

A Project Report on

**DETECTION OF ANOMALOUS BEHAVIOR OF
SMARTPHONE DEVICES USING CHANGE POINT
ANALYSIS & MACHINE LEARNING**

Submitted in partial fulfilment of the requirements for the award of the degree

of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

J.R. SAI LOHITHA (19KA1A0519)

A. SATHWIKA (19KA1A0504)

V. HARI SREE RANI (19KA1A0517)

M. PREMALATHA (19KA1A0555)

Under the esteemed guidance of

Mrs. K.R. LAVANYA, M. Tech.

Assistant Professor (Adhoc)

Department of CSE



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

COLLEGE OF ENGINEERING: KALIKIRI

ANNAMAYYA (Dist.), ANDHRA PRADESH – 517234

2019-2023

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY
ANANTAPUR**

COLLEGE OF ENGINEERING, KALIKIRI

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the project report entitled “**DETECTION OF ANOMALOUS BEHAVIOR OF SMARTPHONE DEVICES USING CHANGE POINT ANALYSIS & MACHINE LEARNING**” that is submitted by

J.R. SAI LOHITHA (19KA1A0519)

A. SATHWIKA (19KA1A0504)

V. HARI SREE RANI (19KA1A0517)

M. PREMALATHA (19KA1A0555)

in partial fulfilment of the requirements for the award of degree of Bachelor of Technology (B. Tech) in **Computer Science and Engineering (CSE)** from **Jawaharlal Nehru Technological University Anantapur College of Engineering, Kalikiri** during the academic year **2022-2023**.

Project Guide

Mrs. K. R. Lavanya, M. Tech

Assistant Professor (Adhoc),

Department of CSE, JNTUACEK

Kalikiri, Annamayya (Dist.)

Head of The Department

Mrs. Shaik Naseera, M. Tech., Ph.D.

Professor & Head of Department,

Department of CSE, JNTUACEK

Kalikiri, Annamayya (Dist.)

Internal Examiner

External Examiner

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

COLLEGE OF ENGINEERING, KALIKIRI

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



DECLARATION

We J.R. SAI LOHITHA (19KA1A0519), A. SATHWIK(19KA1A0504), V. HARI SREE RANI(19KA1A0517), M. PREMALATHA(19KA1A0555) hereby declare that the project work entitled “**DETECTION OF ANOMALOUS BEHAVIOR OF SMARTPHONE DEVICES USING CHANGE POINT ANALYSIS & MACHINE LEARNING**” is carried out under the guidance of **Mrs. K.R.LAVANYA**, M. Tech., Assistant Professor(Ad hoc), Department of CSE, in partial fulfilment for award of the degree of “**BACHELOR OF TECHNOLOGY**” from JNTUA College of Engineering Kalikiri. The results embodied in this project work has not been submitted to any other university or institute for the award of any degree.

| | |
|--------------------------|---------------------|
| J.R. SAI LOHITHA | (19KA1A0519) |
| A. SATHWIK | (19KA1A0504) |
| V. HARI SREE RANI | (19KA1A0517) |
| M. PREMALATHA | (19KA1A0555) |

ACKNOWLEDGEMENTS

An endeavour over a long period can be successful only with advice and support of many well-wishers. The task would be incomplete without mentioning the people who have made it possible, because it is the epitome of hard work. So, with the gratitude, we acknowledge all those whose guidance and encouragement owned our efforts with success.

We are thankful to **Prof. S. V. SATYANARAYANA**, M. Tech., Ph.D., Principal and Professor of Chemical Department, JNTUACE, Kalikiri for his kind and timely help offered to us in projection of our studies and execution.

We are very much obliged to our beloved **Dr. SHAIK NASEERA**, M. Tech., Ph.D., HOD and Professor, Department of Computer Science & Engineering, JNTUACE, Kalikiri for the moral support and invaluable advice provided by her for the success of the project.

We wish to express grateful acknowledgement to our guide **Mrs. K.R. LAVANYA**, M. Tech, Assistant Professor (Adhoc), Department of Computer Science & Engineering, JNTUACE, Kalikiri for her inspiring guidance and continuous encouragement throughout the project.

Finally, we would like to extend our deep sense of gratitude to all the staff members, friends and last but not least we are greatly indebted to our parents who inspired us at all circumstances.

PROJECT ASSOCIATES

| | |
|--------------------------|---------------------|
| J.R. SAI LOHITHA | (19KA1A0519) |
| A. SATHWIK | (19KA1A0504) |
| V. HARI SREE RANI | (19KA1A0517) |
| M. PREMALATHA | (19KA1A0555) |

ABSTRACT

The use of smartphones has increased significantly over the years, and so has the risk of malware attacks on these devices. Traditional methods for detecting malicious activity on smartphones, such as static and dynamic analysis, have proven to be vulnerable and time-consuming. In this project, we propose a generic methodology that uses a data collector and analyser to detect anomalous behaviour on smartphones by analysing changes in power consumption, which can summarize software changes.

Since static analysis is vulnerable to code obfuscation, but dynamic analysis requires harmful activity to cease to dormant in the shortest possible time while data samples are collected. Detecting and recording harmful activity in data samples in dynamic analysis is difficult because we need to produce an efficient mix of user inputs.

To trigger these harmful programmes, we present a generic methodology that employs a data collector and analyser to uncover harmful activity through data analysis. Device's power consumption is employed as this summarises changes made in software. To create user inputs, the data collector employs an automated tool. The data analyser extracts characteristics from power usage using changepoint analysis and trains these features using machine learning techniques.

Two techniques are used in the data analyser step to extract features utilising parametric and non-parametric changepoints. Our procedures take less time to collect data than manual methods, and the data analyser is more accurate than previous ways, for simulated and actual malware

The system includes a user module and a user interface to facilitate data collection, pre-processing, model building, and result visualization. Additionally, we added the use of XAMPP and Docker in the project and provide a detailed explanation of the selected machine learning algorithms. Our project proposes an efficient and accurate methodology for detecting anomalous behaviour on smartphones, which can aid in early detection and prevention of malware attacks

Keywords: Anomalous Behavior, Decision tree, XG Boosting, AdaBoost Classifiers and Support Vector Classifier

TABLE OF CONTENTS

| | |
|---|-----|
| CERTIFICATE | ii |
| DECLARATION | iii |
| ACKNOWLEDGEMENTS | iv |
| ABSTRACT | v |
| LIST OF FIGURES | ix |
| LIST OF ABBREVIATIONS | x |
| CHAPTER 1 | 1 |
| INTRODUCTION | 1 |
| 1.1 Introduction | 1 |
| 1.2 Objectives | 2 |
| CHAPTER 2 | 3 |
| LITERATURE SURVEY | 3 |
| CHAPTER 3 | 5 |
| PROBLEM IDENTIFICATION | 5 |
| 3.1 Existing System | 5 |
| 3.1.1 Disadvantages of Existing Systems | 5 |
| 3.2 Proposeing System | 6 |
| 3.2.1 Advantages of Proposing System | 7 |
| 3.3 Scope | 8 |
| 3.4 Target | 9 |
| 3.5 Feasibility Study | 10 |
| 3.5.1 Economic Feasibility | 10 |
| 3.5.2 Technical Feasibility | 11 |
| 3.5.3 Social Feasibility | 11 |
| 3.5.3 Operational Feasibility | 11 |
| 3.6 Requirements | 11 |

| | |
|---|-----------|
| 3.6.1 Functional Requirements..... | 12 |
| 3.6.2 Non-Functional Requirements | 12 |
| 3.6.3 Hardware & Software Requirements | 13 |
| CHAPTER 4..... | 14 |
| SYSTEM ANALYSIS | 14 |
| 4.1 Block Diagram..... | 14 |
| 4.2 System Architecture | 16 |
| 4.3 UML Diagrams..... | 17 |
| 4.3.1 Use Case Diagram..... | 17 |
| 4.3.2 Class Diagram | 18 |
| 4.3.3 Sequence Diagram between User and System | 19 |
| 4.3.4 Collaboration Diagram between User and System | 20 |
| CHAPTER 5..... | 21 |
| IMPLEMENTATION..... | 21 |
| 5.1 Modules..... | 21 |
| 5.1.1 User module | 21 |
| 5.1.2 System module | 22 |
| 5.2 Technologies Used | 23 |
| 5.2.1 HTML..... | 23 |
| 5.2.2 CSS..... | 23 |
| 5.2.3 XAMPP | 24 |
| 5.2.4 Docker | 24 |
| 5.3 Software Environment..... | 24 |
| 5.3.1 Python..... | 24 |
| 5.3.2 Java Script | 25 |
| 5.3.3 Flask | 25 |
| 5.3.4 Libraries | 26 |

| | |
|--|-----------|
| 5.3.5 Development Environment | 26 |
| CHAPTER 6..... | 27 |
| SAMPLE CODE | 27 |
| CHAPTER 7..... | 34 |
| SYSTEM TESTING | 34 |
| 7.1 Types of Manual testing performed | 34 |
| CHAPTER 8..... | 36 |
| RESULTS | 36 |
| CHAPTER 9..... | 40 |
| CONCLUSION | 40 |
| CHAPTER 10..... | 41 |
| REFERENCES | 41 |

LIST OF FIGURES

| | |
|--|-----|
| Figure 1 Block Diagram | 175 |
| Figure 2 System Architecture | 14 |
| Figure 3 Use Case Diagram..... | 14 |
| Figure 4 Class diagram..... | 19 |
| Figure 5 Sequence diagram between User and System..... | 19 |
| Figure 6 Collaboration diagram between User and System | 20 |
| Figure 7 Correlation ping | 16 |
| Figure 8 Algorithm Comparision..... | 16 |
| Figure 9 Home Page | 36 |
| Figure 10 Load Page..... | 36 |
| Figure 11 View Page..... | 37 |
| Figure 12 Pre-process Page | 37 |
| Figure 13 Model Training Page | 38 |
| Figure 14 Accuracy Score Report..... | 38 |
| Figure 15 Prediction Page..... | 39 |

LIST OF ABBREVIATIONS

ABBREVIATIONS

| | | |
|-----------|---|---|
| SVM | - | SUPPORT VECTOR MACHINE |
| XG BOOST | - | EXTREME GRADIENT BOOST |
| ADA BOOST | - | ADAPTIVE BOOST |
| IOT | - | INTERNET OF THINGS |
| AR | - | ARGUMENTED REALITY |
| LBMAR | - | LOCATION-BASED MOBILE AUGMENTED REALITY |
| DEX | - | DALVIK EXECUTABLE |
| AMD | - | ANDROID MALWARE DATASET |
| TCP | - | TRANSMISSION CONTROL PROTOCOL |
| IDE | - | INTEGRATED DEVELOPMENT ENVIRONMENT |
| SDLC | - | SOFTWARE DEVELOPMENT LIFE CYCLE |
| CSS | - | CASCADING STYLE SHEET |
| HTML | - | HYPERTEXT MAEKUP LANGUAGE |
| HTTP | - | HYPER TEXT TRANSFER PROTOCOL |
| FTP | - | FILE TRANSFER PROTOCOL |
| XSS | - | CROSS-SITE SCRIPTING |
| CSRF | - | CROSS-SITE REQUEST FORGERY |
| VS CODE | - | VISUAL STUDIO CODE |

CHAPTER 1

INTRODUCTION

1.1 Introduction

Smartphones have become an essential part of modern life due to their widespread use for communication and access to information. However, these devices are also a target for cybercriminals, who develop malicious applications to steal user information or harm the performance of cellular networks. Researchers have been developing various methodologies to detect these malicious applications based on the analysis of dynamic characteristics of the device such as power consumption, network traffic, CPU activity, and temperature.

In this project, we propose a novel methodology to detection of anomalous behaviour on smartphones using power consumption as the primary feature. The hypothesis is that the power consumed by a device contains valuable information that can be used to identify the presence of abnormal activities. The proposed methodology uses an offline processing technique and off-device measurement, in which an external device collects the power consumption data to improve the resolution of the power traces.

To extract features from the non-stationary power consumption time series signal, we use the theory of changepoint detection. This theory identifies points in the time series where there is a significant change in the power consumption pattern, which can indicate the presence of a malicious application. The extracted features are used as input to a binary classification problem, where the goal is to detect the presence or absence of anomalous behaviour.

To solve this classification problem, we employ two machine learning algorithms, Support Vector Machines (SVM), Decision Trees, XG Boost and Ada Boost Classifier. SVM is a supervised learning algorithm that can effectively classify non-linear data by finding the optimal hyperplane that separates the two classes. Decision Trees, on the other hand, are a supervised learning algorithm that uses a tree-like model to classify data based on a set of decision rules. XG Boost (Extreme Gradient Boosting) and AdaBoost (Adaptive Boosting) are two popular machine learning algorithms that use boosting techniques to improve the accuracy of models by combining multiple weak classifiers into a stronger ensemble classifier.

We conduct experiments on a dataset of power consumption traces collected from a set of smartphones running both benign and malicious applications. Our results show that the proposed methodology can effectively detect malicious applications with high accuracy using both Ada Boost and Decision Trees. Furthermore, the Decision Trees outperformed all other algorithms in terms of accuracy, precision, recall, and F1-score.

In conclusion, we present a novel methodology for detecting anomalous behaviour of smartphones using change point analysis as the primary feature. We demonstrate that the proposed methodology, combined with Machine learning algorithms, can effectively detect anomalous applications with high accuracy.

1.2 Objectives

The project aims to address the growing concern of cyber security threats on smartphones, which are a common target for cybercriminals due to the sensitive information they store and transmit. The proposed methodology utilizes change point analysis, which involves detecting changes in the behaviour of a time series signal, such as the power consumption of a smartphone, to identify anomalous behaviour. The change points can provide useful features for machine learning algorithms, such as SVM, Decision Trees, XG Boost, and AdaBoost, to improve the accuracy of detecting malicious applications or other anomalous behaviour.

By comparing the performance of different machine learning algorithms, the project aims to identify the most effective approach for detecting anomalous behaviour in smartphones. The SVM algorithm is known for its ability to handle complex and high-dimensional data, while Decision Trees are easy to interpret and can handle both numerical and categorical data. XG Boost is a powerful algorithm that uses gradient boosting and has been shown to achieve state-of-the-art results in various machine learning tasks, while AdaBoost can improve the accuracy of weak classifiers by combining them into a strong ensemble classifier.

The project aims to contribute to the development of effective and efficient methodologies for detecting anomalous behaviour in smartphones, which can help to enhance their security and protect the privacy of their users.

CHAPTER 2

LITERATURE SURVEY

[1] Nowadays, we experience an abundance of Internet of Things middleware solutions that make the sensors and the actuators are able to connect to the Internet. These solutions, referred to as platforms to gain a widespread adoption, have to meet the expectations of different players in the IoT ecosystem, including devices. Low-cost devices are easily able to connect wirelessly to the Internet, from handhelds to coffee machines, also known as Internet of Things (IoT). This research describes the methodology and the development process of creating an IoT platform. This paper also presents the architecture and implementation for the IoT platform. The goal of this research is to develop an analytics engine which can gather sensor data from different devices and provide the ability to gain meaningful information from IoT data and act on it using machine learning algorithms. The proposed system is introducing the use of a messaging system to improve the overall system performance as well as provide easy scalability.

[2] The advancement of virtual reality has sharpened the concept of augmented reality (AR) to new dimension of perceptions of seeing, hearing and immersing in a real world. The evolution of mobile devices has pioneered AR as a state-of-the-art technology in the last decade giving rise to more and more location-based mobile AR (LBMAR) systems. However, notably there are very limited review studies that have focused on investigating factors such as: growth, types, characteristics, features, sensors, application domains and their respective challenges. This study presents a systematic review on location-based mobile augmented reality (LBMAR) system. A total of 35 studies published between the years 2013 and 2018 in top six most popular indexed databases are reviewed. The systematic review has been conducted using Kitchenham method, and the analysis of the findings was carried out using the PRISMA method. This chapter provides a major review of the current state of LBMAR system and outlines the research issues that require more investigation.

[3] This work is part of the research to study trends and challenges of cyber security to smart devices in smart homes. We have seen the development and demand for seamless interconnectivity of smart devices to provide various functionality and abilities to users. While these devices provide more features and functionality, they also introduce new risks and threats. Subsequently, current cyber security issues related to smart devices are discussed and analyzed.

The paper begins with related background and motivation. We identified mobile malware as one of the main issues in the smart devices' security. In the near future, mobile smart device users can expect to see a striking increase in malware and notable advancements in malware-related attacks, particularly on the Android platform as the user base has grown exponentially. We discuss and analysed mobile malware in details and identified challenges and future trends in this area. Then we propose and discuss an integrated security solution for cyber security in smart devices to tackle the issue.

[4] With the widespread use of smartphones, the number of malwares has been increasing exponentially. Among smart devices, Android devices are the most targeted devices by malware because of their high popularity. This paper proposes a novel framework for Android malware detection. Our framework uses various kinds of features to reflect the properties of Android applications from various aspects, and the features are refined using our existence-based or similarity-based feature extraction method for effective feature representation on malware detection. Besides, a multimodal deep learning method is proposed to be used as a malware detection model. This paper is the first study of the multimodal deep learning to be used in the Android malware detection. With our detection model, it was possible to maximize the benefits of encompassing multiple feature types. To evaluate the performance, we carried out various experiments with a total of 41,260 samples. We compared the accuracy of our model with that of other deep neural network models.

[5] The rapid proliferation of Android malware is challenging the classification of the Android malware family. The traditional static method for classification is easily affected by the confusion and reinforcement, while the dynamic method is expensive in computation. To solve these problems, this paper proposes an Android malware familial classification method based on Dalvik Executable (DEX) file section features. First, the DEX file is converted into RGB (Red/Green/Blue) image and plain text respectively, and then, the colour and texture of image and text are extracted as features. Finally, a feature fusion algorithm based on multiple kernel learning is used for classification. In this experiment, the Android Malware Dataset (AMD) was selected as the sample set. Two different comparative experiments were set up, and the method in this paper was compared with the common visualization method and feature fusion method. The results show that our method has a better classification effect with precision, recall and F1 score reaching 0.96. Besides, the time of feature extraction in this paper is reduced by 2.999 seconds compared with the method of frequent subsequence malware family

CHAPTER 3

PROBLEM IDENTIFICATION

3.1 Existing System

In the existing systems, implementation of machine learning algorithms is bit complex to build due to the lack of information about the data visualization.

Mathematical calculations are used in existing systems for model building this may takes the lot of time and complexity.

The existing systems has several machine learning models to classify whether there is anomalous behaviour or not in the android device, but none have adequately addressed this misdiagnosis problem.

Also, similar studies that have proposed models for evaluation of such performance classification mostly do not consider the heterogeneity and the size of the data

To overcome all this, we use machine learning packages available in the scikit-learn library

3.1.1 Disadvantages of Existing Systems

1. Lack of information about data visualization: The existing system faces challenges in implementing machine learning algorithms due to the lack of information about data visualization. This means that it is difficult to understand and visualize the data, which can make it harder to identify patterns or outliers that may be indicative of anomalous behaviour.
2. Use of mathematical calculations for model building: The existing system relies on mathematical calculations for building machine learning models. This approach can be time-consuming and complex, as it involves writing and executing complex mathematical equations to build the models.
3. Misdiagnosis problem: The existing system has several machine learning models for classifying whether there is anomalous behaviour in the Android device. However, these models have not adequately addressed the misdiagnosis problem, which refers to the incorrect identification of anomalous behaviour.

4. Ignoring heterogeneity and data size: Similar studies that have proposed models for evaluating the performance of classification mostly do not consider the heterogeneity and size of the data. The existing system may face similar challenges in addressing these issues.

To overcome these challenges, the proposing system will use machine learning packages available in the scikit-learn library. It will also incorporate change point analysis to detect anomalous behaviour in smartphones, which can improve the accuracy of the models.

3.2 Proposing System

The proposing project aims to provide an end-to-end methodology to automatically collect data and analyse them to detect anomalous behaviour in smartphones. The methodology involves collecting network traffic and TCP packets, filtering them, and then selecting and extracting features from them. The selected functions from various network features are then labelled and stored.

Machine learning classification is then used to build a detection model. The proposed project uses several machine learning models, including Support Vector Machines (SVM), Decision Trees, XG Boost Classifier, and AdaBoost Classifier, to classify whether there is anomalous behaviour or not in the smartphone. The selected features and labelled data are used to train the machine learning models, which are then used to predict whether there is anomalous behaviour in the smartphone.

The use of machine learning classification in the proposed project provides several advantages. First, it offers the highest accuracy compared to traditional methods. Second, it reduces time complexity by automating the process of analysing the collected data. Third, it is easy to use, making it accessible to a wide range of users.

SVM is a popular classification technique used in machine learning. It works by finding a hyperplane that maximizes the margin between the classes. SVM is effective in handling high-dimensional datasets and has been used in many applications, including image recognition, natural language processing, and bioinformatics.

Decision Trees are another classification technique used in machine learning. They work by partitioning the feature space recursively based on the feature that provides the most information gain. Decision Trees are useful when dealing with categorical data and can handle missing data.

XG Boost Classifier is a powerful classification technique that is particularly effective when dealing with large datasets. It works by combining multiple decision trees and correcting their errors through a process called boosting. XG Boost is widely used in various applications, including speech recognition, natural language processing, and computer vision.

Ada Boost is another boosting technique that combines weak classifiers to create a strong classifier. It is particularly useful in handling imbalanced datasets and has been used in applications such as credit scoring, spam filtering, and face detection.

The proposing methodology has the potential to provide an effective and efficient approach to detecting anomalous behaviour in smartphones. The use of machine learning classification techniques can improve the accuracy of the detection and reduce the time and complexity of the analysis process. The proposed approach is also easy to use, making it accessible to a wide range of users.

3.2.1 Advantages of Proposing System

The proposing system for anomalous detection of smartphone applications using changepoint analysis and machine learning techniques offers several advantages over the existing system. Some of the main advantages are:

1. **Higher Accuracy:** The proposed system utilizes changepoint analysis to extract features from power consumption data, and machine learning techniques to train a classifier. This enables the system to identify malicious behaviour more accurately, leading to better detection rates and fewer false positives.
2. **Faster Detection:** The proposed system can recognize malware acting in short periods of time which is a disadvantage of the other methodologies. This leads to faster detection of malicious behaviour, reducing the time taken for detection and remediation.

3. **Greater Flexibility:** The proposed system can be easily adapted to detect anomalous behaviour in different applications, enabling it to be used in a variety of scenarios. This makes it a more versatile solution compared to existing systems that may be limited to specific applications.
4. **Reduced Human Effort:** The proposed system employs an automated tool to create user inputs, and the data analyser is more accurate than previous methods. This reduces the human effort required to collect data, increasing efficiency and reducing costs.
5. **Improved Security:** By detecting malicious behaviour more accurately and quickly, the proposed system can help to improve the overall security of smartphones. This can help to prevent sensitive data from being compromised, reducing the risk of data breaches and other security incidents.

The proposing system offers significant advantages over existing systems, with higher accuracy, faster detection, greater flexibility, reduced human effort, and improved security. These advantages make it a more effective and efficient solution for anomalous detection of smartphone applications.

3.3 Scope

1. The scope of our project is to develop a system for anomalous detection of smartphones using change point analysis and machine learning algorithms. This involves designing and implementing a data collector and analyser that can identify harmful activity through data analysis and power consumption monitoring.
2. Our project also includes the development of a user module that allows users to interact with the system. The user module should be intuitive and easy to use, making it accessible to a wide range of users.
3. Another important aspect of our project is the selection and implementation of machine learning algorithms. We will need to carefully evaluate various algorithms and select the ones that are best suited for detecting anomalous behaviour in smartphone applications. This will require a deep understanding of machine learning principles and techniques.

4. Additionally, our project will require the use of various tools and technologies, such as Python, Scikit-learn, Pandas, NumPy, Flask, and Docker. You will need to be proficient in using these tools and technologies, as well as in programming in general.
5. Finally, the scope of our project includes testing and validation of the system to ensure that it is accurate, reliable, and effective. We will need to design and implement various tests to evaluate the performance of the system, and make any necessary adjustments and improvements to ensure that it meets the required standards.

The scope of our project is broad and requires a range of skills and expertise. By developing a system for anomalous detection of smartphones, we will be contributing to the ongoing efforts to improve cybersecurity and protect users from malicious activity. Our project has the potential to have a significant impact on the field of cybersecurity, and could be used to develop more advanced and effective detection systems in the future.

3.4 Target

The primary target of this project is to develop a methodology for identifying malicious activity on smartphones in real-time. With the increasing number of malware attacks on smartphones, it is important to have a system that can detect these attacks quickly and accurately. The proposed methodology uses changepoint detection theory and machine learning algorithms to analyse the power consumption of a smartphone and identify any anomalous behaviour. The aim is to achieve a high level of accuracy in detecting malware and other malicious activity on smartphones.

Another target of this project is to provide a user-friendly interface for smartphone users to monitor their devices for malicious activity. The user interface will display information about the smartphone's power consumption and highlight any anomalous behaviour. This will help users to detect any potential threats and take necessary actions to prevent further damage. The user interface will also provide guidance on how to safeguard their devices against malware attacks and other security threats.

A key target of this project is to provide a scalable and efficient system that can handle large volumes of data. The proposed methodology uses machine learning algorithms to analyse the power consumption data collected from smartphones. To achieve scalability, we will use distributed computing techniques and cloud-based platforms to process the data.

Another target of this project is to enable researchers to study malware attacks on smartphones and develop new techniques to detect and prevent these attacks. The proposed methodology provides a framework for collecting and analysing data from smartphones, which can be used by researchers to develop new machine learning algorithms and other techniques for detecting malware. The system can also be used to analyse the effectiveness of existing malware detection techniques and identify areas for improvement.

Finally, a key target of this project is to contribute to the development of a more secure and resilient mobile ecosystem. The proposed methodology can help to improve the security of smartphones and prevent cyber-attacks. By providing a reliable and efficient system for detecting and preventing malware attacks, we can help to protect users' personal information and prevent financial losses. Ultimately, the goal of this project is to enhance the overall security and privacy of smartphone users and contribute to a safer and more secure mobile ecosystem.

3.5 Feasibility Study

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ ECONOMICAL FEASIBILITY
- ◆ TECHNICAL FEASIBILITY
- ◆ SOCIAL FEASIBILITY
- ◆ OPERATIONAL FEASIBILITY

3.5.1 Economic Feasibility

Economic feasibility refers to the cost-benefit analysis of the proposed solution. This project aims to develop a methodology that can detect anomalous behaviour in smartphone applications, which can help prevent malware attacks and other cyber threats. The cost of developing the proposed solution includes the cost of hiring skilled professionals and

purchasing hardware and software. However, the benefits of the solution outweigh the cost, as it can prevent data breaches, identity theft, and other cyber threats. Therefore, the economic feasibility of the proposed solution is high.

3.5.2 Technical Feasibility

Technical feasibility refers to the capability of the technology to achieve the objectives of the project. In this project, we aim to develop a methodology to detect anomalous behaviour in smartphone applications using changepoint detection theory and machine learning techniques. The proposed methodology utilizes commonly available technology, such as smartphones, automated tools for creating user inputs, and machine learning libraries. Therefore, the technical feasibility of this project is high.

3.5.3 Social Feasibility

Social feasibility refers to the impact of the proposed solution on society. The proposed methodology can have a significant impact on society, as it can help prevent cyber threats, which have become prevalent in recent times. It can also help protect user privacy and confidential data. However, the methodology may require users' consent to collect power consumption data from their smartphones. Therefore, the social feasibility of the proposed solution is moderate.

3.5.3 Operational Feasibility

Operational feasibility refers to the practicality of implementing the proposed solution. This project aims to develop a methodology for detecting anomalous behaviour in smartphone applications, which requires collecting power consumption data and training a machine learning model. The data collection process may require user inputs and can be automated. Additionally, the trained model can be deployed on a smartphone or in the cloud for real-time detection of anomalous behaviour. Therefore, the operational feasibility of the proposed solution is high.

3.6 Requirements

The requirement analysis of the project involves identifying the necessary components and functionalities to achieve the project goals. Firstly, data collection and pre-processing of network traffic and TCP packets are required to extract relevant features for analysis. This

involves filtering and labelling of the collected data to enable effective analysis using machine learning algorithms. Secondly, change point analysis is used to detect any abnormalities in the collected data. Change point analysis identifies sudden changes or shifts in the data patterns, which could indicate the presence of an anomaly in the smartphone behaviour.

The project also requires the implementation of various machine learning algorithms, including Support Vector Machines, Decision Trees, XG Boost, and AdaBoost. These algorithms are used to build classification models that can detect anomalous behaviour in smartphones. Each of these algorithms has unique strengths and limitations, and their performance is evaluated to determine the most suitable algorithm for the project. Additionally, the accuracy and efficiency of the proposed methodology are crucial requirements of the project, as they determine the effectiveness and practicality of the anomaly detection system.

3.6.1 Functional Requirements

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

3.6.2 Non-Functional Requirements

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project together. They are also called non-behavior requirements. They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability

Examples of non-functional requirements:

- 1) The processing of each request should be done within 10 seconds
- 2) The site should load in 3 seconds whenever of simultaneous users are > 10000

3.6.3 Hardware & Software Requirements

a) Hardware:

For developing the application, the following are the Software Requirements:

- Coding Language : Python 3.9.4
- IDE : PyCharm 2022.2.2
- Frameworks : Flask 2.2
- Tool : WinRunner, Docker, XAMPP Server 3.2.0
- Operating System : Windows 10
- Debugger and Emulator : Any Browser (Particularly Chrome)

b) Software:

For developing the application, the following are the Hardware Requirements:

- Processor : Pentium IV or higher
- RAM : 8 GB
- Hard Disk or SSD : More than 500 GB

CHAPTER 4

SYSTEM ANALYSIS

4.1 Block Diagram

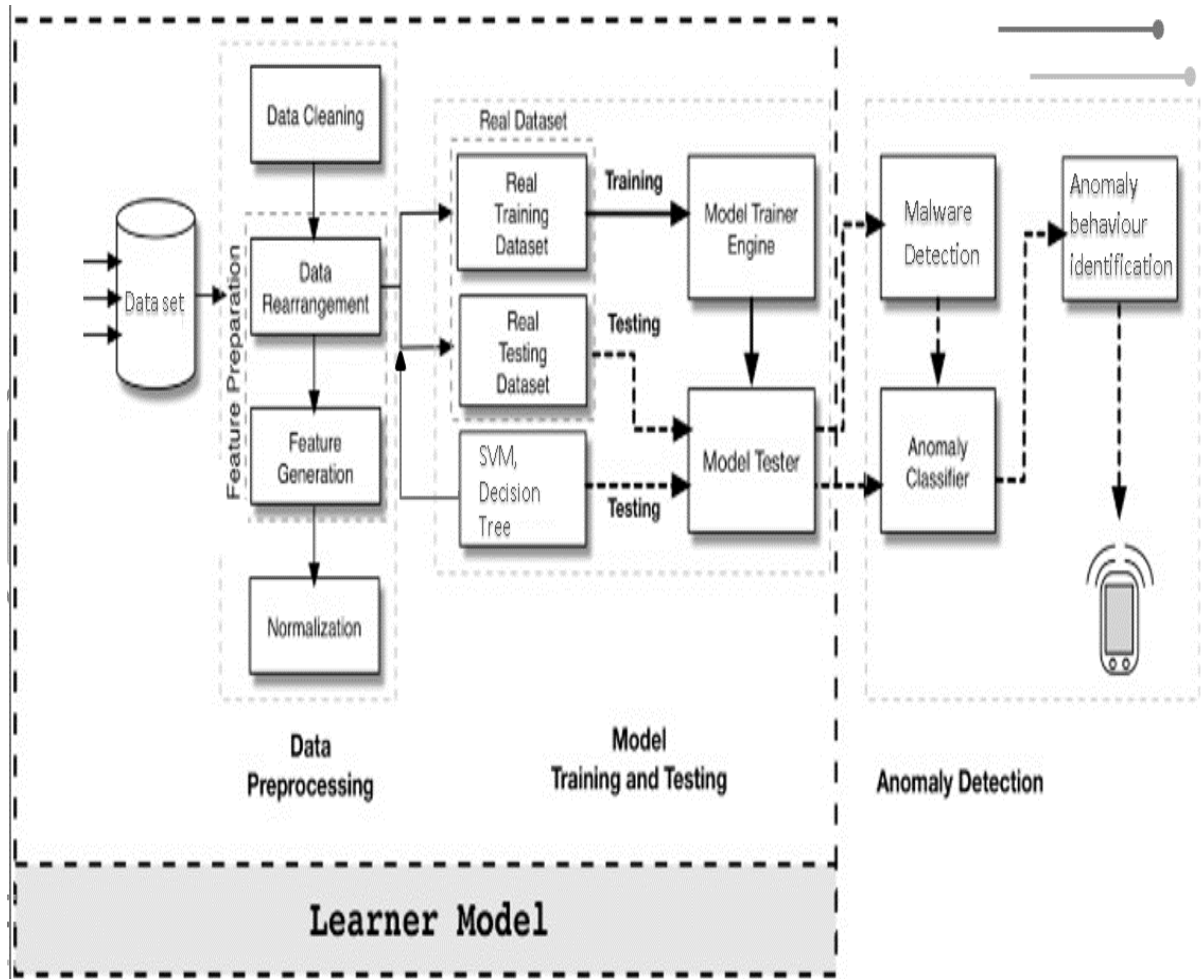


Figure 1 Block Diagram

The block diagram for your project would likely include the following components:

1. Drebin dataset: The first step is to acquire the Drebin dataset, which contains network traffic and TCP packets data of Android devices.
2. Data cleaning: The acquired dataset may contain irrelevant or missing information. Therefore, the next step is to clean the data by removing any inconsistencies or missing values.

3. Data rearrangement: The data is rearranged to prepare it for feature generation. This step may involve merging or splitting different columns, filtering the data, and removing any unnecessary information.
4. Feature generation: The feature generation step involves extracting important information from the dataset.
5. Normalization: The generated features are normalized to bring them to a common scale.
6. Real training dataset: A portion of the pre-processed data is selected for model training. This dataset is used to train the machine learning models using SVM, decision tree, XG Boost, and AdaBoost algorithms.
7. Real testing dataset: Another portion of the pre-processed data is used for testing the trained models.
8. Model trainer engine: The model trainer engine uses the training dataset to train the machine learning models. It applies different machine learning algorithms to generate different models.
9. Model tester: The model tester evaluates the performance of the trained models using the testing dataset. It assesses the accuracy, precision, recall, and F1-score of the models to determine their performance in detecting anomalies.
10. Malware detection: The trained models are used to detect malware in the smartphone by analysing network traffic.
11. Anomaly classifier: The anomaly classifier classifies the behaviour of the smartphone into normal or anomalous based on the generated features and trained machine learning models.
12. Anomaly behaviour identification: Based on the classification output, the smartphone's anomalous behaviour is identified and appropriate actions can be taken to mitigate it.

4.2 System Architecture

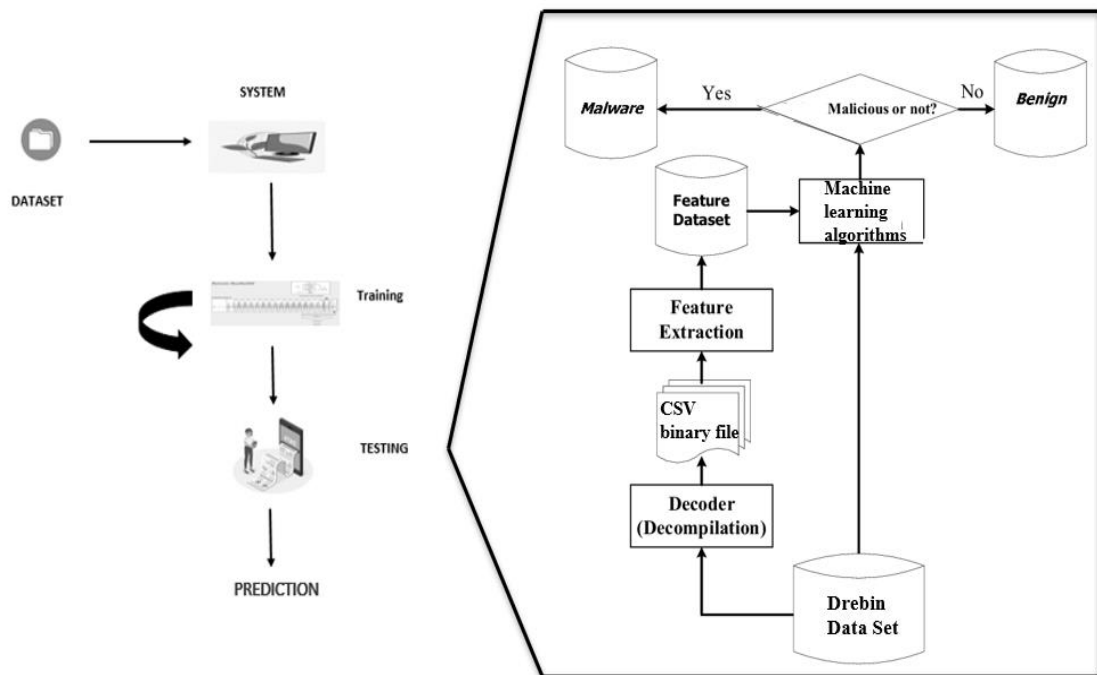


Figure 2 system architecture

The architecture diagram for your project would likely include the following components:

1. **Data Collection:** This component would be responsible for collecting data from the smartphone in question. This could include network traffic and other relevant data points.
2. **Data Filtering and Feature Extraction:** Once the data is collected, it would need to be filtered to remove any irrelevant information. The relevant features would then need to be extracted from the data.
3. **Machine Learning Model Training:** The next component in the architecture would involve training the machine learning models using the extracted features as input. This would involve selecting appropriate algorithms such as SVM, Decision Tree, XG Boost, and AdaBoost, and tuning their parameters for optimal performance.
4. **Model Evaluation:** After training the models, they would need to be evaluated for their accuracy and performance. This could be done using various performance metrics, such as precision, recall, and F1-score.
5. **Anomaly Detection:** The final component in the architecture would involve using the trained models to detect anomalous behaviour in the smartphone.

4.3 UML Diagrams

4.3.1 Use Case Diagram

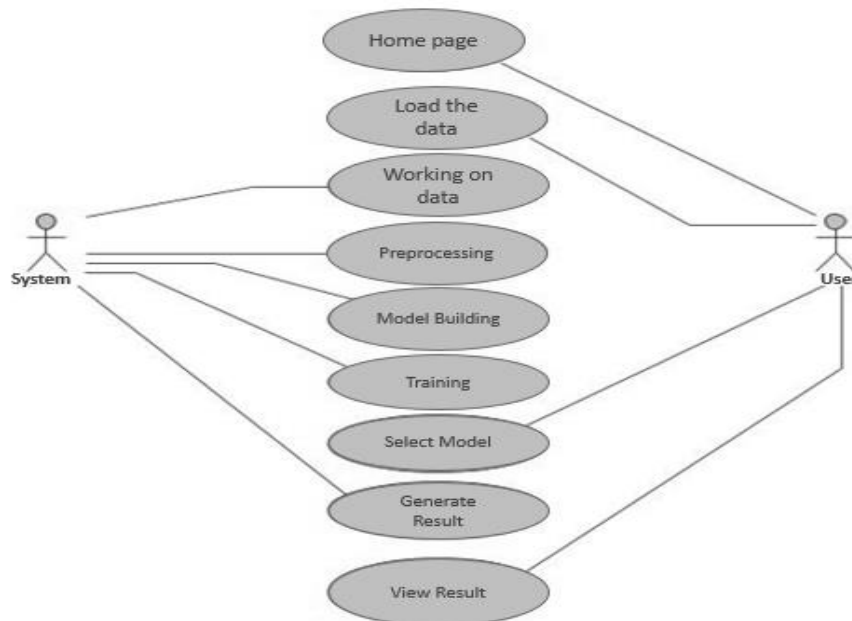


Figure 3 use case

The use case diagram includes the following actors:

1. User
2. System

The following use cases are depicted in the diagram:

1. Home Page: The user can access the home page of the system.
2. Load the Data: The user can load the raw data into the system.
3. Working on Data: The user can perform various operations such as data cleaning, data rearrangement, feature generation, and normalization on the loaded data.
4. Pre-Processing: The user can pre-process the data to remove outliers, null values, and other inconsistencies.

4.3.2 Class Diagram

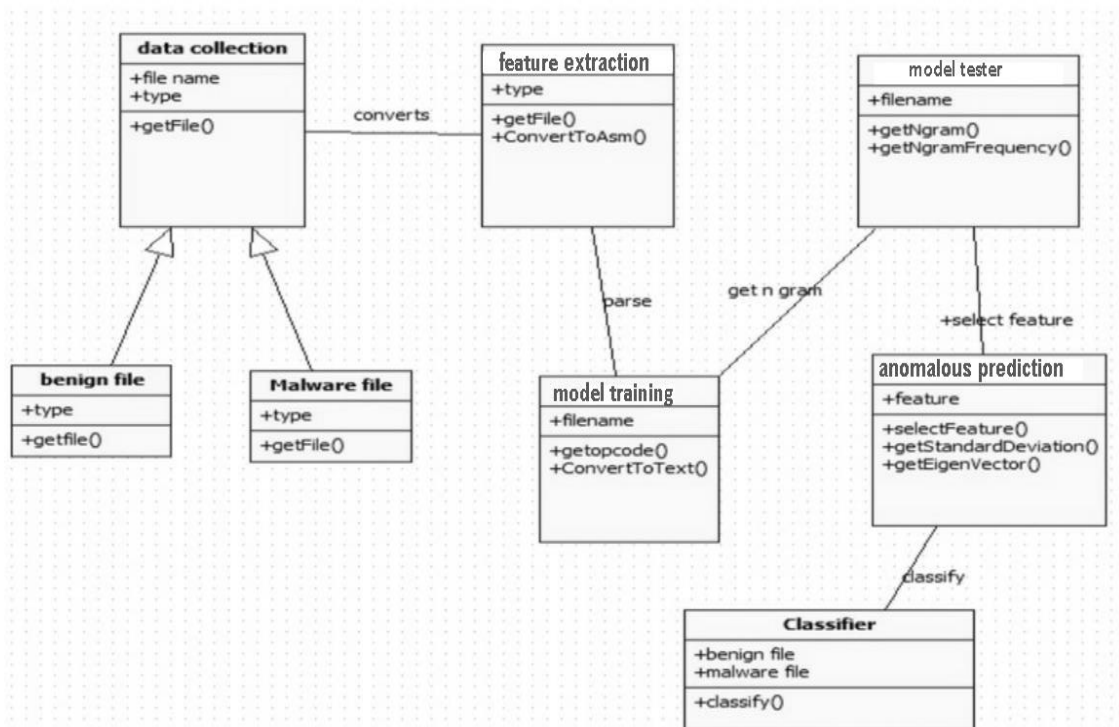


Figure 4 class diagram

The main classes involved in the project are:

1. Data Collection - This class includes two subclasses - 'Benign Files' and 'Malicious Files' for collecting the normal and anomalous data files respectively.
2. Feature Extraction - This class is responsible for extracting relevant features from the collected data files.
3. Model Training - This class is used to train different machine learning models including Support Vector Machine (SVM), Decision Tree, XG Boost and AdaBoost classifiers.
4. Model Tester - This class is used to test the trained models and evaluate their performance.
5. Anomalous Detection - This class is responsible for detecting anomalous behaviour in the smartphone based on the trained models.
6. Anomaly Classifier - This class is used to classify the detected anomalies into different categories.

4.3.3 Sequence Diagram between User and System

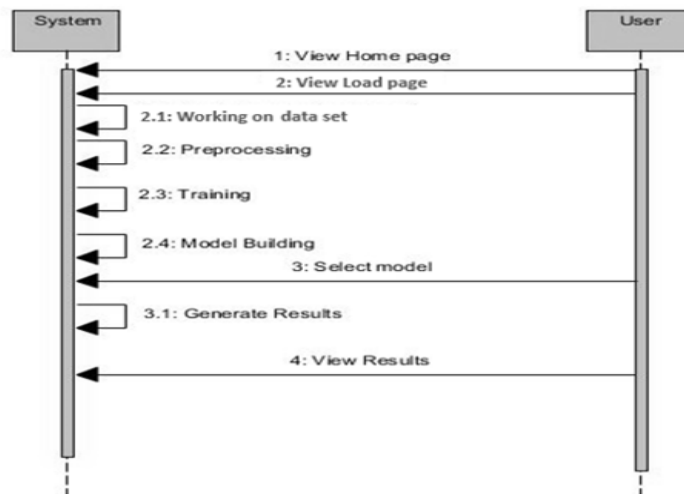


Figure 5 sequence diagram

Here is a sequence diagram for the process you described:

1. Home page: User opens the application and is presented with the home page.
2. Load the data: User selects to load data from their smartphone. The system requests the data from the device.
3. Working on data: The system performs some initial processing on the data to ensure it is in the correct format and remove any irrelevant data.
4. Pre-processing: The system performs pre-processing on the data to prepare it for machine learning algorithms. This includes feature extraction, normalization, and filtering.
5. Model building: The system builds machine learning models using the pre-processed data.
6. Training: The system trains the machine learning models using a portion of the data.
7. Select model: The user selects which machine learning model to use for anomaly detection.
8. Generate result: The system applies the selected machine learning model to the remaining data and generates a result.
9. View result: The user views the result of the anomaly detection.

4.3.4 Collaboration Diagram between User and System

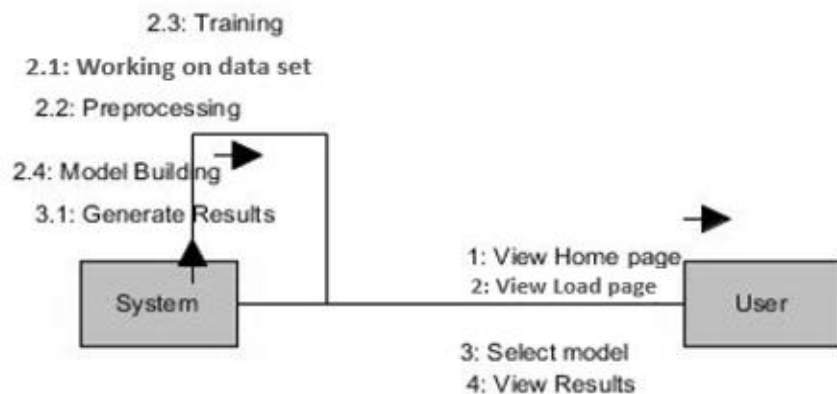


Figure 6 collaboration

Here is a collaboration diagram for the process you described:

1. Home page: User opens the application and is presented with the home page.
2. Load the data: User selects to load data from their smartphone. The system requests the data from the device.
3. Working on data: The system performs some initial processing on the data to ensure it is in the correct format and remove any irrelevant data.
4. Pre-processing: The system performs pre-processing on the data to prepare it for machine learning algorithms. This includes feature extraction, normalization, and filtering.
5. Model building: The system builds machine learning models using the pre-processed data.
6. Training: The system trains the machine learning models using a portion of the data.
7. Select model: The user selects which machine learning model to use for anomaly detection.
8. Generate result: The system applies the selected machine learning model to the remaining data and generates a result.
9. View result: The user views the result of the anomaly detection.

CHAPTER 5

IMPLEMENTATION

5.1 Modules

5.1.1 User module

The user module of the Anomalous Detection of Smartphones using Change Point Analysis and Machine Learning Algorithms project is designed to enable the user to interact with the system and perform various tasks.

The user module includes several functionalities such as loading the data, working on data, selecting models, generating results, and viewing the homepage, among others. The user module is the primary interface between the user and the system.

Users can access the system through the homepage and can upload the required data set. The data set is then pre-processed to eliminate inconsistencies and irrelevant data. The user can then choose a specific model and use it to train the data.

Model selection

Model selection is a critical step in any machine learning project, including the Anomalous behaviour detection of smartphones project. The goal of this step is to select the best machine learning algorithm that can accurately classify the collected data into either anomalous or normal behaviour.

In this project, four different algorithms have been selected for the model selection stage: Support Vector Machines (SVM), Decision Trees, XG Boost, and AdaBoost. SVM is a powerful algorithm that is useful for both linear and non-linear data, and it works by finding a hyperplane that best separates the data points into different classes. Decision Trees is a simple yet effective algorithm that can be easily understood and interpreted, and it works by recursively splitting the data based on the most informative feature. XG Boost and AdaBoost are two boosting algorithms that work by combining multiple weak models to form a stronger model.

To select the best algorithm for this project, several factors need to be considered, including the accuracy, training and testing time, and computational resources required.

5.1.2 System module

Data Collection

- System checks for data whether it is available or not and load the data in csv files.
- As stated earlier, the data for this research consisted of 150,000 malicious files and 87,000 benign executables of Windows format.
- The benign executables were retrieved from fresh installation of Windows 7, Windows 8, Windows 10, Windows Server 2008, and Windows Server 2012.
- To foster research on Android malware and to enable a comparison of different detection approaches, we make the datasets from our project Drebin publicly available. The samples have been collected in the period of August 2010 to October 2019 and were made available to us by the Mobile Sandbox project.

Pre-processing

Data need to be pre-processed. According the models it helps to increase the accuracy of the model and better information about the data.

Feature Extraction

Each sample is described with a features vector to identify potentially malicious applications. Transforming raw data into numerical features that can be processed while preserving the information in original data set. So that potential of feature extraction system can be leveraged to combat with unfamiliar malwares.

- **Feature Selection** A method for automatic feature selection in anomaly detection is proposed which determines optimal mixture coefficients for various sets of features. The method generalizes the support vector data description (SVDD) and can be expressed as a semi-infinite linear program that can be solved with standard techniques.
- **Dataset Standardization** Irrelevant imported functions are disregarded and multiple functions with a similar effect can be mapped to a single action.

Analysing and Detecting

- **Anomalous Analysis:** We demonstrate that with careful selection and extraction of the features combined with SVM, Decision Tree machine learning algorithm, we can build baseline models of benign program execution and use these profiles to detect deviations that occur as a result of malware exploitation
- **Model Building:** Monitoring Power Consumption Battery power consumption is one of the major limitations of mobile phones that limit the complexity of anti-malware solutions. It also brings the challenge for mobile malware as all critical behaviors for malware propagation such as accessing WIFI or Bluetooth consume significant battery power. Any malicious behaviors caused by mobile malware also involve extra power

Generated Score: Here user view the score in % according to the dataset uploaded with significant algorithms accuracies i.e., Support Vector Machine, Decision Tree, XG booster, Ada booster

Generate Results: We train the machine learning algorithm and predict the anomalous behaviour

5.2 Technologies Used

5.2.1 HTML

Hyper Text Markup Language (HTML), the languages of the World Wide Web (WWW), allows users to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Mark-up Language), but specialized to hypertext and adapted to the Web.

5.2.2 CSS

Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable. CSS is a MUST for students and working professionals to become a great Software Engineer especially when they are working in Web Development Domain.

5.2.3 XAMPP

XAMPP is a cross-platform web server solution stack package developed by Apache Friends. It consists of mainly Apache HTTP Server, MySQL database, and interpreters for scripting languages such as PHP and Perl. XAMPP is designed to be an easy-to-install and easy-to-use Apache distribution that is suitable for developers who need a local testing environment for their PHP and MySQL projects.

5.2.4 Docker

Docker is an open-source platform that automates the deployment of applications within software containers. Containers provide a way to package an application's code, libraries, and dependencies into a single object that can run consistently across different environments, such as development, testing, and production.

5.3 Software Environment

- | | | |
|-------------------------------|---|------------------------------|
| 1. Programming languages | : | Python, JavaScript |
| 2. Frameworks | : | Flask, Bootstrap |
| 3. Machine learning libraries | : | Scikit-learn, Pandas, NumPy |
| 4. Operating system | : | Linux, Windows |
| 5. Development environment | : | PyCharm, Visual Studio Code. |
| 6. Deployment tools | : | Docker |

5.3.1 Python

Python is an essential tool in the "Anomalous detection of smartphones using change point analysis and machine learning algorithms" project for several reasons:

Data Analysis and Visualization: Python provides powerful libraries such as NumPy, Pandas, and Matplotlib for data analysis and visualization, which are essential in pre-processing and understanding the dataset.

Machine Learning: Python has several machine learning libraries such as Scikit-learn, TensorFlow, and Keras, which provide a range of tools for implementing and testing different machine learning algorithms, including decision trees, support vector machines, and XG Boost.

Change Point Analysis: Python has libraries such as ruptures and changepoint_detection, which are useful for implementing change point analysis techniques that detect changes in the behaviour of smartphone applications over time.

Rapid Prototyping: Python's simplicity and ease of use allow for quick prototyping and iteration of ideas, enabling us to test and refine algorithms quickly.

5.3.2 Java Script

JavaScript is a crucial programming language for developing dynamic and interactive web pages and applications. In the case of this project JavaScript was used primarily for designing the front-end of the web application that allows users to interact with the system and input data. JavaScript was used to implement features such as form validation, dynamic rendering of data, and handling user interactions.

In the case of this project, JavaScript was used alongside other web technologies such as HTML and CSS to create an intuitive and user-friendly web application. JavaScript was used for form validation using regular expressions, and handling user inputs using event-driven programming.

The importance of JavaScript in this project lies in its ability to create an interactive and dynamic user interface that facilitates data input and analysis. Without JavaScript, the project would not have been able to achieve its goal of creating an effective and user-friendly system for identifying anomalous behaviour in smartphone applications.

5.3.3 Flask

Flask is a popular Python web framework used for developing web applications. In the case of our project flask is used for the back-end development of the web application. Flask helps to simplify the process of creating and managing web applications by providing built-in functionality for handling routing, server-side rendering, and managing data.

Flask was used to build the routes for the different pages of the web application, including the home page, data loading page, pre-processing page, model training page, and prediction page. It also helped in managing the user input and output by handling the HTTP requests and responses. Flask also provided support for integrating the machine learning algorithms and pre-processing functions into the web application.

In addition, Flask was used for front-end development by rendering the HTML templates and allowing for the dynamic generation of web pages. Flask's templating engine, Jinja, allows for the creation of dynamic web pages that can respond to user input and display data generated from the machine learning models.

Flask played a crucial role in the development of the web application. It simplified the process of building the back-end infrastructure, handling user input and output, integrating the machine learning models, and rendering dynamic web pages.

5.3.4 Libraries

Scikit-learn, Pandas, and NumPy are popular Python libraries used for data analysis, manipulation, and machine learning tasks.

Scikit-learn: Scikit-learn is a widely used machine learning library in Python. It provides a range of supervised and unsupervised learning algorithms, as well as tools for model selection, data pre-processing, and evaluation. Some of the algorithms provided by scikit-learn include linear regression, logistic regression, SVMs, decision trees, and neural networks.

Pandas: Pandas is a Python library for data manipulation and analysis. It provides data structures like Series (1-dimensional) and Data Frame (2-dimensional) for handling and analysing large data sets. Pandas also offers tools for data cleaning, data visualization, and data analysis. Some of the key features of Pandas include data alignment, merging and joining of datasets, filtering and grouping of data, and handling of missing data.

NumPy: NumPy is a Python library used for numerical computing. It provides a range of data structures and functions for performing mathematical operations on arrays and matrices.

5.3.5 Development Environment

Development environment: PyCharm, Visual Studio Code

PyCharm and Visual Studio Code are two popular integrated development environments (IDEs) that are commonly used for Python development. PyCharm includes great features such as excellent code completion, inspection with advanced debugger. Whereas VS code includes features like supporting for debugging, syntax highlighting and intelligent code completion.

CHAPTER 6

SAMPLE CODE

Sample Code for individual algorithms

DecisionTreeClassifier

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
dt = DecisionTreeClassifier()
dt = dt.fit(x_train, y_train)
acc_dt = accuracy_score(y_test, y_pred)
print("The accuracy obtained by the Decision Tree Classifier:", acc_dt)
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

XG Boost Classifier

```
from xgboost import XGBClassifier
xgb = XGBClassifier()
xgb = xgb.fit(x_train, y_train)
y_pred = xgb.predict(x_test)
acc_xgb = accuracy_score(y_test, y_pred)
print("The accuracy obtained by the XGBoost Classifier is:", acc_xgb)
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

Ada Boost Classifier

```
from sklearn.ensemble import AdaBoostClassifier
adb = AdaBoostClassifier()
adb = adb.fit(x_train, y_train)
y_pred = adb.predict(x_test)
acc_adb = accuracy_score(y_pred, y_test)
print("The accuracy obtained by the AdaBoost Classifier is :", acc_adb)
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

Support Vector Classifier

```

from sklearn.svm import SVC
svc = SVC(kernel='linear')
svc = svc.fit(x_train,y_train)
y_pred = svc.predict(x_test)
acc_svc = accuracy_score(y_test,y_pred)
print("The accuracy obtained by the Support Vector Classifier is :",acc_svc)

```

Explanation

1.To execute code we should import libraries and a dataset, follow these steps:

- Open a code editor or an integrated development environment (IDE) such as VS code.
- Import the necessary libraries required for the project. For example, to import NumPy, Pandas.

2.Load the dataset that you will be working on. There are several ways to load a dataset, depending on the file format and location of the dataset. For instance, if you have a CSV file named "data.csv" located in your current directory, you can load the dataset into a Pandas Data Frame.

3. we use the head () method on the Data Frame to display the first 5 rows of the dataset.

4.The data. shape attribute is used in Python's Pandas library to determine the dimensions or shape of a Data Frame or NumPy array. Here we have data with 15036 rows and 216 columns, then data. shape will return (15036, 216)

5. the data. duplicated () method is used to find duplicate rows in a data. It returns a Boolean Series that indicates whether each row is a duplicate of a previous row.

6. In machine learning, label encoding is a technique used to convert categorical data (i.e., non-numerical data) into numerical data that can be used in machine learning algorithms. A common use case for label encoding is to encode the class or target variable in a supervised learning problem.

7.In Python's scikit-learn library, you can use the Label Encoder class to perform label encoding on a class or target variable.

8.Next, we create a Label Encoder object and fit it to the class labels using the fit () method. This step maps each unique class label to a unique integer value.

9.Finally, we use the transform () method to transform the class labels using the fitted Label Encoder. The resulting encoded_labels are the numerical representations of the original class labels.

10. the data. corr () method is used to compute pairwise correlation of columns in a data. It returns a new Data frame containing the correlation coefficients between all pairs of columns.

11. The correlation coefficients range between -1 and 1. A coefficient of 1 indicates a perfect positive correlation (i.e., the variables move in the same direction), while a coefficient of -1 indicates a perfect negative correlation (i.e., the variables move in opposite directions). A coefficient of 0 indicates no correlation between the variables.

12. Feature selection is the process of selecting a subset of relevant features (i.e., columns) from a dataset to use in model training. This can help to reduce overfitting, improve model performance, and reduce the computational cost of training

13. Save the selected features to a new CSV file called final.csv

15. To present a correlation matrix as a heatmap using Matplotlib, you can use the matplotlib.pyplot module to plot a 2D heatmap of the correlation matrix which is named as corr. ping

16. A heatmap is a graphical representation of data that uses a color-coding scheme to represent different values. In the context of correlation analysis, a heatmap can be used to visualize the correlation matrix between different variables as shown

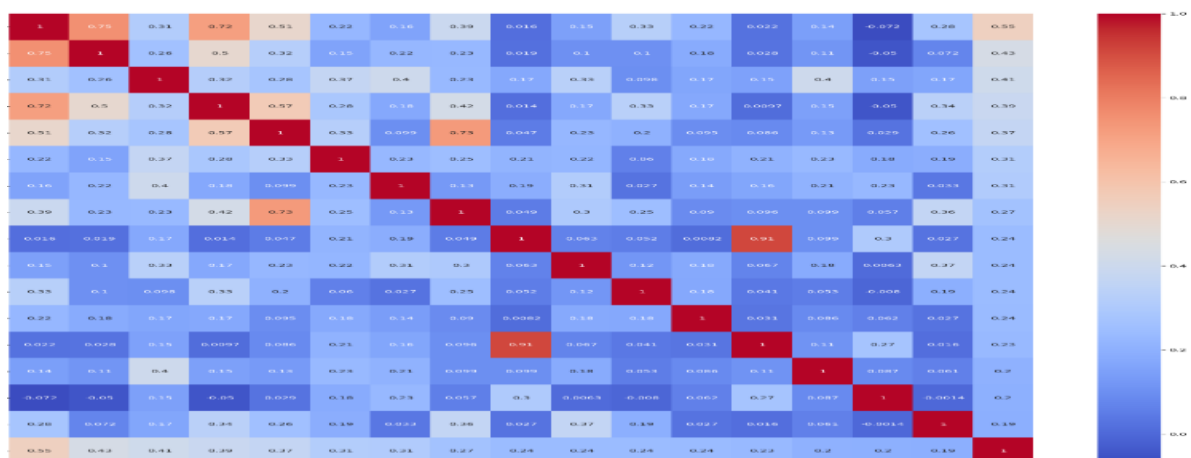


Figure 7 Correlation Ping

17. The `train_test_split ()` function is a utility provided by Scikit-learn library for splitting a dataset into training and testing sets. It takes the input data as well as the target labels and splits them randomly into two separate sets: one for training the model, and another for testing its performance.

18. Next, we load the dataset into a Pandas Data Frame called `dt`. We then separate the features (stored in the `X` variable) from the target variable (stored in the `y` variable).

19. Finally, we call the `train_test_split ()` function and pass in the features and target variables, as well as the test size parameter, which specifies the proportion of the dataset to be used for testing. In this example, we set `test_size` to 0.3, which means 30% of the data will be used for testing. We also set the random state parameter to 42 to ensure that the split is reproducible.

20. The resulting `train_test_split ()` function call will return four variables: `X_train`, `X_test`, `y_train`, and `y_test`, which contain the training and testing sets for the features and target variables. These can then be used to train and evaluate a machine learning model.

21. This code trains a decision tree classifier on a training dataset, uses the trained model to make predictions on a test dataset, and evaluates the performance of the model using accuracy score and confusion matrix.

Here's a step-by-step explanation of the code:

- First, we import the `DecisionTreeClassifier` class from the `sklearn. tree` module and the `accuracy_score` and `confusion_matrix` functions from the `sklearn. metrics` module.
- Next, we create an instance of the `DecisionTreeClassifier` class called `dt`.
- Then, we train the decision tree classifier using the `fit ()` method, passing in the training features (`x_train`) and training labels (`y_train`) as arguments.
- After training the model, we use it to make predictions on the test features (`x_test`) using the `predict ()` method, which returns predicted labels (`y_pred`).
- We then compute the accuracy score by comparing the predicted labels with the true labels from the test set (`y_test`) using the `accuracy_score ()` function.
- Finally, we compute the confusion matrix using the `confusion_matrix ()` function, which returns a matrix showing the number of true positives, false positives, true negatives, and false negatives.

- The `print ()` statements are used to display the results. The first `print ()` statement displays the accuracy obtained by the decision tree classifier, while the second `print ()` statement displays the confusion matrix.

22.This code trains an XG Boost classifier on a training dataset, uses the trained model to make predictions on a test dataset, and evaluates the performance of the model using accuracy score and confusion matrix.

Here's a step-by-step explanation of the code:

- First, we import the XG Boost classifier class from the `sklearn. tree` module and the accuracy score and `confusion_matrix` functions from the `sklearn. metrics` module.
- Next, we create an instance of the XG Boost classifier class called `xgb`.
- Then, we train the decision tree classifier using the `fit ()` method, passing in the training features (`x_train`) and training labels (`y_train`) as arguments.
- After training the model, we use it to make predictions on the test features (`x_test`) using the `predict ()` method, which returns predicted labels (`y_pred`).
- We then compute the accuracy score by comparing the predicted labels with the true labels from the test set (`y_test`) using the `accuracy_score ()` function.
- Finally, we compute the confusion matrix using the `confusion_matrix ()` function, which returns a matrix showing the number of true positives, false positives, true negatives, and false negatives.
- The `print ()` statements are used to display the results. The first `print ()` statement displays the accuracy obtained by the decision tree classifier, while the second `print ()` statement displays the confusion matrix.

23.This code trains an Ada Boost classifier on a training dataset, uses the trained model to make predictions on a test dataset, and evaluates the performance of the model using accuracy score and confusion matrix.

Here's a step-by-step explanation of the code:

- First, we import the Ada Boost classifier class from the `sklearn. tree` module and the accuracy score and `confusion_matrix` functions from the `sklearn. metrics` module.
- Next, we create an instance of the Ada Boost classifier class called `ada`.

- Then, we train the decision tree classifier using the `fit ()` method, passing in the training features (`x_train`) and training labels (`y_train`) as arguments.
- After training the model, we use it to make predictions on the test features (`x_test`) using the `predict ()` method, which returns predicted labels (`y_pred`).
- We then compute the accuracy score by comparing the predicted labels with the true labels from the test set (`y_test`) using the `accuracy_score ()` function.
- Finally, we compute the confusion matrix using the `confusion_matrix ()` function, which returns a matrix showing the number of true positives, false positives, true negatives, and false negatives.
- The `print ()` statements are used to display the results. The first `print ()` statement displays the accuracy obtained by the decision tree classifier, while the second `print ()` statement displays the confusion matrix.

24. This code trains a Support Vector Machine on a training dataset, uses the trained model to make predictions on a test dataset, and evaluates the performance of the model using accuracy score and confusion matrix.

Here's a step-by-step explanation of the code:

- First, we import the Support Vector Machine class from the `sklearn. tree` module and the accuracy score and `confusion_matrix` functions from the `sklearn. metrics` module.
- Next, we create an instance of the Support Vector Machine class called `svc`.
- Then, we train the decision tree classifier using the `fit ()` method, passing in the training features (`x_train`) and training labels (`y_train`) as arguments.
- After training the model, we use it to make predictions on the test features (`x_test`) using the `predict ()` method, which returns predicted labels (`y_pred`).
- We then compute the accuracy score by comparing the predicted labels with the true labels from the test set (`y_test`) using the `accuracy_score ()` function.
- Finally, we compute the confusion matrix using the `confusion_matrix ()` function, which returns a matrix showing the number of true positives, false positives, true negatives, and false negatives.
- The `print ()` statements are used to display the results. The first `print ()` statement displays the accuracy obtained by the decision tree classifier, while the second `print ()` statement displays the confusion matrix.

25.The accuracy score and confusion matrix are obtained with Four algorithms and respective score are shown below with graphical representation using matplotlib.

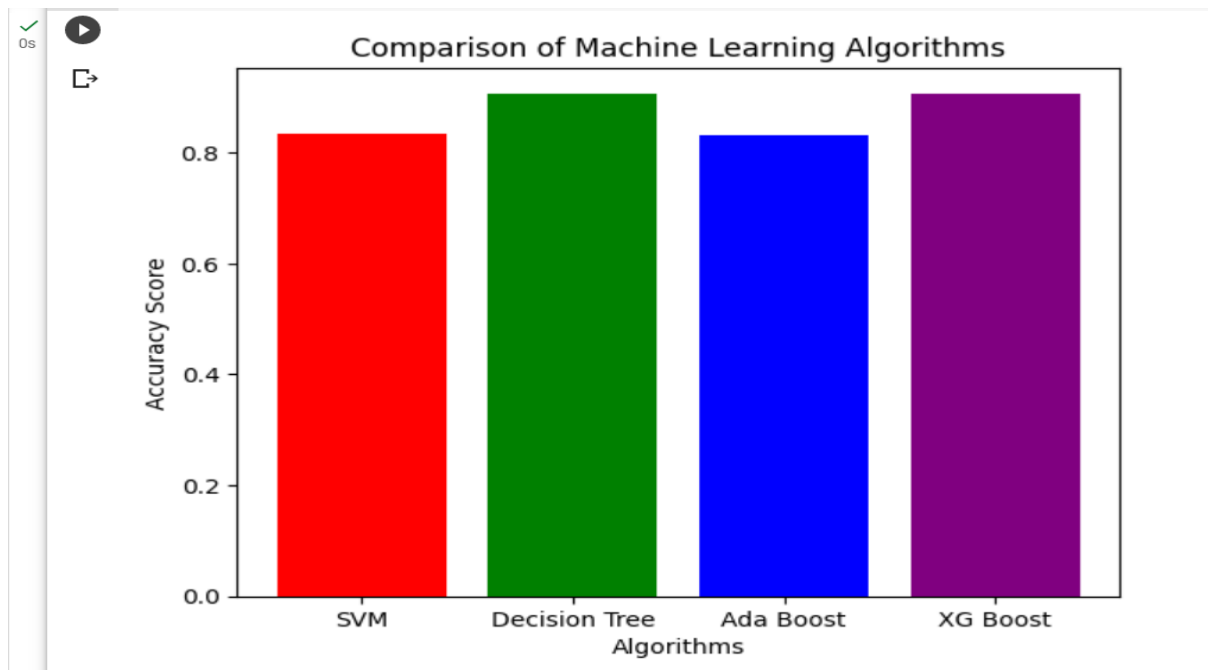


Figure 8 Algorithm Comparison

- Decision tree - 90.7%
- XG Boost - 90.6%
- Ada Boost - 83.2%
- Support Vector Machine - 83.4%

26.Based on above predictions among all, decision tree algorithm is more accurate.

CHAPTER 7

SYSTEM TESTING

Testing is an important part of the software development lifecycle as it ensures that the system is working as intended and meets the project requirements. By performing different types of testing, developers can catch bugs and errors early in the development process, which helps to reduce the cost and time associated with fixing them later.

7.1 Types of Manual testing performed

We performed basically three types of testing for our project, where we tested all the three testing methods i.e., unit test, functional testing and system testing manually. They are explained in detailed below:

Unit Testing

In this project, unit testing could involve testing functions or methods responsible for data pre-processing, model training, and prediction. For example, the unit test for a data pre-processing function could check whether the function correctly scales the data, imputes missing values, or encodes categorical variables. Unit tests performed manually and it can also be performed using framework pytest, which allows developers to write test cases that check the output of each function against expected results.

System testing

In the case of this project, system testing would involve testing the entire web application to ensure that all the pages are loading correctly, the user interface is user-friendly, and the system is responding as expected to user inputs. System testing, we performed manually and can also be performed using automated testing tools like Selenium, which can simulate user interactions with the web application.

Functional Testing

In this project, functional testing would involve testing whether the system can identify anomalous behaviour or not. This could be done by testing the system with a dataset containing known anomalous and non-anomalous behaviours and checking whether the system accurately

identifies them. Functional testing performed manually and can also be performed using automated testing tools like pytest or Robot Framework.

Test Cases:

| Input | Output | Result |
|----------------------------------|---|---------|
| Input for the anomalous behavior | Predicting the anomalous behavior from the user input | Success |

Test cases Model building:

| S.NO | Test cases | I/O | Expected O/T | Actual O/T | P/F |
|------|--------------------|---|---|---|-----|
| 1 | Read the datasets. | Need to provide the dataset path and the data should be in the form of CSV. | Data loaded successfully | Valid data format | P |
| 2 | Read the datasets. | Need to provide the dataset path and the data should be in the form of CSV. | Data loaded successfully | Dataset is not in CSV format | F |
| 3 | Preprocess Data | Need to enter the split size (20-30%) for training and testing | Data preprocessed and splits successfully | Data preprocessed and splits successfully | P |
| 4 | Preprocess Data | If the split size is more or less than the mentioned size | Data preprocessed and splits successfully | Model may get under fit or over fit | F |
| 5 | Model | Models need to trained with the training dataset | Models trained successfully | Accuracy obtained by each model | P |
| 6 | Model | If no model selected for training | Select any model for training | Select any model for training Select any model for training | F |
| 7 | Prediction | Enter details to predict the anomalous behavior | Predict result as anomalous behavior or not | Predict result as anomalous behavior or not | P |

CHAPTER 8

RESULTS

HOME PAGE:

The home page in project serves as the main entry point for the web application. The page includes two main sections: "Let's Start" and "Contact Us". The "Let's Start" and "Contact Us" sections are strategically placed to encourage users to take action and get involved in the project.

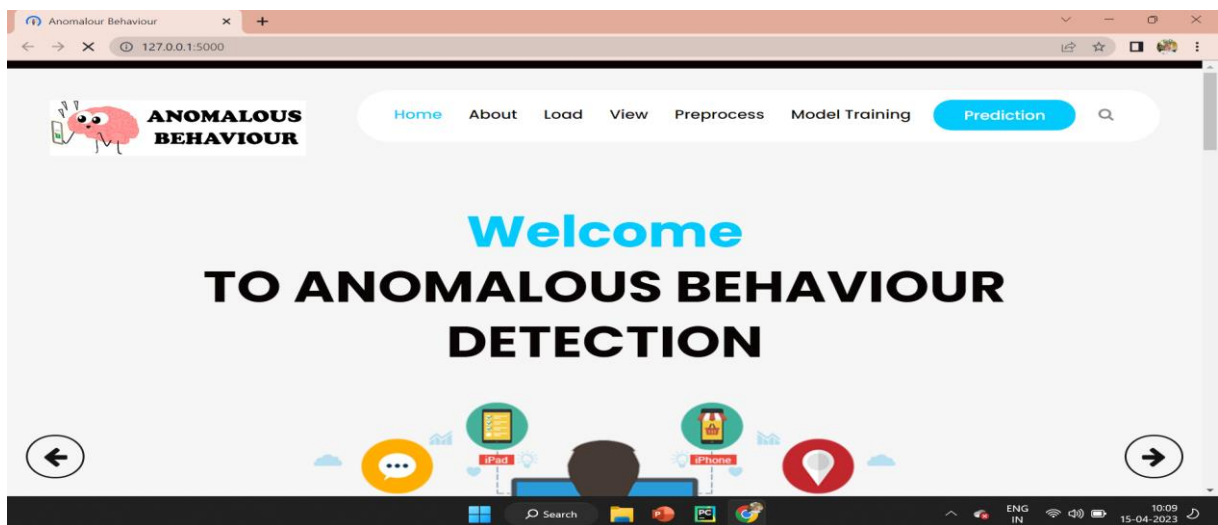


Figure 9 Home Page

LOAD PAGE:

The load page in the project serves as a platform for the user to upload their dataset to the system. The page usually contains a button labeled "Choose File" that users can click to browse their computer and select the file(s) they want to upload.

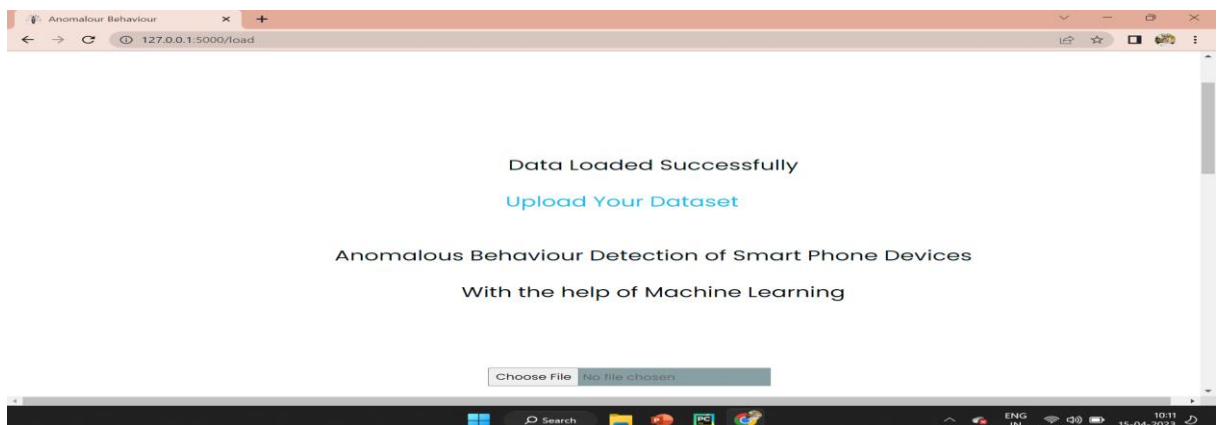
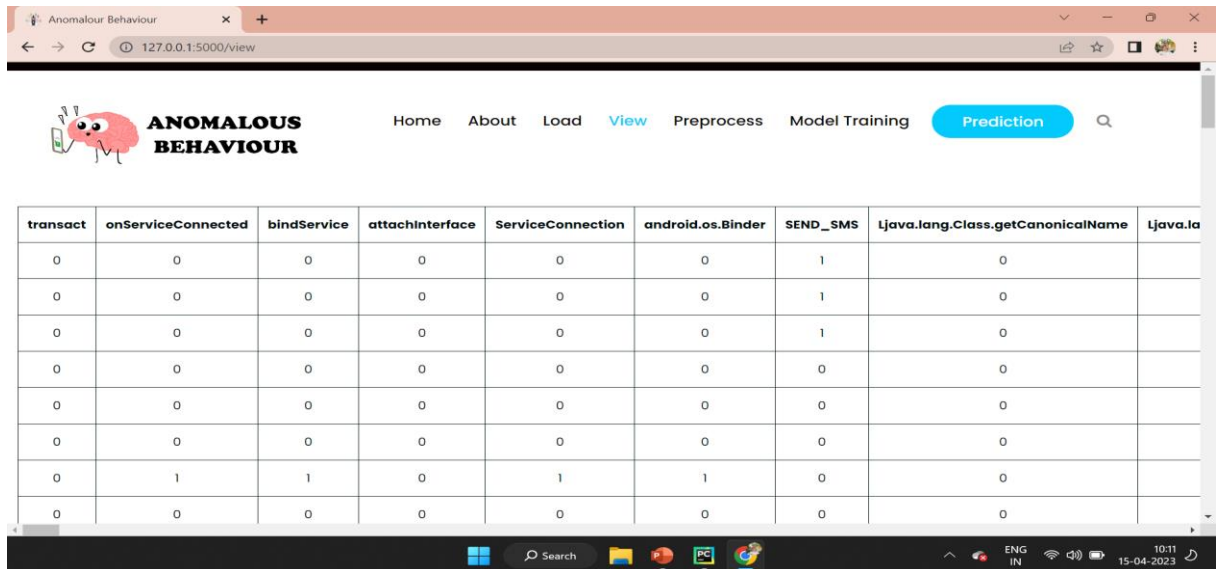


Figure 10 Load Page

VIEW PAGE:

The View page project displays the values stored in the uploaded dataset file. The page is designed to show up to 100 rows and 216 columns of data, making it easy for the user to get a quick overview of the contents of their dataset.

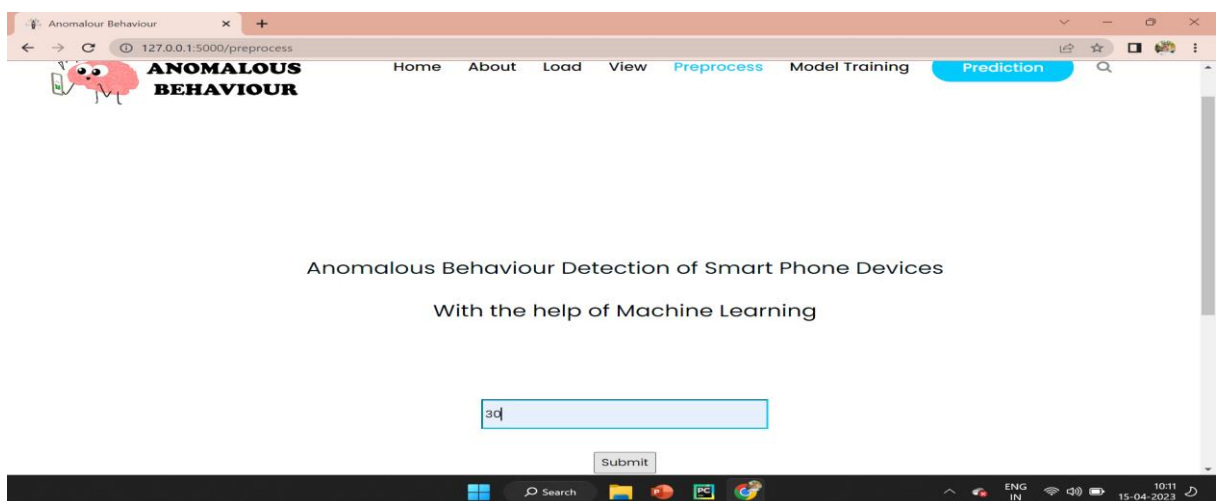


| transact | onServiceConnected | bindService | attachInterface | ServiceConnection | android.os.Binder | SEND_SMS | Ljava.lang.Class.getCanonicalName | Ljava.la |
|----------|--------------------|-------------|-----------------|-------------------|-------------------|----------|-----------------------------------|----------|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Figure 11 View Page

PRE-PROCESS PAGE:

The pre-process page in the project serves the purpose of splitting the dataset into training and testing sets. This page asks the user to input the percentage of data they would like to split for testing purposes, with the range set between 20-30%.



Anomalous Behaviour Detection of Smart Phone Devices
With the help of Machine Learning

30

Submit

Figure 12 Pre-process Page

MODEL TRAINING PAGE:

The Model Training page in the project allows the user to choose from a range of machine learning algorithms like Decision Tree, AdaBoost, XG Boost, and SVM. Once the user selects an algorithm, the page initiates the training process on the pre-processed data and generates an accuracy score for the selected algorithm.

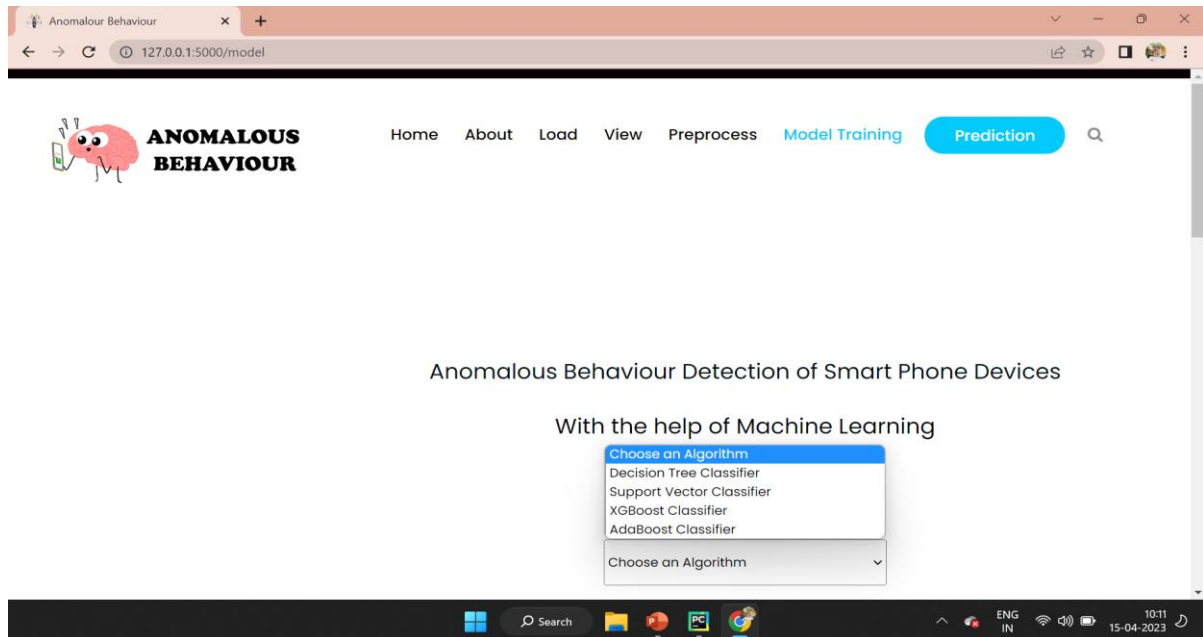


Figure 13 Model Training Page

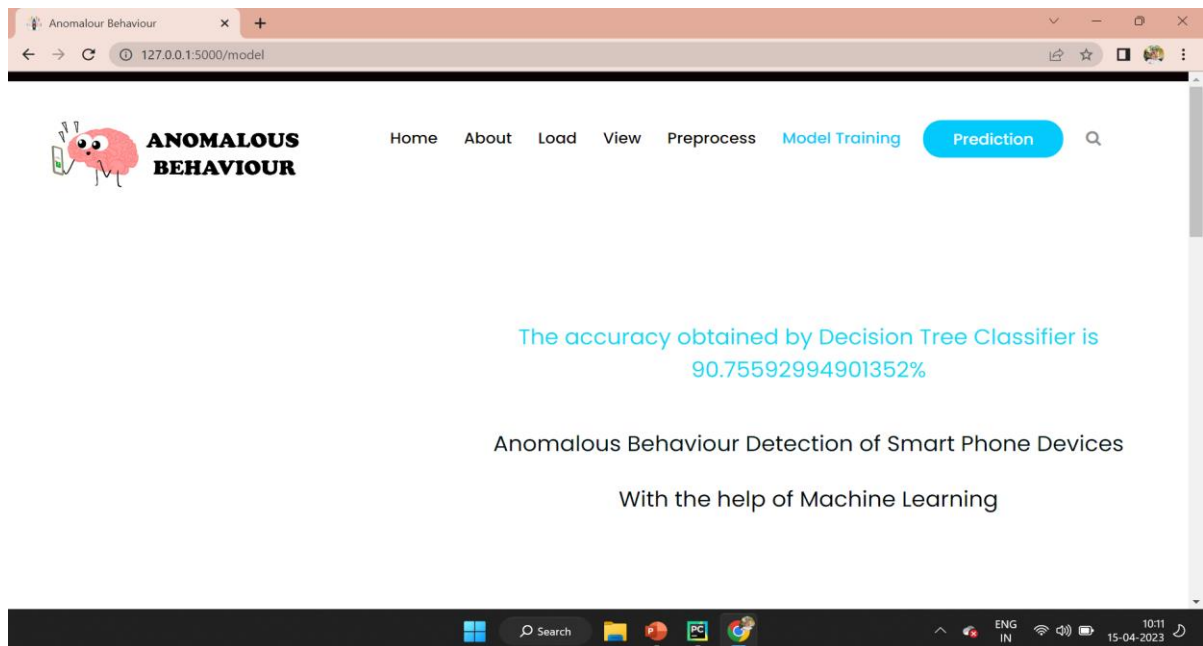


Figure 14 Accuracy Score Report

PREDICTION PAGE:

The prediction page is where users can enter their preferences to predict whether their smartphone has anomalous behaviour or not. The page allows the user to select the columns they want to consider for prediction then report generated indicating whether the smartphone has anomalous behaviour or not.

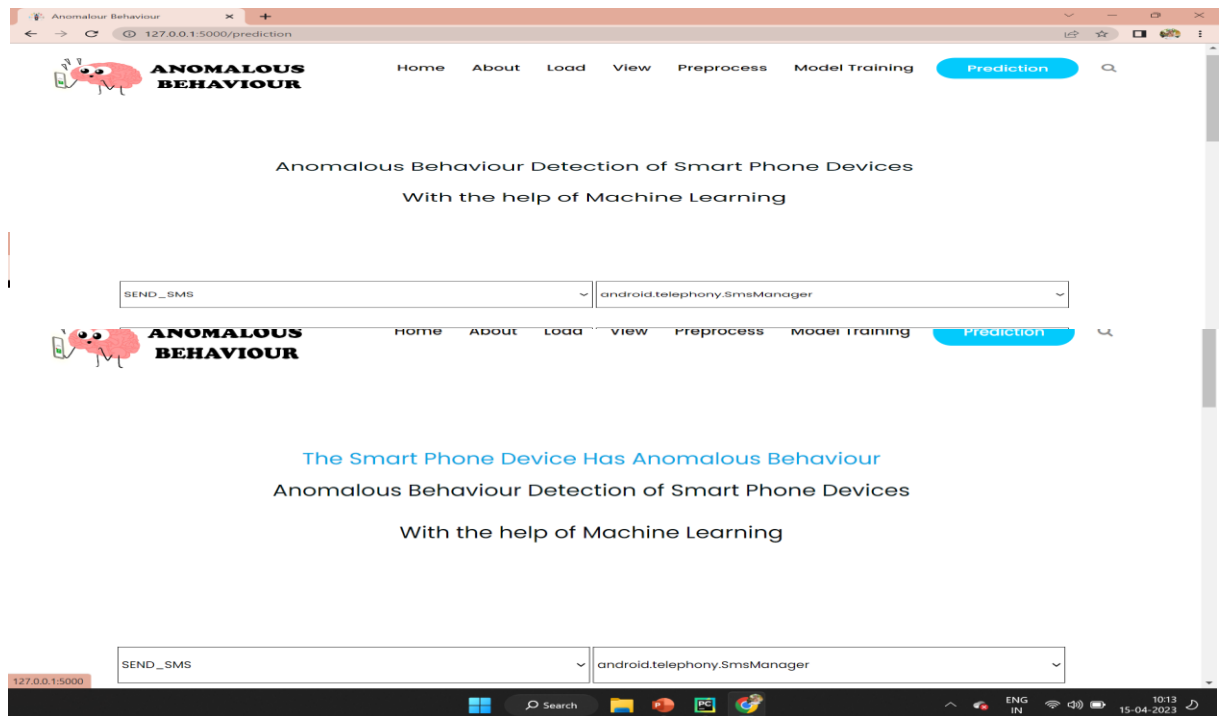


Figure 15 Prediction Page

CHAPTER 9

CONCLUSION

In conclusion, this project aimed to develop a methodology for detecting anomalous activity on smartphones using change point analysis and machine learning algorithms. The approach involved collecting data through an automated tool that generates user inputs to trigger harmful programs, analysing the collected data using change point analysis to extract characteristics from power usage, and training machine learning models to detect anomalous behaviour.

This novel methodology for identifying anomalous smartphone behaviour is proposed. The approach uses changepoint detection theory to extract features and three machine learning techniques to train a classifier based on the power consumed by the smartphone. The proposed methodology outperforms existing methods in terms accuracy score, and it can recognize malware operating within short timeframes, which is a significant advantage over other techniques.

The proposed methodology involves an automated data collection tool, which utilizes an efficient mix of user inputs to trigger malicious behaviour. The data collector then records the device's power consumption, which provides a summary of software changes.

Two different techniques are used in the data analysis step to extract features from the power usage data, including parametric and non-parametric changepoints. These features are then fed into four different machine learning algorithms, including the support vector machine, the decision tree, XG boost and the AdaBoost algorithm. These algorithms have been shown to be effective at accurately detecting anomalous behaviour in smartphone applications.

Future Work

Future work for this project includes applying the methodology to real malware instead of using an emulated malware. This would provide more insight into the effectiveness of the approach in detecting actual malicious behaviour. Additionally, further research could investigate the use of additional machine learning algorithms to improve accuracy and expand the range of anomalous behaviour that can be detected.

CHAPTER 10

REFERENCES

- [1] D. Evans, “The internet of things: How the next evolution of the internet is changing everything,” CISCO white paper, Tech. Rep., 2011.
- [2] Statista, “Number of mobile phone users worldwide from 2015 to 2020 (in billions).” [Online].
- [3] A. Arabo and B. Pranggono, “Mobile malware and smart device security: Trends, challenges and solutions,” in 2013 19th International Conference on Control Systems and Computer Science, May 2013, pp. 526–531.
- [4] T. Kim, B. Kang, M. Rho, and et. all, “A multimodal deep learning method for android malware detection using various features,” IEEE Trans. on Info. Forensics and Security, vol. 14, no. 3, 2019.
- [5] Y.-S. Yen and H.-M. Sun, “An android mutation malware detection based on deep learning using visualization of importance from codes,” Microelectronics Reliability, vol. 93, pp. 109–114, 2019.
- [6] D. Arp, M. Spreitzenbarth, M. Huber, H. Gascon, K. Rieck, and C. Siemens, “Drebin: Effective and explainable detection of android malware in your pocket.” in Ndss, vol. 14, 2014, pp. 23–26.
- [7] P. Faruki, A. Bharmal, V. Laxmi, and et. all, “Android security: A survey of issues, malware penetration, and defenses,” IEEE Communications Surveys Tutorials, vol. 17, no. 2, pp. 998–1022, Secondquarter 2015.
- [8] K. Ariyapala, H. G. Do, H. N. Anh, and et. all, “A host and network- based intrusion detection for android smartphones,” in 30th Int. Conf. on Advanced Info. Net. and Apps Workshops (WAINA), March 2016.
- [9] M. Curti, A. Merlo, M. Migliardi, and S. Schiappacasse, “Towards energy-aware intrusion detection systems on mobile devices,” in Int. Conf. on High Performance Computing Simulation (HPCS), July 2013.