##### REAL TIME MALWARE AND

##### ADWARE PROTECTION

**A PROJECT REPORT**

###### ***Submitted by***

**OISHI BASAK (20BAI10092)**

**SAHIL ARORA (20BAI10264)**

**NIKHIL (20BAI10275)**

**ABHISHEK KUMAR (20BAI10384)**

*in partial fulfillment for the award of the degree*

*of*

##### BACHELOR OF TECHNOLOGY

*in*

# COMPUTER SCIENCE AND ENGINEERING WITH AI & ML

****

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

**VIT BHOPAL UNIVERSITY**

**KOTHRIKALAN, SEHORE**

**MADHYA PRADESH - 466114**

DECEMBER 2021

**VIT BHOPAL UNIVERSITY, KOTHRIKALAN, SEHORE**

**MADHYA PRADESH – 466114**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“REAL TIME MALWARE AND**

**ADWARE PROTECTION”** is the bonafide work of “**OISHI BASAK (20BAI10092), SAHIL ARORA (20BAI10264), NIKHIL (20BAI10275), ABHISHEK KUMAR (20BAI10384)”** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported at this time does not form part of any other project/research work based on which a degree or award was conferred on an earlier occasion on this or any other candidate.

**PROGRAM CHAIR PROJECT GUIDE**

Dr S Sountharrajan, Associate Professor  **DR. LAKSHMI D (ASSOCIATE PROFESSOR) (SCSE)**

School of Computer Science and Engineering School of Computer Science and Engineering

VIT BHOPAL UNIVERSITY VIT BHOPAL UNIVERSITY

The Project Exhibition I Examination is held on 22ND December 2021.

**ACKNOWLEDGEMENT**

First and foremost I would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

I wish to express my heartfelt gratitude to Dr. S. Poonkuntran, Head of the Department, School of Computer Science and Engineering for much of his valuable support and encouragement in carrying out this work.

I would like to thank my internal guide Dr Lakshmi D , for continually guiding and actively participating in my project, giving valuable suggestions to complete the project work.

I would like to thank all the technical and teaching staff of the School of Computer Science and Engineering , who extended directly or indirectly all support.

Last, but not least, I am deeply indebted to my parents who have been the greatest support while I worked day and night for the project to make it a success.

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **ABBREVIATION** | **WORD** |
| ML | Machine Learning |
| AI | Artificial Intelligence |
| KNN | K Nearest Neighbors |
| ANN | Artificial Neural Network |
| ROC Curve | Receiver Operating Characteristic Curve |
| DL | Deep Learning |
| SVM | Support Vector Machine |
| CART | Classification And Regression Tree |
| CNN | Convolutional Neural Network |
| DOM | Document Object Model |
| VCM | Variance Considered Machine |
| Iframe | Inline Frame |
| Ad | Advertisement |
| URL | Uniform Resource Locator |
| HTML | HyperText Markup Language |
| CSS | Cascading Style Sheets |

**LIST OF FIGURES AND GRAPHS**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **TITLE** | **PAGE NO.** |
| 1. | SYSTEM ARCHITECTURE DIAGRAM | 16 |
| 2. | SOFTWARE ARCHITECTURAL DIAGRAM | 29 |
| 3. | CLASSIFICATION MODELS- ACCURACY AND F1 SCORE | 36-38 |
| 4. | COMPARISON OF MODELS | 39 |
| 5. | ROC CURVE | 40 |
| 6. | ACCURACY AND COST OF MODELS | 41 |

**ABSTRACT**

The most threatening problem is, of the different malware that can attack and harm your system. Malware is one of the most serious security threats on the Internet today. In fact, most Internet problems such as spam e-mails and denial of service attacks have malware as their underlying cause. That is, computers that are compromised with malware are often networked together to form botnets, and many attacks are launched using these malicious, attacker-controlled networks. In order to deal with the new malware generated, new techniques to detect them and prevent any damage caused by them. The idea of machine learning is to let the algorithm learn by itself the best parameters from data in order to make good predictions. There are many different applications, in our case we will consider using machine learning algorithm to classify binaries between legitimate and malicious binaries. Our aim is to investigate on how to implement machine learning to malware detection in order to detection unknown malware. To develop a malware detection software that implement machine learning to detect unknown malware. To validate that malware detection that implement machine learning will be able to achieve a high accuracy rate with low false positive rate.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| i.  ii.  iii.  iv. | List of Abbreviations  List of Figures and Graphs  List of Tables  Abstract | 4  5  6  7 |
| 1 | **CHAPTER-1:**  **PROJECT DESCRIPTION AND OUTLINE** Introduction 1.2 Motivation for the work  1.3 About Introduction to the project including  techniques  1.5 Problem Statement  1.6 Objective of the work  1.7 Organization of the project  1.8 Summary | 11  11-12  13-14  14-15  15  15-16  17  17-18 |
| 2 | **CHAPTER-2:**  **RELATED WORK INVESTIGATION**  2.1 Introduction  2.2 Core area of the project  2.3 Existing Approaches/Methods  2.3.1 Approaches/Methods -1  2.3.2 Approaches/Methods -2  2.4 Pros and cons of the stated Approaches/Methods  2.5 Issues/observations from investigation  2.6 Summary | 18  18-19  19  19-20  20-21  21-23    23-24.  24 |
| 3 | **CHAPTER-3:**  **REQUIREMENT ARTIFACTS**  3.1 Introduction  3.2 Hardware and Software requirements  3.3 Specific Project requirements  3.3.1 Data requirement  3.3.2 Functions requirement  3.3.3 Performance and security requirement  3.4 Summary | 25  25  25-26  26  27  27-28  28 |
| 4 | **CHAPTER-4:**  **DESIGN METHODOLOGY AND ITS NOVELTY**  4.1 Methodology and goal  4.2 Functional modules design and analysis  4.3 Software Architectural designs  4.4 Summary | 29  29-30  31  31 |
| 5 | **CHAPTER-5:**  **TECHNICAL IMPLEMENTATION & ANALYSIS**  5.1Outline  5.2 Technical coding and code solutions  5.3 Working Layout of Forms  5.4 Prototype submission  5.5 Test and validation  5.6 Performance Analysis(Graphs/Charts)  5.7 Summary | 32  32  32-35    35-37    38  38-41  42-43  43-44 |
| 6 | **CHAPTER-6:**  **PROJECT OUTCOME AND APPLICABILITY**  6.1Outline  6.2 key implementations outlines of the System  6.3 Significant project outcomes  6.4 Project applicability on Real-world applications  6.4 Inference | 45  45  45-46  46  46 |
| 7 | **CHAPTER-7:**  **CONCLUSIONS AND RECOMMENDATION**  7.1Outline  7.2 Limitation/Constraints of the System  7.3 Future Enhancements  7.4 Inference | 47  47    48    48 |
|  | Related Work Investigation  References | 49  50 |

**CHAPTER 1 : PROJECT DESCRIPTION AND OUTLINE**

**1.1 Introduction**

The Internet has become an indispensable part of our life as it connects us to the entire world virtually but it opens the path for various people with malevolent intents, who strive to attack and harm legitimate users in different ways for various reasons. Most of the time the reason is money. A point to be noted in this respect is the use of malevolent software which is installed in the computer without the consciousness of its owner – this software can steal confidential data and also allow remote access which may cause the denial of services (DNS) in the system . To protect legalized users from various threats, security vendors such as antivirus software provide detection and analysis procedures. Various online tools can dynamically analyze the malware and detect it, the tools use cloud computing hence they are more efficient and safer. The main idea of this study is to identify various online malware analysis tools and compare them based on their analysis

**MALWARE :**

* Malware is an intrusive software designed to damage and destroy computers and computer systems. Malware is **an** **abbreviation** for **"malicious** **software".** It is any type of software created to harm or exploit another piece of software or hardware. Examples of common malware includes viruses, worms, Trojan viruses, spyware, ransomware, etc.
* It is a file or code, typically delivered over a network, that infects, explores, steals or conducts virtually any behavior an attacker wants. Since its birth more than 30 years ago, malware has found several methods of attack. They include email attachments, malicious advertisements on popular sites (malversating), fake software installations, infected USB drives, infected apps, phishing emails and even text messages.
* Malware can be about making money off you, sabotaging your ability to get work done, making a political statement, or just bragging rights. Although malware cannot damage the physical hardware of systems or network equipment, it can steal, encrypt, or delete your data, alter or hijack core computer functions, and spy on your computer activity without your knowledge or permission.

**ADWARE:**

* Adware is a type of malware  that **shows** unwanted **emerging screens (and,** **sometimes,** **irritants)** **that** can **be displayed** on your computer or mobile device. **It** is a type of malicious software that bombards you with incessant pop-ups. Adware, often called advertising-supported software by its developers, is a software that generates revenue for its developer by automatically generating online advertisements in the user interface of the software or on a screen presented to the user during the installation process.
* Adware has the potential to become malicious and harm your device by slowing it down, hijacking your browser and installing viruses and/or spyware. They can come in the form of pop-ups which usually come packaged with other hidden malware threats. The advertisements produced by adware are sometimes in the form of a pop-up, sometimes in an "unclosable window", and sometimes injected into web pages.
* Malware includes viruses, spyware, or Trojan horse programs. Adware can include non-malicious types that are voluntarily downloaded. The problem is that these programs can tie up system resources and slow the computer down. The ads can also be annoying and interfere with productivity.
* Adware has the potential to become malicious and harm your device by slowing it down, hijacking your browser and installing viruses and/or spyware.

**1.2 Motivation for the work**

Some web applications contain code embedded from the ad networks, which provides the interface for the ad networks to serve ads. This capability has been abused by attackers where the landing page of the advertisements coming from ad networks links to malicious content. Moreover, intrusive

advertisements significantly affect the user experience on mobile phones due

to limited screen size. They also drain significant energy and network data. Web advertising also has severe privacy implications for users. A primary motivation for hackers is the money they can obtain by stealing your passwords, bank details, holding your customer information for ransom or selling your data to competitors or on the dark web.

Advertisers use third party web-tracking by embedding code in the websites the users visit, to identify the same users again in a different domain, creating a more global view of the user browsing behavior . Private user information is collected, stored and sold to other third party advertisers. These elaborate user profiles can be used to infer sensitive information about the users like medical history or political views. Communication with these third party services is unencrypted, which can be exploited by attackers.

The security and privacy concerns surrounding web advertising has motivated research in ad blocking tools from both academia and industry notably Adblock Plus, Ghostery, Brave, Mozilla, Opera and Apple. Ad blocking serves to improve web security, privacy, usability, and performance. As of February 2017, 615 million devices had ad blockers installed. However, recently Google Chrome and Safari proposed changes in the API exposed to extensions, with the potential to block extension based ad-blockers. This motivates us for the need of native ad blockers like Brave, Opera, AdGraph and even PERCIVAL.

Finally, in terms of practice, our findings indicate that attempts to encourage more secure behavior should focus on:

1) Emphasizing the effectiveness of threat avoidance safeguards

2) Improving individuals’ beliefs regarding their ability to implement and use threat safeguards

3) Reducing individuals’ perceptions concerning the level of effort needed to implement threat safeguards.

4) Cyber security technology vendors and policy writers can use these findings to develop products, processes, and messages that effectively encourage more secure behavior

**1.3 Introduction to the project including techniques**

* This project is applying ML algorithms in individual malware risk prediction and comparing predictive accuracy with existing models. In our model, unobtrusive ads aren't being blocked in order to support websites. Acceptable Ads are allowed by default to support websites.

We will demonstrate a range of ML algorithms like Random Forest Classification, Naïve Bayes, Decision Tree Classifiers, SVM and Neural Networks (DL) with cross-validations, which is lacking in other applications of ML for malware or adware detection.​ Naïve Bayes is the simplest classification algorithm which is fast to form. It is a popular (baseline) method for text categorization, the matter of judging documents as belonging to a particular category or the opposite with word frequencies due to the features. Decision trees are used for classification and regression. Both the classification or regression challenges are working perfectly for this well-known supervised machine learning algorithm. For SVM however, it’s mostly employed in classification problems. Random forest is like bootstrapping algorithm with a call tree (CART) model. The last word prediction might be a function of each prediction. This final prediction can simply be the mean of every prediction. So it is very effective for filtration system.

* Ranking variable importance may inform algorithm selection with diverse predictive risk factors for future development of new risk prediction models.

**1.4 Problem Statement**

It has been found that Dynamic Malware analysis is always better than Static Malware analysis, but for analyzing the malware in the computers the malicious codes must be executed such that the tools can keep a check on the activities taking place in the computer. This procedure has a high probability of damaging the computer as the malware runs freely on the computer. There are various online-based malware analysis tools identified which work on cloud computing and online virtual machines. These tools provide efficient malware analysis and no harm to the user’s computer. The goal of this paper is to analyze the different dynamic malware analysis tools, mainly the online-based ones and compare them among each other for identifying the best.

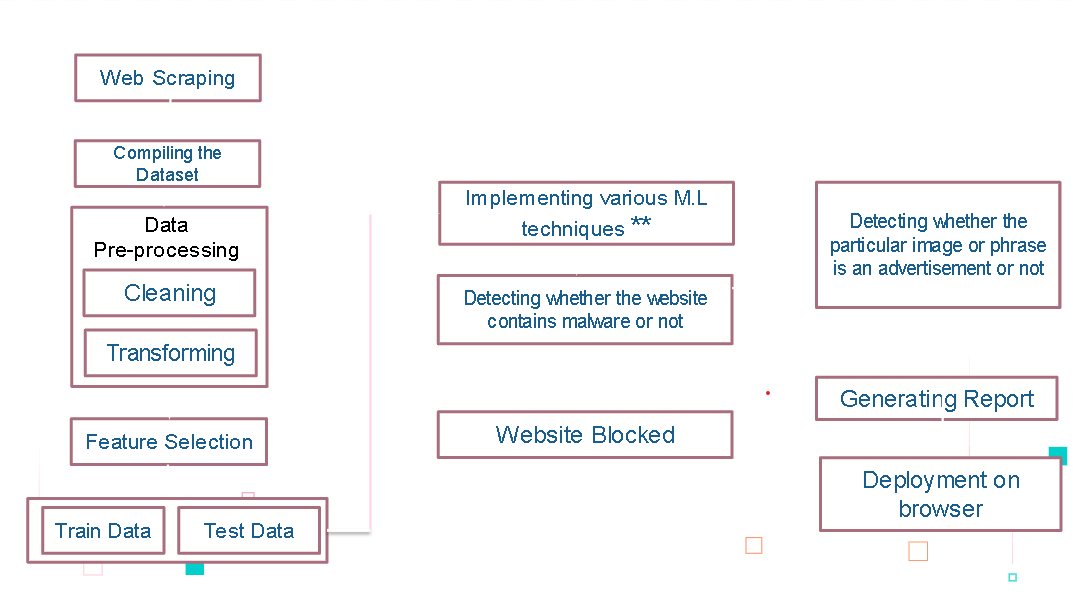
The malware for the experiment purpose was downloaded from Tekdefence.com. The malware 1.exe was used for analysis in all the tools such that the results could be compared more efficiently.

**1.5 Objective of the work**

In the last two decades, a variety of different ML techniques and feature selection algorithms have been widely applied to malware detection, predictions and blocking.

* This model (given that it gets accurate data and prediction after choosing the correct algorithm and proper use of feature engineering) can be used in blocking malicious advertisements and malwares.
* This model will help in reducing the human efforts which will help the user to take proper decisions and steps on time resulting in hassle-free access to websites. It will block ads that interrupt the browsing experience. Blocking annoyances like video ads, pop-ups, flashing banners and more means pages load faster.
* Traditional security product uses virus scanner to detect malicious code, these scanner uses signature which created by reverse engineering a malware. But with malware that became polymorphic or metamorphic the traditional signature-based detection method used by antivirus is no long effective against the current issue of malware (Willems, G., Holz, T. & Freiling, F., 2007). In current anti-malware products, there are two main task to be carried out from the malware analysis process, which are malware detection and malware classification. The main objective of malware detection is to be able to detect malware in the system. There are two type of analysis for malware detection which are dynamic analysis and static analysis. For effective and efficient detection, the uses of feature extraction are recommended for malware detection (Ahmadi, M. et al., 2016). There are various type of detection method, the method that we are using will be detecting through hex and assembly file of the malware. Feature will be extracted from both hex view and assembly view of malware files. After extracting feature to its category, all category is to be combine into one feature vector for the classifier to run on them (Ahmadi, M. et al., 2016). For feature selection, separating binary file into blocks to be compare the similarities of malware binaries. This will reduce the analysis overhead which cause the process to be faster (Kim, T.G., Kang, B. & Imp, E.G., 2013). To build a learning algorithm, feature that are extracted with the label will be undergo classification with using any classification method for example Random Forest, Neural Network, N-gram, KNN and many others, but Support Vector Machine (VCM) is recommended for the presence of noise in the extracted feature and the label (Stevin, P. & Bystrov, I., 2016). As to generate result, the learning model is to test with dataset with label to generate a graph which indicate detection rate and false positive rate. To find the best result, repeat the process using many other classification and create learning model to test on the same dataset. The best result will the one graph that has the highest detection rate and lowest false positive rates (Linzi, A. et al., 2010).

**1.6 Organization of the project**



**1.7 Summary**

In this Chapter , we have discussed about what is malware and adware, how it attacks our systems, what was our motivation for our work i.e. what is the real idea behind our project, then we have talked about the different techniques which will be used in our Project, then we have discussed the problem statement, objective and the organization of the project.

In the last two decades, a variety of different ML techniques and feature selection algorithms have been widely applied to malware detection, predictions and blocking them. ​Our model is a clubbed version of malware detection, adware detection from real data taken from the existing active websites. Also, for user’s discretion we are going to deploy it in the form of an extension on Google chrome browser which 95% of people use worldwide.

This model (given that it gets accurate data and prediction after choosing the correct algorithm and proper use of feature engineering) can be used in blocking malicious advertisements and malwares.​ It will help in reducing the human efforts which will help the user to take proper decisions and steps on time resulting in hassle-free access to websites. It will block ads that interrupt the browsing experience. Blocking annoyances like video ads, pop-ups, flashing banners and more means pages will load faster.​

**CHAPTER-2: RELATED WORK INVESTIGATION**

**2.1 Introduction**

The most threatening problem is, of the different malware that can attack and harm your system. Malware is one of the most serious security threats on the Internet today. In fact, most Internet problems such as spam e-mails and denial of service attacks have malware as their underlying cause. That is, computers that are compromised with malware are often networked together to form botnets, and many attacks are launched using these malicious, attacker controlled networks.

In order to deal with the new malware generated, new techniques to detect them and prevent any damage caused by them.The research papers related to malware analysis stated various tools and techniques which can be potentially followed to detect and analyze the malware. There are two basic methods of analyzing the malware, one is Static and the other is Dynamic. Most of the studies derives that, compared to the static analysis, dynamic analysis is much more efficacious and accurate. Specific malware cases may show the characteristics of various sections at the same time. The tools must be powerful enough to detect different malware efficiently. There are different techniques identified under dynamic analysis – Process Call Monitoring, Process Parameter Analysis, Tracking of information, Instruction Tree, Auto-Start Extensibility.

**2.2 Core area of the project**

The core area of our Project is Machine Learning and Cyber Security. To investigate how to implement machine learning to malware detection in order to detect unknown malware. To develop a malware detection software that implements machine learning to detect unknown malware. To validate that malware detection that implements machine learning will be able to achieve a high accuracy rate with low false positive rate.

**2.3 Existing Approaches/Methods**

There exists some existing models and approaches to detect malware and

prevent it. We will discuss some here:

**2.3.1 Approaches/Methods -1**

**ADBLOCK PLUS →**

**• Perceptual ad blocking in Chromium-based browsers.**

PERCIVAL is deployed in two Chromium-based browsers: Chromium and Brave. We demonstrate two deployment scenarios; first, PERCIVAL blocks ads synchronously as it renders the page, with a modest performance overhead. Second, PERCIVAL classifies images asynchronously and memorizes the results, thus speeding up the classification process1 .

**• Lightweight and accurate deep learning models.**

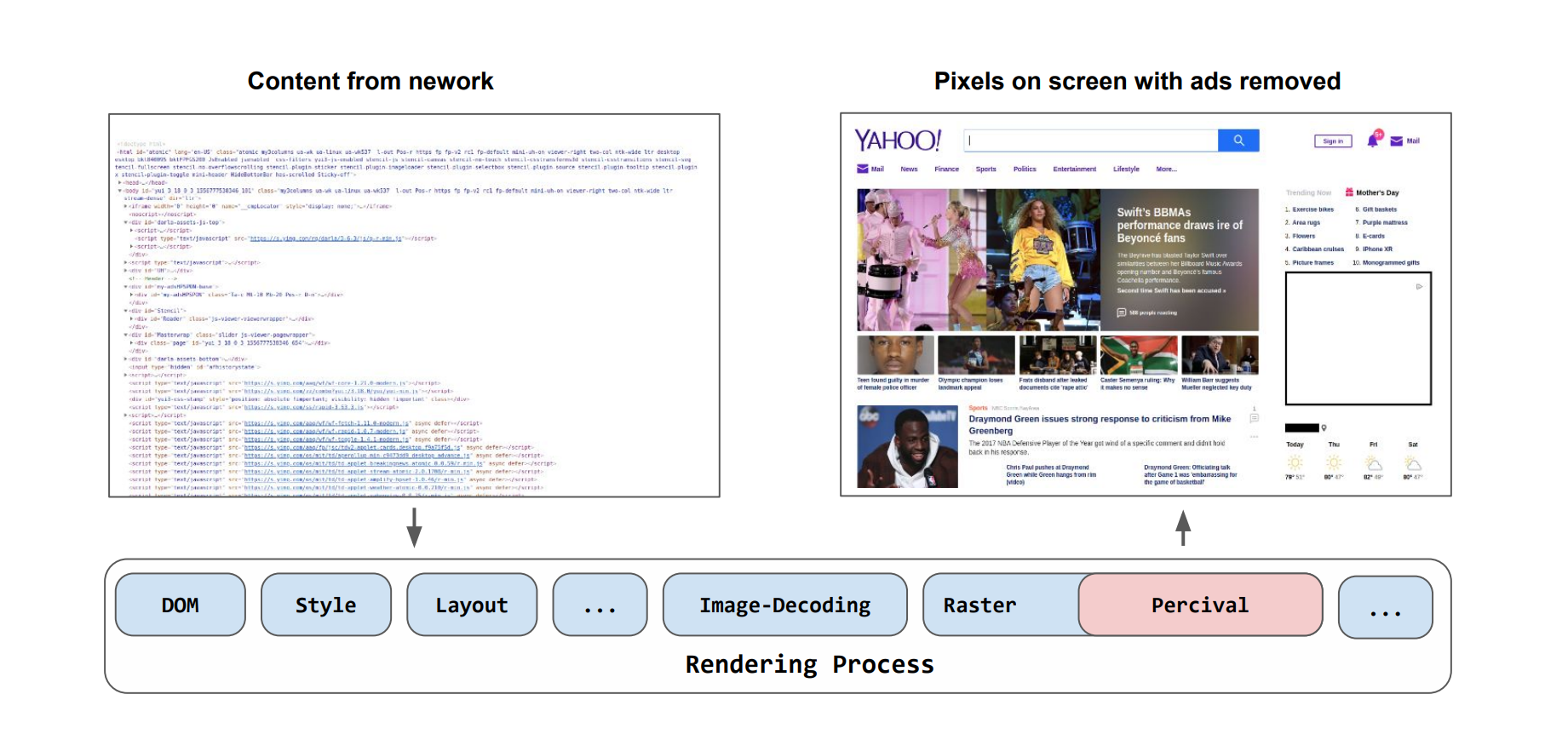
Ad blocking can be done effectively using highly-optimized deep neural network-based models for image processing has been proved by the model. Previous studies suggest that models over 5MB in size become hard to deploy on mobile devices; because of our focus on low-latency detection, we create a compressed in-browser model that occupies 1.76MB on disk, which is smaller by factor of 150 compared to other models of this kind , while maintaining similar accuracy results

**2.3.2 Approaches/Methods -2**

**PERCIVAL →**

This paper presents PERCIVAL, a novel deep-learning based system for blocking ads. Our primary goal is to build a system that blocks ad images that could escape detection by current techniques, while remaining small and efficient enough to run in a mobile browser.

Figure 1 shows how PERCIVAL blocks rendering of ads. First, PERCIVAL runs in the browser image rendering pipeline. By running in the image rendering pipeline, PERCIVAL can inspect all images before the browser shows them to the user. Second, PERCIVAL uses a deep convolutional neural network (CNN) for detecting ad images. Using CNNs enables PERCIVAL to detect a wide range of ad images, even if they are in a language that PERCIVAL was not trained on. This section discusses PERCIVAL’s architecture overview, possible alternative implementations and detection model. We can discuss the detailed design and implementation for our browser modifications and our detection model.



**2.4 Pros and cons of the stated Approaches/Methods:**

**PROS:**

**AD BLOCK PLUS:**

Ad blockers that inspect web pages based on the DOM such as Ad Highlighter are prone to DOM obfuscation attacks. They assume that the elements of the DOM strictly correspond to their visual representation. For instance, an ad blocker that retrieves all img tags and classifies the content contained in these elements does not consider the case, where a rendered image is a result of several CSS or JavaScript transformations and not the source contained in the tag. These ad blockers are also prone to resource exhaustion attacks where the publisher injects a lot of dummy elements in the DOM to overwhelm the ad blocker.

Additionally, a native implementation is much faster than a browser extension implementation with the added benefit of having access to the unmodified image buffers.

**PERCIVAL:**

One alternative to running PERCIVAL directly in the browser could have

been to run PERCIVAL in the browser’s JavaScript layer via an extension. However, this would require scanning the DOM to find image elements, waiting for them to finish loading, and then screenshotting the pixels to run

the detection model. The advantage of a JavaScript-based system is that it works within current browser extensibility mechanisms, but recent work has shown how attackers can evade this style of detection.

Moreover, the native implementation is much faster than Implementation of browser extensions with additional benefits To access the unchanged image buffer.

**CONS:**

**AD BLOCK PLUS:**

Dangling Text: By testing ADBLOCK PLUS integrated into Chromium, we noticed the following limitations. Many ads consist of multiple elements, which contain images and text information layered together. ADBLOCK PLUS is positioned in the rendering engine, and therefore it has access to one image at a time. This leads to situations where we effectively block the image, but the text is left dangling. Although this is rare, we can mitigate this by retraining the model with ad image frames containing just the text. Alternatively, a non-machine learning solution would be to memorize the DOM element that contains the blocked image and filter it out on consecutive page visitations. Although this might provide an unsatisfying experience to the user, we argue that it is of the benefit of the user to eventually have a good ad blocking experience, even if this is happening on a second page visit.

**PERCIVAL:**

Small Images: Currently, images that are below 100×100 size skips PERCIVAL to reduce the processing time. This is a limitation which can be alleviated by deferring the classification and blocking of small images to a different thread, effectively blocking asynchronously. That way we make sure that we don’t regress the performance significantly, while we make sure that consecutive requests will continue blocking small ads.

**2.5 Issues/observations from investigation**

**Accuracy Against EasyList**

For this experiment, we crawl Alexa top 500 news websites as opposed to Alexa top 1000 websites used in the crawl for training. This is because news websites are an excellent source of advertisements and the crawl can be completed relatively quickly. Also, Alexa top 500 news websites serve as a test domain different from the train domain we used previously. For our comparison we create two data sets: First, we apply EasyList rules to select DOM elements that potentially contain ads (IFRAMEs, DIVs, etc.); we then capture screenshots of the contents of these elements. Second, we use resource-blocking rules from EasyList to label all the images of each page according to their resource URL. After crawling, we manually label the images to identify the false positives resulting in a total of 6,930 images

**Blocking Facebook Ads**

To evaluate PERCIVAL’s performance on Facebook, we browse Facebook with PERCIVAL for a period of 35 days using two non-burner accounts that have been in use for over 9 years. Every visit is a typical Facebook browsing session, where we browse through the feed, visit friends’ profiles, and different pages of interest. For desktop computers two most popular places to serve ads is the right-side columns and within the feed (labelled sponsored) . For our purposes, we consider content served in these elements as ad content and everything else as non-ad content. A false positive (FP) is defined as the number of non-ads incorrectly blocked and false negative (FN) is the number of ads PERCIVAL missed to block. For every session, we manually compute these numbers. Figure a shows the aggregate numbers from all the browsing sessions undertaken. Figure b shows PERCIVAL blocking right-side columns correctly

**FIGURE a**

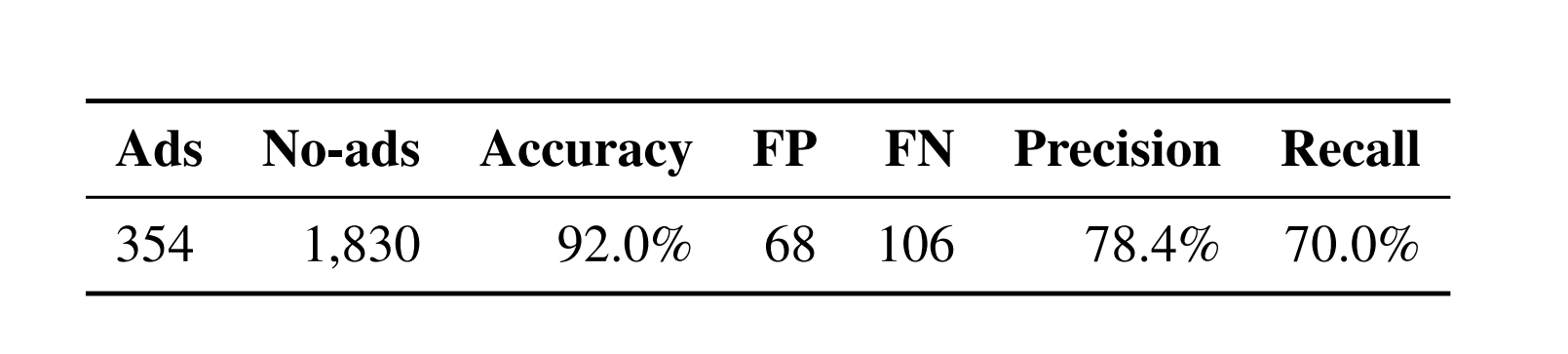


Figure a: Online evaluation of Facebook ads and sponsored content.

**FIGURE b :**

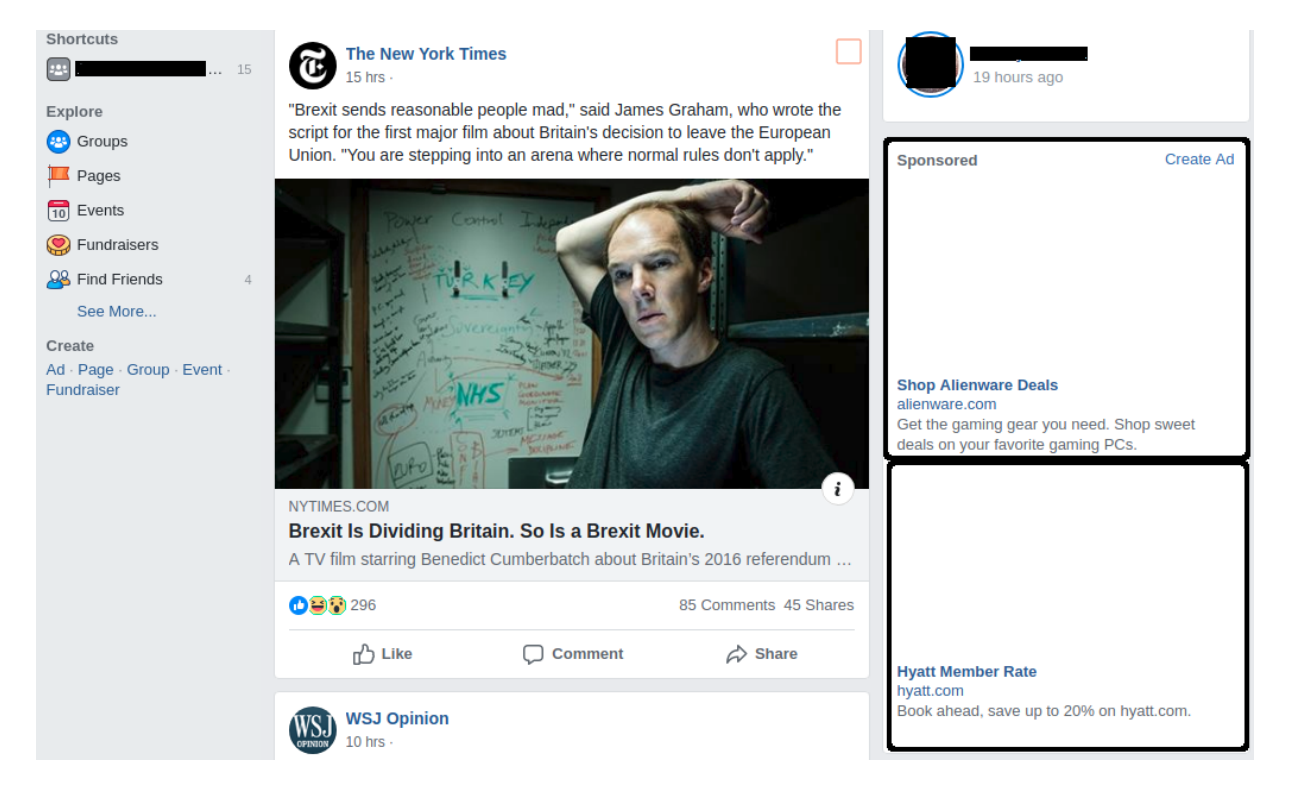
****

Figure b : The screenshots show one of the author’ Facebook home page accessed with PERCIVAL. The black rectangles are not part of the original screenshot.

**2.6 Summary**In this Chapter we have discussed the core area of the project, the existing approaches which we are already in use i.e. AD BLOCK PLUS & PERCIVAL, pros and cons of their approaches and the issues and observation with their used approaches and methods.

**CHAPTER-3:** **REQUIREMENT ARTIFACTS**

**3.1 Introduction**

In this we are going to discuss the Hardware, Software and Data requirements which are used for making our Project i.e. **“REAL-TIME MALWARE AND ADWARE DETECTION”.** Anti-malware companies turned to machine learning, an area of computer science that had been used successfully in image recognition, searching and decision-making, to augment their malware detection and classification. Today, machine learning boosts malware detection using various kinds of data on host, network and cloud-based anti-malware components

**3.2 Hardware and Software requirements**

**3.2.1 Hardware Requirements**

Talking about the minimum Hardware Requirements which we will be using for our project are :

For this we will be using a PC with following specifications:

→ It must have at least 16Gb of DDR4 RAM

→ It must have a NVIDIA GTX 1080 (4 GB RAM) Graphic Card

→ It must have a Intel Core i5-9300H or above Processor.

→ Hard disk : 100 GB (minimum) and above

→ We can use any Operating software like Windows, MacOS, or

Linux (Ubuntu).

**3.2.2 Software Requirements**

Talking about the software requirements which we will be using for our project are:

→ Visual Studio Code (Visual Studio Code is a source-code editor )

→ Jupyter Labs (web-based interactive development environment )

→ PyCharm ( integrated development environment used in computer

programming)

→ Languages: Python and Web browsers: Chrome, Firefox

→ Microsoft Excel ( For creating our .csv dataset)

**3.3 Specific Project requirements**

**3.3.1 Data requirement**

→ For Data Requirement and extraction We will be doing Web Scraping with the help of Beautiful Soup. It extracts content and data from a website. Unlike screen scraping, which only copies pixels displayed onscreen, web scraping extracts underlying HTML codes, and with it data stored in a database. The scraper can then replicate the entire website content elsewhere.

→ Data collection is the process of gathering and measuring information on  variables of interest, in an established systematic fashion that enables one to answer stated research questions, test hypotheses, and evaluate outcomes.

→ Data wrangling is the process of cleaning and unifying messy and complex data sets for easy access and analysis. It is also known as data cleaning. It encompasses all the work done on your data prior to the actual Analysis.

→ Data normalization is a process in which data attributes within a data model are organized  to increase the cohesion of entity types. The goal of data normalization is to reduce and even eliminate data redundancy.

**3.3.2 Functions requirement**

i7 family processors were the best particularly for extreme 3D gaming, intensive graphics tasks, multimedia production in standard computer level. Now, this processor becomes a previous generation CPU, for the reason that Intel introduced new and improved processors called 2nd generation Intel® Core™ processors family. But this doesn’t mean that we shouldn’t buy a Core i7 processor. It still does the job it is designed for. Core i7 processor has several versions both in high end and budget groups. Depending on your work type and budget, you can pick the right one.

**Core i7 General specification**

- All support 64-bit execution

- Integrate 4 Cores (latest Core i7 processor incorporate 6 cores)

- Speed ranges from 2.66GHz to 3.33GHz

- Front Side Bus Speed include 2GHz, 4.8GHz or 6.4GHz

- Support DDR3 main memory

- Support Hyper-threading technology

**3.3.3 Performance and security requirement:**

**Performance And Security of Intel core i7 processor**

The Core i7 processors series targets the gaming industry and for the applications that demand efficient performance and high-end functioning. Generally, Core i7 processor is recommended for:-

- Multitasking, for running multiple programs at the same time

- Multithreading applications

- Intel® hardware-enabled security boosts protection and enables the ecosystem to better defend against evolving and modern cybersecurity threats. Silicon-enabled security technologies help create a trusted foundation, protect workloads, and improve software resilience.

- Creating professional movies and editing graphical tasks

- More than enough for basic tasks such as word processing, internet

browsing and email

* Foundational Security: critical protection to help verify trustworthiness of devices and data.
* Workload and Data Protection: trusted execution for hardware-isolated data protection.
* Software Reliability: platforms that help protect against a range of cybersecurity threats.

**3.4 Summary**

In this Chapter we have discussed the Software and Hardware requirements which will be used in our project , then we have discussed some specific project requirements where we have discussed about the Data Requirements, Function Requirement , and Performance and security requirements. Many kinds of research show that one single malware couldn’t be analyzed in a single tool. Experimental results show that every malware analysis tool has a different metric and way to analyze the malicious code. PERCIVAL was recognized to be the best among the selected ones as it can take the output from Virus Total and simultaneously use a unique Genetic software mapping algorithm. This algorithm is comparing the genes with various other malware for coming to a solid result.

**CHAPTER-4: DESIGN METHODOLOGY AND ITS NOVELTY**

**4.1 Methodology and goal**

* Our Ad Blocker not only blocks ads but also scraps the website, prepares the dataset for malware , checks what is an advertisement and finally returns the output with the help of machine learning.
* It does not actually monitor the browsing history or require any personal information to work properly unlike other ad blockers. Our adblocker will block the annoying, irrelevant and intrusive ads. Therefore, an effective filtering technology is a significant contribution to the sustainability of cyberspace and our society.
* With our ad blocker, avoiding tracking and malware is easy. Blocking intrusive ads reduces the risk of malvertising infections viruses or bugs and also stops companies from following anybody’s online activity and compromise on their online privacy. We will use neural network to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates using multiple hidden layers and non-linear activation functions to detect the malware present inside a website, if it finds positive malware, it blocks the website from the browser but if it cant detect it, it automatically implements the adware function to detect the presence of advertisements in the website.

**4.2 Functional modules design and analysis**

**1.  Web Scraping -**

It extracts content and data from a website. Unlike screen scraping, which only copies pixels displayed on screen, web scraping extracts underlying HTML codes, and with it, data stored in a database. The scraper can then replicate the entire website content elsewhere.

1. **Data Collection –**

Data collection is the process of gathering and measuring

information on variables of interest, in an established systematic

fashion that enables one to answer stated research questions, test

hypotheses, and evaluate outcomes.

**3.   Data Wrangling –**

        Data wrangling is the process of cleaning and unifying messy and

         complex data sets for easy access and analysis. It is also known as

data cleaning. It encompasses all the work done on your data prior to

the actual Analysis.

**4.    Data Normalization -**

Data normalization is a process in which data attributes within a data model are organized to increase the cohesion of entity types. The goal of data normalization is to reduce and even eliminate data redundancy.

**5.  Feature Selection -**

Feature Selection is the process where you automatically or manually select those features which contribute most to your prediction variable or output in which you are interested in.

**6.   Model Building -**

Model building is the process of developing a probabilistic model that best describes the relationship between the dependent and independent variables.

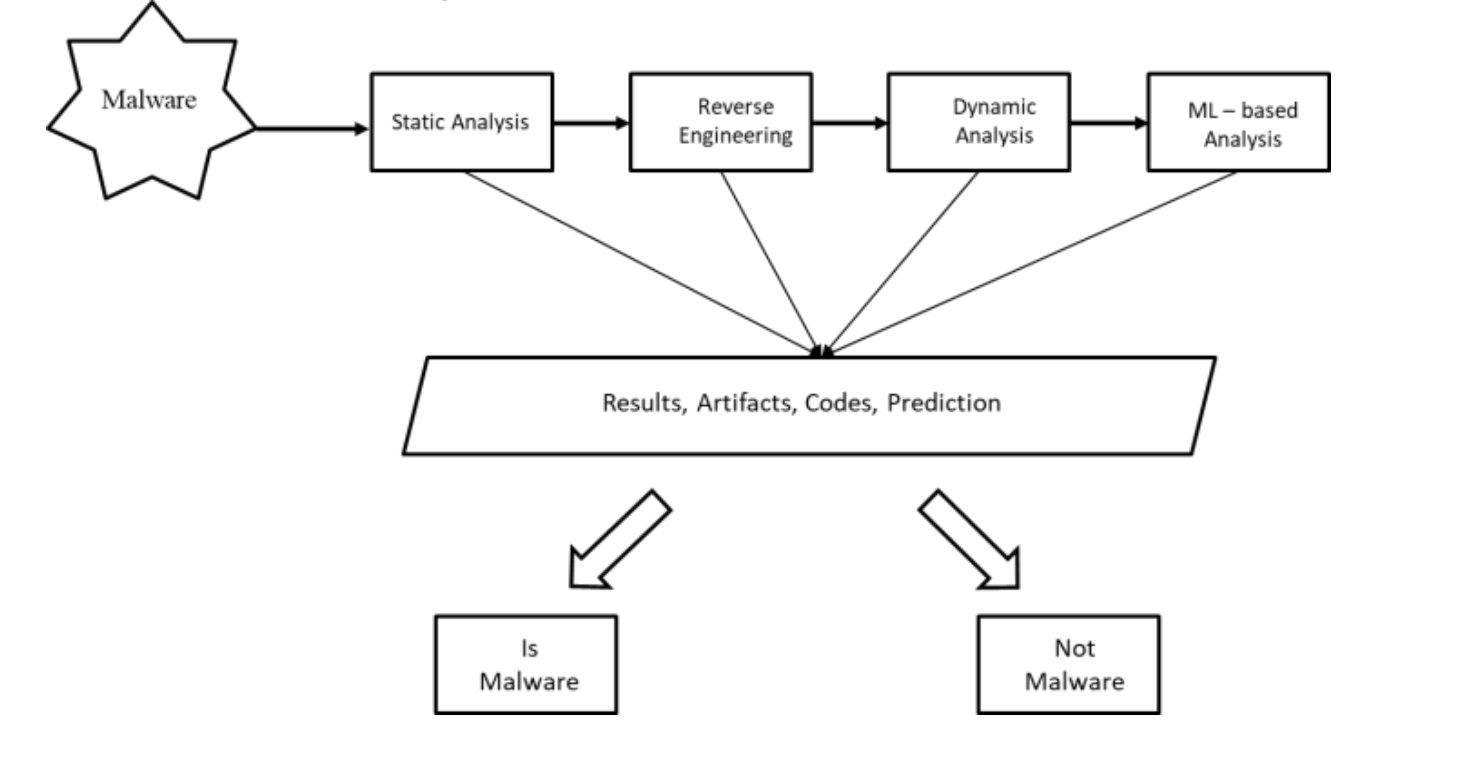
**7.   Data Visualization-**

Data visualization is the process of translating large data sets and metrics into charts, graphs and other visuals. The resulting visual representation of data makes it easier to identify and share real-time trends, outliers, and new insights about the information represented in the data.

**8.   Deployment-**

The concept of deployment in data science refers to the application of a model for prediction using a new data.

**4.3 Software Architectural designs**

****

**This is our Project Software architecture Design**

So first is web scraping which will extract the data from website then the compilation of the data set is done which we got from web scraping then pre processing of the data which will be needed for building our model , and for feature selection we will be training and testing the model by implementing various Machine Learning Algorithms and at last it will detect whether the website contains malware or not if yes then website will be blocked and if no then it will detect whether the particular image or phrase is an advertisement or not and it will generate the report of it and finally the deployment will take place in browser.

**CHAPTER 5: TECHNICAL IMPLEMENTATION AND ANALYSIS**

**5.1 Outline**

An efficient, robust and scalable malware recognition module is the key component of every cybersecurity product. Malware recognition modules decide if an object is a threat, based on the data they have collected on it. This data may be collected at different phases:

– Pre-execution phase data is anything you can tell about a file without executing it. This may include executable file format descriptions, code descriptions, binary data statistics, text strings and information extracted via code emulation and other similar data.

– Post-execution phase data conveys information about behavior or events caused by process activity in a system.

A machine learning algorithm discovers and formalizes the principles that

underlie the data it sees. With this knowledge, the algorithm can ‘reason’ the properties of previously unseen samples. In malware detection, a previously unseen sample could be a new file. Its hidden property could be malware or benign. A mathematically formalized set of principles underlying data properties is called the model.

**5.2 Technical coding and code solutions**

**5.2.1. Web Scraping**

import csv

from bs4 import BeautifulSoup

from selenium import webdriver

from selenium.webdriver.chrome.options import Options

def get\_url(search):

template = 'https://www.amazon.in/s?k={}&ref=nb\_sb\_ss\_ts-doa-p\_1\_7'

search\_term = search.replace(' ', '+')

#adding term query to url

url = template.format(search\_term)

#add page query placeholder

url += '&page{}'

return url

def extract\_record(item):

#Extract and return data from a single record

#description and url

atag = item.h2.a

description = atag.text.strip()

url = 'https://www.amazon.in' + atag.get('href')

#price

price\_parent = item.find('span', 'a-price')

price = price\_parent.find('span', 'a-offscreen').text

#ratings

rating = item.i.text

result = (description, price, rating, url)

return result

def main(search\_term):

#starting the webdriver

driver = webdriver.Chrome()

record = []

url = get\_url(search\_term)

for page in range(1, 21):

driver.get(url.format(page))

soup = BeautifulSoup(driver.page\_source, 'html.parser')

results = soup.find\_all('div', {'data-component-type': 's-search-result'})

for item in results:

record = extract\_record(item)

if record:

records.append(record)

driver.close() #closing of driver

#saving the data to csv file

with open('results.csv', 'w', newline = '', encoding = 'utf-8') as f:

writer = csv.writer(f)

writer.writerow(['Description','Price','Rating','Url'])

writer.writerows(records)

**5.2.2 Initialization of Machine Learning Model**

*The dataset is loaded from the file and is saved in memory.*

import pandas as pd

import matplotlib as plt

import numpy as np

df=pd.read\_csv('MalwareData.csv',sep='|')

**5.2.3 Feature Extraction**

*Since every feature is in numeric form, therefore there is no need for this particular step.*

**5.2.4 Feature Selection**

df=df.drop(['Name','md5'],axis=1)

5.2.4 Splitting the data

X=df.iloc[:,:-1]

y=df.iloc[:,-1]

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2)

**5.2.5 Data Pre-processing**

*Standard Scalar*

Standardize features by removing the mean and scaling to unit variance The standard score of a sample x is calculated as: z = (x - u) / s where u is the mean of the training samples or zero if with\_mean=False, and s is the standard deviation of the training samples or one if with\_std=False. Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using transform. Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data.

from sklearn.preprocessing import StandardScaler

SS = StandardScaler()

Train = SS.fit\_transform(Train)

**5.3 Working Layout of Forms**

*Applying Classification Models*

1. **Logistic Regression:**

Logistic regression is a machine learning algorithm for classification. In this algorithm, the probabilities describing the possible outcomes of a single trial are modelled using a logistic function.

Advantages: Logistic regression is designed for this purpose (classification), and is most useful for understanding the influence of several independent variables on a single outcome variable.

Disadvantages: Works only when the predicted variable is binary, assumes all predictors are independent of each other, and assumes data is free of missing values.

from sklearn.linear\_model import LogisticRegression

lr = LogisticRegression()

lr.fit(X\_train,y\_train)

y\_pred=lr.predict(X\_test)

1. **Naïve Bayes:**

Naïve Bayes algorithm based on Bayes’ theorem with the assumption of independence between every pair of features. Naïve Bayes classifiers work well in many real-world situations such as document classification and spam filtering.

Advantages: This algorithm requires a small amount of training data to estimate the necessary parameters. Naïve Bayes classifiers are extremely fast compared to more sophisticated methods.

Disadvantages: Naïve Bayes is known to be a bad estimator.

from sklearn.naive\_bayes import GaussianNB

nb = GaussianNB()

nb.fit(X\_train,y\_train)

y\_pred1 = nb.predict(X\_test)

1. **K-Nearest Neighbor:**

Neighbors based classification is a type of lazy learning as it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the k nearest neighbors of each point.

Advantages: This algorithm is simple to implement, robust to noisy training data, and effective if training data is large.

Disadvantages: Need to determine the value of K and the computation cost is high as it needs to computer the distance of each instance to all the training samples.

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()

knn.fit(X\_train,y\_train)

y\_pred2=knn.predict(X\_test)

1. **Decision Tree:**

Given a data of attributes together with its classes, a decision tree produces a sequence of rules that can be used to classify the data.

Advantages: Decision Tree is simple to understand and visualize, requires little data preparation, and can handle both numerical and categorical data. Disadvantages: Decision tree can create complex trees that do not generalize well, and decision trees can be unstable because small variations in the data might result in a completely different tree being generated.

from sklearn.tree import DecisionTreeClassifier

tr = DecisionTreeClassifier()

tr.fit(X\_train,y\_train)

y\_pred3=tr.predict(X\_test)

1. **Random Forest:**

Random forest classifier is a meta-estimator that fits a number of decision trees on various sub-samples of datasets and uses average to improve the predictive accuracy of the model and controls over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement.

Advantages: Reduction in over-fitting and random forest classifier is more accurate than decision trees in most cases.

Disadvantages: Slow real time prediction, difficult to implement, and complex algorithm.

from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()

rf.fit(X\_train,y\_train)

y\_pred4=rf.predict(X\_test)

1. **Neural Network:**

Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

Advantages: Most useful for understanding the influence of several independent variables on a single outcome variable. Multiple layers of training and rely on training data to learn and improve their accuracy over time, allowing us to classify and cluster data at a high velocity.

Disadvantages: Time consuming.

model = Sequential()

model.add(Dense(16, input\_dim=54, activation="relu"))

model.add(Dense(8, activation="relu"))

model.add(Dense(4, activation="relu"))

model.add(Dense(1, activation="sigmoid"))

model.summary()#print model Summary

#compile model

model.compile(loss="binary\_crossentropy", optimizer="rmsprop", metrics=["accuracy"])

#Fit model

model.fit(X\_train, y\_train, epochs=5, batch\_size=32)

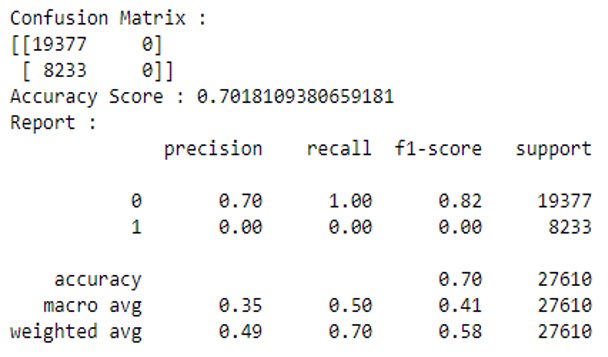
**5.4 Prototype submission**

Python is so flexible and easy to use because of its available packages that are hosted on pypi.org. We will be uploading our package to Pypi. The steps for deployment are:

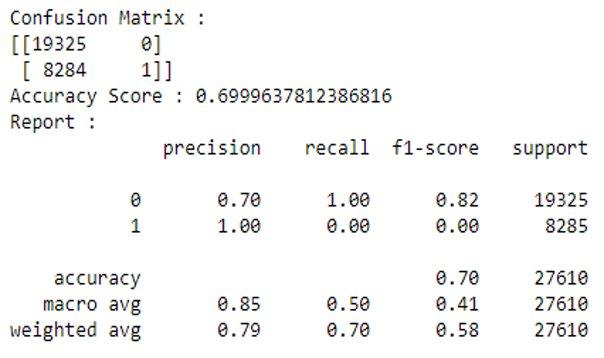
* The twine library is created to simplify uploading packages in Pypi. To install twine library.
* pip install twine
* Get your package ready.
* Make your code publish-ready
* Create a python package
* Create the files PyPi needs
* Create a PyPi account
* Upload your package to github.com
* Upload your package to PyPi
* Install your own package using pip
* Project link: <https://pypi.org/project/malkit/>
* pip install malkit

**5.5 Test and validation**

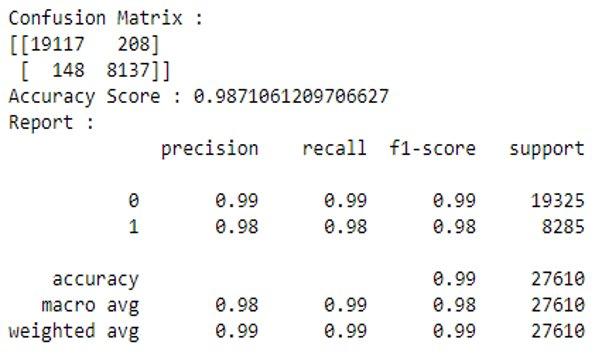
1. **LOGISTIC REGRESSION**

****

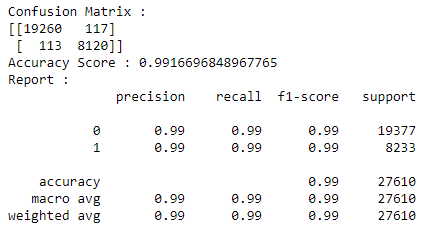
1. **NAÏVE BAYES CLASSIFICATION**



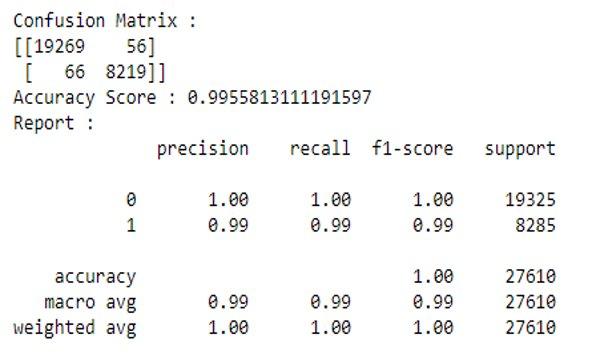
1. **K-NEIGHBOURS CLASSIFICATION​**

****

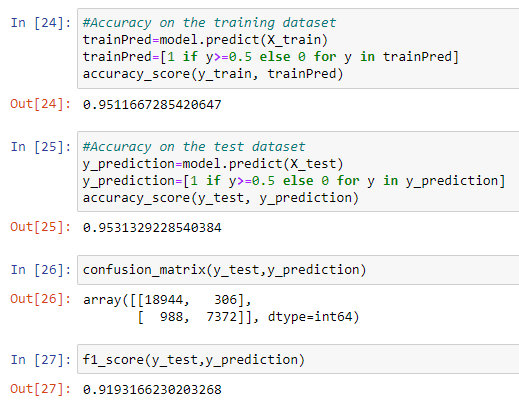
1. **DECISION TREE CLASSIFICATION​**



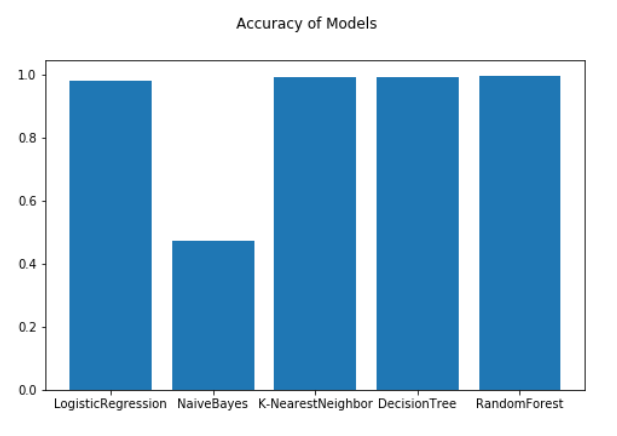
1. **RANDOM FOREST CLASSIFICATION​**

****

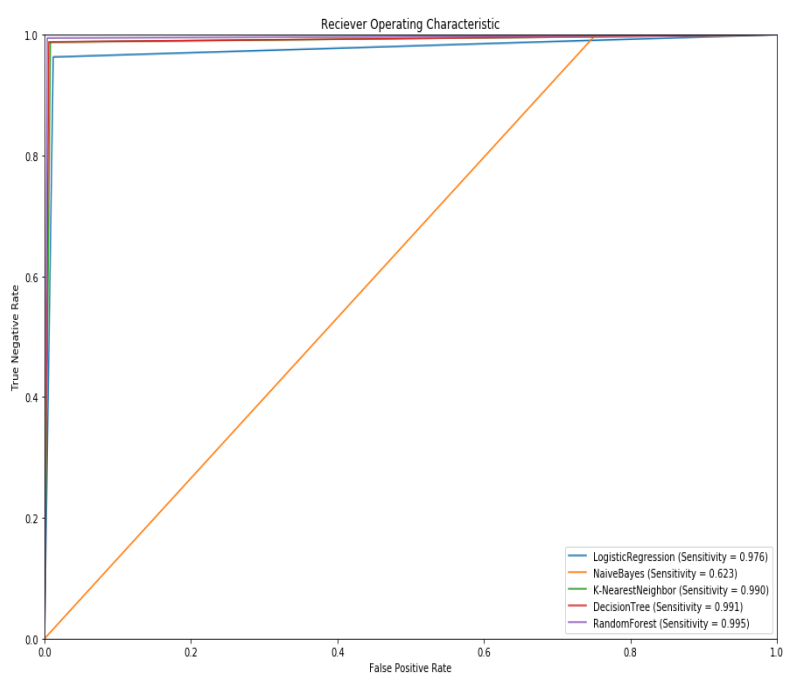
1. **NEURAL NETWORK**



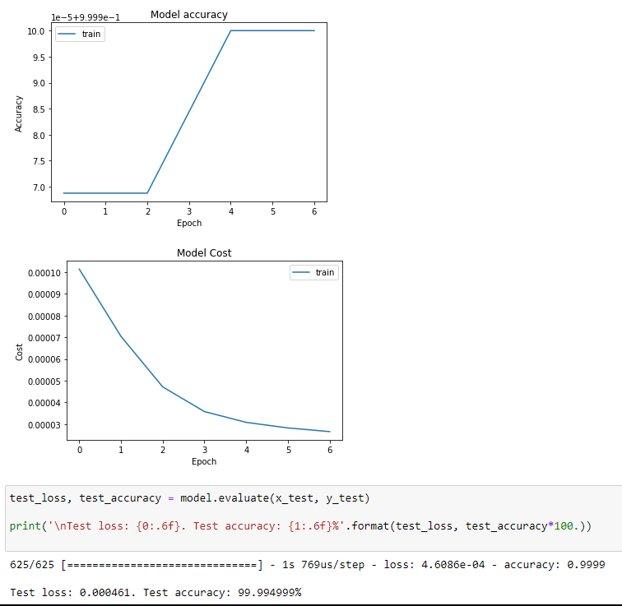
**5.6 Performance Analysis**

*Comparison of Models:* 

*Roc curve:*



*General model cost and accuracy:*



**5.7 Summary**

Our model has shown the different types of tools and ways in which a particular malware can be analyzed. Many kinds of research show that one single malware couldn’t be analyzed in a single tool. Experimental results show that every malware analysis tool has a different metric and way to analyze the malicious code. The possible future work in this domain can be developing an algorithm by which we can use all the software simultaneously and get better results and protect the systems more efficiently. The desired feature extraction and representation methods were selected and the selected machine learning algorithms were applied and evaluated. Collecting a malware dataset is a tedious task that requires a lot of time and effort. For a more accurate evaluation of the predictors, it is advised to test the models on all the possible types of malware: spyware, adware, rootkits, backdoor, banking malware, etc. In addition to that, it is important to understand that the model will only be able to predict the samples of the families that it has seen earlier.​

From the results above its clear that all models except Naïve Bayes are showing the best results on the acquired data. While Naïve Bayes shows poor result because of its probability function on a huge real-life dataset made by us by web scraping.

**CHAPTER 6 : PROJECT OUTCOME AND APPLICABILITY**

**6.1 OUTLINE :**

Remove distracting ads, making pages easier to read. Make web pages load faster. Keep advertisers from tracking you across websites. An ad blocker is a type of software, usually added by a user as an extension to an Internet browser, that prevents ads from appearing on the browsed web pages. When a user with an ad blocker visits a website with ads, the blocker identifies the ad content and prevents it from loading—as a result, the website does not receive ad revenue for that user.

**6.2 KEY IMPLEMENTATION OUTLINE OF THE SYSTEM :**

* Remove annoying ads to make your page easier to read. Speed ​​up website loading. Prevents advertisers from tracking you across your website. Ad blockers are a type of software that users add as an extension of their internet browser that typically prevents ads from appearing on the web pages they visit. When a user uses an ad blocker to visit a website that has ads, the blocker identifies the ad content and prevents it from loading. As a result, the website does not receive advertising revenue for this user.
* To protect our users, our model detects suspicious files and behavior and reports them immediately.
* It also does Automatic threat detection and malware eradication; this feature lags in most of the malware and adware protection software / applications.
* Our project has a Multi-layered protection against malware , which can detect the hidden malwares which are in text format.
* Our artificial intelligence (AI) system uses machine learning to automatically collect and extract data from the entire user base web page— then trains the model. The engines work across devices (both on the cloud and PCs), they use static and dynamic analysis techniques, and they are deployed in many of the layers of our defense engine.
* Malware Behavior Blocking analyzes program behavior to proactively protect against both known and unknown threats. Malware Behavior Blocking observes system events and blocks programs that exhibit malicious activity. Use this feature to ensure a higher level of protection against new, unknown, and emerging threats. After detecting malicious activity, Malware Behavior Blocking performs one of the following actions:

1. Block: Prevents programs exhibiting malicious behavior from making changes to the endpoint.
2. Terminate: Closes programs that exhibit malicious behavior.
3. Clean: Closes programs that exhibit malicious behavior. If a program is verified to be a threat, deletes files and other objects associated with the malicious program.

**6.3 SIGNIFICANT PROJECT OUTCOMES:**

We used three datasets: a training dataset, a test dataset, and a “scale-up” dataset. As stated above, our main goal is to achieve malware detection with only a few (if possible 0) false positives, therefore the clean files in this dataset (and also in the scale-up dataset) is much larger than the number of malware files. From the whole feature set that we created for malware detection, 308 binary features were selected for the experiments to be presented in this paper. Files that generate similar values for the chosen feature set were counted only once. Total number of unique combinations of the 308 selected binary features in the training, test and respectively scale-up datasets. Note that the number of clean combinations — i.e. combinations of feature values for the clean files — in the three datasets is much smaller than the number of malware unique combinations. Additionally, a native implementation is much faster than a browser extension implementation with the added benefit of having access to the unmodified image buffers.

Heuristic-based malware detection focuses on detecting intrusions by monitoring the activity of systems and classifying it as normal or anomalous. The classification is often based on machine learning algorithms that use heuristics or rules to detect misuse, rather than patterns or signatures. One of its shortcomings is that it tends to have a high false positive rate, such that many legitimate actions are classified as intrusive, and that it requires useful training data, which is typically difficult to obtain in large IT environments.

**6.4 PROJECT APPLICABILITY ON REAL WORLD APPLICATIONS :**

* In the last two decades, a variety of different ML techniques and feature selection algorithms have been widely applied to malware detection, predictions and blocking.
* This model (given that it gets accurate data and prediction after choosing the correct algorithm and proper use of feature engineering) can be used in blocking malicious advertisements and malwares. We propose a versatile framework in which one can employ different machine learning algorithms to successfully distinguish between malware files and clean files, while aiming to minimize the number of false positives. In this paper we present the ideas behind our framework by working firstly with cascade one-sided perceptron and secondly with cascade kernelized one-sided perceptron’s. After having been successfully tested on medium-size datasets of malware and clean files, the ideas behind this framework were submitted to a scaling-up process that enable us to work with very large datasets of malware and clean files.
* This model will help in reducing the human efforts which will help the user to take proper decisions and steps on time resulting in hassle-free access to websites. It will block ads that interrupt the browsing experience. Blocking annoyances like video ads, pop-ups, flashing banners and more means pages load faster.

**6.5 INFERENCE:**

As malware becomes more advanced, and with so much of our personal data now being stored online, the threat of malware stealing our private data to use for nefarious means has never been more real or more dangerous.

Prevention is the best strategy for keeping your personal computer free from malware. If your device does become infected though, it's not the end of the line. While manual malware removal is possible, it’s a complicated process even for savvy users. Still, there are ways to return your computer to a normal, functioning state.

**CHAPTER-7: CONCLUSIONS AND RECOMMENDATION**

**7.1 Outline**

• Have the right data. This is the fuel of machine learning. The data must be

representative, relevant to the current malware landscape and correctly labeled

when needed. We became experts in extracting and preparing data and training our

algorithms. We made an efficient collection with billions of file samples to empower machine learning.

• Understand theoretical machine learning and how to apply it to cybersecurity.

We understand how machine learning works in general and keep track of state-of-the-art approaches emerging in the field. On the other hand, we are also experts

in cybersecurity and we recognize the value each innovative theoretical approach

brings to cybersecurity practices.

• Understand user needs and be an expert at implementing machine learning

into products that help users with their practical needs. We make machine learning

work effectively and safely. We build innovative solutions that the cybersecurity

market needs.

• Build a sufficient user base. This introduces the power of ‘crowdsourcing’ to

detection quality and gives us the feedback we need to let us know if we are right or wrong.

**7.2 Limitation/Constraints of the System**

Our main target was to come up with a machine learning framework that generically detects as much malware samples as it can, with the tough constraint of having a zero false positive rate. We were very close to our goal, although we still have a non-zero false positive rate. In order that this framework to become part of a highly competitive commercial product, a number of deterministic exception mechanisms have to be added. In our opinion, malware detection via machine learning will not replace the standard detection methods used by anti-virus vendors, but will come as an addition to them. Any commercial anti-virus product is subject to certain speed and memory limitations, therefore the most reliable algorithms.

Since most antivirus products manage to have a detection rate of over 90%, it follows that an increase of the total detection rate of 3% − 4% as the one produced by our algorithms, is very significant. (Note that the training is performed on the malware samples that are not detected by standard detection methods.)

**7.3 Future Enhancements**

**1. Advanced Preprocessing**

Till now the preprocessing used is of basic ground level. Further advanced preprocessing techniques like Normalization, Encoding and Dimensionality Reduction (e.g. PCA) can be applied on the dataset.

**2. Use a wider dataset**

Currently we have used the data which is in csv format. Now will be using data which is in image format and in ASM & BYTES file format.

Also the dataset that was used in this study is broad, covering most of the malware types that are relevant to the modern world, it does not cover all possible types. Collecting a malware dataset is a tedious task that requires a lot of time and effort. For more accurate evaluation of the predictors, it is advised to test the models on all the possible types of malware: spyware, adware, rootkits, backdoor, banking malware, etc. In addition to that, it is important to understand that the model will only be able to predict the samples of the families that it has seen earlier. In other words, in a real-world application, the maximum amount of possible families should be used before the launch of the project for real-world environments.

**7.4 Inference**

* Large representative datasets are required

In machine learning, we must train our models on data that accurately replicates the conditions in which they will perform in practice. As a result, gathering a representative dataset is critical to the success of machine learning. It is critical to underline the approach's data-driven character.

* The trained model has to be interpretable

The majority of today's model families are referred to as black box models. Black box models execute sophisticated procedures that are difficult for humans to comprehend. The interpretability of a model influences how easy it will be to manage it, analyze its quality, and alter its functioning if a false alarm arises, for example.

* False positive rates must be extremely low

Our goal is to get a false positive rate as close to zero as feasible. False positives occur when an algorithm misidentifies a dangerous label as a harmless file. This is due to the fact that there are a lot of clean files in the world, and they keep coming. We take this into consideration and develop a model that allows us to correct false-positives on the fly.

Coming to Malware detection Methods: -

* Signature based detection technique: - This technique is mainly used for identifying known Malware, as it comes inside the Static analysis. This basically detect some patterns, strings or basically it looks to match signatures found in files with the databases of known Malware, it detects the structure of the Malware , but when some new Malware will come this will be unable to detect.
* Behavior based detection technique: - This technique basically detects the actions of an object before it can execute that behavior or it’s potential behavior is analyzed for suspicious activities. The one of the main disadvantage is it doesn’t show it’s full potential on detecting behaviors in a Virtual Machine or a Sandbox which may lead to some wrong analysis.
* Heuristic based detection technique: - This technique uses some ML algorithms to look for commands which may indicate any kind of malicious intent. As it searches for the commands and instructions in program it is helpful in discovering new viruses and maintaining their signatures.

**RELATED WORK INVESTIGATION**

User products that implement machine learning make decisions autonomously. The

quality of the machine learning model impacts the user system performance and its

state. Because of this, machine learning-based malware detection has specifics. Outside the malware detection domain, machine learning algorithms regularly work under the assumption of fixed data distribution, which means that it doesn’t change with time. When we have a training set that is large enough, we can train the model so that it will effectively reason any new sample in a test set. As time goes on, the model will continue working as expected. The related work investigation and comparison goes as follows:

* Large representative datasets are required:

It is important to emphasize the data-driven nature of this approach. A created model depends heavily on the data it has seen during the training phase to determine which features are statistically relevant for predicting the correct label.

* The trained model has to be interpretable:

Generalizing this, we must train our models on a data set that correctly represents the conditions where the model will be working in the real world. This makes the task of collecting a representative dataset crucial for machine learning to be successful. Most of the model families used currently, like deep neural networks, are called black box models. Black box models are given the input X, and they will produce Y through a complex sequence of operations that can hardly be interpreted by a human. This could pose a problem in real-life applications. For example, when a false alarm occurs, and we want to understand why it happened, we ask whether it was a problem with a training set or the model itself.

* False positive rates must be extremely low:

The interpretability of a model determines how easy it will be for us to manage it, assess its quality and correct its operation. False positives happen when an algorithm mistakes a malicious label for a benign file. Our aim is to make the false positive rate as low as possible, or zero. This is not typical for a machine learning application. This is important, because even one false positive in a million benign files can create serious consequences for users. This is complicated by the fact that there are lots of clean files in the world, and they keep appearing.

* Algorithms must allow us to quickly adapt them to malware writers’ counteractions:

To address this problem, it is important to impose high requirements for both machine learning models and metrics that will be optimized during training, with the clear focus on low false positive rate (FPR) models. This is still not enough, because new benign files that go unseen earlier may occasionally be falsely detected. We take this into account and implement a flexible design of a model that allows us to fix false-positives on the fly, without completely retraining the model.

**REFERENCES**

1. A Survey on Automated Dynamic Malware Analysis Techniques and Tools, https://www.seclab.tuwien.ac.at/papers/malware\_survey.pdf
2. Malware Analysis and Detection in Enterprise Systems by Tebogo Mokoena and Tranos Zuva, published in 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)

<https://ieeexplore.ieee.org/document/8367429>

1. <https://www.blackhat.com/docs/us-17/thursday/us-17-Anderson-Bot-Vs-Bot-Evading-Machine-Learning-Malware-Detection-wp.pdf>
2. [(PDF) An Emerging Malware Analysis Techniques and Tools: A Comparative Analysis (researchgate.net)](https://www.researchgate.net/publication/350886133_An_Emerging_Malware_Analysis_Techniques_and_Tools_A_Comparative_Analysis)

# An Emerging Malware Analysis Techniques and Tools: A Comparative Analysis by Arkajit Datta, Kakelli Anil Kumar and Aju D of VIT University.