T.C.

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

DETECTION AND CLASSIFICATION MALICIOUS URLS ON SOCIAL MEDIA USING MACHINE LEARNING

Ahmed SALEEM

Master of Science

Supervisor

Asst. Prof. Abdullahi Abdu IBRAHIM

Istanbul, 2021

DETECTION AND CLASSIFICATION MALICIOUS URLS ON SOCIAL MEDIA USING MACHINE LEARNING

by

Ahmed Idan Halyout Saleem

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Master of Science

ALTINBAŞ UNIVERSITY

2021

The thesis titled "DETECTION AND CLASSIFICATION MALICIOUS URLS ON SOCIAL MEDIA USING MACHINE LEARNING" prepared and presented by "AHMED SALEEM" was accepted as a Master of Science Thesis in Electrical and Computer Engineering.

____

Asst. Prof. Abdullahi IBRAHIM

Supervisor

Thesis Defense Jury Members:

Asst. Prof. Abdullahi IBRAHIM	School Of Engineering and Natural Sciences, Altinbas University	____________________
Asst. Prof. Muhammad ILYAS	School Of Engineering and Natural Sciences, Altinbas University	____________________
Asst. Prof. Tareq Abed MOHAMMED	Faculty of Engineering, Imam Sadiq University	____________________

I certify that this thesis satisfies all the requirements as a thesis for the degree of ....

Approval Date of Institute of Graduate Studies: ____/____/____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Ahmed Idan Halyout Saleem

Signature

DEDICATION

I would like to thank Allah Almighty for the power of mind, health, strength, guidance, knowledge and skills to complete this study.

This thesis is wholeheartedly dedicated to my parents. There are no words to describe what you mean to me; there is nothing that I can repay for what you have done to me. I will continue to do my best to achieve your expectations.

Lastly, I dedicated this to my brother and sisters, relatives, and friends who have been encouraging me during this study.

ACKNOWLEDGMENTS

I would like to thank my supervisor professor Abdullahi Abdu IBRAHIM for all support during my study. It’s great pleasure to express my deepest gratitude to my friends who have shared with me best moments during my study for the Master degree.

ABSTRACT

DETECTION AND CLASSIFICATION MALICIOUS URLS ON SOCIAL MEDIA USING MACHINE LEARNING

SALEEM, Ahmed

M.Sc., Electrical and Computer Engineering, Altınbaş University

Supervisor: Asst. Prof. Dr. Abdullahi Abdu IBRAHIM

Date: 4/6/2021

Pages: 64

Recently, the variety and size of malware on the social networks has increased dramatically, bearing phrases and headings aimed at attracting attention and pushing them to enter the link that contains malicious software, causing theft (bank accounts, financial transactions, installing malicious software) and therefore it is necessary to discover These risks and threats are addressed. The purpose of this thesis is to discover malware and classify it into benign or malicious URLs using a machine learning algorithm called Support vector machine, which is used in binary classification and the algorithm was utilized for creating a model for malware detection. This study is conducting comprehensive experiments for the purpose of comparing and verifying the suggested method's results with the ones of other techniques. The experimental results are showing that the presented approach is achieving strong detection and high accuracy of up to 93%, and it is a method that achieves strong detection compared to other results for detecting malware.

Keywords: Malicious, Cyber Security, Machine Learning, Social Media

ÖZET

MAKİNE ÖĞRENMEYİ KULLANARAK SOSYAL MEDYADAKİ KÖTÜ AMAÇLI URL'LERİ TESPİT VE SINIFLAMA

SALEEM, Ahmed

Yüksek Lisans, Elektrik ve Bilgisayar Mühendisliği, Altınbaş Üniversitesi

Danışman:: Asst. Prof. Dr. Abdullahi Abdu IBRAHIM

Tarih: 4/6/2021

Sayfalar: 64

Son zamanlarda, sosyal ağlardaki kötü amaçlı yazılımların çeşitliliği ve boyutu önemli ölçüde artmış, dikkat çekmeyi amaçlayan ifadeler ve başlıklar taşıyarak onları kötü amaçlı yazılım içeren bağlantıya girmeye zorlayarak hırsızlığa (banka hesapları, finansal işlemler, kötü amaçlı yazılım yükleme) ve bu nedenle bu risklerin ve tehditlerin ele alındığını keşfetmek gerekir. Bu tezin amacı, kötü amaçlı yazılımları keşfetmek ve onu ikili sınıflandırmada kullanılan Destek vektör makinesi adı verilen ve kötü amaçlı yazılım tespiti için bir model oluşturmak için kullanılan bir makine öğrenme algoritması kullanarak iyi huylu veya kötü niyetli URL'ler olarak sınıflandırmaktır. Bu çalışma, önerilen yöntemin sonuçlarını diğer tekniklerle karşılaştırmak ve doğrulamak amacıyla kapsamlı deneyler yapmaktadır. Deneysel sonuçlar, sunulan yaklaşımın güçlü tespit ve% 93'e varan yüksek doğruluk sağladığını ve kötü amaçlı yazılımları tespit etmek için diğer sonuçlara kıyasla güçlü tespit sağlayan bir yöntem olduğunu göstermektedir.

Anahtar Kelimeler: Kötü Amaçlı, Siber Güvenlik, Makine öğrenimi, Sosyal Medya

TABLE OF CONTENTS

Pages

ABSTRACT .... **vii**

ÖZET .... **viii**

LIST OF TABLES .... **xii**

LIST OF FIGURES .... **xiii**

LIST OF CHARTS .... **xiv**

LIST OF ABBREVIATIONS .... **xv**

1. INTRODUCTION **1**

1.1 BACKGROUND 1

1.2 PROBLEM STATEMENT 2

1.3 MOTIVATION 2

1.4 CONTRIBUTION 3

1.5 THESIS OUTLINE 3

2. LITRITAURE REVIEW **4**

3. MATERIALS AND METHODS **6**

3.1 MALWARE 6

3.2 TYPES OF MALWARE 7

3.3 MALWARE DETECTION 11

3.4 ONLINE SOCIAL NETWORK AND MALICOUS 12

3.5 MALICIOUS SPREED IN SOCIAL MEDIA 14

3.6 MALWARE DISTRIBUTION NETWORK 16

3.7 UNIFORM RESOURCE LOCATOR 18

4. PROPOSED METHOD 19

4.1 MACHINE LEARNING 19

4.1.1 SUPERVISED LEARNING 19

4.1.2 UNSUPERVISEDLEARNING 21

4.1.3 SEMI-SUPERVISEDLEARNING 21

4.1.4 REINFORCEMENTLEARNING 21

4.2 SUPPORT VECTOR MACHINE 22

4.3 MULTINOMIAL NAÏVE BAYES 24

4.4 LOGISTIC REGRESSION 25

4.5 FEATURE EXTRACTION 26

4.6 DATA COLLECTION 28

4.7 TOKENIZATION 29

4.8 VECTORIZATION 30

4.9 EXPERIMENTAL ENVIROMENT 32

4.10 EVALUTION METRICS 33

5. SIMULATION AND RESULTS 37

5.2 CLASSIFIERS 37

5.3 DESCRIBING DATA 37

5.4 RESULTS 39

5.5 CLASSIFIER COMPARISION 42

CONCLUSION **43**

REFERENCES.. **44**

APPENDIX.. **50**

LIST OF TABLE

Pages

Table 4.1: Experimental tools 48

Table 4.2: Advanced metrics of evaluations 50

Table 5.1: Proportions of testing and training Patterns 53

Table 5.2: Results of support vector machine 54

LIST OF FIGURES

Pages

Figure 1.1: Increase Number of people using social media platforms 1

Figure 1.2: Comparison of malware activity to other cyber threats 2

Figure 3.1: Diagram showing Keystroke loggers 8

Figure 3.2: The diagram shows the malicious URL and controlling it by the attacker 9

Figure 3.3: Malware infection chain 10

Figure 3.4: Botnet Architecture 11

Figure 3.5: Top 20 most activities malware program involved in URLs attacks on clients' PCs. 14

Figure 3.6: Example of malicious URLs spread in social media... 16

Figure 3.7: Example of url structure 18

Figure 4.1: Diagram of the Supervised Learning Model 20

Figure 4.2: SVM classification 23

Figure 4.3: Diagram show Feature Extraction & Vectorization 28

Figure 4.4: The database used for the experiment 29

Figure 4.5: Structure of confusion matrix.. 34

Figure 5.1: : Confusion matrix for support vector machine.. 40

Figure 5.2: characteristics (ROC) curve 41

LIST OF CHARTS

Pages

Chart 5.1: Dataset …………………………………………………………………..38

LIST OF ABBREVIATIONS

AI	:	Artificial Intelligence
ML	:	Machine Learning
PC	:	Personal Computer
SVM	:	Support Vector Machine
HTTP	:	Hypertext Transfer Protocol
DDOS	:	Distributed Denial-of-Service
IDS	:	Intrusion Detection Systems
OSN	:	Online Social Network
IT	:	Information Technology
URL	:	Uniform Resource Locator
WWW	:	World Wide Web
TCP	:	Transmission Control Protocol
UDP	:	User Datagram Protocol
HTTPS	:	Hypertext Transfer Protocol Secure
TF-IDF	:	Term Frequency–Inverse Document Frequency
ROC	:	Receiver Operating Characteristic
AUC	:	Area Under the ROC Curve
XSS	:	Cross-Site Scripting

1. INTRODUCTION

1.1 BACKGROUND

The number of social media platforms users has increased Significantly. There are users who attack and abuse the social networking platforms by posting malicious links that allow the attackers partial or complete control over the system. As a result of these cyber crimes theft of (bank accounts, DDOS, and malwares installation), as well as many attacks, therefore, methods must be used to detect malicious links and analyze the content of the sites To protect users from malicious intrusions[1].

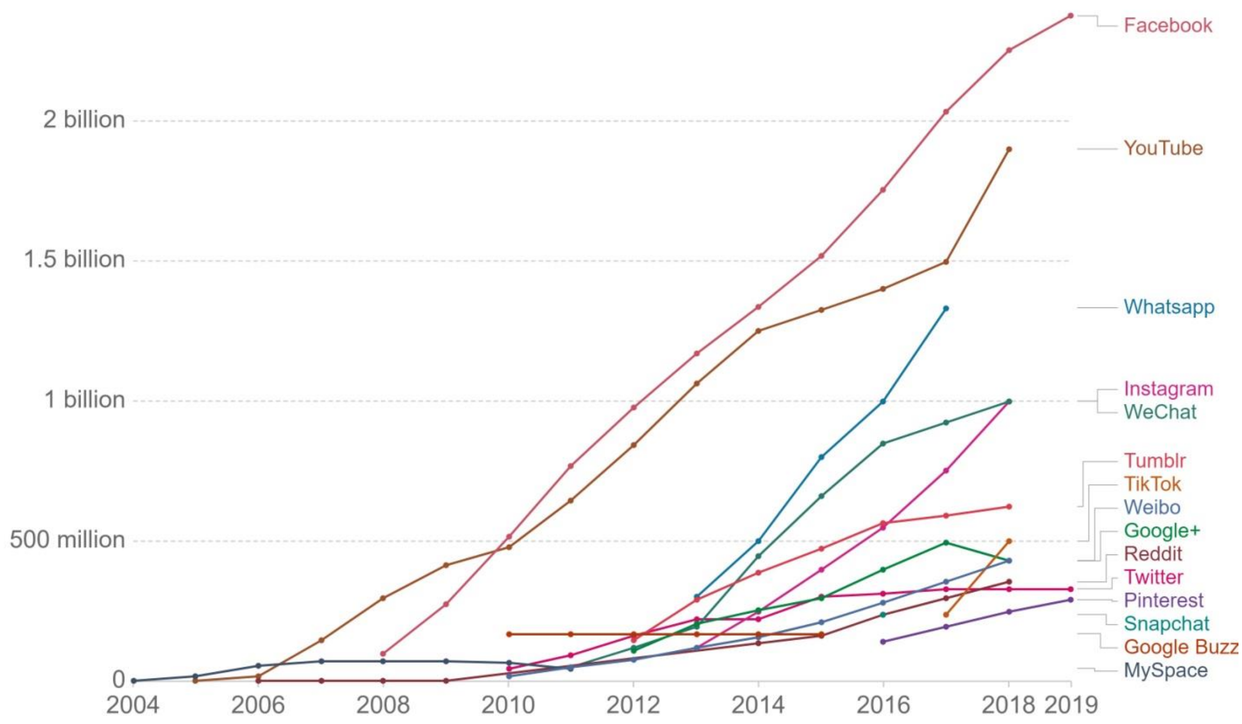

Figure 1.1: Increase Number of people using social media platforms

source : statista and TNW(2019)

1.2 PROBLEM STATEMENT

Malicious software is software that has great damage to systems, Nowadays, malware programmers are working to development on many complex techniques to encode and avoid detection of this software . This is what makes analyzing this software one of the most difficult problems and Challenges [2]. With the increase in services and data provided by social networking platforms, programs have appeared that work, in some way, to exploit vulnerabilities in social media networks, and allow attackers to steal personal data, bank accounts And other security threats. This study aims to uncover methods of analyzing and classifying malware and detecting malicious URLs to protect users on social networking sites**.**

1.3 MOTIVATION

According to, resulting from accidents program activities malicious were fixed (Figure1.2). Studies have shown that more than 34% of malicious programs have no description and that most of them use names that are similar to services and applications on a computer system[2] . Knowing the malicious content on social media and its relationship to computer crimes to avoid the users of this software Malicious and controlling their computers; A machine learning model will be built to provide answers to malicious links to reduce risks to users and to provide safe and secure services.

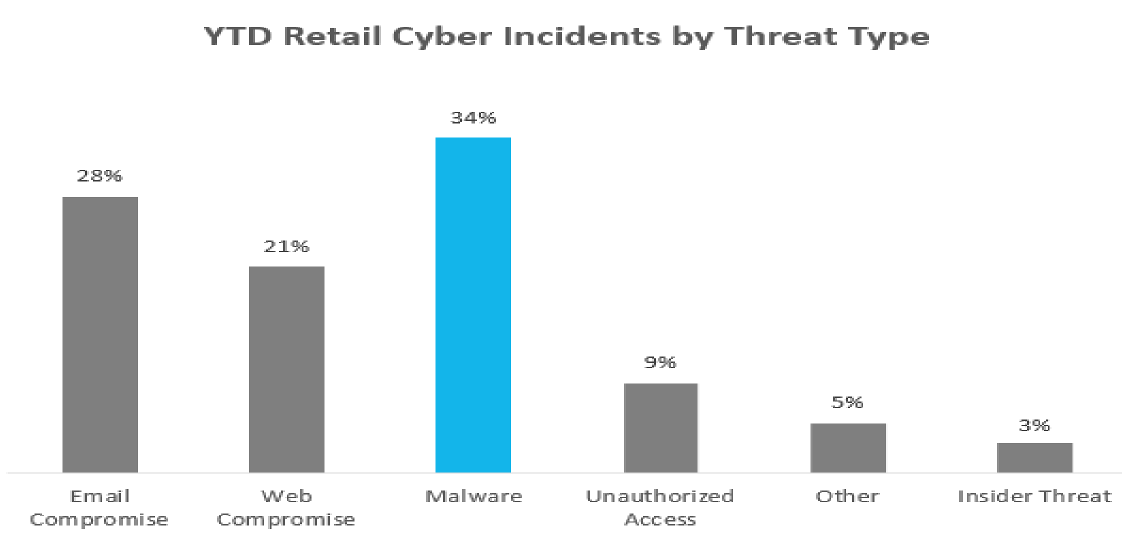

Figure 1.2: Comparison of malware activity to other cyber threats

1.4 CONTRIBUTION

The field of machine learning gives the program experience improving a tool. In this thesis, we take advantage of a number of malware samples based on a Dataset. In this study, we propose a method for detecting malicious URLs by using machine learning technology for the support vector machine algorithm to detect and Malicious URL activity on social media platforms and classify data into benign or Malicious, to reduce risks and threats to social media users.

1.5 THESIS OUTLINE

This thesis is presented in six chapters: Chapter 2 is where we review some of the previous work and implementation on malware detections. Chapter 3 is where we give a background about all the components. Chapter 4 is where we explain our model in details and implementation of that model in details, Chapter 5 is where we simulate our model and record the results obtained. Chapter 6 is where we conclude our work and put a future scope into perspective.

2. LITRITAURE REVIEW

The purpose of this chapter is to explain some of the previous studies that have been conducted in the classification and detection of malware links, the goal is to know the behavior of malicious links and to determine the risks resulting from these attacks.

[Ma et al] Adopted an approach based on automatic URL address for classifications, use statistical methods to discover lexical features to detect malwares sites by predicting and automatically analyzing thousands of features that can indicate harmful addresses, obtained high accuracy in the resulting workbooks and obtained results with high accuracy ,with some positive content false [1].

[Blum et al.] Phishing is a type of cybercrime, for example, making a fake page for the social networking site with a link for the purpose of obtaining the user's password and is sent through conversation or by sending a message to an e-mail to deceive users, and used Lexical feature groups to make blacklist and white list method To detect and classify the link benign or not by using Online-Learning[2].

[Zhang, W.] , This study included detecting threats to websites by using machine learning to classify malicious links and using the features of URLs information to determine if the length of the link URL suspicious or benign content, Used a method to classify malicious links by extracting the URL feature by using on-line learning algorithm [3].

[Chiba et al.] , A developed model for machine learning to detect malicious links, IP address features are used based on blacklists, use Support vector machine algorithm and the results show an accuracy of 83% in detecting malicious links that are not covered by other traditional methods[4].

[Zhao and Hoi], In this work he presented, Detecting malicious URLs using the supervised (CSOAL) algorithm and relying on a lexical feature such as hostname and whois info predicatively identifying malicious or benign URL link content and has proven good efficiency.[5]

[Kandasamy and KorothThe] classification was used in machine learning techniques, the natural language processing algorithm, to classify the random user from the legitimate user in the social networking site Twitter by analyzing the URLS and based on six features of Twitter, and the results of the support vector machine algorithm were 87% and naive bayes algorithm 90% and natural language processing algorithm 97% [6].

[Feroz and Mengel] , The approach that was used to detect malicious URLs is Clustering K-Means algorithm to classified links using machine learning techniques depending on the lexical and host features and achieved results for detecting harmful content links with an accuracy of 93%, with a false positive content rating[7].

[Olalere et al.], Attackers use the Internet to spread malicious content online, and the challenges we face are great. We use cell phones to browse websites by typing the website link in the browser or by clicking on the links, which makes attackers use the links to spread harmful content and penetrate their victim. Accordingly, the researchers used classification and lexical features such as (hostname , URL to path , path)for analyzing the malicious content, the results obtained were a negative error rate (FNR) 0.018, a false positives rate (FPR) 0.046 with high accuracy using the Support vector machine algorithm[8].

[Weedon et al.] , Provided the development of a machine learning model for the Random Forest algorithm to detect malicious content in URL links and classify it into benign or suspicious using the lexical features, the results in this model obtained an accuracy of 86.9%[9].

[Liu et al.] , Proposed a way to discover malicious content in website URLs by combining machine learning and statistical analysis to classify harmful URLs by using the character features of URLs TF-IDF addresses, and achieved the results of the algorithm random forest with high accuracy[10].

[Khan et al.], Depend on the AdaBoost algorithm in building the machine learning model and use the URL feature to classify links into good or bad, the model achieved high results to discover the content of suspicious and valid links[11].

3. MATERIALS AND METHODS

In this chapter, Detection malicious content on social media based on URL-Based Feature and Support vector machine classifier is presented a brief introduction to the components and some of the fundamentals that we adopted to implement our model.

3.1 MALWARE

Malicious programming is an overall term for every malicious program, which with regards to network safety implies: programming that is used for the purpose of penetrating operating systems or to collect sensitive information, attempting to penetrate the security policy of the system in relation to confidentiality, integrity or availability Attackers try to gain unauthorized access to personal information for clients or corporate employees, and this is called confidentiality. Malware has a major impact on integrity because attackers can manipulate and delete content and steal information stored in operating systems. Damage from malware infiltration of security policy for users and organizations Malicious software is used for bad purposes, attackers use many methods to penetrate systems or block service from companies, and limited harm to these actions is caused by stealing specific information from the user's computer, and significant damage may result in disrupting an entire organization and publishing user data, e-mail and passwords. In recent times, malware has been widely misused to achieve political gains and damage the infrastructure, beginning with malware for systems on computer operating systems for a virus called (Brain.A). This malicious program was developed by the brothers Amjad and Basit, they created a virus that spreads through Floppy disk, prove that the computer operating system is not safe[13]. Malware takes different forms that are difficult for users to detect and computer virus may be the most malicious software, malware on Windows operating systems can create copies of itself because it has a feature to spread within the network or automatically spread malicious links on social networking sites, the days of deceivers are gone and Amateurs who develop malicious software, these days, software is being developed by professional criminals for the purpose of achieving their illegal ends. Cyber criminals use advanced methods and techniques that are difficult to detect, which allows them to steal bank accounts online or create fraudulent bills on your bank card to steal the account and they sell those Confidential information on the black market[14].

3.2 TYPES OF MALWARE

There are many forms of malware, and all of them can damage operating systems, Some of the most threatening categories are described below.

Clickjacking malware**:** Clickjacking can be defined as an exploit where multiple opaque or transparent layers have been added for tricking the users to click on a link or button on another page whereas the users aiming to click the presently-displayed page; this, the attacker “hijacks” clicks as well as routing them to another pages.

Extension-based ON malicious: In this types, an extension is installed automatically into the web browser without the knowledge of users for intercepting the social network traffic of users with regard to malicious purposes, for instance, to highjack the credentials of the users.

Magnet malware: Conventional trojans are sending malicious messages/links to friends who are connected (directly) to infected user (ones hop neighbors in networks). In addition, the magnet has the ability for sending malicious messages/links not just to the infect users friend's, yet also to friends (twice hop neighbor). Dramatically, such approach is speeding up the malware’s propagation process. There were no researches on Magnet, extension based OSN malware or Clickjacking malware.

Backdoors: It is malicious code that allows access to, penetration and control of operating systems from another place on the Internet through control commands. Backdoors allow Internet criminals to take passwords or personal information. But there is a peaceful use of the backdoor by software makers to access their own technology to solve software problems[15].

Keystroke loggers: It is a malicious program used to spy on the keyboard and be hidden, and the keyboard monitor is considered illegal because it is associated with cyber criminals to penetrate users and obtain sensitive data such as email passwords, personal accounts, and credit cards. Hackers send keyloggers by email or bulletin on websites Download programs or be within free programs and you do not know[16].

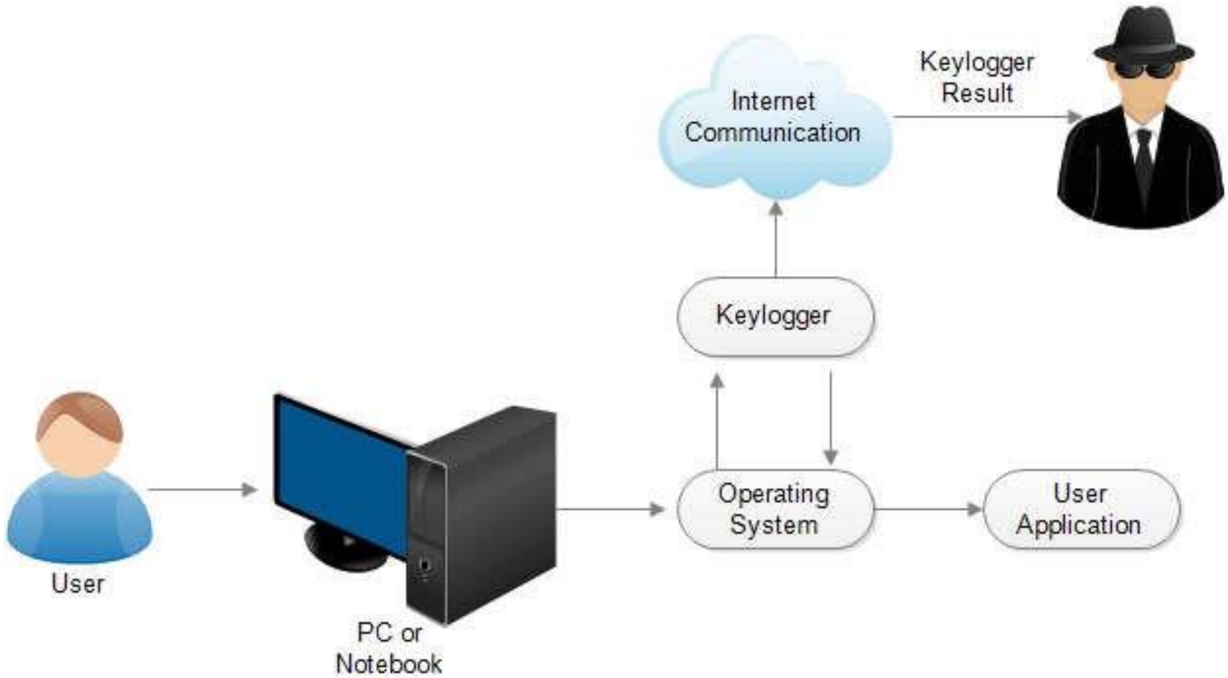

Figure 3.1: Diagram showing Keystroke loggers work [Rahim et al.]

Spam: It is sending unwanted messages to users via e-mail, social networks, or other digital communication tools. Internet users are exposed on a daily basis to piracy operations using unwanted messages, depending on phishing, fraud and social engineering. An example of this is that (PayPal) users are exposed to the theft of their financial accounts. Through these attacks.

Trojan horse: It is a kind of malware that hiding itself like legitimate programming, used by attackers for gain access to systems and users. Users are often hacked with a Trojan horse using social engineering to convince the user that the program performs another function, the Trojan horse works to spy on users and steal sensitive information and it may include copying and modifying data with the possibility of deleting that data or disrupting the network. Trojans cannot reproduce using self-replication

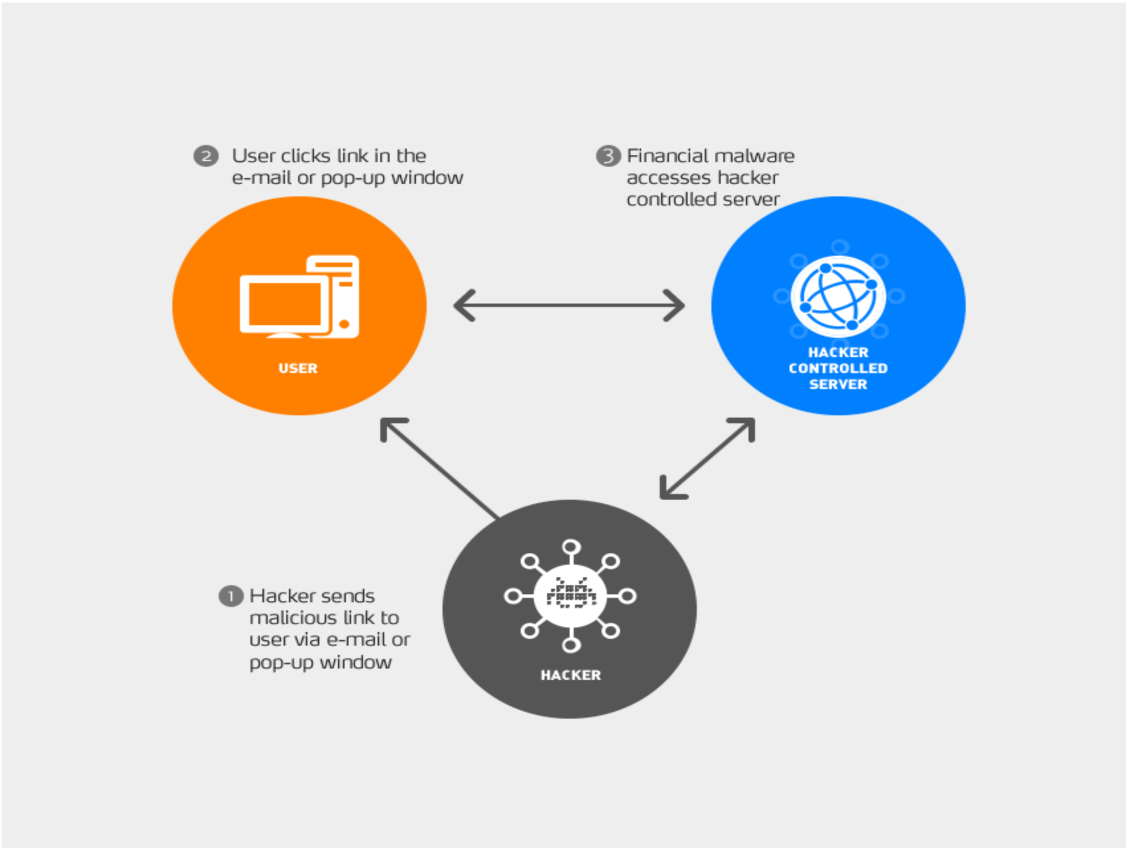

Figure 3.2: The diagram shows the malicious URL and controlling it by the attacker

Remote Access Trojan (RAT): It is a malicious program that is used in the process of spying and monitoring activity and reaches remote computers infected and is considered one of the most dangerous types of Trojans, which makes users vulnerable to violating privacy and stealing their data. Spyware is used by cybercriminals and they have complete control over the operating system, which allows it to record what is written by the keyboard, operate a webcam, and control files and contents[17].

Scareware: It is a kind of malware that deception clients to visit a noxious site, often in the form of a op-up window titled warnings of legitimate anti-virus companies to check your device, and when users download the program, they end up carrying a malicious program.

Worms: It is considered one of the sub-types of viruses, it is characterized by the ability to self-copy the same from one computer to another with the possibility of spread within the network, which leads to infecting the rest of the networks and systems, and thus great damage occurs.

Macro virus: It adds its code macros to documents, data files and spreadsheets by relying on macros, and due to the severity of this command, the macros were disabled by default in Office 2000 by Microsoft, Until now, it is still disrupted by what cybercriminals had to use social engineering to persuade users to enable micro-units. Before the attackers send macro viruses

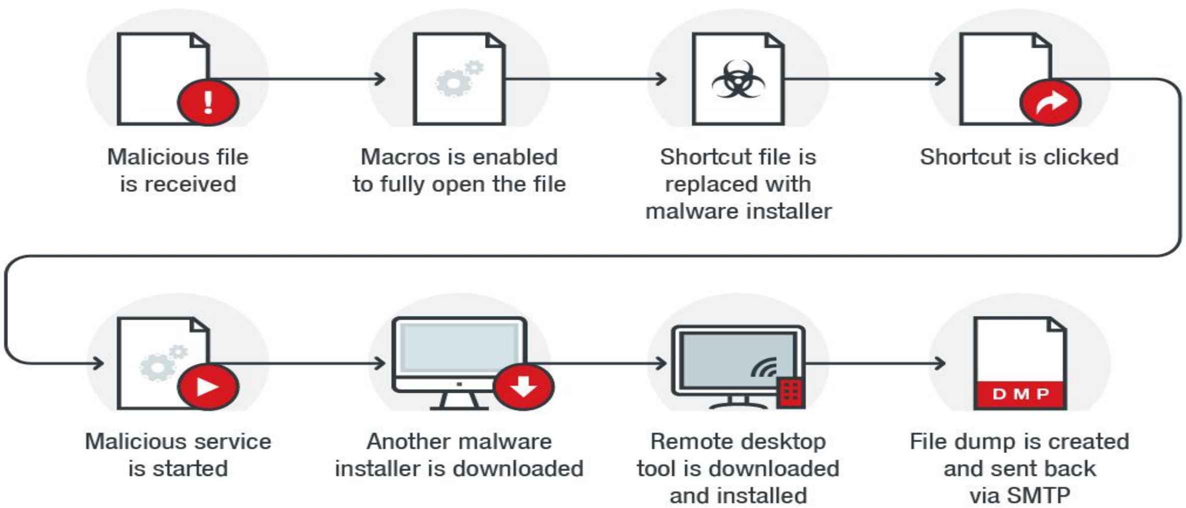

Figure 3.3: Malware infection chain

Social Engineering: Social engineering is the technique of manipulation and psychological deception used by cybercriminals to exploit victims and obtain sensitive information.

Phishing: It is considered one of the types of cybercrimes in which the attacker uses social engineering and pretends to be a reliable source for defrauding the victims by providing sensitive data such as passwords and bank accounts. For example, Kaspersky company explained in its published article that cyber criminals use a method to penetrate Facebook accounts by convincing them that companies have won a cash prize of $ 100 million and you can apply for this grant for companies affected by the Corona virus pandemic The post on CNBC includes a link that allows companies to apply for this grant when filling out the form and sending data. It displays a confirmation message from the site stating that your request has been accepted and we will contact you as soon as possible. This was one of the phishing messages that received great media attention by news agencies and communication networks Social[18] .

Botnet: A group of computers that are hacked by software download sites or social networks, and every victim is called a bot, and the attackers who control the bot are called Botmaster. Internet criminals use the botnet to gain access to illegal people, such as stealing user data and performing denial of service attacks.

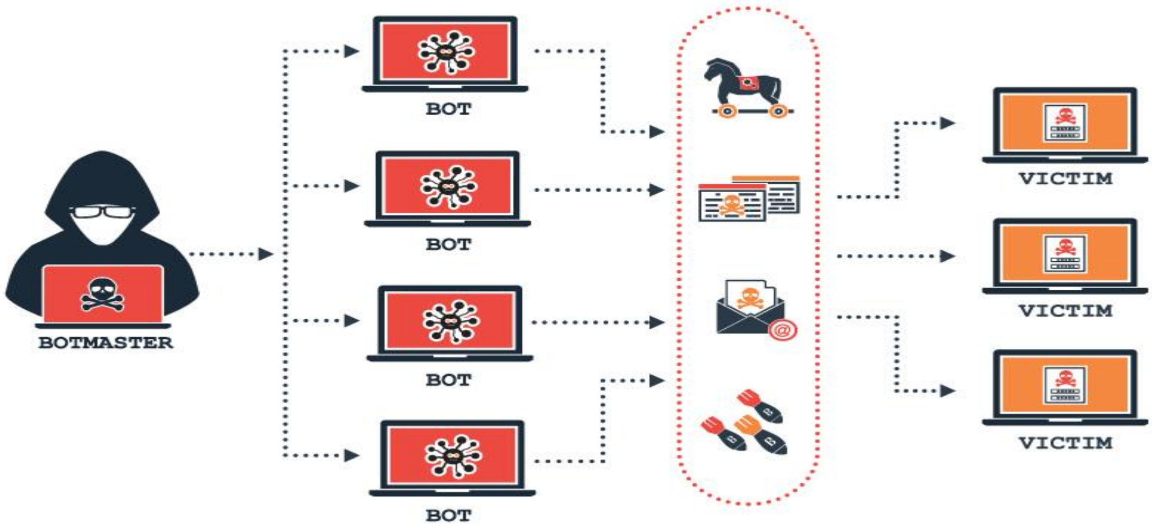

Figure 3.4: Botnet Architecture

Source : The International Council of Electronic Commerce Consultants

3.3 MALWARE DETECTION

As the introduction described, the malware detector can be defined as the implementation regarding a few techniques of malware detection. Also, malware detectors try to detect the malicious behavior; thus, protecting the system. Malware detectors might or might not be residing on the same system that their are trying to protect. Malware detectors are performing their protection via manifested techniques of malware detection, and serving as empirical means of assessing the malware detection methods' detection abilities. The malware detector is taking 2 inputs, one of them is its knowledge regarding malicious behavior. With regard to anomaly based detection, such knowledge's inverse is coming from learning phase. Theoretically, the anomaly based detection recognizes what is an anomalous behavior on the basis of its knowledge regarding what is normal. Due to the fact that the anomalous behavior subsumes a malicious behavior, a few senses of maliciousness might be captured through the anomaly based detection.

In the case when malware detectors are using signature based approach, its knowledge regarding what is malicious is coming from its repository, that was typically maintained/updated via individuals with the ability for identifying malicious behavior as well as expressing it in a form that is considered to be amenable for signature repository, and lastly for machines to read [22]. However, the other input which must be taken via the malware detector is the under-inspection program. As soon as the malware detectors have the knowledge regarding what was specified as normal or malicious behavior and the program under-inspection, it might use its detection approach for deciding if the program was benign or malicious. Even though malware detectors and Intrusion Detection Systems (IDSs) were often synonymously used, the malware detector was typically just one of the complete IDS components**.**

3.4 ONLINE SOCIAL NETWORK AND MALICOUS

In computers, there are 4 groups of security threats: denial of service attacks, spams, malwares and software bugs [22]. Malwares are strongly associated to the other 3 security threat types. Spam might be sent through malwares, whereas malwares might be propagating via spams. Malwares are exploiting the software bugs (vulnerabilities) for the purpose of attacking the computer systems, and might be utilized for mounting DoS attacks. ESET (IT security company), in 2014, specified that approximately 25% of cyber-attacks are resulting from malwares [23]. For extortion and in cyber warfare, malwares were utilized as weapons. For example, certain malware (Stuxnet) targets the software systems of Siemens in Iran nuclear facilities for the purpose of disrupting their services [24], whereas ransomware was utilized for extorting money from organizations and individuals. Also, Symantec documented an increase of approximately 35% in Ransomware attacks in the year 2015 in comparison to the 2014 [25]. In addition, online social networks (OSNs) like Twitter, Facebook and MySpace offered hundreds of millions of individuals globally with ways for connecting and communicating with their colleagues, families and friends who are distributed geographically all over the world. OSNs were of the major significant services provided via WWW. For example, Facebook is considered as the 2nd most-visited website worldwide based on current ranking through Alexa [26], just following Google. Recently, a study conducted via com Score (analytics and media measurements company) - social networking is responsible for one in each five mins spent online

[27]. In addition, the widespread usage and popularity of OSN are attracting cyber criminals and hackers who have utilized OSS as one of the platforms for spreading malwares. Also, an effective attack with the use of malwares on OSNs might result in infecting or compromising tens of millions of computers and OSN accounts. Furthermore, cyber criminals might be mounting a lot of DoS attacks against the internet systems or infrastructures by means of compromised computers and accounts, they might be stealing sensitive information of users for fraudulent activities. Also, the compromised accounts of OSN might be utilized for spreading mis-information to bias the public opinions [28], or influencing automatic trading algorithms which are relying on the public opinions[29]. The algorithms of automatic trading are placing sell/buy stock orders on behalf of human investors). In malwares, there are 2 main types targeting the users of OSN: Trojans and cross site scripting worms.

Cross-site scripting (XSS) worms: The XSS worms can take the advantages of the weak security in the applications where these worms are able to attack and propagate without getting an authorization. Usually, the trojan are installed on the system by cheating the user to open a special social engineering. Furthermore, the typical method to trick the users of OSN is by using a trojan that can download and execute malwares from malicious websites via pretending to be unharmful application such as Adobe Flash Player.

There are different types of trojans that can attack online social network. For example, Clickjacking worms [30] and extension based malware [31]. The enterprise and the individual users are both can be attacked by trojans. For this reason, attacking the social network by a malware could cause violating the privacy of the user and stealing the data as well as making sever operational or financial harm to government agency.

The installed trojan on the victim’s operating system can access the data of the computer such as bank accounts and the password of the email. Moreover, the malware can make multiple copies of itself and transfer it to the other computers on the network which cause the same damage to all other computers. A malware also can make a botnet which allows a hacker to send more malwares for example Ransomware. This kind of malware that detected in 2016 are called Locky ransomware which was transferred to the victim by SVG or JPEG files using the Messenger of the Facebook [32].

Rank	Name*	Number of attacks	% of all attacks**
1	Malicious URL	1 393 829 795	87.36%
2	Trojan.Script.Iframer	58 279 262	3.65%
3	Trojan.Script.Generic	38 948 140	2.44%
4	Trojan.Win32.Generic	5 670 627	0.36%
5	Trojan-Downloader.Script.Generic	4 695 210	0.29%
6	Exploit.Script.Blocker	4 557 284	0.29%
7	Trojan.JS.Popupper.aw	3 355 605	0.21%
8	Exploit.Script.Generic	2 943 410	0.18%
9	Trojan-Downloader.SWF.Voleydaytor.h	2 573 072	0.16%
10	AdWare.Win32.IBryte.x	1 623 246	0.10%
11	Trojan-Downloader.Win32.Generic	1 611 565	0.10%
12	AdWare.Win32.ScreenSaver.e	1 381 242	0.09%
13	Trojan-Downloader.JS.Iframe.cxk	1 376 898	0.09%
14	Trojan-Downloader.JS.Iframe.cyq	1 079 163	0.07%
15	Trojan-Downloader.JS.Expack.sn	1 071 626	0.07%
16	AdWare.Win32.ScreenSaver.i	1 069 954	0.07%
17	Trojan-Downloader.JS.JScript.ag	1 044 147	0.07%
18	Trojan-Downloader.JS.Agent.gmf	1 040 738	0.07%
19	Trojan-Downloader.JS.Agent.gqu	983 899	0.06%

Figure 3.5: Top 20 most activities malware program involved in URLs attacks on clients' PCs.

(Secure list, 2012).

3.5 MALICIOUS SPREAD IN SOCIAL MEDIA

Trojans that attack OSN users then propagate in the online social networks could be generic extension based malware, Koobface [33], or Magnet [34]. The process of describing the real-world trojans can be divided in three stages as following:

First, the trojan developer makes many profiles for unreal persons in the online social network. Then, adding new friends for these fake accounts with many real profiles in the online social network. This technique is called infiltration which approved to be a successful method to spread malicious content in the online social network for example Facebook [35].

Second, the trojan creator makes web links and post it on fake accounts. Then, tricking the friends to click on these links which lead to visiting a web page with malicious content. When one of the friends click on that link and visiting the web page, the trojan will be downloaded and executed on the operating system without giving a permission for this process. This kind of attacking a victim is known as drive-by download. This process only can be done if the user's operating system or the browsers are not updated as well as there are weakness in the security [36].

If the weakness in the security is existed on the victim's computer, then the trojan developer will depend on social engineering techniques to cheat the victim to execute the trojan. After clicking on the link, a malicious web page is opened then the victim will be asked to download a well-known application or watching a video. By clicking on the download link, the victim will download a trojan and execute it on the operating system. This process will allow the trojan creator to run other malicious actions for example, copying the data from the victim's computer or making sever operational or financial harm to other users on the same network.

Third, after a victim's computer is infected, the trojan will start to post a web links on the profile of the online social network. In this case, the victim's friends will think that this post is created by their friend and its unharmful so they will click on that link which cause downloading and executing a trojan on their computer. Consequently, the computers and the profiles of the victim's friends will also be infected, and the same case will be repeated for the other friends as described in the second and the third stage alternately.

Forth, the trojan will post a malicious link on the profile of the latterly victims to recruit their friends and lure them by attracting them to click on the link. This link could be fake web page to play a video then the link will ask the victim to download a plug-in in order to play this video. Then, the chain processes of attacking new victims will continue. In the case that social network does not have a wall to post a link on it, the trojan will send a malicious link to the victim's friends using a private message of the social network.

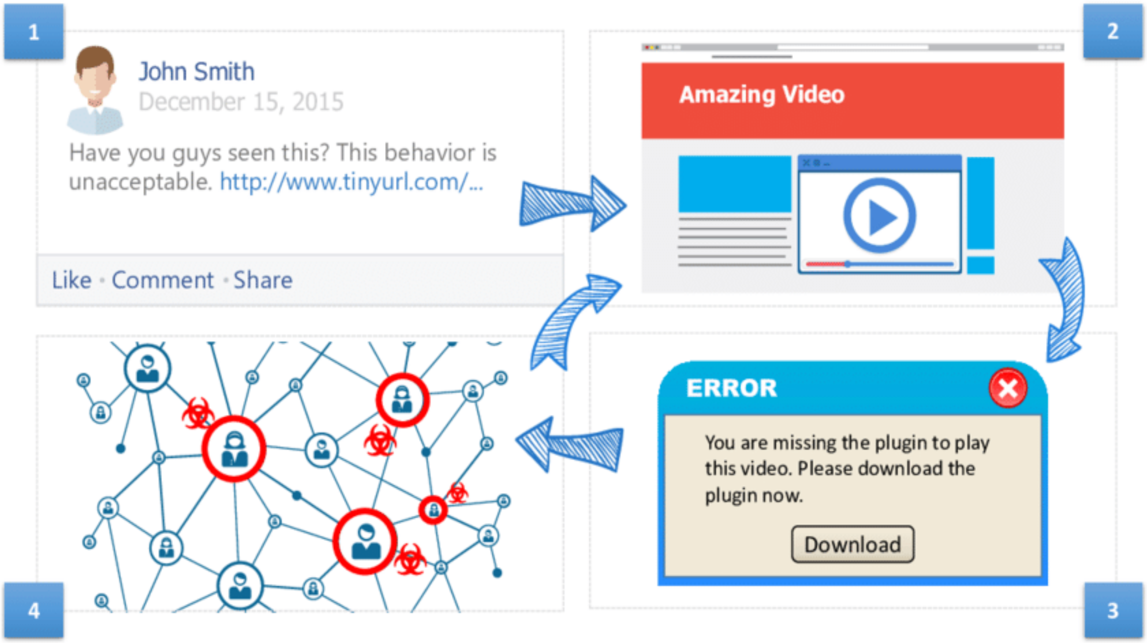

Figure 3.6: Example of malicious URLs spread in social media

3.6 MALWARE DISTRIBUTION NETWORK

A Malware Distribution Network is described as a collection of trojan distribution trees of landing web pages, trojan repository server, and intermediate fake sites [37]. The purpose of the Malware Distribution Network is to transfer the user from a landing page to the intermediate fake sites which lead to trojan repository server. The malicious script of the trojan is downloaded to the user's computer either by clicking on the malicious web site link or automatically by exploiting a malicious script on the malware repository server. Traffic is guided to landing pages in Malware Distribution Network which could be done by using different techniques for example rogue application download link, search engine referrers.

Usually, the procedure of attacking a user in Malware Distribution Network is when a user search in the internet for example about downloading a music player software. The trojan developer exploits SEO to increase the rank of a landing web page in order to making landing page appears as the first result of the search engine. The user will visit the crafted landing page because it is appeared in the top which means it's the best suggestion from the search engine to the requested search query.

After the user visits the crafted landing page where it should lead to downloading the music player software, the Malware Distribution Network redirects the user through a sequence of intermediate landing pages which lead to the malware repository server. After that, a malicious script from the malware repository server is downloaded to the computer of the victim as it's the requested music player software. The user will install the malicious script without knowing it's a malware. Consequently, installing the malicious script causes a several of attacks from keylogging and distribute it to other users. The difficulty in detecting the Malware Distribution Network is cloaking because of dynamically changing the malware repository and the intermediate landing page which is also changed for the same user after two visits.

3.7 UNIFORM RESOURCE LOCATOR

URLs are the page locations on the web, they are displayed on the browser's address bar, in search results, they are utilized in the case when someone is linking to your page within their blog/website, bookmarking the page in their browser or mentioning your page on any of the social-networking sites. The structure of URL is: Protocol + Domain Name +Port + sub domain+ Path to file/resource. Yet, the port number is omitted in the case when typing the address into the web browser: URL Example:

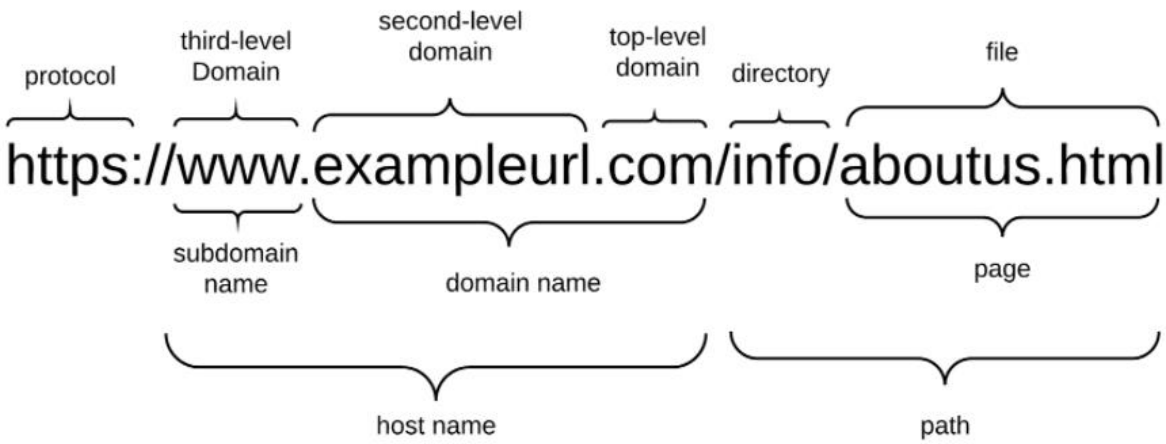

Figure 3.7: Example of url structure

- Protocol= http for websites.
- domain name= exampleul.com
- subdomain=www
- Port Number=TCP/UDP socket or port number.Websites use port 443.
- Path to file=info/aboutus.htm

In this case, the protocol is hypertext transfer protocol secure (https) utilized via web browsers for getting web pages, and the major encountered one [38]. Also, the protocol has been separated from the domain name with **delimiters** that were the colon plus 2 forward slashes://

4. PROPOSED METHOD

4.1 MACHINE LEARNING

Even before the modern concept of computer proposed by Alan Turing, mankind has always searched for ways to either automate or facilitate more complex tasks. With the technological advances that provide higher computation power, computer science fields like Artificial Intelligence (AI) have greatly improved in recent years. One of the AI fields that has benefit from the technological advances is ML. The current subsection will provide a more detailed description of ML and its use for the current work. ML is a subset of AI in which algorithms are applied to learn and make predictions on data, or more formally defined by Tom Mitchell [36]:

“A PC program is said to gain for a fact E concerning some class of assignments T and execution measure P, if its exhibition at errands in T, as estimated by P, improves with experience E.”

A noteworthy nuance about the given definition, and the ML field in general, is that these algorithms are not designed to deal with specific problems. Meaning that as long as a learning problem is well defined through T, P and E, an algorithm based on these three features is able to learn, but not necessarily solve any problem. This creates a contrast to AI fields, where algorithms are built for specific tasks. After defining a learning problem, to obtain a useful outcome (i.e. model) one or more learning algorithms are applied. Learning algorithms fall within one of the following classes: supervised learning; unsupervised learning; semi-supervised learning or reinforcement learning.

Each of the categories is defined on how the problem is learned . A more detailed description, as seen in [39] is as follows:

4.1.1 Supervised Learning

In this category the program is given a corpus (i.e. experience) with input- output pairs and it must learn how to map a never seen input into the correct output. Within supervised learning, a learning problem can be further classified into a classification problem, if the output is a set of finite values (e.g. is it going to rain tomorrow?), or into a regression problem, if the output is continuous (e.g. how likely is it to rain tomorrow?).

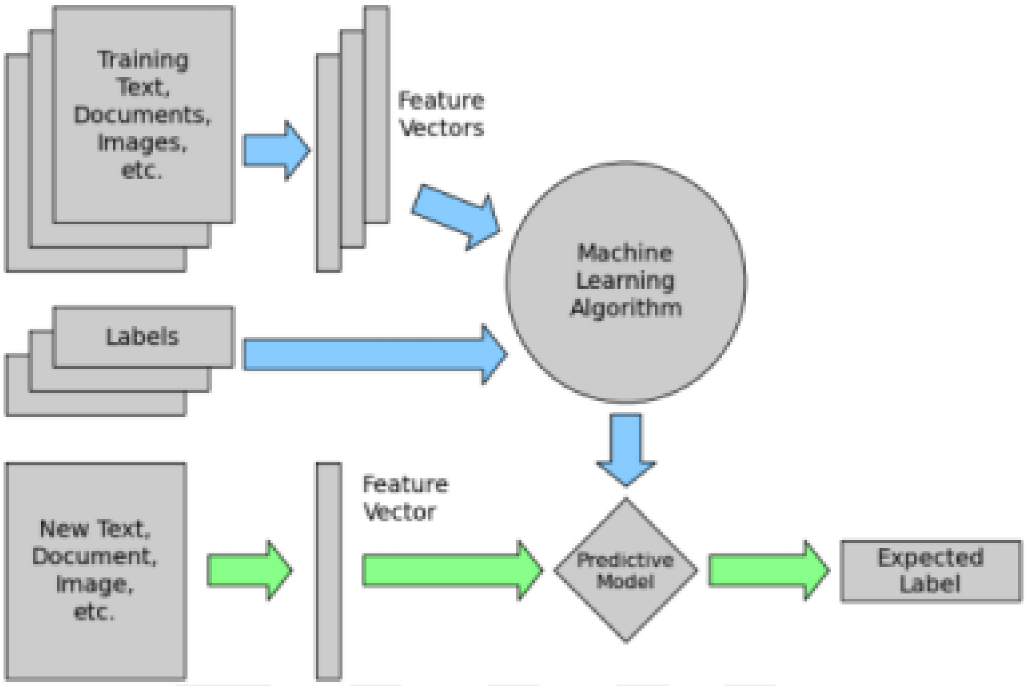

Figure 4.1: Diagram of the Supervised Learning Model

Figure 2. describes the flow of supervised learning. The blue arrows indicate the entry of the training data, which go through the obtaining of characteristics (feature vectors), considered the most descriptive and significant for the data. In the example of the SIEM system, the characteristics can be four:

(1) Number of requests to the server,

(2) hours,

(3) if any vulnerability has been detected and

(4) if the server is up to date. Next, you need to provide it respective label whether or not it corresponds to a security event.

4.1.2 Unsupervised Learning

In such category, the program was given a corpus and it should be learning patterns from input; thus, no wrong or correct output was anticipated, just the way that various inputs are relating. In unsupervised learning, the major task was clustering, in which the input was grouped on the basis of comparable properties.

4.1.3 Semi-supervised Learning

The category is a combination of the previous ones and exists because sometimes the learning problem is based on a dataset that combines both labeled and unlabeled inputs or on a dataset that contains noisy data. The combination of supervised and unsupervised learning provides a better overall result in such scenarios[37].

4.1.4 Reinforcement Learning

In this category the program is given a problem through a set of states furthermore, a bunch of activities it can perform. At any given states the program performs an action which results in a reward or punishment. The cumulative reinforcement at each state teaches the program the best and worse actions for each state. The current work will fit into the supervised learning category, with the purpose of creating a model to assert whether a given sample is malware or not. This issue can be viewed as whether a pure classification issue, with a binary output, or as a regression problem, with the likelihood of a sample being malware. When designing a learning system, multiple choices must be taken into consideration. A brief description of what should be taken into account is given as seen in [38]:

• Training experience is the first choice, as it can altogether affect the achievement or failures of the learner. This feedback provided by the experience can be direct or indirect, affecting the end result of the system. Another important aspect is how the experience represents the distribution over which the performance is measured, meaning how well the experience represents the environment where the system is to be used. Being what is going to be fed to the learner (i.e. features), it must be well thought out and understood.

• Target function is the next choice, representing what is to be learned. The function take as inputs the sets of feature that represents the problem and (ideally) outputs a solution to the task.

• Target function representation is the remaining choice, this is where a learning model is applied to learn the given task. The choice is of great importance, given it will influence how well the system performs under evaluation.

4.2 SUPPORT VECTOR MACHINE

SVMs are the major utilized algorithms showing strengths in such case, also it is the major utilized learning algorithm for regression problems and classification. The model of SVM represents the examples as points in multi-dimensional space, shown in a way that the individual categories' examples were divided into clear boundaries that have been as wide as possible [43]. Put differently, the algorithm is revealing that the maximal margin is separating at least two classes. Assuming that the extracted features related to training dataset vectors from real numbers $x_i \in \mathbb{R}^p$, benign and malicious URLs were categorized into 2 classes, referred to as vector $y \in \{1, -1\}^n$. In addition, SVM is solving the next primal problem:

$$\min_{\omega, b, \zeta} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{n}\zeta_i \quad \text{substitution to } \begin{array}{l} y_i\left(\omega^T\varphi(x_i) + b\right) \geq 1 - \zeta_i \\ \zeta_i \geq 0,\ i = 1, \ldots, n \end{array} \tag{4.2}$$

Its dual is

$$\min_{\alpha} \frac{1}{2}\alpha^T\mathbb{Q}\alpha - e^T\alpha \quad \text{substitution to } \begin{array}{l} y^T\alpha = 0 \\ 0 \leq \alpha_i \leq C,\ i = 1, \ldots, n \end{array} \tag{4. 2}$$

in which e was all ones' vector, $C > 0$ is upper bound, Q represents n-by-n positive semi-definite matrix [44] Pseudo values in 2D Cartesian coordinates were provided in *Figure 5*. In such instance, the number of positive (1 and 2) as well as negative (3 and 4) points, that have been closer to opposing class, form the supporting vectors. Furthermore, such margin's center line was referred to as *hyperplane*; the model makes classification on such bases..

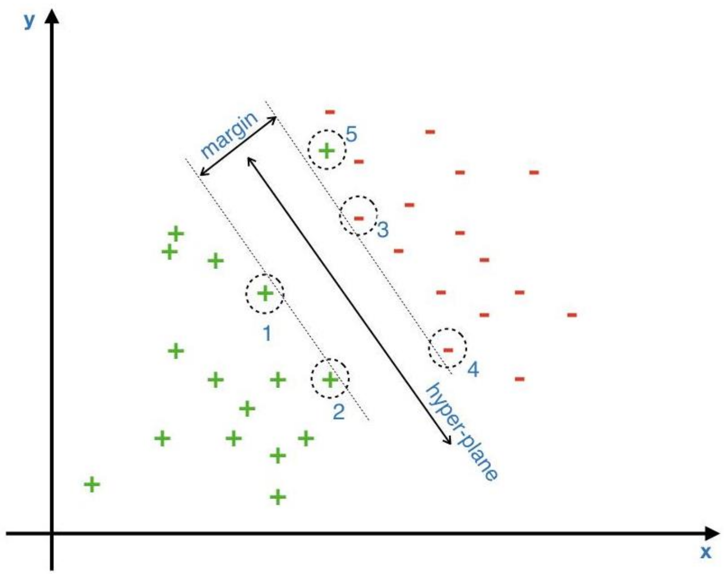

Figure 4.2: SVM classification

The distance between such vectors is considered as a margin which is often strive for attaining maximum value, that has been a constrained optimization problem which might be formulated in the following way [45]:

$$\min f(x) \quad \text{subject to} \quad \begin{array}{lll} g_i(x) = c_i & for\ i = 1, \ldots, n & Equality\ constraints \\ h_i(x) \geq d_j & for\ j = 1, \ldots, m & Inequality\ constraints \end{array} \quad (4.2)$$

in which the inequality and quality constraints needs to be satisfied, also $f(x)$ is specified as one of the objective functions which must be optimized subject to constraints. By using constraints, the algorithm's ratio in terms of outliers might be regulated.

Also, such algorithm has some benefits making it significant for the tasks related to binary as well as multi class classification. An advantage is that the number of data dimensions doesn't impact the classification's accuracy. Put differently, a classification remains efficient even in the case when the number of features was pre-determined via the number of observations.

"Support Vector Machines" conveys high exactness results since it utilizes an enhancement technique. SVM constructs a classifier via looking for an isolating hyperplane (optimal hyperplane) which is optimals and maximium the margins that isolates the classifications (for our study malisious urls and benign urls). Subsequently, SVM has the upside of strength when all is said in done and adequacy when the quantity of measurements is more noteworthy than the quantity of tests.

4.3 MULTINOMIAL NAIVE BAYES

applies term frequency, for instance, the number of times a specified term is appearing in a document. Often, the term frequency was normalized via dividing the raw term frequency by the length of document. Following normalization, the term frequency might be utilized for computing the highest likelihood estimates according to the training data for estimating conditional probability[47]. Naive Bayes classifier its linear classifier that are known for being straightforward yet extremely productive. The probabilistic model of (naive Bayes) classifier depends on (Bayes) hypothesis, and the descriptive word naive comes from the supposition that the highlights in a dataset are commonly autonomous. Practically speaking, the autonomy supposition that is regularly disregarded, however naive Bayes classifiers actually will in general perform very well under this unreasonable presumption [1]. Particularly for little example sizes, naive Bayes classifiers can beat the more remarkable choices . Being moderately strong, simple to execute, quick, and precise, (naive Bayes) classifier are utilized in various fields. A few models incorporate the analysis of sicknesses and settling on choices about therapy measures , the order of RNA groupings in ordered examinations, and spam separating in email customers . Nonetheless, solid infringement of the freedom suppositions and (non linear) order issues can prompt extremely horrible showings of (naive Bayes) classifier. We need to remember that the kind of information and the sort issue to be addressed direct which classifications models we need to pick. Practically speaking, it is constantly prescribed to think about various classifications model on the specific dataset and considers the predicte performance just as computational proficiency.

For what reason was Naive Bayes utilized? Guileless Bayes has highly powerful learning and forecast, it's frequently used to contrast and more refined techniques since it's quick and highly adaptable (functions admirably with highly dimensional data) and as Andrew Ng proposes when managing a ML issue start by attempting with a straightforward brisk and afterward grow starting there**.**

Naive Bayes easy and fast its depends on (Bayes) hypothesis and term (Naive) on the grounds that it accepts that component are free of one another given the class(no/little correlation between highlights), which isn't reasonable. Hence, Naive Bayes can learn singular highlights significance yet can't determine the relationship among highlights. In addition, the preparation time with Naive Bayes is fundamentally more modest rather than elective strategies and it doesn't need a lot preparing information.

4.4 LOGISTIC REGRESSION

analysis examines the relation between categorical dependent variable as well as the set of independent (explanatory) variables. In addition, the name logistic regression was utilized in the case when dependent variable has just 2 values, like 0 and 1 or Yes and No. Usually, the name multinomial logistic regression is reserved for the case when the dependent variable has at least three distinctive values, like Widowed, Single, Married or Divorced. Even though the type related to the data utilized for dependent variable was distinctive from that of multiple regression, the procedure's practical use was comparable [42].

It indicates the regression equation, also the goodness of fit, confidence limits, odds ratios, deviance and likelihood. It is performing comprehensive residual analysis such as diagnostic residual reports as well as plots. It might be performing independent variable subset selection search; it looks for the best regression model with smallest number of independent variables. Also, it is providing confidence intervals on the predicted values along with providing ROC curves to help determining the best cutoff point for classifications. Furthermore, it is allows validating the results via classifying (automatically) the rows which weren't utilized throughout analysis.

4.5 FEATURE EXTRACTION

Text feature extraction can be defined as a process used to tokenize, count and normalize the string data for being fed into the algorithms of machine learning. Generally, just some features' groups for URL classification were majorly utilized.

• URL-Based Features have been extracted from URL. The instances were total address's length, number of digits in URL, legitimate name of brand in sub-domain or number of sub-domains.

• Domain Based Features includes information regarding domain registrant as well as the domain itself, for instance, number of days are passing since the domains was enlisted or if registrant name is covered up.

• Pages-Based Feature were more difficult to acquire, then the others. These features were available from tools such as Google Analytics – number of visited pages, average visit duration, and so on. In the majority of cases, Page Rank was simplest metric to acquire.

• Content-Based Features were collectable with a method referred to as web scraping. Also, the data was extracted from website and analysis has been conducted.

The presented study is focusing on utilizing URL based features (particularly word occurrences) – whereas the dataset does contains already-blocked as well as inactive URLs, there was no possibility for extracting content based, page based or domain-based features. On the other hand, using URL based features was possible safer

(there was no requirement for downloading anything from website) and representing one of the significant challenges for classifying URL with minimal memory and performance requirements.

The research "Heuristics-based Approach's for Phishing website Detect used URLs Feature" that has been published in 2015 described the Korean team's solution for the detection of phishing sites with the use of the features of the URL [48]. The team presented 26 features that have been utilized to detect the mechanism that has been created by them. The first set of the features has been associated with the suggestions of the Google search engine. The first 3 features have been based upon the similarity of the primary domain, sub-domain and the path to Google suggestions. The rest of the 3 features check whether the searched term (sub-domain, path and domain) has been presented in a white list. The Levenshtein distance between both terms is computed. In the case where the search term has been similar to proposed term, then there will be more likelihood that it will be marked as suspicious, due to the fact that the site could be utilized as trap in the case where an address/word is misspelled by user [49]. In the case where the Levenshtein distance has been equal to 0 (i.e. the two terms are identical), then the domain will be added to trustworthy white list, and this site is possibly legitimate.

In the 2nd group, there are 3 features that are obtained by the ranking of the page. The PageRank can be defined as an algorithm, offering a way to measure a web-site's significance. The malicious web-sites typically have quite a low value of the page rank, due to the fact that those sites aren't visit by many peoples, and they only exists for a small period of the time [28]. The 3rd group is about the analysis of URL patterns as well as address structure. The URLs seldom contain special characters and the legitimate sites typically have 1 "top-level domain". Which is why, fundamentally, the TLDs may be signifying fraudulent sites. In the URL property values of the next group are checked. The temporary malicious URLs usually don't utilize the protocol of the HTTPS and doesn't have WHOIS or DNS records. Finally, the sub-domain length is computed, and the terms of phishing in URL are checked. Ultimately, the researchers have performed a comparison of numerous algorithms on a sample that included 6,000 URL.Finally, they have evaluated the "random forest" classifier to be the optimal, for their case. Unfortunately, the parameters of the model haven't been mentioned, and their testing and training data are unavailable, which is why, there is no ability for reproducing their result.

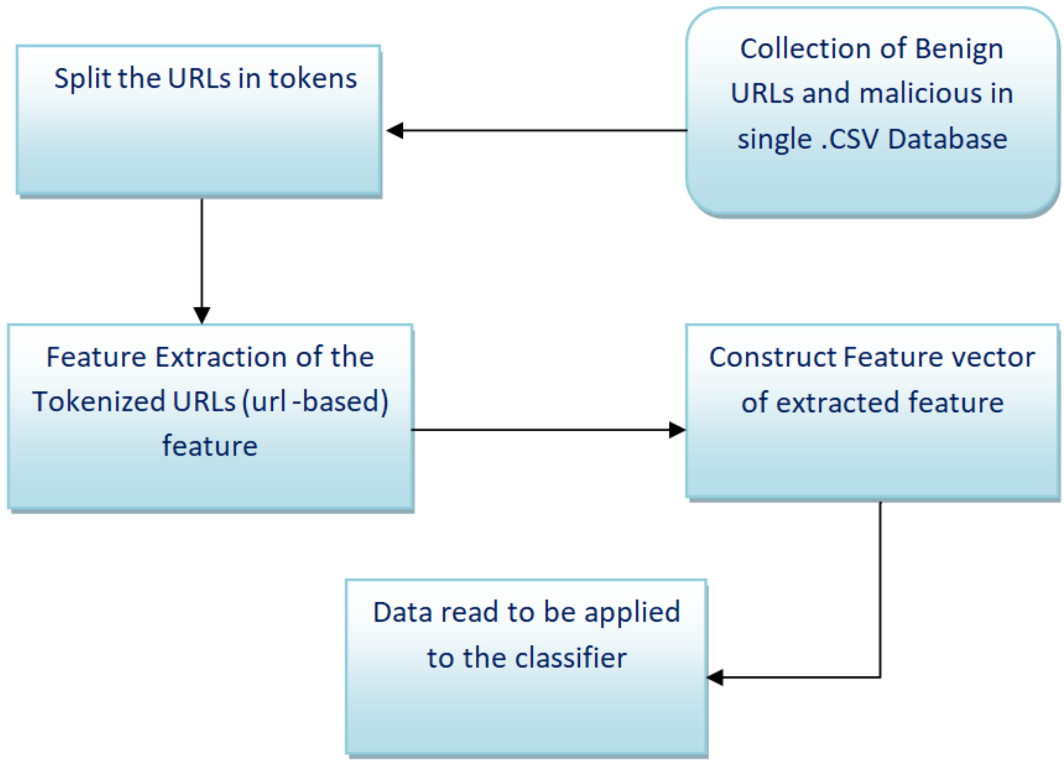

Figure 4.3: Diagram show Feature Extraction & Vectorization

4.6 DATA COLLECTION

The database used in this experiment will be in the form of A comma-separated values (CSV) that contains malicious websites and other benign websites and that the total number of the database is 13002 and the number of malicious samples is 6500 while the number of benign samples is 6501, we obtained from the Kaggle community in this url 'https://www.kaggle.com/antonyj453/URLdataset', which is a free database, In addition to that, One important feature has been extracted

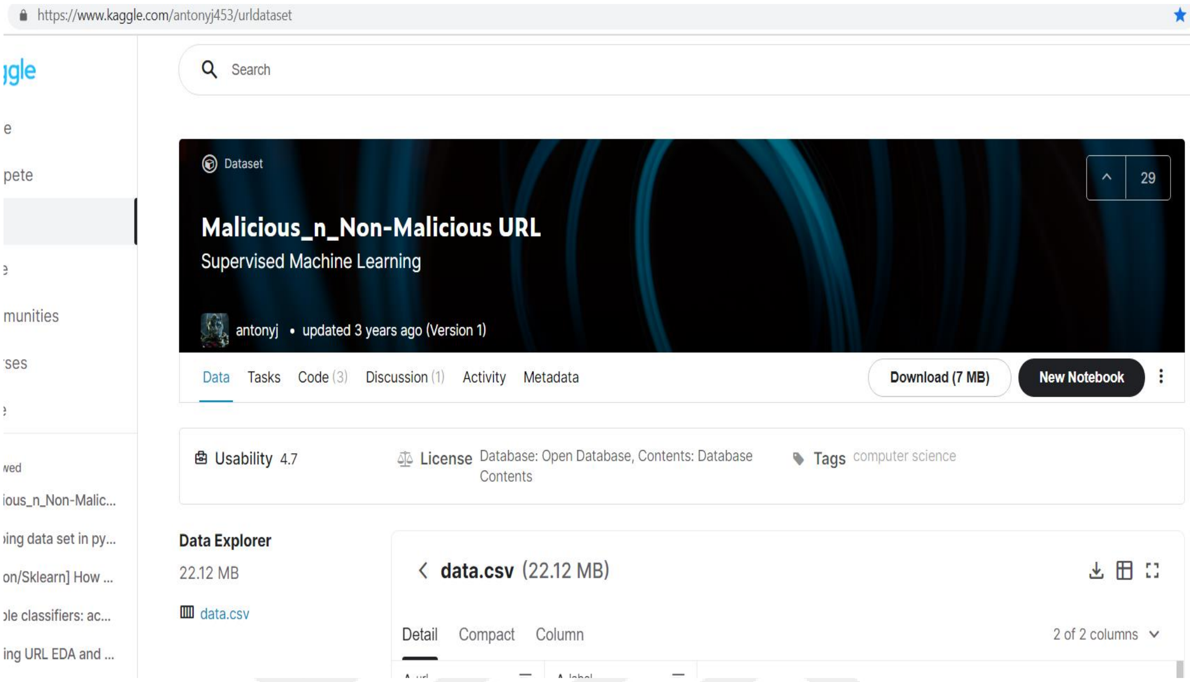

Figure 4.4: The database used for the experiment

4.7 TOKENIZATION

In the case of this study, the URL string tokenization that operates through the modelling of output distribution, considering that all the words are distributed in an independent manner [50]. For the purpose of the proper tokenization of the continuous stream of the words (which are referred to as the string with no spaces), there is a need for having a list of the words that have been sorted from higher to lower with the frequency. The relative word frequency in a certain language may be computed, such as from the data-set of Wikipedia articles. For every one of the words in the set of the frequency, calculation of the "word-cost" is performed by using eq2.1, where N represents the total amount of the words and k ∈ {1, 2, . . . , N} represents the word rank in the data-set.

$$word\ cost = \log(k \times \log(N)) \tag{4.8}$$

Eq. (**4.**) is known as word cost.

Assuming that s is a URL. Dynamic programming is utilized for inferring space positions. As the initial step, the array of the cost has been built, which contains 1 number for every one of the characters in original string s. This array has been created through the iteration through the characters in s, computing minimal cost of first i characters, according to the word-cost, the character value at a position $i - 1$ and each one of the potential contiguous subsequences of the characters with the common upper border, which has been given by index i. Afterwards, the array of the cost is backtracked for the recovery of the string of minimum-cost. The array of the characters, which has been characterized by s is iterated once more, however, backwards. The minimal cost of character set that ends with $j := length(s)$ has been obtained (therefore, the array of cost has been required), in combination with the number of the characters l, representing minimal value. The interval of the characters from the $j - l$ index to j index is one of those words, we have been searching for. Now a new character of ending has been fixed as $j = j - l$ and the backtracking carries on to the point where $j = 0$. The benefit of that solution is the fact that implementation has consumed a linear amount of the memory and time [51]. However, the drawback is the need for having the corpus of the words that have been arranged by the relative frequency of the words, which is similar to what the tokenizer will encounter in fact (for example, the correct language), in the opposite case, results become quite bad [51]. implementing this algorithm can be seen at appendix B. finally, the 4 special tokens have been removed - TLD com, sub domain www and protocols http, https. They exist to a level which will be quite unsuitable for drawing the conclusions on the basis of their existence.

Helloworldmynameisahmed → (Hello, world, my, is, ahmed)

2ZStYTgkFr1U85XjLeE5 → (2, z,sty,tgk, fr, 1, u, 85, xj, lee, 5)

4.8 VECTORIZATION

In order to count and normalize, scikit-learn's TF-IDF vectorizer has been utilized. The TF-IDF is short for the term-frequency times inverse document-frequency, it's one of the statistical measures that are utilized for the evaluation of the significance of the word to the document in corpus (2.3). In this equation, the term-frequency (TD) is the amount of times where the expression t takes place in the record d – which has been referred to as the raw count. By default, the normalized term-frequency has been utilized through the use of the by scikit-learn (2.2). There are other options, which may be utilized, such as the Boolean frequency, in which the term-frequency (t, d) = 1 if t happens in given d and 0 otherwise, or logarithmically scaled frequency where the term-frequency (t, d) is substituted by 1 + log(TF (t, d)).

$$\mathrm{TF}\,(t,d) = \frac{\text{number of times } t \text{ appears in } d}{\text{total number of terms in } d} \tag{2.2}$$

$$\text{TF-IDF}\,(t,d) = \mathrm{TF}\,(t,d) \times \mathrm{IDF}\,(t) \tag{2.3}$$

$$\mathrm{IDF}\,(t) = \log \frac{1+n}{1+\mathrm{DF}\,(t)} + 1 \tag{2.4}$$

The inverse document-frequency has been calculated based on eq. (2.4) where n represents total amount of the documents within the corpus. A function DF (t) represents the amount of the documents in a corpus, containing the term t. after that, the vectors of the TF-IDF have been normalized by Euclidean (L2) norm (2.5). vnorm = v ||v||2 = v √ v1 2 + v2 2 + · · · + vn 2 (2.5)

$$v_{norm} = \frac{v}{||v||_2} = \frac{v}{\sqrt{{v_1}^2 + {v_2}^2 + \cdots + {v_n}^2}} \tag{2.5}$$

As it has been observed seen, the Tf-IDF Vectorizer has 2 beneficial characteristics: (a) it results in lowering common word weights, (b) it applies the Euclidean normalization following the computation of the representation of the TF-IDF. The implementation of the scikit learn TF-IDF vectorizer includes 2 parts. As it has been stated in section2.2.1, every one of the URLs in the corpus is split to separate words – i.e. tokens. After that, the Count Vectorizer performs the conversion of this set of words into a matrix of the token counts .

An encoded matrix will be returned with the length of whole vocabulary and integer count for how many times every one of the words had appeared in a collection. In the case of this study (the URL word occurrences), this matrix will be highly sparse (the majority of matrix elements are equal to 0). The entire number of the features in the data-set that has been presented is approximately 1150000, whereas every one of the URLs includes only a small number of the tokens (in an order of the units). The rest of the constituent parts of the TF-IDF vectorizer is the transformer of the TF-IDF, which performs the conversion of the count matrix into normalized TF or TF-IDF portrayal, rather than raw frequency values of the token occurrence, due to the fact that the aim is scaling words' importance down, which are encountered in a more frequent rate in a certain corpus.

4.9 EXPERIMENTAL ENVIROMENT

The processing of collected data have been performed in Python programming language v 3.6.0. This programming language selected due to the personal preference of the researcher as well as the availability of all of the necessary libraries (such as the Pandas library) for the preprocessing of data. The entire list of utilized libraries of Python with the descriptions may be found in the Appendix .

The procedure of choosing the tool-kit to build classifier has been a subject of the thorough analyses. In a short list of the considered tools have been products like the SciKitLearn, Matlab, RapidMiner, Weko, and R of Python. Following critical analyses, it has been determined to utilize the SciKitLearn that has been implemented in Python environment. The SciKitLearn has been selected according to the following 2 fundamental reasons: initially, the integrity. As it has been stated earlier, Python has been utilized as well for a variety of the experimental tasks. Which is why, there is a convenience in having one integrated environment for the entire experiment. The other reason is the speed of the. The speed of the processing some tool-kits has been examined through running a cross-validation and learning process. Which is why, it has been discovered that the SciKit-Learn has been of the highest speed compared to other libraries, tool-kits, and on-line services seems as one of the common practices amongst the community of the data scientists. The only one of the experimental environment limiting aspects has been the lack in the computational power to train the classifier. The data-set training process has taken a few hours, which definitely had an adverse impact upon the experience of the research. The full list of the tools that have been utilized may be discovered in *Table 4.1*.

Table 4.1: Experimental tools

#	(Tool)	(Name)	(Version)
i.	Programming language	Python	(3.6.0)
ii.	Machine learning library for the Python	scikit-learn	(0.24.1)
iii.	Code editor	Spyder	(4.4.0)

4.10 EVALUTION METRICS

This phase has been considered as one of the main parts of the entire experimentation and answers the question: when this experimentation can be considered as successful? For the purpose of finding the answer for that question, the tasks below have to carried out:

a. Choosing the suitable metric to measure the classifier's accuracy;
b. Comparing the metrics that have been selected against the dissimilar values of a similar experimentation that has given a successful result. As a result, those 2 situations are expected from results:

i. The hypothesis failure – the chosen metrics will have a great degradation of analogous values of the other experimentations;

ii. The justification of the hypothesis – the metric value is near the values of the other researches or even exhibits a performance enhancement. Despite obtaining a certain result, relevant conclusions will be made.

Typically, the predictive models' efficiency is assessed through 4 fundamental metrics: True-Positive (TP), True-Negative (TN) False-Positive (FP), and False-Negative (FN). Those metrics definitions have been represented as:

- TP – the case of the classifier correctly detecting malicious URL
- TN – the case of the classifier correctly detecting a benign URL
- FP – the case of the classifier incorrectly detecting benign URL
- FN – the case of the classifier incorrectly detecting malicious URL

The graphical representation of those metrics for the binary classifiers have been illustrated in Figure12, which can be referred to as Error or Confusion Matrix as well [52]

		Condition	
		Present	Absent
Test	Positive	True-Positive	False-Positive
	Negative	False-Negative	True-Negative

Figure 4.5: Structure of confusion matrix

However, there are as well some sophisticated measures, which have given a more generalized understanding about the model's accuracy, including the False Positive Rate (FPR), True Positive Rate (TPR), Precision and Recall that can be computed from abovementioned fundamental metrics. In general, those advanced metrics with the formulas of calculation can be found in Table 4.2.

Table 4.2: Advanced metrics of evaluations

#	Metric	Formula
1	TPR (True Positive Rate or Sensitivity)	TP / (TP+FN)
2	FPR (False Positive Rate or Specificity)	FP / (FP+TN)
3	Precision	TP/(TP+FP)
4	Recall	TP / (TP+FN)

From the table above, it has been observed that the Recall and the TPR are the same. As a result, it leads to the following question: then why they have been called differently? In fact, this is, in part, as a result of the fact that the FPR and TPR are often utilized to build a ROC curve (i.e. receiver operating characteristic curve), which is why, in literature, they're typically utilized in combination. After that, the community of the data science found the Precision and Recall that have been utilized in pairs too. Which means that it is one of the most common practices for measuring the accuracy of a classifier with the use of the ROC curve (which has been constructed from the FPR and TPR) or the Precision and Recall. Based on numerous sources [53], [54], authors have set out a variety of the recommendations concerning when those metrics have to be utilized. From those recommendations, a number of the simple and important ideas may be obtained in the case where the scholars' argument supplied by math is de-mystified. Fundamentally, the Precision is the recommended metric [53] in which data-set has mainly negative (i.e. benign samples of the URL) compared to the positive class (which mainly includes the malicious URL samples). Which is a result of the fact that the precision is more focused upon positive class, which is why, there is a greater number of the chances for more precisely detecting the malicious URL. Those methods are implemented, for instance, when misdetection of the malicious URLs (i.e. the cases of False Positive) has lamentable results. However, the TPR and the FPR (i.e. the metrics of the ROC) measure the capability for distinguishing between 2 classes (ibid.). Therefore, the metrics of the ROC curve have to be utilized in the case of detecting the two classes has the same importance. Which results from the fact that those metrics provide the same weight to the prediction ability of the two classes. Typically, the metrics of the

ROC curve are utilized in the case where both classes are in balance, or in the case where positive class is larger. In general, considering the recommendations abovementioned, it has been decided measuring the accuracy of the classifier utilizing Recall and Precision, based on the factors below:

i. as it has been shown in the chapter of methodology, the data-set that has been collected is not balanced – due to the fact that it includes more negative samples compared to the positive;

ii. this study has been carried out under an assumption of the fact that a collision with malware URLs will yield in seriously negative results. Which is why, security has a high level of importance, thus, the classifier's end result will be assessed using Recall and Precision. In particular, the results of metrics are going to be compared to similar ones from earlier researches. Following this comparison, important conclusions are going to be drawn, which may be discovered in the chapter of Results and Analysis.

5. SIMULATION AND RESULTS

This chapter present of the experiment's results.

5.1 CLASSIFIERS

There is a wide range of the options of classifiers to select for the text classification. To name a few supervised learning options [44]:

• Naive Bayesian

• Support Vector Machine

• Logistic Regression

Every one of those approaches has benefits and drawbacks, determining when and how these approaches may be utilized. The aim of the present study isn't training a high classification success rate algorithm. The classifiers have been selected for the purpose of presenting a wide range of the potential solutions with rather high rates of classification. The SVM has been utilized for the classification of a sample with matches in the training data and after that, utilizing the label of the closest matches for the class prediction. A comparison has been carried out with each one of the algorithms that have been mentioned and the result has been the maximum accuracy in SVM.

5.2 DESCRIBING DATA

The present section provides more through description of about the manipulation process with data-set prior to the experimentation. The amount of the samples in those sub-sets has been presented in *Table 5.1*.

Table 5.1: Proportions of testing and training Patterns'

	Total	Total %
Train set	9101	70%
Test set	3900	30%
	13002	

The above table illustrates the classes' breakdown in testing and training data-set. As observed from this table, the entire amount of the samples has been 13,002 where 9,101 URL samples have been in training sub-dataset and 3,900 URL samples have been in the sub-set of testing. This proportion has been equal to 30% - 70% - 30% for the testing- training sub-sets.

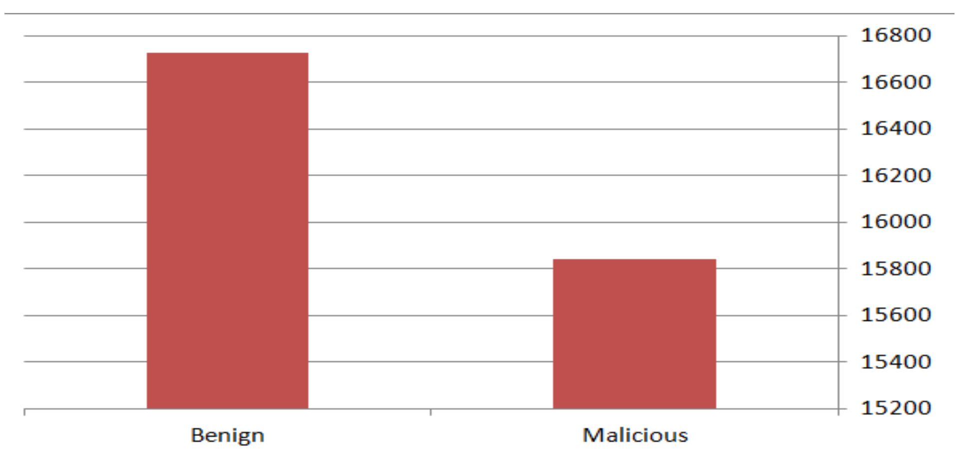

Chart 5.1: Chart column showing the number of benign and Malicious in Dataset

RESULTS

This section includes the information on the efficiency of the classifier. The metrics of Recall and Precision have been selected to evaluate the classifier's efficiency. Those metrics' values have been listed in Table 5.2 .

Table 5.2: Results of support vector machine

Class	Precision	Recall	F1-score	Accuracy
1: benign	0.96	1.00	0.96	
0: malicious	0.93	0.56	0.71	
Total				93%

Table which showed that the general rate of accuracy has been 93%. however, ability to classify the malicious URL samples is low, where the Recall has been 0.56 and Precision has been 0.96. Which may be seen as well in the matrix of Confusion that has been illustrated in Table 5.2.

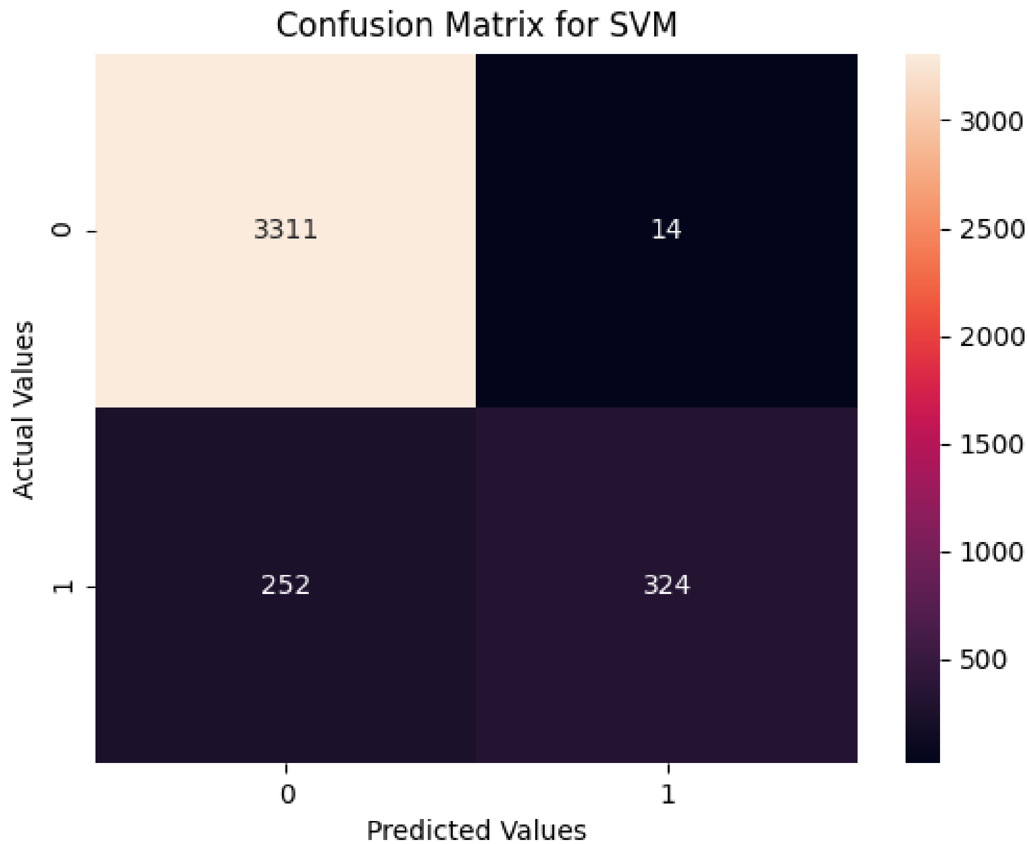

Figure 5.1: Confusion matrix for support vector machine

The abovementioned matrix has shown that the model misclassified (FN) malicious URL samples 252 times. Even though, this classifier copes with classifying benign URLs quite efficiently, the malicious URLs' misclassification rate has been considered quite high.

.

The data from each algorithm that has been utilized for the generation of ROC curve have been utilized in Figure 5.3 The ROC curve can be defined as a metric of performance of the ratings at a variety of the threshold settings, giving information of how well a model can distinguish amongst the classes.

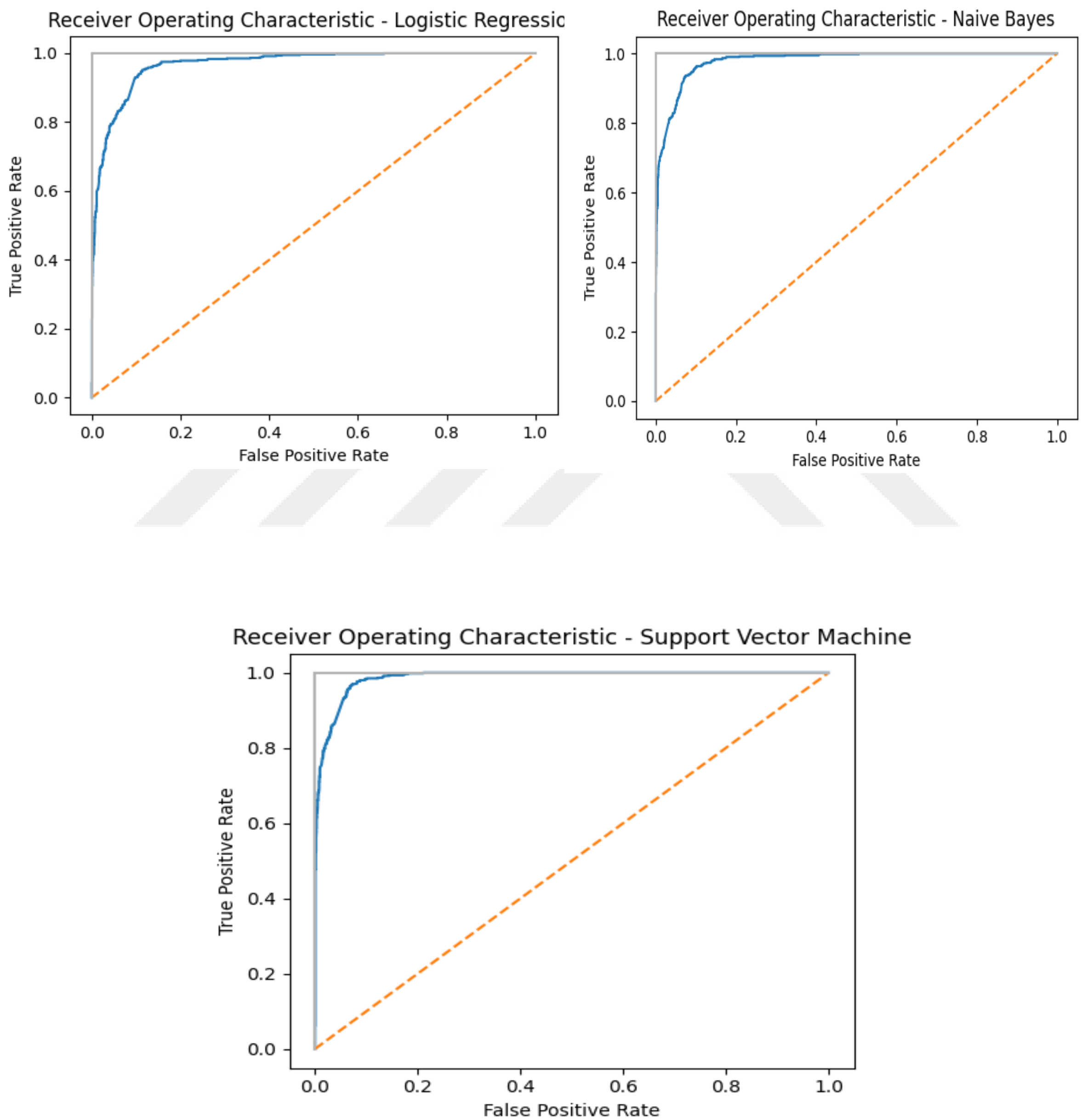

Figure 5.2: characteristics (ROC) curve

5.4 CLASSIFIER COMPARISION

In this section we compare the results of the classifiers that were used in this study, the table below shows the results of the algorithms, the table shows the accuracy of the four algorithms.

Table 5.3: Comparison between our proposed algorithm and other existing algorithms

Algorithm	Accuracy
Naive Bayes	89 %
Logistic Regression	88%
Support Vector Machine	93%

As shown in the above table, the support vector machine algorithm shows the highest accuracy

CONCLUSION

In conclusion, the main objective of the current study was to detection malicious URL on online social Network by developing a predictive model to identify URLs that point to malicious URL using the (URLs Based) Features are extricated from URLs. As an example the total length of address, numbers of digit in URL, legitimate name of brands in sub domain features of URLs .The literature review of previous studies showed the security threats that occur through malicious addresses, and through this knowledge the most appropriate feature was used to build this model. This was achieved through the development of a new model to detect malicious URLs on online social networks, Supervised Machine Learning, in this case with the Support Vector Machines algorithm, was optimal for detecting malicious content, through the selected characteristic. the model was Successful in classifying URLs into malicious or benign with 93% accuracy. however , can be improved by considering other new characteristic. for the model. The malicious content observed corresponds to what can hinder the natural functioning of web browsing, such as the execution of scripts, the spread of viruses, Trojans and exploitkits. However, which should be treated as a potential threat because the final destination of the URL is unknown. Therefore, they must be translated prior to access. The use of URLs by a shortening service that pays for each link accessed is an attractive way to spread malicious content, because in addition to generating a profit it also achieves access to the malicious site. Especially, these short URLs are shared in the viral themes, which are attractive to users. There is a set of short URLs whose translation is another short URL going through more than two levels of translation before obtaining the destination URL. These URLs mostly lead file download repositories. which in turn included other URLs with possible malicious content, whose access should be controlled. It is necessary to implement safeguards in web browsers, specifically they must have the sandboxing function, which show warning messages about the accesses (to the camera, microphone, file downloads, etc.) that the website requires. In this way the user can accept or deny the requested permissions. Nowadays , machine learning technologies are considered an important role to make our use of the Internet in a safe manner, this important technological development through which machine learning techniques have greatly contributed to detection malicious URLs on Online social networks.

REFERENCES

[1] A. B.Sayamber and A. M. Dixit, ‘Malicious URL Detection and Identification’, *IJCA*, vol. 99, no. 17, pp. 17–23, Aug. 2014, doi: 10.5120/17464-8247.

[2] S. Abraham and I. Chengalur-Smith, ‘An overview of social engineering malware: Trends, tactics, and implications’, *Technology in Society*, vol. 32, no. 3, pp. 183–196, Aug. 2010, doi: 10.1016/j.techsoc.2010.07.001.

[3] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, ‘Beyond blacklists: learning to detect malicious web sites from suspicious URLs’, in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD ’09*, Paris, France, 2009, p. 1245, doi: 10.1145/1557019.1557153.

[4] A. Blum, B. Wardman, T. Solorio, and G. Warner, ‘Lexical feature based phishing URL detection using online learning’, in *Proceedings of the 3rd ACM workshop on Artificial intelligence and security - AISec ’10*, Chicago, Illinois, USA, 2010, p. 54, doi: 10.1145/1866423.1866434.

[5] W. Zhang, Y.-X. Ding, Y. Tang, and B. Zhao, ‘Malicious web page detection based on on-line learning algorithm’, in *2011 International Conference on Machine Learning and Cybernetics*, Guilin, China, Jul. 2011, pp. 1914–1919, doi: 10.1109/ICMLC.2011.6016954.

[6] D. Chiba, K. Tobe, T. Mori, and S. Goto, ‘Detecting Malicious Websites by Learning IP Address Features’, in *2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet*, Izmir, Turkey, Jul. 2012, pp. 29–39, doi: 10.1109/SAINT.2012.14.

[7] P. Zhao and S. C. H. Hoi, ‘Cost-sensitive online active learning with application to malicious URL detection’, in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, Chicago Illinois USA, Aug. 2013, pp. 919–927, doi: 10.1145/2487575.2487647.

[8] K. Kandasamy and P. Koroth, ‘An integrated approach to spam classification on Twitter using URL analysis, natural language processing and machine learning techniques’, in *2014*

IEEE Students' Conference on Electrical, Electronics and Computer Science, Bhopal, Mar. 2014, pp. 1–5, doi: 10.1109/SCEECS.2014.6804508.

[9] M. N. Feroz and S. Mengel, 'Phishing URL Detection Using URL Ranking', in *2015 IEEE International Congress on Big Data*, New York City, NY, USA, Jun. 2015, pp. 635–638, doi: 10.1109/BigDataCongress.2015.97.

[10] M. Olalere, M. T. Abdullah, R. Mahmod, and A. Abdullah, 'Identification and Evaluation of Discriminative Lexical Features of Malware URL for Real-Time Classification', in *2016 International Conference on Computer and Communication Engineering (ICCCE)*, Kuala Lumpur, Malaysia, Jul. 2016, pp. 90–95, doi: 10.1109/ICCCE.2016.31.

[11] M. Weedon, D. Tsaptsinos, and J. Denholm-Price, 'Random forest explorations for URL classification', in *2017 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA)*, London, United Kingdom, Jun. 2017, pp. 1–4, doi: 10.1109/CyberSA.2017.8073403.

[12] C. Liu, L. Wang, B. Lang, and Y. Zhou, 'Finding effective classifier for malicious URL detection', in *Proceedings of the 2018 2nd International Conference on Management Engineering, Software Engineering and Service Sciences - ICMSS 2018*, Wuhan, China, 2018, pp. 240–244, doi: 10.1145/3180374.3181352.

[13] M. Nikola, 'History of malware', Feb. 2013, [Online]. Available: https://www.researchgate.net/publication/235666537_History_of_malware.

[14] kaspersky, 'Learn about malware and how to protect all your devices against it', 2019. [Online]. Available: https://www.kaspersky.com/resource-center/preemptive-safety/what-is-malware-and-how-to-protect-against-it.

[15] C. Iwendi *et al.*, 'KeySplitWatermark: Zero Watermarking Algorithm for Software Protection Against Cyber-Attacks', *IEEE Access*, vol. 8, pp. 72650–72660, 2020, doi: 10.1109/ACCESS.2020.2988160.

[16] R. Rahim, H. Nurdiyanto, A. Saleh A, D. Abdullah, D. Hartama, and D. Napitupulu, ‘Keylogger Application to Monitoring Users Activity with Exact String Matching Algorithm’, *J. Phys.: Conf. Ser.*, vol. 954, p. 012008, Jan. 2018, doi: 10.1088/1742-6596/954/1/012008.

[17] D. Adachi and K. Omote, ‘A Host-Based Detection Method of Remote Access Trojan in the Early Stage’, in *Information Security Practice and Experience*, vol. 10060, F. Bao, L. Chen, R. H. Deng, and G. Wang, Eds. Cham: Springer International Publishing, 2016, pp. 110–121.

[18] T. Vladislav, ‘Facebook grant scam acts using phishing’, 2020. [Online]. Available: https://me.kaspersky.com/blog/facebook-grants/8606/.

[19] John. Aycock, *Computer viruses and malware*. .

[20] welivesecurity, ‘Malware behind 25% of cyber attacks’, Aug. 2014. [Online]. Available: https://www.welivesecurity.com/2014/08/06/malware-behind-25-cyber-attacks-dos-last-year-says-cert-team/.

[21] R. Langner, ‘Stuxnet: Dissecting a Cyberwarfare Weapon’, *IEEE Secur. Privacy Mag.*, vol. 9, no. 3, pp. 49–51, May 2011, doi: 10.1109/MSP.2011.67.

[22] B. Morrow, ‘BYOD security challenges: control and protect your most sensitive data’, *Network Security*, vol. 2012, no. 12, pp. 5–8, Dec. 2012, doi: 10.1016/S1353-4858(12)70111-3.

[23] X. Hu and N. Sastry, ‘Characterising Third Party Cookie Usage in the EU after GDPR’, in *Proceedings of the 10th ACM Conference on Web Science - WebSci ’19*, Boston, Massachusetts, USA, 2019, pp. 137–141, doi: 10.1145/3292522.3326039.

[24] M. R. Faghani and H. Saidi, ‘Social Networks’ XSS Worms’, in *2009 International Conference on Computational Science and Engineering*, Vancouver, BC, Canada, 2009, pp. 1137–1141, doi: 10.1109/CSE.2009.424.

[25] W. Fan and K. H. Yeung, ‘Online social networks—Paradise of computer viruses’, *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 2, pp. 189–197, Jan. 2011, doi: 10.1016/j.physa.2010.09.034.

[26] D. Wang, D. Irani, and C. Pu, ‘A social-spam detection framework’, in *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference on - CEAS ’11*, Perth, Australia, 2011, pp. 46–54, doi: 10.1145/2030376.2030382.

[27] L. Wu, B. Brandt, X. Du, and Bo Ji, ‘Analysis of clickjacking attacks and an effective defense scheme for Android devices’, in *2016 IEEE Conference on Communications and Network Security (CNS)*, Philadelphia, PA, USA, Oct. 2016, pp. 55–63, doi: 10.1109/CNS.2016.7860470.

[28] R. Perrotta and F. Hao, ‘Botnet in the Browser: Understanding Threats Caused by Malicious Browser Extensions’, *IEEE Secur. Privacy*, vol. 16, no. 4, pp. 66–81, Jul. 2018, doi: 10.1109/MSP.2018.3111249.

[29] S. Ni, Q. Qian, and R. Zhang, ‘Malware identification using visualization images and deep learning’, *Computers & Security*, vol. 77, pp. 871–885, Aug. 2018, doi: 10.1016/j.cose.2018.04.005.

[30] K. Thomas and D. M. Nicol, ‘The Koobface botnet and the rise of social malware’, in *2010 5th International Conference on Malicious and Unwanted Software*, Nancy, France, Oct. 2010, pp. 63–70, doi: 10.1109/MALWARE.2010.5665793.

[31] M. Reza Faghani and U. Trang Nguyen, ‘Unmasking Magnet, a Novel Malware on Facebook’, in *2020 International Symposium on Networks, Computers and Communications (ISNCC)*, Montreal, QC, Oct. 2020, pp. 1–4, doi: 10.1109/ISNCC49221.2020.9297210.

[32] J. Zhang, R. Zhang, Y. Zhang, and G. Yan, ‘The Rise of Social Botnets: Attacks and Countermeasures’, *IEEE Trans. Dependable and Secure Comput.*, vol. 15, no. 6, pp. 1068–1082, Nov. 2018, doi: 10.1109/TDSC.2016.2641441.

[33] B. Livshits and W. Cui, ‘Spectator: Detection and Containment of JavaScript Worms’, p. 14, 2008.

[34] G. Wang, J. W. Stokes, C. Herley, and D. Felstead, ‘Detecting malicious landing pages in Malware Distribution Networks’, in *2013 43rd Annual IEEE/IFIP International Conference on*

Dependable Systems and Networks (DSN), Budapest, Hungary, Jun. 2013, pp. 1–11, doi: 10.1109/DSN.2013.6575316.

[35] S. Jessica and B. John, ‘URL (Uniform Resource Locator)’. [Online]. Available: https://searchnetworking.techtarget.com/definition/URL.

[36] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[37] N. N. Pise and P. Kulkarni, ‘A Survey of Semi-Supervised Learning Methods’, in *2008 International Conference on Computational Intelligence and Security*, Suzhou, China, Dec. 2008, pp. 30–34, doi: 10.1109/CIS.2008.204.

[38] L. P. Kaelbling, M. L. Littman, and A. W. Moore, ‘Reinforcement Learning: A Survey’, *jair*, vol. 4, pp. 237–285, May 1996, doi: 10.1613/jair.301.

[39] V. Jakkula, ‘Tutorial on Support Vector Machine (SVM)’, p. 13.

[40] V. Yadav, ‘A phosphate transporter from the root endophytic fungus Piriformospora indica plays a role in phosphate transport to the host plant’, *Biol Chem*, 2010.

[41] W. S. Noble, ‘What is a support vector machine?’, *Nat Biotechnol*, vol. 24, no. 12, pp. 1565–1567, Dec. 2006, doi: 10.1038/nbt1206-1565.

[42] Great Learning Team, ‘Multinomial Naive Bayes Explained’, 2020. [Online]. Available: https://www.mygreatlearning.com/blog/multinomial-naive-bayes-explained/.

[43] J. Brownlee, ‘Logistic regression for machine learning. Machine Learning Mastery’, 2016. [Online]. Available: https://machinelearningmastery. com/logistic-regression-for-machine-learning.

[44] C. H. Lee, D. H. Kim, and J. L. Lee, ‘Heuristic based Approach for Phishing Site Detection Using URL Features’, in *Third International Conference on Advances in Computing, Electronics and Electrical Technology - CEET 2015*, Apr. 2015, pp. 131–135, doi: 10.15224/978-1-63248-056-9-84.

[45] A. Zamir *et al.*, ‘Phishing web site detection using diverse machine learning algorithms’, *EL*, vol. 38, no. 1, pp. 65–80, Jan. 2020, doi: 10.1108/EL-05-2019-0118.

[46] stackoverflow, ‘split text without spaces into list of words’, 2012. [Online]. Available: https://stackoverflow.com/questions/8870261/how-to-split-text-without-spaces-into-list-of-words/11642687#11642687.

[47] P. Kolari, ‘Detecting Spam Blogs: A Machine Learning Approach’, p. 6, 2000.

[48] S. Stehman, ‘Selecting and interpreting measures of thematic classification accuracy’, vol. 62 (1), pp. 77–89, 1997, doi: 10.1016/S0034-4257(97)00083-7.

[49] J. Davis and M. Goadrich, ‘The relationship between Precision-Recall and ROC curves’, *University of Wisconsin-Madison, Madison, WI*, Pittsburgh, Pennsylvania, USA, pp. 233–240, 2006.

[50] D. Martin, ‘Evaluation: from Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation’, 2011.

APPENDIX

```
# Import Dependencies
# pandas is a software library for data manipulation and analysis.
import pandas as pd

# library used for arrays., functions for working in domain of linear algebra
import numpy as np

#Convert a collection of raw documents to a matrix of TF-IDF features.
from sklearn.feature_extraction.text import TfidfVectorizer

#a suite that contains libraries and programs for statistical language processing. It is one
of the most powerful NLP libraries
import nltk

#library of functions that can extend the basic features of python
import random

#is a collection of command style functions that make matplotlib work
import matplotlib.pyplot as plt

#Python data visualization library based on matplotlib
import seaborn as sns

#Evaluate a score by cross-validation
from sklearn.model_selection import cross_val_score

#Build a text report showing the main classification metrics
from sklearn.metrics import classification_report, confusion_matrix

#Naive Bayes classifier for multinomial models
from sklearn.naive_bayes import MultinomialNB

#Logistic Regression classifier
from sklearn.linear_model import LogisticRegression

#Support vector machines classifier
from sklearn import svm

#Split Dataset
from sklearn.model_selection import train_test_split

#Compute Area Under the Receiver Operating Characteristic Curve (ROC AUC) from
prediction scores
from sklearn.metrics import roc_curve, roc_auc_score
```

```
#Data vectorization using Tfidfvectorizer
#creating user defined tokenizer(not using default tokenoizer)
#spliting data,removing repetition and "com"
#custom tokenizer for URLs.
#first split - "/"
#second split - "-"
#third split - "."
#remove ".com" (also "http://", but we dont have "http://" in our dataset)
def function_tokens(input):
tokensBySlash = str(input.encode('utf-8')).split('/') # make tokens after splitting by slash
allTokens = []
for i in tokensBySlash:
tokens = str(i).split('-') # make tokens after splitting by dash
tokensByDot = []
for j in range(0,len(tokens)):
tempTokens = str(tokens[j]).split('.')
tokensByDot = tokensByDot + tempTokens
allTokens = allTokens + tokens + tokensByDot
allTokens = list(set(allTokens)) #remove redundant tokens
if 'com' in allTokens:
allTokens.remove('com') #removing .com
if 'org' in allTokens:
allTokens.remove('org') #removing .org
if 'www' in allTokens:
allTokens.remove('www') #removing .www
if 'net' in allTokens:
allTokens.remove('net') #removing .net
if 'edu' in allTokens:
allTokens.remove('edu') #removing .edu
if 'info' in allTokens:
allTokens.remove('info') #removing .info
if 'gov' in allTokens:
allTokens.remove('gov') #removing .gov
#allTokens = (list(nltk.bigrams(allTokens)))
return allTokens


#read from a CSV file
data1 = pd.read_csv("Dataset.csv",',')    # reading training file

#convert it into numpy array and shuffle the dataset
data1 = np.array(data1)
random.shuffle(data1)
```

```
#convert text data into numerical data for machine learning models
y1 = [d[1] for d in data1]
url = [d[0] for d in data1]
vectorizer = TfidfVectorizer(tokenizer=function_tokens)
x1 = vectorizer.fit_transform(url)

#split the data set in to train and test
X_train, X_test, y_train, y_test = train_test_split(x1, y1, test_size=0.3, train_size=0.7)

# Train Machine Learning Models

# Train Machine Learning Models
#checking accuracy of model using test data

#1 naive bayes

print("Naive Bayes\n")
nav=MultinomialNB()
nav.fit(X_train,y_train)
NaiveBayesScores = cross_val_score(nav,x1,y1,cv=10)
print(NaiveBayesScores)
NaiveBayesData_y_pred=nav.predict(X_test)

print(classification_report(y_test,NaiveBayesData_y_pred))
print("Accuracy: %0.2f (+/- %0.2f)\n" % (NaiveBayesScores.mean(),
NaiveBayesScores.std() * 2))


# Display the Confusion Matrix
plt.title('Confusion Matrix for Naive Bayes')
sns.heatmap(confusion_matrix(y_test,NaiveBayesData_y_pred),annot=True,fmt="d")
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()


# Getting predicted probabilities
y_score1 = nav.predict_proba(X_test)[:,1]


# Ploting Receiving Operating Characteristic Curve
# Creating true and false positive rates
false_positive_rate1, true_positive_rate1, threshold1 = roc_curve(y_test, y_score1)

print('roc_auc_score for Naive Bayes: ', roc_auc_score(y_test, y_score1))
```

```
# Ploting ROC curves
plt.subplots(1, figsize=(10,10))
plt.title('Receiver Operating Characteristic - Naive Bayes')
plt.plot(false_positive_rate1, true_positive_rate1)
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()



#2 - Logistic Regression
print("Logistic Regression results\n")

LogisticRegressionModel = LogisticRegression()
LogisticRegressionModel.fit(X_train, y_train)
LogisticRegressionScores = cross_val_score(LogisticRegressionModel,x1,y1,cv=10)
print(LogisticRegressionScores)
logisticRegressionData_y_pred = LogisticRegressionModel.predict(X_test)
print(classification_report(y_test,logisticRegressionData_y_pred))
print("Accuracy: %0.2f (+/- %0.2f)\n" % (LogisticRegressionScores.mean(),
LogisticRegressionScores.std() * 2))


# Display the Confusion Matrix
plt.title('Confusion Matrix for Logistic Regression')
sns.heatmap(confusion_matrix(y_test,logisticRegressionData_y_pred),annot=True,fmt="
d")
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()



# Getting predicted probabilities
y_score2 = LogisticRegressionModel.predict_proba(X_test)[:,1]


# Ploting Receiving Operating Characteristic Curve
# Creating true and false positive rates
false_positive_rate2, true_positive_rate2, threshold2 = roc_curve(y_test, y_score2)
```

```
print('roc_auc_score for Logistic Regression: ', roc_auc_score(y_test, y_score2))


# Ploting ROC curves
plt.subplots(1, figsize=(10,10))
plt.title('Receiver Operating Characteristic - Logistic Regression')
plt.plot(false_positive_rate2, true_positive_rate2)
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()




#3 - SVM
print("Support Vector Model results\n")
svmModel = svm.SVC(kernel='linear', probability=True)
svmModel.fit(X_train, y_train)
SVMScores = cross_val_score(svmModel,x1,y1,cv=10)
print(SVMScores)

SVMdata_y_pred = svmModel.predict(X_test)
print(classification_report(y_test,SVMdata_y_pred))
print("Accuracy: %0.2f (+/- %0.2f)\n" % (SVMScores.mean(), SVMScores.std() * 2))

# Display the Confusion Matrix
plt.title('Confusion Matrix for SVM')
sns.heatmap(confusion_matrix(y_test,SVMdata_y_pred),annot=True,fmt="d")
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()


# Getting predicted probabilities
y_score3 = svmModel.predict_proba(X_test)[:,1]


# Ploting Receiving Operating Characteristic Curve
# Creating true and false positive rates
false_positive_rate3, true_positive_rate3, threshold3 = roc_curve(y_test, y_score3)

print('roc_auc_score for Support Vector Machine: ', roc_auc_score(y_test, y_score3))
```

```
# Ploting ROC curves
plt.subplots(1, figsize=(10,10))
plt.title('Receiver Operating Characteristic - Support Vector Machine')
plt.plot(false_positive_rate3, true_positive_rate3)
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```