DETECTING PHARMING ATTACKS ON WEBSITES USING SVM (SUPPORT VECTOR MACHINE) CLASSIFIER

BY

ISATOYE EMMANUEL OLUWASOGO

CYS/16/9986

SUBMITTED TO

THE DEPARTMENT OF CYBER SECURITY,

THE FEDERAL UNIVERSITY OF TECHNOLOGY AKURE (FUTA),

ONDO STATE, NIGERIA

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE

AWARD OF BACHELOR OF TECHNOLOGY (B. TECH)

IN CYBER SECURITY

FEBRUARY 2023

CERTIFICATION

I certify that this project work was carried out by me and has not been presented elsewhere for the award of any degree or any other degree or any other purpose.

STUDENT'S NAME: **ISATOYE EMMANUEL OLUWASOGO**

SIGNATURE________________ DATE ___________________

This is to certify that this work was carried out by **ISATOYE EMMANUEL OLUWASOGO** with matriculation number **CYS/16/9986** of the Department of Cyber Security The Federal University of Technology, Akure, Nigeria.

SUPERVISOR'S NAME: **DR O. OWOLAFE**

SIGNATURE _________________ DATE __________________

DEDICATION

This report is dedicated to the almighty God, who granted me good health, guided and protected me all through these years and made this project a success. I also dedicate it to my sponsors and guardians for their unending support.

ACKNOWLEDGEMENT

I am deeply grateful to God for His mercy, grace, favor, and love that sustained me throughout my undergraduate studies. I would like to express my appreciation to my project supervisor, Dr. O. Owolafe, for her valuable corrections, direction, guidance, and supervision.

I also extend my thanks to the Head of Department (Dr. A.J Gabriel) and all of the staff in the Cyber Security Department at the Federal University of Technology in Akure for providing me with the knowledge, skills, and values that formed the foundation for this project.

My gratitude also goes to my loved ones and friends for their support, without whom this would not have been possible. I pray that God will bless them all.

ABSTRACT

Cyber attackers are constantly attempting to deceive individuals and organizations in order to cause financial harm, steal sensitive information, and damage their reputation. One specific type of attack, known as a "pharming" attack, has become particularly problematic for website users due to its severe consequences. This type of attack involves stealing a user's login credentials and redirecting them to a malicious website using DNS-based techniques. To enhance security, Transport Layer Security/Secure Sockets Layer (TLS/SSL) was developed. It functions by verifying the authenticity of the web server for the user, rather than the other way around, thus ensuring the reliability of both parties through the use of digital certificates. However, it should be noted that even with SSL in place, pharming attacks can still occur. Results from research have shown that a proposed technique utilizing SVM (Support Vector Machine) provides 99.1% accuracy, along with high performance in terms of precision, sensitivity, and specificity, highlighting the effectiveness of this machine learning method in detecting pharming attacks.

TABLE OF CONTENT

Table of Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background of the study

Pharming is derived from words farming and phishing. Pharming is basically a tricking practice in which malevolent code is installed on victims' computer or server to redirect the user to fake websites without their consent. It is also known as ― phishing without wooing. (Manhas *et al.*, 2019).

Pharming an advance phishing attack. Both Pharming and phishing are used for identity theft. Pharming can also refer as "Phishing without lure". Pharming attacks redirect you to a hacker's site even when you type the address of a real site into your browser. Like phishing, pharming coerces victims into visiting a fake website and supplying information. However, instead of tricking recipients into clicking on an email link, pharming can secretly redirect victims to a fraudulent website directly from their web browser. Pharming effectively eliminates the need for "bait" emails and is therefore potentially more dangerous than "normal" phishing scams and can cast a wider "net" in which to snare victims. (Patel *et al.*, 2013).

Pharming attack is the sophisticated version of phishing attack which redirect users to a clone website that looks exactly the same as the legitimate website a user is trying to reach and it is done by posioning the DNS (Domain Name System) to route the user to that fraudulent site, then steals any information submitted by the user. Pharming can happen either by changing the hosts file on a victim's computer or by exploitation of a vulnerability in DNS (Domain Name System) server software. DNS servers are computers responsible for resolving internet names into their real addresses (IP addresses). Modification of DNS servers are sometimes called "poisoned". The term pharming is new based on farming and phishing. (El-Buhaisi, 2013).

Pharming attack targets the domain name lookup system (DNS) of a victim and compromise the DNS infrastructure so that DNS queries for the victim site's domain return an attacker-controlled IP address. This can be accomplished via several techniques, including DNS cache poisoning and DNS response forgery. Pharming attacks are particularly devious because the browser's URL bar will display the domain name of the legitimate site, potentially fooling even the most meticulous users. The main achievement is identity theft; phishers try to fool web visitors into revealing their login credentials, sensitive personal information, or credit card details with the intent of impersonating their victims for financial gain. (Karlof *et al.,* 2017).

During the time numerous software vendors and managed service providers developed sophisticated tools and techniques to help protect against the threat of Phishing, attackers tended to make use of obfuscation methods to disguise the true destination reached by the customer. Pharming attacks works in the background without the knowledge of the victim by manipulating various components of core domain and host naming systems (HOST file) to misdirect the customer to an alternative destination under their control. (Ollmann, 2015).

Pharming is modern cybercrime involving redirecting victim to fraudulent website to steal their valuable personal information. Users are tricked into handing over their banking info or other account credentials on fake website resembling a legitimate one (of financial institution, for instance). When an attack is carried out at this level, many of the security solutions devised for protecting against Phishing attacks are also deceived. Unfortunately, pharming is hard to detect you can't tell fake website from real one. Sure, there may be some visual differences, but the fake website will have the same URL due to the clever technology utilised by the cybercriminals behind the attack. This complicated modern scam requires redirecting web traffic for victim - or several victims at once - on the DNS server level. Pulling from a pool of well-known exploit techniques includes DNS hijacking, DNS spoofing, cache poisoning, Host file manipulation, and many more. Many users can be victimized because the attacker does not need to target users one by one, one successful attempting poisoning the ISP DNS server can be potentially used to trick all users into getting name resolutions from that DNS server. Since Pharming does not rely on the victim taking action like clicking on a link in an email that later leads to information theft, it is much more challenging to identify and thus far more effective. (Ollmann, 2015).

The term Phishing appeared in early 1996, but it was not until the end of 2003 that email-based phishing attacks began to become a popular attack vector for cybercriminals as a means to conduct financial fraud and identity theft. In 2005 we had the first warning about new attacks that corrupt some DNS servers; in this attack, a colossal number of (.com) sites were directed to forgery servers maintained by attackers. In March 2005, a warning was issued by SANS (SysAdmin, Audit, Network, and Security) organization about new attacks by attackers that corrupted some DNS servers so that requests for several ".com" sites were directed to alternative servers maintained by them - the attackers. Using a rogue DNS server, posing as an authoritative DNS server for a particular .com domain, Pharmers poisoned the cache of several ISP-level DNS servers and requests for more than 900 unique Internet addresses and more than 75,000 email messages redirected using even less complex techniques. In January 2005, the domain name for a large New York ISP (Panix) was hijacked to a site in Australia, while earlier, in 2004, a German teenager hijacked the eBay.de domain name. Through simple social engineering and a reasonable telephone manner, an attacker can often fool a domain registrar and gain control of a domain. Such a thing happened on the 24th April 2005 when users of the secure webmail service - "Hushmail" – were redirected to a "defaced" webpage. Pharming can happen either by changing the hosts' file on a victim's computer or exploiting a DNS (Domain Name System) server software vulnerability. Modification of DNS servers is sometimes called "poisoned."

The term pharming was coined from the word farming and Phishing. In recent years both Pharming and Phishing have been used for online identity theft information. Pharming has become a significant concern to businesses hosting e-commerce and online banking websites. Pharming attacks are particularly devious because the browser's URL bar will display the domain name of the legitimate site, potentially fooling even the most meticulous and intelligent users. Evidence suggests they may become a more severe threat soon. The DNS server plays a significant role in Pharming.

1.2 Motivation

The use of websites in the 21st century is growing widely, and it is not going away anytime soon. Everything is being done over the Internet, and most of them are carried out using websites. Buying and selling, booking flights, hotels, event centers, almost all social media despite having their respective apps are accessible through the web. 80% of organizations have websites running daily on the Internet. Imagine visiting a website trying to buy a Shoe online using a credit card without knowing that the ISP DNS server or the local host file has been corrupted. The URL input (which is correct) has taken you to a clone website without your knowledge, and you input your credit card details intending to shop for a Shoe, and you press "BUY" to submit your details unknowingly to give them to an attacker. Pharming attacks are dangerous to users since they are used to steal sensitive information like usernames, passwords, and credit card numbers, and it is much more dangerous and brutal to detect since the user cannot notice the difference between a standard site and a forged site in URL and website look like a phishing attack. A method based on dual step scrutiny and collaboration of numerous DNS servers explains how IP address checks can be used to check the authenticity of a visited website.

A client-side approach for Detecting pharming attacks by using an Authorized DNS server IP address matching approach. We select IP addresses from the local and legitimate servers and then check them. If the IP address coming from the local match with the one from the Authorized server, then we conclude the website is trustworthy, and then traffic coming in will be allowed, but if the IP addresses did not match then, we know Pharming has occurred (Gastellier *et al.,* 2016).

Bharat Arya (2016) Client-Side Anti Pharming (CSAP) methodology to avert pharming attacks at the client-side explained how multiple DNS servers were used to check the validity of the DNS response given by the local DNS server. If The local server gives iPs different from other DNS server IPs, it is tagged malicious.

Another approach by Shengnan *et al.* (2015) uses a hybrid detection model to detect pharming attacks based on IP address and website content. IP address filtration through different DNS servers is conducted. The problem caused by employing this method sometimes the analysis requires time, and most of the time, they are not accurate.

The primary motivation to research more on SVMs has been their robustness proved in various applications in the past and its increased attention for anomaly detection. The reason to study SVM Classifiers is to overcome the areas others detecting approaches left open. The study of SVM classifiers is not intended to replace the previously proposed methods but to address some of the shortcomings that prevent them from becoming the most robust anomaly detection technique in monitoring the pharming attacks. For example, the IP address check setting up multiple DNS can have a blind spot for a point nearby but still far away from the average data. Most of the time, the users often think that a secure lock (padlock) in front of their URL can protect them from possible attacks, but it is no more the case now as attackers have gone one step ahead to find out the vulnerabilities exploit them. Pharming attacks are found on various SSL-protected websites. There are different methods to enhance the existing machine learning techniques. A random forest technique, a machine learning technique used to solve regression and classification problems, had been used to overcome this problem but unable to overcome input feature values that are similar or very close to each other. This area gives birth to the idea of the proposed work to introduce SVM to overcome overfitting issues.

1.3 Objectives

The objectives are to:

i. Design a SVM-Based model for Pharming attack.

ii. Implement the designed model.

iii. Evaluate the developed model.

1.4 Methodology

The training process adopted by this methodology will be based on training the SVM classifiers with websites legitimate data collected from a trusted source and stored in a database. Then each time we visit any website, we will have to go through the steps: Dataset Collection from the visited website, process them based on some given rules which will be discussed later, and then Extract Relevant Feature like SSL validation, IP address, Domain name, Hidden link or redirecting link, etc. and Classification which focus on generating a hyper-plane splitting two objects and optimally dividing the two objects along with considerable separation distance between them. Evaluation then compares both data using the following parameters Accuracy, Precision, Sensitivity, and Specificity, then head to the last step to detect if the website is a pharming website or a legitimate one.

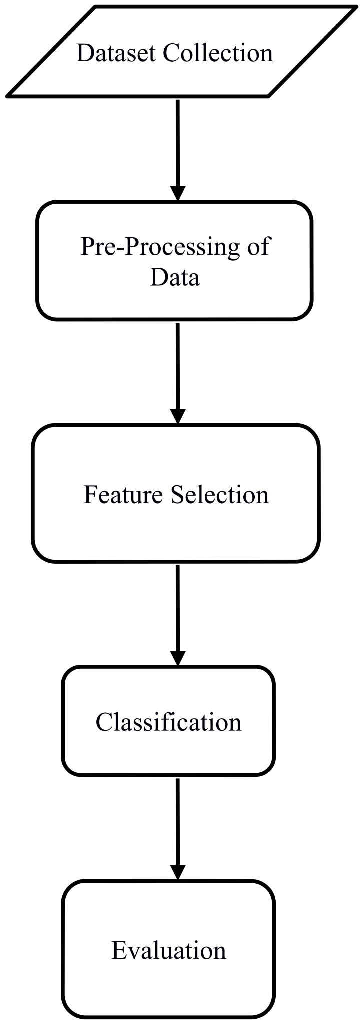


Figure 1.1: SVM Trainning And Detection Process

1.5 Expected Contribution to Knowledge

At the end of this research, a SVM based model for pharming attack would have been developed. This SVM classifiers are trained to detect pharming attacks on websites. This research aims to contribute to the knowledge on the use of machine learning techniques for detecting pharming attacks on websites. The proposed approach, an SVM-based model, is expected to be a valuable contribution to the field of pharming attack detection. Support Vector Machines (SVMs) are a popular and powerful class of machine learning algorithms that have been successfully used in various applications such as image classification, natural language processing, and bioinformatics. In conclusion, this research will provide a new approach for detecting pharming attacks on websites using SVM classifiers. It is expected to offer a powerful and effective solution for protecting users from these types of attacks and will provide a valuable contribution to the field of machine learning and cyber security.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

Pharming is a form of domain spoofing. In simple terms, rather than spamming you with email requests to confirm your financial or personal information, pharmers work invisibly. They change your local DNS server to redirect your Web request to a fake site. This means that when you enter a web address, such as www.futa.edu.ng; you will be taken to a fake website rather than the legitimate website managed by the university. As far as you know, you're connected to the correct site. No email is involved, and if the attacker cloned the appearance of the real site perfectly, victims would have no way to know that anything was wrong.

2.2 Challenges facing the DNS and Different types of attacks on DNS

Pharming attacks focus on changing the Domain Name Sever (DNS) entries on the client side or at the server side. Moving forward witht this project one must understand how the DNS works and how different types of attack can be launched at it to succesfully carry out pharming.

2.2.1 The DNS (Domain Name System)

Domain Name System (DNS) is a hierarchical, distributed naming system that provides a critical Internet service of mapping between two principal namespaces on the Internet: domain name hierarchy and Internet Protocol (IP) address space. By translating the human-friendly (i.e. easy for human to remember) domain names into IP addresses, DNS makes it possible to assign domain names to a group of Internet resources in a meaningful way and independent of entities' physical location(s). This naming mechanism keeps the names of the Internet resources remain consistent even if there are changes in the IP addresses of underlying networks. (Linh Vu Hong 2012). This is advantageous for the users, machines, and services because they can cite Internet resources in the meaningful way, but without having to worry how these resources are actually located. The domain name space is structured in hierarchical manner as a tree. Each domain name consists of multiple domain name labels separated by a ".". A domain name identifies a path from the root node of the DNS hierarchy, denoted by the rightmost ".", to a node representing the domain name (Linh Vu Hong 2012). This node contains a set of resource information associated with the domain name in the form of a collection of resource records (RRs). For example a domain name F.D.B.A. is a path from root node to node F. This node in the DNS tree contains information about the domain name F.D.B.A. The depth of the node in a DNS tree is called the domain level, for example, A. is a Top Level Domain Name (TLD), B.A. is second level domain, and so on.

The DNS system is split into multiple zones by partitioning the domain name space between sibling nodes. Each zone is responsible for a group of nodes, hence their corresponding domain names and associated information for that zone is authoritative. The zone identified by the domain name of the node closet to the root node. Each zone has one or more authoritative name server(s) where RRs of its domain names are stored. These authoritative name servers have complete knowledge of the domain names within their zone and serve this information when requested. The authoritative name servers can further delegate their over part of the zone to other authoritative name servers. This hierarchical model makes DNS the largest distributed system on the Internet. This model and DNS's caching mechanism provides fault-tolerance ability for the DNS system. A client can resolve a domain name www.example.com using a stub resolver that is built-in to all systems that have an Internet connection. This stub resolver sends a DNS query request to a recursive DNS resolver (RDNS). If RDNS has information about the queried domain name cached, then this information will be returned to the stub resolver. Otherwise, the RDNS starts by querying the root name server, then the root name server redirect the RDNS to the name server that is authoritative for the TLD "com.". This process continues until the RDNS reaches a name server that is authoritative for www.example.com. It will query this name server for relevant RRs and return a response to the stub resolver client. The RDNS also caches these RRs for the domain name www.example.com for a certain period of time so that it can immediately respond if there is other request for these RRs for this domain name. The period of time that of the RRs are cached in RDNS depends on the Time To Live (TTL) contained in the information returned for RRs associated with the domain name www.example.com from the authoritative name server. The entire Internet is divided into domains, i.e., name groups that logically belong together. The domains specify whether the names belong to a particular company, country, and so forth. It is possible to create subgroups within a domain that are called subdomains. For example, it is possible to create department subdomains for a company domain. The domain name reflects a host's membership in a group and subgroup. Each group has a name affiliated with it. The domain name of a host is composed from the individual group names. For example, the host named bob.company.com consists of a host named bob inside a subdomain called company , which is a subdomain of the domain com . The domain name consists of strings separated by dots. The name is processed from left to right.

The highest competent authority is the root domain expressed by a dot (.) on the very right (this dot is often left out). Top Level Domains (TLD) are defined in the root domain. We have two kind of TLD, Generic Top Level Domain (gTLD) and Country Code Top Level Domain (ccTLD). Well known gTLDs are edu , com , net , and mil which are used mostly in the USA. According to ISO 3166, we also have two letter ccTLD for individual countries. For example, the us domain is affiliated with USA. However ccTLD are used mostly outside the USA. A detailed list of affiliated ccTLD and their details are listed in Appendix A. The TLD domains are divided into subdomains for particular organizations, for example, coca- Generally, a company subdomain can be divided into lower levels of subdomains, for example, the company Company Ltd. can have its subdomain as company.com and lower levels like bill.company.com for its billing department, sec.company.com for its security department, and head.company.com for its headquarters.

2.2.2 The DNS protocol

A DNS server is used to translate human readable domain name to the corresponding IP address. A sequence of steps is taken to complete the address resolution which are listed below:

1. A client asks its local DNS server for an address resolution.
2. The local DNS server asks the root DNS server for the address resolution.
3. The root DNS server responds with a referral to the top level DNS server.
4. The local DNS server asks the top level DNS server for the address resolution.
5. The top level DNS server will respond with a referral to the second level DNS server.
6. The local DNS server asks the second level DNS server for the address resolution.
7. If this second level DNS server is the authoritative DNS server for the queried address, then it will respond with the IP address of the host or an error. Otherwise it will respond with the address to the third level DNS server.
8. The local DNS server responds to the client with the answer to its query.

In total there can be 127 levels of DNS servers, so a search can continue for several more requests. The local DNS server will store the results of this query in its cache so that if the address is asked again it can give a faster response. The DNS server described above is a recursive DNS server, for it recursively queries DNS servers until it finds the authoritative DNS in which the search host should reside and then it responds. An alternative to recursive querying is iterative querying. With iterative querying the client will do the entire DNS address resolution, instead of the local DNS server.

2.2.3 How the Domain Name System (DNS) Works

Each top-level domain (TLD) is managed by an organization called a domain name registry, which is appointed by the Internet Corporation for Assigned Names and Numbers (ICANN). The most popular TLDs are managed by large organizations such as Verisign (.com and .net) or Public Interest Registry (.org). National domains like .io or .com.au are managed by organizations in their respective countries.

One important thing to understand is registries do not always manage domain name registration. Companies that handle domain registration are called domain name registrars (versus domain name registries) and are usually accredited by registries. Accredited registrars may then subcontract to non-accredited registrars, increasing third-party risks and fourth-party risks and lengthening the time to resolve potential domain name disputes. This is because each registrar has its own rules and requirements for proving domain ownership and approving domain transfers. That said, most TLDS allow anyone to register the domain on one registrar and transfer control of the domain to another registrar (such as from Namecheap to Google domains) for any reason, such as better pricing, better security measures or a better customer experience

2.2.4 Attacks On The Domain Name System

There are multiple ways to carry out attacks on the Domain Name System. Some of the most effective methods are listed and explained below:

a) DNS cache poisoning

The absence of any signature validation on DNS entries makes it possible for an attacker to inject false resolves for a domain or host. Cache poisoning can be done on a DNS server with recursive querying enabled or an end host. The attack is initiated at the moment that a host asks for the address resolve. The cache poisoner will try to react to the query faster than the DNS server that is being queried, because then its answer to the query will be accepted. For this to work the attacker needs to know which DNS server is the Authoritative DNS server for this address resolution and then needs to spoof its source address to this DNS servers address. One approach to make sure that the Authoritative DNS server does not respond in time is to slow down the Authoritative DNS server by performing a DoS attack on it. This type of attack is possible because the only protection against this is the query Id (QID) field in a DNS packet. DNS cache poisoning can be carried out by using malicious responses or taking DNS software vulnerability to "poison" the cache that stores queries made by users in a certain amount of time in order to speed up the user response time for frequently used domains in order to enhance the user experience. After the cache being "poisoned", when the user makes queries at the DNS, the user will be redirected to the fake website where they are asked to update their personal information. When a client waits for a DNS response, it will only accept the information returned if it includes the client’s correct source port and address in addition to the correct DNS transaction ID.

b) DNS Domain Hijacking

Domain hijacking is performed by skipping the confirmation of the old domain registrar and the domain owner where the change of domain registrar can only be made with the confirmation of three parties, the domain owner, old registrar and new registrar. Domain hijacking is the act of changing the registration of a domain name without the permission of the original owner, or by abuse of privileges on domain hosting and domain registrar systems.

c) DNS Server Hijacking

Pharming attacks can be performed through DNS server hijacking. To hijack a DNS server, the attacker will first target the DNS server on the LAN or DNS server hosted by the ISP to change the IP address of an authentic website's domain name to the IP address of the fake website. When the user tries to visit the authentic website, queries will be made on the DNS server for the IP address of the domain name. Since the IP address of the domain name has been changed, it will redirect the user to the fraudulent website. When the user is being redirected to the fraudulent website, they will perform the activities that they wish to perform at the website because the address displayed in the address bar remains the same as the authentic website's address and they think that they are accessing the authentic website. Through the activities that are performed by the user, the attacker will be able to obtain the information from the victim without the victim knowledge.

2.3 Data Mining

It is considered as one of the applications of supervised machine learning, and it plays an important role in the process of retrieving the lost information. Data mining involves the use of sophisticated data analysis tools to discover previously unknown, valid patterns and relationships in large data set. These tools can include statistical models, mathematical algorithm and machine learning methods. Consequently, data mining consists of more than collection and managing data, it also includes analysis and prediction. Classification technique is capable of processing a wider variety of data than regression and is growing in popularity. There are several applications for Machine Learning (ML), the most significant of which is data mining. People are often prone to making mistakes during analyses or, possibly, when trying to establish relationships between multiple features. This makes it difficult for them to find solutions to certain problems. Machine learning can often be successfully applied to these problems, improving the efficiency of systems and the designs of machines. Many terms carry a similar or slightly different meaning to data mining, such as knowledge mining from data, knowledge extraction, data/pattern analysis, data archaeology, and data dredging. Data mining functionalities are used to specify the type of patterns to be found in the data mining tasks. In general data mining tasks can be classified into two main categories: descriptive and predictive. Data mining is the most important machine learning application (Srinivas *et al.,* 2010). It can be defined as extracting information from a large number of data sets (Abdullah, 2016). In other words, data mining is simply defined as " Knowledge mining in data". The main focus in data mining is on pattern recognition (Jothi *et al.,* 2015). Data mining is used in many fields of information technology, science, medicine, biology, education, and human resources to obtain rules and predict certain outcomes (Nagi *et al.,* 2012). The insights gained from data mining can be applied to fraud detection, customer loyalty and market analysis, scientific research, and production control (Priyanka *et al.,* 2012). Data mining includes various techniques such as anomaly detection, classification, regression, clustering, time series analysis, association rules, and summary (Ameta *et al.,* 2017). Descriptive mining tasks characterize the general properties of the data. Predictive mining tasks perform inferences on the current data in order to make predictions. Most of data mining tasks can be one or combination of the following:

a) Classification: used for predictive mining tasks. This method is intended for learning different functions that map each item of the selected data into one of a predefined set of classes. Given the set of predefined classes, a number of attributes, and a “learning (or training) set,” the classification methods can automatically predict the class of other unclassified data of the learning set.

b) Prediction: used for predictive mining tasks. Analysis is related to regression techniques. The key idea of prediction analysis is to discover the relationship between the dependent and independent variables. For example, by using historical data from both sales and profit, either linear or nonlinear regression techniques can produce a fitted regression curve that can be used for profit prediction in the future.

c) Association Rules: used for descriptive mining tasks. It aims to find out the relationship among valuables in database, and produce a set of rules describing the set of features that are strongly related to each other’s, so that the relationship of a particular item in a data transaction on other items in the same transaction is used to predict patterns.

d) Clustering: used for descriptive mining tasks. It is unsupervised, and does not require a learning set. It shares a common methodological ground with Classification. It ungroupes data and uses automatic techniques to put this data into groups. In other words, finds groups of data points (clusters) so that data points that belong to one cluster are more similar to each other than to data points belonging to different cluster.

e) Outlier Analysis: used for predictive mining tasks. Discovers all data points that are different from the rest of data. Such points are known as exceptions or surprises. While outliers can be considered noise and discarded in some applications, they can reveal important knowledge in other domains, and thus can be very significant and their analysis valuable. So it is very important to identify the outliers.

2.4 Classification

Classification, a data mining technique, is the process of classifying and predicting the value of a class attribute based on its predictor value (Romero *et al.,* 2008). A predictor is an attribute used to predict a new record, e.g. Age, gender, harassment, etc. There are two main categories of classification models used for prediction; descriptive and predictive classification models. Descriptive models find relationships or models in the data and even examine the properties of the data being examined. Examples of techniques that support this include summarization, clustering, association rules, etc. While, predictive model conducts prediction of unknown data values by using supervised learning function applied on known values (Jothi *et al.,* 2015). The known data is historical. Example of such techniques includes Time series analysis, Prediction, Classification, Regression, etc. The interest of this study lies in the predictive classification model, where the model is based on the characteristics of historical data and is used to predict future trends (Al-radaideh and Nagi, 2012). Many classification algorithms are used to classify categorical data, e.g. Decision Tree, K-Nearest Neighbor, Naïve Bayes, SVM, J48, Random Forest, Logistic Regression, and many more.

2.5 Machine Learning

Machine learning is a study in the field of artificial intelligence that uses various probabilistic, statistical, and optimization techniques for training computer systems to explore and "learn" different and rigid models in complex, large, and noisy data (Vihinen, 2012). It is about learning how to better face the future based on past experiences (Cruz and Wishart, 2016). For example, learns to act intelligently or to predict harassment accurately based on a series of observations. The goal is to develop learning algorithms that can learn automatically without human assistance or intervention. Machine learning can be used when people are prone to errors in analysis or maybe trying to make connections between multiple functions. It is used to improve the efficiency of system and engine designs (Archana and Elangovan, 2014). Machine learning provides an alternative solution to pharming attacks by using classification techniques applied to past real-world data to predict current attacks.

2.6 Predictive Model

Prediction is the process of studying the present and past states of the attribute to predict its future state (Medhekar *et al.,* 2013). Prediction consists of evaluating classification, pattern matching, trends, and relation through analyzing data of past instances or events to forecast future events using a developed predictive model (Srinivas *et al.,* 2010). Data prediction involves two-step processes: first, the training of the model using a classifier to predict the class label (predicted attribute) of a test data. Then secondly, evaluating performance accuracy of a predictor by calculating the error based on the differences between actual and predicted values for each tuple in test data (Baby and Priyanka, 2012). The predictive model of data mining tasks includes regression, classification, prediction, and time series analysis (Tribhuvan *et al.,* 2015).

2.7 Related Works

Several works have been done on detecting pharming attacks using different types of approach such as IP address detection and web page content analysis. These approach were tested based on their accuracy.

In 2017, S. Stamm *et al.* described the concept of the attack: "Drive by Pharming". In this attack an attacker sets up a web page so that when the victim user views, in the case of javascript enabled browser, the attacker changes the DNS server settings on victim's home broadband router .The authors explain and describe scenarios for that type of attack, and they talk about new attacks like, pharming, growing zombies, and viral spread. So, researchers recommended for any user to change the default password of his own router, and to disable javascript in the browser to avoid these types of attacks. Authors do not introduce an effective solution to protect users from phishing and pharming attacks, they only explain some new attacks, and introduce some advice for users to protect themselves from those attacks.

Li *et al.* (2013) proposed a client pharming attack hybrid detection model which is based on web content and IP addresses. The model mainly consists of IP address filter, web feature extraction, output module and so on. The model analyze the website content extracted from the two DNS setup and compare them using data selection, webpage content recognition and hidden URLs on the websites. An IP address filter and a web feature extraction was developed to extract content from websites and analyze them based of specific reqirements and did not take into consideration that an attacker can also replicate these features into it own pharming website.

In 2010, B. Aslam *et al.* presented a solution to protect users from phishing and pharming attacks. This solution is based on a hashed password which is the hash value of the user-typed password and the authentication server's IP address. The solution rests on the fact that the server connected by a client using TCP connection can't fraud about its IP address. If a user goes to a malicious server (by a Phishing or a Pharming attack), the password obtained by the malicious server will be the hashed password (tied to the malicious server's IP address) and will not be usable by the attacker at the real server, thus defeating Phishing or Pharming attack. Authors used PwdIP-Hash for specific browser, but they do not consider other popular web browsers such as Firefox, Chrome, etc.

In 2011, S. Prevost *et al.* proposed a dual approach to detect pharming attack at the client side. This approach combines the IP address check and the webpage content analysis, using information provided by multiple DNS servers (Local DNS server and Alternate DNS server like GoogleDNS or OpenDNS). The approach is integrated within web browser of the user. Authors validated their proposed approach by conducting a first set of experimentations from continents (North America, South America, Europe, Africa, Asia and Australia). The same third party DNS server was asked to resolve many homepages of legitimate domain names in order to check the IP addresses changes. Authors used webpage content analysis as a second approach of their solution, but the main drawbacks of this solution are to maintain an up-to-date database as well as to protect it against any compromising attacks.

Alfayoumi (2015) wrote a PHP code to develop a plugin that protect user at client-side from pharming attacks by comparing IP addresses, using information provided by local DNS server and a list of IP's provided by the most trusted domain DNS servers – the domain's Authenticated Name Servers. A result of 300 famous and most popular websites in a number of areas such as online banks, search engines, mail bulk services, hosting and E-Commerce Companies with different languages and TLD's in the domain name was examine using two ISPs as real environment. When the location changes the IP address also changes which brings the idea of a reverse lookup to do a trace on the IP address verifying who the address belongs to. The drawback remains cross platform compatibility of the plugin with different web browsers and regular update of the plugins.

Laurent *et al.* (2015) have put forward a two step process to detect pharming by developing a framework that alert the end-user incase of pharming attacks at the client-side using the IP address and webpage content comparison by differentiating legitimate from fraudulent login websites, based on a dual-step analysis performed using multiple DNS servers information. IP address is checked and second, HTML source code analysis. The framework is not easily operable and it is not available for all platforms.

In 2011, S. Prevost *et al.* defined an advanced approach to strengthen their first approach in to alert the end-user in case of pharming attacks at the client- side. They had a success rate over 95%, and they validated their solution that helps users to differentiate legitimate from fraudulent login websites, based on a dual-step analysis (IP address check and webpage content comparison) performed using multiple DNS servers information. Authors used the same webpage content analysis in their first paper in as a second approach of their solution, but still the main drawbacks of this solution are to maintain an up-to-date database as well as to protect it against any compromising attacks.

Gastellier-Prevost *et al.* (2016) proposed a dual approach to provide an anti-pharming protection integrated into the client's browser. The approach combines both an IP address check as well as a webpage content analysis, using the information provided by multiple DNS servers. A test was conducted using DNS from 6 continents (North America, South America, Europe, Africa, Asia and Australia). The same third party DNS server was asked to resolve many homepages of legitimate domain names, in order to check the IP addresses changes. For most of the domain names tested, the third-party DNS server responses greatly vary according to the location from which the DNS query was launched. DNS look up takes time when there is no good network reception which later leads to failure in checking for pharming attacks.

Mahmood in (2016) proposed a new method to detect, alert and protect the user from Internationalized Domain Names (IDN) and Uniform Resource Locator (URL) spoofing, phishing, pharming and man-in-the-middle attacks using a browser plugin which enables the user to validate the website and also provides visual feedback. In case of pharming, where Domain Name System (DNS) is hijacked, the plugin automatically notifies the user of possible attack, posts the attack information to the server and redirects the user to legitimate website. SSL/TLS certificates are also automatically checked, verified and validated by the plugin in order to check for man-in-the-middle and pharming attacks. Compatibility of the plugin with different types of browser becomes the major problem to this approach.

Ramzan *et al.* (2016) describe a web-based automated method to detect how routers are being attacked using JavaScript-generated host scans and HTTP requests by visiting malicious websites, and how the routers DNS are being spoofed by executing the javascripts code without the knowledege of the victim which leads to pharming. Result obained was a DNS configuration change by compromising a D-Link DI- 524 by accessing a website with simple Java and JavaScript to detect the internal network and found the router's IP address and router model. A request to change the DNS server settings can be sent to the router and the attacker will have total control over the DNS server.

In 2016, O. Mahmood, proposed a method to identify, warn and protect users from the attacks: phishing, pharming, and man-in-the-middle attacks, by authenticating the site before the user actually shares personal information. The presented method is based on the use of a browser plug in which enables the user to validate the website and provides visual feedback. The plugin also automatically visually notifies the user of possible attack in case of pharming, where Domain Name System (DNS) is infected. The presented method is divided into three processes: IP address check, SSL certificate validation and verification, and Friend of a Friend evaluation process. Author depend on validating and verifing the SSL certificate by using locally stored information, but that local information may be infected or corrupted on any type of attack.

Karlof *et al.* (2017) developed a two locked same-origin policies for web browsers. The locked same-origin policy enforces access using servers' X.509 certificates and public keys. It helps two existing web authentication mechanisms, client-side SSL and SSL-only cookies to resist both pharming and stronger active attacks. An implemented proof of concept of dynamic pharming attack was tested against two browsers: Firefox 2.0 running on Debian GNU/Linux 3.1 and Microsoft Internet Explorer 7.0 running on Windows Server 2003 SP2 using a pair of Apache SSL web servers (i.e, a pharmer and a target) and round robin DNS. An attacker can deliver a trojan document which can cause the browser to renew its DNS entry for the target domain and connect to the legitimate server, after which the adversary then hijacks the session with the malicious Javascript in the trojan document.

S. Gastellier-prevost *et al.*, proposed client-side framework to authenticate website. It consists of checking IP address and webpage content analysis, using data provided by different DNS servers. Approach is integrated within web browser of user so that notification arises in case of any doubtful website

K. Gajera has used ANN (Artificial Neural Networks) for pharming detection. Features were selected on the basis of URL parameters and fed into neural networks for training. To identify between legitimate and malicious website. For identification of malicious websites, IP addresses were queried to local and global DNS, if they both return same result then it is safe otherwise pharming attack was presented.

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.1 The Design of Support Vector Machine Classifiers

To detect pharming attacks on websites, Support vector machine classifires will be used to analayse datasets from the front-end like website content, hiddens links, website images, text characteristics, classification of images and to the back-end like SSL certificates, HTML Tags, css structures, Tokens etc. Below is the proposed architecture for the system.

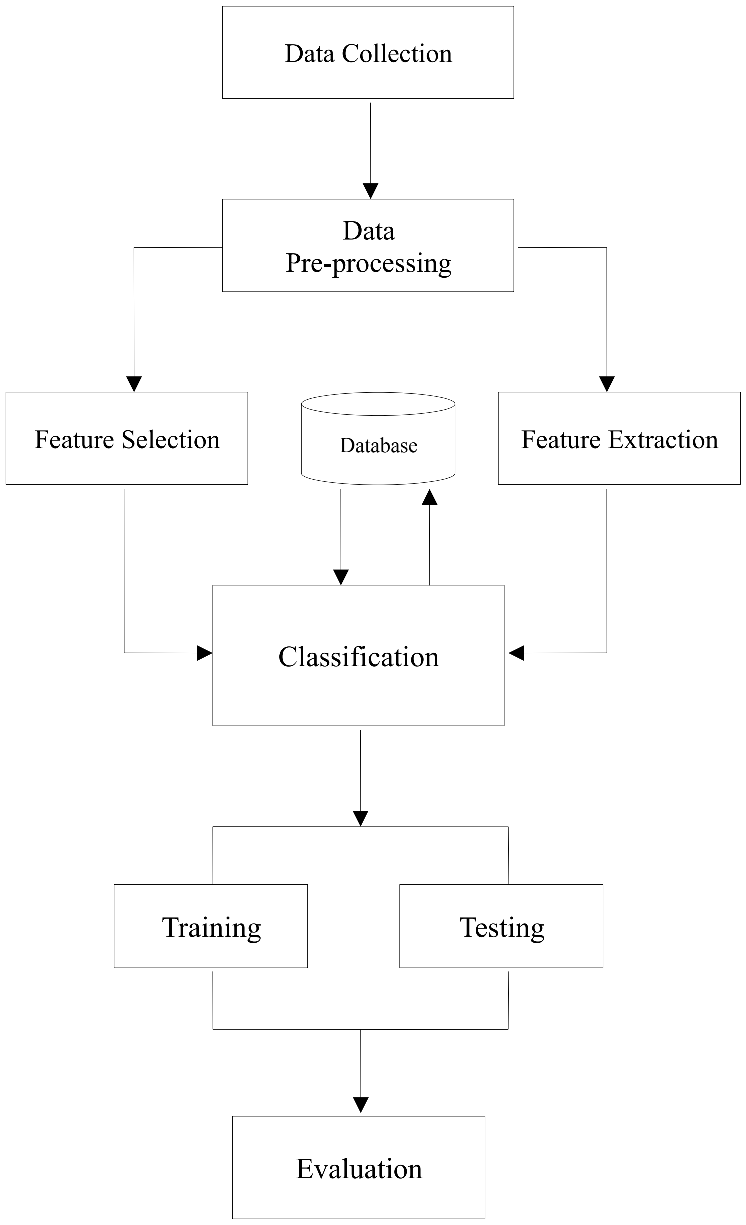


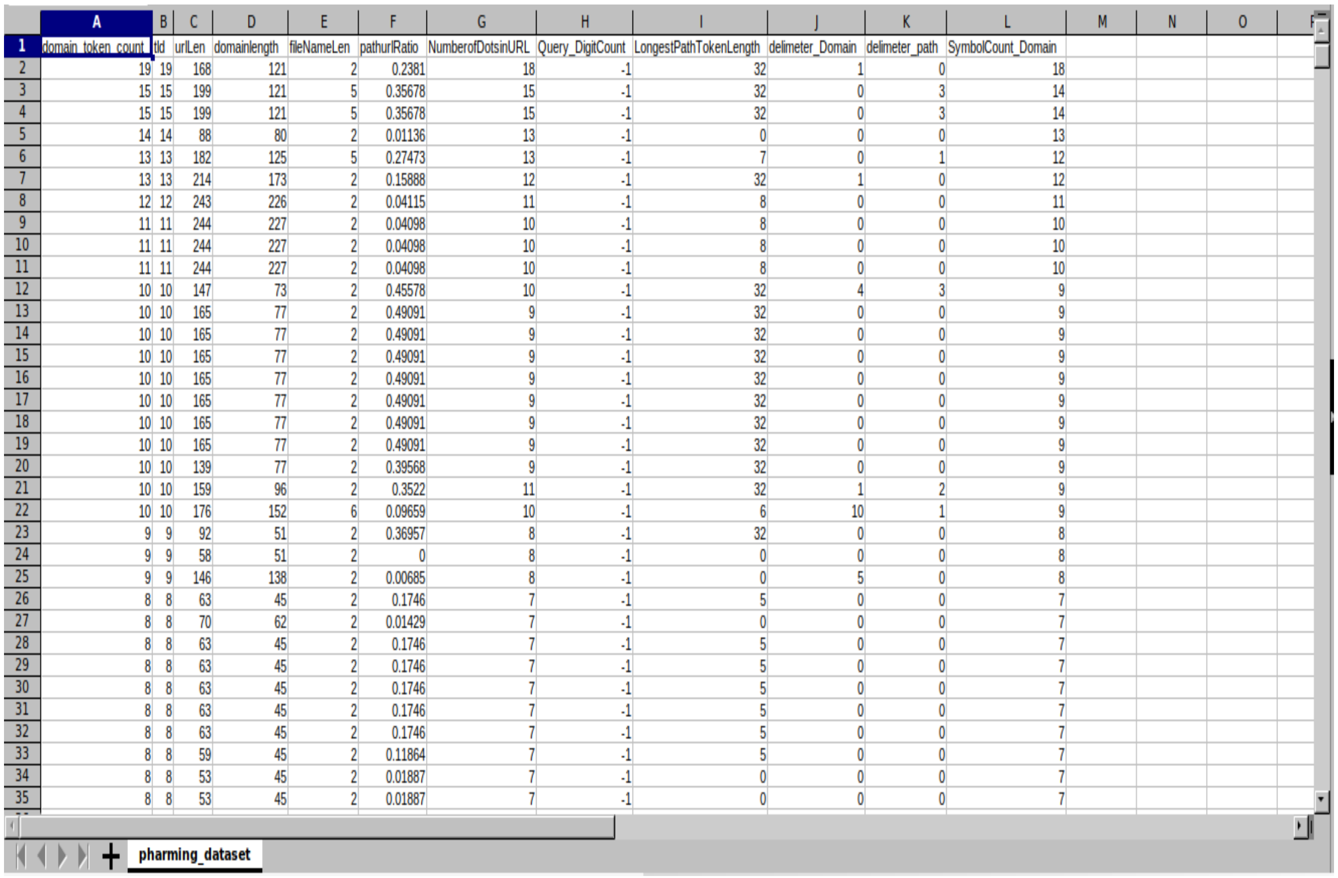
Figure 3.1: Proposed System Architecture

3.2 Data Collection

The dataset of pharming and legitimate websites were collected from the UCI Machine Learning Repository, which is freely available for use. This dataset consists of pharming websites and legitimate websites which were used to extract several website features. Here are some examples of features that can be used to collect data for an SVM classifier:

1. Categorical features: Categorical features can be used to represent the data in the form of categories. Examples of categorical features include the presence of specific keywords in a URL, the top-level domain (TLD) of a URL, etc.
2. Numerical features: Numerical features can be used to represent the data in numerical form. Examples of numerical features include the length of a URL, the number of special characters in a URL, the number of dots in the domain name, etc.
3. Textual features: Textual features can be used to represent the data in the form of text. Examples of textual features include the characters in a URL, the words in a URL, etc.
4. Syntax features: Syntax features can be used to represent the data in the form of syntax. Examples of syntax features include the structure of the domain name, the presence of subdomains, etc.

Selection of appropriate dataset is a prime task because the whole performance of the whole model is dependent on the precision of dataset. Data is needed to be accurate in order to get reliable results. Raw data need to be properly filtered for the unbiased results. To get a real time traffic of websites tend to expensive task. There are some authenticated sources on web which provides reliable datasets.



	A	B	C	D	E	F	G	H	I	J	K	L
1	domain_token_count	tld	urlLen	domainlength	fileNameLen	pathurlRatio	NumberofDotsinURL	Query_DigitCount	LongestPathTokenLength	delimeter_Domain	delimeter_path	SymbolCount_Domain
2	19	19	168	121	2	0.2381	18	-1	32	1	0	18
3	15	15	199	121	5	0.35678	15	-1	32	0	3	14
4	15	15	199	121	5	0.35678	15	-1	32	0	3	14
5	14	14	88	80	2	0.01136	13	-1	0	0	0	13
6	13	13	182	125	5	0.27473	13	-1	7	0	1	12
7	13	13	214	173	2	0.15888	12	-1	32	1	0	12
8	12	12	243	226	2	0.04115	11	-1	8	0	0	11
9	11	11	244	227	2	0.04098	10	-1	8	0	0	10
10	11	11	244	227	2	0.04098	10	-1	8	0	0	10
11	11	11	244	227	2	0.04098	10	-1	8	0	0	10
12	10	10	147	73	2	0.45578	10	-1	32	4	3	9
13	10	10	165	77	2	0.49091	9	-1	32	0	0	9
14	10	10	165	77	2	0.49091	9	-1	32	0	0	9
15	10	10	165	77	2	0.49091	9	-1	32	0	0	9
16	10	10	165	77	2	0.49091	9	-1	32	0	0	9
17	10	10	165	77	2	0.49091	9	-1	32	0	0	9
18	10	10	165	77	2	0.49091	9	-1	32	0	0	9
19	10	10	165	77	2	0.49091	9	-1	32	0	0	9
20	10	10	139	77	2	0.39568	9	-1	32	0	0	9
21	10	10	159	96	2	0.3522	11	-1	32	1	2	9
22	10	10	176	152	6	0.09659	10	-1	6	10	1	9
23	9	9	92	51	2	0.36957	8	-1	32	0	0	8
24	9	9	58	51	2	0	8	-1	0	0	0	8
25	9	9	146	138	2	0.00685	8	-1	0	5	0	8
26	8	8	63	45	2	0.1746	7	-1	5	0	0	7
27	8	8	70	62	2	0.01429	7	-1	0	0	0	7
28	8	8	63	45	2	0.1746	7	-1	5	0	0	7
29	8	8	63	45	2	0.1746	7	-1	5	0	0	7
30	8	8	63	45	2	0.1746	7	-1	5	0	0	7
31	8	8	63	45	2	0.1746	7	-1	5	0	0	7
32	8	8	63	45	2	0.1746	7	-1	5	0	0	7
33	8	8	59	45	2	0.11864	7	-1	5	0	0	7
34	8	8	53	45	2	0.01887	7	-1	0	0	0	7
35	8	8	53	45	2	0.01887	7	-1	0	0	0	7

pharming_dataset

Figure 3.2: Dataset in CSV format

3.3 Data Pre-processing

Data pre-processing is a data mining approach that entails converting raw data into a format that can be understood. Inconsistencies, blunders, out-of-range respect, inconceivable data mixtures, missing qualities, or, more fundamentally, data unsuited for a data mining method will almost certainly be present in the data. Changing the impossible into the possible is accomplished via information handling, which involves modifying data to meet the data requirements of each data mining method. Non-numeric features are typically left out of the dataset because they don't play a significant role in detecting malicious activities. The dataset should be extensively examined to ensure that there are no duplicate records, half-completed queries, or unrelated data. To improve the model's accuracy, we normally remove non-numeric and unnecessary symbols. Raw data should be refined with care because it is the lifeblood of the entire operation. We refined our dataset by taking the above-mentioned factors into account. Data preparation can take a long time to complete. Data decrease assignments, which go to reducing the multifaceted design of the data, perceiving or expelling unessential and uproarious components from the data through element assurance, occurrence choice, or discretization frames, data preprocessing solidifies data planning, exacerbated by mix, cleaning, institutionalization, and change of data, data decrease assignments, which go to reducing the multifaceted design of the data, perceiving or expelling unessential and uproarious components from the data through after a tried and true association of information preparation forms, the desired output is a final informative index that can be thought of correctly and support for advanced data mining techniques (Nitta *et al.,* 2018).

3.4 Feature Extraction

Several features can be extracted from a website to distinguish pharming websites from legitimate ones. The extracted features is crucial for the success of the pharming website detection mechanisms. This approach starts with empty dataset and recursively adds one feature at a time till best accuracy is achieved. When there is no more improvement in the accuracy after adding next feature, at that time process gets stopped. Website features can be used to determine if a website can be trusted or not. Keeping in mind the pharming attack, additional significant criteria were chosen for determining safe and pharming websites. The folowing basic features relevant to this attack from websites was selected

(a) The frequency of keywords.

(b) The total number of words displayed on the web page.

(c) Fake SSL/TLS protocol - Pharmers generally use fraudulent HTTPs protocol to trick users.

(d) Unusual redirects - If website is taking more than usual time to redirect or taking three or four times to redirect you to the desired website then it can be the case of pharming.

(e) IP Address - Usage of IP address in the URL itself is a questionable thing. Muggers use IP addresses of non- working websites. Websites starting with IP addresses are not trustworthy.

(f) Suspicious URL - If URL of the website is not matching with records in WHOIS database, then website is considered unsafe.

(g) Pop-up Window - Any website that prompts you to enter credentials in a pop-up window is a suspect website.

(h) SSL having @ symbol - Genuine SSL doesn’t contain @ symbol in its domain name

(i) Website domain containing hash symbol- Websites having SSL security mechanism doesn’t contain # symbol in its domain name.

(j) Certificate Validity - The certificates used by websites should be valid and not expired.

(k) DNS Record - If DNS records aren't available, there's something suspicious going on.

(l) Uncommon Domain Name - If the rerouted website has an unusual domain name such as fut4.3du.ng instead of futa.edu.ng, you're engaging in pharming.

3.5 Feature selection using SVM-RFE

SVM—RFE stands for Support Vector Machine- Recursive Feature Elimination. It is an embedded approach that recursively removes unimportant features rather than using the weights for ranking criterion. It helps to provide better performance by selecting best features subset. It takes training instances and their class labels as an input to the algorithm and uses ranking criterion based on the weight vector of SVM. From the weight vector of SVM, the ranking of each feature is identified and then features are selected by eliminating those features which have lowest ranking. In bioinformatics, it is a powerful feature selection algorithm to avoid overfitting in case of high number of features. A feature selection process can be used to remove terms in the training dataset that are statistically uncorrelated with the class labels, thus improving both efficiency and accuracy. Pal and Maiti (2010) provided a supervised dimensionality reduction method. The feature selection problem has been modeled as a mixed 0-1 integer program. In addition to reducing classification computational time, it can improve the classification accuracy rate. In recent years, many scholars improved the classification effect in medical diagnosis by taking advantage of this method.

The main purpose of SVM-RFE is to compute the ranking weights for all features and sort the features according to weight vectors as the classification basis. SVM-RFE is an iteration process of the backward removal of features. Its steps for feature set selection are shown as follows.

1. Use the current dataset to train the classifier.

2. Compute the ranking weights for all features.

3. Delete the feature with the smallest weight.

Implement the iteration process until there is only one feature remaining in the dataset; the implementation result provides a list of features in the order of weight. The algorithm will remove the feature with smallest ranking weight, while retaining the feature variables of significant impact. Finally, the feature variables will be listed in the descending order of explanatory difference degree. SVM-RFE's selection of feature sets can be mainly divided into three steps, namely, the input of the datasets to be classified, calculation of weight of each feature, and the deletion of the feature of minimum weight to obtain the ranking of features. The computational step is shown as follows.

1. Input

   a) Training sample: $X_0 = [x_1,x_2,\ldots,x_m]^T$.

   b) Category: $y = [y_1,y_2,\ldots,y_m]^T$.

   c) The current feature set: $s = [1,2,\ldots, n]$.

   d) Feature sorted list: $r = []$.

2. Feature Sorting

   a) Repeat the following process until $s = []$.

b) To obtain the new training sample matrix according to the remaining features: $X = X_0(:, s)$.

c) Training classifier: $\alpha = \text{SVM-train}(X, y)$.

d) Calculation of weight: $w = \sum_k \alpha_k y_k x_k$.

e) Calculation of sorting standards: $c_i = (w_i)^2$.

f) Finding the features of the minimum weight: $f = \arg\min(c)$.

g) Updating feature sorted list: $r = [s(f), r]$.

h) Removing the features with minimum weight: $s = s(1 : -1, f + 1 : \text{length}(s))$.

3. Output: Feature Sorted List r. In each loop, the feature with minimum $(w_i)^2$ will be removed. The SVM then retrains the remaining features to obtain the new feature sorting. SVM-RFE repeatedly implements the process until obtaining a feature sorted list. Through training SVM using the feature subsets of the sorted list and evaluating the subsets using the SVM prediction accuracy, we can obtain the optimum feature subsets.

$$\min_{w \in \mathbb{R}^p, \beta \in \mathbb{R}} |w|_1 + C \sum_{i=1}^{n} \max\left(0, 1 - y_i(x_i^T w + \beta)\right)$$

$$\begin{aligned} &\min_{z \in \mathbb{R}^p, \xi \in \mathbb{R}^n} \textstyle\sum_{i=1}^{p} z_i + C \sum_{i=1}^{n} \xi_i \\ &\text{subject to} \begin{cases} \forall i \leq p,\ z_i \geq w_i,\ z_i \geq -w_i \\ \forall i \leq n,\ \xi_i \geq 0,\ \xi_i \geq 1 - y_i(x_i^T w + \beta) \end{cases} \end{aligned}$$

3.6 Algorithm of the Model

```
Begin

If url 2 eF1(url) // eF1(url) means URL features
Compute total value of eF1(url)
Generate feature vector for eF1(url)
Use invocation connector to communicate eF1(url) to the next atomic subcomponent
Return False
Else
If url 2 eF2(url) // eF2(url) means webpage properties
Compute total value of eF2(url)
Generate feature vector for eF2(url)
Use invocation connector to communicate resultant feature vector (eF1(url) + eF2(url)) to next atomic
subcomponent
Return False
Else
If url 2 eF3(url) // eF3(url) means webpage behaviour
Compute total value of eF3(url)
Generate feature vector for eF3(url)
Use invocation connector to communicate resultant
feature vector (eF1(url) + eF2(url) + eF3(url)) to composite component
Return False
Endif
Normalize and vectorize resultant feature vector
Train the classifier with the resultant feature vector
Generate predictive model for the classifier
Test the predictive model
End
```

3.7 Flowchart of the Model

A flowchart is the diagrammatic representation of an algorithm. Below is the flowchart of the proposed model.

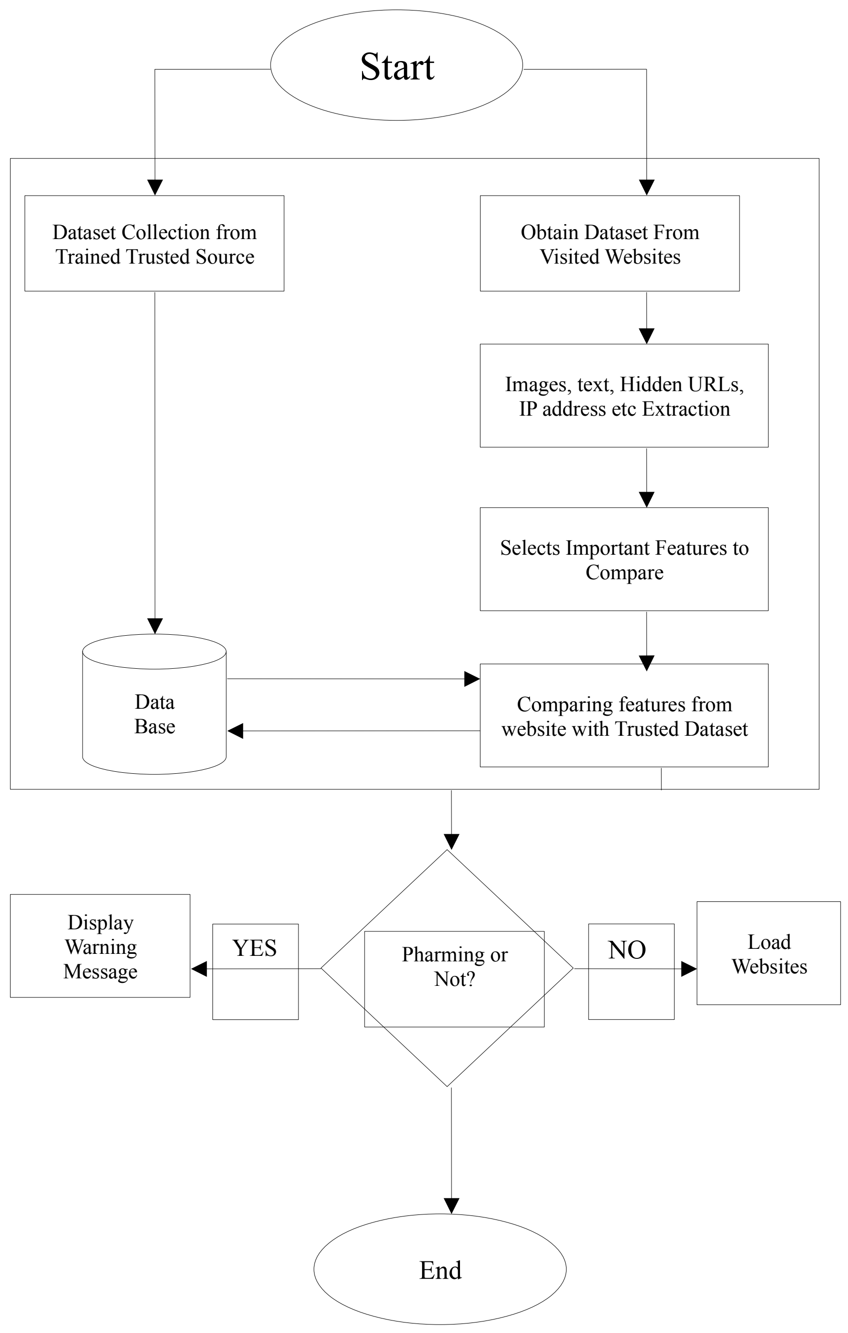

Start
Dataset Collection from Trained Trusted Source
Obtain Dataset From Visited Websites
Images, text, Hidden URLs, IP address etc Extraction
Selects Important Features to Compare
Data Base
Comparing features from website with Trusted Dataset
Display Warning Message
YES
Pharming or Not?
NO
Load Websites
End

Figure 3.3: Flowchart of the Proposed model

3.8 Training

At the core of the machine, the learning process is the training of the model. The bulk of the "learning" is done at this stage. 80 percent of the information was assigned for training to educate our model to assess if pharming occurs or not. It is common practice in machine learning to divide a dataset into two parts: a training set and a test set. The training set is used to train the model and the test set is used to evaluate the performance of the trained model. The typical split ratio is 80/20, where 80% of the data is used for training, and the remaining 20% is used for testing. The model needs to learn the patterns and relationships in the data by using a large portion of the data for training. The more data the model has to learn from, the better it will perform on unseen data. By reserving a portion of the data for testing, we can evaluate the model's performance on unseen data and get a more accurate estimate of its generalization ability. This helps to prevent overfitting, which occurs when a model is too complex and performs well on the training data.

3.9 Testing

Testing is referred to as the procedure when the performance of a fully trained model is assessed on a testing set. That is why 20 percent of the data set produced for assessment is utilized to verify the model's competency. The testing set consisting of a set of testing samples should be segregated from the both training and validation sets, but it should follow the same probability\distribution as the training set. The test set is used to measure the model's generalization ability and simulates the new dataset and provides an estimate of how well the model will perform on unseen data. By testing the model on unseen data, we can get an idea of its ability to generalize to new examples and make accurate predictions on new data. This helps to avoid overfitting, which is when a model is too complex and performs well on the training data but poorly on new data.

3.10 Classification

3.10.1 Support Vector Machine Classifier

Support Vector Machine (SVM) is developed by Boser, Guyon, Vapnik in 1992. The purpose of SVM is to find an optimal hyperplane, which is a hyperplane that has a maximum margin. There will be more than one hyperplane that can divide data into two classes. SVM will choose hyperplane with maximum margin. The primal optimization problem can be written as:

$$min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \xi_i$$
$$s.t.\, y_i(w.x_i + b) \geq 1 - \xi_i, i = 1,2, \dots, N$$
$$\xi_i \geq 0, i = 1,2, \dots, N$$

where the parameter ! > 0 will control the trade-off between the amount of data in the wrong class (narrowed margin) and generalization capabilities (margin widened). C values are not obtained in the learning process but must be determined before learning. ! ! is the slack variable. The slack variable enables misclassification at some distances. The slack variable is formulated as ! ! = |! ! − ! ! ! |. The slack variable is formulated as ! ! = |! ! − ! ! ! |. If ! ! = 0 then ! ! lies in the margin and is classified correctly, and if 0 < ! ! ≤ 1then ! ! lies in the margin but is still classified in the correct class, whereas if ! ! > 1 then ! ! is classified in the wrong class.

$$min \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i \, . \, x_j) - \sum_{i=1}^{N} \alpha_i$$
$$s.t. \sum_{i=1}^{N} y_i \alpha_i = 0 \,, 0 \leq \alpha_i \leq C, i = 1,2, \dots, N$$

SVM is a valuable data categorization tool. Despite the fact that Neural Networks are thought to be easier to utilize than this, sometimes undesirable results are produced. A classification task normally entails training and testing data that includes a variety of data examples. In the training set, each instance has one goal value and multiple characteristics. SVM's purpose is to create a model that predicts the target value of data instances in the testing set given just the characteristics. Classification is used for predictive mining tasks. This method is intended for learning different functions that map each item of the selected data into one of a predefined set of classes. Given the set of predefined classes, a number of attributes, and a "learning (or training) set," the classification methods can automatically predict the class of other unclassified data of the learning set. Support Vector Machine (SVM) is one of the most well known and robust supervised machine learning techniques, which has been utilized effectively in many science and engineering applications. Support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Support Vector Machine based classification is used for the detection of malicious websites. It is a machine learning classification method which generates a hyper-plane splitting two objects. A best hyper-plane is based on how optimally it is dividing two objects along with large separation distance between two objects. In order to obtain an optimal hyper-plane, the width (w) of the margin needs to be maximized, as described in following figure.

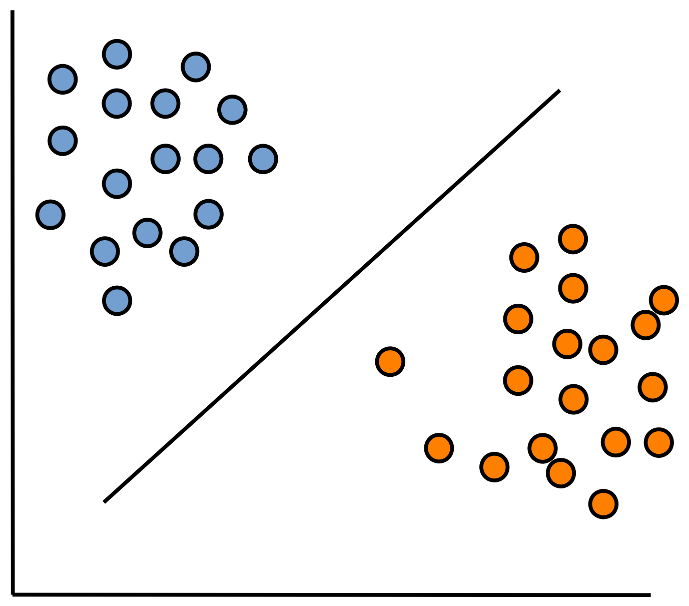


Figure 3.4: Hyper-plane separating two classes

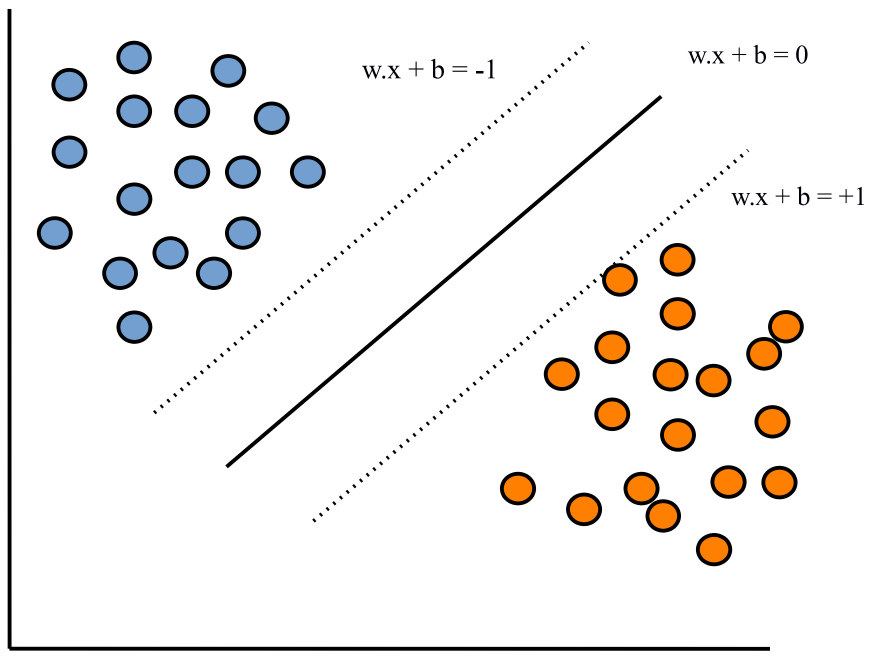


Figure 3.5: Maximization of hyper-plane’s width of margin

3.10.2 Naïve Bayes

Naïve Bayes is an approach for evaluating probabilities of individual variable value where a class is provided from training data which then allows the use of these probabilities to classify new entities which is termed in Bayesian statistics dealing with a coherent probabilistic classifier by applying Bayes‟ theorem (from Bayesian statistics) with strong (naive) independence assumptions. A naïve Bayes classifier infers that the existence (or exclusion) of a attribute of a class is unrelated to the existence (or exclusion) of any other attribute.

3.10.3 Naives bayes Classifier

In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. In the statistics and computer science literature, Naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naive Bayes is not (necessarily) a Bayesian method.

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

$$P(y|x_1, ..., x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods. Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. An analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers. After training, the word probabilities (also known as likelihood functions) are used to compute the probability that a particular set of websites in it belongs to either category. Some software implements quarantine mechanisms that define a time frame during which the user is allowed to review the software's decision. The initial training can usually be refined when wrong judgments from the software are identified (false positives or false negatives).

3.10.4 Naïve Bayes Classification Analysis (Using Pharming Datasets and Value)

The performance of the proposed system is evaluated by using four standards parameters consisting of True Positive Rate, False Positive Rate, True Negative Rate, and False Negative Rate. These are the standard performance metrics to evaluate any phishing detection system. Let P denotes the total number of pharming sites and L represents the total number of legitimate sites.

Using the following notations:

1. P.P as pharming sites classified as pharming
2. P.L as pharming classified as legitimate
3. L.L as legitimate classified as legitimate
4. L.P as legitimate classified as pharming
5. P is the aggregate pharming website
6. L is the aggregate legitimate website

To evaluate the performance of proposed methodology confusion matrix was used. It is a table which provides the performance of classifier on the basis of some parameters on test data containing 924 URLs. It is also known as — Error Matrix. It shows how your classification model gets confused while making predictions. To begin with confusion matrix, one has to follow steps given below:

1. A validation or test dataset with expected outcome results.

2. Predict each row in the test dataset.

3. The expected predictions and expected outcomes give us the number of accurate predictions for each class and the number of inaccurate predictions for each class, ordered by the class that was predicted.

4. The values are organized in matrix containing Predicted class and Actual class, allocated to following terms:

True Positive (TP), False Negative (FN), False Positive (FN), True Negative (TN)

The following confusion matrix derived by model shows values of both predicted and actual class.

Table 3.1: Confusion Matrix

Confusion Matrix		Pharming	
		Yes	No
Prediction	Classified as Pharming	TP	FN
	Classified as Legitimate	FP	TN

Following are some parameters on which we have evaluated the performance:

1. Accuracy - It is calculated as number of correct predictions divided by total number of datasets.

   Acuracy $= \frac{TP+TN}{TP+FP+TN+FN}$

2. Sensitivity - It is calculated by dividing number of correct positives with number of true positives.

   Sensitivity $= \frac{TP}{TP+FN}$

3. Precision - It is calculated by dividing number of correct positive predictions with total number of positive predictions.

   Precision $= \frac{TP}{TP+FP}$

4. Specificity- It is calculated by dividing number of correct negative predictions with total number of negatives.

   Specificity $= \frac{TN}{TN+FP}$

Thus, from the above mentioned results, we can say that the performance given by classifier is quite efficient and reliable and it is able to detect a number of pharming attacks.

3.11 Evaluation

After the model has been trained, it must be tested to determine if it can function in real-world scenarios. This places the model in a situation where it must deal with problems that aren't included in its training. The model, however, should be able to extrapolate the information and detect pharming assaults at various stages and levels as a result of its training. SVM hyper planes were utilised to distinguish legal data from pharming data, regardless of how similar or identical the two results appeared to be.

For SVM classifiers to evaluate and make decision it make use of the following:

1. Prediction is used for predictive mining tasks and Analysis is related to regression techniques. The key idea of prediction analysis is to discover the relationship between the dependent and independent variables.
2. Association Rules is used for descriptive mining tasks. It aims to find out the relationship among valuables in database, and produce a set of rules describing the set of features that are strongly related to each other's, so that the relationship of a particular item in a data transaction on other items in the same transaction is used to predict patterns.
3. Clustering is used for descriptive mining tasks. It is unsupervised, and does not require a learning set. It shares a common methodological ground with Classification. It ungroupes data and uses automatic techniques to put this data into groups. In other words, finds groups of data points (clusters) so that data points that belong to one cluster are more similar to each other than to data points belonging to different cluster.
4. Outlier Analysis is used for predictive mining tasks. It discovers all data points that are different from the rest of data. Such points are known as exceptions or surprises. While outliers can be considered noise and discarded in some applications, they can reveal important knowledge in other domains, and thus can be very significant and their analysis valuable. So it is very important to identify the outliers.

CHAPTER FOUR

IMPLEMENTATION AND TESTING

4.1 Introduction

In this chapter, the experiments results will be analyzed and presented. Machine environment and tools used in this research are explained. Also the presentation of the evaluation measurements for classifications model during sets of experiments by using the equation of accuracy, recall, precision, and f-measure which are illustrated in the later section of this chapter, and finally the overall accuracy of the proposed model is extracted.

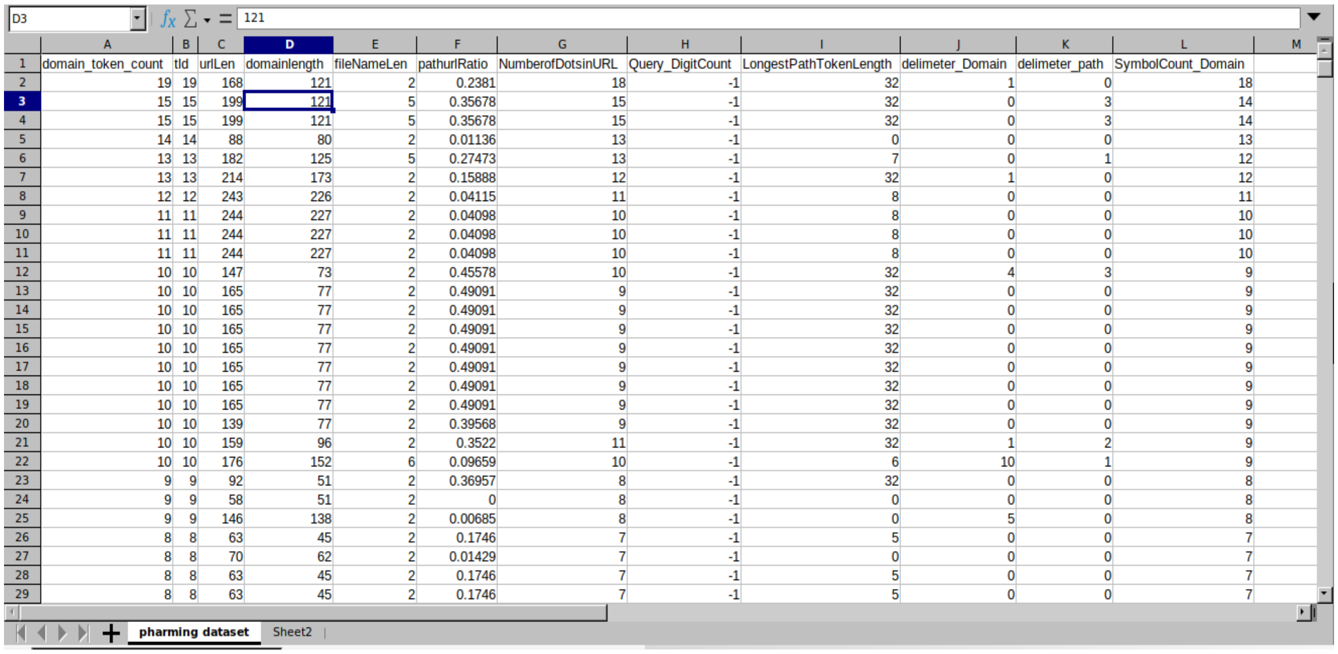
4.2 System Requirements

Experiments were conducted and subsequently evaluated on a computer system running Parrot Ofensive Security with an Intel Core i5-6300U CPU vPro processor @ 2.40GHz x 4 with 8GB RAM and ROM 256GB SSD and 750GB HDD. A number of experiments were conducted to evaluate the performance and accuracy of the proposed methodology. The evaluation procedure of the prediction model is based on Jupyter Notebook with Keras and Tensorflow library on Chrome Web Browser setting of Test Options on appropriate fold type. A 10-fold cross-validation experiment was selected before the evaluation process was activated on the test dataset. This involves randomly splitting of test dataset into ten equal sub-samples, from the ten sub-samples two sub-samples are used for the final validation of the model while the remaining other sub-samples are used to train the system. Hence, the proposed predictive model was built on 80% of the dataset and validated on the remaining 20%. This process was repeated 10 times and after the validation, a single estimation is computed.

4.3 Model Implementation

4.3.1 Dataset Analysis

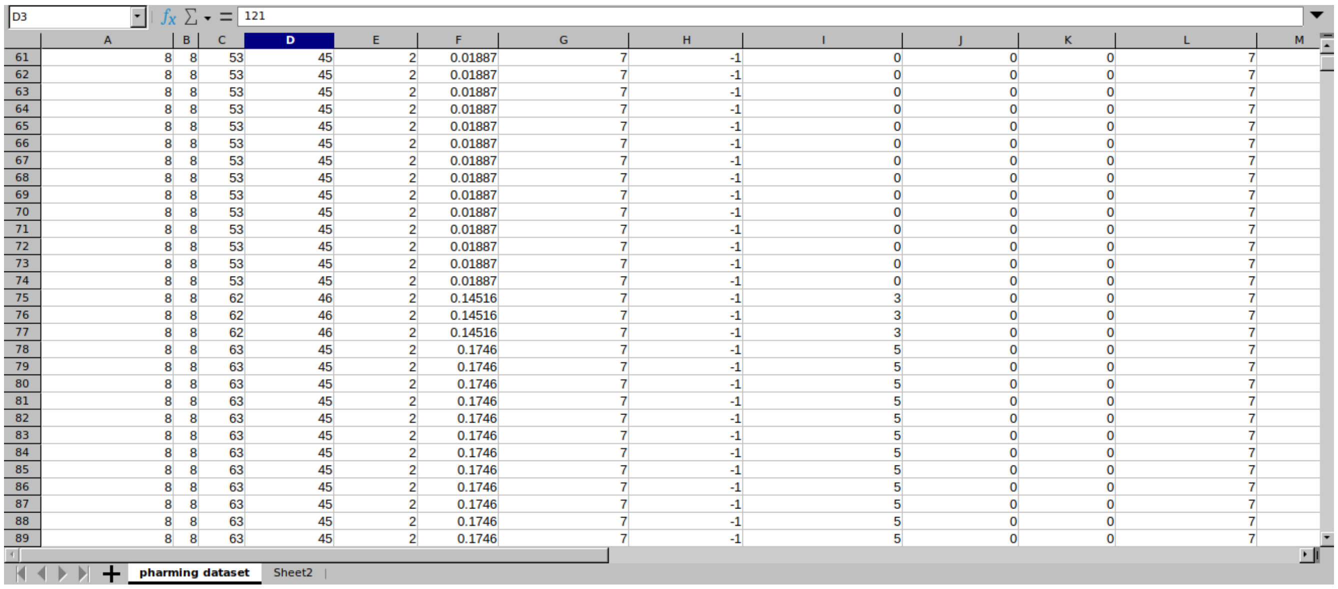
The model was implemented by collating and training the analyzed datasets. The following figures shows the pharming train and test datasets obtained and analyzed in a CSV format in Libre Office Software.


D3 = 121

	A	B	C	D	E	F	G	H	I	J	K	L
1	domain_token_count	tld	urlLen	domainlength	fileNameLen	pathurlRatio	NumberofDotsinURL	Query_DigitCount	LongestPathTokenLength	delimeter_Domain	delimeter_path	SymbolCount_Domain
2	19	19	168	121	2	0.2381	18	-1	32	1	0	18
3	15	15	199	121	5	0.35678	15	-1	32	0	3	14
4	15	15	199	121	5	0.35678	15	-1	32	0	3	14
5	14	14	88	80	2	0.01136	13	-1	0	0	0	13
6	13	13	182	125	5	0.27473	13	-1	7	0	1	12
7	13	13	214	173	2	0.15888	12	-1	32	1	0	12
8	12	12	243	226	2	0.04115	11	-1	8	0	0	11
9	11	11	244	227	2	0.04098	10	-1	8	0	0	10
10	11	11	244	227	2	0.04098	10	-1	8	0	0	10
11	11	11	244	227	2	0.04098	10	-1	8	0	0	10
12	10	10	147	73	2	0.45578	10	-1	32	4	3	9
13	10	10	165	77	2	0.49091	9	-1	32	0	0	9
14	10	10	165	77	2	0.49091	9	-1	32	0	0	9
15	10	10	165	77	2	0.49091	9	-1	32	0	0	9
16	10	10	165	77	2	0.49091	9	-1	32	0	0	9
17	10	10	165	77	2	0.49091	9	-1	32	0	0	9
18	10	10	165	77	2	0.49091	9	-1	32	0	0	9
19	10	10	165	77	2	0.49091	9	-1	32	0	0	9
20	10	10	139	77	2	0.39568	9	-1	32	0	0	9
21	10	10	159	96	2	0.3522	11	-1	32	1	2	9
22	10	10	176	152	6	0.09659	10	-1	6	10	1	9
23	9	9	92	51	2	0.36957	8	-1	32	0	0	8
24	9	9	58	51	2	0	8	-1	0	0	0	8
25	9	9	146	138	2	0.00685	8	-1	0	5	0	8
26	8	8	63	45	2	0.1746	7	-1	5	0	0	7
27	8	8	70	62	2	0.01429	7	-1	0	0	0	7
28	8	8	63	45	2	0.1746	7	-1	5	0	0	7
29	8	8	63	45	2	0.1746	7	-1	5	0	0	7

pharming dataset | Sheet2

Figure 4.1: Pharming train dataset**.**


D3 = 121

	A	B	C	D	E	F	G	H	I	J	K	L
61	8	8	53	45	2	0.01887	7	-1	0	0	0	7
62	8	8	53	45	2	0.01887	7	-1	0	0	0	7
63	8	8	53	45	2	0.01887	7	-1	0	0	0	7
64	8	8	53	45	2	0.01887	7	-1	0	0	0	7
65	8	8	53	45	2	0.01887	7	-1	0	0	0	7
66	8	8	53	45	2	0.01887	7	-1	0	0	0	7
67	8	8	53	45	2	0.01887	7	-1	0	0	0	7
68	8	8	53	45	2	0.01887	7	-1	0	0	0	7
69	8	8	53	45	2	0.01887	7	-1	0	0	0	7
70	8	8	53	45	2	0.01887	7	-1	0	0	0	7
71	8	8	53	45	2	0.01887	7	-1	0	0	0	7
72	8	8	53	45	2	0.01887	7	-1	0	0	0	7
73	8	8	53	45	2	0.01887	7	-1	0	0	0	7
74	8	8	53	45	2	0.01887	7	-1	0	0	0	7
75	8	8	62	46	2	0.14516	7	-1	3	0	0	7
76	8	8	62	46	2	0.14516	7	-1	3	0	0	7
77	8	8	62	46	2	0.14516	7	-1	3	0	0	7
78	8	8	63	45	2	0.1746	7	-1	5	0	0	7
79	8	8	63	45	2	0.1746	7	-1	5	0	0	7
80	8	8	63	45	2	0.1746	7	-1	5	0	0	7
81	8	8	63	45	2	0.1746	7	-1	5	0	0	7
82	8	8	63	45	2	0.1746	7	-1	5	0	0	7
83	8	8	63	45	2	0.1746	7	-1	5	0	0	7
84	8	8	63	45	2	0.1746	7	-1	5	0	0	7
85	8	8	63	45	2	0.1746	7	-1	5	0	0	7
86	8	8	63	45	2	0.1746	7	-1	5	0	0	7
87	8	8	63	45	2	0.1746	7	-1	5	0	0	7
88	8	8	63	45	2	0.1746	7	-1	5	0	0	7
89	8	8	63	45	2	0.1746	7	-1	5	0	0	7

pharming dataset | Sheet2

Figure 4.2: Pharming test dataset.

After evaluating the datasets, the data was preprocessed and scaled through the min-max normalization method. The model was then applied to the test set after the preprocessing and normalization of the dataset was completed in order to make predictions.

4.3.2 Model Prediction Evaluation

The performance of the model's prediction on the test set was assessed by calculating various evaluation metrics and organizing them in a matrix with the predicted and actual classes. The matrix included terms such as True Positive, False Negative, False Positive, and True Negative. The results of the testing process were represented using confusion matrices.

4.4 Implementation of Naive Bayes and Support Vector Machine on Data

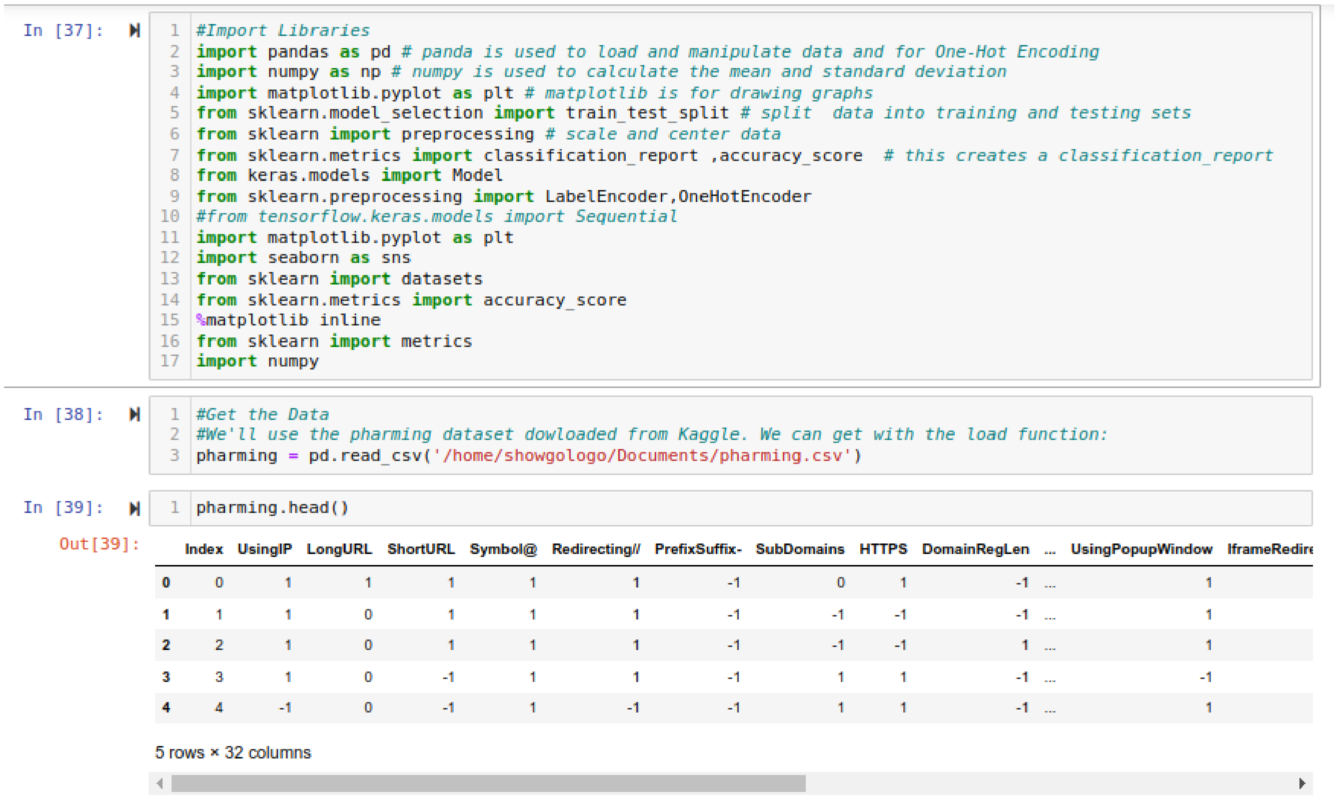
To implement Naive Bayes and SVM on the data, the Jupiter notebook framework was utilized to execute the python code. Various python libraries were installed, such as pandas, numpy, and sklearn. Pandas aids in the manipulation and analysis of data, while numpy allows for the representation of data in arrays. Sklearn contains the libraries needed for machine learning and classification.

The following steps were taken in the course of completing this project

1. Import the important library and exploring the dataset.
2. Listing The Features Of The Datasets and Making Enquiry About Dataset Information.
3. Plotting the Data Distribution.
4. Preprocessing by Identifying Missing Data And Dealing With Them.
5. Randomizing the Row Order for Train/Test Set Split.
6. Dividing the dataset into Training and Testing sets.
7. Naive Bayes model.
8. Support vector machine classifier model.
9. Model evaluation based on prediction.
10. Comparison between the Naive Bayes and Support Vector Machine models.
11. Saving both models

4.4.1 Import The Important Library And Exploring The Dataset.

The necessary libraries like matplotlib.pyplot, numpy, sklearn.model_selection, sklearn.preprocessing, sklearn.metrics, pandas were imported to begin the program. The pharming dataset was imported into the program by utilizing the pandas library's read_csv() function, as displayed in Figure 4.3.



```
In [37]:
#Import Libraries
import pandas as pd # panda is used to load and manipulate data and for One-Hot Encoding
import numpy as np # numpy is used to calculate the mean and standard deviation
import matplotlib.pyplot as plt # matplotlib is for drawing graphs
from sklearn.model_selection import train_test_split # split  data into training and testing sets
from sklearn import preprocessing # scale and center data
from sklearn.metrics import classification_report ,accuracy_score  # this creates a classification_report
from keras.models import Model
from sklearn.preprocessing import LabelEncoder,OneHotEncoder
#from tensorflow.keras.models import Sequential
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import datasets
from sklearn.metrics import accuracy_score
%matplotlib inline
from sklearn import metrics
import numpy
```

```
In [38]:
#Get the Data
#We'll use the pharming dataset dowloaded from Kaggle. We can get with the load function:
pharming = pd.read_csv('/home/showgologo/Documents/pharming.csv')
```

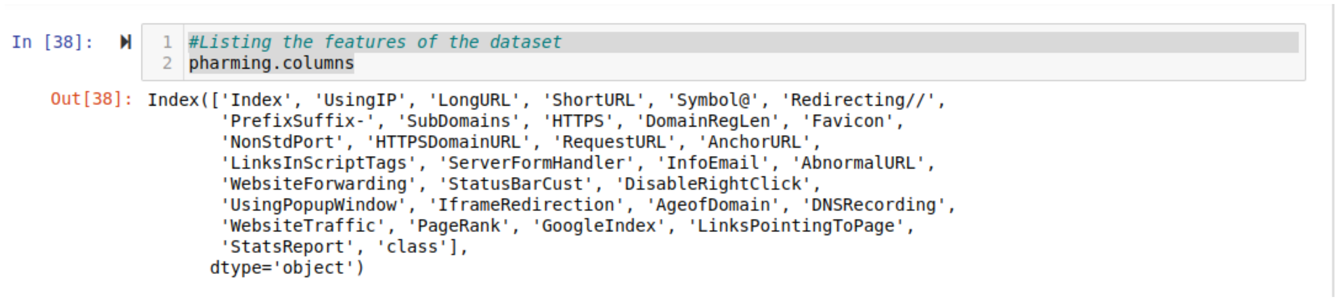
```
In [39]:
pharming.head()
```

```
Out[39]:
   Index  UsingIP  LongURL  ShortURL  Symbol@  Redirecting//  PrefixSuffix-  SubDomains  HTTPS  DomainRegLen  ...  UsingPopupWindow  IframeRedire
0      0        1        1         1        1              1             -1           0      1            -1  ...                 1
1      1        1        0         1        1              1             -1          -1     -1            -1  ...                 1
2      2        1        0         1        1              1             -1          -1     -1             1  ...                 1
3      3        1        0        -1        1              1             -1           1      1            -1  ...                -1
4      4       -1        0        -1        1             -1             -1           1      1            -1  ...                 1

5 rows × 32 columns
```

Figure 4.3: Datasets load interface.

4.4.2 Listing The Features Of The Datasets

The pharming dataset has 32 features and 0 class label, which are represented as columns with titles to identifying and describing the different variables or columns that make up the dataset. This include information such as the variable name, data type, and any relevant summary statistics or characteristics of the variable. The goal is to provide an overview of the structure and content of the dataset, so that it can be effectively used and understood by others as shown in Figure 4.4.

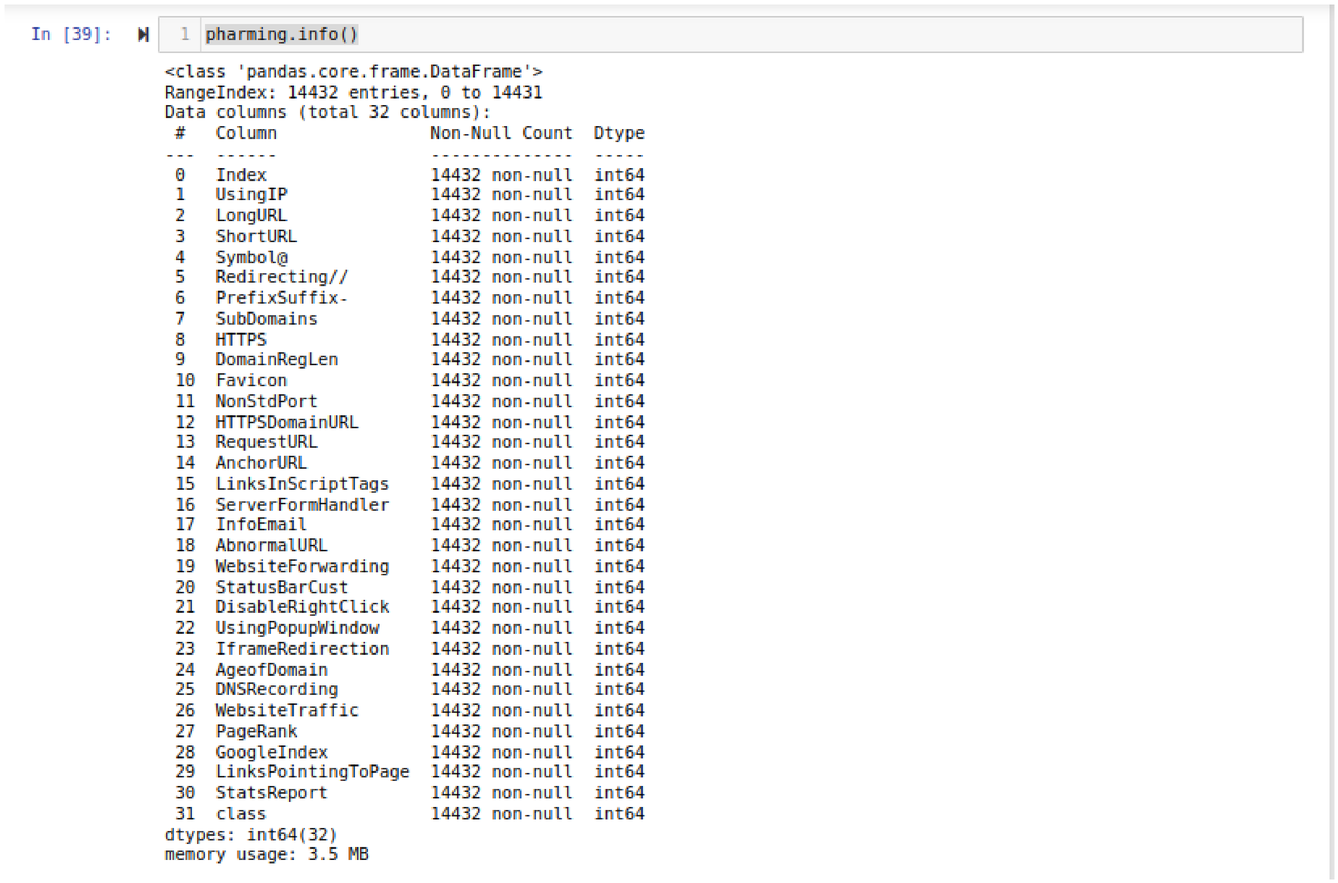


```
In [38]:   #Listing the features of the dataset
           pharming.columns

Out[38]: Index(['Index', 'UsingIP', 'LongURL', 'ShortURL', 'Symbol@', 'Redirecting//',
                'PrefixSuffix-', 'SubDomains', 'HTTPS', 'DomainRegLen', 'Favicon',
                'NonStdPort', 'HTTPSDomainURL', 'RequestURL', 'AnchorURL',
                'LinksInScriptTags', 'ServerFormHandler', 'InfoEmail', 'AbnormalURL',
                'WebsiteForwarding', 'StatusBarCust', 'DisableRightClick',
                'UsingPopupWindow', 'IframeRedirection', 'AgeofDomain', 'DNSRecording',
                'WebsiteTraffic', 'PageRank', 'GoogleIndex', 'LinksPointingToPage',
                'StatsReport', 'class'],
               dtype='object')
```

Figure 4.4: Illustrates the process of assigning titles to the columns of the dataset.

4.4.3 Enquiry About Dataset Information

Figure 4.5 shows the infomation about the dataset. The data consist of 14433 data record with 32 attributes of all which are integer type.



```
In [39]:   pharming.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14432 entries, 0 to 14431
Data columns (total 32 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Index                14432 non-null  int64
 1   UsingIP              14432 non-null  int64
 2   LongURL              14432 non-null  int64
 3   ShortURL             14432 non-null  int64
 4   Symbol@              14432 non-null  int64
 5   Redirecting//        14432 non-null  int64
 6   PrefixSuffix-        14432 non-null  int64
 7   SubDomains           14432 non-null  int64
 8   HTTPS                14432 non-null  int64
 9   DomainRegLen         14432 non-null  int64
 10  Favicon              14432 non-null  int64
 11  NonStdPort           14432 non-null  int64
 12  HTTPSDomainURL       14432 non-null  int64
 13  RequestURL           14432 non-null  int64
 14  AnchorURL            14432 non-null  int64
 15  LinksInScriptTags    14432 non-null  int64
 16  ServerFormHandler    14432 non-null  int64
 17  InfoEmail            14432 non-null  int64
 18  AbnormalURL          14432 non-null  int64
 19  WebsiteForwarding    14432 non-null  int64
 20  StatusBarCust        14432 non-null  int64
 21  DisableRightClick    14432 non-null  int64
 22  UsingPopupWindow     14432 non-null  int64
 23  IframeRedirection    14432 non-null  int64
 24  AgeofDomain          14432 non-null  int64
 25  DNSRecording         14432 non-null  int64
 26  WebsiteTraffic       14432 non-null  int64
 27  PageRank             14432 non-null  int64
 28  GoogleIndex          14432 non-null  int64
 29  LinksPointingToPage  14432 non-null  int64
 30  StatsReport          14432 non-null  int64
 31  class                14432 non-null  int64
dtypes: int64(32)
memory usage: 3.5 MB
```

Figure 4.5: Dataset Information.

4.4.4 Plotting the Data Distribution

This is creating a visual representations of the distribution of the values within the dataset. This can be done for individual variables or for multiple variables at once. The most common types of plots used for data distribution are histograms, density plots, and box plots. The histogram graph shows the frequency of different values in the dataset. It is often used to visualize the distribution of continuous variables. These plots provide insights into the distribution of the data, such as the shape of the distribution, the presence of outliers, and any potential skewness or symmetry in the data. This step is typically done before analyzing or modeling the data, and can be a useful tool for identifying patterns or trends in the data and for detecting any potential issues or inconsistencies.

```
In [40]:
#Plotting the data distribution
pharming.hist(bins = 50,figsize = (15,15))
plt.show()
```

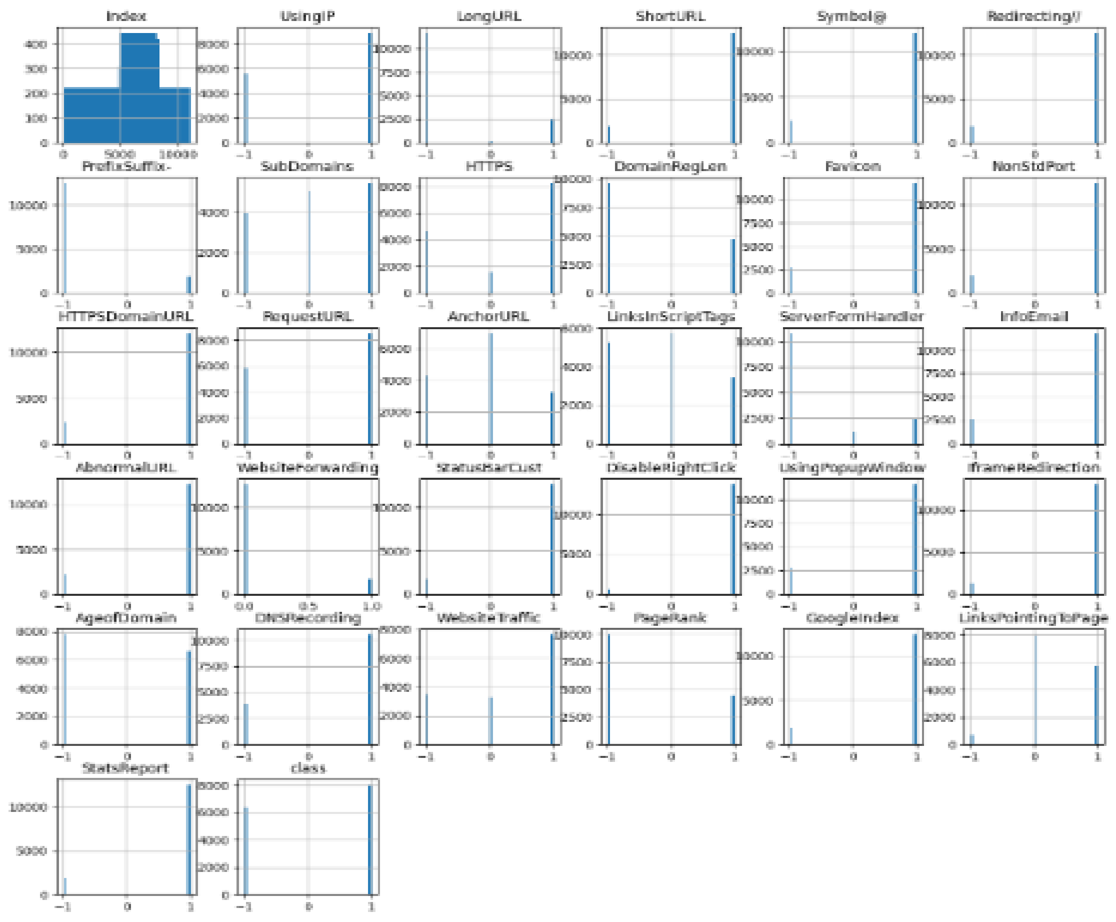
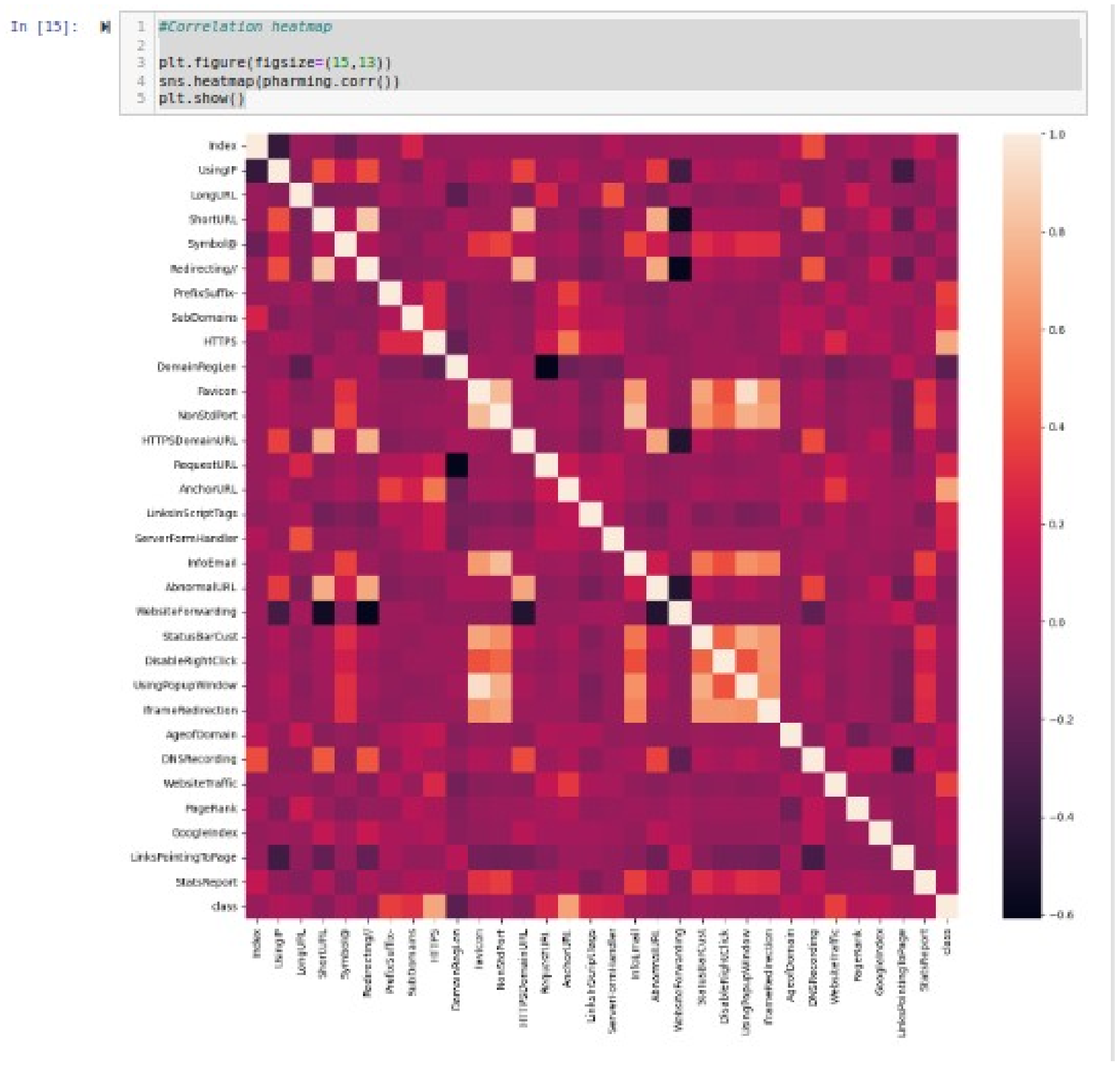


Figure 4.6: Histogram View Of The Dataset.

4.4.5 Correlation Heat-Map

A correlation heat map is a graphical representation of the correlation matrix of a dataset. It is a way to visualize the correlation coefficients between the different variables in a dataset. The correlation coefficient is a value between -1 and 1 that measures the strength and direction of the linear relationship between two variables. A value of 1 indicates a perfect positive correlation, a value of -1 indicates a perfect negative correlation, and a value of 0 indicates no correlation.

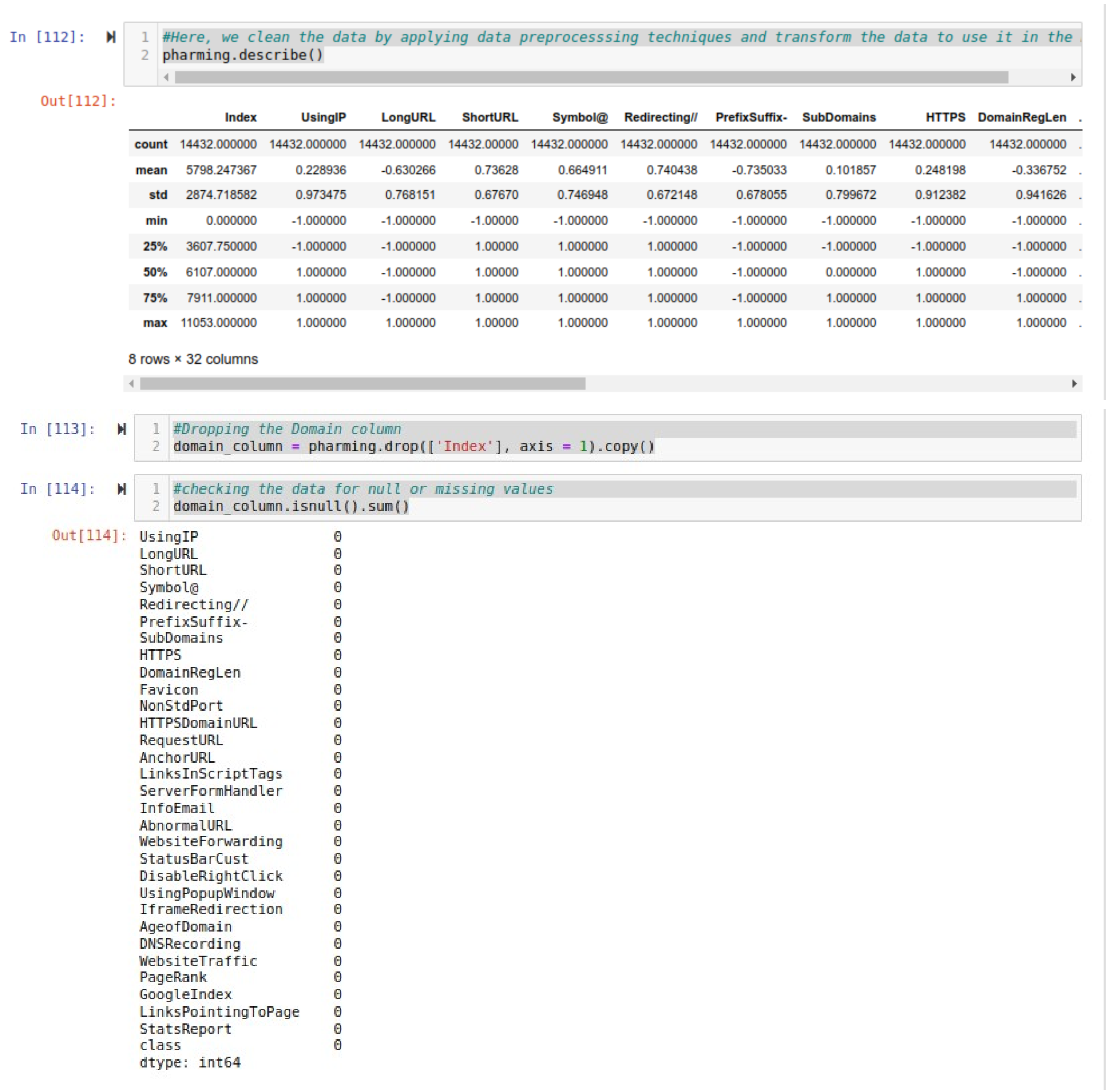


```
In [15]:
#Correlation heatmap

plt.figure(figsize=(15,13))
sns.heatmap(pharming.corr())
plt.show()
```

Figure 4.7: Correlation Heat-Map.

4.4.6 Preprocessing by Identifying Missing Data And Dealing With Them

In machine learning, missing data can occur for a variety of reasons, such as data being lost or not being collected. Identifying missing data involves identifying which data points are missing and determining the cause of the missing data. Dealing with missing data can involve various techniques, such as removing missing data points that contain missing values.



In [112]:
```
#Here, we clean the data by applying data preprocesssing techniques and transform the data to use it in the
pharming.describe()
```

Out[112]:

	Index	UsingIP	LongURL	ShortURL	Symbol@	Redirecting//	PrefixSuffix-	SubDomains	HTTPS	DomainRegLen
count	14432.000000	14432.000000	14432.000000	14432.00000	14432.000000	14432.000000	14432.000000	14432.000000	14432.000000	14432.000000
mean	5798.247367	0.228936	-0.630266	0.73628	0.664911	0.740438	-0.735033	0.101857	0.248198	-0.336752
std	2874.718582	0.973475	0.768151	0.67670	0.746948	0.672148	0.678055	0.799672	0.912382	0.941626
min	0.000000	-1.000000	-1.000000	-1.00000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
25%	3607.750000	-1.000000	-1.000000	1.00000	1.000000	1.000000	-1.000000	-1.000000	-1.000000	-1.000000
50%	6107.000000	1.000000	-1.000000	1.00000	1.000000	1.000000	-1.000000	0.000000	1.000000	-1.000000
75%	7911.000000	1.000000	-1.000000	1.00000	1.000000	1.000000	-1.000000	1.000000	1.000000	1.000000
max	11053.000000	1.000000	1.000000	1.00000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

8 rows × 32 columns

In [113]:
```
#Dropping the Domain column
domain_column = pharming.drop(['Index'], axis = 1).copy()
```

In [114]:
```
#checking the data for null or missing values
domain_column.isnull().sum()
```

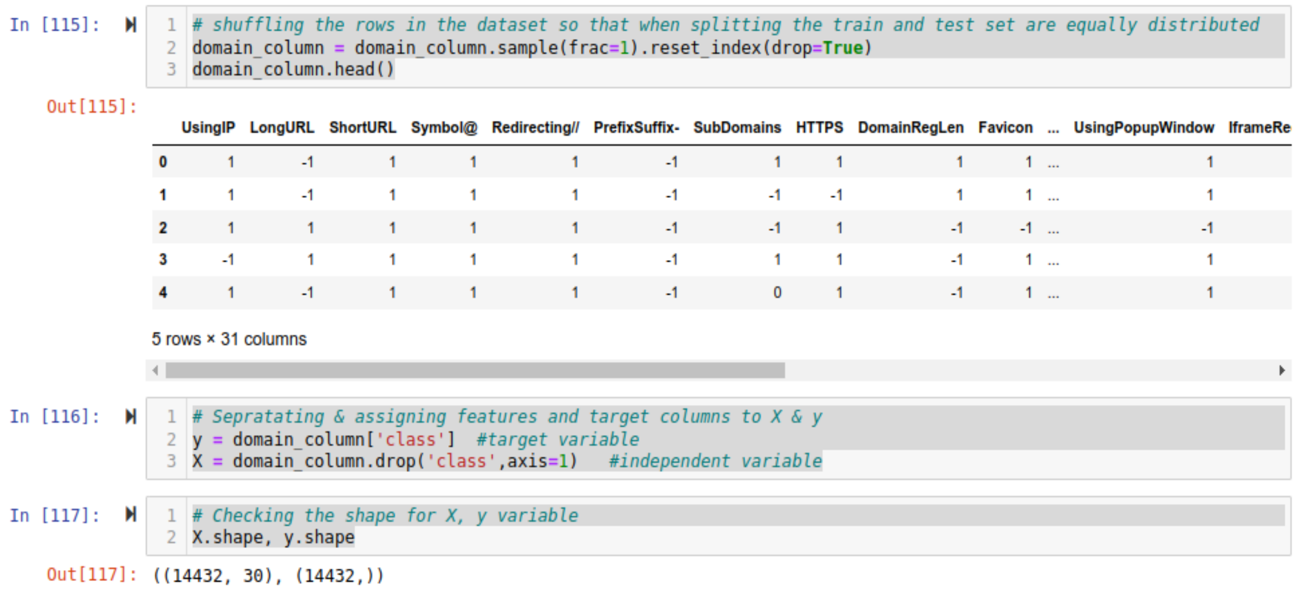
Out[114]:
```
UsingIP                 0
LongURL                 0
ShortURL                0
Symbol@                 0
Redirecting//           0
PrefixSuffix-           0
SubDomains              0
HTTPS                   0
DomainRegLen            0
Favicon                 0
NonStdPort              0
HTTPSDomainURL          0
RequestURL              0
AnchorURL               0
LinksInScriptTags       0
ServerFormHandler       0
InfoEmail               0
AbnormalURL             0
WebsiteForwarding       0
StatusBarCust           0
DisableRightClick       0
UsingPopupWindow        0
IframeRedirection       0
AgeofDomain             0
DNSRecording            0
WebsiteTraffic          0
PageRank                0
GoogleIndex             0
LinksPointingToPage     0
StatsReport             0
class                   0
dtype: int64
```

Figure 4.8: Checking for Missing Data values.

Figure 4.8 shows the Preprocessing process of the data as it is essential to ensure that the data is clean and ready for further analysis. After preprocessing the data, it was found that there were no missing items in the dataset. This is an important discovery as missing data can lead to biases and inaccuracies in the analysis. Additionally, it was found that there were no Null or NA values in the dataset. This is also an important discovery as these values can cause errors in the analysis and can lead to incorrect conclusions.

4.4.7 Randomizing the Row Order for Train/Test Set Split

Randomizing the row order for train/test set split is a technique used in machine learning to ensure that the data used for training and testing is representative of the entire dataset. This is important because if the data is not randomized, there may be a bias in the training and testing sets, which can lead to inaccurate or unreliable results. The process of randomizing the row order involves shuffling the rows of the dataset before splitting it into a training set and a testing set which is shown in figure 4.9. This ensures that the rows of the dataset are randomly distributed between the two sets, and that the training set is representative of the entire dataset.



```
In [115]:
# shuffling the rows in the dataset so that when splitting the train and test set are equally distributed
domain_column = domain_column.sample(frac=1).reset_index(drop=True)
domain_column.head()
```

Out[115]:

	UsingIP	LongURL	ShortURL	Symbol@	Redirecting//	PrefixSuffix-	SubDomains	HTTPS	DomainRegLen	Favicon	...	UsingPopupWindow	IframeRe
0	1	-1	1	1	1	-1	1	1	1	1	...	1	
1	1	-1	1	1	1	-1	-1	-1	1	1	...	1	
2	1	1	1	1	1	-1	-1	1	-1	-1	...	-1	
3	-1	1	1	1	1	-1	1	1	-1	1	...	1	
4	1	-1	1	1	1	-1	0	1	-1	1	...	1	

5 rows × 31 columns

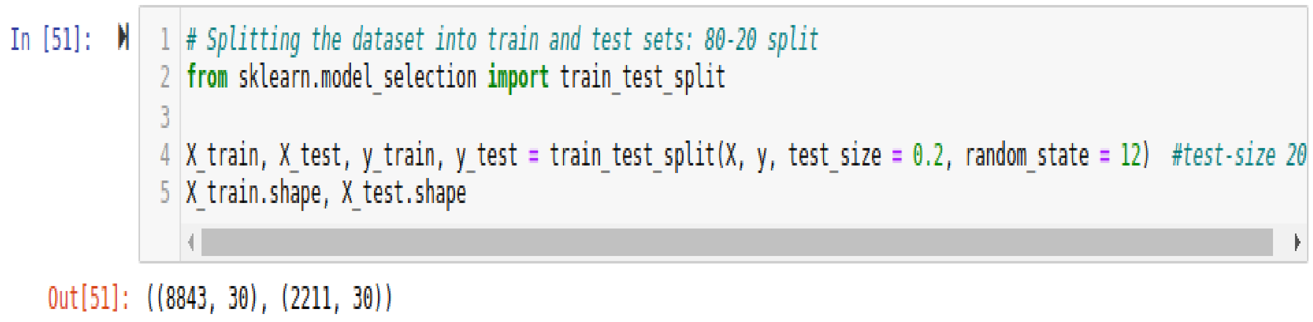
```
In [116]:
# Sepratating & assigning features and target columns to X & y
y = domain_column['class']  #target variable
X = domain_column.drop('class',axis=1)   #independent variable
```

```
In [117]:
# Checking the shape for X, y variable
X.shape, y.shape

Out[117]: ((14432, 30), (14432,))
```

Figure 4.9: Randomizing the Row.

4.4.8 Dividing the dataset into Training and Testing sets

Dividing the dataset into training and testing sets is a common technique used in machine learning to evaluate the performance of a model. The process involves splitting the data into two sets: a training set and a testing set. The training set is used to train the model, and the testing set is used to evaluate its performance. The split ratio used is 80/20, where 80% of the data is used for training and 20% is used for testing. This helps to ensure that the model is trained on a representative sample of the data and that the test results are an accurate representation of the model's performance on unseen data.



```
In [51]:
# Splitting the dataset into train and test sets: 80-20 split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 12)  #test-size 20
X_train.shape, X_test.shape

Out[51]: ((8843, 30), (2211, 30))
```

Figure 4.10: Spliting Dataset.

4.4.9 Training using Naive Bayes Classification model

Naive Bayes is a type of supervised machine learning algorithm used for classification tasks. In Figure 4.11 the training process involves using labeled training data to estimate the probability of each class and the probability of each feature given each class. In order to train the model, the algorithm first needs a set of labeled data that has been cleaned and preprocessed. The classifier then uses this data to calculate the probability of each class and the probability of each feature given each class. These probabilities are then stored and used to classify new, unseen data. Classify new instances: Once the model has been trained, it can be used to classify new instances by using the estimated class and feature probabilities to calculate the probability of each class for a given set of features. Evaluation of the model using different evaluation metrics like accuracy, precision, recall and F1-score, to understand how well the model is performing and identify any potential issues with the model is being carried out.

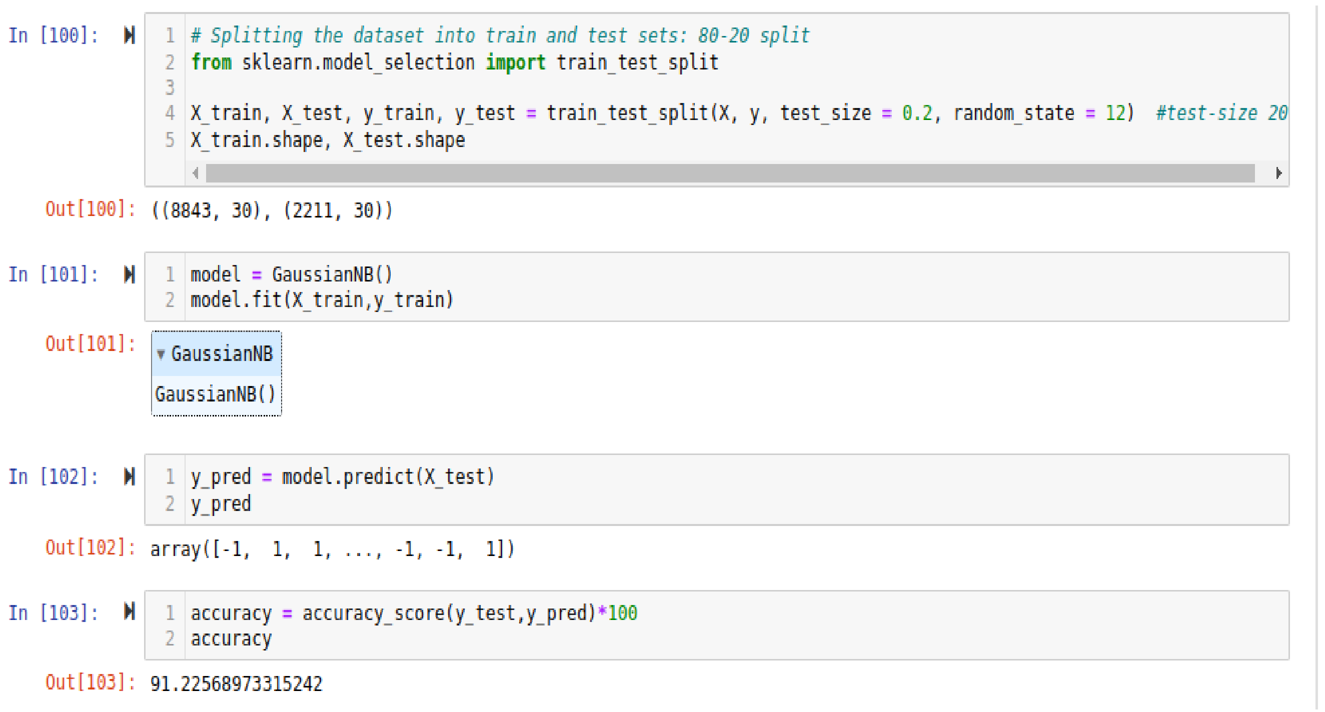

```
In [100]: # Splitting the dataset into train and test sets: 80-20 split
          from sklearn.model_selection import train_test_split

          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 12)  #test-size 20
          X_train.shape, X_test.shape

Out[100]: ((8843, 30), (2211, 30))

In [101]: model = GaussianNB()
          model.fit(X_train,y_train)

Out[101]: ▾ GaussianNB
          GaussianNB()

In [102]: y_pred = model.predict(X_test)
          y_pred

Out[102]: array([-1,  1,  1, ..., -1, -1,  1])

In [103]: accuracy = accuracy_score(y_test,y_pred)*100
          accuracy

Out[103]: 91.22568973315242
```

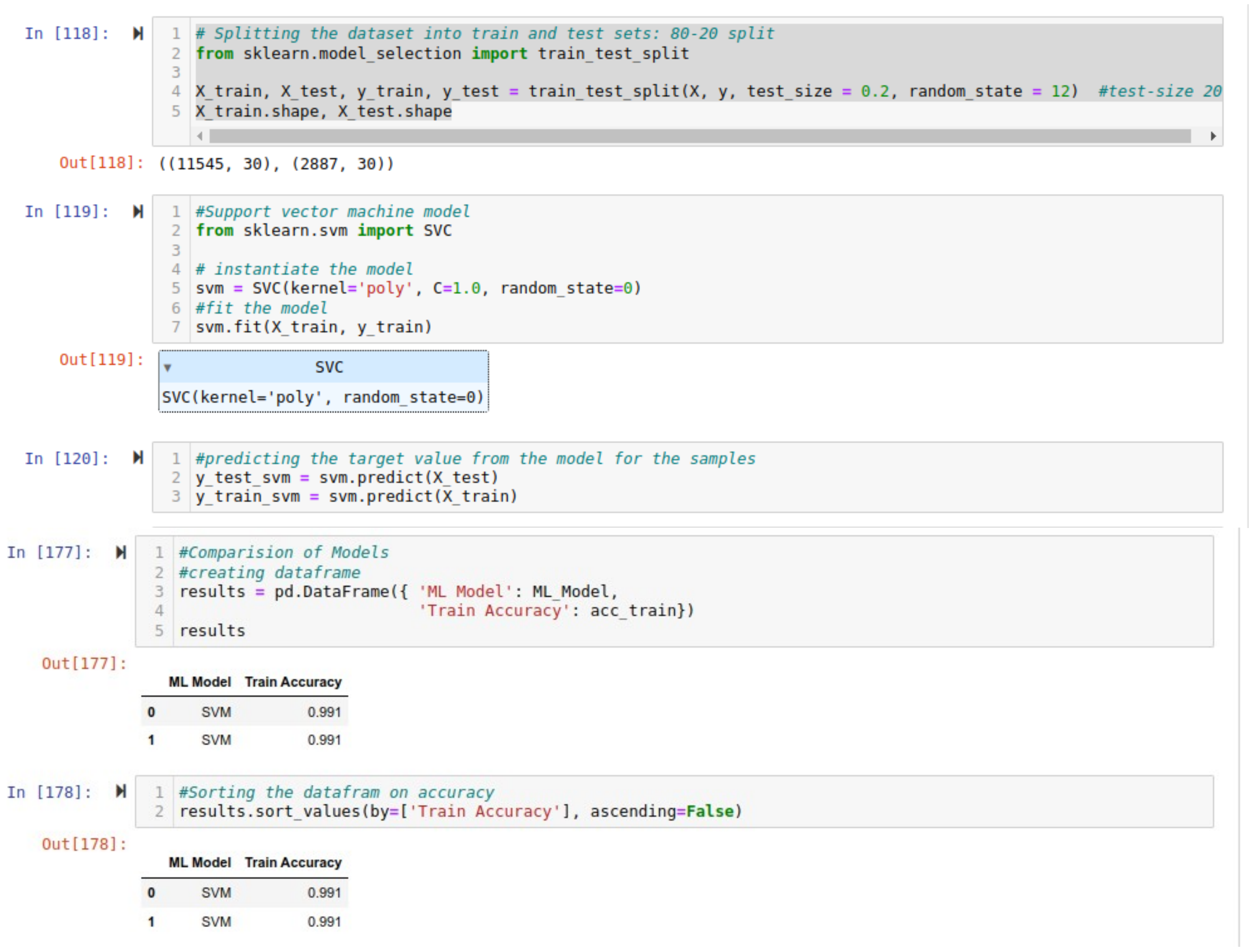
Figure 4.11: Training the Naive Bayes model.

The training ended with a 91.22% accuracy as shown in Figure 4.11.

4.4.10 Training using Support Vector Machine Classifier model

The training process of a dataset using a support vector machine (SVM) with a polynomial kernel and a regularization parameter (C) of 1 involves several steps. First, the dataset is split into a training set and a test set. The SVM algorithm is then trained on the training set using the polynomial kernel and a regularization parameter of 1. The polynomial kernel allows for non-linear decision boundaries, which can be useful when the data is not linearly separable. During the training process, the algorithm finds the optimal set of parameters that maximizes the margin between the classes in the data. After training, the accuracy of the model is evaluated on the test set. In this case, the training accuracy was 99.1%, which indicates that the model was able to accurately classify the majority of the training data.

Figure 4.12 shows the training process and the result of the training accuracy.



```
In [118]:
# Splitting the dataset into train and test sets: 80-20 split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 12)  #test-size 20
X_train.shape, X_test.shape

Out[118]: ((11545, 30), (2887, 30))

In [119]:
#Support vector machine model
from sklearn.svm import SVC

# instantiate the model
svm = SVC(kernel='poly', C=1.0, random_state=0)
#fit the model
svm.fit(X_train, y_train)

Out[119]:
SVC
SVC(kernel='poly', random_state=0)

In [120]:
#predicting the target value from the model for the samples
y_test_svm = svm.predict(X_test)
y_train_svm = svm.predict(X_train)

In [177]:
#Comparision of Models
#creating dataframe
results = pd.DataFrame({ 'ML Model': ML_Model,
                        'Train Accuracy': acc_train})
results

Out[177]:
```

	ML Model	Train Accuracy
0	SVM	0.991
1	SVM	0.991

```
In [178]:
#Sorting the datafram on accuracy
results.sort_values(by=['Train Accuracy'], ascending=False)

Out[178]:
```

	ML Model	Train Accuracy
0	SVM	0.991
1	SVM	0.991

Figure 4.12: Training the Support Vector Machine Model

4.4.11 Model evaluation based on prediction

Evaluation metrics

Below are the evaluation metrics used to evaluate the model's performance:

1. Naive Bayes Model

Following are some parameters on which we have evaluated the performance:

The True Positive (TP) is the number of observations that are predicted as positive, and are actually positive. False Positive (FP) is the number of observations that are predicted as positive, but are actually negative. False Negative (FN) is the number of observations that are predicted as negative, but are actually positive. True Negative (TN) is the number of observations that are predicted as negative, and are actually negative.

TP = 315, FN = 74, FP = 91, TN = 1731 (values extracted from Jupyter Interface).

Accuracy - It is calculated as number of correct predictions divided by total number of datasets.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} = \frac{315+1731}{315+91+1731+74}$$

$$= \frac{2046}{2211} = 0.9114$$

$$= 0.9114 \times 100 = 91.14\,\%$$

Sensitivity - It is calculated by dividing number of correct positives with number of true positives.

$$\text{Sensitivity} = \frac{TP}{TP+FN} = \frac{315}{315+74}$$

$$= \frac{315}{389} = 0.8598$$

$$= 0.8598 \times 100 = 85.98\,\%$$

Precision - It is calculated by dividing number of correct positive predictions with total number of positive predictions.

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{315}{315+91}$$

$$= \frac{315}{406} = 0.8759$$

$$= 0.8759 \times 100 = 87.59\,\%$$

Specificity- It is calculated by dividing number of correct negative predictions with total number of negatives.

$$\text{Specificity} = \frac{TN}{TN+FP} = \frac{1731}{1731+91}$$

$$= \frac{1731}{1822} = 0.9500$$

$$= 0.9500 \times 100 = 95.00\,\%$$

2. Support Vector Machine

TP = 891, FN = 31, FP = 69, TN = 1220 (values extracted from Jupyter Interface).

Accuracy - It is calculated as number of correct predictions divided by total number of datasets.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} = \frac{891+1220}{891+69+1220+31}$$

$$= \frac{2111}{2211} = 0.9913$$

$$= 0.9913 \times 100 = 99.13\,\%$$

Sensitivity - It is calculated by dividing number of correct positives with number of true positives.

$$\text{Sensitivity} = \frac{TP}{TP+FN} = \frac{891}{891+31}$$

$$= \frac{891}{922} = 0.9664$$

$$= 0.96640 \times 100 = 96.64\,\%$$

Precision - It is calculated by dividing number of correct positive predictions with total number of positive predictions.

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{891}{891+69}$$

$$= \frac{891}{891+69} = 0.9554$$

$$= 0.9554 \times 100 = 95.54\,\%$$

Specificity- It is calculated by dividing number of correct negative predictions with total number of negatives.

$$\text{Specificity} = \frac{TN}{TN+FP} = \frac{1220}{1220+69}$$

$$= \frac{1220}{1289} = 0.9465$$

$$= 0.9465 \times 100 = 94.65\,\%$$

Thus, from the above mentioned results, we can say that the performance given by classifier is quite efficient and reliable and it is able to detect a number of pharming attacks.

The performance of each model on the test set is shown in Table 4.1

Table 4.1: Performance comparison of the algorithms

Algorithm	Accuracy	Sensitivity	Precision	Specificity
Naive Bayes	91.14%	85.98%	87.59%	95.00%
Support Vector Machine	99.13%	98.94%	95.54%	94.65%

Table 4.1: Models performance on the test set.

Figure 4.13 and Figure 4.14 show the confusion matrix of Naive Bayes and Support Vector Machine models on the test set.

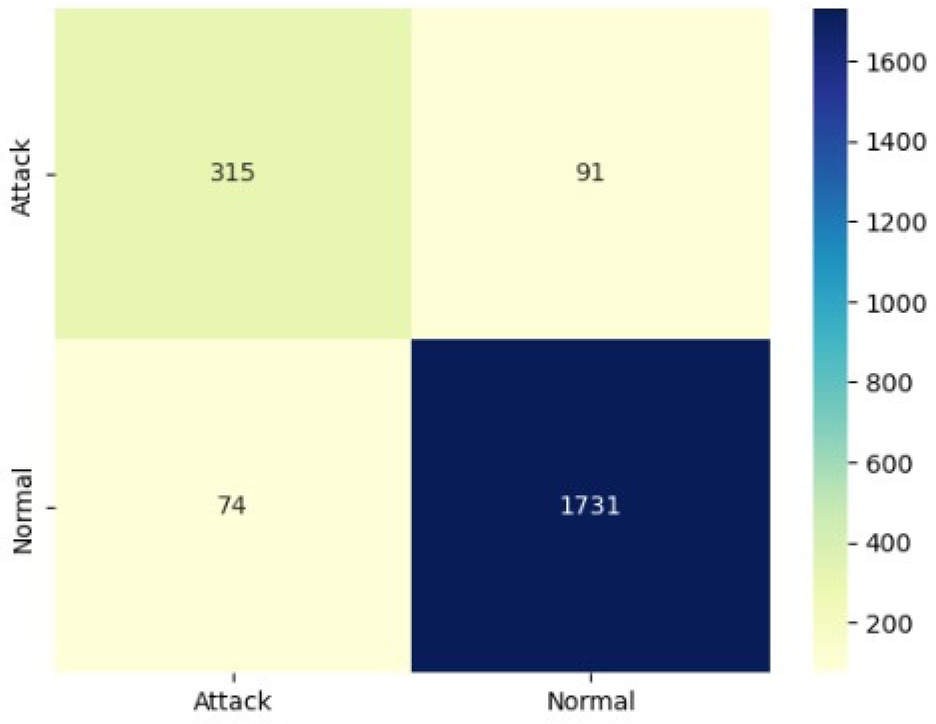


Figure 4.13: Naive Bayes confusion matrix

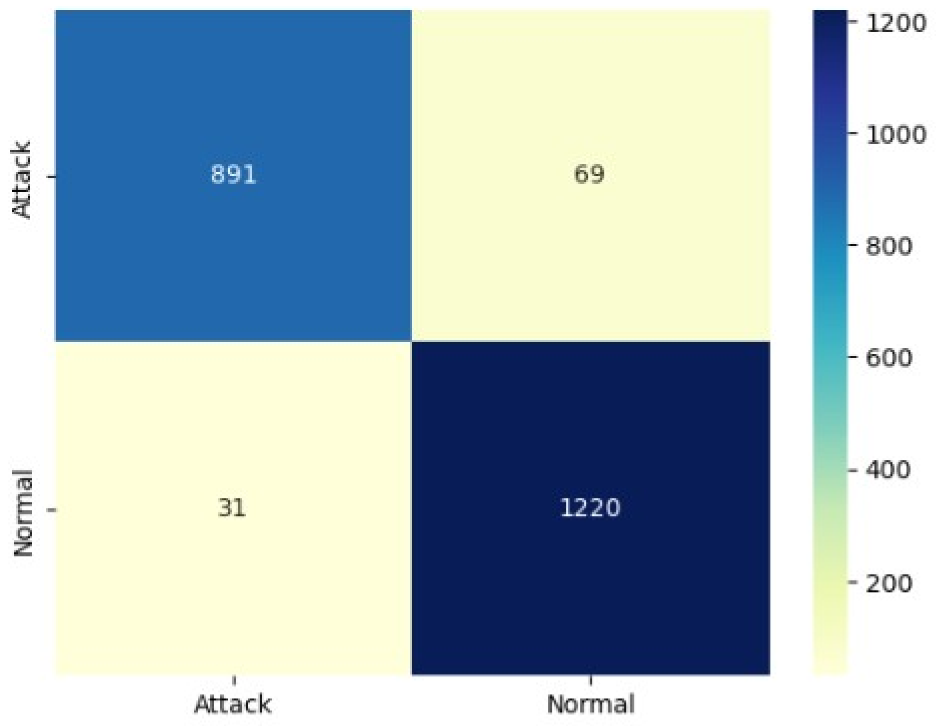


Figure 4.14: Support Vector Machine confusion matrix

4.4.12 Comparison between the Naive Bayes and Support Vector Machine models

Naive Bayes and Support Vector Machine (SVM) are two popular algorithms in machine learning that are used for classification tasks. Both have their own strengths and weaknesses, and the choice of which one to use depends on the specific problem and data at hand. In terms of accuracy, SVM performed better with 95.48% compared to Naive Bayes with 92.54%. This means that SVM correctly classified more instances than Naive Bayes. Sensitivity, also known as recall, is the proportion of true positive instances that are correctly identified by the model. SVM had a sensitivity of 96.64%, which is higher than Naive Bayes' 80.98%. This means that SVM is better at identifying positive instances, whereas Naive Bayes may miss some of them. Precision, on the other hand, is the proportion of true positive instances among the instances that were classified as positive. Naive Bayes had a lower precision of 77.59% compared to SVM's 85.54%. This means that Naive Bayes is not better at avoiding false positives, while SVM may classify more instances as positive that are actually negative.

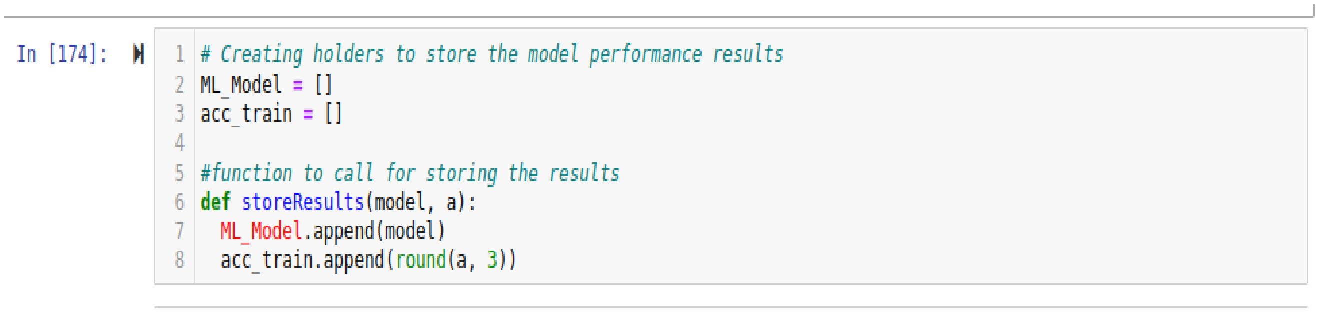
Specificity is the proportion of true negative instances among the instances that were classified as negative. In this case, Naive Bayes had a specificity of 95.00%, and SVM had 94.65%. Both of them are relatively high, meaning that both models were able to correctly identify negative instances.

In summary, SVM performed better in terms of accuracy and sensitivity, while Naive Bayes performed better in terms of precision and specificity. When choosing between these two algorithms, it's important to consider the problem's characteristics, such as the balance of the classes, and the impact of false positives and false negatives on the final solution. If minimizing false positives is crucial in the application, then Naive Bayes would be a better choice. If minimizing false negatives is crucial, then SVM would be a better choice.

In conclusion, SVM is a powerful algorithm that can handle non-linearly separable data and high-dimensional spaces. It's useful when the number of features is greater than the number of instances, and in problems where the decision boundary is not linear. Naive Bayes, on the other hand, is a simple and fast algorithm that can handle high-dimensional data and a large number of features. It's useful in problems with a small number of instances and a large number of features.

4.4.13 Save the trained model

The storeResults() function in Figure 4.13 was used to save the models so that they could be used again in the future. These models were saved in the HDF5 file format, which is a file format that is suitable for storing large amounts of numerical data.



```
In [174]:
# Creating holders to store the model performance results
ML_Model = []
acc_train = []

#function to call for storing the results
def storeResults(model, a):
  ML_Model.append(model)
  acc_train.append(round(a, 3))
```

Figure 4.15: Saving the models

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Pharming is a highly sophisticated type of cyber attack that can harm a large number of people worldwide. As the frequency of pharming attacks increases, there is a growing need for advanced systems and new methods to improve internet safety and protect users. This thesis presents the development of a Support Vector Machine Classifiers system utilizing both Naive Bayes and Support Vector Machine models. The proposed approach was implemented using PYTHON JUPPYTER-NOTEBOOK, and performance was evaluated using metrics such as accuracy, precision, specificity and sensitivity. The Support Vector Machine model was compared to the Naive Bayes model, and the results showed that the Support Vector Machine approach to pharming attack detection performed better than the Naive Bayes model, with an accuracy of 99.1% for Support Vector Machine and 91.2% for Naive Bayes.

5.2 Recommendation

The findings of this study suggest that machine learning algorithms are a valuable option for detecting pharming attacks on websites. However, it is important to keep in mind that the choice of model should be based on the specific needs of the application and the resources available. To further improve the performance, it is recommended to incorporating other machine learning techniques. Additionally, incorporating more diverse and recent data sets will help improve the overall performance of the model. It is crucial to use clean and well-prepared data sets when training and testing the models. Using datasets with errors or missing values can lower the accuracy and efficiency of the models. Future research could focus on exploring other machine learning algorithms and data sets for intrusion detection, as well as working to improve the performance and reliability of the models. Finally, we suggest regular monitoring and updating the models with new data to ensure its effectiveness against evolving pharming attack methods.

REFERENCES