2019

A Phishing detection model based on dynamic-hybrid feature selection

Ruiru, Daniel K
Faculty of Information Technology
Strathmore University

STRATHMORE UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

A Phishing Detection Model Based on Dynamic-Hybrid Feature Selection

By

Ruiru Daniel K

111777

A Thesis Proposal Submitted to the Faculty of Information Technology in partial fulfillment of the requirements for the award of Master of Science in Information Technology.

Master of Science in Information Technology

Strathmore University

July 2019

Declaration and Approval

I Ruiru Daniel K declare that this research has not been submitted to any other University for the award of a Degree in Master of Science in Information Technology

Student Name: Ruiru Daniel

Sign: Date: _____

Supervisor's Name: Dr Omwenga Vincent

Sign: Date: _____

Abstract

Phishing attacks have been a big internet nuisance since the early 1990s when hackers started stealing information from organisations using messaging platforms. At the time, the problem affected large institutions and corporations as the internet was still in its early stages of development and, had minimal individual subscribers. The early 2000s saw the widespread application of the technology (internet) which subsequently saw phishers target individual users using electronic mails (emails). In itself, phishing is a form of cyber-attack that steals personal information from unsuspecting users by duping them using verification or reward emails. This deception process ultimately helps the intruders to access sensitive data that can be used to access financial records for monetary gains or identity theft. Phishing attacks are so prevalent today that over 95 percent of all cyber-attacks are characterised by their intrusion procedures. Moreover, the attacks seem to increase each year and based on recent surveys are said to have a 60 percent annual growth rate. It is because of this outcome that this research proposes a predictive model to detect phishing attacks by implementing a system that pre-empts the intrusion processes before they happen. Unlike conventional methods that rely on human expertise to mitigate the problem, the proposed model automates the identification of the attacks and subsequently their control. This research aims to achieve this goal by optimizing the selection of subset features using a dynamic model that analyses the structural properties of phishing attacks to get adaptive attributes (features) for detecting phishing threats (as highlighted in chapter 4). Random forest is then used as the final classifier owing to its accuracy results (84.13%). Ultimately, the study then proposes the construction of a base model for bootstrapping other detection models in the cyber-security world.

Keywords: *Phishers, Phishing attacks, Neural Networks, E-mail, and Uniform Resource Locator (URL).*

Table of Contents

Table of Figures

Definition of Key Terms

Domain – A character string that identifies web addresses.

Email – Electronic mail.

Neural Networks – Computing systems that are inspired and modelled using the human nervous system.

Phisher – An intruder who uses phishing techniques to fraudulently access data.

Phone Phreaking – The exploration of phones and their functions.

SQL injections – A code injection procedure that allows intruders to interfere with queries.

STMP (Simple Mail Transfer Protocol) Server - A communication protocol that manages the transmission of electronic mail.

URL (Uniform Resource Locator) – A common name for web addresses.

Chapter 1: Introduction

1.1 Background

Internet technology has rapidly grown over the past two decades, enhancing the lives of its users. Today, the concept is more pervasive than ever offering capabilities such as online banking and social networking platforms. Its development has also come with the increase in security threats which using the same operational networks have created inventive methods to steal from online users. Social engineering attacks such as phishing are examples of these threats where intruders try to steal users' credentials using fake emails and websites. This problem continues to evolve each day as the attacks constantly adapt in response to the improved countermeasures used such as blacklists and spam filters (Gupta, Arachchilage & Psannis, 2017). Moreover, the anonymity of the internet, combined with the potential financial gains motivates perpetrators to conduct these heinous acts which presently are growing in numbers.

Phishing can be summed up as an attack that entices victims to submit their personal identity information (PII) as they think they are verifying accounts or getting rewards for answered surveys (Gupta, Arachchilage & Psannis, 2017). In its most recent form, this intrusion occurs in three simple steps; one, a phisher acquires e-mail addresses of their potential victims via social engineering attacks. Two, volumes of e-mails impersonating banks and surveys are sent to people using anonymous STMP servers. Finally, a re-direct to a website containing forms for entering critical personal data (PII) are used to collect users' information. In response to this attack, this paper's present a modern solution based on a two-fold objective: First, of highlighting the taxonomy of phishing attacks and two, a classification of the solution based on the features used to identify a threat.

The solutions presented in this research are a reactive response to the socio-economic effects of phishing attack which today surpass any previous estimations. According to SANS Institute research, over 95 percent of all enterprise attacks are as a result of successful phishing intrusion. Moreover, PhishMe, an intrusion mitigation solution developed by Cofense Inc, in its most recent report highlighted that phishing attacks had grown by over 65 percent within the past year (Katz, 2018). These findings then come just after Kaspersky Lab announced that it had prevented over 46 million phishing attempts in the past quarter of the year (2018), a figure that translated to one intrusion for every ten Kaspersky users (Bisson, 2018). Collectively, this problem is then estimated to cost the average large organizations over $3.7 million annually (Korolov,

2015). Finally, the prevalence of the intrusion is hallmarked by its persistent target on e-mail communication as 92 percent of its delivery are done via mail malware.

This form of delivery (mail) is thus a characteristic attribute of phishing intrusions, however, in their most profound state, Phishers modify the source codes of web pages to mimic legitimate websites. They further spoof organization's credentials to steal from unsuspecting internet users. As such, the manipulations used can be as elusive as the minor changes in web/source codes or as explicit as the removal and addition of content in documents. To try and mitigate this problem, a number of file matching algorithms have been implemented to analyze and detect phishing in websites. Generally, these methods compare the content of web pages with previously documented phishing attacks. Although they can be quite effective, these approaches tend to focus on one feature item which limits their results. The proposal at hand aims to expand on these methods by identifying the most viable features for detecting phishing attacks. Additionally, it aims to lay the foundation for creating a base model that can be used to bootstrap other detection algorithms.

Phishing attacks are among the most popular types of social engineering that not only affect individuals but societies as a whole by infringing on the collective rights of internet users. Phishers violet privacy, confidentiality, and intellectual property rights which heavily affects the overall goals of the internet such as free access, availability of information and anonymity. Today, extra caution and more importantly cost are spent to secure systems due to the impending dangers of scammers. Additionally, research statistics prove the importance of this noteworthy course as they highlight the severity of the situation. In all, this research offers a holistic overview of phishing attacks while developing a predictive detection model. The purpose of the research is therefore aimed to analyzing phishing attacks and developing an algorithm for their identification.

1.2 Problem Statement

Extensive research has been done in the area of phishing detection and has often led to the proposal of heuristic measures for its prevention. In most cases, the solutions have been dependent on training the users and, their abilities to detect phishing activities. In other instances, these methods have tried to equip browsers with toolbars and extensions to broadly identify any threatening events. Although functional, these techniques suffer from low detection accuracy and contain high levels of false results particularly, when introduced for the first time. Furthermore, on top of their inadequacies, these detection techniques employ blacklisting tactics to mitigate

intrusions which is inefficient, especially today when registering new domains is easier than ever before. Thus, based on the current state of events, there is not a truly comprehensive blacklisting tool with an accurate and up to date database to stop phishing attacks which necessitates the development of a new model to mitigate phishing attacks.

Research done by several scholars such as Akanbi and Fazeldehkordi (2015) has shown that the use of software classification methods results in better detection approaches to phishing attacks. Largely, the goal of involving classification technology is to reduce human involvement as it causes many errors depending on the users' levels of understanding. And as such, it eventually results in the identification of phishing messages based on the users' levels of expertise, which is never enough owing to the dynamic nature of the attacks. Additionally, the conventional methods of classification tend to focus on specific features of phishing attacks which limits the viability of the detection methods. By failing to consider multiple phishing features, the available algorithms restrict their roles to specific use cases and contexts.

This research wanted to develop a holistic phishing detection algorithm having the various mechanism of detecting the intrusions. The proposed model collaborated various features of phishing attacks from URL structure to the age of domains to better place the accuracy of intrusion identification. As such, the developed model shifted the role of detection from the users (who are prone to errors) to automated algorithms that over time have been found to have reasonable identification and prevention accuracy.

1.3 Aim

The aim of this study was to develop a model for detecting phishing attacks. The proposed model borrowed from the inherent structure of online facilities such as web pages and their URLs. A greater emphasis was then made on the modern-day structure of e-mails as they are often the bait of the attacks. Web browsers can then apply this model to stop phishing attacks aimed at their users. Moreover, organization and individual's alike can modify their model to meet their operational needs.

1.4 Specific Objectives

i. To classify the structural characteristics of phishing attacks.
ii. To evaluate the vulnerability exploited by Phishers to conduct their attacks.
iii. To analyse features for building a phishing detection model
iv. To develop a base model for bootstrapping phishing detection algorithms.

v. To test the reliability of the proposed model.

1.5 Research Questions

i. How are the structural characteristics of phishing material (Email and URLs) different from legitimate content ((Email and URLs)? Thus are there any attributes that can be used to distinguish between legitimate and illegitimate messages?

ii. How do phishers attack internet users and more specifically, what vulnerabilities do they exploit to gain access into systems?

iii. Which combination of features provides the best model (based on detection accuracy and resource utilization e.g. time)?

iv. Is the proposed model generalizable to other phishing problems/algorithms and what feature/labels can be adjusted to suit its generalizability?

v. How does the proposed model perform in comparison to the existing phishing detection algorithms? (based on the metric of false positives and false negatives).

1.6 Justification

The emergence of consumer oriented-electronic commerce is one of the most recognized successes of the internet. Through this virtual platform, anything can be bought at the click of a button, an outcome that has made life more efficient and propelled the growth of the society. However, this success has come along with a variety of security risks which often target the very core of its socioeconomic functionality. Phishing is one of these threats that in recent years has grown more sophisticated and has adversely affected countless financial institutions causing clients to fall victim to a variety of fraudulent activities.

Tricipher Consumer (2007), an online banking institution, once noted that one in every five adults have been victims of fraud or identity theft in their lifetime. These attacks, broadly financially motivated, affected all aspects of the consumers and organizations, a result that highlighted a fast-growing epidemic. Today, in addition to these economic impacts, phishing is causing severe social effects to internet participants such as diluting the credibility of online systems, the loss of customers' confidence for businesses and the forceful requirement of installing anti-phishing technology. Generally, this form of attack is affecting the very nature of the internet which originally was meant to be open and free, so as to enhance communication. Therefore, studying phishing attacks is a noble course that aims to restore the lost glory of the internet by

mitigating the adverse socio-economic effects of phishing attacks, an objective that is held by this proposal.

1.7 Scope and Limitation

This research focused on developing a base model for identifying phishing attacks based on the various features (characteristics) of the intrusion. Presumably, in this case, a phishing attack occurs when an intruder presents illicit sites to users and uses visual representations to make them look legitimate. Minimal considerations are made on sites that capitalize on browser vulnerability to infect systems with malware, that is, those driven by the download of infected content. As such, while conducting the investigations, the profiled content was analysed in isolated environments to avoid malware infections. Thus, an assumption made throughout the research was that of using an infection-free computer for system analysis. Additionally, this work did not focus on user interface issues and instead concentrated on matching websites and their content. Ultimately, the success of the project depended on the state of false positives as their presence level determined the accuracy of the developed model.

Chapter 2: Literature Review

2.1 Introduction

Security has been at the heart of computer technology since the early 1950s. During these early times, computers had rudimentary techniques of preventing applications from accessing memory not allocated to them. Subsequently, in the 1960s, simple access control and encryption methods were used to protect passwords. The problem of phishing then started to arise in the 1970s when the concept of Phone Phreaking was discovered after telecommunication engineers discovered that telephone switches accidentally crossed frequencies with perfect pitch (Gupta, Arachchilage & Psannis, 2017). Nonetheless, the modern outlook of phishing dates back to the 1990s when in 1995, hackers attempted to break the U.S. Department of Defence (DOD) system more than 250, 000 times. These attacks were then immediately followed by one of the most significant intrusions in computing history in 1996 when phishers stole America Online (AOL) customer's passwords. Broadly, hackers had started to use message boards and news groups to steal information. However, it was not until the year 2000, that phishers started to use mass-mails to spread attack emails in collaboration with spoofed URLs directing internet users to fake websites.

2.2 Theoretical Framework

2.2.1: The Heuristic-Systemic Model

Deception is a hallmark for phishing attacks as intruders use deceit and trickery to acquire sensitive personal information from internet users. The users are in many ways deceived to surrender their data which outlines the close relationship between psychology and this form of attack. Largely, phishers apply persuasion tactics to bait and infringe on people's systems. Psychology through the branch of persuasion studies helps to explain some of the outcomes exhibited in phishing attacks. Specifically, persuasion research through the definitions of the Heuristic-Systemic Model (HSM) outlines how messages sent and received are able to change people's attitudes as well as behaviours. HSM essentially shows that when people are being persuaded they use a combination of various heuristic and systemic processes to determine the validity of received messages (Luo, Zhang, Burd & Seazzu, 2013). The exact mix of these procedures is further linked to other factors which are regularly contextual and situational. The diagram below outlines this blend where persuasion depends on both heuristic and systematic

processing.

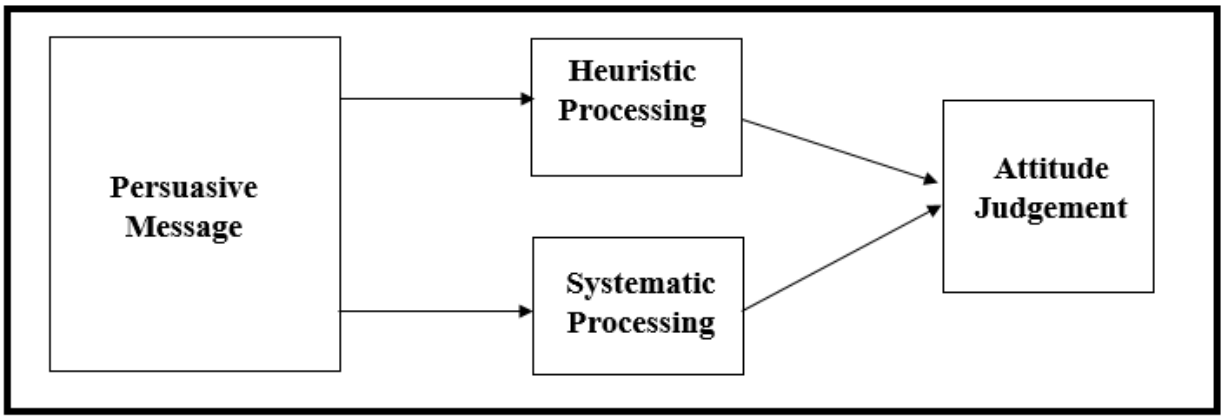

Figure 2-1: Heuristic-Systematic Model

Nonetheless, HSM does hold a familiar structure like other dual processing models because it integrates two different processing modes. First, Heuristic processing which exploits the factors entrenched within the message itself to make a quick validation assessment. Here elements such as the format, source, subject and lengths of the message come in play to evaluate the content received (i.e. the heuristic cues). Secondly, Systematic processing where the content of a message is researched to establish its authenticity. Of the two, the latter takes more effort, time and cognitive resources but is essential to attain an accurate validation of a message. HSM study results also tend to reveal that people often limit their resource investment without adequate motivation. Moreover, the same findings outline a number of factors that determine the cognitive resource investment made by participants and they include; perceived risk, skill level, distraction, time and perceived importance. Therefore, the process of validating a message is always subjective which makes it a highly complex and dynamic procedure.

Additionally, as described by Simon and Newell (1972) in their definition of satisficing people rarely aim to have the highest accuracy or reliability in their validity assessments. Regularly, users will halt the assessment process as soon as they feel they have ‘reliable’ or ‘good enough ‘results. It is this inaccurate notion that made the HSM model to include a unique concept of sufficiency threshold, which is preferred judgemental confidence that people need to make a decision. Thus, among the arguments made by HSM is that people conducting validity tests must surpass the sufficiency threshold to feel comfortable with their final decision(s), otherwise, they will continue with the processing. Therefore, even if the heuristic processing is easier if it does not

achieve the sufficiency threshold people will move to the systematic processing despite its resource requirement.

2.2.2 Phishing and the HSM model

Other than the general dual processing theories that HSM is applied, the model can also play an integral part in explaining the outcomes of other validity-seeking contexts. Phishing attacks and most other common cyber intrusions are good examples of these applications as they utilize the elements discussed in HSM. For instance, a majority of the messages sent to initiate phishing attacks usually contain false information which when thoroughly scrutinized can be easily identified by systematic processing. As such, the truly effective way of maximising the success of phishing attacks is to mislead recipients into making quick and inaccurate assessments to wrongly validate messages (Luo, Zhang, Burd & Seazzu, 2013). Ultimately, therefore, heuristic processing is the most critical element as it plays a vital role in the victimization of unsuspecting internet users.

From the points raised above, the success of all phishing attacks depend on the following objectives which can either be achieved individually or collectively:

i. Promoting heuristic processing having provided false cues to internet users so that they make hasty and inaccurate decisions.
ii. Sending messages that withstand systematic processing ensuring the receiver makes wrong evaluations about the validity of messages.
iii. Suppress systematic processing by all accounts so as to have the victims fully rely on the fast but error-prone heuristic processing.
iv. Finally, try and reduce the sufficiency threshold to make the victim feel confident with their assessments thus avoid to initiate the critical systematic processing.

2.3 Empirical Framework

2.3.1 Structural Characteristics of Phishing Attacks

All phishing attacks rely on deception and social engineering, often using visual similarities to trick internet users. The typical phishing scenario uses a large number of illegitimate emails while calling for action from subscribers by asking them to click on accompanying links. These links then have various variation in addresses such as cousin domain attacks. An example is

illustrated by PayPal website which has an address of *www.paypal.com* but is instead is attached as *wow.paypal.com*. Similarly, other types of intrusion such homograph attacks exploit confusion where an address's characters are replaced with resembling elements for example "1" for "L" (Chandrasekaran, Narayanan & Upadhyaya, 2006). As such, phishing applies the collaboration of emails with illicit websites to steal information following the structure illustrated below.

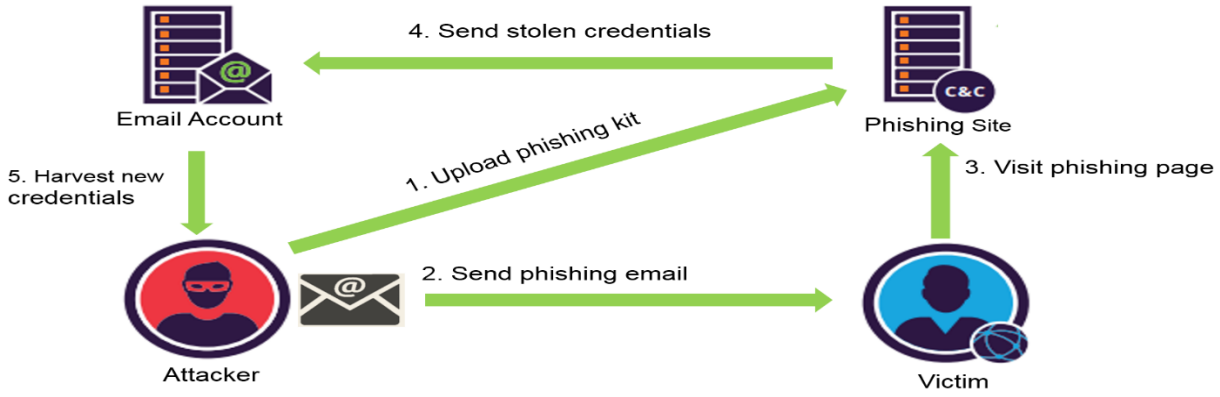

Figure 2-2: Phishing Flow Figure

Structure of Phishing Emails and Websites: Reputable companies are regularly spoofed to gain the trust of the users. Moreover, organizations dealing with financial transactions are often the target as they hold direct and greater rewards for the phishers. Thus, online banking services and retailers' systems are often the most probable targets. The alterations highlighted above are one of the tactics employed by phishers, in other attacks, the links attached to emails can be re-directed to different destinations. For instance, a link to Biashara bank *(www.biasharabank.com*) can be directed to *www.climagro.com.ar/agro/biashara.htm*. In this case, the phishers conceals the destination by obscuring the real URL. Another common technique that is employed by fraudsters is that of hiding the hostname using an IP address in a web site's link (e.g. "http://210.13.227.66/rs/"). Additionally, the URL can contain other forms of representations such as octal numbers, hexadecimal numbers and DWORD to hide the destination website.

Finally, there is generalization which is the ultimate key to addressing recipients as the content depends on the law of large numbers. Most phishing emails or messages never contain any personalized information when contacting the recipients to increase the size of the target audience. They also do not identify users with their identifiers such as names or identification number but instead apply vague terms such as client and customer (Chandrasekaran, Narayanan & Upadhyaya,

2006). However, to lure many victims, the content will contain well defined situational contexts such as threats and false urgencies. Furthermore, deception, elements of wheedling and concerns of deceit will also be used to encourage subscriber to click on highlighted text. These structural elements, therefore, can form the basis of developing contextual models of detecting phishing attacks.

2.3.2 Vulnerabilities used by Phishers

The brief overview presented above provides a simple outline of the evolution of phishing attacks. Largely, intruders overtime adapted to the changing times to meet their goals, an attribute that is observed even today. According to cybersecurity experts, more than 70 percent of all attacks capitalize on patchable vulnerabilities (Verizon Data Breach Investigation Reports (DBIR), 2015). In fact, the majority of web-based intrusion (over 98%) are opportunistic and focus on easy targets. Unlike the broader category of targeted attacks, opportunistic intrusions aim to exploit any vulnerable party thus depend on numbers and not technical prowess. Therefore, in most cases, the attacks in question use a variety of methods to propagate their intentions, starting with phishing campaigns, SQL injections, and notable malware infections.

Phishing on its behalf embodies the attributes of opportunistic attacks and is rarely targeted on any group of people or individual. However, there are incidences that do focus on specific online participants, most notably spear phishing, a much more technical category of phishing attacks. Nonetheless, despite the existence of spear attacks, phishing in itself is a game of numbers often characterized by the distribution of thousands of fraudulent messages in an attempt to scam the recipient. As such, the single most significant vulnerability lies with the users themselves, where their ignorance or naivety is exploited by intruders (Gupta, Arachchilage & Psannis, 2017). Moreover, poorly enforced security policies act as additional weaknesses that are utilized by phishers as their messages are able to reach internet subscribers. Additionally, the current widespread accessibility and availability of information provide the necessary foundation for launching phishing attacks. In this case, private data present in social and networking pages help intruders understand their potential victims, an exploit that further aggravates the results of the attacks as they seem more genuine to the users.

2.3.3 Phishing detection approaches

Based on the structural features identified above, several detection strategies have been devised to mitigate the phishing menace. The variation of these techniques also depends on the

terms of phishing operation and the scenarios used to track attack activities. As such, broadly, detection methods fall into two categories, non-classification and classification approaches. Non-classification methods in most cases include whitelisting trustworthy URLs, blacklisting phisher addresses, information flow detection systems and heuristic evaluations. On the other hand, classification techniques embody the hybridity of machine learning concepts and data mining in collaboration with the features identified above (Zuhair, Selamat & Salleh, 2016). Of the two, classification approaches are better as they outperform their counterparts in all elements. In fact, they have such good accuracy and precision in detection that they are often used by researchers as well as developers in intuitive phishing filters such as those used in modern web browsers. This proposal tries to extend the ideas of phishing classification and eventually culminates the process by developing a predictive model.

2.3.4 Feature selection and Features Used

Phishing detection techniques exploit the features of phishing attacks which broadly can be summarised into the following groups:

- IP-based URLs: Legitimate websites have domain names in their addresses. Phishers thus use IP addresses to identify their sites after hosting them in zombie systems.
- HTML Email: Plaintext emails do not provide the room or scale for the tricks used by phishers. Phishers often use HTML-formatted content which enable the integration of clickable links. Therefore, HTML based emails can be flagged and used as binary features for classification.
- Age of Domain: To avoid being caught, fraudsters use domain names for a limited period of time. This feature can thus be used to identify phishers by highlighting newly registered domains for scrutiny.
- Number of Domains and Sub-domains: One domain is often used but phishers will incorporate more than one to forward users to illicit websites, for instance http://www. biasharabank.com/url?sa=t&ct=res&cd=3&url=http%3A%2F%2Fwww.antifood.org%2G &ei= AVrQFp 5DDUzMs. The same applies to sub-domains as these intruders use them to gain the user trust. In this case, a URL has many dots, a feature that can be exploited by the detection process.
- Number of hyperlinks: Phishers play a game of numbers, therefore, the number of links in an email address can be capitalized to identify phishing websites.

- JavaScript Content: JavaScript can be used to deceive clients by hiding information from them. Any form of JavaScript incorporated in an email can be automatically flagged for assessment, another valuable selection feature.
- URL based images: To further pass the idea of legitimacy, images of real companies are employed in fraudulent emails. Thus, any URL based image can be outlined for thorough evaluation.
- Keywords: Certain words are regularly used in phishing emails, more os, those of urgency such as verify, suspend and username/password. Such terms form a handful feature for detecting phishers and their work (Zuhair, Selamat & Salleh, 2016).
- Matching Source and Body Domains: Finally, phishing emails tend to have different domains in their headers and body section. But, legitimate emails have matching domains in both segments. This distinction can form a basis of identifying fraudulent messages and emails.

2.3.5 Feature Selection Techniques: The Wrapper Approach

Before developing the necessary procedures for prediction, all intelligent agents must encounter the universal problem of focus. In this case, they must determine where to focus most of their attention. Simply put, any problem solving agent must analyse a problem and determine which aspects are relevant and which are not. Through this distinction, the designer of the expert system must choose certain features based on their specified worth. However, an even better agent must learn from experience (data set as training examples) to autonomously discriminate the relevant elements of a problem from the irrelevant ones (Kohavi & John, 1997). This requirement is what introduces machine learning to feature subset selection where an algorithm must select some subset feature(s) upon which it focuses its attention while assuming others.

In wrapper approach, this process (feature subset selection) exists through induction algorithms where a random search of the optimal features is done using reasoning having moved from specific to generalized contexts (induction inference). As shown in the figure below, the induction algorithm forms the heart of the wrapper process where it acts as a black box. This algorithm is ran on a dataset which is split into the elements of internal training and holdout sets. Through these facets, various sets of features are eliminated from a given data until a final optimal solutions (features) is obtained. The final set of features are the ones with the highest evaluation following the execution of the induction algorithm. Also, similar to other predictive models, the

resulting classifier is assessed using an independent data set (not used in the induction process) to evaluate its accuracy.

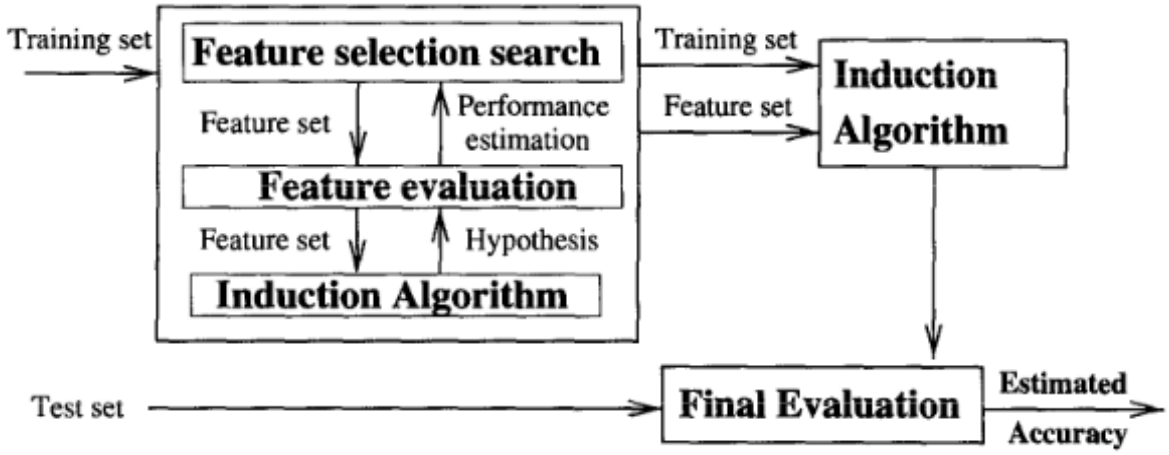

Figure 2-3: Wrapper Approach (Kohavi & John, 1997)

2.3.6 Neural Networks – The Basis of Most Detection Models

Machine learning plays a critical role in the field of data mining and analytics as it enables software to learn while adapting to various inputs. This adaptation helps improve the performance on various tasks which allows machines to mimic human behaviour and make predictions using either supervised or unsupervised algorithms (Sahingoz, Baykal & Bulut, 2018). Neural networks are of machine learning systems that are developed with the biological neural networks in mind. They try to emulate the functions of real neurons where an input of data is processed and then transmitted via electrical signals. The artificial neural networks developed operate using input nodes known as neurons which then subsequently diverge to include edges as either functions, layers or outputs. As such, the entire structure of the artificial neural networks operates with both nodes and edges where the input neurons connect all other neurons through their functions. The diagram below shows a simple representation of neurons (the basic building block of neural

networks).

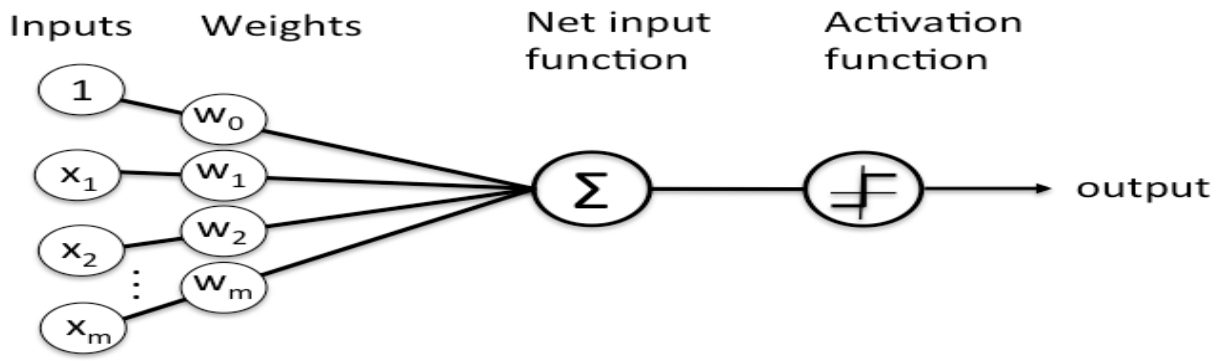

Figure 2-4: A Neuron Structure

Operation model : Given a certain input, neurons determine their output using the weights and bias criteria. In this case, the output is usually given by a summation of the weights multiplied by the input plus the bias criteria. Additionally, an activation function such as RELU or TANH, etc. processes the results before finally the calculation is decreased by a certain loss function. The output is therefore usually a gathering and comparison of real values. The end result (output) is also optimised throughout the process to produce an optimal predicted result. The figure below shows a simple structure of a neural network.

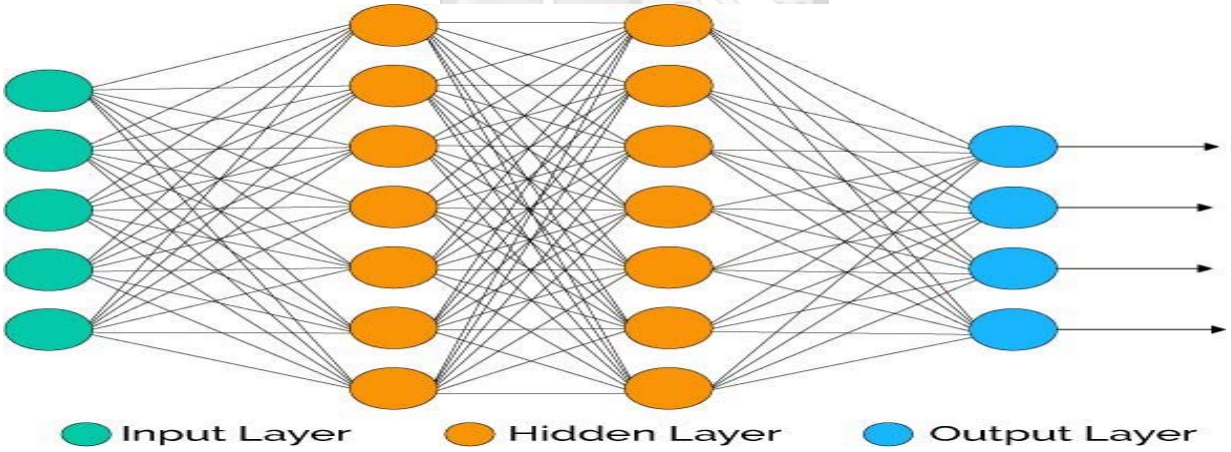

Figure 2-5: Simple structure of a Neural Network

Depending on the number of hidden layers, neural networks are broadly split into two, artificial neural networks and deep neural networks. This study aims to use the latter as it has shown great promise in the field. The model used will also incorporate the concepts of transfer learning where working models are re-used for data-intensive tasks which optimises the time spent on developing a predictive model (Zhang & Yuan, n.d.). Also, to better evaluate the performance

of the model (in phishing detection), the study will make several comparisons with other major machine learning classifiers. As such, decision tree, naive Bayes, support vector machine and K-nearest neighbours will be considered in the study.

2.4 Conceptual Framework

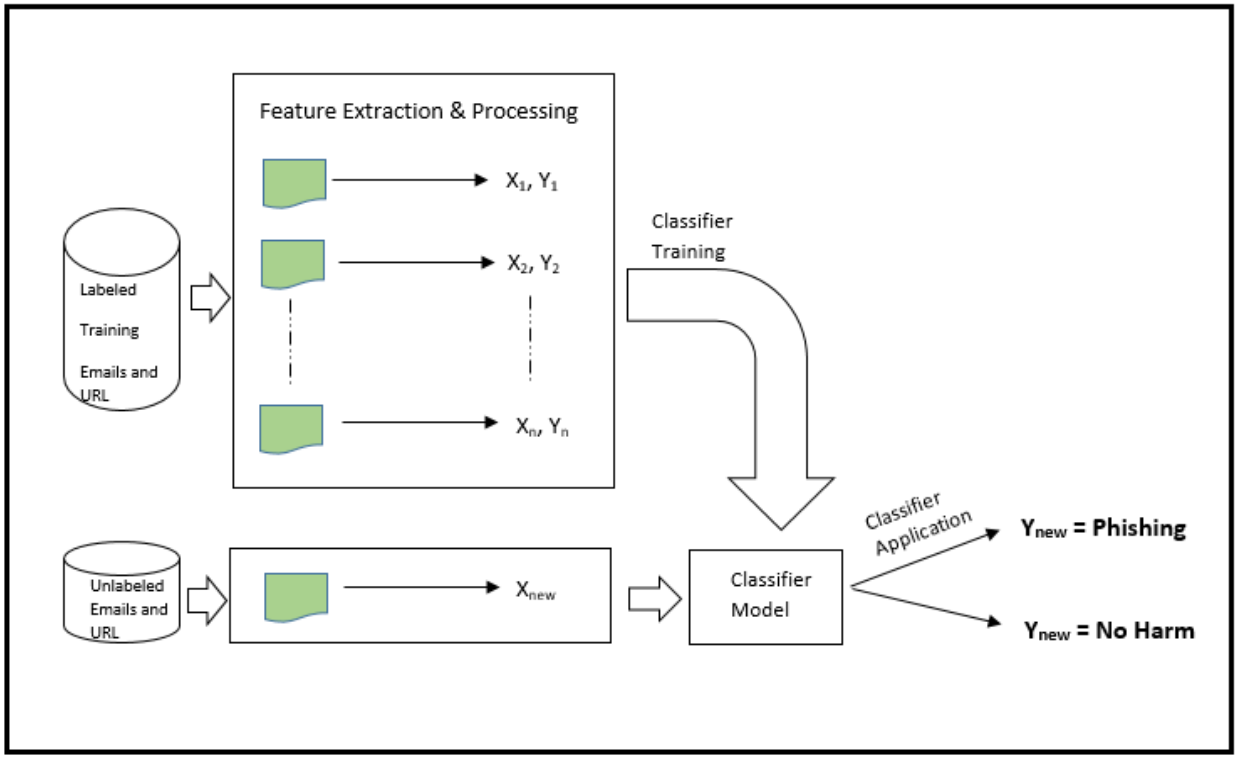

Figure 2-6: Conceptual Framework for the proposed system

The proposed algorithm follows a supervised learning model architecture. Labelled data (URLs and Emails) was first loaded to the system. The favourable features were extracted and processed based on the nature of the data used. The extracted features then formed the basis for the classifier i.e. the classifier model. The unlabelled data (an unknown or new Emails/URLs) were then loaded to the classifier model for classification. In the end, the new content was either classified as harmless (genuine content) or phisher material.

Chapter 3: Research Methodology

3.1 Introduction

The purpose of this research was to develop a dynamic predictive model for detecting phishing attacks. Through the analysis conducted by the study, the various features that influence phishing were identified and evaluated to implement the final solution. The dynamic element came from the envisioned ability of the model to adjust its weights to suit any given context. As such, feature subset selection formed an integral part of the research. Furthermore, because of its dynamic nature, an agile methodology was used to develop the final system (model). This approach also created the necessary conditions for implementing a self-adapting model which over the course of the research became more refined and better with time.

3.2 Research Design

According to De Vaus (2006) research design refers to the general strategy used to integrate the various component of a study to develop a coherent and logical evaluation. It is through this assessment that one addresses the research problem in question by collaborating the different elements of a research. Formally, seen as the blueprints of a study, the elements of data collection, measurements and analysis constitute the biggest part of the research design as they ultimately determine the final results as well as the conclusion of a study. Of note, however, is the research problem which primarily determines the type of research design used.

Based on this definition, a practical problem is highlighted by this proposal where the envisioned study aims to develop a predictive model to detect phishing attacks (Longdom, 2019). An experimental research is inherently defined by the problem statement as the study seeks to venture into the discipline of applied machine learning. To answer the questions asked by the research, systematic experiments are needed. For instance, to determine the optimal number and choice of input features. Experimental trials are therefore going to form the backbone of the research design as machine learning and its affiliated elements, such as deep learning, are highly complex and do not subscribe to other common methods of analysis (Brownlee), 2018. Fortunately, applied machine learning offers the researcher complete control over the experiments thus one can run as many trials as they wish which will further enhance the results of the proposed study. This control over the experiments, specifically, the possible variables in question additionally categories this study as controlled experimental research.

As such, the attributes of controlled experiments will be readily visible in the research. In particular, its subsequent elements of Choose-Feature Experiments and Tune-Model Experiments will be used. The former is applied in instances where a researcher wants to determine the data features to use that is, the input variable which is one of the key objectives in this proposal. In all, the most useful and relevant variables are to determined and used as the inputs for the predictive model. The latter then focuses on fine-tuning a machine learning model through the adjustments of variable and parameters to enhance the estimation skills of the algorithm. It is this improved performance that enables a model classify unknown and unseen data. To meet this need of iterative experiment, an agile and well defined tool is needed. This requirement necessitates the development tool highlighted below.

3.2.1 Prototyping/ Model Development

The proposed model was developed using Pytorch. Pytorch is an open source library that is used to develop machine learning algorithms such as those of natural language processing. It has elaborate features and capabilities having been developed with the needs of research. The library has broadly two high levels of capabilities; one, a Tensor computation that applies strong GPU acceleration such as Numpy and two, deep neural networks which are developed on tape-based auto-diff systems (Pytorch, 2018).

3.2.2 System Development

The proposed model was developed on the basis of the characterising data. Here, phishing features were identified and analysed to give predictive results. This structure of system design can be broadly summarised as a data-driven model owing to the large attribution of data in the implementation process. The model developed, therefore, was determined by the connections made between the state variables such as inputs, outputs and internal factors as outlined by the diagram below (Solomatine & Ostfeld, 2008). Moreover, there were minimal assumptions on the physical behaviours of the model as they were of less importance to the research.

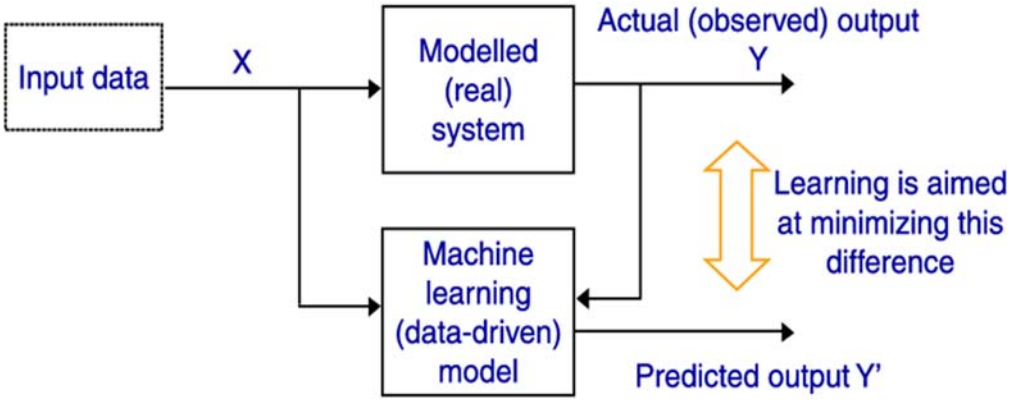

Figure 3.1: Data-driven Modelling (Adapted from Solomatine & Ostfeld, 2008)

Techniques used: Data-driven models use a variety of techniques in their development process. They include fuzzy rule-based systems, genetic algorithms, and neural networks. As previously highlighted, the proposed Phishing detection model used the final technique (neural networks) but with a greater emphasis on deep learning (more than two hidden layers).

Testing and Deploying the Model: The publicly accessed data (PhishTank and ICU repository) was split into several categories. The first being the training data set to train the developed model. Thereafter, a validation data set was used to evaluate the model and finally, a test data set assessed the efficiency of the final model based on its accuracy to predict Phishing attacks. Therefore, the test data was used to test the final model developed.

3.2.3 System Analysis

A series of steps were used to analyse the requirements of the proposed model. Yes, the general needs were understood but to implement a working algorithm, all the functional and non-functional elements needed to be clearly outlined.

First, the requirement determination which included both the functional and non-functional elements. The former, a process and information-oriented model was followed to ensure the algorithm performed the processes it envisioned. It also helped the system have the necessary data for its operations. The latter then held the attributes of performance, speed, and security among others.

Secondly, requirement specification where the questions of what the model should do and how it should behave were at the centre of the analysis.

Finally, a feasibility analysis that tested the workability of the model based on its social impact, utilization of resources, and the ability to meet the initial objectives.

3.2.4 System Design

To define the elements of the model such as architecture, components, interfaces, and modules, various design approaches were used. They included Use case diagrams, sequence diagrams, Data Flow Diagrams and Entity Relation Diagrams (ERDs). Use case diagrams outline the action sets (behaviour) of a system while it interacts with external users. As such, it was an important tool for describing the corporation of the model with external actors such as email users. The sequence diagrams, on the other hand, shows the interaction of objects, in this case, the interrelation between the detection algorithm, user, database and the internet platform (Email). The DFD then portrayed the flow of data across the web and also in the model. DFDs are efficient methods of graphically representing information in a system which were key in this case to understand the limitations of the available containers (Jilani, Usman & Nadeem, 2011). Finally, the ERDs which define the relationships of the various entity sets recorded in databases. This element was key in understanding how user access accounts (e.g. Emails) and the relationship between Login, Email access, and threat tables.

3.3 Target Population

While there are different features that characterise the internet and thus, the properties of detecting phishing attacks, Emails stand out owing to their widespread use by phishers. Predominantly, the intruders use this form of communication to bait users and acquire their data. Thereafter, these emails re-direct the unknowing users into rogue websites (URLs) which completes the attacks. Thus, while a variety of elements were considered by the study, Emails and URLs were the main targets for the study, including the sampling process. This target population was sourced from three publicly available data sources PhishTank archives, MillerSmiles archive and UCI's Machine Learning Repository.

Several attributes characterized the data which in this case was the target population. For one, it came from publicly available dataset. These types of sources were selected because they contain large dataset which are unbiased to any class. Secondly, the target population (data) had webpage address features such as domain name, domain age and other URL items. It is these items that were used to classify a single data label as either phished or legitimate. Finally, the target population was not from any context or situation as established by the age of the datasets selected (they were from 2006 to 2019). This final attribute improved the impartially state of the dataset which further improved the reliability of the data.

Due to the magnitude of the data available (as highlighted in the next section), several statistical techniques were used to obtain an optimal number of records to meet the necessary model's accuracy. Although, the exact number of data labels is not fully specified at this time, the quality of the data labels would ultimately determine the quality of the research. As such, labelling the collected data, as seen in numerous ML studies determined the bounds and the performance of the model (Mendels, 2019). Nonetheless, all the identified data had more than 2,500 data labels as highlighted in UCL ML repository (2019).

Like most other machine learning models, the data was structured using a probability distribution such as p (x, y; Θ). To this end, to evaluate the size of data used (sample) analysis was done on several instances determined at the start of the experiments. Here, the influence of each instance on a learning procedure was analysed which ended up creating a learning curve for the entire dataset (based on the instances, sub-datasets). Each of these sets of data was then fed into the learning model which gave better results by avoiding the limitations of overfitting.

3.4 Data collection

Several publicly available datasets were used in the study to evaluate the developed model. PhishTank archives and MillerSmiles archive were the primary sources of data as they had extensive resources that dated back to the early 2000s. These dataset were then complemented by UCI's Machine Learning Repository which over time has developed significant phishing records from other sources such as Google, MillerSmiles and even PhishTank archives.

Obtaining and Pre-Processing Data: PhishTank contains a large dataset with a wide range of variables (over 2,500 data labels for each individual dataset/segment). Thus, before using its content, an analysis was done on the general record to generate a specialised dataset for the proposed model. Nonetheless, majority of PhishTank records were characterized by URLs from both legitimate and phishers websites. Most of these URLs had been verified by human experts thus were valid to give accurate results.

On its behalf, the UCI dataset had a combination of both Email and URL datasets. A broad distinction was however made having the categories of good and spam emails. Together, these two data sub-segments facilitated the training of the classification model.

3.5 Data Analysis

3.5.1 Data Cleaning

As a preparation steps towards the final classification, data cleaning was done on the PhishTank and UCI datasets. This cleaning process involved the matching and selecting of data files as well as instance cases. Additionally, the selected records were assessed for missing and erroneous data as they posed a huge problem to the final results of the experiments conducted. Erroneous data was discarded either using deletion of replacement techniques (chosen instance(s)). Regression imputation was thus used to replace the deleted instances. Regression imputation replaces missing data with estimation values (Kang, 2013). It therefore, helps to improve the quality of research by minimising the effects of missing and erroneous values. The same procedure (imputation) was also used for missing values. In this case, erroneous data largely emanated from missing data segments and wrong formats of URLs as well as Emails. The same was additionally done for missing values. Eventually, this step (data cleaning) ensured all data points were consistent with the objectives of the study and meet the identified classification criteria.

3.5.2 Data Classification

A predictive analysis characterised this step, however, to facilitate a thorough analysis of the data, two other evaluations were conducted; descriptive and exploratory. Descriptive analysis, quantitatively describes the core features of a dataset particularly, when an experiment is faced with a voluminous amount of data as in this context. The exploratory evaluation, on the other hand, is done to find any unknown, previous relationships in datasets. Finally, the predictive assessment examines both historical and current data (facts) to help make predictions about future outcomes/events.

Broadly, a descriptive analysis was first done based on multivariate statistical analyses. This analysis process helped to establish the relationship between the various measurements (features). In line with these relationships, the exploratory analysis was incorporated in the general evaluation to discover any new connections between the phishing features (Leek, 2013). These connections further helped to redefine the study's questions and those of the future models. Nevertheless, in the end, the predictive analysis was used to measure the right variable which again re-empathizes on the importance of the previous two analyses methods (descriptive and exploratory). The importance (the why) of these data analyses methods was defined by the need to meet the objectives of the research. More accurately, the first method defined the data by

generally describing its elements which laid the foundation of data evaluation. The second approach then provided the key relationships of the features (variables) in use and the last technique provided the means to predict future results based on historical data.

3.6 Research Quality

There exists a number of definitions for the term research quality. These descriptions even go further to highlight what good quality research is and what it is not. However, despite their disparity, they all seem to agree on the following elements. The first being the ability to stand the test of time and peer scrutiny. Secondly, the impact on a particular research field and finally, the contribution to society more so the scientific community (Carlsson, Kettis & Soderholm, 2011). This study envisioned to meet these criteria by applying the utmost standards in its development process. For one, it utilized reputable material (sources) in its analysis such as those in the literature review. Moreover, verified data was used to develop the model which further enhanced the quality of final results. Finally, a validation method, using an errors metric (MSE) was used to assess the quality of the detection process (research model).

3.6.1 Reliability

Once a model is implemented, one has to test its performance. Typically, a dataset is split into two; training and test data thus, the two components help assess how the algorithm performs. The method used in this case was MSE where a new data set (testing data) x was used. All the values in the data set were indexed and labelled accordingly (x_i). To evaluate the model, the x_i was compared with another categorized data y_i. This comparison applied MSE where the error (the difference between f (x_i) and y) was computed. The resultant value was then squared, followed by the averaging of all available values (mean).

Mathematically, this is defined as:

$$MSE = \frac{1}{n}\Sigma_{i=1}^{n}(y - \hat{f}(x_i))^2$$

3.6.2 Validity

Several training instances (epochs) were used in the model. At each instance (iteration), the values will be recorded based on a controlled environment (i.e. variables and features, etc.). The consistency of this value with guarantee the validity of the results obtained.

3.7 Ethical Consideration

The research primarily used publicly available data, a process that only occurred after formal consent was granted by the caretakers of the content. For the UCI ML Repository, any research and publications made using their data requires formal acknowledgment as stipulated in their citation policy. This study met this requirement by citing the necessary content used based on the recommended format. Additionally, the research disclosed all applications of the said data highlighting the scope and duration of the study. In terms of personal information contained in the dataset. The pre-processing stage of this research filtered all personal content present in the database. However, majority of the data sources (PhishTank, UCI ML Repository and MillerSmiles) had redacted all personal information from their dataset before publishing them online. This redacted information included addresses of the victims of the attacks.

Chapter 4: System Analysis and Design

4.1 System Analysis

This chapter offers a general overview of the model based on its design and application. This section outlines the system requirements in terms of; data, functional and non-functional requirements. Moreover, a general description and analysis of the algorithm is given in the end.

4.1.1 Data Analysis

4.1.1.1 Data Collection

Data is always an integral part of a study as it provides the basic asset to test hypotheses and theories. The asset was particularly important for this research as it not only tested the postulations made earlier but also provided the means to predict future outcomes. The model developed would pick out the defining elements of the data (features) to classify raw content. Therefore, the data used for the study needed not only to be accurate (based of the objectives of the study), but also impartial (unbiased to any classification). This requirement meant equal number of data labels for both phishing and legitimate content.

There are various open source databases that hold this form of data which were used for this study. PhishTank archives and MillerSmiles archives are good example of these databases as they hold great reputation in the computing world having collected accurate records over the past two decades. UCI's Machine Learning Repository is an even better data store as it has not only. accumulated the records from the former two repositories but has also performed basic pre-processing activities which makes their dataset more refined. Nonetheless, after acquiring the data from UCI's database a few pre-processing activities were needed on the collected data before being employed in the model.

4.1.1.2 Data Evaluation

Factor Analysis: Factor analysis was conducted to help identify the key elements of the data. These items would form the basis of identifying the system requirements of the study i.e. functional and non-functional requirements.

```
In [3]:  1 #load data
         2 computer_raw_data=pd.read_csv(r"C:\Users\RDK-PC\Phishing-Website-Detection-master\raw_datasets\1000-phishing.txt",header=
         3 #computer_raw_data

In [5]:  1 #Preprocess the data
         2 computer_raw_data.columns

Out[5]: Index(['urls'], dtype='object')
```

Figure 4.1: Loading and Preprocessing Data

After loading and pre-processing the data, missing values were dealt with to avoid errors in the final data analysis process. Generally, there were columns without any value and were thus eliminated.

```
In [6]:
# Dropping missing values rows
computer_raw_data.dropna(inplace=True)

In [7]:
computer_raw_data.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 998 entries, 0 to 997
Data columns (total 1 columns):
urls    998 non-null object
dtypes: object(1)
memory usage: 15.6+ KB
```

Figure 4.2: Dropping Missing Values and Raw Data Info

Finally, the factor analysis and its output

```
In [8]:
computer_raw_data.head()
```

Out[8]:

	urls
0	http://asesoresvelfit.com/media/datacredito.co/
1	http://caixa.com.br.fgtsagendesaqueconta.com/c...
2	http://hissoulreason.com/js/homepage/home/
3	http://unauthorizd.newebpage.com/webapps/66fbf/
4	http://133.130.103.10/23/

```
In [27]:
#Choosing the number of factors
fa = FactorAnalyzer()
fa.analyze(computer_raw_data, 25, rotation=None)
# Check Eigenvalues
ev, v = fa.get_eigenvalues()
ev
```

Figure 4.3: Factor Analysis

The output

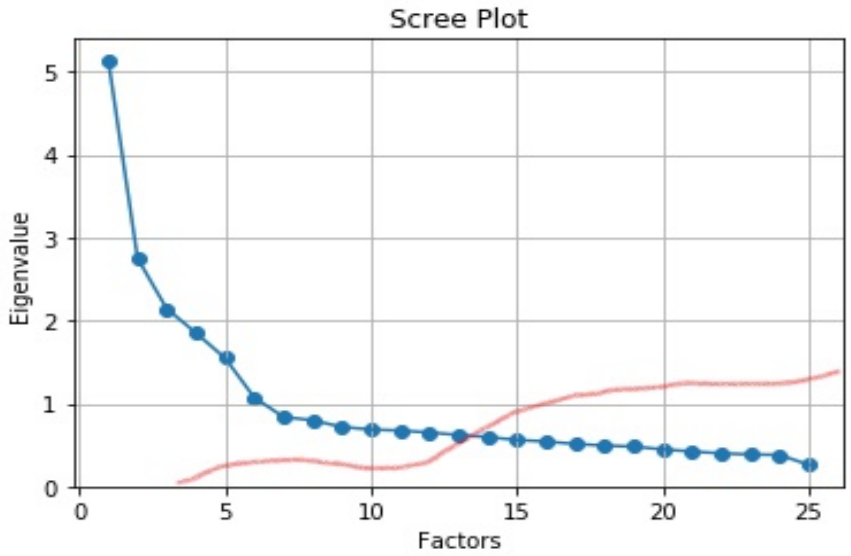

Figure 4.4: Factor Analysis Output

An optimal number of features to use is between 12 and 14. This research settled for 13 features as highlighted throughout the report.

4.1.2 System Requirements and Analysis

Phishing takes on multiple forms depending on the vulnerabilities exploited by the phishers. As highlighted in chapter 2 of this report, various exploits are used to target and steal personal information from users. These exploits form the structures used in the malicious techniques of phishing attacks more so, through phishing emails. Thus, to develop a credible algorithm capable of capturing and preventing phishing attacks, it is important to highlight these exploits and their subsequent controls. This component forms the basis of the requirement analysis which then define the appropriate resources needed to meet the objectives set by this study.

System requirements minimize the efforts of model development by optimizing the available resources, especially time. They expand simple objectives (like those of this study) into practical procedures that eventually achieve the desired system goals. System requirements thus generally reduce the cost of developing systems (Nascenia, 2018). This outcome is important in this study as it is constrained in terms of resources including the budget and time. Therefore, even the requirements set are based on a prioritization basis to achieve the most effective solution. Since an agile development process is employed, these prioritized requirements are refined to achieve practical solutions for the end user. This strict and precise development process necessitates the need to define the term requirement(s) which based on a worldwide acknowledgement of business analysis body is viewed as the following: One, a capability or condition needed to solve a problem to achieve certain objectives. Two, a condition that must be met for solution to satisfy certain specifications. Finally, it is a document representing the said conditions (Semusheva, 2019).

All these definitions of requirement(s) are met by the description of the system requirements used in this research. They are broadly split into two: *Functional* and *Non-functional requirements*. Moreover, to implement the prioritization scheme, they are further individually grouped into the following classes of requirements: Essential, Desirable and Optional.

Table T1: Functional Requirements

No.	Requirement Description	Priority
T1-1	The model should have access to user's emails based on set time and access requirements	Essential
T1-2	Extracting email details	Essential
T1-3	Perform evaluation and phishing check	Essential
T1-4	Store phished emails details	Essential
T1-5	Store non-phished email details	Optional
T1-6	Flag identified phishing email details	Essential
T1-7	Enable threat intelligence analysis to enable to identification of root cause of phishing attacks as well as the prediction of future attacks	Desirable
T1-8	Provide feedback of system performance e.g. presence of false positives	Optional

Table T2: Non-Functional Requirements

ssNo.	Requirement Description	Priority
T2-1	A high usability based on its ease to use	Essential
T2-2	The model should be reliable and provide accurate results	Essential
T2-3	The model must be always available to the end user	Essential
T2-4	The model must maintain the security of the user (their emails etc.)	Essential
T2-5	A high interoperability	Desirable
T2-6	Adjustable preferences on the end user module	Optional
T2-7	Recoverability and maintainability	Essential

4.2 System Design

To further define and understand the model, various pictorial and graphical representations/diagrams were used in the design stage. This section therefore distinctively showcases the use case diagrams, sequence diagrams, activity diagram, data flow diagrams, class, diagram and the entity relation diagram.

4.2.1 Use Case Diagram

Careful modelling of system is crucial to obtain the correct and most efficient model architecture. Use case diagrams facilitate this process by defining the actions and behaviours of a

system as it interacts with some external parties (users). Use case diagrams are therefore high level requirement analysis of systems or models that are developed to capture its dynamic view. Largely, they are used to capture requirements and functionalities of systems as described by various use cases. This role subsequently helps to identify the external and internal agents engaging with a system (or model in this case) (Miloudi & Ettouhami, 2018). These agents are also sometimes called *actors* as depicted in the diagrams below.

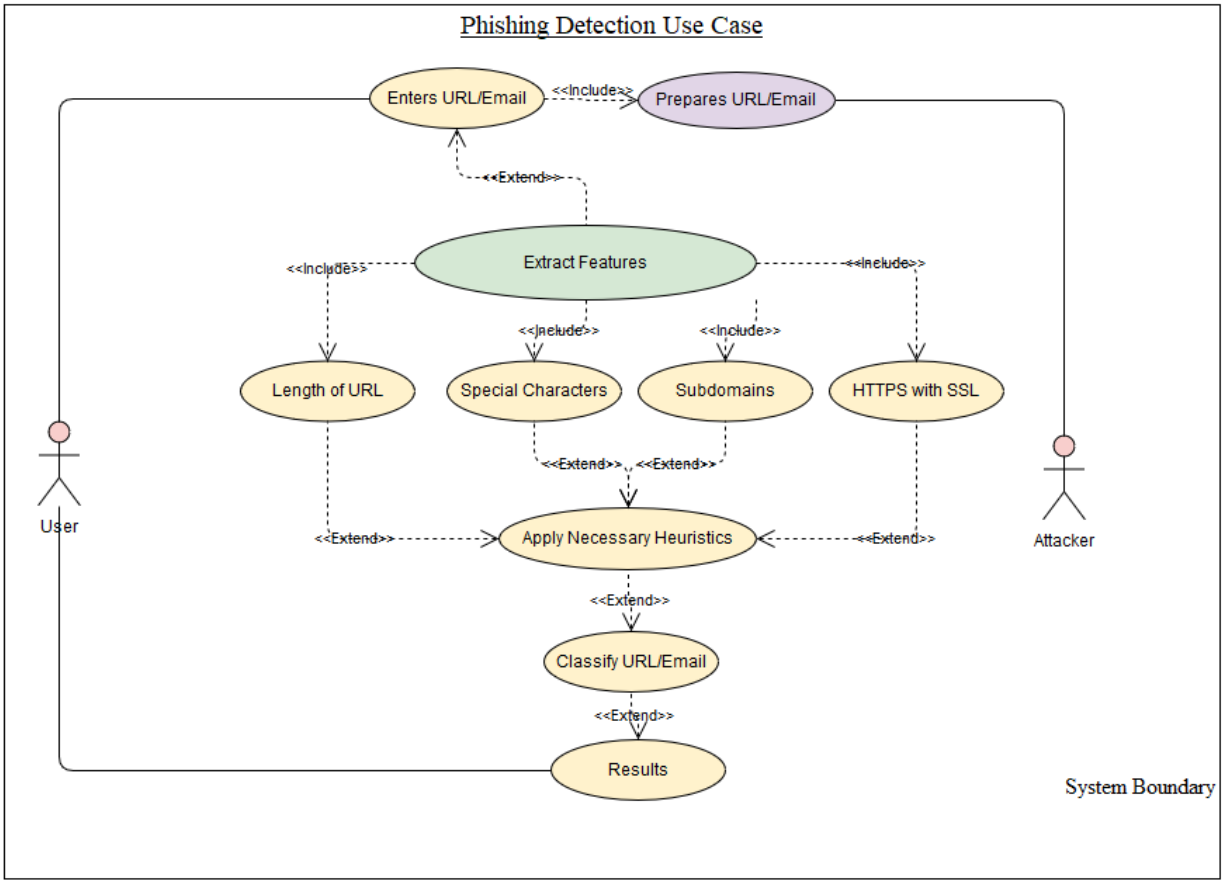

Figure 4.6: Use Case Diagram

4.2.3 Sequence Diagram

The sequence diagram below highlights the flow of actions, events and messages across the model. Sequence diagrams are commonly used to represent these functions as well as to design, document and even validate a system's architecture. They also describe the interfaces and

operational logics based on the flow of actions for a certain set scenario. As such, they also provide a dynamic view of the model and its behaviour.

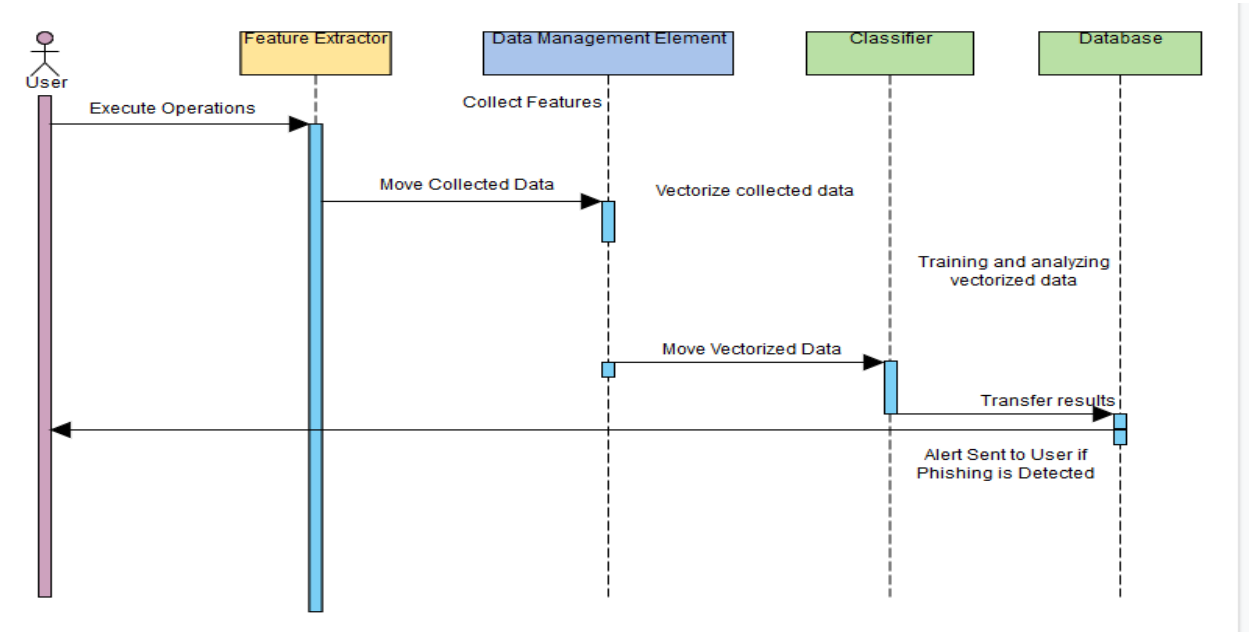

Figure 4.7: Sequence Diagram

4.2.4 Data Flow Diagrams

Data flow diagrams (DFD) are an efficient way of simplifying design management and the decision support of systems. They also simplify design work as well as optimise time by reducing coding routines that require lots of time to develop and execute.

4.2.4.1 Level 0

Figure 4.8 shows an abstract view of the model by highlighting the general system. Majorly, this DFD shows a single process (the model) and its interaction with external entities. Level 0 DFD are commonly called *context diagrams* are they generally represent an entire system outlined by input and output data.

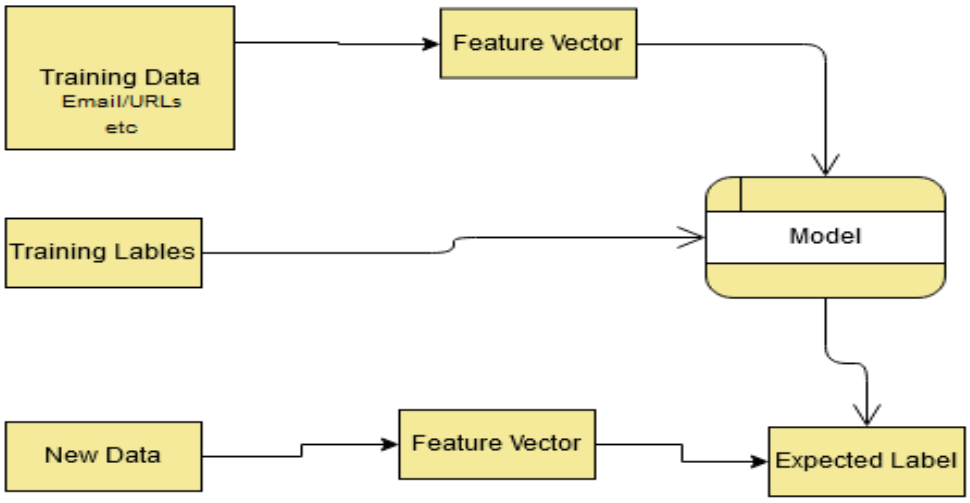

Figure 4.8: Level 0 DFD

4.2.4.2 Level 1

Figure 4.10 is a decomposition of the level 0 DFD (above), where more specific interactions are defined. An even deeper description of the entities used is provided to meet the objectives of the model.

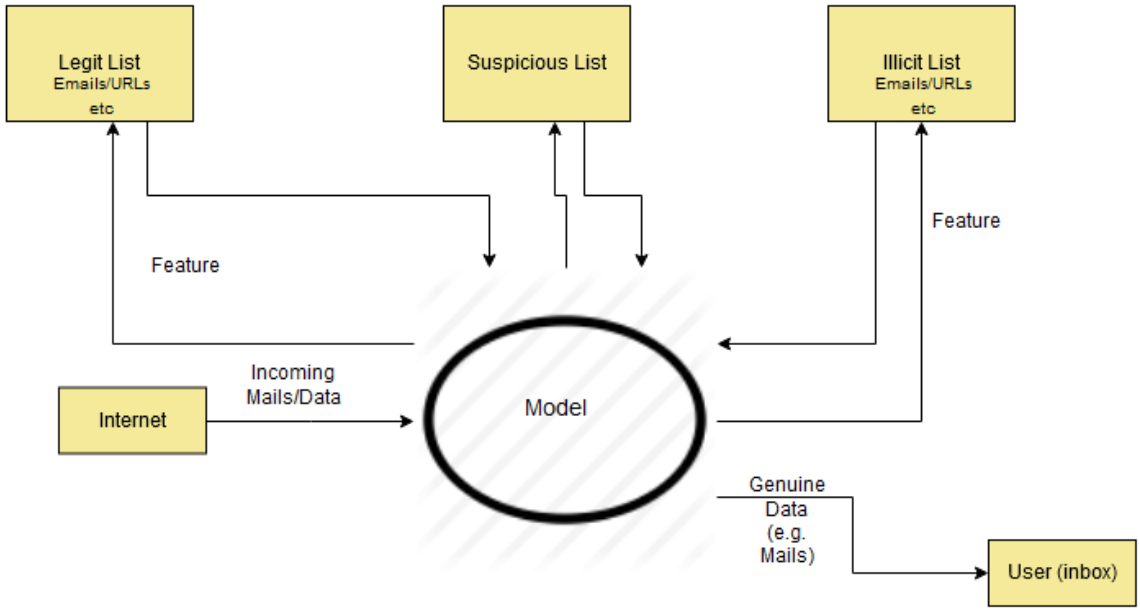

Figure 4.9: Level 1 DFD

4.2.4.3 Level 2

Figure 4.11 shows a deeper outline of the model where it further decomposes the roles of the model based on an interaction of a user and a device (which is connected to the internet).

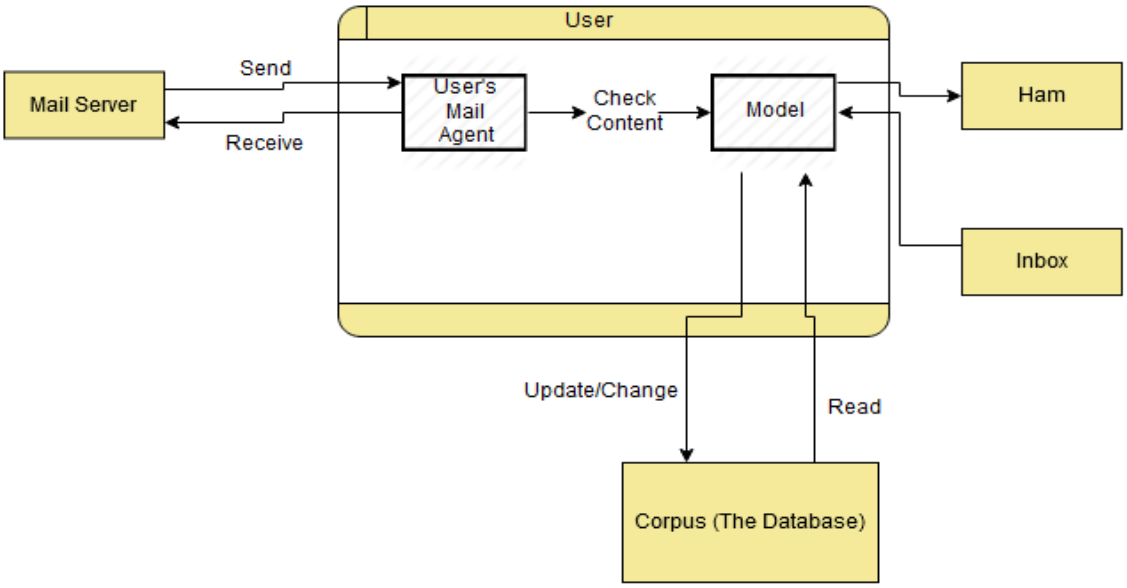

Figure 4.10: Level 2 DFD

4.2.5 Activity Diagram

The models operations are in this diagram defined by a flow of events. Thus, figure 4.12 defines a sequence of actions as a component of the process flow. Like most other activity diagrams, it models a sequence of events to outline the process flow of actions and their results (Bhattacharjee & Shyamasundar, 2009). There is a greater focus on the work performed during the implementation of operations as well as the activities of use case scenarios or objects in general.

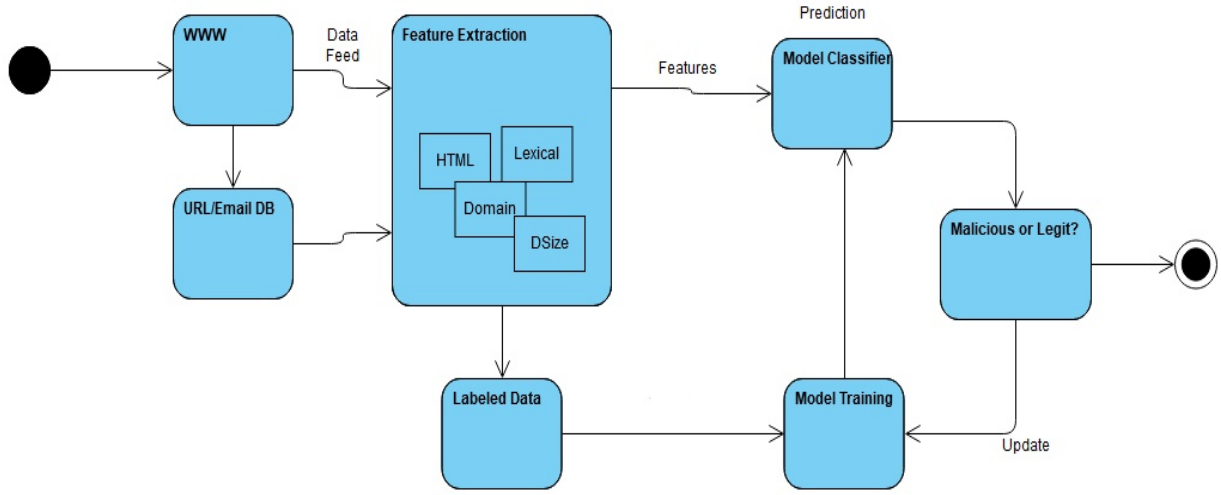

Figure 4.11: Activity Diagram

4.2.6 Class Diagram

Figure 4.13 defines a structure diagram that shows classes, operations, attributes and associations of the model. Class diagrams highlights artefacts (classes) that create objects based on common (shared) operations, relationships and attributes (Alhumaidan, 2012). The same is outlined by the figure below.

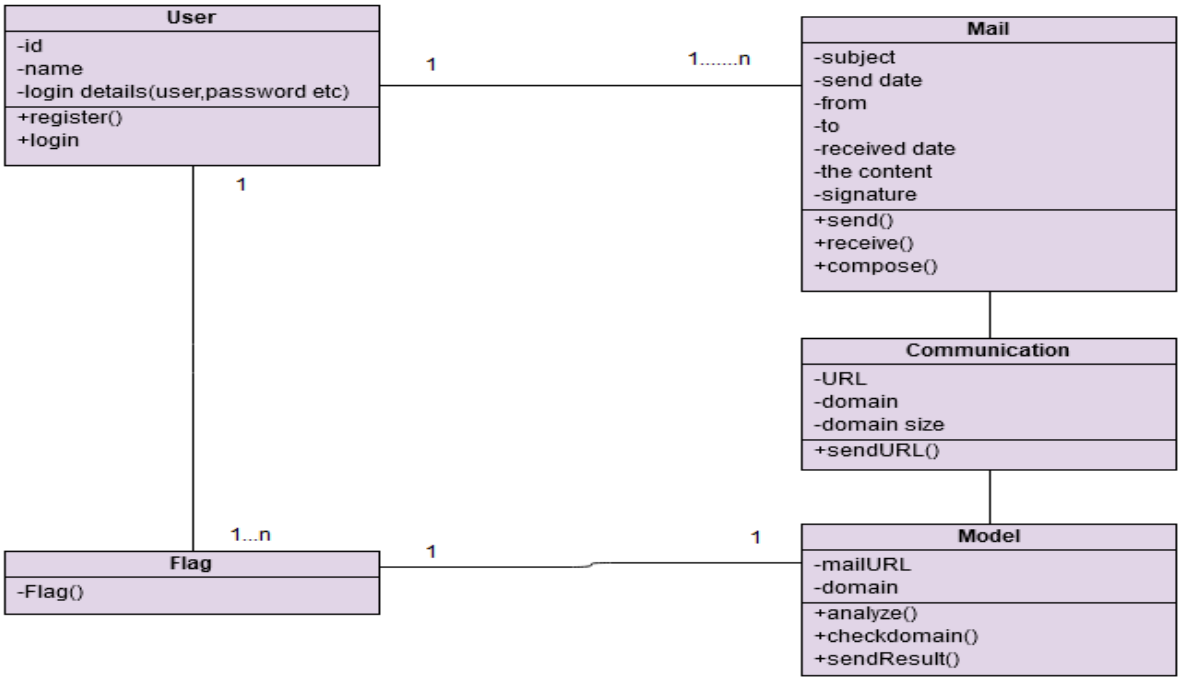

Figure 4.12: Class Diagram

4.2.7 Entity Relation Diagram

Figure 4.14 describes information using a wide range of entities, attributes and relationships. Entity Relationships (ER) are high level conceptual models designed to facilitate the development of databases. The ERD diagram below therefore forms the blueprint of the database used in the detection model.

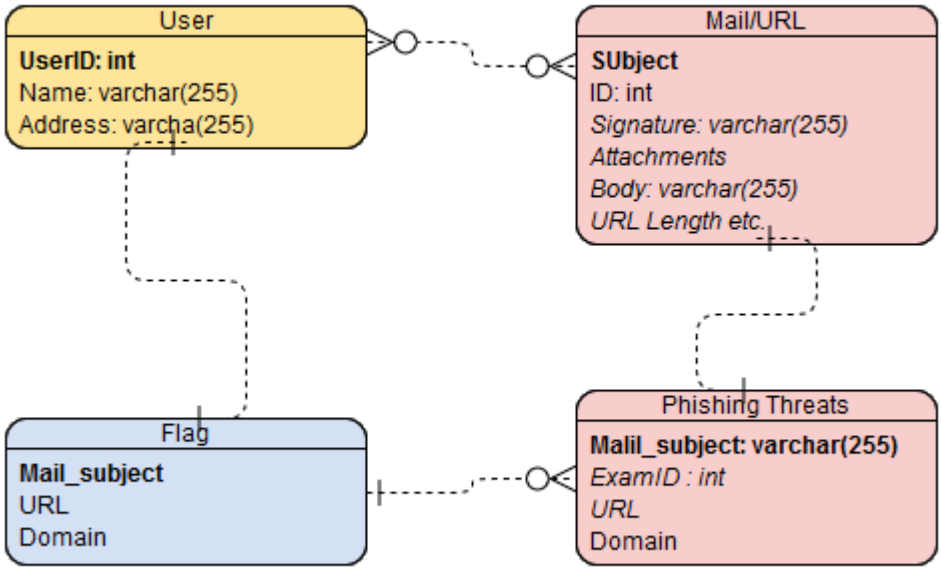

Figure 4.13: Entity Relation Diagram

4.2.8 Database Schema

Following the abstract view of the database in figure 4.14 (ERD). The database schema below, figure 4.15, outlines the metadata that defines the relationships formed between the objects and information of the detection model.

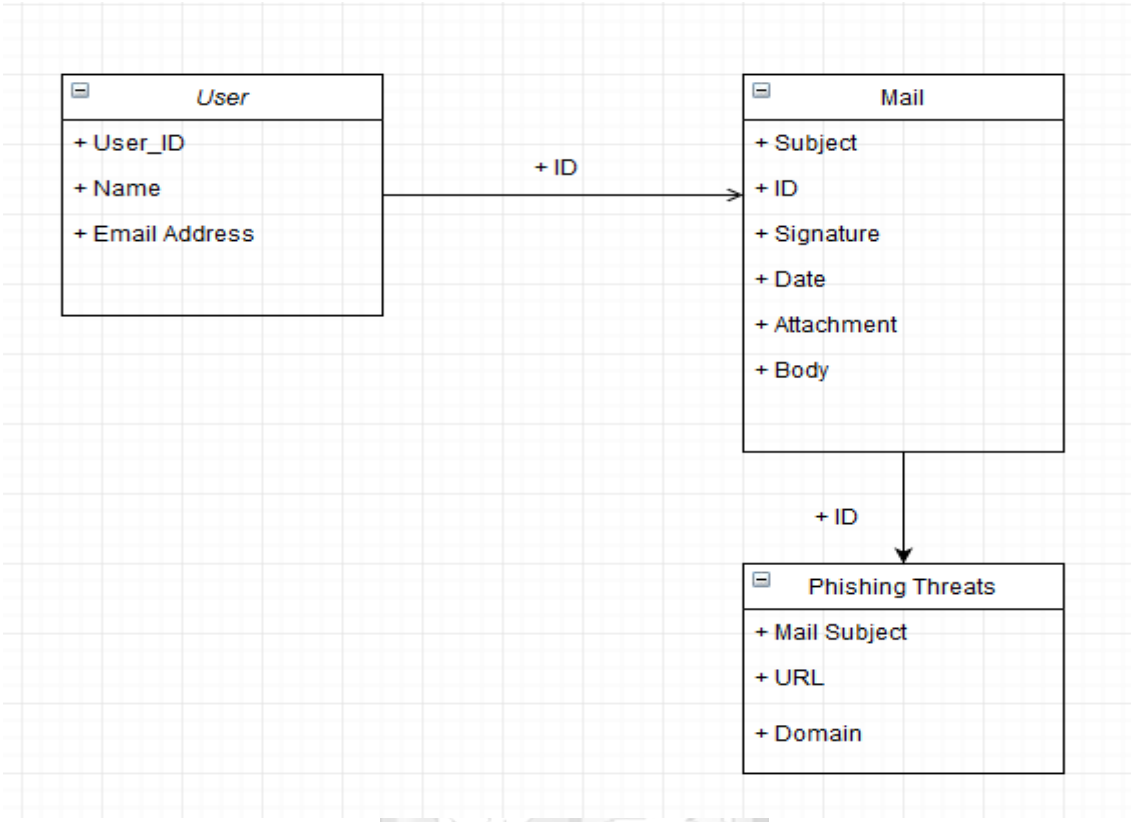

Figure 4.14: Database Schema

Chapter 5: Implementation and Testing

5.1 Introduction

This section highlights the steps used to implement and test the proposed model. First, a description of the model is given. This initial sections outlines all the major phases of phishing detection. Thereafter, the actual implementation is provided having all the accompanying codes. The final section then showcases the tests performed on the model.

5.1 Description of the Algorithm

Figure 5-0 provides a basic description of the model where it outlines the steps for URL evaluation and phishing detection. Generally, the model's processing stages include: Pre-processing of data (URLs), feature extraction/selection (wrapper approach as defined before), feature assessment, classification and eventually the evaluation of the classification results.

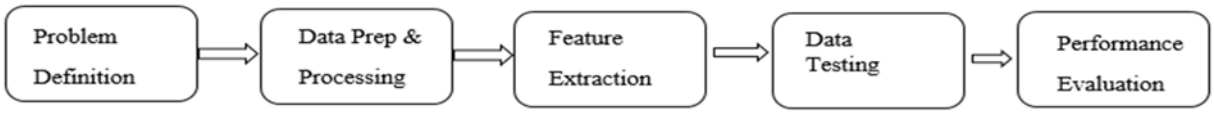

Figure 5.0: Phishing Detection

To optimise the results, the Random Forest algorithm is used as it holds powerful knowledge representation attributes as well as a strong reasoning mechanisms (Hamid, Abawajy & Kim, 2013). Additionally, the algorithm is widely used owing to its simplicity which provides the study a large support community. Random Forest algorithm is however not only used due to its simplicity but also because of its manipulating capabilities which extend to the tokens and affiliated probabilities directed by user's classification needs. Therefore, any classification decisions and empirical performances can be adequately tested and adopted using the algorithm.

Mathematics behind Random Forest:

Random forest as a predictor consists of a group of randomized regression trees which can be defined as $\{r_n(x, \theta_m, D_n)$, where m >=1$\}$. θ_m which can take the values $\theta_1, \theta_2,$ and above, defines the various outputs of the randomized variable θ. A combined regression estimation can be derived from these random elements.

$$\acute{r}_n(X, D_n) = E_\theta[r_n(X, \theta, D_n)]$$

Where: E_θ is the expectation with relation to the random parameter, considering the input X and a dataset D_n.

A coordinate of $X = (X^{(1)},\ldots, X^{(d)})$ can be selected at each node. After selecting the coordinate, a split is done at the midsection of the selected side. From this selection, all randomized trees ($r_n(X, \theta)$), have an average output of Y_i which corresponds to the vectors X_i. Simply, therefore having $A_n(X, \theta)$ as a rectangular cell of random partition having X gives.

$$r_n(\mathbf{X}, \Theta) = \frac{\sum_{i=1}^{n} Y_i \mathbf{1}_{[\mathbf{X}_i \in A_n(\mathbf{X},\Theta)]}}{\sum_{i=1}^{n} \mathbf{1}_{[\mathbf{X}_i \in A_n(\mathbf{X},\Theta)]}} \mathbf{1}_{\mathcal{E}_n(\mathbf{X},\Theta)}$$

With event $\mathcal{E}n(X, \theta)$ being defined as:

$$\mathcal{E}_n(\mathbf{X}, \Theta) = \left[\sum_{i=1}^{n} \mathbf{1}_{[\mathbf{X}_i \in A_n(\mathbf{X},\Theta)]} \neq 0\right]$$

Therefore, the final random forest regression expression takes the following from:

$$\bar{r}_n(\mathbf{X}) = \mathbb{E}_\Theta\left[r_n(\mathbf{X}, \Theta)\right] = \mathbb{E}_\Theta\left[\frac{\sum_{i=1}^{n} Y_i \mathbf{1}_{[\mathbf{X}_i \in A_n(\mathbf{X},\Theta)]}}{\sum_{i=1}^{n} \mathbf{1}_{[\mathbf{X}_i \in A_n(\mathbf{X},\Theta)]}} \mathbf{1}_{\mathcal{E}_n(\mathbf{X},\Theta)}\right]$$

Description of Random Forest

Random Forest is defined as a collection of Classification and Regression Trees (CART). This definition adopted by Breiman et al in 1984 highlights the overall working procedure of the algorithm. Random Forest is usually trained on datasets of the same size as the general training sets, known as bootstraps, which are also developed from random resampling of data training sets. When the tree is finally constructed, bootstraps which do not have any record of the original dataset (out of bag samples (OOB) are then used to test the system. The error rate from the classification of all the test set is taken as the OBB estimate and the generalization error. Empirical evidence gathered by Breiman (1996) suggested that for a range of classifiers, the accuracy of OOB error matches that of a test set having the same size as the training set. It is therefore, not necessary to use a separate test set when having an OOB estimate. Eventually, when classifying a new input, each single CART tree elects one class and the forest final results (prediction) is based on the popularity of votes.

Random Forest uses a certain set of rules to grow, combine, test and process the tree. The algorithm is often robust to overfitting and is viewed as the most stable method of classification particularly, in the presence of high dimensional parameters and outliers. The idea of variable importance is an implicit feature selection process performed by the algorithm which is based on a random subspace of the methodology. The concept of selection is also assessed by Gini impurity criterion index. From a machine learning background, the Gini index is an overall measure of prediction power of variable in a classification process that is based on the principles of impurity reduction. As a measure, it is non-parametric which means it does not rely on any data coming from a particular type of distribution. In a given binary split, the Gini index having n nodes is calculated as shown below:

$$Gini\,(n) = 1 - \sum_{j=1}^{2} (p_j)^2$$

Where, the pj represents the relative frequency of classes (j) in the n node.

Based on this description, improving the Gini index results in better binary node splits. That is, a low Gini index (decreasing this value) means that a given predictor feature has greater role in dividing (partitioning) the data into two classes. The Gini index, therefore can be used to rank the importance of attributes in a classification problem. This functionality is used in this study where the value of each URL feature is tested and its importance to the research is then outlined.

5.1 Implementation

Python was used to implement the model using a series of functions and classes. The general goal was to collect data, pre-process this data, extract the necessary features and use these features to identify three different categories of URLs. The three categories were legitimate, phishing and suspicious. From these groups, a model was then developed to predict the category a new URL belonged to, based on the training done by the data collected (training labels). This section, highlights the implementation process of this project.

The final classification was done using Random Forest, even though other algorithms were tested in the training process. Random Forest in the end proved to be more efficient achieving higher levels of accuracy and had minimal resource requirements. The former benefits proved important and were particularly influenced by the algorithm's fundamental design. Random forests are integrated tree predictors where each tree works on the premise of values through random vectors which are sampled independently while having the same distribution of all the trees

involved (Breiman, 2001). This technical structure was originally developed by Leo Breiman who in the early 2000s showed significant gains in classification and regression accuracy using ensemble trees. What makes Random forest even more practical and compatible with this project is their foundational development which is influenced by geometric feature selection, random subspace methods and random split selection approaches (Biau, 2012). These pre-requisite elements facilitate the most vital aspect of this project which is feature selection having collected raw data.

Back to the implementation process, the model was implemented using Python programming language. Several libraries were used to meet this objective and are well highlighted in the implementation journal (Jupyter Notebook) attached to this report. Three main steps were followed: *Feature Extraction, Pre-processing* and *The Model*.

5.1.1 Stepwise System Implementation

First, a series of packages were installed to implement the necessary functions of the model. They included; *pandas, whois, datetime* and *socket* etc. Thereafter, raw data, sourced from UCI Repository (which combines content from Google, MillerSmiles and even PhishTank archives) was loaded into the programming environment. It is important to note that the raw data was split into two categories; legitimate and phishing URLs, each class having 1000 samples.

Feature Extraction: A class name FeatureExtraction was created having as series of functions to identify and source features such as Domain, Long URL, at (@) symbol and many others.

```
cd = None
class FeatureExtraction:
    def __init__(self):
        pass

    def getProtocol(self,url):
        return urlparse(url).scheme

    def getDomain(self,url):
        return urlparse(url).netloc

    def getPath(self,url):
        return urlparse(url).path

    def havingIP(self,url):
        """If the domain part has IP then it is phishing otherwise legitimate"""
        match=re.search('(([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|2
                        '((0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\/)'  #I
                        '(?:[a-fA-F0-9]{1,4}:){7}[a-fA-F0-9]{1,4}',url)     #Ipv6
        if match:
            #print match.group()
            return 1            # phishing
        else:
            #print 'No matching pattern found'
            return 0            # legitimate

    def long_url(self,url):
        """This function is defined in order to differntiate website based on the length of the URL"""
        if len(url) < 54:
            return 0            # legitimate
        elif len(url) >= 54 and len(url) <= 75:
            return 2            # suspicious
        else:
            return 1            # phishing
```

Figure 5.1: A Small Snap shoot of Features Extraction Code (entire code is attached to the report)

The whole idea in this segment is to use common attributes of these address components to identify a link as phished or legitimate. For instance, it is common to have the at (@) symbol in rogue URL as commonly seen in emails. Thus the presence of the at (@) symbol in a URL was grounds to classify it as phishing. In the end, the following features were selected and implemented by the code. In the end, new datasets were developed having the extracted features which were saved in csv format.

```
#The Feature in Use:
protocol = []
domain = []
path = []
having_ip = []
len_url = []
having_at_symbol = []
redirection_symbol = []
prefix_suffix_separation = []
sub_domains = []
tiny_url = []
abnormal_url = []
web_traffic = []
domain_registration_length = []
dns_record = []
statistical_report = []
age_domain = []
http_tokens = []
#google_index = []
```

Figure 5.2: Features Used

Pre-processing: Before pre-processing, it is good practice to view the state of the data being used. Thus, both data frames (legitimate and phishing) were viewed.

	Domain	Having_@_symbol	Having_IP	Path	Prefix_suffix_separation	Protocol	Redirection_//_symbol	Sub_domains
0	asesoresvelfit.com	0	0	/media /datacredito.co/	0	http	0	0
1	caixa.com.br.fgtsagendesaqueconta.com	0	0	/consulta8523211 /principal.php	0	http	0	1
2	hissoulreason.com	0	0	/js/homepage /home/	0	http	0	0
3	unauthorizd.newebpage.com	0	0	/webapps/66fbf/	0	http	0	0
4	133.130.103.10	0	1	/23/	0	http	0	2
5	dj00.co.vu	1	0	/css/	0	http	0	0

Figure 5.3: Initial View of Data Frames

The single most important activity of the pre-processing stage was merging of the two data frames in preparation for the model. This action was efficiently conducted more so, considering the two data frames had similar column identifiers. After merging, some columns were found to be irrelevant as they had zero contribution to the classification process. They include things like the numbering of the rows and some features such prefix-suffix-separation which had null values. Thereafter, the new dataset (single, combined frame) was reshuffled as the environment had combined the previous two data frames sequentially (1000 rows of legitimate and then phishing) which could have led to a biased results(model) if trained with the combined dataset. It was only after the reshuffling process that the new dataset was split into training and test classes. A split that followed the 70:30 rule.

Verifying the split. It must be in equal proportions for the two classes.

Phishing - 1

Legitimate - 0

```
1  labels_train.value_counts()
```

```
0    726
1    684
Name: label, dtype: int64
```

```
1  labels_test.value_counts()
```

```
1    314
0    291
Name: label, dtype: int64
```

```
1  #The split is almost equal for both training and testing.
2  #Therefore, we can create the model to execute the data
```

Figure 5.4: The Final Dataset Split

The Model: Random forest classifier was adopted for this model and used to group the dataset having extracted the features, combined and reshuffled the data frames.

```
1 random_forest_classifier.fit(data_train,labels_train)
```

```
C:\Users\RDK-PC\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The default value of n_estimator
s will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)

RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

Figure 5.5: The Model

5.2 Testing

Tests were conducted in two separate phases – the first was to validate the accuracy of the algorithm for the final detection system. The second phase was done on the final model (one with possible deployment to the end user). The former test was conducted using the testing dataset that had been set aside during the data preparation/analysis stage of developing the model. The latter tests were then executed using random website links sourced from the web.

5.2.1: Testing the Algorithm

Following the split,

```
In [21]:
1 from sklearn.model_selection import train_test_split
2 data_train, data_test, labels_train, labels_test = train_test_split(urls_without_labels, labels, test_size=0.20, random_s
```

```
In [22]:
1 #Showing the outcome
2 print(len(data_train),len(data_test),len(labels_train),len(labels_test))

1612 403 1612 403
```

```
In [23]:
1 labels_train.value_counts()
2
3 #labels_train[labels_train == 0].count()
4 #labels_train[labels_train == 1].count()

Out[23]: 0    808
         1    804
         Name: label, dtype: int64
```

```
In [24]:
1 labels_test.value_counts()

Out[24]: 0    209
         1    194
         Name: label, dtype: int64
```

Figure 5.6: Splitting the Dataset

The prediction labels for the model were established and the output was as follows

Predictions

```
In [25]:  1  prediction_label = random_forest_classifier.predict(data_test)

In [27]:  1  print(prediction_label),print(list(labels_test))
```

```
[0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0
 1 1 0 1 0 0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 1 1 1 0 0 1 0 0 0 0 0 0 1 0 1 0
 1 0 1 1 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 1 0 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0
 1 1 1 0 0 1 0 0 1 1 0 1 1 0 0 1 0 1 1 1 1 0 1 1 0 1 0 0 1 0 0 0 0 1 1 1 1 0
 0 0 1 1 0 0 1 0 1 0 0 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 0 1 0 1 1 0 0 1 0
 1 0 1 1 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 1 0 0 1 1
 0 1 1 1 1 0 0 0 1 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 1 0 1
 1 1 0 1 0 1 0 1 0 0 0 1 0 1 1 1 0 0 1 0 1 0 1 1 1 0 1 0 1 1 1 0 1 0 1 0 1
 0 1 0 1 0 1 1 1 1 1 0 1 1 1 1 0 0 1 1 0 1 1 1 0 0 0 0 1 1 0 1 1 0 0 0 0 0
 1 0 0 0 0 0 0 1 1 1 0 1 0 1 0 1 0 0 1 0 0 1 1 1 0 0 0 1 1 1 1 0 1 1 0 0 1
 1 1 0 0 0 0 0 1 0 1 0 1 1 0 0 1 0 0 0 0 1 0 0 0 0 1 0 1 0 1 1 0 0 0 1 1 1 0
 0 0 0 0 1 1 1 0 0 1 1 0 0 1 1 1 1 1 0 1 1 1 0 0 0 1 0 0 1 0 1 0 1 0 0 0 1 1 1
 1 1 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0 0 0 1 1 1 1 1 1 1 0 0 1 1 0 1 0 0 1 1 0 0
 1 0 1 1 0 0 1 1 0 0 1 0 1 0 1 0 1 1 1 0 0 1 0 1 0 1 0 0 0 1 1 1 1 0 0 1 0
 1 0 0 1 1 1 1 0 0 0 1 0 0 0 0 0 1 1 0 0 1 1 1 0 1 0 0 0 0 0 0 1 1 1 0 1 0 0
 1 1 1 0 1 1 1 1 1 1 0 1 0 0 0 0 1 0 0 1 0 0 1 1 0 1 0 1 1 1 1 1 0 1 1 0 0 1
 1 0 0 0 1 1 1 0 1 1 0 0 1]
```

Figure 5.7: Prediction Labels

Thereafter the confusion matrix was ran to establish the initial accuracy of the algorithm.

Confusion Matrix

```
In [28]:  1  from sklearn.metrics import confusion_matrix,accuracy_score
          2  cpnfusionMatrix = confusion_matrix(labels_test,prediction_label)
          3  print(cpnfusionMatrix)
          4  accuracy_score(labels_test,prediction_label)
```

```
[[257  34]
 [ 68 246]]

Out[28]: 0.8314049586776859
```

Figure 5.8: Random Forest Accuracy

Of all the algorithms considered, random forest gave the highest prediction accuracy, about 83 percent, hence its choice. Several steps were taken to try and improve this accuracy, including, changing the features, increasing the number of data labels and finally adjusting the attributes of the algorithm itself (trees, maximum depth etc.). The first two items of this modification are attached as appendix as they yielded minimal results. Also, in some instances, they led to lower accuracy levels. The last step (adjusting the algorithm's attributes), however, led to improved accuracy. Following several alterations, figure 5.9 highlights the optimal results obtained.

```
In [29]: 1 #Improving the accuracy by giving a max_depth and the number of trees.

In [30]: 1 custom_random_forest_classifier = RandomForestClassifier(n_estimators=500, max_depth=20, max_leaf_nodes=10000)

In [31]: 1 custom_random_forest_classifier.fit(data_train,labels_train)

Out[31]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                     max_depth=20, max_features='auto', max_leaf_nodes=10000,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, n_estimators=500, n_jobs=None,
                     oob_score=False, random_state=None, verbose=0,
                     warm_start=False)

In [32]: 1 custom_classifier_prediction_label = custom_random_forest_classifier.predict(data_test)

In [33]: 1 confusionMatrix2 = confusion_matrix(labels_test,custom_classifier_prediction_label)
         2 print(confusionMatrix2)
         3 accuracy_score(labels_test,custom_classifier_prediction_label)

         [[257  34]
          [ 62 252]]

Out[33]: 0.8413223140495868
```

Figure 5.9: Improving the Algorithm's Accuracy

As observed in figure 5.9, the algorithm's variables; max_depth and number of trees were changed to specific values to optimize the results (max_depth = 20 and max number of leaf nodes = 10000).

Although the features of the model had been selected using the factor analysis process and further optimized at the implementation stage, their importance was also tested. Here, the research tried to establish the importance of each feature to the final results obtained by the model. The results were as shown below by figures 5.10 and 5.11.

```
***Feature ranking: ***
Feature name : Importance
1 URL_Length    :  0.21423689987092182
2 web_traffic    :  0.18650776802479382
3 statistical_report    :  0.14478064012142144
4 age_domain    :  0.10029618071055392
5 Sub_domains    :  0.08687722067040515
6 domain_registration_length    :  0.07457599992464341
7 dns_record    :  0.06588413694096013
8 tiny_url    :  0.05235770029231327
9 Prefix_suffix_separation    :  0.05100802733847292
10 Having_IP    :  0.008531469712408875
11 Having_@_symbol    :  0.007002889731583401 5
12 Redirection_//_symbol    :  0.006535308354691134
13 http_tokens    :  0.0014057583068308472
```

Figure 5.10: Features Importance

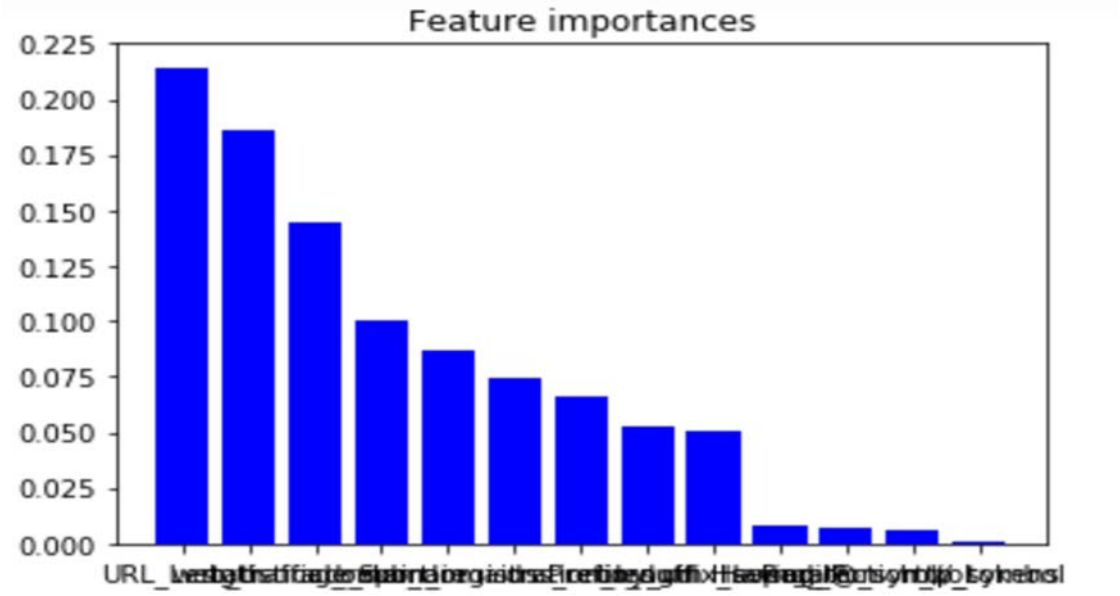

Figure 5.11: Features Importance Chart

In the end, URL Length was the most important feature in the classification process.

Chapter 6: Discussions

6.1 Overview

This section tries to consolidate the findings of this research and compares them with the objectives stated at the start of project. Largely, this chapter tries to evaluate whether the aim of this study was met and if the research questions were answered. Moreover, this section also tries to explain the key items uncovered by the research as well as any anomalies encountered.

6.2 Discussion of Findings

The original aim for this research was to develop a machine learning model to detect phishing attacks. The model needed to employ the inherent structures of web pages and in particular their URLs. Emails are used to deliver these URLs thus were also considered in the implementation process. But since phishing can also occur through baited links in legitimate website, a greater emphasis on the actual links (the resource used to steal information) of phishing attacks was given. Furthermore, to achieve the aims for this research, certain objectives and questions were set which are first analysed in this section before any further discussion.

The first objective wanted to classify the structural attributes of phishing attacks to develop the basis of detection. Phishers operate by compromising legitimate domains to create rogue websites. They even develop scripts to redirect users to malicious server where a user's data can be downloaded. Their actions is also often confined to certain regions through the use of IP filters which are elements that operate at the TCP/IP stack (Namasivayan, 2017). Broadly, therefore, they use the complete connectivity framework to compromise system which outlines the structural characteristics of phishing attacks to the actual features of internet/website connectivity. Starting with URL design, DNS records and even the whois items (domain specifics such as registration date, current register etc.) among others. These structural elements are then combined with persuasion techniques of impersonation, duping and cloning among others to steal data from users. This explains why user's naivety and ignorance are the biggest vulnerabilities exploited by phishers, which was the second objective of the study. The characteristics of website as mentioned above also forms the basis of building the model as they are the ones extracted to form the features of the algorithm used (objective 3).

Following a rigorous analysis of the phishing attacks and their characteristics, this research settled on detecting phishing attacks from webpage access. Essentially, URLs, either sent via emails or conveniently attached as ads in legitimate websites, were evaluated to verify their

legitimacy. This phishing website content was obtained from UCI Machine Learning Repository which over the years has consolidated a large database of phishing attacks. This helped develop a robust feature categories as the practicality of the final algorithm used depended on the features employed. From the experiments conducted, these features were majorly consistent although a few scams used directory generation to develop unique paths for each baited users. It was therefore, easy to detect most phishing links using the various features extracted at the first stage of the model development. The combination of multiple features and not one (like in most detection models) further helped improve the adaptability of feature detection. It is commonly known that URL features are not entirely robust in predicting phishing incidences involving shortened URL operations (Namasivayan, 2017). Thus, by employing other features beyond URL elements such as DNS, a greater robustness is achieved including a greater detection accuracy.

After comparing the work of this research with that done by other researchers such as Waleed Ali (2017) and Weiping Wang et al (2019) which greatly relied on URL features, the study at hand extracted other more relevant features for phishing detection. The model developed thus had an even higher level of robustness that could even detect spam content typically engineered to evade security protocols through well-crafted URLs. In the end, the combination of URL features, IP and DNS attributes seem to give better results. Website popularity features as well as whois were also employed although they seemed to introduce feature set biases which was a significant problem faced by this study.

Chapter 7: Conclusion, Recommendations and Future Works

7.1 Conclusion

This research presented many opportunities to improve the existing algorithm for detecting phishing attacks. Extensive work was conducted on machine learning environments using Python programming language to implement various classification algorithms. The same language was also used to identify the crucial features of phishing attacks. It is through these features that the model for detecting phishing attacks was developed. The features were however constantly adjusted because the phisher methods were found to be highly dynamic than initially thought. Some intrusion could even adjust their properties to suit the detection model in place. As such, a dynamic detection process was also required to cope with the ever changing menace of phishing attacks.

The research began identifying as set of objectives that could help identify phishing attacks. Among these objectives was the classification of the structural characteristics of phishing attacks (content). While a range of attributes were found with regards to these elements, they majorly followed the existing structures of online content. In particular, phishers developed their baiting systems (websites and URLs) using the existing building blocks of legitimate websites. As such, this objective could be easily fulfilled by highlighting the properties of legitimate websites and identifying any deviation from the norm. This objective eventually helped develop the final model, as the attributes of URLs and other affiliated elements of websites were used to code the classification algorithm. Here, after pre-processing data, feature extraction was conducted to identify the defining elements of a website, similar to what was done to the training dataset.

There were however, challenges to this procedure as many attributes/features existed, each with its own dynamics. Moreover, some were not useful to the final system as they even reduced the accuracy of the model. Finding an optimal balance was thus the ultimate challenges of the research. Additionally, a lot of processing power was needed to handle all the available data labels and their feature extraction. In some instances, some data labels were assumed due to the resource limitation which may have affected the final result. Nonetheless, good results and findings were made by the study despite the limitations.

7.2 Recommendations

A successful research was conducted in the end because the developed model was able to add new components to the process of phishing detection. For one, the exclusive use of URL features was found to be a limiting factor for detection. Secondly, the use of whois and popularity features gave biased detection results. However, a combination of these features together with those of DNS servers among other attributes gave the best results. Based on the outcomes of these study, the following would improve the adaptability of the final system.

- Develop dynamic tools of dealing with the dynamism of phishing attacks more so, the directory generation. In fact, one could choose to focus their research on this particular aspect of phishing attacks.
- Develop individual models that use URL features, whois features, popularity features and DNS server features separately, then compare the results to find the best feature combinations. This work requires a bit of time to get longitudinal records.
- Finally, combine multiple machine learning and deep learning algorithms where an output of one is used as the input of another. This may result in a much superior model and detection systems.

7.3 Future Works

Based on the recommendation made above, this research proposes the following improvements for future phishing detection work:

- Phishing detection studies should try developing models using multiple features. There is a tendency to use specific types or forms of features (it is easier to get results) which limits and even biases results. A combination of different features gives the best results even though it requires time and is more complex to design.
- Extending the coding process of feature extraction. Only binary options are used which in some instances does not reflect the cases found in the real world.
- Finally, combine multiple machine learning and deep learning algorithms when building the final model.

References

Alhumaidan, F. (2012). A critical analysis and treatment of important UML diagrams enhancing modeling power. *Intelligent Information Management*, 4, 231-237.

Ali, W. (2017). Phishing website detection based on supervised machine learning with wrapper features selection. *International Journal of Advanced Computer Science and Applications,* 8(9).

Akanbi, O & Fazeldehkordi, E. (2015). *A Machine-Learning Approach to Phishing Detection and Defense*. Elsevier Inc. Retrieved 28 March, 2019, from: https://www.sciencedirect.com/topics/computer-science/phishing-detection

Bhattacharjee, A & Shyamasundar, R. (2009). Activity diagrams: A formal framework to model business processes and code generation. *Journal of Object Technology*, 8(1).

Biau, G. (2012). Analysis of random forests model. *Journal of Machine Learning Research*, 13, 1063-1095.

Bisson, D. (2018). Three-Quarters of Organizations Experienced Phishing Attacks in 2017, Report Uncovers. *The State of Security*. Retrieved 28 March, 2019, from: https://www.tripwire.com/state-of-security/security-data-protection/three-quarters-organizations-experienced-phishing-attacks-2017-report-uncovers/

Breiman, L. (2001). Random Forests. *Machine Learning* 45, 5–32.

Carlsson, H., Kettis, A & Soderholm, A. (2011). Research Quality and the Role of the University Leadership. *The Association of Swedish Higher Education (SUHF).* Retrieved 28 March, 2019, from: https://gupea.ub.gu.se/bitstream/2077/27835/1/gupea_2077_27835_1.pdf

Chandrasekaran, M., Narayanan, K & Upadhyaya, S. (2006). Phishing E-mail Detection Based on Structural Properties. *Department of Computer Science and Engineering*. Retrieved 28 March, 2019, from: https://pdfs.semanticscholar.org/d6e6/d07f47245974ebc9017c5295a6574286d8f1.pdf

De Vaus, D. (2006). *Research Design in Social Research*. London: SAGE.

Gupta, B., Arachchilage, N & Psannis, K. (2017). Defending against Phishing Attacks: Taxonomy of Methods, Current Issues and Future Directions. *India-Australian Centre for Cyber Security (ACCS)*. Retrieved 28 March, 2019, from: https://arxiv.org/ftp/arxiv/papers/1705/1705.09819.pdf

Hamid, I.R., Abawajy, J.H., & Kim, T. (2013). Using feature selection and classification scheme for automating phishing email detection. *Computer Science*.

Jilani, A., Usman, M & Nadeem, A. (2011). Comparative Study on DFD to UML Diagrams Transformations. *World of Computer Science and Information Technology Journal*, 1(1), 10-16.

Kang H. (2013). The prevention and handling of the missing data. *Korean journal of anesthesiology*, 64(5), 402–406.

Katz, E. (2018). Phishing Statistics: What Every Business Needs to Know. *Dashlane*. Retrieved 28 March, 2019, from: https://blog.dashlane.com/phishing-statistics/

Kohavi, R & John, G. (1997). Wrappers for Feature Subset Selection. *Artificial Intelligence*, 273-274. Retrieved 28 March, 2019, from: http://ai.stanford.edu/~ronnyk/wrappersPrint.pdf

Korolov, M. (2015). Phishing is a $3.7-million Annual Cost for Average Large Company. *CSO From IDG*. Retrieved 28 March, 2019, from: https://www.csoonline.com/article/2975807/phishing-is-a-37-million-annual-cost-for-average-large-company.html

Luo, R., Zhang, W., Burd, S & Seazzu, A. (2013). Investigating phishing victimization with the Heuristic–Systematic Model: A theoretical framework and an exploration. *Elsevier*. Retrieved 28 July, 2019, from: https://www.researchgate.net/publication/278394379_Investigating_phishing_victimization_with_the_Heuristic-Systematic_Model_A_theoretical_framework_and_an_exploration

Miloudi, K. (2018). A Multiview formal model of use case diagrams using Z notation: Towards improving functional requirements quality. *Journal of Engineering*.

Mendels, G. (2019). Organizing machine learning projects: project management guidelines. *Medium*. Retrieved 12 November, 2019, from: https://medium.com/comet-ml/organizing-machine-learning-projects-project-management-guidelines-2d2b85651bbd

Namasivayam, B. (2017). Categorization of phishing detection features and using the feature vectors to classify phishing websites. *Arizona State University*. Retrieved 20 February,

2020, from: https://repository.asu.edu/attachments/189603/content/Namasivayam_asu_0010N_17146.pdf

Nascenia. (2018). Software requirements specification: What it is and why it is important. *Official Website*. Retrieved 20 February, 2020, from: https://www.nascenia.com/the-importance-of-software-requirements-specification/

Pytorch. (2018). Pytorch. *Internet Archives Wayback Machine*. Retrieved 28 March, 2019, from: https://web.archive.org/web/20180615190804/https://pytorch.org/about/

Sahingoz, O., Baykal, S & Bulut, D. (2018). Phishing Detection From URLs By Using Neural Networks. *AIRCC Publishing Corporation*. Retrieved 28 March, 2019, from: https://www.academia.edu/38008213/PHISHING_DETECTION_FROM_URLS_BY_USING_NEURAL_NETWORKS

Sarica, A., Cerasa, A & Quattrone, A. (2017). Random forest algorithm for the classification of neuroimaging data in alzheimer's disease: A systematic review. *A Frontiers in Aging Neuroscience*. Retrieved 20 February, 2020, from: https://www.frontiersin.org/articles/10.3389/fnagi.2017.00329/full

Semusheva, O. (2019). Requirements. Why is it important? *Steel Kiwi*. Retrieved 24 February. 2020, from: https://steelkiwi.com/blog/requirements-why-it-important/

Solomatine, D & Ostfeld, A. (2008). Data-Driven Modelling: Some Past Experiences and New Approaches. *Journal of Hydro informatics - J HYDROINFORM*. Retrieved 28 March, 2019, from: https://www.researchgate.net/publication/228742526_Data-Driven_Modelling_Some_Past_Experiences_and_New_Approaches

Wang, W., Zhang, F., Luo, X & Zhang, S. (2019). PDRCNN: Precise phishing detection with recurrent convolutional neural networks. *Security and Communication Networks*. Retrieved 20 February, 2020, from: https://www.hindawi.com/journals/scn/2019/2595794/

Zhang, N & Yuan, Y. (n.d). Phishing Detection Using Neural Network. *Department of Computer Science, Department of Statistics, Stanford University*. Retrieved 28 March, 2019, from: http://cs229.stanford.edu/proj2012/ZhangYuan-PhishingDetectionUsingNeuralNetwork.pdf

Zuhair, Hiba & Selamat, Ali & Salleh, Mazleena. (2016). Feature selection for phishing detection: A review of research. *International Journal of Intelligent Systems Technologies and Applications*, 15(2).