

A project report on

IDENTIFYING PHISHING URL's USING DIFFERENT ML ALGORITHMS

Submitted in partial fulfilment for the award of the degree of

INTEGRATED M. TECH

(Software Engineering)

by

NEELAMRAJU S M KARTHIK (19MIS7023)

VENKATA SAI GURU PARDHU MADDU (19MIS7099)

P CHANDHU VARDHAN REDDY (19MIS7081)

Under the Guidance of:

Dr. Syed Khasim



INAVOLU, AMARAVATI

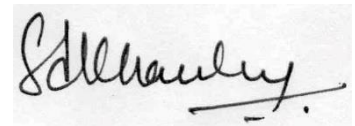
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

May, 2023

CERTIFICATE

This is to certify that the Capstone Project work titled “**IDENTIFYING PHISHING URL's USING DIFFERENT ML ALGORITHMS**” that is being submitted by VENKATA SAI GURU PARDHU MADDU (19MIS7099) is in partial fulfilment of the requirements for the award of Master of Technology (Integrated 5 Year) Software Engineering, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Guide Name: Dr. Syed Khasim



Guide Signature

The thesis is satisfactory

Approved by

PROGRAM CHAIR

M. Tech. SE

DEAN

School Of Computer Science and Engineering

ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to **Prof. Syed Khasim**, Associate Professor Grade – 1, SCOPE, VIT-AP, for his constant guidance, continual encouragement, understanding; more than all, he taught me patience in my endeavour. My association with him is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Internet of Things.

I would like to express my gratitude to **Dr. G. Viswanathan, Dr. Sekar Viswanathan, Sankar Viswanathan, G. V. Selvam, Kadhambari S. Viswanathan, Dr. S. V. Kota Reddy, and Dr. Sudha S V**, School of Computer Science and Engineering, for providing with an environment to work in and for his inspiration during the tenure of the course.

In jubilant mood I express ingeniously my whole-hearted thanks to **Dr. Reeja S R**, Program Chair Master of Technology in Integrated Software Engineering (MTSE), Professor Grade-1, all teaching staff and members working as limbs of our university for their not-self-centred enthusiasm coupled with timely encouragements showered on me with zeal, which prompted the acquirement of the requisite knowledge to finalize my course study successfully. I would like to thank my parents for their support.

It is indeed a pleasure to thank my friends who persuaded and encouraged me to take up and complete this task. At last, but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Amaravati

Date: 18 May 2023

ABSTRACT

Phishing attacks are a growing concern for individuals and organizations alike, as attackers use fake websites and emails to steal sensitive information such as login credentials and financial data. One approach to combating these attacks is to use machine learning (ML) algorithms to automatically identify phishing URLs. In this paper, we explore the effectiveness of various ML algorithms for identifying phishing URLs. We use a dataset of over 10,000 URLs, including both legitimate and phishing URLs, and evaluate the performance of several algorithms, including logistic regression, decision trees, random forests, and support vector machines etc. Our results show that all the algorithms were able to achieve high accuracy in identifying phishing URLs, with random forests and support vector machines achieving the highest accuracy. We also analyse the features that were most important in determining whether a URL was phishing or not, including the length of the URL, the presence of certain words or characters, and the URL's domain name. Our findings suggest that ML algorithms can be effective in identifying phishing URLs and can be a valuable tool in the fight against cybercrime.

In addition to identifying phishing URLs, our study also highlights the importance of feature selection and preprocessing techniques in achieving accurate results. We found that removing certain features, such as those that were highly correlated or had little predictive power, improved the performance of the algorithms. Preprocessing techniques such as feature scaling and data normalization also played a role in improving the accuracy of the models. Furthermore, we demonstrate the potential of using ensemble methods, such as combining multiple algorithms, to further improve the accuracy of phishing URL detection. Overall, our study highlights the effectiveness of machine learning algorithms in identifying phishing URLs and provides insights into the feature selection and preprocessing techniques that can be used to optimize their performance. As phishing attacks continue to evolve, the use of advanced machine learning techniques will be crucial in staying ahead of attackers and protecting against these threats.

LIST OF FIGURES		
Figure	Title	Page. No
Figure 2.1	Worldwide financial losses (in billion) due to phishing attacks	4
Figure 2.2	growth of phishing attacks from 2005 to 2015	5
Figure 2.3	Mean imputation	6
Figure 2.4	Zero Imputation	6
Figure 2.5	Feature selection process	7
Figure 2.6	Feature Selection Models	8
Figure 2.7	Detection accuracy comparison	22
Figure 2.8	Phishing Mechanism	25
Figure 2.9	Types of Phishing attack	26
Figure 2.10	Life cycle of phishing attack	27
Figure 2.11	Legitimate PayPal webpage and phishing webpage of PayPal	28
Figure 3.1	Machine Learning development process	36
Figure 3.2	Architectural Design of the Proposed System	38
Figure 3.3	Use Case diagram for Proposed System	38
Figure 3.4	Flowchart of the proposed System	39
Figure 3.5	Flowchart of the web interface	40
Figure 4.1	Dataset of Phishing URLs	42
Figure 4.2	Dataset of Legitimate URLs	43
Figure 4.3	Code for Address bar-based feature extraction	44
Figure 4.4	Code for domain-based features extraction	45
Figure 4.5	Code for Html & java-script based features extraction	46
Figure 4.6	Code computation for all the feature extraction used dataset.	47
Figure 4.7	Distribution plot of dataset base on the features selected	48
Figure 4.8	Correlation heat map of the dataset	49
Figure 4.9	Feature importance for Decision Tree classifier	49
Figure 4.10	Feature importance for Random Forest classifier	50
Figure 4.11	Summary of the dataset	50
Figure 4.12	Number of missing values in the dataset	51
Figure 4.13	Accuracy performance of models	52

CONTENTS

Acknowledgement

Abstract

List of Figures

CHAPTER 1: INTRODUCTION		
1.1	Background Information	1
1.2	Statement of Problem	2
1.3	Aim of Study	2
1.4	Objectives of the Study	2
1.5	Methodology	2
1.6	Significance of the Project	3
1.7	Scope of the Study	3
1.8	Project Report Arrangement	3

CHAPTER 2: LITERATURE REVIEW		
2.1	Overview of the Study	4
2.2	Theoretical Review	4
2.2.1	Data Imputation	5
2.2.2	Feature Selection	7
2.2.3	Feature Extraction	8
2.2.3.1	Address Bar-Based Features	9
2.2.3.2	Abnormal Based Features	13
2.2.3.3	Html and JavaScript-Based Features	14
2.2.3.4	Domain-Based Features	16
2.2.4	Algorithm and Model Evaluation (Performance Metrics)	19
2.2.4.1	Implementation and Result	21
2.3	Conceptual Review	24
2.3.1	Phishing Mechanism	24
2.3.2	Taxonomy of Phishing Attack	25
2.3.3	Anti-Phishing Technique	26
2.3.4	Visual Similarity-Based Phishing Detection and Filtering Approaches	27
2.4	Empirical Review	28

CHAPTER 3: SYSTEM ANALYSIS AND DESIGN		
3.1	Overview of System Analysis	34
3.2	Analysis of Existing System	34
3.3	Proposed System	34
3.3.1	Benefits of the New System	35
3.4	Model Development	35
3.5	System Modelling	37
3.5.1	System Architecture	37
3.5.2	Use a Case Diagram of the System	37
3.5.3	Flowchart of the System	39

CHAPTER 4: SYSTEM IMPLEMENTATION AND RESULTS		
4.1	Installation Requirements	41
4.1.1	Hardware Requirements	41
4.1.2	Software Requirements	41
4.2	Model Development	41
4.2.1	Data Collection	42
4.2.2	Feature Extraction on the Datasets	43
4.2.2	Data Analysis & Visualization	47
4.2.3	Data Pre-Processing	50
4.2.4	Phishing Detection Model	51

CHAPTER 5: SUMMARY		
5.1	Summary	53
5.2	Contribution to Knowledge	53

CHAPTER 6: CONCLUSION AND RECOMMENDATIONS		
6.1	Conclusion	54
6.2	Recommendation	54

CHAPTER 7: REFERENCE		55
-----------------------------	--	----

CHAPTER ONE

INTRODUCTION

1.1 Background Information

The Internet has become an important part of our lives for gathering and disseminating information, particularly through social media. According to Pamela (2021), the Internet is a network of computers containing valuable data, so there are many security mechanisms in place to protect that data, but there is a weak link: the human. When a user freely gives away their data or access to their computer, security mechanisms have a much more difficult time protecting their data and devices.

Therefore, Imperva (2021) defines social engineering (a type of attack used to steal user data, including login credentials and credit card numbers) as a type of attack that is one of the most common social engineering attacks. The attack happens when an attacker fools a victim into opening an email, instant message, or text message as if it were from a trusted source. When the recipient clicks the link, they mistakenly think they have received a gift and unknowingly click a harmful link, which leads to the installation of malware, the freezing of the machine during a ransomware assault, or the release of private data.

Due to the rapid adoption of technological advancements, there has been a significant growth in computer security threats in recent years, which has also increased the vulnerability of human exploitation. Users should be informed of the methods used by phishers as well as ways to help against falling victim to phishing.

Computer security threats have increased substantially in recent years, owing to the rapid adoption of technology improvements, while simultaneously increasing the vulnerability of human exploitation. Users should know how the phishers do it, and they should also be aware of techniques to help protect themselves from becoming phished.

The strategies employed by cybercriminals are becoming more complex as technology advances. Other than phishing, there are a variety of methods for obtaining personal information from users. KnowBe4 (2021) stated the following techniques:

- a) **Vishing (Voice Phishing):** This kind of phishing includes the phisher calling the victim to get personal information about the bank account. The use of a fake caller ID is the most typical method of phone phishing.
- b) **Smishing (SMS Phishing):** Phishing via Short Message Service (SMS) is known as Smishing. It is a method of luring a target through the SMS text message service by sending a link to a phishing website.
- c) **Ransomware:** A ransomware attack is a type of attack that prevents users from accessing a device or data unless they pay up.

d) Malvertising: Malvertising is malicious advertising that uses active scripts to download malware or push undesirable information onto your computer. The most prevalent techniques used in malvertisements are exploits in Adobe PDF and Flash.

Hence, this is a rapidly evolving threat to individuals as well as big and small corporations. Criminals now have access to industrial-strength services on the dark web, resulting in an increase in the amount of these phishing links and emails, and, more frighteningly, they are increasing in 'quality,' making them tougher to detect.

1.2 Statement of Problem

Phishing attacks have gotten increasingly complex, it is very difficult for an average person to determine if an email message link or website is legitimate. Cyber-attacks by criminals that employ phishing schemes are so prevalent and successful nowadays.

Hence, this project seeks to address fake URLs and domain names by identifying phishing website links. Therefore, having a web application that provides the user an interface to check if a URL is Phishing or legitimate will help decrease security risks to individuals and organizations.

1.3 Aim of Study

This project aims to detect phishing websites using machine learning and deep neural networks by allowing users to check if a URL is phishing or legitimate and have access to resources to help tackle phishing attacks.

1.4 Objectives of the Study

To accomplish the project's purpose, the following objectives have been established:

- i. Dataset collection and pre-processing;
- ii. Machine-learning model selection and development;
- iii. Integration of the developed model to a web application.

1.5 Methodology

An extensive review was done on related topics and existing documented materials such as journals, e-books, and websites containing related information gathered which was examined and reviewed to retrieve essential data to better understand and know how to help improve the system.

The methodology used to achieve the earlier stated objectives is explained below.

The dataset collection consists of phishing and legitimate URLs which were obtained from open-source platforms. The dataset was then pre-processed that is cleaned up from any abnormality such as missing data to avoid data imbalance. Afterward, expository data analysis was done on the dataset to explore and summarize the dataset. Once the dataset was free from all anomalies, website content-based features were extracted from the dataset to get accurate features to train and test the model. An extensive review was done on existing works of literature and machine

learning models on detecting phishing websites to best decide the classification models to solve the problem of detecting phishing websites. Hence, Series of these machine learning classification models such as Decision Tree, Support Vector Classifier, XG-Booster, CatBoost, Naïve Bayes, K-Nearest neighbors, Multilayer perceptions, Gradient Boosting, Logistic Regression and Random Forest was deployed on the dataset to distinguish between phishing and legitimate URLs. The best model with high training accuracy out of all the deployed models was selected.

1.6 Significance of the Project

According to Abdelhamid, Thabtah, and Abdel-jaber (2017), various fraudulent websites have been built on the World Wide Web in the previous decade to resemble reputable websites and steal financial assets from users and organizations. This type of online scam is known as phishing, and it has cost the internet community and other stakeholders hundreds of millions of dollars. As a result, robust countermeasures that can identify phishing are required.

These are the challenges to be addressed in this project:

- a) Reduce the rate of financial theft from users and organizations online.
- b) Educate Internet Users on the deception of phishers.
- c) Educate Internet users on the countermeasures of a phishing attack.

1.7 Scope of the Study

This study explores data science and machine learning models that use datasets gotten from open-source platforms to analyze website links and distinguish between phishing and legitimate URL links.

The model will be integrated into a web application, allowing a user to predict if a URL link is legitimate or phishing. Numerous browsers are supported by this web application.

1.8 Project Report Arrangement

This report is structured into five chapters. The rest of the chapters are stated as follows:

Chapter two describes the summary of works of literature review on the research and related works which is divided into five areas which are: Introduction, Theoretical Review, Conceptual Review, and Empirical Review and Summary of the chapter. Chapter three describes extensively the methodology, system analysis, and design of the system model. Chapter four describes the implementation of the machine learning model in the methodology and discussion on the result. Chapter five and six discusses the Summary, Conclusion, Recommendation, Limitation of Study, and Contribution to knowledge.

CHAPTER TWO

LITERATURE REVIEW

2.1 Overview of the Study

This chapter offers an insight into various important studies conducted by excellent scholars from articles, books, and other sources relevant to the detection of phishing websites. It also provides the project with a theoretical review, conceptual review, and empirical review to demonstrate understanding of the project.

2.2 Theoretical Review

Ankit and Gupta (2017) mentioned that Statistics show that according to Internet world stats ("Internet world stats usage and population statistics", 2014), 2.97 billion people use the internet globally as of 2014, which equates to more than 38% of the world's population. Because the Internet is not secure, hackers can trick unsuspecting people into falling for phishing scams. A phishing e-mail is used to defraud both individuals and financial organizations on the Internet. ("RSA Anti-Fraud Command Center", n.d.) Said the Anti-Phishing Working Group (APWG) is an international consortium that is dedicated to promoting research, education, and law enforcement to eliminate online fraud and cyber-crime. 2012 was a record year for phishing assaults, with a 160% rise in overall attacks over 2011. Over 5.9 billion dollars in losses were incurred as a result of the nearly 450,000 phishing attacks that were discovered in 2013 ("RSA Anti-Fraud Command Centre", n.d.). Total attack increases by 1% in 2013 as compared to 2012. A total of 125,215 phishing assaults were discovered in the first quarter of 2014, up 10.7% from the fourth quarter of 2013. According to a 2014 survey by the Anti-Phishing Working Group (APWG), 99.4% of phishing websites use port 80 and more than 55% of them attempt to trick consumers by include the target website's name in some way. Payment services are the most often targeted industry, and the first quarter of 2014 saw the second-highest number of phishing attempts ever recorded ("Anti- Phishing Working Group (APWG) Phishing activity trends report first quarter", 2014). In the second half of 2014, 123,972 different phishing assaults were recorded ("APWG report", 2014). Total financial losses were 1.2 billion in 2011; by 2013, they had increased to 5.9 billion. According to Figure 2.1 ("The RSA Current State of Cybercrime", n.d.), the financial damages resulting from phishing attacks in 2014 and 2015 were 4.5 and 4.6, respectively. Figure 2.2 depicts the increase in phishing attacks from 2005 to 2015.



Figure 2.1 Worldwide financial losses (in billion) due to phishing attacks.

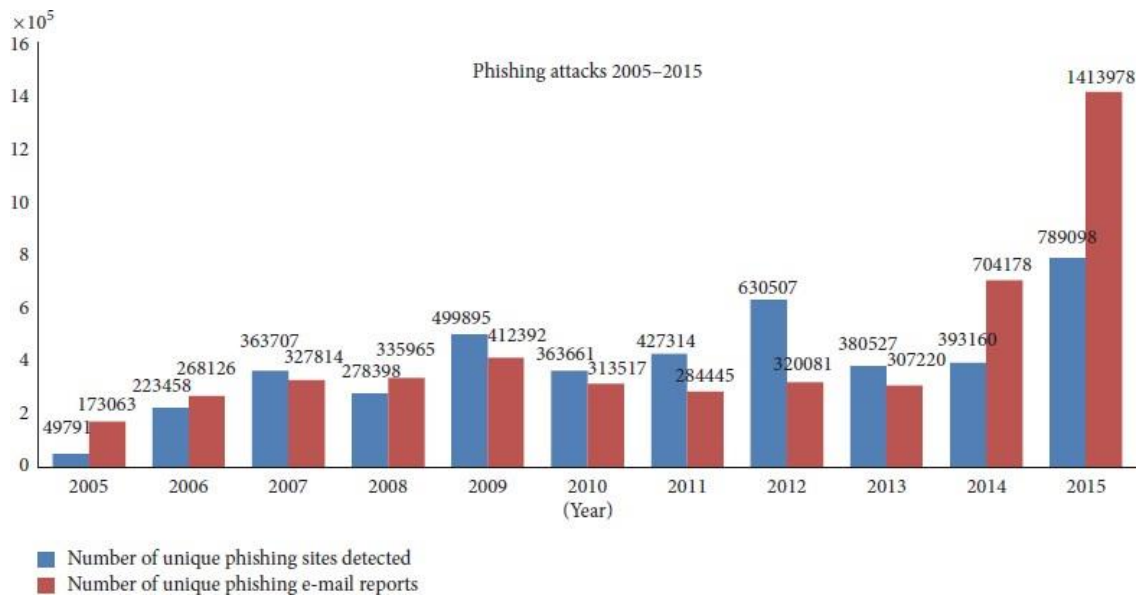


Figure 2.2 growth of phishing attacks from 2005 to 2015

2.2.1 Data imputation

Joachim (n.d). Has defined missing data imputation as a statistical method that replaces missing data points with substituted values. Missing values can appear in real-world datasets for a variety of reasons. They are frequently encoded as NaNs, blanks, or other omissions. Training a model with a dataset with several missed values can have a dramatic impact on the quality of the machine learning algorithm.

There are three main types of missing data:

1) Missing completely at random (MCAR)

MCAR occurs when the missing variable is completely unsystematic. The chance of missing data is unrelated to any other variable and unrelated to the variable with missing values itself when our dataset contains missing values that are fully random. For instance, MCAR would happen if data were missing as a result of depression study survey replies getting misplaced in the mail. (Kyaw, 2020)

2) Missing at random (MAR)

MAR happens when a variable's chance of having missing values is connected to another measured variable but not to the variable's own missing values. Because men are less likely to answer to a poll about depression, for instance, the data values are absent. In this instance, the respondents' gender is related to the missing data. The amount of depression itself is unrelated to the missing data, though. (Kyaw, 2020).

3) Missing Not at Random (MNAR)

When a variable's missing values are connected to the variable's own missing values, MNAR arises. Since the respondents were unable to complete the survey due to their level of depression, the data values in this instance are missing. (Kyaw, 2020)

Popular ways for data imputation for cross-sectional datasets:

I. Imputation using (Mean/Median) Values:

It works by measuring the mean/median of the non-missing values in a column and extracting the missing values separately and independently from each other within each column. As seen in Figure 2.3, only quantitative data may be employed (Will, 2019).

II. Imputation using (Most Frequent) or (Zero/Constant) Values:

Most popular is another statistical technique for imputing missing values and this deals with categorical characteristics (strings or numerical representations) by replacing missing data from each column with its most frequent values (Will, 2019). As shown in Figure 2.4.

III. Imputation using k-NN:

The k nearest neighbor is an algorithm that is used for simple classification. The method predicts the values of any additional data points using "feature similarity." In other words, the value given to the new point depends on how much it resembles the points in the training set. (Will, 2019).

	col1	col2	col3	col4	col5			col1	col2	col3	col4	col5	
0	2	5.0	3.0	6	NaN	mean()		0	2.0	5.0	3.0	6.0	7.0
1	9	NaN	9.0	0	7.0			1	9.0	11.0	9.0	0.0	7.0
2	19	17.0	NaN	9	NaN			2	19.0	17.0	6.0	9.0	7.0

Figure 2.3 Mean imputation

Source: <https://towardsdatascience.com/6-different-ways-to-compensate-for-missing-values-data-imputation-with-examples-6022d9ca0779>

	col1	col2	col3	col4	col5			col1	col2	col3	col4	col5	
0	2	5.0	3.0	6	NaN	df.fillna(0)		0	2	5.0	3.0	6	0.0
1	9	NaN	9.0	0	7.0			1	9	0.0	9.0	0	7.0
2	19	17.0	NaN	9	NaN			2	19	17.0	0.0	9	0.0

Figure 2.4 Zero Imputation

Source: <https://towardsdatascience.com/6-different-ways-to-compensate-for-missing-values-data-imputation-with-examples-6022d9ca0779>

2.2.2 Feature selection

With the use of just pertinent data and the elimination of irrelevant data, feature selection is a technique for lowering the input variable for your model.

It is also the process of automatically choosing relevant features for your machine- learning model based on the type of problem you are trying to solve. We do this by including or excluding important features without changing them. It assists in minimizing the amount of noise in our data and the quantity of our input data. The feature selection procedure is depicted in Figure 2.5.

Feature selection models are of two types:

I. Supervised Models:

The term "supervised feature selection" refers to a technique that chooses features using the output label class. The factors that can improve the model's efficacy are found using the goal variables.

II. Unsupervised Models:

Unsupervised The term "feature selection" refers to a procedure that does not require an output label class. We use them for unlabeled data. Figure 2.6 shows the flow of the feature selection model.

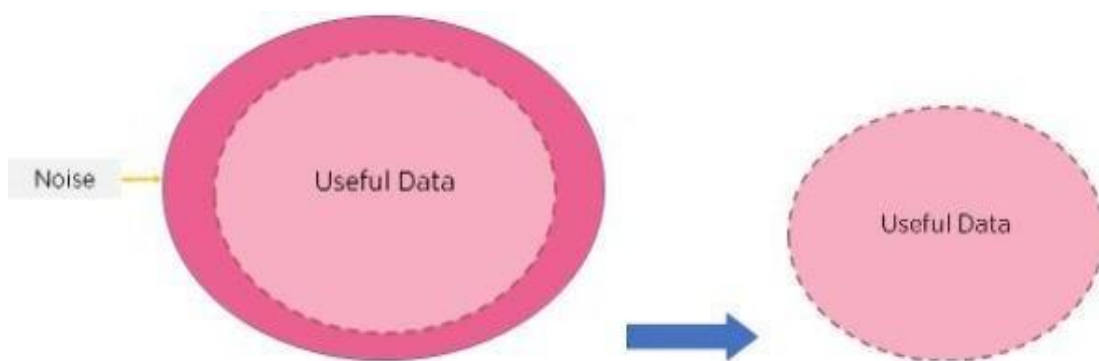


Figure 2.5 Feature selection process

Source: <https://www.simplilearn.com/tutorials/machine-learning-tutorial/feature-selection-in-machine-learning>

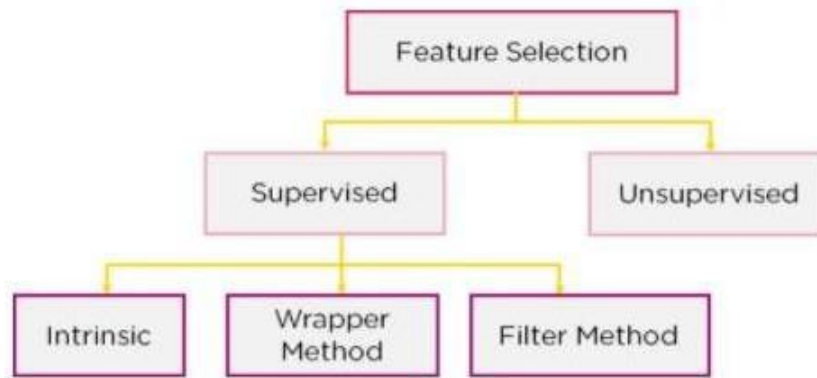


Figure 2.6 Feature Selection Models

Source: <https://www.simplilearn.com/tutorials/machine-learning-tutorial/feature-selection-in-machine-learning>

2.2.3 Feature extraction

Large amounts of raw data are divided into more manageable groupings using a dimensionality reduction approach called feature extraction. These enormous data sets have a lot of variables, which means processing them takes a lot of computing power. In order to reduce the quantity of data that needs to be processed while still properly and thoroughly describing the original data set, approaches that select and/or combine variables into features are known as feature extraction (deepAI, n.d.).

Why is Feature Extraction Useful? When fewer resources are required for processing without losing crucial or pertinent data, the feature extraction procedure is helpful. Additionally, feature extraction can help an analysis by reducing the amount of redundant data. Additionally, the machine's efforts to generate variable combinations (features) and the reduction of the data speed up the learning and generalisation stages of the machine learning process. (DeepAI, undated)

(Rami, Fadi, & Lee, 2015) claim that they have combined significant aspects that have been proven to be reliable and efficient at identifying phishing websites. Additionally, they have updated several other features and proposed some new ones. They have also experimentally given certain well-known features new rules.

These feature selections include:

- i. Address Bar based Features
- ii. Abnormal Based Features
- iii. HTML and JavaScript-based Features
- iv. Domain-based Features

All the listed feature selection above consists of feature extraction which are guided by rules. The feature extraction is as follows:

2.2.3.1 Address bar-based features

(1) Using the IP Address

Users can be sure that someone is attempting to steal their personal information if an IP address is used in place of the domain name in the URL, such as "http://125.98.3.123/fake.html". The link "http://0x58.0xCC.0xCA.0x62/2/paypal.ca/index.html" illustrates how the IP address can occasionally even be converted to hexadecimal form.

$$\text{Rule: if } \begin{cases} \text{If The Domain Part has an IP Address} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(2) Long URL to Hide the Suspicious Part

Phishers can conceal the dubious portion of a URL in the address bar by using a lengthy URL. For example:

http://federmacedoadv.com.br/3f/aze/ab51e2e319e51502f416dbe46b773a5e/?cmd=h_ome&dispatch=11004d58f5b74f8dc1e7c2e8dd4105e811004d58f5b74f8dc1e7c2e8dd4105e8@phishing.website.html

We determined the average URL length for the dataset after measuring each URL's length to confirm the validity of our analysis. According to the findings, phishing URLs are those that have more than 54 characters and are longer than that. By looking over our dataset, we were able to identify 1220 URLs with lengths equal to or greater than 54, or 48.8% of the entire dataset.

$$\text{Rule: IF } \begin{cases} \text{URL length} < 54 \rightarrow \text{feature} = \text{Legitimate} \\ \text{else if URL length} \geq 54 \text{ and } \leq 75 \rightarrow \text{feature} = \text{Suspicious} \\ \text{otherwise} \rightarrow \text{feature} = \text{Phishing} \end{cases}$$

We have been able to update this feature rule by using a method based on the frequency and thus improving its accuracy.

(3) Using URL Shortening Services "Tiny-URL"

The "World Wide Web" has a technology known as URL shortening that allows a URL to be significantly reduced in size while still directing to the desired webpage. This is accomplished utilizing an "HTTP Redirect" on a short domain name, which links to the webpage that has a long URL. The URL "http://portal.hud.ac.uk/" can be condensed to "bit.ly/19DXSk4", for instance.

$$\text{Rule: IF } \begin{cases} \text{TinyURL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(4) URL's having "@" Symbol

The "@" character in a URL causes the browser to ignore everything before it, and the actual address frequently comes after the "@" symbol.

$$\text{Rule: IF} \begin{cases} \text{Url Having @ Symbol} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(5) Redirecting using "/"

If there is a "/" in the URL route, the visitor will be redirected to another website. An example of such URLs is: "http://www.legitimate.com/<http://www.phishing.com>". We examine the location where the "/" appears. We discover that the "/" should be in the sixth position if the URL begins with "HTTP". However, if the URL employs "HTTPS" then the "/" should appear in the seventh position.

$$\text{Rule: IF} \begin{cases} \text{the position of the Last Occurrence of "/" in the URL} > 7 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(6) Including a Prefix or Suffix in the Domain, Spaces (-) Between Each

Legitimate URLs rarely employ the dash symbol. Prefixes or suffixes, separated by (-), are frequently added by phishers to the domain name to give users the idea that they are visiting a reliable website. For example, <http://www.Confirme-paypal.com/>.

$$\text{Rule: IF} \begin{cases} \text{Domain Name Part Includes (-) Symbol} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(7) Sub Domain and Multi-Sub Domains

Assume we already have access to <http://www.hud.ac.uk/students/>. The country-code top-level domain (ccTLD), "UK" in our example, may be present in a domain name. Academic is abbreviated as "ac" in the "ac" portion. The domain's real name is "hud," and "UK" is referred to as its second-level domain (SLD). It is necessary to remove the (www.) from the URL, which is a subdomain in and of itself, before we can create a rule for extracting this feature. The (ccTLD) must then be deleted if it already exists. We then add up the leftover dots. If there are more than one dot, the URL is considered "Suspicious" because it only has one subdomain. The classification changes to "Phishing" nevertheless if there are more than two dots because this indicates the presence of many subdomains. Otherwise, we shall label the feature as "Legitimate" if the URL has no subdomains.

$$\text{Rule: IF} \begin{cases} \text{Dots In Domain Part} = 1 \rightarrow \text{Legitimate} \\ \text{Dots In Domain Part} = 2 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

(8) HTTPS (Hypertext Transfer Protocol with Secure Sockets Layer)

The presence of HTTPS is crucial in creating the impression that a website is legitimate, but it is not sufficient. The authors of Muhammad, Thabtah, and McCluskey (2012) and Muhammad, Thabtah, and McCluskey (2013) advise examining the certificate's validity, issuer's reputation, and age. "GeoTrust, GoDaddy, Network Solutions, Thawte, Comodo, Doster, and VeriSign" are just a few of the certificate authorities that are frequently mentioned as being among the most reliable brands. Furthermore, we discover that the minimal age of a trustworthy certificate is two years by putting our datasets to the test.

Rule: IF $\left\{ \begin{array}{l} \text{Use HTTPS and Issuer Is Trusted and Age of Certificate} \geq 1 \text{ Years} \rightarrow \text{Legitimate} \\ \text{Using https and Issuer Is Not Trusted} \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{array} \right.$

(9) Domain Registration length

We assume that reliable domains are frequently purchased for several years in advance based on the fact that phishing websites only exist for a brief amount of time. The longest fraudulent domains in our sample were only active for a single year.

Rule: IF $\left\{ \begin{array}{l} \text{Domains Expire on} \leq 1 \text{ years} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{array} \right.$

(10) Favicon

A favicon is a visual representation (icon) connected to a particular website. Favicons are currently displayed in the address bar of many user agents as a visual reminder of the website identity, such as graphical browsers and newsreaders. The webpage is probably a Phishing attempt if the favicon loads from a different domain than what is displayed in the URL bar.

Rule: IF $\left\{ \begin{array}{l} \text{Favicon Loaded From External Domain} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{array} \right.$

(11) Using Non-Standard Port

This capability can be used to verify whether a given service, such HTTP, is available or unavailable on a particular server. To control intrusions, it is much better to merely open the ports that you need. By default, many firewalls, proxy servers, and Network Address Translation (NAT) servers will block all or most of the ports and only open the ones that are explicitly chosen. If all ports are exposed, phishers can operate nearly any service they want, endangering the security of user data. The most important ports and their preferred status are shown in Table 2.1

$$Rule: IF \begin{cases} \text{Port is of the Preferred Status} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(12) The “HTTPS” Token is present in the URL’s Domain Section

In order to deceive users, phishers may append the "HTTPS" token to the domain portion of a URL. For example,

[http://https-www-paypal-it-webapps-mpp-home.soft-hair.com/.](http://https-www-paypal-it-webapps-mpp-home.soft-hair.com/)

$$Rule: IF \begin{cases} \text{Using HTTP Token in Domain Part of The URL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

Table 2.1: Common ports to be checked

PORT	Service	Meaning	Preferred Status
21	FTP	Transfer files from one host to another	Close
22	SSH	Secure File Transfer Protocol	Close
23	Telnet	Provide a bidirectional interactive text-oriented communication	Close
80	HTTP	Hyper test transfer protocol	Open
443	HTTPS	Hypertext transfer protocol secured	Open
445	SMB	Providing shared access to files, printers, serial ports	Close
1433	MSSQL	Store and retrieve data as requested by other software applications	Close
1521	ORACLE	Access oracle database from the web.	Close
3306	MySQL	Access MySQL database from the web.	Close
3389	Remote Desktop	allow remote access and remote collaboration	Close

2.2.3.2 abnormal based features

(1) Request URL

Request URL checks to see if the webpage's external media files—such as photos, videos, and sounds—are loaded from a different domain. In legitimate web pages, the webpage address and most of the objects embedded within the webpage are sharing the same domain.

$$\text{Rule: IF} \begin{cases} \% \text{ of Request URL} < 22\% \rightarrow \text{Legitimate} \\ \% \text{ of Request URL} \geq 22\% \text{ and } 61\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{feature} = \text{Phishing} \end{cases}$$

(2) URL of Anchor

Request URL checks to see if the webpage's external media files—such as photos, videos, and sounds—are loaded from a different domain.

1. If the website's domain name and the a> tags have distinct names. This is like the request URL feature.
2. If the anchor points to no website, for instance:
 - a)
 - b)
 - c)
 - d)

$$\text{Rule: IF} \begin{cases} \% \text{ of URL of Anchor} < 31\% \rightarrow \text{Legitimate} \\ \% \text{ of URL of Anchor} \geq 31\% \text{ and } 67\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

(3) Links in <Meta>, <Script> and <Link> tags

Given that our investigation covers all angles likely to be used in the webpage source code, we find that it is common for legitimate websites to use <Meta> tags to offer metadata about the HTML document; <Script> tags to create a client-side script; and <Link> tags to retrieve other web resources. It is expected that these tags are linked to the same domain of the webpage.

$$\text{Rule: IF} \begin{cases} \% \text{ of Links in " < Meta > ", " < Script > " and " < Link > " } < 17\% \rightarrow \text{Legitimate} \\ \% \text{ of Links in " < Meta > ", " < Script > " and " < Link > " } \geq 17\% \text{ And } \leq 81\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

(4) Server Form Handler (SFH)

SFHs that contain an empty string or “about: blank” are considered doubtful because action should be taken upon the submitted information. Additionally, since external

domains almost never handle supplied information, it is suspicious if the domain name in SFHs differs from the domain name of the webpage.

$$Rule: IF \begin{cases} SFH \text{ is "about blank" Or Is Empty} \rightarrow \text{Phishing} \\ SFH \text{ Refers To A Different Domain} \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(5) Submitting Information to Email

A user can enter his personal information through a web form, which is then sent to a server for processing. The user's details might be forwarded to his email by a phisher. A server-side scripting language, such as PHP's "mail ()" function, may be employed to that effect. Another client-side functionality that could be used in this circumstance is the "mailto:" function.

$$Rule: IF \begin{cases} \text{Using mail ()or mailto:Function to Submit User Information} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(6) Abnormal URL

This feature can be extracted from the WHOIS database. For a legitimate website, identity is typically part of its URL.

$$Rule: IF \begin{cases} \text{The Host Name Is Not Included In URL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

2.2.3.3 html and JavaScript-based features

(1) Website Forwarding

The fine line that distinguishes phishing websites from legitimate ones is how many times a website has been redirected. In our dataset, we find that legitimate websites have been redirected one-time max. On the other hand, phishing websites containing this feature have been redirected at least 4 times.

$$Rule: IF \begin{cases} \text{ofRedirect Page} \leq 1 \rightarrow \text{Legitimate} \\ \text{of Redirect Page} \geq 2 \text{ And } < 4 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

(2) Status Bar Customization

Phishers may use JavaScript to show a fake URL in the status bar to users. To extract this feature, we must dig out the webpage source code, particularly the "onMouseOver" event, and check if it makes any changes on the status bar.

$$Rule: IF \begin{cases} \text{onMouseOver Changes Status Bar} \rightarrow \text{Phishing} \\ \text{It Does't Change Status Bar} \rightarrow \text{Legitimate} \end{cases}$$

(3) Disabling Right Click

Phishers use JavaScript to disable the right-click function so that users cannot view and save the webpage source code. This feature is treated exactly as “Using onMouseOver to hide the Link”. Nonetheless, for this feature, we will search for the event “event.Button==2” in the webpage source code and check if the right-click is disabled.

$$Rule: IF \begin{cases} \text{Right Click Disabled} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(4) Using Pop-up Window

It is unusual to find a legitimate website asking users to submit their personal information through a pop-up window. On the other hand, this feature has been used in some legitimate websites and its main goal is to warn users about fraudulent activities or broadcast a welcome announcement, though no personal information was asked to be filled in through these pop-up windows.

$$Rule: IF \begin{cases} \text{Popup Window Contains Text Fields} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(5) Iframe Redirection

The Iframe is an HTML tag used to display an additional webpage into one that is currently shown. Phishers can make use of the “iframe” tag and make it invisible i.e. without frame borders. In this regard, phishers make use of the “frame border” attribute which causes the browser to render a visual delineation.

$$Rule: IF \begin{cases} \text{Using iframe} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

2.2.3.4 domain-based features

(1) Age of Domain

This feature can be extracted from the WHOIS database (Whois 2005). Most phishing websites live for a short period. By reviewing our dataset, we find that the minimum age of the legitimate domain is 6 months

$$Rule: IF \begin{cases} \text{Age Of Domain} \geq 6 \text{ months} \rightarrow \text{Legitimate} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

(2) DNS Record

For phishing websites, either the claimed identity is not recognized by the WHOIS database (Whois 2005) or no records are found for the hostname (Pan and Ding 2006). If the DNS record is empty or not found then the website is classified as “Phishing”, otherwise it is classified as “Legitimate”.

$$Rule: IF \begin{cases} \text{no DNS Record For The Domain} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(3) Website Traffic

This feature measures the popularity of the website by determining the number of visitors and the number of pages they visit. However, since phishing websites live for a short period, they may not be recognized by the Alexa database (Alexa the Web Information Company., 1996). By reviewing our dataset, we find that in the worst scenarios, legitimate websites ranked among the top 100,000. Furthermore, if the domain has no traffic or is not recognized by the Alexa database, it is classified as “Phishing”. Otherwise, it is classified as “Suspicious”.

$$Rule: IF \begin{cases} \text{Website Rank} < 100,000 \rightarrow \text{Legitimate} \\ \text{Website Rank} > 100,000 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

(4) PageRank

PageRank is a value ranging from “0” to “1”. PageRank aims to measure how important a webpage is on the Internet. The greater the PageRank value the more important the webpage. In our datasets, we find that about 95% of phishing web pages have no PageRank. Moreover, we find that the remaining 5% of phishing webpages may reach a PageRank value up to “0.2”.

$$Rule: IF \begin{cases} \text{PageRank} < 0.2 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(5) Google Index

This feature examines whether a website is in Google's index or not. When a site is indexed by Google, it is displayed on search results (Webmaster resources, 2014). Usually, phishing webpages are merely accessible for a short period, and as a result, many phishing webpages may not be found on the Google index.

$$\text{Rule: IF } \begin{cases} \text{Webpage Indexed by Google} \rightarrow \text{Legitimate} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

(6) Number of Links Pointing to Page

The number of links pointing to the webpage indicates its legitimacy level, even if some links are of the same domain (Dean, 2014). In our datasets and due to their short life span, we find that 98% of phishing dataset items have no links pointing to them. On the other hand, legitimate websites have at least 2 external links pointed to them.

$$\text{Rule: IF } \begin{cases} \text{Of Link Pointing to The Webpage} = 0 \rightarrow \text{Phishing} \\ \text{Of Link Pointing to The Webpage} > 0 \text{ and } \leq 2 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

(7) Statistical-Reports Based Feature

Several parties such as Phish Tank (Phish Tank Stats, 2010-2012), and Stop-Badware (Stop-Badware, 2010-2012) formulate numerous statistical reports on phishing websites at every given period; some are monthly and others are quarterly. In our research, we used 2 forms of the top ten statistics from Phish Tank: "Top 10 Domains" and "Top 10 IPs" according to statistical reports published in the last three years, starting in January 2010 to November 2012. Whereas for "Stop-Badware", we used "Top 50" IP addresses.

$$\text{Rule: IF } \begin{cases} \text{Host Belongs to Top Phishing IPs or Top Phishing Domains} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

Rishikesh and Irfan (2018), stated the feature extraction used for detecting phishing URLs, are as follows:

- I. Presence of IP address in URL: If IP address is present in URL, then the feature is set to 1 else set to 0. Most benign sites do not use an IP address as an URL to download a webpage. The use of an IP address in a URL indicates that the attacker is trying to steal sensitive information.
- II. Presence of @ symbol in URL: If @ symbol is present in URL then the feature is set to 1 else set to 0. When phishers add a specific @ sign to a URL, the browser ignores everything before the "@" character and frequently skips over the true address [4].

- III. The number of dots in Hostname: Phishing URLs have many dots in URL. For example, <http://shop.fun.amazon.phishing.com>, in this URL phishing.com is an actual domain name, whereas the use of the “amazon” word is to trick users to click on it. The average number of dots in benign URLs is 3. The feature is set to 1 if there are more than 3 dots in the URLs; otherwise, it is set to 0.
- IV. A prefix or suffix to the domain may be separated by the dash (-) sign, in which case the characteristic is set to 1 otherwise. Legitimate URLs rarely employ the dash symbol. Phishers include the dash symbol (-) in the domain name to give users the impression that they are visiting a trustworthy website. For instance, the real website address is <http://www.onlineamazon.com>, but phishers can construct a phony version of it called <http://www.online-amazon.com> to trick unwary people.
- V. URL redirection: If “//” is present in the URL path then the feature is set to 1 else to 0. If the URL route contains the character “//,” the visitor will be redirected to another website.
- VI. HTTPS token in URL: If HTTPS token is present in URL, then the feature is set to 1 else to 0. To deceive users, phishers may tack on the “HTTPS” token to the domain portion of a URL. For example, <http://https-www-paypal-it-mpp-home.soft-hair.com>.
- VII. Information submission to Email: Phishers might use “mail ()” or “mailto:” functions to redirect the user’s information to his email. If such functions are present in the URL then the feature is set to 1 else to 0.
- VIII. URL Shortening Services “Tiny-URL”: Tiny-URL service allows the phisher to hide long phishing URLs by making them short. The goal is to redirect the user to phishing websites. If the URL is crafted using shortening services (like bit.ly) then the feature is set to 1 else 0.
- IX. Hostname Length: The benign URLs' average length is discovered to be 25, and if the URL is longer than 25, the feature is set to 1; otherwise, it is set to 0.
- X. Presence of sensitive words in URL: Phishing sites use sensitive words in its URL so that users feel that they are dealing with a legitimate webpage. The following words are frequently seen in phishing URLs: "confirm," "account," "banking," "secure," "ebyisapi," "webscr," "signin," "mail," "install," "toolbar," "backup," "Paypal," "password," "username," etc.;
- XI. XI. The number of slashes in the URL: The average benign URL contains 5 slashes; if the amount is higher, the feature is set to 1, otherwise it is set to 0.
- XII. Unicode characters are present in the URL: Phishers can employ Unicode characters in the URL to fool visitors into clicking on them. The domain "xn--80ak6aa92e.com," for instance, is equal to "appe.com." The user can see the URL "appe.com," but when they click it, they are taken to the phishing website "xn--80ak6aa92e.com."
- XIII. XIII. SSL Certificate Age: The presence of HTTPS is crucial in creating the impression that a website is legitimate. However, the SSL certificate for safe websites must be at least one to two years old.
- XIV. URL of Anchor: We have extracted this feature by crawling the source code on the URL. URL of the anchor is defined by <a> tag. If the <a> tag has a maximum

number of hyperlinks that are from the other domain then the feature is set to 1 else to 0.

- XV. IFRAME: We have extracted this feature by crawling the source code of the URL. This tag is used to add another web page to the existing main webpage. The "iframe" tag allows phishers to make a frame invisible, or without frame boundaries. Since the border of an inserted webpage is invisible, the user seems that the inserted webpage is also part of the main web page and can enter sensitive information.
- XVI. Website Rank: We extracted the rank of websites and compare it with the first One hundred thousand websites of the Alexa database. If the rank of the website is greater than 10,000 then the feature is set to 1 else to 0.

(Shreya, 2020), Stated feature selection used in her project which is based on the listed categories below which can be used to make predictions for phishing websites. These categories are similar to (Rami, Fadi & Lee, 2015) compounded principle for predicting phishing websites.

- 1. Address Bar based Features
- 2. Domain-based Features
- 3. HTML & JavaScript based Feature

2.2.4 Algorithm and model evaluation (Performance Metrics)

Algorithms or Models are used when it comes to machine learning includes the most important topics in Artificial Intelligence. Supervised and Unsupervised learning, which relate to labeled and unlabeled data analysis or data prediction, are further divisions of it. Regression and classification are two additional business problem categories in supervised learning.

The classification method uses labeled data as input and requires us to forecast the output into a class. Binary Classification is used when there are just two classes. Multi-Class Classification is used when there are more than two classes.

There are so many classification algorithms available but for this project, let focus on the commonly used algorithms use by researchers to predict phishing websites.

- 1. According to Rishikesh and Irfan (2018), three machine learning classification models such as Decision Tree, Random Forest, and Support vector machine was selected to detect phishing websites.

- i. Decision Tree Algorithm (Rahul, 2017)

One of the most widely used algorithms in machine learning technology. The decision tree algorithm is easy to understand and easy to implement. The decision tree begins its work by choosing the best splitter from the available attributes for classification which is considered as a root of the tree. The algorithm continues

to build a tree until it finds the leaf node. With the help of a decision tree, a training model may be created that predicts target values or classes. The interior nodes of the tree represent attributes, whereas the leaf nodes represent class labels. These nodes are computed in the decision tree algorithm using the Gini index and information gain techniques.

ii. Random Forest Algorithm (Saimadhu, 2017)

One of the most potent machine learning algorithms, the random forest algorithm is built on the idea of the decision tree algorithm. Random forest algorithm creates the forest with several decision trees. A high number of a tree gives high detection accuracy. The bootstrap method is used to create trees. In the bootstrap method features and samples of a dataset are randomly selected with replacement to construct a single tree. A random forest algorithm will select the best splitter for the classification from among randomly chosen features, and like the decision tree approach, the random forest algorithm also uses the Gini index and information gain techniques to discover the best splitter. This process will get continue until a random forest creates N number of trees. The method will calculate the votes for each projected target once each tree in the forest forecasts the target value. Finally, the random forest algorithm considers the high-voted predicted target as a final prediction.

iii. Support Vector Machine Algorithm (Noel, 2016)

Another potent algorithm in machine learning is the support vector machine. In the support vector machine algorithm, each data item is plotted as a point in n-dimensional space, and the support vector machine algorithm constructs separating lines for the classification of two classes, this separating line is well known as a hyperplane. Support vector machine seeks for the closest points called support vectors and once it finds the closest point it draws a line connecting to them. The separating line is then built by the support vector machine, cutting through and parallel to the connecting line. Another potent algorithm in machine learning is the support vector machine. In a real scenario, it is not possible to separate complex and nonlinear data, to solve this problem support vector machine uses kernel trick which transforms lower- dimensional space to higher-dimensional space.

2. Kondeti, Konka, and Kavishree (2021) argued that phishing website detection is best done using Machine learning supervised classification models. Models such as Decision Tree Classifier, Random Forest Model, Multilayer Perceptron's, XGBoost Classifier, Auto Encoder, and Support Vector Machines are trained on a categorical input URL such as phishing (1) and legitimate (0). Since Rishikesh et al. (2018) has stated the significance of Decision tree classifier, Random Forest, and Support Vector Machines models on detecting phishing websites, let consider other classification models stated by Kondeti et al. (2021) for detecting phishing websites which are as follows:

I. XGBooster is best used because it is not any different for classification or regression processes, it is meant for speed and performance. It will add gradient boosting to decision trees.

II. Auto Encoder Neural Network

It is like a neural network that has the same no of input neurons as that of output neurons. It has fewer neurons in the hidden layers of the network that are called predictors. The input neurons process the output and transmit information to the predictors.

III. Multilayer Perceptrons

Multilayer perceptrons are known as feed-forward neural networks. They are used to process multiple stages simultaneously and result in an optimal decision for the processed stage.

2.2.4.1 Implementation and result

1. (Rishikesh & Irfan, 2018) stated the implementation and result for detecting phishing websites. Let consider and evaluate the models used by Rishikesh et al. (2018) to carry out phishing websites detection. Rishikesh et al. (2018) said that the scikit-learn tool is best used to import Machine learning algorithms for phishing detection. A Dataset is divided into a training set and testing set in 50:50, 70:30, and 90:10 ratios respectively. Each classifier is trained using the training set and a testing set is used to evaluate the performance of classifiers. The performance of classifiers has been evaluated by calculating the classifier's accuracy score, false-negative rate, and false-positive rate from Table 2.2.

The outcome demonstrates that compared to decision tree and support vector machine methods, the Random Forest approach provides greater detection accuracy, 97.14%, with the lowest false-negative rate. The results also demonstrate that as additional datasets are utilized as the training dataset, the detection accuracy of phishing websites improves. All classifiers perform well when 90% of data is used as a training dataset. Fig. 2.7 shows the detection accuracy of all classifiers when 50%, 70%, and 90% of data is used as a training dataset, and the graph clearly shows that detection accuracy increases when 90% of data is used as a training dataset and random forest detection accuracy is maximum than other two classifiers.

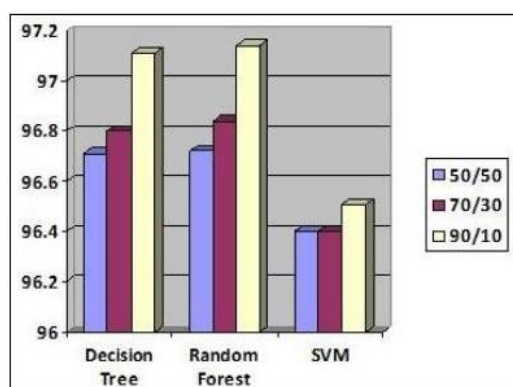


Figure 2.7 Detection accuracy comparison (Source: Rishikesh and Irfan, 2018)

Table 2.2: Classifier's performance

Dataset Split Ratio	Classifiers	Accuracy Score	False Negative Ratio	False Positive Ratio
50:50	Decision Tree	96.71	3.69	2.93
	Random Forest	96.72	3.69	2.91
	Support Vector Machine	96.40	5.26	2.08
70:30	Decision Tree	96.80	3.43	2.99
	Random Forest	96.84	3.35	2.98
	Support Vector Machine	96.40	5.13	2.17
90:10	Decision Tree	97.11	3.18	2.66
	Random Forest	97.14	3.14	2.61
	Support Vector Machine	96.51	4.73	2.34

2. According to (Ashritha, Chaithra, Mangala & Deekshitha, 2019), many algorithms are used to detect phishing websites accurately. Few of them are discussed which can be used to classify the URL as legitimate or phished. The publicly available phishing website's data set from the UCI machine learning repository can be used for training and testing. The features of the dataset are used to predict the result.

Different algorithms that can be used to detect phishing websites are:

I. Artificial Neural Networks (ANN)

An artificial neural network (ANN), inspired by biological neural networks, is a set of interconnected nodes (neurons). Each connection between nodes is typically assigned weights. The network learns by adjusting the weights, in the learning phase for the correct prediction process. ANNs were thought to be less appropriate for data mining due to their difficult interpretation and extensive training times. However, their advantages include the ability to classify patterns on which they have not been trained and a high tolerance for noisy data.

II. K-Nearest Neighbor (k-NN)

K-NN classifiers learn by analogy by contrasting the test tuple with similar training tuples. Since these are distance-based comparisons and each factor is essentially given the same weight, accuracy could be affected by noisy or unimportant data. Editing and pruning approaches have been developed, respectively, to address the problems of useless and noisy data tuples. N attributes are used to describe the training tuples. Each tuple is a representation of a point in an n-dimensional space. Through experimentation, it is feasible to determine the appropriate number of neighbors.

III. Support Vector Machine (SVM)

SVMs are capable of classifying both linear and nonlinear data. In short, when given original training data, the algorithm uses a nonlinear mapping to transform it into a higher dimension. In this dimension, a linear optimal hyperplane is searched, to keep the data of any two classes separate. SVMs can be used for classification and numerical prediction. The simplest SVM problem is a two-class problem with linearly separable classes. For a 2-D problem, a straight line can be drawn to separate the classes multiple lines could be drawn.

IV. Random Forests (RF)

Along with random attribute selection, Random Forests can be built using bagging. Random Forests follow an ensemble approach to learning, which is a divide and conquer approach for improving performance in a straightforward decision tree, the input or test is introduced at the top and travels downward, arriving at smaller subgroups. The ensemble mechanism in a random forest mixes several random subsets of trees. The input/test traverses through all the trees. The result is calculated based on an average or weighted average of the individual

results, or the voting majority in the case of categorical data. The strength of each individual classifier and the degree of classifier reliance determine the accuracy of a random forest, which also solves the issue of decision trees' overfitting.

2.3 Conceptual Review

2.3.1 phishing mechanism

Figure 2.8 below shows a fake website is the clone of a targeted genuine website, and it always contains some input fields (e.g., textbox). When the user submits his/her details, the information is transferred to the attacker. The following actions are taken by an attacker to steal the user's credentials from the innocent party:

1. Construction of Phishing Site. The attacker recognizes the target as a well-known company in the first phase. The attacker then visits the organization's website to get comprehensive information about it. The attacker then creates the bogus website using this information.
2. URL Sending. In this step, the attacker creates a fake email and distributes it to many users. In the phony email, the attacker included the URL of the false website. In a spear-phishing attack, the email is sent to a specific group of users. With the use of blogs, forums, and other online communities, an attacker can also disseminate the URL of phishing websites (Kruegel, Kirda, Mutz, Robertson, & Vigna, 2005).
3. Stealing of the Credentials: when the user clicks on the attached URL, consequently, fake site is opened in the web browser. A phony login form on the false website is used to steal a user's credentials from an unsuspecting visitor. Additionally, the attacker has access to the user-filled information.
4. Identity Theft: attacker uses this credential for malicious purposes. For example, an attacker purchases something by using the credit card details of the user. (Ankit & Gupta, 2017).

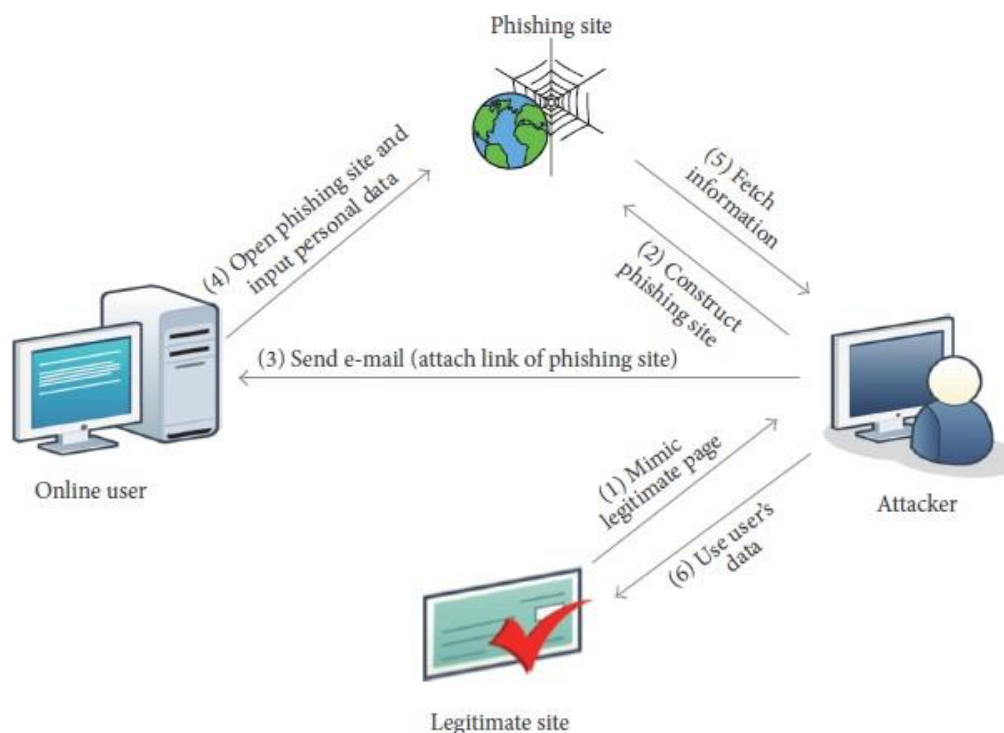


Figure 2.8 Phishing Mechanism (Source: Ankit & Gupta, 2017)

2.3.2 Taxonomy of phishing attack

An Attacker performs a phishing attack by utilizing technical subterfuge and social engineering techniques [(“RSA Anti-Fraud Command Center”, n.d.), (Almomani, Gupta, Atawneh, Meulenberg, & Almomani, 2013)]. Attackers use social engineering techniques to execute this attack by sending phony email. Attackers frequently persuade victims to react using the names of financial institutions, credit card providers, online retailers, and other entities (Tewari, Jain, & Gupta, 2016). Technical deception techniques use Trojan and key logger spyware to directly steal credentials from the user's system by installing malware (Gupta, Tewari, Jain, & Agrawal, 2016). Additionally, the spyware sends users to bogus websites or proxy servers. Attackers attached malware or embedded malicious links in the fraudulent emails and when the user opens the fraud hyperlink, malicious software is installed on the user's system, which collected the confidential information from the system and sent it to the attacker (e.g., key logger software sends the details of every key hit by the user). Attackers may also get remote access to the victim's computer and collect data whenever attackers want. In this paper, we focus on social engineering schemes, as it is the most popular way to steal victim's information by phishing. Classification of various phishing attacks is shown in Figure 2.9. (Ankit & Gupta, 2017).

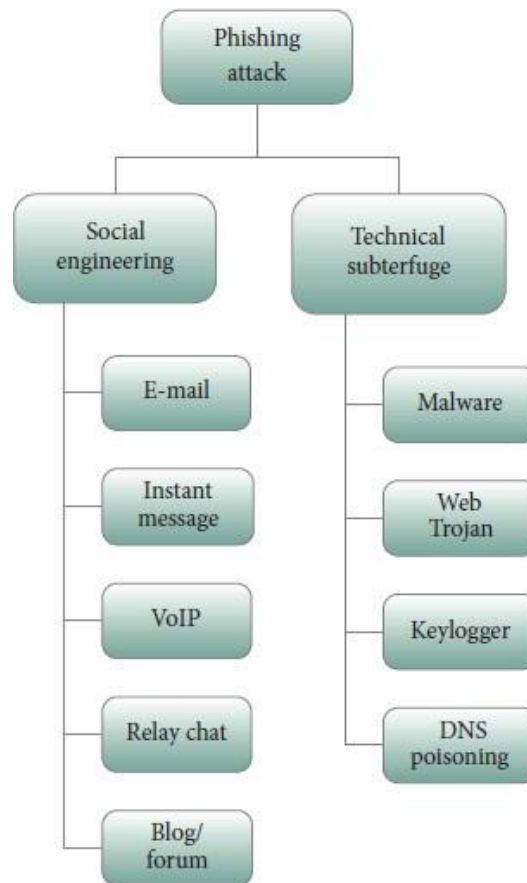


Figure 2.9 Types of Phishing attack (Source: Ankit & Gupta, 2017).

2.3.3 Anti-phishing technique

The method to a phishing scam starts with spreading bogus e-mail. To prevent a phishing attack, after receiving such e-mail, anti-phishing techniques need to be activated, either by redirecting the phishing mail in the spam folder or by showing a warning when an online user clicks on the link of the phishing URL. The lifecycle of phishing attacks is shown in Figure 2.10. The following steps are involved in the phishing lifecycle:

1. **Step 1.** The attacker creates the fake copy of a popular organization and sends the URL of the fake website to many Internet users using e-mail, blogs, social networking sites, and so forth.
2. **Step 2.** In the case of a fake e-mail, every e-mail is first to pass through the DNS-based blacklist filters. If the domain is found in the blacklist, then e-mail is blocked before it reached the SMTP mail server. There are also various solutions available which block fake e-mail based on structural features of mail (Almomani, Gupta, Atawneh, Meulenberg, & Almomani, 2013).
3. **Step 3.** Some browser-based blacklist techniques block the site on the client-side if a phony email gets past blacklist and feature-based solutions and the user clicks on the email's attached link.
4. **Step 4.** Other approaches, including those relying on heuristics and visual analogies, likewise blocked the website only when the browser requested a questionable webpage.

5. **Step 5.** The phishing attack obtains the credentials of innocent users and delivers them to the attacker if it manages to get past all the defenses. The attacker makes use of this information for personal or other gain. 2017 (Ankit & Gupta).

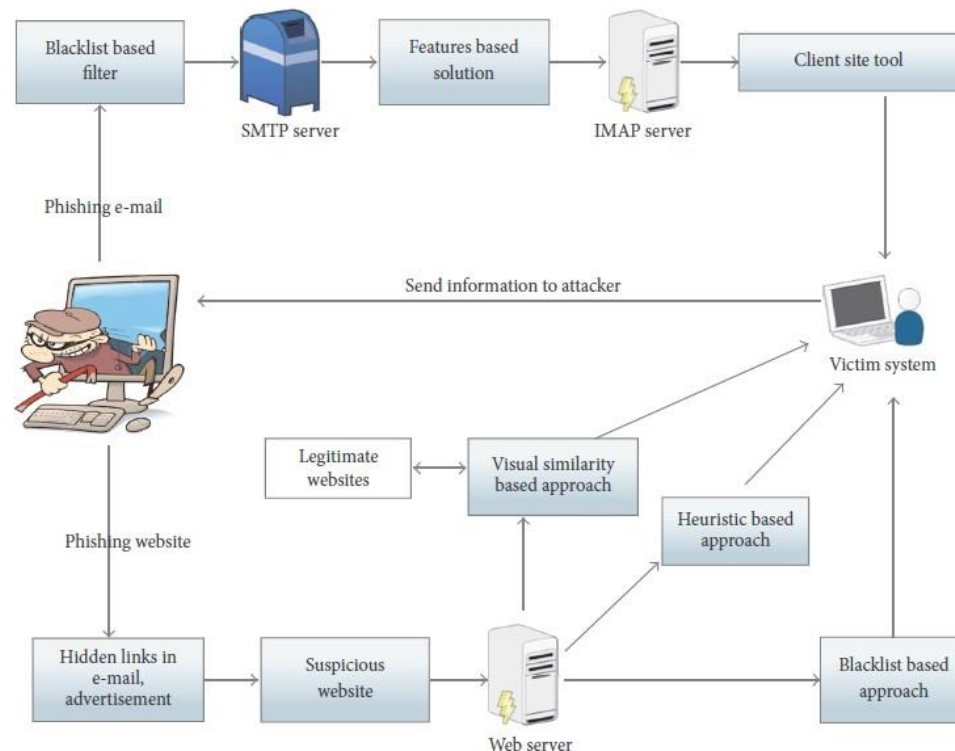


Figure 2.10 Life cycle of phishing attack (Source: Ankit & Gupta, 2017).

2.3.4 Visual similarity-based phishing detection and filtering approaches

By comparing the phishing websites' striking visual similarity to the targeted legitimate website in terms of page designs, graphics, text content, font size, and font color, a user may fall victim to the assault. Take for example the fake and genuine web pages of PayPal as shown in Figure 2.11.

Both pages have the same visual appearance but different URLs. It is not always necessary that people carefully notice on URL and SSL (Secure Socket Layer) certificates of websites. The likelihood of Internet users entering their credentials is relatively low if an attacker does not accurately mimic the visual design of a targeted website. An attacker fools the user in the following ways:

- a) **Visual Appearance:** The phishing website looks like its legitimate website. Attackers used to copy the HTML source code of a genuine website to build the fake website.
- b) **Address Bar:** Attackers also cover the address or URL bar of the website by script or image. The user would think they are entering their data on the correct website. 2017 (Ankit & Gupta).

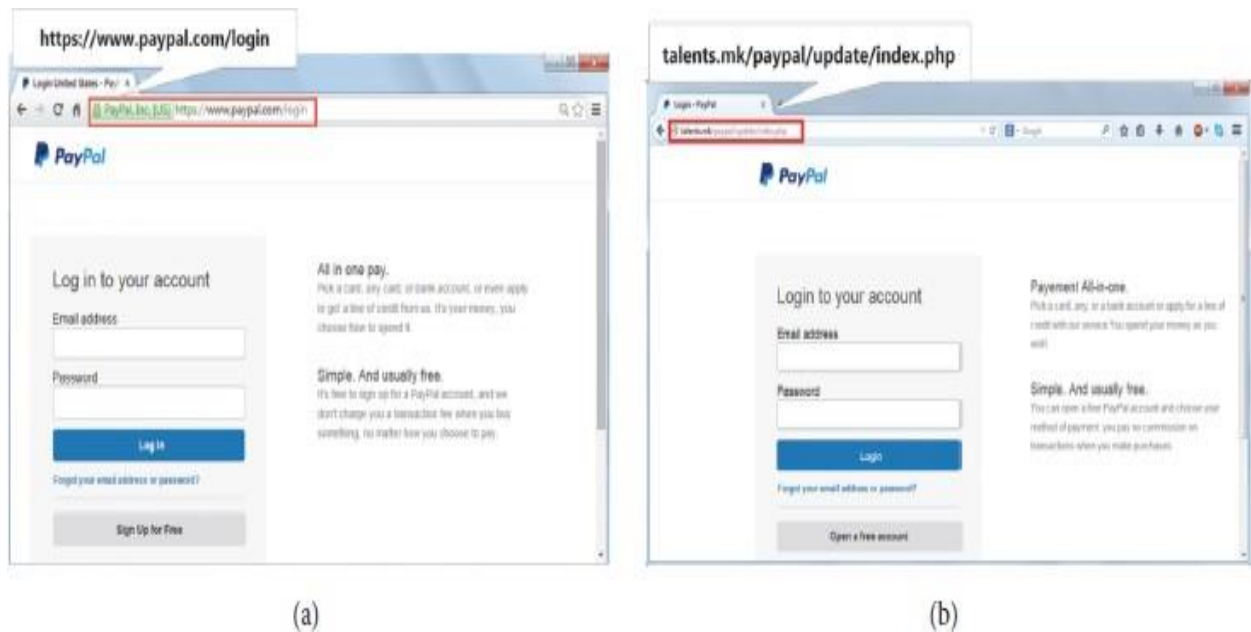


Figure 2.11 (a) Legitimate PayPal webpage and (b) phishing webpage of PayPal. (Source: Ankit & Gupta, 2017).

2.4 Empirical Review

In 2017, the paper titled A Theoretical Study on Different ways to Identify the Phishing URL and Its Prevention Approaches by V. Karamchand Gandhi and Dr.

M. Suriakala works on identifying the general format and procedure of phishing URLs by analyzing 200 phishing email archives provided by APWG. Based on the analysis, they find out the suitable approaches to prevent them from the fraudulent process.

The strengths of the paper are that the approaches, technical techniques, and classification of phishing hyperlinks in phishing e-mails enhance better results in the identification and prevention of phishing attacks.

The weakness of the paper is that the paper faces the challenge of inadequate data, as any machine learning algorithm can only give correct readings/predictions if it is applied to reliable data.

To address this weakness, this project will ensure proper and accurate dataset collection and pre-processing of updated phishing emails from the Anti-Phishing Working Group (APWG).

In 2016, the paper titled A Literature Review on Phishing Crime, Prevention Review and Investigation of Gaps by Anjum N. Shaikh, Antesar M. Shabat, and M.A. Hossain produced comprehensive research in which different reviews of previous pieces of literature are presented. In order to combat phishing, the report suggests using CRI, which stands for Crime, Prevention Review, and Investigation of Research Gap.

The strengths of the paper state the theory of phishing crime, a review of the anti-phishing techniques offered by different research, and an investigation of the research gaps. This research aims to develop a practice to understand the growing threats of phishing under a theoretical model of CRI and discover new literature.

The weakness of the paper is that there is need for a thorough research in terms of evaluating the effectiveness of countermeasures for phishing and developing a holistic technique to generating blacklists that are free from the demerits mentioned in the paper. Also, the paper didn't implement the research into a phishing detection system particularly against phishing emails since it is considered the most common way of attack.

To address this weakness, this project will ensure proper and accurate evaluation from different resources on Blacklisting and Countermeasures to a Phishing attack. Also, this project will develop a phishing detection system to validate phishing emails

In 2005, the paper titled Protecting users against phishing attacks with AntiPhish by Engin Kirda and Christopher Kruegel presents AntiPhish, a browser extension that aims to protect inexperienced users against spoofed website-based phishing attacks. AntiPhish keeps track of the sensitive information of a user and generates warnings whenever sensitive information is typed into a form on a website that is considered untrusted. The program is available for free public use and has been built as a Mozilla Firefox plug-in.

The strengths of the paper give an overview of the different types of phishing attacks and discuss a real-world example. It also describes the solution of AntiPhish, and provides details about its implementation, as well as presents an example that shows how AntiPhish mitigates a typical phishing attempt.

The weakness of the paper is that despite that AntiPhish is free for public use, the AntiPhish browser extension does not support other Web browsers such as Chrome, Brave, and Internet Explorer but is limited to only the Mozilla browser.

To address this weakness, a cross-platform plug-in is developed for any browser to ensure AntiPhish is not limited to a specific browser since it is free for public use.

In 2019, a paper by Ashritha Jain R, Chaithra Kulal, Deekshitha S, and Mrs. Mangala Kini titled A Review Paper on Detection of Phishing Websites Using Machine Learning proposed various algorithms (models), as well as various features of phishing attacks and techniques to detect phishing websites.

The strengths of the paper discuss works and different methods for phishing detection. It also introduces the proposed methodology that can be implemented to predict the phishing website accurately. The investigation gap provides more scope to study phishing detection.

The weakness of the paper is that it is limited to few feature selections of URL websites as well as four models for the prediction of phishing websites which might not produce the best accuracy of the model.

To address this weakness, several resources are sourced from updated published work to uncover different Machine learning models and features of a website to test and train the dataset for accuracy of the model. This will help find the best accuracy of the model for the detection of a phishing website.

The article Phishing Websites Detection Using Machine Learning by Kirthiga R. and Akila D. was published in 2019. outlined an innovative technique for employing machine learning algorithms to identify phishing websites. They also contrasted the performance of five machine learning algorithms: Generalized Linear Model (GLM), Generalized Additive Model (GAM), Gradient Boosting (GBM), Random Forest (RF), and Decision Tree (DT) (Shad & Sharma, 2018). Each algorithm's accuracy, precision, and recall assessment metrics were computed and compared. Website attributes (30) are extracted with the help of Python and performance evaluation done with open-source programming language R. Top three algorithms namely Decision Tree, Random Forest, and GBM performance were compared in the table. The Random Forest algorithm yielded the maximum 98.4% accuracy, 98.59% recall, and 97.70% precision, according to the tables of accuracy, recall, and performance.

The strengths of the paper are that different authors proposed different approaches for feature selection algorithm, automatic feature extraction, and phishing detection model to detect phishing performance effectively by using different machine learning models.

In this paper, the authors Sönmez, Tuncer, Gökal, & Avci (2018) proposes a classification model to classify phishing attacks. This model comprises feature extraction from sites and the classification of websites. Thirty features have been collected from the UCI Irvine machine learning repository data set for feature extraction, and the principles for phishing feature extraction have been outlined. Support Vector Machine (SVM), Naive Bayes (NB), and Extreme Learning Machine (ELM) were used to classify the data using these features (Sönmez et al., 2018). Six activation functions were utilized in the Extreme Learning Machine (ELM), which outperformed SVM and NB in accuracy with 95.34%. MATLAB was used to help with the results.

The Authors Peng, Harris, & Sawa, (2018) present an approach to detect phishing email attacks using natural language processing and machine learning. This is used to perform the semantic analysis of the text to detect malicious intent. A natural Language Processing (NLP) technique is used to parse each sentence and finds the semantic jobs of words in the sentence in connection to the predicate. This technique determines whether the statement is a command or an inquiry based on the function of each word in the sentence. The blacklist of harmful pairs is created using supervised machine learning (Peng et al., 2018). For the purpose of identifying phishing emails, authors developed the SEAHound algorithm (Peng et al., 2018), and the Net-craft Anti-Phishing Toolbar is used to validate a URL. The Nazario phishing email set dataset is utilized in conjunction with Python programs to create this technique. Results of Net-craft and SEAHound (Peng et al., 2018) are compared and obtained precision 98% and 95% respectively.

The weakness of the paper is that it is limited to a few feature selections and extraction of URL websites.

To address this weakness, new techniques and features should be added or replaced to help detect new phishing URL attacks.

The Table below shows related algorithms proposed by several researchers in Machine Learning to detect phishing websites. On reviewing their papers, they concluded that most of the work done is by using familiar machine learning algorithms like Naïve Bayesian, SVM, Decision Tree, and Random Forest. Some authors suggested a brand-new system for fraud detection, similar to Phish Score and Phish Checker. The combinations of features with regards to accuracy, precision, recall, etc. were used. Experimentally successful techniques in detecting phishing website URLs were summarized in Table 2.3. As phishing websites increase day by day, some features may be included or replaced with new ones to detect them.

Table 2.3: Outline of related algorithms used to detect phishing website

Algorithm	Reference paper	No. of Features	Dataset	Language/Tools	Conclusion
Decision Tree (DT), Random Forest (RF), Gradient Boosting (GBM), Generalized Linear Model (GLM),	J. Shad and S. Sharma,	30	Not Mentioned	Python, R Language	Random Forest highest accuracy 98.4%
Support vector Machine (SVM), Naïve Bayes (NB) and Extreme Learning Machine(ELM)	Y. Sönmez, T. Tuncer, H. Gökal, and E. Avci,	30	UCI-Machine Learning Repository	MATLAB	ELM achieved 95.34% accuracy.
Natural Language Processing	T. Peng, I. Harris, and Y.	-	Nazario Phishing Email set	Python	Proposed SEAHound

	Sawa,				provides 95% accuracy
Bayesian Network, Stochastic Descent (SGD), lazy. K. Star, Randomizable Filtered Classifier, Logistic model tree (LMT) and ID3 (iterative Dichotomiser), Multilayer Perception, JRip, PART. J48, Random Forest and Random Tree	M. Karabatak and T. Mustafa	27	UCI-Machine Learning Repository	MATLAB, WEKA	Lazy. K.Star obtained 97.58% accuracy
Random Forest	S. Parekh, D. Parikh, S. Kotak, and P. S. Sankhe	8	Phishtank,	RStudio	95% accuracy
Neural network model Adam, AdaDelta and SGD	K. Shima	URL length	Phishtank	Chainer	Accuracy of Adam 94.18%
Convolution neural network (CNN) and SNN long short-term memory (CNN-LSTM)	A. Vazhayil, R. Vinayakumar, and K. Soman,	-	Phishtank, OpenPhish, MalwareDomainlist, MalwareDomain	TensorFlow in conjunction with Keras	CNN-LSTM obtained 98% accuracy

Logistic Regression and Support Vector Machine (SVM)	W. Fadheel, M. Abusharkh, and I. Abdel-Qader,	19	UCI machine learning repository	Big Data	SVM accuracy 95.62%
AdaBoost, Bagging, Random Forest, and SMO	X. Zhang, Y. Zeng, X. Jin, Z. Yan, and G. Geng	11	DirectIndustry Anti-Phishing Alliance of China	Big Data	Only Semantic Features of word embedding obtained high accuracy.
C4.5 decision tree	L. MacHado and J. Gadge,	9 features and heuristic values	Phishtank Google	-	89.40%
KNN, SVM and Random Forest	A. Desai, J. Jatakia, R. Naik, and N. Raul	22	UCI-Machine Learning Repository	HTML, JavaScript, CSS, Python	Random Forest high accuracy
Naïve Bayes and Sequential Minimal Optimization (SMO)	M. Aydin and N. Baykal	133	Phishtank Google	C# programming and R programming WEKA	SMO Beat accuracy than NB
Heuristic feature root mean square Error (RMSE)	L. A. T. Nguyen, B. L. To, H. K. Nguyen, and M. H. Nguyen,	6	PhishTank	MYSQL. PHP	97%
PhishScore	S. Marchal, J. Francois, R. State,	12	PhishTank	-	94.91%
PhishChecker	A. A. Ahmed and N. A. Abdullah,	5	PhishTank and Yahoo directory set	Microsoft Visual Studio Express 2013 and C# language	96%

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.1 Overview of System Analysis

This chapter describes the various process, methods, and procedures adopted by the researcher to achieve the set aim and objectives and the conceptual structure within which the research was conducted.

The methodology of any research work refers to the research approach adopted by the researcher to tackle the stated problem. Since the efficiency and maintainability of any application are solely dependent on how designs are prepared. This chapter includes thorough explanations of the techniques used to offer solutions to the research's stated aims.

According to the Merriam-Webster dictionary's 11th edition, system analysis is "the process of studying a procedure or business to identify its goals and purposes and create systems and procedures that will efficiently achieve them." It is also the act, practice, or profession of investigating an activity (such as a procedure, a business, or a physiological function) using mathematical methods in order to specify its objectives and identify the best ways to carry them out. System analysis is used in every field where the development of something is done. Before planning and development, you need to understand the old systems thoroughly and use the knowledge to determine how best your new system can work.

3.2 Analysis of Existing System

When new phishing strategies are developed, the current system of phishing detection techniques suffers from low detection accuracy and high false alarm rates. Beyond that, the most popular solution is the blacklist-based approach, which is ineffective at stopping phishing attempts from spreading because it is now simpler to register a new domain. No exhaustive blacklist can provide a perfect up-to-date database for phishing detection.

3.3 Proposed System

The proposed phishing detection system utilizes machine learning models and deep neural networks. The system comprises one major part, which is the machine learning models. These models consist of Decision Tree, Support Vector Classifier, XG-Booster, CatBoost, Naïve Bayes, K-Nearest neighbors, Multilayer perceptions, Gradient Boosting, Logistic Regression and Random Forest.

These models are selected after different comparison-based performances of multiple machine learning algorithms. Each of these models is trained and tested on a website content-based feature, extracted from both phishing and legitimate dataset.

Hence, the model with the highest accuracy is selected and integrated into a web application that will enable a user to predict if a URL link is phishing or legitimate.

3.3.1 Benefits of the new system

- I. Will be able to differentiate between phishing (0) and legitimate (1) URLs
- II. It Will help reduce phishing data breaches for an organization
- III. It Will be helpful to individuals and organizations
- IV. It is easy to use

3.4 Model Development

The model development method takes several models, tests them, and adds them to an iterative process until a model that meets the required requirements is developed. Figure 3.1 shows the steps used in the development of machine learning models using both supervised and unsupervised learning.

The stages of developing a machine learning model for phishing detection systems are as follows:

i. Data Collection

The data used to generate the datasets on which the models are trained are gotten from different open-source platforms. The dataset collection consists of phishing and legitimate URL dataset.

The set of phishing URLs are collected from an open-source service called Phish Tank. This site offers a collection of phishing URLs that are updated every hour in a variety of forms, including CSV, JSON, and others. You can get this dataset by visiting the phishtank.com website. Over 5000 randomly selected phishing URLs are gathered from this dataset to train the ML models.

The set of legitimate URLs are obtained from the open datasets of the University of New Brunswick, this dataset is accessible on the university website. This dataset has a collection of benign, spam, phishing, malware & defacement URLs. The benign URL dataset is taken into consideration for this study out of all of these types. From this dataset, over 5000 random legitimate URLs are collected to train the ML models.

ii. Preprocessing

Data preprocessing is the first and crucial step after data collection. The raw dataset obtained for phishing detection was prepared by removing redundant and irregular data and encoded using the One-Hot Encoding technique into a useful and efficient format suitable for the machine learning model.

iii. Exploratory data analysis

Exploratory data analysis (EDA) technique was used on the dataset after series of data cleaning. The data visualization method was employed to analyze, explore, and summarize the dataset. These visualizations consist of heat-map, histograms, box plots, scatter plots, and pair plots to uncover patterns and insights within data.

iv. Feature Extraction

By generating new features from the current ones, feature extraction tries to decrease the number of features in a dataset. Thus, Website content-based features were extracted from phishing and legitimate datasets such as the Address bar-based feature which consists of 9 features, Domain-based feature which consists of 4 features, and HTML & JavaScript-based Feature which consists of 4 features. So, altogether 17 features were extracted for phishing detection.

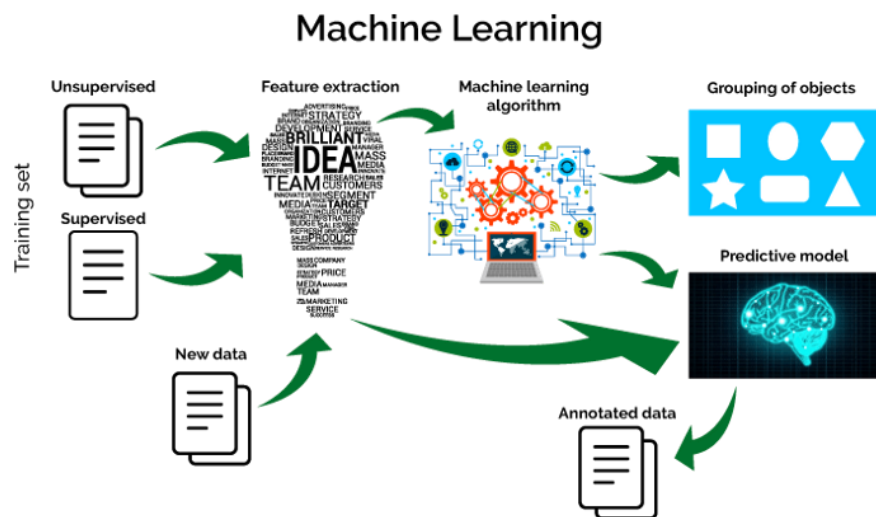


Figure 3.1 Machine Learning development process (Source: Ayush, 2019)

v. Model Training

In order to help find and understand positive qualities of the dataset, model training entails supplying machine learning algorithms with data. This research problem is a product of supervised learning, which falls under the classification problem. The algorithms used for phishing detection consist of supervised machine learning models (4) and deep neural network (2) which was used to train the dataset. These algorithms include Decision Tree, Support Vector Classifier, XG-Booster, CatBoost, Naïve Bayes, K-Nearest neighbors, Multilayer perceptions, Gradient Boosting, Logistic Regression and Random Forest. All these models were trained on the dataset. Thus, the dataset is spitted into a training and testing set. The training model consists of 80% of the dataset to enable the machine learning models to learn more about the data and be able to distinguish between phishing and legitimate URLs.

vi. Model Testing

Model Testing involves the process where the performance of a fully trained model is evaluated on a testing set.

Thus, after 80% of data has been trained, 20% of the dataset is used to evaluate the trained dataset to see the performance of the models.

vii. Model Evaluation

Model Evaluation involves estimating the generalization accuracy of models and deciding whether the model performs better or not.

Thus, Scikit-learn (sklearn matrices) module was used to implements several score and utility functions to measure the classification performance to properly evaluate the models deployed for phishing detection.

3.5 System Modelling

System modeling involves the process of developing an abstract model of a system, with each model presenting a different view or perspective of the system. It is the process of representing a system using various graphical notations that shows how users will interact with the system and how certain parts of the system function. The proposed system was modeled using the following diagrams:

1. Architecture diagram
2. Use case diagram
3. Flowcharts

The proposed system will be implemented using Python Programming language along with different machine learning models and libraries such as pandas, scikit-learn, python who-is, beautiful-Soup, NumPy, seaborn, and matplotlib. Etc.

3.5.1 System architecture

Understanding how a system should be configured and creating its general structure are the main goals of architectural design. It demonstrates how the system's various parts interact to accomplish its primary goals. It is the procedure for determining the various components that make up a system as well as the framework for sub-system coordination and communication. The architectural design of the suggested system is shown graphically in the diagram below.

Figure 3.1 shows the architecture view of the proposed phishing detection system such that a user enters a URL link and the link moves through different trained machine learning and deep neural network models and the best model with the highest accuracy is selected. Thus, the selected model is deployed as an API (Application Programming Interface) which is then integrated into a web application. Hence, a user interacts with the web application which is accessible across different display devices such as computers, tablets, and mobile devices.

3.5.2 Use a case diagram of the system

The Use Case diagram describes the functionality of the system as designed from the requirements, it summarizes the details of a system and the users within the system. It represents the observable interactions between actors and the developing system as a

behavior diagram. Actors, the system, associated use cases, and relationships between them are all included in the use case diagram.

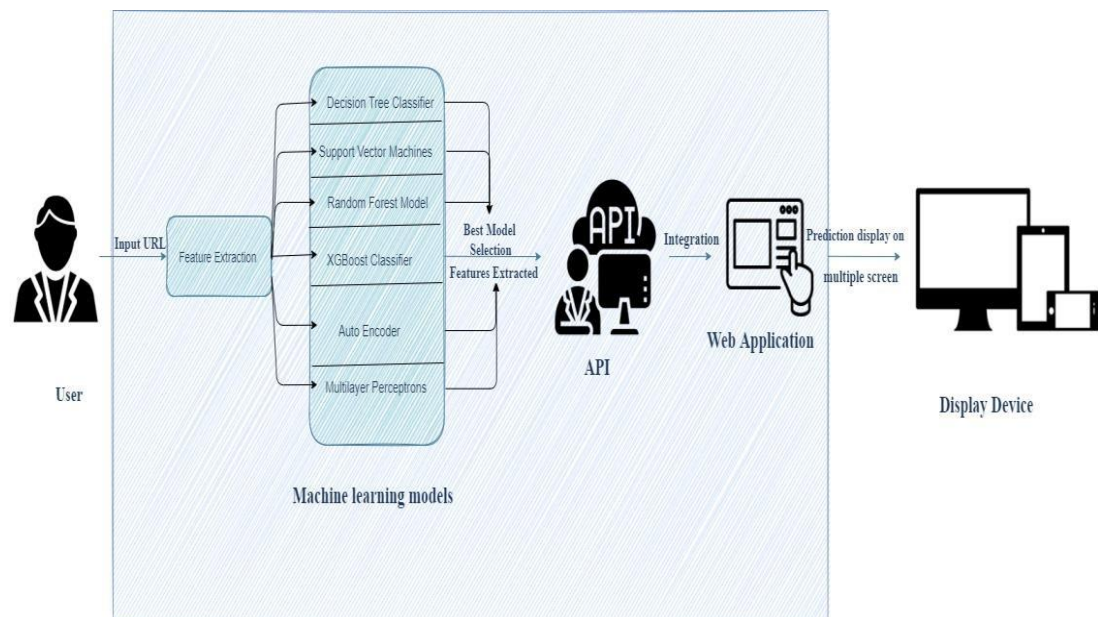


Figure 3.2 Architectural Design of the Proposed System

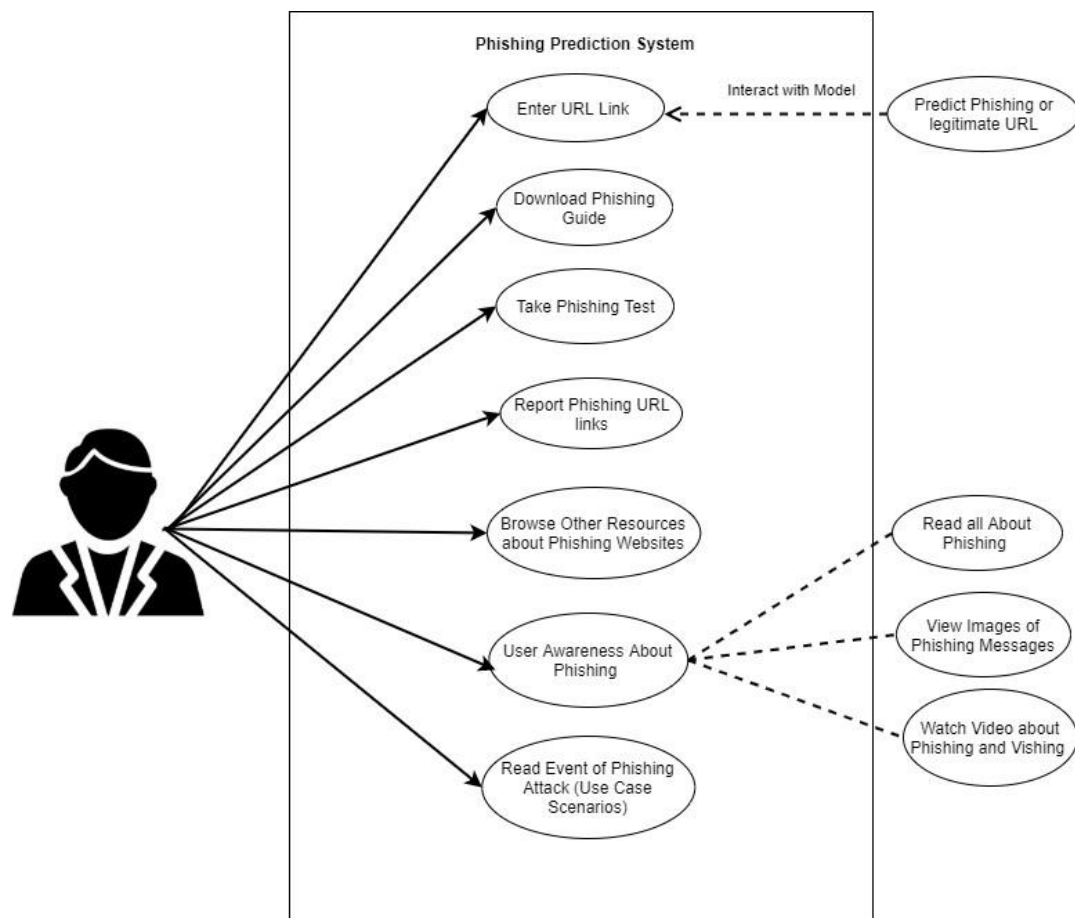


Figure 3.3 Use Case diagram for Proposed System

3.5.3 Flowchart of the system

An illustration of a system, computer algorithm, or process' operation is called a flowchart. It is a graphical representation of the steps that are to be performed in a system, it shows the steps in sequential order. It is used in presenting the flow of algorithms and to communicate complex processes in clear, easy-to-understand diagrams.

Figure 3.3 shows the flow of phishing detection systems using the machine learning process.

Figure 3.4 shows the phishing detection web interface system. The user inputs a URL link and the website validates the format of the URL and then predicts if the link is phishing or legitimate.

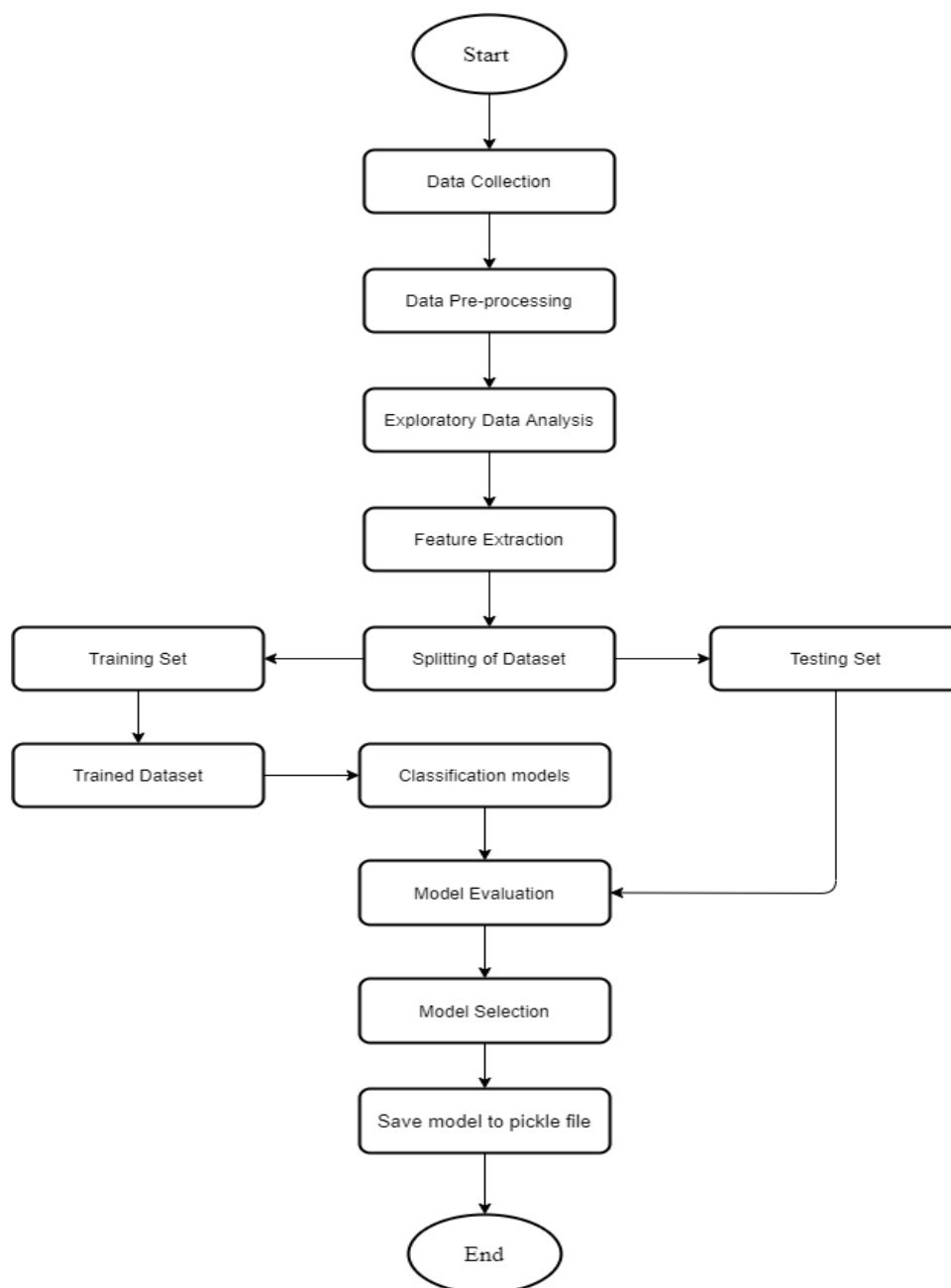


Figure 3.4 Flowchart of the proposed System

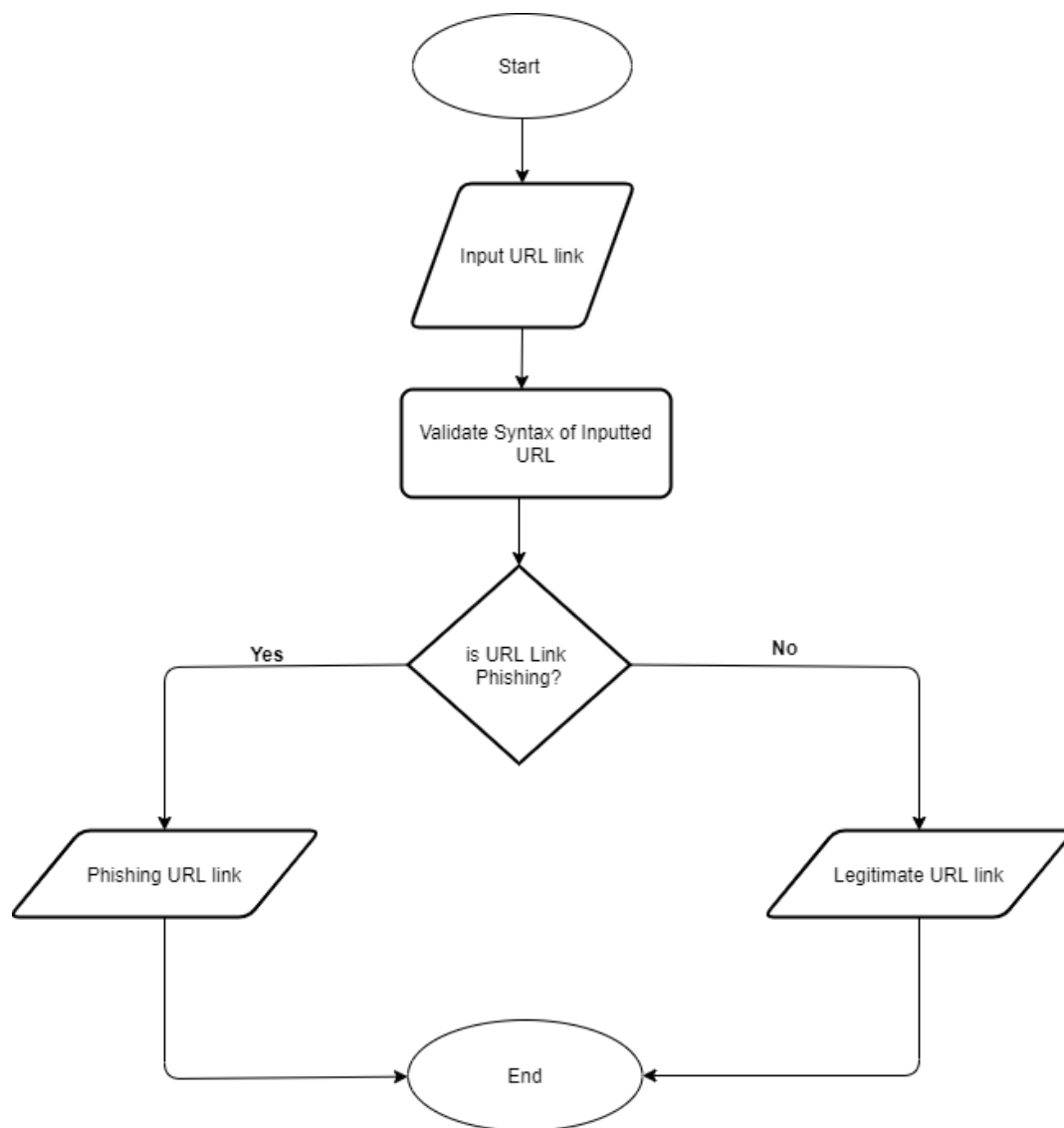


Figure 3.5 Flowchart of the web interface

CHAPTER FOUR

SYSTEM IMPLEMENTATION AND RESULTS

This chapter deals with the implementation of multiple machine learning models for the detection of underlying diseases and illnesses as earlier designed in the preceding chapters.

The implementation is concerned with all the activities that took place to put up the newly developed system into operation (using the approach that was stated in the methodology (e.g., architectural diagram, flowchart, uses case, etc.)) to achieve the objectives of the project to convert the theoretical design into a working system. The components of the system were also tested and evaluated.

4.1 Installation Requirements

The hardware (physical components of a computer system that can be seen, touched, or felt) and software (both system software and the application software installed and used in the system development) tools needed to satisfy these objectives highlighted below:

4.1.1 Hardware requirements

The hardware requirement includes:

1. A laptop or desktop computer (Preferably 64bit)
2. Random Access Memory (RAM): 8 Gigabytes Minimum
3. Processor: Intel Core i5, 2.4 GHz Minimum

4.1.2 Software requirements

The software requirements for the development of this system include:

1. Windows Operating System (8/10)
2. Anaconda Navigator (Jupyter Notebook)
3. PyCharm Community edition
4. Web browser (Preferably Chrome)
5. Visual Studio Code
6. Postman

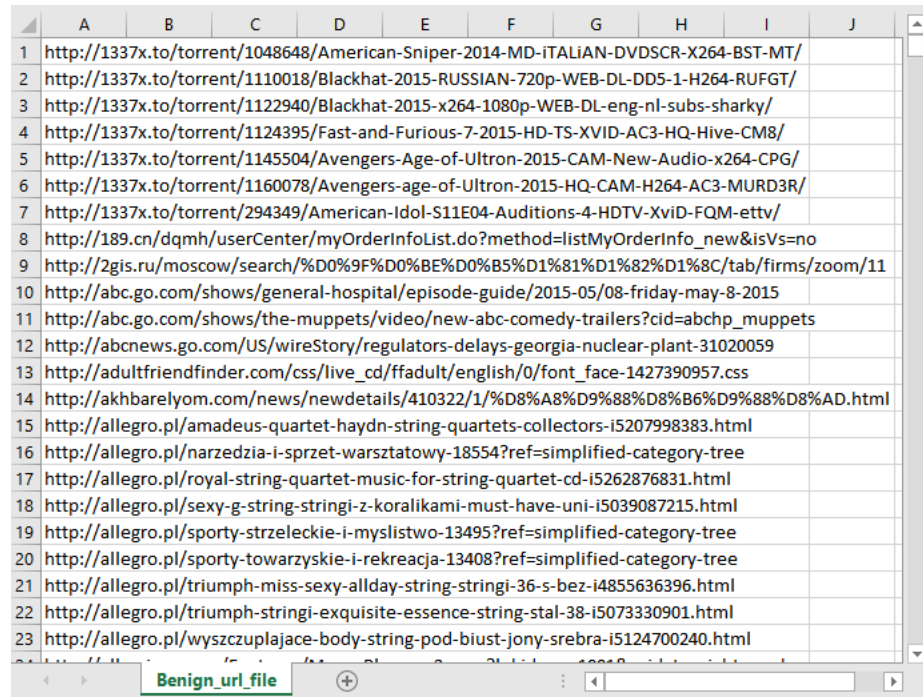
4.2 Model Development

The model for detecting phishing URL websites was built using a python programming language with over six (6) machine learning models and deep neural network algorithms altogether and the most accurate test score on the tested 5,000 datasets were used.

4.2.1 Data collection

The dataset used for the classification was sourced from was gotten from multiple sources listed in the earlier stated methodology.

The dataset used for classifying the dataset into phishing and legitimate URLs was sourced from open-source websites, samples of which are shown below in figure 4.1 and 4.2 respectively.



	A	B	C	D	E	F	G	H	I	J
1	http://1337x.to/torrent/1048648/American-Sniper-2014-MD-ITALIAN-DVDSCR-X264-BST-MT/									
2	http://1337x.to/torrent/1110018/Blackhat-2015-RUSSIAN-720p-WEB-DL-DD5-1-H264-RUFGT/									
3	http://1337x.to/torrent/1122940/Blackhat-2015-x264-1080p-WEB-DL-eng-nl-subs-sharky/									
4	http://1337x.to/torrent/1124395/Fast-and-Furious-7-2015-HD-TS-XVID-AC3-HQ-Hive-CM8/									
5	http://1337x.to/torrent/1145504/Avengers-Age-of-Ultron-2015-CAM-New-Audio-x264-CPG/									
6	http://1337x.to/torrent/1160078/Avengers-age-of-Ultron-2015-HQ-CAM-H264-AC3-MURD3R/									
7	http://1337x.to/torrent/294349/American-Idol-S11E04-Auditions-4-HDTV-XviD-FQM-ettv/									
8	http://189.cn/dqmh/userCenter/myOrderInfoList.do?method=listMyOrderInfo_new&isVs=no									
9	http://2gis.ru/moscow/search/%D0%9F%D0%BE%D0%B5%D1%81%D1%82%D1%8C/tab/firms/zoom/11									
10	http://abc.go.com/shows/general-hospital/episode-guide/2015-05/08-friday-may-8-2015									
11	http://abc.go.com/shows/the-muppets/video/new-abc-comedy-trailers?cid=abchp_muppets									
12	http://abcnews.go.com/US/wireStory/regulators-delays-georgia-nuclear-plant-31020059									
13	http://adultfriendfinder.com/css/live_cd/ffadult/english/0/font_face-1427390957.css									
14	http://akhbarelyom.com/news/newdetails/410322/1/%D8%A8%D9%88%D8%B6%D9%88%D8%AD.html									
15	http://allegro.pl/amadeus-quartet-haydn-string-quartets-collectors-i5207998383.html									
16	http://allegro.pl/narzedzia-i-sprzet-warsztatowy-18554?ref=simplified-category-tree									
17	http://allegro.pl/royal-string-quartet-music-for-string-quartet-cd-i5262876831.html									
18	http://allegro.pl/sexy-g-string-stringi-z-koralikami-must-have-uni-i5039087215.html									
19	http://allegro.pl/sporty-strzeleckie-i-myslistwo-13495?ref=simplified-category-tree									
20	http://allegro.pl/sporty-towarzyskie-i-rekreacja-13408?ref=simplified-category-tree									
21	http://allegro.pl/triumph-miss-sexy-allday-string-stringi-36-s-bez-i4855636396.html									
22	http://allegro.pl/triumph-stringi-exquisite-essence-string-stal-38-i5073330901.html									
23	http://allegro.pl/wyszczuplajace-body-string-pod-biust-jony-srebra-i5124700240.html									

Figure 4.1 Dataset of Phishing URLs

Source: The Dataset is collected from an open-source service called Phish-Tank. This dataset consists of 5,000 random phishing URLs which are collected to train the ML models.

	A	B	C	D	E	F	G	H	I	J
1	phish_id	url	phish_det	submissio	verified	verificatio	online	target		
2	6911546	https://ja	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
3	6911545	http://po	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
4	6911536	https://sp	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
5	6911494	https://hy	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
6	6911483	http://sto	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
7	6911482	https://oc	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
8	6911481	https://tri	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
9	6911480	https://jo	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
10	6911467	https://su	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
11	6911466	https://cr	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
12	6911465	http://est	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
13	6911424	https://ek	http://ww	2021-01-0	yes	2021-01-0	yes	eBay, Inc.		
14	6911414	https://is	http://ww	2021-01-0	yes	2021-01-0	yes	Development Bank of Singa		
15	6911408	http://wir	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
16	6911403	http://ma	http://ww	2021-01-0	yes	2021-01-0	yes	ABSA Bank		
17	6911400	http://ww	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
18	6911398	https://bi	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
19	6911395	https://fg	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
20	6911394	http://fgh	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
21	6911389	https://wl	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
22	6911388	https://wl	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
23	6911382	http://clfi	http://ww	2021-01-0	yes	2021-01-0	yes	Other		

Figure 4.2 Dataset of Legitimate URLs

Source: The Dataset were obtained from the open datasets of the University of New Brunswick, the dataset consists of collections of benign, spam, phishing, malware & defacement URLs. Out of all these types, the benign URL dataset is considered for this project. This dataset consists of 5,000 random legitimate URLs which are collected to train the ML models.

4.2.2 Feature extraction on the datasets

The features extraction used on the dataset are categorized into

1. Address bar-based features
2. Domain-based features
3. Html & java-script based features

In figure 4.3, figure 4.4, and figure 4.5 the images show the list of code feature extraction done on the dataset while figure 4.6 shows the code computation for all the feature extraction used on the dataset.

3.1. Address Bar Based Features:

Many features can be extracted that can be considered as address bar base features. Out of them, below mentioned were considered for this project.

- Domain of URL
- IP Address in URL
- "@" Symbol in URL
- Length of URL
- Depth of URL
- Redirection "/" in URL
- "http/https" in Domain name
- Using URL Shortening Services "TinyURL"
- Prefix or Suffix "-" in Domain

Each of these features are explained and the coded below:

```
In [12]: # importing required packages for this section
import urllib.parse
import urlparse,urllib
import ipaddress
import re
```

3.1.1. Domain of the URL

Here, we are just extracting the domain present in the URL. This feature doesn't have much significance in the training. May even be dropped while training the model.

```
In [13]: # 1.Domain of the URL (Domain)
def getDomain(url):
    domain = urlparse(url).netloc
    if re.match(r"^(www\.)",domain):
        domain = domain.replace("www.", "")
    return domain
```

```
In [14]: # 2.Checks for IP address in URL (Have_IP)
def havingIP(url):
    try:
        ipaddress.ip_address(url)
        ip = 1
    except:
        ip = 0
    return ip
```

3.1.3. "@" Symbol in URL

Checks for the presence of "@" symbol in the URL. Using "@" symbol in the URL leads the browser to ignore everything preceding the "@" symbol and the real address often follows the "@" symbol.

If the URL has "@" symbol, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

```
In [15]: # 3.Checks the presence of @ in URL (Have_At)
def haveAtsign(url):
    if "@" in url:
        at = 1
    else:
        at = 0
    return at
```

3.1.4. Length of URL

Computes the length of the URL. Phishers can use long URL to hide the doubtful part in the address bar. In this project, if the length of the URL is greater than or equal 54 characters then the URL classified as phishing otherwise legitimate.

If the length of URL >= 54, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

```
In [16]: # 4.Finding the length of URL and categorizing (URL_Length)
def getLength(url):
    if len(url) < 54:
        length = 0
    else:
        length = 1
    return length
```

Figure 4.3 Code for Address bar-based feature extraction

```
In [26]: # 12.Web traffic (Web_Traffic)
def web_traffic(url):
    try:
        #Filling the whitespaces in the URL if any
        url = urllib.parse.quote(url)
        rank = BeautifulSoup(urllib.request.urlopen("http://data.alexa.com/data?cli=10&dat=s&url=" + url).read(), "xml").find(
            "REACH")["RANK"]
        rank = int(rank)
    except TypeError:
        return 1
    if rank < 100000:
        return 1
    else:
        return 0
```

3.2.3. Age of Domain

This feature can be extracted from WHOIS database. Most phishing websites live for a short period of time. The minimum age of the legitimate domain is considered to be 12 months for this project. Age here is nothing but different between creation and expiration time.

If age of domain > 12 months, the value of this feature is 1 (phishing) else 0 (legitimate).

```
In [27]: # 13.Survival time of domain: The difference between termination time and creation time (Domain_Age)
def domainAge(domain_name):
    creation_date = domain_name.creation_date
    expiration_date = domain_name.expiration_date
    if (isinstance(creation_date,str) or isinstance(expiration_date,str)):
        try:
            creation_date = datetime.strptime(creation_date,'%Y-%m-%d')
            expiration_date = datetime.strptime(expiration_date,'%Y-%m-%d')
        except:
            return 1
    if ((expiration_date is None) or (creation_date is None)):
        return 1
    elif ((type(expiration_date) is list) or (type(creation_date) is list)):
        return 1
    else:
        ageofdomain = abs((expiration_date - creation_date).days)
        if ((ageofdomain/30) < 6):
            age = 1
        else:
            age = 0
    return age
```

3.2. Domain Based Features:

Many features can be extracted that come under this category. Out of them, below mentioned were considered for this project.

- DNS Record
- Website Traffic
- Age of Domain
- End Period of Domain

Each of these features are explained and the coded below:

```
In [23]: !pip install python-whois

Requirement already satisfied: python-whois in c:\users\goodness\anaconda3\lib\site-packages (0.7.3)
Requirement already satisfied: future in c:\users\goodness\anaconda3\lib\site-packages (from python-whois) (0.18.2)

In [24]: # importing required packages for this section
import re
from bs4 import BeautifulSoup
import whois
import urllib
import urllib.request
from datetime import datetime
```

Figure 4.4 Code for domain-based features extraction

3.3.2. Status Bar Customization

Phishers may use JavaScript to show a fake URL in the status bar to users. To extract this feature, we must dig-out the webpage source code, particularly the "onMouseOver" event, and check if it makes any changes on the status bar

If the response is empty or onmouseover is found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

```
In [31]: # 16.Checks the effect of mouse over on status bar (Mouse_Over)
def mouseOver(response):
    if response == "":
        return 1
    else:
        if re.findall("<script>.+onmouseover.+</script>", response.text):
            return 1
        else:
            return 0
```

3.3.3. Disabling Right Click

Phishers use JavaScript to disable the right-click function, so that users cannot view and save the webpage source code. This feature is treated exactly as "Using onMouseOver to hide the Link". Nonetheless, for this feature, we will search for event "event.button==2" in the webpage source code and check if the right click is disabled.

If the response is empty or onmouseover is not found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

```
In [32]: # 17.Checks the status of the right click attribute (Right_Click)
def rightClick(response):
    if response == "":
        return 1
    else:
        if re.findall(r"event.button ?== ?2", response.text):
            return 0
        else:
            return 1
```

3.3. HTML and JavaScript based Features

Many features can be extracted that come under this category. Out of them, below mentioned were considered for this project.

- IFrame Redirection
- Status Bar Customization
- Disabling Right Click
- Website Forwarding

Each of these features are explained and the coded below:

```
In [29]: # importing required packages for this section
import requests
```

3.3.1. IFrame Redirection

IFrame is an HTML tag used to display an additional webpage into one that is currently shown. Phishers can make use of the "iframe" tag and make it invisible i.e. without frame borders. In this regard, phishers make use of the "frameBorder" attribute which causes the browser to render a visual delineation.

If the iframe is empty or response is not found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

```
In [30]: # 15. IFrame Redirection (iFrame)
def iframe(response):
    if response == "":
        return 1
    else:
        if re.findall(r"<iframe>[<frameBorder>]", response.text):
            return 0
        else:
            return 1
```

Figure 4.5 Code for Html & java-script based features extraction


```

In [40]: ▶ #Function to extract features
# There are 17 features extracted from the dataset
def featureExtractions(url):

    features = []
    #Address bar based features (9)
    features.append(getDomain(url))
    features.append(havingIP(url))
    features.append(haveAtSign(url))
    features.append(getLength(url))
    features.append(getDepth(url))
    features.append(redirection(url))
    features.append(httpDomain(url))
    features.append(prefixSuffix(url))
    features.append(tinyURL(url))

    #Domain based features (4)
    dns = 0
    try:
        domain_name = whois.whois(urlparse(url).netloc)
    except:
        dns = 1

    features.append(dns)
    features.append(web_traffic(url))
    features.append(1 if dns == 1 else domainAge(domain_name))
    features.append(1 if dns == 1 else domainEnd(domain_name))

    # HTML & Javascript based features (4)
    try:
        response = requests.get(url)
    except:
        response = ""
    features.append(iframe(response))
    features.append(mouseOver(response))
    features.append(rightClick(response))
    features.append(forwarding(response))
    # features.append(Label)

    return features

featureExtractions('http://www.facebook.com/home/service')

Out[40]: ['facebook.com', 0, 0, 0, 2, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0]

```

Figure 4.6 Code computation for all the feature extraction used dataset.

4.2.2 Data Analysis & Visualization

The image as shown in figure 4.7 shows the distribution plot of how legitimate and phishing datasets are distributed base on the features selected and how they are related to each other.

In figure 4.8 shows the plot of a correlation heat-map of the dataset. The plot shows correlation between different variables in the dataset.

In figure 4.9 and figure 4.10, it shows the feature importance in the model for Decision tree classifier and Random Forest classifier respectively.

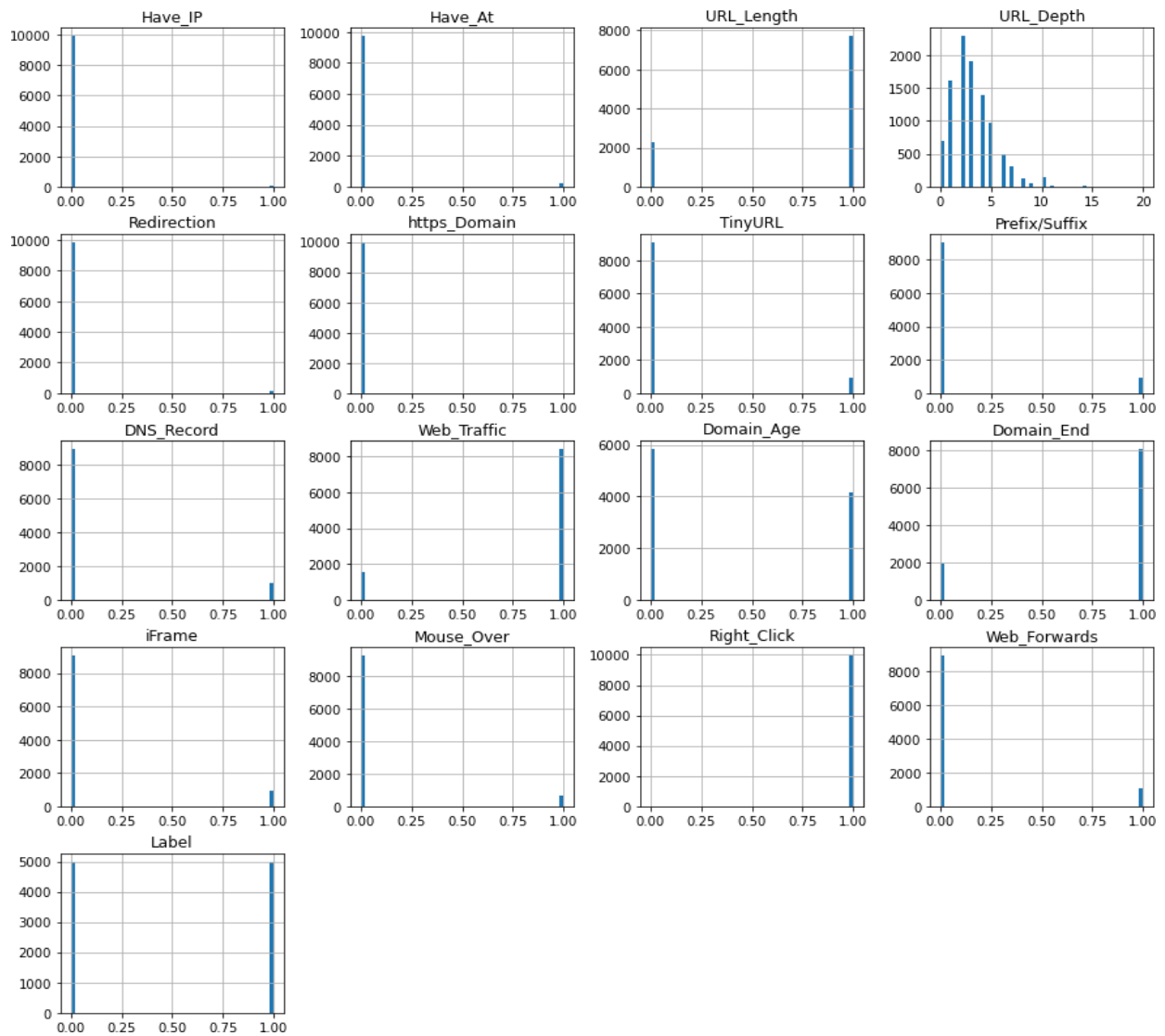


Figure 4.7 Distribution plot of dataset base on the features selected

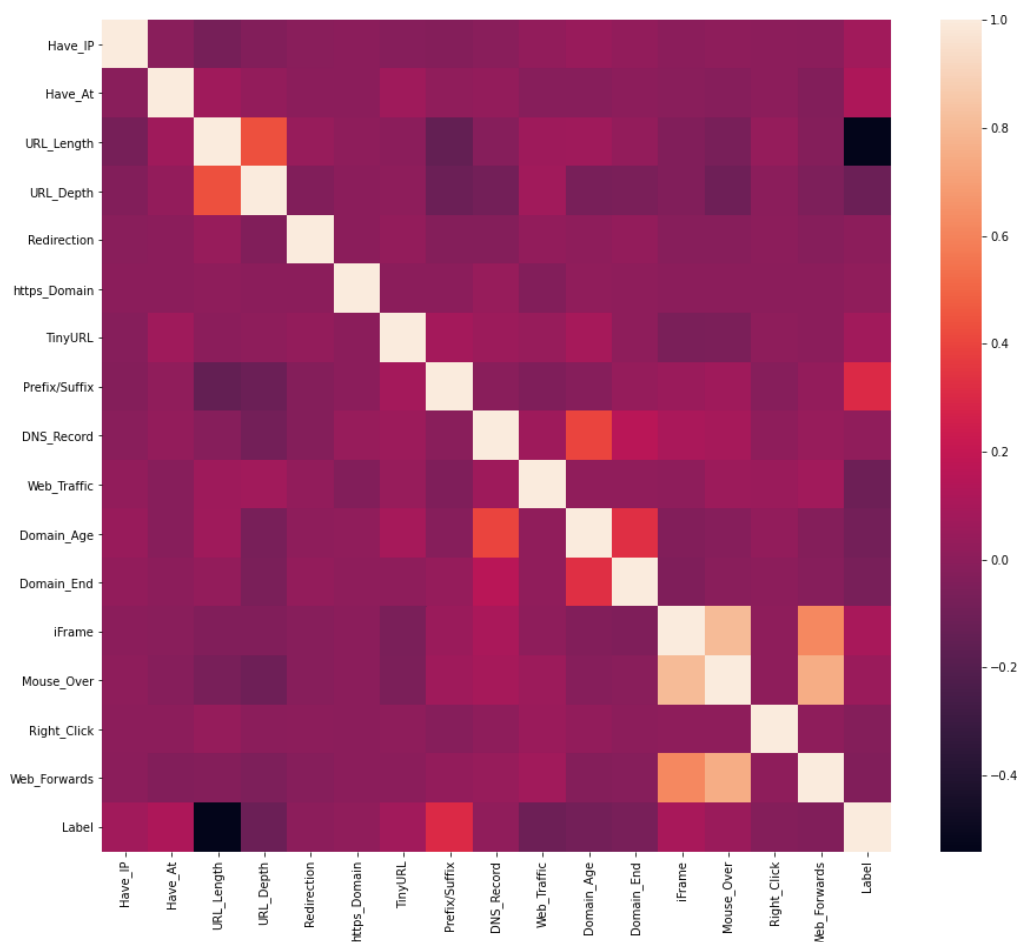


Figure 4.8 Correlation heat map of the dataset

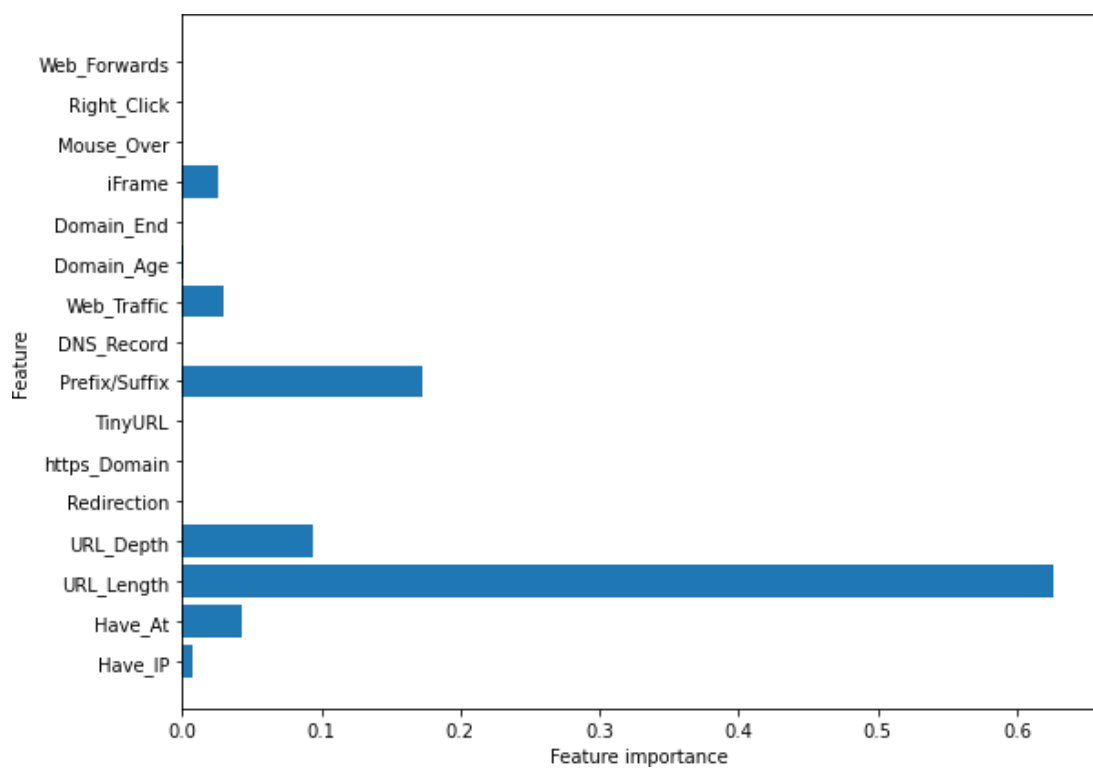


Figure 4.9 Feature importance for Decision Tree classifier

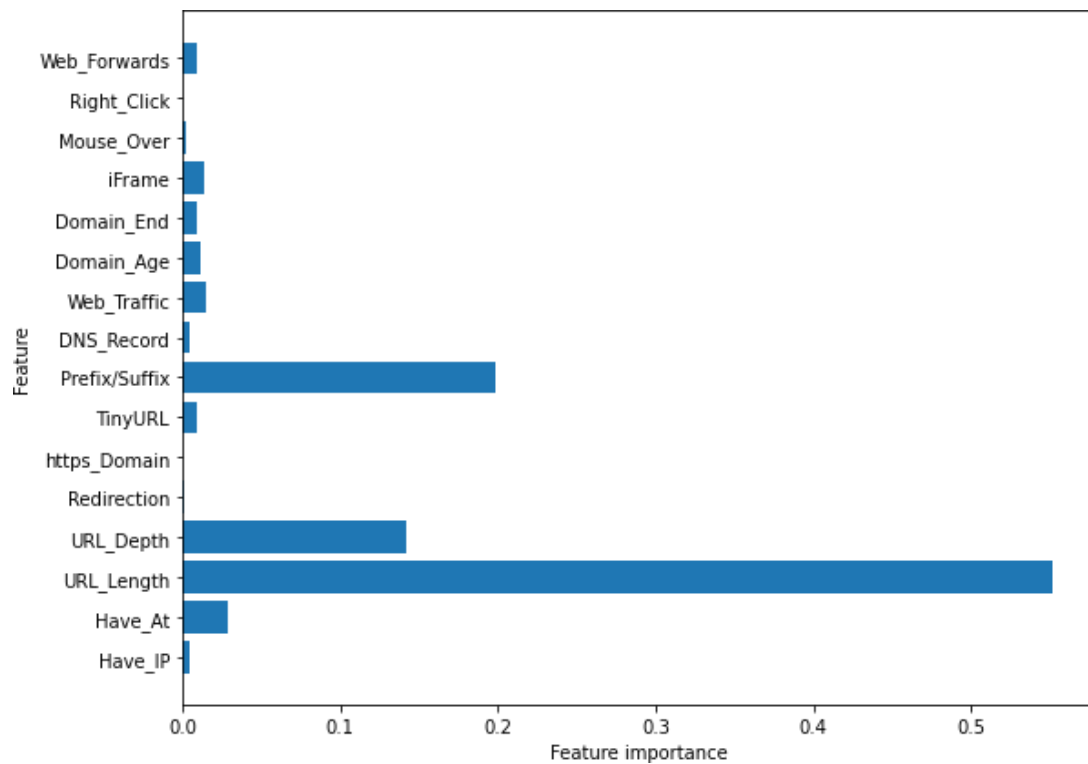


Figure 4.10 Feature importance for Random Forest classifier

4.2.3 Data pre-processing

The datasets were first cleaned to remove empty entries and fill some entries by applying data pre-processing techniques and transform the data to use in the models. Figure 4.11 shows the summary of the dataset while figure 4.12 shows the number of missing values in the dataset which all appear to be zero.

```
In [8]: tuna.describe()
```

Out[8]:

	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Do
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	100
mean	0.005500	0.022600	0.773400	3.072000	0.013500	0.000200	0.090300	0.093200	0.100800	0.845700	
std	0.073961	0.148632	0.418653	2.128631	0.115408	0.014141	0.286625	0.290727	0.301079	0.361254	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	1.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	
50%	0.000000	0.000000	1.000000	3.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	
75%	0.000000	0.000000	1.000000	4.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	
max	1.000000	1.000000	1.000000	20.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	

Figure 4.11 Summary of the dataset

```

In [10]: #checking the data for null or missing values
         dfsa.isnull().sum()

Out[10]: Have_IP          0
         Have_At          0
         URL_Length       0
         URL_Depth        0
         Redirection       0
         https_Domain      0
         TinyURL           0
         Prefix/Suffix     0
         DNS_Record        0
         Web_Traffic       0
         Domain_Age        0
         Domain_End        0
         iFrame            0
         Mouse_Over        0
         Right_Click       0
         Web_Forwards      0
         Label             0
         dtype: int64

```

Figure 4.12 Number of missing values in the dataset

4.2.4 Phishing detection model

The based methodology stated that the proposed system utilizes machine learning models and deep neural networks. These models consist of Decision Tree, Support Vector Classifier, XG-Booster, CatBoost, Naïve Bayes, K-Nearest neighbors, Multilayer perceptions, Gradient Boosting, Logistic Regression and Random Forest

The models determine whether a website URL is phishing or legitimate. The models help give a 2-class prediction (legitimate (0) and phishing (1)). In the model development process, over six (6) machine learning models and deep neural network algorithms all together were used to detect phishing URLs using Jupyter notebook IDE with packages such as pandas, Beautiful Soup, who-is, urllib, etc.

Here are the models, their accuracy was tested using sklearn matrices with an accuracy score and their matrices are shown in figure 4.13. The XGBooster model had the highest performance score of 86.6%, the Multilayer Perceptions model had an accuracy of 86.5%, the Decision Tree model had an accuracy of 81.4%, the Random Forest model had an accuracy of 81.8%, the Support Vector Machine model had an accuracy of 80.4%, and the Auto Encoder Neural Network model had an accuracy of 16.1%.

```

▶ #Sorting the dataframe on accuracy
results.sort_values(by=['Test Accuracy', 'Train Accuracy'], ascending=False)
49]:

```

	ML Model	Train Accuracy	Test Accuracy
3	XGBoost	0.866	0.864
2	Multilayer Perceptrons	0.865	0.864
0	Decision Tree	0.814	0.812
1	Random Forest	0.818	0.811
5	SVM	0.804	0.794
4	AutoEncoder	0.161	0.177

For the above comparison, it is clear that the XGBoost Classifier works well with this dataset.

So, saving the model for future use.

Figure 4.13 Accuracy performance of models

CHAPTER FIVE

SUMMARY

5.1 Summary

Phishing attacks are a rapidly expanding threat in the cyber world, costing internet users billions of dollars each year. It involves the use of a variety of social engineering tactics to obtain sensitive information from users. Hence, Phishing techniques can be detected using a variety of types of communication, including email, instant chats, pop-up messages, and web pages.

This project was able to categorize and recognize how phishers carry out phishing attacks and the different ways in which researchers have helped to solve phishing detection. Hence, the proposed system of this project worked with different feature selection and machine learning and deep neural networks such as Decision Tree, Support Vector Machine, XGBooster, Multilayer Perceptions, Auto Encoder Neural Network, and Random Forest to identify patterns in which URL links can be detected easily.

Users can enter website URL links to determine if they are authentic or phishing by using a web application that integrates the model with the best accuracy based on the feature extraction method used to distinguish phishing URL from legitimate URL links.

5.2 Contribution to Knowledge

This project provides a new and faster way to help users detect if a URL link is phishing or legitimate and provide them access to educational resources about phishing attacks.

CHAPTER SIX

CONCLUSION, AND RECOMMENDATIONS

6.1 Conclusion

The system developed detects if a URL link is phishing or legitimate by using machine learning models and deep neural network algorithms. The feature extraction and the models used on the dataset helped to uniquely identify phishing URLs and also the performance accuracy of the models used. It is also surprisingly accurate at detecting the genuineness of a URL link.

6.2 Recommendation

Through this project, one can know a lot about phishing attacks and how to prevent them. This project can be taken further by creating a browser extension that can be installed on any web browser to detect phishing URL Links.

CHAPTER SEVEN

REFERENCES

-  Almomani, A., Gupta, B. B., Atawneh, S., Meulenberg, A., & Almomani, E. (2013). A survey of phishing email filtering techniques, Proceedings of IEEE Communications Surveys and Tutorials, vol. 15, no. 4, pp. 2070–2090. <https://romisatriawahono.net/lecture/rm/survey/information%20retrieval/Almonani%20-%20Phishing%20Email%20Filtering%20Techniques%20-%202013.pdf>
-  Ashritha, J. R., Chaithra, K., Mangala, K., & Deekshitha, S. (2019). A Review Paper on Detection of Phishing Websites using Machine Learning. Proceedings of International Journal of Engineering. <https://www.ijraset.com/best-journal/phishing-websites-spotting-with-help-of-using-machine-learning-tools>
-  Workflow of a Machine Learning project. Retrieved from <https://towardsdatascience.com/workflow-of-a-machine-learning-project-ec1dba419b94>
-  Engine K., & Christopher K. (2005). Protecting Users Against Phishing Attacks. Proceedings of the Oxford University Press on behalf of The British Computer Society, Oxford University, 0, 2005, Retrieved from: https://sites.cs.ucsb.edu/~chris/research/doc/cj06_phish.pdf
-  KnowBe4 (2021). Phishing Techniques. Retrieved from <https://www.phishing.org/phishing-techniques>
-  Kondeti, P. S., Konka, R. C., & Kavishree, S. (2021). Phishing Websites Detection using Machine Learning Techniques. International Research Journal of Engineering and Technology, 08(4), Page 1471-1473. Retrieved from <https://www.irjet.net/archives/V8/i4/IRJET-V8I4274.pdf>
-  Will, B. (2019). 6 Different Ways to Compensate for Missing Values In a Dataset (Data Imputation with examples). Retrieved from <https://towardsdatascience.com/6-different-ways-to-compensate-for-missing-values-data-imputation-with-examples-6022d9ca0779>