

*Project Report on*

**DETECTION OF PHISHING WEBSITES USING**

**MACHINE LEARNING**



**BAKHTIYARPUR COLLEGE OF ENGINEERING**

BAKHTIYARPUR, PATNA, 803212

*A project report submitted in partial fulfilment of the requirement  
for the award of the degree*

**BACHELOR OF TECHNOLOGY**

*In*

**COMPUTER SCIENCE AND ENGINEERING**

(SESSION 2019-2023)

Under the Esteemed Supervision of Project Mentor

**PROF. PRAGYA CHAUDHARY**

**Submitted By:**

- **NITISH KUMAR** (19105126022)
- **YOGESH KUMAR SINGH** (19105126035)
- **SHREYA SOURABH** (19105126040)
- **ABHISHEK KUMAR** (19105126060)

## CERTIFICATION

This is to certify that the project, “**Detection of Phishing Websites Using Machine Learning**” was carried out by the project team members:

Name	Registration No.	Roll No.
Nitish Kumar	19105126022	19CS05
Yogesh Kumar Singh	19105126035	19CS37
Shreya Sourabh	19105126040	19CS44
Abhishek Kumar	19105126060	19CS01

For partial fulfillment of the requirement for B.Tech degree in **Computer Science and Engineering** from **Bakhtiyarpur College of Engineering**.

The project report demonstrates the participant's understanding and proficiency in the field of machine learning and its application in detecting and mitigating the threats posed by phishing websites. Through diligent research, analysis, and implementation, the participant has showcased their ability to develop an effective solution for combating online phishing attacks.

.....  
Signature of  
Project Supervisor

.....  
Signature of  
External Examiner

## DECLARATION

We hereby declare that this project report being submitted to the partial fulfilment of the 8th semester Project of Bihar Engineering University during the Academic Session 2019-23, is carried out under Supervision of **Miss Pragya Chaudhary**.

As the matter in the project has not yet been previously published or written by any other person as well as the details provided in the project report hold true to the best of our knowledge and belief.

We would also like to ensure that all the experiments, analyses, and evaluations conducted in this project report were performed with integrity and in accordance with ethical standards. Any data, results, or conclusions presented are authentic and have not been falsified or manipulated.

Name	Registration No.	Roll No.	Signature
Nitish Kumar	19105126022	19CS05	.....
Yogesh Kumar Singh	19105126035	19CS37	.....
Shreya Sourabh	19105126040	19CS44	.....
Abhishek Kumar	19105126060	19CS01	.....

.....  
**Prof. Pragya Chaudhary**  
Bakhtiyarpur College of Engineering

## ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to everyone who have contributed to the successful completion of our group project on "Detection of Phishing Websites Using Machine Learning". Their support, guidance, and expertise have been invaluable throughout this endeavor.

We would also like to thank our supervisor, Miss Pragya Chaudhary for her constant guidance and support throughout the duration of this project. We are indebted to the numerous researchers and experts in the field of machine learning and cybersecurity, whose pioneering work and publications have served as invaluable resources.

Most of all, we are grateful for our harmonious coordination and contribution that lead to the successful completion of the project within stipulated time.

## **ABSTRACT**

Phishing attacks are a rapidly expanding threat in the cyber world, costing internet users billions of dollars each year. It is a criminal crime that involves the use of a variety of social engineering tactics to obtain sensitive information from users. Phishing techniques can be detected using a variety of types of communication, including email, instant chats, pop-up messages, and web pages. This study develops and creates a model that can predict whether a URL link is legitimate or phishing.

The data set used for the classification was sourced from an open-source service called ‘Phish Tank’ which contain phishing URLs in multiple formats such as CSV, JSON, etc. and also from the University of New Brunswick dataset bank which has a collection of benign, spam, phishing, malware & defacement URLs. Over six (6) machine learning models and deep neural network algorithms all together are used to detect phishing URLs. This study aims to develop a web application software that detects phishing URLs from the collection of over 5,000 URLs which are randomly picked respectively and are fragmented into 80,000 training samples & 20,000 testing samples.

# **TABLE OF CONTENT**

<b><u>TITLE</u></b>	<b><u>PAGE NO.</u></b>
CERTIFICATION	i
DECLARATION	iii
ACKNOWLEDGEMENTS	iv
ABSTRACT	v
<b>1. CHAPTER 1: INTRODUCTION</b>	<b>9</b>
1.1 Statement of Problem	10
1.2 Aim of Study	10
1.3 Objective of the Study	10
1.4 Significance of the Project	11
1.5 Scope of the Study	11
1.6 Methodology	11
1.7 Architectural Design	12
<b>2. CHAPTER 2: CONCEPTUAL REVIEW</b>	<b>14</b>
2.1 Data Collection	14
2.2 Preprocessing	14
2.3 Exploratory Data Analysis	14
2.4 Feature Extraction	15
2.5 Model Training	16

<b>3. CHAPTER 3: FEATURE SELECTION</b>	<b>16</b>
<b>3.1 Address Bar-Based Features</b>	<b>18</b>
<b>3.2 HTML and JavaScript Based Features</b>	<b>21</b>
<b>3.3 Domain Based Features</b>	<b>22</b>
<b>4. CHAPTER 4: PROJECT REPORT ARRANGEMENT</b>	<b>25</b>
<b>4.1 Algorithm and Model Evaluation</b>	<b>25</b>
<b>4.2 Use a Case Diagram of the System</b>	<b>28</b>
<b>4.3 Hardware Requirement</b>	<b>30</b>
<b>4.4 Software Requirement</b>	<b>30</b>
<b>4.5 Prototype Development</b>	<b>30</b>
<b>5. CHAPTER 5: SYSTEM IMPLEMENTATION AND RESULTS</b>	<b>31</b>
<b>5.1 Installation Requirements</b>	<b>31</b>
<b>5.2 Model Development</b>	<b>32</b>
<b>5.3 General Working of the System</b>	<b>43</b>
CONCLUSION	45
BIBLIOGRAPHY	46

## **FIGURE OF CONTENT**

<b>Fig. No.</b>	<b>Figure Title</b>	<b>Page No.</b>
1.1	Architectural design of the proposed system	12
1.2	Flowchart of the proposed system	13
3.1	Feature selection process	17
3.2	Feature selection model	17
4.1	Use case of model accuracy	27
4.2	Use case diagram for proposed system	28
4.3	Flowchart of the web interface	29
5.1	Dataset of phishing URLs	32
5.2	Dataset of legitimate URLs	33
5.3	Code for address bar-based feature extraction	34
5.4	Code for domain-based feature extraction	35
5.5	Code for html & java-script based features extraction	36
5.6	Code computation for all the feature extraction used dataset.	37
5.7	Distribution plot of dataset base on the features selected	38
5.8	Correlation heat map of the dataset	39
5.9	Feature importance for Decision Tree classifier	40
5.10	Feature importance for Random Forest classifier	40
5.11	Summary of the dataset	41
5.12	Number of missing values in the dataset	41
5.13	Accuracy performance of models	42
5.14	Use case diagram of website	44



# CHAPTER 1

## INTRODUCTION

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. Upon clicking the link, the recipient is fooled into believing that they've received a gift and unsuspectingly clicks a malicious link, resulting in the installation of malware, the freezing of the system as part of a ransomware attack, or the disclosure of sensitive information.

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 most common method of phone phishing is to use a phony caller ID.
- 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.

### **1.1 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.

### **1.2 Aim of Study**

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

### **1.3 Objectives of the Study**

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

- i. dataset collection and pre-processing;
- ii. machine-learning model selection and development;
- iii. development of a web-based application for detection;
- iv. Integration of the developed model to a web application.

#### **1.4 Significance of the Project**

A phishing website is a fraudulent website designed to mimic the appearance and functionality of a legitimate website with the intent to deceive visitors and trick them into revealing sensitive information or performing certain actions. 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.

#### **1.5 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. This online application is compatible with a variety of browsers.

#### **1.6 Methodology**

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.

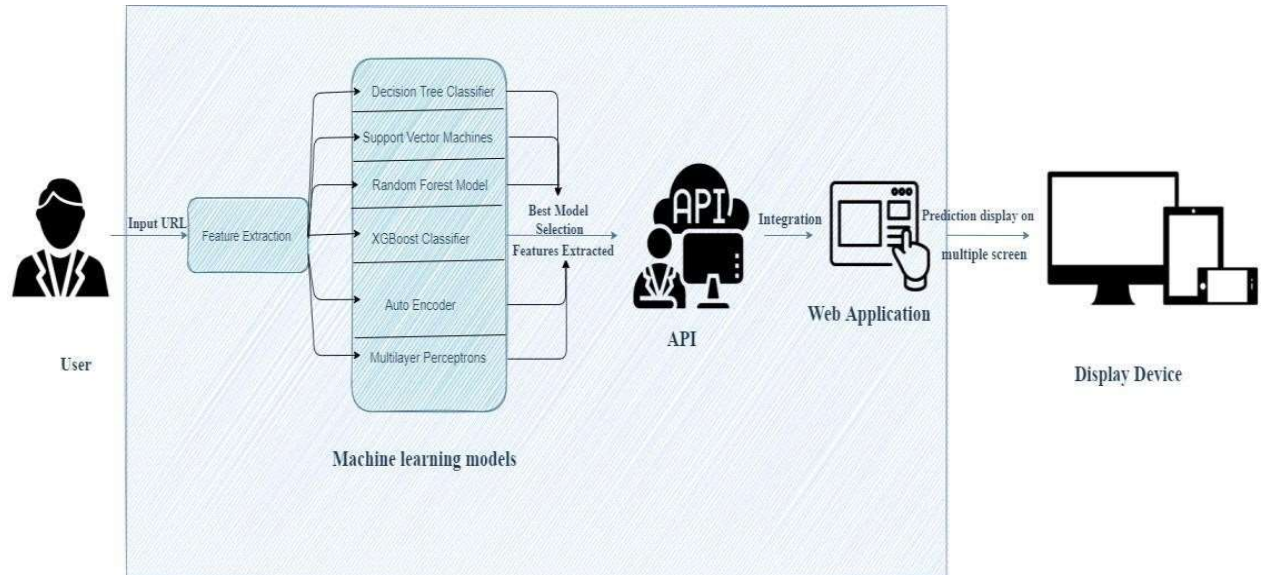


Figure 1.1: Architectural design of the proposed system

## 1.7 Architectural Design

- **Models:** Include the machine learning models you want to integrate into the system. These could be models for tasks like image recognition, natural language processing, recommendation systems, etc.
- **API Integration:** Create APIs (Application Programming Interfaces) for each model to enable communication between the models and the web application.
- **Web Application:** Develop a web application that serves as the user interface. Users will interact with the web application to perform various tasks that involve the integrated models.
- **Display Design:** Design the user interface of the web application, including layout, colors, typography, and overall user experience.

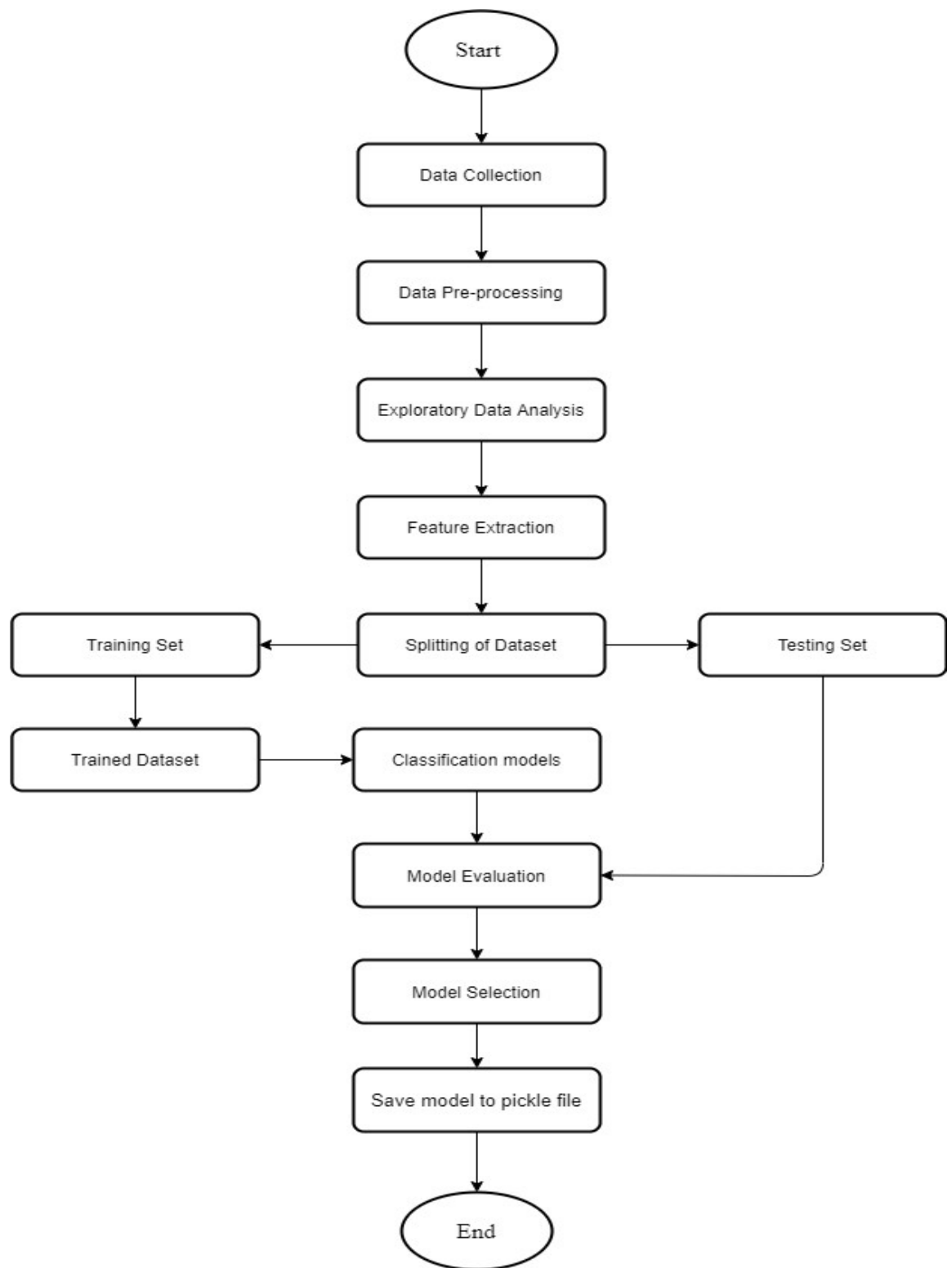


Figure 1.2: Flowchart of the proposed System

## **CHAPTER 2**

### **CONCEPTUAL REVIEW**

#### **2.1 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 service provides a set of phishing URLs in multiple formats like CSV, JSON and so on that gets updated hourly. This dataset is accessible from the phishtank.com website. From this dataset, over 5000 random phishing URLs are collected 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. Out of all these types, the benign URL dataset is considered for this project. From this dataset, over 5000 random legitimate URLs are collected to train the ML models.

#### **2.2 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 also encoded using the One-Hot Encoding technique into a useful and efficient format suitable for the machine learning model.

#### **2.3 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, boxplots, scatter plots, and pair plots to uncover patterns and insights within data.

## 2.4 Feature Extraction

Feature Selection is the method of reducing the input variable to your model by using only relevant data and getting rid of noise in data. 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. Feature Extraction aims to reduce the number of features in a dataset by creating new features from the existing ones. Thus, Website content-based features were extracted from phishing and legitimate datasets

- Address bar-based features – 9
- Domain-based features – 4
- Html & java-script based features - 4

## 2.5 Model Training

Model Training involves feeding Machine learning algorithms with data to help identify and learn good attributes of dataset. 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, Random Forest, Support Vector Machines, XGBooster, Multilayer Perceptron, and Auto-encoder Neural Network. 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.

## **CHAPTER 3**

### **FEATURE SELECTION**

Feature Selection is the method of reducing the input variable to your model by using only relevant data and getting rid of noise in data. 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 helps in cutting down the noise in our data and reducing the size of our input data. Figure 3.1 shows the feature selection process.

Feature selection models are of two types:

i. Supervised Models:

Supervised feature selection refers to the method which uses the output label class for feature selection. They use the target variables to identify the variables which can increase the efficiency of the model. The goal of the algorithm is to learn a mapping from inputs to outputs, so that it can make accurate predictions or classifications on new, unseen data. It is a foundational approach in machine learning where algorithms are trained on labeled data to make predictions or classifications on new data. It plays a crucial role in various real-world applications and continues to be a focus of research and development in the field of artificial intelligence.

ii. Unsupervised Models:

Unsupervised Feature selection refers to the method which does not need the output label class for feature selection. Unlike supervised learning, the algorithm works with unlabeled data, seeking to uncover inherent patterns or groupings without specific guidance. Unsupervised learning involves exploring and understanding data without explicit labels. It's used for tasks like clustering, dimensionality reduction, and anomaly detection, enabling insights into the underlying patterns of complex datasets.



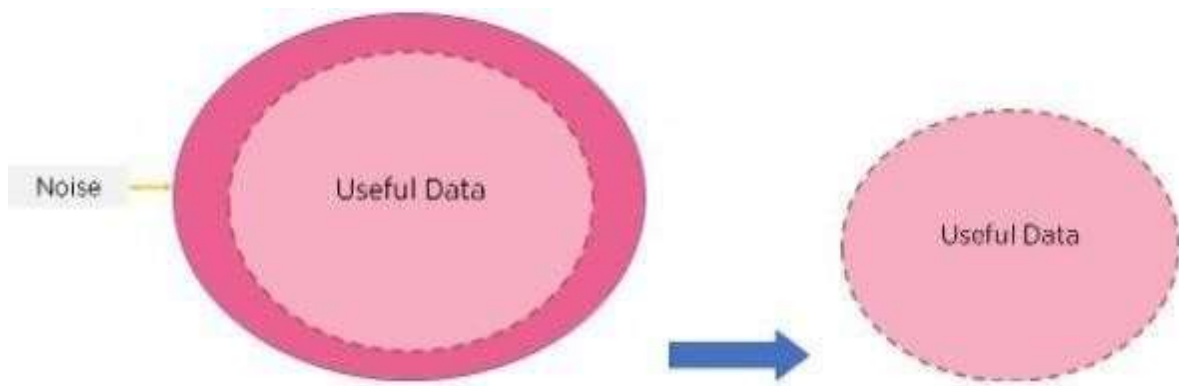


Figure 3.1: Feature selection process

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

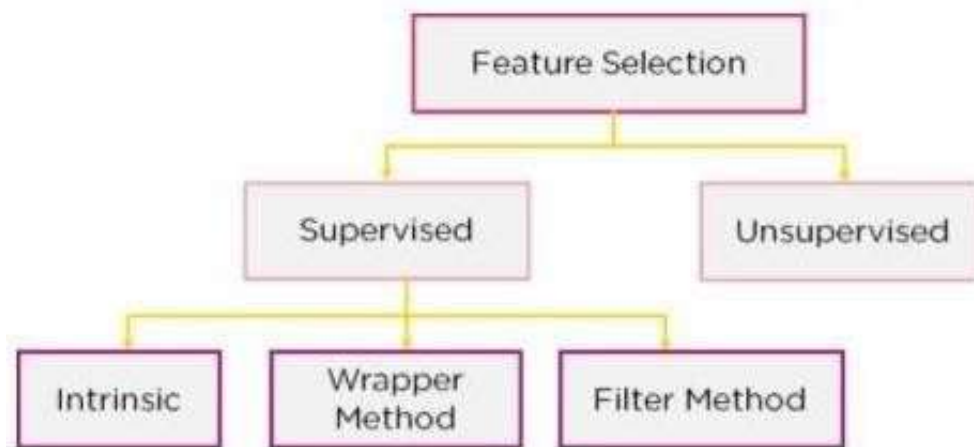


Figure 3.2: Feature selection models

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

### 3.1 Address Bar-Based Features

#### (1) Using the IP Address

If an IP address is used as an alternative to the domain name in the URL, such as “http://125.98.3.123/fake.html”, users can be sure that someone is trying to steal their personal information.

Rule: IF { If The Domain Part has an IP Address  $\rightarrow$  Phishing  
Otherwise  $\rightarrow$  Legitimate

#### (2) Long URL to Hide the Suspicious Part

Phishers can use a long URL to hide the doubtful part in the address bar to ensure the accuracy of our study, we calculated the length of URLs in the dataset and produced an average URL length.

$URL\ length < 54 \rightarrow feature = \text{Legitimate}$

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

#### (3) URL's having “@” Symbol

Using the “@” symbol in the URL leads the browser to ignore everything preceding the “@” symbol and the real address often follows the “@” symbol.

Url Having @ Symbol  $\rightarrow$  Phishing

Rule: IF { Otherwise  $\rightarrow$  Legitimate

#### (4) Redirecting using “//”

The existence of “//” within the URL path means that the user will be redirected to another website.

the position of the Last Occurrence of “//” in the URL  $> 7 \rightarrow$  Phishing

Rule: IF { Otherwise  $\rightarrow$  Legitimate

### **(5) Sub Domain and Multi-Sub Domains**

Let us assume we have the following link: <http://www.hud.ac.uk/students/>. A domain name might include the country-code top-level domains (ccTLD), which in our example is “UK”. The “ac” part is shorthand for “academic”, the combined “ac. UK” is called a second-level domain (SLD) and “hud” is the actual name of the domain. To produce a rule for extracting this feature, we first have to omit the (www.) from the URL which is a subdomain in itself. Then, we have to remove the (ccTLD) if it exists. Finally, we count the remaining dots. If the number of dots is greater than one, then the URL is classified as “Suspicious” since it has one subdomain. However, if the dots are greater than two, it is classified as “Phishing” since it will have multiple subdomains. Otherwise, if the URL has no subdomains, we will assign “Legitimate” to the feature.

Dots In Domain Part = 1 → Legitimate

Rule: IF { Dots In Domain Part = 2 → Suspicious

Otherwise → Phishing

### **(6) HTTPS (Hypertext Transfer Protocol with Secure Sockets Layer)**

The existence of HTTPS is very important in giving the impression of website legitimacy, but this is not enough so by testing out our datasets, we find that the minimum age of a reputable certificate is two years.

Use HTTPS and Issuer Is Trusted and Age of Certificate  $\geq$  1 Years → Legitimate

Rule: IF { Using https and Issuer Is Not Trusted → Suspicious

Otherwise → Phishing

### **(7) Domain Registration Length**

Based on the fact that a phishing website lives for a short period, we believe that trustworthy domains are regularly paid for several years in advance. In our dataset, we find that the longest fraudulent domains have been used for one year only.

Domains Expire on  $\leq 1$  years  $\rightarrow$  Phishing

Rule: IF {  
Otherwise  $\rightarrow$  Legitimate

### **(8) The Existence of “HTTPS” Token in the Domain Part of the URL**

The phishers may add the “HTTPS” token to the domain part of a URL to trick users. For example,

<http://https-www-paypal-it-webapps-mpp-home.soft-hair.com/>.

Rule: IF { Using HTTP Token in Domain Part of The URL  $\rightarrow$  Phishing  
Otherwise  $\rightarrow$  Legitimate

### **(9) Adding Prefix or Suffix Separated by (-) to the Domain**

The dash symbol is rarely used in legitimate URLs. Phishers tend to add prefixes or suffixes separated by (-) to the domain name so that users feel that they are dealing with a legitimate webpage. For example, <http://www.Confirme-paypal.com/>.

Domain Name Part Includes (–) Symbol  $\rightarrow$  Phishing

Rule: IF {  
Otherwise  $\rightarrow$  Legitimate

### 3.2 HTML and JavaScript-Based Features

#### a. 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.

ofRedirect Page  $\leq 1 \rightarrow$  Legitimate

Rule: IF { of Redirect Page  $\geq 2$  And  $< 4 \rightarrow$  Suspicious

Otherwise  $\rightarrow$  Phishing

#### b. 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.

OnMouseOver Changes Status Bar  $\rightarrow$  Phishing

Rule: IF { It Does't Change Status Bar  $\rightarrow$  Legitimate

#### c. 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.

Right Click Disabled  $\rightarrow$  Phishing

Rule: IF {  
Otherwise  $\rightarrow$  Legitimate

#### **d. 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.

Popup Window Contains Text Fields  $\rightarrow$  Phishing

Rule: IF { Otherwise  $\rightarrow$  Legitimate

#### **e. 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 { Popup Window Contains Text Fields  $\rightarrow$  Phishing  
Otherwise  $\rightarrow$  Legitimate

### **3.3 Domain-Based Features**

#### **a. 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.

Age Of Domain  $\geq$  6 months  $\rightarrow$  Legitimate

Rule: IF { Otherwise  $\rightarrow$  Phishing

### **b. 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”.

no DNS Record For The Domain  $\rightarrow$  Phishing

Rule: If { Otherwise  $\rightarrow$  Legitimate

### **c. 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”.

Website Rank  $< 100,000 \rightarrow$  Legitimate

Rule: IF { Website Rank  $> 100,000 \rightarrow$  Suspicious  
Otherwise  $\rightarrow$  Phish

### **d. 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”.

PageRank  $< 0.2 \rightarrow$  Phishing

Rule: IF { Otherwise  $\rightarrow$  Legitimate

#### **e. 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.

Rule: IF { Webpage Indexed by Google  $\rightarrow$  Legitimate  
Otherwise  $\rightarrow$  Phishing

#### **f. 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 pointing to them.

Of Link Pointing to The Webpage = 0  $\rightarrow$  Phishing

Rule: IF { Of Link Pointing to The Webpage  $> 0$  and  $\leq 2 \rightarrow$  Suspicious  
Otherwise  $\rightarrow$  Legitimate



## **CHAPTER 4**

### **PROJECT REPORT ARRANGMENT**

#### **4.1 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.

Classification is a machine learning algorithm where we get the labeled data as input and we need to predict the output into a class. If there are two classes, then it is called Binary Classification. If there are more than two classes, then it is called Multi-Class Classification.

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.

i.      **Decision Tree Algorithm**

One of the most widely used algorithms in machine learning technology. The decision tree algorithm is easy to understand and also 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. Decision tree creates training model which is used to predict target value or class in tree representation each internal node of the tree belongs to attribute and each leaf node of the tree belongs to the class label. In the decision tree algorithm, Gini index, and information gain methods are used to calculate these nodes.

ii.     **XGBooster**

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.

iii. Random Forest Algorithm

Random forest algorithm is one of the most powerful algorithms in machine learning technology and it is based on the concept of decision tree algorithm. Random forest algorithm creates the forest with several decision trees. A high number of a tree gives high detection accuracy. The creation of trees is based on the bootstrap method. In the bootstrap method features and samples of a dataset are randomly selected with replacement to construct a single tree. Among randomly selected features, a random forest algorithm will choose the best splitter for the classification, and as the decision tree algorithm; the Random Forest algorithm also uses Gini index and information gain methods to find the best splitter. This process will get continue until a random forest creates N number of trees. Each tree in the forest predicts the target value and then the algorithm will calculate the votes for each predicted target. Finally, the random forest algorithm considers the high-voted predicted target as a final prediction.

iv. Support Vector Machine Algorithm

Support vector machine is another powerful algorithm in machine learning technology. 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. 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.

v. 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 pass information to the predictors and process the output.

vi. 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.

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 is a behavior diagram and visualizes the observable interactions between actors and the system under development. The Use case diagram consists of the system, the related use cases, and actors and relates to each other.

```
#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.

Figure 4.1: Use case of model accuracy

#### 4.2 Use a case diagram of the system

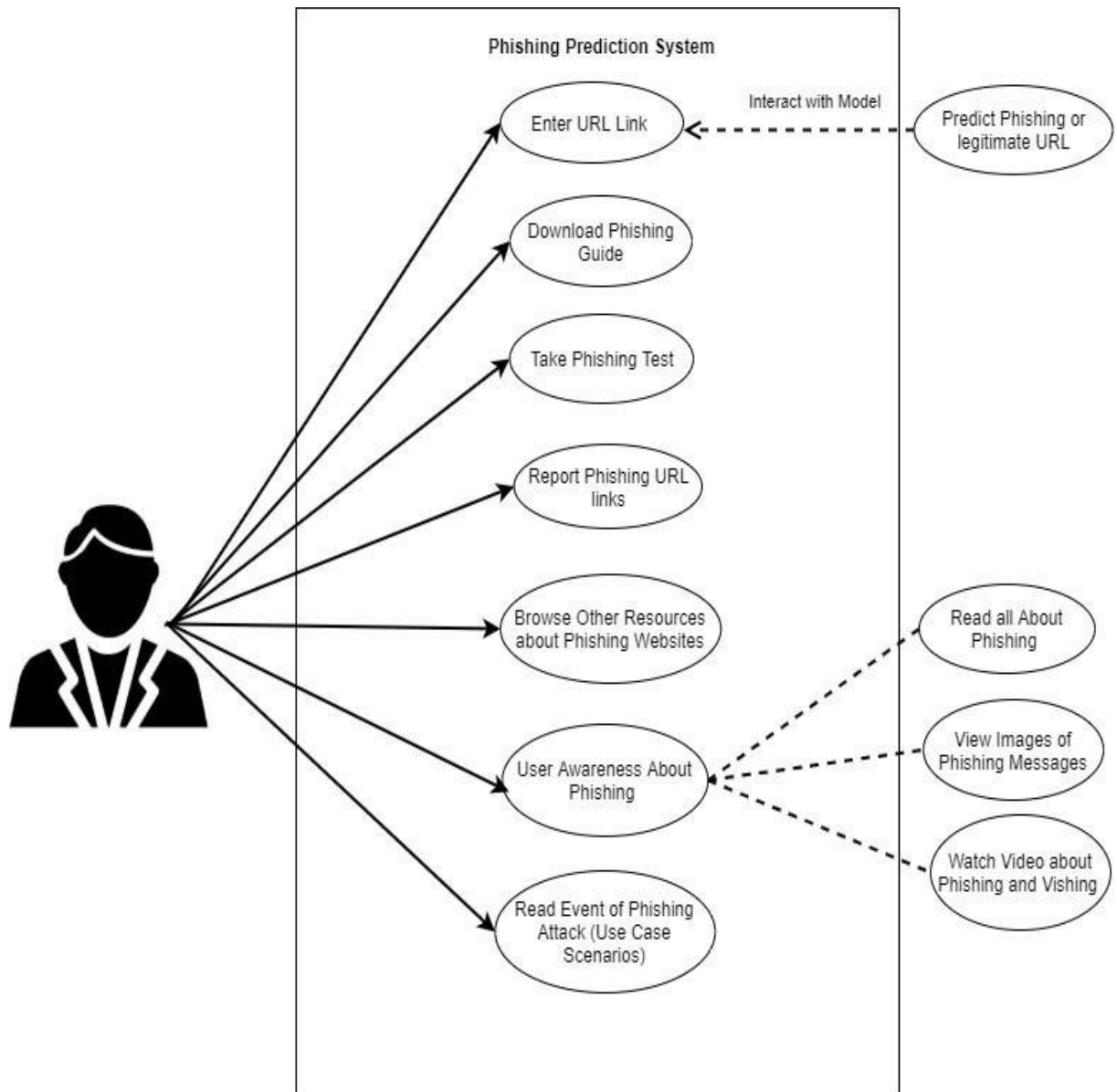


Figure 4.2: Use Case diagram for Proposed System

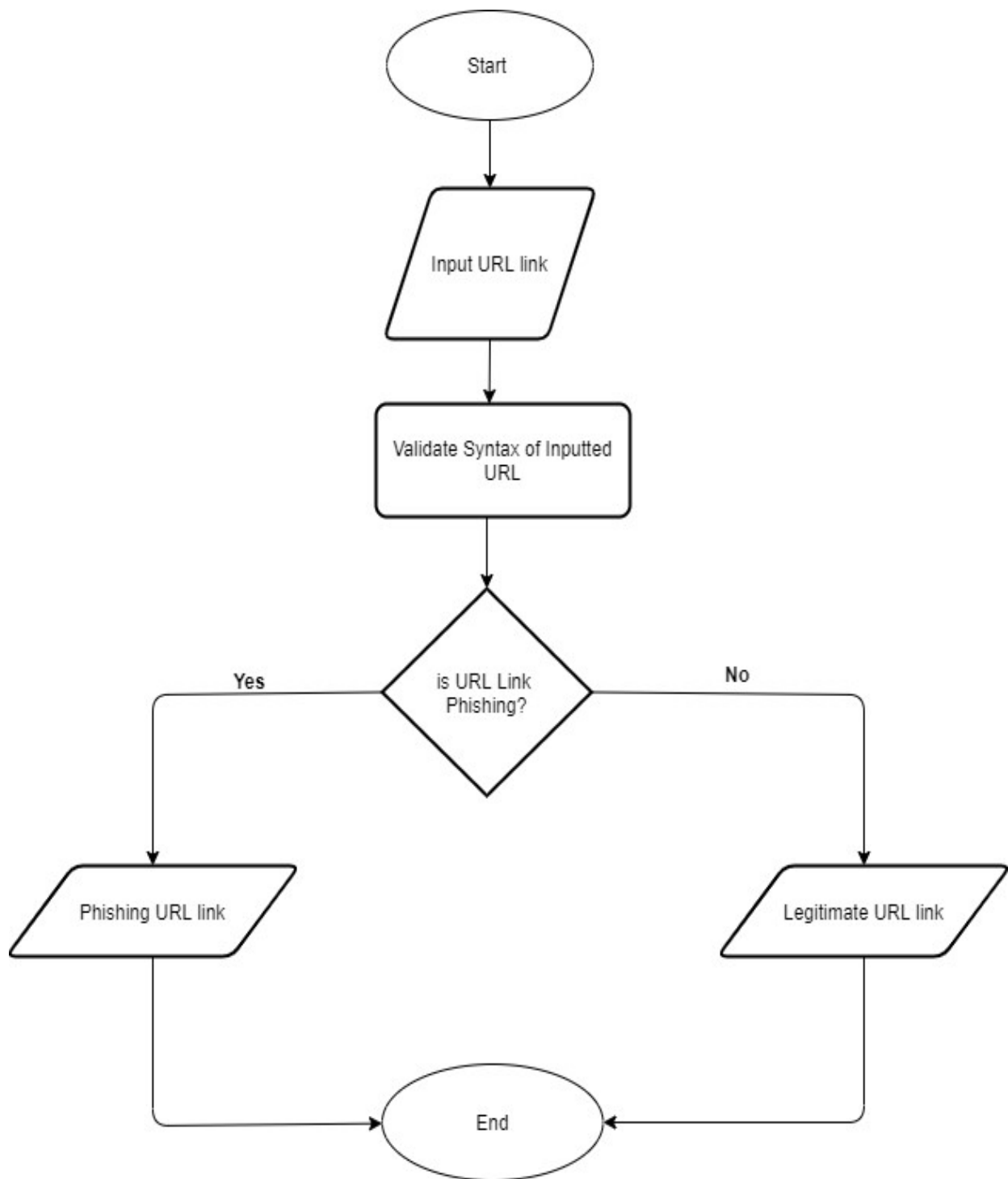


Figure 4.3: Flowchart of the web interface

### **4.3 Hardware requirements**

The hardware requirement includes:

- i. A laptop or desktop computer (Preferably 64bit)
- ii. Random Access Memory (RAM): 8 Gigabytes Minimum
- iii. Processor: Intel Core i5, 2.4 GHz Minimum

### **4.4 Software requirements**

The software requirements for the development of this system include:

- i. Windows Operating System (8/10)
- ii. Anaconda Navigator (Jupyter Notebook)
- iii. PyCharm Community edition
- iv. Web browser (Preferably Chrome)
- v. Visual Studio Code

### **4.5 Prototype 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. To evaluate the model's effectiveness, various evaluation metrics such as accuracy, precision, recall, and F1 score were computed using the testing set. The model's ability to accurately detect phishing websites was assessed, and efforts were made to strike a balance between precision and recall to minimize false positives and false negatives. Throughout the development process, extensive cross-validation techniques were employed to validate the model's performance and mitigate overfitting. The model's robustness and generalizability were tested using unseen data to ensure its reliability.

## **CHAPTER 5**

### **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.

#### **5.1 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:

##### **5.1.1 Hardware requirements**

The hardware requirement includes:

- i. A laptop or desktop computer (Preferably 64bit)
- ii. Random Access Memory (RAM): 8 Gigabytes Minimum
- iii. Processor: Intel Core i5, 2.4 GHz Minimum

##### **5.1.2 Software requirements**

The software requirements for the development of this system include:

- i. Windows Operating System (8/10)
- ii. Anaconda Navigator (Jupyter Notebook)
- iii. PyCharm Community edition
- iv. Web browser (Preferably Chrome)
- v. Visual Studio Code

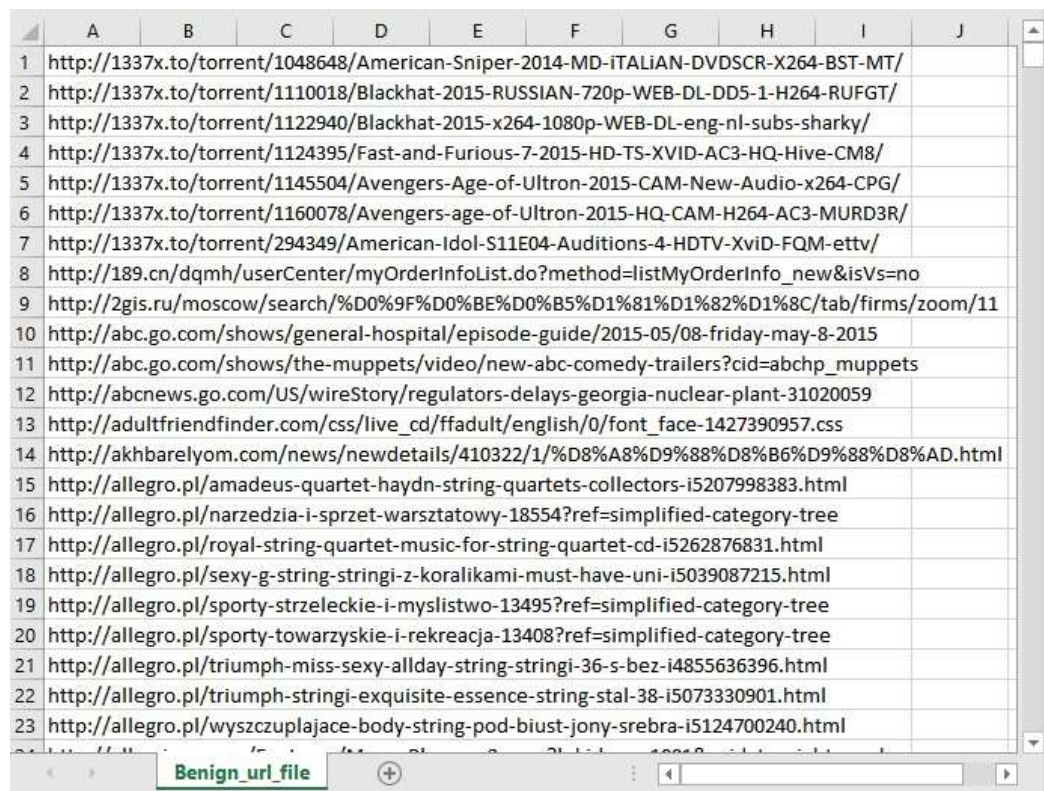
## 5.2 Model Development

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.

### 5.2.1 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 5.1 and 5.2 respectively.



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

Figure 5.1 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://tr	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://w	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
22	6911388	https://w	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 5.2 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

### 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
from urllib.parse import urlparse,urllencode
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  $\geq 54$ , 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 5.3: Code for Address bar-based feature extraction

### 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
```

```
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.alex.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
```

Figure 5.4: Code for domain-based features extraction

### 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
```

#### 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
```

Figure 5.5: 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 5.6: Code computation for all the feature extraction used dataset.

### 5.2.2 Data Analysis and Visualization

The image as shown in figure 5.7 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 5.8 shows the plot of a correlation heat-map of the dataset. The plot shows correlation between different variables in the dataset.

In figure 5.9 and figure 5.10, it shows the feature importance in the model for Decision tree classifier and Random Forest classifier respectively.

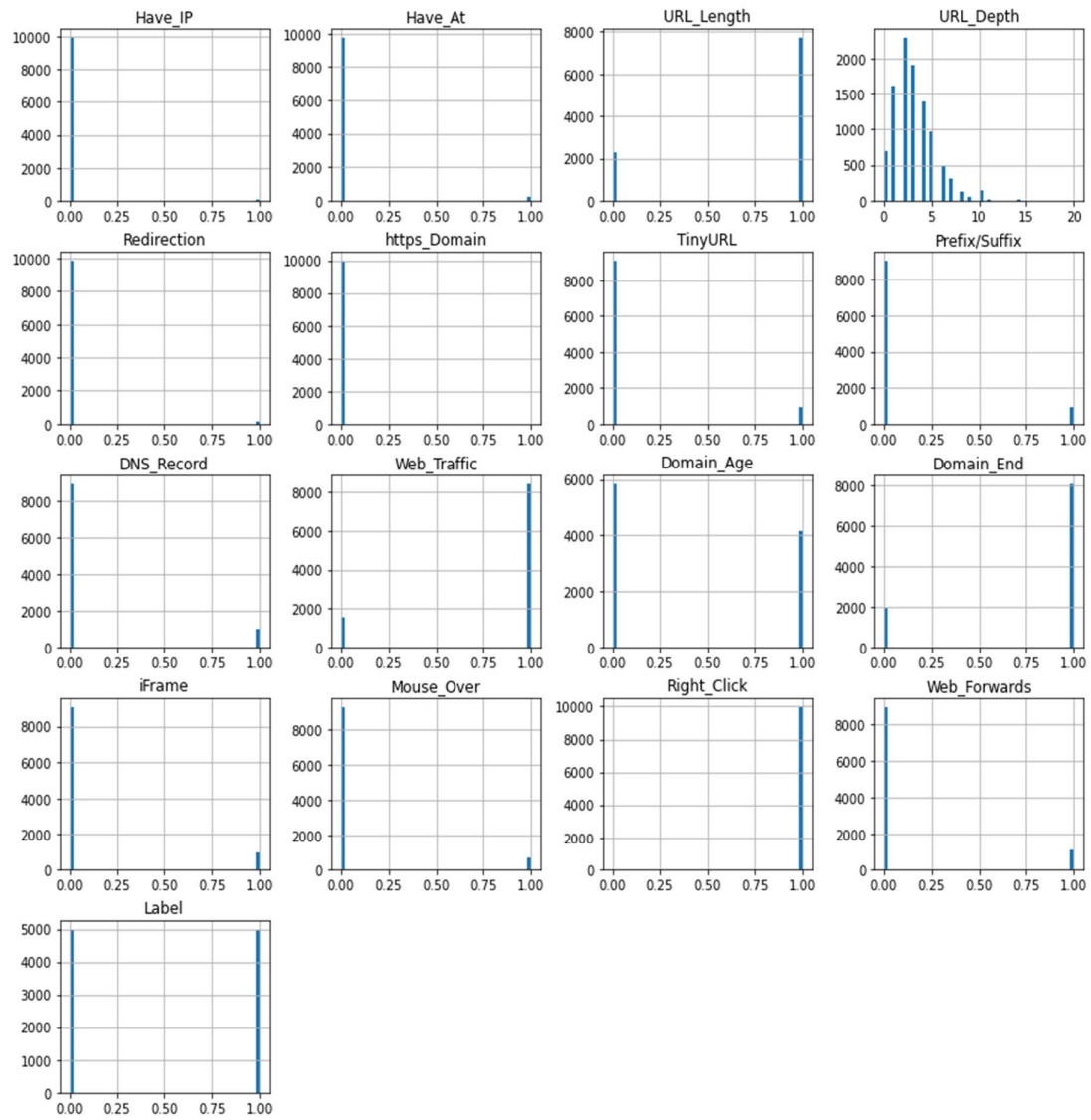


Figure 5.7: Distribution plot of dataset base on the features selected

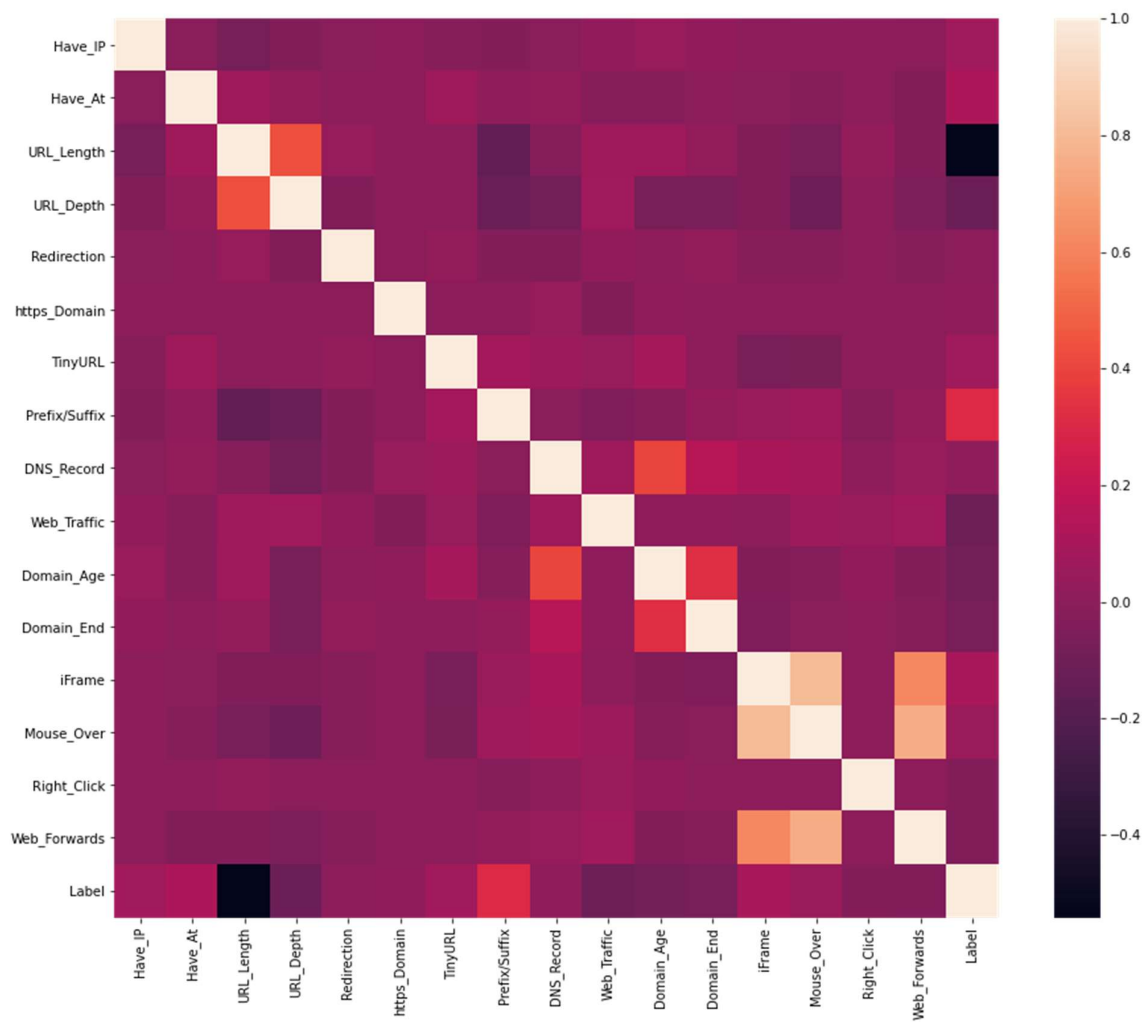


Figure 5.8: Correlation heat map of the dataset

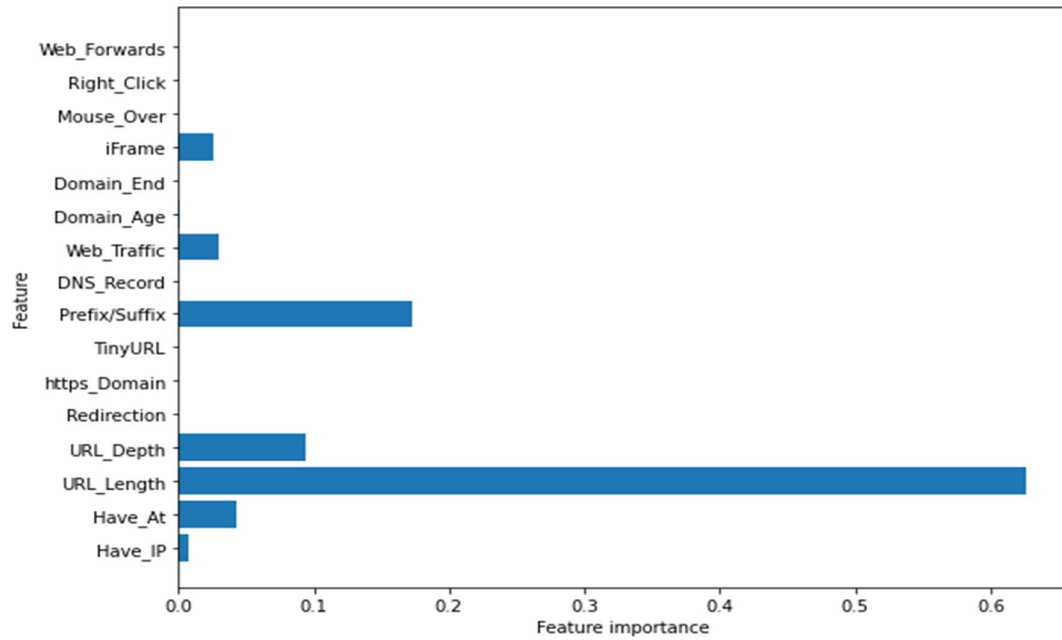


Figure 5.9: Feature importance for Decision Tree classifier

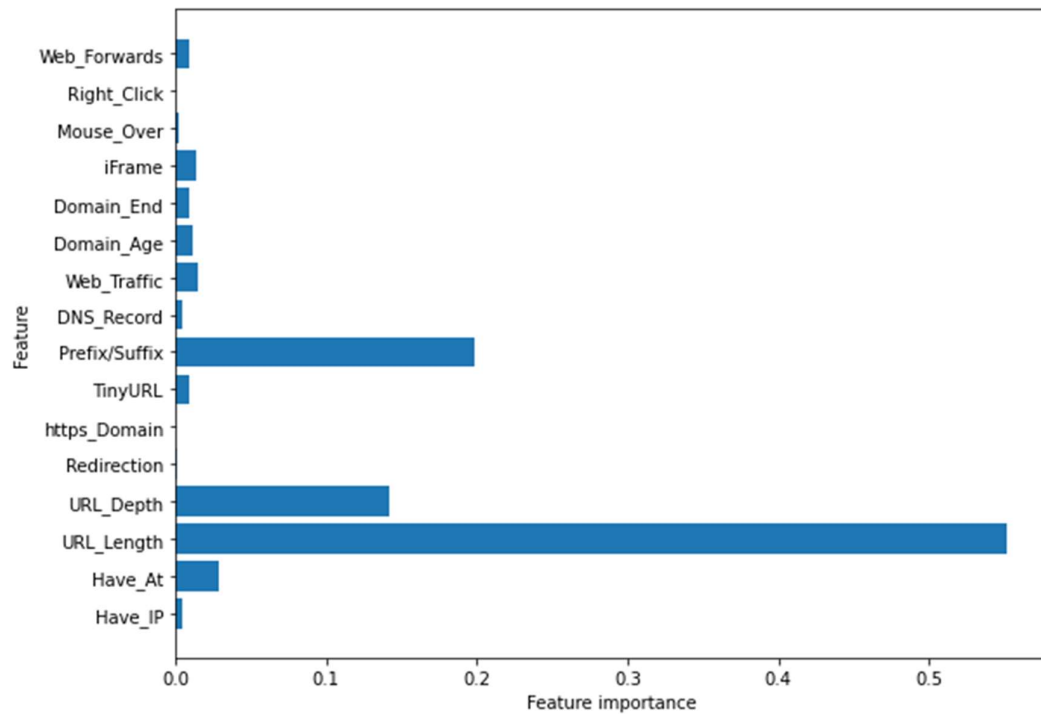


Figure 5.10: Feature importance for Random Forest classifier



### 5.2.3 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 5.11 shows the summary of the dataset while figure 5.12 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 5.11: 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 5.12: Number of missing values in the dataset

### 5.2.3 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 Machine, XGBooster, Multilayer Perceptions, Auto Encoder Neural Network, 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 5.13. 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 5.13: Accuracy performance of models

### **5.3 General Working of The System**

The A one-page phishing detection web application called “Phish-Buster” has been developed to run on any browser. The application was developed using programming languages such as HTML, CSS, PHP, and JavaScript.

The phishing detection web application has the following pages:

#### **5.3.1 Phishing detection model**

The home page contains a session for a user to enter a URL and predict if it is phishing or legitimate. It predicts the state of the URL base on the feature selection. The purpose of this page is to help its users validate a URL link and also provide various resources on phishing attacks. The User can also take a google phishing test to help understand how to detect phishing messages and URLs. Also, users can download a book that contains information and other resources on phishing.

#### **5.3.2 The about page**

The about page contains details about the application and also information about the author of the project.

#### **5.3.3 The use case page**

The use case page contains different case scenarios of phishing attacks that happened to various companies in previous years back. Also, it contains images of phishing attacks messages sent by phishers.

#### **5.3.4 Resource page**

The resource page contains different resources regarding phishing such as definition, types, and techniques of a phishing attack as well as reference links to the source in which the content where retrieved. Also, it contains two (2) sub-session link: the first session reports phishing case and the second session consists of a phishing website.

The First Session: The Report phishing Case link redirects users to submit any case of phishing attack be it URL or a phishing email to Google Support and Google safe browsing.

The Second Session: Phishing website link redirects users to two (2) websites that contain resourceful information and phishing test.

### 5.3.3 The use case page

The use case page contains different case scenarios of phishing attacks that happened to various companies in previous years back. Also, it contains images of phishing attacks messages sent by phishers.

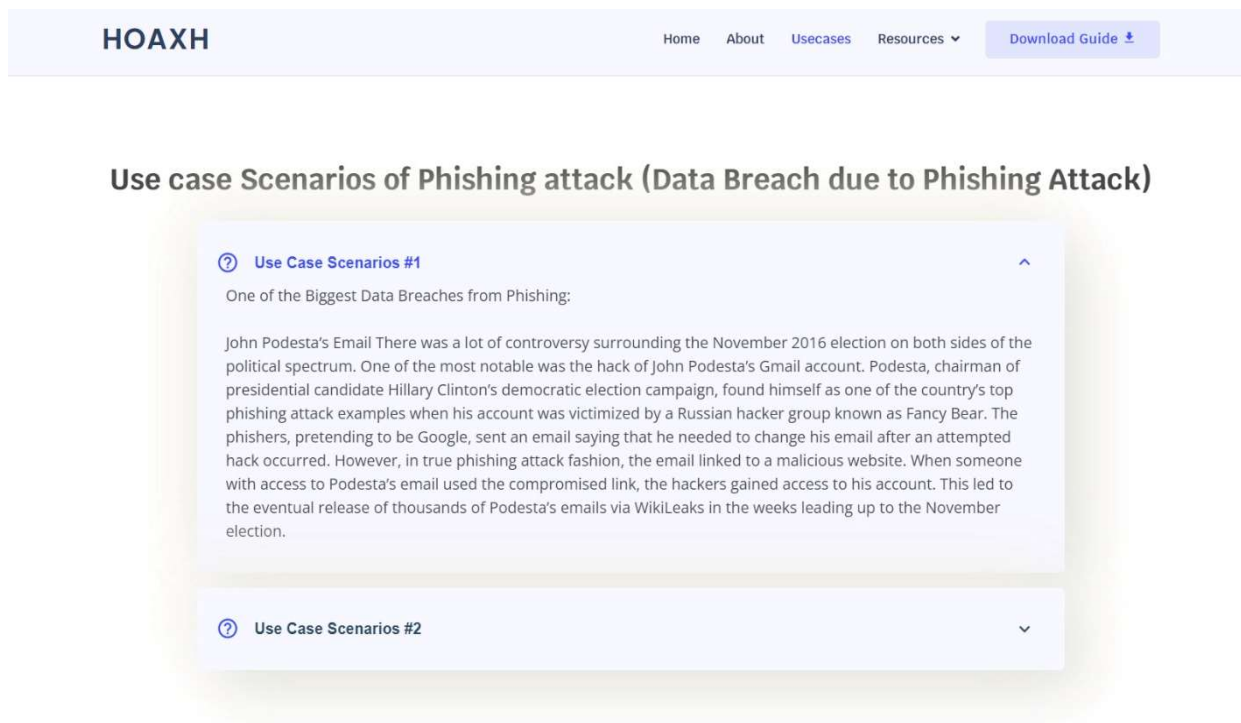


Figure 5.14: Use case diagram of website

## 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. Through comprehensive research and analysis, various techniques and algorithms have been explored and implemented to identify and classify phishing URLs accurately. The findings highlight the significance of combining machine learning approaches with feature engineering to improve the effectiveness of phishing detection systems. With the continued advancements in technology, it is crucial to stay vigilant and update security measures to safeguard users from falling victim to phishing scams. This project serves as a foundation for further research and development in the field of cybersecurity, aiming to create a safer online environment for all users.

## BIBLIOGRAPHY

- Machine Learning Retrieved from  
[https://www.researchgate.net/publication/337049054\\_Phishing\\_Websites\\_Detection\\_using\\_Machine\\_Learning](https://www.researchgate.net/publication/337049054_Phishing_Websites_Detection_using_Machine_Learning).
- Phishing Technique Retrieved From  
<https://www.phishing.org/phishing-techniques>
- Fighting against phishing attacks: state of the art and future challenges, Neural Computing and Applications. Internet world stats usage and population statistics.(2014). Retrieved from  
<http://www.internetworldstats.com/stats.html>
- Machine Learning Retrieved from  
<https://www.researchgate.net/publication/337049054>
- Phishing Websites Detection using Machine Learning
- Various other research papers from ResearchGate and IEEE
- Websites: Google, TowardsDataScience, AnalyticsVidya, ChatGPT
- 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-withexamples-6022d9ca0779>
- Workflow of a Machine Learning project (2019). Retrieved from  
<https://towardsdatascience.com/workflow-of-a-machine-learning-projectecd419b94>