

A Project Report on

Heuristic-based Approach for Phishing Site Detection

Submitted in partial fulfillment of the requirements for the award
of the degree of

Bachelor of Engineering

in

Information Technology

by

Chirag L. Chaudhari(15204030)
Swapnil S. Ghawali(15204009)
Swapnil R. Kshetre(15204003)
Akash A. Sane(15204012)

Under the Guidance of

Guide:- Prof. Sadanand L. Shelgaonkar
Co-Guide:- Prof. Sunil A. Sushir



Information Technology

A.P. Shah Institute of Technology
G.B.Road,Kasarvadavli, Thane(W), Thane-400 615
UNIVERSITY OF MUMBAI

Academic Year 2018-2019

Approval Sheet

This Project Report entitled “*Heuristic-based Approach for Phishing Site Detection*” Submitted by “*Chirag L. Chaudhari*”(15204030), “*Swapnil S. Ghawali*”(15204009), “*Swapnil R. kshtre*”(15204003), “*Akash A. Sane*”(15204012) is approved for the partial fulfillment of the requirement for the award of the degree of *Bachelor of Engineering* in *Information Technology* from *University of Mumbai*.

(Prof. Sunil A. Sushir)
Co-Guide

(Prof. Sadanand L. Shelgaonkar)
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Place :-A.P.Shah Institute of Technology, Thane

Date :-

CERTIFICATE

This is to certify that the project entitled “**Heuristic-based Approach for Phishing Site Detection** ” submitted by “**Chirag L. Chaudhari**” (15204030), “**Swapnil S. Ghawali**” (15204009), “**Swapnil R. Kshetre**” (15204003), “**Aakash A. Sane**” (15204012) for the partial fulfillment of the requirement for award of a degree **Bachelor of Engineering** in **Information Technology**, to the University of Mumbai, is a bonafide work carried out during academic year 2018-2019.

(Prof. Sunil A. Sushir)
Co-Guide

(Prof. Sadanand L. Shelgaonkar)
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Dr. Uttam D. Kolkar
Principal

External Examiner(Signature)

1.

2.

Place :- A.P.Shah Institute of Technology, Thane

Date :-

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)
(Chirag L. Chaudhari(15204030))

(Signature)
(Swapnil S. Ghawali(15204009))

(Signature)
(Swapnil R. Kshetre(15204003))

(Signature)
(Aakash A. Sane(15204012))

Date :-

Acknowledgement

We have great pleasure in presenting the report on **Heuristic-based Approach for Phishing Site Detection**. We take this opportunity to express our sincere thanks towards our guide **Prof. Sadanand L. Shelgaonkar** & Co-Guide **Prof. Sunil A. Sushir** Department of IT, APSIT thane for providing the technical guidelines and suggestions regarding line of work. We would like to express our gratitude towards his constant encouragement, support and guidance through the development of project.

We thank **Prof. Kiran B. Deshpande** Head of Department, IT, APSIT for his encouragement during progress meeting and providing guidelines to write this report.

We thank **Prof. Vishal S. Badgujar** BE project co-coordinator, Department of IT, APSIT for being encouraging throughout the course and for guidance.

We also thank the entire staff of APSIT for their invaluable help rendered during the course of this work. We wish to express our deep gratitude towards all our colleagues of APSIT for their encouragement.

Name :- CHIRAG LAHU CHAUDHARI
Id :- 15204030

Name :- SWAPNIL SHIVAJI GHAWALI
ID :- 15204009

Name :- SWAPNIL RAOSAHEB KSHETRE
ID :- 15204003

Name :- AAKASH ANIL SANE
ID :- 15204012

Abstract

Phishing is one of the most potentially disruptive actions that can be performed on the Internet. Intellectual property and other pertinent business information could potentially be at risk if a user falls for a phishing attack. The most common way of carrying out a phishing attack is through email. The adversary sends an email with a link to a fraudulent site to lure consumers into divulging their confidential information. While such attacks may be easily identifiable for those well-versed in technology, it may be difficult for the typical Internet user to spot a fraudulent email. The emphasis of this research is to detect phishing attempts within emails. To date, various phishing detection algorithms, mostly based on the blacklists, have been reported to produce promising results. Yet, the phishing crime rates are not likely to decline as the cyber-criminals devise new tricks to avoid those phishing filters. Since the early non-text based approaches do not address the text content of the email that actually deludes users, this paper proposes a text-based phishing detection algorithm. In particular, this research focuses on improving upon the previously published text-based approach. The algorithm in the previous work analyzes the body text in an email to detect whether the email message asks the user to do some action such as clicking on the link that directs the user to a fraudulent website. This work expanded the text analysis portion of that algorithm, which performed poorly in catching phishing emails. The modified algorithm generated considerably higher results in filtering out malicious emails than the original algorithm did; but the rate of text incorrectly identified as phishing, which is the FPR, was slightly worse. To address the FP problem, a statistical approach was adopted and the method ameliorated the FPR while minimizing the decrease in the phishing detection accuracy.

Contents

1	Introduction	1
1.1	Library system :	1
1.2	Related Work:	3
1.3	Objective:	3
1.4	Scope:	3
2	Literature Review	5
2.1	The secured Anti-phishing Approach Using Image based Validation.Y.Yesu Jyothi, D. Srinivas k. GovindaRaju,2013	5
2.2	Protecting users Against phishing attacks, En-gin kirda Chistopher Kruegel,2012	5
2.3	phishing Anti -Phishing Technique(Prakash etal., 2010)	5
2.4	RDF based anti phishing framework(Vamsee etal.,2013)	6
3	Proposed System/ Problem Statement	8
3.1	URL Structure:	12
3.2	URL Features:	12
3.3	Architecture	13
3.4	Algorithm	14
4	Diagram	15
4.1	Activity Diagram	15
4.2	Use-Case Diagram	16
4.3	Sequence Diagram	17
4.4	Class Diagram	18
4.5	Data Flow Diagram	18
5	Implementation	19
5.1	Result	27
5.2	REQUIREMENTS ANALYSIS(Hardware And Software Requirements) . . .	28
5.2.1	Hardware Description	28
5.2.2	Software Description	28
6	Testing Implementation	29
6.1	SOFTWARES TO BE USED FOR THE PROJECT	29
6.2	STEPS INVOLVED IN THE SYSTEM DEVELOPMENT LIFE CYCLE . .	31
6.2.1	System Development Life Cycle:	33
7	Methodology	36

8	Conclusions and Future Scope	47
8.1	Conclusions	47
8.2	Future scope:	47
	Bibliography	48
	Publication	52

List of Figures

1.1	Phishing Attacks per Year	2
3.1	Intrusion Detection System	8
3.2	Architecture	13
4.1	Activity Diagram	15
4.2	Use-Case Diagram	16
4.3	Sequence Diagram	17
4.4	Class Diagram	18
4.5	Data Flow Diagram	18
5.1	Main Screen	19
5.2	Training system upload database	20
5.3	Extract the features using ID3	20
5.4	View of Extracted features	21
5.5	View of uploaded database	21
5.6	Generate Rules	22
5.7	Check URL	22
5.8	Test URL	23
5.9	Search bar	23
5.10	Legitimate site	24
5.11	Open in Browser output	24
5.12	Phishing Site	25
5.13	Test with graph	25
5.14	Upload Test database	26
5.15	Show the Calculation	26
6.1	Waterfall Model	33
6.2	System Development Life Cycle (SDLC)	34
8.1	Detect Phishing Site	51

List of Tables

2.1	Literature Review	7
5.1	TP,TN,FP,FN Matrix	27

List of Abbreviations

URL:	Uniform Resource Locator
HTML:	Hyper text markup language
SMS:	Short Message Service
DOM:	Document Object Model
SVM:	Support Vector Machine
IP:	Internet Protocol
RDF:	Resource Description Framework
HTTP:	Hyper Text Transfer Protocol
HTTPS:	Hyper Text Transfer Protocol over secure socket layer
FTP:	File Transfer Protocol
SEO:	Search Engine Optimization
WOT:	Web Of Trust
ODP:	Open Directory Project
WWW:	World Wide Web
DNS:	Domain Name System
ID3:	Iterative Dichotomiser 3
WEKA:	Waikato Environment For Knowledge Analysis
TP:	True Positive
TN:	True Negative
FP:	False Positive
FN:	False Negative
ASP:	Active Server Page
SQL:	Sequential Query Language
RDBMS:	Relational Database Management Sysytem
XML:	Extensible Markup Language
ETL:	Extract Transform Load
TSDLC:	Software Development Life Cycle

Chapter 1

Introduction

1.1 Library system :

With the recent growth of the Internet environment and diversification of available web services, web attacks have increased in quantity and advanced in quality. Phishing is a type of social engineering attack that targets a users sensitive information through a phony website that appears similar to a legitimate site, or by sending a phishing email[1]. Phishing is a website forgery technique with an intention to track and steal the sensitive information of online users. The hacker fools the user with social engineering techniques such as SMS, voice,email, website and malware[2].Various approaches have been proposed and implemented to detect a variety of phishing attacks such as use of blacklists and white lists to name a few. We propose a desktop application called PhishDetect, which focuses on URL and website content of the phishing web-page. We aim at detecting phishing websites with the help of a desktop application named PhishDetect. PhishDetect use a heuristic features to detect a number of phishing attacks[2]. We will create a Application which name is PhishDetect. PhishDetect is a desktop application to eectively detect phishing websites and practices. PhishDetect is capable to detect most of the phishing techniques and aims at providing full-proof security from all kinds of phishing attacks.PhishDetect is not a traditional Anti-virus software and does not guarantee any protection from any kind of virus attacks. PhishDetect is based on the idea that users should be able to browse the internet safely and access websites without getting concerned about the legitimacy of the websites. The system works by taking an input from the user in the form of URL of the website for which legitimacy needs to be determined. The system then outputs the state of the website as phishing, legitimate or unknown[6]. The existing systems have several limitations which can be overcome by PhishDetect. The main advantages of PhishDetect over normal phishing detectors are as following:

- It is based on heuristic approach that is capable of detecting Zero hour phishing attacks that is phishing attacks that are relatively new which is not possible for most of the other phishing detectors.
- Users just need to provide the URL of the website whose legitimacy needs to be determined. Nothing else needs to be done by the user.
- The main advantage of our application is that it can detect phishing sites which tricks the users by replacing content with images, which most of the existing anti phishing

techniques are not able to detect, even if they can, they take more execution time than our application.

Heuristic based methods extract features of a web-page to decide the legitimacy of the website in-stead of depending on any pre-compiled lists. Most of these features are extracted from URL and HTML Document Object Model (DOM) of the given web-page. The extracted features are compared with known features collected from phishing and legitimate pages to decide its legitimacy. Some of these approaches use heuristics to calculate spoof score of a given web-page to check its genuineness[6].

Machine learning approach :- The machine learning approach exploits many characteristics of the URL and the websites by using machine learning techniques such as Support Vector Machines (SVM), Decision tree algorithms, Random forest classification method etc. These characteristics are combined to use to detect the phishing websites. However, there are some limitations in this approach. First, the machine learning-based techniques might fail in the case that attackers compromise legitimate domains and host phishing attacks on those servers. Second, because of text-based analysis mechanism, these phishing detection techniques cannot detect the phishing websites which are purely made up of images[7].

Heuristics based Phishing Detection :- Heuristic based methods extract features of a web page to decide the legitimacy of the website in-stead of depending on any precompiled lists. Most of these features are extracted from URL and HTML Document Object Model (DOM) of the given web page. The extracted features are compared with known features collected from phishing and legitimate pages to decide its legitimacy. Some of these approaches use heuristics to calculate spoof score of a given web page to check its genuineness[7]



Figure 1.1: Phishing Attacks per Year

1.2 Related Work:

Phishing is an attempt to steal a users personal information typically through a fraudulent email or website [1]. We conducted a study on phishing sites, which are either fake sites that are designed to appear similar to legitimate sites or sites that simply have phishing-related behaviors. Almost all phishing sites include the functionality in which users enter sensitive information, such as their personal identification, password, and/or account number. These sites can include links to connect to other phishing sites and malicious code that contaminates a users computer. Phishing detection techniques can be generally divided into blacklist-based and heuristic-based approaches. The blacklist-based approach maintains a database list of addresses (URLs) of sites that are classified as malicious. If a user requests a site that is included in this list, the connection is blocked [3]. The blacklist-based approach has the advantages of easy implementation and a low false positive rate; however, it cannot detect phishing sites that are not listed in the database, including temporarily sites [4]. The heuristic based approach analyzes phishing site features and generates a classifier using those features [5]. When a user requests a web page, the classifier determines whether that page is a phishing site. This approach can detect new phishing sites and temporary phishing sites because it extracts features from the requested web page. Nevertheless, it has the disadvantage of being difficult to implement; moreover, generating a classifier is time intensive. Thus, the two approaches have both advantages and disadvantages. Therefore, these approaches are selectively employed in the proposed technique depending on the application.

1.3 Objective:

In this project URL heuristic approach will be used along with the ranking of sites to extract the features from the URL. All the extracted features along with the phishing and legitimate sites URLs will be stored in database. Next, a classifier will be generated using decision tree algorithm which will classify the URLs as phishing and legitimate. When new URL is receive edit will extract the features and will compare them with the features stored in database, thus classifying the incoming site as phishing or legitimate.

1.4 Scope:

We intend to address the time-intensive disadvantage of the heuristic-based technique. With a large number of features, it is time-consuming for the heuristic based approach to generate classifiers and perform classification. Therefore, we will apply algorithms to reduce the number of features and thereby improve performance. In addition, we will examine a new phishing detection technique that uses not only URL-based features, but also HTML and JavaScript features of web pages to improve performance.

1. Economic Feasibility :- This system can be used by the E-commerce enterprise in order to carryout the whole transaction process securely. This system will increase the productivity and protability of the E-commerce enterprise. This will provide economic benets. It includes quantication and identification of all the benets expected.
2. Operational Feasibility :- This system is more reliable, maintainable, aordable and producible. These are the parameters which are considered during design and develop-

ment of this project. During design and development phase of this project there was appropriate and timely application of engineering and management eorts to meet the previously mentioned parameters.

3. Technical Feasibility :- The back end of this project is SQL server which stores details which is related to this project. There are basic requirement of hardware to run this application. This system is developed in .Net Framework using C. This application will be online so this application can be accessed by using any device like (Personal Computers, laptop)

Chapter 2

Literature Review

2.1 The secured Anti-phishing Approach Using Image based Validation.Y.Yesu Jyothi, D. Srinivas k. GovindaRaju,2013

- Problem Identified: To solve the problem of phishing protect individual personal private information
- Methodology: Visual Cryptography(image based validation)
- Strength: It prevents attack of phishing websites on financial portal, banking portal online shopping market.
- Weakness: Inability to recover missing or corrupt share

2.2 Protecting users Against phishing attacks, Engin kirda Chistopher Kruegel,2012

- Problem Identified: Increased email linked to phishing scams
- Methodology: Browser Extension
- Strength: It protect users against spoofed website-based phishing attacks
- Weakness: It requires that user support to capture store sensitive information rather than automatically capturing storing the sensitive information

2.3 phishing Anti -Phishing Technique(Prakash etal., 2010)

- Problem Identified: predicts variation of URLs
- Methodology: heuristics

- Strength: It replace Top level domain(TLD), Directory structure similarity, IP address equivalence, Query staring substitution brands name equivalence
- Weakness: It cannot detect zero day phishing

2.4 RDF based anti phishing framework(Vamsee etal.,2013)

- Problem Identified: To differentiate a phishing site from a legitimate site
- Methodology:Resource Description framework model
- Strength: It uses nineteen properties that describes the characteristics of a web page to distinguish between a legitimate a phishing site
- Weakness: Building RDF can be cuber some

Title of Paper	Problem Identified	Methodology	Strength	Weakness
The secured Anti-phishing Approach Using Image based Validation.Y.Yesu Jyothi, D. Srinivas k. GovindaRaju,2013	To solve the problem of phishing protect individual personal private information	Visual Cryptography(image based validation)	It prevents attack of phishing websites on nancial portal, banking portal on-line shopping market.	Inability to recover missing or corrupt share.
Protecting users Against phishing attacks, Engin kirda Chistopher Kruegel,2012	Increased email linked to phishing scams	Browser Extension	It protect users against spoofed website-based phishing attacks.	It requires that user support to capture store sensitive information rather than automatically capturing storing the sensitive information.
phishing Anti-Phishing Technique(Prakash etal., 2010)	Predicts variation of URLs	heuristics	It replace Top level domain(TLD), Directory structure similarity, IP address equivalence, Query staring substitution brands name equivalence.	It cannot detect zero day phishing.
RDF based anti phishing framework(Vamsee etal.,2013)	To diifferentiate a phishing site from a legitimate site	Resource Description framework model	It uses nineteen properties that describes the characteristics of a web page to distinguish between a legitimate a phishing site.	Building RDF can be cuber some.

Table 2.1: Literature Review

Chapter 3

Proposed System/ Problem Statement

The proposed architecture for System is given in Figure. The system provides an interface where the user can write his/her query. Once the search button is clicked, it gives the list of URLs on the same page . A URL is a protocol that is used to indicate the location of data on a network. The URL is composed of the protocol, sub-domain, primary domain, top-level domain (TLD), and path domain. In meantime; it saves all the URLs in the database. The protocol refers to a communication protocol for exchanging information between information devices; e.g., HTTP, FTP, HTTPS, etc. Protocols are of various types and are used in accordance with the desired communication method. For each URL, all mentioned eighteen factors are calculated and saved as total score in the database. Then based on the total score value of each URL, they are rearranged in the descending order, which means if URL has high total score value then it will appear as the top most result and accordingly rest comes as per their total score value.

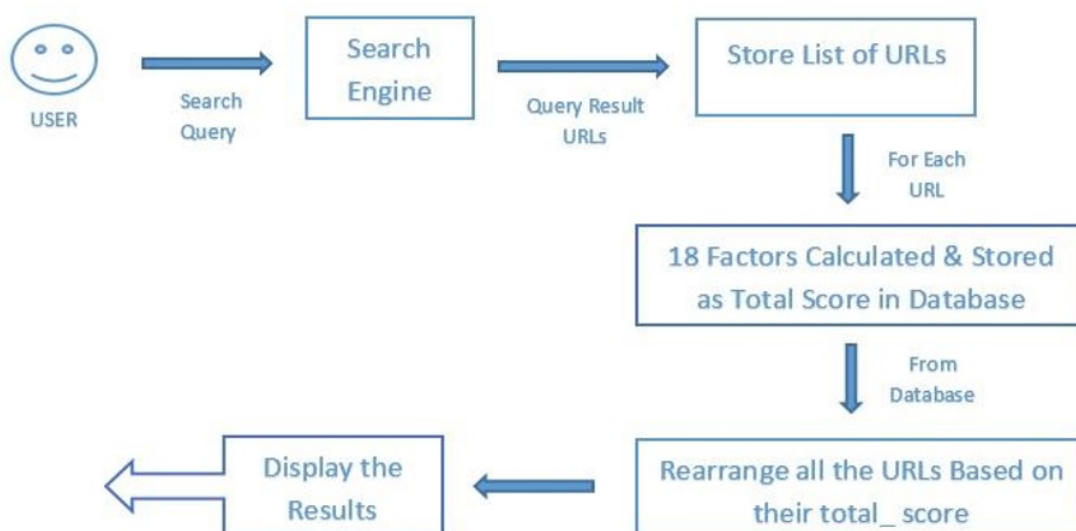


Figure 3.1: Intrusion Detection System

ANALYSIS OF EIGHTEEN IMPACT FACTORS:

Once user entered his/her query, list of URLs will be available on the same screen. Then each URL is extracted to retrieve the eighteen factors value for calculating them. Then it stores all the calculated factors in the database. Next and final step is that it shows the new result (URLs) in descending order of their Total score. That means if record has high score then it will come to the top and rest comes as per the total score value. Calculations of all 18 factors are as follows:

1. Page Title: Every HTML document must have a TITLE Element in the head section. From a search engine's point of view, page title is the first indication of the contents of the page. Additionally, page title is the key information returned when search engines list results to a keyword search. If the searched keyword is in the title tag, then it means the page is most matched.

2. Meta Keyword: A Meta keywords tag is supposed to be a brief and concise list of the most important themes of the web-page. Long ago in Internet time, the Meta keywords tag was very useful in helping pages to win on search engines. But many unscrupulous web-masters have abused the Meta keywords tag that search engines have had to DE-emphasize their importance. Though Meta keywords tags are not a major factor search engines consider when ranking sites, they should not be left off the page. If the searched keyword is in the Meta tag then it means the page is most matched.

3. Meta Description: The Meta description tag is intended to be a brief and concise summary of web page's content. The Meta description tag is designed to provide a brief description of the website which can be used by search engines or directories. If the searched keyword is in the Meta description then it means the page is most matched.

4. Alexa Rank: Alexa is a very powerful tool used to rank web site traffic. This is one of the most accurate freely available tools to find out how well your site ranks up against millions of other sites on the web. The lower the Alexa ranking number the more heavily visited the site.

5. Google Page Rank: Google Page Rank is one of the methods Google uses to determine relevance or importance of a page. It matters because it is one of the factors that determine a page's ranking in the search results. If the page rank of a particular page is closer to 5 then it is considered as more popular.

6. Google Inbound Links: Defines number of third party websites link to particular website that is identified by Google. Back links are important for SEO because some search engines, especially Google, will give more credit to websites that have a good number of quality back links, and consider those websites more relevant than others in their results pages for a search query. More number of Google in bounds links then more Weight-age is given to it.

7. Google Indexed Pages: Google Indexed Pages gives an indication of number of pages indexed by Google and available in Google servers. There are various on-page SEO factors helping to get higher search engine rankings including the number of web pages indexed by Google and other search engines (indexation). It plays an important role in the SEO score

of particular site. In many cases, the winning factor of a site compared to its competitor is the number of its pages indexed by Google or other search engines.

8. Last Modified Date: Last Modified Date is the date of the particular page created or modified by the website owner. From an informational point of view, newer content is usually better than older content within a web page. Whenever user tries to search some information and anticipate that newer information rather than information that is few-years-old .

9. Domain Age: Domain Age defines how old a particular website is. A long-lasting and well-kept domain name on the cyberspace reflects its importance to search engine optimization .

10. Yahoo Inbound Links: Yahoo Inbound Links defines number of third party websites link to particular website that is identified by Yahoo . More number of Yahoo inbound links then more Weight-age is given to it.

11. Web Of Trust (WOT) Rating: WOT ratings are powered by a global community of millions of trustworthy users who have rated millions of websites based on their experiences. Website's reputation rating is based on ratings from the WOT community, tells how much other users trust your site. It defines that particular site is safe or not as Trustworthy, Mostly, Suspicious, Untrustworthy, Dangerous, Unknown and depending upon this Weight-age is assigned to it.

12. Yahoo Indexed Pages: Yahoo Indexed Pages gives an indication of number of pages indexed by yahoo and available in yahoo servers. If more pages are indexed by Yahoo and available in Yahoo servers, more it is better.

13. Alexa Inbound Links: Alexa Inbound Links defines number of third party websites link to particular website that is identified by Alexa. Quality inbound links are an essential element of web site marketing and search engine optimization programs to increase traffic and online sales. The greater the number of relevant and authoritative links to a web page, the greater the potential for higher search engine rankings and qualified traffic. More number of Alexa inbounds links then more Weight-age is given to it.

14. Dmoz Listing: Dmoz Listing is the Open Directory Project (ODP) is a multilingual open content directory of WWW links. That means site comes under the particular category. DMOZ is the most respected online directory. All major search engines like Google, Yahoo, and Bing give a lot of importance to websites having links from DMOZ. Inclusion in DMOZ can dramatically increase your website ranking and traffic. If the site comes under the ODP category that means marks will be 100

15. Site Advisor Rating: Ratings from SiteAdvisor.com are based on a variety of measures. It claims to protect user by labeling Web sites green, yellow, or red to indicate that they are safe, questionable, or dangerous. Site Advisor Rating defines that site is good (Green), bad (Red), compromised (Yellow) and (Grey). Depending upon the color Weight-age is assigned.

16. Bing Indexed Pages: Bing Indexed Pages gives an indication of number of pages indexed by Bing and available in Bing servers. If more pages are indexed by Bing and available in Bing servers, more it is better.

17. Bing Inbound Links: Bing Inbound Links defines number of third-party websites link to particular website that is identified by Bing. More number of Bing inbounds links then more Weight-age is given to it.

18. Ask Indexed Pages: Ask Indexed Pages gives an indication of number of pages indexed by Ask and available in Ask servers. If more pages are indexed by Ask and available in Ask servers, more it is better.

3.1 URL Structure:

A URL (uniform resource locator) is used to locate the resources. The structure of URL is as follows: protocol : // sub-domain . primary-domain . TLD / path-domain For example, with the URL: `http://www.paypal.abc.net/login/web/index.html`, there are six components as follows: Protocol is `http`, Sub-domain is `paypal`, Primarydomain is `abc`, TLD is `net`, Domain is `abc.net`, Path-domain is `login/web/index.html`

3.2 URL Features:

1. Similarity of primary domain and Google suggestion(primary domain) :- Levenshtein distance between primary domain and Google suggestion(primary domain).
2. Similarity of subdomain and Google Suggestion (subdomain) :- Levenshtein distance between primary domain and Google Suggestion (subdomain).
3. Similarity of path domain and Google Suggestion (path domain) :- Levenshtein distance between primary domain and Google Suggestion (path domain).
4. Safety of Google Suggestion (primary domain) :- Whether result of Google Suggestion (primary domain) is present in the whitelist .
5. Safety of Google Suggestion (subdomain) :- Whether result of Google Suggestion (subdomain) is present in the whitelist.
6. Number of TLD and out of TLD position :- More than one TLD in URL, and out of TLD position .
7. Safety of Google Suggestion (path domain) :- Whether result of Google Suggestion (path domain) is present in the whitelist.
8. Google page rank :- PageRank value of domain
9. Via IP address :- Whether domain is in the form of an IP address
10. Length of URL :- Length of URL
11. Suspicious character :- Whether URL has @, //
12. Prefix and suffix :- Whether URL has -
13. Number of subdomain :- Number of dots in domain
14. Length of subdomain :- Length of subdomain
15. Port number matching :- Whether explicit port number and protocol port number are equal
16. Phishing words in URL :- Whether URL has phishing terms

17. Primary domain spelling mistake :-Whether primary domain is similar to whitelisted domains
18. Number of / :- Number of / in URL
19. Country matching :- TLD country, and domains country are equal or not
20. HTTPS protocol :- Whether URL use https
21. DNS record :- Whether URL has DNS record
22. WHOIS record :- Domain age in WHOIS record

3.3 Architecture

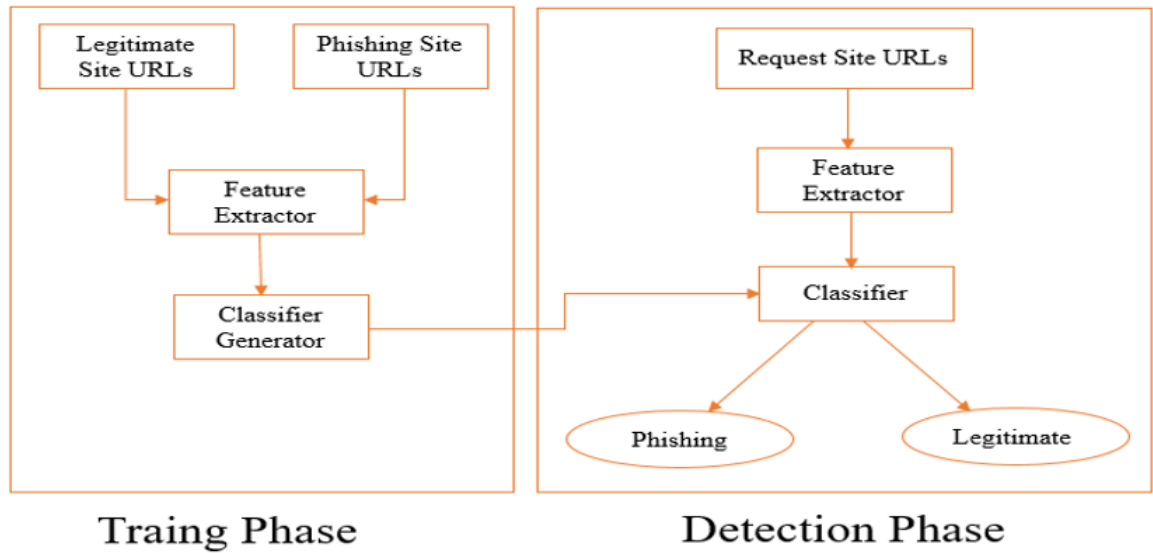


Figure 3.2: Architecture

In the training phase, a classifier is generated using URLs of phishing sites and legitimate sites collected in advance. The collected URLs are transmitted to the feature extractor, which extracts feature values through the predefined URL-based features. The extracted features are stored as input and passed to the classifier generator, which generates a classifier by using the input features and the machine learning algorithm. In the detection phase, the classifier determines whether a requested site is a phishing site. When a page request occurs, the URL of the requested site is transmitted to the feature extractor, which extracts the feature values through the predefined URL-based features. Those feature values are inputted to the classifier. The classifier determines whether a new site is a phishing site based on learned information. It then alerts the page-requesting user about the classification result.

3.4 Algorithm

Decision tree:

Machine learning technique:- Machine learning algorithms are used to build an efficient classifier which would decide whether a given URL is phishing or not. Decision tree is a classification method that was introduced in 1992 by Quinlan[8] . It creates a tree form for classifying samples. Each internal node of the tree corresponds to a feature, and the edges from the node separate the data based on the value of the feature [8]. Decision tree includes a decision area and leaf node. The decision area checks the condition of the samples and separates them into each leaf node or the next decision area. The decision tree is very fast and easy to implement; however, it has the risk of over fitting we propose a new heuristic-based phishing detection technique that resolves the limitation of the blacklist-based technique. Even complex phishing attacks can be easily determined by Decision Tree algorithm. This algorithm has relatively less false positive and false negative rates[8].

We implemented the proposed technique and conducted an experimental performance evaluation. The proposed technique extracts features in URLs of user-requested pages and applies those features to determine whether a requested site is a phishing site. This technique can detect phishing sites that cannot be detected by blacklist-based techniques; therefore, it can help reduce damage caused by phishing attacks.

Chapter 4

Diagram

4.1 Activity Diagram

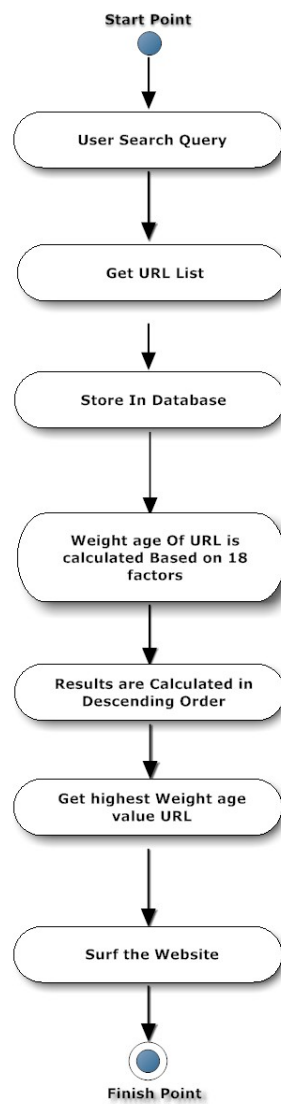


Figure 4.1: Activity Diagram

4.2 Use-Case Diagram

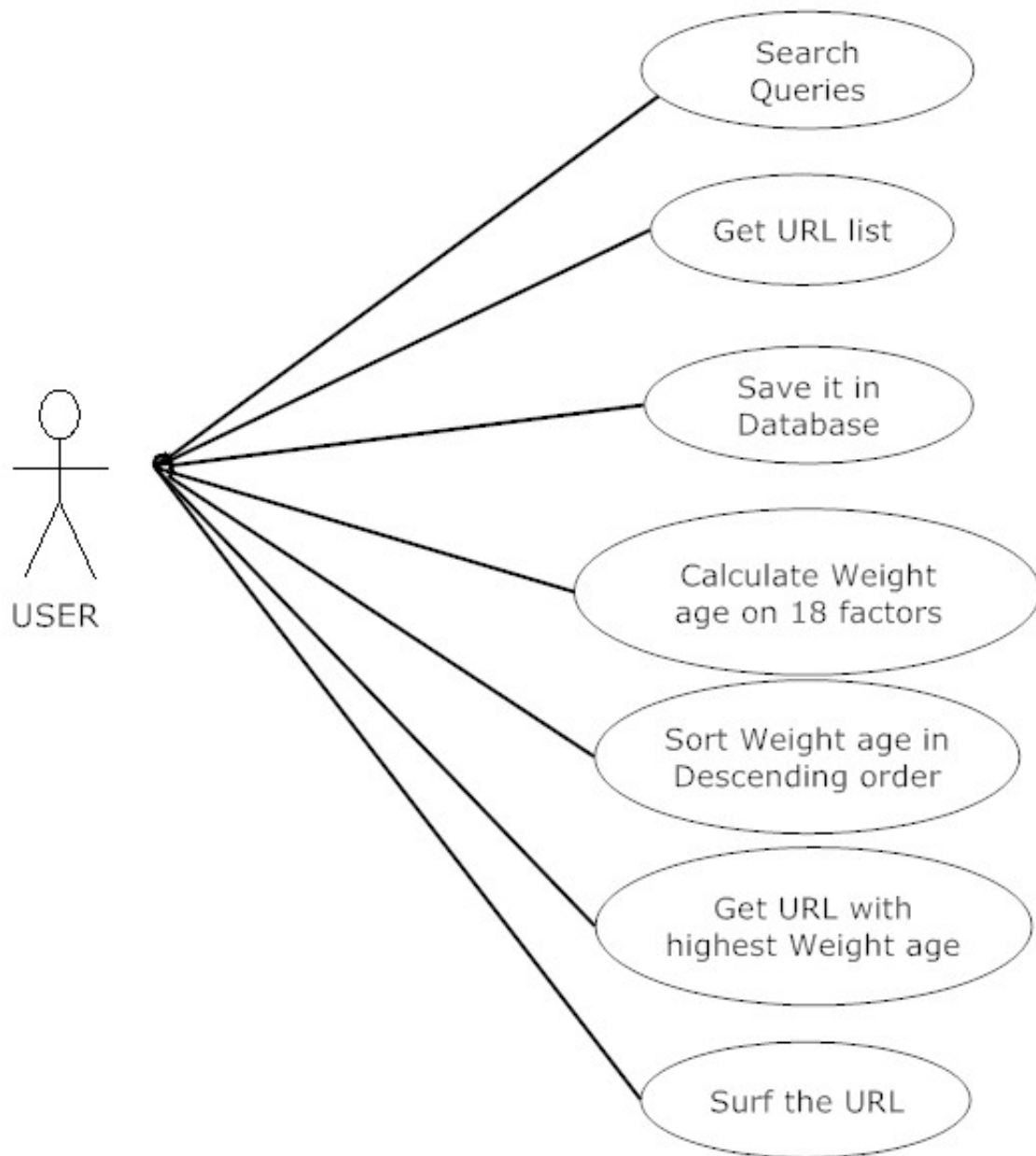


Figure 4.2: Use-Case Diagram

The use case diagram for user is depicted in Figure 4.2. He is responsible for the general maintenance of the site as well as giving out important announcements if and when required. He also manages the addition or removal of blacklisted websites. They can check whether a particular website is a phishing website or not.

4.3 Sequence Diagram

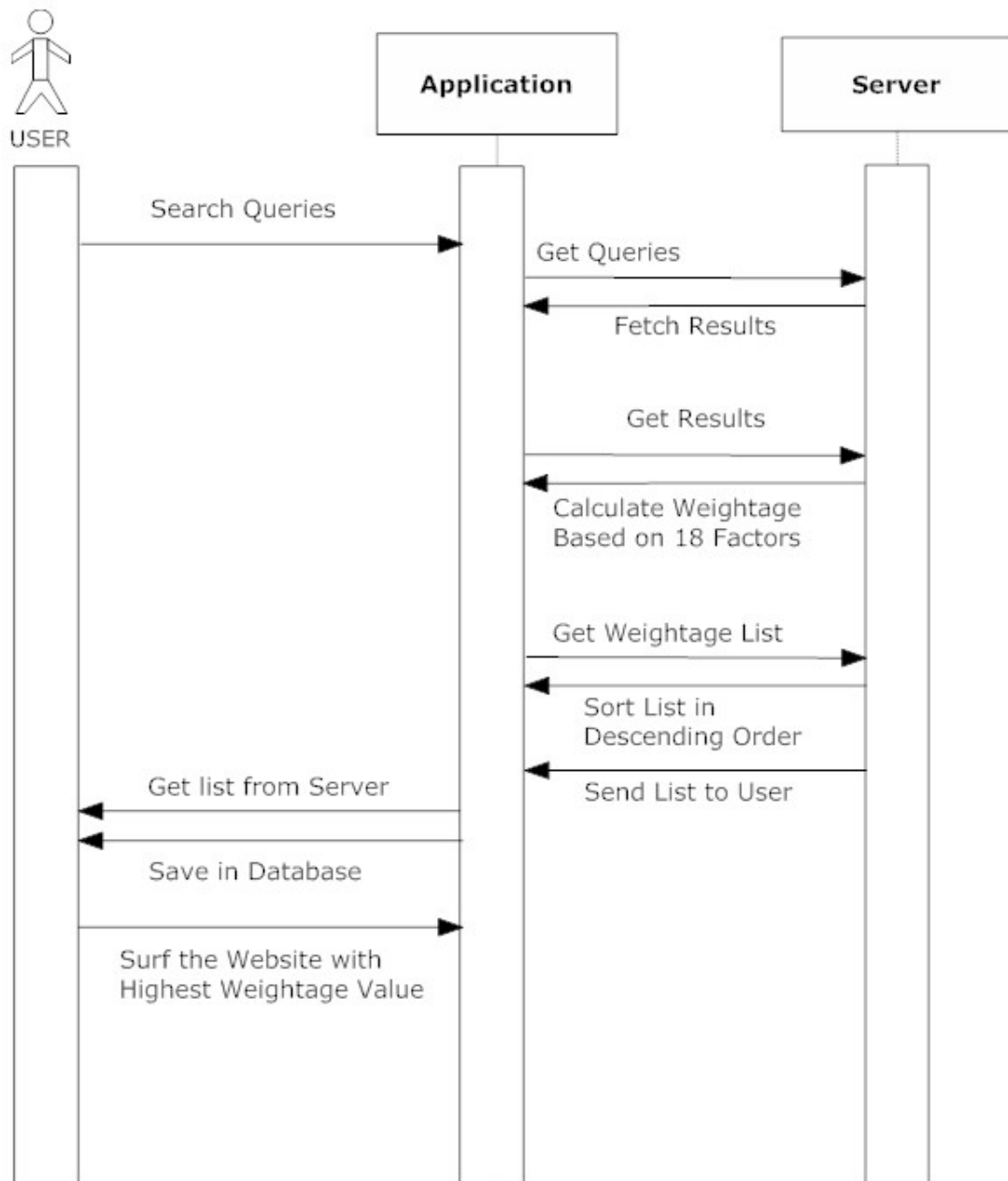


Figure 4.3: Sequence Diagram

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence. The sequence diagram for user is depicted in Figure 4.3. They may login to view the site if they have registered earlier. They can check if the website is phished or not and can add them to blacklisted websites. Then after checking the websites, they can give feedback. They can complete their work as per their convenience. They can log out if they do not wish to continue.

4.4 Class Diagram

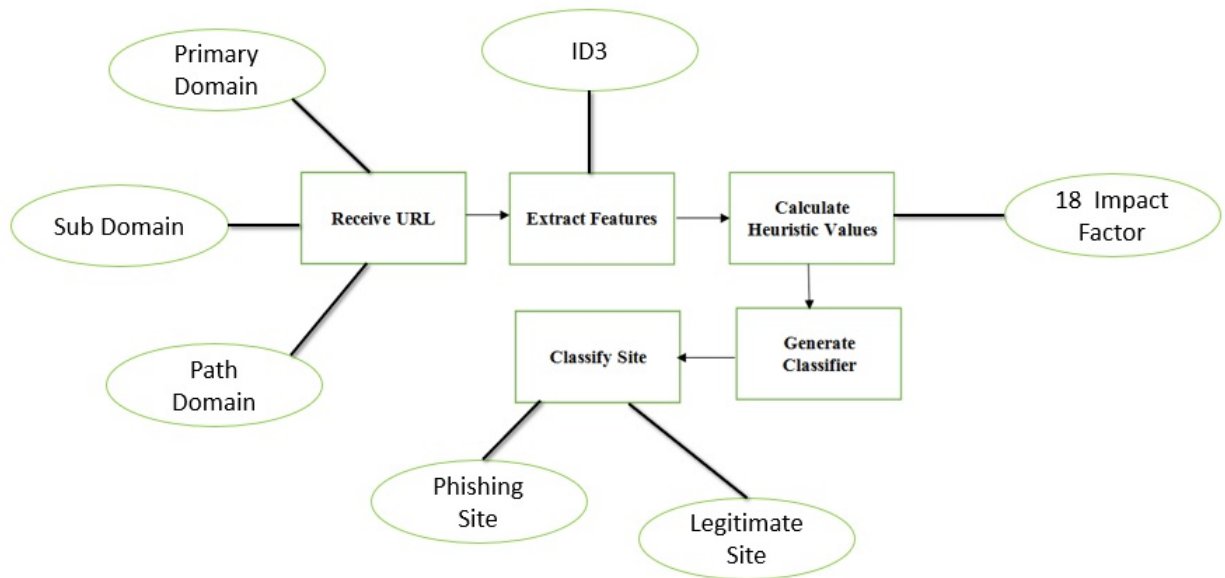


Figure 4.4: Class Diagram

4.5 Data Flow Diagram

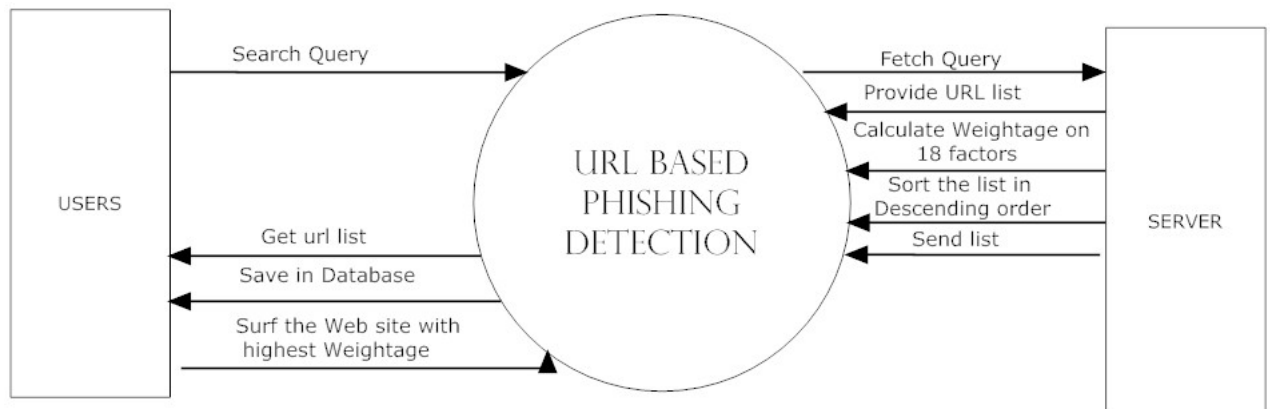


Figure 4.5: Data Flow Diagram

Chapter 5

Implementation

1. These in main Screen of our desktop application, They have two stages: Training and detection. We want to first train our system then we move to detection part of our application.

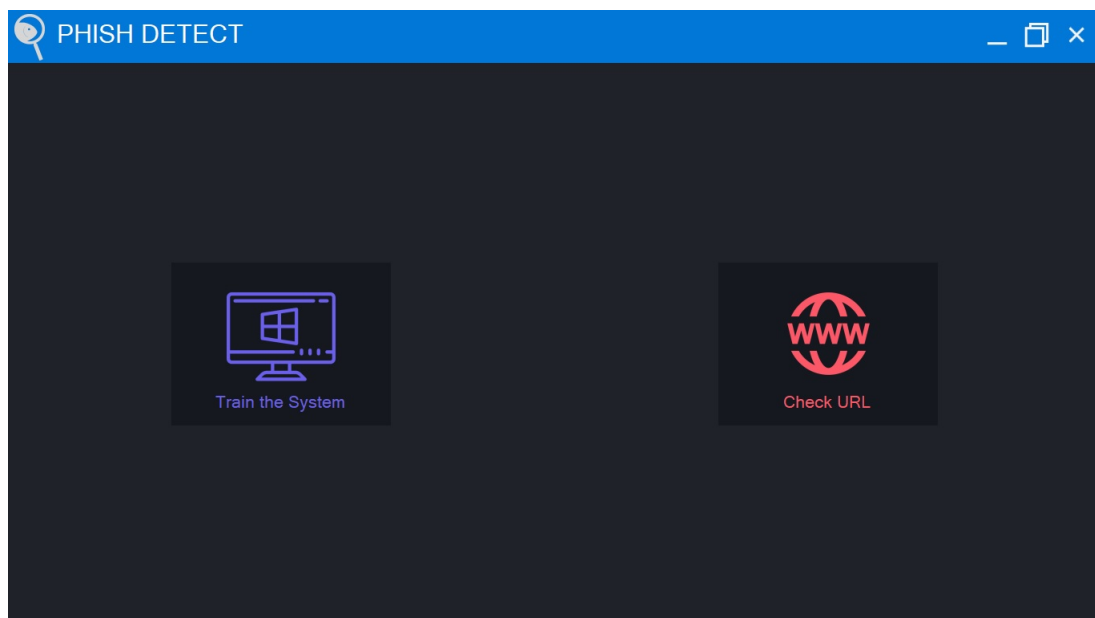


Figure 5.1: Main Screen

The following shows two button training system and detection system that is check URL. Select the button test file system start the training state.

2. After selecting test with system we want to upload the database which store in folders. We have collection of database which collect in our system .The file are in Excel format which is upload in application

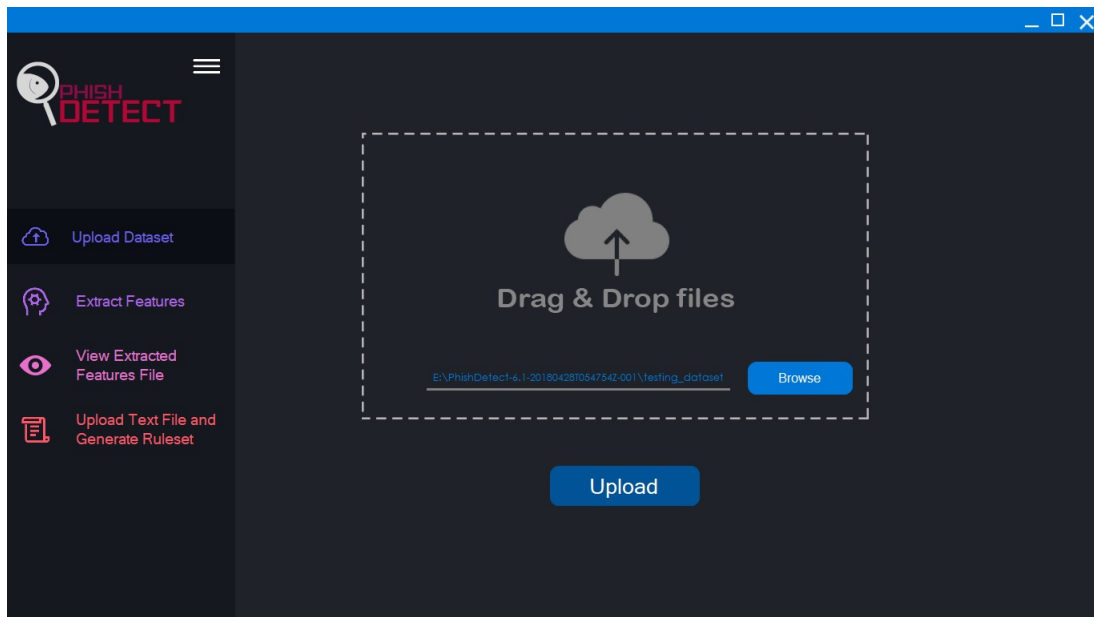


Figure 5.2: Training system upload database

Browse the database then we uploaded.

3. Extracting features of database Using ID3 algorithm of decision tree .Click the button EXTRACT FEATURES OF ID3 and extracted.

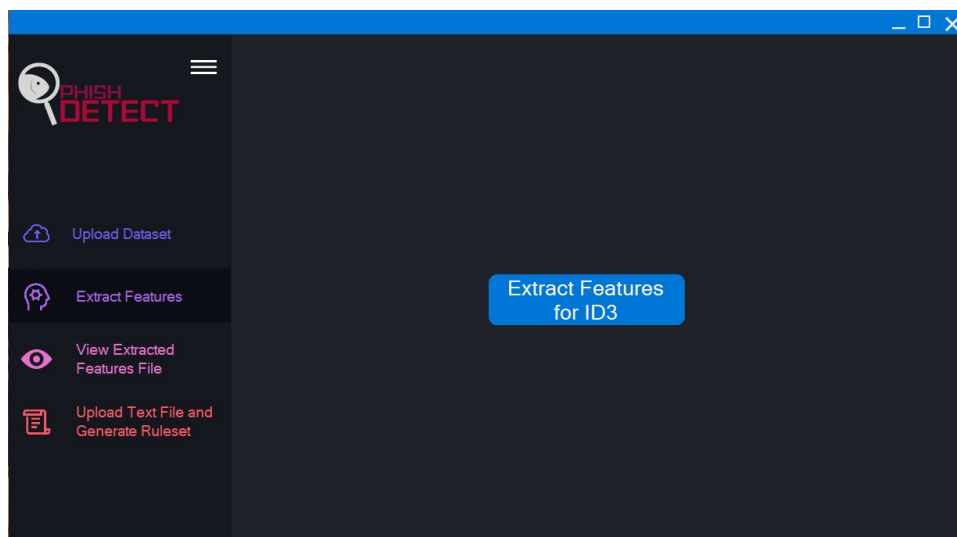



Figure 5.3: Extract the features using ID3

4. View of extracted featured data in terms of 1 and 0. 1 is stands for the value is true and 0 is stands for false value. 1 is stands of the value are present in URL and 0 is stands for the value of URL are not present. It also be categorized by URL which is IP contain, length of URL, suspicious character, prefix suffix, dots, slashes, HTTP present are not etc.


☰


View





	ip_contains	length_of_URL	suspicious_char	prefix_suffix	dots	sub_domain	slash	http_has
	0	0	0	0	1	0	1	1
	0	1	0	1	1	0	1	1
	0	1	0	0	1	0	1	1
	0	1	0	0	1	1	1	1
	0	1	0	0	1	0	1	1
	0	0	0	1	0	0	0	1
	0	1	0	0	1	1	1	1
	0	1	0	0	1	1	1	1
	0	1	0	1	1	1	1	1
	0	1	0	1	1	0	1	1
	0	1	0	0	1	0	1	1
	0	0	0	0	0	0	0	1
	0	1	0	0	1	0	1	1
	0	0	1	0	1	0	1	1
	0	0	0	0	0	0	0	1
	0	1	1	0	1	0	1	1
	0	1	0	0	1	0	1	1
	0	1	0	0	1	0	1	1
	0	1	0	0	1	0	1	1
	0	0	0	1	1	0	1	1
	0	0	0	1	1	0	1	1
	0	0	0	1	1	0	1	1
	0	0	0	1	1	0	1	1
	0	0	0	0	1	0	1	1
	0	0	0	1	1	0	1	1
	0	0	0	1	1	0	1	1

📁 Upload Dataset
⚙️ Extract Features
👁 View Extracted
Features File
📄 Upload Text File and
Generate Ruleset

Figure 5.4: View of Extracted features

5. There are Extracted feature is also generates some rule we applied in URL .first we upload the file which generates from extracting features and set some rule of URL which applied on phishing URL.



-  Upload Dataset
-  Extract Features
-  View Extracted Features File
-  Upload Text File and Generate Ruleset

ip	in	sc	pr	d0	ad	sl	fr	ph
0	1	0	1	1	0	1	1	0
0	1	0	0	1	0	1	1	0
0	1	0	0	1	1	1	1	0
0	1	0	0	1	0	1	1	0
0	0	0	1	0	0	0	1	1
0	1	0	0	1	1	1	1	1
0	1	0	0	1	1	1	1	0
0	1	0	1	1	0	1	1	0
0	1	0	0	1	0	1	1	0
0	1	0	0	1	1	1	1	0
0	0	0	0	0	0	0	1	0
0	1	1	0	1	0	1	1	0
0	1	0	0	1	0	1	1	0
0	1	0	0	1	0	1	1	0
0	0	0	1	1	0	1	1	1
0	0	0	1	1	0	1	1	1
0	0	0	1	1	0	1	1	1
0	0	0	0	1	0	1	1	0
0	0	0	1	1	0	1	1	0
0	0	0	1	1	0	1	1	1
0	0	0	1	1	0	1	1	1
0	0	0	1	1	0	1	1	1
0	0	0	1	1	0	1	1	0
0	0	0	1	1	0	1	1	0
0	0	0	1	1	0	1	1	1
0	1	0	0	1	0	1	1	0

ID3

Figure 5.5: View of uploaded database

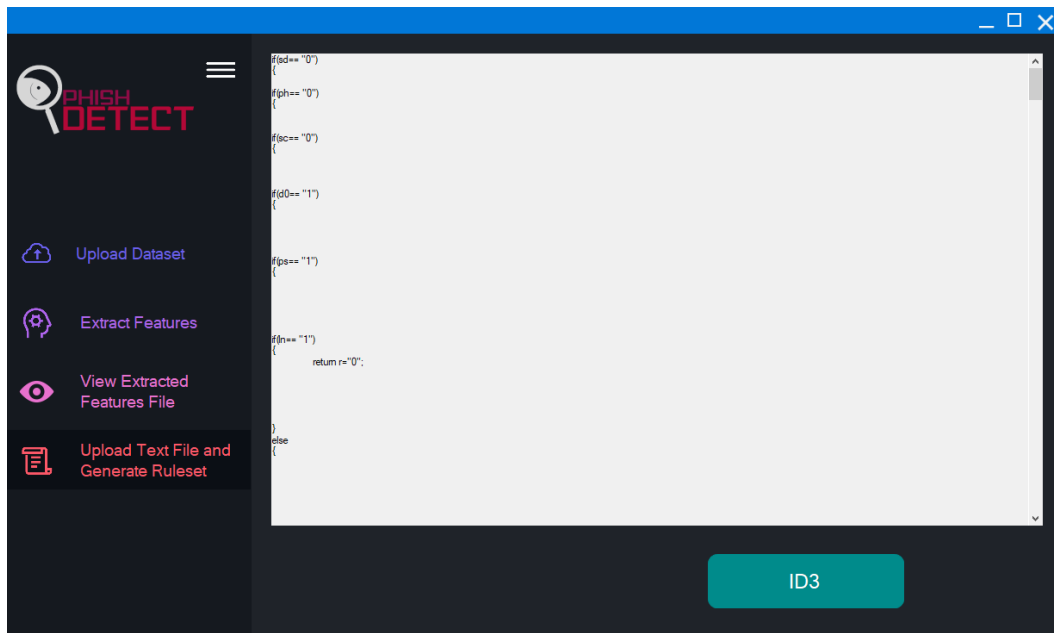


Figure 5.6: Generate Rules

6. There is a End of training part which done successfully in project Using ID3 algorithm.
7. Now, We move to detection part of software. In detection part we check URL which is phishing or Legitimate. click check URL button and move forward.

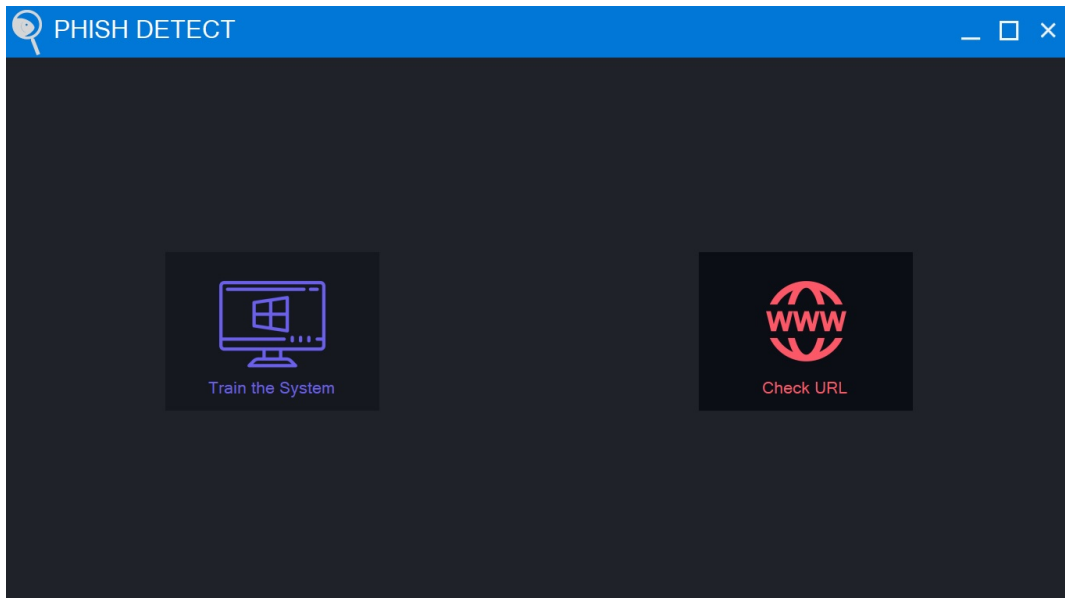


Figure 5.7: Check URL

8. Click Test URL button to check phishing site or not.

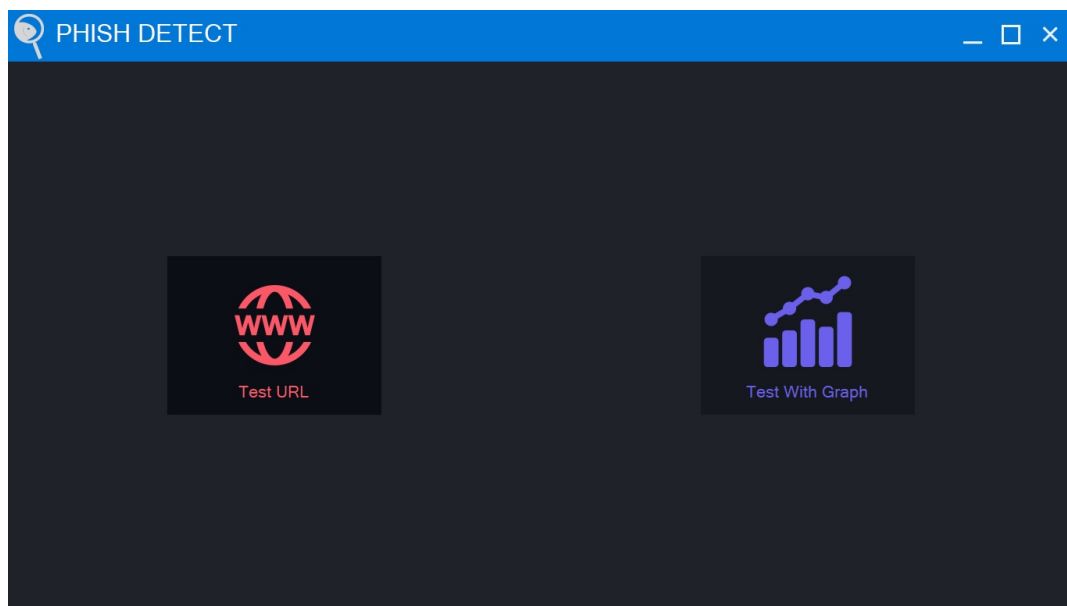


Figure 5.8: Test URL

9. In TEST URL there are one search button to input the value of URL.

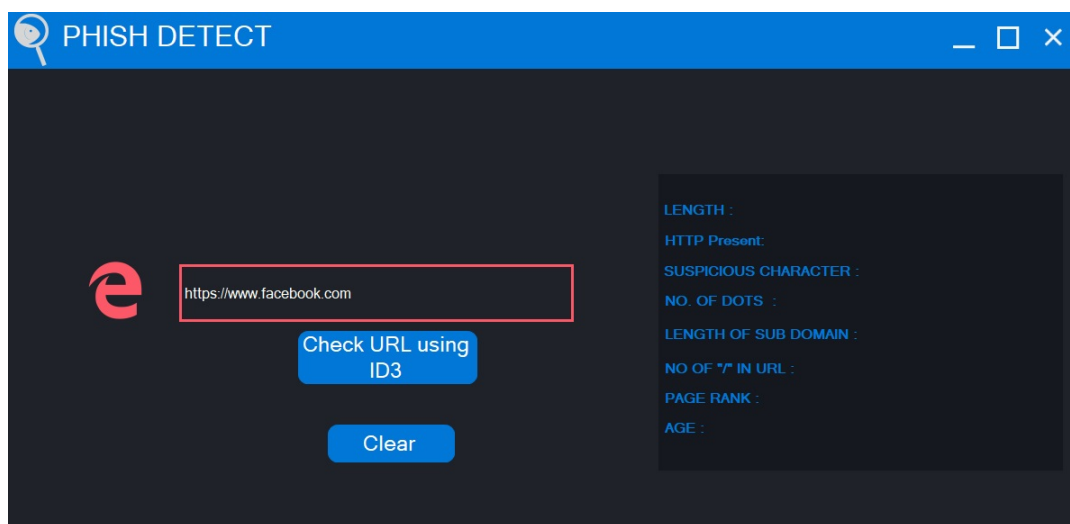


Figure 5.9: Search bar

After the putting URL we Check the URL using ID3 algorithm. After check we shown the result which is length of URL , HTTP present or not, suspicious character , Dots available or not , length of sub domain, number of URL, Page rank, Age of website.

10. Check with Id3 algorithm we know the current URL are phishing site or Legitimate site . If the site will be Legitimate it run successfully and open it into a browser are shown in follow:

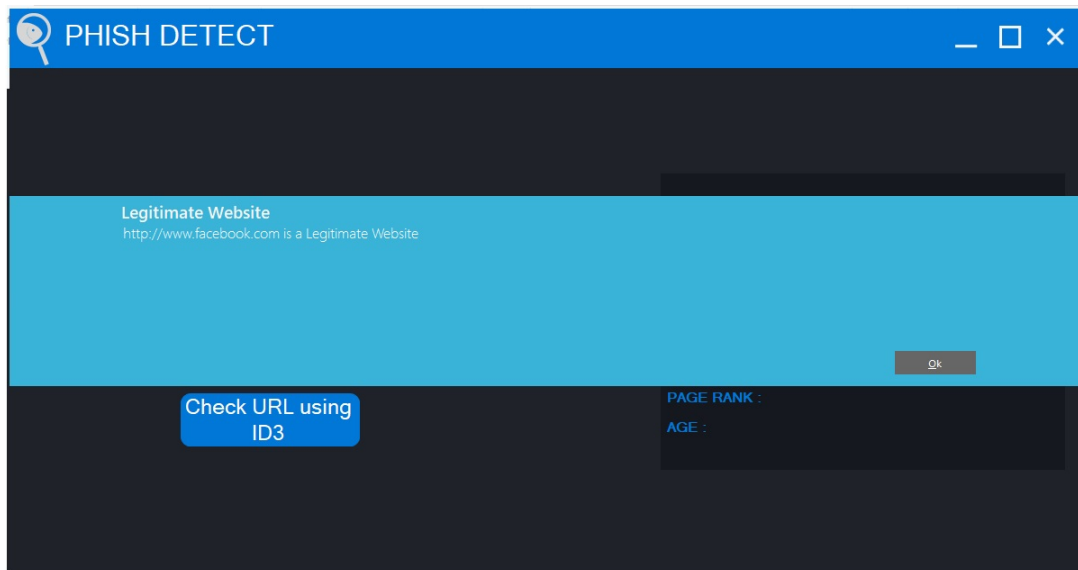


Figure 5.10: Legitimate site

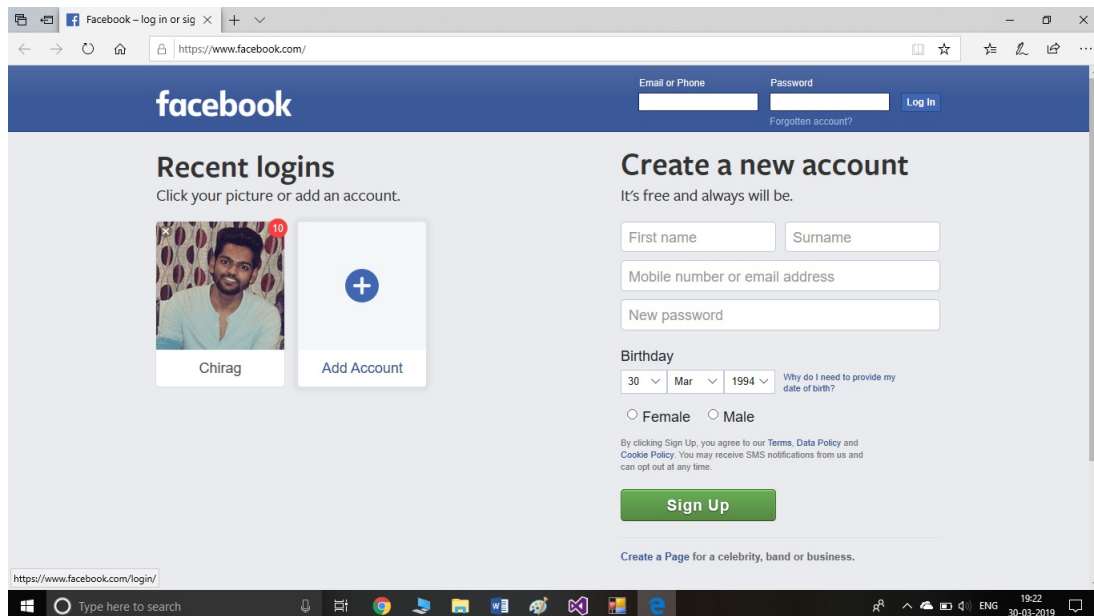


Figure 5.11: Open in Browser output

11. If the site will be phishing then it gives error message.

This is end of detection part, After completing detection part we know which URL

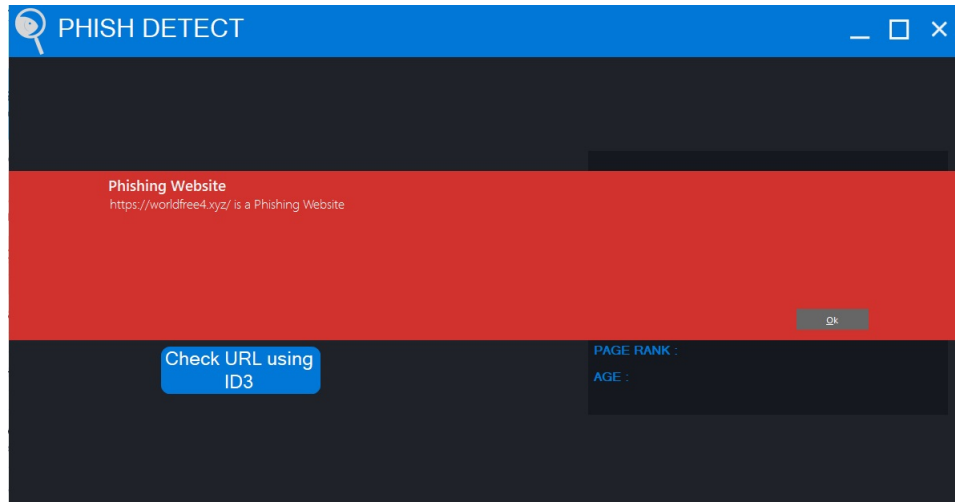


Figure 5.12: Phishing Site

sites are phishing or Legitimate. Then we skip the phishing site and done work proper on legitimate site.

12. After completing training and detection technique phase, We have accurate result of URL site. But, some site is pure legitimate and some site part will OK to use there for we done some calculation to create statistics of URL site. The calculation is used by decision tree to conclude TRUE POSITIVE, FALSE POSITIVE, TRUE NEGATIVE AND FALSE NEGATIVE web sites in database and categorized by them as following result. We found Accuracy, Specificity, Sensitivity and False measure of database and also shown in graph.

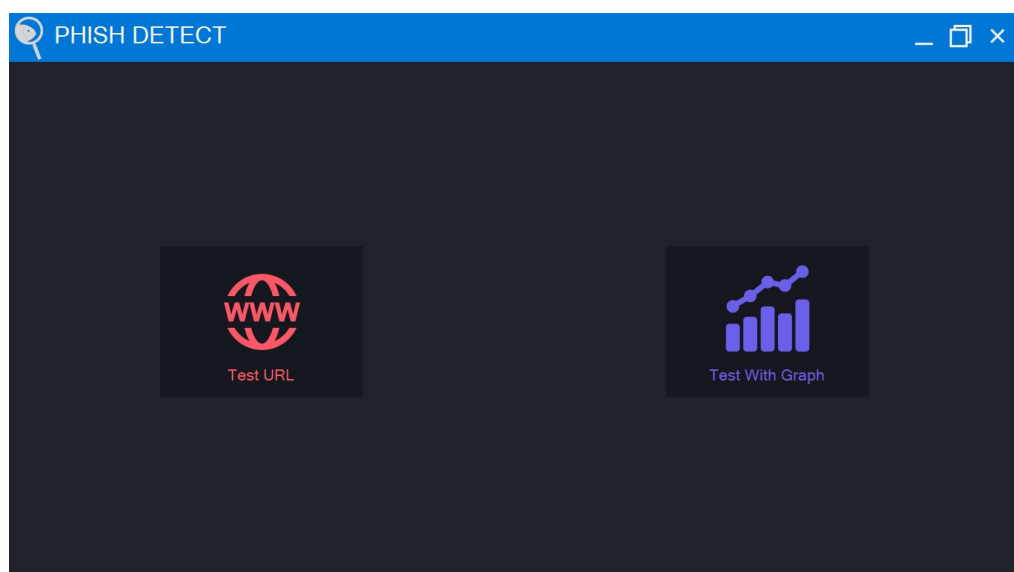


Figure 5.13: Test with graph

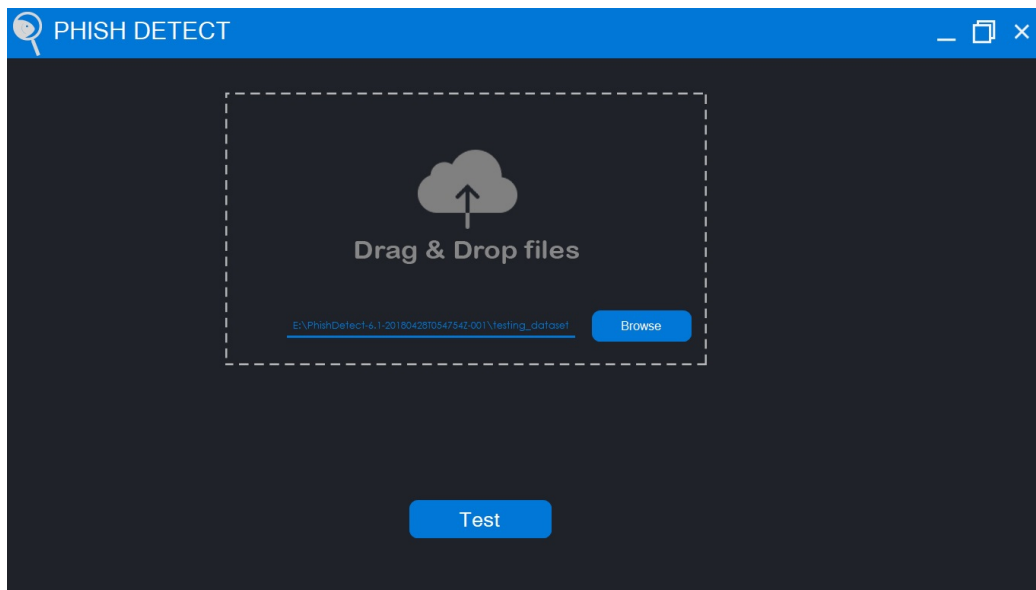


Figure 5.14: Upload Test database

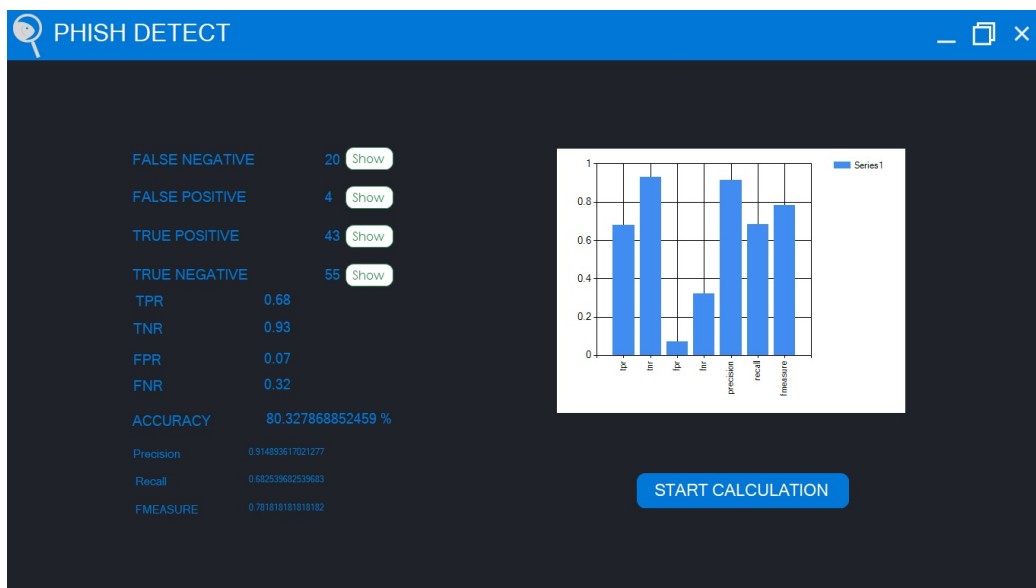


Figure 5.15: Show the Calculation

5.1 Result

To conduct classifier training and evaluation through an experimental data-set, we collected the URLs of phishing and legitimate sites. We gathered 3,000 phishing site URLs from PhishDetect and 3,000 legitimate site URLs from DMOZ. The evaluation was conducted using k-fold cross validation. Kfold cross validation divides the input data into k; k - 1 data sets are used for training, and the remaining one is used for validation. This process is performed k times, such as the number in the divided data-set, because all data-sets can be used for training and validation. This method is typically used to evaluate the accuracy of the classifier with a small dataset. In this study, we used ten-fold cross validation to evaluate our detection technique. We performed the testing with the WEKA open-source machine learning tool, and we analyzed the performance of each of the machine learning algorithms noted in Section 3. The accuracy was calculated as TP (true positive), TN (true negative), FP (false positive), and FN (false negative). We compared the performance of each classifier using the calculated accuracy. Fig. 5.1 depicts the TP, TN, FP, and FN matrix.

		Prediction	
		POSITIVE	NEGATIVE
Actual	TRUE	TRUE POSITIVE	FALSE NEGATIVE
	FALSE	FALSE POSITIVE	TRUE NEGATIVE

Table 5.1: TP,TN,FP,FN Matrix

TP is the ratio of the prediction that a determined phishing site is indeed a phishing site, and FN is the ratio of the prediction that a determined phishing site is actually a legitimate site. In addition, FP is the ratio of the prediction that a truly legitimate site is a phishing site, and TN is the ratio of prediction that a determined legitimate site is indeed a legitimate site. Table 2 shows the TP, TN, FP, FN ratios of each machine learning algorithm. As a result of the experiments, we obtained TP, TN, FP, and FN ratios to calculate three measurements that we used to compare the performance of each algorithm. The first measurement was for the specificity of the true negative rate. The second was the sensitivity of the true positive rate. The third was the accuracy of the total ratio of the prediction that a determined phishing site is actually a phishing site, and that a determined legitimate site is indeed legitimate. In measuring the classifier performance, (1) was the equation of specificity, (2) was the equation of sensitivity, and (3) was the equation of accuracy.

$$\text{Specificity} = \text{TN} / (\text{TN} + \text{FP}) \dots (1)$$

$$\text{Sensitivity} = \text{TP} / (\text{TP} + \text{FN}) \dots (2)$$

$$\text{Accuracy} = (\text{TP} + \text{TN} + \text{FP} + \text{FN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) \dots (3)$$

writing Figure axis labels to avoid confusing the reader. As an example, write the quantity Magnetization, or Magnetization, M, not just M. If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write Magnetization (A/m) or Magnetization A[m(1)], not just A/m. Do not label axes with a ratio of quantities and units. For example, write Temperature (K), not Temperature/K.

5.2 REQUIREMENTS ANALYSIS(Hardware And Software Requirements)

5.2.1 Hardware Description

The selection of hardware is very important in the existence and proper working of any software. When selecting hardware, the size and requirements are also important.

Minimum Requirements :

Processor	: Intel 1.66 GHz Processor Pentium 4
RAM	: 1 GB
Hard Disk Drive	: 250 GB

5.2.2 Software Description

The selection of software is very important in the existence and proper working of any hardware.

Minimum Requirements :

Operating System	: Windows XP/7/8/10
Front - End	: Visual Studio 2013
Back - End	: MS SQL Server 2008

Chapter 6

Testing Implementation

6.1 SOFTWARES TO BE USED FOR THE PROJECT

1. ASP.NET

- ASP.NET is more than the next version of Active Server Pages (ASP); it is a unified Web development platform that provides the services necessary for developers to build enterprise-class Web applications. While ASP.NET is largely syntax-compatible with ASP, it also provides a new programming model and infrastructure that enables a powerful new class of applications. You can migrate your existing ASP applications by incrementally adding ASP.NET functionality to them. ASP.NET is a compiled .NET Framework -based environment. You can author applications in any .NET Framework compatible language, including Visual Basic and Visual C. Additionally, the entire .NET Framework platform is available to any ASP.NET application. Developers can easily access the benefits of the .NET Framework, which include a fully managed, protected, and feature-rich application execution environment, simplified development and deployment, and seamless integration with a wide variety of languages.
- **.net Framework:**The .NET Framework is Microsoft's Managed Code programming model for building applications on Windows clients, servers, and mobile or embedded devices. Microsoft's .NET Framework is a software technology that is available with several Microsoft Windows operating systems. In the following sections describes, the basics of Microsoft .Net Frame work Technology and its related programming models. C is a language for professional programming. C (pronounced C sharp) is a programming language designed for building a wide range of enterprise applications that run on the .NET Framework. The goal of C is to provide a simple, safe, modern, object-oriented, high-performance, robust and durable language for .NET development. Also it enables developers to build solutions for the broadest range of clients, including Web applications, Microsoft Windows Forms-based applications, and thin- and smart-client devices.

2. Microsoft SQL Server:

- Relational database management: A relational database management system uses only its relational capabilities to manage the information stored in its database. Information Representation: All information stored in a RDBMS is represented only by data item values, which are stored in the tables that makeup the database.
- Logical Accessibility: Every data item value stored in a relational database is accessible by stating the name of the table it is stored in, the name of the column under which it is stored and the value of the primary key that defines the row in which it is stored.
- Representation of Null Values: The database management system has a consistent method for representing null values. For example, null values for numeric data must be distinct from zero or any other numeric value and for character data it must be different from a string of blanks or any other character value
- Catalog Facility: The logical description of the relational database management system is represented in the same manner as ordinary data. This is done so that the facilities of the relational database management system itself can be used to maintain database description.
- Data Language: A relational database management system may support many types of languages for description data and accessing the database. However, there must be at least one language that uses ordinary character strings to support the definition of data, the definition of views, the manipulation of data, constraints on data integrity, information concerning authorization and the boundaries for recovery of units.
- View Updating: Any view that can be defined using combinations of base tables, that are theoretically update-able, is capable of being updated by the relational database management system.
- Insert, Update and Delete: Any operand that describes the results of a single retrieval operation is capable of being applied to an insert, update or delete operation as well.
- Physical Data Independence: Changes made to physical storage representations or access methods do not require changes to be made to application programs. Logical Data Independence: Changes made to tables, that do not modify any data stored in that table, do not require changes to be made to application programs. Integrity Constraints: Constraints that apply to entity integrity and referential integrity are specifiable by the data language implemented by the database management system and not by the statements coded into the application programs.
- Database Distribution: The data language implemented by the relational database management system supports the ability to distribute the database without requiring changes to be made application programs.
- Not Subversion: If the relational database management supports facilities that allow application programs to operate on the tables a row at a time, an application program using this type of database access is prevented from bypassing entity integrity or referential integrity constraints that are defined for the database. Business today demands a different kind of data management solution.

Performance scalability, and reliability are essential, but businesses now expect more from their key IT investment. SQL Server 2005 exceeds dependability requirements and provides innovative capabilities that increase employee effectiveness, integrate heterogeneous IT ecosystems, and maximize capital and operating budgets. SQL Server 2005 provides the enterprise data management platform your organization needs to adapt quickly in a fast changing environment. Bench marked for scalability, speed, and performance, SQL Server 2005 is a fully enterprise-class database product, providing core support for Extensible Markup Language (XML) and Internet queries.

- **Easy-to-use Business Intelligence (BI) Tools:** Through rich data analysis and data mining capabilities that integrate with familiar applications such as Microsoft Office, SQL Server 2005 enables you to provide all of your employees with critical, timely business information tailored to their specific information needs. Every copy of SQL Server 2005 ships with a suite of BI services
- **Self-Tuning and Management Capabilities:** Revolutionary self-tuning and dynamic self-configuring features optimize database performance, while management tools automate standard activities. Graphical tools and performance, wizards simplify setup, database design, and performance monitoring, allowing database administrators to focus on meeting strategic business needs.
- **Data Management Application and Services:** Unlike its competitors, SQL Server 2005 provides a powerful and comprehensive data management platform. Every software license includes extensive management and development tools, a powerful extraction, transformation, and loading (ETL) tool, business intelligence and analysis services, and analysis service, and such as Notification Service. The result is the best overall business value available.

6.2 STEPS INVOLVED IN THE SYSTEM DEVELOPMENT LIFE CYCLE

Below are the steps involved in the System Development Life Cycle. Each phase within the overall cycle may be made up of several steps.

(a) Step 1: Software Concept

- The first step is to identify a need for the new system. This will include determining whether a business problem or opportunity exists, conducting a feasibility study to determine if the proposed solution is cost effective, and developing a project plan. This process may involve end users who come up with an idea for improving their work. Ideally, the process occurs in tandem with a review of the organization's strategic plan to ensure that IT is being used to help the organization achieve its strategic objectives. Management may need to approve concept ideas before any money is budgeted for its development.

(b) Step 2: Requirements Analysis

- Requirements analysis is the process of analyzing the information needs of the end users, the organizational environment, and any system presently being used, developing the functional requirements of a system that can meet the needs of the users. Also, the requirements should be recorded in a document, email, user interface storyboard, executable prototype, or some other form. The requirements documentation should be referred to throughout the rest of the system development process to ensure the developing project aligns with user needs and requirements. Professionals must involve end users in this process to ensure that the new system will function adequately and meets their needs and expectations.

(c) Step 3: Architectural Design

- After the requirements have been determined, the necessary specifications for the hardware, software, people, and data resources, and the information products that will satisfy the functional requirements of the proposed system can be determined. The design will serve as a blueprint for the system and helps detect problems before these errors or problems are built into the final system. Professionals create the system design, but must review their work with the users to ensure the design meets users' needs.

(d) Step 4: Coding and Debugging

- Coding and debugging is the act of creating the final system. This step is done by software developer.

(e) Step 5: System Testing

- The system must be tested to evaluate its actual functionality in relation to expected or intended functionality. Some other issues to consider during this stage would be converting old data into the new system and training employees to use the new system. End users will be key in determining whether the developed system meets the intended requirements, and the extent to which the system is actually used.

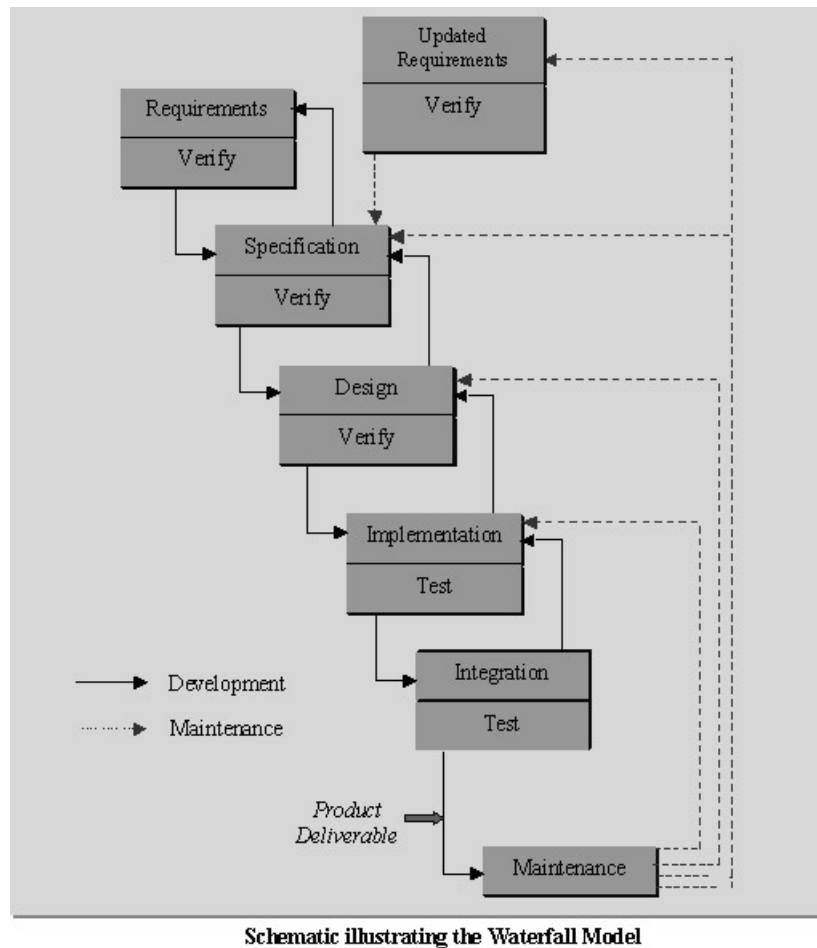
(f) Step 6: Maintenance

- Inevitably the system will need maintenance. Software will definitely undergo change once it is delivered to the customer. There are many reasons for the change. Change could happen because of some unexpected input values into the system. In addition, the changes in the system could directly affect the software operations. The software should be developed to accommodate changes that could happen during the post implementation period. There are various software process models like:-

1. Prototyping Model
2. RAD Model
3. The Spiral Model
4. The Waterfall Model
5. The Iterative Model

Of all these process models we've used the Iterative model (The Linear Sequential Model) for the development of our project.

Iterative model: The waterfall model derives its name due to the cascading effect from one phase to the other as is illustrated in Figure 1.1. In this model each phase well defined starting and ending point, with identifiable deliveries to the next phase. This model is sometimes referred to as the linear sequential model or the software life cycle.



Schematic illustrating the Waterfall Model

Figure 6.1: Waterfall Model

6.2.1 System Development Life Cycle:

The System Development Life Cycle is the process of developing information systems through investigation, analysis, design, implementation, and maintenance. The System Development Life Cycle (SDLC) is also known as Information Systems Development or Application Development. The model consists of six distinct stages, namely:

- (a) In the requirements analysis phase.
 - The problem is specified along with the desired service objectives (goals).
 - The constraints are identified.

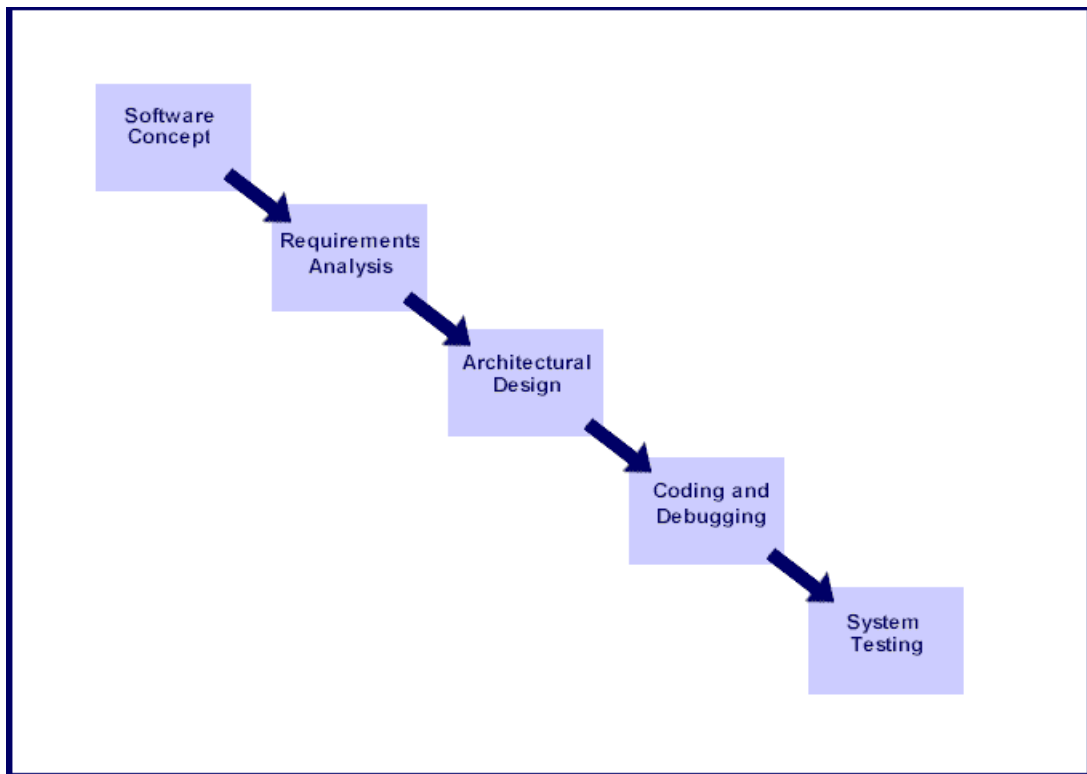


Figure 6.2: System Development Life Cycle (SDLC)

- (b) In the specification phase the system specification is produced from the detailed definitions of (a) and (b) above. This document should clearly define the product function.
- (c) In the system and software design phase, the system specifications are translated into a software representation. The software engineer at this stage is concerned with:
- Data structure
 - Software architecture
 - Algorithmic detail
 - Interface representations

The hardware requirements are also determined at this stage along with a picture of the overall system architecture. By the end of this stage should the software engineer should be able to identify the relationship between the hardware, software and the associated interfaces. Any faults in the specification should ideally not be passed down stream.

- (d) In the implementation and testing phase stage the designs are translated into the software domain
- Detailed documentation from the design phase can significantly reduce the coding effort.
 - Testing at this stage focuses on making sure that any errors are identified and that the software meets its required specification.

- (e) In the integration and system testing phase all the program units are integrated and tested to ensure that the complete system meets the software requirements. After this stage the software is delivered to the customer [Deliverable The software product is delivered to the client for acceptance testing.]
- (f) The maintenance phase the usually the longest stage of the software. In this phase the software is updated to:
 - Meet the changing customer needs
 - Adapted to accommodate changes in the external environment
 - Correct errors and oversights previously undetected in the testing phases
 - Enhancing the efficiency of the software

Observe that feed back loops allow for corrections to be incorporated into the model. For example a problem/update in the design phase requires a revisit to the specifications phase. When changes are made at any phase, the relevant documentation should be updated to reflect that change.

Advantages of the Iterative Model:-

- Testing is inherent to every phase of the Iterative model
- It is an enforced disciplined approach
- It is documentation driven, that is, documentation is produced at every stage

Disadvantages of the Iterative Model:- The waterfall model is the oldest and the most widely used paradigm. However, many projects rarely follow its sequential flow. This is due to the inherent problems associated with its rigid format. Namely:

- It only incorporates iteration indirectly, thus changes may cause considerable confusion as the project progresses.
- As The client usually only has a vague idea of exactly what is required from the software product, this IM has difficulty accommodating the natural uncertainty that exists at the beginning of the project.
- The customer only sees a working version of the product after it has been coded. This may result in disaster any undetected problems are precipitated to this stage.

Chapter 7

Methodology

ID3 Algorithm

This is my implementation of ID3 algorithm from Machine Learning field. As I was unable to find similar solutions for .NET platform I decided to write my own solution that uses ID3 algorithm to calculate decision tree for a given input.

Class Attribute is used as a container for input instances and classes TreeNode and Branch are used for Tree construction purposes inside DecisionTree class. As you can see, most of the work is done inside DecisionTree class. But although it looks like there are many methods inside it, in fact, it has a lot of helper methods that offload work from more important methods. Although it looks complicated, it is a very simple algorithm. Here is my implementation using recursion as that is the only possible way of forming the tree.

Coding:-

ID3.cs

```
using System;
using System.Collections;
using System.IO;
using System.Linq;

namespace ConsoleApplication1
{
    public class ID3
    {
        public string ReturnString { get; set; }

        private readonly bool _instanceFieldsInitialized = false;

        public ID3()
        {
            if (!_instanceFieldsInitialized)
            {
                InitializeInstanceFields();
                _instanceFieldsInitialized = true;
            }
        }

        private void InitializeInstanceFields()
        {
            _root = new TreeNode(this);
        }

        private int _numAttributes;
        private string[] _attributeNames;
        private ArrayList[] _domains;

        private class DataPoint
        {
            private readonly ID3 _outerInstance;

            public readonly int[] Attributes;
            public DataPoint(ID3 outerInstance, int numAttributes)
            {
                _outerInstance = outerInstance;
                Attributes = new int[numAttributes];
            }
        }

        public class TreeNode
        {
            private readonly ID3 _outerInstance;
```



```

        public double Entropy;
        public ArrayList Data;
        public int DecompositionAttribute;
        public int DecompositionValue;
        public TreeNode[] Children;
        public TreeNode Parent;
        public TreeNode(ID3 outerInstance)
        {
            _outerInstance = outerInstance;
            Data = new ArrayList();
        }
    }

    private TreeNode _root;

    protected virtual int GetSymbolValue(int attribute, string
symbol)
    {
        int index = _domains[attribute].IndexOf(symbol);
        if (index < 0)
        {
            _domains[attribute].Add(symbol);
            return _domains[attribute].Count - 1;
        }
        return index;
    }

    protected virtual int[] GetAllValues(ArrayList data, int
attribute)
    {
        var values = new ArrayList();
        int num = data.Count;
        for (int i = 0; i < num; i++)
        {
            var point = (DataPoint)data[i];
            var symbol = (string)_domains[attribute]
[point.Attributes[attribute]];

            int index = values.IndexOf(symbol);
            if (index == -1)
            {
                values.Clear();
                values.Add(symbol);
            }
            else
            {
                values.Clear();
            }
        }
    }

```

```

        public double Entropy;
        public ArrayList Data;
        public int DecompositionAttribute;
        public int DecompositionValue;
        public TreeNode[] Children;
        public TreeNode Parent;
        public TreeNode(ID3 outerInstance)
        {
            _outerInstance = outerInstance;
            Data = new ArrayList();
        }
    }

    private TreeNode _root;

    protected virtual int GetSymbolValue(int attribute, string
symbol)
    {
        int index = _domains[attribute].IndexOf(symbol);
        if (index < 0)
        {
            _domains[attribute].Add(symbol);
            return _domains[attribute].Count - 1;
        }
        return index;
    }

    protected virtual int[] GetAllValues(ArrayList data, int
attribute)
    {
        var values = new ArrayList();
        int num = data.Count;
        for (int i = 0; i < num; i++)
        {
            var point = (DataPoint)data[i];
            var symbol = (string)_domains[attribute]
[point.Attributes[attribute]];

            int index = values.IndexOf(symbol);
            if (index == -1)
            {
                values.Clear();
                values.Add(symbol);
            }
            else
            {
                values.Clear();
            }
        }
    }

```

```

protected virtual double CalculateEntropy(ArrayList data)
{
    int numdata = data.Count;
    if (numdata == 0)
    {
        return 0;
    }
    int attribute = _numAttributes - 1;
    int numvalues = _domains[attribute].Count;
    double sum = 0;
    for (int i = 0; i < numvalues; i++)
    {
        int count = 0;
        for (int j = 0; j < numdata; j++)
        {
            var point = (DataPoint)data[j];
            if (point.Attributes[attribute] == i)
            {
                count++;
            }
        }
        double probability = 1.0 * count / numdata;
        if (count > 0)
        {
            sum += -probability * Math.Log(probability);
        }
    }
    return sum;
}

protected virtual bool AlreadyUsedToDecompose(TreeNode node,
int attribute, int value)
{
    if (node.Children != null)
    {
        if (node.DecompositionAttribute == attribute &&
            node.DecompositionValue == value)
        {
            return true;
        }
    }
    if (node.Parent == null)
    {
        return false;
    }
    return AlreadyUsedToDecompose(node.Parent, attribute,
        value);
}

```

```

protected virtual double CalculateEntropy(ArrayList data)
{
    int numdata = data.Count;
    if (numdata == 0)
    {
        return 0;
    }
    int attribute = _numAttributes - 1;
    int numvalues = _domains[attribute].Count;
    double sum = 0;
    for (int i = 0; i < numvalues; i++)
    {
        int count = 0;
        for (int j = 0; j < numdata; j++)
        {
            var point = (DataPoint)data[j];
            if (point.Attributes[attribute] == i)
            {
                count++;
            }
        }
        double probability = 1.0 * count / numdata;
        if (count > 0)
        {
            sum += -probability * Math.Log(probability);
        }
    }
    return sum;
}

protected virtual bool AlreadyUsedToDecompose(TreeNode node,
int attribute, int value)
{
    if (node.Children != null)
    {
        if (node.DecompositionAttribute == attribute &&
            node.DecompositionValue == value)
        {
            return true;
        }
    }
    if (node.Parent == null)
    {
        return false;
    }
    return AlreadyUsedToDecompose(node.Parent, attribute,
        value);
}

```

```

    }
    protected virtual void DecomposeNode(TreeNode node)
    {
        double bestEntropy;
        bool selected = false;
        int selectedAttribute = 0;
        int selectedValue = 0;
        int numdata = node.Data.Count;
        int numinputattributes = _numAttributes - 1;
        double initialEntropy = bestEntropy = node.Entropy =
            CalculateEntropy(node.Data);
        System.Console.WriteLine(@"Entropy of " + node + @"=" +
            + node.Entropy);
        if (node.Entropy == 0)
        {
            return;
        }
        for (int i = 0; i < numinputattributes; i++)
        {
            int numvalues = _domains[i].Count;
            for (int j = 0; j < numvalues; j++)
            {
                if (AlreadyUsedToDecompose(node, i, j))
                {
                    continue;
                }
                var subset = GetSubset(node.Data, i, j);
                if (subset.Count == 0)
                {
                    continue;
                }
                var complement = GetComplement(node.Data, subset);
                var e1 = CalculateEntropy(subset);
                var e2 = CalculateEntropy(complement);
                var entropy = (e1 * subset.Count + e2 * complement.Count)
                    / numdata;
                if (!(entropy < bestEntropy)) continue;
                selected = true;
                bestEntropy = entropy;
                selectedAttribute = i;
                selectedValue = j;
            }
        }
        if (selected == false)
        {
            return;
        }
    }

```

```

node.DecompositionAttribute = selectedAttribute;
node.DecompositionValue = selectedValue;
node.Children = new TreeNode[2];
node.Children[0] = new TreeNode(this)
{
    Parent = node,
    Data = GetSubset(node.Data, selectedAttribute, selectedValue)
};
node.Children[1] = new TreeNode(this) { Parent = node };

for (int j = 0; j < numdata; j++)
{
    var point = (DataPoint)node.Data[j];
    if (node.Children[0].Data.IndexOf(point) >= 0)
    {
        continue;
    }
    node.Children[1].Data.Add(point);
}
DecomposeNode(node.Children[0]);
DecomposeNode(node.Children[1]);
node.Data = null;
}

public virtual int ReadData(string filename)
{
    System.IO.FileStream @in = null;
    try
    {
        Stream s = File.Open(filename, FileMode.Open, FileAccess.Read,
            FileShare.None);
        s.Close();
        @in = new FileStream(filename, FileMode.Open);
    }
    catch (Exception e)
    {
        Console.Error.WriteLine("Unable to open data file: " + filename
            + "\n" + e);
        return 0;
    }
    var bin = new System.IO.StreamReader(@in);
    string input;
    while (true)
    {
        input = bin.ReadLine();
        if (input == null)
        {

```

```

        Console.Error.WriteLine("No data found in the data file: "
            + filename + "\n");
        return 0;
    }
    if (input.StartsWith("//", StringComparison.Ordinal))
    {
        continue;
    }
    if (input.Equals(""))
    {
        continue;
    }
    break;
}
String[] tokenizer = input.Split('\t');
_numAttributes = tokenizer.Count();
if (_numAttributes <= 1)
{
    Console.Error.WriteLine("Read line: " + input);
    Console.Error.WriteLine("Could not obtain the names of
        attributes in the line");
    Console.Error.WriteLine("Expecting at least one input attribute
        and one output attribute");
    return 0;
}
_domains = new ArrayList[_numAttributes];
for (int i = 0; i < _numAttributes; i++)
{
    _domains[i] = new ArrayList();
}
_attributeNames = new string[_numAttributes];
for (int i = 0; i < _numAttributes; i++)
{
    _attributeNames[i] = tokenizer[i];
}
while (true)
{
    input = bin.ReadLine();
    if (input == null)
    {
        break;
    }
    if (input.StartsWith("//", StringComparison.Ordinal))
    {
        continue;
    }
    if (input.Equals(""))

```

```

    {
        continue;
    }

    tokenizer = input.Split('\t');
    int numtokens = tokenizer.Count();
    if (numtokens != _numAttributes)
    {
        Console.Error.WriteLine("Read " + _root.Data.Count + "data");
        Console.Error.WriteLine("Last line read: " + input);
        Console.Error.WriteLine("Expecting " + _numAttributes + "attributes");
        return 0;
    }
    var point = new DataPoint(this, _numAttributes);
    for (int i = 0; i < _numAttributes; i++)
    {
        point.Attributes[i] = GetSymbolValue(i, tokenizer[i]);
    }
    _root.Data.Add(point);
}
bin.Close();
return 1;
}

```

```

protected virtual void PrintTree(TreeNode node, string tab)
{
    int outputattr = _numAttributes - 1;
    if (node.Children == null)
    {
        int[] values = GetAllValues(node.Data, outputattr);
        if (values.Length == 1)
        {
            ReturnString += "\t" + "return" + " " + _attributeNames[outputattr] +
                "=\"" + _domains[outputattr][values[0]] + "\";" + Environment.NewLine;

            return;
        }
        ReturnString += tab + "\t" + _attributeNames[outputattr] + "= " +
            Environment.NewLine + "{" + Environment.NewLine;

        for (int i = 0; i < values.Length; i++)
        {
            ReturnString += "\"" + _domains[outputattr][values[i]] + "\"" +
                Environment.NewLine;
            if (i != values.Length - 1)
            {
                ReturnString += " , " + Environment.NewLine;
            }
        }
    }
}

```



```

        }
    }
    ReturnString += " };" + Environment.NewLine;
    return;
}
ReturnString += tab + "if(" + _attributeNames[node.DecompositionAttribute]
    + "==" + "\"" + _domains[node.DecompositionAttribute][node.DecompositionValue]
    + "\"" + Environment.NewLine + "{" + Environment.NewLine;

PrintTree(node.Children[0], tab + "\t" + Environment.NewLine);
ReturnString += tab + "}" + Environment.NewLine + "else" + Environment.NewLine +
    "{" + Environment.NewLine;

PrintTree(node.Children[1], tab + "\t" + Environment.NewLine);
ReturnString += tab + "}" + Environment.NewLine;
}

public virtual void CreateDecisionTree()
{
    DecomposeNode(_root);
    PrintTree(_root, "");
}
}
}

```

Chapter 8

Conclusions and Future Scope

8.1 Conclusions

The phishing attacks cause severe loss to companies, customers and web users. Social networking sites such as Facebook, Twitter and LinkedIn have been the victims of phishing. However, there are anti-phishing tools available which can help users to detect phishing attacks and prevent them. The Heuristic based Phishing Detection System detects the malicious URLs and specify the reason for classifying a URL as phishing which will help the users to be aware of such malicious and suspicious URLs. After the user enters a URL, the system provides 16 heuristics for the user to select and apply on input URL to determine whether a URL is malicious or legitimate. It stores all the input URLs in the database which can be retrieved for the later purpose. Apart from this, the system also displays the phishing results in form of bar graphs. The testing is done on all legitimate websites as well as malicious websites which are collected from phishDetect. The testing is done on combination of multiple heuristics as well as on individual heuristics to ensure the efficient functionality of system. From a set of URLs tested, majority of the URLs have been classified as correctly by the system. The evaluation of the system is done using a confusion matrix which lists the True Positives, True Negatives, False Positives and False Negatives. Once all this information is collected, the precision and recall are calculated for the system. Based on heuristics selected by the user, the precision and recall varies accordingly. For a better precision and recall, the false positives and false negatives can be reduced which will improve the accuracy of the classification.

8.2 Future scope:

- Certain additional heuristics such as Pop-up window, Abnormal URL, TTL value of the domain can be implemented in addition to the heuristics in the proposed system.
- The discriminative classifier algorithms such as Decision Tree, SVM, Boosting can be used to predict the URL category by training huge amount of the data extracted from the datasets.
- The WEKA (Waikato Engine for Knowledge Analysis) tool can be used in order to evaluate the precision and recall of heuristic based phishing detection system.

Bibliography

- [1] Khonji, Mahmoud, Youssef Iraqi, and Andrew Jones. Phishing detection: a literature survey. *Communications Surveys Tutorials*, IEEE 15.4(2013): 2091-2121.
- [2] APWG, Phishing activity trends paper.[online]. [http://docs.apwg.org/reports/APWG Global Phishing Report 1H 2014.pdf](http://docs.apwg.org/reports/APWG%20Global%20Phishing%20Report%201H%202014.pdf).
- [3] Huang, Huajun, Junshan Tan, and Lingxi Liu. "Countermeasure techniques for deceptive phishing attack." *New Trends in Information and Service Science*, 2009. NISS'09. International Conference on. IEEE, 2009.
- [4] Ma, Justin, et al. "Beyond blacklists: learning to detect malicious web sites from suspicious URLs." *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009 .
- [5] Nguyen, Luong Anh Tuan, et al. "A novel approach for phishing detection using URL-based heuristic." *Computing, Management and Telecommunications (ComManTel)*, 2014 International Conference on. IEEE, 2014.
- [6] Suman Bhattacharyya, Chetan kumar Pal, Praveen kumar Pandey, Detecting Phishing Websites, a Heuristic Approach, *International Journal of Latest Engineering Research and Applications (IJLERA)* ISSN: 2455-7137 Volume 02, Issue 03, March 2017, PP 120-129.
- [7] Wikipedia. (2015. March) Uniform Resource Locator. Available: [http://en.Wikipedia.org/wiki/Uniform resource locator](http://en.Wikipedia.org/wiki/Uniform_resource_locator).
- [8] Hou, Yung-Tsung, et al. Malicious web content detection by machine learning. *Expert Systems with Applications* 37.1 (2010): 55-60.
- [9] Jin-Lee Lee, Dong-Hyun Kim, Chang-Hoon, Lee, Heuristic-based Approach for Phishing Site Detection Using URL Features, *Proc. of the Third Intl. Conf. on Advances in Computing, Electronics and Electrical Technology - CEET 2015* Copyright Institute of Research Engineers and Doctors, USA .All rights reserved. ISBN: 978-1-63248-0569 doi: 10.15224/ 978-1-63248-056-9-84.
- [10] Kausar, Firdous, et al. "Hybrid Client Side Phishing Websites Detection Approach." *International Journal of Advanced Computer Science and Applications (IJACSA)* 5.7 (2014).

Appendices

Appendix-A

Installation steps of Visual studio 2017

Step 1:- Visual Studio can be downloaded from the following link <https://www.visualstudio.com/downloads/> You can select Visual Studio 2017 Community Edition Visual Studio 2017 Professional Edition (30 Day Free Trial) In this tutorial, we will install the Professional Edition

Step 2:- Click on the downloaded .exe file

Step 3:- In the next screen, click continue

Step 4:- Visual Studio will start downloading the initial files. Download speed will vary as per your internet connection.

Step 5:- In next screen, click install

Step 6:- In next screen, 1. Select ".Net desktop development" 2. Click install

Step 7:- Visual Studio will download the relevant files based on the selection in step 6

Step 8:- Once the download is done, you will be asked to reboot the PC

Step 9:- Post reboot, open the Visual Studio IDE 1. Select a theme of your choice 2. Click Start Visual Studio

Step 10:- In Visual Studio IDE, you can navigate to File menu to create new C applications

Appendix-B

Install Office 2019

step 1:- Press Windows + R to open Runprompt in Microsoft Edge or Internet Explorer.

step 2:- Alternatively, Chrome users selectSetup option, and Firefox users choose Save File option.

step 3:- Next, a User Control prompt will appear on your screen and says, Do you want to allow this app to make changes to your device?

step 4:- Then, choose Yes option.

step 5:- Now, the process of installation begins.

step 6:- Wait for few minutes until the installation process is completed, then you see a phrase says, Youre all set! The office is successfully installed now.

step 7:- After that, an animation plays show you where to find Office applications on your computer system.

step 8:- Choose Close tab.

step 9:- Lastly, follow the instruction appears on your window to easily find your Office apps.

Appendix-C

Screen Shot

phishing site detecting screen shot

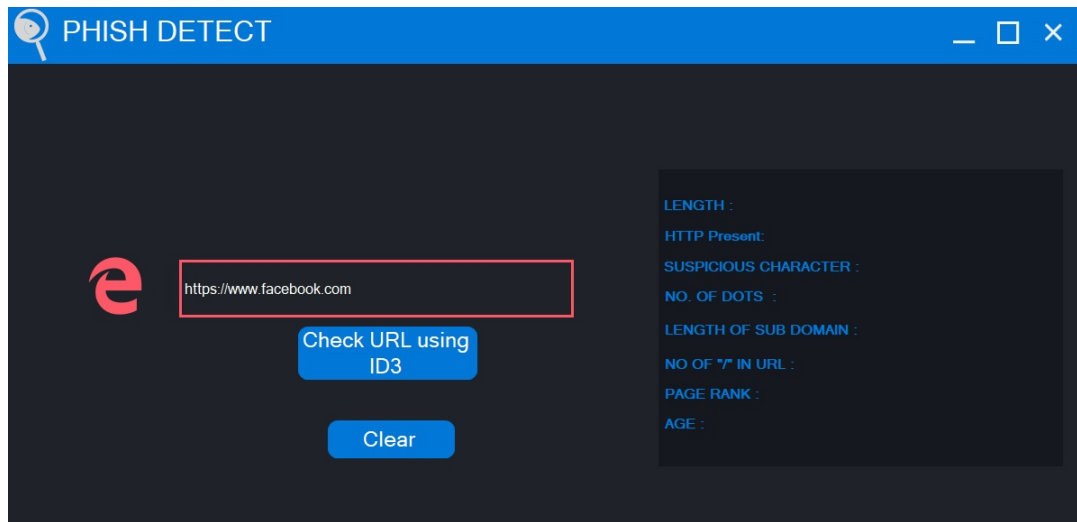


Figure 8.1: Detect Phishing Site

Publication

Paper entitled ” **Heuristic-based Approach for Phishing Site Detection** ” is published at ” **International Conference on Advances in Science, Technology and Engineering (ICASTe-2019)** ” by ”Chirag L. Chaudhari, Swapnil S. Ghawali, Swapnil r. Kshetre, Aakash A. Sane”.

Paper entitled ” **Heuristic-based Approach for Phishing Site Detection** ” is published at ” **International Research Journal of Engineering and Technology (IRJET-2019)** ” by ” Chirag L. Chaudhari, Swapnil S. Ghawali, Swapnil R. Kshetre, Aakash A. Sane”.