

## **WEB PHISHING DETECTION**

## **INTRODUCTION**

### **1.1 PROJECT OVERVIEW**

This project describes the machine learning based Over the last decade, many cyber-attacks start with a poisoned link in a seemingly harmless email. When you click on the link, it could be a phishing or malicious site. Phishing websites try to hook Internet surfers into revealing their sensitive information including credentials, bank account, and other personal information and malicious sites try to install malware onto your devices. These new, short-lived phishing URLs can easily bypass signature-based detectors. To combat this problem, researchers have also used machine learning methods to detect phishing websites. Nevertheless, there is still no definitive solution with machine learning or another approach.

The main objective of the web phishing process consists of

1. To create a dataset and apply necessary preprocessing followed by feature selection.
2. To apply various machine learning models and compare them based on different metrics.
3. To run all the algorithms in the selected cloud platform using IBM Cloud's AutoAI feature to validate the choose obtained previously.
4. To implement and deploy the selected machine learning model onto a cloud-based platform.

5. To predict the probability of a website being legitimate or phishing based on the URL of the website.

## 1.2 **PURPOSE**

The main purpose of the web phishing project Phishing is a form of fraud in which an attacker masquerades as a reputable entity or person in email or other forms of communication. Attackers will commonly use phishing emails to distribute malicious links or attachments that can perform a variety of functions. Some will extract login credentials or account information from victims.

## **LITERATURE SURVEY**

Construction of Phishing Site. In the first step attacker identifies the target as a well-known organization. Afterward, attacker collects the detailed information about the organization by visiting their website. The attacker then uses this information to construct the fake website URL Sending. In this step, attacker composes a bogus e-mail and sends it to the thousands of users. Attacker attached the URL of the fake website in the bogus e-mail. In the case of spear phishing attack, an attacker sends the e-mail to selected users. An attacker can also spread the link of phishing website with the help of blogs, forum, and so forth Stealing of the Credentials. When user clicks on attached URL, consequently, fake site is opened in the web browser. The fake website contains a fake login form which is used to take the credential of an innocent user. Furthermore, attacker can access the information filled by the user Identity Theft. Attacker uses this credential of malicious purposes. For example, attacker purchases something by using credit card details of the user. Although attacks use different techniques to create phishing websites to deceive users, most have similarly designed phishing website features. Therefore, researchers have conducted extensive anti-phishing research using phishing website features. Current methods for phishing detection include black and whitelists, heuristics, visual similarity, and machine learning, among which heuristics and machine learning are more widely used. The following is an introduction to the aforementioned phishing detection techniques Black and whitelist To prevent phishing attack threats, many anti-phishing methods have been proposed. Blacklisting methods are the most straightforward ways to prevent phishing attacks and are widely used in the industry. Google Safe Browsing uses a blacklist-based phishing detection method to check if

the URL of the matching website exists in the blacklist. If it does, it is considered a phishing website.

## **2.1 EXISTING WORK PROBLEM**

Many researchers have been working on phishing website detection for more than a decade now. Phishing site detection can be achieved from many perspectives using different sets of features, i.e., search-based, URL-based, content-based, or hybrid.

An influential search-based framework, CANTINA [48], uses TF-IDF scores of each term on the web page, then generates a lexical signature by taking the five terms with highest TF-IDF weights to feed into a search engine (Google). Detection is based on whether the domain of the current web page matches one of the domains in the top 30 search results

In the real world, there are many types of legitimate and phishing websites. Many new legitimate websites exist, which use very generic terms in their website content, e.g., nonprofit websites do this frequently, and have no logos in the web content. Such domains may not be easy to find, if the corresponding websites are not popular. Therefore, such methods tend to have a relatively high false positive rate, and must be complemented with other 9 features. Although our search-based features are inspired by [3, 48], they are novel, since we look for domain emails and subdomains rather than keywords from the content.

## **2.2 CONTENT-BASED**

a typical content-based phishing detector. The system will get HTML source code and URL of input webpage first. URL features normally just check internal and external links from HTML source code based on domain name. In HTML source code, there normally are four types of features that will be investigated and extracted, namely login forms, hyperlinks, CSS and JavaScript, and web identity features

## **2.2 REFERENCES**

1. J. Alamelu Mangai, V. Santhosh Kumar, and S. Appavu alias Balamurugan. A novel feature selection framework for automatic web page classification. *International Journal of Automation and Computing*, 9(4):442–448, Aug 2012
2. Ankesh Anand, Kshitij Gorde, Joel Ruben Antony Moniz, Noseong Park, Tanmoy Chakraborty, and Bei-Tseng Chu. Phishing URL detection with oversampling based on text generative adversarial networks. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1168–1177, Dec 2018
3. Choon Lin Tan and Kang Leng Chiew and San Nah Sze. Phishing website detection using URL-assisted brand name weighting system. In *2014 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, pages 054–059, Dec 2014.
4. Mehdi Babagoli, Mohammad Pourmahmood, Aghababa, and Vahid Solouk. Heuristic nonlinear regression strategy for detecting phishing websites. In *Soft Computing*, pages 4315–4327. Springer Berlin Heidelberg, June 2019.

5. Alejandro Correa Bahnsen, Eduardo Contreras Bohorquez, Sergio Villegas, Javier Vargas, and Fabio A. González. Classifying phishing URLs using recurrent neural networks. In 2017 APWG Symposium on Electronic Crime Research (eCrime). IEEE, 2017

### **2.3 THE PROBLEM CHALLENGES OF PROBLEM**

The problem of web phishing system Phishing is a major problem, which uses both social engineering and technical deception to get users' important information such as financial data, emails, and other private information. Phishing exploits human vulnerabilities; therefore, most protection protocols cannot prevent the whole phishing attacks. Phishing is a major threat to all Internet users and is difficult to trace or defend against since it does not present itself as obviously malicious in nature. In today's society, everything is put online and the safety of personal credentials is at risk. Phishing can be seen as one of the oldest and easiest ways of stealing information from people and it is used for obtaining a wide range of personal details. It also has a fairly simple approach – send an email, email sends victim to a site, site steals information

### 2.3.1 IDEATION & PROPOSED SOLUTION





## 2.3.2 PROPOSED SOLUTIONS

Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> Who is your customer? i.e. working parents of 3-5 y.o. kids <ul style="list-style-type: none"><li>C-suite executives</li><li>Internet based financial services business</li><li>Online payment service users</li></ul>	<b>6. CUSTOMER CONSTRAINTS</b> <span>CC</span> What constraints prevent your customers from taking action or limit their choices of solutions? (i.e. spending power, budget, no cash, network connection, available devices). <ul style="list-style-type: none"><li>Prevent access to third party websites</li><li>multi step verification</li><li>prevent entry to unwanted websites</li><li>Frequent change of passcodes</li></ul>	<b>5. AVAILABLE SOLUTIONS</b> <span>AS</span> Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? (i.e. pen and paper is an alternative to digital note taking) <ul style="list-style-type: none"><li>Web security gateway</li><li>Secure web gateway</li><li>Spam filter</li><li>use of VPN</li><li>Check for site seals</li><li>Firewalls and proxy</li><li>Antispyware software</li></ul>	Explore AS, differentiate
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>JBP</span> Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different angles. <ul style="list-style-type: none"><li>Prevent personal data getting stolen</li><li>Ensure user safety</li><li>Intimating the suspicious activity or log in attempts</li></ul>	<b>9. PROBLEM ROOT CAUSE</b> <span>RC</span> What is the real reason that this problem exists? What is the back story behind the need to do this job? <ul style="list-style-type: none"><li>Large user base</li><li>Leniency in the adoption of security measure</li><li>Lack of awareness where layman prohibits every websites looking legitimate</li></ul>	<b>7. BEHAVIOUR</b> <span>BE</span> What does your customer do to address the problem and get the job done? <ul style="list-style-type: none"><li>Using instant firewalls</li><li>Back up files</li><li>Scan System for malware</li><li>Change credentials</li><li>Set up a fraud alert</li></ul>	
Focus on CS, fit into BE, mention RC				Focus on BE, fit into RC, understand CC

<b>3. TRIGGERS</b> What triggers customers to act? <ul style="list-style-type: none"><li>• Coupons and gift voucher s</li><li>• Attractive advertisement and pop - ups</li><li>• Assuming everything is legitimate website</li></ul>	<b>TR</b>	<b>10. YOUR SOLUTION</b> If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour. <ul style="list-style-type: none"><li>• Pop - up alert for fake websites</li><li>• Check websites authenticity</li><li>• Whitelist filtering</li><li>• Blacklist interception</li></ul>	<b>SL</b>	<b>8. CHANNELS of BEHAVIOUR</b> <b>8.1 ONLINE</b> What kind of actions do customers take online? Extract online channels from <ul style="list-style-type: none"><li>• Don't use insecure public channels while doing transactions</li><li>• Back up files</li><li>• Scan system for malware</li><li>• Set up a fraud alert</li></ul> <b>8.2 OFFLINE</b> What kind of actions do customers take offline? Extract offline channels from 8.1 and use them for customer development.	<b>CH</b>
<b>4. EMOTIONS: BEFORE / AFTER</b> How do customers feel when they face a problem or a job and afterwards? <b>Before the job is done:</b> Threatened, scared, anxious, stressed, lost <b>After the job is done:</b>	<b>EM</b>				

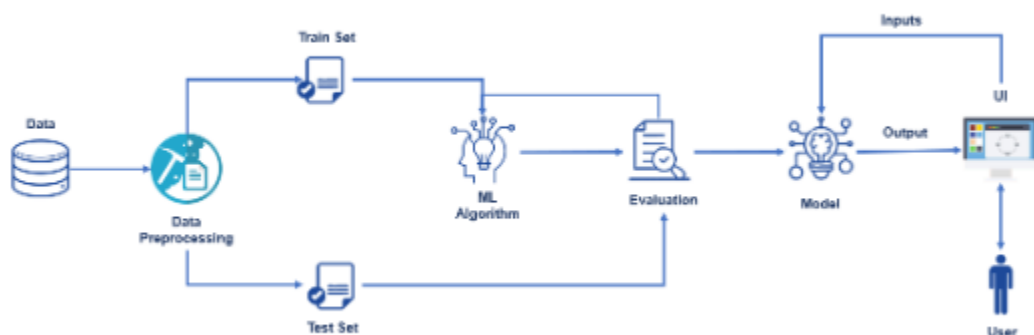


Figure: technical architecture

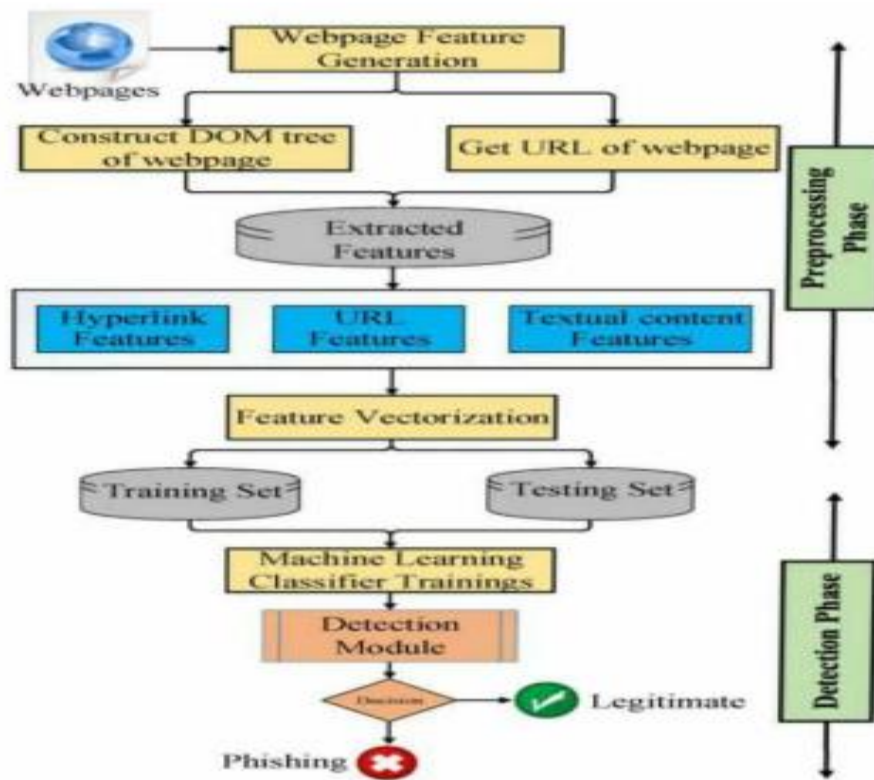
## **4 REQUIREMENT ANALYSIS**

### **4.1 FUNCTIONAL REQUIREMENT**

#### **Solution Architecture:**

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

Find the best tech solution to solve existing business problems. ● Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders. ● Define features, development phases, and solution requirements. ● Provide specifications according to which the solution is defined, managed, and delivered.



## 4.2 NON FUNCTIONL REQUIRMENT

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story/Task	Story Points	Priority	Team Members
Sprint-1	Login	USN-1	As a user, can navigate into the website	1	High	Gowtham
Sprint-1	Dashboard	USN-2	As a user, I will input any sites url, in the form to check it's genuineness	1	High	Arunkumar
Sprint-1		USN-3	As a user, I can see the output	2	High	Blesson Joshua
Sprint-2	Backend	USN-4	As an admin, if a new URL is found, I can add the new state into the database	3	Medium	Dhiyaneshwar
Sprint-3	Report	USN-5	As a user, I can ask my queries and report suspicious site in the report box	1	Low	Blesson Joshua
Sprint-4		USN-6	As an admin, I can take actions to the queries asked by the user	2	Low	Dhiyaneshwar

A characteristic of a quality SRS is that in addition to describing the functional requirements of a system, It will also provide detailed coverage of the non-functional requirements. In practice, this would entail detailed analysis of issues such as availability, security, usability and maintainability. However, as this document is only an outline specification, it does not contain the same degree of rig our that would normally be expected in a formal SRS. Therefore, the sections below should be seen as indicative rather than providing specific (l.e. testable) requirements.

## **5 PROJECT DESIGN**

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

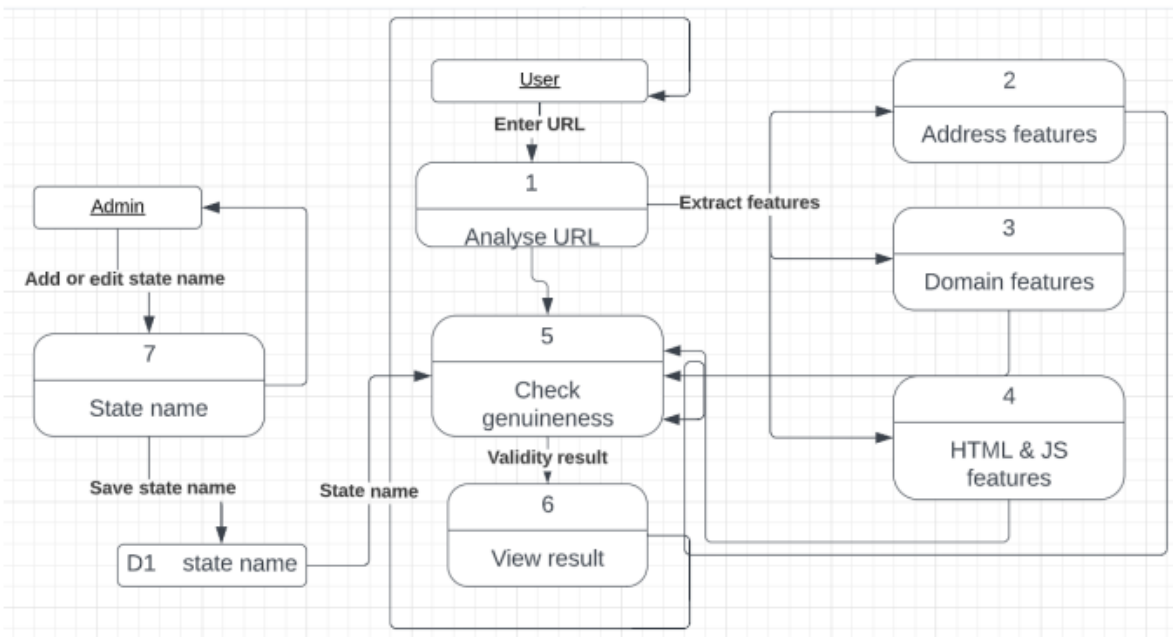
### **5.1 DATA FLOW DIAGRAMS**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the

system,

W

### DFD level 0



## 5.2 SOLUTIONS & TECHNICAL ARCHITECTURE

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Login	USN-1	As a user, I can navigate into the website	I can access the page	High	Sprint-1
	Dashboard	USN-2	As a user, I will paste the URL that needs to be checked if it's a phishing website or not	I can paste the URL in the text box	High	Sprint-1
		USN-3	As a user, I can see the output	I can see if it's a safe site	High	Sprint-1
Administrator		USN-4	If a new URL is found, I can add the new state into the database	I can add the new URL	Medium	Sprint-2

## PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Planning & Estimation

### 6.2 sprint delivery Schedule

## Project planning and scheduling

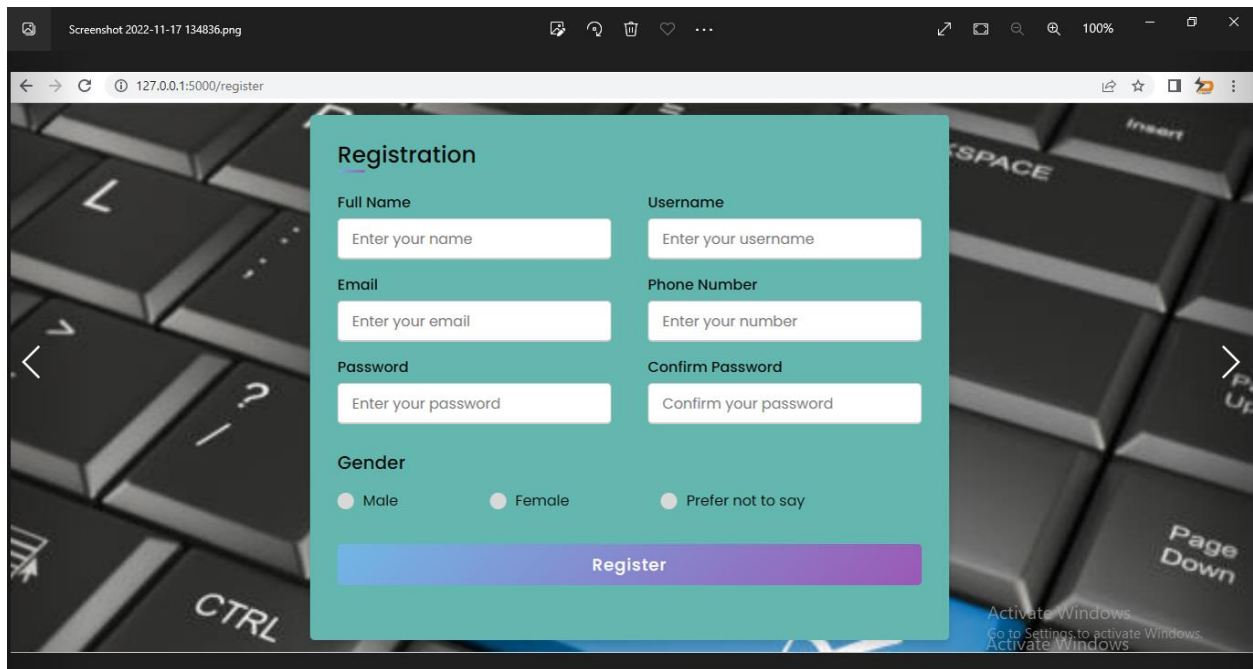
<b>Milestones</b>	<b>Activities</b>
Project development phase	Delivery of sprint- 1,2,3,4
Create and configure and IBM cloud services	Create IBM Watson
Create and access deep learning	Create v1 to interact with app deploy
	Create IBM and connect with python
Create & database in cloud and DB	Launch the cloudant DB and Create database
Develop the python flask	Install the python software
	Develop python code
Create the web application	Develop the web application

<b>Milestones</b>	<b>Activities</b>	<b>Description</b>
Ideation phase	literature	Literature survey on the selected project & information gathering
	Empathy Map	Prepare empathy map to capture the user pains & gains, prepare list of problem statement
	Ideation	Organizing the brainstorming session and priorities the top 3 ideas based on feasibility & importance

<b>sprint</b>	<b>Functional requirement</b>	<b>User story/task</b>	<b>Story priority points</b>	<b>Team member</b>
<b>Sprint 1</b>	Registration	User enter the details can register with details	20 high	<b>Blesson Joshua Dhiyaneshwar Arun kumar gowtham</b>
<b>Sprint 2</b>	Training of dataset	We can collect the dataset train the model using data	20 high	<b>Blesson Joshua Dhiyaneshwar Arun kumar gowtham</b>
<b>Sprint 3</b>	Prediction	Based on the model we can build the model	20 high	<b>Blesson Joshua Dhiyaneshwar Arun kumar gowtham</b>

# RESULTS

## Registration Page



The screenshot shows a web browser window with a dark theme. The address bar displays "127.0.0.1:5000/register". A teal-colored registration form is centered over a background image of a computer keyboard. The form contains the following fields and options:

- Full Name**: Input field with placeholder "Enter your name".
- Username**: Input field with placeholder "Enter your username".
- Email**: Input field with placeholder "Enter your email".
- Phone Number**: Input field with placeholder "Enter your number".
- Password**: Input field with placeholder "Enter your password".
- Confirm Password**: Input field with placeholder "Confirm your password".
- Gender**: Three radio button options: "Male", "Female", and "Prefer not to say".
- Register**: A large, wide button with a blue-to-purple gradient.

At the bottom right of the keyboard background, there is a watermark that reads "Activate Windows Go to Settings to activate Windows Activate Windows".

## Prediction result



127.0.0.1:5000/index

### Prediction

SSLfinal_State	URL_of_Anchor
<input type="text" value="SSLfinal_State"/>	<input type="text" value="URL_of_Anchor"/>
Prefix_Suffix	web_traffic
<input type="text" value="Prefix_Suffix"/>	<input type="text" value="Web_Traffic"/>
Domain_registration_length	
<input type="text" value="Domain_registration_length"/>	

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

Find schemas or tables Refresh

<input checked="" type="checkbox"/>	Name	Type	Tables
<input checked="" type="checkbox"/>	RCV94964	User	2

Total: 1, selected: 1

<input type="checkbox"/>	Name	Schema	Properties
<input type="checkbox"/>	PROTB	RCV94964	...
<input type="checkbox"/>	REGTB	RCV94964	...

Total: 2, selected: 0

Activate Windows  
Go to Settings to activate Windows

## TRAINING OF DATASET

```
import numpy as np
import pandas as pd
% loading data
raw_data = pd.read_csv('100-legitimate-art.txt')
msg = 'http://www.emuck.com:3000/archive/egan.html'
msg = 'http://www.emuck.com:3000/archive/egan.html'
%websites
data = {'websites':[msg]}
raw_data = pd.DataFrame(data)
% raw_data['websites'].str.split("://").head()

seperation_of_protocol = raw_data['websites'].str.split("://",expand = True)

%seperation_of_protocol.head()

type(seperation_of_protocol)
%
seperation_domain_name = seperation_of_protocol[1].str.split("/",1,expand
= True)

def long_url(l):
    """This function is defined in order to differntiate website based on
the length of the URL"""
    if len(l) < 54:
        return 0
    elif len(l) >= 54 and len(l) <= 75:
        return 2
    return 1

splitted_data['long_url'] = raw_data['websites'].apply(long_url)
def have_at_symbol(l):
    """This function is used to check whether the URL contains @ symbol or
not"""
    if "@" in l:
        return 1
    return 0

%
e have imported re module in the above feature. So need not to import again
```

```
splitted_data['shortening_service'] = raw_data['websites'].apply(shortening_service)
```

```
def shortening_service(url):  
    match=re.search('bit\.ly|goo\.gl|shorte\.st|go2l\.ink|x\.co|ow\.ly|t\.co|tinyurl|tr\.im|is\.gd|cli\.gs|'  
                    'yfrog\.com|migre\.me|ff\.im|tiny\.cc|url4\.eu|twit\.a|c|su\.pr|twurl\.nl|snipurl\.com|'  
                    'short\.to|BudURL\.com|ping\.fm|post\.ly|Just\.as|bkit e\.com|snipr\.com|fic\.kr|loopt\.us|'  
                    'doiop\.com|short\.ie|kl\.am|wp\.me|rubyurl\.com|om\.l|y|to\.ly|bit\.do|t\.co|lnkd\.in|'  
                    'db\.tt|qr\.ae|adf\.ly|goo\.gl|bitly\.com|cur\.lv|tinyurl\.com|ow\.ly|bit\.ly|ity\.im|'  
                    'q\.gs|is\.gd|po\.st|bc\.vc|twitthis\.com|u\.to|j\.mp|buzurl\.com|cutt\.us|u\.bb|yourls\.org|'  
                    'x\.co|prettylinkpro\.com|scrnch\.me|filoops\.info|vzturl\.com|qr\.net|lurl\.com|tweez\.me|v\.gd|tr\.im|link\.zip\.net',url)  
    if match:  
        return 1  
    else:  
        return 0
```

## CODE

```
from flask import Flask,request, url_for, redirect, render_template import pickle

import numpy as np

app = Flask(__name__)

model=pickle.load(open('model.pkl','rb'))

@app.route('/')

def login():

    return render_template("login.html")

@app.route('/register')

def register():

    return render_template("register.html")

@app.route('/index')

def index(): return render_template("view.html")

@app.route('/predict',methods=['POST','GET'])

def predict():

    int_features=[int(x) for x in request.form.values()]

    final=[np.array(int_features)]

    print(int_features)

    print(final)

    prediction=model.predict(final)[0]

    if prediction==1:

        return render_template('view.html',pred='This website is safe.'.format(prediction))

    else:

        return render_template('view.html',pred='This website is not safe.'.format(prediction))

if __name__ == '__main__':

    app.run(debug=True)

import socket
def statistical_report(url):
```

```

hostname = url
h = [(x.start(0), x.end(0)) for x in re.finditer('https://|http://|www
.|https://www.|http://www.', hostname)]
z = int(len(h))
if z != 0:
    y = h[0][1]
    hostname = hostname[y:]
    h = [(x.start(0), x.end(0)) for x in re.finditer('/', hostname)]
    z = int(len(h))
    if z != 0:
        hostname = hostname[:h[0][0]]
    url_match=re.search('at\.ua|usa\.cc|baltazarpresentes\.com\.br|pe\.hu|
esy\.es|hol\.es|sweddy\.com|myjino\.ru|96\.lt|ow\.ly',url)
    try:
        ip_address = socket.gethostbyname(hostname)
        ip_match=re.search('146\.112\.61\.108|213\.174\.157\.151|121\.50\
168\.88|192\.185\.217\.116|78\.46\.211\.158|181\.174\.165\.13|46\.242\.145
\.103|121\.50\.168\.40|83\.125\.22\.219|46\.242\.145\.98|107\.151\.148\.44
|107\.151\.148\.107|64\.70\.19\.203|199\.184\.144\.27|107\.151\.148\.108|1
07\.151\.148\.109|119\.28\.52\.61|54\.83\.43\.69|52\.69\.166\.231|216\.58\
.192\.225|118\.184\.25\.86|67\.208\.74\.71|23\.253\.126\.58|104\.239\.157\
.210|175\.126\.123\.219|141\.8\.224\.221|10\.10\.10\.10|43\.229\.108\.32|1
03\.232\.215\.140|69\.172\.201\.153|216\.218\.185\.162|54\.225\.104\.146|1
03\.243\.24\.98|199\.59\.243\.120|31\.170\.160\.61|213\.19\.128\.77|62\.11
3\.226\.131|208\.100\.26\.234|195\.16\.127\.102|195\.16\.127\.157|34\.196\
.13\.28|103\.224\.212\.222|172\.217\.4\.225|54\.72\.9\.51|192\.64\.147\.14
1|198\.200\.56\.183|23\.253\.164\.103|52\.48\.191\.26|52\.214\.197\.72|87\
.98\.255\.18|209\.99\.17\.27|216\.38\.62\.18|104\.130\.124\.96|47\.89\.58\
.141|78\.46\.211\.158|54\.86\.225\.156|54\.82\.156\.19|37\.157\.192\.102|2
04\.11\.56\.48|110\.34\.231\.42',ip_address)
    except:
        return 1

    if url_match:
        return 1
    else:
        return 0

```

## **HTML**

**<!DOCTYPE html>**

```
<html lang="en" >

<head>

  <meta charset="UTF-8">

  <title>Classic Login Form Example</title>

  <link href="https://fonts.googleapis.com/css?family=Assistant:400,700" rel="stylesheet">

  <link rel="stylesheet" href="{{ url_for('static', filename='css/login.css') }}">

</head>

<body style="background-image:
url('../static/images/360_F_119115529_mEnw3lGpLdlDkflGRCvSbFRuVl6sMDty.jpg'); ">

<!-- partial:index.partial.html -->

<section class='login' id='login'>

  <div class='head'>

    <h1 class='company'>User Login</h1>

  </div>

  <p class='msg'>Welcome back</p>

  <div class='form'>

    <form>

      <input type="text" placeholder='Username' class='text' id='username' required><br>

      <input type="password" placeholder='.....' class='password'><br>

      <a href="/index" class='btn-login' >Login</a>

      <a href="/register" class='btn-login' >Register</a>

    </form>

  </div>

</section>

</body>

</html>
```

```

<!DOCTYPE html>

<!-- Created By CodingLab - www.codinglabweb.com -->

<html lang="en" dir="ltr">

  <head>

    <meta charset="UTF-8">

    <!--<title> Responsive Registration Form | CodingLab </title>-->

    <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

  </head>

  <body style="background-image: url('../static/images/istockphoto-869283118-612x612.jpg');
  background-repeat: no-repeat; background-size: cover;">

    <div class="container" style="background-color: #68EA9F;">

      <div class="title">Prediction</div>

      <div class="content">

        <form class="contact2-form validate-form" action="{{ url_for('predict')}}" method="post" >

          <div class="user-details">

            <div class="input-box">

              <span class="details">SSLfinal_State</span>

              <input class="input2" type="text" name="SSLfinal_State" required="required"
placeholder="SSLfinal_State"/>

            </div>

            <div class="input-box">

              <span class="details">URL_of_Anchor</span>

              <input class="input2" type="text" name="URL_of_Anchor" required="required"
placeholder="URL_of_Anchor"/>

            </div>

            <div class="input-box">

              <span class="details">Prefix_Suffix</span>

              <input class="input2" type="text" name="Prefix_Suffix" required="required"
placeholder="Prefix_Suffix" />

            </div>


```

```
<div class="input-box">

    <span class="details">web_traffic</span>

    <input class="input2" type="text" name="web_traffic" required="required"
placeholder="Web_Traffic"/>

</div>

<div class="input-box">

    <span class="details">Domain_registration_length</span>

    <input class="input2" type="text" name="Domain_registration_length"
required="required" placeholder="Domain_registration_length"/>

</div>


</div>


<div class="button">

    <input type="submit" value="Predict">

</div>

<br>

<br>

<h4 class="predict">{{ pred }}</h4>

</form>

</div>

</div>

</body></html>
```



## **ADVANTAGES**

- High Accuracy
- High Prediction Rate
- We can prevent the attacks
- High reliability

## **DISADVANTAGES**

- High redundancy
- We can store less amount of data
- Reduced reliability

## **CONCLUSION**

The phishing detection process using our model from the user prospective can be explained in the following steps: The end-user clicks on a link within an email or browses the internet. He will be directed to a website that could be legitimate or phishy. This website is basically the test data. A script written in PHP that is embedded within the browser starts processing to extract the features of the test data (current website) and saves them in a data structure. Now, the intelligent model will be active within the browser to guess the type of the website based on rules learnt from historical websites (previous data collected). The rules of the classifier are utilised to predict the type of the test data based on features similarity. When the browsed website is identified as legitimate no action will be taken. On the other hand, when the website turned to be phishy, the user will be warned by the intelligent method that he is under risk.

## **FUTURE WORK**

- **In future work the web phishing system is implemented with new algorithm**  
High detection efficiency: To provide high detection efficiency, incorrect classification of benign sites as phishing (false-positive) should be

minimal and correct classification of phishing sites (true-positive) should be high.

- Real-time detection: The prediction of the phishing detection approach must be provided before exposing the user's personal information on the phishing website.
- Target independent: Due to the features extracted from both URL and HTML the proposed approach can detect new phishing websites targeting any benign website (zero-day attack).
- Third-party independent: The feature set defined in our work are lightweight and client-side adaptable, which do not rely on third-party services such as blacklist/whitelist, Domain Name System (DNS) records, WHOIS record (domain age), search engine indexing, network traffic measures, etc. Though third-party services may raise the effectiveness of the detection approach, they might misclassify benign websites if a benign website is newly registered. Furthermore, the DNS database and domain age record may be poisoned and lead to false negative results (phishing to benign).

Hence, a light-weight technique is needed for phishing websites detection adaptable at client side. The major contributions in this paper are itemized as follows.

- We propose a phishing detection approach, which extracts efficient features from the URL and HTML of the given webpage without relying on third-party services. Thus, it can be adaptable at the client side and specify better privacy.

- We proposed eight novel features including URL character sequence features (F1), textual content character level (F2), various hyperlink features (F3, F4, F5, F6, F7, and F14) along with seven existing features adopted from the literature.
- We conducted extensive experiments using various machine learning algorithms to measure the efficiency of the proposed features. Evaluation results manifest that the proposed approach precisely identifies the legitimate websites as it has a high true negative rate and very less false positive rate.
- We release a real phishing webpage detection dataset to be used by other researchers on this topic.