

**PROJECT TITLE: WEB PHISHING DETECTION**

**TEAM ID : PNT2022TMID28015**

**Team members**

R.PRAVEEN(Team Leader)

M.HARISH

G.V.MAVEERA

K.VENKATESH

P.PARTHIBAN

# Project Report

1. **INTRODUCTION**
  - 1.1 Project Overview
  - 1.2 Purpose
2. **LITERATURE SURVEY**
  - 2.1 Existing problem
  - 2.2 References
  - 2.3 Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
  - 3.1 Empathy Map Canvas
  - 3.2 Ideation & Brainstorming
  - 3.3 Proposed Solution
  - 3.4 Problem Solution fit
4. **REQUIREMENT ANALYSIS**
  - 4.1 Functional requirement
  - 4.2 Non-Functional requirements
5. **PROJECT DESIGN**
  - 5.1 Data Flow Diagrams
  - 5.2 Solution & Technical Architecture
  - 5.3 User Stories
6. **PROJECT PLANNING & SCHEDULING**
  - 6.1 Sprint Planning & Estimation
  - 6.2 Sprint Delivery Schedule
  - 6.3 Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
  - 7.1 Feature 1
  - 7.2 Feature 2
  - 7.3 Database Schema (if Applicable)
8. **TESTING**
  - 8.1 Test Cases
  - 8.2 User Acceptance Testing
9. **RESULTS**
  - 9.1 Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**
  - Source Code
  - GitHub & Project Demo Link

## 1.INTRODUCTION

### 1.1 Project Overview

# Web Phishing Detection

There are a number of users who purchase products online and make payments through e-banking. There are e-banking websites that ask users to provide sensitive data such as username, password & credit card details, etc., often for malicious reasons. This type of e-banking website is known as a phishing website. Web service is one of the key communications software services for the Internet. Web phishing is one of many security threats to web services on the Internet.

Common threats of web phishing:

- Web phishing aims to steal private information, such as usernames, passwords, and credit card details, by way of impersonating a legitimate entity.
- It will lead to information disclosure and property damage.
- Large organizations may get trapped in different kinds of scams.

This Guided Project mainly focuses on applying a machine-learning algorithm to detect Phishing websites.

In order to detect and predict e-banking phishing websites, we proposed an intelligent, flexible and effective system that is based on using classification algorithms. We implemented classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy. The e-banking phishing website can be detected based on some important characteristics like URL and domain identity, and security and encryption criteria in the final phishing detection rate. Once a user makes a transaction online when he makes payment through an e-banking website our system will use a data mining algorithm to detect whether the e-banking website is a phishing website or not.

### 1.2 Purpose

- You'll be able to understand the problem to classify if it is a regression or a classification kind of problem.
- You will be able to know how to pre-process/clean the data using different data pre-processing techniques.
- Applying different algorithms according to the dataset
- You will be able to know how to find the accuracy of the model.
- You will be able to build web applications using the Flask framework.

## 2.LITERATURE SURVEY

PROJECT TITLE	AUTHOR	OBJECTIVE/OUTCOME
Phishing Detection (2013)	Youssef Iraqi	Many Cyber-attacks are spread via mechanism that exploit weaknesses found in end-users which makes users the weakest element in the security chain.
A survey and classification of web phishing detection schemes (2016)	Manoj Misra	Phishing is a fraudulent technique that is used over the Internet to deceive users with the goal of extracting their personal information such as username, passwords, credit card, and bank account information.
Phishing websites detection using a novel multipurpose dataset and web technologies features. (2019)	Sahingoz et al	Phishing attacks are one of the most challenging social engineering cyberattacks due to the large amount of entities involved in online transactions and services.

### Project flow

- Download the dataset.
- Preprocess or clean the data.
- Analyze the pre-processed data.
- Train the machine with preprocessed data using an appropriate machine learning algorithm.
- Save the model and its dependencies.
- Build a Web application using a flask that integrates with the model built.

### 3.IDEATION AND PROPOSED SOLUTION

#### 3.1 Empathy Map Canvas

##### Ideation Phase Empathize & Discover

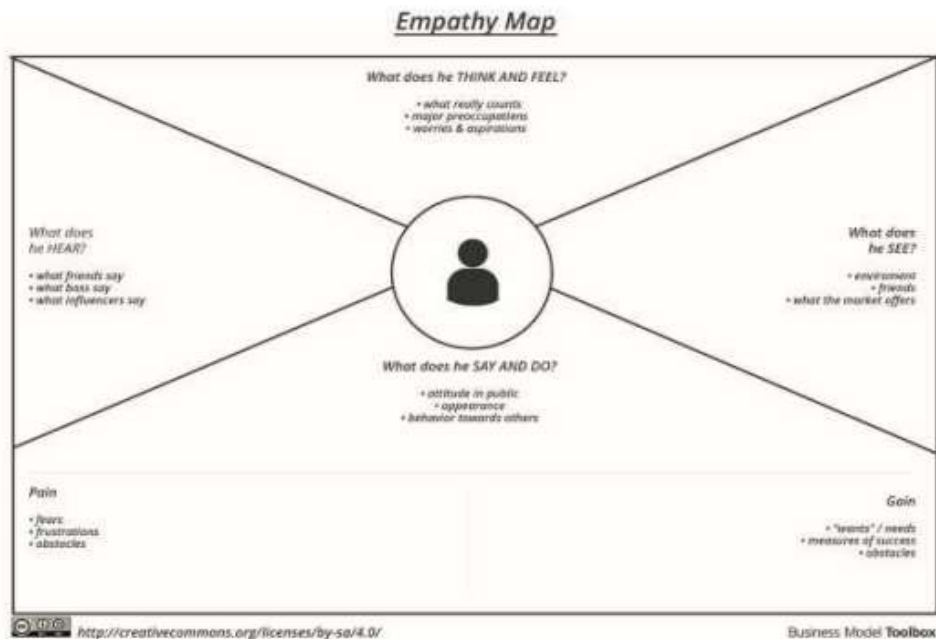
Date	19 September 2022
Team ID	PNT2022TMID28015
Project Name	WEB PHISHING DETECTION
Maximum Marks	4 Marks

##### Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

##### Example:



Reference: <https://www.mural.co/templates/empathy-map-canvas>

## Example: WEB PHISHING DETECTION



### 3.2 Ideation and Brainstorming

#### Ideation Phase Brainstorm & Idea Prioritization Template

Date	19 September 2022
Team ID	PNT2022TMID28015
Project Name	WEB PHISHING DETECTION
Maximum Marks	4 Marks


#### Brainstorm & Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Reference: <https://www.mural.co/templates/empathy-map-canvas>

#### Step-1: Team Gathering, Collaboration and Select the Problem Statement



### Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

🕒 10 minutes to prepare  
🕒 1 hour to collaborate  
👥 3-6 people recommended

**Before you collaborate**  
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

- 1 **Team gathering**  
Define who should participate in the session and send an invite. Share relevant information to get work started.
- 2 **Set the goal**  
Think about the problem you'll be focusing on looking at for brainstorming session.
- 3 **Learn how to use the facilitation tools**  
Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) ➔

**1 Define your problem statement**  
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

🕒 5 minutes

How might we [your problem statement]?

**2 Key rules of brainstorming**  
To run an smooth and productive session:

- 1. Stay on topic.
- 2. Defier judgment.
- 3. Be for volume.
- 4. Encourage wild ideas.
- 5. Listen to others.
- 6. If possible, test ideas.

## Step-2: Brainstorm, Idea Listing and Grouping

### Brainstorm

Write down any ideas that come to mind that address your problem statement.

15 minutes

Use all information to help you create your problem statement and brainstorm ideas.

**PROBLEMS**

1. The company's website is slow and unreliable, leading to lost sales and customer frustration.

2. The company's mobile app is buggy and crashes frequently, leading to a poor user experience.

3. The company's customer support is slow and unhelpful, leading to negative reviews and lost business.

4. The company's marketing campaigns are not reaching the target audience, leading to low conversion rates.

5. The company's product is outdated and lacks features that competitors offer, leading to a loss of market share.

**GOALS**

1. Increase website speed and reliability to reduce bounce rates and improve conversion.

2. Improve mobile app performance and stability to increase user engagement and retention.

3. Enhance customer support response time and quality to improve customer satisfaction and loyalty.

4. Refine marketing strategy and targeting to reach the right audience and increase sales.

5. Develop new product features and update the existing product to stay competitive in the market.

**CONSIDERATIONS**

1. Limited budget for website development and hosting.

2. Limited resources for mobile app development and testing.

3. Limited staff for customer support and training.

4. Limited reach of current marketing channels.

5. Limited time for product development and testing.

**ASSUMPTIONS**

1. The company has a clear understanding of its target audience and their needs.

2. The company has the necessary skills and resources to implement the proposed solutions.

3. The company is committed to providing excellent customer support and service.

4. The company is willing to invest in marketing and advertising to reach its target audience.

5. The company is open to feedback and iteration to improve its products and services.

**QUESTIONS**

1. How can we improve website speed and reliability without increasing costs?

2. How can we improve mobile app performance and stability without sacrificing features?

3. How can we enhance customer support response time and quality without hiring more staff?

4. How can we refine our marketing strategy and targeting without wasting budget?

5. How can we develop new product features and update the existing product without losing sight of our core values?

### Group ideas

Take time clustering your ideas into clusters or related notes as you go. In this step 10 minutes, give each cluster a name that is clearly visible, by and use it to generate.

10 minutes

Use all information to help you create your problem statement and brainstorm ideas.

**Web phishing aims to steal private information, such as usernames, passwords, and credit card details, by way of impersonating a legitimate entity.**

**Phishing attacks are one of the most challenging social engineering cyberattacks due to the large amount of entities involved in online transactions and services.**

**Phishing is a form of fraud in which the attacker tries to learn sensitive information such as login credentials or account information by sending as a reputable entity or person in email or other communication channels.**

100%

Navigation Settings

## Step-3: Idea Prioritization

### Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

25 minutes

Use all information to help you create your problem statement and brainstorm ideas.

**Importance**

If any of these items could get done, it'll put you in a good position to have the most positive impact.

**Web phishing aims to steal private information, such as usernames, passwords, and credit card details, by way of impersonating a legitimate entity.**

**Phishing attacks are one of the most challenging social engineering cyberattacks due to the large amount of entities involved in online transactions and services.**

**Phishing is a form of fraud in which the attacker tries to learn sensitive information such as login credentials or account information by sending as a reputable entity or person in email or other communication channels.**



### 3.3 Proposed Solution

#### **Project Design Phase-I Proposed Solution Template**

<b>Date :</b>	19 September 2022
<b>Team ID :</b>	PNT2022TMID28015
<b>Project Name :</b>	WEB PHISHING DETECTION
<b>Maximum Marks :</b>	2 Marks

#### **Proposed Solution :**

<b>S.No.</b>	<b>Parameter</b>	<b>Description</b>
1.	Problem Statement (Problem to be solved)	A measurement for phishing detection is the number of suspicious e-mails reported to the security team. This measurement is designed to evaluate the number of employees who followed the proper procedure for reporting suspicious messages..
2.	Idea / Solution description	Phishing is described as a fraudulent activity that is done to steal confidential user information such as credit card numbers, login credentials, and passwords. It is usually done by using email or other forms of electronic communication by pretending to be from a reliable business entity.
3.	Novelty / Uniqueness	Phishing is the most widespread and pernicious cyberattack which mostly targets the human rather than the computer by exploiting their vulnerabilities.
4.	Social Impact / Customer Satisfaction	Web spam has a negative impact on the search quality and users' satisfaction and forces search engines to waste resources to crawl, index, and rank it. Thus search engines are compelled to make significant efforts in order to fight web spam.
5.	Business Model (Revenue Model)	The web phishing detection website platform no cost usage.
6.	Scalability of the Solution	length of an URL, URL has HTTP, URL has suspicious character, prefix/suffix, number of dots, number of slashes, URL has phishing term, length of subdomain,

### 3.4 Problem Solution Fit

Project Title:

**Web phishing Detection Project Design Phase-I - Solution Fit Template**

Team ID: PNT2022TMID28015

<b>Define CS, fit into CC</b>	<b>1. CUSTOMER SEGMENT(S)</b> <ul style="list-style-type: none"><li>• Users</li><li>• Business email Compromise</li><li>• Public</li></ul>	<b>6. CUSTOMER CONSTRAINTS</b> <ul style="list-style-type: none"><li>• To prevent spam link.</li><li>• Prevent user data.</li></ul>	<b>5. AVAILABLE SOLUTIONS</b> <ul style="list-style-type: none"><li>• Install firewalls</li><li>• Change our password regularly.</li></ul>	<b>Explore AS, differentiate</b>
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <ul style="list-style-type: none"><li>• Protect your data by backing it up.</li><li>• Protect your device and confidential information.</li></ul>	<b>9. PROBLEM ROOT CAUSE</b> <ul style="list-style-type: none"><li>• Security</li><li>• Prevent unwanted link</li></ul>	<b>7. BEHAVIOUR</b> <p>Phishing detection systems are principally based on the analysis of data moving from phishers to Victims.</p>	
<b>Identify strong</b>	<b>3. TRIGGERS</b> <p>Phishing is a type of social engineering attack often used to steal user data including login credentials and credit card</p>	<b>10. YOUR SOLUTION</b> <ul style="list-style-type: none"><li>• Use anti-Phishing protection and anti-spam Software to protect yourself when malicious messages slip through to your computer.</li></ul>	<b>8. CHANNELS of BEHAVIOUR</b> <ul style="list-style-type: none"><li>• Spear Phishing</li><li>• Whaling</li><li>• Vishing</li></ul>	<b>Extract online &amp; offline</b>
	<b>4. EMOTIONS: BEFORE / AFTER</b> <p>Security purpose to prevent our data.</p>			

## 4.REQUIREMENT ANALYSIS

### 4.1 Functional Requirements

### Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Input	User inputs an URL in required field to check its validation.
FR-2	Website Comparison	Model compares the websites using Blacklist and Whitelist approach.
FR-3	Feature extraction	After comparing, if none found on comparison then it extracts feature using heuristic and visual similarity approach.
FR-4	Prediction	Model predicts the URL using Machine Learning algorithms such as Logistic Regression, KNN
FR-5	Classifier	Model sends all output to classifier and produces final result.
FR-6	Announcement	Model then displays whether website is a legal site or a phishing site.
FR-7	Events	This model needs the capability of retrieving and displaying accurate result for a website

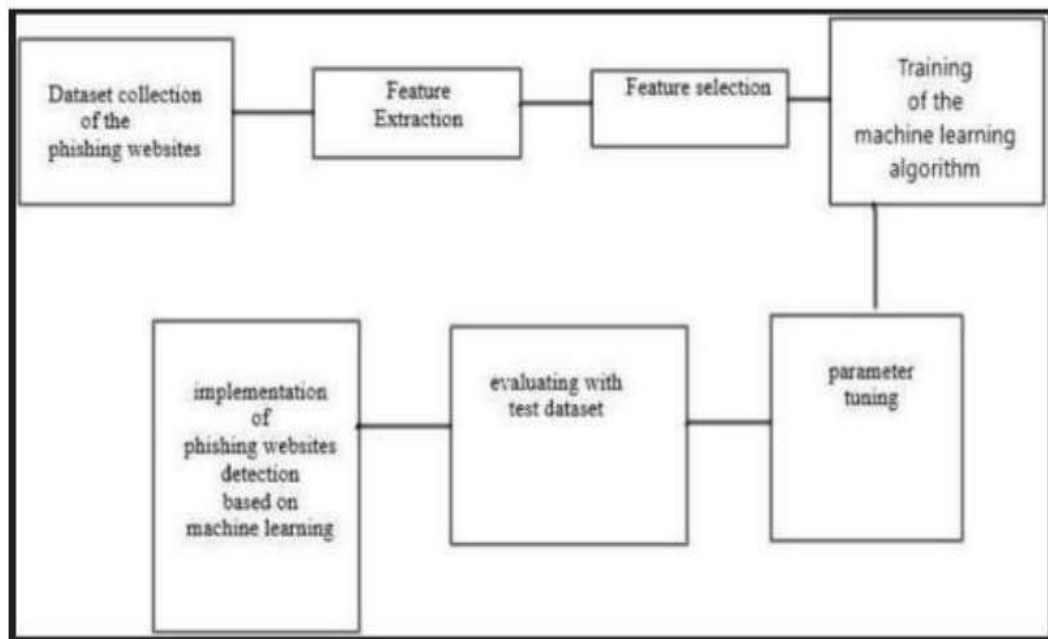
## 4.2 Non Functional Requirements

Following are the non-functional requirements of the proposed solution.

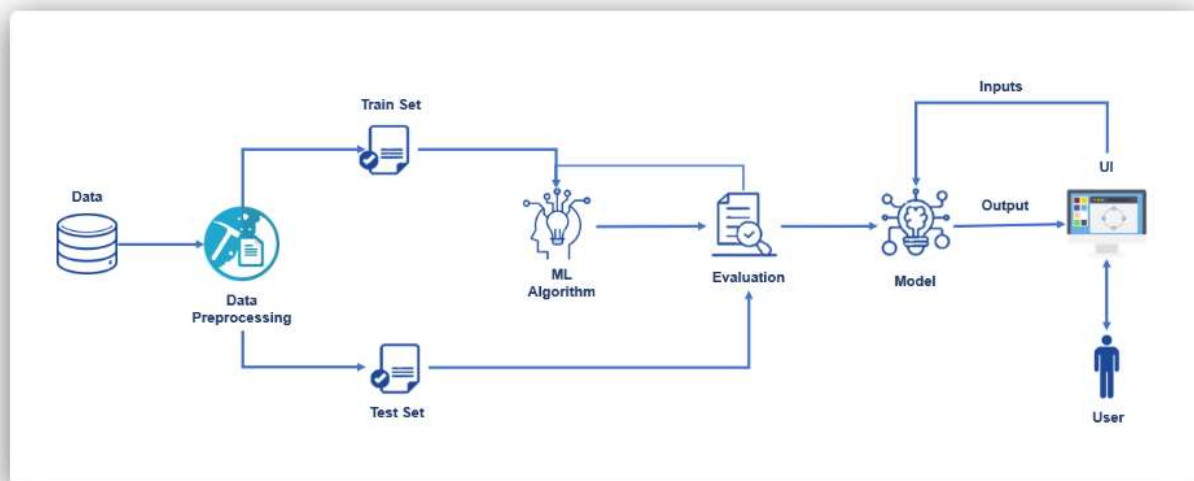
FR No.	Non-Functional Requirement	Description
NFR-1	High detection efficiency	To provide high detection efficiency, incorrect classification of benign sites as phishing (false-positive) should be minimal and correct classification of phishing sites (true-positive) should be high.
NFR-2	Real-time detection	The prediction of the phishing detection approach must be provided before exposing the user's personal information on the phishing website
NFR-3	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).

## 5.PROJECT DESIGN

### 5.1 DATA FLOW DIAGRAM



## 5.2 Solution and Technical Architecture



## 5.3 User Stories

User Story / Task
User inputs an URL in the required field to check its validation.
Model compares the websites using Blacklist and Whitelist approach.
After comparison, if none found on comparison then it extract feature using heuristic and visual similarity.
Model predicts the URL using Machine learning algorithms such as logistic Regression, KNN.
Model sends all the output to the classifier and produces the final result.
Model then displays whether the website is legal site or a phishing site.
This model needs the capability of retrieving and displaying accurate result for a website.

## 6.PROJECT PLANNING AND SCHEDULING

### 6.1 Sprint Planning and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	User input	USN-1	User inputs an URL in the required field to check its validation.	1	Medium
Sprint-1	Website Comparison	USN-2	Model compares the websites using Blacklist and Whitelist approach.	1	High
Sprint-2	Feature Extraction	USN-3	After comparison, if none found on comparison then it extract feature using heuristic and visual similarity.	2	High
Sprint-2	Prediction	USN-4	Model predicts the URL using Machine learning algorithms such as logistic Regression, KNN.	1	Medium
Sprint-3	Classifier	USN-5	Model sends all the output to the classifier and produces the final result.	1	Medium
Sprint-4	Announcement	USN-6	Model then displays whether the website is legal site or a phishing site.	1	High
Sprint-4	Events	USN-7	This model needs the capability of retrieving and displaying accurate result for a website.	1	High

### 6.2 Sprint Delivery Schedule



Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

### 6.3 Reports from JIRA

Velocity: Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day) We have a 6-day sprint duration, and the velocity of the team is 20 (points per sprint). So our team's average velocity (AV) per iteration unit (story points per day)  $AV = (\text{Sprint Duration} / \text{Velocity}) = 20 / 6 = 3.33$

Burndown Chart: A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time

## 7.CODING AND SOLUTIONING

### Phishing Detection

```
if request.method == "POST":
    url = request.form["url"]
    obj = FeatureExtraction(url)
    x = np.array(obj.getFeaturesList()).reshape(1,30)
    y_pred = gbc.predict(x)[0]
```

```
#1 is safe

#-1 is unsafe

y_pro_phishing = gbc.predict_proba(x)[0,0]
y_pro_non_phishing = gbc.predict_proba(x)[0,1]

# if(y_pred ==1 ):

pred = "It is {0:.2f} % safe to go ".format(y_pro_phishing*100)

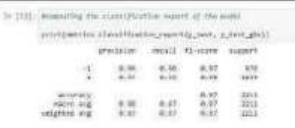

return render_template('index.html',xx =round(y_pro_non_phishing,2),url=url )

return render_template("index.html", xx =-1)
```

## 8.TESTING

### 8.1 Test cases

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	<b>Classification Model:</b> <b>Gradient Boosting Classification</b> Accuracy Score- 97.4%	
2.	Tune the Model	Hyperparameter Tuning - 97% Validation Method – KFOLD & Cross Validation Method	

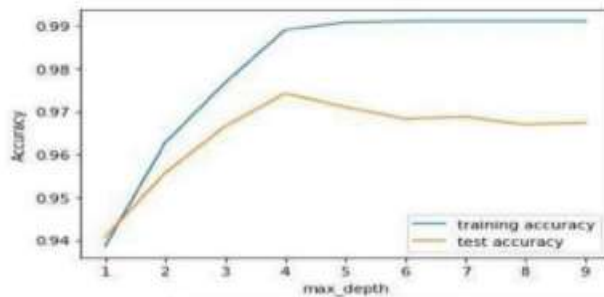
## 1. METRICS: CLASSIFICATION REPORT:

In [52]: `#computing the classification report of the model`

```
print(metrics.classification_report(y_test, y_test_gbc))
```

	precision	recall	f1-score	support
-1	0.99	0.96	0.97	976
1	0.97	0.99	0.98	1235
accuracy			0.97	2211
macro avg	0.98	0.97	0.97	2211
weighted avg	0.97	0.97	0.97	2211

## PERFORMANCE :





Out[83]:

	ML Model	Accuracy	f1_score	Recall	Precision
0	Gradient Boosting Classifier	0.974	0.977	0.994	0.986
1	CatBoost Classifier	0.972	0.975	0.994	0.989
2	Random Forest	0.969	0.972	0.992	0.991
3	Support Vector Machine	0.964	0.968	0.980	0.965
4	Decision Tree	0.958	0.962	0.991	0.993
5	K-Nearest Neighbors	0.956	0.961	0.991	0.989
6	Logistic Regression	0.934	0.941	0.943	0.927
7	Naive Bayes Classifier	0.605	0.454	0.292	0.997
8	XGBoost Classifier	0.548	0.548	0.993	0.984
9	Multi-layer Perceptron	0.543	0.543	0.989	0.983

## 2. TUNE THE MODEL – HYPERPARAMETER TUNING

```
In [58]: #HYPERPARAMETER TUNING
         grid.fit(X_train, y_train)
```

Out[58]:

```
* GridSearchCV
GridSearchCV(cv=5,
             estimator=GradientBoostingClassifier(learning_rate=0.7,
                                                    max_depth=4),
             param_grid={'max_features': array([1, 2, 3, 4, 5]),
                         'n_estimators': array([10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130,
140, 150, 160, 170, 180, 190, 200])})
* estimator: GradientBoostingClassifier
  GradientBoostingClassifier(learning_rate=0.7, max_depth=4)
* GradientBoostingClassifier
  GradientBoostingClassifier(learning_rate=0.7, max_depth=4)
```

```
In [59]: print("The best parameters are %s with a score of %.2f"
              % (grid.best_params_, grid.best_score_))

The best parameters are {'max_features': 5, 'n_estimators': 200} with a score of 0.97
```

## VALIDATION METHODS: KFOLD & Cross Folding

### Wilcoxon signed-rank test

```
In [78]: #KFOLD and Cross Validation Model

from scipy.stats import wilcoxon
from sklearn.datasets import load_iris
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import cross_val_score, KFold

# Load the dataset
X = load_iris().data
y = load_iris().target

# Prepare models and select your CV method
model1 = GradientBoostingClassifier(n_estimators=100)
model2 = XGBClassifier(n_estimators=100)
kf = KFold(n_splits=20, random_state=None)
# Extract results for each model on the same folds
results_model1 = cross_val_score(model1, X, y, cv=kf)
results_model2 = cross_val_score(model2, X, y, cv=kf)
stat, p = wilcoxon(results_model1, results_model2, zero_method='zsplit');
stat
```

Out[78]: 95.0

### 5x2CV combined F test

```
In [89]: from mlxtend.evaluate import combined_ftest_5x2cv
from sklearn.tree import DecisionTreeClassifier, ExtraTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from mlxtend.data import iris_data

# Prepare data and clfs
X, y = iris_data()
clf1 = GradientBoostingClassifier()
clf2 = DecisionTreeClassifier()

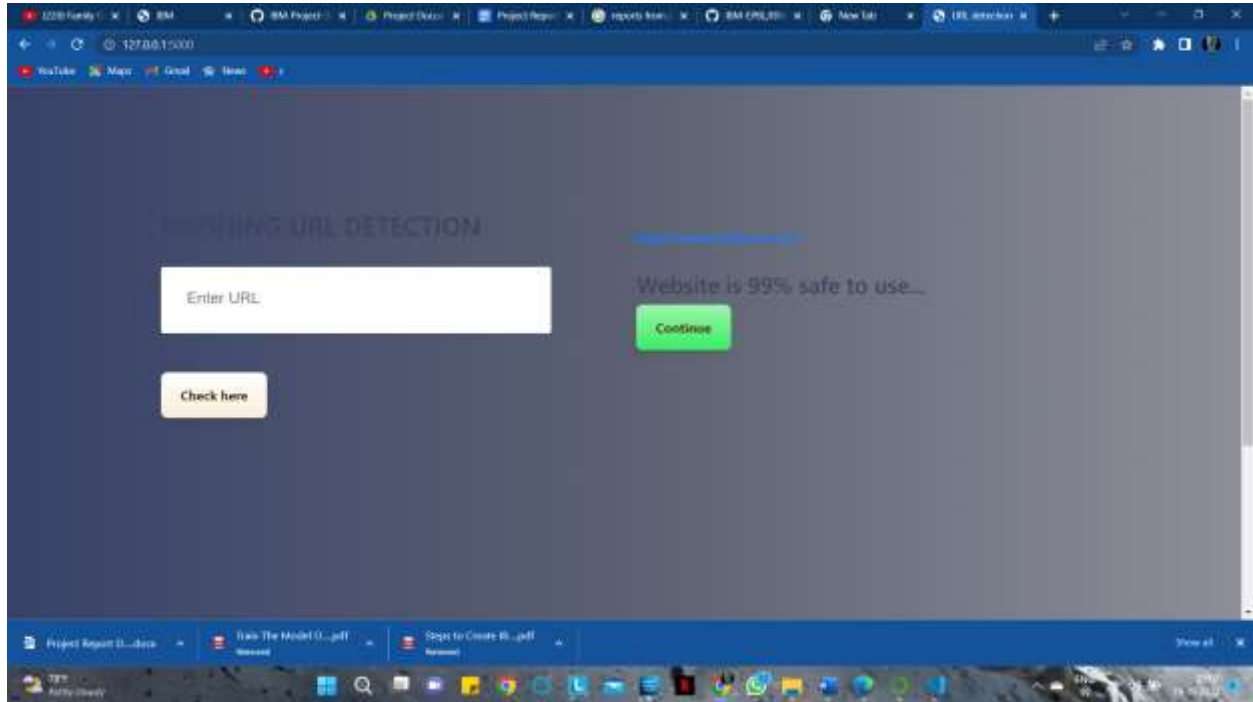
# Calculate p-value
f, p = combined_ftest_5x2cv(estimator1=clf1,
                             estimator2=clf2,
                             X=X, y=y,
                             random_seed=1)

print('f-value:', f)
print('p-value:', p)

f-value: 1.727272727272733
p-value: 0.2840135734291782
```

## 9.RESULTS.

### 9.1 PERFORMANCE METRICES.



## 10.ADVANTAGES AND DISADVANTAGES.

### ADVANTAGES

#### *1. Improve on Inefficiencies of SEG and Phishing Awareness Training*

Secure email gateway's (SEG's) and phishing awareness training remain a critical tool in the fight against phishing and malware.

As increasingly-sophisticated phishing attacks, such as BEC, become more difficult to detect, even by trained security personnel. Thus there is an urgent need for the channel to provide customers with technology that not only strives to prevent intrusion, but can also help users after an attack has passed through the secure email gateway.

A mailbox-level anti-phishing solution offers an additional layer of protection by analyzing account information and understanding users' communication habits. This delivers an enhanced level of phishing protection to detect attacks faster, alert users and remediate threats as quickly as possible. Machine learning scores sender reputation enabling a baseline for what "normal

communications” with a user should look like. It can then compare correspondence and incoming messages with multiple data points to identify and learn from anomalies.

## *2. It Takes a Load off the Security Team*

Customers now have many tools on the market to enhance their email security. The best of these use artificial intelligence and machine learning to better identify some of the suspected threats. This not only improves security, but can significantly reduce the workloads of IT and security teams. According to a **survey by Fidelis Cybersecurity**, less than one in five organizations have a dedicated threat hunting team, and only half of those could handle more than eight investigations per day.

Security teams need all the help they can get, and must look beyond human intelligence to additional aids that protect the integrity of their corporate information. Automation can improve efficiency by flagging and analyzing the growing number of investigations, and by filtering out false positives. An automation detection and mitigation system reduces response time, identifies threats and can classify and remediate them with one click. It also delivers better metrics and intelligence with real-time insight into the strategies employed by the latest threats. Pushing these duties to an innovative solution not only increases security, but also enables security teams to focus more on policy and prevention.

## *3. It Offers a Solution, Not a Tool*

The goal of security isn’t solely to bring tools to the table, it is to offer solutions. resellers ultimately need to ensure that solutions work for the organization, and that calls for listening to customers, understanding the channels, the issues and how they are impacting the organization. Whereas basic tools simply offer information and basic applications, automated and advanced phishing threat protection solutions can help solve the challenges that customers face. This ultimately helps the channel discuss solutions with their customers and offer an overall picture and architecture of solving the challenges that are now inherent to email security. Automated advanced phishing threat protection defends against today’s and tomorrow’s threats with a system that continually learns and makes the entire organization more security conscious and aware.

## *4. Separate You from Your Competitors*

As the channel is sometimes slow to move to some of the more advanced technologies, those that are fast will gain the upper hand and a competitive advantage. Merely having these conversations will automatically distinguish you from many of your competitors. It demonstrates that you have a pulse on the latest threats and an insight into how artificial intelligence and machine learning can improve the security posture, without adding more burden to them.

## DISADVANTAGES

Techniques	False positives	Zero day attacks	Fake interface attack	Slow response time
Blacklist	No	Yes	No	No
Heuristics	Yes	Maybe	No	Maybe
User polling	Yes	Yes	Yes	Maybe
Third-party certification	No	No	Yes	Maybe

## 11.CONCLUSION

This is the concluding chapter of this study, which implies that it includes a recap of all the previous chapters. First, a concluding remark is given discussing the importance of this study in mitigating the risk incurred by online users to phishing websites. Furthermore, the objectives of the study are summarized in phases of our research methodology discussed . Second, our contribution to this research area is analyzed and discussed in different stages of implementation. However, because of the broadness of the research area, we discussed the scope of our research. to a time-feasible scope. Furthermore, during the course of the study, we realized prospective field of study to expand on our research in the future. This brings us to the concluding section of our work, which discusses our recommendation and future research area that can improve website phishing detection and mitigation. Finally, our closing remark that includes a brief on the problem and the effectiveness of our solution approach is given.

## 12. FUTURE SCOPE

In future if we get structured dataset of phishing we can perform phishing detection much more faster than any other technique. In future we can use a combination of any other two or more classifier to get maximum accuracy. We also plan to explore various phishing techniques that use Lexical features, Network based features, Content based features, Webpage based features and HTML and JavaScript features of web pages which can improve the performance of the system. In particular, we extract features from URLs and pass it through the various classifiers.

## 13 APPENDIX.

### SOURCE CODE

#### App.py

```
from flask import Flask, request, render_template

import numpy as np

import pandas as pd

from sklearn import metrics

import warnings

import pickle

warnings.filterwarnings('ignore')

from feature import FeatureExtraction
```

```
file = open("pickle/model.pkl","rb")
```

```
gbc = pickle.load(file)
```

```
file.close()
```

```
app = Flask(__name__)
```

```
@app.route("/", methods=["GET", "POST"])
```

```
def index():
```

```
    if request.method == "POST":
```

```
        url = request.form["url"]
```

```
        obj = FeatureExtraction(url)
```

```
        x = np.array(obj.getFeaturesList()).reshape(1,30)
```

```
        y_pred =gbc.predict(x)[0]
```

```
        #1 is safe
```

```
        #-1 is unsafe
```

```
        y_pro_phishing = gbc.predict_proba(x)[0,0]
```

```
        y_pro_non_phishing = gbc.predict_proba(x)[0,1]
```

```
        # if(y_pred ==1 ):
```

```
            pred = "It is {0:.2f} % safe to go ".format(y_pro_phishing*100)
```

```
            return render_template('index.html',xx =round(y_pro_non_phishing,2),url=url )
```

```
        return render_template("index.html", xx =-1)
```

```
if __name__ == "__main__":  
    app.run(debug=True)
```

### **Feature.py**

```
import ipaddress  
  
import re  
  
import urllib.request  
  
from bs4 import BeautifulSoup  
  
import socket  
  
import requests  
  
from googlesearch import search  
  
import whois  
  
from datetime import date, datetime  
  
import time  
  
from dateutil.parser import parse as date_parse  
  
from urllib.parse import urlparse  
  
  
class FeatureExtraction:  
  
    features = []  
  
    def __init__(self,url):  
  
        self.features = []
```

```
self.url = url

self.domain = ""

self.whois_response = ""

self.urlparse = ""

self.response = ""

self.soup = ""


try:

    self.response = requests.get(url)

    self.soup = BeautifulSoup(response.text, 'html.parser')

except:

    pass


try:

    self.urlparse = urlparse(url)

    self.domain = self.urlparse.netloc

except:

    pass


try:

    self.whois_response = whois.whois(self.domain)

except:

    pass
```



```
self.features.append(self.UsingIp())
```

```
self.features.append(self.longUrl())
```

```
self.features.append(self.shortUrl())
```

```
self.features.append(self.symbol())
```

```
self.features.append(self.redirecting())
```

```
self.features.append(self.prefixSuffix())
```

```
self.features.append(self.SubDomains())
```

```
self.features.append(self.Hppts())
```

```
self.features.append(self.DomainRegLen())
```

```
self.features.append(self.Favicon())
```

```
self.features.append(self.NonStdPort())
```

```
self.features.append(self.HTTPSDomainURL())
```

```
self.features.append(self.RequestURL())
```

```
self.features.append(self.AnchorURL())
```

```
self.features.append(self.LinksInScriptTags())
```

```
self.features.append(self.ServerFormHandler())
```

```
self.features.append(self.InfoEmail())
```

```
self.features.append(self.AbnormalURL())  
self.features.append(self.WebsiteForwarding())  
self.features.append(self.StatusBarCust())  
  
self.features.append(self.DisableRightClick())  
self.features.append(self.UsingPopupWindow())  
self.features.append(self.IframeRedirection())  
self.features.append(self.AgeofDomain())  
self.features.append(self.DNSRecording())  
self.features.append(self.WebsiteTraffic())  
self.features.append(self.PageRank())  
self.features.append(self.GoogleIndex())  
self.features.append(self.LinksPointingToPage())  
self.features.append(self.StatsReport())
```

# 1.UsingIp

```
def UsingIp(self):  
    try:  
        ipaddress.ip_address(self.url)  
        return -1  
    except:  
        return 1
```

```
# 2.longUrl
```

```
def longUrl(self):
```

```
    if len(self.url) < 54:
```

```
        return 1
```

```
    if len(self.url) >= 54 and len(self.url) <= 75:
```

```
        return 0
```

```
    return -1
```

```
# 3.shortUrl
```

```
def shortUrl(self):
```

```
    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\.ac|su\.pr|twurl\.nl|snipurl\.com|'
```

```
'short\.to|BudURL\.com|ping\.fm|post\.ly|Just\.as|bkite\.com|snipr\.com|fic\.kr|loopt\.us|'
```

```
        'doiop\.com|short\.ie|kl\.am|wp\.me|rubyurl\.com|om\.ly|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', self.url)
```

```
    if match:
```

```
        return -1
```

```
    return 1
```

# 4.Symbol@

def symbol(self):

if re.findall("@",self.url):

return -1

return 1

# 5.Redirecting//

def redirecting(self):

if self.url.rfind('/')>6:

return -1

return 1

# 6.prefixSuffix

def prefixSuffix(self):

try:

match = re.findall("-", self.domain)

if match:

return -1

return 1

except:

return -1

# 7.SubDomains

def SubDomains(self):

dot\_count = len(re.findall("\.", self.url))

if dot\_count == 1:

return 1

elif dot\_count == 2:

return 0

return -1

# 8.HTTPS

def Hppts(self):

try:

https = self.urlparse.scheme

if 'https' in https:

return 1

return -1

except:

return 1

# 9.DomainRegLen

def DomainRegLen(self):

try:

expiration\_date = self.whois\_response.expiration\_date

```

        creation_date = self.whois_response.creation_date

    try:
        if(len(expiration_date)):
            expiration_date = expiration_date[0]
    except:
        pass

    try:
        if(len(creation_date)):
            creation_date = creation_date[0]
    except:
        pass

    age = (expiration_date.year-creation_date.year)*12+ (expiration_date.month-
creation_date.month)

    if age >=12:
        return 1
    return -1
except:
    return -1

# 10. Favicon

def Favicon(self):
    try:
        for head in self.soup.find_all('head'):

```

```
        for head.link in self.soup.find_all('link', href=True):

            dots = [x.start(0) for x in re.finditer('\.', head.link['href'])]

            if self.url in head.link['href'] or len(dots) == 1 or domain in head.link['href']:

                return 1

        return -1

    except:

        return -1
```

# 11. NonStdPort

```
def NonStdPort(self):

    try:

        port = self.domain.split(":")

        if len(port)>1:

            return -1

        return 1

    except:

        return -1
```

# 12. HTTPSDomainURL

```
def HTTPSDomainURL(self):

    try:

        if 'https' in self.domain:

            return -1
```

```
        return 1

    except:

        return -1
```

# 13. RequestURL

```
def RequestURL(self):
```

```
    try:
```

```
        for img in self.soup.find_all('img', src=True):
```

```
            dots = [x.start(0) for x in re.finditer('\.', img['src'])]
```

```
            if self.url in img['src'] or self.domain in img['src'] or len(dots) == 1:
```

```
                success = success + 1
```

```
            i = i+1
```

```
        for audio in self.soup.find_all('audio', src=True):
```

```
            dots = [x.start(0) for x in re.finditer('\.', audio['src'])]
```

```
            if self.url in audio['src'] or self.domain in audio['src'] or len(dots) == 1:
```

```
                success = success + 1
```

```
            i = i+1
```

```
        for embed in self.soup.find_all('embed', src=True):
```

```
            dots = [x.start(0) for x in re.finditer('\.', embed['src'])]
```

```
            if self.url in embed['src'] or self.domain in embed['src'] or len(dots) == 1:
```

```
                success = success + 1
```



```
i = i+1
```

```
for iframe in self.soup.find_all('iframe', src=True):
```

```
    dots = [x.start(0) for x in re.finditer('\.', iframe['src'])]
```

```
    if self.url in iframe['src'] or self.domain in iframe['src'] or len(dots) == 1:
```

```
        success = success + 1
```

```
    i = i+1
```

```
try:
```

```
    percentage = success/float(i) * 100
```

```
    if percentage < 22.0:
```

```
        return 1
```

```
    elif((percentage >= 22.0) and (percentage < 61.0)):
```

```
        return 0
```

```
    else:
```

```
        return -1
```

```
except:
```

```
    return 0
```

```
except:
```

```
    return -1
```

```
# 14. AnchorURL
```

```
def AnchorURL(self):
```

```
try:
```

```
    i,unsafe = 0,0
```

```
    for a in self.soup.find_all('a', href=True):
```

```
        if "#" in a['href'] or "javascript" in a['href'].lower() or "mailto" in a['href'].lower() or not  
(url in a['href'] or self.domain in a['href']):
```

```
            unsafe = unsafe + 1
```

```
    i = i + 1
```

```
try:
```

```
    percentage = unsafe / float(i) * 100
```

```
    if percentage < 31.0:
```

```
        return 1
```

```
    elif ((percentage >= 31.0) and (percentage < 67.0)):
```

```
        return 0
```

```
    else:
```

```
        return -1
```

```
except:
```

```
    return -1
```

```
except:
```

```
    return -1
```

```
# 15. LinksInScriptTags
```

```
def LinksInScriptTags(self):
```

try:

i,success = 0,0

for link in self.soup.find\_all('link', href=True):

dots = [x.start(0) for x in re.finditer('\.', link['href'])]

if self.url in link['href'] or self.domain in link['href'] or len(dots) == 1:

success = success + 1

i = i+1

for script in self.soup.find\_all('script', src=True):

dots = [x.start(0) for x in re.finditer('\.', script['src'])]

if self.url in script['src'] or self.domain in script['src'] or len(dots) == 1:

success = success + 1

i = i+1

try:

percentage = success / float(i) \* 100

if percentage < 17.0:

return 1

elif((percentage >= 17.0) and (percentage < 81.0)):

return 0

else:

return -1

```

        except:

            return 0

    except:

        return -1

# 16. ServerFormHandler

def ServerFormHandler(self):

    try:

        if len(self.soup.find_all('form', action=True))==0:

            return 1

        else :

            for form in self.soup.find_all('form', action=True):

                if form['action'] == "" or form['action'] == "about:blank":

                    return -1

                elif self.url not in form['action'] and self.domain not in form['action']:

                    return 0

                else:

                    return 1

    except:

        return -1

# 17. InfoEmail

def InfoEmail(self):

```

```
try:

    if re.findall(r"[mail\(\)|mailto:?}", self.soap):

        return -1

    else:

        return 1

except:

    return -1
```

# 18. AbnormalURL

```
def AbnormalURL(self):

    try:

        if self.response.text == self.whois_response:

            return 1

        else:

            return -1

    except:

        return -1
```

# 19. WebsiteForwarding

```
def WebsiteForwarding(self):

    try:

        if len(self.response.history) <= 1:

            return 1
```

```
elif len(self.response.history) <= 4:  
    return 0  
  
else:  
    return -1  
  
except:  
    return -1
```

# 20. StatusBarCust

```
def StatusBarCust(self):  
    try:  
        if re.findall("<script>.+onmouseover.+</script>", self.response.text):  
            return 1  
    except:  
        return -1
```

# 21. DisableRightClick

```
def DisableRightClick(self):  
    try:  
        if re.findall(r"event.button ?== ?2", self.response.text):  
            return 1  
    except:
```

```
        return -1
```

```
    except:
```

```
        return -1
```

```
# 22. UsingPopupWindow
```

```
def UsingPopupWindow(self):
```

```
    try:
```

```
        if re.findall(r"alert\(", self.response.text):
```

```
            return 1
```

```
        else:
```

```
            return -1
```

```
    except:
```

```
        return -1
```

```
# 23. IframeRedirection
```

```
def IframeRedirection(self):
```

```
    try:
```

```
        if re.findall(r"<iframe>|<frameBorder>", self.response.text):
```

```
            return 1
```

```
        else:
```

```
            return -1
```

```
    except:
```

```
        return -1
```

# 24. AgeofDomain

```
def AgeofDomain(self):
```

```
    try:
```

```
        creation_date = self.whois_response.creation_date
```

```
    try:
```

```
        if(len(creation_date)):
```

```
            creation_date = creation_date[0]
```

```
    except:
```

```
        pass
```

```
    today = date.today()
```

```
    age = (today.year-creation_date.year)*12+(today.month-creation_date.month)
```

```
    if age >=6:
```

```
        return 1
```

```
    return -1
```

```
    except:
```

```
        return -1
```

# 25. DNSRecording

```
def DNSRecording(self):
```

```
    try:
```

```
        creation_date = self.whois_response.creation_date
```



```
try:
```

```
    if(len(creation_date)):
```

```
        creation_date = creation_date[0]
```

```
except:
```

```
    pass
```

```
today = date.today()
```

```
age = (today.year-creation_date.year)*12+(today.month-creation_date.month)
```

```
if age >=6:
```

```
    return 1
```

```
return -1
```

```
except:
```

```
    return -1
```

```
# 26. WebsiteTraffic
```

```
def WebsiteTraffic(self):
```

```
    try:
```

```
        rank =
```

```
BeautifulSoup(urllib.request.urlopen("http://data.alexa.com/data?cli=10&dat=s&url=" +  
url).read(), "xml").find("REACH")["RANK"]
```

```
        if (int(rank) < 100000):
```

```
            return 1
```

```
return 0
```

```
except :
```

```
return -1
```

## # 27. PageRank

```
def PageRank(self):  
  
    try:  
  
        prank_checker_response = requests.post("https://www.checkpagerank.net/index.php",  
        {"name": self.domain})  
  
        global_rank = int(re.findall(r"Global Rank: ([0-9]+)", rank_checker_response.text)[0])  
  
        if global_rank > 0 and global_rank < 100000:  
  
            return 1  
  
            return -1  
  
    except:  
  
        return -1
```

## # 28. GoogleIndex

```
def GoogleIndex(self):  
  
    try:  
  
        site = search(self.url, 5)  
  
        if site:  
  
            return 1  
  
        else:  
  
            return -1
```

```
except:
```

```
    return 1
```

```
# 29. LinksPointingToPage
```

```
def LinksPointingToPage(self):
```

```
    try:
```

```
        number_of_links = len(re.findall(r"<a href=", self.response.text))
```

```
        if number_of_links == 0:
```

```
            return 1
```

```
        elif number_of_links <= 2:
```

```
            return 0
```

```
        else:
```

```
            return -1
```

```
    except:
```

```
        return -1
```

```
# 30. StatsReport
```

```
def StatsReport(self):
```

```
    try:
```

```
        url_match = re.search(
```

```
'at\.ua|usa\.cc|baltazarpresentes\.com\.br|pe\.hu|esy\.es|hol\.es|sweddy\.com|myjino\.ru|96\.lt|ow\.l  
y', url)
```

```
        ip_address = socket.gethostbyname(self.domain)
```

```
        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|107\  
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|103\.232\.215\.140|69\.172\.201\.153|'
```

```
'216\.218\.185\.162|54\.225\.104\.146|103\.243\.24\.98|199\.59\.243\.120|31\.170\.160\.61|213\.1  
9\.128\.77|62\.113\.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\.141|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|204\.11\.56\.48|110\.34\.231\.42', ip_address)
```

```
        if url_match:
```

```
            return -1
```

```
        elif ip_match:
```

```
            return -1
```

```
        return 1
```

```
    except:
```

```
        return 1
```

```
    def getFeaturesList(self):
```

```
        return self.features
```

## Index.html

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
  <meta charset="UTF-8">
```

```
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

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

```
  <meta name="description" content="This website is develop for identify the safety of url.">
```

```
  <meta name="keywords" content="phishing url,phishing,cyber security,machine  
learning,classifier,python">
```

```
  <meta name="author" content="VAIBHAV BICHAVE">
```

```
<!-- BootStrap -->
```

```
  <link rel="stylesheet"  
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
```

```
    integrity="sha384-  
9aIt2nRpC12Uk9gS9baDI411NQApFmC26EwAOH8WgZl5MYYYxFfc+NcPb1dKGj7Sk"  
crossorigin="anonymous">
```

```
  <link href="static/styles.css" rel="stylesheet">
```

```
  <title>URL detection</title>
```

```
</head>
```

```
<body>
```

```
<div class=" container">
```

```
<div class="row">
```

```
<div class="form col-md" id="form1">
```

```
<h2>PHISHING URL DETECTION</h2>
```

```
<br>
```

```
<form action="/" method ="post">
```

```
<input type="text" class="form__input" name ='url' id="url" placeholder="Enter URL"
required="" />
```

```
<label for="url" class="form__label"> Enter URL</label>
```

```
<button class="button" role="button" >Check here</button>
```

```
</form>
```

```
</div>
```

```
<div class="col-md" id="form2">
```

```
<br>
```

```
<h6 class = "right "><a href= {{ url }} target="_blank">{{ url }}</a></h6>
```

```
<br>
```

```
<h3 id="prediction"></h3>
```

<button class="button2" id="button2" role="button" onclick="window.open('{{url}}')" target="\_blank">Still want to Continue</button>

<button class="button1" id="button1" role="button" onclick="window.open('{{url}}')" target="\_blank">Continue</button>

</div>

</div>

<br>

<h1></h1>

</div>

<!-- JavaScript -->

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"

integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"

crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"

integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"

crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"

integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI"

crossorigin="anonymous"></script>

```
<script>
```

```
let x = '{{xx}}';
```

```
let num = x*100;
```

```
if (0<=x && x<0.50){
```

```
    num = 100-num;
```

```
}
```

```
let txtx = num.toString();
```

```
if(x<=1 && x>=0.50){
```

```
    var label = "Website is "+txtx +"% safe to use...";
```

```
    document.getElementById("prediction").innerHTML = label;
```

```
    document.getElementById("button1").style.display="block";
```

```
}
```

```
else if (0<=x && x<0.50){
```

```
    var label = "Website is "+txtx +"% unsafe to use..."
```

```
    document.getElementById("prediction").innerHTML = label ;
```

```
    document.getElementById("button2").style.display="block";
```

```
}
```

```
</script>
```

```
</body>
```



```
</html>
```

## **Style.css**

```
,
```

```
*::after,
```

```
*::before {
```

```
margin: 0;
```

```
padding: 0;
```

```
box-sizing: inherit;
```

```
font-size: 62,5%;
```

```
}
```

```
body {
```

```
padding: 10% 5%;
```

```
background: #494D5F;
```

```
background: linear-gradient(to right,#384269, #5b6172, #8d8e99);
```

```
justify-content: center;
```

```
align-items: center;
```

```
height: 100vh;
```

```
color: rgb(63, 67, 92);
```

```
}
```

```
.form__label {
```

```
font-family: 'Roboto', sans-serif;
```

```
font-size: 1.2rem;

margin-left: 2rem;

margin-top: 0.7rem;

display: block;

transition: all 0.3s;

transform: translateY(0rem);
}

.form__input {

top: -24px;

font-family: 'Roboto', sans-serif;

color: #333;

font-size: 1.2rem;

padding: 1.5rem 2rem;

border-radius: 0.2rem;

background-color: rgb(255, 255, 255);

border: none;

width: 75%;

display: block;

border-bottom: 0.3rem solid transparent;

transition: all 0.3s;
}
```

```
.form__input:placeholder-shown + .form__label {  
  
  opacity: 0;  
  
  visibility: hidden;  
  
  -webkit-transform: translateY(+4rem);  
  
  transform: translateY(+4rem);  
  
}
```

```
.button {  
  
  appearance: button;  
  
  background-color: transparent;  
  
  background-image: linear-gradient(to bottom, #fff, #f8eedb);  
  
  border: 0 solid #e5e7eb;  
  
  border-radius: .5rem;  
  
  box-sizing: border-box;  
  
  color: #482307;  
  
  column-gap: 1rem;  
  
  cursor: pointer;  
  
  display: flex;  
  
  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica  
Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI  
Symbol","Noto Color Emoji";  
  
  font-size: 100%;  
  
  font-weight: 700;
```

```
line-height: 24px;

margin: 0;

outline: 2px solid transparent;

padding: 1rem 1.5rem;

text-align: center;

text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1);

user-select: none;

-webkit-user-select: none;

touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);
}
```

```
.button:active {

background-color: #f3f4f6;

box-shadow: -1px 2px 5px rgba(81,41,10,0.15),0px 1px 1px rgba(81,41,10,0.15);

transform: translateY(0.125rem);
}
```

```
.button:focus {

box-shadow: rgba(72, 35, 7, .46) 0 0 0 4px, -6px 8px 10px rgba(81,41,10,0.1), 0px 2px 2px
rgba(81,41,10,0.2);
}
```

```
.main-body{  
  
  display: flex;  
  
  flex-direction: row;  
  
  width: 75%;  
  
  justify-content:space-around;  
  
}
```

```
.button1{  
  
  appearance: button;  
  
  background-color: transparent;  
  
  background-image: linear-gradient(to bottom, rgb(160, 245, 174), #37ee65);  
  
  border: 0 solid #e5e7eb;  
  
  border-radius: .5rem;  
  
  box-sizing: border-box;  
  
  color: #482307;  
  
  column-gap: 1rem;  
  
  cursor: pointer;  
  
  display: flex;  
  
  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica  
Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI  
Symbol","Noto Color Emoji";  
  
  font-size: 100%;  
  
  font-weight: 700;
```

```
line-height: 24px;

margin: 0;

outline: 2px solid transparent;

padding: 1rem 1.5rem;

text-align: center;

text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1);

user-select: none;

-webkit-user-select: none;

touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);

display: none;

}
```

```
.button2{

appearance: button;

background-color: transparent;

background-image: linear-gradient(to bottom, rgb(252, 162, 162), #ee3737);

border: 0 solid #e5e7eb;

border-radius: .5rem;

box-sizing: border-box;

color: #482307;

column-gap: 1rem;
```

```
cursor: pointer;

display: flex;

font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";

font-size: 100%;

font-weight: 700;

line-height: 24px;

margin: 0;

outline: 2px solid transparent;

padding: 1rem 1.5rem;

text-align: center;

text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1);

user-select: none;

-webkit-user-select: none;

touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);

display: none;

}
```

```
.right {

right: 0px;

width: 300px;
```

```
}
```

```
@media (max-width: 576px) {
```

```
  .form {
```

```
    width: 100%;
```

```
  }
```

```
}
```

```
.abc{
```

```
  width: 50%;
```

```
}
```

**Github Link:**

[IBM-EPBL/IBM-Project-30630-1660151305](#)

**PROJECT DEMO LINK:**

[\*\*https://drive.google.com/file/d/1w\\_cckznfW1HGM55w4uftbC7W1h5tvtk2/view?usp=sharing\*\*](https://drive.google.com/file/d/1w_cckznfW1HGM55w4uftbC7W1h5tvtk2/view?usp=sharing)