CAPE INSTITUTE OF

TECHNOLOGY

LEVENGIPURAM

IBM PROJECT REPORT

WEB PHISHING DETECTION

Team ID:PNT2022TMID3422

Team Members

Aishwarya.V

Amisha.G

Nanthini.K

Ponvijaya.P

<u>TABLE OF CONTENTS</u>

ABSTRACT

Phishing is the most commonly used social engineering and cyber attack. Through such attacks, the phisher targets naive online users by tricking them into revealing confidential information, with the purpose of using it fraudulently. In order to avoid getting phished, Users should haveawareness of phishing websites. Have a blacklist of phishing websites which requires the knowledge of website being detected as phishing.
Detect them intheir early appearance, using machine learning and deep neuralnetwork algorithms. Of the above three, the machine learningbased method is provento be most effective than the other methods. A phishing website is a commonsocial engineering method that mimics trustful uniformresource locators (URLs) and webpages. The objective of this project is to train machine learning models anddeep neural nets on the dataset created to predict phishing websites. Both phishing and benign URLs of websites are gathered to form a dataset and from them required URL and website content-based features are extracted. The performance level of each model is measured and compared.

Keywords: Deep learning, Machine learning,Phishing website attack,Phishing website detection, Anti-phishing website, Legitimate website , Phishingwebsite datasets, Phishingwebsite features.

1.INTRODUCTION

a.PROJECT OVERVIEW

There are a number of users who purchase products online and make payments through e-banking. There are ebanking 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 are

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

b. It will lead to information disclosure and property damage.

c. Large organizations may get trapped in different kinds of scams.

d. 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.

b. PURPOSE

As technology is growing, phishing methods have started to progress briskly and this should be avoided by making use of anti-phishing techniques to detect phishing. Machine learning is a authoritative tool that can be used to aim against phishing assaults. There are several methods or approaches to identify phishing websites.

The purpose of Phishing Domain Detection is **detecting phishing domain names**. Therefore, passive queries related to the domain name, which we want to classify as phishing or not, provide useful information to us.

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.LITERATURE SURVEY

a. EXISTING PROBLEM

Phishing attacks are one of the most common security challenges that both individuals and companies face in keeping their information secure. Whether it's getting access to passwords, credit cards, or other sensitive information, hackers are using email, social media, phone calls, and any form of communication they can to steal valuable data. Businesses, of course, are a particularly worthwhile target.

COMMON TYPES OF PHISHING ATTACKS AGAINST BUSINESSES

Company Impersonation

One of the most common forms of phishing is where attackers impersonate your brand. This is typically done with an email connected to a domain very similar to the target company (e.g., “first.name@amazon-support”). It’s also a difficult attack for companies to look out for due to the fact that you won’t know until someone falls for it or alerts you.

Spear phishing

This type of scheme involves using a fake company name (impersonation) but also key details about the target. Muck like in sales, a rep finds the name, position and other personalization and includes that in a pitch email. Attackers find those same tokens and use it to compel more victims into their trap. It’s an especially dangerous ploy.

Email Account Takeover

All members of your executive and management team are vulnerable. If a phishing scammer acquires the email credentials of high-profile leadership, it’s likely they’ll target anyone they can using that very email address. Potential targets would be: colleagues, team members and even customers (if they’ve already obtained this information via hack).

Phishing Emails

Similar to the email account takeover scam, this phishing attack is done via email. The difference is the phishing scammer uses an email address that resembles a legitimate email address, person or company. The email will include a request to click a link, change a password, send a payment, respond with sensitive information, or open a file attachment.

Phone Phishing or Voice Phishing

Using Voice over Internet Protocol (VoIP) technology, scammers, again, impersonate companies. This technique also employs the other types of phishing including using personal details about targets and impersonating individuals of the company (e.g., the CEO) in order to get a higher take on the overall scam.

b. REFERENCES

a. Routhu Srinivasa Rao1 , Alwyn Roshan Pais :Detection of phishing websites using anefficient feature-based machine learning framework :In Springer 2018. Volume 3, Issue 7, September-october-2018 | http:// ijsrcseit.com Purvi Pujara et al. Int J S Res CSE & IT. 2018September-October-2018; 3(7) : 395-399 399

b. Chunlin Liu, Bo Lang :Finding effective classifier for malicious URL detection :

c. InACM,2018

d. Sudhanshu Gautam, Kritika Rani and Bansidhar Joshi : Detecting Phishing WebsitesUsingRule-Based Classification Algorithm: A Comparison : In Springer,2018.

e. M. Amaad Ul Haq Tahir, Sohail Asghar, Ayesha Zafar, Saira Gillani : A Hybrid Model to Detect Phishing-Sites using Supervised Learning Algorithms :In International Conference on Computational Science and Computational Intelligence IEEE ,2016.

f. Hossein Shirazi, Kyle Haefner, Indrakshi Ray: Fresh-Phish: A Framework for Auto- Detection of Phishing Websites: In (International Conference on Information Reuse andIntegration (IRI)) IEEE,2017.

g. Ankit Kumar Jain, B. B. Gupta : Towards detection of phishing websites on client-sideusing machine learning based approach :In Springer Science+Business Media, LLC, part ofSpringer Nature2017

h. Bhagyashree E. Sananse, Tanuja K. Sarode : Phishing URL Detection: A MachineLearning and Web Mining-based Approach : In International Journal of ComputerApplications,2015

i. Mustafa AYDIN, Nazife BAYKAL : Feature Extraction and Classification PhishingWebsites Based on URL : IEEE,2015

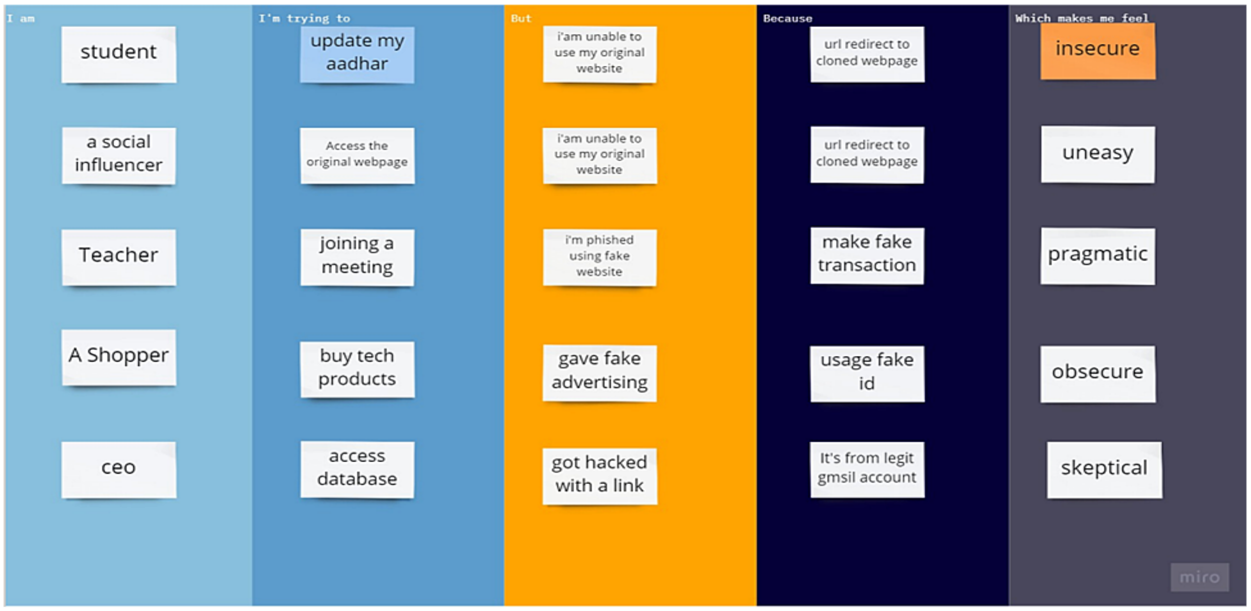
j. Priyanka Singh, Yogendra P.S. Maravi, Sanjeev Sharma : Phishing Websites Detectionthrough Supervised Learning Networks : In IEEE,2015

k. Pradeepthi. K V and Kannan. A: Performance Study of Classification Techniques forPhishing URL Detection: In 2014 Sixth International Conference on Advanced Computing(ICoAC) IEEE,2014

l. Luong Anh Tuan Nguyen†, Ba Lam To† ,Huu Khuong Nguyen† and Minh Hoang Nguyen : Detecting Phishing Web sites: A Heuristic URL-Based Approach: In The 2013International Conference on Advanced Technologies for Communications (ATC'13)

m. Ahmad Abunadi, Anazida Zainal ,Oluwatobi Akanb: Feature Extraction Process: APhishing Detection Approach :In IEEE,2013.

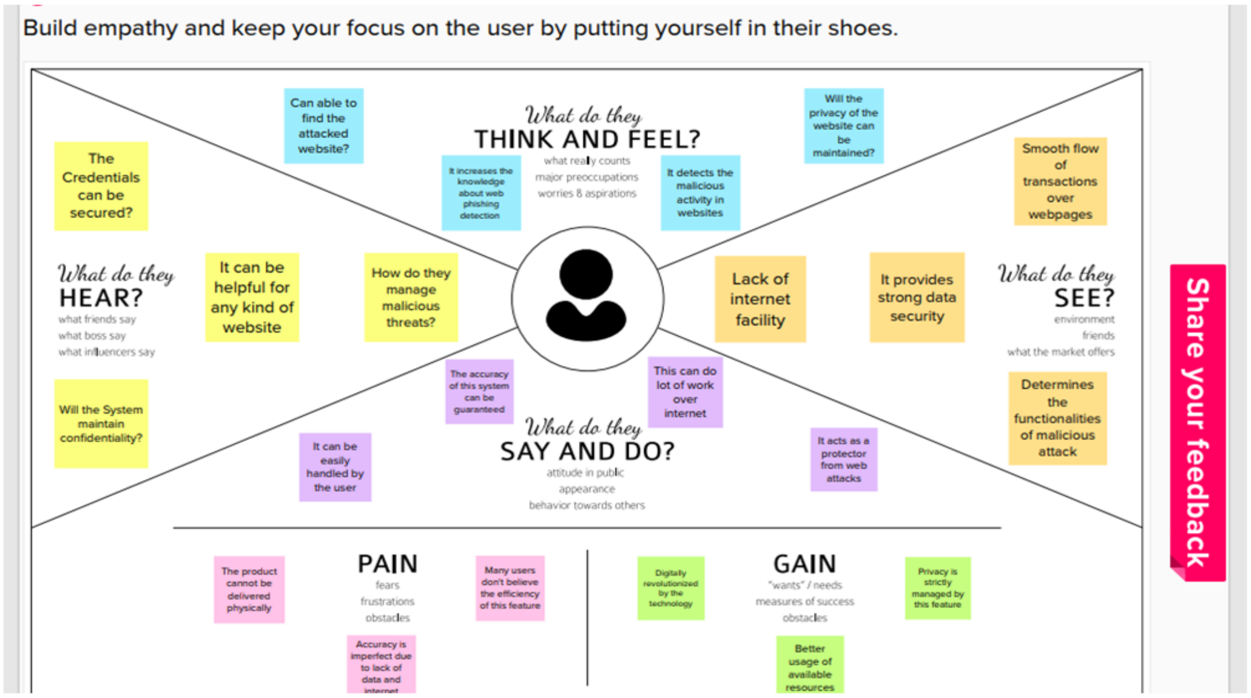
c.PROBLEM STATEMENT DEFINITION



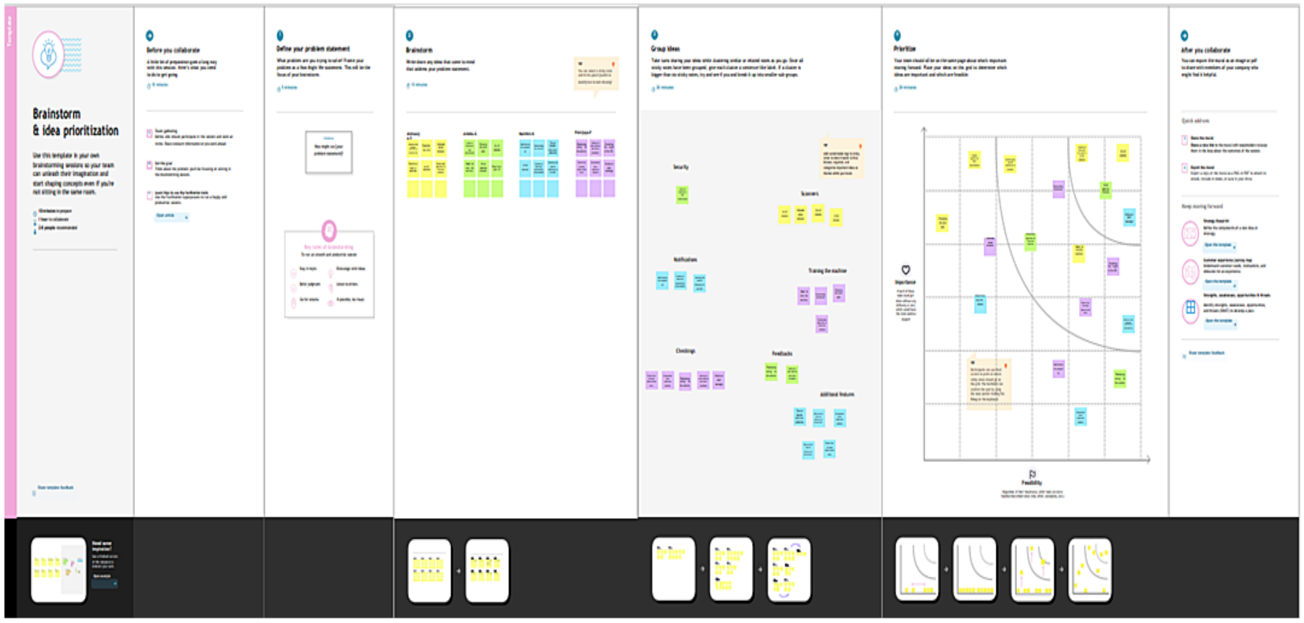
I am	I'm trying to	But	Because	Which makes me feel
student	update my aadhar	i'am unable to use my original website	url redirect to cloned webpage	insecure
a social influencer	Access the original webpage	i'am unable to use my original website	url redirect to cloned webpage	uneasy
Teacher	joining a meeting	i'm phished using fake website	make fake transaction	pragmatic
A Shopper	buy tech products	gave fake advertising	usage fake id	obscure
ceo	access database	got hacked with a link	It's from legit gmsil account	skeptical

3.IDEATION& PROPOSED SOLUTION

a.EMPATHY MAP CANVAS


Build empathy and keep your focus on the user by putting yourself in their shoes.
What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Can able to find the attacked website?
It increases the knowledge about web phishing detection
It detects the malicious activity in websites
Will the privacy of the website can be maintained?
What do they HEAR?
what friends say
what boss say
what influencers say
The Credentials can be secured?
It can be helpful for any kind of website
How do they manage malicious threats?
Will the System maintain confidentiality?
What do they SEE?
environment
friends
what the market offers
Smooth flow of transactions over webpages
Lack of internet facility
It provides strong data security
Determines the functionalities of malicious attack
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
The accuracy of this system can be guaranteed
This can do lot of work over internet
It can be easily handled by the user
It acts as a protector from web attacks
PAIN
fears
frustrations
obstacles
The product cannot be delivered physically
Many users don't believe the efficiency of this feature
Accuracy is imperfect due to lack of data and internet
GAIN
"wants" / needs
measures of success
obstacles
Digitally revolutionized by the technology
Privacy is strictly managed by this feature
Better usage of available resources
Share your feedback

b.IDEATION& BRAINSTORMING



c.PROPOSED SOLUTION

S.NO.	PARAMETER	DESCRIPTON
1.	**Problem Statement(Problem to be solved)**	● 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.

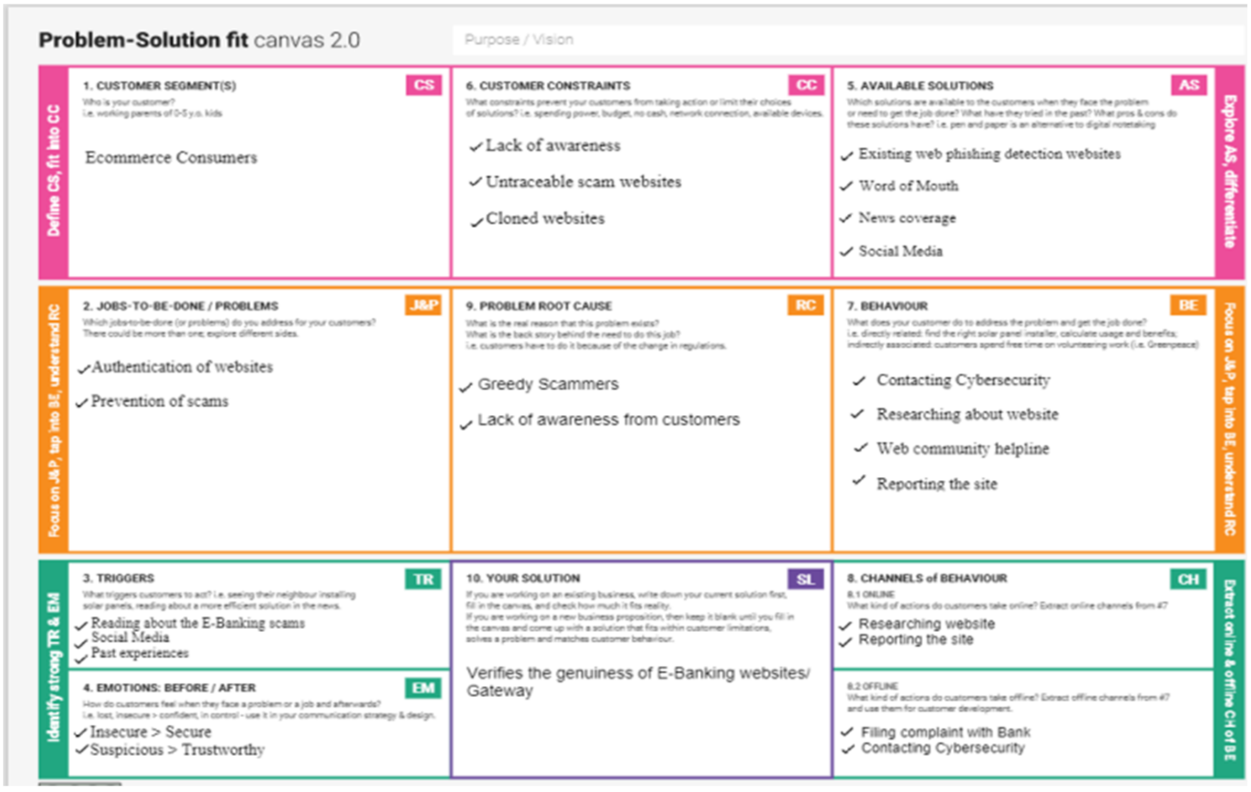
2.	**Idea /Solution description**	a. In order to detect and predict e-banking phishing websites, we proposed an intelligent, flexible and effective system that is based on using classification algorithms. We implemented classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy.

3.	**Novelty /Uniqueness**	b. 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.
4.	**Social Impact /Customer Satisfaction**	c. The feasibility of implementing this idea is moderate neither easy nor tough because the system needs to satisfy the basic requirements of the customer as well as it should act as a bridge towards achieving high accuracy on predicting and analysing the detected websites or files to protect our customer to the fullest.

5.	**Business Model (Revenue Model)**	d. People buy subscription annually,to protect their files both locally and at remote location with the help of our cloud integrated flask app for web phishing detection.

6.	**Scalability of the Solution**	a. By implementing this system, the people can efficiently and effectively to gain knowledge about the web phishing techniques and ways to eradicate them by detection . This system can also be integrated with the future technologies.

d.PROBLEM SOLUTION FIT



Problem-Solution fit canvas 2.0

Purpose / Vision

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS
Who is your customer?
i.e. working parents of 0-5 y.o. kids

Ecommerce Consumers

6. CUSTOMER CONSTRAINTS CC
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.

✓ Lack of awareness
✓ Untraceable scam websites
✓ Cloned websites

5. AVAILABLE SOLUTIONS AS
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 notetaking

✓ Existing web phishing detection websites
✓ Word of Mouth
✓ News coverage
✓ Social Media

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P
Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

✓ Authentication of websites
✓ Prevention of scams

9. PROBLEM ROOT CAUSE RC
What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

✓ Greedy Scammers
✓ Lack of awareness from customers

7. BEHAVIOUR BE
What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

✓ Contacting Cybersecurity
✓ Researching about website
✓ Web community helpline
✓ Reporting the site

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR
What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

✓ Reading about the E-Banking scams
✓ Social Media
✓ Past experiences

4. EMOTIONS: BEFORE / AFTER EM
How do customers feel when they face a problem or a job and afterwards?
i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

✓ Insecure > Secure
✓ Suspicious > Trustworthy

10. YOUR SOLUTION SL
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.

Verifies the genuiness of E-Banking websites/ Gateway

8. CHANNELS of BEHAVIOUR CH
8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

✓ Researching website
✓ Reporting the site

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

✓ Filing complaint with Bank
✓ Contacting Cybersecurity

Extract online & offline CH of BE

4.REQUIREMENT ANALYSIS

a.FUNCTIONAL REQUIREMENTS

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Verifying input	User inputs an URL (Uniform Resource Locator) in necessary field to check its validation.
FR-2	Website Evaluation	Model evaluates the websiteusing Blacklist and Whitelist approach
FR-3	Extraction and Prediction	It retrieves features based on heuristics and visual similarities. The URL is predicted by the model using Machine Learning methods such as Logistic Regression and KNN.
FR-4	Real Time monitoring	The use of Extension plugin should provide a warning pop-up when they visit a website that is phished. Extension plugin will have the capability to also detect latest and new phishing websites
FR-5	Authentication	Authentication assures secure site, secure processes and enterprise information security.

b.NONFUNCTIONAL REQUIREMENTS

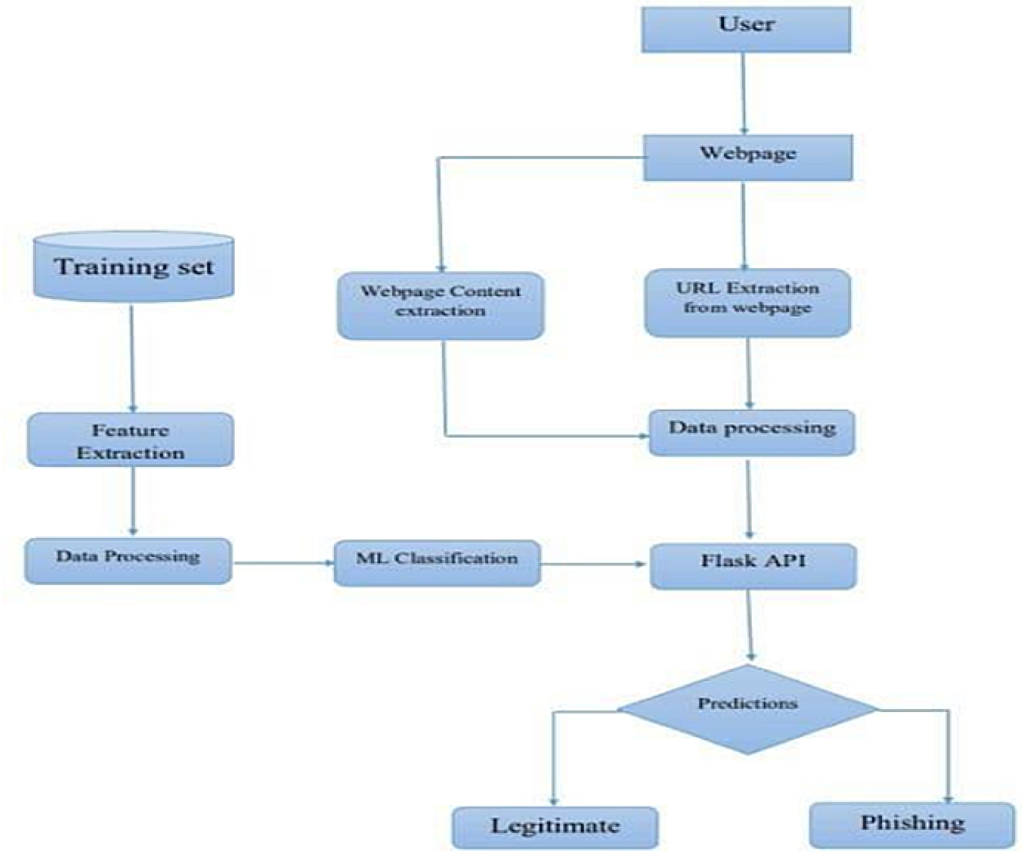
FR No.	Non-Functional Requirement	Description

NFR-1	**Usability**	Analysis of consumers' product usability in the design process with user experience as the core may certainly help designers better grasp users' prospective demands in web phishing detection, behaviour, and experience.
NFR-2	**Security**	It guarantees that any data included within the system or its components will be safe from malware threats or unauthorised access.If you wish to prevent unauthorised access to the admin panel, describe the login flow and different user roles as system behaviour or user actions.
NFR-3	**Reliability**	It specifies the likelihood that the system or its component will operate without failure for a specified amount of time under prescribed conditions.
NFR-4	**Performance**	It is concerned with a measurement of the system's reaction time under various load circumstances.
NFR-5	**Availability**	It represents the likelihood that a user will be able to access the system at a certain moment in time. While it can be represented as an expected proportion of successful requests, it can also be defined as a percentage of time the system is operational within a certain time period.
NFR-6	**Scalability**	It has access to the highest workloads that will allow the system to satisfy the performance criteria. There are two techniques to enable the system to grow as workloads increase: Vertical and

		horizontal scaling.

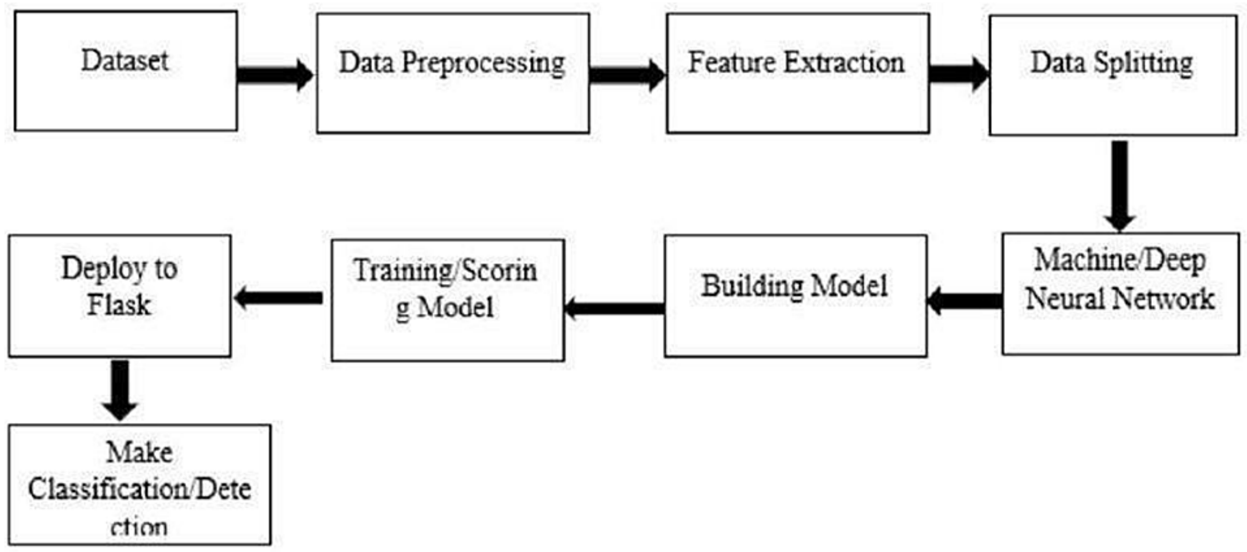
5.PROJECT DESIGN

a.DATAFLOW DIAGRAM

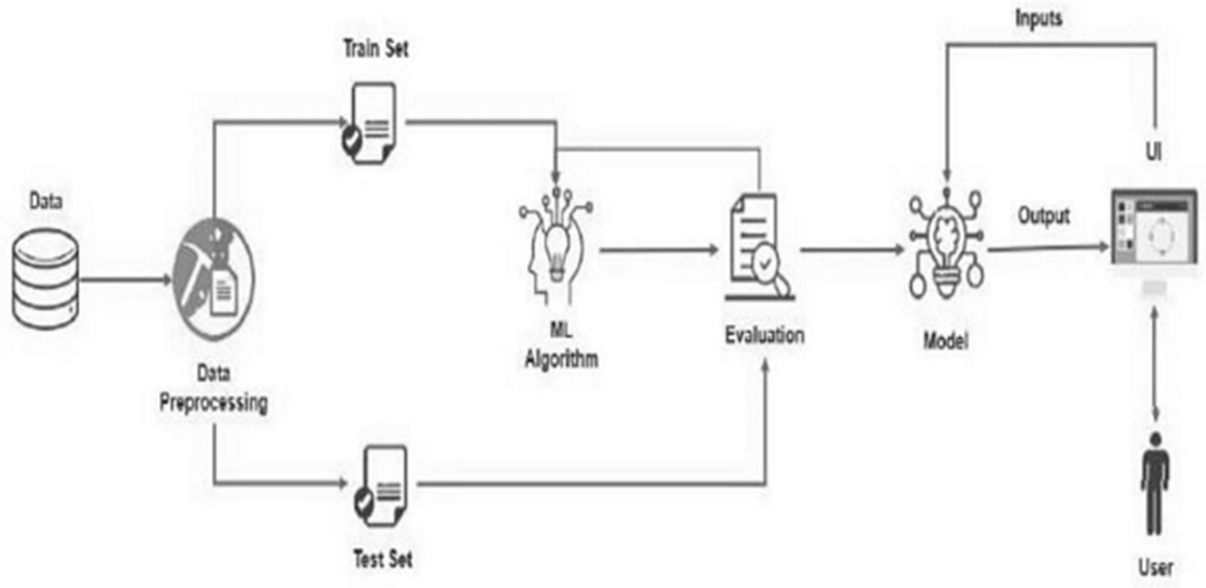


b.SOLUTION & TECHNICAL ARCHITECTURE

Solution Architecture

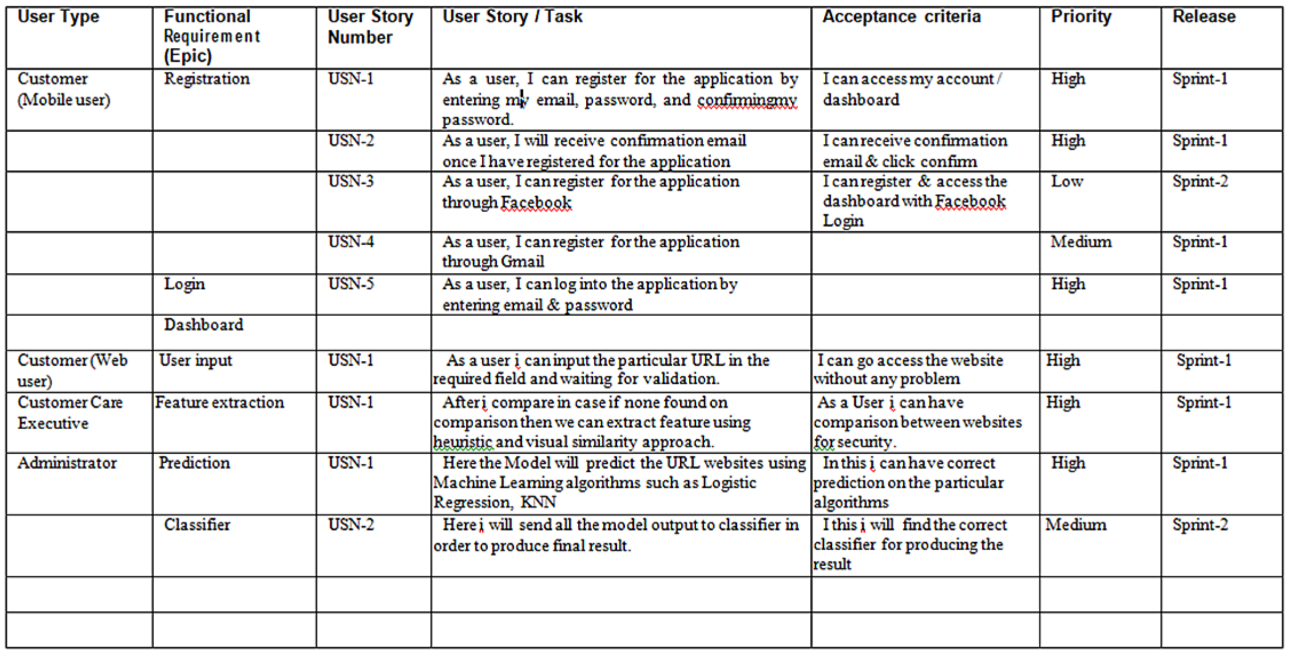


Technical architecture



c.USER STORIES

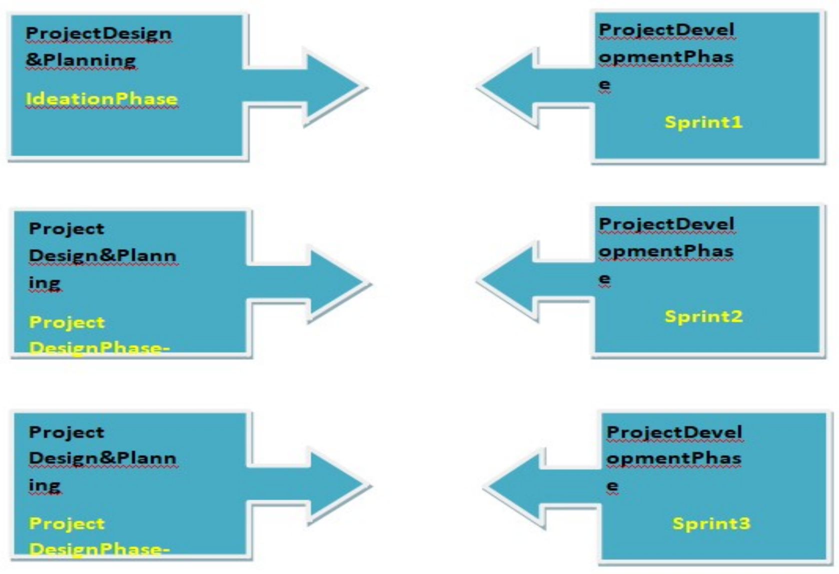
Use the below template to list all theuser stories for the product.



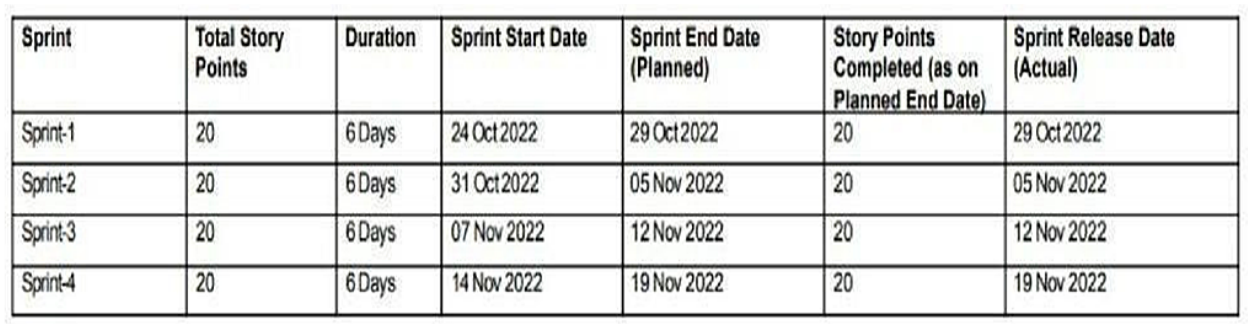
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirmingmy password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web user)	User input	USN-1	As a user i can input the particular URL in the required field and waiting for validation.	I can go access the website without any problem	High	Sprint-1
Customer Care Executive	Feature extraction	USN-1	After i compare in case if none found on comparison then we can extract feature using heuristic and visual similarity approach.	As a User i can have comparison between websites for security.	High	Sprint-1
Administrator	Prediction	USN-1	Here the Model will predict the URL websites using Machine Learning algorithms such as Logistic Regression, KNN	In this i can have correct prediction on the particular algorithms	High	Sprint-1
	Classifier	USN-2	Here i will send all the model output to classifier in order to produce final result.	I this i will find the correct classifier for producing the result	Medium	Sprint-2

6.PROJECT PLANNING & SCHEDULING

a.SPRINT PLANNING & ESTIMATION

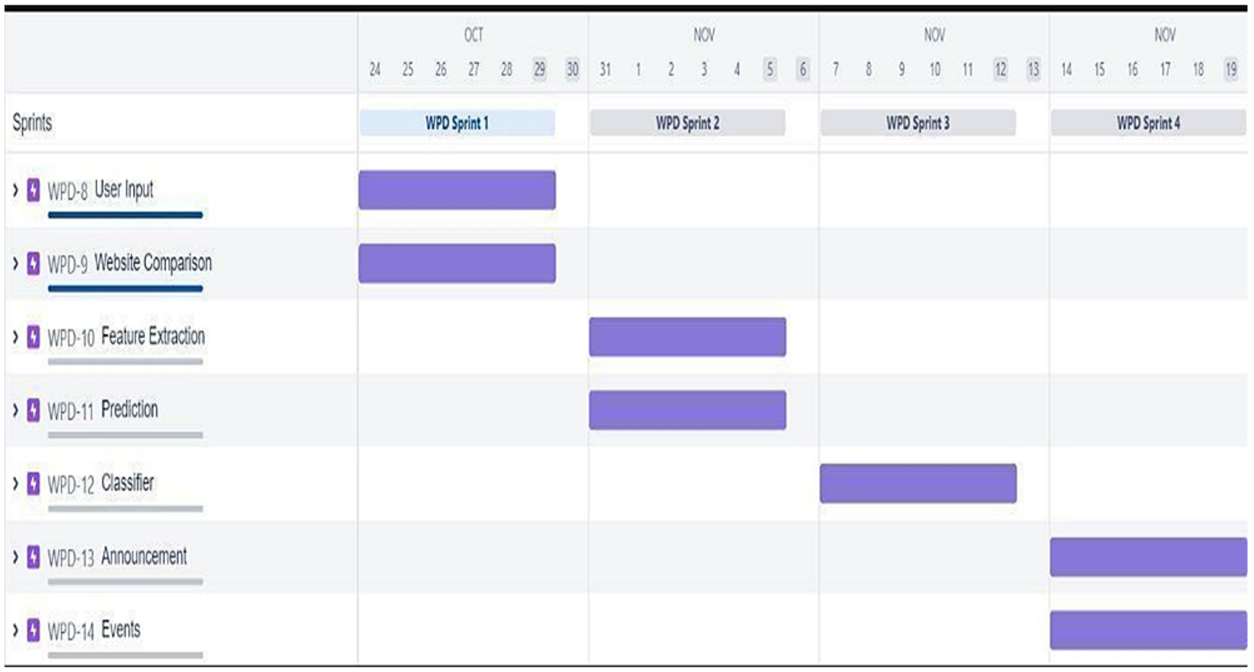


b.SPRINT DELIVERY SCHEDULE



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

c. Reports from JIRA



7.CODING& SOLUTIONING

a. **Feature 1**

#app.py

# importing requiredlibraries

```
from feature import FeatureExtraction
from flask import Flask, request, render_template
import numpy asnp
import pandasas pd
from sklearn import metrics
importwarnings
import pickle
warnings.filterwarnings('ignore')

file = open("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, port=2002)
```

b. **Feature 2**

```
#feature.py
import ipaddress
import re
import urllib.request
from bs4 import BeautifulSoup
import socket
import requests
from googlesearch importsearchimport
whois
from datetime importdate, datetime
importtime
```

```
from dateutil.parser importparse 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|1url\.com|tweez\.me|v\.g
d|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.schemeif
        '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']:
                    return1
        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)forx 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)forx 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.urlin 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 xin re.finditer('\.', iframe['src'])]
            if self.urlin iframe['src'] or self.domain in iframe['src'] or len(dots) == 1:
                success= success + 1
            i = i+1


        try:
            percentage = success/float(i) * 100if
            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 (urlin
a['href'] or self.domain in a['href']):
                     unsafe = unsafe + 1i
                 = 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 forminself.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:
            return1
        else:
            return -1
    except:
        return -1


# 19. WebsiteForwarding
def WebsiteForwarding(self):
    try:
        if len(self.response.history) <= 1:
            return1
        elif len(self.response.history) <= 4:
            return0
        else:
            return -1
    except:
        return -1


# 20. StatusBarCust
def StatusBarCust(self):
    try:

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

```
# 21. DisableRightClick
def DisableRightClick(self):
    try:
        if re.findall(r"event.button ?== ?2", self.response.text):
            return1
        else:
            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
defIframeRedirection(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\.ly', 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\.21
1\.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|10
7\.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
\.19\.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\.1
56\.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):
        returnself.features
```

8.TESTING

a.TEST CASES

Date	15-Nov-22
Team ID	PNT2022TMID34222
Project Name	Project - Web Phishing Detection
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_ OO1	Functional	Home Page	Verify user is able to see theLanding Page when user can type the URL in thebox		1.Enter URL and click go 2.Type theURL 3.Verify whetherit is processing or not.	https://phishing-shield.herokuapp.com/	Should Display the Webpage	Working as expected	Pass		N		V.Aishwarya
LoginPage_TC_ OO2	UI	Home Page	Verifythe UI elements is Responsive		1.Enter URLand click go 2.Typeor copy pastethe URL 3.Check whether the button isresponsive or not 4.Reload and Test Simultaneously	https://phishing-shield.herokuapp.com/	Should Wait for Response and thengets Acknowledge	Working as expected	Pass		N		G.Amisha
LoginPage_TC_ OO3	Functional	Home page	Verify whether the link islegitimate or not		Enter URL and click go Typeor copy pastethe URL Check the website is legitimateor not Observe the results	https://phishing-shield.herokuapp.com/	User should observe whether thewebsite is legitimate or not.	Working as expected	Pass		N		K.Nanthini
LoginPage_TC_ OO4	Functional	Home Page	Verify user is able to access thelegitimate website or not		Enter URLand click go Typeor copy pastethe URL Check the website is legitimateor not Continue if the website is legitimate or be cautious if it isnot legitimate.	https://phishing-shield.herokuapp.com/	Application shouldshow that SafeWebpageor Unsafe.	Working as expected	Pass		N		V.Aishwarya

LoginPage_TC_ OO5	Functional	Home Page	Testing the website with multiple URLs		Enter URL (https://phishing-shield.herokuapp.com/) and click go Type or copy paste the URL totest Check the website is legitimateor not Continue if the website is secure or be cautious if it is notsecure	1. https://avbalajee.github.io /welcom 2.totalpad.com https://www.klnce.edu salescript.info 5. https://www.google.com/6. delgets.com	User can able to identify thewebsites whether it is secure ornot	Working as expected	Pass		N		G.Amisha

b.USER ACCEPTANCE TESTING

User Acceptance Testing

UAT Execution & Report Submission

Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [Web Phishing Detection] project at the time of the release to User Acceptance Testing (UAT).

Defect Analysis

This report shows the number of resolvedor closed bugs at each severity level, and howtheywere resolved

1.Test Case Analysis

This reportshows the numberof test cases that have passed, failed,and untested

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
ByDesign	10	4	2	3	20
Duplicate	1	0	3	0	4

External	2	3	0	1	6
Fixed	10	2	4	20	36
NotReproduced	0	0	1	0	1

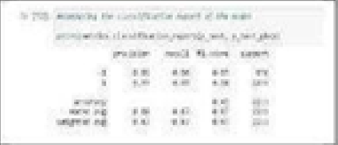
Test Case Analysis

This report shows the number of testcases that have passed, failed,and untested

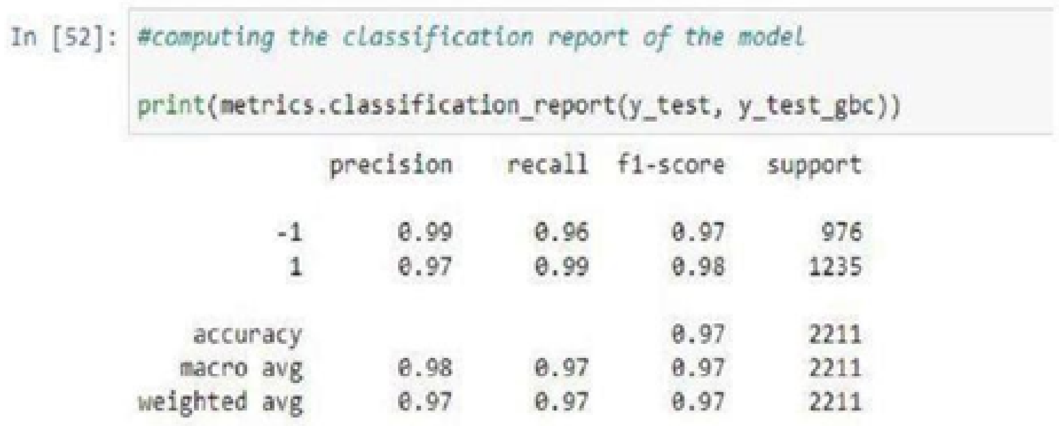
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	10	0	0	10
Client Application	50	0	0	50
Security	5	0	0	4
Outsource Shipping	3	0	0	3
Exception Reporting	10	0	0	9
Final Report Output	10	0	0	10
Version Control	4	0	0	4

9.RESULTS

a.PERFORMANCE METRICES

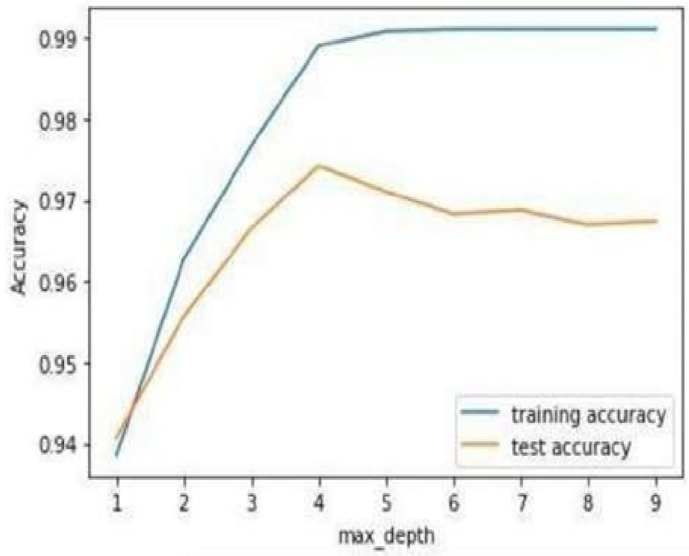
S.No.	Parameter	Values	Screenshot
1.	Metrics	**Classification Model:** **Gradient Boosting Classification** Accuray Score- 97.4%	 [illegible]
2.	Tune the Model	Hyperparameter Tuning - 97% Validation Method – KFOLD & Cross Validation Method	[illegible]

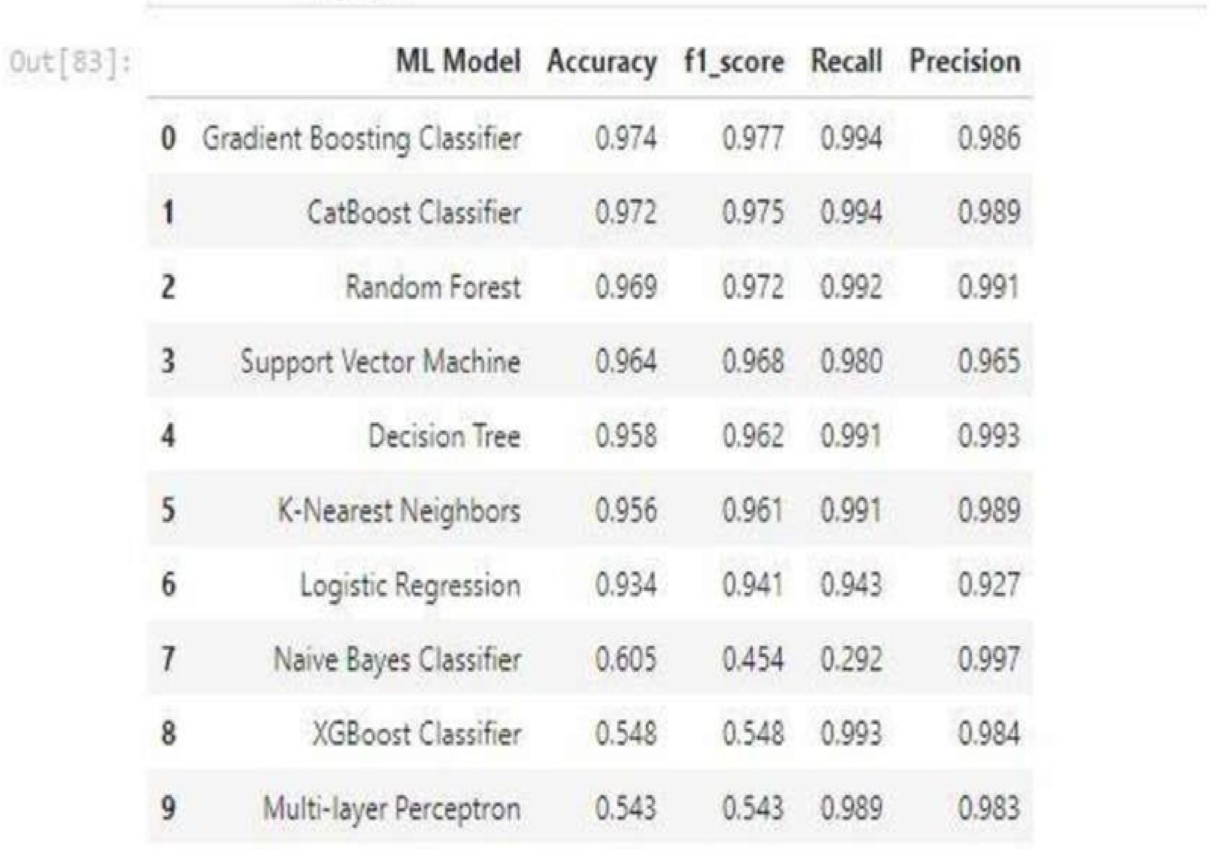
1. METRICS:
CLASSIFICATION REPORT:



```
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
```

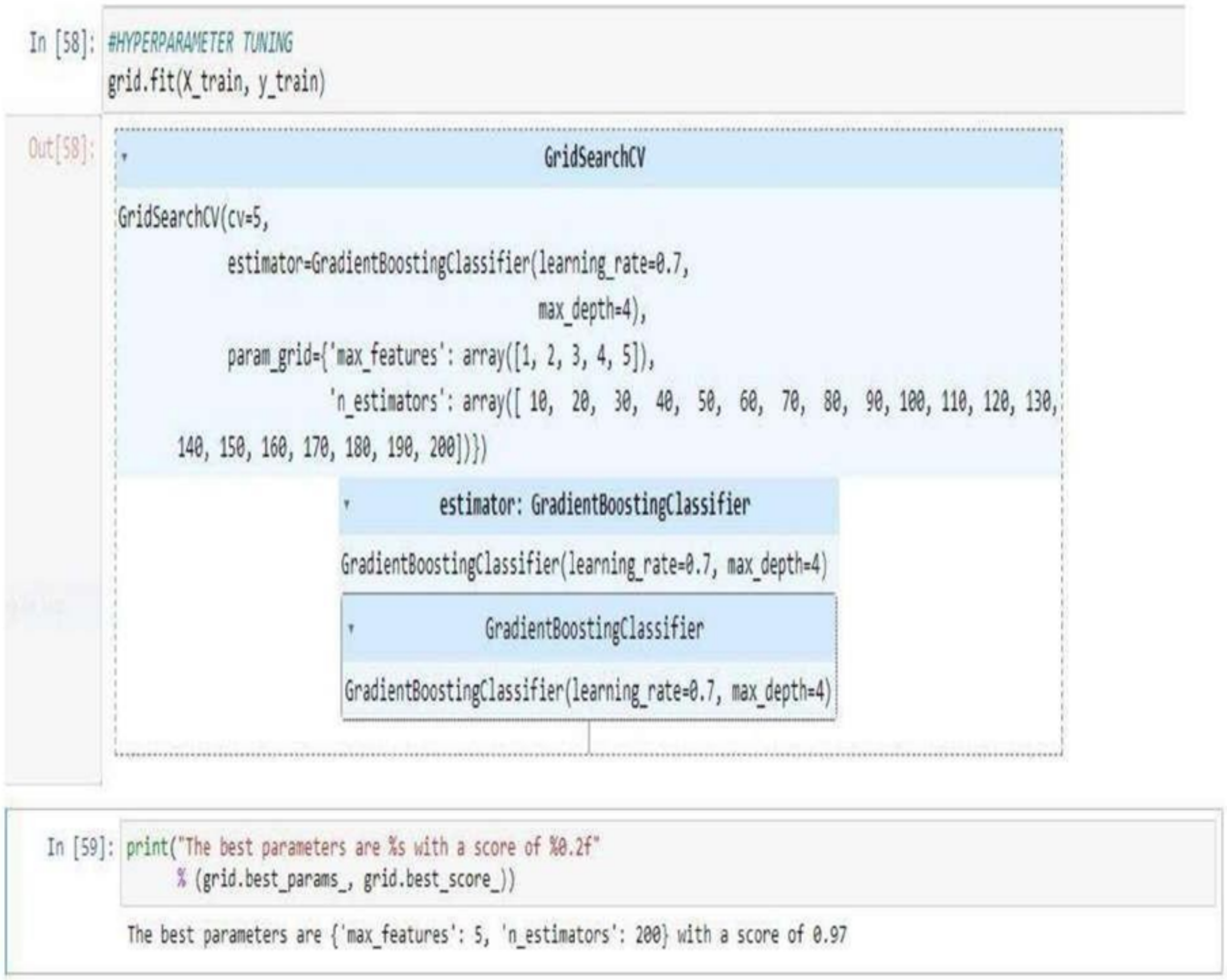
PERFORMANCE :




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

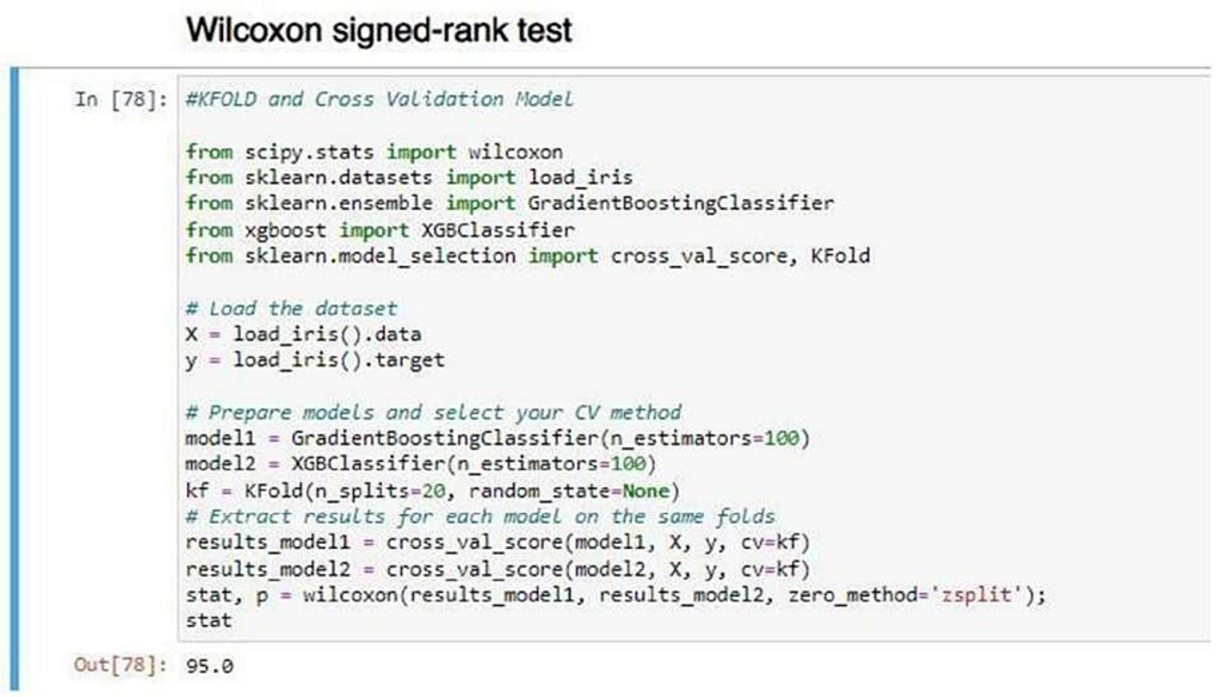


```
In [58]: #HYPERPARAMETER TUNING
         grid.fit(X_train, y_train)
```

```
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)
```

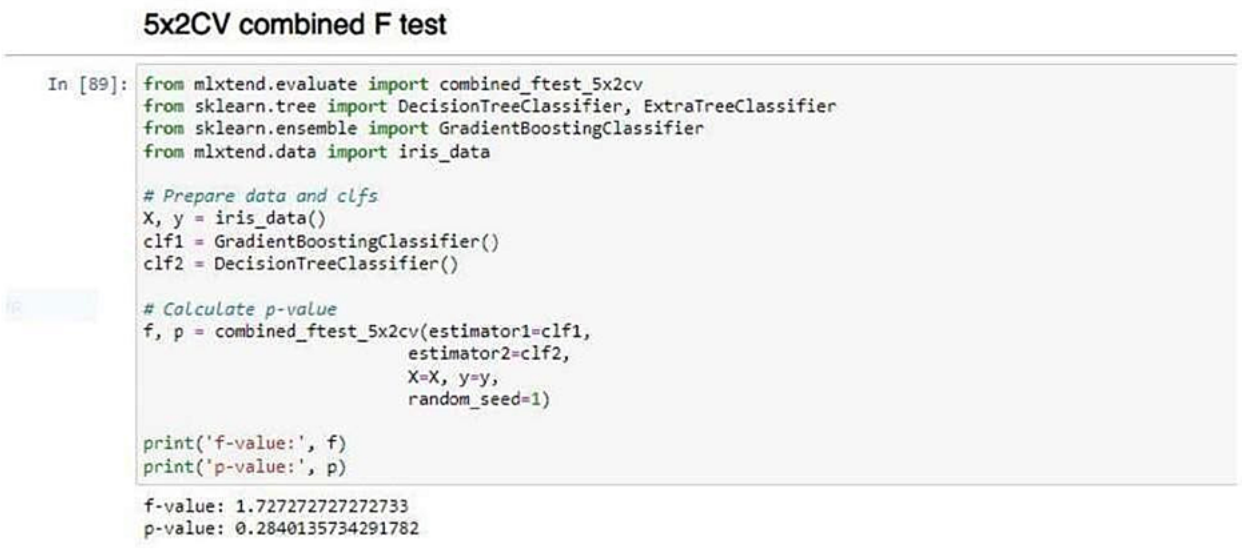
```
In [59]: print("The best parameters are %s with a score of %0.2f"
               % (grid.best_params_, grid.best_score_))

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
```

1O.ADVANTAGES & DISADVANTAGES

Advantages of web phishingdetection

i. Improve on Inefficiencies of SEG and Phishing Awareness Training

ii. It Takesa Load off the SecurityTeam
iii. It Offers a Solution, Not a Tool
iv. Separate You from Your Competitors
v. Thissystem can be used by many e-commerce websites in order to have goodcustomer relationships.
vi. If internetconnectionfails this system will work

Disadvantages of web phishingdetection

1. All website related data will be stored in one place.
2. It is a very time-consuming process.

11.CONCLUSION

It is outstanding that a decent enemy of phishing apparatus ought to anticipate the phishing assaults in a decent timescale. We accept that the accessibility of a decent enemy of phishing device at a decent time scale is additionally imperative to build the extent of anticipating phishing sites. This apparatus ought to be improved continually through consistent retraining.As a matter of fact, the accessibility of crisp and cutting-edge preparing dataset which may gained utilizing our very own device [30, 32] will help us to retrain our model consistently and handle any adjustments in the highlights, which are influential in deciding the site class. Albeit neural system demonstrates its capacity to tacklea wide assortment of classification issues, the procedure of finding the ideal structure is verydifficult, and much of the time, this structure is controlled by experimentation.

Our model takes care of this issue via computerizing the way toward organizing a neural system conspire; hence, on the off chance that we construct an enemy of phishing model and for any reasons we have to refresh it, at that point our model will encourage this procedure, that is, since our model will mechanize the organizing procedure and will request scarcely any client definedparameters.

12.FUTURE SCOPE

There is a scope for future development of this project. We will implement this using advanced deep learning method to improve the accuracy and precision. Enhancements canbe done in an efficient manner. Thus, the project is flexible and can be enhanced at any timewith more advancedfeatures.

13.APPENDIX

1. Application Building
2. Collection of Dataset
3. Data Pre-processing
4. Integration of Flask App with IBM Cloud
5. Model Building
6. Performance Testing
7. Training the model on IBM
8. User Acceptance Testing
9. Ideation Phase
10. Preparation Phase
11. Project Planning
12. Performance Testing
13. User Acceptance Testing

Project Link:https://github.com/IBM-EPBL/IBM-Project-50534-1660915255

Project Demo Link:https://drive.google.com/file/d/1yjxSQQdXFSFi2F-N1cwtyjYZOH_AAVS9/view?usp=drivesdk