Detecting Phishing Websites Using ML-Algorithms

Minor Project II

Submitted By :

Manan Chaudhary (9919103178)

Kartikeya Consul (9919103183)

Megha Jain (9919103188)

Under the Supervision of :

Mr. Surendra Kumar

Department of CSE/IT

Jaypee Institute of Information Technology , Noida

MAY 2022

ACKNOWLEDGEMENT

I would like to place on record my deep sense of gratitude to **Mr. Surendra Kumar**, Assistant Professor (Grade-II), Jaypee Institute of Information Technology, India for her generous guidance, help and useful suggestions.

I express my sincere gratitude to M**r. Surendra Kumar** , Dept. of CSE/IT, Jaypee Institute of Information Technology, India for her stimulating guidance, continuous encouragement and supervision throughout the course of present work.

I also wish to extend my thanks to friends and other classmates for their insightful comments and constructive suggestions to improve the quality of this project work.

Signature(s) of Students

Manan Chaudhary (9919103178)

Kartikeya Consul (9919103183)

Megha Jain (9919103188)

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and beliefs, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma from a university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place : Noida

Date : 16-05-2022

Name : Manan Chaudhary

Enrolment No.: 9919103178

Name : Kartikeya Consul

Enrolment No.: 9919103183

Name : Megha Jain

Enrolment No.: 9919103188

CERTIFICATE

This is to certify that the work titled "**Detecting Phishing Websites Using ML-Algorithms** " submitted by Manan Chaudhary, Kartikeya Consul and Megha Jain of B.Tech of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of any other degree or diploma.

Digital Signature of Supervisor
Mr. Surendra kumar
Assistant Professor (Grade-II)
Date : 16-05-2022

ABSTRACT OF PROJECT

Phishing is an illegitimate method to collect secret information of any person or organization. Information like debit card, credit card details, PIN no, OTP, passwords, etc. are stolen by the attackers through phishing sites. Researchers have used different techniques to detect those phishing sites. But it is difficult to stay on a particular technique as attackers come with new tactics. In this paper, phishing and legitimate URL classifications are performed based on the lexical features of URLs. To evaluate the performance of our system, we have taken 30 features from URL to detect a website as a phishing or non-phishing. The system is trained using more than 11,055 phishing and legitimate URLs with XG-Boost classifier. Our experiment results show more than 97.47% accuracy in detecting phishing websites using XG-Boost Classifier.

LIST OF FIGURES

Fig. No.	Topic Name	Page No.
1	Supervised Learning	6
2	Decision tree	7
3	Gradient boosting decision Tree	7
4	Work flow Design	9
5	Accuracy scores	18
6	Website Interface	19
7	Result for legitimate website	20
8	Result for phishing website	20

CONTENTS

		Page No.
Abstract		*i*
List of Figures		*ii*
Table of Contents		*iii*
Chapter 1 :	**Introduction**	
	1.1 Introduction	1
	1.2 Type of phishing	1
	1.2.1 Email Phishing	1
	1.2.2 Spear Phishing	1
	1.2.3 whaling and ceo fraud	2
	1.2.4 Clone Phishing	2
	1.3 State of art	
Chapter 2 :	**Background Study**	
	2.1 Phishing Website detection techniques	3
	2.2 State of Art	4
Chapter 3 :	**Requirement Analysis**	
	3.1 Software Requirements	5
	3.2 Libraries Used	5
	3.3 Classifier Used	7
Chapter 4 :	**Detailed Design**	
	4.1 Objectives of study	9
	4.2 Work Flow	10
	4.3 Dataset description	11
Chapter 5 :	**Implementation**	
	5.1 Feature Extraction	12
	5.1.1 Address bar based features	13
	5.1.2 Abnormal based features	14
	5.1.3 Html and javascript based features	15
	5.1.4 Domain based features	16
	5.2 Implementing Various models to compare accuracy	17
	5.3 Creating a website to test the phishing URL	20
	5.4 Integrating machine learning model with website	20
Chapter 6 :	**Results**	
	6.1 Results	21
Chapter 7 :	**Future work and conclusion**	
	7.1 Future work	23
	7.2 Conclusion	23
	References	**24**

INTRODUCTION

In recent years, advancements in Internet and cloud technologies have led to a significant increase in electronic trading in which consumers make online purchases and transactions. This growth leads to unauthorized access to users' sensitive information and damages the resources of an enterprise. Phishing is one of the familiar attacks that trick users to access malicious content and gain their information.Phishing attacks have become increasingly sophisticated and often transparently mirror the site being targeted, allowing the attacker to observe everything while the victim is navigating the site, and transverse any additional security boundaries with the victim. In terms of website interface and uniform resource locator (URL), most phishing webpages look identical to the actual webpages. Various strategies for detecting phishing websites, such as blacklist, heuristic, Etc., have been suggested. However, due to inefficient security technologies, there is an exponential increase in the number of victims. The anonymous and uncontrollable framework of the Internet is more vulnerable to phishing attacks. Existing research works show that the performance of the phishing detection system is limited. There is a demand for an intelligent technique to protect users from the cyber-attacks. As of 2020, phishing is by far the most common attack performed by cybercriminals, the FBI's Internet Crime Complaint Centre recording over twice as many incidents of phishing than any other type of computer crime.

TYPES OF PHISHING

EMAIL PHISHING

Most phishing messages are delivered by email, and are not personalized or targeted to a specific individual or company–this is termed "bulk" phishing The content of a bulk phishing message varies widely depending on the goal of the attacker–common targets for impersonation include banks and financial services, email and cloud productivity providers, and streaming services. Attackers may use the credentials obtained to directly steal money from a victim, although compromised accounts are often used instead as a jumping-off point to perform other attacks, such as the theft of proprietary information, the installation of malware, or the spear phishing of other people within the target's organization.Compromised streaming service accounts are usually sold directly to consumers on darknet markets.

SPEAR PHISHING

Spear phishing involves an attacker directly targeting a specific organization or person with tailored phishing communications. This is essentially the creation and sending of emails to a particular person to make the person think the email is legitimate. In contrast to bulk phishing, spear phishing attackers often gather and use personal information about their target to increase their probability of success of the attack. Spear phishing typically targets executives or those that work in financial departments that have access to the organization's sensitive financial data and services. A 2019 study showed that accountancy and audit firms are frequent targets for spear phishing owing to their employees' access to information that could be valuable to criminals.

Threat Group-4127 (Fancy Bear) used spear phishing tactics to target email accounts linked to Hillary Clinton's 2016 presidential campaign. They attacked more than 1,800 Google accounts and implemented the accounts-google.com domain to threaten targeted users.

A recent study tested the susceptibility of certain age groups against spear fishing. In total, 100 young and 58 older users received, without their knowledge, daily simulated phishing emails over 21 days. A browser plugin recorded their clicking on links in the emails as an indicator of their susceptibility. Forty-three percent of users fell for the simulated phishing emails, with older women showing the highest susceptibility. While susceptibility in young users declined across the study, susceptibility in older users remained stable.

WHALING AND CEO FRAUD

Whaling refers to spear phishing attacks directed specifically at senior executives and other high-profile targets.The content will be likely crafted to be of interest to the person or role targeted - such as a subpoena or customer complaint.

CEO fraud is effectively the opposite of whaling; it involves the crafting of spoofed emails purportedly from senior executives with the intention of getting other employees at an organization to perform a specific action, usually the wiring of money to an offshore account. While CEO fraud has a reasonably low success rate, criminals can gain very large sums of money from the few attempts that do succeed. There have been multiple instances of organizations losing tens of millions of dollars to such attacks.

CLONE PHISHING

Clone phishing is a type of phishing attack whereby a legitimate, and previously delivered email containing an attachment or link has had its content and recipient address(es) taken and used to create an almost identical or cloned email. The attachment or link within the email is replaced with a malicious version and then sent from an email address spoofed to appear to come from the original sender. It may claim to be a resend of the original or an updated version to the original. Typically this requires either the sender or recipient to have been previously hacked for the malicious third party to obtain the legitimate email.

BACKGROUND STUDY

PHISHING WEBSITE DETECTION TECHNIQUES

Method-1

Pop-up window showing while opening a website can be a sign of attempt to gain some of your personal information.

Links and website addresses that contain malicious content may look almost identical to legitimate sites, normally using slight spelling changes, additional special characters, and/or a different domain (ex: .com instead of .gov).

Check that the URL begins with an “https://” or “shttp://”.but this is not 100% foolproof and there have been a significant rise in the number of phishing sites using SSL certificate, so we need to look for additional evidence.

Limitation :

We cannot do manual checking for each website we visit and also with advancements phishers create URLs almost identical which cannot be detected just by looking at the url .

Method-2

One solution approach is to use a blacklist of malicious URLs developed by anti-virus groups. The problem with this approach is that the blacklist cannot be exhaustive because new malicious URLs keep cropping up continuously. Thus, approaches are needed that can automatically classify a new, previously unseen URL as either a phishing site or a legitimate one. Such solutions are typically machine-learning based approaches where a system can categorize new phishing sites through a model developed using training sets of known attacks.

Limitation :

Cyber criminal use domain generation algorithm (DGA) to circumvent the blacklist by generating new malicious urls.

Method-3

Machine learning techniques that identify phishing URLs typically evaluate a URL based on some feature or set of features extracted from it. There are two general types of features that can be extracted from URLs, namely host-based features and lexical features. Host based features describe characteristics of the website, such as where it is located, who manages it, and when was the site installed. Alternatively, lexical features describe textual properties of the URL. Since URLs are simply text strings that can be divided into subparts including the protocol, hostname, and path, a system can assess a site’s legitimacy based on any combination of those components.

Phishing detection techniques do suffer low detection accuracy and high false alarm especially when novel phishing approaches are introduced. Besides, the most common technique used, blacklist-based method is inefficient in responding to emanating phishing attacks since registering new domain has become easier, no comprehensive blacklist can ensure a perfect up-to-date database. Furthermore, page content inspection has been used by some strategies to overcome the false negative problems and complement the vulnerabilities of the stale lists. Moreover, page content inspection algorithms each have different approach to phishing website detection with varying degrees of accuracy.

STATE OF ART

Sudhanshu et al.[1] used association data mining approach. They have proposed rule based classification technique for phishing website detection. They have concluded that association classification algorithm is better than any other algorithms because of their simple rule transformation. They achieved 92.67% accuracy by extracting 16 features but this is not up to mark so proposed algorithm can be enhanced for efficient detection rate.

M. Amaad et al.[2] presented a hybrid model for classification of phishing website. In this paper, proposed model carried out in two phase. In phase 1,they individually perform classification techniques, and select the best three models based on high accuracy and other performance criteria. While in phase 2, they further combined each individual model with best three model and makes hybrid model that gives better accuracy than individual model. They achieved 97.75% accuracy on testing dataset. There is limitation of this model that it requires more time to build hybrid model.

Ahmad et al.[3] proposed three new features to improve accuracy rate for phishing website detection. In this paper, Author used both type of features as commonly known and new features for classification of phishing and non-phishing site. At the end author has concluded this work can be enhanced by using this novel features with decision tree machine learning classifiers.

Many machine learning techniques have been used for detection of malicious URLs. Sadeh et al. [4] proposed a system called PILFER for classifying phishing URLs. They extracted a set of ten features that are specifically designed to highlight deceptive methods used to fool users. The data set consists of approximately 860 phishing e-mails and 6950 non- phishing emails. They used a Support Vector Machine (SVM) as a classifier in the implementation. They trained and tested the classifier using 10-fold cross validation and obtained 92 percent accuracy.

Authors in this paper [5] also proposed reduced feature selection model to detect phishing websites. They used Logistic Regression and Support Vector Machine (SVM) as classification methods to validate the feature selection method. 19 features reduced from 30 site features have been selected and used for phishing detection. The LR and SVM calculations performance was surveyed dependent on precision, recall, f-measure and accuracy. Study shows that SVM algorithm achieved best performance over LR algorithm.

REQUIREMENT ANALYSIS

SOFTWARE REQUIREMENTS

Programming language	:	python
Operating system	:	windows
Tools	:	python libraries and machine learning classifier

LIBRARIES USED

RE (REGULAR EXPRESSION LIBRARY)

A regular expression (or RE) specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression (or if a given regular expression matches a particular string, which comes down to the same thing).
re.search()
Re.findall()
Re.finditer()

IPADDRESS

ipaddress is a module for inspecting and manipulating IP addresses, the first thing you’ll want to do is create some objects. You can use ipaddress to create objects from strings and integers.

Addresses, often referred to as “host addresses” are the most basic unit when working with IP addressing. The simplest way to create addresses is to use the ipaddress.ip_address() factory function, which automatically determines whether to create an IPv4 or IPv6 address based on the passed in value.

URLLIB.REQUEST

The urllib.request module defines functions and classes which help in opening URLs (mostly HTTP) in a complex world — basic and digest authentication, redirections, cookies and more.

WHOIS

WHOIS is a short form for “Who is responsible for this domain name?” and is not an acronym.
Python-whois module is used for this protocol, which is available at given below link:
https://pypi.python.org/pypi/python-whois.
It doesn’t have any external dependencies it only requires python’s standard library. It is able to extract for all TLDs (org, com, net,...)

BEAUTIFULSOUP

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work. It creates a parse tree from page source code that can be used to extract data in a hierarchical and more readable manner.
It was first introduced by Leonard Richardson, who is still contributing to this project and this project is additionally supported by Tidelift (a paid subscription tool for open-source maintenance) .

GOOGLESEARCH

googlesearch is a Python library for searching Google, easily. googlesearch uses requests and BeautifulSoup4 to scrape Google .To get results for a search term, simply use the search function in googlesearch. googlesearch supports a few additional options. By default, googlesearch returns 10 results. This can be changed. In addition, you can change the language google searches in.

MACHINE LEARNING CLASSIFIER USED :

XGBOOST

XGBoost is an open-source software library that implements optimized distributed gradient boosting machine learning algorithms under the Gradient Boosting framework.

XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems.

It’s vital to an understanding of XGBoost to first grasp the machine learning concepts and algorithms that XGBoost builds upon : supervised machine learning, decision trees, ensemble learning, and gradient boosting[6] .

Supervised machine learning uses algorithms to train a model to find patterns in a dataset with labels and features and then uses the trained model to predict the labels on a new dataset’s features.

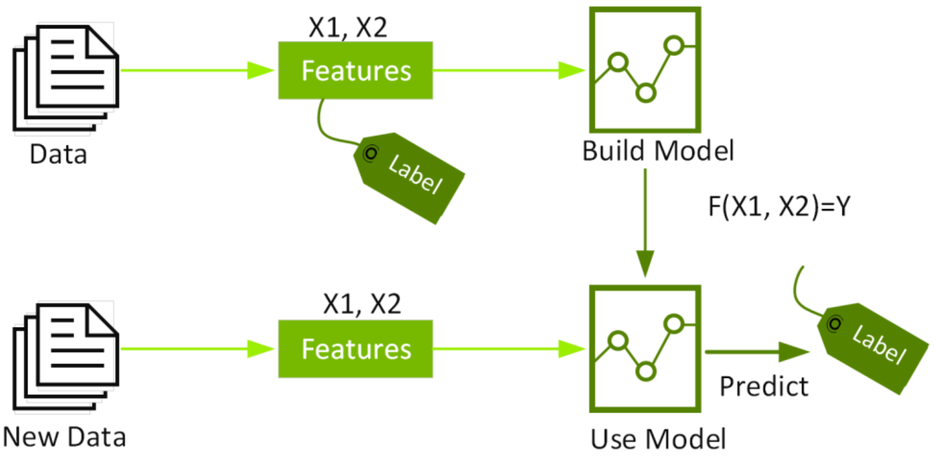


Image 1 : Supervised learning

Decision trees create a model that predicts the label by evaluating a tree of if-then-else true/false feature questions, and estimating the minimum number of questions needed to assess the probability of making a correct decision. Decision trees can be used for classification to predict a category, or regression to predict a continuous numeric value. In the simple example below, a decision tree is used to estimate a house price (the label) based on the size and number of bedrooms (the features).

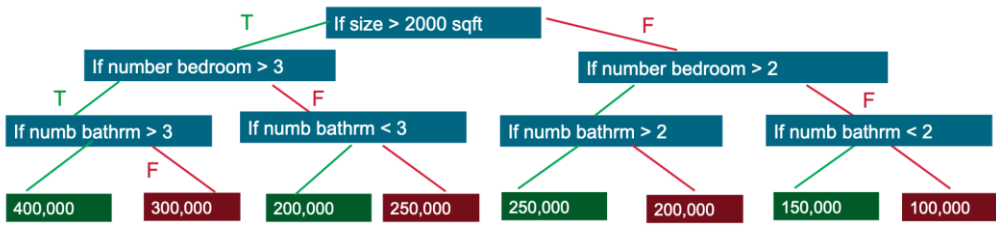


Image 2 : Decision Tree

A Gradient Boosting Decision Trees (GBDT) is a decision tree ensemble learning algorithm similar to random forest, for classification and regression. Ensemble learning algorithms combine multiple machine learning algorithms to obtain a better model.

Both random forest and GBDT build a model consisting of multiple decision trees. The difference is in how the trees are built and combined.

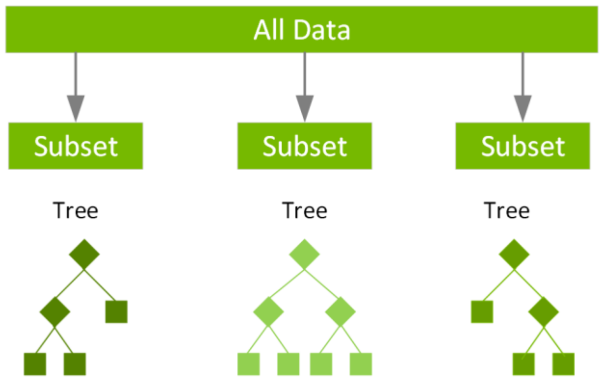


Image 3 : Gradient boosting decision tree

Random forest uses a technique called bagging to build full decision trees in parallel from random bootstrap samples of the data set. The final prediction is an average of all of the decision tree predictions.

The term “gradient boosting” comes from the idea of “boosting” or improving a single weak model by combining it with a number of other weak models in order to generate a collectively strong model. Gradient boosting is an extension of boosting where the process of additively generating weak models is formalized as a gradient descent algorithm over an objective function. Gradient boosting sets targeted outcomes for the next model in an effort to minimize errors. Targeted outcomes for each case are based on the gradient of the error (hence the name gradient boosting) with respect to the prediction.

GBDTs iteratively train an ensemble of shallow decision trees, with each iteration using the error residuals of the previous model to fit the next model. The final prediction is a weighted sum of all of the tree predictions. Random forest “bagging” minimizes the variance and overfitting, while GBDT “boosting” minimizes the bias and underfitting.

XGBoost is a scalable and highly accurate implementation of gradient boosting that pushes the limits of computing power for boosted tree algorithms, being built largely for energizing machine learning model performance and computational speed. With XGBoost, trees are built in parallel, instead of sequentially like GBDT. It follows a level-wise strategy, scanning across gradient values and using these partial sums to evaluate the quality of splits at every possible split in the training set.

XGBOOST BENEFITS AND ATTRIBUTES

The list of benefits and attributes of XGBoost is extensive, and includes the following:

- A large and growing list of data scientists globally that are actively contributing to XGBoost open source development
- Usage on a wide range of applications, including solving problems in regression, classification, ranking, and user-defined prediction challenges
- A library that’s highly portable and currently runs on OS X, Windows, and Linux platforms
- Cloud integration that supports AWS, Azure, Yarn clusters, and other ecosystems
- Active production use in multiple organizations across various vertical market areas
- A library that was built from the ground up to be efficient, flexible, and portable

DETAILED DESIGN

OBJECTIVES OF THE STUDY

The objectives of the study are as follows:

To develop a approach to detect malicious URL and alert users.

To apply ML techniques in the approach in order to analyse the real time URLs and produce effective results.

To implement the concept classifiers , this is a familiar ML technique that has the capability to handle huge amount of data.

WORK FLOW

1. Firstly we have collected the data of legitimate and phishing URL's.
2. Then, we performed feature extraction on those URL's
3. Then, we have trained them using different Machine Learning Models such as: Gradient Boost, Random Forest, Decision Tree, Support Vector, KNN, ADA Boost and XG Boost.
4. Then, we have compared the accuracy we got for all these models
5. Then, we chose the best one, which in this case is XG Boost Classifier with 97.47% accuracy.
6. Then we will implement this model with a website which will tell that the inputted URL is a legitimate or a phished URL.

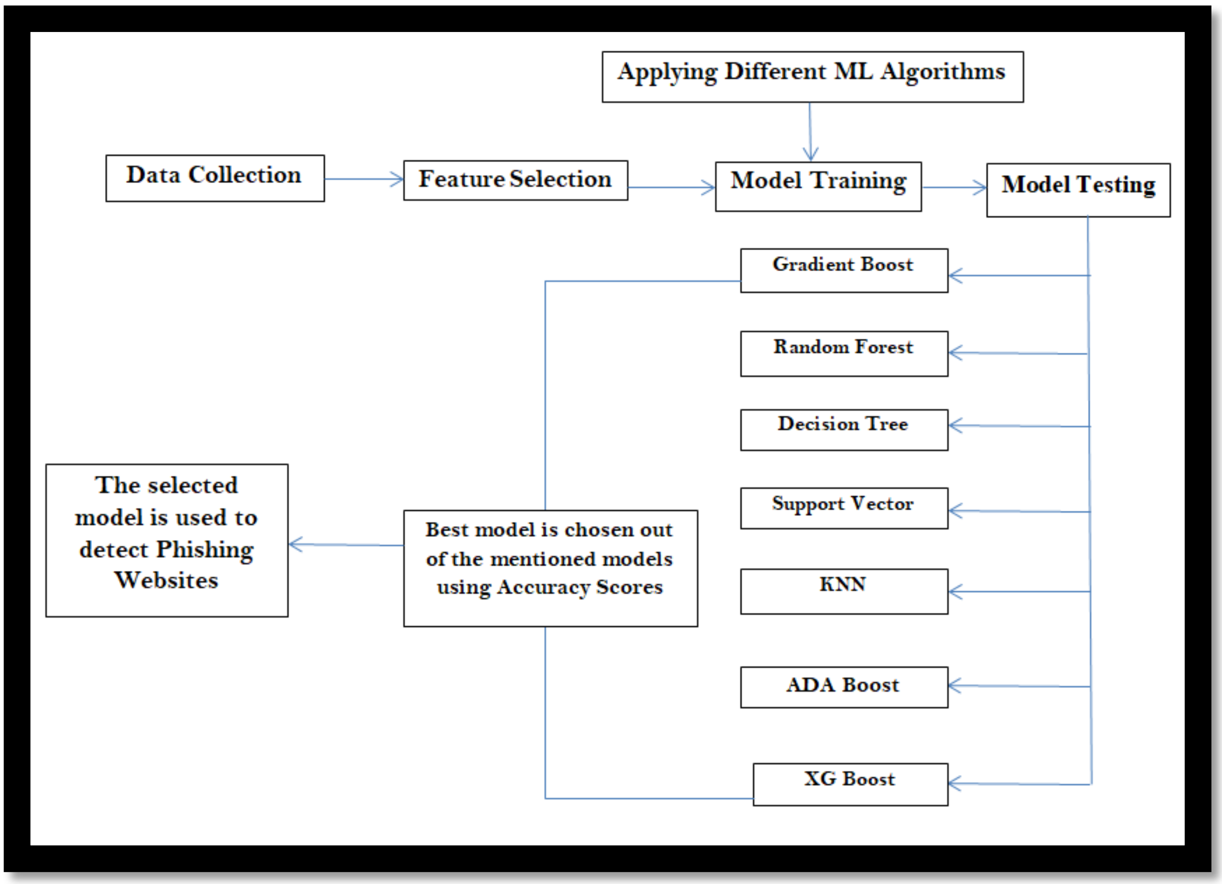

Applying Different ML Algorithms
Data Collection
Feature Selection
Model Training
Model Testing
Gradient Boost
Random Forest
Decision Tree
Support Vector
KNN
ADA Boost
XG Boost
Best model is chosen out of the mentioned models using Accuracy Scores
The selected model is used to detect Phishing Websites

Image 4 : Work flow design

DATASET DESCRIPTION

We used the dataset provided by UCI Machine Learning repository collated by Mohammad et al. The dataset has 11055 data-points with 6157 legitimate URLs and 4898 phishing URLs. Each data-point had 30 features subdivided into following three categories:

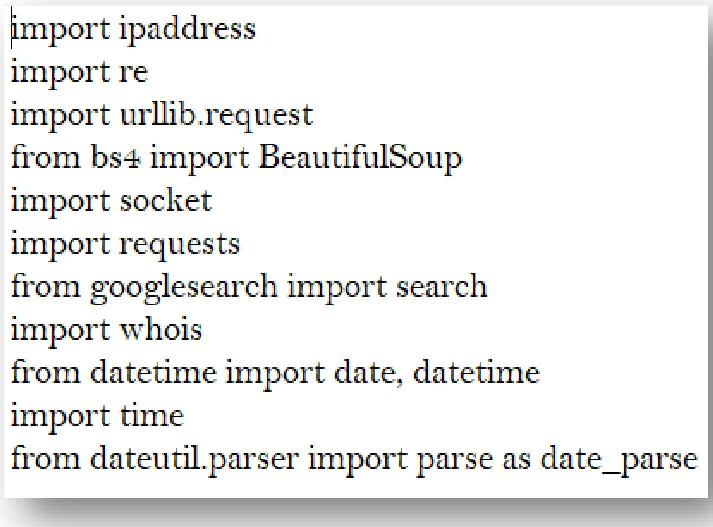
- URL and derived features
- Page's source code based features : Includes URLs embedded in the webpage and HTML and JavaScript based features.
- Domain based features

Studying the way of extraction and relevance of features, we dropped 5 features out of 30, namely: Port Number, Abnormal URL, Pop-up Window, Google Index and Number of Links Pointing to a Page. Port Number was dropped due to feature drift. Rest were dropped due to unavailability of methods to extract them programmatically or absence of public APIs.

IMPLEMENTATION

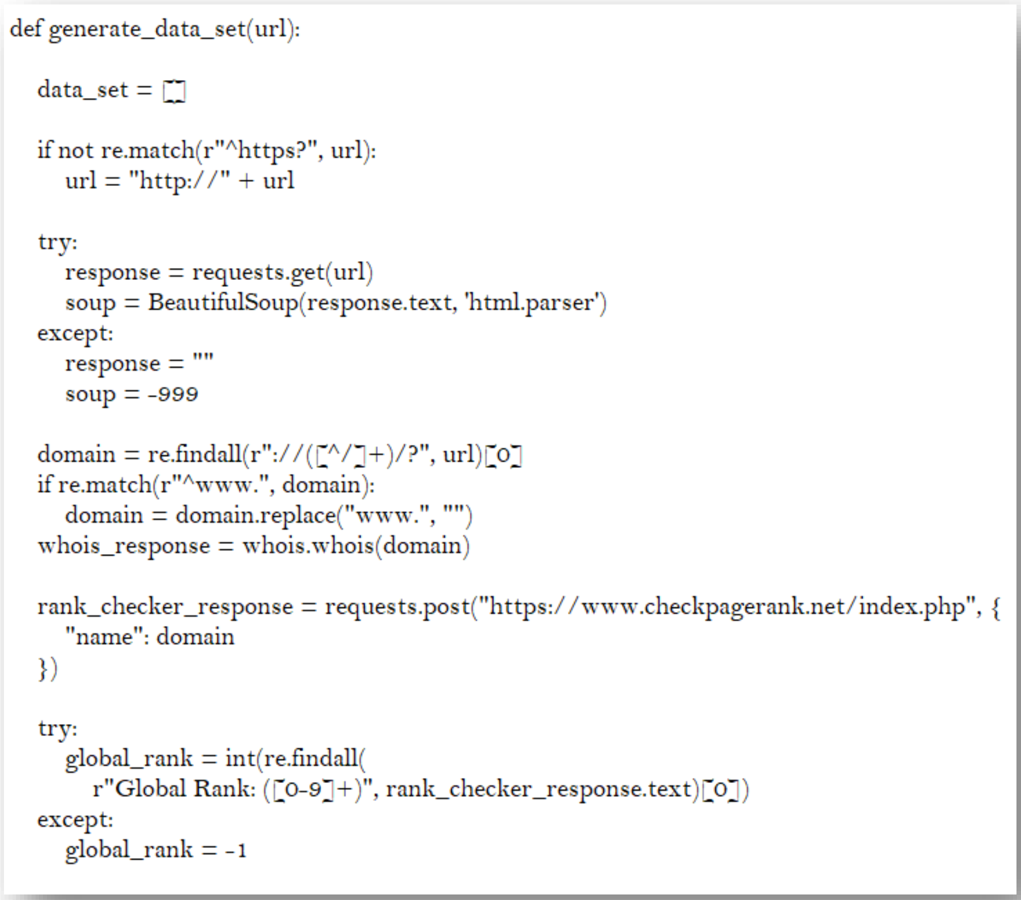
FEATURE EXTRACTION

Firstly, we have imported the required libraries :



```
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
```

Then, we have initialized a function with 'URL' as parameter (generate_data_set(url)) for extracting the features which are necessary for detecting Phishing Websites.



```
def generate_data_set(url):

    data_set = []

    if not re.match(r"^https?", url):
        url = "http://" + url

    try:
        response = requests.get(url)
        soup = BeautifulSoup(response.text, 'html.parser')
    except:
        response = ""
        soup = -999

    domain = re.findall(r"://([^/]+)/?", url)[0]
    if re.match(r"^www.", domain):
        domain = domain.replace("www.", "")
    whois_response = whois.whois(domain)

    rank_checker_response = requests.post("https://www.checkpagerank.net/index.php", {
        "name": domain
    })

    try:
        global_rank = int(re.findall(
            r"Global Rank: ([0-9]+)", rank_checker_response.text)[0])
    except:
        global_rank = -1
```

Those features are mentioned below :

<u>Address Bar - Based Features</u>

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

2. Long URL: to hide the Suspicious Part Phishers can use long URL to hide the doubtful part in the address bar. To ensure accuracy of our study, we calculated the length of URLs in the dataset and produced an average URL length. The results showed that if the length of the URL is greater than or equal 54 characters then the URL classified as phishing.

3. URL Shortening Services: “TinyURL” URL shortening is a method on the “World Wide Web” in which a URL may be made considerably smaller in length and still lead to the

required webpage. This is accomplished by means of an “HTTP Redirect” on a domain name that is short, which links to the webpage that has a long URL.

4. URL’s having “@” Symbol: Using “@” symbol in the URL leads the browser to ignore everything preceding the “@” symbol and the real address often follows the “@” symbol.

5. Redirecting using “//”: The existence of “//” within the URL path means that the user will be redirected to another website. An example of such URL’s is: “http://www.legitimate.com//http://www.phishing.com”. We examine the location where the “//” appears. We find that if the URL starts with “HTTP”, that means the “//” should appear in the sixth position. However, if the URL employs “HTTPS” then the “//” should appear in seventh position.

6. Adding Prefix or Suffix Separated by (-) to the Domain: The dash symbol is rarely used in legitimate URLs. Phishers tend to add prefixes or suffixes separated by (-) to the domain name so that users feel that they are dealing with a legitimate webpage.

7. Sub Domain and Multi Sub Domains: To produce a rule for extracting this feature, we firstly have to omit the (www.) from the URL which is in fact a sub domain in itself. Then, we have to remove the (ccTLD) if it exists. Finally, we count the remaining dots. If the number of dots is greater than one, then the URL is classified as “Suspicious” since it has one sub domain. However, if the dots are greater than two, it is classified as “Phishing” since it will have multiple sub domains. Otherwise, if the URL has no sub domains, we will assign “Legitimate” to the feature.

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

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

10. Favicon: A favicon is a graphic image (icon) associated with a specific webpage. Many existing user agents such as graphical browsers and newsreaders show favicon as a visual reminder of the website identity in the address bar. If the favicon is loaded from a domain other than that shown in the address bar it is considered a Phishing attempt.

11. Non-Standard Port: This feature is useful in validating if a particular service (e.g. HTTP) is up or down on a specific server. In the aim of controlling intrusions, it is much better to merely open ports that you need. Several firewalls, Proxy and Network Address Translation (NAT) servers will, by default, block all or most of the ports and only open the ones selected. If all ports are open, phishers can run almost any service they want and as a result, user information is threatened.

12. "HTTPS" Domain URL: The phishers may add the "HTTPS" token to the domain part of a URL in order to trick users.

<u>Abnormal-Based Features</u>

13. Request URL: It examines whether the external objects contained within a webpage such as images, videos and sounds are loaded from another domain. In legitimate webpages, the webpage address and most of objects embedded within the webpage are sharing the same domain.

14. URL of Anchor: An anchor is an element defined by the tag. This feature is treated exactly as "Request URL". However, for this feature we examine: 1. If the tags and the website have different domain names. This is similar to request URL feature.

15. Links in Script Tags: Given that our investigation covers all angles likely to be used in the webpage source code; we find that it is common for legitimate websites to use tags to offer metadata about the HTML document.

16. Server from Handler: SFHs that contain an empty string or "about:blank" are considered doubtful because an action should be taken upon the submitted information. In addition, if the domain name in SFHs is different from the domain name of the webpage, this reveals

that the webpage is suspicious because the submitted information is rarely handled by external domains.

17. Information to Email: Web form allows a user to submit his personal information that is directed to a server for processing. A phisher might redirect the user's information to his personal email. To that end, a server-side script language might be used such as "mail()" function in PHP. One more client-side function that might be used for this purpose is the "mailto:" function.

18. Abnormal URL: This feature can be extracted from WHOIS database. For a legitimate website, identity is typically part of its URL.

HTML & JavaScript Based Features

19. Website Forwarding: The fine line that distinguishes phishing websites from legitimate ones is how many times a website has been redirected. In our dataset, we find that legitimate websites have been redirected one time max. On the other hand, phishing websites containing this feature have been redirected at least 4 times.

20. Status Bar Customization: Phishers may use JavaScript to show a fake URL in the status bar to users. To extract this feature, we must dig-out the webpage source code, particularly the "onMouseOver" event, and check if it makes any changes on the status bar.

21. Disabling Right Click: Phishers use JavaScript to disable the right-click function, so that users cannot view and save the webpage source code. This feature is treated exactly as "Using onMouseOver to hide the Link". Nonetheless, for this feature, we will search for event "event.button==2" in the webpage source code and check if the right click is disabled.

22. Using Pop-up Window: It is unusual to find a legitimate website asking users to submit their personal information through a pop-up window. On the other hand, this feature has been used in some legitimate websites and its main goal is to warn users about fraudulent activities or broadcast a welcome announcement, though no personal information was asked to be filled in through these pop-up windows.

23. IFrame Redirection: IFrame is an HTML tag used to display an additional webpage into one that is currently shown. Phishers can make use of the "iframe" tag and make it invisible i.e. without frame borders. In this regard, phishers make use of the "frameBorder" attribute which causes the browser to render a visual delineation

<u>Domain Based Features:</u>

24. Age of Domain: This feature can be extracted from WHOIS database (Whois 2005). Most phishing websites live for a short period of time. By reviewing our dataset, we find that the minimum age of the legitimate domain is 6 months.

25. DNS Record: For phishing websites, either the claimed identity is not recognized by the WHOIS database (Whois 2005) or no records founded for the hostname (Pan and Ding 2006). If the DNS record is empty or not found then the website is classified as "Phishing", otherwise it is classified as "Legitimate".

26. Website Traffic: This feature measures the popularity of the website by determining the number of visitors and the number of pages they visit. However, since phishing websites live for a short period of time, they may not be recognized by the Alexa database (Alexa the Web Information Company., 1996). Furthermore, if the domain has no traffic or is not recognized by the Alexa database, it is classified as "Phishing". Otherwise, it is classified as "Suspicious".

27. PageRank: It is a value ranging from "0" to "1". PageRank aims to measure how important a webpage is on the Internet. The greater the PageRank value the more important the webpage. In our datasets, we find that about 95% of phishing webpages have no PageRank. Moreover, we find that the remaining 5% of phishing webpages may reach a PageRank value up to "0.2".

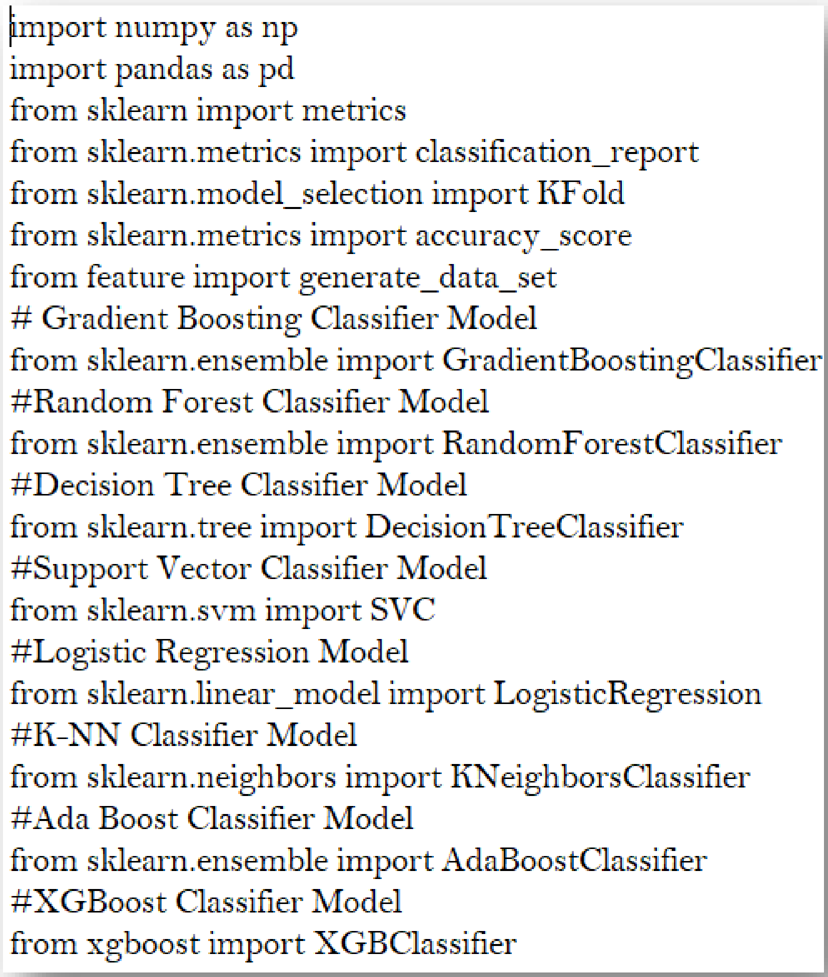
28. Google Index: This feature examines whether a website is in Google's index or not. When a site is indexed by Google, it is displayed on search results (Webmaster resources, 2014). Usually, phishing webpages are merely accessible for a short period and as a result, many phishing webpages may not be found on the Google index.

29. Number of Links Pointing to Page: The number of links pointing to the webpage indicates its legitimacy level, even if some links are of the same domain (Dean, 2014). In our datasets and due to its short life span, we find that 98% of phishing dataset items have no links pointing to them. On the other hand, legitimate websites have at least 2 external links pointing to them.

30. Statistical-Reports: Several parties such as PhishTank (PhishTank Stats, 2010-2012), and StopBadware (StopBadware, 2010-2012) formulate numerous statistical reports on phishing websites at every given period of time; some are monthly and others are quarterly. In our research, we used 2 forms of the top ten statistics from PhishTank: “Top 10 Domains” and “Top 10 IPs” according to statistical-reports published in the last three years, starting in January2010 to November 2012. Whereas for “StopBadware”, we used “Top 50” IP addresses.

IMPLEMENTING VARIOUS MODELS TO COMPARE THE ACCURACY

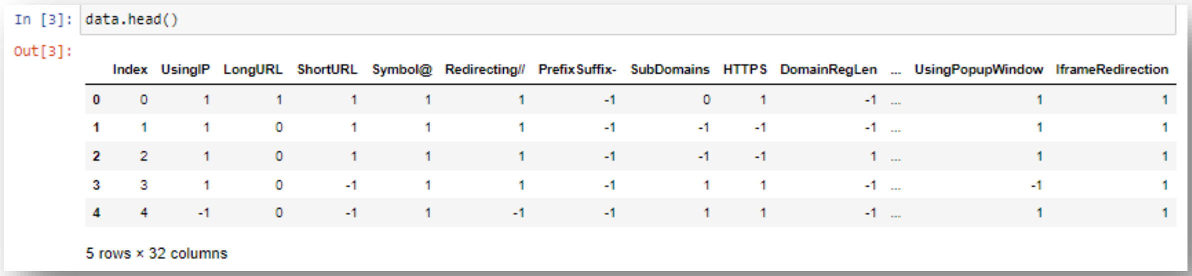
Firstly, we have imported the required libraries:



```
import numpy as np
import pandas as pd
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.model_selection import KFold
from sklearn.metrics import accuracy_score
from feature import generate_data_set
# Gradient Boosting Classifier Model
from sklearn.ensemble import GradientBoostingClassifier
#Random Forest Classifier Model
from sklearn.ensemble import RandomForestClassifier
#Decision Tree Classifier Model
from sklearn.tree import DecisionTreeClassifier
#Support Vector Classifier Model
from sklearn.svm import SVC
#Logistic Regression Model
from sklearn.linear_model import LogisticRegression
#K-NN Classifier Model
from sklearn.neighbors import KNeighborsClassifier
#Ada Boost Classifier Model
from sklearn.ensemble import AdaBoostClassifier
#XGBoost Classifier Model
from xgboost import XGBClassifier
```

Then, we have imported the dataset "phishing.csv"

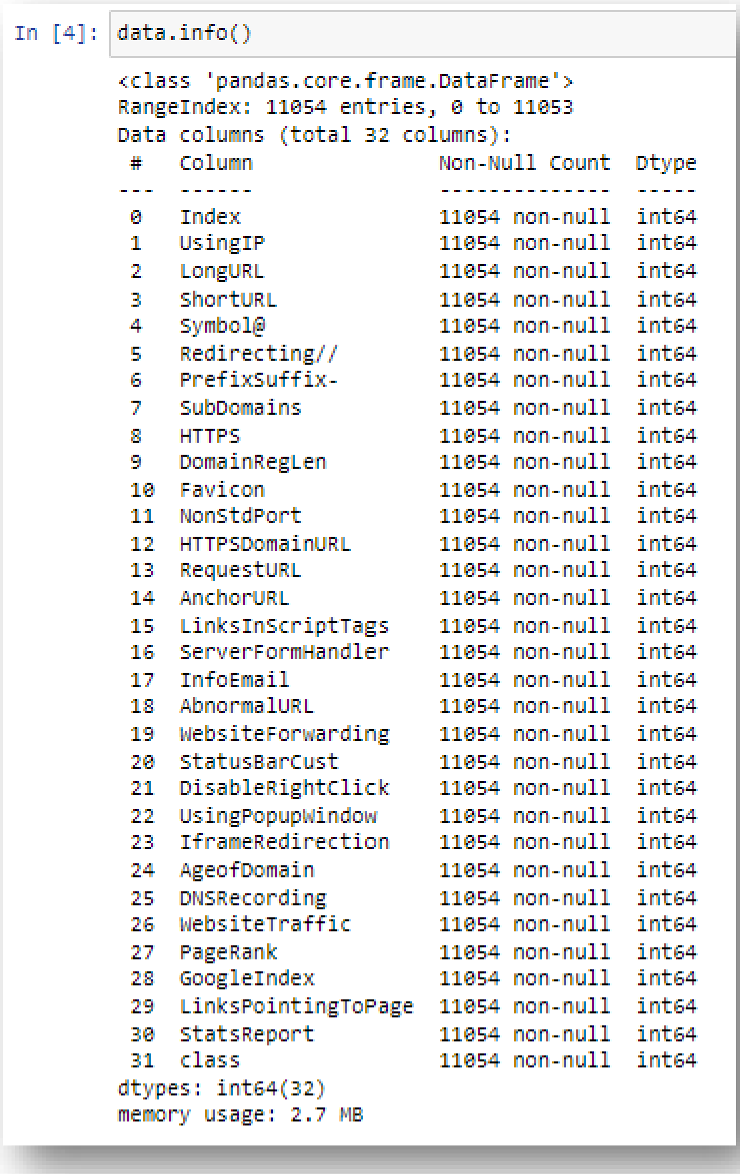
A glimpse at the dataset:



```
In [3]: data.head()
Out[3]:
```

	Index	UsingIP	LongURL	ShortURL	Symbol@	Redirecting//	PrefixSuffix-	SubDomains	HTTPS	DomainRegLen	...	UsingPopupWindow	IframeRedirection
0	0	1	1	1	1	1	-1	0	1	-1	...	1	1
1	1	1	0	1	1	1	-1	-1	-1	-1	...	1	1
2	2	1	0	1	1	1	-1	-1	-1	1	...	1	1
3	3	1	0	-1	1	1	-1	1	1	-1	...	-1	1
4	4	-1	0	-1	1	-1	-1	1	1	-1	...	1	1

5 rows × 32 columns



```
In [4]: data.info()

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 11054 entries, 0 to 11053
        Data columns (total 32 columns):
         #   Column               Non-Null Count  Dtype
        ---  ------               --------------  -----
         0   Index                11054 non-null  int64
         1   UsingIP              11054 non-null  int64
         2   LongURL              11054 non-null  int64
         3   ShortURL             11054 non-null  int64
         4   Symbol@              11054 non-null  int64
         5   Redirecting//        11054 non-null  int64
         6   PrefixSuffix-        11054 non-null  int64
         7   SubDomains           11054 non-null  int64
         8   HTTPS                11054 non-null  int64
         9   DomainRegLen         11054 non-null  int64
         10  Favicon              11054 non-null  int64
         11  NonStdPort           11054 non-null  int64
         12  HTTPSDomainURL       11054 non-null  int64
         13  RequestURL           11054 non-null  int64
         14  AnchorURL            11054 non-null  int64
         15  LinksInScriptTags    11054 non-null  int64
         16  ServerFormHandler    11054 non-null  int64
         17  InfoEmail            11054 non-null  int64
         18  AbnormalURL          11054 non-null  int64
         19  WebsiteForwarding    11054 non-null  int64
         20  StatusBarCust        11054 non-null  int64
         21  DisableRightClick    11054 non-null  int64
         22  UsingPopupWindow     11054 non-null  int64
         23  IframeRedirection    11054 non-null  int64
         24  AgeofDomain          11054 non-null  int64
         25  DNSRecording         11054 non-null  int64
         26  WebsiteTraffic       11054 non-null  int64
         27  PageRank             11054 non-null  int64
         28  GoogleIndex          11054 non-null  int64
         29  LinksPointingToPage  11054 non-null  int64
         30  StatsReport          11054 non-null  int64
         31  class                11054 non-null  int64
        dtypes: int64(32)
        memory usage: 2.7 MB
```

Then , we have dropped the index column as it doesn't have any impact on predicting the accuracy of the model and also we have split our data into dependant and independent features.

Then , since our data is small so we have used K-Fold Cross Validation to evaluate Machine Learning Models.

Then , we have implemented different Machine Learning Classifier Models:

Gradient Boost Classifier

Random Forest Classifier

Decision Tree Classifier

Support Vector Classifier

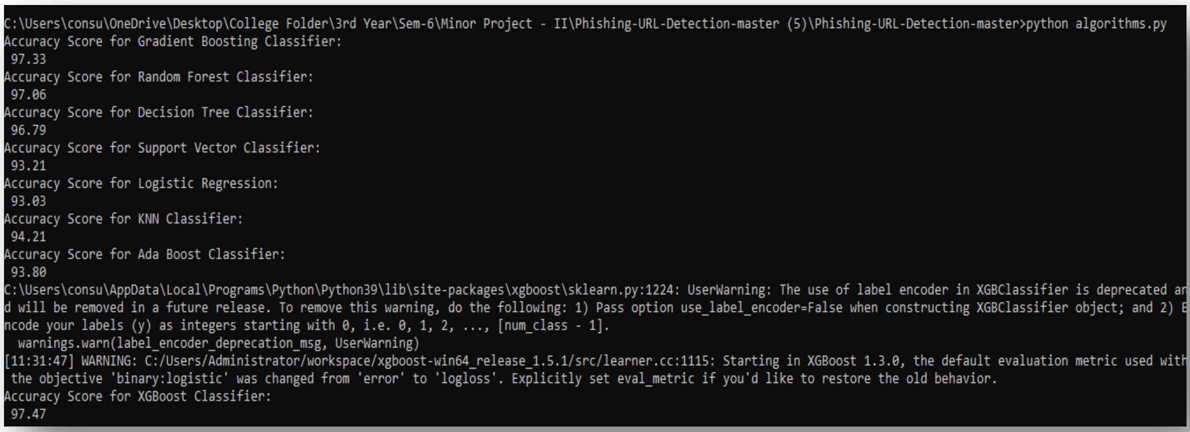
KNN Classifier

Logistic Regression

ADA Boost Classifier

XG Boost Classifier

Comparison of Accuracy Scores of above mentioned models:



```
C:\Users\consu\OneDrive\Desktop\College Folder\3rd Year\Sem-6\Minor Project - II\Phishing-URL-Detection-master (5)\Phishing-URL-Detection-master>python algorithms.py
Accuracy Score for Gradient Boosting Classifier:
 97.33
Accuracy Score for Random Forest Classifier:
 97.06
Accuracy Score for Decision Tree Classifier:
 96.79
Accuracy Score for Support Vector Classifier:
 93.21
Accuracy Score for Logistic Regression:
 93.03
Accuracy Score for KNN Classifier:
 94.21
Accuracy Score for Ada Boost Classifier:
 93.80
C:\Users\consu\AppData\Local\Programs\Python\Python39\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated an
d will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) E
ncode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[11:31:47] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.1/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with
 the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
Accuracy Score for XGBoost Classifier:
 97.47
```

Image 5 : Accuracy scores

CREATING A WEBSITE TO DETECT PHISHING URL

We have used HTML for designing the website and used css and javascript for styling and functionality

INTEGRATING MACHINE LEARNING MODEL WITH WEBSITE

Since, clearly XG Boost Classifier serves the highest accuracy of 97.47% outperforming Gradient Boosting Classifier and Decision Tree Classifier as well. So, therefore we will use XG Boost Classifier for detecting phishing websites.

Implemented xgboost classifier

```
# instantiate the model

xgb = XGBClassifier(random_state=0)

xgb.fit(X_train, y_train)
```

Integrating website

```
@app.route("/")
```

```
def index():
    return render_template("index.html", xx= -1)
@app.route("/predict", methods=["GET", "POST"])
def predict():
    if request.method == "POST":
```

RESULTS

We have created a website to test if the 'URL' is fake or legitimate and then integrating our Machine Learning Model with the website.

Here, are some screenshots of the work done:

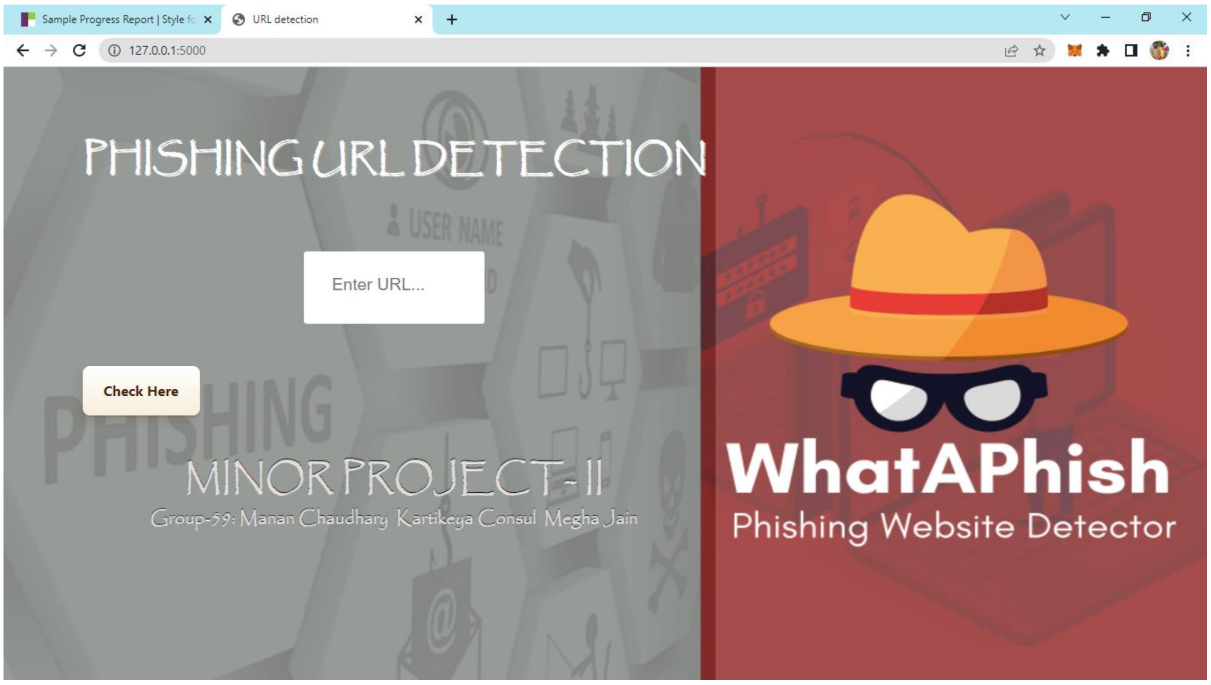

Sample Progress Report | Style fo
URL detection
127.0.0.1:5000
PHISHING URL DETECTION
Enter URL...
Check Here
MINOR PROJECT - II
Group-59: Manan Chaudhary Kartikeya Consul Megha Jain
WhatAPhish
Phishing Website Detector

Image 6 : Website Interface

Checking a Non-Phished URL: “linkedin.com”

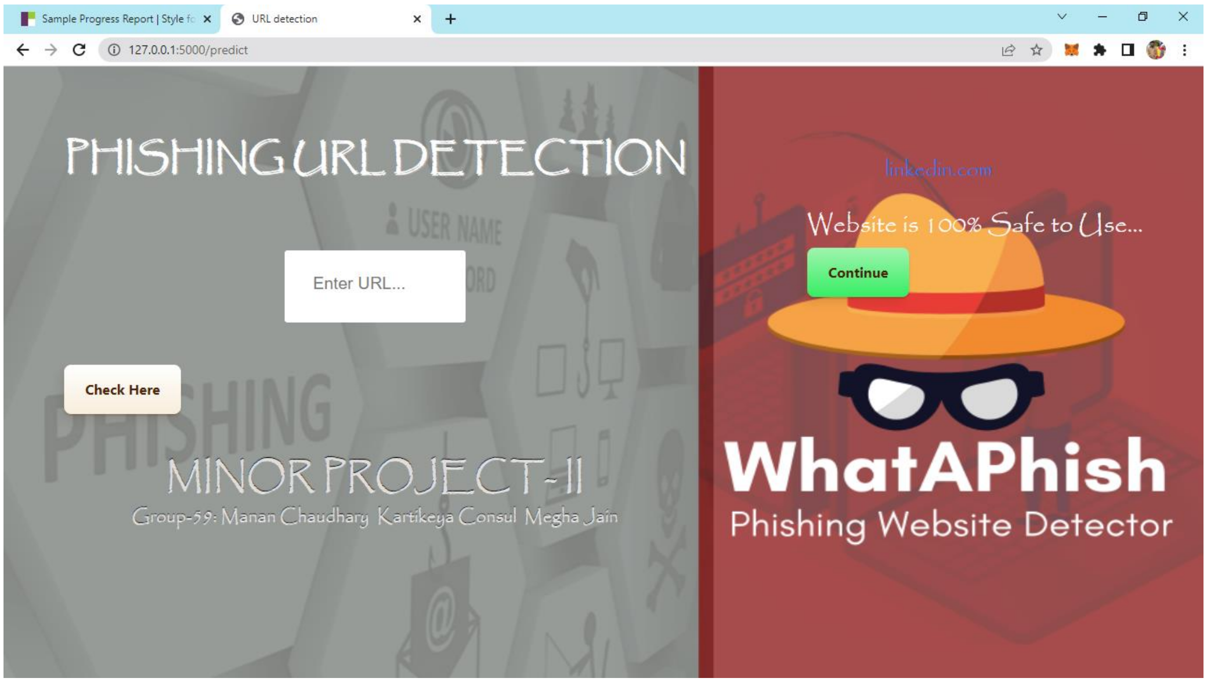

Sample Progress Report | Style fo
URL detection
127.0.0.1:5000/predict
PHISHING URL DETECTION
Enter URL...
Check Here
MINOR PROJECT-II
Group-59: Manan Chaudhary Kartikeya Consul Megha Jain
linkedin.com
Website is 100% Safe to Use...
Continue
WhatAPhish
Phishing Website Detector

Image 7 : Result for legitimate Website

Checking a Phished URL: “https://xavier-net.gq/?login=do”

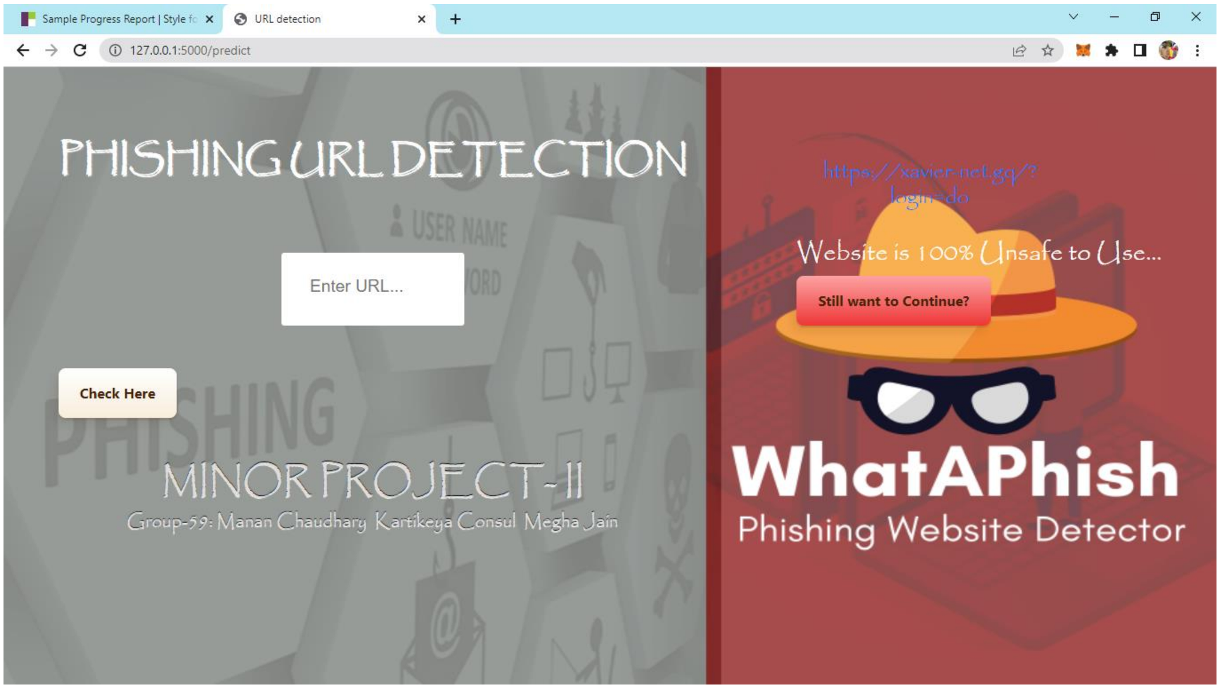

Sample Progress Report | Style fo
URL detection
127.0.0.1:5000/predict
PHISHING URL DETECTION
Enter URL...
Check Here
MINOR PROJECT-II
Group-59: Manan Chaudhary Kartikeya Consul Megha Jain
https://xavier-net.gq/?login=do
Website is 100% Unsafe to Use...
Still want to Continue?
WhatAPhish
Phishing Website Detector

Image 8 : Result for phishing website

FUTURESCOPE

The research work presented here has some limitations and it can be extended further. The first limitation is that although there are 30 features for each URL but we used features that were already extracted from URLs, features need to be discrete. Classifiers are more suitable when features are discrete. We can evaluate classifiers using a large data set and extract more number of features that may be significant in decision making. In order to avoid the problem of overfitting a classifier, we need to include a pre-process stage. In processing, we can use clustering to find out outliers or noisy data samples. Such samples should not be used in the training set data.

For future enhancements, we intend to build the phishing detection system as a scalable web service which will incorporate online learning so that new phishing attack patterns can easily be learned and improve the accuracy of our models with better feature extraction.

CONCLUSION

Our project's aim was to compare pre-defined different machine learning classifiers and then select the best algorithm based on their performance. The ML based phishing techniques depend on website functionalities to gather information that can help classify websites for detecting phishing sites. The problem of phishing cannot be eradicated, none the-less can be reduced by combating it in two ways, improving targeted anti-phishing procedures and techniques and informing the public on how fraudulent phishing websites can be detected and identified. To combat the ever evolving and complexity of phishing attacks and tactics, ML anti-phishing techniques are essential. We have detected phishing websites using XG Boost Classifier with the accuracy of 97.47%. Then, we built a Chrome extension for detecting phishing web pages. The extension allows easy deployment of our phishing detection model to end users. The future direction of this study is to develop an unsupervised deep learning method to generate insight from a URL.

REFERENCES

[1] https://link.springer.com/chapter/10.1007/978-981-10-3932-4_3

[2] https://ieeexplore.ieee.org/document/7881507

[3] https://ieeexplore.ieee.org/document/6920759

[4] N. Sadeh, A. Tomasic, and I Fette, “Learning to detect phishing emails”, Proceedings ofthe16thinternational conference on world wide web, pp.649–656, 2007.

[5] W. Fadheel, M. Abusharkh, and I. Abdel-Qader, “On Feature Selection for the Prediction of Phishing Websites,” 2017 IEEE 15th Intl Conf Dependable, Auton. Secur. Comput. 15th Intl Conf Pervasive Intell. Comput. 3rd Intl Conf Big Data Intell. Comput. Cyber Sci. Technol. Congr., pp. 871–876, 2017.

[6] https://www.nvidia.com/en-us/glossary/data-science/xgboost/