PHISHING WEBSITE DETECTION USING MACHINE LEARNING AIGORITHM

Submitted in partial fulfilment of the requirements for the
award of
Bachelor of Engineering degree in Computer Science and Engineering

By

MANOJ SV (38110687)

RISHI KUMAR V (38110681)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119

MARCH - 2022

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENTOF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **MANOJ SV (38110687)** and **RISHI KUMAR V (38110681)** who carried out the project entitled "**PHISHING WEBSITE DETECTION USING MACHINE LEARNING ALGORITHMS**" under my supervision from November 2021 to March 2022.

Internal Guide

Dr. S. L. JANY SHABU M.Phil., Ph.D.,

Head of the Department

Dr. S. VIGNESWARI M.E., Ph.D.,

Submitted for Viva voce Examination held on_____

Internal Examiner **External Examiner**

DECLARATION

We **RISHI KUMAR V** and **MANOJ SV** hereby declare that the Project Report entitled **"PHISHING WEBSITE DETECTION USING MACHINE LEARNING ALGORITHMS"** done by me under the guidance of **Dr. S. L. JANY SHABU M.Phil., Ph.D.,** and Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA INSTITUTE OF SCIENCE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.SASIKALA M.E, Ph.D., Dean, School of Computing** and **Dr. S.VIGNESHWARI M.E., Ph.D., and Dr. L.LAKSHMANAN, M.E., PhD., Head of the Department, Dept. of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S. L. JANY SHABU M.Phil., Ph.D.,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Phishing websites have proven to be a major security concern. Several cyberattacks risk the confidentiality, integrity, and availability of company and consumer data, and phishing is the beginning point for many of them. Many researchers have spent decades creating unique approaches to automatically detect phishing websites. While cutting-edge solutions can deliver better results, they need a lot of manual feature engineering and aren't good at identifying new phishing attacks. As a result, finding strategies that can automatically detect phishing websites and quickly manage zero-day phishing attempts is an open challenge in this field. The web page in the URL which hosts that contains a wealth of data that can be used to determine the web server's maliciousness. Machine Learning is an effective method for detecting phishing.It also eliminates the disadvantages of the previous method. We conducted a thorough review of the literature and suggested a new method for detecting phishing websites using features extraction and a machine learning algorithm. The goal of this research is to use the dataset collected to train ML models and deep neural nets to anticipate phishing websites.

TABLE OF CONTENTS

1. INTRODUCTION

Phishing has become the most serious problem, harming individuals, corporations, and even entire countries. The availability of multiple services such as online banking, entertainment, education, software downloading, and social networking has accelerated the Web's evolution in recent years. As a result, a massive amount of data is constantly downloaded and transferred to the Internet. Spoofed emails pretending to be from reputable businesses and agencies are used in social engineering techniques to direct consumers to fake websites that deceive users into giving financial information such as usernames and passwords. Technical tricks involve the installation of malicious software on computers to steal credentials directly, with systems frequently used to intercept users' online account usernames and passwords.

1.1 TYPES OF PHISHING

• Deceptive Phishing: This is the most frequent type of phishing assault, in which a Cyber criminal impersonates a well-known institution, domain, or organization to acquire sensitive personal information from the victim, such as login credentials, passwords, bank account information, credit card information, and so on. Because there is no personalization or customization for the people, this form of attack lacks sophistication.

• Spear Phishing: Emails containing malicious URLs in this sort of phishing email contain a lot of personalization information about the potential victim. The recipient's name, company name, designation, friends, co-workers, and other social information may be included in the email.

• Whale Phishing: To spear phish a "whale," here a top-level executive such as CEO, this sort of phishing targets corporate leaders such as CEOs and top-level management employees.

• URL Phishing: To infect the target, the fraudster or cyber-criminal employs a URL link. People are sociable creatures who will eagerly click the link to accept friend

invitations and may even be willing to disclose personal information such as email addresses.

This is because the phishers are redirecting users to a false web server. Secure browser connections are also used by attackers to carry out their unlawful actions. Due to a lack of appropriate tools for combating phishing attacks, firms are unable to train their staff in this area, resulting in an increase in phishing attacks.Companies are educating their staff with mock phishing assaults, updating all their systems with the latest security procedures, and encrypting important Information as broad countermeasures. Browsing without caution is one of the most common ways to become a victim of this phishing assault. The appearance of phishing websites is like that of authentic websites.

1.3 EXISTING SYSTEM

Anti-phishing strategies involve educating netizens and technical defense. In this paper, we mainly review the technical defense methodologies proposed in recent years. Identifying the phishing website is an efficient method in the whole process of deceiving user informationAlong with the development of machine learning techniques, various machine learning based methodologies have emerged for recognizing phishing websites to increase the performance of predictions.The primary purpose of this paper is to survey effective methods to prevent phishing attacks in a real-time environment

1.4 PROPOSED SYSTEM

The most frequent type of phishing assault, in which a cybercriminal impersonates a well-known institution, domain, or organization to acquire sensitive personal information from the victim, such as login credentials, passwords, bank account information, credit card information, and so on.Emails containing malicious URLs in this sort of phishing email contain a lot of personalization information about the potential victim.To spear phish a "whale," here a top-level executive such as CEO, this sort of phishing targets corporate leaders such as CEOs and top-level management employeesTo infect the target, the fraudster or cyber-criminal employs a URL link.

1.5 ADVANTAGES

there is no personalization or customization for the people, this form of attack lacks sophistication.social information may be included in the email.The recipient's name, company name, designation, friends, co-workers may be missing click the link to accept friend invitations and may even have other people information.

2. LITERATURE REVIEW

Many scholars have done some sort of analysis on the statistics of phishing URLs. Our technique incorporates key concepts from past research. We review past work in the detection of phishing sites using URL features, which inspired our current approach. Happy describe phishing as "one of the most dangerous ways for hackers to obtain users' accounts such as usernames, account numbers and passwords, without their awareness." Users are ignorant of this type of trap and will ultimately, they fall into Phishing scam. This could be due to a lack of a combination of financial aid and personal experience, as well as a lack of market awareness or brand trust. In this article, Mehmet et al. suggested a method for phishing detection based on URLs. To compare the results, the researchers utilized eight different algorithms to evaluate the URLs of three separate datasets using various sorts of machine learning methods and hierarchical architectures. The first method evaluates various features of the URL; the second method investigates the website's authenticity by determining where it is hosted and who operates it; and the third method investigates the website's graphic presence.We employ Machine Learning techniques and algorithms to analyse these many properties of URLs and websites. Garera et al. classify phishing URLs using logistic regression over hand-selected variables. The inclusion of red flag keywords in the URL, as well as features based on Google's Web page and Google's Page Rank quality recommendations, are among the features. Without access to the same URLs and features as our approach, it's difficult to conduct a direct comparison.

In this research, Yong et al. created a novel approach for detecting phishing websites that focuses on detecting a URL which has been demonstrated to be an accurate and efficient way of detection. To offer you a better idea, our new capsule-based neural network is divided into several parallel components. One method involves removing shallow characteristics from URLs. The other two, on the other hand, construct accurate feature representations of URLs and use shallow features to evaluate URL legitimacy. The final output of our system is calculated by adding the outputs of all divisions. Extensive testing on a dataset collected from the Internet indicate that our system can compete with other cutting-edge detection methods

while consuming a fair amount of time. For phishing detection, Vahid Shahrivari et al. used machine learning approaches. They used the logistic regression classification method, KNN, Adaboost algorithm, SVM, ANN and random forest. They found random forest algorithm provided good accuracy. Dr.G. Ravi Kumar used a variety of machine learning methods to detect phishing assaults. For improved results, they used

NLP tools. They were able to achieve high accuracy using a Support Vector Machine and data that had been pre-processed using NLP approaches. Amani Alswailem et al. tried different machine learning model for phishing detection but was able to achieve more accuracy in random forest. Hossein et al. created the “Fresh-Phish” open-source framework. This system can be used to build machine-learning data for phishing websites. They used a smaller feature set and built the query in Python. They create a big, labelled dataset and test several machine-learning classifiers on it. Using machine-learning classifiers, this analysis yields very high accuracy. These studies look at how long it takes to train a model. X. Zhang suggested a phishing detection model based on mining the semantic characteristics of word embedding, semantic feature, and multi-scale statistical features in Chinese web pages to detect phishing performance successfully. To obtain statistical aspects of web pages, eleven features were retrieved and divided into five classes. To obtain statistical aspects of web pages, eleven features were retrieved and divided into five classes. To learn and evaluate the model, AdaBoost, Bagging, Random Forest, and SMO are utilized. The legitimate URLs dataset came from DirectIndustry online guides, and the phishing data came from China's Anti-Phishing Alliance. With novel methodologies, M. Aydin approaches a framework for extracting characteristics that is versatile and straightforward. Phish Tank provides data, and Google provides authentic URLs. C# programming and R programming were utilized to btain the text attributes. The dataset and third-party service providers yielded a total of 133 features. The feature selection approaches of CFS subset based and Consistency subset-based feature selection were employed and examined with the WEKA tool. The performance of the Nave Bayes and Sequential Minimal Optimization (SMO) algorithms was evaluated, and the author prefers SMO to NB for phishing detection.

3. METHODOLOGY

A phishing website is a social engineering technique that imitates legitimate webpages and uniform resource locators (URLs). The Uniform Resource Locator (URL) is the most common way for phishing assaults to occur. Phisher has complete control over the URL's sub-domains. The phisher can alter the URL because it contains file components and directories.This research used the linear-sequential model, often known as the waterfall model. Although the waterfall approach is considered conventional, it works best in instances where there are few requirements. The application was divided into smaller components that were built using frameworks and hand-written code.

3.1 RESEARCH FRAMEWORK

the steps of this research in which some selected publications were read to determine the research gap and, as a result, the research challenge was defined. Feature selection, classification and phishing website detection were all given significant consideration. It's worth noting that most phishing detection researchers rely on datasets they've created. However, because the datasets utilized were not available online for those who use and check their results, it is difficult to assess and compare the performance of a model with other models. As a result, such results cannot be generalized. For the preparation of this dissertation, I used Python as the primary language. Python is a language that is heavily focused on machine learning. It includes several machine learning libraries that may be utilized straight from an import. Python is commonly used by developers all around the world to deal with machine learning because of its extensive library of machine learning libraries. Python has a strong community, and as a result, new features are added with each release.

Data Collection

The phishing URLs were gathered using the open source tool Phish Tank. This site provides a set of phishing URLs in a variety of forms, including csv, json, and others, which are updated hourly. This dataset is used to train machine learning models with 5000 random phishing URLs.

Data Cleaning

Fill in missing numbers, smooth out creaking data, detect and delete outliers, and repair anomalies to clean up the data.

Data Pre-processing

Data preprocessing is a cleaning operation that converts unstructured raw data into a neat, well-structured dataset that may be used for further research. Data preprocessing is a cleaning operation that transforms unstructured raw data into well-structured and neat dataset which can be used for further research.

The data is split into 8000 training samples and 2000 testing samples, before the ML model is trained. It is evident from the dataset that this is a supervised machine learning problem. Classification and regression are the two main types of supervised machine learning issues. Because the input URL is classed as legitimate or phishing, this data set has a classification problem. The following supervised machine learning models were examined for this project's dataset training: Decision Tree , Multilayer Perceptron, Random Forest,Autoencoder Neural Network , XGBoost and Support Vector Machines.

3.2 ADDRESS BASED CHECKING

Below are the categories been extracted from address based

1. Domain of the URL :Where domain which is present in the URL been extracted

2. IP Address in the URL :The presence of an IP address in the URL is checked. Instead of a domain name, URLs may contain an IP address. If an IP address is used instead of a domain name in a URL, we can be certain that the URL is being used to collect sensitive information.

3. "@" Symbol in URL :The presence of the'@' symbol in the URL is checked. When the “@” symbol is used in a URL, the browser ignores anything before the “@” symbol, and the genuine address is commonly found after the “@” symbol.

4. Length of URL :Calculates the URL's length. Phishers can disguise the suspicious element of a URL in the address bar by using a lengthy URL. If the length of the URL is larger than or equal to 54 characters, the URL is classed as phishing in this project.

5. Depth of URL :Calculates the URL's depth. Based on the'/', this feature determines the number of subpages in the given address.

6. Redirection "//" in URL :The existence of"//" in the URL is checked. The presence of the character"//" in the URL route indicates that the user will be redirected to another website. The position of the"//" in the URL is calculated. We discovered that if the URL begins with “HTTP,” the “//” should be placed in the sixth position. If the URL uses “HTTPS,” however, the “//” should occur in the seventh place.

7. Http/Https in Domain name :The existence of "http/https" in the domain part of the URL is checked. To deceive users, phishers may append the “HTTPS” token to the domain section of a URL.

8. Using URL Shortening Services :URL shortening is a means of reducing the length of a URL while still directing to the desired webpage on the "World Wide Web." This is performed by using a “HTTP Redirect” on a short domain name that points to a webpage with a long URL.

9. Prefix or Suffix "-" in Domain :Checking for the presence of a '-' in the URL's domain part. In genuine URLs, the dash symbol is rarely used. Phishers frequently append prefixes or suffixes to domain names, separated by (-), to give the impression that they are dealing with a legitimate website.

3.3 DOMAIN BASED CHECKING

This category contains a lot of features that can be extracted. This category contains a lot of features that can be extracted. The following were considered for this project out of all of them.

1. DNS Record :In the case of phishing websites, the WHOIS database either does not recognize the stated identity or there are no records for the host name .

2. Web Traffic :This function determines the number of visitors and the number of pages they visit to determine the popularity of the website. In the worst-case circumstances, legitimate websites placed among the top100,000, according to our data. Furthermore, it is categorized as "Phishing" if the domain has no traffic or is not recognized by the Alexa database.

3. Age of Domain :This information can be retrieved from the WHOIS database. Most phishing websites are only active for a short time. For this project, the minimum age of a legal domain is deemed to be 12 months. Age is simply the difference between the time of creation and the time of expiry.

4. End Period of Domain :This information can be gleaned from the WHOIS database. The remaining domain time is calculated for this feature by determining the difference between the expiry time and the current time. For this project, the valid domain's end time is regarded to be 6 months or fewer.

3.4 HTML AND JAVASCRIPT BASED CHECKING

Many elements that fall within this group can be extracted. The following were considered for this project out of all of them.

1. IFrame Redirection :IFrame is an HTML tag that allows you to insert another webpage into the one you're now viewing. The “iframe” tag can be used by phishers to make the frame invisible, i.e., without frame borders. Phishers employ the “frame border” attribute in this case, which causes the browser to create a visual boundary.

2. Status Bar Customization :Phishers may utilize JavaScript to trick visitors into seeing a false URL in the status bar. To get this feature, we'll need to delve into the webpage source code, specifically the "on Mouseover" event, and see if it alters the status bar.

3. Disabling Right Click :Phishers disable the right-click function with JavaScript, preventing users from viewing and saving the webpage source code. This functionality is handled in the same way as "Hiding the Link with on Mouseover." Nonetheless,we'll look for the ``event"event. button==2" in the webpage source code and see if the right click is disabled for this functionality.

4. Website Forwarding :The number of times a website has been redirected is a narrow line that separates phishing websites from authentic ones. We discovered that authentic websites were only routed once in our sample. Phishing websites with this functionality, on the other hand, have been redirected at least four times.

5. Implementation :We'll examine the implementation component of our artefact in this area of the report, with a focus on the description of the developed solution. This is a task that requires supervised machine learning.

List-Based Approaches:Jain and Gupta proposed an auto-updated, whitelist-based approach to protect against phishing attacks on the client side in 2016. The experimental results demonstrate that it achieved 86.02% accuracy and less than a 1.48% false-positive rate, which indicates a false warning for phishing attacks. The other benefit of this approach is fast access time, which guarantees a real-time environment and products.

Heuristic Strategies: Tan et al. introduced a phishing detection approach named PhishWHO, which consists of three phases. First, it obtains identity keywords by a weighted URL token system and ensembles the N-gram model from the page's HTML. Secondly, it puts the keywords into mainstream search engines to find the legitimate website and the legal domain. Next, it compares the legal domain and the target website's domain to determine if the target website is a phishing website or not. Chiew et al. used a logo image from the website to distinguish if the website was legal. In this paper, the authors extracted a logo from web page images by some machine learning algorithms and then queried the domain via the Google search engine with a logo as a keyword. Therefore, some researchers also called this category search engine-based approach.

Machine Learning-Based Methods: Machine learning-based countermeasures are proposed to address dynamic phishing attacks with higher accuracy performance and lower false positive rates than other methods. Consequently, the machine learning approach consists of six components: data collection, feature extraction, model training, model testing, and predicting.The flowchart of each part. Existing machine

learning-based phishing website detection solutions are based on this flowchart to optimize one or more parts to obtain better performance.

Tiny URL Detection:Since tiny URLs do not present the real domain, resource direction, or search parameters, rule-based feature selection techniques might be useless for tiny URLs. Due to tiny URLs generated by different services, it is hard to convert them to original URLs. Furthermore, tiny URLs are short strings that are unfriendly for natural language processing to extract character-level features. If tiny URLs are not specially processed during data cleansing and preprocessing, they are likely to cause false or missed alarms. Internet products are also essential in terms of user experience, and users are also sensitive to false alarms of Internet security products. 6.4. Response Time for Real-Time Systems Rule-based models depend on rule parsing and third-party services from a URL string. Therefore, they demand a relatively long response time in a real-time prediction system that accepts a single URL string as an input in each request from a client. Phishing attacks spread to various communication media and target devices, such as personal computers and other smart devices. It is a big challenge for developers to cover all devices with one solution. Language independence and running environment independence should be taken into consideration to reduce system development complexity and late maintenance costs.

3.5 DATASET

We collected the datasets from the open-source platform called Phishing tank. The dataset that was collected was in csv format. There are 18 columns in the dataset, and we transformed the dataset by applying data pre-processing technique. To see the features in the data we used few of the data frame methods for familiarizing. For

visualization, and to see how the data is distributed and how features are related to one another, a few plots and graphs are given. The Domain column has no bearing on the training of a machine learning model. We now have 16 features and a target column. The recovered features of the legitimate and phishing URL datasets are simply concatenated in the feature extraction file, with no shuffling. We need to shuffle the data to balance out the distribution while breaking it into training and testing sets. This also eliminates the possibility of over fitting during model training.

3.6 MACHINE LEARNING MODELS

Decision Tree Classifier :For classification and regression applications, decision trees are commonly used models. They basically learn a hierarchy of if/else questions that leads to a choice. Learning a decision tree is memorizing the sequence of if/else questions that leads to the correct answer in the shortest amount of time. The method runs through all potential tests to discover the one that is most informative about the target variable to build a tree.

Random Forest Classifier :Random forests are one of the most extensively used machine learning approaches for regression and classification. A random forest is just a collection of decision trees, each somewhat different from the others. The notion behind random forests is that while each tree may do a decent job of predicting, it will almost certainly overfit on some data. They are incredibly powerful, frequently operate effectively without a lot of parameters adjusting, and don't require data scalability.

3.7 LIBRARIES USED

Pandas:It's a Python-based machine learning library. Pandas is a free and open-source programming language. Pandas is a programming language that is commonly used for

dataset loading and data analytics. Pandas is used for machine learning in a variety of domains, including economics, finance, and others. It is extremely user-friendly and can display datasets in a tabular style for easier comprehension.

Sklearn: Sklearn is one of the most essential Python libraries for machine learning. Sklearn includes several tools for statistical classification, modelling, regression, dimensionality reduction and clustering.

Numpy: Numpy is a Python-based machine learning package. In Python, Numpy is used to deal with arrays. NumPy is used for all calculations using 1-d or 2-d arrays. Numpy also has routines for working with linear algebra and the Fourier transform.

MAPTlotlib: MAPTlotlib is a library for data visualization. It's a Python open-source module for plotting graphs from model results. These diagrams can aid in comprehending the circumstance of the outcomes. For easier comprehension, several components of the results can be graphically formatted.

3.8 EVALUATION

In this section, we use different models of machine learning for evaluating the accuracy. It has been explained about the different models in below sections. Where in this project the models are examined, with accuracy as the primary metric. In final stage we have compared the model accuracy. In all circumstances the testing and training datasets are splinted into 20:80 ratio.

Experiment 1/ Feature Distribution :Here in below figure shows how the data is distributed and how features are related to one another, a few plots and graphs are given.

Experiment 2/ Decision Tree Classifier :The method runs through all potential tests to discover the one that is most informative about the target variable to build a tree.

Where we are predicting the accuracy of the model on the samples collected on both trained and test samples. On this we found accuracy of test and training datasets are 82.6% and 81%. Below is the execution of Decision tree classifier algorithm. To generate model various parameters are set and the model is fitted in the tree. The samples are divided into X and Y train, X and Y test to check the accuracy of the model.

Experiment 3/ Random Forest Classifier :We can limit the amount of over fitting by averaging the outcomes of numerous trees that all operate well and over fit in diverse ways. To construct a random forest model, you must first determine the number of trees to construct. They are incredibly powerful, frequently operate effectively without a lot of parameters adjusting, and don't require data scalability. Where we are predicting the accuracy of the model on the samples collected on both trained and test samples. On this we found accuracy of test and training datasets are 83.4% and 81.4%.

Experiment 4/ MLP :MLPs can be thought of as generalized linear models that go through numerous phases of processing before deciding. Below is the execution of the MLP algorithm. To generate a model various parameters are set and the model is fitted in the tree. The samples are divided into X and Y train, X and Y test to check the accuracy of the model. Where we are predicting the accuracy of the model on the samples collected on both trained and test samples. On this we found accuracy of test and training datasets are 86.3% and 85.9%.

Experiment 5/ XGBoost :Below is the execution of XGBoost algorithm. To generate a model various parameters are set and the model is fitted in the tree. The samples are divided into X and Y trains, X and Y tests to check the accuracy of the model. Where we are predicting the accuracy of the model on the samples collected on both trained and test samples. On this we found accuracy of test and training datasets are 86.4% and 86.6%.

Experiment 6/ Auto encoder :The auto-encoder must learn to encode the input to the hidden neurons with fewer neurons. In an auto encoder, the predictors (x) and output (y) are identical. To generate model various parameters are set and the model is fitted in the tree. The samples are divided into X and Y train, X and Y test to check the accuracy of the model. Where we are predicting the accuracy of the model on the samples collected on both trained and test samples. On this we found accuracy of test and training datasets are 81.8% and 81.9%.

Experiment 7/ SVM :An SVM training algorithm creates a model that assigns new examples to one of two categories, making it a non-probabilistic binary linear classifier, given a series of training examples that are individually designated as belonging to one of two categories. To generate model various parameters are set and the model is fitted in the tree. The samples are divided into X and Y train, X and Y test to check the accuracy of the model. Where we are predicting the accuracy of the model on the samples collected on both trained and test samples. On this we found accuracy of test and training datasets are 81.8% and 79.8%.

Performance Evaluation:

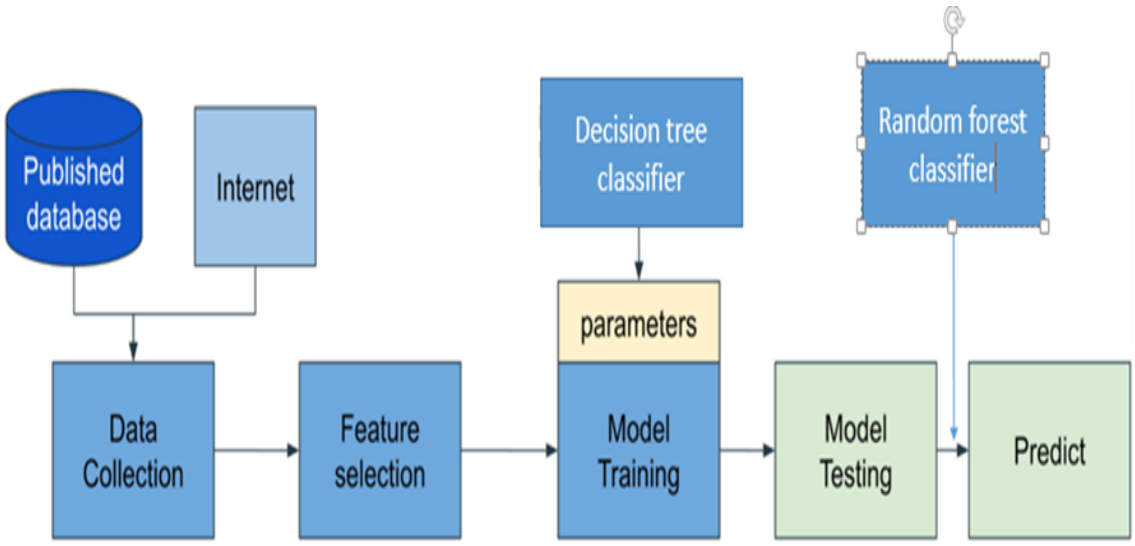
The evaluation of performance was carried out during the testing process. The original dataset would be divided into training data and test data, usually 80% and 20%, respectively. When evaluating the classifier's behavior on the testing dataset, there were four statistical numbers: the number of correctly identified positive data points (TP), the number of correctly identified negative data points (TN), the number of negative data points labeled by the classifier as positive (FP), and the number of positive data points labeled by the model as negative (FN).

There are several broadly used metrics to evaluate performance. The classification accuracy is the ratio of correct predictions to total predictions: accuracy = TP + TN /TP + TN + FN + FP.In binary classification cases, it is known that random selection has 50% accuracy. In unbalanced datasets, sometimes high accuracy does not mean that the model is excellent. For instance, among the 10,000 data, 9000 were legitimate websites, and 1000 were phishing websites, so when the prediction model did nothing, it could reach 90%. Accuracy is misleading when the class sizes are substantially different. Precision is the percentage of correctly identified positive data points among those predicted as positive by the model. The number of false-positive cases (FP) reflects the false warning rate. In real-time phishing detection systems, this directly affects the user experience and trustworthiness: Precision = TP/TP + FP.The recall is the portion of positive data points labeled as such by the model among all truly positive data points. The number of false-negative cases (FN) represents the number of phishing URLs that has not been detected. Leak alarms mean that users are likely to receive an attack that could result in the theft of sensitive information. Misleading users can do more harm to users than not detecting them:

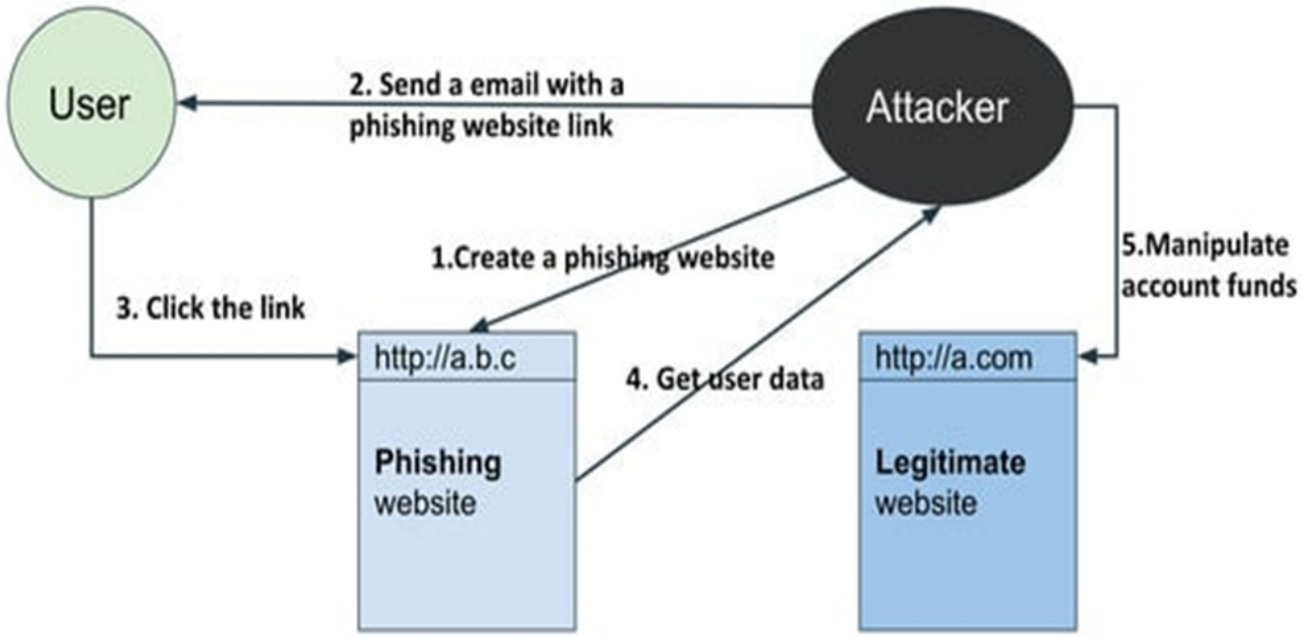
Recall = TP /TP + FN.The F-measure or F-score is the combination of precision and recall. Generally, it is formulated as shown below: Fβ = (β2 + 1)× Precision × Recall/ β2 × Precision + Recall β ∈ (0, ∞).Here, β quantifies the relative importance of the precision and recall such that β = 1 stands for the precision and recall being equally important, which is also called F1. The F-score does the best job of any single statistic, but all four work together to describe the performance of a classifier: F1 = 2 × Precision × recall /Precision + recall = TP /TP + 1/2 (FP + FN).In addition, many researchers use the N-fold cross-validation technique to measure performance for phishing detection [4,30,31]. The N-fold cross-validation

technique is widely used on small datasets for evaluating machine learning models' performance. It is a resampling procedure that divides the original data samples into N pieces after shuffling the dataset randomly. One of the pieces is used in the testing process, and others are applied to the training process. Commonly, N is set as 10 or 5.

3.9 ARCHITECTURE DIAGRAM



3.10 FLOW DIAGRAM



3.11 DECISION TREE ALGORITHM

There's not much mathematics involved here. Since it is very easy to use and interpret it is one of the most widely used and practical methods used in Machine Learning.It is a tool that has applications spanning several different areas. Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.Root Nodes – It is the node present at the beginning of a decision tree from this node the population starts dividing according to various features.Decision Nodes – the nodes we get after splitting the root nodes are called Decision Node.Leaf Nodes – the nodes where further splitting is not possible are called leaf nodes or terminal nodes. Sub-tree – just like a small portion of a graph is called sub-graph similarly a subsection of this decision tree is called sub-tree.Pruning – is nothing but cutting down some nodes to stop overfitting.

Entropy:Entropy is nothing but the uncertainty in our dataset or measure of disorder. Let me try to explain this with the help of an example.Suppose you have a group of friends who decides which movie they can watch together on Sunday. There are 2 choices for movies, one is “Lucy” and the second is “Titanic” and now everyone has to tell their choice. After everyone gives their answer we see that “Lucy” gets 4 votes and “Titanic” gets 5 votes. Which movie do we watch now? Isn’t it hard to choose 1 movie now because the votes for both the movies are somewhat equal.This is exactly what we call disorderness, there is an equal number of votes for both the movies, and we can’t really decide which movie we should watch. It would have been much easier if the votes for “Lucy” were 8 and for “Titanic” it was 2. Here we could easily say that the majority of votes are for “Lucy” hence everyone will be watching this movie.

Information Gain:Information gain measures the reduction of uncertainty given some feature and it is also a deciding factor for which attribute should be selected as a decision node or root node.It is just entropy of the full dataset – entropy of the dataset given some feature. To understand this better let’s consider an example:Suppose our entire population has a total of 30 instances. The dataset is to predict whether the person will go to the gym or not. Let’s say 16 people go to the gym and 14 people don’tNow we have two features to predict whether he/she will go to the gym or not. Feature 1 is “Energy” which takes two values “high” and “low”Feature 2 is “Motivation” which takes 3 values “No motivation”, “Neutral” and “Highly motivated”.Let’s see how our decision tree will be made using these 2 features. We’ll use information gain to decide which feature should be the root node and which feature should be placed after the split.Let’s calculate the entropy:To see the weighted average of entropy of each node we will do as follows:Now we have the value of E(Parent) and E(Parent|Energy), information gain will be:Our parent entropy was near 0.99 and after looking at this value of information gain, we can say that the entropy of the dataset will decrease by 0.37 if we make “Energy” as our root node.Similarly, we will do this with

the other feature “Motivation” and calculate its information gain.In this example “Energy” will be our root node and we’ll do the same for sub-nodes. Here we can see that when the energy is “high” the entropy is low and hence we can say a person will definitely go to the gym if he has high energy, but what if the energy is low? We will again split the node based on the new feature which is “Motivation”.

You must be asking this question to yourself that when do we stop growing our tree? Usually, real-world datasets have a large number of features, which will result in a large number of splits, which in turn gives a huge tree. Such trees take time to build and can lead to overfitting. That means the tree will give very good accuracy on the training dataset but will give bad accuracy in test data.There are many ways to tackle this problem through hyperparameter tuning. We can set the maximum depth of our decision tree using the max_depth parameter. The more the value of max_depth, the more complex your tree will be. The training error will off-course decrease if we increase the max_depth value but when our test data comes into the picture, we will get a very bad accuracy. Hence you need a value that will not overfit as well as underfit our data and for this, you can use GridSearchCV.

Another way is to set the minimum number of samples for each spilt. It is denoted by min_samples_split. Here we specify the minimum number of samples required to do a split. For example, we can use a minimum of 10 samples to reach a decision. That means if a node has less than 10 samples then using this parameter, we can stop the further splitting of this node and make it a leaf node.

Pruning:It is another method that can help us avoid overfitting. It helps in improving the performance of the tree by cutting the nodes or sub-nodes which are not significant. It removes the branches which have very low importance.

There are mainly 2 ways for pruning:Pre-pruning – we can stop growing the tree earlier, which means we can prune/remove/cut a node if it has low importance while growing the tree.Post-pruning – once our tree is built to its depth, we can start pruning the nodes based on their significance.

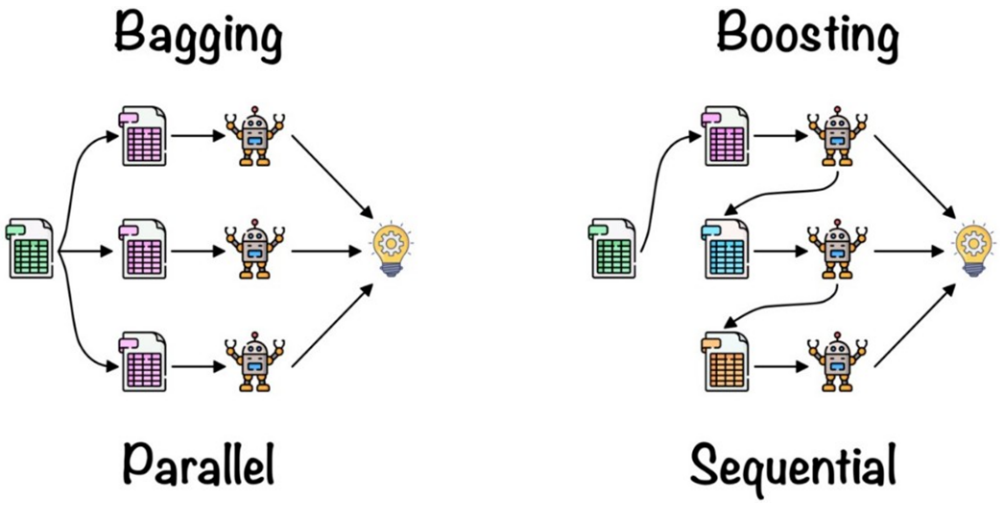
3.12 RANDOM FOREST

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.Let's dive into a real-life analogy to understand this concept further. A student named X wants to choose a course after his 10+2, and he is confused about the choice of course based on his skill set. So he decides to consult various people like his cousins, teachers, parents, degree students, and working people. He asks them varied questions like why he should choose, job opportunities with that course, course fee, etc. Finally, after consulting various people about the course he decides to take the course suggested by most of the people.

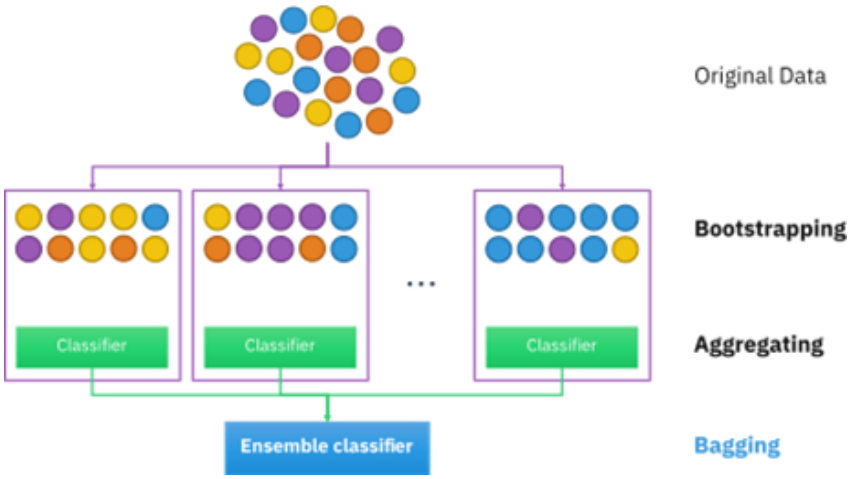
Ensemble uses two types of methods:

1.Bagging– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest.

2.Boosting– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.



Bagging:Bagging, also known as Bootstrap Aggregation is the ensemble technique used by random forest. Bagging chooses a random sample from the data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as row sampling. This step of row sampling with replacement is called bootstrap. Now each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation.



Steps involved in random forest algorithm:

Step 1: In Random forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

For example: consider the fruit basket as the data as shown in the figure below. Now a number of samples are taken from the fruit basket and an individual decision tree is constructed for each sample. Each decision tree will generate an output as shown in the figure. The final output is considered based on majority voting. In the below figure you can see that the majority decision tree gives output as an apple when compared to a banana, so the final output is taken as an apple.

Important Features of Random Forest

Diversity- Not all attributes/variables/features are considered while making an individual tree, each tree is different..Immune to the curse of dimensionality- Since each tree does not consider all the features, the feature space is reduced.Parallelization-Each tree is created independently out of different data and attributes.This means that we can make full use of the CPU to build random forests.Train-Test split- In a random forest we don't have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.Stability- Stability arises because the result is based on majority voting/ averaging.

4. CONCLUSION

This survey presented various algorithms and approaches to detect phishing websites by several researchers in Machine Learning.On reviewing the papers, we came to a conclusion that most of the work done by using familiar machine learning algorithms like Naïve Bayesian, SVM, Decision Tree and Random Forest.Some authors proposed a new system like Phish Score and Phish Checker for detection. The combinations of features with regards to accuracy, precision, recall etc. were used. Experimentally successful techniques in detecting phishing website URLs were summarized As phishing websites increases day by day, some features may be included or replaced with new ones to detect them.

5. REFERENCES

[1] 'APWG | Unifying The Global Response To Cybercrime' (n.d.) available: https://apwg.org/

[2] 14 Types of Phishing Attacks That IT Administrators Should Watch For [online] (2021)https://www.blog.syscloud.com,available:https://www.blog.syscloud.comtypes-of-phishing/

[3] Lakshmanarao, A., Rao, P.S.P., Krishna, M.M.B. (2021) 'Phishing website detection using novel machine learning fusion approach', in 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), Presented at the 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), 1164–1169

[4] H. Chapla, R. Kotak and M. Joiser, "A Machine Learning Approach for URL Based Web Phishing Using Fuzzy Logic as Classifier", 2019 International Conference on Communication and Electronics Systems (ICCES), pp. 383-388, 2019, July

[5] Vaishnavi, D., Suwetha, S., Jinila, Y.B., Subhashini, R., Shyry, S.P. (2021) ‘A Comparative Analysis of Machine Learning Algorithms on Malicious URL Prediction’, in 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Presented at the 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), 1398–1402

[6] Microsoft, Microsoft Consumer safety report. https://news.microsoft.com/en-sg/2014/02/11/microsoft-consumersafety-index-reveals-impact-of-poor-online-safety-behaviours-in-singapore/sm.001xdu50tlxsej410r11kqvksu4nz.

[7] Internal Revenue Service, IRS E-mail Schemes. Available at https://www.irs.gov/uac/newsroom/consumers-warnedof-new-surge-in-irs-email-schemes-during-2016-tax-season-tax-industry-also-targeted.

[8] Abu-Nimeh, S., Nappa, D., Wang, X., Nair, S. (2007), A comparison of machine learning techniques for phishing detection.

Proceedings of the Anti-phishing Working Groups 2nd Annual ECrime Researchers Summit on - ECrime ’07.

doi:10.1145/1299015.1299021.

[9] E., B., K., T. (2015)., Phishing URL Detection: A Machine Learning and Web Mining-based Approach. International Journal of

Computer Applications,123(13), 46-50. doi:10.5120/ijca2015905665.

[10] Wang Wei-Hong, L V Yin-Jun, CHEN Hui-Bing, FANG Zhao-Lin., A Static Malicious Javascript Detection Using SVM, In

Proceedings of the 2nd International Conference on Computer Science and Electrical Engineering (ICCSEE 2013).

[11] Ningxia Zhang, Yongqing Yuan, Phishing Detection Using Neural Net-work, In Proceedings of International Conference on Neural Information Processing, pp. 714–719. Springer, Heidelberg (2004).

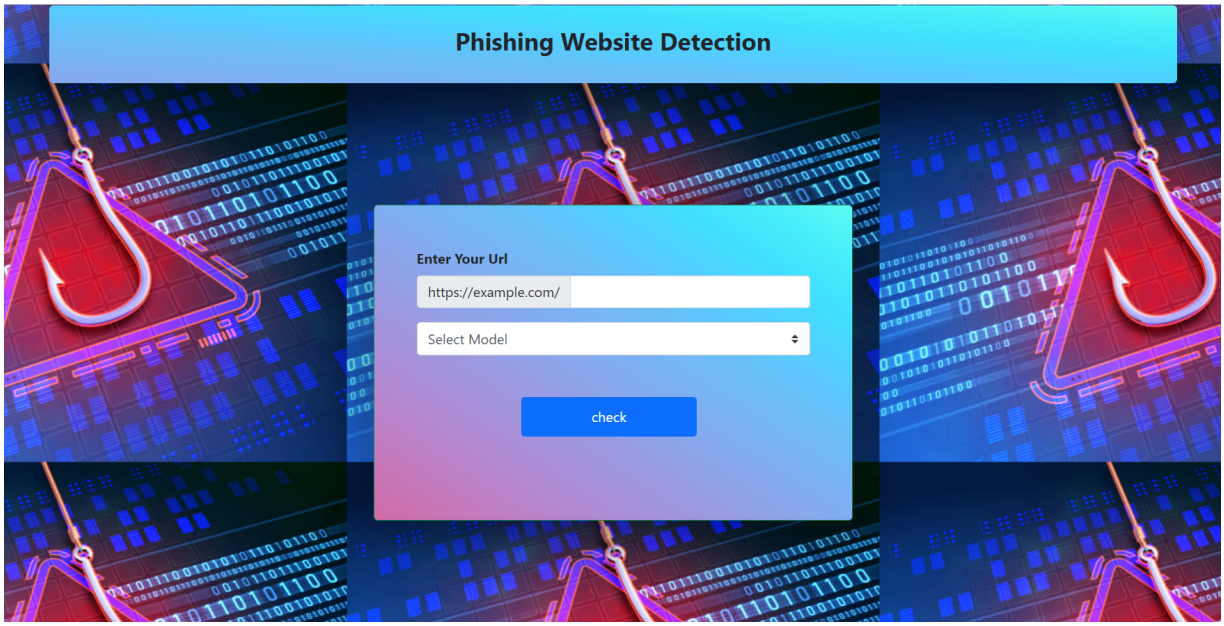
[12] Ram Basnet, Srinivas Mukkamala et al, Detection of Phishing Attacks: A Machine Learning Approach, In Proceedings of the International World Wide Web Conference (WWW), 2003.

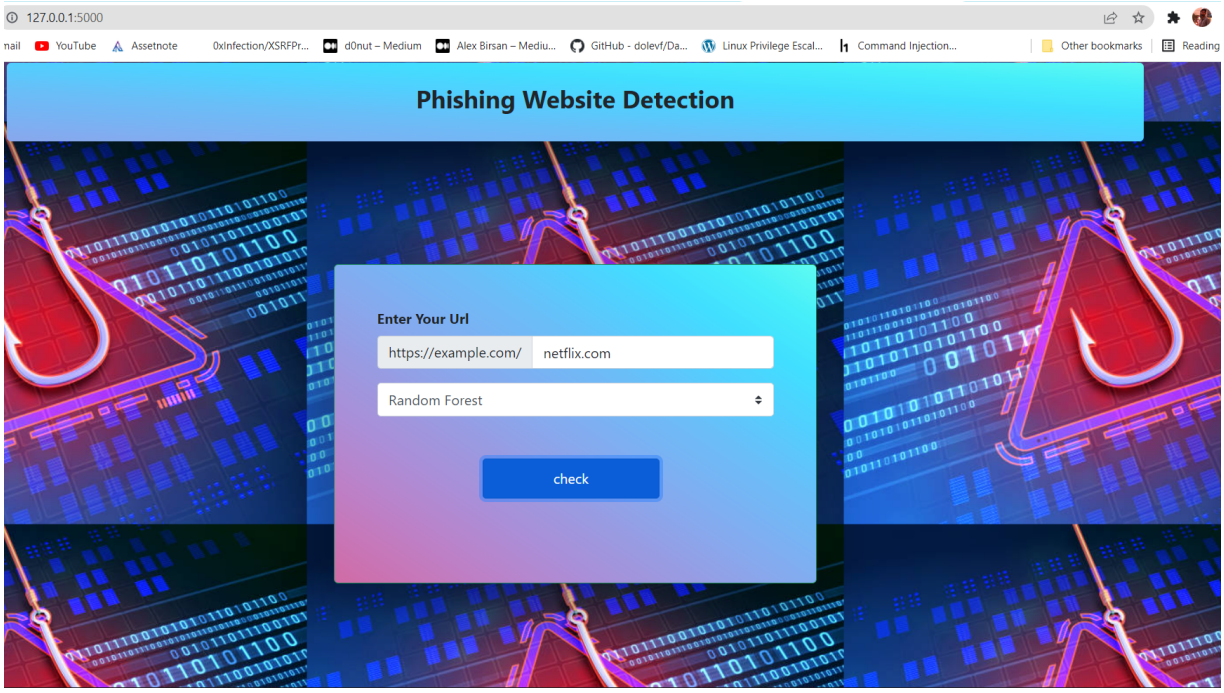
[13] Sci-kit learn, SVM library. http://scikit-learn.org/stable/modules/svm.html.

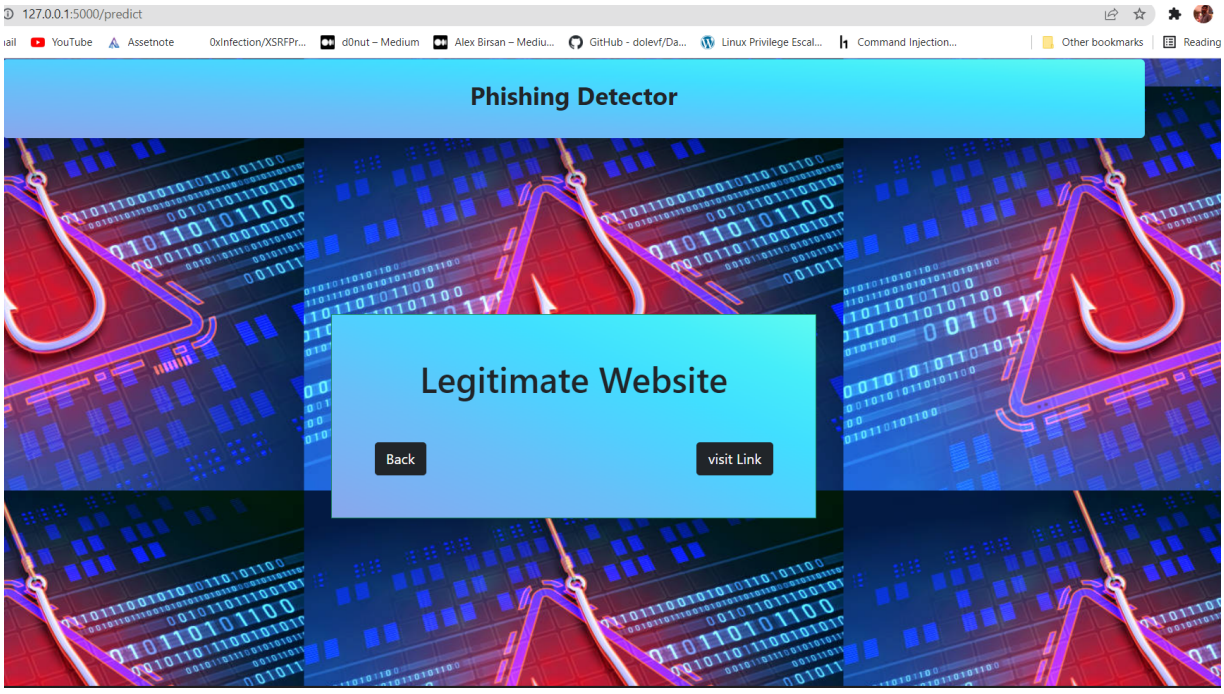
[14] Sklearn, ANN library. http://scikit-learn.org/stable/modules/ann.html.

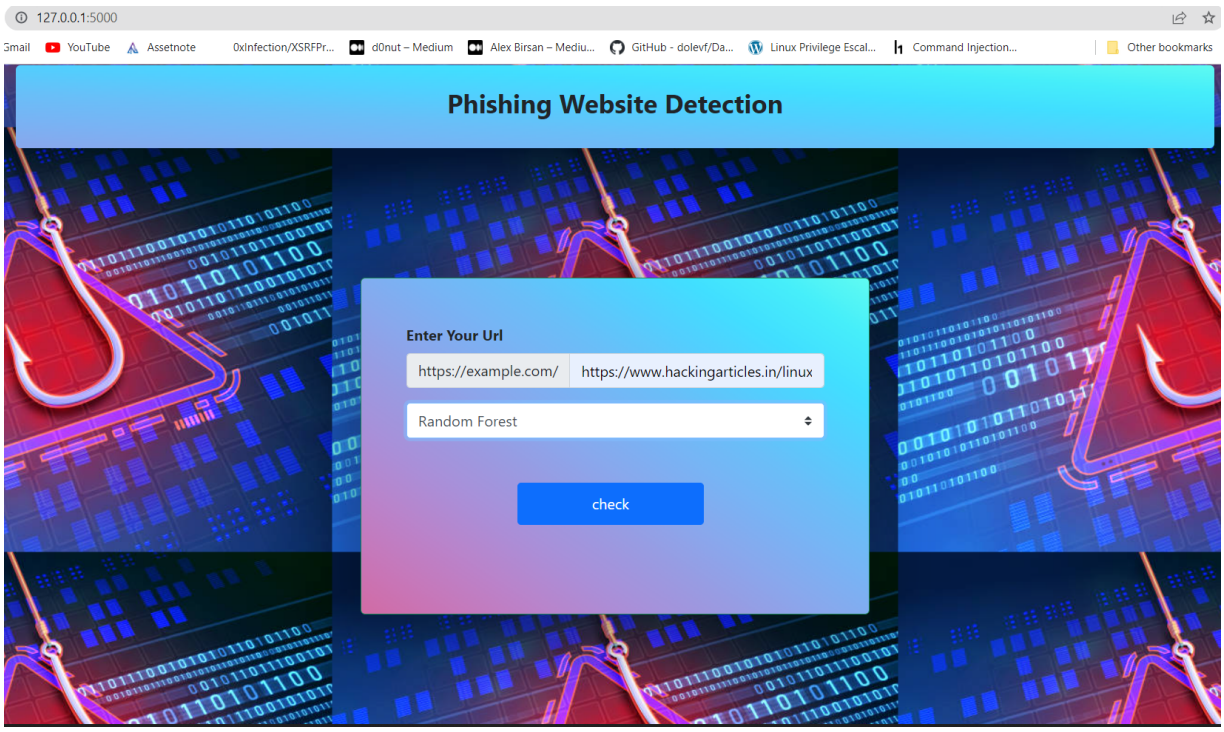
[15]Sclera,Randomforestlibrary.http://scikitlearn.org/stable/modules/Randomforesets.html.

6. OUTPUT SCREENSHOTS




127.0.0.1:5000
YouTube
Assetnote
0xInfection/XSRFPr...
d0nut – Medium
Alex Birsan – Mediu...
GitHub - dolevf/Da...
Linux Privilege Escal...
Command Injection...
Other bookmarks
Reading
Phishing Website Detection
Enter Your Url
https://example.com/
netflix.com
Random Forest
check


127.0.0.1:5000/predict
YouTube
Assetnote
0xInfection/XSRFPr...
d0nut – Medium
Alex Birsan – Mediu...
GitHub - dolevf/Da...
Linux Privilege Escal...
Command Injection...
Other bookmarks
Reading
Phishing Detector
Legitimate Website
Back
visit Link



7. APPENDIX

main.py

```
from flask_cors import CORS,cross_origin

import pickle

from features import *

app = Flask(__name__) # initializing a flask app

@app.route('/',methods=['GET'])  # route to display the home page

@cross_origin()

def homePage():
```

```
    return render_template("index.html")

@app.route('/predict',methods=['POST','GET']) # route to show the predictions in a web UI

@cross_origin()

def index():

    if request.method == 'POST':

        try:

            #  reading the inputs given by the user

            web_link=str(request.form['link'])

            print(web_link)

            model = int(request.form['model'])

            print(model)

            data=featureExtraction(web_link)

            print(data)

            if model==1:

                filename = 'DecisionTree.pickle'

                    loaded_model = pickle.load(open(filename, 'rb')) # loading the model file from the storage

                # predictions using the loaded model file
```

```
            prediction=loaded_model.predict([data])
            print('prediction is', prediction[0])
        else:
            filename = 'RandomForest.pickle'
                loaded_model = pickle.load(open(filename, 'rb')) # loading the model file from the storage
            # predictions using the loaded model file
            prediction=loaded_model.predict([data])
            print('prediction is', prediction[0])
        if prediction[0]==1:
            return render_template('result.html', pred =True  ,link=web_link )
        else:
             return render_template('result.html', pred =False  ,link=web_link )
    except Exception as e:
        print('The Exception message is: ',e)
        return 'something is wrong'
# return render_template('results.html')
else:
```

```
        return render_template('index.html')

if __name__ == "__main__":

    #app.run(host='127.0.0.1', port=8001, debug=True)

        app.run(debug=False) # running the app
```

features.py

```
from datetime import datetime

# 2.Checks for IP address in URL (Have_IP)

def havingIP(url):

  try:

    ipaddress.ip_address(url)

    ip = 1

  except:

    ip = 0

  return ip



"""#### **3.1.3. "@" Symbol in URL**
```

Checks for the presence of '@' symbol in the URL. Using “@” symbol in the URL leads the browser to ignore everything preceding the “@” symbol and the real address often follows the “@” symbol.

If the URL has '@' symbol, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

"""

3.Checks the presence of @ in URL (Have_At)

def haveAtSign(url):

  if "@" in url:

    at = 1

  else:

    at = 0

  return at

"""#### **3.1.4. Length of URL**

Computes the length of the URL. Phishers can use long URL to hide the doubtful part in the address bar. In this project, if the length of the URL is greater than or equal 54 characters then the URL classified as phishing otherwise legitimate.If the length of URL >= 54 , the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

"""

4.Finding the length of URL and categorizing (URL_Length)

def getLength(url):

  if len(url) < 54:

```
    length = 0
  else:
    length = 1
  return length


"""#### **3.1.5. Depth of URL**

Computes the depth of the URL. This feature calculates the number of sub pages in the given url based on the '/'.

The value of feature is a numerical based on the URL.

"""

# 5.Gives number of '/' in URL (URL_Depth)
def getDepth(url):
  s = urlparse(url).path.split('/')
  depth = 0
  for j in range(len(s)):
    if len(s[j]) != 0:
```

```
    depth = depth+1
  return depth


"""#### **3.1.6. Redirection "//" in URL**


Checks the presence of "//" in the URL. The existence of “//” within the URL path means that the user will be redirected to another website. The location of the “//” in URL is computed. 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.


If the "//" is anywhere in the URL apart from after the protocal, thee value assigned to this feature is 1 (phishing) or else 0 (legitimate).

"""


# 6.Checking for redirection '//' in the url (Redirection)

def redirection(url):

  pos = url.rfind('//')

  if pos > 6:

    if pos > 7:
```

```
    return 1
  else:
    return 0
else:
  return 0


"""#### **3.1.7. "http/https" in Domain name**

Checks for the presence of "http/https" in the domain part of the URL. The phishers may add the “HTTPS” token to the domain part of a URL in order to trick users.

If the URL has "http/https" in the domain part, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).
"""

# 7.Existence of “HTTPS” Token in the Domain Part of the URL (https_Domain)
def httpDomain(url):
  domain = urlparse(url).netloc
  if 'https' in domain:
```

```
    return 1
  else:
    return 0


"""#### **3.1.8. Using 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.


If the URL is using Shortening Services, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).
"""


#listing shortening services
shortening_services                                                    =
r"bit\.ly|goo\.gl|shorte\.st|go2l\.ink|x\.co|ow\.ly|t\.co|tinyurl|tr\.im|is\.gd|cli\.gs|" \


r"yfrog\.com|migre\.me|ff\.im|tiny\.cc|url4\.eu|twit\.ac|su\.pr|twurl\.nl|snipurl\.com|" \
```

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

r"doiop\.com|short\.ie|kl\.am|wp\.me|rubyurl\.com|om\.ly|to\.ly|bit\.do|t\.co|lnkd\.in|db\.tt|" \

r"qr\.ae|adf\.ly|goo\.gl|bitly\.com|cur\.lv|tinyurl\.com|ow\.ly|bit\.ly|ity\.im|q\.gs|is\.gd|" \

r"po\.st|bc\.vc|twitthis\.com|u\.to|j\.mp|buzurl\.com|cutt\.us|u\.bb|yourls\.org|x\.co|" \

r"prettylinkpro\.com|scrnch\.me|filoops\.info|vzturl\.com|qr\.net|1url\.com|tweez\.me|v\.gd|" \
                    r"tr\.im|link\.zip\.net"


# 8. Checking for Shortening Services in URL (Tiny_URL)
def tinyURL(url):
    match=re.search(shortening_services,url)
    if match:
        return 1
    else:
```

```
    return 0
```

"""#### **3.1.9. Prefix or Suffix "-" in Domain**

Checking the presence of '-' in the domain part of URL. 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.

If the URL has '-' symbol in the domain part of the URL, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

"""

```
# 9.Checking for Prefix or Suffix Separated by (-) in the Domain (Prefix/Suffix)
def prefixSuffix(url):
    if '-' in urlparse(url).netloc:
        return 1            # phishing
    else:
        return 0            # legitimate
```

"""### **3.2. Domain Based Features:**

Many features can be extracted that come under this category. Out of them, below mentioned were considered for this project.

* DNS Record
* Website Traffic
* Age of Domain
* End Period of Domain

Each of these features are explained and the coded below:

"""

"""#### **3.2.1. DNS Record**

For phishing websites, either the claimed identity is not recognized by the WHOIS database or no records founded for the hostname.

If the DNS record is empty or not found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

"""

11.DNS Record availability (DNS_Record)

obtained in the featureExtraction function itself

"""#### **3.2.2. Web 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). By reviewing our dataset, we find that in worst scenarios, legitimate websites ranked among the top 100,000. Furthermore, if the domain has no traffic or is not recognized by the Alexa database, it is classified as “Phishing”.

If the rank of the domain < 100000, the vlaue of this feature is 1 (phishing) else 0 (legitimate).

"""

```
# 12.Web traffic (Web_Traffic)

def web_traffic(url):
  try:
    #Filling the whitespaces in the URL if any
    url = urllib.parse.quote(url)
                                                                    rank                    =
BeautifulSoup(urllib.request.urlopen("http://data.alexa.com/data?cli=10&dat=s&url="
+ url).read(), "xml").find(
        "REACH")['RANK']
    rank = int(rank)
  except TypeError:
        return 1
  if rank <100000:
    return 1
  else:
    return 0


"""#### **3.2.3. Age of Domain**
```

This feature can be extracted from WHOIS database. Most phishing websites live for a short period of time. The minimum age of the legitimate domain is considered to be 12 months for this project. Age here is nothing but different between creation and expiration time.

If age of domain > 12 months, the vlaue of this feature is 1 (phishing) else 0 (legitimate).

"""# 13.Survival time of domain: The difference between termination time and creation time (Domain_Age)

```
def domainAge(domain_name):
  creation_date = domain_name.creation_date
  expiration_date = domain_name.expiration_date
  if (isinstance(creation_date,str) or isinstance(expiration_date,str)):
    try:
      creation_date = datetime.strptime(creation_date,'%Y-%m-%d')
      expiration_date = datetime.strptime(expiration_date,"%Y-%m-%d")
    except:
      return 1
  if ((expiration_date is None) or (creation_date is None)):
      return 1
  elif ((type(expiration_date) is list) or (type(creation_date) is list)):
```

```
    return 1
  else:
    ageofdomain = abs((expiration_date - creation_date).days)
    if ((ageofdomain/30) < 6):
      age = 1
    else:
      age = 0
  return age


"""#### **3.2.4. End Period of Domain**

This feature can be extracted from WHOIS database. For this feature, the remaining domain time is calculated by finding the different between expiration time & current time. The end period considered for the legitimate domain is 6 months or less  for this project.

If end period of domain > 6 months, the vlaue of this feature is 1 (phishing) else 0 (legitimate).

"""


# 14.End time of domain: The difference between termination time and current time (Domain_End)
```

```
def domainEnd(domain_name):
  expiration_date = domain_name.expiration_date
  if isinstance(expiration_date,str):
    try:
      expiration_date = datetime.strptime(expiration_date,"%Y-%m-%d")
    except:
      return 1
  if (expiration_date is None):
      return 1
  elif (type(expiration_date) is list):
      return 1
  else:
    today = datetime.now()
    end = abs((expiration_date - today).days)
    if ((end/30) < 6):
      end = 0
    else:
      end = 1
```

return end

"""## **3.3. HTML and JavaScript based Features**Many features can be extracted that come under this category. Out of them, below mentioned were considered for this project.

* IFrame Redirection

* Status Bar Customization

* Disabling Right Click

* Website Forwarding

Each of these features are explained and the coded below:

"""

importing required packages for this section

import requests

"""### **3.3.1. 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.

If the iframe is empty or repsonse is not found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

"""

15. IFrame Redirection (iFrame)

```
def iframe(response):
  if response == "":
      return 1
  else:
      if re.findall(r"[<iframe>|<frameBorder>]", response.text):
          return 0
      else:
          return 1
```

"""### **3.3.2. 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

If the response is empty or onmouseover is found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

"""

16.Checks the effect of mouse over on status bar (Mouse_Over)

```
def mouseOver(response):
  if response == "" :
    return 1
  else:
    if re.findall("<script>.+onmouseover.+</script>", response.text):
      return 1
    else:
      return 0
```

"""### **3.3.3. 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.

If the response is empty or onmouseover is not found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

"""

```
# 17.Checks the status of the right click attribute (Right_Click)
def rightClick(response):
```

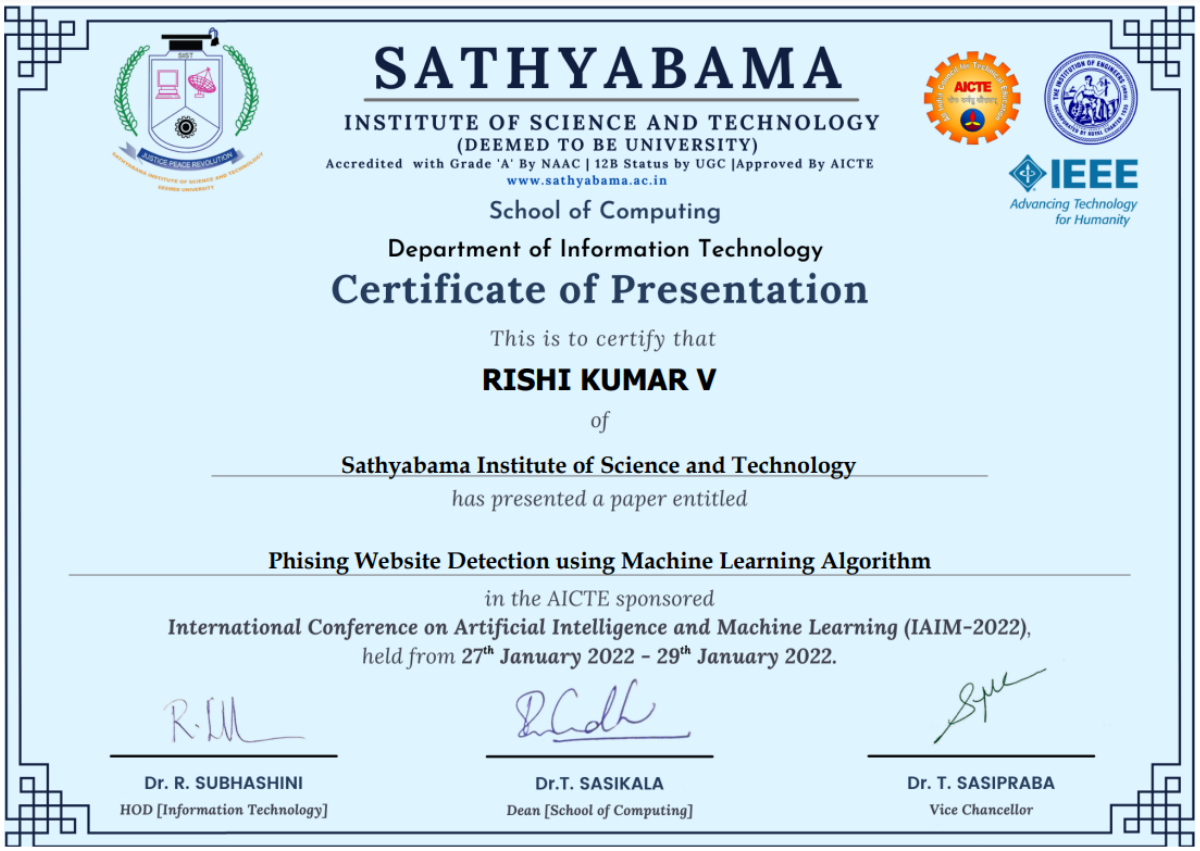
```
  if response == "":
    return 1
  else:
    if re.findall(r"event.button ?== ?2", response.text):
      return 0
    else:
      return 1


"""### **3.3.4. 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.

"""


# 18.Checks the number of forwardings (Web_Forwards)

def forwarding(response):
  if response == "":
    return 1
```

```
  else:
    if len(response.history) <= 2:
      return 0
    else:
      return 1


"""## **4. Computing URL Features**


Create a list and a function that calls the other functions and stores all the features of the URL in the list. We will extract the features of each URL and append to this list.
"""


#Function to extract features
def featureExtraction(url, whois=None):


  features = []
  #Address bar based features (10)
  #features.append(getDomain(url))
```

```
features.append(havingIP(url))

features.append(haveAtSign(url))

features.append(getLength(url))

features.append(getDepth(url))

features.append(redirection(url))

features.append(httpDomain(url))

features.append(tinyURL(url))

features.append(prefixSuffix(url))


#Domain based features (4)

dns = 0

try:

  domain_name = whois.whois(urlparse(url).netloc)

except:

  dns = 1


features.append(dns)

features.append(web_traffic(url))
```

```
  features.append(1 if dns == 1 else domainAge(domain_name))
  features.append(1 if dns == 1 else domainEnd(domain_name))


  # HTML & Javascript based features
  try:
    response = requests.get(url)
  except:
    response = ""features.append(iframe(response))
  features.append(mouseOver(response))
  features.append(rightClick(response))
  features.append(forwarding(response))
  return features
#""" converting the list to dataframe
""" feature_names = ['Domain', 'Have_IP', 'Have_At', 'URL_Length', 'URL_Depth','Redirection',
                    'https_Domain', 'TinyURL', 'Prefix/Suffix', 'DNS_Record', 'Web_Traffic',
                        'Domain_Age', 'Domain_End', 'iFrame', 'Mouse_Over','Right_Click', 'Web_Forwards', 'Label'] """
```

8. PUBLICATION AND PLAGIARISM REPORT



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade 'A' By NAAC | 12B Status by UGC |Approved By AICTE
www.sathyabama.ac.in

School of Computing

Department of Information Technology

Certificate of Presentation

This is to certify that

RISHI KUMAR V

of

Sathyabama Institute of Science and Technology

has presented a paper entitled

Phising Website Detection using Machine Learning Algorithm

in the AICTE sponsored
International Conference on Artificial Intelligence and Machine Learning (IAIM-2022),
held from **27th January 2022 - 29th January 2022.**

Dr. R. SUBHASHINI
HOD [Information Technology]

Dr.T. SASIKALA
Dean [School of Computing]

Dr. T. SASIPRABA
Vice Chancellor

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade 'A' By NAAC | 12B Status by UGC |Approved By AICTE
www.sathyabama.ac.in

School of Computing

Department of Information Technology

Certificate of Presentation

This is to certify that

MANOJ SV

of

Sathyabama Institute of Science and Technology

has presented a paper entitled

Phising Website Detection using Machine Learning Algorithm

in the AICTE sponsored
International Conference on Artificial Intelligence and Machine Learning (IAIM-2022),
held from **27th January 2022 - 29th January 2022.**

Dr. R. SUBHASHINI	Dr.T. SASIKALA	Dr. T. SASIPRABA
HOD [Information Technology]	Dean [School of Computing]	Vice Chancellor

ORIGINALITY REPORT

13%	11%	2%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	norma.ncirl.ie Internet Source	10%
2	Submitted to Gitam University Student Paper	1%
3	A. Lakshmanarao, P.Surya Prabhakara Rao, M M Bala Krishna. "Phishing website detection using novel machine learning fusion approach", 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), 2021 Publication	1%
4	www.jetir.org Internet Source	1%
5	"Phishing Websites Detection using Machine Learning", International Journal of Recent Technology and Engineering, 2019 Publication	1%