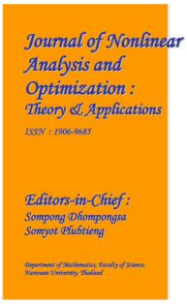
Journal of **N**onlinear **A**nalysis and **O**ptimization

Vol. 14, Issue. 2 : 2023

ISSN : **1906-9685**



DETECTION OF FRAUDULENT WEBSITES USING MACHINE LEARNING

[1]P. Gayatri, [2]A.Vaishnavi, [3]G.Sahruthi, [4]P.Varshini, [5]A.Nikshitha [1]Assistant Professor, [2,3,4,5]UG Students, Dept. of CSE (AI & ML), Malla Reddy Engineering College for Women (Autonomous), Hyderabad, India. E-mail: gaya3pillutla@gmail.com

ABSTRACT

Phishing is an online scam when a perpetrator sends out phoney messages that appear to be from a reliable source. The email will contain a link or file that, when clicked, can steal personal data or infect a machine with malware. In the past, phishing attacks were carried out through massive spam operations that indiscriminately targeted large populations of people. To get as many individuals to open a malicious file or click on a malicious link was the objective. There are numerous methods to find this category of attack. The utilization of machine learning is one strategy. The URLs that the user receives will be entered to the machine learning model, which will then process the input and present the results, indicating whether they are phishing or not. These URLs can be categorized by means of a collection of machine learning (ML) approaches, together with SVM, Neural Networks, Random Forest, Decision Tree, XG boost, etc. The Gradient Boost classifier, Random Forest, Decision Tree classifier, and seven more classifiers are all covered by the suggested method. The proposed method successfully distinguished between legitimate and fraudulent URLs with a Gradient Boost Classifier accuracy of 97.4%.

Keywords: ML, Website, SVM, Neural network, Random forest, Decision tree, XG boost

INTRODUCTION

Due of how simple it is to expand a phoney website that strongly look like a legitimate website, phishing is now a top worry for defense researchers. Even though specialist know how to spot fraudulent websites, not all users can, and as a result, some users fall prey to phishing scams. The attacker's primary goal is to obtain login information for bank accounts. Every year, phishing scams cost American businesses $2 billion in lost revenue from clients who fall for them. The 3rd Microsoft Computing Safer Index Report, which was released in February 2014, estimates that phishing may have an annual global economic impact of $5 billion. Users are falling victim to phishing attacks more frequently since they have no idea of them. Since phishing attacks take advantage of customer vulnerabilities, it is extremely difficult to stop them; currently it is important to advance phishing detection techniques. The "blacklist" method, which is the standard technique for detecting phishing websites, involves adding blacklisted URLs and Internet Protocol (IP) addresses to the antivirus database. To avoid blacklists, offenders alter the URL so it looks genuine using obfuscation and a variety of additional simple techniques, including fast-flux, whereby proxies are instantly built to host the website.

This method's primary flaw is its inability to identify zero-hour phishing attacks. Heuristic-based detection, which takes into account traits that have been shown to exist in actual phishing assaults, is capable of spotting Phishing attacks at zero hour. The false positive rate for identification is relatively

http://doi.org/10.36893/JNAO.2023.V14I2.0161-0170

high, and the features can sometimes fail to be included during such attacks. Many security experts are now focusing on machine learning techniques to conquer the limitations of blacklist and heuristics depending methods.

Numerous algorithms used in machine learning technologies depend on historical data in order to anticipate or make decisions about future data. In order to accurately identify phishing websites, including zero-hour phishing websites, a technique might scrutinize numerous blacklisted and valid URLs and their attributes.

LITERATURE SURVEY

S.No	Authors	Contribution	confines
1.	Jain A.K., and Gupta B.B (2017)	A combination of the NB and SVM techniques were used to find the dangerous websites.	SVM and NB are both slow learners and do not keep track of earlier outcomes in memory. As a result, the URLdetector's effectiveness can be diminished.
2.	Purbay M., and Kumar D (2022)	URL classification was done using a variety of ML techniques.	The effectiveness of several ML method types was compared. The algorithms' retrieval capabilities, however, were not discussed.
3.	Gandotra E., and Gupta D. (2020)	a variety of categorization algorithms were used to find hazardous URLs.	The consequences of the studies demonstrate that the system's performance was superior to that of other ML techniques. It cannot handle bigger amounts of data, though.
4.	Hung Le et al. (2018)	a deep learning-based URL detector was proposed. According to authors, the approach can generate approaching as of URL.	Deep learning techniques require added time to complete a task. Additionally, it parses the URL and compares it to the library to produce a result.
5.	Hong J. et al., (2021)	created a crawler to harvest URLs as of data storage systems. Lexical features were used to locate the phishing websites.	Presentation was assessed using a crawler-collected dataset. Therefore, here is no guarantee as to how well the URL detector will work by real-time URLs.
6.	Kumar J. et al. (2020)	created a URL detector using a collection of blacklisted websites. In order to distinguish between dangerous and trustworthy websites, a lexical feature method is moreover utilized.	The presentation of the detector using real-time URLs can be affected by the authors' use of an older dataset.
7.	Hassan Y.A. and Abdelfettah B.	For classifying webpages and identifying phishing websites, a URL detector has been proposed. The	Although the GA dependent URL detector performed better, forecast time is quite long for complex sets of URLs.

	(2021)	performance was enhanced using GA method.	
8.	Rao RS and Pais AR. (2021)	The page characteristics used by authors include the logo, favicon, scripts, and styles.	Because a server was used in the approach to update the page properties, the detecting system performed less well.
9.	Aljofey A et al. (2022)	a phishing page detection system based on CNN. To discover URLs, a sequential prototype is engaged.	As on the available studies, CNN works enhanced at fetching images than words.
10.	AlEroud A and Karabatis G (2020)	The research uses a generative adversarial network to get around a detection mechanism.	By learning about the environment around it, a neural network-based recognition system be able to recognize the thought of an unfavorable network

Problem Definition

The goal of phishing websites is to trick users into divulging critical information, such as login credentials, money, or personal information. These websites are intended to seem like official websites, frequently copying well-known businesses, organizations, or financial institutions. Phishing websites often utilise a variety of false approaches to persuade visitors to disclose sensitive information.

We propose a work called Phishing website detection to detect these websites, which entails the process of identifying and stopping people from accessing phoney websites meant to fool and steal critical information. The goal is to identify and save people from phishing websites, hence safeguarding individuals and organizations from phishing assaults.

Phishing website detection entails using many techniques, technologies, and analysis methodologies to differentiate between legal and malicious websites; the most popular way is machine learning.

Proposed System

The goal of the "Detection of Fraudulent Websites Using ML" work is to identify fraudulent or phishing websites and offer users with a safety % indicating the amount of danger associated with the website. Based on the safety percentage displayed, an individual can proceed to the website with one click. We can accomplish precise identification of fraudulent websites by training machine learning algorithms, making them an effective and efficient method for detecting and flagging potentially misleading online platforms.

One of the most popular and effective kinds of machine learning is manage machine learning. Supervised learning is utilised anytime a given collection of characteristics is used to predict a certain outcome/label. Creating a machine learning model using our training set of features-label pairs in order to produce accurate predictions for fresh, never-before-seen data.

System Architecture

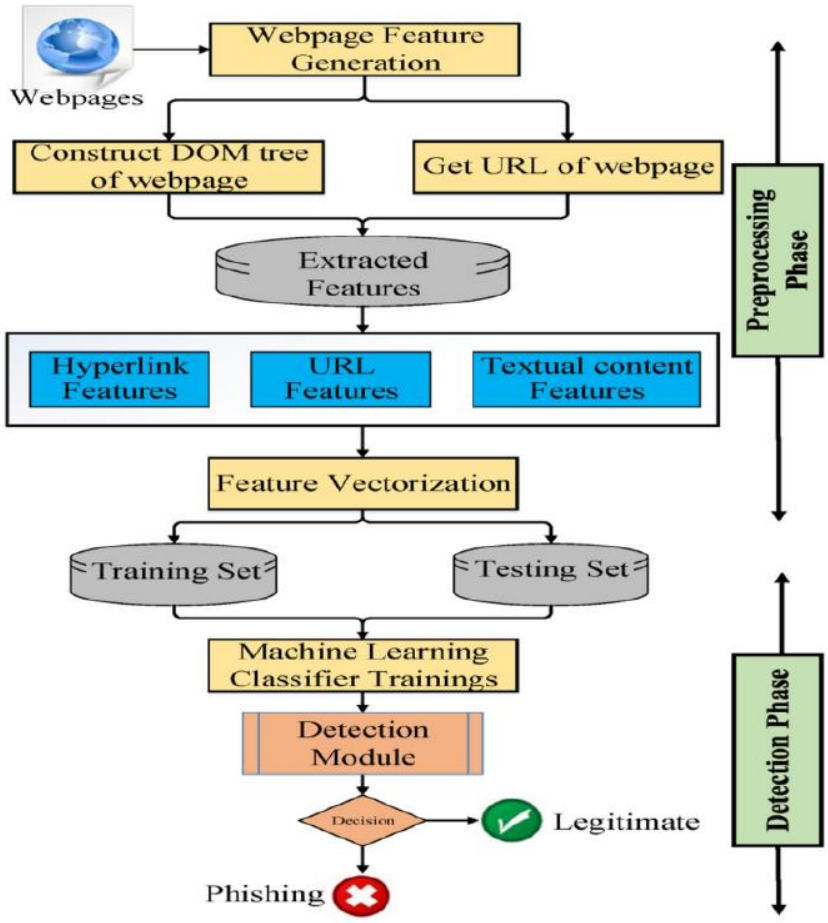


Fig.1. System Architecture

A Python programmed has been created to haul out skin texture as of URLs. The skin texture that we have culled for phishing URL detection is listed below.

IP address in URL: The attribute is set to 1 if an IP address is present in the URL; alternatively, it is set to 0. A URL to download a webpage that contains an IP address is rarely utilized by reliable websites. If an IP address appears in a URL, the attacker may be looking for sensitive information.

@ Symbol in the URL: The attribute is set to 1 if the @ symbol appears in the URL; alternatively, it is set to 0. The web browser overlooks everything before the "@" characters and usually skips over the actual address whenever phishes add a particular @ symbol to a URL. [4].

Hostname dot count: Phishing URLs frequently include a lot of dots in them. For instance, in the URL http://shop.fun.amazon.phishing.com, the word "amazon" is used to deceive users into clicking on it even though phishing.com is a real domain name. The typical amount of dots in safe URLs is 3. The feature is set to 1 if there are more than 3 dots in the URLs; otherwise, it is set to 0.

The dash (-) character is used to denote the prefix or suffix in the domain designation: If the domain address has the attribute set to 1, alternatively it is set to 0. The dash symbol is rarely used in legitimate URLs. For customers to believe they have reached a reliable website, phishers add the dash mark (-) to the domain name. For occurrence, the real website address is http://www.onlineamazon.com, but phishers can construct a phoney version of it called http://www.online-amazon.com to trick unwary people.

Redirecting URLs: If "//" is here in the URL path, the attribute is set to 1 otherwise. The user would be encouraged to an added website if the URL path include the character "//" [4].

http://doi.org/10.36893/JNAO.2023.V16I2.0161-0170

If the URL for International Journal of Computer Applications (0975 - 8887) Volume 181 - No. 23, October 2018 46 contains an HTTPS token, the attribute is set to 1; otherwise, it is set to 0. Phishers may add the "HTTPS" tokens to the domain section of a URL so as to trick consumers. Check out http://https-wwwpaypal-it-mpp-home.soft-hair.com [4] as an illustration.
The user's information may be forwarded to the phisher's personal email account using the "mail()" or "mailto:" functions[4]. If the URL contains such functions, the attribute is set to 1; otherwise, it is set to 0.
URL shortening services like "TinyURL" enable phishers to conceal lengthy phishing URLs by making them brief. User traffic is being diverted to fraudulent websites. If the URL is shortened by means of a service similar to bit.ly, then the attribute is set to 1, otherwise it is set to 0.
The server name's length: The typical length of innocent URLs is found to be 25, and if the URL is for a longer time the attribute is set to 1; alternatively, it is set to 0.
The URL contains hazardous terms: In order to give consumers the idea that they are hitting a reliable website, phishing websites add crucial terms in their URLs. The words "confirm," "account," "banking," "secure," "ebyisapi," "webscr," "signin," "mail," "install," "toolbar," "backup," "Paypal," "password," "username," others. are frequently seen in phishing URLs;
It has been determined that innocent URLs have five slashes; if the URL contains more slashes, the attribute has been set to one; otherwise, it is set to zero.
Unicode characters in the URL: Phishers can utilize Unicode characters in the URL to fool visitors keen on clicking on it. The domain "xn--80ak6aa92e.com," for instance, is equal to "appe.com." The user can see the URL "appe.com," but when they click it, they are taken to the phishing website "xn--80ak6aa92e.com."
Age of SSL Certificate: The presence of HTTPS is crucial in conveying the credibility of a website [4]. However, a benign website's SSL certificate must be at least one to two years old.
URL of Anchor: By crawling the source code of the URL, we were able to extract this feature. The anchor's URL is specified by the tag. The feature is set to 1 if the tag holds a maximum amount of backlinks from another site; otherwise, it is set to 0.
IFRAME: By crowded the URL's source code, we were able to extract this feature. Using this element, one web page can be added to the primary webpage already there. The "iframe" tag allows phishers to make their content invisible, or without frame boundaries. The user may enter critical information because the added webpage's border is invisible and appears to be a part of the original webpage.
Website Rank: We took the rank of individual websites and compared it to the top 100,000 websites in the Alexa database. The feature is set to 1 if the website's rank is more than 10,0000; otherwise, it is set to 0.

Modules and Functionalities:
Data Collection: Gather a dataset of labeled URLs, including both legitimate and phishing URLs. This dataset will serve as the basis for training your machine learning model.
Feature Extraction: Extract relevant features from the URLs that can help distinguish between legitimate and phishing URLs. Common features include domain information, URL length, presence of suspicious keywords, and use of special characters.
Preprocessing: Clean and preprocess the extracted features to ensure they are in a suitable format for training the machine learning model. This may involve tasks such as normalization, encoding categorical variables, and handling missing data.
Machine Learning Algorithms: Choose and implement appropriate machine learning techniques for classification, as phishing URL detection is essentially a binary classification problem. Commonly used techniques comprises decision trees, random forests, support vector machines (SVM), and neural networks.

Divide your dataset into training and testing sets for training. To train your machine learning model on the labelled samples, use the training set. The model gains the ability to spot patterns and traits that distinguish between authentic and phishing URLs.

Evaluation: Employing the testing set, assess how well your model who was trained performed. A few frequent assessments are F1 score, recall, efficiency, and accuracy. This process enables you to gauge how successfully your model generalizes to fresh, untested data.

Deployment: Once you are satisfied with the performance of your model, deploy it in a production environment where it can be used to classify new URLs in real-time. This may involve integrating the model into an existing system or building a new application around it.

IMPLEMENTATION

Libraries

beautifulsoup4==4.9.3-BeautifulSoup: A Python library for parsing HTML and XML documents. It helps extract relevant information from webpages, such as form fields or specific elements, which can aid in identifying phishing elements.

Scikit_learn==1.0.1- A popular machine learning library in Python that provides various algorithms for classification tasks. It be able to be worn to build machine learning representations to detect phishing URLs based on features extracted from URLs.

Flask==2.0.2-Flask is a popular web framework for building Python web applications. However, Flask itself does not provide built-in functionality for detecting phishing URLs or malicious websites.

Flask can be worn as a framework to build a web application that incorporates these techniques for phishing URL detection. However, the actual implementation would involve using other libraries or services to perform the detection, while Flask would handle the web application's routing, request handling, and response generation.

It's important to note that phishing detection is a complex problem, and no single approach can guarantee 100% accuracy. It's always recommended to combine multiple techniques and stay updated with the latest security practices to enhance the effectiveness of phishing URL detection.

Numpy:numpy library is not directly used for phishing URL detection, it can be used as a supporting tool for various data manipulation tasks that are part of the overall phishing URL detection process. Here are a few examples of how numpy can be utilized:

Pandas:The pandas library is a prevailing tool for data manipulation and investigation in Python. Although it is not directly used for phishing URL detection, it can be helpful in several aspects of the process

Ggunicorn:Gunicorn (Green Unicorn) is a Python WSGI (Web Server Gateway Interface) HTTP server commonly used to deploy web applications. While Gunicorn itself is not directly used for phishing URL detection, it plays a role in the deployment and hosting of web applications that may include phishing URL detection functionality.

Ddateutil:The python_dateutil library is not specifically designed for phishing URL detection in machine learning (ML). However, it is a powerful Python library commonly used for parsing, manipulating, and working with dates and times in various formats.

Rrequests: The requests library is a popular Python library used for making HTTP requests to interact with web resources. While it doesn't directly specialize in phishing URL detection, it can be utilized as part of the data collection and retrieval process in machine learning-based phishing URL detection systems.

Whois:The whois command-line utility and protocol are used to retrieve registration information about domain names, IP addresses, and autonomous system numbers from the Internet's domain name registrars and registries. While whois itself is not a Python library, there are Python packages available,

such as python-whois and dnspython, that provide a Pythonic interface to query and retrieve WHOIS information.

Algorithms

The algorithms used are:

Random Forest

The random forest method, one of the most effective machine learning algorithms, is based on the concept of the decision tree algorithm. The method known as random forest produces the forest with numerous decision trees. Numerous instances of trees provide excellent accuracy for detection. The process of creating a tree employs the bootstrap methodology. Employing attributes and samples from the dataset that are randomly selected, a single tree is built utilizing the technique known as bootstrap. The random forest methodology, like the decision tree methods, employs the Gini index and information gain techniques to identify the best splitter among all of the determined criteria for categorization. This process will continue until n trees are produced by the random forest. The target value is predicted by each tree in the forest, and an algorithm then calculates the number of votes for each of the targets estimate. Last but not least, the random forest algorithm makes its final forecast based on the target obtaining the most votes.

Logistic Regression

Logistic regression represents one of the Machine Learning techniques that is most frequently employed in the Supervised Learning group. It is employed to forecast a categorical dependent variable using a specified set of variables that are independent. A logistic regression model is used to forecast the output for a dependent variable that is categorical. This type of statistical model, often known as a logistic model, is extensively used in categorization and analytics for prediction.

Logistic regression determines the chance probability an event, such as voting or not voting, will take place based on a collection of independent variables. Since the outcome is a probability, the range of the variable that is dependent is 0 to 1. The chances are altered using the logistic formula in logistic regression, which is the probability that something will work reduced by the probability of failure.

XG Boost Classifier

XGBoost (eXtreme Gradient Boosting) is a popular and successful open-source version of the gradients boosted trees technique. In order to accurately predict a target variable, the supervised learning method known as gradient boosting integrates a collection of predictions from various weaker and simpler models. The algorithm known as XGBoost performs exceptionally well in machine learning contests due to its robust handling of a variety of data types, correlations, distributions, and a wide range of hyper parameter settings that you may fine-tune. Using XGBoost, problems with regression, classification (binary and multiclass), and ranking can all be resolved. A networked gradient booster library called XGBoost has been enhanced to be exceedingly efficient, versatile, and portable. It implements machine learning methods using the framework known as Gradient Boosting. A parallel tree boosting method called XGBoost (also known as GBDT or GBM) is available to rapidly and precisely address a assortment of data science issues.

Support Vector Machine Algorithm

A support vector machine is a powerful machine learning technique. Using the support vector machine approach, each item of data is represented as a point in n-dimensional space, and a line is also produced to separate the data among two groups. A hyperplane is the name for this line. Support vector machines search for the nearest points, often referred to as support vectors, and after finding them, connect them with a line.

Afterward, an independent line that cuts through and is horizontal to the line that connects them is made employing a support vector machine. Data be supposed to be perfectly classified with the biggest margin possible. In this case, the margin is the separation between the support and hyperplane vectors. Complex and nonlinear data cannot be separated in the real world, hence support vector machines employ a kernel

method to convert lower dimensional room to superior dimensional space in order to tackle this complexity.

K Nearest Neighbours

The non-parametric supervised learning algorithm k-nearest neighbours (k-NN) was developed. Regression and classification are two uses for it. The input in both situations consists of a data set's k closest training samples. Whether k-NN is applied for regression or classification determines the results. After storing all the previous data, an additional point of data is categorized using the algorithm known as K-NN based on similarity. This indicates that new data can be reliably and quickly categorized using the K-NN approach. The K-NN approach is frequently utilized for categorization jobs yet it can be employed as well for regression.

Testing

s.no	Test scenario	Test steps	Test data	Expected Result	Success/ Failure
1	Check with valid phishing websites	Test with phishing dataset	URL 1	Successful in displaying the result	Success
2	Check with legitimate websites	Test without phishing dataset	URL 2	Successful in displaying the result	Success

Table.1. Testing the dataset with phishing and safe websites

s.no	Test scenario	Test steps	Test data	Expected Result	Success/ Failure
1	Check with valid phishing websites	Test entering URL to user interface	URL 3	Successful in displaying the result	Success
2	Check with legitimate websites	Test entering URL to user interface	URL 4	Successful in displaying the result	Success

Table.2. Testing with website URL in the user interface

Results

The user interface

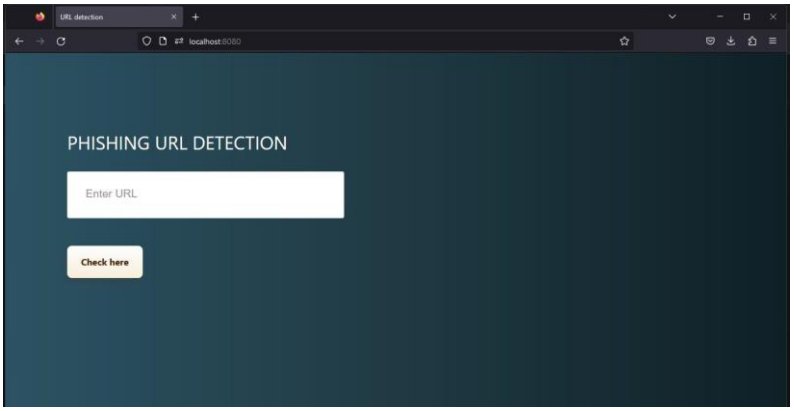


Fig.2. User interface

http://doi.org/10.36893/JNAO.2023.V16I2.0161-0170

Test Case 1

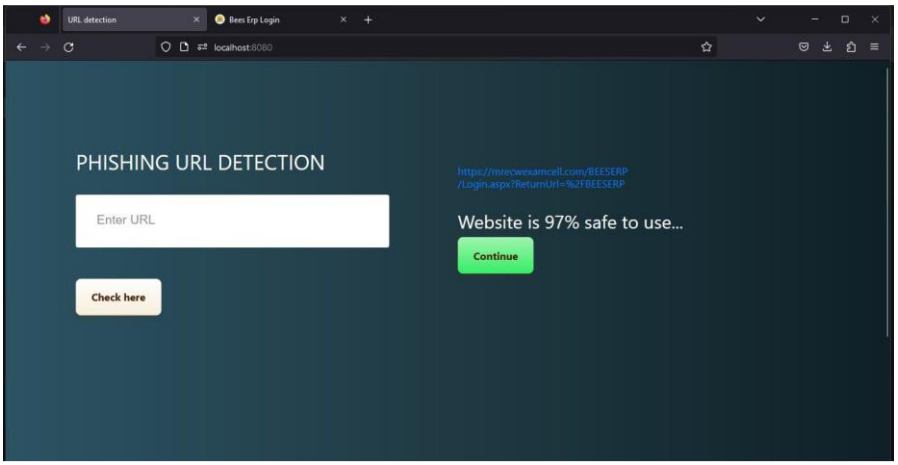


Fig.3. Test case 1

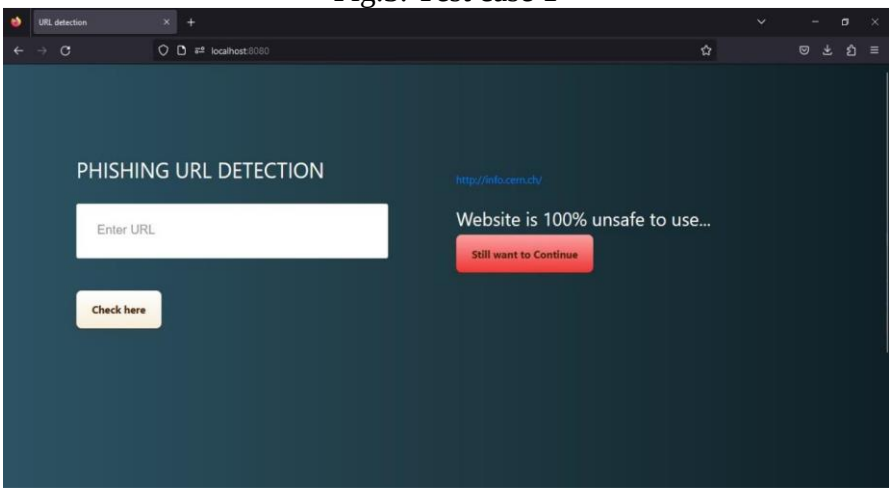


Fig.4. Test case 2

	ML Model	Accuracy	f1_score	Recall	Precision
0	Gradient Boosting Classifier	0.974	0.977	0.994	0.986
1	CatBoost Classifier	0.972	0.975	0.994	0.989
2	XGBoost Classifier	0.969	0.973	0.993	0.984
3	Multi-layer Perceptron	0.969	0.973	0.995	0.981
4	Random Forest	0.967	0.971	0.993	0.990
5	Support Vector Machine	0.964	0.968	0.980	0.965
6	Decision Tree	0.960	0.964	0.991	0.993
7	K-Nearest Neighbors	0.956	0.961	0.991	0.989
8	Logistic Regression	0.934	0.941	0.943	0.927
9	Naive Bayes Classifier	0.605	0.454	0.292	0.997

Fig.5. Comparison of various algorithms accuracy levels

http://doi.org/10.36893/JNAO.2023.V16I2.0161-0170

CONCLUSION

In result, this system is built to ensure that resources are utilized as intended, prevent the loss of important data, provide improved control mechanisms, and notify users when their personal information is at risk. This system might use some changes, similar to any other programmed. Text message integration would be a fantastic suggestion to make to enhance the programmed in the future according to the abilities that the existing system processes. Future iterations of the application might potentially include a feature that allows users to send a text message instantly to a website that has been banned. The list of names could be made available to the programmed as an attachment. The application's functionality would be improved by the ability to integrate text messages.

With the aid of machine learning technologies, this initiative seeks to get better the identification process for phishing websites. By means of the random forest technique, which has the lowest false positive rate, we were able to detect with 97.14% accuracy. Additionally, the consequences reveal that classifiers execute in enhanced while added data is used as training data.

FUTURE SCOPE

In the future, phishing detection will be considerably faster than with any other technique if we have access to structured datasets of phishing. In the future, we can combine any other two or more classifiers to achieve the highest accuracy. In regulate to enhance the system's presentation, we also intend to investigate numerous phishing strategies that make use of web pages' HTML and JavaScript code, network-based features, content-based features, and lexical features. We specifically extract information from URLs and subject them to different classifiers. Although it has been demonstrated that using only URL lexical features can achieve high accuracy (97%), phishers have figured out how to make it challenging to anticipate a URL destination by carefully modifying the URL to avoid detection. Therefore, the best strategy is to combine these traits with others, including host. In order to learn new phishing assault patterns quickly and increase the accuracy of our models with greater feature extraction, we want to construct the phishing detection system as a scalable web service that incorporates online learning.

REFERENCES

1. Gunter Ollmann, “The Phishing Guide Understanding & Preventing Phishing Attacks”, IBMInternet Security Systems, 2007.
2. Mohammad R., Thabtah F. McCluskey L., (2015) Phishing websites dataset. Available: https://archive.ics.uci.edu/ml/datasets/Phishing+Websites Accessed January 2016
3. Mahmoud Khonji, Youssef Iraqi, "Phishing Detection: A Literature Survey IEEE, and Andrew Jones, 2013
4. Matthew Dunlop, Stephen Groat, David Shelly (2010) " GoldPhish: Using Images for Content-Based Phishing Analysis”
5. Rishikesh Mahajan (2018) “Phishing Website Detection using Machine Learning Algorithms”
6. Purvi Pujara, M. B.Chaudhari (2018) “Phishing Website Detection using Machine Learning : A Review”
7. D., Akila. (2019). Phishing Websites Detection Using Machine Learning. 8. 111-114. 10.35940/ijrte.B1018.0982S1119.
8. Parekh, S., Parikh, D., Kotak, S., Sankhe, S.: A new method for detection of phishing websites: URL detection. In: IEEE 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT), April 2018