Detecting a Phishing URL using Machine learning Techniques

-Arun saravanakumar

Phishing:

- This is a social engineering attack that tricks victims into revealing personal information or clicking malicious links.
- Phishing emails are the most common method, and statistics suggest they account for around 30% of all cyberattacks.
- Phishing URLs are frequently used to steal login credentials, such as usernames and passwords, for online accounts.

Overview

- Detecting phishing URLs using machine learning involves training a model on features extracted from URLs and then using that model to classify whether a given URL is phishing or not.
- Data is collected from the kaggle, Extracted relevant features from the URLs. Example long or short urls, HTTPS, Symbol @, domains.
- The performance level of each technique is measured and compared. We select the algorithm with highest accuracy.

Machine learning algorithm for classification

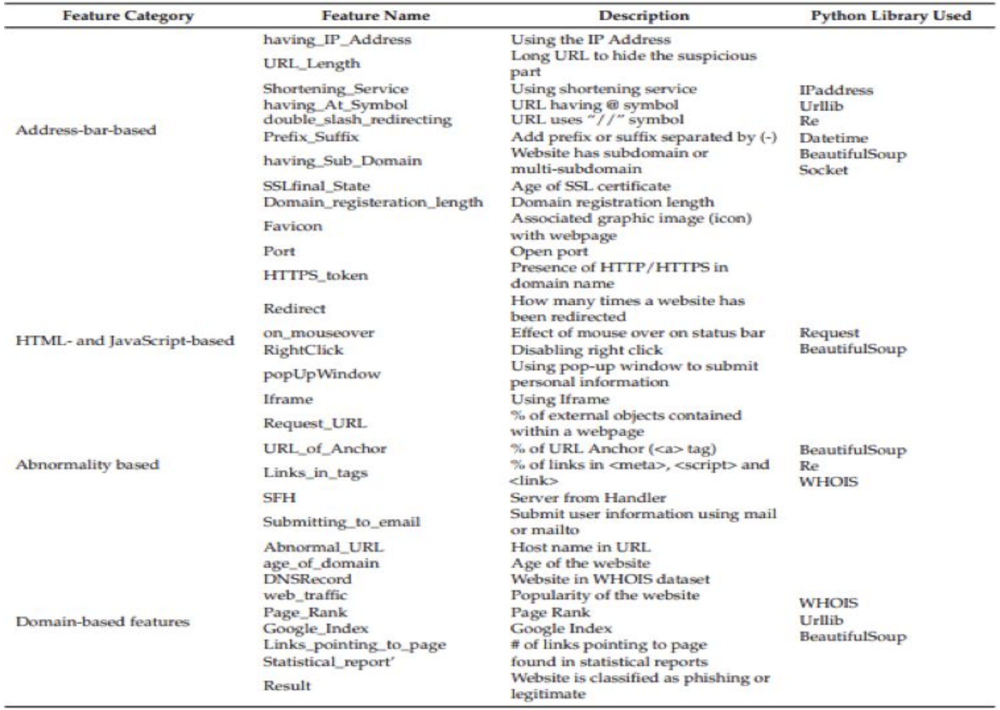
- Gradient Boosting Classifier
- Random Forest
- Decision Tree
- K-Nearest Neighbors
- Logistic Regression

Resources

- Dataset from Kaggle: https://www.kaggle.com/code/eswarchandt/website-phishing/notebook
- Python Interpreter for Code Implementation
- **Input:** Train and Test Datasets with over 11000+ entries
- **Output:** Probability of the given URL being malicious.

Packages:

- Classifier algorithms were imported using sklearn.



Feature Category	Feature Name	Description	Python Library Used
Address-bar-based	having_IP_Address	Using the IP Address	IPaddress Urllib Re Datetime BeautifulSoup Socket
	URL_Length	Long URL to hide the suspicious part	
	Shortening_Service	Using shortening service	
	having_At_Symbol	URL having @ symbol	
	double_slash_redirecting	URL uses “//” symbol	
	Prefix_Suffix	Add prefix or suffix separated by (-)	
	having_Sub_Domain	Website has subdomain or multi-subdomain	
	SSLfinal_State	Age of SSL certificate	
	Domain_registeration_length	Domain registration length	
	Favicon	Associated graphic image (icon) with webpage	
	Port	Open port	
	HTTPS_token	Presence of HTTP/HTTPS in domain name	
HTML- and JavaScript-based	Redirect	How many times a website has been redirected	Request BeautifulSoup
	on_mouseover	Effect of mouse over on status bar	
	RightClick	Disabling right click	
	popUpWindow	Using pop-up window to submit personal information	
	Iframe	Using Iframe	
Abnormality based	Request_URL	% of external objects contained within a webpage	BeautifulSoup Re WHOIS
	URL_of_Anchor	% of URL Anchor (<a> tag)	
	Links_in_tags	% of links in <meta>, <script> and <link>	
	SFH	Server from Handler	
	Submitting_to_email	Submit user information using mail or mailto	
	Abnormal_URL	Host name in URL	
Domain-based features	age_of_domain	Age of the website	WHOIS Urllib BeautifulSoup
	DNSRecord	Website in WHOIS dataset	
	web_traffic	Popularity of the website	
	Page_Rank	Page Rank	
	Google_Index	Google Index	
	Links_pointing_to_page	# of links pointing to page	
	Statistical_report'	found in statistical reports	
	Result	Website is classified as phishing or legitimate	

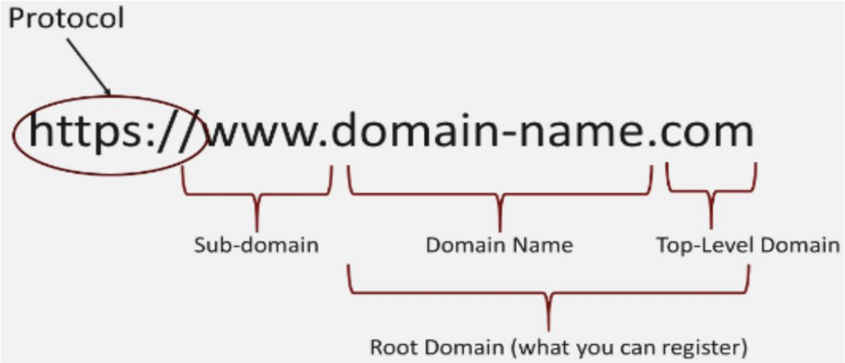
1. **BeautifulSoup**:
   - **Prefix_Suffix**: Used to parse HTML content and identify specific patterns or structures in URLs.
   - **HTTPS_token**: Ued to parse URLs and verify the presence or format of HTTPS in links.
   - **Request_URL**: Used to parse HTML documents and identify external objects referenced in the webpage.
   - **URL_of_Anchor**: Used for parsing anchor elements in HTML to analyze the links.
   - **Links_in_tags**: Used to parse and count links embedded within script and meta tags in HTML documents.
2. **Requests**:
   - **Redirect**: Used to make HTTP requests and follow redirects to count how many times a website has been redirected, which can help determine if redirection is being abused for phishing.
3. **Urllib & Datetime**:
   - **having_At_Symbol**: Libraries such as urllib are used to analyze URLs and datetime might be involved in timestamping and handling time-based features.
4. **Re (Regular Expression)**:
   - **Shortening_Service**: Used for pattern matching to check if a URL is shortened, which often involves regular expressions to detect typical patterns of URL shorteners.
5. **Socket**:
   - **SSLfinal_State**: This attribute involves using sockets to establish a connection to the server and check the details of the SSL certificate.
6. **WHOIS**:
   - **age_of_domain, DNSRecord, and web_traffic**: These attributes is related to analyzing WHOIS data, such as checking the domain age, DNS records, and traffic data related to the domain's popularity and legitimacy.

Attributes:

1. **UsingIP**: Checks whether the URL contains an IP address instead of a domain name. Using an IP address can be a sign of phishing, as it can obscure the real domain
2. **LongURL**: Evaluates the length of the URL. Longer URLs are often used by phishers to hide suspicious parts.
3. **ShortURL**: Determines if the URL has been shortened using services like bit.ly or goo.gl. Phishers use this technique to disguise malicious links.
4. **Symbol@**: Check for the presence of '@' in the URL. The '@' symbol can redirect the browser to a different website, as it is often used in phishing to trick users.
5. **Redirecting//**: Checks if the URL has "//" after the protocol. This could be an attempt to confuse the browser about which part of the URL is the actual domain.
6. **PrefixSuffix**: Looks for '-' in the domain name, e.g., example-phishing-site.com. Phishers use such URLs to create deceptive versions of legitimate websites.
7. **SubDomains**: Examines the number of subdomains in the URL. Multiple subdomains can be a sign of complexity used to confuse users.
8. **HTTPS**: Indicates whether the website uses HTTPS, providing secure communication. Phishing sites might not use HTTPS, or they might use it improperly.
9. **DomainRegLen**: Refers to the length of the domain registration. Short-term registrations might indicate a phishing site, as phishers often use domains for a brief time.
10. **Favicon**: Checks if the website's favicon is loaded from a different domain other than the website's domain, which might indicate a phishing attempt.
11. **NonStdPort**: Examines if the website uses a non-standard port. This is uncommon for most legitimate sites and can be a phishing indicator.
12. **HTTPSDomainURL**: Reviews if the domain in the URL matches the domain in the SSL certificate. Mismatches could signal a phishing site.
13. **RequestURL**: Assesses how many external objects are requested from different domains. Phishing sites often gather content from various unsecured sources.
14. **AnchorURL**: Looks at the percentage of hidden links or links going to different domains. A high percentage can be indicative of a phishing site.
15. **LinksInScriptTags**: Evaluates the ratio of links in scripts that lead to external websites. A high ratio could be suspicious.
16. **ServerFormHandler**: Checks if the data submitted through forms is sent to an external domain, which is highly suspicious and indicative of data theft.

DataSet:

- A comprehensive dataset from Kaggle, containing over 11,000 entries of phishing and legit URLs. Values 1 and -1 are used to classify the websites as legitimate and phishing respectively


Protocol
https://www.domain-name.com
Sub-domain
Domain Name
Top-Level Domain
Root Domain (what you can register)

Approach:

❑ The dataset which is a combination of Phishing and Legit URLs.

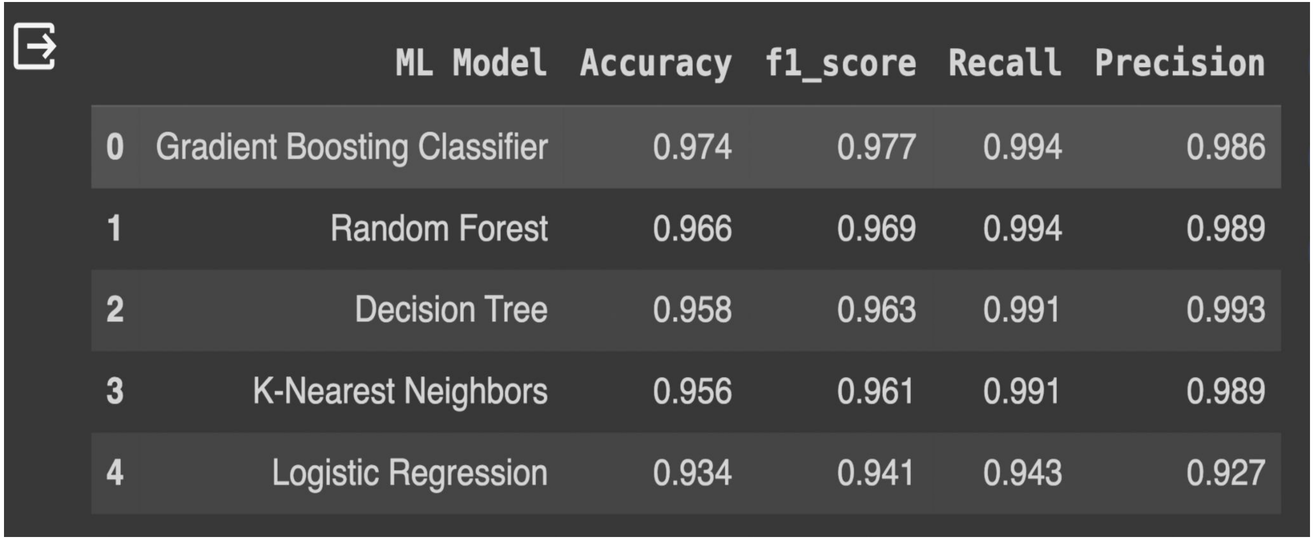
❑ Implementing the code to extract the required features from the database.

❑ Data Splitting

❑ List of Models: Logistic Regression, Decision Trees, K-Nearest Neighbors, Random Forest, Gradient Boosting.

❑ When it comes to Model Evaluation we check for performance metrics like Accuracy, Precision, Recall, F1-Score.

Evaluation:



	ML Model	Accuracy	f1_score	Recall	Precision
0	Gradient Boosting Classifier	0.974	0.977	0.994	0.986
1	Random Forest	0.966	0.969	0.994	0.989
2	Decision Tree	0.958	0.963	0.991	0.993
3	K-Nearest Neighbors	0.956	0.961	0.991	0.989
4	Logistic Regression	0.934	0.941	0.943	0.927

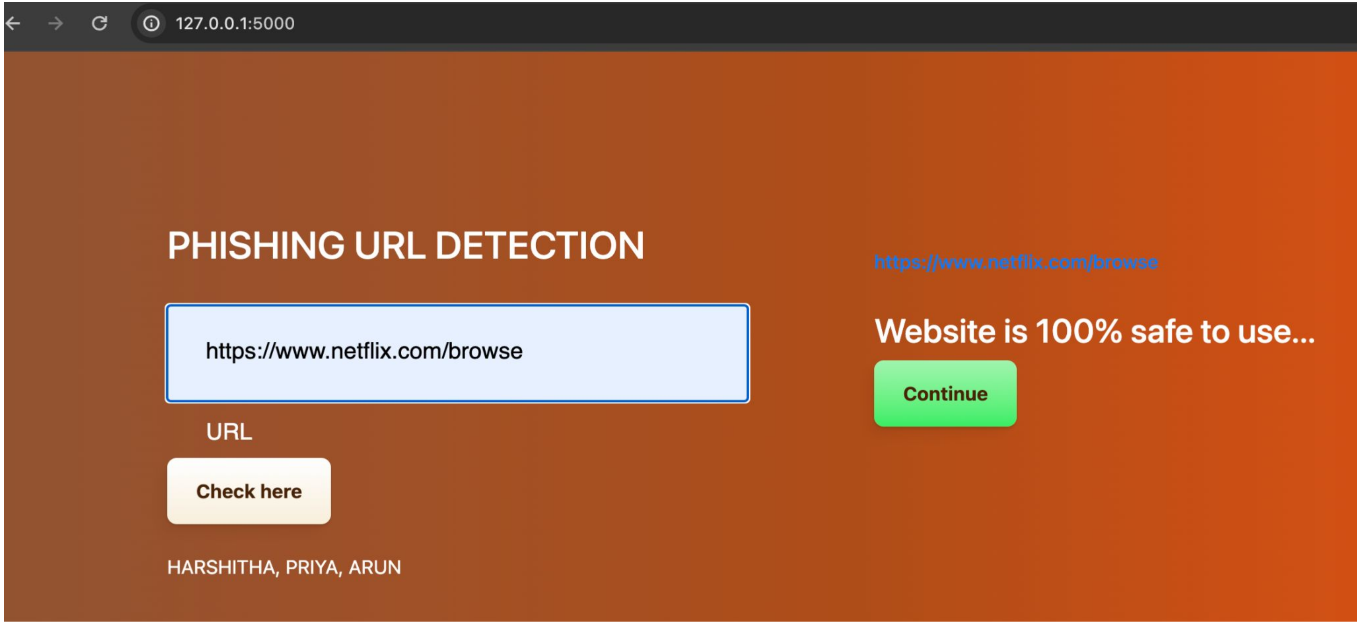
Models:

- Gradient Boosting is very effective at handling a mix of categorical and continuous data, as seen in our dataset which includes URL features like HTTP status, URL length, etc.
- KNN is sensitive to the scale of the data and noisy features, which can reduce its effectiveness in a dataset with diverse attribute scales.
- Phishing URL detection often involves complex patterns that are better captured by non-linear models.
- Decision Trees can easily overfit on training data, especially with a complex feature set like URL attributes, making them less generalizable to unseen data.
- While Random Forest reduces variance by averaging multiple decision trees, it can still be relatively complex and computationally expensive, requiring more resources for training and inference.

Overview of System Components:

- Front-End: The user interface where URLs are entered.
- Processing the URL (Back-End): The back-end of our website, built using a Python framework Flask which receives the URL.
- Sending Results Back to Front-End: Once the prediction is made, the result (safe or unsafe) along with a probability score or a certainty level (like 100% safe) is packaged into a response and sent back to the front-end.
- Displaying the Results: The front-end displays the result in a designated area on the webpage.

When website is safe to use:



PHISHING URL DETECTION

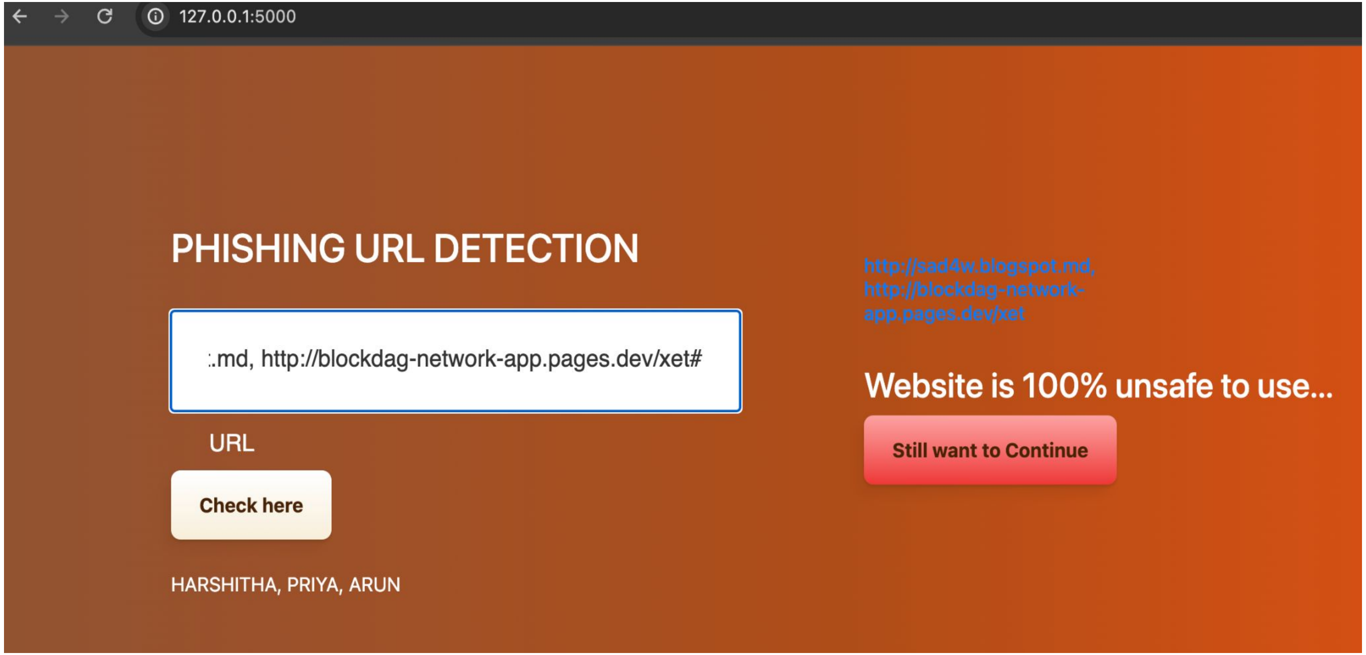
https://www.netflix.com/browse

HARSHITHA, PRIYA, ARUN

https://www.netflix.com/browse

Website is 100% safe to use...

When website is not safe to use:



127.0.0.1:5000

PHISHING URL DETECTION

:.md, http://blockdag-network-app.pages.dev/xet#

URL

HARSHITHA, PRIYA, ARUN

http://sad4w.blogspot.md,
http://blockdag-network-app.pages.dev/xet

Website is 100% unsafe to use...

References:

[1] J. Gu and H. Xu, “An ensemble method for phishing websites detection based on XGBoost,” in *2022 14th international conference on computer research and development (ICCRD)*, 2022, pp. 214–219.

[2] A. Maini, N. Kakwani, B. Ranjitha, M. Shreya, and R. Bharathi, “Improving the performance of semantic-based phishing detection system through ensemble learning method,” in *2021 IEEE mysore sub section international conference (MysuruCon)*, 2021, pp. 463–469.

[3] A. Pandey, N. Gill, K. Sai Prasad Nadendla, and I. S. Thaseen, “Identification of phishing attack in websites using random forest-svm hybrid model,” in *International conference on intelligent systems design and applications*, 2018, pp. 120–128.

[4] https://towardsdatascience.com/phishing-domain-detection-with-ml-5be9c99293e5

[5] https://www.sciencedirect.com/science/article/abs/pii/S0957417418306067

Thank You.. :)