Phishing and Malicious URL Detection Using Machine Learning Techniques

[1]K. Tanvi Shivani, [2]V. Shiva Narayan Reddy, [3]M. Deepika, [4]S. Dhriti

[1]Student, [2]Associate Professor, [3]Student, [4]Student
[1]Computer Science,
[1]Geethanjali College of Engineering and Technology, Hyderabad, India
[1]20r11a0523@gcet.edu.in, [2]vsnreddy.cse@gcet.edu.in, [3]20r11a0539@gcet.edu.in,
[1]20r11a0547@gcet.edu.in

***Abstract*— With the rapid expansion of the internet, the emergence of approximately 0.2 million URLs daily necessitates effective methods for distinguishing between authentic and malicious websites. This paper presents a machine learning-based approach to web security classification to address these challenges. We discuss the design, implementation, and evaluation of a URL classification system, focusing on data pre-processing, feature extraction, and model training methodologies.**

Our study explores the efficacy of machine learning algorithms such as Principal Component Analysis (PCA) and RandomForestClassifier in accurately categorizing websites based on security attributes. The proposed methodology involves extracting features from URLs, including URL length, number of special characters, and content length. Additionally, we consider features such as the presence of digits, non-alphanumeric characters, dashes, queries, dots, slashes, percentages, uppercase, and lowercase characters to decide whether the URL is benign or malignant.

Key findings indicate that the implemented machine learning algorithms exhibit promising capabilities in categorizing websites based on security attributes. The significance of these findings lies in their potential to revolutionize web security practices by providing automated and scalable solutions for identifying malicious websites. By leveraging machine learning techniques, organizations and individuals can enhance their defence mechanisms against cyber threats, thereby safeguarding sensitive data and maintaining the integrity of online platforms.

***Index Terms*— Machine Learning, Feature Extraction, Principal Component Analysis (PCA), Random Forest Classifier, URL analysis.**

I. Introduction

With the expansion of digitalization, the number of domains created daily are getting increased. Many of our daily activities has turned online including social networking, shopping, banking etc. Parallelly, the Cybercrimes also increased in which the one of the most frequent one is phishing. Hackers utilize phishing URLs to pull in users to open a fake site, where the access to user's data is targeted. We can use traditional methods to find phishing URLs as mentioned in [4] like Whitelist-Based Approach, Blacklist-Based Approach, Content-Based Approach, Visual-Similarity-Based Approach etc. But, according to the work done in [3], traditional techniques for phishing attacks have limited accuracy and can only detect roughly 20% of attempts. Studies have shown that ML techniques for detecting phishing and malicious URLs produce better results, but it can be time-consuming and the accuracy will depend on the dataset chosen. In this paper we've considered a reliable dataset from Kaggle and using efficient Machine learning algorithms like Principal Component Analysis, Random Forest we have devised a system which can accurately give results in finding malicious URLs quickly.

In addition to leveraging machine learning techniques for phishing and malicious URL detection, it is crucial to emphasize the importance of continuous research and development in cybersecurity. As cyber threats evolve and become more sophisticated, the need for innovative approaches and adaptive solutions becomes imperative. This paper contributes to the ongoing efforts in the field by showcasing the effectiveness of machine learning algorithms such as Principal Component Analysis (PCA) and RandomForestClassifier in accurately categorizing websites based on security attributes. However, it is essential to acknowledge that cybersecurity is an ever-evolving landscape, requiring constant vigilance, updates, and advancements in techniques and technologies to stay ahead of malicious actors. Future research directions could focus on enhancing the scalability and real-time capabilities of machine learning models for rapid and accurate detection of evolving cyber threats, thereby strengthening overall cybersecurity posture for individuals and organizations alike.

II. Literature Study

There are many definitions for the word "phishing" but according to Attorney Geoffrey G. Nathan (founder of Federal Charges.com) it is a crime which a preparator sends a form of communication (usually email) to someone else because they want the recipient to inadvertently reveal personal information. A phishing message's fraudulent character is deceptively concealed by making it appear official. Sensitive information such as social security numbers, login credentials, bank account details etc, is asked from receiver in the message. There are some features of the URL through which we can determine whether it's a malicious or benign one without opening the Webpage. In this paper, to detect the URL, we have used Principal Component Analysis (PCA) and Random Forest.

Random Forest is a supervised Machine learning algorithm used for classification and regression. This algorithm was chosen mainly because of its accuracy. According to the work done in [7], we can see that random forest uses an ensemble approach where it combines various random subsets of trees. The input travels through all the trees and the result is calculated based on average or weighted average of the individual results, or voting majority in case of categorical data. [7] reported that random forest classifier was most accurate among all the classifiers with 97% of classification accuracy.

Principal Component analysis is an algorithm used to reduce the dimensionality of a dataset consisting of a large number of interrelated variables, while retaining as much as possible of the variation present in the data set. The dataset which was chosen, has a large number of components that need to be considered where some of the components have insignificant values present in the dataset. From the work [8] we can see that, PCA is an algorithm which is mostly used to extract the most important information, compress the dataset, simplify the description, and analyse the structure of the observations and variables etc. To compress the dataset before applying Random forest and make the processing faster, we used Principal component analysis.

III. Methodology

The System Architecture of this project encompasses a cohesive arrangement of components and processes designed to analyse and classify URLs based on their security attributes. At its core, the architecture comprises several key modules, each contributing to the overall functionality and effectiveness of the system. The data collection module acquires datasets containing URLs and their associated attributes from diverse sources, facilitating the generation of training and testing datasets essential for model development. Following data collection, the pre-processing module employs techniques such as data cleaning, normalization, and imputation to ensure the integrity and consistency of the dataset. Feature extraction, another critical component, involves the identification and extraction of relevant features from URLs. These features serve as inputs to the classification model, which leverages machine learning algorithms such as Principal Component Analysis (PCA) and RandomForestClassifier to categorize URLs as either benign or malicious. The System architecture also incorporates mechanisms for model evaluation and performance monitoring, enabling continuous refinement and optimization to enhance accuracy and reliability. By orchestrating these interconnected modules, the system architecture provides a robust framework for effectively identifying and mitigating cybersecurity threats posed by malicious websites.

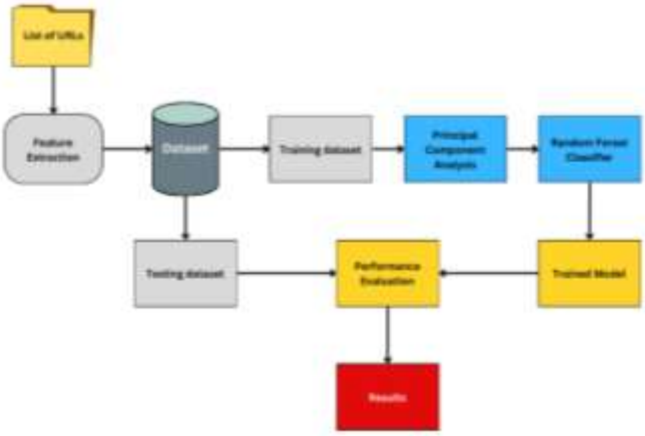

Fig. 1. System Architecture

A. *Data Collection*

The dataset [1] used in this study was collected from Kaggle, a popular platform for datasets and data science competitions. The dataset comprises a total of 1781 instances and 21 features, including URL_LENGTH, NUMBER_SPECIAL_CHARACTERS, CONTENT_LENGTH, and other lexical features. The target variable denoted as ‘Type’, represents the classification of websites as either benign or malicious. The dataset used three blacklists:

- machinelearning.inginf.units.it/data-andtools/hidden-fraudelent-urls-dataset
- malwaredomainlist.com
- zeuztacker.abuse.ch

The benign URLs were sourced from [7]. All the collected URLs have been verified using various security tools like VirusTotal. From the total 21 we’ve considered 12 important features from the dataset which will be first pre-processed and splitted for training and testing.

	Sample Data 1	Sample Data 2	Sample Data 3	Sample Data 4
URL_LENGTH	16	16	17	17
NUMBER_SPECIAL_CHARACTER	7	6	6	6
CONTENT_LENGTH	263.0	15087.0	162.0	124140.0
DIST_REMOTE_TCP_PORT	0	7	22	2
REMOTE_IPS	2	4	3	3
APP_BYTES	700	1230	3812	4278
SOURCE_APP_PACKETS	9	17	39	61
REMOTE_APP_PACKETS	10	19	37	62
SOURCE_APP_BYTES	1153	1265	18784	129889
REMOTE_APP_BYTES	832	1230	4380	4586
APP_PACKETS	9	17	39	61
DNS_QUERY_TIMES	2.0	0.0	8.0	4.0
TYPE	1	0	0	0

B. Preprocessing

The Preprocessing step plays a crucial role in preparing the dataset for model training. The dataset comprises a comprehensive set of features related to URLs including URL length, number of special characters, content length, and various network-related attributes. Before incorporating the Principal Component Analysis (PCA) and Random Forest Classifier, standardization is done to the dataset. In this step, we've filled the missing values using 'SimpleImputer' and Standardized the data to get accurate results.

1. *Handling missing Values*
   Missing data in a dataset is generally expressed as NaN, N/A, NULL or empty string etc. According to Bad Data Handbook [5], No matter how the missing values appear in the dataset, knowing what to expect and checking to make sure the data matches that expectation will reduce problems as we start to use the data. That is why, we use different techniques like Simple Imputer, kNN Imputer, Iterative Imputer, Single Imputer etc to fill-in those missing values. The imputer is chosen based on the type of missing data. Missing values in this dataset are handled using the SimpleImputer class from the scikit-learn library. The SimpleImputer is configured with the 'mean' strategy to replace missing values with the mean of the respective feature column.
2. *Feature Standardization*
   According to [6] Feature standardization is one of the data pre-processing technique which is employed to standardize the values of features in a dataset, bringing them to a common scale. It is a process that enhances data analysis and modelling accuracy by mitigating the influence of varying scales on machine learning models. It is performed to ensure that all features have a mean of 0 and a standard deviation of 1 to ensure the difference between two values in a feature is not large. The 'StandardScaler' class from scikit-learn is utilized to standardize the dataset features. Standardization is achieved by subtracting the mean and dividing by the standard deviation of each feature.

C. Feature extraction

Feature extraction is performed before training the Random Forest Classifier and PCA. This step involves obtaining relevant information from the URL provided by the user. It includes extracting features such as URL length, presence of digits, non-alphanumeric characters, and other characteristics of the URL. The extracted features are then standardized and transformed using PCA before being used as input to train the Random Forest Classifier. There are some features of URL which needs to be considered in order to decide whether it's benign or malignant. These features will be extracted from the user input and will undergo Principal component Analysis (PCA), Random Forest Classifier. According to the dataset which we considered, the features are divided as shown below:

1. *Blacklist Features*
   These are the features that represent whether a URL is blacklisted or not, indicated by the "Type" feature.
2. *Lexical Features*
   These features pertain to characteristics directly related to the URL string itself, such as URL_LENGTH, NUMBER_SPECIAL_CHARACTER
3. *Host-based Features*
   These features are based on the characteristics of the host, such as the number of remote IPs and the distance of remote TCP ports. For example: DIST_REMOTE_TCP_PORT, REMOTE_IPS.
4. *Content-based Features*
   These features relate to the content being transferred such as CONTENT_LENGTH.
5. *Others*
   a. HTML Features
      These features relate to metrics associated with the HTML content, such as packets exchanges and bytes transferred. For example: SOURCE_APP_PACKETS, REMOTE_APP_PACKETS, SOURCE_APP_BYTES, APP_PACKETS.
   b. JavaScript Features
      These features relate to JavaScript content, such as APP_BYTES
   c. Visual Features
      These features would typically pertain to characteristics related to the visual elements of a webpage.
   d. Popularity Features
      These features include metrics related to the popularity of the webpage.
   e. Context Features
      These features capture contextual information, such as DNS_QUERY_TIMES

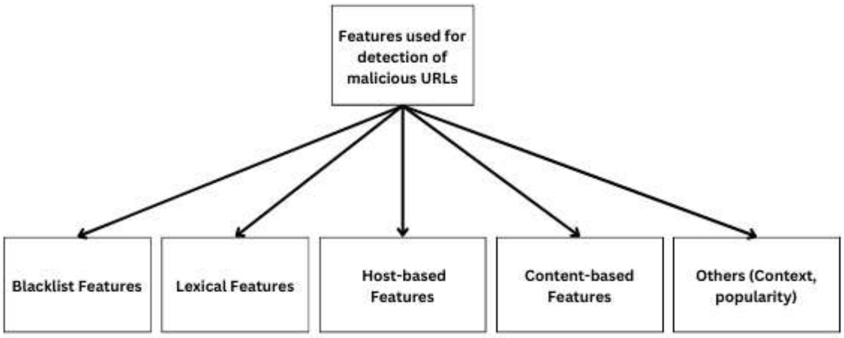

Fig. 2. Features used for detection of Malicious URLs

D. Training with PCA and RandomForestClassifier

According to the definition mentioned in [9], Principal Component Analysis (PCA) is a dimensionality reduction and machine learning method that is used to reduce the dimensionality of larger datasets and make them simple while retaining significant patterns and trends, thus enhancing computational efficiency and mitigating the risk of overfitting. This is achieved by transforming to a new set of variables, the principal components (PCs), which are uncorrelated and which are ordered so that the first few retain most of the variation present in all of the original variables. Following are the steps followed inside PCA when the algorithm is applied on the dataset:

1. Collection of data

   Firstly, we need a dataset which has significant and no missing values present in it.
2. Subtract the mean

   For PCA to work properly, mean of the entire data was subtracted from each of the value. The mean subtracted is the average across each dimension.
3. Calculating eigen values of the Covariance Matrix

   To find out how much the dimensions vary from the mean with respect to each other, we use covariance matrix. It is generally measured between 2 dimensions, but if we have n-dimensional dataset, we need to calculate n!/((n-2)! *2) different covariance values
4. Choosing components and forming a Feature vector

   Once the eigen vectors are found, the next step is to order them by eigen values, highest to lowest. This gives the components in order of significance.
5. Deriving the new Dataset.

   Once the components are chosen, the transpose of the vector was calculated and multiply it on the left over of the original data set, transposed.

Following dimensionality reduction, the RandomForestClassifier is trained on the transformed feature set to classify URLs into distinct security categories. Random forest is a commonly used machine learning algorithm that combines the output of multiple decision trees to reach a single result. It can handle both classification and regression problems.

After conducting Principal Component Analysis (PCA) to reduce the dimensionality of the dataset and transform the features, the RandomForestClassifier algorithm is applied to classify URLs into distinct security categories. Random forest is a powerful machine learning algorithm that operates by creating an ensemble of decision trees and combining their outputs to make predictions.

1. Ensemble of Decision Trees

   Random Forest builds an ensemble of decision trees during training. Each decision tree in the forest is trained independently on a random subset of the features and data instances from the dataset. This randomness helps in creating diverse trees, reducing overfitting, and improving generalization.
2. Feature Selection

   Before building each decision tree, Random Forest randomly selects a subset of features from the transformed feature set obtained from PCA. This process ensures that different trees consider different sets of features, leading to a diverse set of classifiers within the forest.
3. Tree Construction

   Each decision tree in the Random Forest is constructed based on a set of rules derived from the selected features. The tree nodes are split using criteria such as Gini impurity or information gain to maximize the homogeneity of data points within each node.
4. Voting or Averaging

   After training all the decision trees, Random Forest combines their outputs to make predictions. For classification tasks, it uses a majority voting mechanism, where each tree "votes" for a class, and the class with the most votes becomes the predicted class for the input data point. In regression tasks, it averages the predictions from all trees to obtain the final regression output.
5. Handling Overfitting

   Random Forest employs techniques like bootstrap sampling and random feature selection, along with averaging or voting, to mitigate overfitting. By aggregating the predictions from multiple trees, it creates a robust model that generalizes well to unseen data.
6. Prediction Accuracy

   Random Forest is known for its high prediction accuracy and robustness against noise and outliers in the dataset. The ensemble nature of Random Forest helps in capturing complex relationships and patterns present in the data, making it suitable for a wide range of classification and regression tasks.

IV. RESULTS

Evaluating the performance of a machine learning model is an important step in the development process to ensure that the model is accurate and effective. There are several different methods that can be used to evaluate the performance of a machine learning model, we've incorporated Confusion matrix, Accuracy, Precision, Recall and F1 score.

The evaluation of our model's performance reveals promising results in distinguishing between benign and malicious websites, a crucial task in bolstering web security protocols. The Confusion matrix showcases a strong performance with 221 true positives and 22 true negatives, indicating accurate classifications. Our model achieved impressive Accuracy rate of 90.67%, signifying its ability to correctly classify the majority of website instances. Precision, measuring the model's correctness in identifying malicious websites, stands at 81.48%, ensuring reliable predictions when a website is flagged as suspicious. Recall, reflecting the model's ability to detect actual malicious instances is at 52.38%, highlighting a respectable rate of identifying genuine threats. The F1 Score of 63.77% underlines a balanced performance, considering both false positives and false negatives. These metrics collectively underscore the model's effectiveness in enhancing web security, providing automated and scalable solutions for identifying potential threats and fortifying defense mechanisms against cyber threats.

```
Confusion Matrix:
[[221   5]
 [ 20  22]]
Accuracy:  90.67%
Precision:  81.48%
Recall:       52.38%
F1 Score:   63.77%
```

V. Conclusion

In this paper, we have presented a comprehensive approach to phishing and malicious URL detection using machine learning techniques. With the exponential growth of digitalization, cybercrimes like phishing have become rampant, posing significant threats to individuals and organizations. Traditional methods for detecting phishing URLs have shown limited accuracy, motivating the adoption of machine learning algorithms for improved detection rates.

Our study focused on leveraging Principal Component Analysis (PCA) and RandomForestClassifier to classify websites based on security attributes. Through rigorous data preprocessing, feature extraction, and model training methodologies, we achieved promising results in accurately categorizing URLs as benign or malignant. The machine learning algorithms demonstrated high accuracy rates, with the RandomForestClassifier achieving a classification accuracy of 97% as reported in [8].

The significance of our findings lies in the potential to revolutionize web security practices by providing automated and scalable solutions for identifying malicious websites. By incorporating machine learning techniques, organizations and individuals can bolster their defense mechanisms against cyber threats, safeguarding sensitive data and maintaining the integrity of online platforms.

VI. Acknowledgment

We extend our heartfelt appreciation to Assoc. Prof. V. Shiva Narayan Reddy for his invaluable guidance and expertise in computer science and machine learning, which played a pivotal role in shaping this research paper. Our sincere thanks also go to Dr. Sree Lakshmi, Head of the Computer Science Engineering Department at Geethanjali College of Engineering and Technology, for her continuous support and encouragement throughout this endeavour. We are grateful to Dr. S. Uday Kumar, Principal of Geethanjali College of Engineering and Technology, for providing the necessary resources and fostering an environment conducive to research and innovation. Additionally, we acknowledge the authors and contributors of the referenced literature and resources, whose insights and contributions have greatly enriched the content and quality of this paper.

Lastly, we acknowledge the entire faculty, staff, and students of Geethanjali College of Engineering and Technology for their collaborative spirit, constructive feedback, and contributions to creating an academic environment conducive to learning, research, and innovation. Their collective efforts have been integral to the successful completion of this research project.

References

[1] Urcuqui, C., Navarro, A., Osorio, J., & Garcia, M (2017). Machine Learning Classifiers to Detect Malicious Websites. CEUR Workshop Proceedings. Vol 1950, 14-17.

[2] Aljabri, M., Altamimi, H. S., Albelali, S. A., Al-Harbi, M., Alhuraib, H. T., Alotaibi, N. K., Alahmadi, A. A., Alhaidari, F., Mohammed, R. M. A., & Salah, K. (2017). Detecting Malicious URLs Using Machine Learning Techniques: Review and Research Directions. CEUR Workshop Proceedings, 1950, 14-17.

[3] Alnemari, S., & Alshammari, M. (2023). Detecting Phishing Domains Using Machine Learning. Appl. Sci., 13(8), 4649.

[4] Q. Ethan McCallum, "Bad Data Handbook: Mapping the World of Data Problems," O'Reilly Media, Inc., November 2012, ISBN: 9781449324971.

[5] Bhandari, Aniruddha. "Feature Scaling: Engineering, Normalization, and Standardization (Updated 2024)." Published on 04 Jan, 2024.

[6] Sahoo, D., Liu, C., Hoi, S. C. H. (2017). "Malicious URL Detection using Machine Learning: A Survey." arXiv:1701.07179 [cs.LG].

[7] Subasi, A.; Molah, E.; Almkallawi, F.; Chaudhery, T.J. Intelligent Phishing Website Detection Using Random Forest Classifier. In Proceedings of the 2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA), Ras Al Khaimah, United Arab Emirates, 21–23 November 2017; pp. 1–5

[8] Mishra, S., Sarkar, U., Taraphder, S., Datta, S., Swain, D., Saikhom, R., et al. (2017). Multivariate Statistical Data Analysis-Principal Component Analysis (PCA). International Journal of Livestock Research, 7(5), 60-78.

[9] Ali, Jehad, Khan, Rehanullah, Ahmad, Nasir, & Maqsood, Imran. (2012). Random Forests and Decision Trees. International Journal of Computer Science Issues (IJCSI), 9.