Phishing Email Text and URL Detection using Machine Learning Models

Dr. Amit Ramesh Khaparde
Assistant Professor, Computer Science & Engg.
G. B. Pant DSEU Okhla -I Campus
New Delhi, India
khaparde.amit@gmail.com

Yash Sharma
Final Year Graduate, Computer Science Engineering
G. B. Pant DSEU Okhla -I Campus
New Delhi, India
Yash27jan@gmail.com

Keshav Raturi
Final Year Graduate, Computer Science Engineering
G. B. Pant DSEU Okhla -I Campus
New Delhi, India
kshvraturi@gmail.com

Lovneesh Badhawn
Final Year Graduate, Computer Science Engineering
G. B. Pant DSEU Okhla -I Campus
New Delhi, India
lovee.b123@gmail.com

Authors Name/s per 2nd Affiliation (Author)
line 1 (of Affiliation): dept. name of organization
line 2: name of the organization, acronyms acceptable
line 3: City, Country
line 4: e-mail address if desired

Authors Name/s per 3rd Affiliation (Author)
line 1 (of Affiliation): dept. name of organization
line 2: name of the organization, acronyms acceptable
line 3: City, Country
line 4: e-mail address if desired

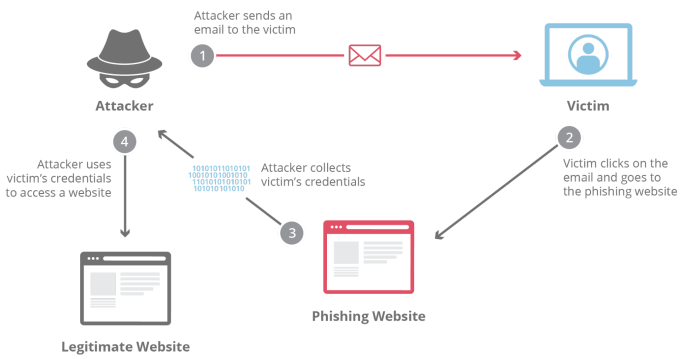
Abstract—.This paper presents a study conducted on a phishing dataset using various machine learning models, such as SVM, LR, RFC, DTC, NB, and NN. The objective was to identify the best model for detecting phishing emails based on their text and URL components. In the process of phishing mail detection, we imported several Python libraries and extracted a tar file to analyze its content and quantity. We utilized word cloud techniques to identify words frequently employed by attackers. Additionally, we applied the aforementioned machine learning algorithms to obtain accuracy scores for the text and URL parts of phishing mails from the dataset..The results revealed that the Decision Tree Classifier (DTC) achieved the highest accuracy for the text component, with a score of 99.87%. As for the URL part, the Neural Network (NN) model demonstrated the best performance with an accuracy of 97.89%

Keywords - Machine learning, phishing mail classification, URL classification, SVM, Logistic Regression, Random Forest, Decision Trees, Naive Bayes, Neural Networks.

I. INTRODUCTION

Phishing is a cybercrime in which a target or targets are contacted by email, telephone or text message by someone posing as a legitimate institution to lure individuals into providing sensitive data such as personally identifiable information, banking and credit card details, and passwords. The information is then used to access important accounts and can result in identity theft and financial loss.

The overall goal of a phishing attack is usually to gain sensitive data such as logins and passwords from their victims in order to access the targeted network or company . Fraudsters often do it for financial gain by stealing credit card details. Emails are a very widespread form of communication with no cyber security. Doesn't require much technical skill. Hackers prefer phishing as it is less complex and it relies on human manipulation.



[1] Fig 1. Overview of a Phishing Attack

There are many supervised techniques generally used for analyzing the phishing detection in Mails. Popularity wise these are Support Vector Machine, Logistic Regression,

Random Forest Classification, Decision Tree and Naive Bayes from shallow part and Neural Network from Deep [2].

Support Vector Machines (SVM) is a commonly used technique for both classification and regression tasks. It works by representing each data item as a point in a multi-dimensional space. The goal of SVM is to find the best hyperplane that can separate the data into two classes. In cases where the data is not linearly separable, SVM uses a kernel function to transform it into a higher-dimensional space where a separating hyperplane exists.

In Logistic Regression (LR) models the probability of an event occurring based on independent variables. It is widely used in various fields, including predicting binary outcomes and understanding the relationship between variables. In the context of phishing email detection, some researchers [2] have utilized logistic regression to train and test their models. By analyzing the characteristics and features of phishing emails, logistic regression can learn patterns and make predictions on whether an email is likely to be a phishing attempt.

A Random Forest Classifier (RFC) is a type of classifier that makes predictions by combining multiple decision trees. It works by creating several decision trees on different subsets of the dataset. Each tree is built using a random selection of the best attributes. During the training phase, the decision trees are constructed based on predefined rules, and these trees are then used to predict the class of new instances.

The Decision Tree Classifer (DTC) is like a flowchart where each node represents a question and the branches represent the possible answers. The algorithm learns from the data to find the best questions to ask and the optimal order of the questions to make accurate predictions. It classifies new data points by following the path through the decision tree based on the answers to the questions until it reaches a leaf node, which represents the predicted class or outcome.

II. RELATED WORK

The research paper "A Systematic Literature Review on Phishing Email Detection Using Natural Language Processing Techniques" by Said Salloum et al. [2] provides a detailed literature review of studies that utilize NLP and ML methods for phishing email detection. It covers various machine learning techniques, datasets used for evaluation, data sources, and search strategies. The review highlights the prevalence of SVM and K-means clustering in ML and the use of the Bag-of-Words approach in NLP.

The research paper "PhishHaven—An Efficient Real-Time AI Phishing URLs Detection System" by Maria Sameen et al. [3] introduces PhishHaven, a real-time AI-based system for detecting phishing URLs. It utilizes lexical features-based extraction, URL HTML encoding, and an ensemble-based machine learning approach. PhishHaven achieved 98.00% accuracy in detecting both AI-generated and human-crafted phishing URLs.

"Learning from the Ones that Got Away: Detecting New Forms of Phishing Attacks" by Christopher N. Gutierrez et al. [4] presents SAFE-PC, a system for extracting features to detect new forms of phishing attacks. It addresses challenges associated with detection using URLs and blacklists, specifically the increase in URL redirection as an obfuscation technique.

The research paper titled "Systematization of Knowledge (SoK): A Systematic Review of Software-Based Web Phishing Detection" by Zuochao Dou et al. [5] provides an overview of software-based phishing detection schemes, their features, life cycle, and evaluation. It offers insights into the different phishing attacks, players involved, and the increasing number of phishing attacks.

"Who Are the Phishers? Phishing Scam Detection on Ethereum via Network Embedding" by Jiajing Wu et al. [6] proposes an approach to detect phishing scams on Ethereum by leveraging network embedding. It uses transactional features, user features, and the Ethereum transaction graph to detect malicious accounts involved in phishing scams.

"SPWalk: Similar Property Oriented Feature Learning for Phishing Detection" by Xiuwen Liu and Jianming Fu [7] presents SPWalk, a phishing detection algorithm based on network embedding. It leverages the structural regularities and URL information of webpages to achieve high precision in phishing detection.

The research paper titled "Phishing-Aware: A Neuro-Fuzzy Approach for Anti-Phishing on Fog Networks" by Chuan Pham et al. [8] proposes a neuro-fuzzy framework called Fi-NFN for detecting phishing websites on fog networks. The framework utilizes URL features, web traffic features, and fog computing to enhance the security of the network.

The research paper titled "AI Meta-Learners and Extra-Trees Algorithm for the Detection of Phishing Websites" by Yazan Ahmad Alsariera et al. [9] proposes four meta-learner models developed using the extra-tree base classifier. These models achieved a detection accuracy of at least 97% with a low false-positive rate, outperforming existing ML-based models in phishing attack detection.

"Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity" by Jian Mao et al. [10] presents an algorithm to quantify the suspiciousness ratings of web pages based on visual similarity. It utilizes CSS and a weighted page-component similarity rating method.

"An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs" by Ayman El Aassal et al. [11] introduces Phishbench, a benchmarking framework for phishing detection. It evaluates over 200 features and 30 classification algorithms and compares PhishBench to previous studies. The performance of PhishBench on balanced and imbalanced datasets showed slightly lower accuracy in most cases and a decline in performance as the ratio of phishing to legitimate class increased.

TABLE 1 : COMPARATIVE STUDY OF SIMILAR SYSTEMS

S No.	Research Paper	Author	Description	
			Method	Result
1	AI Meta-Learners and Extra-Trees Algorithm for the Detection of Phishing Websites	Yazan Ahmad Alsariera , Victor Elijah Adeyemo , Abdullateef Oluwagbemiga Balogun ,, (Member, Ieee), And Ammar Kareem Alazzawi	This study proposed four (4) meta-learner models (AdaBoost-Extra Tree (ABET), Bagging –Extra tree (BET), Rotation Forest – Extra Tree (RoFBET) and LogitBoost-Extra Tree (LBET)) developed using the extra-tree base classifier.	The proposed AI-based meta-learners were fitted on phishing website datasets (currently with the newest features) and their performances were evaluated. The models achieved a detection accuracy not lower than 97% with a drastically low false-positive rate of not more 0.028. In addition, the proposed models outperform existing ML-based models in phishing attack detection.
2	An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs	Ayman El Aassal, Shahryar Baki , Avisha Das, And Rakesh M. Verma , (Member, Ieee)	This study proposed a novel taxonomy about phishing emails and websites and their respective features : 1.**Syntactic 2.Semantic and 3.Pragmatic.** Then it introduces **Phishbench** : an easy-to-use benchmarking framework which has state of the art phishing detection methods and extraction of over 200 features and 30 classification algorithms. It's arcitecture consists of 5 modules : **1. Input module 2. Feature Extraction 3. Feature Ranking 4. Classification Module 5. Evaluation Module.** This study also compares PhishBench to previous studies.	After vigorous experimentation and benchmarking on balanced and imbalanced datasets using **PhishBench**, and comparing them to the **previous reported studies**, the performance for balanced datasets was slightly lower in most cases. A decline in performance was noticed in imbalanced datasets as the ratio of phishing to legitimate class increased.
3	A Systematic Literature Review on Phishing Email Detection Using Natural Language Processing Techniques	Said Salloum, Tarek Gaber, Sunil Vadera & Khaled Shaalan Turn On Screen Reader Support	They have given a detailed literature review of studies that aim to utilise natural language processing (NLP) and machine learning (ML) methods for detecting phishing emails. Comprehensive study of phishing email detection comprises of : 1. Various machine learning techniques that is the methods used in phishing email detection. (supervised & unsupervised classical ML algorithms) 2. Various datasets used for evaluation 3. Data sources and search strategies 4. Quality assessments	According to this Paper : 1. Most common technique* used in phishing email detection in ML is **SUPPORT VECTOR MACHINE (SVM)** [Supervised] & **K-MEANS CLUSTERING** [Unsupervised] and in NLP is **Bag-of-Words (BoW) *In all of this Basic NLP technique was more common** 2. Adam optimizer is most popular among other optimizations techniques. 3 The **Nazario phishing corpus (datasets and resources)** has been in the majority of studies (at 42 studies) 4. **Python** was widely used tool 5. Most Researches focused on **Accuracy** while less in F-score, Confusion Matrix etc.

4	Learning from the Ones that Got Away: Detecting New Forms of Phishing Attacks	Christopher N. Gutierrez , Taegyu Kim, Raffaele Della Corte , Member, Ieee, Jeffrey Avery, Dan Goldwasser, Marcello Cinque, And Saurabh Bagchi , Senior Member, Ieee	They design SAFE-PC (Semi-Automated Feature generation for Phish Classification), a system to extract features, elevating some to higher level features, that are meant to defeat common phishing email detection strategies	They reported increase in URL redirection as a technique to obfuscate phishing websites exemplifies the challenges associated with detection using URLs and blacklists
5	PhishHaven —An Efficient Real-Time AI Phishing URLs Detection System	Maria Sameen 1 , Kyunghyun Han 2 , And Seong Oun Hwang 3 , (Senior Member, Ieee)	PhishHaven system uses **lexical features-based extraction** and analysis techniques. To proactively detect and classify a URL on-the-fly, they additionally introduce **URL HTML Encoding** as a lexical feature to further boost PhishHaven. In addition to this, they introduce **URL Hit**, an approach to effectively detect tiny URLs. Furthermore, they also design a new paradigm for executing **ensemble-based machine learning** for PhishHaven. PhishHaven also employs **unbiased voting** concept in decision-making process to assign final labels (i.e., either phishing or normal) to the URL(s).	PhishHaven (an efficient real-time AI-generated Phishing URLs detection system) identifies AI-generated (generated by **DeepPhish**) as well as human-crafted phishing URLs achieving **98.00%** accuracy, outperforming the existing lexical-based human-crafted phishing URLs detection systems. It can majorly do two things : (1) it can always **detect tiny URLs**, and (2) it can detect future **AI-generated Phishing URLs** based on our selected lexical features with 100.00% accuracy.
6	Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity	Jian Mao1 , (Member, Ieee), Wenqian Tian1 , Pei Li1 , Tao Wei2 , (Member, Ieee), And Zhenkai Liang3 , (Member, Ieee)	It presents an algorithm to quantify the suspiciousness ratings of Web pages based on the similarity of visual appearance between the Web pages.	Their approach uses CSS as the basis to accurately quantify the visual similarity of each page element. As page elements do not have the same influence to pages, they base the rating method on weighted page-component similarity prototyped in the Google Chrome browser.
7	Phishing-Aware: A Neuro-Fuzzy Approach for Anti-Phishing on Fog Networks	Chuan Pham , Luong A. T. Nguyen, Nguyen H. Tran , Eui-Nam Huh, and Choong Seon Hong	They use URL features and Web traffic features to detect phishing websites based on a designed neuro-fuzzy framework (dubbed Fi-NFN). Based on the new approach, fog computing as encouraged by Cisco, & designed an anti-phishing model to transparently monitor and protect fog users from phishing attacks.	The experiment results of their proposed approach, based on a large-scale dataset collected from real phishing cases, have shown that the system can effectively prevent phishing attacks and improve the security of the network.

8	SPWalk: Similar Property Oriented Feature Learning for Phishing Detection	Xiuwen Liu And Jianming Fu	In SPWalk, similar property nodes refer to a collection of phishing webpages or legitimate webpages. SPWalk applies the network **embedding technique** to mapping nodes into a low-dimensional vector space. A biased random walk procedure efficiently integrates both structural information between nodes and URL information of each node. They first constructed a weblink network with nodes representing webpages. The edges between nodes represent the reference relationships that connect webpages through hyperlinks or similar textual content.	The effectiveness and robustness of SPWalk come from 3 points : 1. Phishing attackers do not have full control over **reference relationships**. 2. The **structural regularities** generated by diverse reference relationships can be exploited to discriminate between phishing and legitimate webpages. 3. **Node URL information** (using node as numeric feature) makes the learned node representations more suited for phishing detection.They demonstrated the superiority of SPWalk over **state-of-the-art techniques** on phishing detection, especially in terms of precision (over 95%).
9	Systematization of Knowledge (SoK): A Systematic Review of Software-Based Web Phishing Detection	Zuochao Dou, Student Member, IEEE, Issa Khalil, Member, IEEE, Abdallah Khreishah, Member, IEEE, Ala Al-Fuqaha, Senior Member, IEEE, and Mohsen Guizani, Fellow, IEEE	This study educates us about what **phishing**, the players involved in phishing, the different strategies and targets of phishers, the various state of the art phishing attacks and the steep increase in the number of phishing attacks, an overview of software based phishing detection schemes (its features , their life cycle , the features used, etc.) and its evaluation.	NA
10	Who Are the Phishers? Phishing Scam Detection on Ethereum via Network Embedding	Jiajing Wu , Senior Member, IEEE, Qi Yuan, Dan Lin , Wei You, Weili Chen ,Chuan Chen ,Member, IEEE, and Zibin Zheng , Senior Member, IEEE	They proposes an approach to detect phishing scams on Ethereum by mining its transaction records. Method Summarising in 3 Steps : 1. Crawling the labeled phishing addresses from two authorized websites and reconstruct the **transaction network** according to the **collected transaction records.** 2. Taking the transaction amount and timestamp into consideration, they propose a novel network embedding algorithm called **trans2vec** to extract the features of the addresses for subsequent phishing identification. 3. They adopted the one-class **support vector machine (SVM)** to classify the nodes into normal and phishing ones.	Experimental results demonstrate that the phishing detection method works effectively on Ethereum, and indicate the efficacy of **trans2vec** over existing **state-of-the-art** algorithms on feature extraction for transaction networks. This work is the first investigation on phishing detection on Ethereum via **network embedding** and provides insights into how features of large-scale transaction networks can be embedded.

III. EXPERIMENTAL SETUP

A. Dataset Analysis

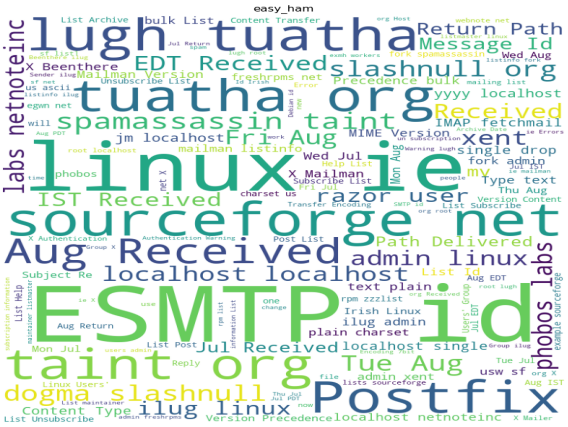
The proposed system used the dataset provided by the Apache SpamAssassin publiccorpus [12] for the experimentation. The dataset contains the real-life sample of email messages that are used for phishing detection and email classification purposes. The dataset includes both phishing (phish) and non-phishing (ham) email samples, which we have used to train machine learning models and evaluate the performance of phishing filters and email classifiers.

The dataset has 3891 total emails on which easy ham (which are easier to detect as non-phishing) are 1401, hard ham (which are hard to detect as non-phishing) are 250 and phishing emails are 2239 in the tar folders.

TABLE I. DATASET

S.No	Type of Datasets	Amount of Data
1	Easy Ham	1401
2	Hard Ham	250
3	Phish Mail	2239

Phishing emails in the "phishingMails" dataset are listed with the ".eml" extension in the "phishingMails/phishingMails" folder, while easy and hard ham files are simple text files with gibberish extensions for example, 0001.f0c..48e is the first mail dataset. Here we performed the word cloud analysis on the text content of all the emails. The results are below.



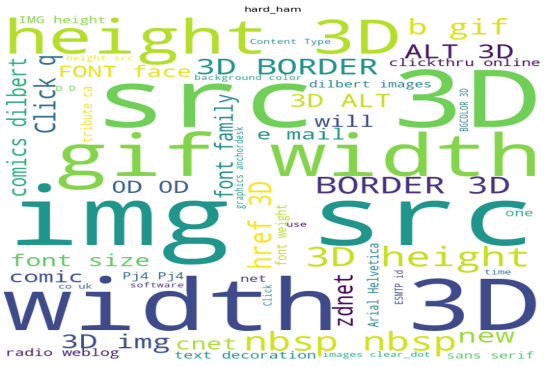


Fig 2. Word Cloud for Easy Ham, Hard Ham and Phishing Email Dataset

B. Data Munging

After observing the word cloud of all these three datasets, we conducted the root word extraction by counting the frequency of each root word. To do this, the NLTK (Natural Language Toolkit) library was used for stemming and count vectorization.

We fit each vectorizer on the corresponding list of root words and transform the data into word counts. Finally, we calculate the word frequency for each dataset by summing the word counts and storing the results in dictionaries.

```
Easy ham 2 word frequency:
{'00': 210, '000': 60, '0000': 22, '00000000': 1,
003': 2, '00004': 2, '00005': 2, '00006': 2, '0000
'000101c231b8': 1, '00011': 2, '0001107': 1, '0001
0017': 2, '00018': 2, '00019': 2, '0002': 1, '0002
025': 2, '00026': 2, '00027': 2, '00028': 2, '0002
56626': 2, '00033': 2, '00034': 2, '00035': 2, '00
0042': 2, '00043': 2, '00044': 2, '00045': 2, '000
052': 2, '00053': 2, '00054': 2, '00055': 2, '0005
62': 2, '00063': 2, '00064': 2, '00065': 2, '00066
2': 2, '00073': 2, '00074': 2, '00075': 2, '00076'
```

Fig 3. Word frequency into dictionaries results

We sorted the word frequencies in descending order and selected the top 20 most frequent words for each dataset. We store the sorted words and their corresponding frequencies in separate lists to create a bar chart using matplotlib.pyplot. We initialize a figure and axis object, set the figure size, and create three bar plots. Each bar plot represents the word frequencies for a specific dataset. Labels are for the x-axis ("Words") and y-axis ("Frequency"), and provide a title for the chart.

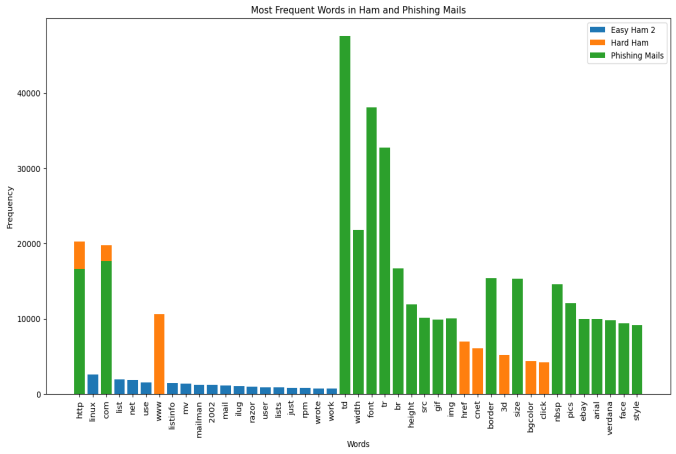


Fig 4. Bar Plot representing the word frequencies for a specific dataset

IV. METHODOLOGY

C. *Evaluation Model*

The dataset extracted from the Tar file was utilized for ML-based models in our research. We specifically extracted features such as the text body and URL. Upon analysis, we observed various shapes within the data. To gain further insights, we generated a word cloud to identify the most frequently used words and their corresponding root word frequencies.

For the email body analysis, we employed the decision tree algorithm and achieved an impressive accuracy of 99.8%. This accuracy outperformed other algorithms such as Support Vector Machines (SVM), Random Forest Classifier (RFC), Logistic Regression (LR), and Naive Bayes (NB).

Regarding the URL analysis, we employed a neural network model, which yielded an accuracy of 97.8%. This accuracy surpassed other algorithms such as Naive Bayes (NB), Decision Trees (DT), Random Forest Classifier (RFC), Logistic Regression (LR), and Support Vector Machines (SVM).

These findings indicate the efficacy of our approach in accurately classifying and analyzing email body content and URLs. The high accuracy rates achieved by the decision tree algorithm for email body analysis and the neural network for URL analysis underscore the suitability of these models for our research objectives.

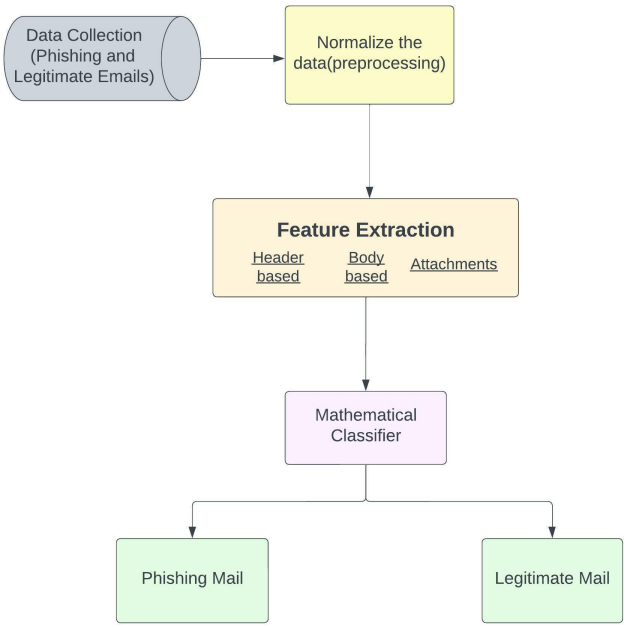


Fig 5. Flowchart to explain the workflow, used dataset ,extracted features, and later found using ML models that mail is phishing or not

D. *Text and Url based Analysis*

Text based filtering was done using Neural Network. After importing the necessary libraries, including TensorFlow, NumPy, pandas, and scikit-learn. We load and read each file in these folders, store the email content in the 'data' list, and assign labels of 0 for non-phishing emails and 1 for phishing emails in the 'target' list.

The 'target' list is converted to a numpy array for further processing. The data is then split into training and testing sets using the 'train_test_split' function from scikit-learn.

To tokenize the text data, the code uses the 'Tokenizer' class from TensorFlow, which converts the text into sequences of integers. The tokenizer is fitted on the training data, and both the training and testing data are transformed into sequences. These sequences are padded to a fixed length using 'pad_sequences' to ensure consistent input shape for the neural network.

The model architecture includes an embedding layer, a flatten layer, a dense layer with ReLU activation, and a dense output layer with sigmoid activation. The model is then compiled with binary cross-entropy loss, the Adam optimizer, and accuracy as the evaluation metric.

This model is trained using the training data, specifying the number of epochs and providing validation data. Once training is complete, the model is used to predict the target values for the test set. The predicted probabilities are thresholded at 0.5 to obtain the final predicted labels.

```
98/98 [==============================] - 1s 7ms/step - loss: 1.8706e-04 - a
acy: 0.9961
25/25 [==============================] - 0s 3ms/step
Metric          Value
Accuracy        0.996144
Precision       0.995370
Recall          0.997680
F1-score        0.996524
Support          [347 431]
ROC AUC         0.995958
              precision    recall  f1-score   support

           0       1.00      0.99      1.00       347
           1       1.00      1.00      1.00       431

    accuracy                           1.00       778
   macro avg       1.00      1.00      1.00       778
weighted avg       1.00      1.00      1.00       778
```

Fig 6. Various metrics such as accuracy, precision, recall, F1-score, support, and ROC AUC are calculated using scikit-learn functions based on the predicted labels and the true labels.

Similarly other machine learning models were also used to find out the results.

For analyzing the URL part we write the unique URLs and their labels to a CSV file named "url.csv". The CSV file is structured with the headers "URL" and "Label". We iterate through each unique URL and label in the unique_urls set and write them as rows in the CSV file.

Multinomial Naive Bayes was applied by performing label mapping by associating the string labels ('Ham' and 'Phish') with numeric labels (0 and 1). We split the data into training and testing sets. In this process, the 'URL' column serves as the feature input (X) while the 'Label' column represents the target output (y). Further we convert the URLs present in the training and testing sets into feature vectors by utilizing the CountVectorizer() class from scikit-learn. This is achieved by applying the fit_transform() method to the training set and the transform() method to the testing set.

To train the Naive Bayes classifier using the MultinomialNB() class from scikit-learn. The classifier is fitted on the training data, comprising X_train and y_train. Once the classifier is trained, we make predictions for the labels of the testing set by invoking the predict() method on X_test. Below is the confusion matrix.

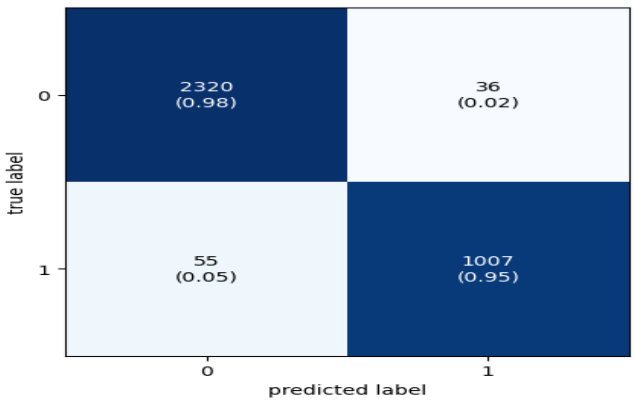


Fig 7.Evaluation Matrix for Multinomial Naive Bayes

V. Results and Discussion

The SVM classifier achieved an accuracy of 99.74%, precision of 99.77%, recall of 99.77%, F1-score of 99.77%, and ROC AUC of 99.74%. The Logistic Regression classifier achieved an accuracy of 99.23%, precision of 98.85%, recall of 99.77%, F1-score of 99.31%, and ROC AUC of 99.16%.

The Random Forest Classifier achieved the same performance as the SVM classifier. The Decision Tree Classifier achieved an accuracy of 99.87%, precision of 100%, recall of 99.77%, F1-score of 99.88%, and ROC AUC of 99.88%.

The Naive Bayes Classifier achieved an accuracy of 95.89%, precision of 93.28%, recall of 99.77%, F1-score of 96.41%, and ROC AUC of 95.42%. Finally, the Neural Network model achieved an accuracy of 99.36%, precision of 99.09%, recall of 99.77%, F1-score of 99.43%, and ROC AUC of 99.35%.

All the classifiers performed well in classifying the emails, with high accuracy and other evaluation metrics. The decision tree classifier achieved the highest performance, closely followed by the SVM and random forest classifiers.

However, the Naive Bayes classifier had slightly lower performance compared to the other classifiers. The neural network model also performed well but had slightly lower performance compared to the SVM and decision tree classifiers. These results demonstrate the effectiveness of these algorithms in email classification tasks, with the decision tree classifier being the most accurate in this particular case.

VI. Conclusion

In order to assess the accuracy of different popular machine learning models for classifying phishing email text and URLs, we referred to [2]. The dataset was split into a training set and a testing set in an 80:20 ratio. Upon evaluation, it was found that the Decision Tree Classifier (DTC) achieved an accuracy of 97.87% for the text classification task. As the URL component is considered crucial in phishing emails, it was given special attention using the same dataset.. In this case, the Neural Network (NN) model achieved the highest accuracy of 97.99% for URL classification.

In conclusion, based on the evaluation of various machine learning models on both text and URL parts of phishing emails, the Decision Tree Classifier (DTC) exhibited an accuracy of 97.87% for text classification, while the Neural Network (NN) model achieved the highest accuracy of 97.99% for URL classification.

VII. Acknowledgment

It gives us immense pleasure to express my deepest sense of gratitude and sincere thanks to our respected guide Dr. Amit Khaparde Sir for their valuable guidance, encouragement and help for completing this work. Their useful suggestions for this whole work and cooperative behavior are sincerely

acknowledged. We also wish to express our indebtedness to our parents as well as our family members.

VIII. References

[1]https://www.cloudflare.com/learning/access-management/phishing-attack/

[2] S. Salloum, T. Gaber, S. Vadera, and K. Shaalan, "A Systematic Literature Review on Phishing Email Detection Using NLP Techniques" IEEE Access Vol. 10 , 2022

[3] M. Sameen, K. Han, and S. Oun Hwang (PhishHaven—An Efficient Real-Time AI Phishing URLs Detection System), "PhishHaven—An Efficient Real-Time AI Phishing URLs Detection System"

[4] C. N. Gutierrez, T. Kim, R.D. Corte, J. Avery, D. Goldwasser, M. Cinque, and S. Bagchi, "Learning from the Ones that Got Away: Detecting New Forms of Phishing Attacks" in IEEE Transactions On Dependable And Secure Computing, Vol. 15, No. 6, November/December 2018

[5] Z. Dou, I. Khalil, A. Khreishah, A. Al-Fuqaha, and M. Guizani, "Systematization of Knowledge (SoK): A Systematic Review of Software-Based Web Phishing Detection" in IEEE Communications Surveys & Tutorials, Vol. 19, No. 4, Fourth Quarter 2017

[6] J. Wu, Q. Yuan, D. Lin, W. You, W. Chen, C. Chen, Z. Zheng, "Who Are the Phishers? Phishing Scam Detection on Ethereum via Network Embedding" in IEEE Transactions On Systems, Man, And Cybernetics: Systems, Vol. 52, No. 2, February 2022

[7] X. Liu, J. Fu, "SPWalk: Similar Property Oriented Feature Learning for Phishing Detection" IEEE Access Vol. 8 , 2020

[8] C. Pham, L.A.T. Nguyen, N.T. Tran, E.-N. Huh, and C.S. Hong, "Phishing-Aware: A Neuro-Fuzzy Approach for Anti-Phishing on Fog Networks" in IEEE Transactions On Network And Service Management, Vol. 15, No. 3, September 2018

[9] Y. Alsariera, V.A. Ademyemo, A.O. Balogun, and A.K. Alazzawi, "AI Meta-Learners and Extra-Trees Algorithm for the Detection of Phishing Websites" IEEE Access Vol 8, 2020

[10] J. Mao, W. Tian, P. Li, T. Wei, Z. Liang, "Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity" IEEE Access Vol. 5, 2017

[11] A.E. Aassal, S. Baki, A. Das, and R.M. Verma, "An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs" in IEEE Access Vol. 8, 2020

[12] spamassassin.apache.org/old/publiccorpus/