KLE Society's
KLE Technological University

Information Security Course Project Report

On

Detection of Phishing Websites using Machine Learning Techniques

Submitted By

Roll No	Name	USN
501	Halli Harshita	01FE20BCS242
509	Vishal Farande	01FE20BCS251
515	Vivek Kumar	01FE20BCS263
518	Chandan Srivastava B	01FE20BCS274

Faculty In-charge

Mrs Pooja Shettar

School of Computer Science and Engineering

Vidyanagar, Hubballi – 580031, India.

Academic year 2023-24

CONTENTS

INTRODUCTION

The proliferation of online activities has led to an increase in cyber threats, with phishing attacks posing a significant risk to individuals and organizations. Phishing websites, designed to deceive users into divulging sensitive information, continue to evolve in sophistication, making traditional detection methods less effective. This paper explores the application of machine learning techniques for the detection of phishing websites, leveraging the inherent ability of machine learning models to discern patterns and anomalies in large datasets.

The proposed approach involves the extraction of relevant features from website content, URL structures, and other contextual information. A diverse set of machine learning algorithms, including but not limited to decision trees, support vector machines, and neural networks, are employed to train models on labeled datasets comprising both phishing and legitimate websites. The models are fine-tuned using feature selection and hyperparameter optimization to enhance their accuracy, precision, and recall.

To evaluate the effectiveness of the machine learning models, extensive experiments are conducted on real-world datasets, simulating various phishing scenarios. The results demonstrate the ability of the proposed approach to effectively distinguish between phishing and legitimate websites, showcasing its potential for proactive threat mitigation. Furthermore, the paper explores the adaptability and scalability of the machine learning models to evolving phishing tactics and emerging threats.

The research contributes to the ongoing efforts in cybersecurity by providing a robust and dynamic solution for the timely detection of phishing websites. The incorporation of machine learning techniques not only improves the accuracy of detection but also enables the system to learn and adapt to new phishing strategies. The findings of this study have implications for the development of more resilient and adaptive cybersecurity systems, ultimately enhancing the protection of users and organizations against the ever-evolving landscape of cyber threats.

LITERATURE SURVEY

Browsers like Microsoft Edge, Firefox, and Google Chrome utilize List Based methods to detect phishing websites. Whitelisting and blacklisting are two types of List Based approaches. The whitelist contains a list of valid URLs that browsers can access, which means if the URL is in the whitelist, the browser can download the web page. At the same time, the blacklisting database includes phishing or fraudulent URLs that stop browsers from downloading the web pages. The major disadvantage is that a minor modification in the URL is enough to bypass the List Based techniques and to prevent new phishing URLs, these lists must frequently be updated [1](Yang et al., 2021).

Visual Similarity approach assesses the suspect and authentic websites based on various visual characteristics. Because the phishing web page appears to be highly similar to its legitimate page, these tools compare similarities: This approach uses CSS, text layout, source code, the website logo, screenshots of the web page, and other visual elements. Because these techniques compare the suspect web page to previously visited or saved web pages, they cannot detect zero-hour phishing attacks [2](Jain and Gupta, 2018).

Heuristic approach uses features derived from the phishing website. This strategy is based on several attributes that can differentiate a phishing website from a genuine one. These methods gather data from various sources, such as URLs, text content, DNS, digital certificates, and website traffic. The feature set, training samples, and classification algorithms all influence the success of this method. One of the advantages of this technique is that it can detect Zero-hour phishing attacks [2](Jain and Gupta, 2018).

Nowadays, Machine Learning is a prevalent approach for detecting phishing websites [3](Sindhu et al., 2020). Common attributes such as URL information, website structure, and JavaScript features are collected to represent phishing URLs and related websites. Then, based on those features, phishing data sets are obtained. After that, Machine Learning classifiers are trained to detect the phishing website based on those features [4](Zhu et al., 2020). This technique works very well with Big Data sets (having high Velocity, Variety, Volume, Value, and Veracity). Machine Learning-based classifiers achieved more than 99% accuracy, which proved to be the most effective method [5](Alkawaz et al., 2021).

Deep Learning: According to recent developments in Deep Learning methods, Deep Neural Network should perform better than conventional Machine Learning techniques in detecting phishing websites. Some well-known Deep Learning algorithms used for phishing detection are the Deep Neural Network, Recurrent Neural Network, Feed-Forward Deep Neural Network, limited Boltzmann machine, Convolutional Neural Network, deep belief network, and deep auto-encoder [6](Basit et al., 2020).

PROBLEM STATEMENT

Developing an effective machine learning and deep learning-based system for the detection and classification of phishing websites, with a focus on improving accuracy, speed, and robustness, to protect users from online threats and cyber attacks.

IMPLEMENTATION

A phishing website is a common social engineering method that mimics trustful uniform resource locators (URLs) and webpages. The objective of this notebook is to collect data & extract the selected features from the URLs.

- Below mentioned are the steps involved in the completion of this project:
- Collect dataset containing phishing and legitimate websites from the open source platforms.
- Write a code to extract the required features from the URL database.
- Analyze and preprocess the dataset by using EDA techniques.
- Divide the dataset into training and testing sets.
- Run selected machine learning and deep neural network algorithms like SVM, Random Forest, Autoencoder on the dataset.
- Write a code for displaying the evaluation result considering accuracy metrics.
- Compare the obtained results for trained models and specify which is better.

The phishing URLs are collected from various sites. The csv file of phishing URLs is obtained by using wget command. After downloading the dataset, it is loaded into a DataFrame. So, the data has thousands of phishing URLs. Without getting into the risk of data imbalance, considering a margin value of 10,000 phishing URLs & 5000 legitimate URLs. Thereby, picking up 5000 samples from the above dataframe randomly.

Feature Extraction:

In this step, features are extracted from the URLs dataset. The extracted features are categorized into

- ❖ Address Bar based Features
- ❖ Domain based Features
- ❖ HTML & Javascript based Features

Address Bar Based Features:

Many features can be extracted that can be considered as address bar base features. Out of them, below mentioned were considered for this project.

- Domain of URL

  Here, we are just extracting the domain present in the URL. This feature doesn't have much significance in the training. May even be dropped while training the model.

- IP Address in the URL

  Checks for the presence of an IP address in the URL. URLs may have IP address instead of domain name. If an IP address is used as an alternative of the domain name in the URL, we can be sure that someone is trying to steal personal information with this URL.

  If the domain part of the URL has an IP address, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

- "@" Symbol in URL

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

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

- Length of URL

  Computes the length of the URL. Phishers can use long URLs to hide the doubtful part in the address bar. In this project, if the length of the URL is greater than or equal 54 characters then the URL is classified as phishing otherwise legitimate.

  If the length of URL >= 54 , the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

- Depth of URL

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

  The value of the feature is numeric based on the URL.

- Redirection "//" in URL

  Check the presence of "//" in the URL. The existence of “//” within the URL path means that the user will be redirected to another website. The location of the “//” in the URL is computed. We find that if the URL starts with “HTTP”, that means the “//” should appear in the sixth position. However, if the URL employs “HTTPS” then the “//” should appear in seventh position.

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

- "http/https" in Domain name

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

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

- Using URL Shortening Services “TinyURL”

  URL shortening is a method on the “World Wide Web” in which a URL may be made considerably smaller in length and still lead to the required webpage. This is accomplished by means of an “HTTP Redirect” on a domain name that is short, which links to the webpage that has a long URL.

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

- Prefix or Suffix "-" in Domain

  Checking the presence of '-' in the domain part of the URL. The dash symbol is rarely used in legitimate URLs. Phishers tend to add prefixes or suffixes separated by (-) to the domain name so that users feel that they are dealing with a legitimate website.

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

Domain Based Features:

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

- DNS Record

  For phishing websites, either the claimed identity is not recognized by the WHOIS database or no records found for the hostname. If the DNS record is empty or not found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

- Web Traffic

  This feature measures the popularity of the website by determining the number of visitors and the number of pages they visit. However, since phishing websites live for a short period of time, they may not be recognized by the Alexa database (Alexa the Web Information Company., 1996). By reviewing our dataset, we find that in the worst scenarios, legitimate websites ranked among the top 100,000. Furthermore, if the domain has no traffic or is not recognized by the Alexa database, it is classified as “Phishing”.

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

- Age of Domain

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

  If age of domain > 12 months, the value of this feature is 1 (phishing) else 0 (legitimate).
- End Period of Domain

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

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

HTML and JavaScript based Features:

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

- IFrame Redirection

  IFrame is an HTML tag used to display an additional webpage into one that is currently shown. Phishers can make use of the “iframe” tag and make it invisible i.e. without frame borders. In this regard, phishers make use of the “frameBorder” attribute which causes the browser to render a visual delineation.

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

- Status Bar Customization

  Phishers may use JavaScript to show a fake URL in the status bar to users. To extract this feature, we must dig-out the webpage source code, particularly the “onMouseOver” event, and check if it makes any changes on the status bar.

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

- Disabling Right Click

  Phishers use JavaScript to disable the right-click function, so that users cannot view and save the webpage source code. This feature is treated exactly as “Using onMouseOver to hide the Link”. Nonetheless, for this feature, we will search for the event “event.button==2” in the webpage source code and check if the right click is disabled.

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

- Website Forwarding

  The fine line that distinguishes phishing websites from legitimate ones is how many times a website has been redirected. In our dataset, we find that legitimate websites have been redirected one time max. On the other hand, phishing websites containing this feature have been redirected at least 4 times.

Computing URL Features

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

```
#Function to extract features
def featureExtraction(url,label):

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

```
  features.append(redirection(url))
  features.append(httpDomain(url))
  features.append(tinyURL(url))
  features.append(prefixSuffix(url))

  #Domain based features (4)
  dns = 0
  try:
    domain_name = whois.whois(urlparse(url).netloc)
  except:
    dns = 1

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

  # HTML & Javascript based features (4)
  try:
    response = requests.get(url)
  except:
    response = ""
  features.append(iframe(response))
  features.append(mouseOver(response))
  features.append(rightClick(response))
  features.append(forwarding(response))
  features.append(label)

  return features
```

A phishing website is a common social engineering method that mimics trustful uniform resource locators (URLs) and webpages. The objective of this project is to train machine learning models and deep neural nets on the dataset created to predict phishing websites. Both phishing and benign URLs of websites are gathered to form a dataset and from them required URL and website content-based features are extracted. The performance level of each model is measured and compared.

This project is worked on by Google Collaboratory.
The required packages for this notebook are imported when needed.

The features are extracted and stored in the csv file. The working of this can be seen in the 'Phishing Website Detection_Feature Extraction.ipynb' file. The resulting csv file is uploaded to this notebook and stored in the dataframe.

Visualizing the data

Few plots and graphs are displayed below to find how the data is distributed and how features are related to each other shown in figure 1 and figure 2.

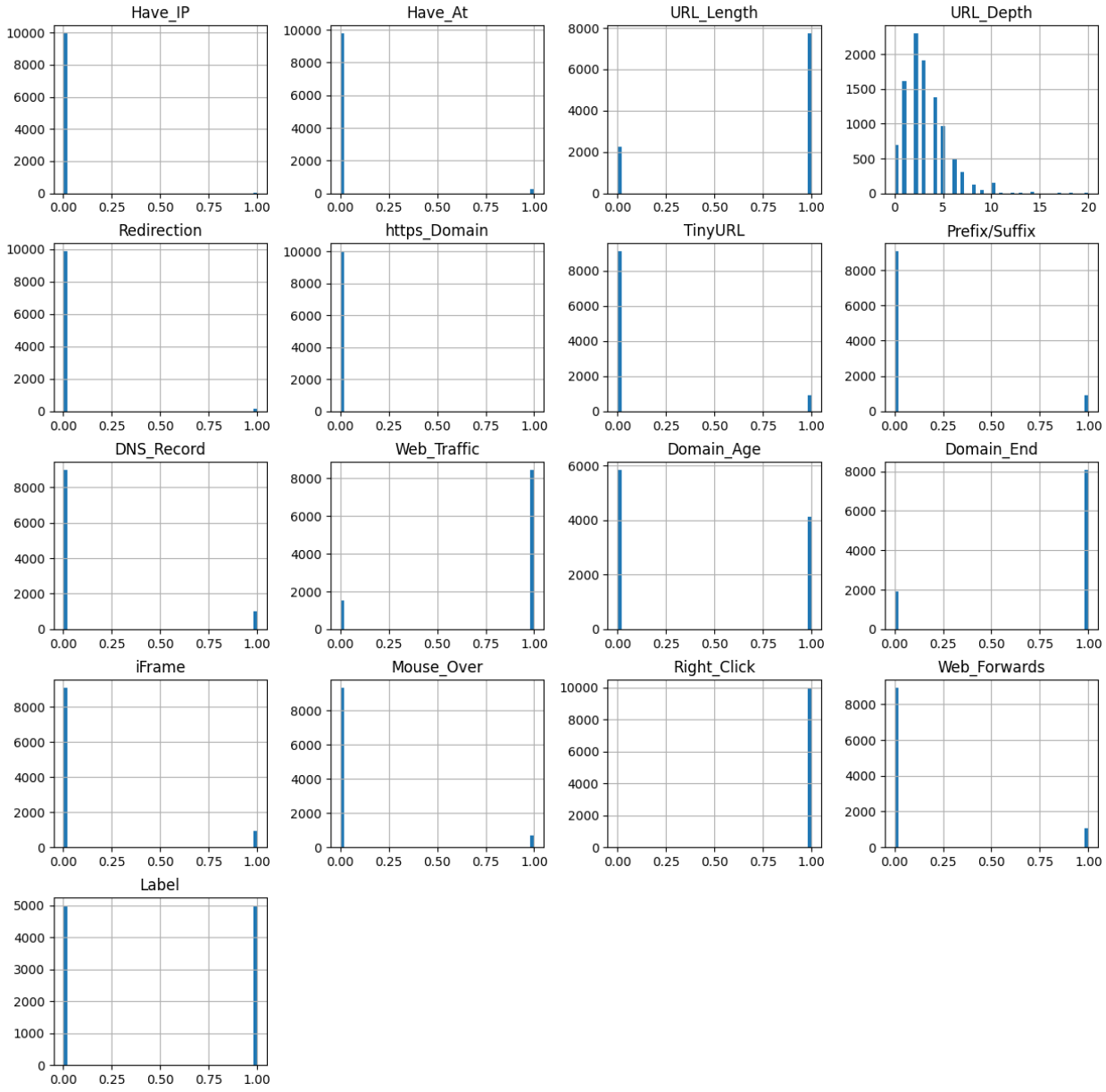

Figure 1: Graphs showing the feature relevance

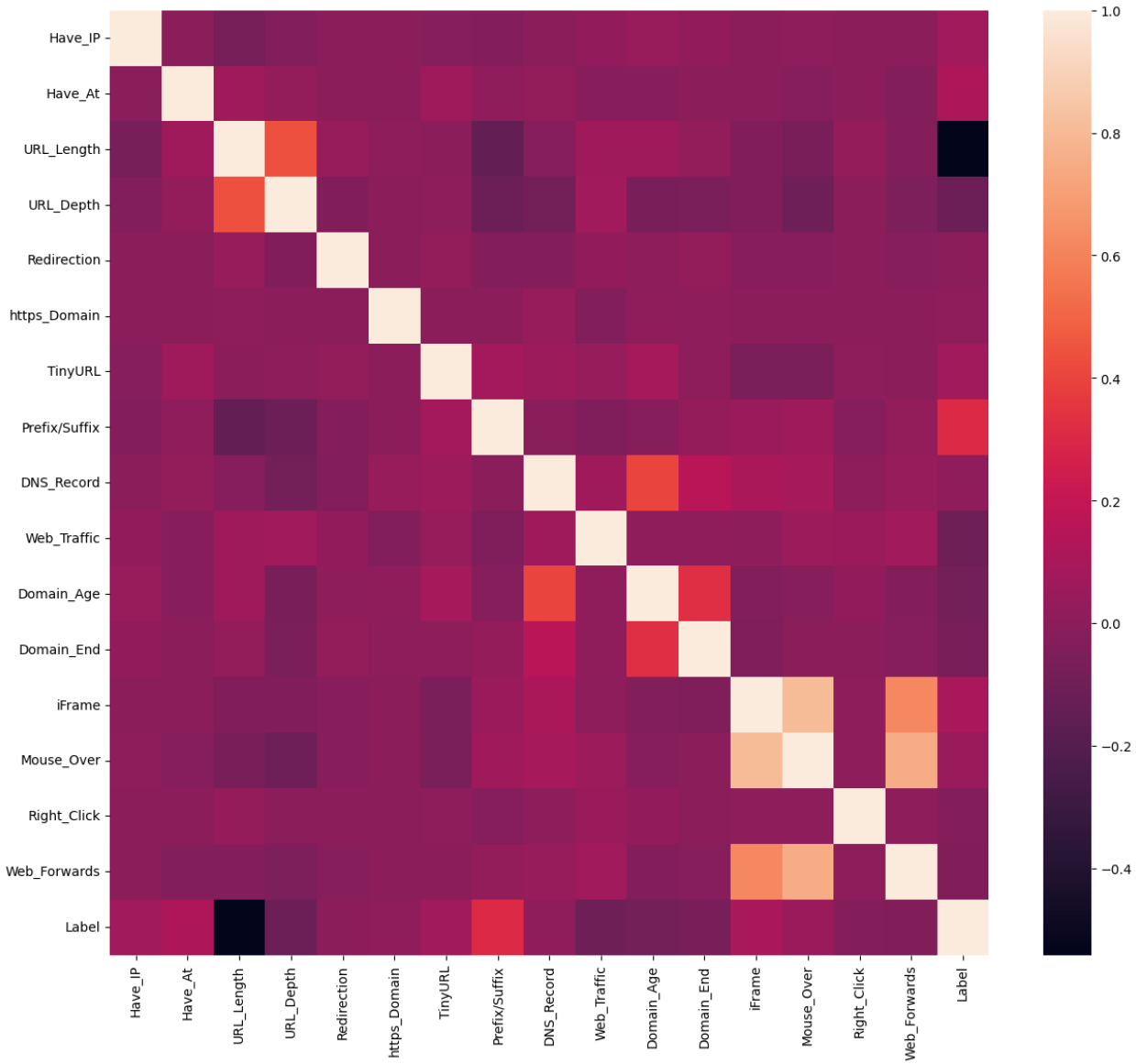

Figure 2: Features Correlation Matrix

Data Preprocessing & EDA

Here, we clean the data by applying data preprocessing techniques and transform the data to use it in the models.

The obtained result shows that most of the data is made of 0's & 1's except 'Domain' & 'URL_Depth' columns. The Domain column does not have any significance to the machine learning model training. So dropping the 'Domain' column from the dataset. This leaves us with 16 features & a target column. The 'URL_Depth' maximum value is 20. According to my understanding, there is no necessity to change this column.

In the feature extraction file, the extracted features of legitimate & phishing url datasets are just concatenated without any shuffling. This resulted in top 5000 rows of legitimate url data & bottom 5000 of phishing url data.

To even out the distribution while splitting the data into training & testing sets, we need to shuffle it. This even evades the case of overfitting while model training.

Splitting the Data

```
# Splitting the dataset into train and test sets: 80-20 split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
random_state = 12)
X_train.shape, X_test.shape
```

Machine Learning Models & Training

From the dataset above, it is clear that this is a supervised machine learning task. There are two major types of supervised machine learning problems, called classification and regression.

This data set comes under classification problems, as the input URL is classified as phishing (1) or legitimate (0). The supervised machine learning models (classification) considered to train the dataset in this notebook are:

- Decision Tree
- Random Forest
- Multilayer Perceptrons
- XGBoost
- Autoencoder Neural Network
- Support Vector Machines

Decision Tree Classifier

Decision trees are widely used models for classification and regression tasks. Essentially, they learn a hierarchy of if/else questions, leading to a decision. Learning a decision tree means learning the sequence of if/else questions that gets us to the true answer most quickly.

In the machine learning setting, these questions are called tests (not to be confused with the test set, which is the data we use to test to see how generalizable our model is). To build a tree, the algorithm searches over all possible tests and finds the one that is most informative about the target variable as shown in Figure 3.

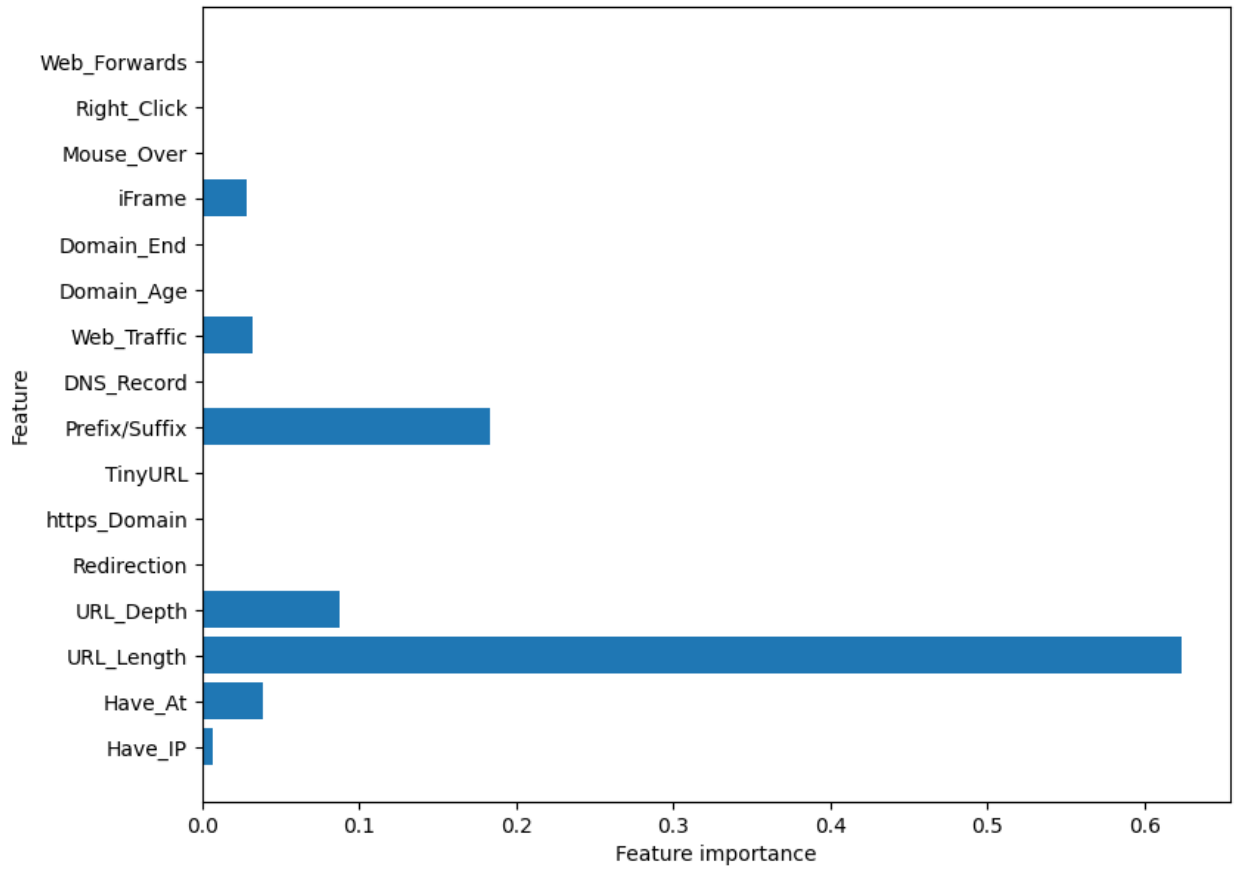

Figure 3: Feature importance for Decision Tree Classifier

Random Forest Classifier

Random forests for regression and classification are currently among the most widely used machine learning methods.A random forest is essentially a collection of decision trees, where each tree is slightly different from the others. The idea behind random forests is that each tree might do a relatively good job of predicting, but will likely overfit on part of the data.

If we build many trees, all of which work well and overfit in different ways, we can reduce the amount of overfitting by averaging their results. To build a random forest model, you need to decide on the number of trees to build shown in figure 4 (the n_estimators parameter of RandomForestRegressor or RandomForestClassifier). They are very powerful, often work well without heavy tuning of the parameters, and don't require scaling of the data.

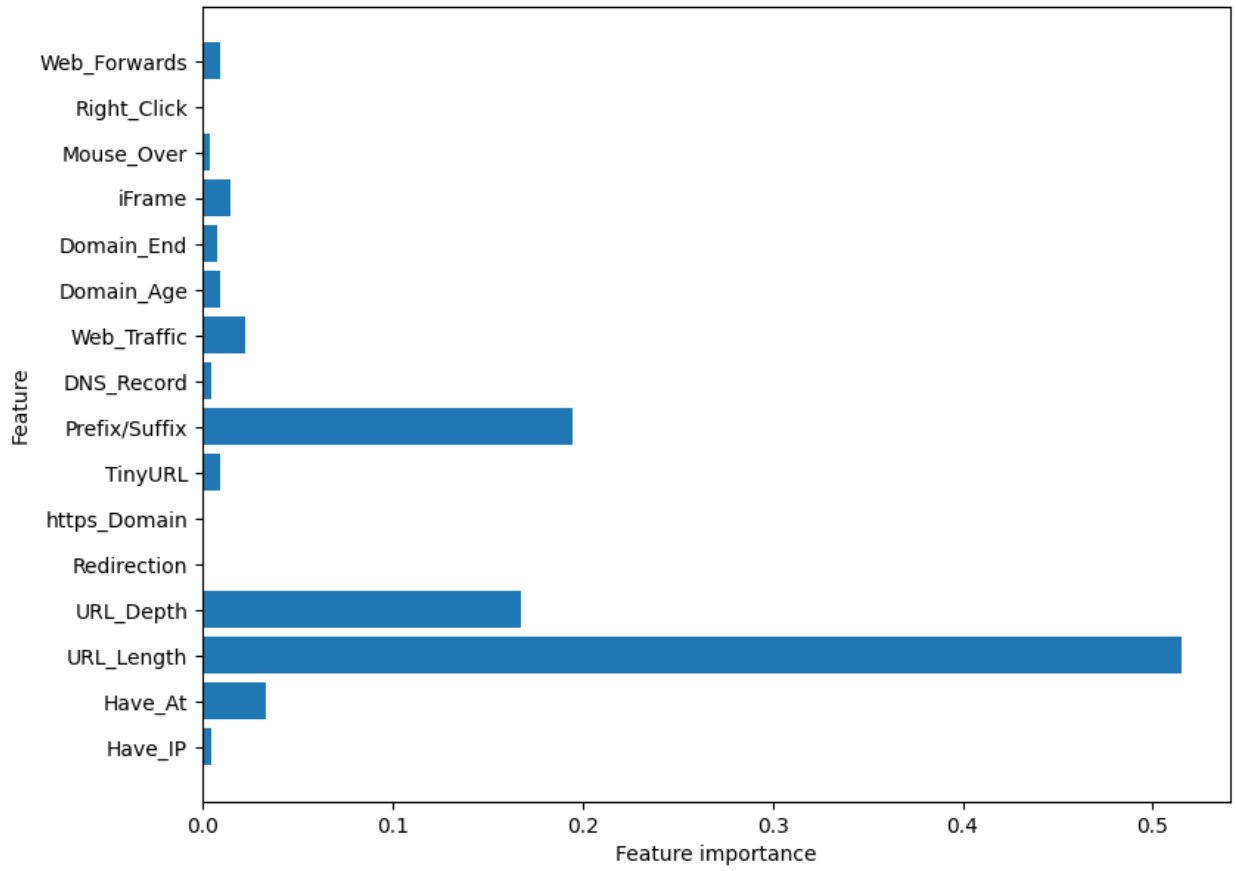

Figure 4: Feature importance for Random Forest Classifier

Multilayer Perceptrons (MLPs): Deep Learning

Multilayer perceptrons (MLPs) are also known as (vanilla) feed-forward neural networks, or sometimes just neural networks. Multilayer perceptrons can be applied for both classification and regression problems.

MLPs can be viewed as generalizations of linear models that perform multiple stages of processing to come to a decision.

XGBoost Classifier

XGBoost is one of the most popular machine learning algorithms these days. XGBoost stands for eXtreme Gradient Boosting. Regardless of the type of prediction task at hand; regression or classification. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

Autoencoder Neural Network

An auto encoder is a neural network that has the same number of input neurons as it does outputs. The hidden layers of the neural network will have fewer neurons than the input/output neurons. Because there are fewer neurons, the auto-encoder must learn to encode the input to the fewer hidden neurons. The predictors (x) and output (y) are exactly the same in an auto encoder.

Support Vector Machines

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

RESULTS & DISCUSSIONS

Comparison of Models

To compare the model's performance, a data frame is created. The columns of this dataframe are the lists created to store the results of the model.

```
#creating dataframe
results = pd.DataFrame({ 'ML Model': ML_Model,'Train Accuracy': acc_train,
    'Test Accuracy': acc_test})
results
```

The models are evaluated and considered metric is **Accuracy**.

Below Figure shows the training and test dataset accuracy by the respective models:

	ML Model	Train Accuracy	Test Accuracy
0	Decision Tree	0.817	0.800
1	Random Forest	0.820	0.800
2	Multilayer Perceptrons	0.863	0.854
3	XGBoost	0.869	0.850
4	AutoEncoder	0.003	0.003
5	SVM	0.804	0.793

CONCLUSION & FUTURE SCOPE

In conclusion, the use of machine learning techniques for the detection of phishing websites has shown promising results in enhancing cybersecurity efforts. The deployment of sophisticated algorithms and models has significantly improved the accuracy and efficiency of identifying fraudulent websites that aim to deceive users. One key advantage lies in the ability of machine learning systems to adapt and evolve, learning from new phishing patterns and staying ahead of constantly evolving threats.

Additionally, the integration of feature engineering and data preprocessing techniques has played a crucial role in enhancing the overall performance of these models. By extracting meaningful features from various aspects of web content, including URLs, text, and images, machine learning algorithms can better discern the subtle nuances that distinguish phishing sites from legitimate ones. This has led to a more robust and reliable detection system, reducing false positives and negatives.

The collaborative efforts between researchers, cybersecurity experts, and data scientists have contributed to the development of more advanced and accurate machine learning models. The continuous refinement of these models through feedback loops and real-time updates ensures that they remain effective in the face of emerging phishing tactics. This dynamic approach is crucial in the fast-paced landscape of cybersecurity, where threat actors are constantly devising new strategies to bypass traditional security measures.

However, it is important to acknowledge the challenges that still exist in the field of phishing website detection. Adversarial attacks and the constant evolution of phishing techniques pose ongoing challenges for machine learning models. Researchers must remain vigilant in addressing these challenges and devising innovative solutions to stay one step ahead of cybercriminals.

In conclusion, while machine learning has significantly bolstered the capabilities of phishing website detection, it should be seen as part of a comprehensive cybersecurity strategy. Combining machine learning with other security measures, such as user education, secure browsing practices, and regular security updates, will create a more holistic defense against phishing threats. As technology continues to advance, the synergy between human expertise and machine learning capabilities will be crucial in maintaining a resilient cybersecurity infrastructure.

The future of detecting phishing websites using machine learning is a dynamic landscape, encompassing advancements in algorithms, techniques, collaboration, and adaptability. As the field continues to evolve, the integration of cutting-edge technologies and the collaborative efforts of researchers and

cybersecurity professionals will play a pivotal role in shaping a more resilient and effective defense against phishing threats.

- Working on this project is very knowledgeable and worth the effort.
- Through this project, one can know a lot about the phishing websites and how they are differentiated from legitimate ones.
- This project can be taken further by creating browser extensions or developing a GUI.
- These should classify the inputted URL to legitimate or phishing with the use of the saved model.

REFERENCES

1. L. Yang, J. Zhang, X. Wang, Z. Li, Z. Li, Y. He; *An improved ELM-based and data preprocessing integrated approach for phishing detection considering comprehensive features;* Expert Syst. Appl., 165 (July 2020).
2. A.K. Jain, B.B. Gupta; *Two-level authentication approach to protect from phishing attacks in real time;* J. Ambient Intell. Hum. Comput., 9 (6).
3. S. Sindhu, S.P. Patil, A. Sreevalsan, F. Rahman, A.N. Saritha; *Phishing detection using random forest, SVM and neural network with backpropagation;* Proceedings of the International Conference on Smart Technologies in Computing, Electrical and Electronics, ICSTCEE 2020 (2020), pp. 391-394.
4. E. Zhu, Y. Ju, Z. Chen, F. Liu, X. Fang; *DTOF-ANN: an artificial neural network phishing detection model based on decision tree and optimal features;* Appl. Soft Comput. J., 95 (2020), Article 106505.
5. M.H. Alkawaz, S.J. Steven, A.I. Hajamydeen, R. Ramli; *A comprehensive survey on identification and analysis of phishing websites based on machine learning methods;* ISCAIE 2021 - IEEE 11th Symposium on Computer Applications and Industrial Electronics (2021), pp. 82-87.
6. A. Basit, M. Zafar, X. Liu, A.R. Javed, Z. Jalil, K. Kifayat; *A comprehensive survey of AI-enabled phishing attacks detection techniques;* Telecommun. Syst., 76 (1) (2020), pp. 139-154.