Design and prototype innovation app or software-based solutions that can detect the use, type, and scale of dark patterns one-commerce platforms

Project Submitted in Partial Fulfilment of the Requirements for the Degree of Bachelor of Technology in the field of Computer Science and Engineering

BY
Auditee Saha Chowdhury (123211003035)
Mallika Aich (123211003073)
Priti Sarkar (123211003100)
Megha Saha (123211003080)

Under the supervision of

DEBASREE MITRA

Department of Computer Science and Engineering
JIS College of Engineering

Block-A, Phase-III, Kalyani, Nadia, Pin-741235
West Bengal

ACKNOWLEDGEMENT

The analysis of the project work wishes to express our gratitude to Prof. Ira Nath for allowing the degree attitude and providing effective guidance in development of this project work. Her conscription of the topic and all the helpful hints, she provided, contributed greatly to successful development of this work, without being pedagogic and overbearing influence.

We also express our sincere gratitude to **Dr. Bikramjit Sarkar**, Head of the Department of Computer Science and Engineering of JIS College of Engineering, and all the respected faculty members of the Department of CSE for giving the scope of successfully carrying out the project work.

Finally, we take this opportunity to thank Prof. **(Dr.) Partha Sarkar**, Principal of JIS College of Engineering for giving us the scope of carrying out the project work.

CONTENTS

List of Figures & Tables :

Abstract:

Phishing is an internet scam in which an attacker sends out fake messages that look to come from a trusted source. Phishing attack is a simplest way to obtain sensitive information from innocent users. Aim of the phishers is to acquire critical information like username, password and bank account details. A URL or file will be included in the mail, which when clicked will steal personal information or infect a computer with a virus. Traditionally, phishing attempts were carried out through wide-scale spam campaigns that targeted broad groups of people indiscriminately. The goal was to get as many people to click on a link or open an infected file as possible. There are various approaches to detect this type of attack. One of the approaches is machine learning. The URLs received by the user will be given input to the machine learning model then the algorithm will process the input and display the output whether it is phishing or legitimate. Various ML algorithms like SVM, Neural Networks, Random Forest, Decision Tree, XG boost, etc. can be used to classify these URLs. The proposed approach deals with the XGBoost, The proposed approach effectively classified the Phishing and Legitimate URLs with an accuracy of 86.0% for the XGBoost.

1.Introduction:

Nowadays Phishing has become a main area of concern for security researchers because it is not difficult to create a fake website that looks so close to legitimate website. Experts can identify fake websites but not all the users can identify the fake website and such users become the victim of phishing attack. Main aim of the attacker is to steal banks account credentials. In United States businesses, there is a loss of US$2billion per year because their clients become victim to phishing . In 3rd Microsoft Computing Safer Index Report released in February 2014, it was estimated that the annual worldwide impact of phishing could be as high as $5 billion . Phishing attacks are becoming successful because lack of user awareness. Since phishing attack exploits the weaknesses found in users, it is very difficult to mitigate them but it is very important to enhance phishing detection techniques. The general method to detect phishing websites by updating blacklisted URLs, Internet Protocol
(IP) to the antivirus database which is also known as “blacklist" method. To evade blacklists attackers uses creative techniques to fool users by modifying the URL to appear legitimate via obfuscation and many other simple techniques including: fastflux, in which proxies are automatically generated to host the web-page; algorithmic generation of new URLs; etc. The major drawback of this method is that it cannot detect zero-hour phishing attacks. Heuristic-based detection includes characteristics that are found to exist in phishing attacks in reality and can detect

zero-hour phishing attacks, but the characteristics are not guaranteed to always exist in such attacks and false positive rate in detection is very high . To overcome the drawbacks of blacklisting and heuristics-based method, many security researchers now focused on machine learning techniques. Machine learning technology consists of many algorithms that require past data to make a decision or prediction on future data. Using this technique, the algorithm will analyze various blacklisted and legitimate URLs and their features to accurately detect phishing websites including zero-hour phishing websites.

2.Literature Survey:

A very effective detection of phishing website model which is focused on optimal feature selection technique and also based on neural network (OFS-NN) is proposed. In this proposed model, an index called feature validity value(FVV) has been generated to check the effects of all those features on the detection of such websites. Now, based on this newly generated index, an algorithm is developed to find from the phishing websites,the optimal features. This selected algorithm will be able to overcome the problem of over-fitting of the neural network to a great extend. These optimal features are then used to build an optimal classifier that detects phishing URLs by training the neural network. A theory called Fuzzy Rough Set(FRS) was devised to a tool that finds the most appropriate features from a few standardised dataset. These features are then sent to a few classifiers for detection of phishing. To investigate the feature selection for FRS in building a generalized detection of phishing, the models by a different dataset of 14,000 website samples are trained. Feature engineering plays a vital role in finding solutions for detection of phishing websites, although the accuracy of the model greatly will be based on knowledge of the features. though the features taken from all these various dimensions are understandable, the limitation lies in the time taken to collect these features. To fix this drawback, the authors have proposed a multidimensional phishing detection feature approach that concentrates on a rapid detection technique by making use of deep learning (MFPD) To detect phishing occurrence accurately, a three phase detection called Web Crawler based Phishing Attack Detector (WC-PAD) has been proposed. This takes the web's content, traffic and URL as input features. Now considering these features, classification is done.

.

A model as a solution was the focus in a study that uses Random Forest classifier for detection of phishing websites by URL method. An approach that combines to form an online tool, the collection, validation and detection of phishing websites. This online tool monitors in real-time the blacklist of PhishTank, validates and

detects phishing website. A framework was developed, known as ”Fresh-Phish” , that generates for phishing websites, present machine learning data. By using 30 various features of website which can be queried using Python, a very large dataset is built and the various ML classifiers are analyzed against this generated dataset to find out which has highest accuracy. This model analyzes both the accuracy as well as the time taken by the model to train. A determined bond was built between the content-based heuristics and the authenticity of the website by evaluating both the phishing and legitimate websites' training set. A framework called PhishingDetective is presented which detects the websites as phishing based on existing heuristics as well as new heuristics A productive way using C4.5 decision tree classifier as well as certain features of the URL was proposed to detect websites that are phishing. There are many schemes for detection of phishing websites, among which the visual similarity scheme is collecting glances. The screenshot of the website is taken and stored in a database. It checks if the input screenshot of the website is same as the one stored in the database. If yes, then that website is predicted as phishing. But, if there are several similar websites, which ever is the first website that is given as input is taken as legitimate. Hence, it cannot predict correctly the authentic website and therefore recognising the goal website becomes tedious. This detection method is proposed with target website finder by making use of images and CSS.

- **”Phishing Sites Detection based on C4.5 Decision Tree Algorithm”** :- The approach proposed makes use of features that were extracted from the URL to make decision about the legitimacy of the URL given as input. To generate the rules, the c4.5 algorithm was used. The rules produced are utilized to order the submitted URL as genuine or phishing with better productivity. Overall accuracy is less as the paper considers limited URL features.
- **”Visual Similarity-based Phishing Detection Scheme using Image and CSS with Target Website Finder”:-**
  The main focus is on the fact that authentic websites are usually linked by many websites and those websites are regarded as legitimate, the screenshots and CSS of which are stored in a database. Because CSS is a file that characterizes the site's visual substance, assaulter regularly take real CSS to imitate the real site. Hence, by finding the site which counterfeits appearance or CSS of real site, we identify phishing site and its objective at the same time. 2017 The websites that are linked at least by one website are also recorded in the white list assuming it to be legitimate.

From the above, ML methods plays a vital role in many applications of cybersecurity and shall remain an encouraging path that captivates more such investigations. When coming to the reality, there are several barriers that are limitations during implementations. As discussed, there are many approaches earlier proposed for

detecting phishing website attack and they also have their own limitations. Therefore, the aim of the project is detection of phishing website attack using a novel Machine learning technique.

3. Proposed Method:

The problem is derived after making a thorough observation and study about the method of classification of phishing websites that makes use of machine learning techniques. +

Below mentioned are the steps involved in the completion of this project:

- Collect datasets containing phishing and legitimate websites from 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, and XGBoost 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.

4. Methodology:

Data Collection

The set of phishing URLs are collected from an open-source service. This service provides a set of phishing URLs in multiple formats like CSV, JSON, etc. From this dataset, 5000 random phishing URLs are collected to train the ML models.

This dataset has a collection of benign, spam, phishing, malware & defacement URLs. Out of all these types, the benign url dataset is considered for this project. From this dataset, 5000 random legitimate URLs are collected to train the ML models.

Feature Extraction

The below-mentioned category of features are extracted from the URL data:

1. AddressBarbasedFeatures

   In this category 9 features are extracted.

2. Domain-basedfeatures

   In this category, 4 features are extracted.

3. HTML&JavascriptbasedFeatures

   In this category, 4 features are extracted.

1) Presence of IP address in URL: If IP address present in URL then the feature is set to 1 else set to 0. Most of the benign sites do not use IP address as an URL to download a webpage. Use of IP address in URL indicates that attacker is trying to steal sensitive information.

2) Presence of @ symbol in URL: If @ symbol present in URL then the feature is set to 1 else set to 0. Phishers add special symbol @ in the URL leads the browser to ignore everything preceding the "@" symbol and the real address often follows the "@" symbol .

3) Number of dots in Hostname: Phishing URLs have many dots in URL. For example http://shop.fun.amazon.phishing.com, in this URL phishing.com is an actual domain name, whereas use of "amazon" word is to trick users to click on it. Average number of dots in benign URLs is 3. If the number of dots in URLs is more than 3 then the feature is set to 1 else to 0.

4) Prefix or Suffix separated by (-) to the domain: If domain name separated by dash (-) symbol then feature is set to 1 else to 0. The dash symbol is rarely used in legitimate URLs. Phishers add dash symbol (-) to the domain name so that users feel that they are dealing with a legitimate webpage. For example Actual site is http://www.onlineamazon.com but phisher can create another fake website like http://www.online-amazon.com to confuse the innocent users. **5) URL redirection**: If "//" present in URL path then feature is set to 1 else to 0. The existence of "//" within the URL path means that the user will be redirected to another website.

6) **HTTPS token in URL**: If HTTPS token present in URL then the feature is set to 1 else to 0. Phishers may add the "HTTPS" token to the domain part of a URL in order to trick users. For example, http://https-wwwpaypal-it-mpphome.soft-hair.com.

7) **Information submission to Email:** Phisher might use "mail()" or "mailto:" functions to redirect the user's information to his personal email. If such functions are present in the URL then feature is set to 1 else to 0.

8) **URL Shortening Services "TinyURL":** TinyURL service allows phisher to hide long phishing URL by making it short. The goal is to redirect user to phishing websites. If the URL is crafted using shortening services (like bit.ly) then feature is set to 1 else 0

9) **Length of Host name:** Average length of the benign URLs is found to be a 25, If URL's length is greater than 25 then the feature is set to 1 else to 0

10) **Presence of sensitive words in URL:** Phishing sites use sensitive words in its URL so that users feel that they are dealing with a legitimate webpage. Below are the words that found in many phishing URLs :- 'confirm', 'account', 'banking', 'secure', 'ebyisapi', 'webscr', 'signin', 'mail', 'install', 'toolbar', 'backup', 'paypal', 'password', 'username', etc;

11) **Number of slash in URL:** The number of slashes in benign URLs is found to be a 5; if number of slashes in URL is greater than 5 then the feature is set to 1 else to 0.

12) **Presence of Unicode in URL:** Phishers can make a use of Unicode characters in URL to trick users to click on it. For example the domain "xn-80ak6aa92e.com" is equivalent to "apple.com". Visible URL to user is "apple.com" but after clicking on this URL, user will visit to "xn-80ak6aa92e.com" which is a phishing site.

13) **Age of SSL Certificate:** The existence of HTTPS is very important in giving the impression of website legitimacy . But minimum age of the SSL certificate of benign website is between 1 year to 2 year.

14) **URL of Anchor**: We have extracted this feature by crawling the source code oh the URL. URL of the anchor is defined by tag. If the tag has a maximum number of hyperlinks which are from the other domain then the feature is set to 1 else to 0.

15) **IFRAME**: We have extracted this feature by crawling the source code of the URL. This tag is used to add another web page into existing main webpage. Phishers can make use of the "iframe" tag and make it invisible i.e. without frame borders . Since border of inserted webpage is invisible, user seems that the inserted web page is also the part of the main web page and can enter sensitive information.

16) **Website Rank**: We extracted the rank of websites and compare it with the first One hundred thousand websites of database. If rank of the website is greater than 10,0000 then feature is set to 1 else to 0.

MACHINE LEARNING ALGORITHM:

Machine learning classification model Decision Tree, Random forest, Multilayer Perceptrons, XGBoost, AutoEncoder, LIGHTGBM, GBM and has been selected to detect phishing websites.

1)Decision Tree Algorithm:

One of the most widely used algorithm in machine learning technology. The decision tree algorithm is easy to understand and also easy to implement. The decision tree begins its work by choosing best splitter from the available attributes for classification which is considered as a root of the tree. Algorithm continues to build a tree until it finds the leaf node. Decision tree creates a training model which is used to predict the target value or class in tree representation each internal node of the tree belongs to an attribute and each leaf node of the tree belongs to class label. In the decision tree algorithm, gini index and information gain methods are used to calculate these nodes.

2) Random Forest Algorithm:

Random forest algorithm is one of the most powerful algorithms in machine learning technology and it is based on concept of decision tree algorithm. Random forest algorithm creates the forest with number of decision trees. High number of tree gives high detection accuracy. Creation of trees are based on bootstrap method. In bootstrap method features and samples of dataset are randomly selected with replacement to construct single tree. Among randomly selected features, random forest algorithm will choose best splitter for the classification and like decision tree algorithm; Random forest algorithm also uses gini index and information gain methods to find the best splitter. This process will get continue until random forest creates n number of trees. Each tree in forest predicts the target value and then algorithm will calculate the votes for each predicted target. Finally, random forest algorithm considers high voted predicted target as a final prediction.

3) Multilayer Perceptrons Algorithm :

A multilayer perceptron (MLP) is a feed-forward artificial neural network that generates a set of outputs from a set of inputs. An MLP is characterized by several layers of input nodes connected as a directed graph between the input nodes connected as a directed graph between the input and output layers. MLP uses backpropagation for training the network. MLP is a deep learning method.

4) XGBoost Algorithm :

XGBoost is an optimized distributed gradient boosting library designed for efficient and scalable training of machine learning models. It is an ensemble learning method that combines the predictions of multiple weak models to produce a stronger prediction. XGBoost stands for “Extreme Gradient Boosting” and it has become one

of the most popular and widely used machine learning algorithms due to its ability to handle large datasets and its ability to achieve state-of-the-art performance in many machine learning tasks such as classification and regression.
One of the key features of XGBoost is its efficient handling of missing values, which allows it to handle real-world data with missing values without requiring significant pre-processing. Additionally, XGBoost has built-in support for parallel processing, making it possible to train models on large datasets in a reasonable amount of time.

XGBoost can be used in a variety of applications, including Kaggle competitions, recommendation systems, and click-through rate prediction, among others. It is also highly customizable and allows for fine-tuning of various model parameters to optimize performance. **5) AutoEncoder Algorithm :**

An **autoencoder** is a type of artificial neural network used to learn efficient codings of unlabeled data (unsupervised learning). An autoencoder learns two functions: an encoding function that transforms the input data, and a decoding function that recreates the input data from the encoded representation. The autoencoder learns an efficient representation (encoding) for a set of data, typically for dimensionality reduction.

Variants exist, aiming to force the learned representations to assume useful properties. Examples are regularized autoencoders
(*Sparse*, *Denoising* and *Contractive*), which are effective in learning representations for subsequent classification tasks, and *Variational* autoencoders, with applications as generative models. Autoencoders are applied to many problems, including facial recognition, feature detection, anomaly detection and acquiring the meaning of words. Autoencoders are also generative models which can randomly generate new data that is similar to the input data (training data).

6) Multilayer Perceptrons Algorithm :

Multi-Layer perceptron defines the most complex architecture of artificial neural networks. It is substantially formed from multiple layers of the perceptron. TensorFlow is a very popular deep learning framework released by, and this notebook will guide to build a neural network with this library. If we want to understand what is a Multi-layer perceptron, we have to develop a multi-layer perceptron from scratch using Numpy.

6) LIGHTGBM Algorithm :

LightGBM is a gradient-boosting framework based on decision trees to increase the efficiency of the model and reduces memory usage.
It uses two novel techniques:

- Gradient-based One Side Sampling(GOSS)

- Exclusive Feature Bundling (EFB)

These techniques fulfill the limitations of the histogram-based algorithm that is primarily used in all GBDT (Gradient Boosting Decision Tree) frameworks. The two techniques of GOSS and EFB described below form the characteristics of the LightGBM Algorithm. They comprise together to make the model work efficiently and provide it a cutting edge over other GBDT frameworks.

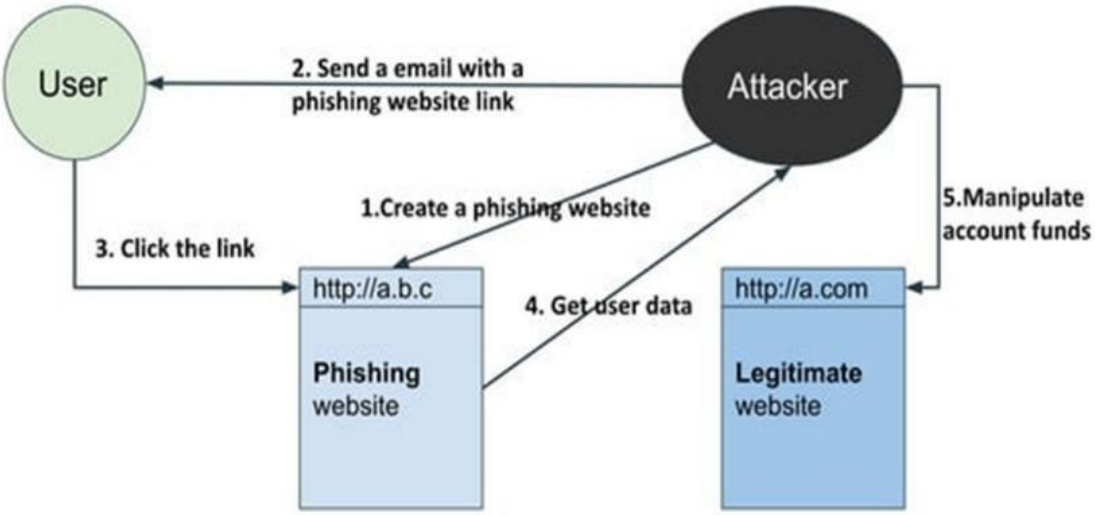
7) GBM Algorithm :

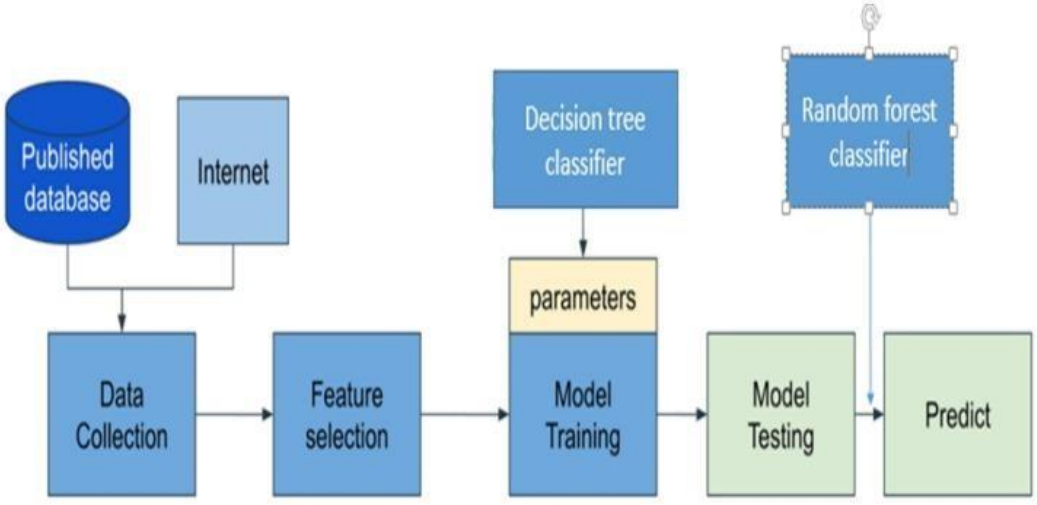
Gradient Boosting Machine (GBM) is considered one of the most powerful boosting algorithms. Boosting technique follows the concept of ensemble learning, and hence it combines multiple simple models (weak learners or base estimators) to generate the final output. It starts with building a primary model from available training data sets then it identifies the errors present in the base model. After identifying the error, a secondary model is built, and further, a third model is introduced in this process. In this way, this process of introducing more models is continued until we get a complete training data set by which model predicts correctly.

8) Support Vector Machine Algorithm :

Support vector machine is another powerful algorithm in machine learning technology. In support vector machine algorithm each data item is plotted as a point in n-dimensional space and support vector machine algorithm constructs separating line for classification of two classes, this separating line is well known as hyperplane. Support vector machine seeks for the closest points called as support vectors and once it finds the closest point it draws a line connecting to them. Support vector machine then construct separating line which bisects and perpendicular to the connecting line. In order to classify data perfectly the margin should be maximum. Here the margin is a distance between hyperplane and support vectors. In real scenario it is not possible to separate complex and non linear data, to solve this problem support vector machine uses kernel trick which transforms lower dimensional space to higher dimensional space.

WORKFLOW DIAGRAM :





Architecture of diagram

Models & Training

Before stating the ML model training, the data is split into 80-20 i.e., 8000 training samples & 2000 testing samples. From the dataset, 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 problem, 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 project are:

- Decision Tree
- Random Forest
- Multilayer Perceptron
- XGBoost
- Autoencoder Neural Network
- Support Vector Machines
- LIGHTGBM
- GBM

All these models are trained on the dataset and evaluation of the model is done with the test dataset.

5. Result and Discussion:

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.

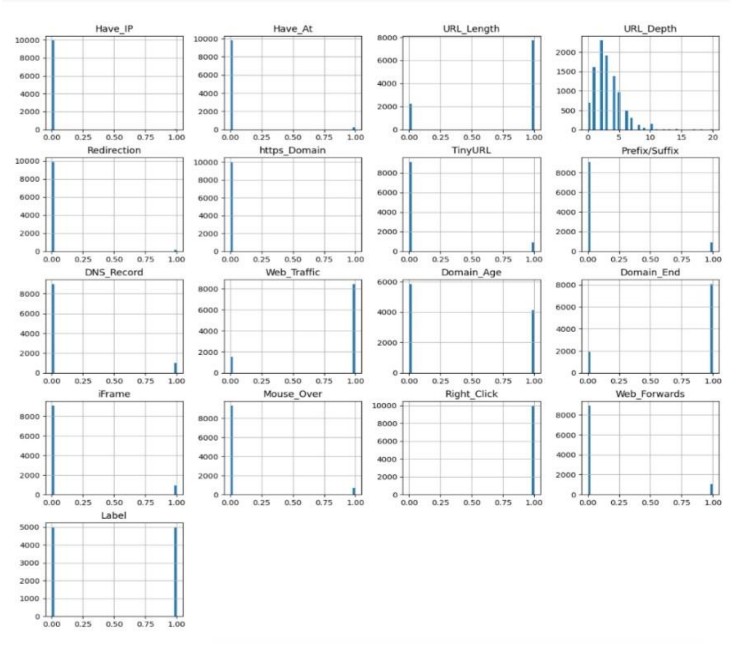


Fig 1 :- Plotting the Data Distribution

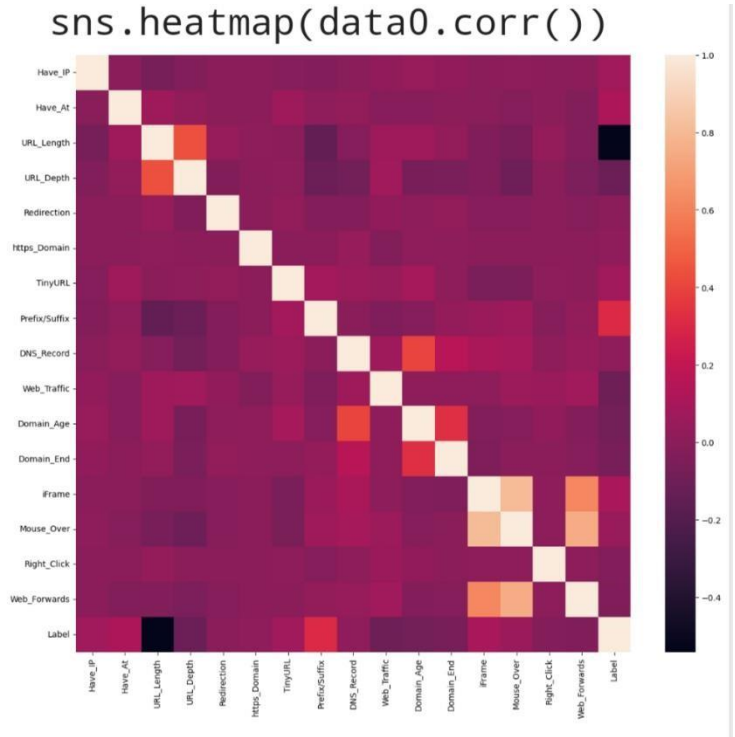


Fig 2 :-

Correlation Heatmap

Scikit-learn tool has been used to import Machine learning algorithms. 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. The supervised machine learning models (classification) considered to train the dataset in this notebook are:

- Decision Tree
- Random Forest
- Multilayer Perceptron
- XGBoost
- Autoencoder Neural Network
- Support Vector Machines
- LIGHTGBM
- GBM

- **Decision Tree**

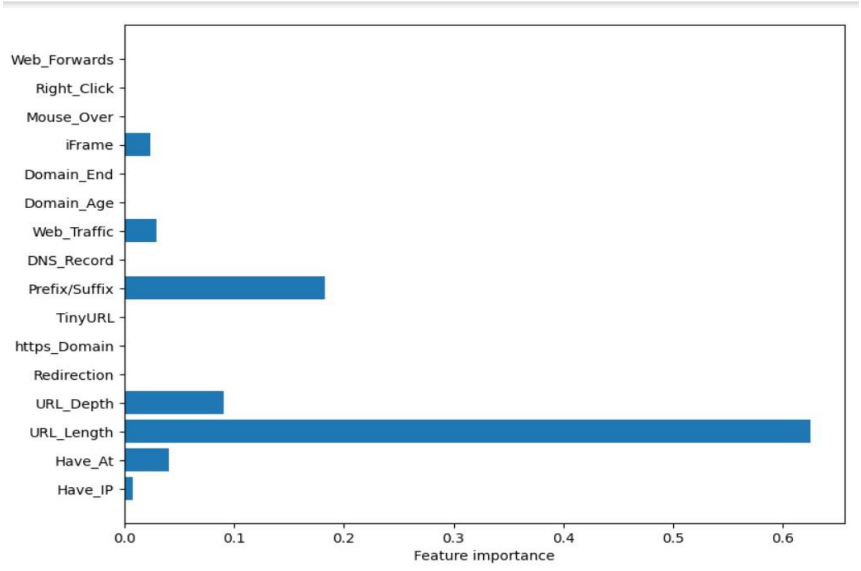


Figure3

```
[ ]  #computing the accuracy of the model performance
     acc_train_tree = accuracy_score(y_train,y_train_tree)
     acc_test_tree = accuracy_score(y_test,y_test_tree)

     print("Decision Tree: Accuracy on training Data: {:.3f}".format(acc_train_tree))
     print("Decision Tree: Accuracy on test Data: {:.3f}".format(acc_test_tree))
```

```
Decision Tree: Accuracy on training Data: 0.814
Decision Tree: Accuracy on test Data: 0.809
```

- **Random Forest**

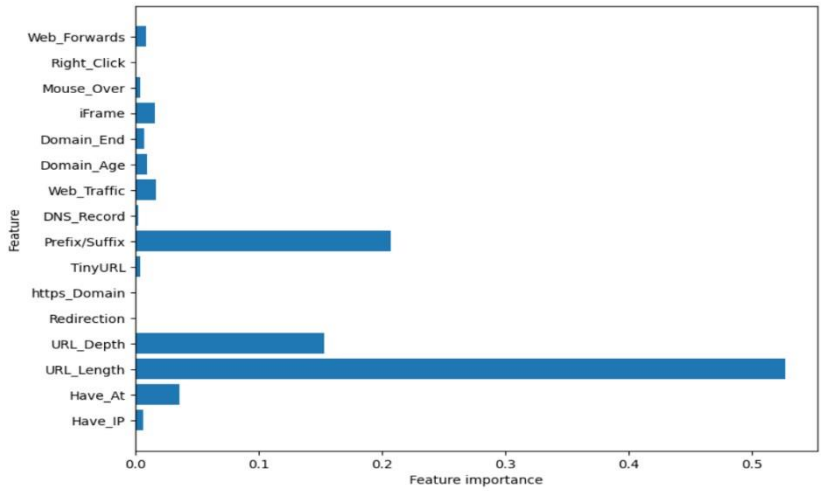


Figure4

```
#computing the accuracy of the model performance
acc_train_forest = accuracy_score(y_train,y_train_forest)
acc_test_forest = accuracy_score(y_test,y_test_forest)

print("Random forest: Accuracy on training Data: {:.3f}".format(acc_train_forest))
print("Random forest: Accuracy on test Data: {:.3f}".format(acc_test_forest))
```

```
Random forest: Accuracy on training Data: 0.815
Random forest: Accuracy on test Data: 0.805
```

- **Multilayer Perceptrons**

```
[ ]  #computing the accuracy of the model performance
     acc_train_mlp = accuracy_score(y_train,y_train_mlp)
     acc_test_mlp = accuracy_score(y_test,y_test_mlp)

     print("Multilayer Perceptrons: Accuracy on training Data: {:.3f}".format(acc_train_mlp))
     print("Multilayer Perceptrons: Accuracy on test Data: {:.3f}".format(acc_test_mlp))
```

```
Multilayer Perceptrons: Accuracy on training Data: 0.868
Multilayer Perceptrons: Accuracy on test Data: 0.848
```

XGBoost

```
[ ] #computing the accuracy of the model performance
    acc_train_xgb = accuracy_score(y_train,y_train_xgb)
    acc_test_xgb = accuracy_score(y_test,y_test_xgb)

    print("XGBoost: Accuracy on training Data: {:.3f}".format(acc_train_xgb))
    print("XGBoost : Accuracy on test Data: {:.3f}".format(acc_test_xgb))
```

```
XGBoost: Accuracy on training Data: 0.870
XGBoost : Accuracy on test Data: 0.850
```

Autoencoder Neural Network

```
[ ] acc_train_auto = autoencoder.evaluate(X_train, X_train)[1]
    acc_test_auto = autoencoder.evaluate(X_test, X_test)[1]

    print('\nAutoencoder: Accuracy on training Data: {:.3f}' .format(acc_train_auto))
    print('Autoencoder: Accuracy on test Data: {:.3f}' .format(acc_test_auto))
```

```
250/250 [==============================] - 1s 2ms/step - loss: 3.2671 - accuracy: 0.0052
63/63 [==============================] - 0s 2ms/step - loss: 3.2046 - accuracy: 0.0040

Autoencoder: Accuracy on training Data: 0.005
Autoencoder: Accuracy on test Data: 0.004
```

Support Vector Machines

```
[ ] #computing the accuracy of the model performance
    acc_train_svm = accuracy_score(y_train,y_train_svm)
    acc_test_svm = accuracy_score(y_test,y_test_svm)

    print("SVM: Accuracy on training Data: {:.3f}".format(acc_train_svm))
    print("SVM : Accuracy on test Data: {:.3f}".format(acc_test_svm))
```

```
SVM: Accuracy on training Data: 0.805
SVM : Accuracy on test Data: 0.789
```

LIGHTGBM

```
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred_binary)
print("LightGBM Accuracy:", accuracy)
```

```
[LightGBM] [Warning] Auto-choosing row-wise multi-threading, the overhead of testing was 0.002656 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
LightGBM Accuracy: 0.851
```

- **GBM**

```
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy of Gradient Boosting Machine:", accuracy)
```

```
Accuracy of Gradient Boosting Machine: 0.8265
```

```
[ ] #Sorting the datafram on accuracy
    results.sort_values(by=['Test Accuracy', 'Train Accuracy'], ascending=False)
```

	ML Model	Train Accuracy	Test Accuracy
3	XGBoost	0.870	0.850
2	Multilayer Perceptrons	0.868	0.848
0	Decision Tree	0.814	0.809
1	Random Forest	0.815	0.805
5	SVM	0.805	0.789
4	AutoEncoder	0.005	0.004

From the obtained results of the above models, the XGBoost Classifier has highest model performance of 86.4%.

6. Conclusion

The demonstration of phishing is turning into an advanced danger to this quickly developing universe of innovation. Today, every nation is focusing on cashless exchanges, business online, tickets that are paperless and so on to update with the growing world. Yet phishing is turning into an impediment to this advancement. Individuals are not feeling web is dependable now. It is conceivable to utilize AI to get information and assemble extraordinary information items. A lay person, completely unconscious of how to recognize a security danger shall never invite the danger of making money related exchanges on the web. Phishers are focusing on instalment industry and cloud benefits the most.

The project means to investigate this region by indicating an utilization instance of recognizing phishing sites utilizing ML. It aimed to build a phishing detection mechanism using machine learning tools and techniques which is efficient, accurate and cost effective.

This project aims to enhance detection method to detect phishing websites using machine learning technology. The proposed method used machine learning classifiers to achieve this and a comparative study of the algorithms was made. A good accuracy score was also achieved. We achieved 86.4% detection accuracy

using the XGBoost Classifier algorithm. Also result shows that classifiers give better performance when we used more data as training data. In future hybrid technology will be implemented to detect phishing websites more accurately, for which the XGBoost Classifier algorithm of machine learning technology will be used.

7.Future Enhancement

Further work can be done to enhance the model by using ensembling models to get greater accuracy score. Ensemble methods is a ML technique that combines many base models to generate an optimal predictive model. Further reaching future work would be combining multiple classifiers, trained on different aspects of the same training set, into a single classifier that may provide a more robust prediction than any of the single classifiers on their own. The project can also include other variants of phishing to complete the system. Looking even further out, the methodology needs to be evaluated on how it might handle collection growth. The collections will ideally grow incrementally over time so there will need to be a way to apply a classifier incrementally to the new data, but also potentially have this classifier receive feedback that might modify it over time.

Reference:

[1] Gunter Ollmann, "The Phishing Guide Understanding & Preventing Phishing Attacks", IBMInternet Security Systems, 2007.

[2] http://dataaspirant.com/2017/01/30/how-decision-tree algorithm-works/

[3] http://dataaspirant.com/2017/05/22/random-forest algorithm-machine-learning/

[4] https://www.kdnuggets.com/2016/07/support-vectormachines-simple-explanation.html

[5] https://blog.keras.io/building-autoencoders-in-keras.html

[6] https://en.wikipedia.org/wiki/Autoencoder

[7] https://mc.ai/a-beginners-guide-to-build-stacked-autoencoder-and-tying-weights-with-it/

[8]https://github.com/shreyagopal/t81_558_deep_learning/blob/master/t81_558_class_14_03_anomaly.ipynb

[9] https://machinelearningmastery.com/save-gradient-boosting-models-xgboost-python/