A MACHINE LEARNING MODEL TO DETECT PHISHING WEBSITES

Swarupa Gowri Adiseshachalam
Dept of Applied Data Science
San Jose State University
swarupagowri.adiseshachalam@sjsu.edu

Mounica Bachu
Dept of Applied Data Science
San Jose State University
mounica.bachu@sjsu.edu

Ramya Sree Modadugu
Dept of Applied Data Science
San Jose State University
venkatasairamyasree.modadugu@sjsu.edu

Shiva Abhishek Varma Penmetsa
Dept of Applied Data Science
San Jose State University
shivaabhishekvarma.penmetsa@sjsu.edu

***Abstract*—The Internet has become one of the most common things every person around the world uses regularly. Especially with the pandemic, there is more possibility that cybercrime will skyrocket in the coming years. Phishing is a type of online fraud in which attackers attempt to obtain confidential information from internet users. Although there are several traditional strategies for detecting and reporting phishing websites, machine learning approaches outperform statistical approaches in terms of accuracy and scale. This comparative study uses a dataset of 95,910 separate phishing and legitimate URLs, and lexical features are extracted from these URLs, which are then used to train machine learning models. On the task of detecting a phishing website, we compare different classification machine learning algorithms, such as Logistic Regression, Random Forest, NaiveBayes, Support Vector Machine, XGBoost, Decision tree, and neural network, using various metrics. XGBoost outperformed the other versions in terms of accuracy and ease of data classification.**

***Index Terms*—Phishing, Machine Learning, XGBoost**

I. Introduction

Security researchers have a ubiquitous concern regarding Phishing. It is a simple task to create a fake website that resembles a legitimate website. Security experts through their expertise can identify fake websites but not all users can identify them and such users become the victim of a phishing attack. In a survey by Avanan, Phishing attacks increased rapidly by 65% from 2016 to 2017. It harms efficiency (67%) and data loss (54%) as well as reputational effects (50%). Phishing reports have been on the rise for the past few years, according to the FBI. Businesses in the United States have lost about US$2 billion a year as a result of their customers falling victim to phishing. The main aim of the attacker is to steal personal credentials which are confidential to the user. Financial services such as banking are now easily available over the Internet making the lives of people easy. Thus, the security and safety of such services must be maintained by detecting phishing websites from legitimate websites.

Phishing has been one of the biggest threats to web security over the past few decades. Using this technique, the user credentials are extracted by masquerading as a genuine website or service over the web. Several types of Phishing are:

- **Spear phishing**: targeting a specific person in an organization or companies.
- **Whaling**: Even more dangerous which targets the CEO, CFO of an organization.
- **Vishing**: attackers steal users information through a voice call.
- **Email phishing**: Most common type of phishing. Hacker acquires the control when the user clicks any 'Link'.

The annual global impact of Phishing and other identity breaches, for example, was reported to be nearly USD 5 billion in the Microsoft Consumer Safer Index (MCSI) study published in 2014. Similarly, the IRS has issued an alert about a rise in phishing attacks, citing a 400% increase in confirmed cases. To combat phishing, a variety of solutions have been suggested, ranging from web user education to improved phishing detection techniques. Because of the complex and changing nature of phishing attacks, the traditional approach to phishing detection has failed. Following further analysis, it was discovered that each phishing attack was distinct from the others.

Machine learning algorithms are an excellent solution to the problem of phishing detection because they enable a system to learn new patterns from data. Although several articles have attempted to detect phishing attacks using machine learning in recent years, we want to go one step further and develop applications that can be quickly deployed in end-user systems to detect phishing attacks.

II. Literature Survey

Many researchers have already performed experiments and addressed their machine learning approach to addressing the issue of phishing websites. The ongoing research will help

us better understand the problem definition and our machine learning approach to solving it. [1] uses a hybrid approach with heuristic features, visual features, and a blacklist and whitelist system to identify features for detecting phishing websites, then feeds these features to machine learning algorithms like logistic regression, decision tree, and random forest to compare results. [2] describes a machine learning method focused on SVMs. It has a low false-positive rate and can spot new temporary phishing websites, minimizing the amount of damage caused by phishing attacks. In [3,] the features are extracted using Lexical Feature Extraction, and the performance of SVM, Random Forest, and Neural Network classification algorithms using the extracted features is compared to find the best one. Since SVM outperformed the others in terms of precision, it was chosen as the final classifier. After a detailed data analysis, the authors [4] suggest a logistic regression model as the best classifier among SVM, Naive Bayes, and Decision tree classifiers. It also measures the success of the four models over time, as well as True Positive, False Positive, True Negative, and False Negative. [5] employs a random forest classifier to identify phishing websites. It employs a variety of metrics to provide a clear image of the detection's efficiency and accuracy. [6] shows how to classify phishing websites with a small dataset by using a single-layer neural network and heuristic functionality.

Researchers addressed various approaches in [7], including Rule-based or Heuristics-based approaches [9], Blacklisting approaches [10], Content-based approaches [11], and Machine Learning-based approaches [12] [13]. In the study of CANTINA [8,] the original and spoofed website was separated using Term Frequency Inverse Document Frequency (TFIDF). This technique results in a 6% of the false-positive rate before adding a few simple heuristics, and the false positive rate decreases significantly when some heuristics are added. In a Heuristic-based approach, Spoof guard, an anti-phishing browser plugin, is used as a solution. SpoofGuard solution has a limitation of generating a high rate of false positives in case of a sophisticated phishing attack. In [10] the Blacklist approach is described, the method used for a long time adds all the entries of Phishing URLs in the blacklist. The URLs are obtained from phishing emails, spam. However, the problem with this approach is all phishing websites are not covered. Blacklist involves time-consuming human feedback; due to this reason, they are ineffective in blocking. In [11] each classifier compares a test website with the original website with its features to find the ground truth based on voting.

In [12] the Naïve Bayes algorithm performance is compared with the SMO algorithm where the SMO algorithm excelled better in terms of accuracy, precision. In [13], the researchers contrasted and recorded the accuracy of each classification algorithm on the dataset for each of the preprocessing algorithms (feature collection and dimensionality reduction). In the trial, Random Forest outperformed Decision Tree, SVM, Naive Bayes, and AdaBoost in terms of precision and AUC.

III. Proposed Approach

We intend to compare few supervised machine learning algorithms over their performance of classifying phishing websites. These machine learning models will be designed with the help of python using its "sklearn" library. Feature Engineering is performed using the python libraries BeautifulSoup, urllib, whois, IP address, tldextract. From the URLs obtained in the base data source, we extracted more lexical features to improve the model. The data with updated features are used to train the machine learning models which would be thoroughly evaluated on different evaluation metrics. Grid Search would be further applied to the Machine Learning model with high accuracy to fine-tune the hyperparameters. The updated relatively high accurate machine learning model will be converted to a pickle file.

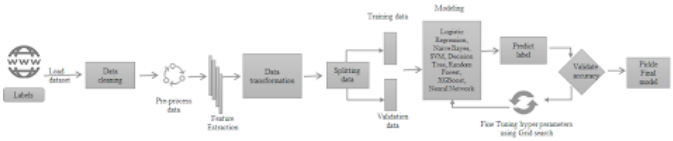

Fig. 1. Architecture

IV. Project Development Methodology

The remote Pair Programming method is used to develop this project. It is an agile development methodology where two developers work together in one workstation. This method has two roles namely driver and navigator. The driver writes the code and the navigator gives instant feedback on the implementation and develops ideas to solve the task. The role is changed regularly between the developers. Through this approach, it was easy to establish teamwork and team coordination. The knowledge sharing was done effectively among all the developers in the team. There was a significant improvement in the efficiency of the code. The time consumed to code the project was comparatively low by following this method.

To establish the pair programming method amidst the pandemic, zoom calls with screen sharing were conducted between the developer-navigator pairs, where the developer will share the screen while coding and the navigator views it to simultaneously give any ideas which improve the code and its methodology.Apart from pair programming, regular scrum meetings were conducted every week to share the status of the tasks and for the assignment of new tasks. The evidence for the same is attached along with this project document in an agile user stories document.Four

data analysts worked on this project to efficiently build the classification model. The analysts worked by sharing the work across all the stages of the project and in its deliverables.

V. Data Engineering

A. Data Collection

The goal of this experiment is to detect if the given website is a phishing website or not. In this experiment, we are using machine learning to train the model which could be further used to detect the given URL. For the same, our model should be trained on large amounts of distinct URLs over different features such as subdomains, the rank of the URL, length of the URL, etc and we should have enough data regarding both the phishing websites and non-phishing websites. From the literature survey, we could not find any open source data source which has a large number of URLs with as many features as possible in Kaggle and GitHub but we came across a labelled dataset in Kaggle which has 95,911 URLs but not with many features[14]. After obtaining the labelled data we did a literature search on how we can extract lexical features from the given URL.

Based on the features required to detect a phishing website explained by [15] we can briefly categorize these features into four categories; Any of these features are derived from URLs and joined to our internal dataset, including address bar-based features, abnormal-based features, HTML and JavaScript-based features, and domain-based features.

Address bar based features: These are the visual features of the URL, some of these distinct features are considered to identify the phishing URL. ex: length of the URL, number of subdomains, if the domain part has IP address, has_@, is_redirecting etc, has_https.
Abnormal features: request URL, Anchor URL, is_abnormal (is Identity part of the URL).
HTML and javascript based features: is the URL redirecting to another URL, does it have any pop- up window, customizing the status bar to display a different URL name.
Domain-based features: features corresponding to the domain of the URL ex: age of the domain, PageRank of the domain, Alexa rank of the URL, google index of the URL etc. and this research paper was used to extract some features along with the features present in the base data source. Features in the base data source are:

- **URL**: URL name
- **The rank of the URL**: rank of the URL
- **Is_ip**: boolean value of whether the URL has an IP address in use other than the URL.
- **Valid**: is the URL details available in the google "Whois" API
- **active duration**: duration of the URL since it's registration in days
- **URL length**: length of the URL
- **is@** : the boolean value of whether the URL has @ in it
- **Isredirect**: the boolean value of whether the URL is redirecting to another URL
- **have dash**: the boolean value of whether the URL has - in it
- **domain length**: length of the domain name.
- **noOfSubdomain**: number of sub-domains present in the URL.
- **Label**: boolean value where the "0" label represents a legitimate website and the "1" label represents the Phishing website.

	domain	ranking	isIp	valid	activeDuration	urlLen	is@	isredirect	haveDash	domainLen	nosOfSubdomain	label
12639	www.paypal.com.nl.cgi.bin.webscr.cmd.login.pro...	10000000	0	0	0	116	0	0	1	115	15	1
47426	songxia-battery.com/js/?mod=www&ssl=0	10000000	0	1	3288	41	0	0	1	19	1	1
49818	netinterbank.org/user/	10000000	0	0	0	22	0	0	0	16	1	1
54035	www.aol.com.update.gatestohell.co.in/AolServic...	10000000	0	0	0	68	0	0	0	36	6	1
43185	bsacodo.fr/wp-content/upgrade/f.htm	10000000	0	0	0	35	0	0	0	10	1	1

Fig. 2. Data frame of the base data source

B. Data Pre-processing

Data is initially cleaned by removing the duplicate values and null values. The count of the dataset decreased from 95,911 to 72,363 entities. The null value count for each column is computed and it was found that no null values are present in the dataset.

C. Random Sampling

We observed that the obtained dataset consists of 31025 legitimate URLs and 41338 phishing URLs i.e. a bias exists towards the phishing websites in the data, to remove the imbalance in the data we applied the under-sampling technique to balance the dataset.

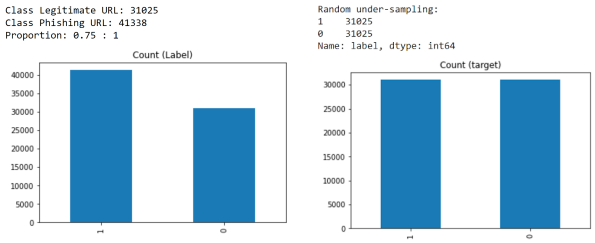

Fig. 3. Distribution before and after Undersampling

D. Data Transformation

From the data quality verification, we found that the "noOfSubdomain" feature values are not correct corresponding to the URL's and they have been re-calculated to maintain the data quality by omitting the "www." which is a subdomain in itself along with the ccTLD such as country code "UK" or "in" etc. New Features extracted based on the definitions present in [15] are:

All the features are pre-processed to maintain a similar range of the values across all features such that all features consist of just “0” or “1” values.

- **Domain_http**: boolean value of whether the URL has ”HTTP” in its domain.
- **LongURL**: boolean value of whether the URL length is less than 54 characters.
- **TinyURL**: boolean value which checks whether the URL consists of shortening services like bit.ly, loopt, .zip, .net etc.
- **Rank**: 0 if rank ¡ 100000 and 1 if rank ¿= 100000
- **isValid**: i) if creation or expiration date values were not found or could not be extracted-1.
  ii) 1 if the age of the domain is less than 6 months else 0.
- **Domain_reg_len**: i) 1 if expiration date was not found or not present in the list.
  ii)It gives 1 if the domain has expired for more than 6 months else 0.
- **SubDomain** : 1, if the URL has one subdomain else 0.
- **activeduration**: 1, if the active duration is less than or equal to one year else 0.

As we extracted the required features from the dataset, the URLs in the feature ‘domain’ were dropped. Since the features longURL and TinyURL have been added, urlLen and domainLen features were no longer required and were removed.

	Rank	isIP	isValid	Domain_reg_len	is@	isRedirect	haveDash	SubDomain	Domain_http	LongURL	TinyURL	Label
0	1	0	1	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	1	1	0	0	0	1
2	0	0	1	0	0	0	0	1	0	0	0	1
3	0	0	1	0	0	0	0	1	0	0	0	0
4	1	0	1	0	0	0	0	0	0	0	0	1

Fig. 4. Data frame of the base data after feature engineering

E. Data Statistics

We explored a relatively new technology called “Lux” to visually explore the data. Lux internally accesses all features and gives the plots of correlations, distributions, and filters by class values, etc. All these plots will be available on the go instead of manually plotting all of the features concerning the kind of plot we want to visualize. Though it is complex to use Lux, this extension can be directly added to the jupyter notebook which gives the plot by just running our dataframe. The distribution of each feature value according to the labels is shown below.

Lux has an internal function termed “Intent”, this function takes an input feature and generates correlation graphs with the input feature. In the plots below, the target variable “Label” is given as the input of the intended function.

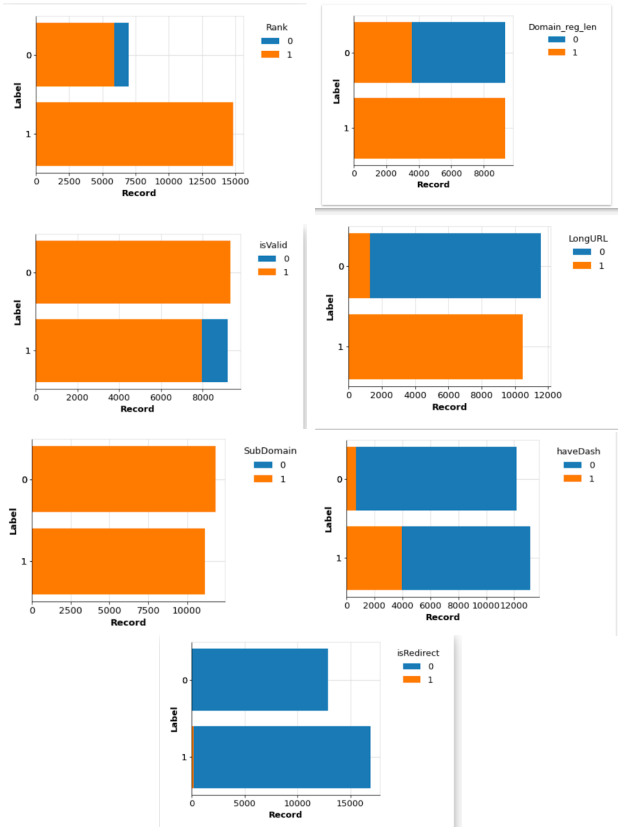

Fig. 5. Distribution of each feature value according to the labels

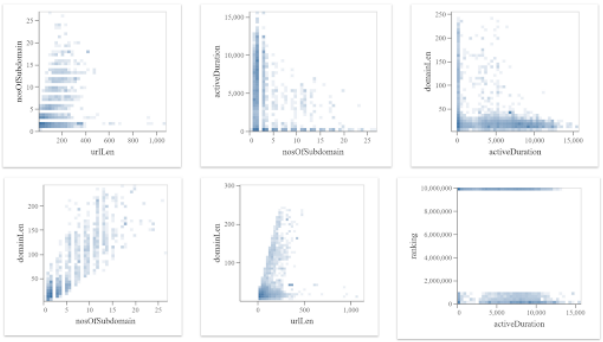

Fig. 6. Correlation of few quantitative features is obtained using the Lux visualization tool

Some of the insights inferred are:

- The number of subdomains in the URL, length of the URL, and length of the domain are in a linear relationship with each other.
- All of the URLs have a subdomain of 1 and most of the features have a rank of 1, if we have ranked as 0 we can consider the URL as a legitimate one.
- If the URL is long, high chances exist that the URL could be a phishing website.
- If the URL has a dash in it and the domain registration length is 1 i.e age of the URL is less than one year, it is more probable that it is a phishing website.

F. Data Exploration

To visually explore the relationships between each feature in the data, a heat map and the correlation available in Lux are used.

Fig. 7. Heat Map representing the relationships among each feature

The below figures are the word clouds of legitimate and phishing websites with 150 words.

Fig. 8. Word cloud of legitimate websites with 150 words

Fig. 9. Word cloud of Phishing websites with 150 words

The decision tree algorithm was used to determine the feature importance of all features to reduce the dimensionality. Based on the feature importance obtained, we dropped the features, isIP, Domain_http, and is@. We observed that 'isValid' and 'isRedirect' have relatively less importance compared to other features, dropping these features will depend on the results obtained from comparing model accuracies with different selected features.

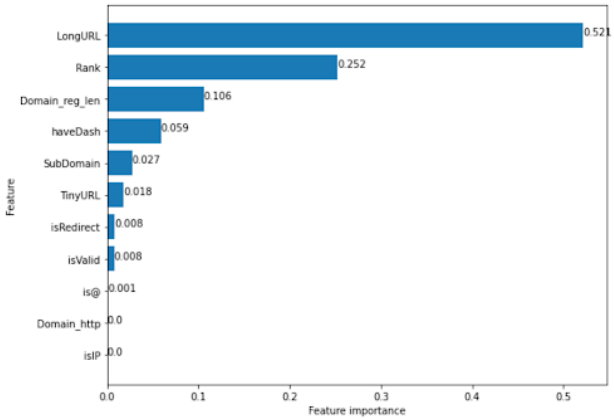

Fig. 10. Feature Importance Plot

VI. Model Development

Various supervised machine learning models are compared to identify the best-suited model for the classification of phishing websites. This section provides the proposed machine learning models that are used for this comparison.

A. *Logistic Regression*

Logistic regression is a classification algorithm that uses supervised machine learning. It explains the relationship between one dependent binary variable and one or more nominal, ordinal independent variables by describing the data. To perform this classification, Logistic Regression employs logistic functions such as the Sigmoid equation.

$$f(x) = \frac{1}{1 + e^{-(x)}}$$

We used this algorithm in our project because of the following reasons.

- It looks at the relationship between one dichotomous dependent variable and one (categorical or continuous) independent variable.
- The number of noise variables is less than or equal to the number of explanatory variables.
- Very fast at classifying unknown records.
- Easy to interpret, implement, and very efficient to train.
- Less prone to over-fitting in a low dimensional dataset having sufficient/enough training samples.

B. *Naïve Bayes*

The Naive Bayes algorithm is a supervised learning algorithm based on the Bayes theorem for solving classification problems. The Naive Bayes Classifier is a simple and successful classification algorithm that aids in the creation of fast

machine learning models that can make accurate predictions quickly.

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Using Bayesian probability terminology, the above equation can be written as

$$Posterior = \frac{Prior * likelihood}{evidence}$$

We tried the Naive Bayes algorithm in our project because

- It is better suited for categorical input variables than numerical variables.
- Easy to implement as only probability needs to be calculated.
- Works very fast and time saving.

C. *Support Vector Machines*

Because of its broad margin classification, Support Vector Machine (SVM) is a common supervised machine learning algorithm. Binary classification is supported by the basic SVM, and multiclass classification is supported by the extension. Since it has boundary vectors (points) that support the boundary line that runs along the class borders, it's called a support vector machine.

SVMs was used to find the best classifier (hyperplane) for maximizing the distance between two sets of training results (margin). A Hyperplane or H, a dividing structured thread, should be equidistant in both groups. Choosing the most functional linear function is crucial.

$$K(X,Y) = (\gamma.X^T Y + r)^d, \gamma > 0$$

SVM justifies being apt for our project because:

- High performance while dealing with higher dimensions.
- If there is a definite distinction between groups, it functions reasonably well.
- Relatively memory efficient.

D. *Decision Tree*

A supervised learning algorithm, a decision tree, is intuitive and has no assumptions or constraints, making it simple to understand and allowing users to see the results. Both classification and regression data can be used with decision trees. Pre-processing requires less time for data preparation. It works well for huge datasets and takes less time.

$$Entropy(S) = \sum_{i=1}^{c} -p_i log_2 p_i$$

$$Gain(S,A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gini = 1 - \sum_{i=1}^{c} (p_i)^2$$

The Decision tree is used in our project to,

- Identify important features required for the classification.
- No use of parameters and hence is one of the simplest algorithms.
- Robust; can manage a huge dataset and make accurate predictions in a limited amount of time.

E. *Random Forest*

Random forest is a classification and regression ensemble learning system that employs several models of many Decision Trees to improve prediction accuracy and is typically trained using the "bagging" method. The fundamental principle of the bagging approach is that integrating multiple learning models increases the result. The main reason for us to choose a random forest algorithm is that,

- The relative importance assigned to the input features can be viewed easily.
- It helps to increase decision tree consistency by reducing overfitting.
- Runs efficiently on large datasets.
- Robust to outliers.

F. *XGBoost*

The gradient boosting system is used in eXtreme Gradient Boosting (XGBoost), a decision-tree-based ensemble machine learning algorithm. Other machine learning algorithms are known to have better results than XGBoost. It has grown in popularity as a machine learning algorithm for dealing with small volumes of structured data and for producing highly accurate results since its inception.

$$\begin{aligned} \mathcal{L}(\phi) &= \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \\ \text{where } \Omega(f) &= \gamma T + \frac{1}{2}\lambda \|w\|^2 \end{aligned} \tag{2}$$

We choose this model for our project because,

- It has the in-built capability to handle missing values.
- Uses parallel computing to its full potential. The model is executed on several CPU cores.
- It supports regularization and can cache.
- Allows the user to do cross-validation at each step of the boosting phase.

G. *Neural Network*

Neural networks are a type of machine learning algorithm that uses several hidden layers and non-linear activation functions to model complex patterns in datasets. A neural network takes an input, runs it through multiple layers of hidden neurons (mini functions with special coefficients that must be learned), and then outputs a projection that represents the sum of the inputs of all the neurons. We wanted to try neural networks for its,

- Ability to learn and model non-linear and complex relationships.

- They have a high fault tolerance so they are often able to adapt, and minor changes in input do not usually result in a change in output.

Apart from the mentioned importances, each algorithm has its limitations. Logistic regression and Naive Bayes are easy and relatively simple to implement but their assumptions may not hold to promptly classify the phishing websites based on their features. SVM, on the other hand, may be time-consuming both while training and testing the data. It is theoretically evident that decision trees tend to overfit. The real-time predictions of the random forest algorithm might be slow and inconsistent if there are a large number of trees in the forest. XGBoost consumes a high training time whereas overfitting can be controlled by hyperparameter tuning. Finally, the neural network is a black box in nature. It can approximate any function but it will not give any insights into the structure of the approximated function. The final model of the project will be the one that has the least limitations as stated.

VII. Evaluation and Comparison

he implementation of phishing websites detection was done using seven different models. The performance metrics accuracy, precision, recall, f1-score, confusion matrix, kappa score, ROC curve, precision-recall curve, time are calculated for measuring the performance of the model.

A. *Confusion Matrix*

The true positives, false positives, true negatives and false negatives predicted by the model are shown in a 2x2 matrix with actual values in one axis and the predicted values in the other.

Actual

Predicted ⇓	Positive	Negative
Positive	True Positive	False Positive
Negative	False Negative	True Negative

B. *Accuracy*

Accuracy is the proportion of the true results predicted among the total number of data examined. The accuracy was calculated for both the training data and the test data prediction for each model.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

C. *Precision*

Precision identifies what proportion of true positives are actual positives. It is a valid choice when we want our prediction to be completely sure.

$$Precision = \frac{TP}{TP + FP}$$

D. *Recall*

Recall states what portion of the actual positives are correctly classified. It is a good choice when we want to capture as many positives as possible.

$$Recall = \frac{TP}{TP + FN}$$

E. *F1-Score*

F1-score is the harmonic mean of precision and recall. It is used when we want the model to have good precision and recall.

$$F1 - score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

F. *ROC-AUC curve*

AUC-ROC curve is the performance measurement for the classification problems at various thresholds. AUC stands for Area Under the Curve and ROC is the abbreviation for Receiver Operating Characteristic. ROC is the probability curve and AUC is the degree of separability. Higher the AUC means that the model is predicting as expected. The ROC curve is a graphical plot that illustrates the trade-off between sensitivity and specificity and plotted with FPR in the x-axis and TPR in the y-axis.

$$TPR/Recall/Sensitivity = \frac{TP}{TP + FN}$$

$$FPR = 1 - Specificity$$

$$Specificity = \frac{TN}{TN + FP}$$

G. *Cohen's Kappa Score*

The Kappa Statistic measures the agreement between the evaluations of actual and predicted values. It describes agreement achieved beyond chance, as a proportion of that agreement that is possible beyond chance.

$$K = \frac{p_o - p_e}{1 - p_e}$$

$$p_o - theproportionofobservedagreements$$

$$p_e - Proportionofagreementsexpectedbychance$$

H. *Accuracy Comparison with Feature Importance*

To select the best model among the different models used, performance metrics were compared and came up with one final model with high performance. Two different comparisons are made here, one is to test the accuracy of the model by dropping the least important features and another comparison is with the performance metrics. Initially, we calculated the importance of each feature from the decision tree feature importance method. From that, we have got the five least important features, among which 'isIP', 'is@' and 'Domain_http' are having '0.0' importance value towards the classification, whereas 'isValid' and 'isRedirect' is having 0.009%. As the first three features are having zero importance, they have been dropped directly from the list of features. Before dropping the features 'isValid' and 'isRedirect', accuracy was compared before and after

dropping the features.

Initial accuracy of all the models was stored, compared with accuracy values after dropping both the 'isValid' and 'isRedirect' features and observed that all the other model's accuracy was dropped down by noticeable difference and the minimal difference was observed for XGBoost and Random-Forest models. To be more precise with the accuracy values, each feature was dropped at a time and compared the results. First, the 'isValid' is dropped and kept the 'isRedirect' as it is and noticed the only minimal difference without causing an impact on XGBoost, RandomForest, and neural network, but when the 'isRedirect' is removed keeping the 'isValid' feature, all the model's accuracy was dropped by 1%. But the performance of XGBoost remained consistent with not much difference.

I. Comparison of Model Performance with Time

After the above-mentioned performance metrics are compared, four models are performing consistently which are Decision tree, XGBoost, Random Forest, and Neural Network. Accuracy, Kappa score, roc curve, and AUC score values are very close and it was hard to decide the final model alone with these metrics. So, the 'time' measure was added to our model to check how much time each model takes to get trained and predict the data. After the time calculation, the neural network is taking more time when compared to the other three models. Among the XGBoost, Random Forest, and Decision tree, all these three took less time and performed with consistent and high accuracy.

Although the decision tree is taking the least time, it was not considered as the final model because it might be unstable for real-time implementation. Along with these metrics, grid search with cross-validation was implemented to identify the hyperparameters and finetune them to improve the accuracy. After fine-tuning, no increase in the accuracy was observed. So, we finally chose the XGBoost as our final model due to its consistent high accuracy as well as the less prediction time. Along with the performance, it also does parallel processing and caching, which will be reliable for real-time streaming data, as well it does regularization by avoiding overfitting and not harming classification.

J. Results of the XGBoost Model

The classification report, confusion matrix and the precision-recall curve of the XGBoost algorithm are shown below.

The consolidated ROC curve and the accuracies of all the models are shown below.

VIII. Conclusion

The first step of protecting ourselves from phishing attacks is to identify the phishing websites and algorithms should be capable enough accurately predicting the URL as False predictions could lead to phishing attacks. The machine

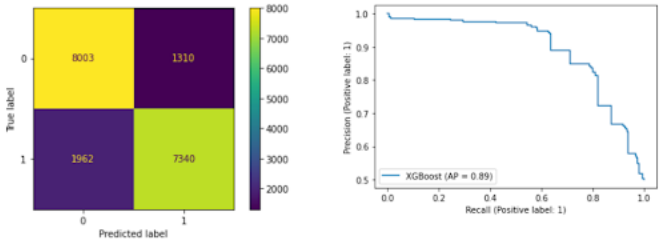

Fig. 11. Confusion Matrix and Precision Recall curve

---------- XGBoost ----------

Classification Report:

	precision	recall	f1-score	support
0	0.80	0.86	0.83	9313
1	0.85	0.79	0.82	9302
accuracy			0.82	18615
macro avg	0.83	0.82	0.82	18615
weighted avg	0.83	0.82	0.82	18615

Fig. 12. Classification report

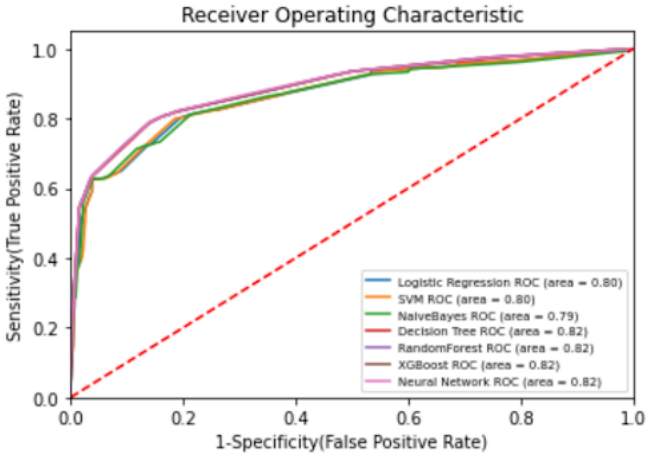

Fig. 13. Consolidated ROC curve of all models

	ML Model	Train Accuracy	Test Accuracy	Kappa Score	Training Time	Testing Time
0	Logistic Regression	0.798273	0.796992	0.593956	0.098634	0.001747
1	Naive Bayes	0.800829	0.799570	0.599210	0.008865	0.004034
2	SVM	0.820859	0.820091	0.640429	26.751432	9.290254
3	Decision Tree	0.819708	0.818587	0.637370	0.009785	0.000915
4	Random Forest	0.820859	0.820252	0.640750	0.646564	0.117824
5	XGBoost	0.820813	0.820252	0.640748	1.107178	0.010425
6	Neural Network	0.820582	0.820091	0.640424	2.995940	0.010825

Fig. 14. Accuracies of all the models

learning model created helps the user to identify a URL before visiting it if the user is suspicious about the URL and helps users to safeguard their sensitive information. We intended to create a machine learning model that could be used to detect a phishing website and give back to the research community by making our raw dataset and Transformed datasets accessible to the public on GitHub. We built this dataset using lexical features and it could be further used to increase the features and create more machine learning models by researchers on the fly without doing all the feature engineering which we had to do due to lack of quality resources initially.

Based on the evaluation of metrics, we can conclude that the XGBoost model is Performing better in terms of accuracy and time metrics. Though the accuracy of XGBoost is similar to the Random Forest and Neural Networks i.e. 82%, the time taken to train the neural network model was comparatively more than the other two models. We chose to select XGBoost over the random forest as XGBoost has capabilities like parallel processing, cache optimization. We observed that the testing time for XGBoost was far less compared to the Random Forest model. Though training time is higher for XGBoost, the model is trained relatively on a less frequent basis and time taken is not that significant a loss.

To succeed in identifying the features relatively more accurately, we need to extract more lexical features about the URLs and add more labelled data to the Model.

IX. Future Work

- Data Collection: The data set used to train and test the models has around 95,000 URLs but collecting more URLs would improve the models' performance on classification.
- This machine learning model could further be used to label new data.
- Further visualization approaches should be used to analyze the test data and identify any patterns in the URLs that are misclassified in the validation data.
- Shifting the entire process of training and testing the models along with the dataset to the cloud, so that it can easily integrate or interact with other applications.
- The main goal of this model is to classify the URL and it warns the user if he is trying to visit a phishing site, More work should be done with help of the tensorFlow.Js or HTML to make this model available as a browser extension.

References

[1] V. Patil, P. Thakkar, C. Shah, T. Bhat and S. P. Godse, "Detection and Prevention of Phishing Websites Using Machine Learning Approach," International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, 2018.

[2] J. Rashid, T. Mahmood, M. W. Nisar and T. Nazir, "Phishing Detection Using Machine Learning Technique," International Conference of Smart Systems and Emerging Technologies (SMARTTECH), Riyadh, 2020.

[3] S. Sindhu, S. P. Patil, A. Sreevalsan, F. Rahman and M. S. A. N., "Phishing Detection using Random Forest, SVM and Neural Network with Backpropagation," International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE), Bengaluru, 2020.

[4] W. Bai, "Phishing Website Detection Based on Machine Learning Algorithm," International Conference on Computing and Data Science (CDS), California, 2020.

[5] S. Parekh, D. Parikh, S. Kotak and S. Sankhe, "A New Method for Detection of Phishing Websites: URL Detection," International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, 2018.

[6] L. A. T. Nguyen, B. L. To, H. K. Nguyen and M. H. Nguyen, "An efficient approach for phishing detection using single-layer neural network," International Conference on Advanced Technologies for Communications (ATC 2014), Hanoi, 2014.

[7] Srushti Patil, Sudhir Dhage "A Methodical Overview on Phishing Detection along with an Organized Way to Construct an Anti-Phishing-Framework", 5th International Conference on Advanced Computing Communication Systems (ICACCS), 2019

[8] Zhang, Y., Hong, J. I., Cranor, L. F. (2007, May). Cantina: a content-based approach to detecting phishing web sites. In Proceedings of the 16th international conference on World Wide Web (pp. 639-648). ACM.

[9] A. A. Ahmed and N. A. Abdullah, "Real Time Detection of Phishing Websites", 2016 IEEE 7th Annual Information Technology Electronics and Mobile Communication Conference (IEMCON), pp. 1-6, 2016.

[10] A. Naga Venkata Sunil and A. Sardana, "A PageRank Based Detection Technique for Phishing Web Sites", 2012 IEEE Symposium on Computers Informatics (ISCI), pp. 58-63, 2012.

[11] N. Shrestha, R. K. Kharel, J. Britt and R. Hasan, "High-performance Classification of Phishing URLs Using a Multi-modal Approach with MapReduce", 2015 IEEE World Congress on Services, pp. 206-212, 2015.

[12] M. Aydin and N. Baykal, "Feature Extraction and Classification Phishing Websites Based on URL", 2015 IEEE Conference on Communications and Network Security

(CNS), pp. 769-770, 2015.

[13] P. Singh, N. Jain and A. Maini, "Investigating the Effect Of Feature Selection and Dimensionality Reduction On Phishing Website Classification Problem", 2015 1st International Conference on Next Generation Computing Technologies (NGCT), pp. 388-393, 2015.

[14] A. Nagariya, "Phishing websites Data," Kaggle, 31-Aug-2020. [Online]. Available: https://www.kaggle.com/aman9d/phishing-data.

[15] Mohammad, R., 2021. Phishing Websites Features. Available at: eprints.hud.ac.uk/id/eprint/24330/6/MohammadPhishing14July2015.pdf .

appendix

APPENDIX

A. *Visualisations*

Many visualisations have been done to analyze the data. Following are the brief explanations of the figures.

- Figure 1 explains the overall architecture of the proposed approach of the project.
- Figure 3 explains the conversion of an unbalanced dataset to a balanced dataset using the bar plots.
- The distribution of each feature across the two classes in the dataset is represented in figure 5.
- The correlation of the quantitative features is described in figure 6.
- The heatmap is used to view the relationship between the features of the dataset. The same is shown in figure 7.
- Figure 8,9 depicts the word cloud for the top 150 words of the legitimate and the phishing sites in the dataset.
- Features are extracted from the feature importance provided by the decision tree classification algorithm and the same is represented in figure 10.
- The classification report of the XGB classifier containing the precision, recall and f1-score across the two classes in the dataset is shown in figure 11,12.
- Figure 11 depicts the precision-recall curve of the XGBoost classifier.
- The consolidated AUC-ROC curve for all the proposed models is displayed in figure 13.
- The table provided in figure 14 lists the training and the test accuracy time, kappa statistic of all the proposed models. This is used to finalize the model and hyperparameter tuning is done for the same.

B. *Significance to the real world*

- CSV file generated with lexical features and transformed dataset is accessible to the public and they can start training their machine learning models on the fly without having to spend time on extracting these features from URLs.
- We Compared seven models and found a better model to detect a Phishing website and researchers can further build over it to find a relatively better model compared to XGBoost without having to test these models if they are using the same dataset.
- In the process of creating a browser extension, we did the first task to identify phishing websites to identify in real time.
- This model could be further used for companies to send a set of URLs in their repository and get labelled data.
- With some enhancements, the final model of this project can be used in internet security services to automatically warn the user by identifying the phishing sites.

C. *Saving the final model for a quick demo*

The final model, which is the XGBoost classifier is saved after hyperparameter tuning using the pickle package in Python. The saved pickle file is submitted along with this report.

D. *Report*

The report is framed by the manual effort of all four members of the team. We referred to many sources to frame this document and the references are mentioned in the 'References' section of this project report.

E. *Version control*

We have published our project code in GitHub publicly. We have also mentioned the softwares and packages required to run our project code. The GitHub link for our project is https://github.com/srimathigowri/MLProjectRepo/tree/master

F. *Key Learnings*

- Domain Knowledge on URLs and lexical features of URL.
- How to implement the Knowledge gained about lexical features in python.
- Our literature search on new visualization tools that can be applied to dataframe led us to "Lux", which can generate various kinds of graphs on the fly and this can help a lot in the data exploration stage.
-
- Applying different machine learning techniques, helped us to correlate between accuracy and time requirements for each model practically.
- Pickle File
- Handling exceptions
- Using Applications like GitHub in real time to manage the project between team members
- With the help of the ROC curve, we observed that Decision Tree, Random Forest and Neural networks gave similar results and It gave us new learning of how we should choose the model based on its simplicity and time taken to train the machine learning model and test with new data.
- Choosing the right kernel in SVM, because depending on the problem each kernel type either increases or

decreases the accuracy. For our problem, the linear kernel was giving a poor performance so replaced it with a polynomial kernel.
- Having not been introduced to neural networks in our course, we got a chance to apply neural networks from Sklearn and Keras and pick the better ones out of these.
- When we trained neural network models with both Keras and sklearn, we learnt that in Keras we can select different activation functions for each layer which could not be done in neural networks trained from sklearn modules.
- Neural networks trained with Keras took more time to train the neural network trained with sklearn, but the accuracy was similar for both the models, so we chose to implement just one neural network model trained with the help of sklearn modules.
- Even though Decision Tree performed better than the Random forest in time requirements, Based on the literature survey we found that, in real time It is better to pick RF over DT because of its ability to reduce the variance in the error.

G. *Prospects of winning a competition*

- Most of the existing (approaches/ research papers) haven't used XGBoost for phishing website detection, we included XGBoost in our model list and it achieved the highest accuracy with the least execution time.
- Grid search cross validation for fine tuning the hyperparameters.

H. *Velocity*

- Not only the chosen model XGBoost can parallel process and cache optimization, which will be reliable for real time streaming data, but it also applies regularization techniques internally to avoid overfitting by not causing any negative impact on the performance in classifying the phishing websites.
- The URLs in the dataset are ones that were already web scraped from the available open source service on the internet. Hence the final model classifying the data will be relevant to real time data and the model will learn over time.

I. *Innovation*

- Instead of directly proceeding with the available dataset, we did a literature search to know about the process of extracting more features and added lexical features to the dataset.
- New visualisation technology 'LUX' was used to get the correlation and distribution among the features and labels.

J. *Evaluation of Performance*

- All the models are compared using different performance metrics like accuracy, precision, recall, Roc curve, precision-recall curve and kappa score.
- Along with the efficiency, the execution time is also an important metric, which was considered along with accuracy comparison. The final model was chosen with high accuracy with the least execution time which is ideal for real time executions.

K. *Team work*

- We are a team of four data analysts, developed this project by sharing equal responsibilities and collaborative effort to complete a task most effectively and efficiently.
- Each team member's opinion and feedback were taken into consideration during the entire process. All the team members were committed to work and timelines to finish the assigned tasks by having weekly meetings, to keep the status updated.

L. *Technical Difficulties*

- One of the challenges we faced in this experiment was feature engineering as we had to learn domain knowledge regarding the URLs to get lexical features in Python. As the dataset is large and many functionalities are happening in the model to get lexical features we encountered exceptions like time out error when we implemented the pipeline to classify large numbers of test data.
- Identifying the packages used to perform URL based operations.
- Implementing Lux was a difficult task as it was a relatively new technology which none of the team members was familiar with, and the challenging part is that when we add lux extension to the jupyter notebook, pandas tend to override the lux dataframe, so at the end, we had to a new file which consists of appropriate visualizations so that neither pandas dataframe nor lux dataframe will not override each other.
- Implementation of new techniques like neural networks, Grid Search.

M. *Practiced Pair Programming*

We did pair programming for this project to establish team coordination and to produce the working code rapidly. A detailed explanation of how we established this approach is mentioned in the 'Project Development Methodology' section of this document.

N. *Practiced Agile/scrum*

Scrum meetings were held regularly weekly to address each task of the project. The user stories of the scrum meetings are attached along with the submission of this document.

O. *Used Grammarly*

After writing the report, a Grammarly check is done to correct all the grammatical errors, improvise the text and get a plagiarism report to finalize the content without any issues. Evidence for the same is submitted along with this document.

P. *Elevator pitch video*

A video on the purpose of our project and the efficiency of the model with the real time data that might impress any phishing sites security provider has been published on YouTube. The Link will be provided in another document.

Q. Used Latex

Latex is used for preparing the whole document with high quality typesetting by following the IEEE format.

R. Creative Presentation techniques

We have used the Prezi tool for our presentation to elevate our project with good animations.

S. Literature Survey

A detailed literature survey was done before we proceeded with this project. We studied around 13 research papers to get an idea of phishing websites and how they are handled via machine learning. We also looked for various machine learning ways through which the detection of phishing websites was implemented in the real time world. Finally, consolidating the literature survey, we came up with a proposed approach that extracts the most relevant and required features from the URLs and efficiently classifies the phishing websites. Detailed information about the study is explained in the 'Literature Survey' section of this document.