Feature Engineering Framework to detect Phishing Websites using URL Analysis

N. Swapna Goud¹
Research scholar
Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation
Vaddeswaram, AP, India

Dr. Anjali Mathur²
Assistant Professor
Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation
Vaddeswaram, AP, India

Abstract—Phishing is a most popular and dangerous cyber-attack in the world of internet. One of the most common attacks in cyber security is to access the personal information of internet users through “Phishing Website”. The major element through which hacker can do this job is through URL. Hacker creates an almost replica of original URL in which there is a very small difference, generally not revealed without keen observation. By pipelining various machine learning algorithms, the proposed model aims to recognize the important features to classify the URL using a recursive feature elimination process. In this work, the data set of various URL records has been collected with 112 features including one target value. In this work, a Machine Learning based model is proposed to identify the significant features, used to classify a URL, the wrapper method recursive feature elimination compares different bagging and boosting machine learning approaches. Ensemble algorithms, Bootstrap Aggregation Algorithms, Boosting and stacking algorithms are used for feature selection. The proposed work has five sections: work on the pre-processing phase, finding the relation between the features of the dataset, automatic selection of number of features using Extra Tree Classifier, comparison of the various ensemble algorithm and finally generates the best features for URL analysis. This paper, designs meta learner with XG BOOST classifier as base classifier and achieved an accuracy of 95%. Out of 112 features, this model has performed an extensive comparative study on feature selection and identified 29 features as core features by performing URL analysis.

Keywords—Recursive feature elimination; principal component analysis; standard scalar transformation; eXtreme gradient boosting classifier; correlation matrix

I. INTRODUCTION

The world of digital suffers a lot from cyber security attacks. The phishing attack can be handled based on source code or URL or image. This research designed the model based on URL features. These URL features are further classified into 4 sub categories. The sub categorization is represented in the below Fig. 1.

With the increase of E-commerce applications, cybercrimes are also increasing rapidly [1]. To solve this issue, researchers are focusing on the detection of phishing websites using Machine Learning and Deep Learning techniques. The dataset contains 112 attributes but all the attributes may not be important. This research tries to find the most important attributes that can determine whether it is a phishing website or not. Some of the websites access are marked as unauthorized by Google but it is difficult to identify all the unauthorized sites by Google search engine. In order to prevent those types of sites, the model compares the every component of the URL to mark it as “Phish Website” [7].

Fig. 1. Classification of URL based Features.

In the proposed research, the feature selection to detect the phishing website is solved using the ensemble algorithms. Ensemble algorithms are popular for their robust results and it designs the meta learners by combining the weak learners, which performed on the same dataset. There are three popular ensemble techniques. The algorithms are as follows.

1) Bootstrap aggregation algorithms: Bootstrap Aggregation is also known as “Bagging”, in which execution of weak classifiers occurs concurrently (or) in parallel. In this method, a sample randomized subset is generated from the entire dataset. The traditional algorithms suffer from high variance. To solve this problem, a component known as “estimator” is added, which is used to create a random sub sample based on the classifier passed.

2) Boosting algorithms: In this type of algorithms, classifiers of same type are combined sequentially to generate a new model. In general, it solves the problem of label incorrectness, which is defined by the one or more models. Boosting also, address the problem of high variance by generating the output of various classifiers and then the average of their predictions are taken into account. It also plays an vital role in reduction of bias.
3) **Stacking algorithms:** It is more advanced than bagging and boosting algorithms because it combines heterogeneous classifiers rather than homogenous. It also designs a model in such way that it combines a meta learners with base learners. In these algorithms, after every iterations, it applies meta model based on the output generated by the previous iteration.

II. **RELATED WORK**

In [8] Mehmet Korkmaz et al. analyzed URL features by comparing the performance of the model using three different datasets and eight machine learning algorithms. This model has analyzed every part of the URL for detection of fake website. The model has recognized 58 features as important features in URL analysis.

Among all the algorithms, Random forest has given best performance on all the three datasets.

In [2] M Somesha et al, proposed an efficient deep learning techniques in which URL’s are passed as the input to the model and features are extracted from the selenium testing tool. The major components of extraction are:

a) **Obfuscation Features:** These are features are extracted from the URL itself. In general, there are 5 types of features as shown in Table I.

| S.NO | Feature Name                  |
|------|-------------------------------|
| 1    | Number of dots in host name   |
| 2    | @symbol in URL                |
| 3    | URL Length                    |
| 4    | IP presence                   |
| 5    | HTTPS presence                |

b) **Hyperlink-based Features:** These features are extracted from the code snippet of the HTML page. Among 8 features, this model has identified six features as important to design the concept of information gain.

c) **Third-Party-based Features:** These features are extracted from the search engines and assisting devices. In this type of features, Alexa has obtained highest information gain. In information gain, it assumes that every feature is dependent on class label and all other features are independent of other features. Every feature is allocated with rank and the ranks less than the threshold value are ignored.

In the next step, the model splits the data into trained and cross validated data. The model had designed a multi feed forward network, with five DNN layers with hyper parameters. In this design, weight values are considered randomly. A Recurrent Neural Network as LSTM is designed to find the relation between the extracted. LSTM is good at handling the vanishing gradient problem. To avoid long term dependencies, LSTM has designed three gates with every gate has its own equation. A CNN with eight layers are designed with back propagation method to classify the URL as suspicious or not.

In [3] Paulius Vaitkevicius et al. conducted a comparative study on various machine learning algorithms and designed a unified ranking model for the detection of phishing website. During the training phase, the model has implemented a cross validation process with setting of hyper parameters and also it has the solved the problem of memorizing the data by decreasing the number of weak learning algorithms. Similarly, at the same time, overfitting and underfitting issues are handled by defining the high bias and high variance. Welch’s T-test, compare the accuracies generated by the two classifiers and compute the statistical difference. Based on the accuracies produced, each classifier has assigned a unique value. Finally, the model has given unique ranking depending on the related work and libraries used.

In [9] Jitendra Kumar et al. proposed five classifiers for detecting phishing websites. Among these random forest and decision tree classifiers are given almost the same accuracy. Author used regular expressions to extract the components from the URL. The author has mainly focused on the three important features namely, URL based features, page based features and domain based features. The data are randomly distributed for forming training and testing dataset.

In [4] Ammar Odeh et al. designed a Multi-Layer Perceptron by considering the URL features. On the extracted features, performed selection, combination of ranking and single attribute evaluation is performed. Later, a subset from these features is generated and is passed as input to the neural network. In this, model is designed with fixed values of hyper parameters and has obtained an accuracy of 93.7%.

In [5] Yazan A et al, proposed AI meta learners combined with base algorithm known as “Extra Tree Classifier”. The first meta learner is “ABET”, this process is carried for 100 iterations and later normal distribution is performed. After training the classifier, it generated a hypothesis and computed the error rate. For every iteration, it updated the weight value and checked whether the threshold value 0.5 is satisfied or not. For the entire generated hypothesis, it computed the argmax function. The second meta learner is “RoFET”, in which the dataset is randomly divided into five subsets, with each containing six features. It created a new dataset by using the bootstrap induction. The newly constructed dataset generated coefficients with the help of sparse rotation matrix. It generated class confidence to determine the class label of each record. The third meta learner is “BET”, 150 iterations are performed over the training dataset and a new dataset is generated by inducer for base algorithm. Argmax concept is applied to find the most frequent predicted class label of the record. The fourth meta classifier is “LBET”, all the weights are initialized to 1/n, where n is number of records in the dataset. The probability estimators are initialized as 0.5. For all the 100 iterations, calculated the weight based probability estimators, then fitted the least square regression function to the weights and updated the values of the weight. Summation of all the classifiers are used to predict the class label. Out of these 4 meta classifiers, LBET has performed better with 97.5% accuracy.
In [6] Waleed Ali et al. proposed PSO based feature weighting approach by encoding the features in a particle. The position and velocities are generated randomly to calculate the fitness of the function. The major goal of the fitness function is to update the local and global values by checking the threshold values regularly. After reaching the termination condition, it generates the optimal weights of all the features. Based on the weights, important features are considered for classification process. These features are passed as input to the five traditional machine learning algorithms and one neural network algorithm. The model has compared all the six algorithms in terms of all evaluation metrics and stated that back propagation neural network has achieved highest accuracy.

In [10] Suleiman Y. Yerima et al, designed 2-layer CNN and 3-layer CNN to detect the phishing websites. The 2-layered architecture contains one convolution and one max pooling layer. The flattened data is transferred to the eight units of neurons that are activated with the help of ReLu function. To prove the efficiency, author has constructed the network with different number of neurons and has achieved 96.6% accuracy for 64 neurons. In the 3-layered architecture additional max pooling layer is added and has achieved 97.1% accuracy for 64 neurons. This model clearly depicted that with the increase of layers and neurons, the accuracy of the model also increases.

III. METHODOLOGY

In this research, the major focus is to reduce the number of features for classifying a URL as phishing website or not. This research work is organized into five sections: Section 1 describes the pre-processing, Section 2 describes the process of finding the relation between the features of the dataset using correlation and principal component analysis (PCA), Section 3 describes the automatic selection of number of features using Extra Tree Classifier, Section 4 compares the bagging and boosting algorithms on the number of features generated using the pipeline mechanism and Section 5 generates the best features based on the best algorithm generated in Section 4.

The feature engineering helps the model to construct the explanatory attributes which plays a vital role in training the model. The major goal of feature selection is to reduce the computation time of the complex models. The process of feature engineering is illustrated in the Fig. 2.

A. Pre-Processing

The process of cleaning and transforming the data is known as “Pre-processing”. A dataset with missing and inconsistent values leads to the poor performance of the model. The dataset used in this research contains 112 features (Table VII in Appendix A) and all the features are numerical values. Used wrangling process to deal with missing and inconsistent values to reduce the computational time, the data should be normally distributed. Normal distribution is achieved by applying standard scalar mechanism.

The equation is

$$SC = \frac{A[i]-\mu}{\sigma}$$ (1)

Where,

A[i], represents attribute(columns) of i\textsuperscript{th} index

\(\mu\), represents the mean

\(\sigma\), represents the standard deviation

After applying standard scalar mechanism, for further processing of the model, the dataset is divided into 80% training data and 20% testing data.

B. Dependency between the Features

Implementing correlation function and PCA functions to find the relation among the features of the dataset. In correlation, the coefficient \(r\), is denoted by 3 possible values [-1, 0, 1], which indicates negative, no relation and positive relation. In this research, Pearson method is implemented, which tries to draw a best fit line between the features. The calculation of Pearson coefficient is represented in the equation 2.

$$PC = \frac{np \sum_{i=1}^{n} A[i]A[j] - \sum_{i=1}^{n} A[i]\sum_{j=1}^{n} A[j]}{\sqrt{(np \sum_{i=1}^{n} (A[i])^2 - (np \sum_{i=1}^{n} A[i] )^2)(np \sum_{j=1}^{n} (A[j])^2 - (np \sum_{j=1}^{n} A[j] )^2)}}$$ (2)

Where,

\(np\), represents number of pairs.

\(A[i], A[j]\), represents attribute values of corresponding indexes.

After applying PCA on our dataset, the result is observed as followed (the attributes whose coefficient value, PC is less than 0.95 and self-denoting correlation values are ignored) Table II.

Principal Component Analysis (PCA) creates a linear relationship among the attributes. The explained variance measures as rank in terms of PCA. It iterates from 1 to \(n\) components specified in the algorithm. It is a two-step process in which first one calculates the covariance matrix and second one computes the Eigen values, Eigen vectors for the covariance matrix. This proposed research made the threshold for covariance as 0.90 and obtained 20 features as important features. The results are tabulated in Table III.
### TABLE II. PEARSON COEFFICIENT VALUES IN CORRELATION MATRIX

| S.No | Attribute 1             | Attribute 2             | PC Value  |
|------|-------------------------|-------------------------|-----------|
| 1    | qty_percent_url         | qty_percent_params      | 0.950514  |
| 2    | qty_equal_url           | qty_and_params          | 0.960387  |
| 3    | qty_and_params          | qty_equal_params        | 0.966137  |
| 4    | tld_present_params      | qty_at_params           | 0.967140  |
| 5    | qty_plus_params         | qty_questionmark_params | 0.969952  |
| 6    | qty_plus_params         | qty_exclamation_params  | 0.969952  |
| 7    | qty_equal_url           | qty_equal_params        | 0.970317  |
| 8    | qty_equal_directory     | qty_asterisk_directory  | 0.974202  |
| 9    | qty_hashtag_params      | qty_plus_params         | 0.976315  |
| 10   | qtySlash_directory      | qty_slash_url           | 0.977222  |
| 11   | qty_equal_directory     | qty_tilde_directory     | 0.978526  |
| 12   | qty_equal_params        | qty_params              | 0.979225  |
| 13   | qty_equal_directory     | qty_and_file            | 0.980746  |
| 14   | qty_and_directory       | qty_equal_directory     | 0.982878  |
| 15   | qty_questionmark_directory | qty_equal_directory     | 0.983013  |
| 16   | qty_asterisk_directory  | qty_and_directory       | 0.986722  |
| 17   | qty_exclamation_params  | qty_questionmark_params | 0.987444  |
| 18   | qty_asterisk_directory  | qty_and_file            | 0.988919  |
| 19   | qty_dollar_directory    | qty_asterisk_directory  | 0.988919  |
| 20   | qty_tilde_directory     | qty_and_directory       | 0.991102  |
| 21   | qty_comma_file          | qty_asterisk_directory  | 0.991163  |
| 22   | qty_percent_file        | qty_percent_directory   | 0.991288  |
| 23   | qty_and_directory       | qty_dollar_directory    | 0.993309  |
| 24   | qty_and_file            | qty_tilde_directory     | 0.993309  |
| 25   | qty_dollar_params       | qty_questionmark_params | 0.993776  |
| 26   | qty_exclamation_params  | qty_space_params        | 0.993776  |
| 27   | qty_dollar_directory    | qty_and_file            | 0.995510  |
| 28   | qty_hashtag_directory   | qty_and_directory       | 0.995562  |
| 29   | qty_dollar_file         | qty_tilde_directory     | 0.995562  |
| 30   | qty_plus_directory      | qty_and_file            | 0.997758  |
| 31   | qty_space_file          | qty_dollar_directory    | 0.997758  |
| 32   | qty_and_directory       | qty_and_file            | 0.997778  |

### TABLE III. EXPLAINED VARIANCE MEASURE OF PCA

| S.No | Explained Variance Ratio |
|------|--------------------------|
| 1    | 0.40925027               |
| 2    | 0.16526287               |
| 3    | 0.05252565               |
| 4    | 0.03512697               |
| 5    | 0.03213711               |
| 6    | 0.02153855               |
| 7    | 0.02043802               |
| 8    | 0.01887185               |
| 9    | 0.01696678               |
| 10   | 0.01528985               |
| 11   | 0.01430187               |
| 12   | 0.01369579               |
| 13   | 0.01319472               |
| 14   | 0.01253416               |
| 15   | 0.01178374               |
| 16   | 0.01158423               |
| 17   | 0.01125422               |
| 18   | 0.01058307               |
| 19   | 0.01013929               |
| 20   | 0.0098498                |

### C. Automatic Selection of Number of Features

In this research, an ensemble technique known as “Extra Tree Classifier” is implemented. This model constructs unpruned decision trees and it considers majority voting to predict the class labels. It follows reverse approach to decision trees and random forest; it tries to fit the decision trees generated by the dataset. The algorithm has three hyper parameters, change of which may lead different evaluations. The hyper parameter k, determines the attribute selections, n-min determines the output noise average and m determines variance level. To evaluate the model, repeated stratified cross validation is performed. This research paper has opted repeated cross validation because in K-fold cross validation, because of the random distribution of the data, every time it is executed, and the output values may vary. To reduce this noisy data problem, in the proposed model the repeated cross validation performs the same task for multiple times and the average of all the executions is taken into account. The number of models generated and executed is shown in the equation 3.

\[
\text{Number of models generated} = k \times r
\]  

(3)

Where,

- k, represents K-folded cross validation
- r, represents number of repetitions
D. Comparative Study

Bagging and boosting algorithms are identified as powerful ensemble algorithms. Pipelined four ensemble algorithms to study the performance of an algorithms on the number of features generated by the Extra Tree Classifier among them three are boosting algorithms and other is bagging algorithm with meta base estimator each algorithm is illustrated.

1) Bagging with meta base estimator: This model fits the randomly generated subsets to the base classifiers. The average predicted values of all the models generated by the subsets are taken in to consideration and the outcome is predicted based on the majority voting. The random subsets are generated based on the estimator specified in the model. In this research, the estimator considered is “Logistic Regression”, which is better to draw the relationships between nominal, interval and ordinal data. The base classifier considered is “Decision Tree Classifier”, in this at every node to split the data, it takes number of conditions into consideration. Using the concept of divide and conquer, the decision tree recursively partitions the data. The splitting of the data is taken care by the parameter, information gain. Here, the information gain is computed by subtracting the weighted sum node impurity of two child nodes from the node impurity of the parent node. The node impurity defines the homogeneity of the labels to which it belongs to.

2) Adaptive boost (AdaBoost) classifier: This algorithm trains the model sequentially by correcting the errors generated by the previous models. In this classifier, initially every record is assigned with a random weight (Generally, it is considered as 1/n, where n represents number of record in the dataset). A decision tree is constructed and is used to classify the records. The predictive labels generated by the decision tree are compared against the actual class labels of the training dataset. This comparison gives the results of correctly and incorrectly classified records. This misclassified data weights summation is assumed as error rate and theses errors are corrected by increasing the weights of incorrect decisions and decreasing the weights of correct decisions. This process is repeated continuously until correct predictions are made.

3) Gradient Boosting (GBM) Classifier: It is almost similar to AdaBoost classifier but it specifies the target values to the next model generator. The target value always depends on the rate of variation among the prediction models. If the error rate is high the target value is set to high otherwise it is set to low. The error value is determined by the gradient coefficient in the loss function. The loss function in this algorithm is considered as mean square error, the objective of this algorithm is to minimize it. It always tries to bring the error rate close to zero.

4) eXtreme Gradient BOOSTing (XGBOOST) Classifier: It is an extended version of gradient boosting algorithm. The major goal of this algorithm is to increase the computational power of the system and optimize the performance of the model as minimum as possible. All the trees construction occurs concurrently in this model. This parallelism improves the CPU utilization time.

| Algorithm 1: Explore Number of Features |
|----------------------------------------|
| **Input**: Phishing Dataset, D         |
| **Output**: Accuracy for 2 to n-1 features |

| Start |
|-------|
| Step 1: Load the dataset, D           |
| Step 2: number_features=111           |
| Step 3: create an empty dictionary, models |
| Step 4: for i=1 to number_features:    |
|   a. call ExtraTreeClassifier(max_features=i) |
|   b. append to the models             |
| Step 5: cv=repeatedstratifiedcrossvalidation() |
| Step 6: scores=cross_val_score()      |
| Step 7: for names, model in models.items(): |
| Step 8: print scores                  |
| Stop                                  |

| D. Comparative Study |

| Algorithm 2: Recursive Feature Elimination with pipeline |
|----------------------------------------------------------|
| **Architecture**                                         |
| **Input**: Phishing Dataset, D                           |
| **Output**: Accuracies of different machine learning algorithm |

| Start |
|-------|
| Step 1: Load the Dataset, D                             |
| Step 2: create a two dimensional matrix, ID store all the columns except the target column |
| Step 3: create a one dimensional matrix, DD store the target column |
| Step 4: ID_train, DD_train, ID_test,DD_test=split the dataset |
| Step 5: Update the ID_train and DD_train with the standard scalar values and fit the transformation |
| Step 6: alg=get_alg()                                    |
| Step 7: create two empty lists, res and name            |
| Step 8: for n,m in alg:                                  |
|   a. sco=eval_alg(m,ID_train,DD_train)                   |
|   b. append sco value to the res list                     |
|   c. append n to the name list                           |
|   d. print the mean of scores, mean(sco) and names n     |
| Stop                                                     |
The feature selection has done using filtering methods, wrapper methods and embedded methods. In this research, the popular wrapper method known as “Recursive Feature Elimination” (RFE) is implemented. In this method, different classifiers are applied on the same model to find the number of important features. Initially, RFE considers all the features as important and it starts constructing subset by eliminating the unwanted features. RFE ranks the features based on their importance. The features with least absolute value are removed.

Procedure for get_alg():
1) create an empty list, algorithms
2) add logistic regression with decision tree classifier as meta estimator to pipeline
3) add adaboost algorithm to pipeline
4) add XGBoost classifier to pipeline
5) add Gradient Boosting classifier to pipeline

Procedure for eval_alg(m, ID_train, DD_train):
   a) call the repeated cross validation function
   b) call the cross evaluation score function

E. Generation of Best Features
The best features are obtained by passing XGBoost as estimator to the recursive feature elimination algorithm and number of significant features are 29. Support and rank are achieved for all the features.

Algorithm 3: Recursive Feature Elimination with XGBOOST Estimator

Input ← Phishing Dataset, D
Output ← Accuracies of different machine learning algorithm

| Feature Number | Accuracy Score | Feature Number | Accuracy Score | Feature Number | Accuracy Score | Feature Number | Accuracy Score | Feature Number | Accuracy Score | Feature Number | Accuracy Score |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 1              | 0.921          | 24             | 0.933          | 47             | 0.937          | 70             | 0.936          | 93             | 0.934          |
| 2              | 0.927          | 25             | 0.934          | 48             | 0.936          | 71             | 0.936          | 94             | 0.934          |
| 3              | 0.922          | 26             | 0.935          | 49             | 0.937          | 72             | 0.938          | 95             | 0.935          |
| 4              | 0.925          | 27             | 0.934          | 50             | 0.937          | 73             | 0.938          | 96             | 0.935          |
| 5              | 0.927          | 28             | 0.928          | 51             | 0.937          | 74             | 0.937          | 97             | 0.935          |
| 6              | 0.926          | 29             | 0.940          | 52             | 0.936          | 75             | 0.933          | 98             | 0.934          |
| 7              | 0.923          | 30             | 0.933          | 53             | 0.939          | 76             | 0.936          | 99             | 0.935          |
| 8              | 0.923          | 31             | 0.936          | 54             | 0.937          | 77             | 0.935          | 100            | 0.935          |
| 9              | 0.927          | 32             | 0.932          | 55             | 0.940          | 78             | 0.937          | 101            | 0.934          |
| 10             | 0.931          | 33             | 0.935          | 56             | 0.937          | 79             | 0.935          | 102            | 0.935          |
| 11             | 0.928          | 34             | 0.935          | 57             | 0.938          | 80             | 0.932          | 103            | 0.933          |
| 12             | 0.926          | 35             | 0.937          | 58             | 0.936          | 81             | 0.932          | 104            | 0.933          |
| 13             | 0.927          | 36             | 0.936          | 59             | 0.940          | 82             | 0.933          | 105            | 0.937          |
| 14             | 0.926          | 37             | 0.935          | 60             | 0.936          | 83             | 0.935          | 106            | 0.934          |
| 15             | 0.930          | 38             | 0.936          | 61             | 0.935          | 84             | 0.937          | 107            | 0.935          |
| 16             | 0.933          | 39             | 0.935          | 62             | 0.936          | 85             | 0.936          | 108            | 0.935          |
| 17             | 0.929          | 40             | 0.934          | 63             | 0.937          | 86             | 0.934          | 109            | 0.934          |
| 18             | 0.932          | 41             | 0.937          | 64             | 0.935          | 87             | 0.936          | 110            | 0.936          |
| 19             | 0.933          | 42             | 0.936          | 65             | 0.935          | 88             | 0.934          | 111            | 0.934          |
| 20             | 0.930          | 43             | 0.937          | 66             | 0.937          | 89             | 0.935          |
| 21             | 0.932          | 44             | 0.938          | 67             | 0.937          | 90             | 0.937          |
| 22             | 0.930          | 45             | 0.933          | 68             | 0.937          | 91             | 0.936          |
| 23             | 0.930          | 46             | 0.939          | 69             | 0.935          | 92             | 0.937          |

IV. EXPERIMENTAL RESULTS
In this work the algorithms are good enough to find the significant features useful for classification process. The feature selection is based on their accuracy score and score for every feature is tabulated in Table IV. When the accuracy is calculated for all 111 features, it is observed that accuracy is more i.e., 94% accuracy when the number of features are 29 and are tabulated in table new. The accuracy scores are plotted in Fig. 3. In the below graph, x-axis represents the number of features and y-axis represents the accuracy.
On the selected number of features, four different algorithms are passed to the recursive feature elimination and the accuracies are tabulated in Table V, and the comparative study is plotted in Fig. 4.

The XGBoost classifier identifies the top 29 features and remaining is allocated with different ranking. The results are tabulated in Table VI.

---

**TABLE V. MACHINE LEARNING ALGORITHM ACCURACIES**

| S.No | Algorithm Name     | Accuracy |
|------|--------------------|----------|
| 1    | Logistic Regression (LR) | 89.8     |
| 2    | Ada Boost (Ada)    | 90.0     |
| 3    | XGBoost            | 93.0     |
| 4    | Gradient Boost (gbm) | 92.8     |

---

**TABLE VI. RANK AND SUPPORT VALUES FOR THE FEATURES**

| Feature Number | Feature Name          | Support | Rank | Feature Number | Feature Name          | Support | Rank |
|----------------|-----------------------|---------|------|----------------|-----------------------|---------|------|
| 1              | qty_dot_url           | True    | 1.00 | 57             | qty_percent_directory | True    | 1.00 |
| 2              | qty_hyphen_url        | True    | 1.00 | 58             | directory_length     | True    | 1.00 |
| 3              | qty_underline_url     | True    | 1.00 | 59             | qty_dot_file         | False   | 69.00 |
| 4              | qty_slash_url         | True    | 1.00 | 60             | qty_hyphen_file      | False   | 43.00 |
| 5              | qty_questionmark_url  | False   | 3.00 | 61             | qty_underline_file   | False   | 41.00 |
| 6              | qty_equal_url         | False   | 2.00 | 62             | qty_slash_file       | False   | 59.00 |
| 7              | qty_at_url            | True    | 1.00 | 63             | qty_questionmark_file| False   | 37.00 |
| 8              | qty_and_url           | True    | 1.00 | 64             | qty_equal_file       | False   | 6.00  |
| 9              | qty_exclamation_url   | False   | 11.00| 65             | qty_at_file          | False   | 10.00 |
| 10             | qty_space_url         | False   | 13.00| 66             | qty_and_file         | False   | 12.00 |
| 11             | qty_tilde_url         | False   | 15.00| 67             | qty_exclamation_file | False   | 14.00 |
| 12             | qty_comma_url         | False   | 17.00| 68             | qty_space_file       | False   | 34.00 |
| 13             | qty_plus_url          | False   | 24.00| 69             | qty_tilde_file       | False   | 35.00 |
| 14             | qty_asterisk_url      | False   | 19.00| 70             | qty_comma_file       | False   | 16.00 |
| 15             | qty_hashtag_url       | False   | 21.00| 71             | qty_plus_file        | False   | 33.00 |
| 16             | qty_dollar_url        | False   | 25.00| 72             | qty_asterisk_file    | False   | 18.00 |
| 17             | qty_percent_url       | False   | 4.00 | 73             | qty_hashtag_file     | False   | 20.00 |
| 18             | qty_tld_url           | False   | 5.00 | 74             | qty_dollar_file      | False   | 22.00 |
| 19             | length_url            | True    | 1.00 | 75             | qty_percent_file     | True    | 1.00  |
| 20             | qty_dot_domain        | True    | 1.00 | 76             | file_length          | True    | 1.00  |
| 21             | qty_hyphen_domain     | False   | 38.00| 77             | qty_dot_params       | False   | 48.00 |
| 22             | qty_underline_domain  | False   | 40.00| 78             | qty_hyphen_params    | False   | 50.00 |
| 23             | qty_slash_domain      | False   | 42.00| 79             | qty_underline_params | False   | 52.00 |
| 24             | qty_questionmark_domain| False| 47.00| 80             | qty_slash_params     | False   | 54.00 |
| 25             | qty_equal_domain      | False   | 49.00| 81             | qty_questionmark_params | False   | 56.00 |
| 26             | qty_at_domain         | False   | 51.00| 82             | qty_equal_params     | False   | 58.00 |
| 27             | qty_and_domain        | False   | 53.00| 83             | qty_at_params        | False   | 60.00 |
| 28             | qty_exclamation_domain| False   | 55.00| 84             | qty_and_params       | False   | 62.00 |
| 29             | qty_space_domain      | False   | 57.00| 85             | qty_exclamation_params| False   | 64.00 |
| 30             | qty_tilde_domain      | False   | 59.00| 86             | qty_space_params     | False   | 66.00 |
| 31             | qty_comma_domain      | False   | 61.00| 87             | qty_tilde_params     | False   | 68.00 |
| 32             | qty_plus_domain       | False   | 63.00| 88             | qty_comma_params     | False   | 28.00 |
| 33             | qty_asterisk_domain   | False   | 65.00| 89             | qty_plus_params      | False   | 7.00  |
| 34             | qty_hashtag_domain    | False   | 67.00| 90             | qty_asterisk_params  | False   | 23.00 |

---
This proposed model of feature engineering, trained the dataset in split ratio of 80% and 20%. This model evaluates the correlation matrix values and principal component analysis values. The values don’t specify the relation between the features clearly. While comparing PCA values and filter values with wrapper methods we found more clarity between the features in terms of exploratory and performance by using wrapper methods. The proposed developed algorithm eXtreme Gradient finds the important features among all the existing 112 features. The major reason for selecting the XGB classification algorithm is when meta classifier i.e., the bagging algorithm, Ada Boost (Ada), XGBoost (XGB), and Gradient Boost (gbm) classifiers are applied, the XGB algorithm has achieved an accuracy of 93%. The Ensemble based recursive feature elimination mechanism constructs a subset, by eliminating the weak features. RFE identify significant feature with minimum number so good classifier is designed. Recursive feature elimination is developed by combining Meta classifier with base classifier and decision tree as a bagging algorithm. The most significant features are highlighted under the column support on the basis of the rank values. All the true values in support column are considered as significant values, these significant features has accuracy value as 1.00. The bagging algorithm, Logistic regression obtained 89.8% accuracy whereas boosting algorithms achieved more than 90% accuracy apart from all XGBoost gave 93.0% accuracy to detect Phishing websites.

V. CONCLUSION

TABLE VII. DATASET USED IN RESEARCH WITH 112 FEATURES

| Feature Number | Feature Name           | Value |
|----------------|------------------------|-------|
| 1              | qty_dot_url            | False |
| 2              | qty_hyphen_url         | False |
| 3              | qty_underline_url      | False |
| 4              | qtySlash_url           | False |
| 5              | qty_questionmark_url   | False |
| 6              | qty_equal_url          | False |
| 7              | qty_at_url             | False |
| 8              | qty_and_directory      | False |
| 9              | qty_exclamation_directory | False |
| 10             | qty_space_directory    | False |
| 11             | qty_tikle_directory    | False |
| 12             | qty_comma_directory    | False |
| 13             | qty_plus_directory     | False |
| 14             | qty_asterisk_directory| False |
| 15             | qty_hashtag_directory  | False |

REFERENCES

[1] Abdulhamit Saberi, Emir Kremenic, Comparison of Adaboost with MultiBoosting for Phishing Website Detection, Procedia Computer Science, Volume 168, 2020, Pages 272-278, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.02.251.

[2] Somesha, M., Pais, A.R., Rao, R.S.et al. Efficient deep learning techniques for the detection of phishing websites. Sādhanā 45, 165 (2020), https://doi.org/10.1007/s12046-020-01392-4

[3] Vaitekivicius, Paulius, and Virginijus Marcinkievicius. “Comparison of Classification Algorithms for Detection of Phishing Websites.” Informatica, 2020, pp. 143-60. DOI.org (Crossref), doi:10.15388/20-INFORM404.

[4] Odeh, A., Keshta, I. & Abdelfattah, E. (2020). Efficient Detection of Phishing Websites Using Multilayer Perceptron. International Association of Online Engineering. Retrieved March 27, 2021 from https://www.learntechlib.org/p/217754/.

[5] Y. A. Alsiariwa, V. E. Adeyemo, A. O. Balogun and A. K. Alazzawi, “AI Meta-Learners and Extra-Trees Algorithm for the Detection of Phishing Websites,” in IEEE Access, vol. 8, pp. 142532-142542, 2020, doi: 10.1109/ACCESS.2020.3013699.

[6] W. Ali and S. Malebary, "Particle Swarm Optimization-Based Feature Weighting for Improving Intelligent Phishing Website Detection," in IEEE Access, vol. 8, pp. 116766-116780, 2020, doi: 10.1109/ACCESS.2020.3005569.

[7] C. Singh and Meenu, "Phishing Website Detection Based on Machine Learning: A Survey," 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2020, pp. 398-404, doi: 10.1109/ICACCS48705.2020.9074400.

[8] M. Korkmaz, O. K. Sahingozi and B. Diri, “Detection of Phishing Websites by Using Machine Learning-Based URL Analysis,” 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2020, pp. 1-7, doi: 10.1109/ICCCNT49239.2020.9225561.

[9] J. Kumar, A. Santhanavijayan, B. Janet, B. Rajendran and B. S. Bindhumadhavu, "Phishing Website Classification and Detection Using Machine Learning," 2020 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2020, pp. 1-6, doi: 10.1109/ICCCI48352.2020.9104161.

[10] S. Y. Yerima and M. K. Alzaylaee, "High Accuracy Phishing Detection with AI Meta Learners and Extra-Trees Algorithm for the Detection of Phishing Websites," in IEEE Access, vol. 8, pp. 142532-142542, 2020, doi: 10.1109/ACCESS.2020.3013699.
| Column       | Value                      |
|-------------|---------------------------|
| 11          | qty_tilde_url              |
| 12          | qty_comma_url              |
| 13          | qty_plus_url               |
| 14          | qty_asterisk_url           |
| 15          | qty_hashtag_url            |
| 16          | qty_dollar_url             |
| 17          | qty_percent_url            |
| 18          | qty_tld_url                |
| 19          | length_url                 |
| 20          | qty_dot_domain             |
| 21          | qty_hyphen_domain          |
| 22          | qty_underline_domain       |
| 23          | qty_slash_domain           |
| 24          | qty_questionmark_domain    |
| 25          | qty_equal_domain           |
| 26          | qty_at_domain              |
| 27          | qty_and_domain             |
| 28          | qty_exclamation_domain     |
| 29          | qty_space_domain           |
| 30          | qty_tilde_domain           |
| 31          | qty_comma_domain           |
| 32          | qty_plus_domain            |
| 33          | qty_asterisk_domain        |
| 34          | qty_hashtag_domain         |
| 35          | qty_dollar_domain          |
| 36          | qty_percent_domain         |
| 37          | qty_vowels_domain          |
| 38          | domain_length              |
| 39          | domain_in_ip               |
| 40          | server_client_domain       |
| 41          | qty_dot_directory          |
| 42          | qty_hyphen_directory       |
| 43          | qty_underline_directory    |
| 44          | qty_slash_directory        |
| 45          | qty_questionmark_directory |
| 46          | qty_equal_directory        |
| 47          | qty_at_directory           |
| 48          | qty_and_directory          |
| 49          | qty_exclamation_directory  |
| 50          | qty_space_directory        |
| 51          | qty_tilde_directory        |
| 52          | qty_comma_directory        |
| 53          | qty_plus_directory         |
| 54          | qty_asterisk_directory     |
| 55          | qty_hashtag_directory      |
| 56          | qty_dollar_directory       |
| 57          | qty_percent_directory      |
| 58          | directory_length           |
| 59          | qty_dot_file               |
| 60          | qty_hyphen_file            |
| 61          | qty_underline_file         |
| 62          | qty_slash_file             |
| 63          | qty_questionmark_file      |
| 64          | qty_equal_file             |
| 65          | qty_at_file                |
| 66          | qty_and_file               |
| 67          | qty_exclamation_file       |
| 68          | qty_space_file             |
| 69          | qty_tilde_file             |
| 70          | qty_comma_file             |
| 71          | qty_plus_file              |
| 72          | qty_asterisk_file          |
| 73          | qty_hashtag_file           |
| 74          | qty_dollar_file            |
| 75          | qty_percent_file           |
| 76          | file_length                |
| 77          | qty_dot_params             |
| 78          | qty_hyphen_params          |
| 79          | qty_underline_params       |
| 80          | qty_slash_params           |
| 81          | qty_questionmark_params    |
| 82          | qty_equal_params           |
| 83          | qty_at_params              |
| 84          | qty_and_params             |
| 85          | qty_exclamation_params     |
| 86          | qty_space_params           |
| 87          | qty_tilde_params           |
| 88          | qty_comma_params           |
| 89          | qty_plus_params            |
| 90          | qty_asterisk_params        |
| 91          | qty_hashtag_params         |
| 92          | qty_dollar_params          |
| 93          | qty_percent_params         |
| 94          | params_length              |
| 95          | tld_present_params         |
| 96          | qty_params                 |
| 97          | email_in_url               |
| 98          | time_response              |
| 99          | domain_spl                 |
| 100         | asa_ip                     |
| 101         | time_domain_activation     |
| 102         | time_domain_expiration     |
| 103         | qty_ip_resolved            |
| 104         | qty_nameservers            |
| 105         | qty_mx_servers             |
| 106         | ttl_hostname               |
| 107         | tls_ssl_certificate        |
| 108         | qty_redirects              |
| 109         | url_google_index           |
| 110         | domain_google_index        |
| 111         | url_shortened              |
| 112         | Phishing(Target value)     |