*

Discover Malicious Websites Using Data Mining Algorithms *

Sai Gowtham Ande
MS in Computer Engineering
Fall 2021
San Jose State University
saigowtham.ande@sjsu.edu

Poorna Srinivas Gutta
Ms in Software Engineering
Fall 2021
San Jose State University
poornasrinivas.gutta@sjsu.edu

Manish Mapakshi
MS in Software Engineering
Fall 2021
San Jose State University
manish.mapakashi@sjsu.edu

Pratibha Awasthi
Masters in Computer Engineering
Fall 2021
San Jose State University
pratibha.awasthi@sjsu.edu

***Abstract*—Phishing is one of the most dangerous threats to your online accounts and data, because these kind of exploits hide behind the guise of being from a reputable company or person,and use elements of social engineering to make victims far more likely to fall for the scam. In order to prevent the user from these practices we try to predict the data based on the URL of the website. In this project initially we try to clean the data and then impute the missing values.Later the Discovery of malicious websites is done by using algorithms like logistic regression,XGBoost, Random Forest and they are evaluated before and after parameter tuning. This project concludes that Random forest gives high accuracy of 95.8 %.**

I. Introduction

Phishing is a deceptive practice in which an attacker attempts to obtain sensitive information from a victim. Emails, text messages, and websites are commonly used in these types of assaults. Phishing websites, which are on the rise these days, have the same appearance as real websites. Their backend, on the other hand, is geared to harvest sensitive information provided by the victim. The machine learning community, which has constructed models and performed classifications of phishing websites, has recently become interested in discovering and detecting phishing websites. This research includes two dataset versions with 58,645 and 88,647 websites categorized as real or fraudulent, respectively. These files contain a collection of legitimate and phishing website examples. Each website is identified by a collection of characteristics that indicate whether it is real or not. Thus, data can be used as a source of information in the machine learning process.

II. Data Description

The data in this presentation was gathered and compiled to develop and analyze several categorization algorithms for detecting phishing websites using URL characteristics, URL resolving metrics and external services. Six groups of attributes can be found in the prepared dataset:

- attributes based on the whole URL properties.
- attributes based on the domain properties
- attributes based on the URL directory properties.
- attributes based on the URL file properties.
- attributes based on the URL parameter properties.
- attributes based on the URL resolving data and external metrics.

As shown in Figure 1, the first group is based on the values of the characteristics on the entire URL string, but the values of the next four groups are based on specific sub-strings. The final set of attributes is based on URL resolve metrics as well as external services like Google's search index.

Fig. 1. Parts of URL

The dataset has 111 features in total, except the target phishing attribute, which indicates if the instance is legitimate (value 0) or phishing (value 1). We have two versions of the dataset, one with 58,645 occurrences with a more or less balanced balance between the target groups, with 30,647 instances categorized as phishing websites and 27,998 instances labeled as legitimate. The second dataset has 88,647 cases, with 30,647 instances tagged as phishing and 58,000 instances identified as valid, with the goal of simulating a real-world situation in which there are more legitimate websites present. We have used dataset small for further analysis and model building as it has more balanced classes.

III. Methods

The methods planned for this project are: Initially, for Data Loading, we have taken the raw data from the data sources available and loaded it into the notebook. After doing a brief analysis, we found that the number of -1's in each feature is comparatively higher but as per the dataset description, the -1's values do not have any

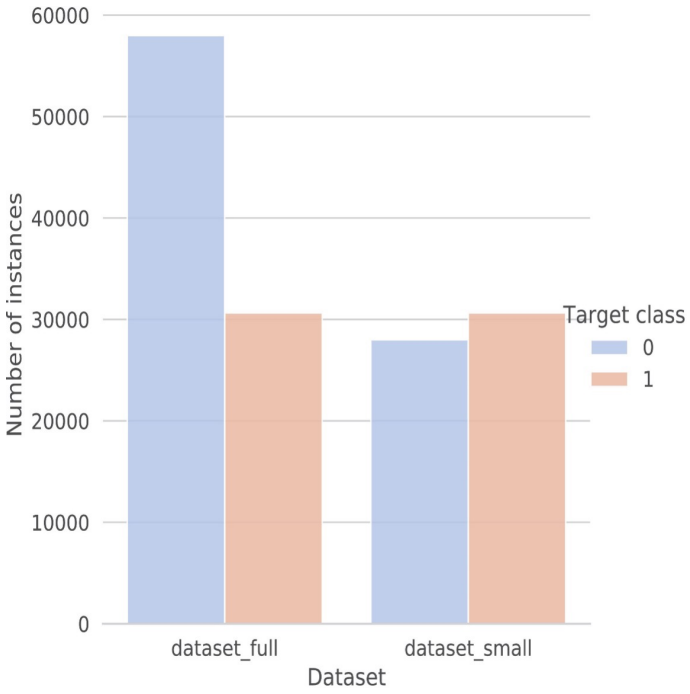

Fig. 2. Phishing feature classification in big dataset and small dataset

significance. So, we decided to divide the analysis into two major methods. The first one is where we consider the -1's as an accurate value and then the other is to treat -1 as a missing value.

A. *Method 1*

In this method, we have tried simulate real-life scenario by considering following: (1) A considerable amount of data involves '-1'. Removing such data will compromise the accuracy of the result, and hence the -1 should be considered. (2) The data set with 88,647 provides sufficient data for training. It also sets realistic accuracy and maintains bias variance tradeoff. Following steps are performed to achieve the accuracy in this method:

- **Data Load and Manipulation**: Data in its true form is raw and not usable. It needs to be cleaned and produced in a form that is more readable and usable. The practice of modifying or altering data in order to make it more understandable and structured is known as data manipulation. Following steps are performed to clean the data:
- **Removing missing or NA values**: In this method we have checked for NA or missing values because they hold no significance. The 'is.na()' method has been used to check for missing data or null values. This approach goes over each and every column of the data set, looking for outliers with NA values that may have an impact on the calculation. As we have considered '-1' as a value with meaning, no such null values were found.
- **Removing Duplicates and Printing Unique Values**: We often come across redundant or repeated values. The purpose here is find the unique value and print them. The unique() function returns the unique values present in a data set. It essentially employs a hash table-based approach to extract non-redundant values from a set of values in the data frame/series data structure.
- **Exploratory Data Analysis**: Exploratory data analysis is used to analyze the data and synthesize the important conclusions. I t will provide us with a fundamental grasp of your data, including its distribution, null values, and more. You can either use graphs or python functions to investigate data. We have analysed the data to find the value counts for '0s' and '1s'. The '0s' represent to be non phishing website whereas the '1s' represent as phishing. It was found that there are 58000 instances for '0s' and 30647 instances for '1s'. The visuals for these values are plotted in figure 3. It can be seen that the x axis represent the number of Phishing websites whereas y gives its count.
- **Correlation**: A correlation heatmap is a heatmap that depicts a two-dimensional correlation matrix between two discrete dimensions, with colored pixels representing data on a monochromatic scale. The first dimension's values display as rows in the table, while the second dimension's values appear as columns. The amount of measurements that match the dimensional value determines the color of the cell. This makes correlation heatmaps great for data analysis since they show differences and variance in the same data while making patterns clearly visible. A colorbar aids the readability and comprehension of data in a correlation heatmap, just as it does in a standard heatmap. The correlation heatmap is plotted and the correlations are printed.
- **Outliers**: An outlier is a value in a random sampling from a population that deviates abnormally from other values. In some ways, this definition delegated the decision of what constitutes abnormality to the analyst (or a consensus procedure). It is required to describe normal observations before anomalous observations may be identified. We have identified the outliers using the z score. In this methodology we have kept the threshold to 3. Also an attempt is made of removing these outliers by IQR. Finally a new data set is generated.

B. *Method 2*

- **Data Preprocessing**: It is the process of transforming raw data into an understandable format. Data pre-processing is used to enhance the quality of the data for future modeling purposes.Several methods have been used in this step to clean and increase the quality of the data. The methods are as follows:
- **Feature Selection using Variance Threshold**: There are several features in the data frame that consist of only a unique value throughout the column. These values have to be dropped as they do not make any impact on the target feature. We have used a method called "Variance Threshold" to remove unique value columns. Variance

Threshold sets up a threshold value and any feature whose variance doesn't meet the threshold. By default, it removes all the zero-variance features. In this model, 13 features don't meet the threshold and these features are dropped.

- **Dropping duplicate rows**: We check for the duplicate rows, where we compare the individual row with another row and check if both the rows are identical. If we found the rows to be identical we try to remove them. By the above method, we have cleaned 1240 rows from the data frame.
- **Eliminating Missing Values**: Most of the features in the data have "-1" as a value. URL attributes can never have negative values. For example, features like quantity, length, and params can never be negative. It is highly illogical to consider these negative values. These "-1" values are considered as missing values here. We calculated the percentage of "-1" values in each and every feature. Features that have more than 80 percent of their values as "-1" are dropped. Later, all the other features consisting of "-1" values are replaced with "NAN". As you can see in the figure below, the missing number library is used to plot all the missing values. It can clearly be noted that a lot of params values are missing. All the "params" features consisting of missing values are dropped. We try to visualize the missing data in the data frame using the missing number library to plot missing values.

Fig. 3. Visualization of missing values

- From the fig. 3, we can understand that some missing values still exist. To deal with these missing data we can use different imputation techniques such as Mean/Median/Mode imputation or use imputer algorithms like KNNimputer(), MissForest(). We have decided to approach this problem by using both the Mean impuation and KNN imputation and later compare results.
- **Mean Imputed Data Analysis**:
- The Null values are replaced using the Mean value. This process is called Mean Imputation.
- **KNN Imputed Data Analysis**:
- The Null data is imputed using KNN imputer with nneighbhors:3. The distance measure used is euclidean distance.
- Stratified split of this data to train and test data with test data size as 25
- Standardizing the data using Standardscaler().
- Three classifiers namely Logistic Regression, RandomForestClassifier, and XGBoost classifier are implemented to analyze this data.
- We are initializing these 3 classifiers with default parameters. Hyperparameter tuning will be done in the next phase after feature selection. Below figures are the metrics obtained after training the models and scoring on test data.
- We try to visualize the data of the categorical columns in KNN Imputed data using a count plot.

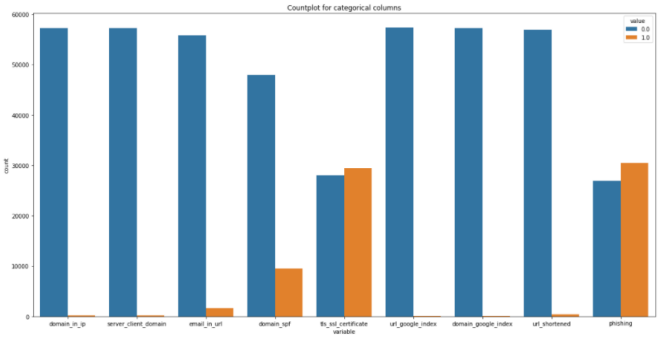

Fig. 4. Countplot for categorical columns

IV. Feature Selection and Importance

In this section, we compute the feature selection and model training for mean imputed data and KNN imputed data. Later improvements are made by hyper-tuning the parameters for better analysis. In machine learning, hyperparameter optimization or tuning is the process of selecting a group of optimal constraints of data patterns for a learning algorithm. A hyperparameter's value controls the learning process, and the other parameter's values are derived accordingly.There are several parameter tuning techniques, but in this project, we shall focus on two of the most widely-used parameter optimizing techniques :

- **Grid Search**: We try every combination of a preset list of hyper-parameter values in Grid Search and evaluate the model for each combination. The pattern is similar to that of a grid, in which all of the values are arranged in a matrix. Each set of parameters is considered, and the precision is recorded. After all of the combinations have been evaluated, the model with the set of parameters that provides the highest accuracy is deemed the best.
- **Random Search**: Random search is a technique for finding the best solution for a built model by using random

combinations of hyperparameters. It tries a variety of value combinations at random. The function is evaluated at several random configurations in the parameter space to optimize with random search.

A. Feature Selection & Importance for Mean Imputed Data using Random Search

- **XGBoost Hyperparameter Tuning**: To get the best optimal results, we tune the XGB using a random search for fitting 3 folds for each of the 100 candidates totaling 300 fits. The optimal parameters obtained are: subsample: 0.1, n_estimators: 500, min_child_weight: 1, max_depth: 5, eta: 0.05, colsample_bytree: 0.1. Using these parameters in the XGBClassifier gives us the feature importance of the data frame.
- **Feature Importance XGBOOST for Mean Imputed data** :

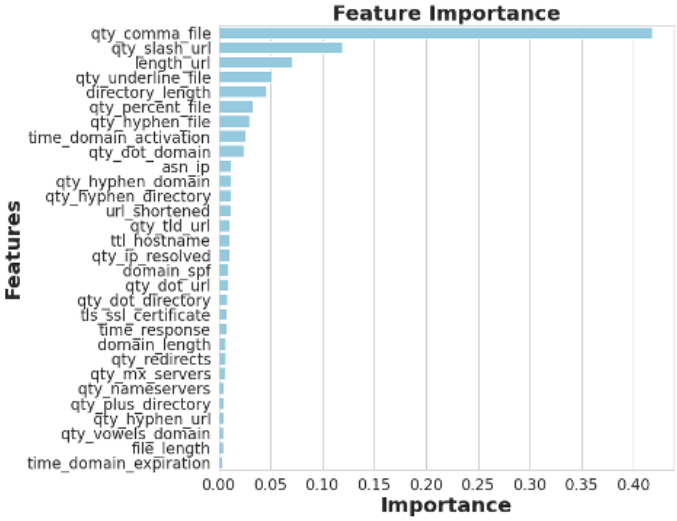

Fig. 5. Feature importances obtained from coefficients

- From the above graph, we can observe that the first 15 features have high importance and for the next feature's importance becomes constant. To continue the analysis, we filter out the important features.
- **Exploratory Data Analysis on Important features**: To find the outliers of the data set, we perform analysis on data using a box plot. This plot shows the data distribution and derives the data points outside the box plot. To observe the obtained important feature data distribution, we performed analysis using a Violin plot. It provides a comparison of the data distribution of different features and helps to determine the outliers.
- **Eliminating Outliers using Inter Quartile Range (IQR) method**: Quartiles can be used to divide each dataset. The first quartile point denotes that 25% of the data points fall below that value, whereas the second quartile is the dataset's median point. On numerical datasets, the inter quartile method detects outliers. The detected outliers are eliminated using the IQR method.

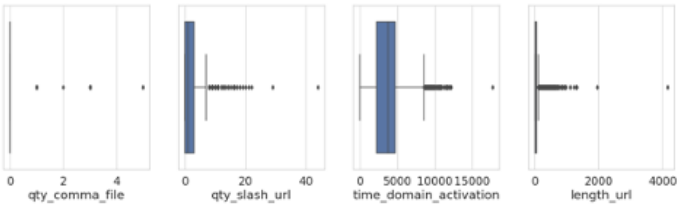

Fig. 6. Boxplot depecting the outliers

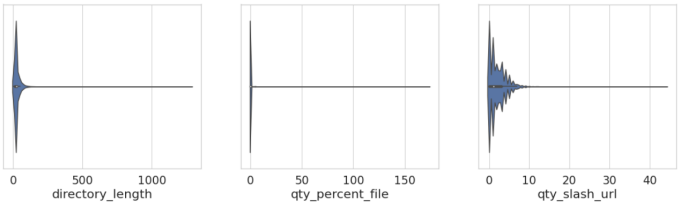

Fig. 7. Violin Plot

- **Correlation**: Correlation describes the relationship between two or more variables. These variables are features of the input data that were used to forecast our target variable. After the data has been modified we try to find the correlation between the features and the target variable 'Phishing'. We can use heatMap to visualise this. From the heatMap, we can infer that time domain activation has negative correlation with phishing feature but qty slash url relatively has a positive correlation.
- **Relativity**: Here, we observe the relation between phishing and other features such as time domain activation and directory_length. From Fig.11, we can infer that the probability of the website being a phishing website is high when the domain activation time is between 3000 to 4000 days. Fig. 12 indicates that the increase in the directory length increases the probability of the website being a phishing website.

B. Feature Selection & Importance for KNN Imputed Data using Random Search

The performance metrics of Logistic Regression can be seen in Fig 13.

- **Feature Importance and Selection**: A serially allocated number that is used for recognition.
- Examining the model's coefficients is the simplest technique to analyze feature importances. It has some influence on the forecast if the assigned coefficient is a large (negative or positive) number. If the coefficient is zero, on the other hand, it has no bearing on the forecast.For Logistic Regression the feature importances is derived from their respective coefficients.
- RandomForest Classifier XGBoost Classifier have built-in feature importance. The decrease in node impurity is weighted by the likelihood of accessing that node to compute feature significance. The number of samples that reach the node divided by the total number of samples yields the node probability. The more significant the

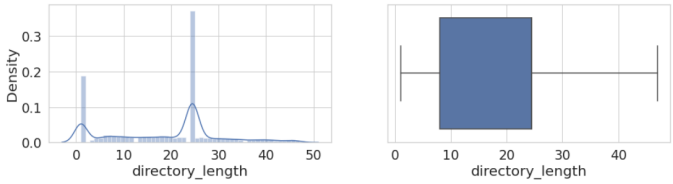

Fig. 8. Box plot of directory_length data after outlier removal

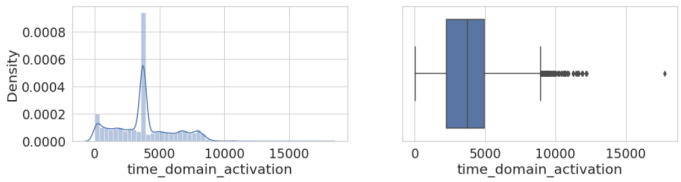

Fig. 9. Box plot of time_domain_activation data after outlier removal

feature, the higher the value. Below are figures of feature importances using 3 models:

- **Feature Importance Logistic Regression**: The feature importance achieved from Logistic Regression can be seen in Fig 16.
- **Feature Importance XGBOOST**: The feature importance achieved from XGBoost Classifier can be seen in Fig 5. A total of 15 features are considered important and those features are selected.
- **Feature Importance RandomForest**: The feature importance achieved from RandomForest can be seen in Fig 17.

From the figure, We can observe that the cross-validation accuracy score doesn't change after selecting 30 features; it flattens as the number of features selected increases. Training our models on these top 30 features should increase the performance of the model.

- **Mean Imputed Data Analysis**:
- This attribute includes details about previous vaccine events.
  From the graph we can clearly depict the important features and then we consider the first 9 features because after that we can clearly see that the importance becomes constant.
- **Analysis of the Important Features**:
  Now we do the basic analysis on the features which we have selected and then try to find the distribution of data and then we try to Feature engineering.So, at first we plot the box plots inorder to depict the outliers.
  we can clearly see from Fig.11 that the outliers exist in the above features like length_url,qty_slash_url. Now we try to plot the Violin plot for the remaining features inorder to find the distribution of the data
- **Outliers**: Earlier we observed the outliers on the selected features.So, now we try to remove the outliers from the selected columns using the Inter Quartile Range (IQR) methodology.

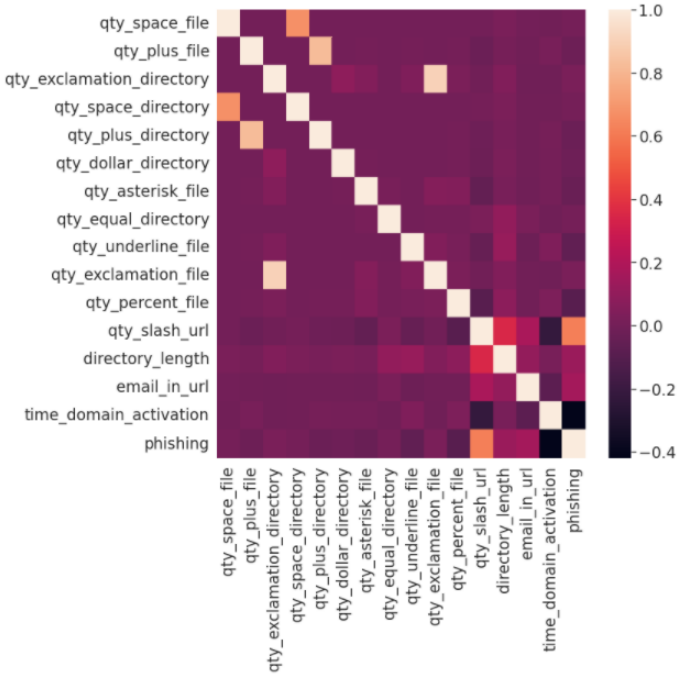

Fig. 10. Feature importances obtained from coefficients

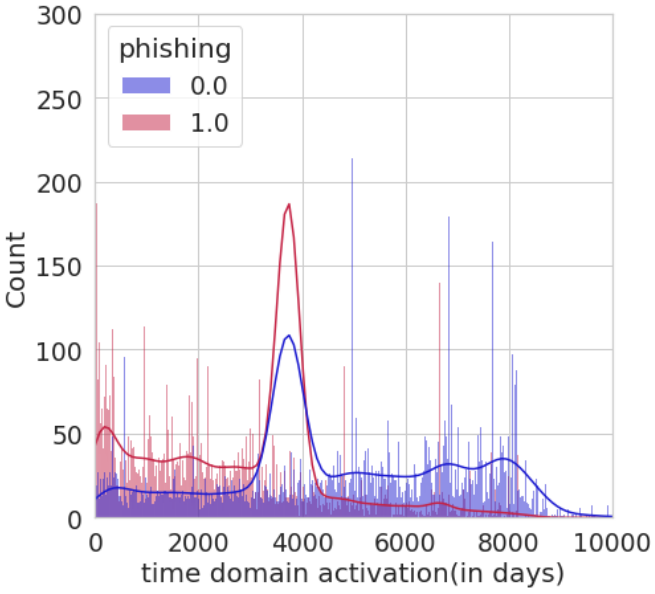

Fig. 11. Relation between phishing feature and time domain activation

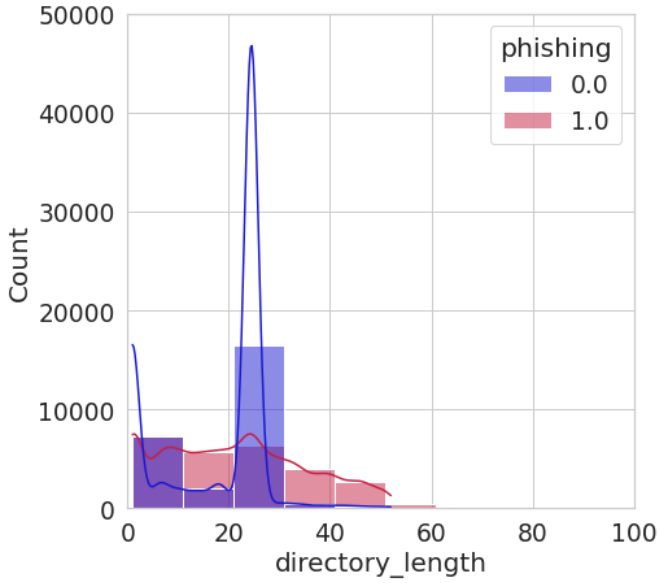

Fig. 12. Relation between phishing feature and directory_length

```
Training accuracy: 0.8889508280491487
test accuracy: 0.8862876254180602
precision score: 0.8870946906084485
recall: 0.9005901639344263
f1 score: 0.8937914877001171
              precision    recall  f1-score   support

         0.0       0.89      0.87      0.88      6727
         1.0       0.89      0.90      0.89      7625

    accuracy                           0.89     14352
   macro avg       0.89      0.89      0.89     14352
weighted avg       0.89      0.89      0.89     14352
```

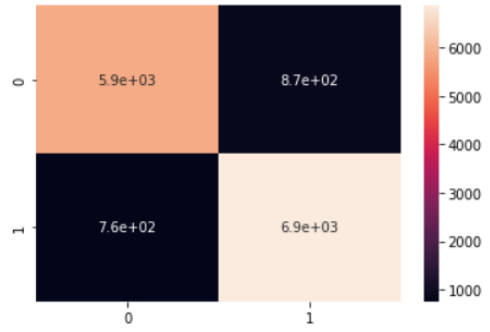

Fig. 13. Logistic Regression performance metrics

From the Fig. 13 we can clearly say that there a negative correlation between for the feature time_domain_activation and phishing, a positive correlation between features like qty_slash_url and phishing is found.

- From the fig.14 we can clearly see that if the length of the directory is more than 30 there is a high probability chance of URL being a phishing website.
- From the fig.15 we can clearly see that as the length of URL increases, it is most likely to be a phishing website.

```
Training accuracy: 0.9733119643230437
test accuracy: 0.9494147157190636
precision score: 0.9538218655440074
recall: 0.9508196721311475
f1 score: 0.9523184027321686
              precision    recall  f1-score   support

         0.0       0.94      0.95      0.95      6727
         1.0       0.95      0.95      0.95      7625

    accuracy                           0.95     14352
   macro avg       0.95      0.95      0.95     14352
weighted avg       0.95      0.95      0.95     14352
```

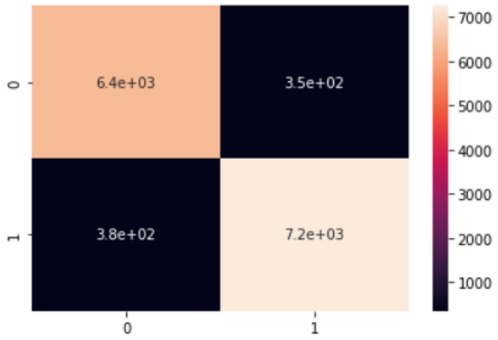

Fig. 14. XGBoost performance metrics

```
Training accuracy: 0.9999767728149026
test accuracy: 0.9509476031215162
precision score: 0.9522938178015946
recall: 0.9555409836065574
f1 score: 0.9539146373396177
              precision    recall  f1-score   support

         0.0       0.95      0.95      0.95      6727
         1.0       0.95      0.96      0.95      7625

    accuracy                           0.95     14352
   macro avg       0.95      0.95      0.95     14352
weighted avg       0.95      0.95      0.95     14352
```

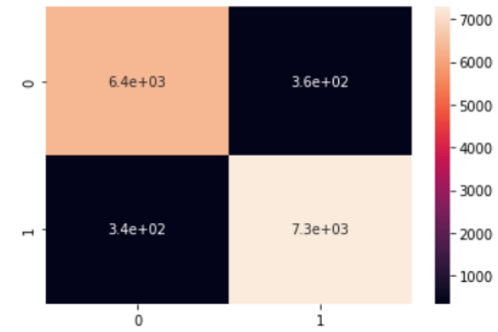

Fig. 15. RandomForest performance metrics

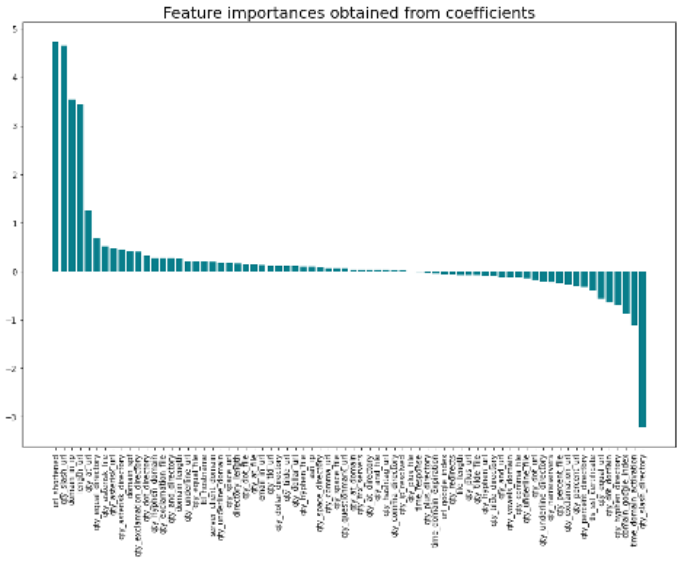

Fig. 16. Feature importances obtained from coefficients

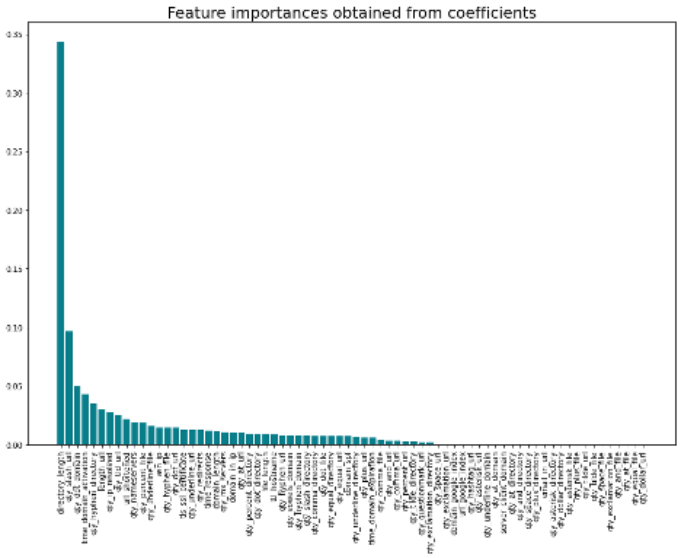

Fig. 17. Feature importances obtained from coefficients

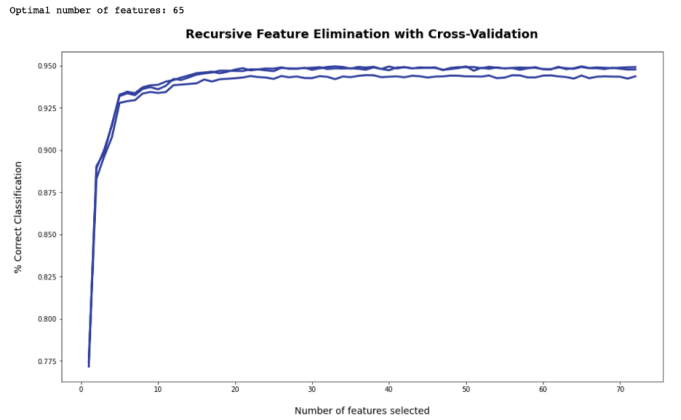

Fig. 18. Recursive Feature Elimination with Cross-Validation

- From the scatter plot we can see that the probability of URl is phishing if the value lies between 0 and 5 for the features qty_percent_file,qty_hyphen_directory.
- Likewise the below scatter plot we can see that the probability of URl is phishing is more as the value of qty_slash_url is more than 4.

V. Modeling

- The accuracy is obtained using various models. The first two models were trained on Mean imputed data while the other models were trained on KNN imputed data. The following are the models:
- **Logistic Regression**: It is used in statistical software to understand the relationship between the dependent variable and one or more independent variables by estimating probabilities using a logistic regression equation. The accuracy found through Logistic Regression is 99.99 percent. Hence, it can be understood that this model is over-fitting.
- **KNN Classifier**: The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. The accuracy found KNeighbour classifier is 95.75 percent.
- **XGB Classifier**: XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. The accuracy found through XGB CLassifier is 95.11 percent.
- **Logistic Regression**: It is used in statistical software to understand the relationship between the dependent variable and one or more independent variables by estimating probabilities using a logistic regression equation. The accuracy found through Logistic Regression is 89.05 percent.
- **Random Forest**: When a large number of decision tree operate as an ensemble, they make up Random Forest. Each tree in the random forest produces a class prediction, and the class with the most votes becomes the prediction of our model. The accuracy found through Random Forest is 95.63 percent.

VI. Comparisons

When the analysis of both the imputation methods are observed, it is understood that same models produce different outputs on both the imputed data. For example, Logistic Regression applied on KNN imputation has produced an accuracy of 95 percent, but Logistic Regression applied on Mean Imputation was overfitting the model. However, from the analysis of all 5 models it can be inferred that models built with both the methods have achieved the highest accuracy of 95 percent.

VII. Conclusion

From the above analysis, it is understood that the highest accuracy for mean imputed data is obtained through KNN

Classifier. KNN Classifier has produced 95.75 percent accuracy. The highest accuracy for KNN imputed data is produced by RandomForest Classifier. RandomForest Classifier has produced 95.63 percent of accuracy. It can be observed that the important features for both KNN imputed analysis and Mean imputed analysis is based on the attributes relating to URL and External services.

VIII. References

- https://www.sciencedirect.com/science/article
- https://data.mendeley.com/datasets/72ptz43s9v/1
- https://scikit-learn.org/stable/modules/impute.html
- https://www.sciencedirect.com/topics/mathematics/imputation-method
- https://machinelearningmastery.com/calculate-feature-importance-with-python/
- https://towardsdatascience.com/explaining-feature-importance-by-example-of-a-random-forest-d9166011959e
- https://machinelearningmastery.com/data-visualization-methods-in-python/
- https://www.geeksforgeeks.org/interquartile-range-and-quartile-deviation-using-numpy-and-scipy/
- https://resources.infosecinstitute.com/topics/phishing/gref
- https://arxiv.org/pdf/2009.11116.pdf