*

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*—**

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.

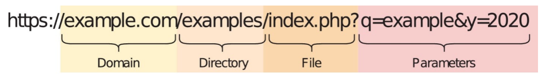


Fig. 1. Parts of URL

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

A. *Method 1*

- **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. Our dataset consists of imbalanced data. 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 that consist of duplicate values in it's columns. These values have to be dropped to reduce the dimensionality. We have used a method called "Variance Threshold" for feature selection. 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, there are 13 features

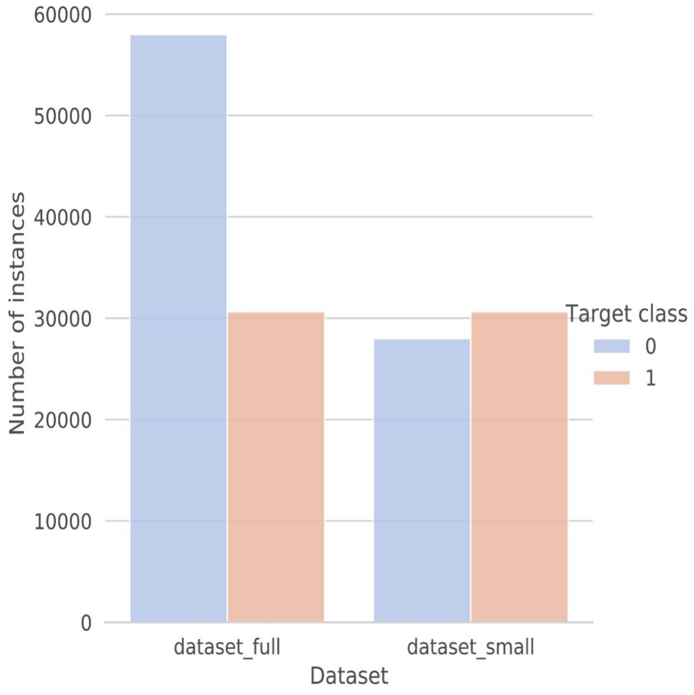


Fig. 2. Phishing feature classification in big dataset and small dataset

which don't meet the threshold and these features are dropped.

- **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 which have more than 80 percentage of its 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.
- 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.
- **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().

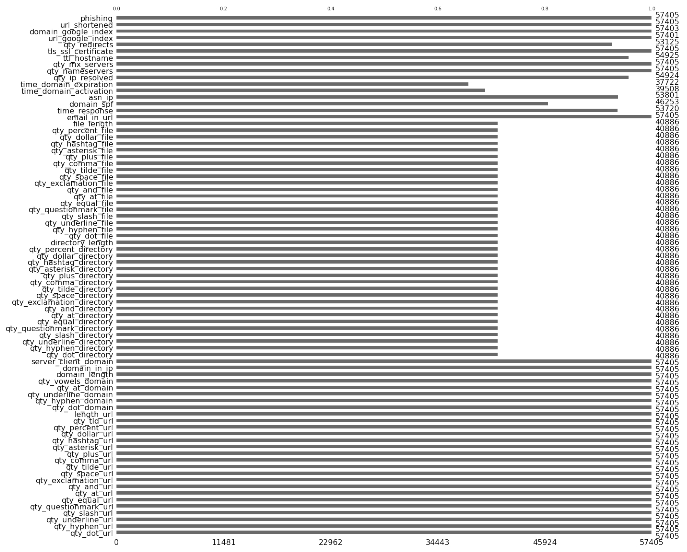


Fig. 3. Visualization of missing values

- 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.
- **Logistic Regression performance metrics**: Fig. 4

```
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
```

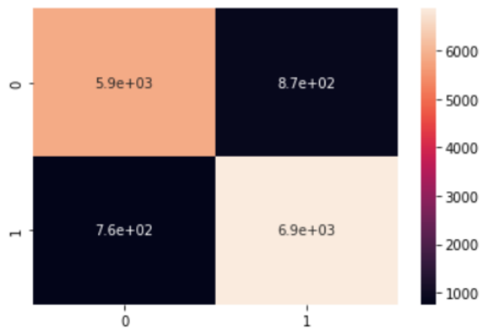
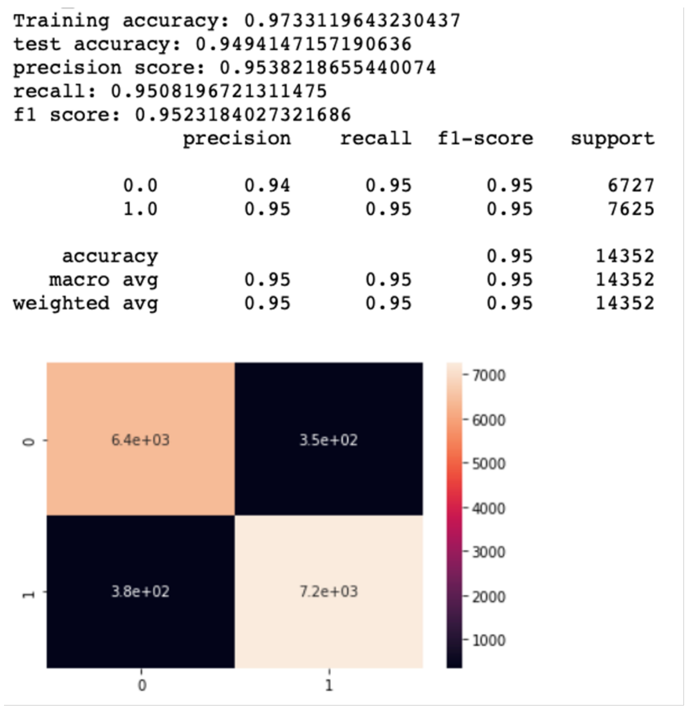


Fig. 4. Logistic Regression performance metrics

- **XGBoost performance metrics**: Fig. 5
- **RandomForest performance metrics**: Fig. 6

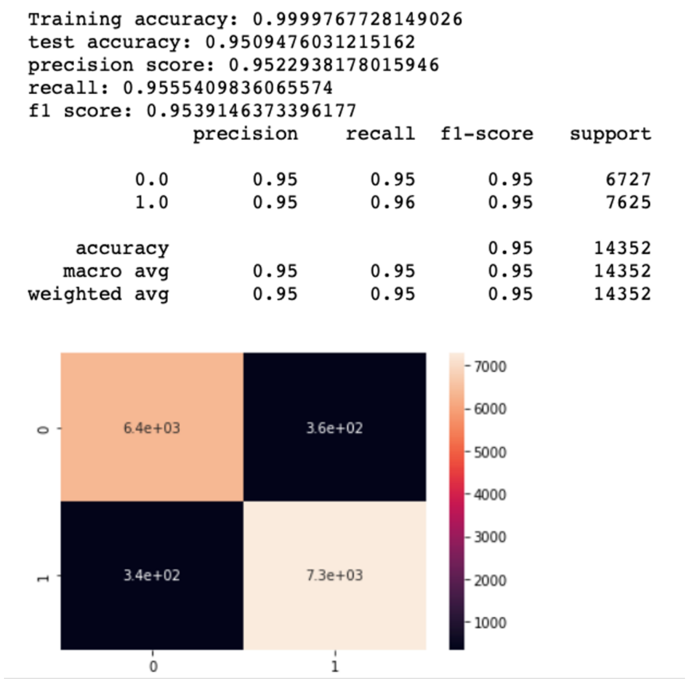


```
Training accuracy: 0.9733119643230437
test accuracy: 0.9494147157190636
precision score: 0.9538218655440074
recall: 0.9508196721311475
f1 score: 0.9523184027321686
              precision    recall  f1-score   support

         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. 5. 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. 6. RandomForest performance metrics

- **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 feature, the higher the value. Below are figures of feature importances using 3 models:
- **Feature Importance Logistic Regression**:Fig. 7

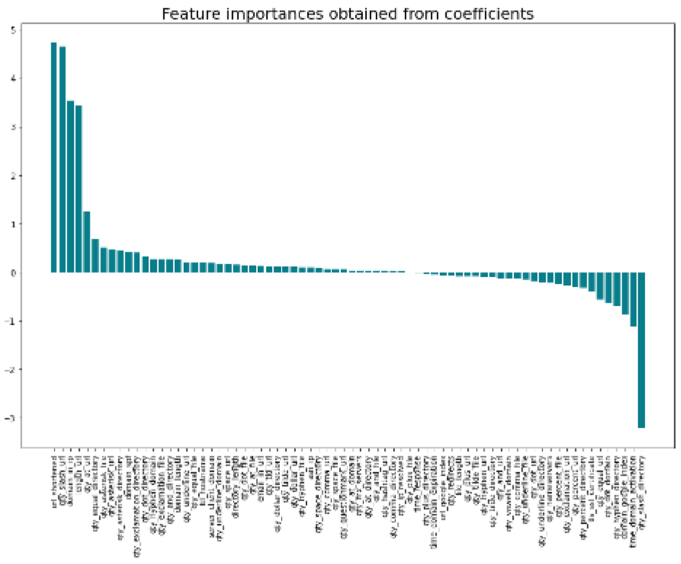


Fig. 7. Feature importances obtained from coefficients

- **Feature Importance XGBOOST**: Fig. 8
- **Feature Importance RandomForest**:Fig. 9
  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.
- **Feature Importance XGBOOST** : Fig.10
  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

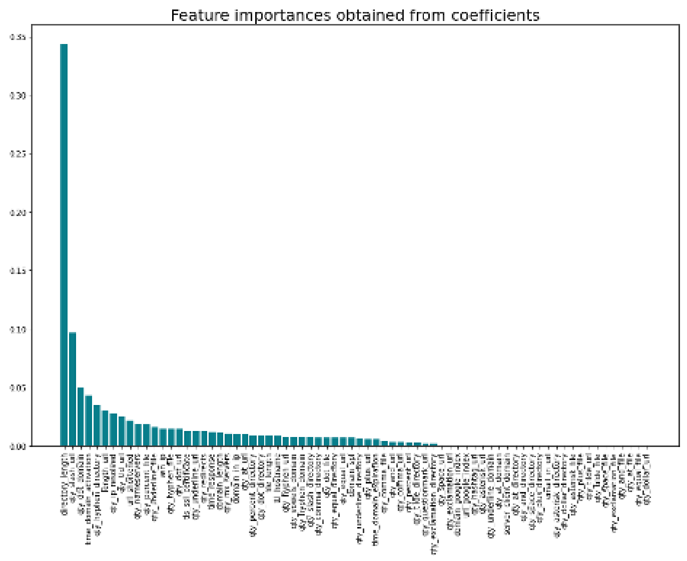


Fig. 8. Feature importances obtained from coefficients

```
Optimal number of features: 65
```

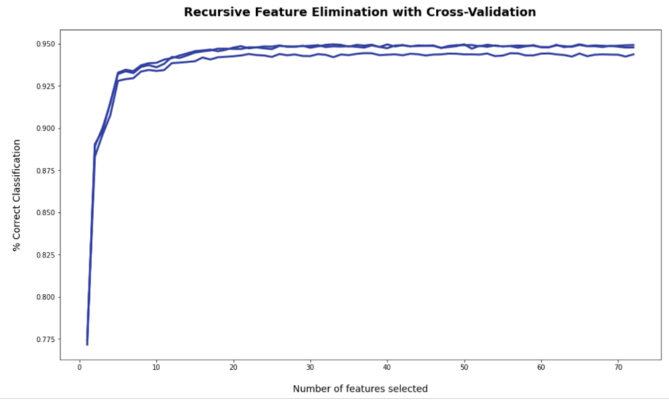


Fig. 9. Recursive Feature Elimination with Cross-Validation

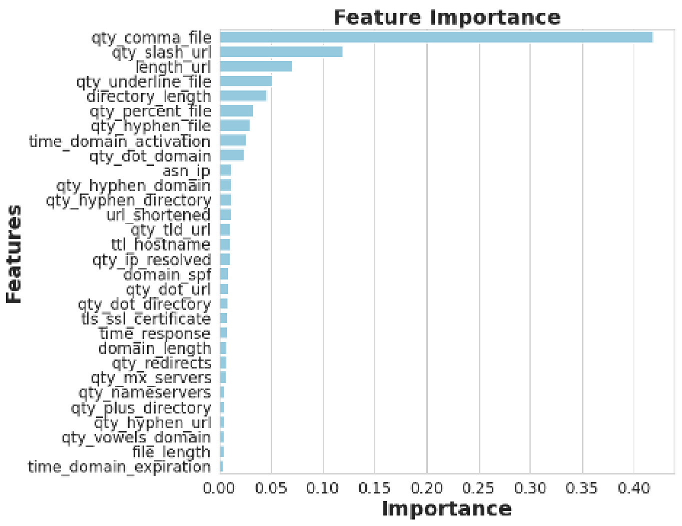


Fig. 10. Feature importances obtained from coefficients

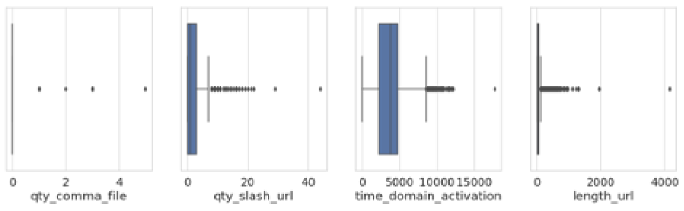


Fig. 11. Boxplot depecting the outliers

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

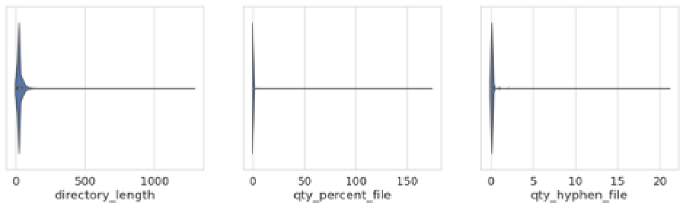


Fig. 12. Violin Plot

- **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.
- **Correlation**: 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 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.
- **Relativity**: Here we try to find the relation between the length features and the phishing feature.
- 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.
- 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.

IV. Modeling

- The accuracy is retrieved using different models. The first two models were trained on Mean imputed data while the

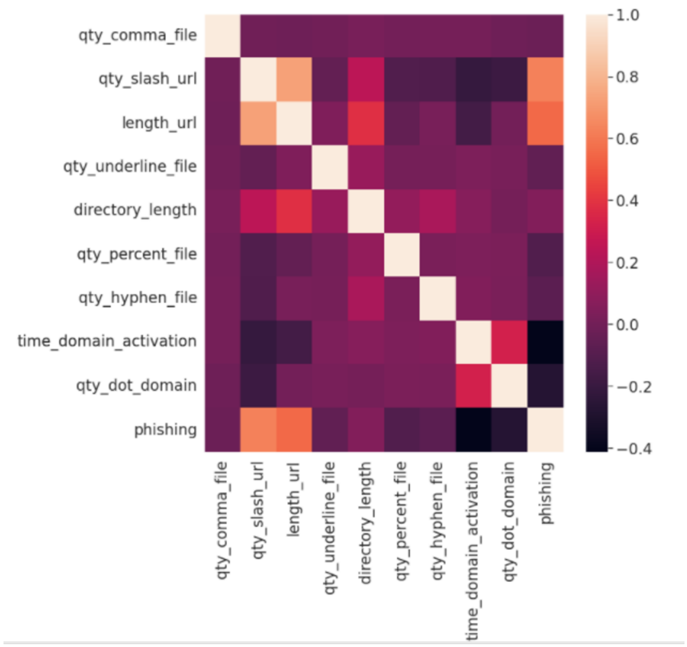


Fig. 13. Feature importances obtained from coefficients

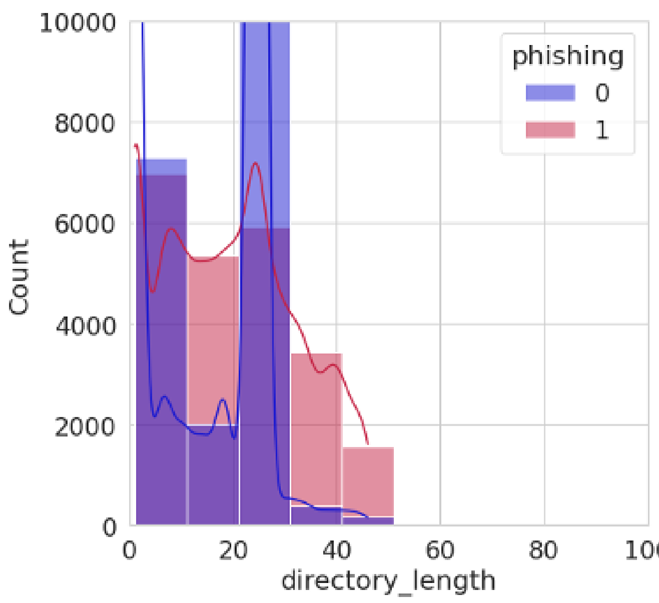


Fig. 14. Relation between phishing feature and directory_length

other models were trained on KNN imputed data. They are:

- **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 Random Forest is 85.6
- **KNeighbors Classifier**: The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification

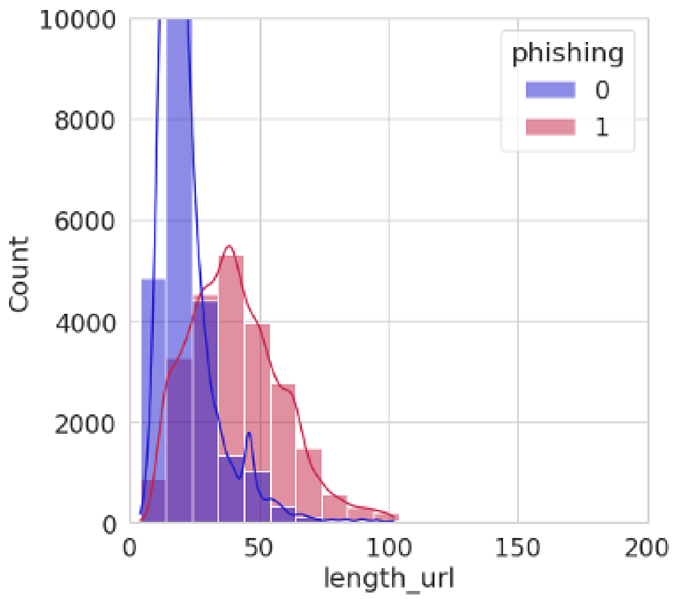


Fig. 15. Relation between the phishing feature and length_url

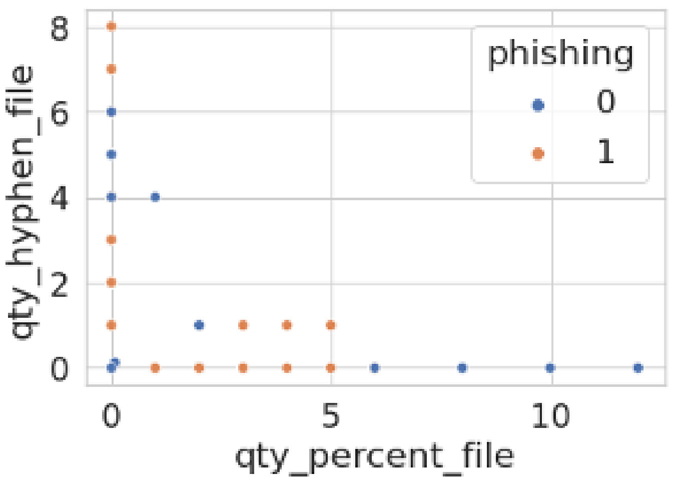


Fig. 16. Scatter Plot for qty_percent_file and qty_hyphen_file

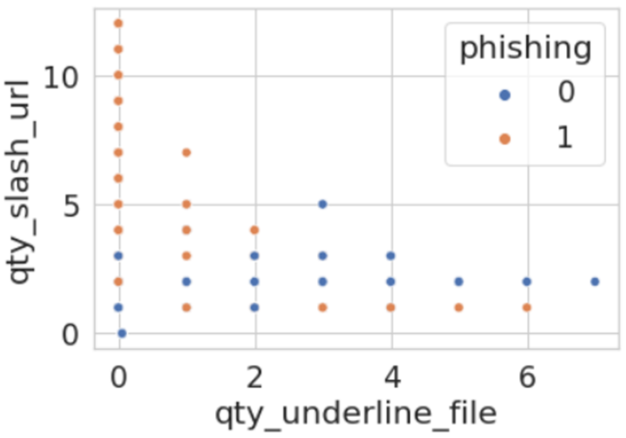


Fig. 17. Scatter Plot for qty_underline_file and qty_slash_url

and regression problems. The accuracy found through decision tree is 89.61

- **Decision Tree**: Decision tree is a predictive modelling approaches used in data mining. It is constructed through algorithmic approach that identifies differents methods of splitting a data set based on different conditions. The accuracy found through decision tree is 95.65
- **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 97.52
- **KNN**: KNN is a machine learning algorithm based on Supervised Learning technique. It assumes similarity between new data and available data and put new data into category that seems most similar to available categories. The accuracy found is 97.29
- It is concluded that the KNeighbors for KNN imputed dataset detected the best accuracy.

V. Comparisons

VI. Example Analysis

VII. Conclusions

VIII. References