National College of Ireland

Improving the detection method of phishing URLs through feature extraction and Machine Learning

MSc Research Project

Cyber Security

Martin Casey

Student ID: x20256604

School of Computing

National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

Project Submission Sheet – 2021/2022

Student Name:	Martin Casey
Student ID:	X20256604
Program:	MSc Cybersecurity
Year:	2022
Module:	Research Project
Lecturer:	Vikas Sahni
Submission Due Date:	4th September 2022
Project Title:	Improving the detection method of phishing URLs through feature extraction and Machine Learning
Word Count:	6744
Page count	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:

Date:

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. Projects should be submitted to your Program Coordinator.
3. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**
5. All projects must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Improving the detection method of phishing URLs through feature extraction and Machine Learning

Abstract

Attackers have been taking advantage of the increased use of the internet due to more people working remotely and accessing the internet from their homes, this increase in use of the internet is also due to the increase in number of electronic devices. Many of these people are also not sufficiently trained on detecting social engineering attacks. This research looks at how these emails can be identified by developing a framework in which phishing emails can be identified through the URL in the email. The goal of this study is to detect and mitigate against these attacks, where these types of attacks are expanding

The research carried out in this paper has shown different precision and accuracy scores for both different machine learning algorithms and for the number of features used in the algorithm. The best model from the results was the XGBoost algorithm this proved to be the most accurate model. The best result for each model was not necessarily the one using the most features from the dataset this can be clearly seen in the results.

1 Introduction

The internet has become the largest method of information exchange and largest communication method (Krombholz et al., 2015). This is more so the case since the Covid-19 pandemic with more and more people accessing the internet for work purposes (Shalke and Achary, 2022). With more flexibility employees are using their personal devices for work purposes both in the office and outside of the office environment (Krombholz et al., 2015). This means that companies are at risk from viruses and potential data leaks and social engineering attacks. The most common form of social engineering is phishing emails, where the attacker will send an email looking for sensitive information such as PPS number or credit card number or requesting the user to open a malicious URL or open a malicious attachment, both of which can lead to the user losing control of their machine to the attacker. Kevin Mitnick in the mid-nineties was one of the most wanted hackers and is best known for social engineering, this proves that social engineering is not a new problem and has existed for decades if not centuries. These attack techniques are described in the cyber kill chain where information gathering/reconnaissance is step one and exploitation is step three

Users that are not cyber aware can click on this malicious email and open them and fall for the intended scam. This results in data leakage or initial exploitation of the user's machine. A study carried out by the NCBI (National Centre for Biotechnology Information) where more than 2.9 million simulated emails are sent to employees at 6 hospitals with a median click rate of 16.7% (Gordon et al., 2019). This is a worrying statistic for all those involved in securing these networks and sensitive data. As mentioned above, opening and acting on these malicious emails can lead to systems being infiltrated by adversaries.

Users can be easily deceived by webpages where login information is requested, this is because that the phishing site is often a copy of the legitimate site and to the unsuspecting user, they see no difference and will proceed with entering their credentials or downloading some malicious software off the site

It is therefore incredibly important to have the appropriate measures to protect against phishing emails, some of these measures include having a spam filter that detects these emails, ensure users machines have the latest XDR (extended detection and response) solution and antivirus solution with the latest signatures installed, educating users on phishing emails is one of the most effective methods of reducing the click rate within an organisation and finally, what I propose here is another method of protecting against phishing emails where phishing emails are detected using the URL in the email to decipher if the message is a phishing email or not.

1.1 Research Question

1. **Question:** Can embedded URLs be used to classify whether an email is a phishing email or not.

Description: Many papers in the literature look specifically at the text used within emails to determine if they are phishing or not. In this paper the URL in emails will be isolated to aid in determining if the mail is phishing or not.

2. **Question:** What is the computational cost of implementing an effective method of detecting phishing emails?

Description: In this paper compute time will be analysed for each of the models. This is important for two main reasons. Firstly, the longer it takes to run the longer the delivery time of the mail increasing the likelihood that the model won't be used and secondly, the rising cost of computation, the longer it takes for the model the run the more expensive it is to run.

3. **Question:** What machine learning model is most appropriate for detecting phishing emails, where URLs are used as the classifier.

Description: In the paper some of the most popular Machine learning models will be used to determine which is the most accurate at detecting phishing emails.

4. **Question:** What is the effect on the number of features used in the machine learning model on the accuracy of the model.

Description: In this study the effect of varying the number of features will be analysed and the effect it has on the accuracy of the model. Having the number of features where the model is most accurate allows for faster computational time and therefore reduced cost.

2 Related Work & Background

2.1 Introduction

In this section a discussion of the previous work will be critically analysed and the gaps in the research will be identified. Some of these gaps were identified at an earlier stage when developing the research question. Here more depth into these gaps will be provided and identify failings in other papers and the merits of other papers that can be utilised in this study. Key definitions are also discussed at the beginning of this section. This is to give the reader some background, as these terms will be discussed throughout the paper.

2.2 Background

2.2.1 Objectives of Phishing emails

Attackers try to lure innocent computer users to give up sensitive information or downloading malicious software. Data loss can occur though users replying to the attacker or supplying the attacker with credentials. Phishing can be either targeted to a particular user this is known as spear phishing or where a phishing email can be sent to hundreds of users where the attacker hopes for a high click rate, these email addresses can be obtained from sites like hunter.io[1] where the attacker can get the format to an organisations email address and then look people up on LinkedIn and create their own distribution list.

2.2.2 Types of attacks

Typically, there are two different forms of phishing attacks they are

- Malware based – this is where an email has attached malicious software to the email and entices the recipient to download and install the software. This software then then takes advantage of vulnerabilities that exist on the machine and can then communicate

back to the attacking machine this allows the attacker to gain access to the machine and to the data that sits on the machine also.

- Deceptive email – this is where the attacker sends the recipient an email containing a URL. When the recipient clicks on the URL is looks like a legitimate site. For example, the attacker will clone a well-known site such as a banking site or Microsoft login page to lure the recipient to enter their credentials. These credentials are then harvested by the attacker and are used to sign in to steal data.

There are many ways in which an attacker can deliver these:

- Social engineering – this is where the convinced by the attacker using believable stories or creating a sense of urgency in the email,
- URL Hiding – this is where an attacker hides a URL in the email and the actual URL is behind this not visible to the non-tech savvy user.
- Cloning – this is where the attacker clones a website to look like the legitimate one, this may be used in harvesting credentials.
- Spoofing – where phishers mask their actual sending email address and make it look like they are sending from an official email address.

This paper focuses on phishing URLs where the recipient receives a phishing email, and the recipient is requested to click on a URL in the email. Accessing this URL can download malware on their machine, harvest credentials etc.

2.3 Previous Work

There are two main methods of detecting phishing emails, these methods are analysing the content of the target URL, that is the embedded URL in the phishing email. There is also another method of detection where the written content in the email is analysed (Verma et al., 2012). In the literature four of the main categories of phishing email detection they are:

- Machine learning techniques
- String patterns
- Information retrieval
- Blacklists

Before the introduction of these schemes integrations of blacklists into browsers was the preferred technique to block these URLs. A study carried out by (Ludl et al., 2007) analysed the effectiveness of Microsoft's and Google's blacklists and found that they are effective in blocking phishing URLs, however it has been shown that initially these blacklists do not give users protection and the effectiveness of the blacklists improve over a period of time (Ludl et al., 2007). This is also described as a limitation in (Nguyen et al., 2018) to manually find representative features and small datasets. Using blacklists in todays environment is not all that practical as they are hard to keep up to date due to the ease of spinning up new phishing sites by the attackers.

Work carried out by (Chandrasekar and Priyatharsini, 2018) uses characteristic within the email to determine if the mail is spam/phishing or not. These characteristics include subject line and keywords within the body of the mail. This filter however is quite simple in design where it only looks at the senders IP address, URLs.

Machine learning models are generally used by email filtering systems that function at the email level, this can be seen in multiple papers (Bergholz et al., 2010, 2008; Fette et al., 2007). This works by training a classifier on a set of extracted features from the set of emails. Once the training process of the model is complete the classifier is then utilized to detect phishing emails. These methods of detection differ in both the number and type of features used in the training process. The filters can either be installed on the client side or server side. These filters need to be updated on a regular basis; this is an important maintenance aspects of machine learning. It can be seen in (Vaitkevicius and Marcinkevicius, 2020) that the authors do a

comparison between different studies carried out and the various models and datasets used. This is useful information for this study as it will provide an insight to how the models presented in this study will perform

In (Shalke and Achary, 2022) uses a clustering strategy that gives the author a greater level of accuracy. This involved developing signatures similar to how anti-virus works where it develops signatures on malware, here in (Shalke and Achary, 2022) the authors develop signatures to detect phishing emails. Carrying out tokenization where the emails are split into single words or groups of words, this helps comparing them against generated signatures and also create new signatures. Similar to blacklists these signatures need to kept up to date.

In (Garera et al., 2007) the authors use a logistic regression model to classify phishing URLs. The paper uses features to classify the variables such as red flag or banned words in the URL and known features based on Google's page rank quality. In this paper it is hard to make a direct comparison as access to the URLs is not given. In (Shahrivari et al., 2020) the authors use machine learning techniques such as KNN (k-nearest neighbour), random forest, Adaboost and SVM (Support vector machine). Random forest proved to be the most accurate.

In (Almeida et al., 2011) looks at the increasing threat of web phishing attacks. The system in this paper looks at phishing detection using machine learning classifiers and wrapper feature selection strategy (Almeida et al., 2011).

A study carried out by (Kumar et al., 2015) describes that organisations can have the latest tools to prevent social engineering attacks on the users, these tools make the organisation less vulnerable to these types of attacks. However in (Kumar et al., 2015) describe that the best form of defence against these types of attacks is having an adequately trained staff in detecting phishing emails or social engineering attacks, ultimately even if a phishing email gets through the latest filters having an educated user recognising these mails means they won't click on malicious links or open malicious attachments.

Authors in (Chen and Guo, 2006; Garera et al., 2007) look at how models can used in the detection of phishing emails by carrying out analysis on the structure of the phishing URL. A logistic regression model is developed in (Garera et al., 2007), this model gave high accuracy in the detection.

Another approach taken to detect phishing emails can be seen by (Whittaker et al., n.d.), this research was carried out by a research team at Google. This paper focuses more on analysing the URL of the email and the content of the webpage. This model achieved an accuracy level of 90% in classifying phishing emails.

The authors in (Kulkarni and Brown, 2019) propose methods of identifying phishing emails. This was done through classifying the URLs through machine learning (ML) this looks at alerting on phishing emails by analysing the domain the email is sent from, this is done using WHOIS database. Certain features in results of a WHOIS lookup can used to tell if a phishing email is real or fake, for example when using WHOIS it may show a completely different domain name than that of the one in the phishing mail this would indicate the email is a phishing mail. Relying on WHOIS however for phishing detection may have its downfalls such as ensuring the WHOIS database is up to date and accurate.

Authors in (Kulkarni and Brown, 2019) use different methods to categorize websites. In this paper they use four classifiers they are decisions trees, Neural networks, Naïve Bayesian classifier and Support vector machine. For this study the Support vector machine came out as the best classifier. For the decision trees the issue of overfitting made the project unworkable, neural network did not work well for the dataset in this paper

The authors in (Nguyen et al., 2018) developed a framework with hierarchical long-short term memory (LSTM) and attention mechanisms to model the emails simultaneously at the word level of the email (Nguyen et al., 2018). It is expected to develop models for anti-phishing and prove how deep learning can be used for such problems in cybersecurity particularly in

detecting phishing emails. In this paper they look at F1-score and precision to determine the performance of the model in detecting phishing emails.
Researchers in (Khonji et al., 2012) suggest using lexical URL analysis approach. Lexical URL analysis (LUA) proven to have increased detection of phishing emails, all the time while reducing the number of false positives being detected. According to the authors the reason behind running feature subset selection methods on the feature sets is that unnecessary features can increase the time and space complexity of the classifiers and also reducing the accuracy of the classifiers. The authors use two datasets made up of harmless (4150) and phishing emails (4116). The proposed method was successful in increasing the detection rate and quality of identification.

2.4 Gap Analysis

From having read through the above literature as can be seen many methods have been chosen in detecting such as creating signatures to identify phishing emails in (Shalke and Achary, 2022). It is also possible to see in (Khonji et al., 2012) that they use lexical identifiers in the URL to identify phishing URLs, this is slightly different to the proposed study where in this paper a larger set of features both lexical and host based classifiers will be used in the training of the ML models. This should result in enhanced results and better detection.
Many ML models use all the features available in the dataset for the purposes of training the model. In this study an emphasis will be put on the effect the number of features used effects the model, for example is using 25 features better than using 40. Also, for the purposes of computational cost this is important, as there may be non-significant improvements in the model in letting it run through an additional 15 features but may add considerable time to the computational time.
Improving the detection method of phishing URLs through machine learning looking specifically at the number of features required to get the optimum scores is what is done in this research paper and compared against what is in the literature.

3 Research Methodology

In this section will look at data selection, data pre-processing, visualization, and feature extraction. In respect to model selection, this paper will mainly focus on machine learning focusing on different ML models such as random forest classifier (RFC), naïve bayes, decision tree classifier, XGBoost and logistic regression.

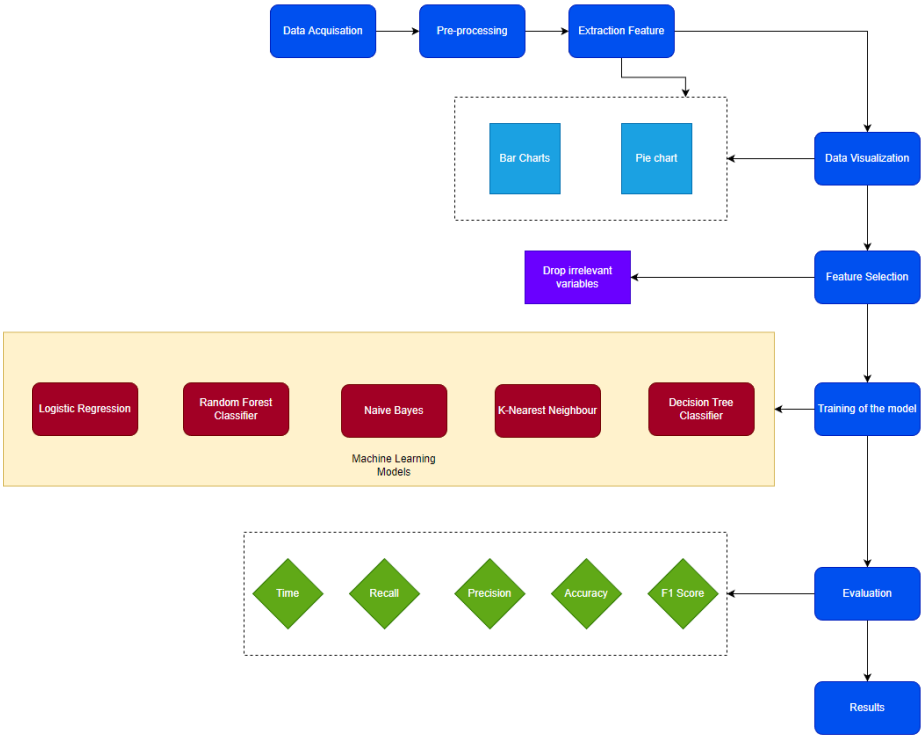

Figure 1 Methodology for detecting malicious emails through URL scanning

3.1 Data Selection

To be able to detect phishing emails it is necessary to obtain a large dataset of phishing emails for the purposes of training for ML. For this research a dataset was acquired from Kaggle. The dataset contains 10,000 email URLs which are labelled as non-phishing or phishing URLs, this is represented in the data as a 0 or a 1 respectively. The dataset contains other data that can contribute to whether it is a phishing URL or not, such as number of characters in the URL, number of dashes in the URL, whether the site has an SSL cert or not etc. The dataset is found to be balanced in terms of there is equal number of phishing URLs as non-phishing URLs. This is useful as there is no need to manipulate the data to remove any imbalance.

3.2 Pre-processing

Data pre-processing is an important step in any analysis of data. It involves looking at the data to identify null values in the data. It was found in this dataset that there is no null values in the dataset as seen in Figure 3. The data was also balanced in terms of number of phishing emails and the number of non-phishing emails. The fact that the dataset is balanced is very convenient as Symmetric Minority Oversampling Technique is not required. Also, as part of the pre-processing it was important to remove any irrelevant variables from the dataset.

Also, as part of the pre-processing the mutual information classifier is calculated. This classifier is used to identify linear and non-linear correlations between the features and the labels. Below in Figure 2 the MI scores can be seen for each of the features in the dataset, plotted in descending order. This MI classifier is later used when looping through the features to find the most accurate number of features to run for a particular model.

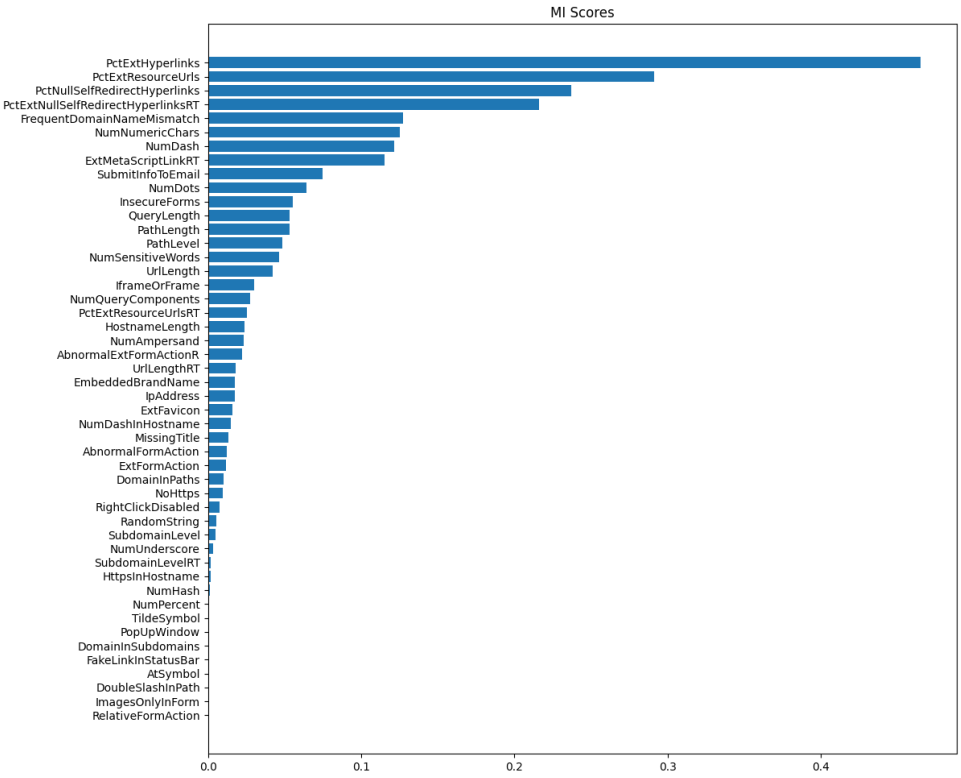

Figure 2 MI scores

3.3 Feature Extraction

The obtained dataset contains multiple features. Within the dataset it is primarily made up of lexical features. These features include URL length, number of sensitive word in the email etc. A list of the features can be seen below in Figure 3. As can be seen from the feature extraction they are all type 32, these were converted from type 64. Doing this allows to save on memory usage and the data can be prepared using cUML later for the purposes of training.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 50 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              10000 non-null  int32
 1   NumDots                         10000 non-null  int32
 2   SubdomainLevel                  10000 non-null  int32
 3   PathLevel                       10000 non-null  int32
 4   UrlLength                       10000 non-null  int32
 5   NumDash                         10000 non-null  int32
 6   NumDashInHostname               10000 non-null  int32
 7   AtSymbol                        10000 non-null  int32
 8   TildeSymbol                     10000 non-null  int32
 9   NumUnderscore                   10000 non-null  int32
 10  NumPercent                      10000 non-null  int32
 11  NumQueryComponents              10000 non-null  int32
 12  NumAmpersand                    10000 non-null  int32
 13  NumHash                         10000 non-null  int32
 14  NumNumericChars                 10000 non-null  int32
 15  NoHttps                         10000 non-null  int32
 16  RandomString                    10000 non-null  int32
 17  IpAddress                       10000 non-null  int32
 18  DomainInSubdomains              10000 non-null  int32
 19  DomainInPaths                   10000 non-null  int32
 20  HttpsInHostname                 10000 non-null  int32
 21  HostnameLength                  10000 non-null  int32
 22  PathLength                      10000 non-null  int32
 23  QueryLength                     10000 non-null  int32
 24  DoubleSlashInPath               10000 non-null  int32
 25  NumSensitiveWords               10000 non-null  int32
 26  EmbeddedBrandName               10000 non-null  int32
 27  PctExtHyperlinks                10000 non-null  float32
 28  PctExtResourceUrls              10000 non-null  float32
 29  ExtFavicon                      10000 non-null  int32
 30  InsecureForms                   10000 non-null  int32
 31  RelativeFormAction              10000 non-null  int32
 32  ExtFormAction                   10000 non-null  int32
 33  AbnormalFormAction              10000 non-null  int32
 34  PctNullSelfRedirectHyperlinks   10000 non-null  float32
 35  FrequentDomainNameMismatch      10000 non-null  int32
 36  FakeLinkInStatusBar             10000 non-null  int32
 37  RightClickDisabled              10000 non-null  int32
 38  PopUpWindow                     10000 non-null  int32
 39  SubmitInfoToEmail               10000 non-null  int32
 40  IframeOrFrame                   10000 non-null  int32
 41  MissingTitle                    10000 non-null  int32
 42  ImagesOnlyInForm                10000 non-null  int32
 43  SubdomainLevelRT                10000 non-null  int32
 44  UrlLengthRT                     10000 non-null  int32
 45  PctExtResourceUrlsRT            10000 non-null  int32
 46  AbnormalExtFormActionR          10000 non-null  int32
 47  ExtMetaScriptLinkRT             10000 non-null  int32
 48  PctExtNullSelfRedirectHyperlinksRT  10000 non-null  int32
 49  CLASS_LABEL                     10000 non-null  int32
dtypes: float32(3), int32(47)
memory usage: 1.9 MB
```

Figure 3 Feature extraction from the dataset

3.4 Data Visualization

In this section the data will be visualized, this will give a greater insight into the data. Below it can be seen that various features have been visualised. Of course, there are more features that could have been visualized but, in this section, it was decided to just look at the important features in the dataset. In this section pie charts and stack charts are used to visualise the data.

With respect to the use of SSL certificates, it can be seen in Figure 4 and Figure 5 that just under 99% of emails had no SSL certificate. Those emails that had URLs with no SSL certificate are split almost in half between phishing and non-phishing. However, for the 1% of emails that have an SSL certificate it is much more likely that they are phishing emails, as can be seen from the pie chart on the left in Figure 4.

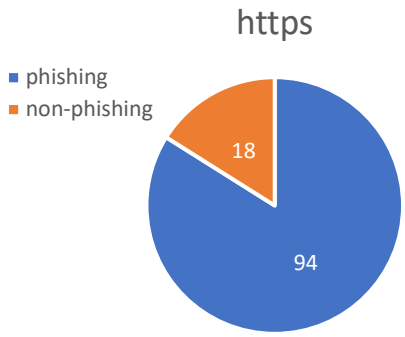

Figure 4 Shows distribution of phishing and non-phishing mails for HTTPS websites

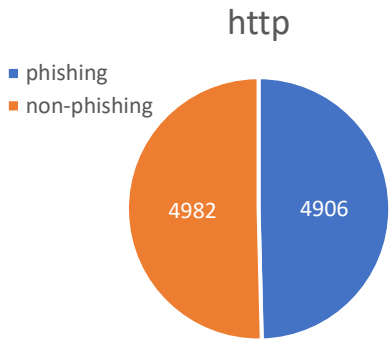

Figure 5 Shows distribution of phishing and non-phishing mails for HTTP websites

It can be seen from Figure 6 and Figure 7 when there is an IP address in the URL it is 100% a phishing email. However, this is a small proportionate of the data at just over 1.5% of all phishing emails had a URL with an IP address.

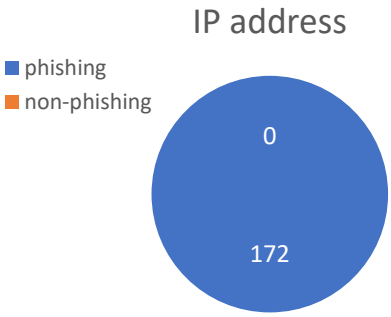

Figure 6 Shows distribution of phishing and non-phishing where there is an IP address in the URL

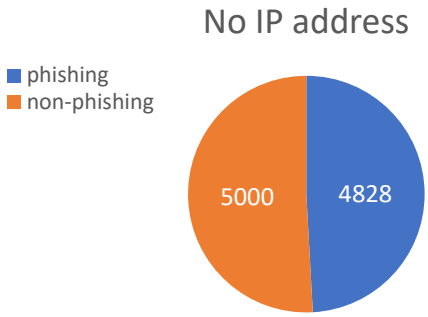

Figure 7 Shows distribution of phishing and non-phishing where there is no IP address in the URL

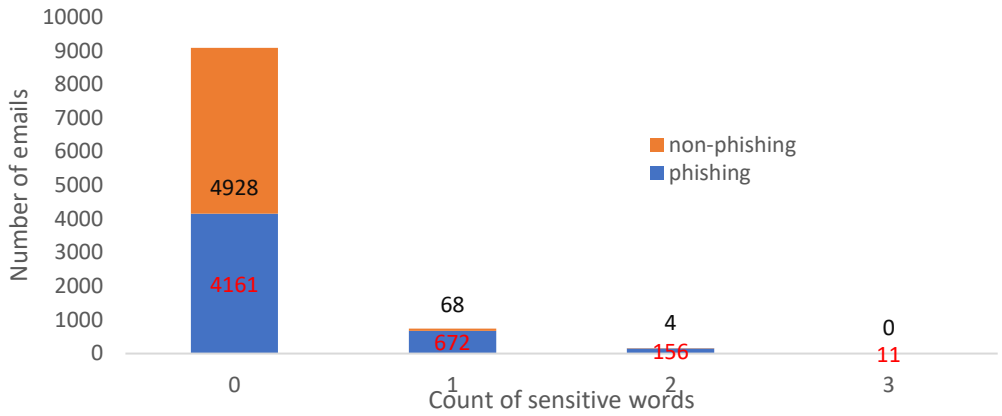

Figure 8 Shows the number of sensitive words in a URL and if they are phishing or non-phishing emails

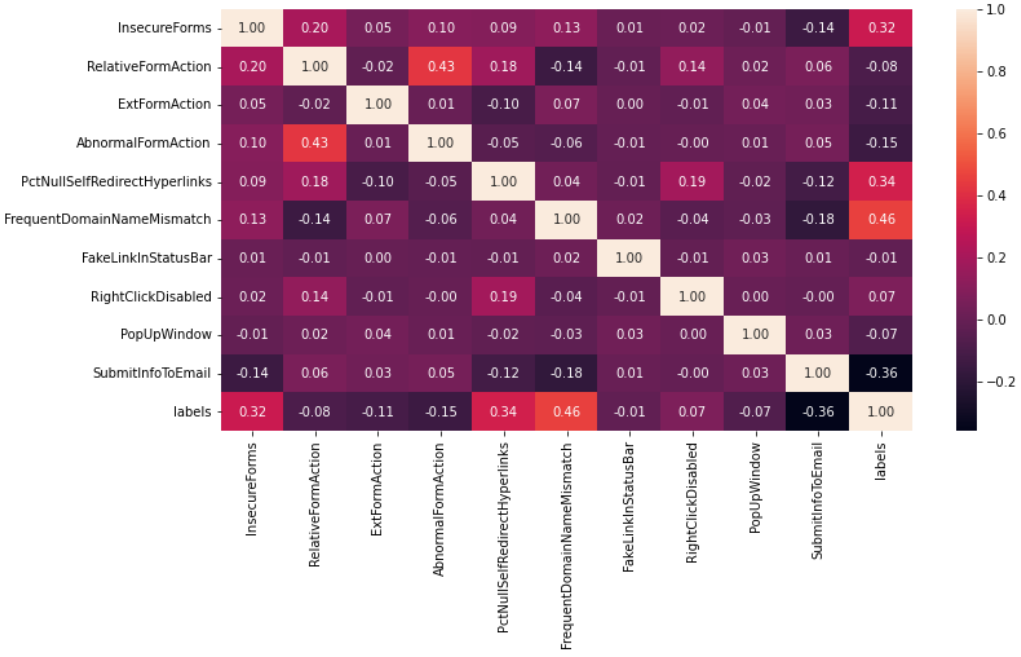

Figure 9 Correlation between columns 30 to 40 and the labels

In columns 30 to 40 in Figure 9 it can seen that there are some correlations as described below:

- Percent null self-redirect hyperlinks shows the same correlation as insecure forms
- Frequent domain name mismatch shows that it has a medium correlation
- Submit info to email indicates that sites that ask the user to submit information seems to be more probable to be a phishing email.
- Insecure Forms shows that the higher the value the more probable it is a phishing email.

4 Machine Learning Models

4.1 Detection Accuracy

To evaluate the accuracy of the model an approach needs to be developed to measure true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Explanation of each is given below.

For each of the machine learning models Accuracy, Precision, Recall Score, F1 Score and Time were calculated.

4.2 Logistic regression

Logistic regression is a statistical model often used for predictive analysis. Logistic regression estimates the probability of an event occurring or not. Since the outcome of logistic regression is a probability the dependant variable is restricted between 0 and 1 ("What is Logistic regression?," n.d.). Logistic regression is also known as the log odds or natural logarithm of odds, this is the probability of success divided by the probability of failure and is represented in the below formulas:

$$logit(\eta) = \frac{1}{1 + \exp(\eta)} \tag{5}$$

$$logit(\eta) = log\left(\frac{P(y=1)}{1-P(y=1)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \cdots \ldots + \beta_p X_p \tag{6}$$

Below demonstrates the effect of logistic regression versus linear regression. As can be seen from Figure 10 the fit is much better for a logistic regression model.

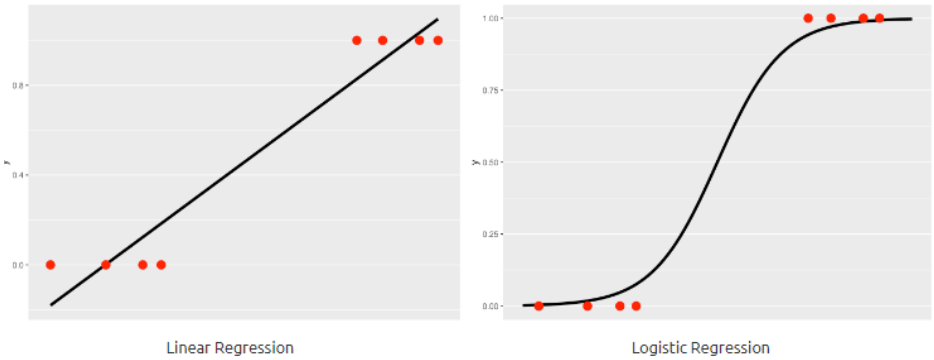

Figure 10 Shows the fit for logistic regression is much better as opposed to linear regression

4.3 Naïve Bayes

Naïve Bayes methods are a set of supervised learning algorithms based on applying Bayes theorem (seen in formula 3) with naïve assumption of conditional independence between every pair of features given the value class. Naïve Bayes are very efficient in terms of speed compared to more advanced methods.

$$P(A|B) = \frac{P(B|A)\ x\ P(A)}{P(B)} \tag{7}$$

For a given dataset $X = \{x_1, x_2, x_3, x_4, x_5, \ldots \ldots . x_n\}$ $\{y\}$ it can therefore be said using Bayes theorem:

$$P(y|x_1, x_2, x_3, x_4, x_5, \ldots \ldots . x_n) = \frac{P(x_1|y)P(x_2|y)P(x_3|y)P(x_4|y) \ldots . P(x_n|y)\ x\ P(y)}{P(x_1)P(x_2)P(x_3) \ldots . P(x_n)} \tag{8}$$

This can then be rewritten as

$$P(y|x_1, x_2, x_3, x_4, x_5, \ldots \ldots . x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1)P(x_2)P(x_3) \ldots . P(x_n)} \tag{9}$$

Since $P(x_1, x_2, x_3, x_4, x_5, \ldots \ldots . x_n)$ is constant given the input the below can be used:

$$P(y|x_1, x_2, x_3, x_4, x_5, \ldots\ldots x_n)\ \alpha\ P(y)\prod_{i=1}^{n} P(x_i|y) \tag{10}$$

$$\hat{y} = \arg max\ P(y)\prod_{i=1}^{n} P(x_i|y) \tag{11}$$

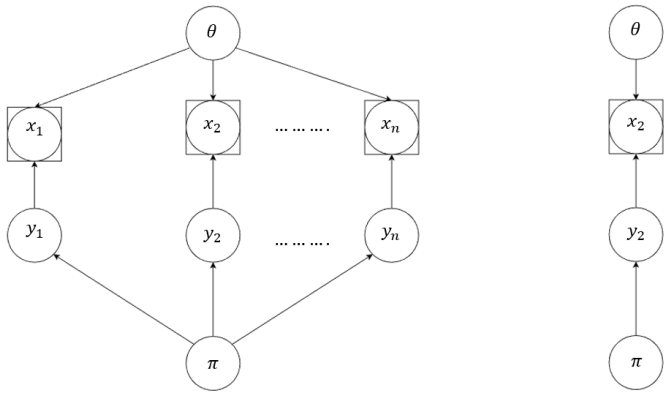

Figure 11 Shows the Naive Bayes train (left) and test (right) model (Smolyakov, 2018)

4.4 Random Forest Classifier (RFC)

Random forest classifier (RFC) is a process of classification, regression and other tasks that construct decision trees at training time. Random forest is primarily based on the idea of ensemble learning, this is where several classifiers are utilised to resolve a complicated problem and increase the performance of the model. Random forest comprises of decision tree models on different subsets of the data. Datapoints are then passed through each tree one by one and note down the predictions. The predictions are combined, as it's a classification problem the majority voting will be taken. Below in Figure 12 it can seen, an example of a random forest classifier.

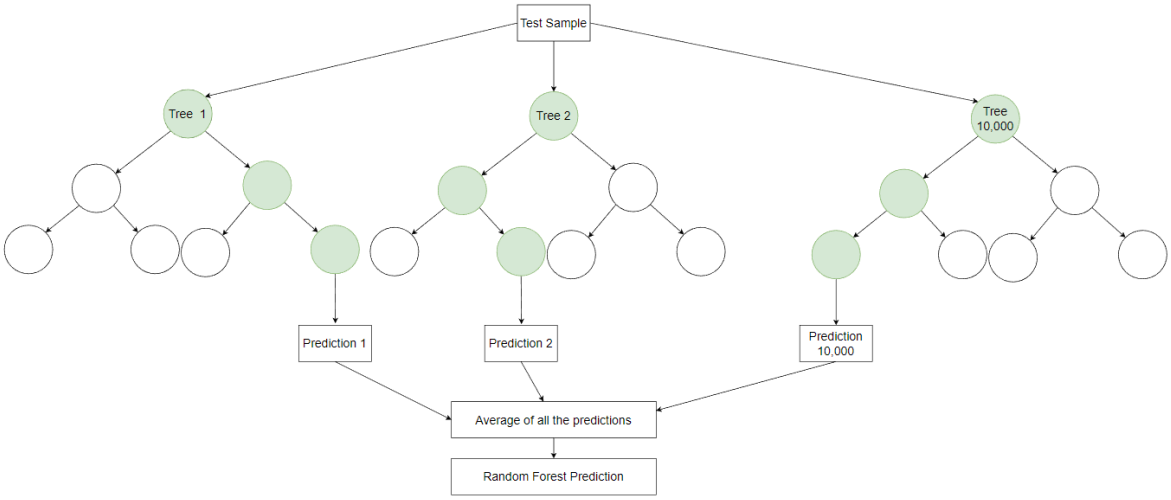

Figure 12 Random Forest Classifier model

4.5 Decision Tree Classifier

Decision tree classifier is a supervised machine learning algorithm (Bento, 2021) and is uses a set of rules in order to make decisions. Decision tree can both perform classification tasks and regression tasks, this is also referred to as CART (Classification and Regression Tree) (Bento, 2021). Decision trees work on the basis that a dataset with features are used to create yes or no questions to continuously split the data, this gives the tree structure.

When a question is asked this creates a node the first question being the root node. Asking additional questions creates additional nodes, and when it is decided to stop splitting the data the last node is called the leaft node (Bento, 2021).

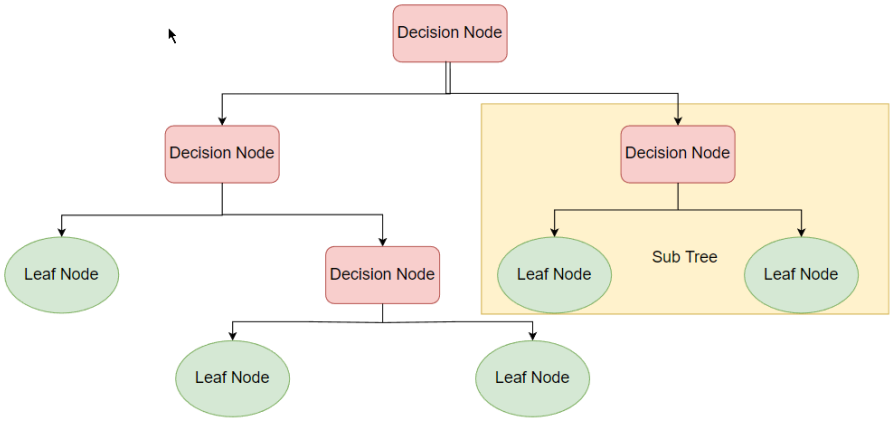

Figure 13 Decision Tree Classifier ("Python Decision Tree Classification Tutorial," n.d.)

4.6 Extreme Gradient Boosting (XGBoost)

XGBoost is the leading machine learning library for classification ranking and regression problems, it provides parallel tree boosting ("What is XGBoost?," n.d.). XGBoost has many benefits listed below:

- There is large contributions being made to XGBoost open source library
- Has a wide range of applications for example, classification ranking, regression.
- Runs on OS X, Windows and Linux
- Cloud integration AWS, Azure and Yarn clusters

In this study XGBoost algorithm was run with a GPU (graphics processing unit) this significantly increases the accuracy and reduces the run time compared to a standard CPU. RAPIDS uses optimized NVIDIA CUDA primitives and high-bandwidth GPU memory to accelerate data preparation and machine learning ("Getting Started," n.d.)

4.7 K-Nearest Neighbour Classifier

KNN or K-nearest neighbour is a non parametric learning uses proximity to make classification or predictions about the grouping of an individual data point and is typically used in classification algorithms.

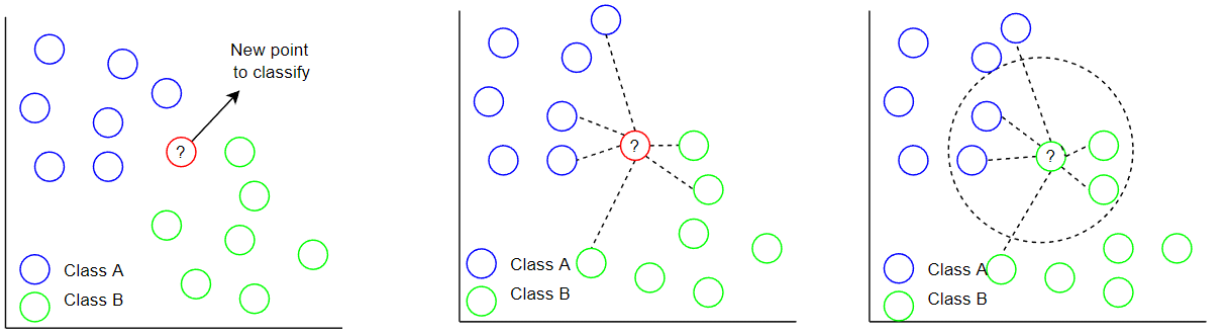

Figure 14 KNN diagram ("What is the k-nearest neighbours algorithm?," n.d.)

When using the KNN Classifier we need to calculate the value for k that is how many nearest neighbour we will consider or the distance. This distance can be calculated using one of two methods, Euclidean distance or Manhattan distance. These formulas are represented below:

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2} \quad (12)$$

$$d(x,y) = \sum_{i=1}^{m}|x_i - y_i| \quad (13)$$

Euclidean formula is limited to real valued vectors and is the most commonly used distance measure. Manhattan distance on the other hand measures the absolute distance between two points.

5 Implementation

As mentioned previously the primary goal of this paper is to investigate the machine learning models to classify emails as phishing or non-phishing based on the URL found in the body of an email. New URLs are created each day, it is therefore important to be able to extract this information as efficiently as possible.

Various libraries were used throughout the analysis of the data and the implementation of the ML algorithms.

The models were implemented on Google Colab. Colab allows the user to write Python code with little to no configuration. It also gives the user access to GPUs for running machine learning models.

6 Evaluation

In this section the results from the implemented models will be displayed. The accuracy of the model along with the recall, F1 score and precision will be shown for each model. 5 models are analysed in this section and are listed below:

- Logistic regression
- Random Forest Classifier (RFC)
- Naïve Bayes
- K-Nearest Neighbour
- Decision Tree Classifier

In the summary section an evaluation and comparison of the models will be made. Compute time is also calculated which can dictate which model is used, a long compute time can in this case lead to a long time in the mail being delivered resulting in a poor service.

6.1 Logistic Regression

In this experiment, focus will be on the performance of a logistic regression model in respect to detecting phishing emails that have a malicious URL in them. Recall, F1 Score, Precision and Accuracy were calculated for the dataset. Also, as part of this study, the effect the number of features has on these final scores will also be analysed. The key to this is therefore picking the best number of features that result in the highest of scores across the board. It can seen from Figure 15 that the recall performs consistently well. However, there are dips in the other KPIs (key performance indicators). An area where all the metrics are performing well is around 47 features. The model took 91 seconds to run (wall time) with a CPU time of 164 seconds.

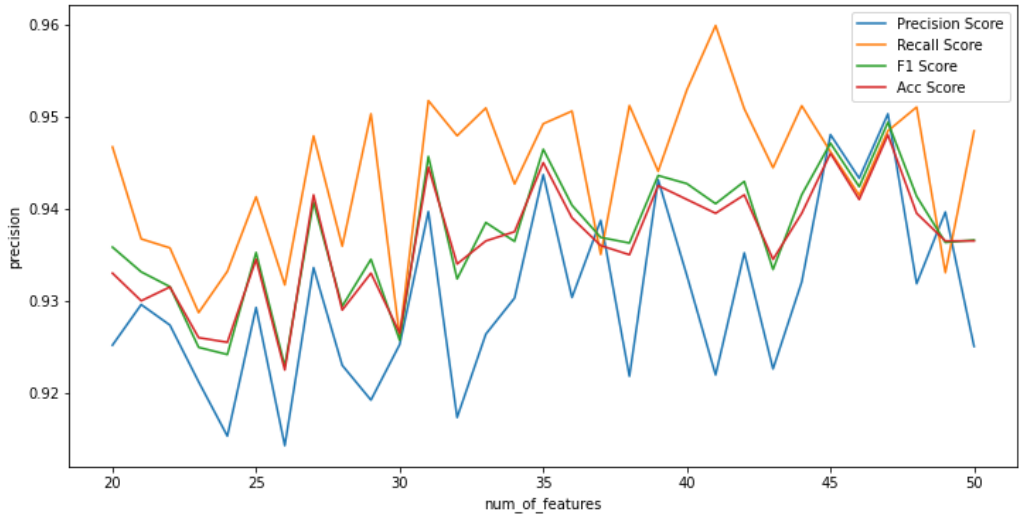

Figure 15 Performance of the logistic regression model as the number of features in the model increase.

6.2 Random Forest Classifier

In this experiment, focus will be on the performance of a Random Forrest Classifier model in respect to detecting phishing emails that have a malicious URL in them. Recall, F1 Score, Precision and Accuracy were calculated for the dataset. It can seen from Figure 16 that the recall performs consistently well. However, there are dips in the other KPIs. An area where all the metrics are performing well is around 27 features. The model took 125 seconds to run (wall time) with a CPU time of 178 seconds.

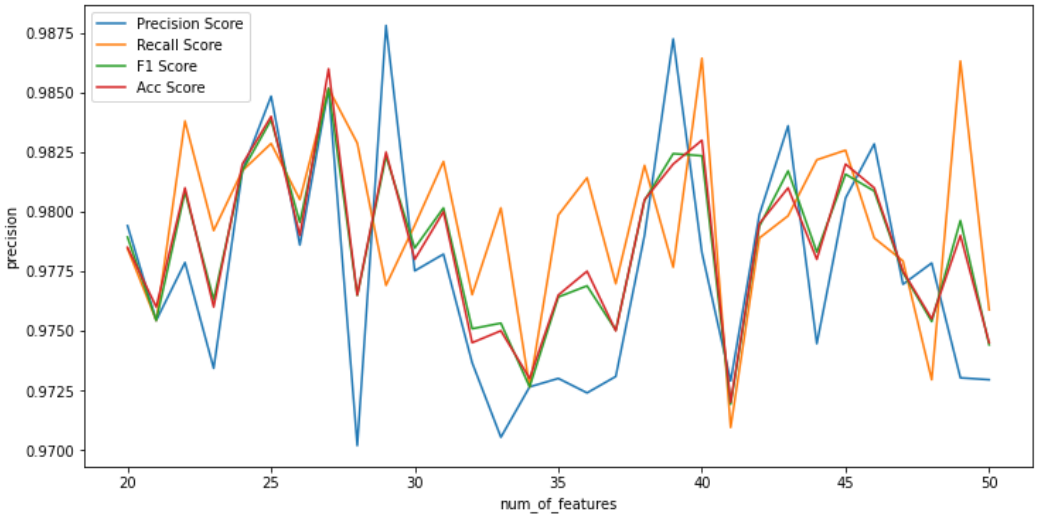

Figure 16 Performance of the Random Forrest Classifier model as the number of features in the model increase.

6.3 Naïve Bayes

In this experiment, focus will be on the performance of a Naïve Bayes model in respect to detecting phishing emails that have a malicious URL in them. Recall, F1 Score, Precision and Accuracy were calculated for the dataset. It can seen from Figure 17 that the recall performs consistently well. However, there are dips in the other KPIs. An area where all the metrics are performing well is around 27 features. The model took 0.543 seconds to run (wall time) with a CPU time of 0.553 seconds.

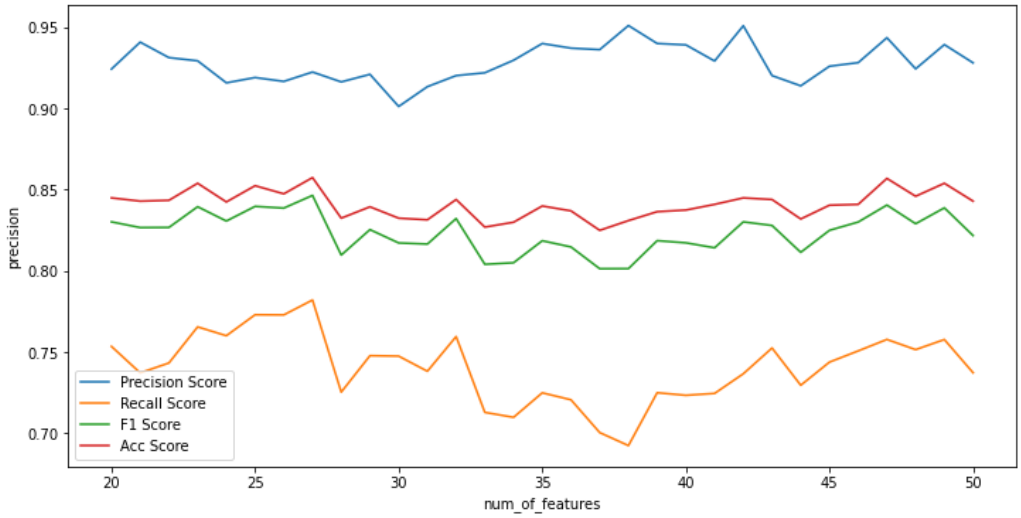

Figure 17 Performance of the Naïve Bayes model as the number of features in the model increase.

6.4 K-Nearest Neighbour Classifier (KNN)

In this experiment, focus will be on the performance of a K-Nearest Neighbour model in respect to detecting phishing emails that have a malicious URL in them. Recall, F1 Score, Precision and Accuracy were calculated for the dataset. It can seen from Figure 18 that the recall performs consistently well. However, there are dips in the other KPIs. An area where all the metrics are performing well is around 48 features. The model took 14.1 seconds to run (wall time) with a CPU time of 18.5 seconds.

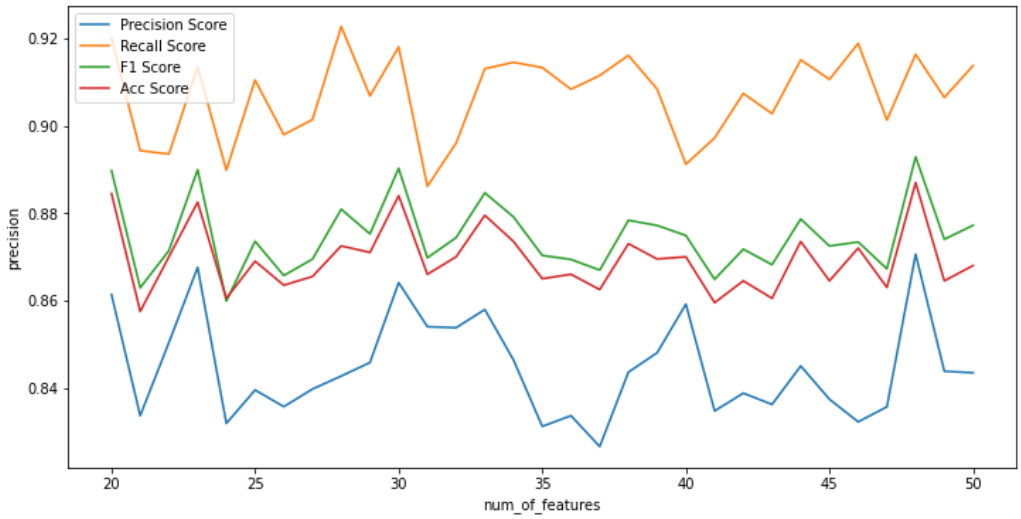

Figure 18 Performance of the K-Nearest Neighbour Classifier model as the number of features in the model increase.

6.5 Decision Tree Classifier

In this experiment, focus will be on the performance of a Decision Tree model in respect to detecting phishing emails that have a malicious URL in them. Recall, F1 Score, Precision and Accuracy were calculated for the dataset. It can seen from Figure 17 that the recall performs consistently well. However, there are dips in the other KPIs. An area where all the metrics are performing well is around 41 features. The model took 2.1 seconds to run (wall time) with a CPU time of 2.09 seconds.

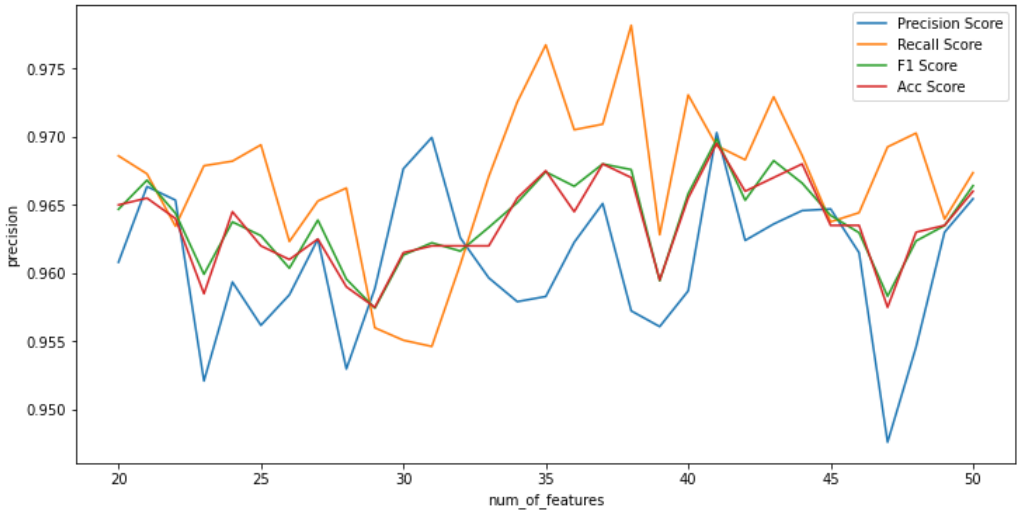

Figure 19 Performance of the Decision Tree Classifier model as the number of features in the model increase.

6.6 XGBoost

In this experiment, focus will be on the performance of a XGBoost model in respect to detecting phishing emails that have a malicious URL in them. Recall, F1 Score, Precision and Accuracy were calculated for the dataset. It can seen from Figure 20 that the recall performs consistently well. However, there are dips in the other KPIs. An area where all the metrics are performing well is around 41 features. The model took 48.3 seconds to run (wall time) with a CPU time of 88 seconds.

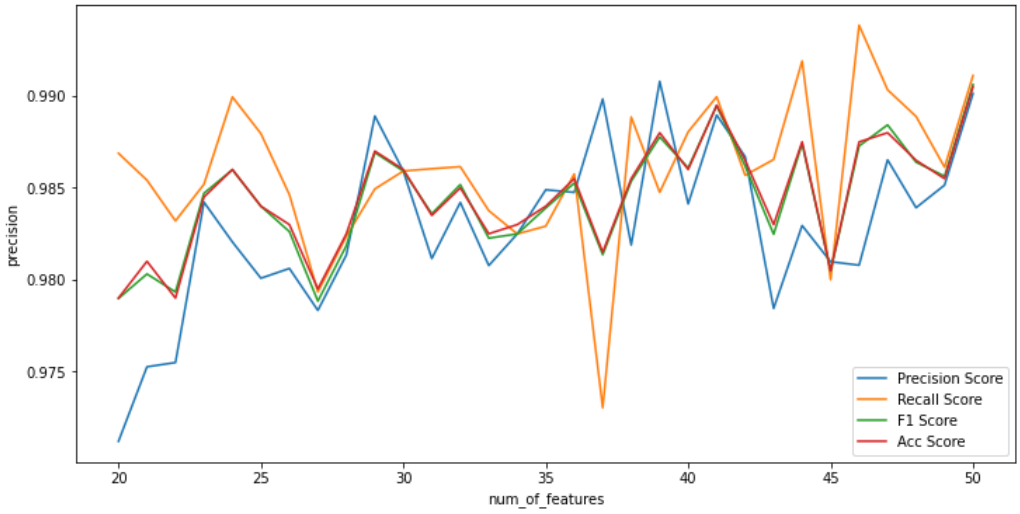

Figure 20 Performance of the XGBoost Classifier model as the number of features in the model increase.

6.7 Summary and Discussion

In this section all the results will be summarized and analysed to see what the best performing model was in detecting phishing URLs.

Table 1 Summary of all models

Model	Precision Score	Recall Score	F1 Score	Accuracy Score	CPU Time (s)	Wall Time (s)	Optimal number of features
Logistic Regression	95.00%	94.80%	94.90%	94.80%	164	91	47
Naïve Bayes	92.20%	78.20%	84.60%	85.80%	0.53	0.54	27
Random Forrest Classifier	98.50%	98.50%	98.50%	98.60%	178	125	27
K-Nearest Neighbour Classifier	87.60%	91.60%	87.30%	88.70%	18.5	14.1	48
Decision Tree Classifier	97.00%	96.90%	96.90%	96.90%	2.1	2.1	41
XGBoost Classifier	98.80%	98.90%	98.90%	98.90%	88	48.3	41

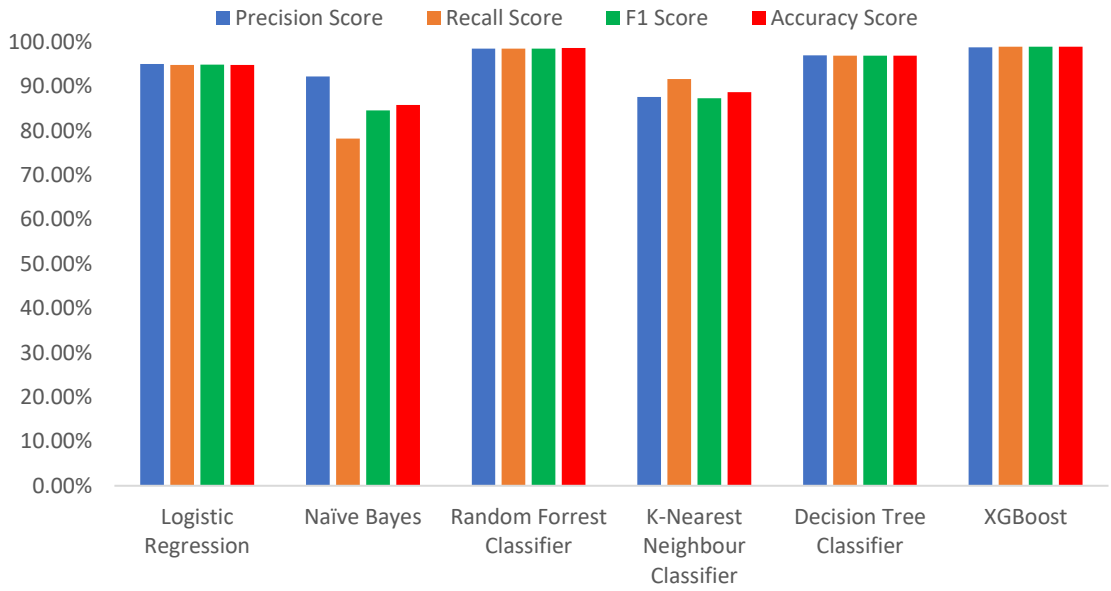

Figure 21 Shows the performance of each of the models

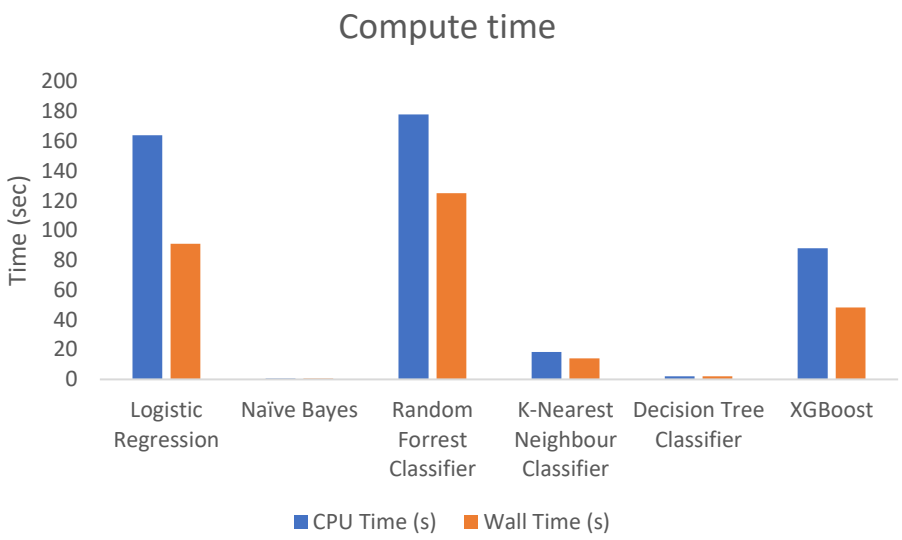

Figure 22 Shows the CPU time and Wall time for the different models

It can be seen from the graph in Figure 21 Shows the performance of each of the models Figure 21 that the XGBoost algorithm is the best performing model, it is however the third slowest of the models to run. This compute time can be seen in Figure 22 where random forest classifier is the longest to run with Naïve Bayes the quickest at just 0.5 seconds for both the CPU time and the wall time. Compute time is incredibly important as this will determine the delivery time of the email.

Looking at (Vaitkevicius and Marcinkevicius, 2020) that XGBoost is one of the more accurate algorithms, this is backed up in this paper. It can also be seen from (Vaitkevicius and Marcinkevicius, 2020) that Naïve Bayes doesn't perform as good as the others with small datasets (<10,000) this is also visible in this paper where Naïve Bayes is the least accurate.

We can see from Table 1 that K-nearest neighbour is the worst performing model and the only model where recall is larger than the precision. This can be dangerous in terms of detecting phishing URLs as when the recall is larger it means the false negative is larger than the true negative, meaning that there are phishing URLs not being detected by the model.

7 Conclusions & Future Work

Firstly, we can see that embedded URLs in phishing emails can be used to classify emails if they are phishing related or not. From the above it can be seen using feature extraction from these URLs machine learning algorithms can be utilised to carry out this detection with high precision scores from some of the models.

The computational cost is relatively small. In the experiments run in this paper a relatively low spec machine was used. The CPU was an Intel(R) Xeon(R) CPU @ 2.20GHz with 14GB of RAM. Improving the spec of this machine it would be expected to see faster computational time and also in these experiments a NVIDIA Tesla T4 graphics card was utilized, again if a superior graphics card was used better results could be expected. However, the results in this paper show that the best equipment is not necessary to obtain good results. This is more important than ever with high demand on graphics cards and processors in mining crypto currency for example. This would be one area for future work where the equipment used in the experiments are of a higher spec.

It can be seen from the results that the XGBoost algorithm is the preferred model for predicting phishing URLs. This model gives the greatest accuracy of the 5 models chosen and also the computational time is not considered high by todays email filtering times.

It can be seen from this research that there is a significant impact on the number of features used to determine if an email is phishing or not. This can be seen in the logistic regression model as an example

where the precision score ranges from 91.5% to 95% for different number of features used. It is also important to note that the most accurate model is not necessarily using all the features in the dataset. An area for further work would be to implement the best performing model into an application to be able to use this in a corporate environment potentially.

8 Video Presentation

https://www.youtube.com/watch?v=3QJiok8SVic

9 References

Almeida, T.A., Almeida, J., Yamakami, A., 2011. Spam filtering: how the dimensionality reduction affects the accuracy of Naive Bayes classifiers. J Internet Serv Appl 1, 183–200. https://doi.org/10.1007/s13174-010-0014-7

Bento, C., 2021. Decision Tree Classifier explained in real-life: picking a vacation destination [WWW Document]. Medium. URL https://towardsdatascience.com/decision-tree-classifier-explained-in-real-life-picking-a-vacation-destination-6226b2b60575 (accessed 8.4.22).

Bergholz, A., Chang, J.H., Paass, G., Reichartz, F., Strobel, S., 2008. Improved Phishing Detection using Model-Based Features.

Bergholz, A., De Beer, J., Glahn, S., Moens, M.-F., Paaß, G., Strobel, S., 2010. New filtering approaches for phishing email. Journal of Computer Security 18, 7–35. https://doi.org/10.3233/JCS-2010-0371

Chandrasekar, Dr.C., Priyatharsini, P., 2018. CLASSIFICATION TECHNIQUES USING SPAM FILTERING EMAIL. ijarcs 9, 402–410. https://doi.org/10.26483/ijarcs.v9i2.5571

Chen, J., Guo, C., 2006. Online Detection and Prevention of Phishing Attacks, in: 2006 First International Conference on Communications and Networking in China. Presented at the 2006 First International Conference on Communications and Networking in China, pp. 1–7. https://doi.org/10.1109/CHINACOM.2006.344718

Cyber Kill Chain® [WWW Document], 2022. . Lockheed Martin. URL https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html (accessed 7.30.22).

Fette, I., Sadeh, N., Tomasic, A., 2007. Learning to detect phishing emails, in: Proceedings of the 16th International Conference on World Wide Web, WWW '07. Association for Computing Machinery, New York, NY, USA, pp. 649–656. https://doi.org/10.1145/1242572.1242660

Garera, S., Provos, N., Chew, M., Rubin, A.D., 2007. A framework for detection and measurement of phishing attacks, in: Proceedings of the 2007 ACM Workshop on Recurring Malcode, WORM '07. Association for Computing Machinery, New York, NY, USA, pp. 1–8. https://doi.org/10.1145/1314389.1314391

Getting Started [WWW Document], n.d. . RAPIDS. URL https://rapids.ai/start.html (accessed 8.10.22).

Gordon, W.J., Wright, A., Aiyagari, R., Corbo, L., Glynn, R.J., Kadakia, J., Kufahl, J., Mazzone, C., Noga, J., Parkulo, M., Sanford, B., Scheib, P., Landman, A.B., 2019. Assessment of Employee Susceptibility to Phishing Attacks at US Health Care Institutions. JAMA Netw Open 2, e190393. https://doi.org/10.1001/jamanetworkopen.2019.0393

Khonji, M., Iraqi, Y., Jones, A., 2012. Enhancing Phishing E-Mail Classifiers: A Lexical URL Analysis Approach. International Journal for Information Security Research 2, 236–245. https://doi.org/10.20533/ijisr.2042.4639.2013.0029

Krombholz, K., Hobel, H., Huber, M., Weippl, E., 2015. Advanced social engineering attacks. Journal of Information Security and Applications, Special Issue on Security of Information and Networks 22, 113–122. https://doi.org/10.1016/j.jisa.2014.09.005

Kulkarni, A.D., Brown, L.L., 2019. Phishing Websites Detection using Machine Learning. International Journal of Advanced Computer Science and Applications 10, 7.

Kumar, A., Chaudhary, M., Kumar, N., 2015. Social Engineering Threats and Awareness: A Survey.

Ludl, C., McAllister, S., Kirda, E., Kruegel, C., 2007. On the Effectiveness of Techniques to Detect Phishing Sites, in: M. Hämmerli, B., Sommer, R. (Eds.), Detection of Intrusions and Malware, and Vulnerability Assessment, Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp. 20–39. https://doi.org/10.1007/978-3-540-73614-1_2

Nguyen, M., Nguyen, T., Nguyen, T.H., 2018. A Deep Learning Model with Hierarchical LSTMs and Supervised Attention for Anti-Phishing. https://doi.org/10.48550/arXiv.1805.01554

Python Decision Tree Classification Tutorial: Scikit-Learn DecisionTreeClassifier [WWW Document], n.d. URL https://www.datacamp.com/tutorial/decision-tree-classification-python (accessed 8.4.22).

Shahrivari, V., Darabi, M.M., Izadi, M., 2020. Phishing Detection Using Machine Learning Techniques.

Shalke, C.J., Achary, R., 2022. Social Engineering Attack and Scam Detection using Advanced Natural Langugae Processing Algorithm, in: 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI). Presented at the 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI), pp. 1749–1754. https://doi.org/10.1109/ICOEI53556.2022.9776697

Smolyakov, V., 2018. Information Planning and Naive Bayes [WWW Document]. Medium. URL https://towardsdatascience.com/information-planning-and-naive-bayes-380ee1feedc7 (accessed 7.30.22).

Vaitkevicius, P., Marcinkevicius, V., 2020. Comparison of Classification Algorithms for Detection of Phishing Websites. Informatica 31, 143–160. https://doi.org/10.15388/20-INFOR404

Verma, R., Shashidhar, N., Hossain, N., 2012. Detecting Phishing Emails the Natural Language Way, in: Foresti, S., Yung, M., Martinelli, F. (Eds.), Computer Security – ESORICS 2012, Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 824–841. https://doi.org/10.1007/978-3-642-33167-1_47

What is Logistic regression? | IBM [WWW Document], n.d. URL https://www.ibm.com/topics/logistic-regression (accessed 8.31.22).

What is the k-nearest neighbors algorithm? | IBM [WWW Document], n.d. URL https://www.ibm.com/topics/knn (accessed 8.3.22).

What is XGBoost? [WWW Document], n.d. . NVIDIA Data Science Glossary. URL https://www.nvidia.com/en-us/glossary/data-science/xgboost/ (accessed 8.10.22).

Whittaker, C., Ryner, B., Nazif, M., n.d. Large-Scale Automatic Classification of Phishing Pages 14.