Cardiff University with the School of Computer Science and Informatics

Detecting Phishing Scams using machine learning

Cameron Williams

C2058507

CM3203

Project supervisor: Amir Javed

May 2023

Abstract

Phishing emails are a huge problem within the digital age causing a huge amount of monetary loss to both the public and businesses every day. This project looks at using machine learning algorithms to predict and classify emails as normal emails (ham) or phishing scam emails. I collected online data from phishing email corpuses and legitimate ham email collections to analyse and extract features from such as keywords, spelling mistakes and other lexical metrics as well as the emotional context of the email and parts of speech used in the communication. I trained a few machine learning models to classify emails as phishing or ham and the random forest algorithm produced an accuracy of over 99%. When used to predict unseen ham emails it returned a large number of false positives for phishing (66% FP). This indicates the model is overfitting and more work needs to be done to balance the datasets and improve upon the features extracted.

Acknowledgements

I'd like to thank Amir Javed, my project supervisor, for giving me the opportunity to work on this project and the guidance throughout.

I would also like to thank my family for the enormous support throughout and during the last term of my time at university.

Table of Contents

Table of Figures

1. Introduction

Phishing Emails are a common, widespread scam targeting everyone with an email address by pretending to be from a trusted source, like their bank, social media accounts, online shopping, delivery services, or government services. These scams can have a huge impact on victim's lives when their information is stolen, leading to their online accounts being comprised and inevitably being tricked into giving away a huge amount of their money.

As of March 2023 the number of phishing scams reported to the UK's National Cyber Security Centre (NCSC) is over 19 million(NCSC 2023). Scammers use highly emotional attacks exploiting current events such as Covid 19, to make messages believable and more plausible. Over the pandemic there has been more phishing emails pretending to be delivery companies asking for payment, and government emails based around Covid 19, worrying the victim into thinking they owe money. Fear tactics are used in over 60% of phishing emails(Norris and Brookes 2021) and 6.3% of UK cyber fraud in 2022 was coronavirus based(ONS 2022) . Around 320,000 people fall for phishing scams every year, costing the world $44.2 million (Griffiths 2023).

It's important that the public are informed and have knowledge of scams so that they are less likely to fall for them. One way of making the public more aware of the risks of phishing scams is to implement tools directly into the email client which raise awareness of the likely hood that the email is phishing.

This project aims to provide a new method of detecting phishing emails by extracting lexical and emotional features from the emails and processing different attributes of the email. The features will include lexical metrics, part of speech word counts and emotional analysis of the email which could allow for a more in-depth analysis of the email compared to current solutions.

2. Background

Phishing scams land in people's inboxes every day impersonating big companies such as PayPal, the government and delivery companies such as Royal Mail and FedEx. Along with the public, phishing can also affect businesses with business impersonation pretending to be a member of the organisation asking to make payments to the scammers. It is important that when scam emails do get through into the inbox that the user can be aware that it may be phishing.

2.1 Current Solutions

Gmail is one of the most popular email providers with more than 1.8billion active users and 29.5% of all the emails globally (Ruby 2023). Whilst google says they prevent 99.9% of phishing and spam from getting into the inbox, phishing scams still get through. Gmail checks incoming emails for suspicious email headers such as mismatched sender addresses and suspicious server paths (Poston 2020).

A modern method for improving trust of sender addresses is using DMARC (Domain based message authentication reporting and conformance). It denies scammers use of spoofed domains to look like it is coming from a real company. It uses already implemented security protocols (DKIM and SPF) to validate where the emails are coming from using the ISP's DNS and who is authorised to use the domains(Nagele 2023). DMARC works by telling the email provider to block emails that do not pass both security protocols to help prevent phishing attempts. The introduction of DMARC helped prevent approximately 25 million attacks in 2013 and dropped spoofed PayPal emails by 70% instantly (Agari 2014). This doesn't prevent all attempts at phishing as some phishing does not use accurate spoofed addresses and can use different styles to make the email look legit.

Some of the other techniques Gmail use in phishing prevention, include user-based feedback, where the user can report emails as phishing so that google can work to prevent them in the future. This comes with issues such as trusting the user to report phishing emails accurately, which could lead to false positives if not done properly, meaning the user could be missing important emails.

2.2 Current research

This section includes other research and programs that have been developed for analysis of spam emails.

Apache Spam Assassin is an open-source enterprise local mail filter server which blocks spam from entering the inbox. It uses a combination of header and text analysis along with checking DNS records to give the email a total score to predict if it is spam. It's been running since 2001 and is regularly updated. It uses public contributions to train its score generation and allows user submissions for new or modified rules to classify spam emails (ApacheSpamAssassin 2001).

Mail trout is a browser extension tool that uses a machine learning model to analyse the attributes of an email to predict if it's a scam. It looks for 4 different classes of spam, business email compromise, extortion, impersonation, and unexpected winnings. It removes stop words from the emails and limits the body length to 500 words.

Diego O'Campoh released a project on using machine learning to detect phishing emails by doing analysis on the emails. (O'Campoh 2017) They used many features measured by Yes/No such as presence detections of Html, JavaScript, URLs ,flash content and any attachments in the emails. They used an objected orientated approach and used RapidMiner to execute different machine learning algorithms such as decisions trees and random forests. It uses datasets from a phishing corpus and non-phishing from Enron data set.

2.3 Missing from current solutions.

Missing from Gmail, SpamAssassin, Mail Trout and O'Campoh' s project is any kind of sentimental analysis on the content of the email.

Whilst they do analyse the content such as attachments and suspicious links they do not take in the context of what the email body is asking for. The current spam filters have only a basic check on the types of email being sent with many of the features measured are a Yes/No presence check or a total amount of score to test if its spam or phishing.

Also missing from current solutions is any kind of lexical checks on the content, such as tagging words with their speech types to analyse the text. Also, they could be checking for spelling mistakes and punctuation. Spam Assassin uses a total score to predict if the emails are spam whereas my project will aim to use machine learning to categorise the email.

SpamAssassin and Mail Trout do not have a focus on impersonation phishing emails and use datasets that contain plain text personal emails and other types of spam and marketing emails whereas my project will only have a focus on phishing emails, where the datasets are the difference between a real email and a phishing email.

Other solutions use the whole email as the data input for a machine learning model and classify that way, but it has no extra analysis on the sentiment.

My project will be loosely based upon Diego Ocampohs project where I am using features from emails to train a machine learning model, which will also be like spam assassin where each feature is a number however instead of scoring it, it will directly input into the machine learning model.

This project will aim to be able to show that analysing a combination of features from the email and the text will make a more accurate spam filter, with a focus on phishing emails.

3.Approach

It will use a python script (Python Version 3.10.0) (Python 2021) to input the mbox files with the emails and run feature functions to analyse the values. It will then output the values into a csv file full of all the email's features. The csv of features can then be inputted into the machine learning program to train a model.

3.1 Features

All the features extracted by the program and outputted into the csv.

Feature	**Description**
Length of address	Length of sender's email address
Numbers In Email Address	Numbers in senders email address
Average word Length	total amount of characters in the body of the email divided by the total amount of words.
Keyword Count	Number of key words found the body text of the email
Spelling Mistakes	Count of spelling mistakes
Capital letters	Count of capital letters in body
URL Length	Length of URLs in body
Numbers in URL	Numbers in URLs
Fake Links	Count of links using mouse over to hide the original URL
Emotion affects (Anger, Anticipation, Disgust, Fear, Joy Negative, Positive, Sadness, Surprise, Trust)	Emotional Affect frequencies from the NRCLex module (metalcorebear 2019)
Parts of Speech	Categorises each word into its part of speech tag and outputs a total count for each tag
Tag	Tagged "phishing" or "ham"

Figure 1 Table of Features

3.1.1 Most notable Features

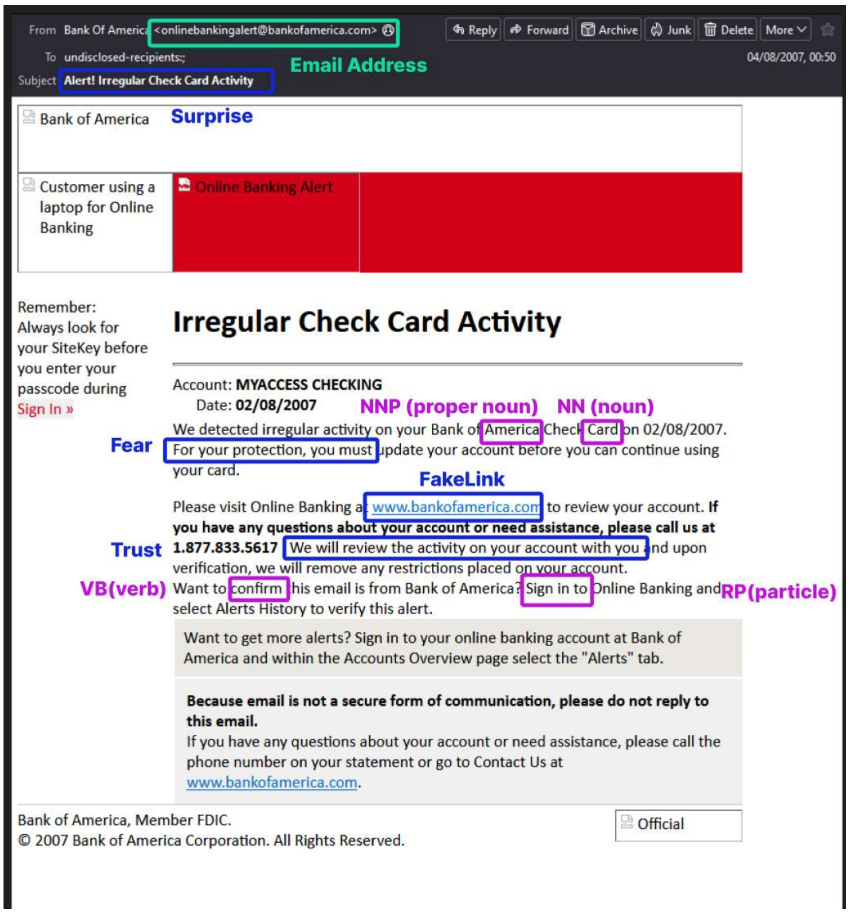
The fake link feature will check when email senders are trying to hide the hyperlinks actual destination by having the highlighted link display a different URL to the one it will direct you to. It checks for an onmouseover attribute and will return a count of onmouseover attributes found.

The emotional affects feature will analyse the text from the email and will return frequencies of each emotion into the csv. This allows for a measure for the tone of the email and gives the machine learning model more information than simple text features like capitals letters and spelling mistakes. The emotions in a phishing email are an important part of the impersonation scam working. Fear has been the most impactful emotional in phishing scams over the past three years where it can make use of current situations such as pandemics, natural disasters or government policies to manipulate the victims and convince them they need to take action and make them click through to the scam and coercing them into sending money(Kofod 2020).

The part of speech tags will take the text from the email and tokenise it into words, then each word will be categorised by its part of speech tag. For example (Singular noun)NN:8, (proper noun) NNP:10. (Harrison 2015). This allows for a representation of how the email is structured and the tone that is used in the email (Park and Julia 2015). It also gives the features a lot more data for the machine learning model to train on.

The keywords feature will input a keywords text file that contains common words that appear in phishing emails. It will count the number of keywords matched within the text of the email. This will allow the model to have more information about the likeliness of the email being phishing.

This is an example of what the features would be analysing from an email.



From Bank Of America <onlinebankingalert@bankofamerica.com>
To undisclosed-recipients;;
04/08/2007, 00:50
Subject Alert! Irregular Check Card Activity

Bank of America

Customer using a laptop for Online Banking

Online Banking Alert

Remember: Always look for your SiteKey before you enter your passcode during Sign In »

Irregular Check Card Activity

Account: **MYACCESS CHECKING**
Date: **02/08/2007**

We detected irregular activity on your Bank of America Check Card on 02/08/2007. For your protection, you must update your account before you can continue using your card.

Please visit Online Banking at www.bankofamerica.com to review your account. **If you have any questions about your account or need assistance, please call us at 1.877.833.5617** We will review the activity on your account with you and upon verification, we will remove any restrictions placed on your account.
Want to confirm this email is from Bank of America? Sign in to Online Banking and select Alerts History to verify this alert.

Want to get more alerts? Sign in to your online banking account at Bank of America and within the Accounts Overview page select the "Alerts" tab.

Because email is not a secure form of communication, please do not reply to this email.
If you have any questions about your account or need assistance, please call the phone number on your statement or go to Contact Us at www.bankofamerica.com.

Bank of America, Member FDIC.
© 2007 Bank of America Corporation. All Rights Reserved.

Official

Figure 2 Example of Features being extracted from an email.

3.2 Datasets

The first dataset to be used is a phishing corpus with around 2000 phishing emails from the Nazario phishing corpus files (Nazario 2004-2018). These contain phishing scams impersonating companies such as PayPal, eBay, and multiple Banks. I chose this dataset as it contains HTML content and represents a good amount of the impersonation phishing that lands in inboxes every day.

The second dataset which contains the Ham Emails are from the old Apache mail assassin corpus with around 500 emails(SpamAssassin 2004). I chose to use this dataset oppose to a larger ham dataset is it doesn't contain as many personal plain text emails and contains more emails that would typically come from companies like account updates or marketing emails so it would compare better with the phishing emails.

it's difficult to find more ham that is comparable to the phishing emails from companies with personal accounts due to privacy concerns and people not wanting to share that kind of content. The Enron dataset contains around 33,000 emails with 33% being ham emails (Enron 2000). These emails are mostly personal emails from 2000 with people communicating with each other like they would over text so is not useful in this case for training against impersonation emails. There are more ham datasets available such as Hillary Clintons emails but this would not be representative of a normal inbox (Kaggle 2015).

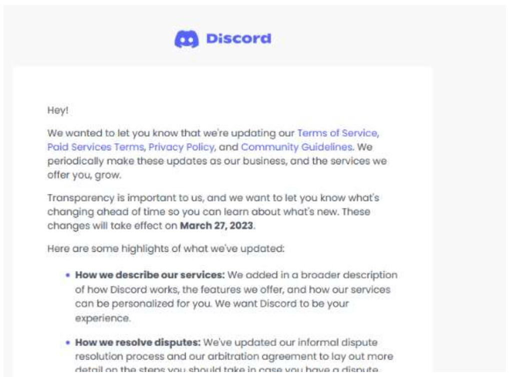
3.2.1 New data for testing

To begin with I set up 3 different honey pot email addresses to attract phishing emails and ham emails. I set up a Gmail, ZohoMail and an AOL account.

To attract scammers and phishing emails I posted my email to multiple forums such as twitter, reddit and some leaks forums. I also signed up for spam accounts such as marketing deals and online surveys which could be survey scams, which give your email to a huge amount of marketing companies and collect your personal data for passing onto scammers. They use tricks like "Free gifts" and impersonate real companies like supermarkets promising to give you gift cards. (KAHN 2022)

Unfortunately, no phishing emails were sent to the emails in time for this project. The ideal situation would be a data breach involving my honeypot email addresses as then the addresses would be on a publicly available list for scammers to target and I would be up to five times more likely to be phished according to (Google 2021). However, I still was able to collect my own phishing emails from my personal inboxes which have been subject to many data breaches over the previous decade.

These Ham emails contain different types of personal business emails such as account updates, terms and conditions updates, and some newsletters. The phishing collected from multiple of my email addresses contains emails claiming they are from eBay, FedEx Royal Mail etc.



Discord

Hey!

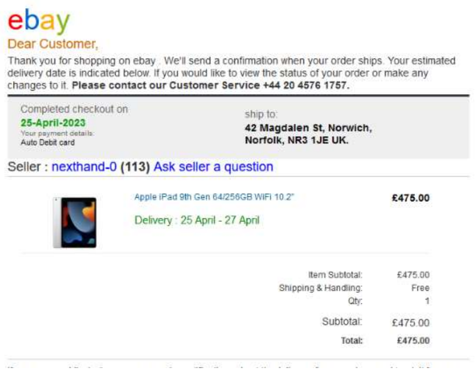
We wanted to let you know that we're updating our Terms of Service, Paid Services Terms, Privacy Policy, and Community Guidelines. We periodically make these updates as our business, and the services we offer you, grow.

Transparency is important to us, and we want to let you know what's changing ahead of time so you can learn about what's new. These changes will take effect on **March 27, 2023**.

Here are some highlights of what we've updated:

- **How we describe our services:** We added in a broader description of how Discord works, the features we offer, and how our services can be personalized for you. We want Discord to be your experience.
- **How we resolve disputes:** We've updated our informal dispute resolution process and our arbitration agreement to lay out more

Figure 3 Example of Personal Ham Collected



ebay

Dear Customer,

Thank you for shopping on ebay . We'll send a confirmation when your order ships. Your estimated delivery date is indicated below. If you would like to view the status of your order or make any changes to it. **Please contact our Customer Service +44 20 4576 1757.**

Completed checkout on
25-April-2023
Your payment details:
Auto Debit card

ship to:
42 Magdalen St, Norwich, Norfolk, NR3 1JE UK.

Seller : nexthand-0 **(113)** Ask seller a question

Apple iPad 9th Gen 64/256GB WiFi 10.2" **£475.00**

Delivery : 25 April - 27 April

Item Subtotal:	£475.00
Shipping & Handling:	Free
Qty:	1
Subtotal:	£475.00
Total:	**£475.00**

Figure 4 Example of Personal Phishing Collected

3.3 Machine Learning

With the features extracted from the emails each row will have an additional feature for its classification- “phishing” or “ham”. The model will have a binary classification task where this will be the feature that it is predicting.

The feature data will be split 80% for training and 20% for testing for the machine learning model to return statistics of its performance.

3.3.1 Models

The machine learning program will use a couple of different algorithms to make a model.

The first model I will use is Nearest neighbour. It checks the nearest neighbouring datapoints using Euclidean distance to predict what to classify a datapoint.

Another model I will use is the Random Forest Algorithm. This uses grouping to construct multiple decision trees to predict the classification of the data.

The other model I will use will be the Naïve Bayes algorithm. This takes the assumption that each datapoint is independent from each other and then is worked backwards through each datapoint which then updates the probability that they are independent. It classifies the data based upon the probability given.

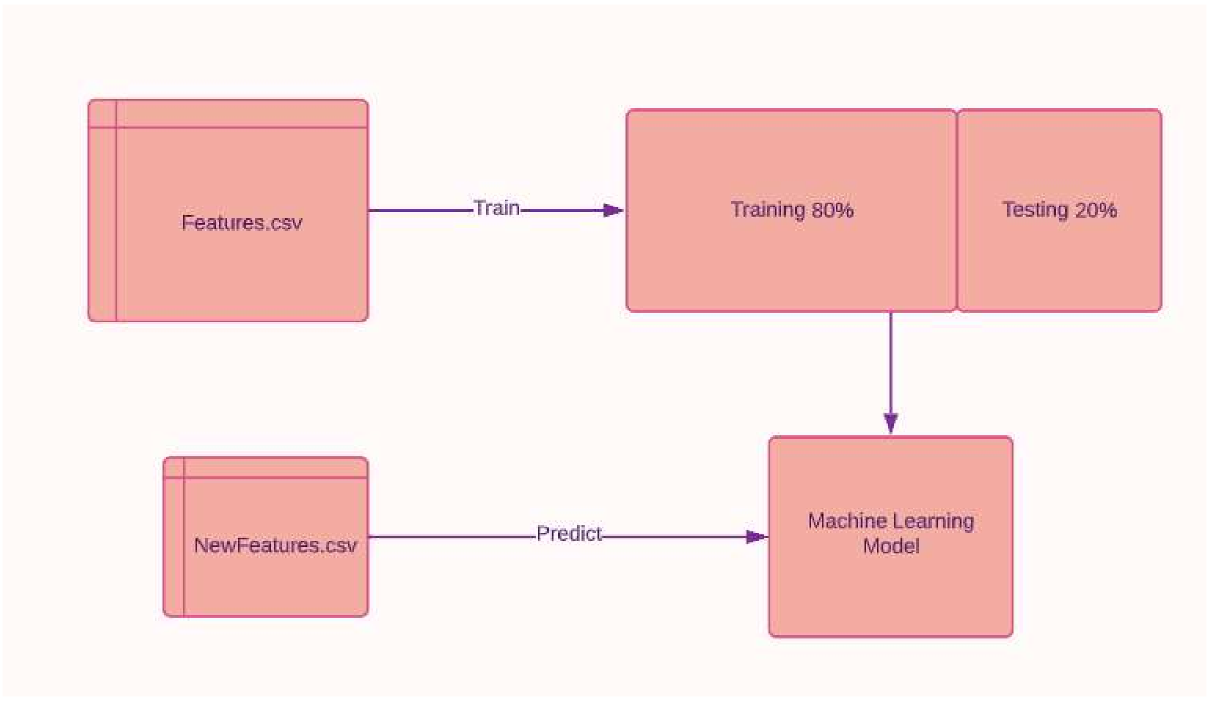


Figure 5 Flowchart of Data for Machine Learning

4. Design and Implementation

I have designed the python programs to be dynamic in nature, with it being very simple to make tweaks and modifications to the feature extraction algorithm. Each feature is a separate function where they can be added or removed by changing the list of features in the main program.

I decided to put emails into the mbox file format. This is a simple format where it can hold multiple messages in one file. It makes it easy to read as the email messages are all in the same format and can be parsed easier with python's mailbox library. To create the mbox files I used an email client called Thunderbird (Mozilla 2023). This is an open-source program which allows easy manipulation of email messages and can be modified with plugins for importing and exporting mbox files.

The feature extraction script takes four email mbox files as input for Phishing data, Ham Data and the new Phishing and ham data for prediction. The mbox file names can be dynamically changed on one line so that different datasets and corpuses can be evaluated into features and then trained into a machine learning model.

When the program has finished processing the email features it will output 2 csv files. One for the first phishing and ham data, the second one is for the new phishing and ham data. It appends "ham" or "phishing" to each row depending on which data the row is from.

Within the machine learning program, it takes in the training data and runs three machine learning models on the data and uses the 20% testing to test. As the data is inputted filters may need to be added to the data to remove any rows that did not give sufficient data. Once the models have been trained it takes in the new data from the second csv and predicts its classification, "ham" or "phishing".

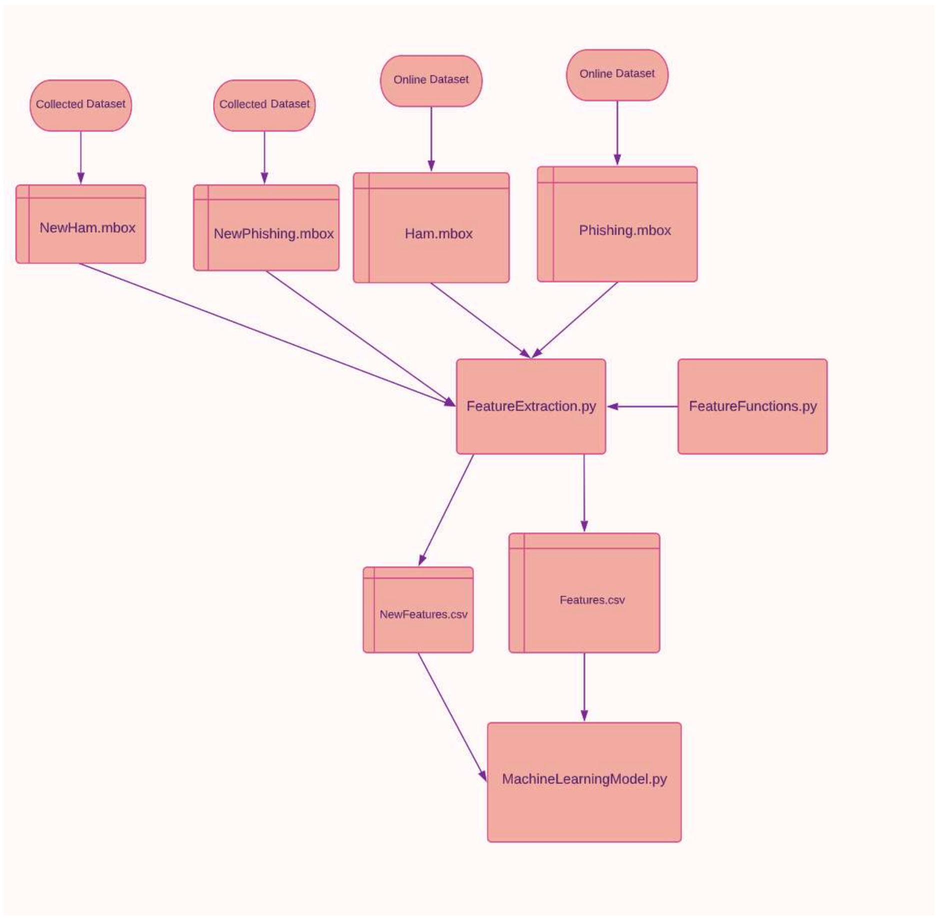


Figure 6 Flow Chart of the system's Data Design

4.1 Code Analysis

Within the main feature extraction program is a list of feature functions from their python file. This allows for the separate of the features for modular approach.

One of the first problems encountered during programming was being able to parse the mbox files to get each message. To do this the program uses a python library called mailbox (Python) which can take an mbox file and split it into messages which can be iterated through.

Another challenge was getting the html emails plain text out of the html code to be able to analyse the text. To do this it uses a HTML parser called Beautiful Soup (Richardson 2023). This allows the function getText() to return the plain text from the HTML emails for analysis in other functions for different features.

The program makes use of the natural language tool kit from the NLTK Team (NLTKTeam 2023) which contains the part of speech tags for the feature. It takes in a body of text and returns a count of each speech tag as a dictionary; this is then added to the features with each tag being its own column.

The emotional affect frequencies are generated using NRCLex (metalcorebear 2019). It takes a body of text and measures the emotional affects. It uses the NLTK library to map words and phrases to different emotions and returns a rating for each emotion plus a positive and negative sentiment.

The keywords feature takes in a keyword text file which contains words that are commonly found in scamming and phishing emails. The text file is a collection of around 800 words from online sources which describe them as high risk keywords for scams.(RAY PUGH 2021) (KAEVAND 2022). The function checks the text for each keyword and returns a count.

4.2 Features

There are 12 features being extracted which produces 69 values excluding the classification tag "phishing" or "ham".

File	Name	Description
Features.py	test(message, title=0)	A testing function that returns the current message for debugging
	getText(message)	Uses the beautiful soup module to parse the html from the body of the email and returns the plain text.
	Length(message,title=0)	Uses a regular expression to take the email address from the header and returns the length
	numbersinEmailAddr(message, title=0)	Uses a regular expression to take the email address from the header and returns a count of numbers
	avgWordLength(message, title=0)	Uses getText() to get the length of the body and divides by the number of words to get an average word length
	keywordCount(message, title=0)	Goes through each word in the keyword list (from a txt file) and takes a count of how many are in the email body text
	spellingMistakes(message,title=0)	Uses NRCLex to tokenize the email body into words then uses Spellchecker (metalcorebear 2019)to make a list of incorrect words. The function returns the length of the list.
	linkLength(message,title=0)	Uses beautiful soup to parse the HTML and find link tags ("a") with the HREF attribute. Returns the length of the URL.
	numbersinLink(message,title=0)	Uses beautiful soup to parse the HTML and find link tags ("a") with the HREF attribute. It then checks through the link for any numbers and returns a count
	fakeLinks(message,title=0)	Uses beautiful soup to parse the HTML and find link tags ("a") with the 'onmouseover' attribute. Returns a count of links found
	emotion(message)	Returns a dictionary of affect frequencies found using NRCLex's emotion analyser
	pos(message)	Returns a dictionary of part of speech counts from the natural language tool kit's part of speech tagger.
Emailinput.py	extractFeatures(message)	Takes a list of feature finders imported from features.py and iterates through each one collecting the returned values and adds the values to a dictionary
	getMbox(mbox)	Opens the Mbox file. For each message in the Mbox file calls extractFeatures(message) and adds the returned dictionary to a list.

Figure 7 Table of Functions and Feature explanations

4.3 Machine Learning Models

To implement the machine learning models into python I am using an open source library called Sci kit Learn. (developers 2023).

To explain the choices for the machine learning algorithm parameters I first need to explain overfitting. Overfitting is when the model is trained too specifically on the data given and struggles to generalise to new data. It can happen when there is limited data, or the data is too tight together. The risk of overfitting within my algorithms is quite high due to the limited amount of "ham" data available and the imbalance with the phishing data.

When implementing the Knn Nearest neighbour's algorithm I changed the nearest neighbour's parameter to 7. This makes it so it doesn't learn the data as intensely as a value of 3 with the data being unbalanced to help reduce the overfitting.

When implementing the random forest algorithm, I used the class weight parameter to help the imbalance in the data. By setting it to "balanced" it adjusts the weights inversely proportional to class frequencies (SciKit-Learn 2023). The max depth value is set to 5. This is so the algorithm does not get too deep into the data as it is unbalanced and reduces overfitting.

5.Results and Evaluation

Here are the results from extracting the feature metrics from all the emails with a machine learning model trained on the features extracted and analysed by different sentiment modules and lexical evaluation.

5.1 Machine Learning Results

When looking at the feature data that has been extracted, several plain text emails are not being analysed by the program and a majority of feature values for specific emails are null and no emotions or parts of speech have been measured.

To reduce the noise within the results data I have added a filter to remove any rows where the average word length is below "1". This makes sure that any emails that were not properly read and the emails that have a huge number of null values will not be used in the training data to help increase its accuracy.

The accuracy is how many times the model has made a correct prediction, this should be relatively high due to the unbalanced dataset.

The precision score is a ratio of the number of true positives predicted vs false positives.

The F1 score is a metric based upon the number of True positives, False Positives, True Negatives, and false negatives the model has whilst training.

With around 2250 phishing emails and 500 ham emails to begin with, after the word length filter there are 2449 emails in total with 326 being ham emails. I ran 3 different machine learning models to find which one performs the best.

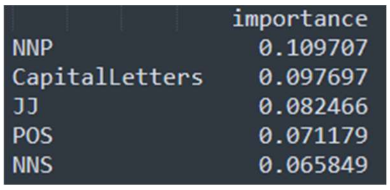
Model	Accuracy	Precision	F1 score
Nearest Neighbour	0.9836	0.988	0.9907
Random Forest	0.9979	0.9976	0.9988
Gaussian Naive Bayes	0.95510	0.9701	0.9746

Figure 8 Table of Machine learning results

As shown by the table, Random Forest performed the best with the highest scores together. This will be used to calculate feature importance and new data prediction.

5.2 Feature analysis

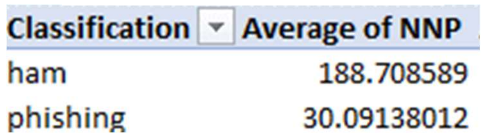
With the random forest algorithm, I was able to calculate what it calculates are the most important features using the feature_importances_ attribute.



```
                importance
NNP               0.109707
CapitalLetters    0.097697
JJ                0.082466
POS               0.071179
NNS               0.065849
```

Figure 9 Feature Importance Calculated with Random Forest

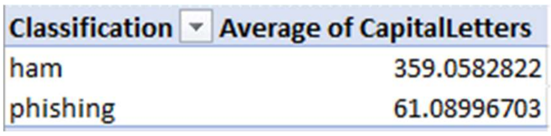
This shows that the most important feature is NNP (Singular proper Nouns). This is a count of the number of singular pronouns found in the body of the email. As shown by the data in the csv file, the average number of proper nouns in ham emails is significantly more than phishing emails. This may be down to the ham emails directly addressing people by their name whereas the phishing emails are sent out on mass so are not directly addressed.



Classification	Average of NNP
ham	188.708589
phishing	30.09138012

Figure 10 Average Noun Count for each classification

The second most important feature it shows is capital letters. When analysing the data to explain why this feature may be important, it is clear that ham emails contain a huge amount of capital letters compared to phishing emails. I believe this is down to when companies are sending out emails they want them to look professential and are proof read many times with correct punctuation, whereas phishing emails may come from people who are not as fluent in english and have many punctuation and spelling mistakes.



Classification	Average of CapitalLetters
ham	359.0582822
phishing	61.08996703

Figure 11 Average Capital Letters for each classification

5.3 Predicting with new data.

The machine learning model inputs the feature values extracted from the ham and phishing emails I collected. It then uses the trained machine learning model to classify the new data.

When given 12 emails of 6 phishing emails and 6 ham emails it removes one phishing email due to the average word length being below 1. In total it predicts the 5 phishing emails correctly with 100% success rate.

With the ham emails it only predicted 2 out of 6 to be ham. This means it has 4 false positives. With a success rate of 33% for predicting ham correctly.

Whilst the model does predict all phishing emails correctly (true positives), it also predicts 67% of ham as phishing (False Positives). Whilst it may stop all phishing emails, this would be very inconvenient to the user as their important emails would be going missing into the spam folder.

It is clear that the machine learning model is overfitting to its data and cannot be combatted by different parameters used during the training process. More work would need to be done to create a more balanced model.

5.4 Evaluation

A limitation of this project is that it is not up to date with current scams due to the datasets it uses for training being over 10 years old. The model is not able to adapt to new tactics scammers are using in real time and has to have seen emails with the same attributes before to be able to classify it correctly. If the model was given new datasets from recent times it may be better at recognising new scams as they adapt but it will still be behind the security curve due to the time needed to collect a relevant and large enough sample of emails.

This project will only be useful for emails that are in English. The features extracted are based around English lexicon and the modules that evaluate the emotions of the email are made for English text, The python mailbox library does also not support non-ascii characters. Whilst the features extracted do analyse the email for its emotional context it does not look at the wider context such as suspicious attachments, DNS records or any other vulnerabilities within email protocols.

Another limitation of this project is that it does not show the user why it thinks an email may be phishing or ham. For an end user it might be useful to educate them as to the reasons why it may be a scam which after a long period of use they will learn to be able to identify phishing themselves.

Evaluating the results, I think this could be useful to improve upon general email security where it classifies emails as either phishing or ham. Although it does not check for other types of spam or scam emails I think it could be a useful tool in providing more analysis on fraudulent emails and being able to find the most prominent parts of a phishing scam which could help educate the general public on the signs that may indicate it is not legitimate. As more data becomes available the tool can be used to keep up to date with modern impersonation techniques to help improve protection against phishing scams.

6. Conclusion

In conclusion the concept of extracting text-based features from emails and analysing the emotional context within the text can give a good machine learning model basis, however more work is needed to improve upon the model. Some of the features such as capital letters and the part of speech tags come out on top as good ways for observing phishing emails. When using the model to predict new data it has a high false positive rate for predicting ham emails which would cause inconvenience to a user's inbox. The model is overfitting to the training data and to overcome this more work is needed to collect ham emails. More work will also be needed to modify and tweak the features to make sure the entire breath of the email is analysed and given a metric. With these additions this method which fully encompasses the entire email with extracting multiple features and doing extra lexical and emotional analysis from emails could be an accurate tool for classifying phishing emails.

7. Future Work

Future work for this project could contain multiple advancements to improve upon the performance of the feature extractions and the results of the machine learning model.

Firstly, more features can be added to improve the scope of the email being analysed such as word frequency , sentence lengths and presence of @'s in URL's. Work can also be done to modify the existing features, for example some emails are not read correctly and return mostly 0's, stop words could be removed from the text before analysing it and capital letters could be a proportion of text instead a general count, to account for the longer emails.

More work could also include fetching more up to date phishing corpuses to improve the accuracy of the model and allow it to make better predictions about new emails coming into inboxes. There are newer corpuses than the ones used in this project such as a phishing corpus from Jose in 2018 but the python mailbox does not support non Ascii headers (flokli 2020) which many of the emails in the corpus contain due to them being in different languages.

Another option for future work would be to incorporate the model into a browser tool or email plugin which could be used directly in the browser to analyse new emails when they come into the inbox to allow the user to see the likelihood of the email being phishing.

8. Reflection on Learning

On reflection ive learnt that everything takes longer than planned and will be a bigger issue than first perceived. When I started this project I planned to be able to collect a huge amount of phishing emails with honeypot email addresses very quickly in around two to three weeks. I thought because phishing was such a huge portion of scams it would be fast, however with my new emails having zero digital footprint that would be a bigger task than expected and one that I could have parallelised whilst also starting to develop the code, instead of doing one task at a time.

With more time I also could have tweaked the program to get better features and had time to go back over the tasks and iterate to improve upon them. I have definitely learnt that time management is very important given a project and the ability to adapt to changes along the way, but also having time to go backwards to check what has been done and how to improve it as I am creating the project.

I have learnt to adapt to change throughout the project when difficult problems come up, such as having to use online datasets instead of being able to use my data on its own. It's clear that when something isn't working as intended and is not reasonable to fix within the time frame it is a good idea to be able to adapt and take a new approach to be able to reach the deadline with a working process, sometimes leading to a better outcome than the original methodology I was trying to take.

I've also learnt that given an idea and organising a plan into steps I can create programs from start to end, unlike previous coursework within my degree where I have only worked on a small portion of a program. I have enjoyed working on the project from start to finish.

References

Agari. 2014. *DMARC by the Numbers*. Available at: https://www.agari.com/blog/dmarc-numbers

ApacheSpamAssassin. 2001. *SpamAssassin*. Available at: https://spamassassin.apache.org/index.html

developers, s.-l. 2023. *SciKitLearn*. Available at: https://scikit-learn.org/

Enron. 2000. *Enron corpus*. Available at: http://nlp.cs.aueb.gr/software_and_datasets/Enron-Spam/index.html

flokli. 2020. *mailbox.mbox fails on non ASCII characters #86599*. Available at: https://github.com/python/cpython/issues/86599

Google. 2021. *New research reveals who's targeted by email attacks*. Available at: https://workspace.google.com/blog/identity-and-security/how-gmail-helps-users-avoid-email-scams

Griffiths, C. 2023. *The Latest 2023 Phishing Statistics*. Available at: https://aag-it.com/the-latest-phishing-statistics

Harrison. 2015. *Part of Speech Tagging with NLTK*. Available at: https://pythonprogramming.net/part-of-speech-tagging-nltk-tutorial/

KAEVAND, R. 2022. *The Ultimate List of 700+ Spam Trigger Words to Avoid in 2022*. Available at: https://instantly.ai/blog/spam-trigger-words#the-ultimate-list-of-700-spam-trigger-words

Kaggle. 2015. *Hillary Clinton's Emails*. Available at: https://www.kaggle.com/datasets/kaggle/hillary-clinton-emails

KAHN, R. 2022. *Fraud Detection: Online Survey Scams and How To Spot Them*. Available at: https://www.anura.io/blog/fraud-detection-online-survey-scams

Kofod, A. 2020. *Phishing with Fear: How scammers use our emotions against us*. Available at: https://dev.to/leading-edje/phishing-with-fear-how-scammers-use-our-emotions-against-us-8me

metalcorebear. 2019. NRCLEX 4.0.

Mozilla. 2023. ThunderBird.

Nagele, C. 2023. *DMARC: What is it and why do you need it?* Available at: https://postmarkapp.com/guides/dmarc

Nazario, J. 2004-2018. *nazario phishingcorpus ~jose/phishing*. Available at: https://monkey.org/~jose/phishing/

NCSC. 2023. *Phishing: Spot and report scam emails, texts, websites and calls*. Available at: https://www.ncsc.gov.uk/collection/phishing-scams

NLTKTeam. 2023. *Natural Language Toolkit*. Available at: https://www.nltk.org/index.html

Norris, G. and Brookes, A. 2021. Personality, emotion and individual differences in response to online fraud. *Personality and Individual Differences* 169, p. 109847. doi: 10.1016/j.paid.2020.109847

O'Campoh, D. 2017. *MachineLearningPhishing*. Available at: https://github.com/diegoocampoh/MachineLearningPhishing

ONS. 2022. *Phishing attacks – who is most at risk?* Available at: https://www.ons.gov.uk/peoplepopulationandcommunity/crimeandjustice/articles/phishingattackswhoismostatrisk/2022-09-26

Park, G. and Julia. 2015. Using Syntactic Features for Phishing Detection. *arXiv pre-print server*, doi: None

arxiv:1506.00037

Poston, H. 2020. *How to scan email headers for phishing and malicious content*. Available at: https://resources.infosecinstitute.com/topic/how-to-scan-email-headers-for-phishing-and-malicious-content/

Python. mailbox.

Python. 2021. Python 3.10.0.

RAY PUGH, S. W. 2021. *The top phishing keywords in the last 10k+ malicious emails we investigated*. Available at: https://expel.com/blog/top-phishing-keywords/

Richardson, L. 2023. Beautiful Soup4.

Ruby, D. 2023. *67 Gmail Statistics For 2023*. Available at: https://www.demandsage.com/gmail-statistics/

SciKit-Learn. 2023. *RandomForest*. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier

SpamAssassin. 2004. */old/publiccorpus*. Available at: https://spamassassin.apache.org/old/publiccorpus/