Detecting Spam Emails Using Machine Learning Classification Methods

Victor Zeddy’s Ochieng Onyango

Date of Submission: May 6, 2021

Abstract

Electronic mails, also known as emails, are a form of online communication where a user can send text, files, images or attachments over a network to a specific recipient or group. It is a very popular method of communication which makes it prone to various forms of misuse e.g. spam email. Spam emails refer to any unsolicited email sent in bulk to many random users. There are various forms of spam emails mainly ads, chain letters, email spoofs,hoaxes, money scams, malware warnings and porn spam. Some of these types have led to the spread of malware, phishing scams, social engineering exploits, spread of misinformation, fraud, etc. As a result, the need for ways to detect spam emails is needed in order to reduce the prevalence of the aforementioned exploits. Machine learning algorithms were introduced in order to create spam email filters. The paper aims to study some of the algorithms used and test their diagnostic abilities on detecting spam email.

Contents

List of Figures

List of Tables

Chapter 1: Introduction

2.1 Background

Spam emails are emails that are unsolicited and are sent in bulk to indiscriminate recipients (Comodo, 2021). There are many different types of email spam but the most common are ads, chain letters, email spoofing, hoaxes, money scams, malware warnings and porn spam (Gatefy, 2021). Email IDs are collected by automated software called spam bots which crawl the internet for them. Spam messages made up 47.3% of the emails sent between 2014 and 2020 (Johnson, 2021). Russia generated 23.52% of the global spam value making it the largest generator of spam emails (Johnson, 2021). The spam email rate has been decreasing from over the years. However, a substantial portion of spam emails constitute emails of malicious intent comprising of spyware, trojans and ransomware (Johnson, 2021).

Spam emails have been used as a medium to spread malware. Examples of ransomware that were spread by spam emails were locky, troldesh, cryptolocker and petya among others (Kaspersky, 2021). Some spam emails were used for phishing scams. 96% of phishing scams are done by email (Verizon, 2020) thus showing the gravity of the situation.

There are various methods used to detect spam emails some of which include looking for spelling mistakes, setting spam filters in front of the mail server and/or on the mail server and using spam detection software that use machine learning among other solutions (Wikipedia, 2021).The common machine learning methods used to detect spam emails include KNeighbors, Random forests, Naive Bayes and Support Vector Machines with linear kernels among others. This paper seeks to investigate the various machine learning methods used to detect spam emails.

2.2 Problem Statement

Spam emails are used to send malware, initiate phishing scams, facilitating money scams, send unsolicited ads and spread false information. This has led to theft, fraud, black mailing through stealing user information from spywares and communications overload which eventually leads time wastage on manual deletion of the mails (Kaspersky, n.d.).

2.3 Objectives

The purpose of the study is to investigate the various methods used in spam email detection.

2.4 Specific Objectives

i To study the effects of spam email on businesses

ii To create various machine learning methods to detect the spam emails

iii To evaluate the performance of the various machine learning models

2.5 Research Questions

i What are the effects of spam emails on businesses?

ii How effective are machine learning methods on spam email detection?

iii How can we evaluate the machine learning models to ensure they work as required?

2.6 Justification

There is need to develop algorithms that can efficiently detect spam emails. The model should be able to identify a high percentage of emails that are spams. The detection of spam emails may assist in reducing the infection radius of malware and also assist in preventing communication overload. This may also reduce the amount of money businesses and people lose to ransomware and scams.

2.7 Scope and Limitations

The study focuses on a few of the machine learning algorithms and their capability to detect spam email. The success of the study highly depends on the source of data due to the impact data has on the performance of a model (Ragab et al., 2019).

Chapter 2: Literature Review

The chapter explores the various effects of spam email and different implementations of spam email detection but with a wider scope. Successes and limitations reported in the documents of study will also be outlined in this chapter.

3.1 Effects of Email Spam

Spam email refers to any kind of unsolicited email that often comes in form of commercialized adverts (Park et al., 2016). Email spam costs European companies $2.8 billion in relation to lost productivity and US based companies $20 billion as a result of purchasing more mail servers and storage equipment to be able to stabilize with the inflow of spam messages (Park et al., 2016). In addition, more money is lost in time spent having staff free networks overloaded with spam. There are several dangers that are accompanied with email spam.

First and foremost is the prevalence of malware in the emails. Malware refers to any software designed to damage a computer, network or server (Moir, 2009). It is a catch-all term that represents software like viruses, trojans etc. A study done under the Australian Internet Security Initiative in 2020 discovered that out of 21,131,389 unique emails obtained from participating Australian internet service providers, 0.54% of the emails had attachments of which 31% were identified to be compromised by malware. 10% of the sample spam emails embedded with URLs directed to a compromised website (Broadhurst and Trivedi, 2020). Some of the common ransomwares identified by the study in the spam were Locky, NEMUCOD, Cerber, W97M and the Razy Trojan. Once infected with the ransomware, the victim is required to pay a certain amount before the file system can be decrypted. An example is an incident in 2016 when 3 hospitals were infected with the Locky ransomware (BBC, 2016). One of the hospitals was forced to pay $17,000 for access to they're files.

According to the Cisco's journal on email security, spam emails contribute to 96% of the phishing scams (Cisco, 2019). The most common types of attacks are: Office 365 phishing where the attacker tries to social engineer the victim into giving them they're Office 365 credentials, business emails compromise(BEC) where an attacker impersonates a C-level executive or above in order to trick the victim into performing a business function, digital extortion where the aim is to scare the victim into sending money to the attacker, packaging and invoice spam where the attacker impersonates a service and sends a false email to the victim with an invoice document infected with malware and advance fee fraud where the attacker sends an false email about a pending transfer which requires a certain amount to be sent before the transfer can be completed (Cisco, 2019). In 2018, the document states that $1.3 billion was lost to email phishing especially BEC and email account compromise methods.

As a result of such loses, the need for better spam email detection was required in order to detect these emails before any more damage could be caused.

3.2 Spam Detection Methods and Techniques

There are various methods and techniques devised to detect spam emails. The section below discusses some of these methods.

3.2.1 Manual Detection of Spam Emails

There are various methods used to detect phishing emails. The first method is by investigating the email. Some of the characteristics of the emails are (Cisco, 2019):

i The From: address does not align with the actual company's address.

ii The presence of numerous spelling and grammatical errors as well as blurry and/or incorrect logos.

iii Sense of urgency where the email dictates immediate action or captures your curiosity.

iv The email requests for personal or sensitive information.

v The embedded URLs do not match those of the company's domain.

vi The file types of the attachments are not those that are usually expected.

However, people behave differently during the time of the attack thus the use of software to aid users in detecting such emails Boag, 2020.

3.2.2 Machine Learning Algorithms in Spam Email Detection

Spam email filters are programs created to filter out spams from the user's inbox. There are various categories of filters that are used: content-based filters which include word-based, heuristic and Bayesian filters which attempt to filter spam emails using the content of the email, list-based filters which attempt to filter spam using a predefined list of potential spam senders or legitimate senders, challenge/response system which force a sender to perform a task before sending the email to the recipient, collaborative filters that attempt to filter spam using a data collected from a pool of volunteers and DNS Lookup Systems that attempt to verify the domain name of the sender (Tewari and Jangale, 2016). However some of these filters are prone to produce false positives or to not be aggressive as required.

Machine learning has been used to increase the efficiency of spam email filtering for example the Bayesian filter that learns what the user considers as spam and attempts to filter emails using the experience it has garnered. The two main learning styles used are supervised learning by the use of algorithms like KNN, Random Forest, Logistic regression etc and unsupervised learning by use of algorithms like the Apriori and k-Means.

3.3 Related Works

The following are research paper on various attempts to train machine learning algorithms to detect spam email.

3.3.1 Hybrid spam detection using machine learning

This research aimed at creating a hybrid machine learning solution by combining Support Vector Machines (SVM) and Naive Bayes(NB) algorithm with the aim to create a solution that utilizes the advantages of the SVM's high precision and recall rate and Naive Bayes' fast classification whilst requiring a small training set (Jawale et al., 2018). The NB algorithm was used to process the training data by calculating the probabilities of each word and the composite probability of each messages. The NB algorithm afterwards classify the dataset into spam or ham. The NB classified dataset is fed into the SVM which the performs binary classification on the NB classified dataset input. The NB-SVM solution had a 99.74% on the training set and 97.57% on the test set.

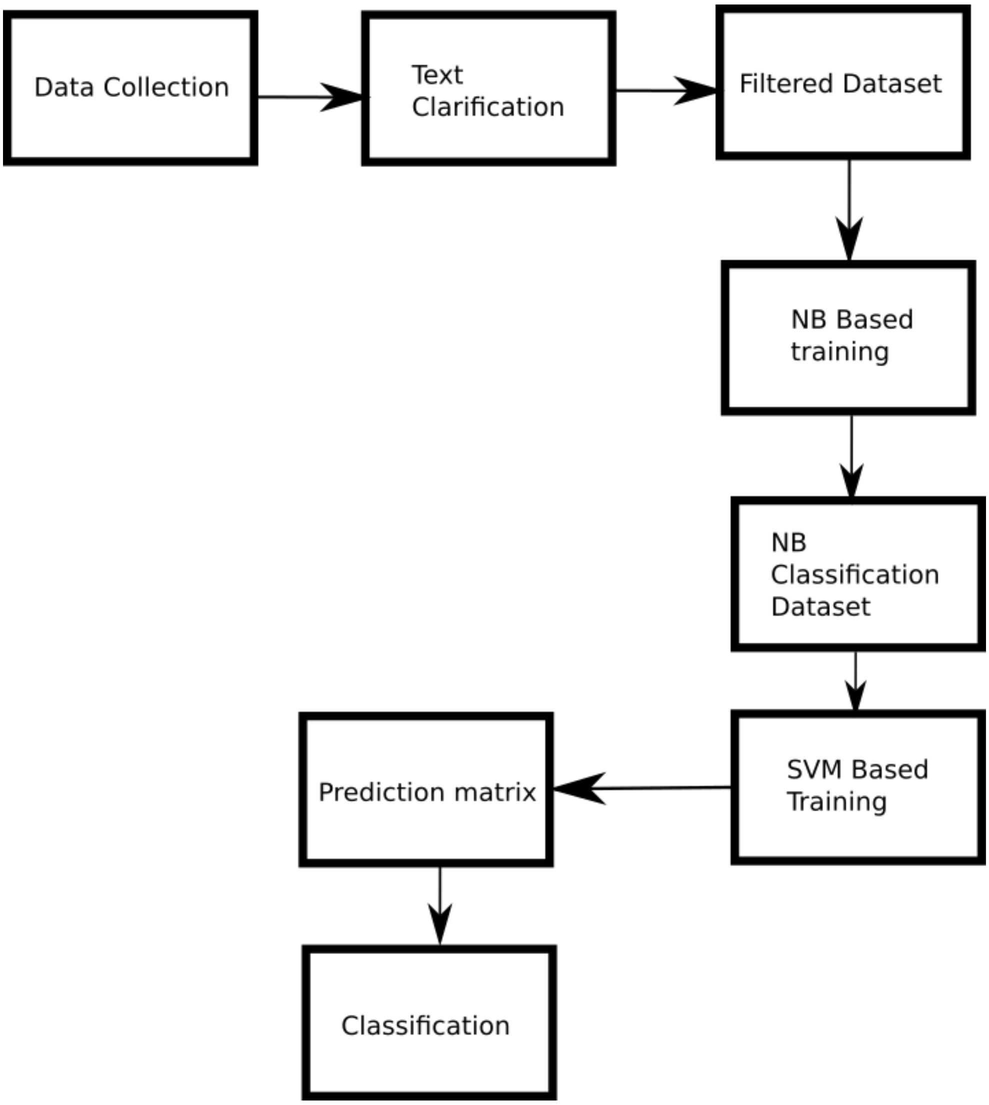


Figure 1: NB-SVM Architecture (Jawale et al., 2018)

3.3.2 E-Mail Spam Classification Using Long Short-Term Memory Method

The research aimed to create a text-based dataset module architecture based on Multi Modal Architecture Model Fusion(MMA-MF) with the aim to be able to effectively filter spam even if it's hidden in text (Dongre and Patidar, 2019). The research argued that spammers had learned to evade the single model spam filtering systems by injecting junk information into the multi-modal part of an email thereby making the spam evade detection. This type of spam is called hybrid spam and they explained that it is more harmful than traditional spam since it contains more information than the latter thereby requiring more network bandwidth and storage space. The MMA-MF architecture combined both CNN and Long Short Term Memory (LSTM) to handle image and text data respectively to generate the class probabilities of the words and images in the email. A fully connected neural network with a logistic regressor at the output layer was added to the architecture to classify the class probabilities to either spam or ham (Dongre and Patidar, 2019). Below is a figure of the architecture.

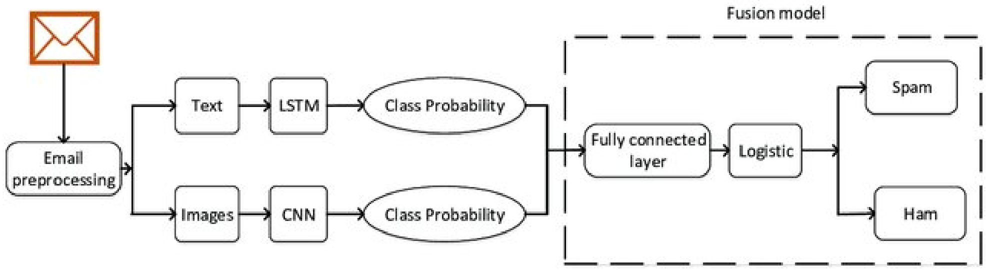


Figure 2: MMA-MF Architecture (Dongre and Patidar, 2019)

The model was evaluated using 5-fold cross validation. Below is the performance of the model.

Fold	Accuracy	Recall	F1-Score	Precision
MMA-MF Model for Text Dataset 1				
1	98.42	97.64	97.24	98.5
2	98.67	98.15	97.47	98.5
3	98.67	98.19	97.65	99
4	98.25	97.71	97.27	98
5	98.42	97.89	97.53	98.5
MMA-MF Model for Text Dataset 2				
1	93.35	92.64	92.89	90.5
2	92.56	92.63	92.75	90.01
3	91.5	92.33	91.83	93.5
4	92.35	92.83	92.97	92
5	93.44	92.72	92.71	92.5

Table 1: Fold Cross-Validation Chart for Text Dataset 1 and 2

3.3.3 Spam Mail Classification Using Ensemble and Non-Ensemble Machine Learning Algorithms

The researchers aimed to explore ensemble machine learning algorithms and compare they're performance with non-ensemble algorithms (Agarwal et al., 2020). They were able to achieve 98.47% accuracy with the SVM which is a non-ensemble algorithm and the ensemble voting algorithm achieved a 96.80% accuracy on the test dataset. K-Nearest Neighbor(KNN), Naive Bayes and SVM were the the non-ensemble algorithms used. The different approaches used to create the ensemble classifiers were: Majority Voting where they took the best performing classifiers from each model and have them make a "vote" on each test instance making the final output prediction with the one with more than half the votes; Bootstrap Aggregating or Bagging where n models are created from one model and the same bootstrap sampling algorithm(Adaboost.SAMME) creates random n sub-samples drawn from the original dataset. Afterwards the n models are fitted to the n bootstrap samples which are then averaged by voting; Boosting approach trains each model with the same data set while adjusting the instance weights in relation to the error rate of the previous prediction (Agarwal et al., 2020). Below is a table showing the results obtained from the study.

Algorithm	Accuracy	TP	FP	FN	TN
SVM	98.47	962	6	14	133
KNN	94.90	1928	0	142	159
Naive Bayes	95.61	982	0	31	102
Voting Classifier	96.80	968	0	23	124
Boosting Method	<90%	955	0	160	0
Bagging Method	97.3%	958	1	29	127

Table 2: Accuracy obtained from the different models in the Test set

Chapter 3: Implementation and Testing

This chapter studies the data pre-processing methods used during the study, the implementation of different models and the various testing methods used to measure the diagnostic capabilities of the models.

4.1 Data Collection

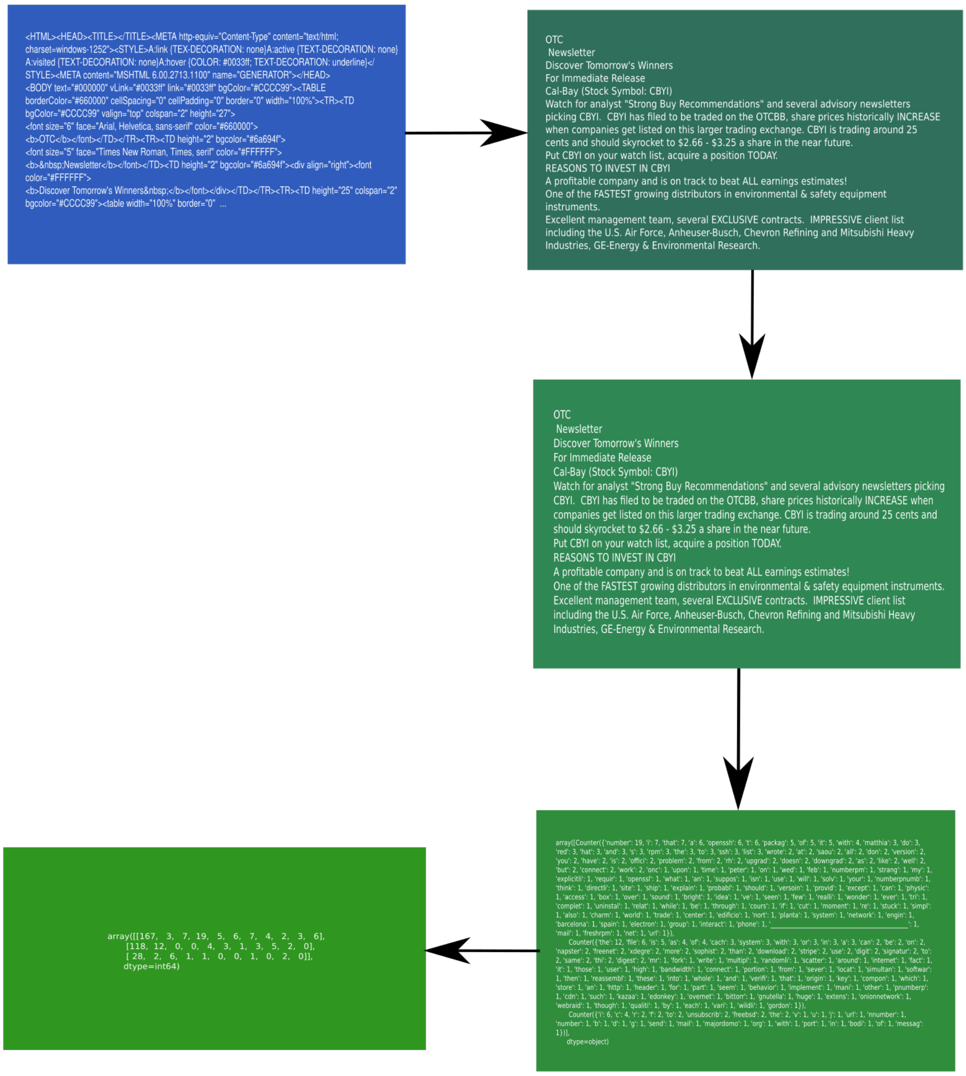
4.1.1 Primary Data

This is first hand data collected by the researcher from various sources like databases, interviews, surveys, etc. The study used the Apache SpamAssasins Dataset. 500 spam emails and 2750 ham emails(emails that are not spam) were used during the study.

4.2 Data Preparation

4.3 Data Preparation

The aim of this step was to get the words in the emails and use their frequency to determine whether an email is spam or ham. The HTML emails are converted to plain text to enable the building of a better vocabulary since HTML tags will not be of any use in creating the vocabulary space. Afterwards we get the word counts to vectors and output a sparse matrix. The text in the emails are then converted to their word count by stemming. Stemming allows us to create a shortened vocabulary space by getting the root of the words in the emails which in turn improves the feature space of the dataset.



```
<HTML><HEAD><TITLE></TITLE><META http-equiv="Content-Type" content="text/html;
charset=windows-1252"><STYLE>A:link {TEX-DECORATION: none}A:active {TEXT-DECORATION: none}
A:visited {TEXT-DECORATION: none}A:hover {COLOR: #0033ff; TEXT-DECORATION: underline}</
STYLE><META content="MSHTML 6.00.2713.1100" name="GENERATOR"></HEAD>
<BODY text="#000000" vLink="#0033ff" link="#0033ff" bgColor="#CCCC99"><TABLE
borderColor="#660000" cellSpacing="0" cellPadding="0" border="0" width="100%"><TR><TD
bgColor="#CCCC99" valign="top" colspan="2" height="27">
<font size="6" face="Arial, Helvetica, sans-serif" color="#660000">
<b>OTC</b></font></TD></TR><TR><TD height="2" bgcolor="#6a694f">
<font size="5" face="Times New Roman, Times, serif" color="#FFFFFF">
<b>&nbsp;Newsletter</b></font></TD><TD height="2" bgcolor="#6a694f"><div align="right"><font
color="#FFFFFF">
<b>Discover Tomorrow's Winners&nbsp;</b></font></div></TD></TR><TR><TD height="25" colspan="2"
bgcolor="#CCCC99"><table width="100%" border="0"  ...
```

```
OTC
 Newsletter
Discover Tomorrow's Winners
For Immediate Release
Cal-Bay (Stock Symbol: CBYI)
Watch for analyst "Strong Buy Recommendations" and several advisory newsletters
picking CBYI.  CBYI has filed to be traded on the OTCBB, share prices historically INCREASE
when companies get listed on this larger trading exchange. CBYI is trading around 25
cents and should skyrocket to $2.66 - $3.25 a share in the near future.
Put CBYI on your watch list, acquire a position TODAY.
REASONS TO INVEST IN CBYI
A profitable company and is on track to beat ALL earnings estimates!
One of the FASTEST growing distributors in environmental & safety equipment
instruments.
Excellent management team, several EXCLUSIVE contracts.  IMPRESSIVE client list
including the U.S. Air Force, Anheuser-Busch, Chevron Refining and Mitsubishi Heavy
Industries, GE-Energy & Environmental Research.
```

```
OTC
 Newsletter
Discover Tomorrow's Winners
For Immediate Release
Cal-Bay (Stock Symbol: CBYI)
Watch for analyst "Strong Buy Recommendations" and several advisory newsletters picking
CBYI.  CBYI has filed to be traded on the OTCBB, share prices historically INCREASE when
companies get listed on this larger trading exchange. CBYI is trading around 25 cents and
should skyrocket to $2.66 - $3.25 a share in the near future.
Put CBYI on your watch list, acquire a position TODAY.
REASONS TO INVEST IN CBYI
A profitable company and is on track to beat ALL earnings estimates!
One of the FASTEST growing distributors in environmental & safety equipment instruments.
Excellent management team, several EXCLUSIVE contracts.  IMPRESSIVE client list
including the U.S. Air Force, Anheuser-Busch, Chevron Refining and Mitsubishi Heavy
Industries, GE-Energy & Environmental Research.
```

```
array([Counter({'number': 19, 'i': 7, 'that': 7, 'a': 6, 'openssh': 6, 't': 6, 'packag': 5, 'of': 5, 'it': 5, 'with': 4, 'matthia': 3, 'do': 3,
'red': 3, 'hat': 3, 'and': 3, 's': 3, 'rpm': 3, 'the': 3, 'to': 3, 'ssh': 3, 'list': 3, 'wrote': 2, 'at': 2, 'saou': 2, 'all': 2, 'don': 2, 'version': 2,
'you': 2, 'have': 2, 'is': 2, 'offici': 2, 'problem': 2, 'from': 2, 'rh': 2, 'upgrad': 2, 'doesn': 2, 'downgrad': 2, 'as': 2, 'like': 2, 'well': 2,
'but': 2, 'connect': 2, 'work': 2, 'onc': 1, 'upon': 1, 'time': 1, 'peter': 1, 'on': 1, 'wed': 1, 'feb': 1, 'numberpm': 1, 'strang': 1, 'my': 1,
'explicitli': 1, 'requir': 1, 'openssl': 1, 'what': 1, 'an': 1, 'suppos': 1, 'isn': 1, 'use': 1, 'will': 1, 'solv': 1, 'your': 1, 'numberpnumb': 1,
'think': 1, 'directli': 1, 'site': 1, 'ship': 1, 'explain': 1, 'probabl': 1, 'should': 1, 'versoin': 1, 'provid': 1, 'except': 1, 'can': 1, 'physic':
1, 'access': 1, 'box': 1, 'over': 1, 'sound': 1, 'bright': 1, 'idea': 1, 've': 1, 'seen': 1, 'few': 1, 'realli': 1, 'wonder': 1, 'ever': 1, 'tri': 1,
'complet': 1, 'uninstal': 1, 'relat': 1, 'while': 1, 'be': 1, 'through': 1, 'cours': 1, 'if': 1, 'cut': 1, 'moment': 1, 're': 1, 'stuck': 1, 'simpl':
1, 'also': 1, 'charm': 1, 'world': 1, 'trade': 1, 'center': 1, 'edificio': 1, 'nort': 1, 'planta': 1, 'system': 1, 'network': 1, 'engin': 1,
'barcelona': 1, 'spain': 1, 'electron': 1, 'group': 1, 'interact': 1, 'phone': 1, '__________________________________________': 1,
'mail': 1, 'freshrpm': 1, 'net': 1, 'url': 1}),
       Counter({'the': 12, 'file': 6, 'is': 5, 'as': 4, 'of': 4, 'cach': 3, 'system': 3, 'with': 3, 'or': 3, 'in': 3, 'a': 3, 'can': 2, 'be': 2, 'on': 2,
'napster': 2, 'freenet': 2, 'xdegre': 2, 'more': 2, 'sophist': 2, 'than': 2, 'download': 2, 'stripe': 2, 'use': 2, 'digit': 2, 'signatur': 2, 'to':
2, 'same': 2, 'thi': 2, 'digest': 2, 'mr': 1, 'fork': 1, 'write': 1, 'multipl': 1, 'randomli': 1, 'scatter': 1, 'around': 1, 'internet': 1, 'fact': 1,
'it': 1, 'those': 1, 'user': 1, 'high': 1, 'bandwidth': 1, 'connect': 1, 'portion': 1, 'from': 1, 'sever': 1, 'locat': 1, 'simultan': 1, 'softwar':
1, 'then': 1, 'reassembl': 1, 'these': 1, 'into': 1, 'whole': 1, 'and': 1, 'verifi': 1, 'that': 1, 'origin': 1, 'key': 1, 'compon': 1, 'which': 1,
'store': 1, 'an': 1, 'http': 1, 'header': 1, 'for': 1, 'part': 1, 'seem': 1, 'behavior': 1, 'implement': 1, 'mani': 1, 'other': 1, 'pnumberp':
1, 'cdn': 1, 'such': 1, 'kazaa': 1, 'edonkey': 1, 'overnet': 1, 'bittorr': 1, 'gnutella': 1, 'huge': 1, 'extens': 1, 'onionnetwork': 1,
'webraid': 1, 'though': 1, 'qualiti': 1, 'by': 1, 'each': 1, 'vari': 1, 'wildli': 1, 'gordon': 1}),
       Counter({'i': 6, 'c': 4, 'r': 2, 'f': 2, 'to': 2, 'unsubscrib': 2, 'freebsd': 2, 'the': 2, 'v': 1, 'u': 1, 'j': 1, 'url': 1, 'nnumber': 1,
'number': 1, 'b': 1, 'd': 1, 'g': 1, 'send': 1, 'mail': 1, 'majordomo': 1, 'org': 1, 'with': 1, 'port': 1, 'in': 1, 'bodi': 1, 'of': 1, 'messag':
1})],
      dtype=object)
```

```
array([[167,   3,   7,  19,   5,   6,   7,   4,   2,   3,   6],
       [118,  12,   0,   0,   4,   3,   1,   3,   5,   2,   0],
       [ 28,   2,   6,   1,   1,   0,   0,   1,   0,   2,   0]],
      dtype=int64)
```

Figure 3: Data Preprocessing Steps

4.4 Model Training and Testing

4.5 Training and Testing

The models used for the study were Logistic Regression, KNeigbhor's Classifier(KNN), Decision Tree, Random Forests, Naive Bayes variants, Support Vector Machine Classifier and Embedders. GridSearchCV was used for hyperparameter tuning. The table below shows the performance of the different models.

model	Precision	Recall	f1 score
Logistic Regression	98.02	95.19	96.59
KNN	92.59	72.12	81.08
Decision Tree	80.36	86.54	83.33
Random Forest	98.97	92.31	95.52
Gaussian Naive Bayes (GNB)	42.13	95.19	58.41
Multinomial Naive Bayes	90.10	87.50	88.78
Complement Naive Bayes	90.10	87.50	88.78
Bernoulli Naive Bayes	90.53	82.69	86.43
SVC	88.89	92.31	90.57
Ensemble: Boosting	87.16	91.35	89.20
Ensemble: Bagging	97.03	94.23	95.61
Ensemble: Majority Voting	98.02	95.19	96.59

Table 3: Model Performances

For the ensemble methods, Gaussian Naive Bayes was used for the Boosting method since it was the weakest classifier. For bootstrap aggregation/bagging and majority voting methods, the models used were complement naive bayes, logistic regression, random forest and SVC.

Chapter 4: Conclusion

The primary goal of the paper was to study the performance of some of the machine learning algorithms in spam email classification. The results showed that the machine learning algorithms especially the logistic regressor and random forest were highly capable of detecting spam emails thereby their use in creating spam filters will be highly effective.

For future works, I intend to study the more complex machine learning algorithms used in detecting spam emails and in turn attempt to create a similar complex algorithm that can be used for spam email classification and compare it to the studied algorithms.

References

Agarwal, K., Uniyal, P., Virendrasingh, S., Krishna, S., & Dutt, V. (2020). Spam mail classification using ensemble and non-ensemble machine learning algorithms. *Machine Learning for Predictive Analysis*, 179–189. https://doi.org/10.1007/978-981-15-7106-0_18

BBC. (2016). Three us hospitals hit by ransomware. https://www.bbc.com/news/technology-35880610

Boag, P. (2020). 6 reasons you can never trust users! https://mediatemple.net/blog/web-development-tech/6-reasons-can-never-trust-users/

Broadhurst, R., & Trivedi, H. (2020). Malware in spam email: Risks and trends in the australian spam intelligence database, trends & issues in crime and criminal justice. *Trends and Issues in Crime and Criminal Justice*, *603*, 1–18.

Cisco. (2019). Email: Click with caution.

Comodo. (2021). What is email spam: Best ways to prevent spam emails. https://blog.comodo.com/email-security/what-is-email-spam/

Dongre, S., & Patidar, K. (2019). E-mail spam classification using long short-term memory method. *International Journal of Scientific Research &amp; Engineering Trends*, *5*(5). https://ijsret.com/

Gatefy. (2021). 7 most common types of email spam and how to identify. https://gatefy.com/blog/most-common-types-email-spam/

Jawale, D. S., Mahajan, A. G., Shinkar, K. R., & Katdare, V. V. (2018). Hybrid spam detection using machine learning. *International Journal of Advance Research, Ideas and Innovations in Technology*, *4*(2). https://www.ijariit.com/%5C?utm_source=pdf%5C&amp;utm_ medium=edition%5C&amp;utm_campaign=OmAkSols%5C&amp;utm_term=V4I2-1925

Johnson, J. (2021). Spam statistics: Spam e-mail traffic share 2019. https://www.statista.com/statistics/420391/spam-email-traffic-share/

Kaspersky. (n.d.). Damage caused by spam. https://encyclopedia.kaspersky.com/knowledge/damage-caused-by-spam/

Kaspersky. (2021). What are the different types of ransomware. https://www.kaspersky.com.au/resource-center/threats/ransomware-examples

Moir, R. (2009). Defining malware: Faq. https://docs.microsoft.com/en-us/previous-versions/tn-archive/dd632948(v=technet.10)%5C?redirectedfrom=MSDN

Park, I., Sharman, R., Rao, R., & Upadhyaya, S. (2016). The effect of spam and privacy concerns on e-mail users' behavior. *ACM Transactions on Information and System Security - TISSEC*.

Ragab, D. A., Sharkas, M., Marshall, S., & Ren, J. (2019). Breast cancer detection using deep convolutional neural networks and support vector machines. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6354665/

Tewari, A., & Jangale, S. (2016). Spam filtering methods and machine learning algorithm - a survey. *International Journal of Computer Applications*, *154*(6), 8–12. https://doi.org/10.5120/ijca2016912153

Verizon. (2020). 2020 data breach investigations report. https://enterprise.verizon.com/en-gb/resources/reports/dbir/

Wikipedia. (2021). Email spam. https://en.wikipedia.org/wiki/Email_spam

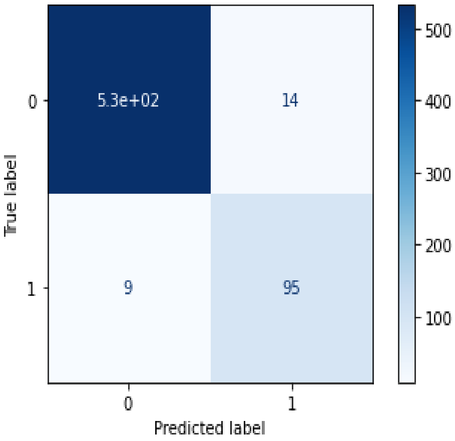
Appendix

6.1 Github Links

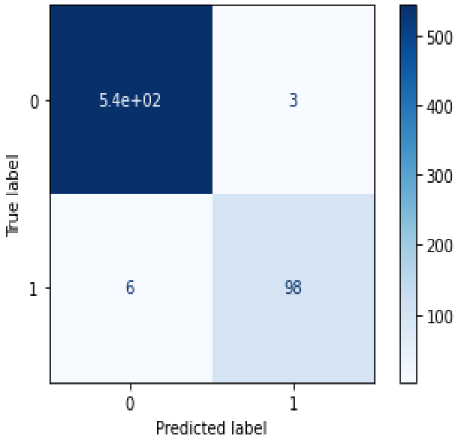
Notebook: Spam Classifier Notebook

Reference Notebooks: Hands On ML Notebook

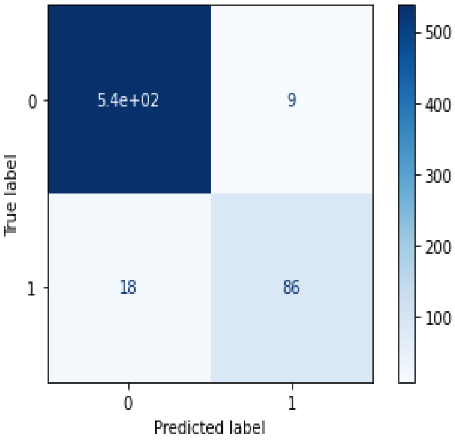
6.2 Confusion Matrices



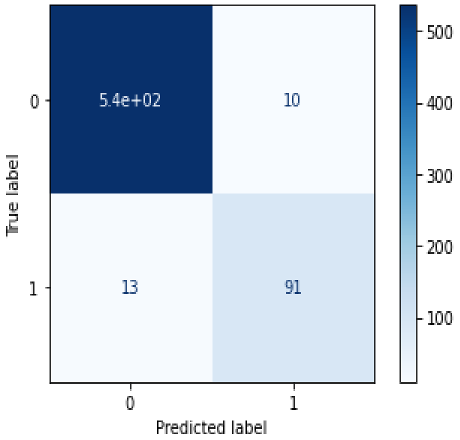
Adaboost with Gaussian Naive Bayes



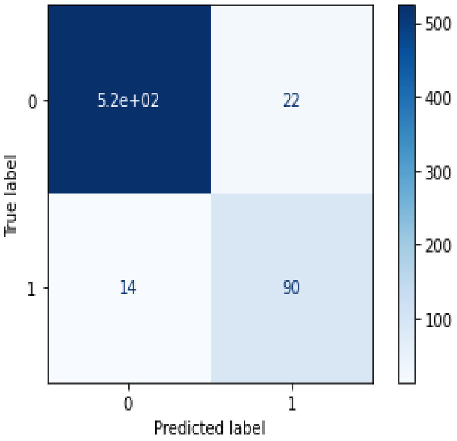
Bootstrap Aggregation



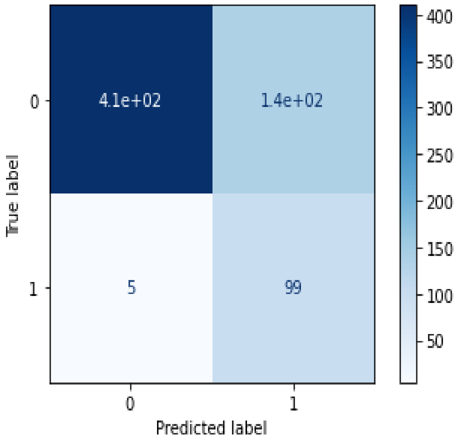
Bernoulli Naive Bayes



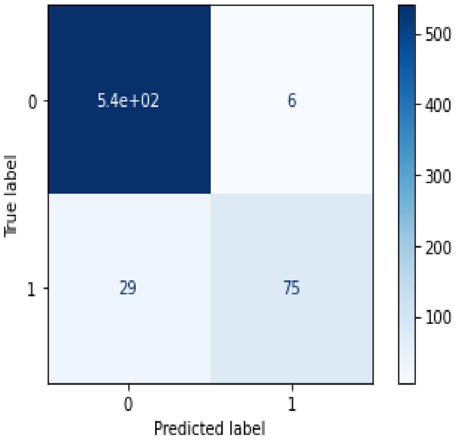
Complement Naive Bayes



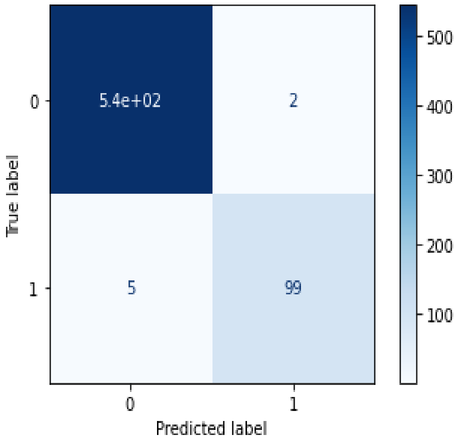
Decision Tree



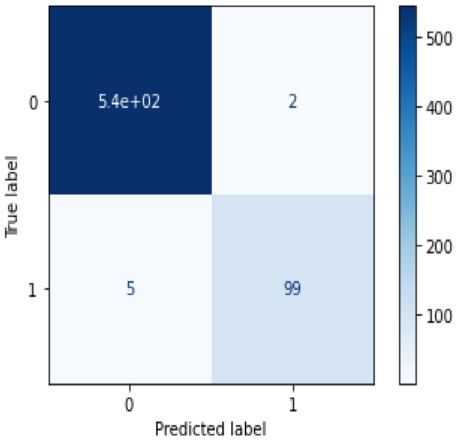
Gaussian Naive Bayes



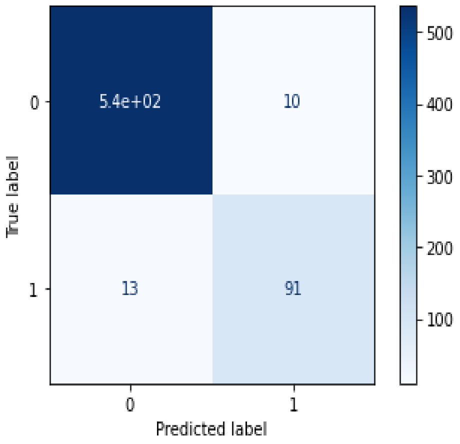
KNN



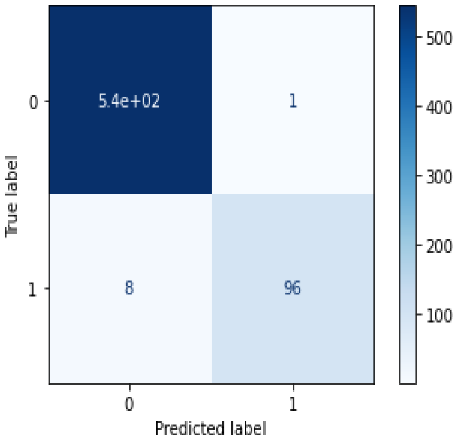
Logistic Regression



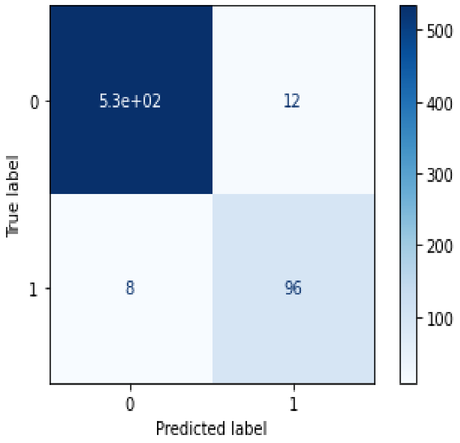
Majority Voting



Multinomial Naive Bayes



Random Forest



SVC