# Spam Mail Detector
# Processing and Classification Based on Inbox E-Mails

Shrut Patel
2015A7PS0032G

Semester 1, 2017-18

Study Oriented Project
CS F266

Submitted to Dr. Sanjay Kumar Sahay
November, 2017

# 1 Acknowledgements

I am extremely thankful to Dr. Sanjay K. Sahay for providing me with the ideas and direction which form the base of this project. His constant supervision and feedback were of tremendous help and the inputs received have shaped the project to the state it is in now.
I am sure that the experience gained through this project would aid me for such research projects in the future with the insights received during our work.

# 2 Abstract

In this project, we have created a script to classify mail into the category of non-spam (i.e. desirable) and spam (i.e. undesirable). A big data set, consisting of .eml files, also separated as non-spam and spam was used as input to algorithms which trained the computer to then make predictions on unclassified mail based on the results. The results obtained consisted of an accuracy of 83-88% on average on non-classified mail.

Along with text of the body, any found URL links are also handed over to the script for further processing and classification of mail - and it was found to enhance the accuracy of results to 88-91%. This report contains idea, methodology and statistical result of the project.

Scripts to obtain actual email source code from the web, and then their further cleaning and processing were done and then the data analysis part was coded and performed. Apart from particular domains and individual contributions, the team contributed as a whole towards the progress of the project with knowledge, criticism and novel ideas.

# Contents

# 3 Introduction

Spam mails are often a nuisance and sometimes even dangerous, where they guide a users to malicious websites to carry out cyber attacks, pass on malware, steal credentials and other nefarious activities. In this project, the team has attempted and has come up with scripts and models to detect and filter spam with nearly 90% accuracy using data extraction and machine learning. The rest of the report publishes our methodology, steps, results and various statistics at the relevant places.

# 4 Extraction of E-mail Source Code (GMAIL)

The early phase of the project was spent on extracting source code of the email in raw format from the mail inboxes directly. To do this, we wrote a script in Google Apps Script - a Google scripting language based on JavaScript. It contains the APIs required to extract and download mails directly from GMAIL. inboxes.

```
function getUrlFromGmail() {
    var raw_source;
    var threads \= GmailApp.getInboxThreads(0,1);
    // get first thread
    for(var i = 0; i < threads.length; i++){
        var messages = threads[i].getMessages();
        // get all messages in the thread
        for(var j =0; j< messages.length; j++){
            var content = messages[j].getRawContent();
            //get the raw html content from the mail
            raw_source+=content;
        }
    }
    return str(raw_source);
}

function downloadRawSource(){
       return ContentService.createTextOutput(
       getUrlFromGmail()).downloadAsFile(”emails.txt”);
}
```

Once we were convinced that mails can be obtained easily through the mentioned method, we shifted focus to the processing of data for implementing the spam filter. The method to obtain e-mails through Gmail does require more work to be completely robust, i.e. it is not completely self sufficient. The knowledge that e-mails can be obtained in real time through Google Apps Script was considered sufficient for the implementation for the project.

# 5 Pre-Processing

For training and testing purposes, a bulk of e-mails as a dataset was obtained from an online repository, reference of which is given at the end of the report.

The data set contains two folders: "TRAIN" and "TEST". The TRAIN folder contains mails separated in folder according to spam and non-spam mails. The TEST folder is for testing the result of the machine learning algorithm implemented and to gauge the accuracy.

| Folder | SPAM | NON-SPAM |
|---|---|---|
| Train | 1214 | 1432 |
| Test | 1185 | 1369 |

The table shows the number of e-mails present in the used data-set.

The data set of spam and non spam mails that was retrieved cannot be used for classification until the mails in the data set are pre-processed, therefore, the nltk package has been utilized to perform this part of the project. Also, visualisation tools such as matplotlib had been used extensively to have an overview of our obtained dataset.

Pre-processing of text data involves:-

1. Stop word removal
2. Stemming
2. Vectorizing the bag of words

# 6 Probabilistic Algorithm on E-mail Content

Henceforth, I used this pre-processed training data set for classification based on email content. He has used a Naive Bayes classifier for content based classification. Naive Bayes is a simple but powerful algorithm for predictive modeling and is particularly suited when the dimensionality of the inputs is high.

Naive Bayes classifier had been used to predict based on either, Word presence and Word frequency methods. In Word presence method the probability is calculated using the presence of word in data. While in Word frequency method probability is calculated using the frequency of word in data. We then save the trained model into a JSON file which is then to be loaded for the purpose of running the test dataset. The whole purpose is to reduce the number of times the srcipt has to create the model where it had been unnecessary.

To Handle the case of zero probability values for a event we have used **Laplace Smoothing**. This is a technique for parameter estimation which accounts for unobserved events. It is more robust and will not fail completely when data that has never been observed in training shows up.

| Model | Accuracy |
|---|---|
| Word Presence | 88.05% |
| Word Frequency | 83.71% |

## 7 Supervised Multivariate Classifications for the URL Analysis

For the URL based classification, we have used Supervised Multivariate Classifications Linear Support Vector Machine and the Random Forest Algorithm.In order to perform Supervised Multivariate Classifications we need to select features from the data.
There are 9 features selected to be used for classification.

1 Host count
2 Average token length
3 URL length
4 IP address
5 Dots count
6 Critical words
7 Host rank
8 Host country
9 token count

The function to calculate host rank is shown below

```
def popularity(url):
  temp_url = urlparse(url)
  host = temp_url.netloc
  xmlpath = 'http://data.alexa.com/data?cli=10&dat=snbamz&url=' + host
  print xmlpath
  try:
      xml = urllib2.urlopen(xmlpath)
      dom = minidom.parse(xml)
      rank_host = -1
      for sub in dom.getElementsByTagName('REACH'):
          if sub.hasAttribute('RANK'):
              rank_host = sub.attributes['RANK'].value
      for sub in dom.getElementsByTagName('COUNTRY'):
          if sub.hasAttribute('RANK'):
              rank_country = sub.attributes['RANK'].value
      return [rank_host, rank_country]
```

```
    except:
        return [-1, -1]
```

Below is the number if urls that have been scraped from the emails.

| Folder | SPAM | NON-SPAM |
|---|---|---|
| Train | 2430 | 4452 |
| Test | 1628 | 2651 |

## 7.1 Support Vector Machine

Support Vector Machine(SVM)is one of the most efficient machine learning algorithms to classify data based on either a priori knowledge or statistical information extracted from raw data.

SVM is defined by a convex optimization problem (no local minima) for which there are efficient methods SMO.It is an approximation to a bound on the test error rate, and there is a substantial body of theory behind it which suggests it should be a good idea.

As we have multiple features it is our advantage to use SVM algorithms as SVM uses Kernel methods which works well with multidimensional data.

## 7.2 Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of over fitting to their training set.

The advantages of random forest are:

1.It is one of the most accurate learning algorithms available. For many data sets, it produces a highly accurate classifier.<br>
2.It runs efficiently on large databases.<br>
3.It can handle thousands of input variables without variable deletion.<br>
4.It gives estimates of what variables are important in the classification.<br>
5.It generates an internal unbiased estimate of the generalization error as the forest building progresses.<br>
6.It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.

## 7.3 Performance

It can be seen the support vector machine, had both a better accuracy and precision than the random forest algorithm suggesting that for a multi-dimensional classification like this a SVM would yield a better model.

| Model | Accuracy | Precision |
|---|---|---|
| LinearSVM | 91.23% | 0.769 |
| Random Forest | 88.89% | 0.708 |

## 8 Conclusion

In this report, design and implementation of a multi-equipped spam detection software is presented. Techniques from domains such as Information Retrieval, Data Mining and Machine Learning have been used to accomplish the goal of detecting spam mails and phishing URL. The system was checked against a set of 2554 emails with a near average success rate of 90%.
The study project taken under Dr. Sahay has been a tremendous learning experience for me. Along with learning about text prcoessing and computer-aided predictions using machine learning, we also learnt about various methodologies used in the field of cyber security and electronic communications.

## 9 References

https://archive.ics.uci.edu/ml/machine-learning-databases/0032 (Dataset)

Mustafa Aydin, Nazife Baykal. Feature Extraction and Classification Of Phishing Websites Based on URL. IEEE (2015)