Email Spam Detection

Submitted by:

Yatika Taneja

ACKNOWLEDGMENT

.

I am using this opportunity to express my gratitude to everyone who supported me throughout the course of this project. I am thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project.
I wish to thank, all the faculties in data trained academy as this project utilized knowledge gained from every course that formed the Data science program.

INTRODUCTION

Objective of the study

Spam email is one of the biggest issues in the world of internet. Spam emails not only influence the organizations financially but also exasperate the individual email user. This study aims to build a Spam Mail Detection System by implementing various machine learning algorithms and comparing them in terms of precision, recall, accuracy, f-measure, true negative rate, false positive rate and false negative rate

Business Model

Email system is one of the most effective and commonly used sources of communication. The reason of the popularity of email system lies in its cost effective and faster communication nature. Unfortunately, email system is getting threatened by spam emails. Spam emails are the uninvited emails sent by some unwanted users also known as spammers with the motive of making money. The email users spend most of their valuable time in sorting these spam mails. Multiple copies of same message are sent many times which not only affect an organization financially but also irritates the receiving user. Spam emails are not only intruding the user's emails but they are also producing large amount of unwanted data and thus affecting the network's capacity and usage. In this paper, a Spam Mail Detection system is proposed which will classify email data into spam and ham emails. The process of spam filtering focuses on three main levels: the email address, subject and content of the message.

All mails have a common structure i.e. subject of the email and the body of the email. A typical spam mail can be classified by filtering its content. The process of spam mail detection is based on the assumption that the content of the spam mail is different than the legitimate or ham mail

Analytical Problem Framing

Data Sources

The sample data is provided to us from our client database. The dataset has 2893 observations and 3 features. In this case, Label ‘1’ indicates Spam Emails, while, Label ‘0’ indicates Ham (not spam) Emails.

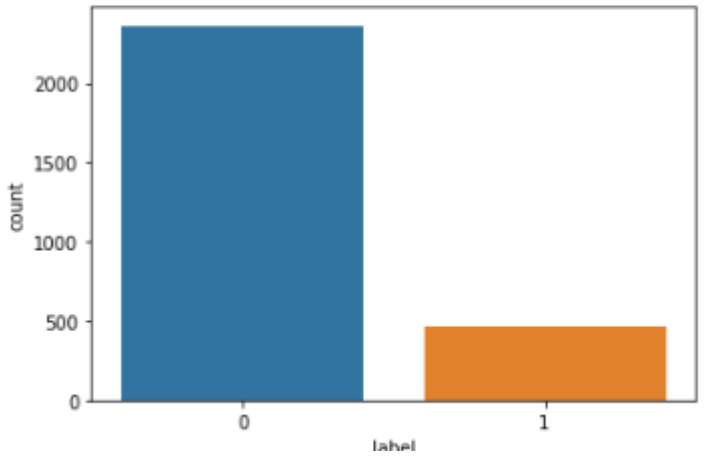
The dataset is imbalanced. Label ‘0’ has 83% records, while, label ‘1’ has approximately 17% records.

```
print('spam ratio = ',round(len(df[df['label']==1])/len(df.label),2)*100,'%')
print('ham ratio = ',round(len(df[df['label']==0])/len(df.label),2)*100,'%')
```

```
spam ratio =  17.0 %
ham ratio =  83.0 %
```

```
sns.countplot(x='label',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x173b9447048>
```



Data Preprocessing

The fundamental steps involved in data pre-processing are,:

1. Cleaning the raw data
2. Tokenizing the cleaned data

➢ Cleaning the Raw Data

This phase involves the deletion of words or characters that do not add value to the meaning of the text. Some of the standard cleaning steps are listed below:

- Lowering case
- Removal of special characters
- Removal of stop words
- Removal of hyperlinks
- Removal of numbers
- Removal of whitespaces

Lowering Case

Lowering the case of text is essential for the following reasons:

- The words, 'TEXT', 'Text', 'text' all add the same value to a sentence
- Lowering the case of all the words is very helpful for reducing the dimensions by decreasing the size of the vocabulary

```
df['full_message']=df['full_message'].str.lower()
```

Removal of special characters

```
#remove punctuation
df['full_message']=df['full_message'].str.replace(r'[^\w\d\s]',' ')

#replace whitespace with a single space
df['full_message']=df['full_message'].str.replace(r'\s+',' ')

#remove leading and trailing whitespace
df['full_message']=df['full_message'].str.replace(r'^\s+|\s+?$','')
```

```
#Replace email adress with email
df['full_message']=df['full_message'].str.replace(r'^.+@[^\.].*\.[a-z]{2,}$','emailaddress')

#repace urls with 'webaddress'
df['full_message']=df['full_message'].str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$','webaddress')

#replace money symbols
df['full_message']=df['full_message'].str.replace(r'£|\$','dollers')

#replace 10 digit phone numbers with phonenumber
df['full_message']=df['full_message'].str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$','phonenumber')

#replace numbers with 'numbr
df['full_message']=df['full_message'].str.replace(r'\d+(\.\d+)?','numbr')
```

Removal of stop words

Stop words are commonly occurring words in a language like ‘the’, ‘a’, and so on.

Most of the time they can be removed from the text because they don’t provide valuable information.

```
#remove stopwords
import string
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english')+['u','ur','4','2','in','dont','doin','ure'])
df['full_message']=df['full_message'].apply(lambda x: ' '.join(term for term in x.split() if term not in stop_words))
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\YaTiKa\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

- Tokenizing the Cleaned Data

Tokenization is the process of splitting text into smaller chunks, called tokens.

Each token is an input to the machine learning algorithm as a feature.

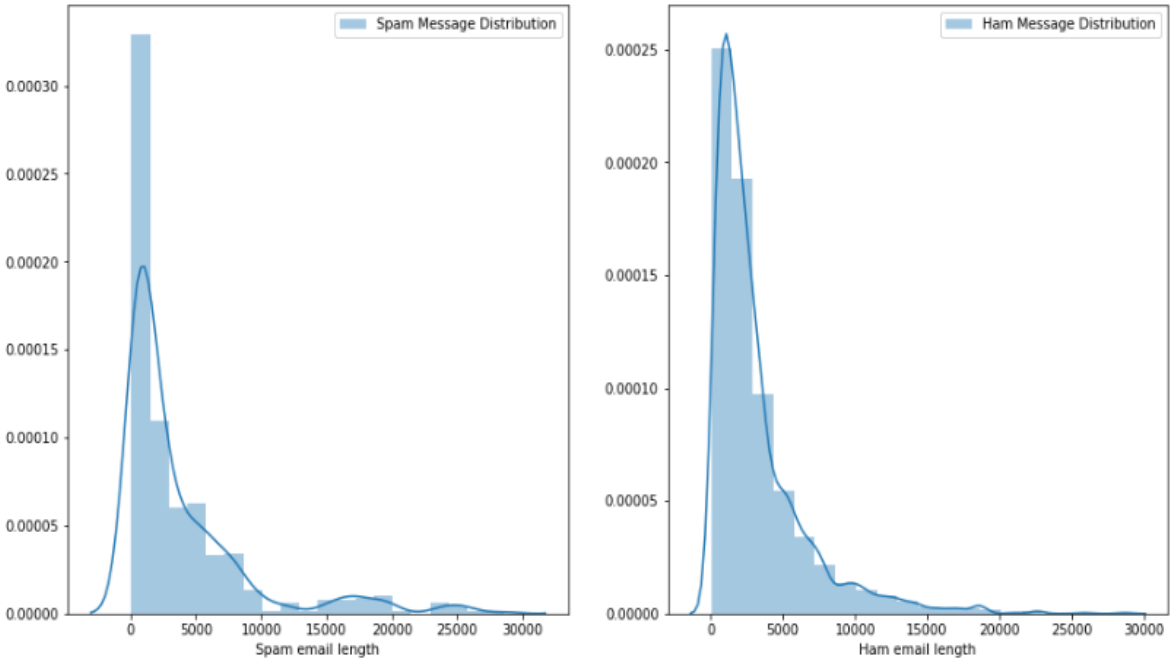
```
from sklearn.feature_extraction.text import TfidfVectorizer
tf_vec = TfidfVectorizer()
features = tf_vec.fit_transform(df['full_message'])
X = features
y=df['label']
```

Tools Used

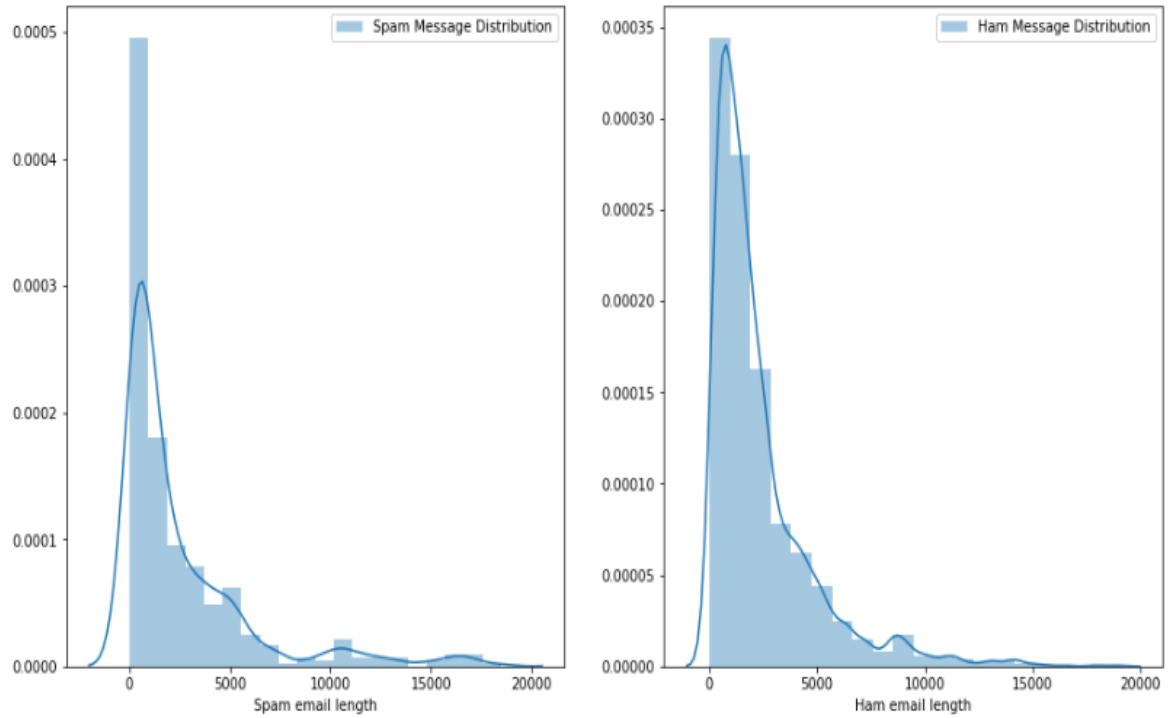
Python 3.7.2,
Jupyter Notebook
Numpy
Pandas
Matplotlib
Seaborn
Scikit-learn
Scipy
NLTK

Data Visualization

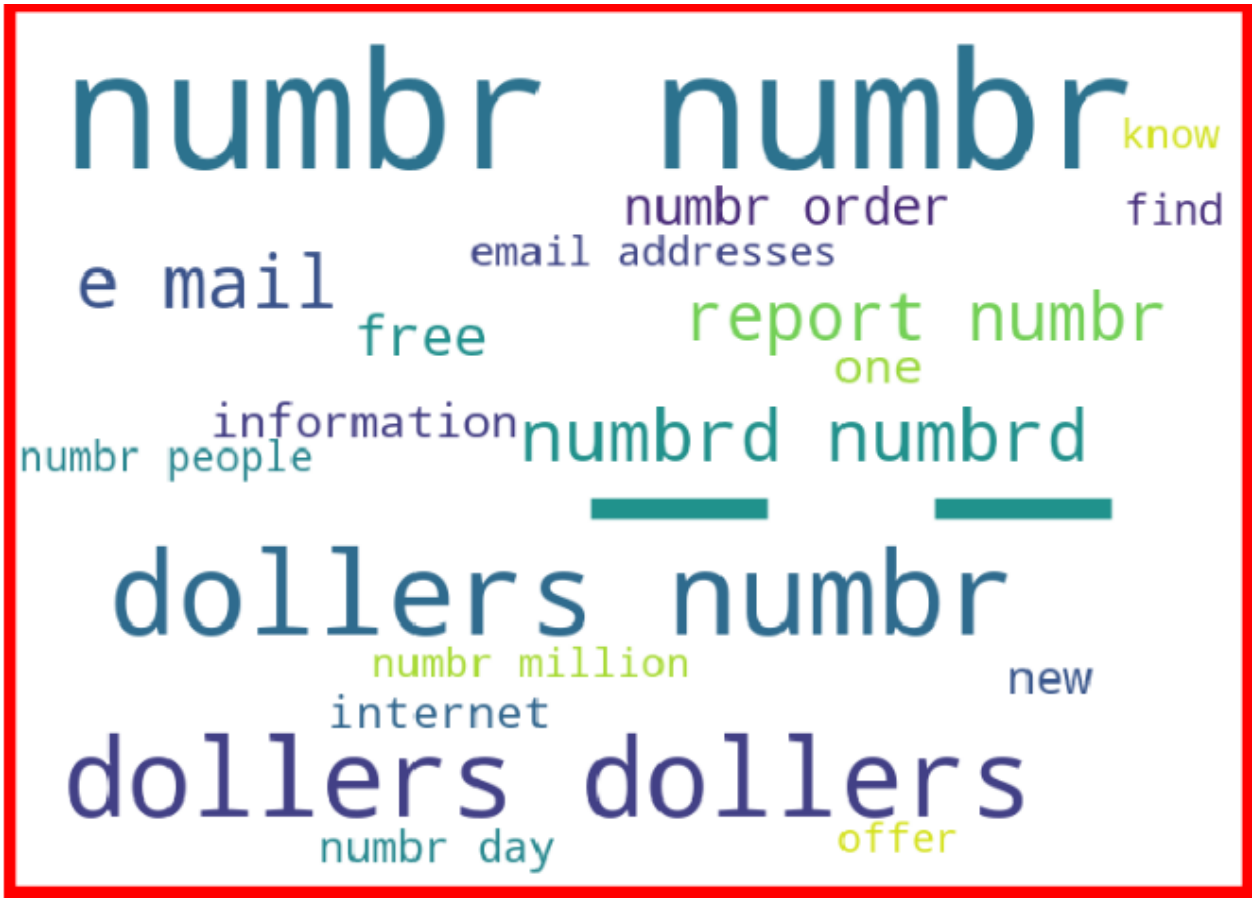
1. Message distribution before cleaning



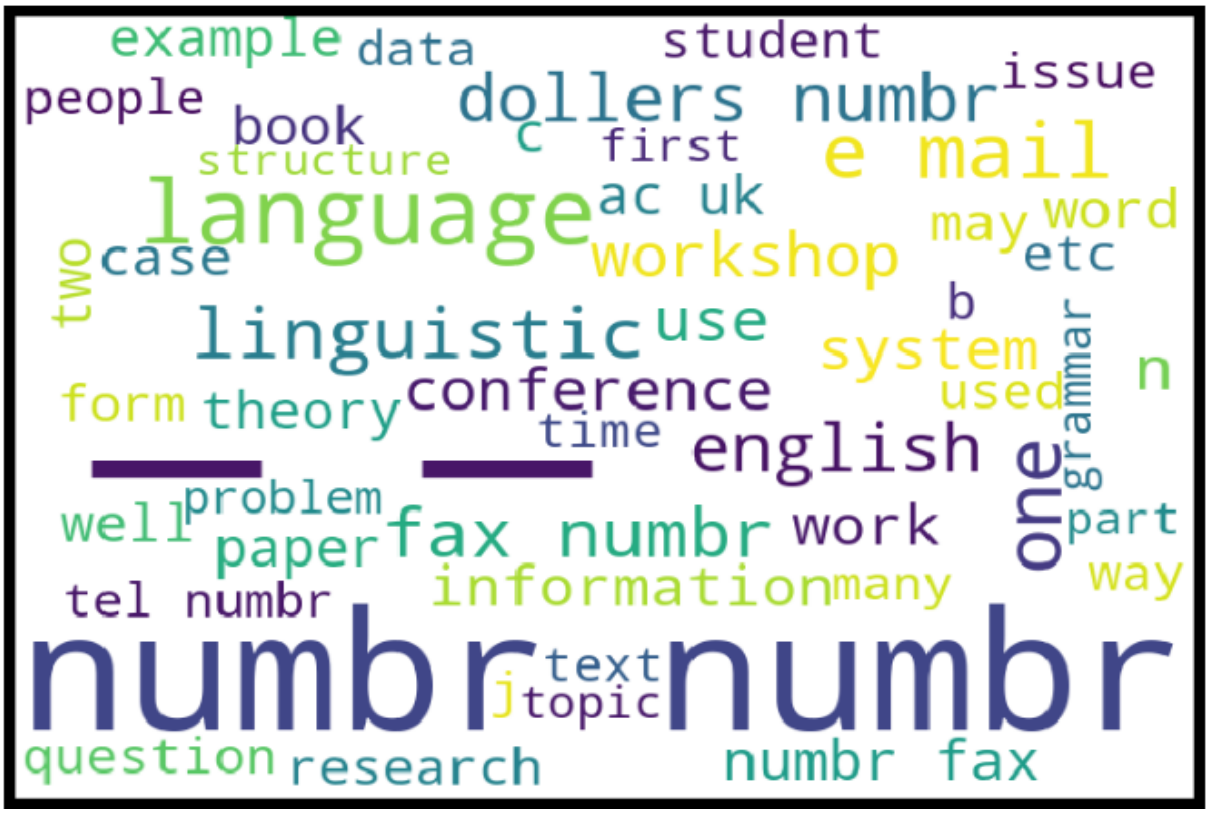
2. Message distribution after cleaning



3. Spam Word Cloud



4. Ham Word Cloud



Model/s Development and Evaluation

Models Applied

Email spam detection is a classification problem. Some algorithms like Naive Bayes Classifier, Decision Trees work well for spam detection. Algorithms like KNN, Linear Regression don't really work well due to inherent disadvantages such as curse of dimensionality.

Naive Bayes

Naive Bayes is the easiest classification algorithm (fast to build, regularly used for spam detection). It is a popular (baseline) method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features.

Why use Naive Bayes?

1. NB is very simple, easy to implement and fast because essentially you're just doing a bunch of counts.
2. If the NB conditional independence assumption holds, then it will converge quicker than discriminative models like logistic regression.
3. NB needs works well even with less sample data.
4. NB is highly scalable. It scales linearly with the number of predictors and data points.
5. NB can be used for both binary and multi-class classification problems and handles continuous and discrete data .
6. NB is not sensitive to irrelevant features.

Decision Trees

Decision trees are used for classification and regression. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. The decision of making strategic splits heavily affects a tree's

accuracy. The decision criteria is different for classification and regression trees. Information theory is a measure to define this degree of disorganization in a system known as Entropy. If the sample is completely homogeneous, then the entropy is zero and if the sample is an equally divided (50% – 50%), it has entropy of one. It chooses the split which has lowest entropy compared to parent node and other splits. The lesser the entropy, the better it is.

Why use Decision Trees?

1. Decision trees implicitly perform variable screening or feature selection. When we fit a decision tree to a training dataset, the top few nodes on which the tree is split are essentially the most important variables within the dataset and feature selection is completed automatically
2. Decision trees are easy to understand, easy to represent visually and easy to communicate.
3. Nonlinear relationships between parameters do not affect tree performance. Also trees can explain the non-linearity in an intuitive manner.

Random Forest

Random forest is like bootstrapping algorithm with Decision tree (CART) model. Random forest tries to build multiple CART model with different sample and different initial variables. For instance, it will take a random sample of 100 observation and 5 randomly chosen initial variables to build a CART model. It will repeat the process 10 times and then make a final prediction on each observation. Final prediction is a function of each prediction. This final prediction can simply be the mean of each prediction. Random forest gives much more accurate predictions when compared to simple CART/CHAID or regression models in many scenarios. These cases generally have high number of predictive variables and huge sample size. This is because it captures the variance of several input variables at the same time and enables high number of observations to participate in the prediction.

Model Evaluation

Metrics Used:

Following are the metrics we used to evaluate the performance of ML techniques:

1. Precision

Precision refers to the closeness of two or more measurements to each other. In Machine Learning, precision is the fraction of relevant instances among the retrieved instances. Precision = TP / (TP + FP) (Where TP = True Positive, TN = True Negative, FP = False Positive, FN = False Negative).

2. Accuracy

Accuracy refers to the closeness of a measured value to a standard or known value. Accuracy = (TP+TN) / ALL

3. Recall

Recall is how many of the true positives were recalled (found), i.e. how many of the correct hits were also found. Recall = TP / (TP + FN)

4. F-Score

F-scores are a statistical method for determining accuracy accounting for both precision and recall. It is essentially the harmonic mean of precision and recall.

Comparison of Performance of the Models:

1. Naive Bayes

For this problem, Naive Bayes algorithm did not work well. Following were the confusion matrix and scores:

	precision	recall	f1-score	support
0	0.85	1.00	0.92	584
1	1.00	0.14	0.24	124
accuracy			0.85	708
macro avg	0.92	0.57	0.58	708
weighted avg	0.87	0.85	0.80	708

High accuracy suggests that the model is very good at correctly classifying the mails as ham or spam. Precision value is also good at 0.85, means the model has a low false positive rate. But the model has a very low recall value. This indicates that results are complete to a large extent. In other words, probability of detection is low. Together, having a high precision and low recall means that the most of ham predictions are correct, but the model is not predicting all the ham in the test data.

3. **Decision Tree**

	precision	recall	f1-score	support
0	0.97	0.98	0.98	584
1	0.91	0.87	0.89	124
accuracy			0.96	708
macro avg	0.94	0.93	0.93	708
weighted avg	0.96	0.96	0.96	708

Accuracy, Precision, Recall and F-Score of the decision tree are higher than Naive Bayes model. Good recall of 0.87 means predictions of decision tree are more complete compared to Naive Bayes. Good precision of 0.97 indicates that model has low false positive rate. High F-Score indicates a very good model in terms of precision and recall both.

4. **Random Forest**

	precision	recall	f1-score	support
0	0.97	1.00	0.98	584
1	1.00	0.85	0.92	124
accuracy			0.97	708
macro avg	0.99	0.93	0.95	708
weighted avg	0.98	0.97	0.97	708

Random Forest algorithm had a highest precision, accuracy, recall and f-score out of 3. Having high precision and recall suggests that the model is correctly predicting positive class (ham) and also capturing most ham in the test data. It follows that the model also has a highest F-score, since it is directly proportional to Precision and Accuracy.

Result:

Since Random Forest Classifier have highest accuracy, precision, recall and f-score. It is the best fit for Email Spam Detection.

CONCLUSION

In this study, a Spam Mail Detection system is introduced which makes use of a NLP and Machine Learning for its implementation. The classification algorithms used in this approach are Naïve Bayes, Decision Tree Classifier and Random Forest Classifier. It was observed that Random forest Classifier is the best fit for Spam Mail Detection System with 97% accuracy and 0.98 f1-score.

..