Arsadur Rahman
AIT 580
Assignment – Spam or Ham
November 21, 2018

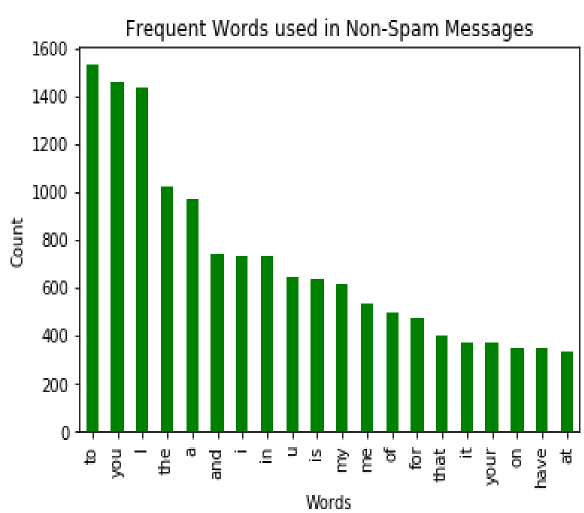
After downloading the spam dataset from Kaggle, I decided to use Naive Bayes Model to build a spam detection model. In this assignment I am mainly dealing with text data and in my understanding Naïve Bayes works better in terms of prediction to classify certain types, in this case detecting spam depends on conditional probability of having certain words or lengths or character to an email. Naïve ayes predict the unknown dataset class using probability, so let's dive in.

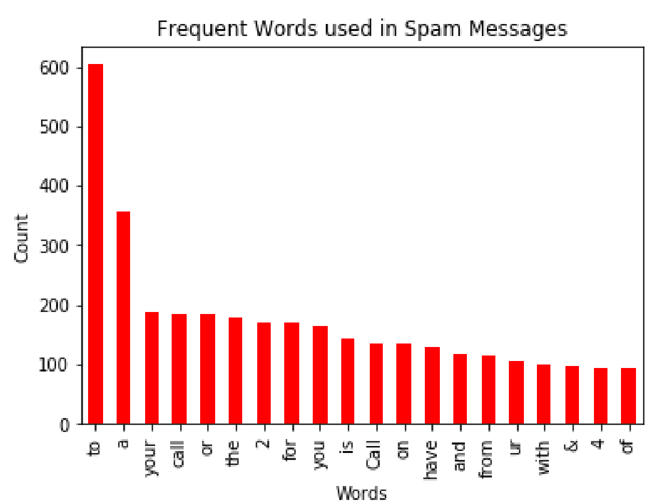
In the beginning of the process, after importing all the necessary libraries in python such as pandas, numpy, matplotlib.pyplot, seaborn, using different features and loading the data, everything was set to build the model.

I started with eliminating unnecessary columns and rows and renaming the column header using data cleaning technique. In the original dataset I have found 3 unnecessary columns and I had to remove that and rename the column header V1 to class and v2 o messages for better text analysis. Using python coding which is attached in a different file associated to this assignment, the dataset looked like this

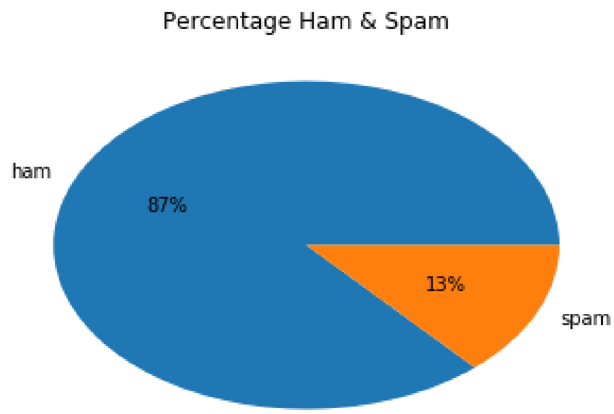
| | class | message |
|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... |
| **1** | ham | Ok lar... Joking wif u oni... |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... |
| **3** | ham | U dun say so early hor... U c already then say... |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... |

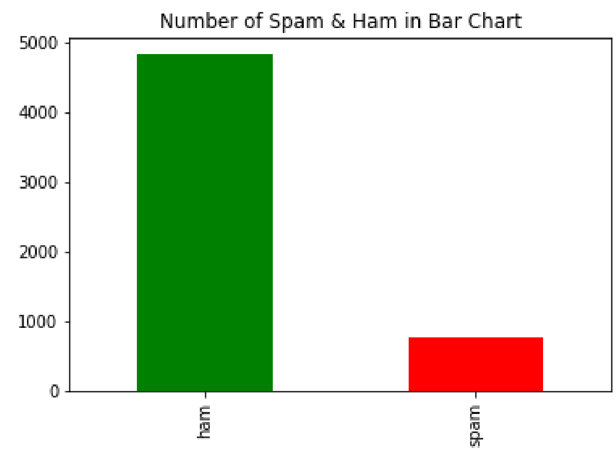
The table above shows the new look of dataset after initial polishing. I went ahead to explore more with the data set, such how many ham and spam are in the data set, what's the percentage, which words were frequently used in spam emails and ham emails, using python matplotlib.Pyplot and pie chart. The graphs are shown below.



Arsadur Rahman
AIT 580
Assignment – Spam or Ham
November 21, 2018




The 4 graph shows a big picture of the dataset, we can see the dataset has 13% of spam emails and 87% of ham emails, and the total number of class and messages are 5572. Looking at the percentage I could not use accuracy to classify the labels because if a classifier returns hap the accuracy will be 87% which may not be the case for an outcome, so I need to use F-1 score which would be discussed later. The first two graph shows which words were frequently used in ham and spam emails. (code for the graphs are attached in a different file)

After exploring the data, I did some cleaning with the data, during the cleaning process I found no missing data and o other classes rather than ham and spam. Cleaning and normalizing took a lot of time, to train the dataset for future perdition. I have altered the ham and spam class with "0" and "1" using lambda function and this helped to prepare the dataset for the training purpose.

| | class | message | label |
|---|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... | 0 |
| **1** | ham | Ok lar... Joking wif u oni... | 0 |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | 1 |
| **3** | ham | U dun say so early hor... U c already then say... | 0 |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | 0 |

To normalize the texts, I needed to remove the stop words (words that are frequently used in English language, e.g. I, to, yes, no. you etc.), take out the punctuations, and reduced words to its root word. Importing string from python libraries the text normalization was done, also I imported "stopwords" from nltk.corpus and prterstemmer from nltk as Stemmer to remove the punctuations, stopwrods, etc. Once, the normalization was done the data frame was ready. At this stage I needed to convert messages to vectors so that it can be used to train the existing data for future prediction using Naïve Bayes Model. Using bag of words model, as in how many words and what kinds of words are usually we see in spam or ham emails. I used Term Frequency(TF) and Inverse Document Frequency(IDF) vectorizer feature where TF deals with how many times a term occurred in a dataset and on the other hand IDF deals with the significant terms in the bag of words. This whole process will help train the data to predict future emails as spam or ham. Using the TfidfVectorizer feature from sklearn.feature_extraction the messages was transformed in the form of vector.

At this stage I needed to use pipeline feature of sklearn so that the TfidfVectorizer can transfer it to Naïve Bayes classifier. Now the dataset was ready for the training by importing MultinomialNB from sklearn.naive_Bayes and using train_test_split .

```
0     [go, jurong, point, crazi, avail, bugi, n, gre...
1                        [ok, lar, joke, wif, u, oni]
2     [free, entri, 2, wkli, comp, win, fa, cup, fin...
3         [u, dun, say, earli, hor, u, c, alreadi, say]
4     [nah, dont, think, goe, usf, live, around, tho...
5     [freemsg, hey, darl, 3, week, word, back, id, ...
6     [even, brother, like, speak, treat, like, aid,...
7     [per, request, mell, mell, oru, minnaminungint...
8     [winner, valu, network, custom, select, receiv...
9     [mobil, 11, month, u, r, entitl, updat, latest...
10    [im, gonna, home, soon, dont, want, talk, stuf...
11    [six, chanc, win, cash, 100, 20000, pound, txt...
12    [urgent, 1, week, free, membership, â£100000, ...
13    [ive, search, right, word, thank, breather, pr...
14                                     [date, sunday]
15    [xxxmobilemovieclub, use, credit, click, wap, ...
16                                   [oh, kim, watch]
17    [eh, u, rememb, 2, spell, name, ye, v, naughti...
18    [fine, thatåõ, way, u, feel, thatåõ, way, gota...
19    [england, v, macedonia, dont, miss, goalsteam,...
Name: message, dtype: object
```

```
index   idf     tfidf   term
72      8.5271  0.2330  08452810075over18
413     3.6544  0.0999  2
420     8.2394  0.2252  2005
433     8.2394  0.2252  21st
833     8.0163  0.2191  87121
1180    6.0993  0.1667  appli
2076    7.1408  0.1952  comp
2246    7.4285  0.2030  cup
2748    6.5346  0.3572  entri
2868    8.5271  0.4661  fa
2969    6.0993  0.1667  final
3091    4.2096  0.1151  free
4592    5.8190  0.1590  may
5768    8.5271  0.2330  questionstd
5815    8.5271  0.2330  ratetc
5856    5.8645  0.1603  receiv
6959    4.3027  0.1176  text
7099    8.0163  0.2191  tkt
7276    4.5137  0.1234  txt
7708    5.2950  0.1447  win
7741    6.9176  0.1891  wkli
```

The F-1 formula follows as: F-1 Formula is a function of recall and precision. Where recall and precision deals true positive, false positive, and false negative. F-1 score used to find balance between recall and precision. F-1 is used when the class distribution is uneven, which in our case the distribution is uneven giving spam only 13% and ham 87%. Accuracy always not the best model in uneven case so the F-1 score can help us with avoiding the effects of large number of true negatives.

$$\text{F1} = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

Using the classification_report library from sklern.metrics, I find he average f-1 score for the model is .96

```
             precision    recall  f1-score   support

        ham       1.00      0.95      0.98      1006
       spam       0.70      0.99      0.82       109

avg / total       0.97      0.96      0.96      1115
```

***Python version 3.7