**Introduction and Data Processing**

This paper will include the analysis and classification of emails as spam or ham. We loaded in data from 2 separate files within the Spam Corpora File and included 1500 spam and 1500 emails for further analysis.

New number of spam files: 1500

New number of ham files: 1500

Initially we ran into some challenges getting the code to run by accessing the terminal because of some subtle differences between PC and Mac. After successfully getting the data to load, we were trying to incorrectly use the emaildocs list and therefor getting an emaildocs not defined error when trying to define our bag of words feature. To solve for this we defined a get\_emaildocs function that loads the tokenized text and spam/ham labels from emails in both spam texts and ham texts into one list and then randomizes it. We then created a function to define our bag of words feature with unigrams.

**Word Frequency**

We used the nltk word frequency for each spam and ham words. The word frequency is a lot of punctuation and a few words like the, to, and a that were common across ham and spam emails. One thing we noticed is the @ symbol came up in ham, but not spam. Since you typically see @ in email addresses, we expected to see this in both.

ham\_dist = nltk.FreqDist(ham\_words)

spam\_dist = nltk.FreqDist(spam\_words)

ham spam

('-', 33560) ('.', 19517)

('/', 15466) (',', 11251)

('.', 14286) ('-', 7942)

(',', 11988) ('the', 7297)

(':', 10303) (':', 6114)

('the', 8115) ('/', 5707)

('to', 6631) ('to', 5173)

('ect', 5684) ('and', 4917)

('@', 4496) ('of', 4511)

('and', 3500) ('?', 4107)

('for', 3490) ('a', 3794)

('hou', 2968) ('in', 3147)

('enron', 2599) ('=', 3091)

('on', 2568) ('you', 2797)

('a', 2563) ('for', 2526)

For the model evaluation, we created a cross validation function that loads in a feature set and number of folds to be used and then splits into training and testing sets and run naïve bayes classifier. We then ran this function with our bag of words feature set and applied 10 folds. Instead of dividing up the code As a team we did this project with paired programming working sessions via Zoom.

**Model Evaluation Analysis**

We had excellent results from our model evaluation. We had 10 folds of our training data with an average accuracy of 96.7%, with the highest accuracy at 99% for Fold 9 and Fold 0 and Fold 2 on the lower end at 95%. This further validates the importance and usefulness of the K folds method. Had we done just the initial training sample we would have ended up with a model that was much less accurate. While overall our accuracy is high, looking into our confusion matrix our model there was a slight discrepancy between our spam/ham accuracies. Only 2 spam emails were incorrectly identified as ham, while 97 ham emails were incorrectly identified as spam. If we were to continue this study, it would be worth looking into those 97 emails and understanding what was causing the discrepancy.

Accuracy for Fold 0: 0.95

Accuracy for Fold 1: 0.97

Accuracy for Fold 2: 0.95

Accuracy for Fold 3: 0.9866666666666667

Accuracy for Fold 4: 0.97

Accuracy for Fold 5: 0.9566666666666667

Accuracy for Fold 6: 0.97

Accuracy for Fold 7: 0.97

Accuracy for Fold 8: 0.9566666666666667

Accuracy for Fold 9: 0.99

---Average Accuracy: 0.967---

Precision Recall F1

ham 0.935 0.999 0.966

spam 0.999 0.939 0.968

| s |

| h p |

| a a |

| m m |

-----+-----------+

ham |<1403> 97 |

spam | 2<1498>|

-----+-----------+

(row = reference; col = test)

| s |

| h p |

| a a |

| m m |

-----+---------------+

ham | <46.8%> 3.2% |

spam | 0.1% <49.9%>|

-----+---------------+

(row = reference; col = test)