**Email Spam Classification**

Mohammad Abubakar (15L-4155), Nouman Ali (15L-4160), Talha Akbar (15L-4283)

***Abstract*—Email classification especially that of spam, is a very important topic of text classification. A lot of research has been done, employing various machine learning techniques. In our project we will perform feature extraction on the Enron corpus and then use the Naïve Bayes and Support Vector Machine Algorithms for classification of emails into two classes: spam or ham. The peak accuracy achieved was approx. 97 percent.**

***Keywords—head pose estimation; neural network; head pose classification, face orientation***

# Introduction

Classification of emails is an important topic area of research in text classification because there is great need for it as the advancement of internet and electronic communication has led to increased volume of emails flowing into our mail boxes. The most common dataset that is used for building email classification models is that of Enron Corpus and that is what will be used in this project. A word dictionary will be created from the content of the emails which will then be then used for feature extraction from the emails. After these two processes the Multinomial Naïve Bayes and Support Vector Machine Algorithm from sci-kit learn will be used to create models and classify the emails into spam or ham. Both these algorithms have been widely used for email classification.

This article is framed as follows. Section II reviews Related Work, Section III describes the creating dictionary feature extraction and the algorithms used, and Section IV explains the experimental setup, dataset and results. Section V conclude this document.

# Literature Review

[1] In exploration of the Enron dataset, Bryan Klimt and Yiming Yang basically tried to research email folder prediction. SVM was used first to classify the folder of each email based solely on a particular field of data from the email. The fields used were “From”, “Subject”, “Body”, and “To, CC”. The weights for each section were learned for each folder of a particular user, using ridge regression on the training data. They obtained thresholds using score-based local optimization, known as SCut, and evaluated using F1 scores, which measure both precision and recall. The results indicated that the number of messages a user has is clearly not strongly correlated with the performance of the text classifier on his or her email. The result was reasonable, though. If a user has many messages, but they are all in the same folder, classification is trivial.

[2] In their research Mehran Sahami, Susan Dumais, David Heckerman, Eric Horvits have used the Bayesian approach to filter out unwanted spam email. Bayesian filter hast be trained to work effectively. Every word had a certain probability of occurring in spam or ham email in the database. The technique used was that if the total of words probabilities exceeds a certain limit, the filter will mark the e-mail to either two categories that are: spam or ham. The results showed that modeling the subcategories for applying Bayes made the results only worse. The accuracy was highest on words only feature.

# Problem Statement

The problem being addressed in this project is that of classifying emails into two categories: spam or ham. A lot of email passes through the email system that is unnecessary and repetitive and it is a very basic requirement to filter out that set of emails. This project intends to create working models which are trained on the labeled and then eventually classify the emails.

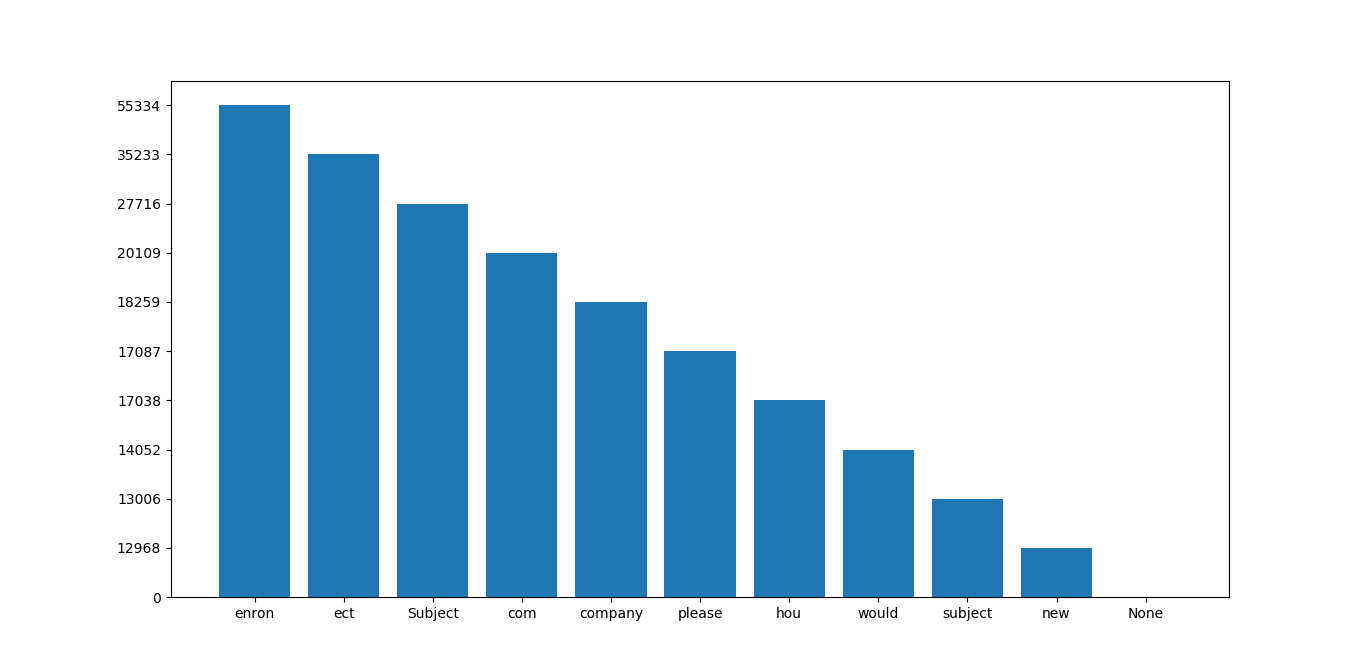
We have the Enron Corpus which basically has 33,716 emails. The data here is labeled according to the categories. The problem is to perform successful feature extraction and then use the feature matrix to classify the emails into the correct categories with a high accuracy.

# Methodology

1. **Creating Word Dictionary**

This is the very initial part of the project. A word dictionary is to be created which will contain 1000 most common words extracted from the emails. Basically all the emails were iterated and all the words were stored in an array. After that the stop words were removed through the nltk library. After that any words consisting of numbers or words that were single letters were removed. And finally the dictionary was obtained from the array.

The following figure shows the frequency of the top 10 words of the dictionary.



1. **Feature Extraction**

This is the most important part of the email classification process. Basically in the last step we obtained a dictionary which contained the 1000 most common words extracted from all the mails whether they were ham or spam. In this step we created a feature matrix which was a 2-D numpy array, having dimensions (No. of emails) x 1000. The 1000 is basically the number of words in the dictionary.

In the feature extraction process, every single directory was browsed through code to access every single mail. After the mail was opened, the content was tokenized using the nltk library. The tokenization provided us with an array of words. For every single word in the array, we iterated over the content of the dictionary to check whether if the word matched the ith word of the dictionary or not and if it did, we recorded the number of occurrences of that word in our 2-D array in the respective index. This strategy basically provided us with a 2-D array whose indexes were filled with number of instances of the dictionary’s words for every single email of the corpus.

Simultaneously, in the loops, we recorded the label of each email and stored it in an 1-D array. In conclusion, the feature extraction process was able to provide us with a 2-D feature matrix and all the labels of the train data.

1. **Classification**

This is the last process of the project. In the last step we successfully obtained a feature matrix and the labels through a feature extraction process. Basically our feature extraction process was generic so through multiple calls, we obtained a train feature matrix and train labels and also the test feature matrix and test labels.

The first classifier that was used was the Multinomial Naïve Bayes classifier. The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. This algorithm was suitable for our classification process because Naïve Bayes is one of the most widely used algorithms for text classification and our feature matrix had words counts in it so the Multinomial version was suitable. The default parameters were used for classification that is the Laplace smoothing parameter was set to 1 and class prior probabilities were not used.

The second classifier used was the Support Vector Machine Classifier. The SVC variant of the sklearn.svm was used as it allows enabling the probability estimates which is integral for text classification and has the C regularization term. It has a linear kernel and is well suited for text classification. Some parameters were tuned to achieve the proper classification.

# Experimental Setup

The project was set up in the PyCharm environment which employed the Python programming language. The code was compiled in Python 3.6.

# Dataset

The dataset used in this project is the Enron-Spam dataset which is basically a subset if the Enron Email Corpus obtained from the site aueb.gr [3]. The dataset has a total of 33,716 emails, organized into 6 folders which each in turn are divided into two folders, ham or spam. So the dataset is totally labeled into the respective categories pf ham and spam. The "preprocessed" subdirectory contains the messages in the preprocessed format that is used in the experiments of this project. Each message is in a separate text file. The number at the beginning of each filename is the "order of arrival" while the end specifies whether it is ham or spam.

The dataset had 6 folders as described above and the content of 5 folders was used as training data while the content of 1 folder was used as test data.

|  |  |  |
| --- | --- | --- |
| Email Category | Training Dataset | Test Dataset |
| Spam | 12671 | 4500 |
| Ham | 15045 | 1500 |

# Results

The results are shown in terms of accuracy in the below tables

# SVM RESULTS

|  |  |
| --- | --- |
| Parameters | Accuracy |
| svm.SVC(C=1.0, cache\_size=200, class\_weight=None, coef0=0.0,  decision\_function\_shape='ovr', degree=2, gamma='auto', kernel='rbf',  max\_iter=10, probability=False, random\_state=None, shrinking=False,  tol=0.001, verbose=False) | 96.6833333333 |
| svm.SVC(C=5.0, cache\_size=200, class\_weight=None, coef0=0.0,  decision\_function\_shape='ovr', degree=2, gamma='auto', kernel='rbf',  max\_iter=-1, probability=False, random\_state=None, shrinking=False,  tol=0.001, verbose=False) | 97.25 |
| svm.SVC(C=3.0, cache\_size=200, class\_weight=None, coef0=0.0,  decision\_function\_shape='ovr', degree=2, gamma='auto', kernel='rbf',  max\_iter=-1, probability=False, random\_state=None, shrinking=False,  tol=0.001, verbose=False) | 97.1 |
| svm.SVC() | 96.6833333333 |

# Multinomial Results

|  |  |
| --- | --- |
| Parameters | Accuracy |
| MultinomialNB() | 95.3166666667 |
| MultinomialNB(fit\_prior= False) | 95.2833333333 |
| MultinomialNB(alpha=3.0, class\_prior=None, fit\_prior=True  ) | 95.2166666667 |
| MultinomialNB(alpha=5.0, class\_prior=None, fit\_prior=True  ) | 95.1166666667 |

# Conclusion

The project was an email classification problem in identifying whether the emails were spam or not. The famous Enron Spam email dataset was used in this project. After doing the processes of feature extraction and classification using Naïve Bayes and Support Vector Machine algorithms. The results obtained after classification were very impressive as we were able to achieve a peak accuracy of 97 percent. Other research papers indicated that an accuracy of 99 percent was possible also. Our experiment can be called a success as we were able to almost accurately predict whether the emails of the corpus were spam or ham. As the results indicated Support Vector Machine algorithm performed better than the Naïve Bayes Algorithm. The results could have been improved further by refining the words present in the dictionary or more optimal number of words could have been used which would in turn would improve our feature matrix. Overall, email classification is an important part of text classification and using better algorithms will help achieve better results.

##### **References**

1. Bryan Klimt and Yiming Yang, “The Enron Corpus: A New Dataset for Email Classification Research”, European Conference on Machine Learning, 2004
2. Mehran Sahami, Susan Dumais, David Heckerman, Eric Horvits, “A Bayesian Approach To Filtering Junk E-mail”, AAAI Workshop on Learning for Text Categorization, 1998
3. http://www2.aueb.gr/users/ion/data/enron-spam/