Investigation into the Application of Personality Insights and Language Tone Analysis in Spam Classification

Student Name

Colm McGetrick

A dissertation submitted in partial fulfilment of the requirements of Dublin Institute of Technology for the degree of M.Sc. in Computing (Data Analytics)

April 2017

Declaration of Authorship

I certify that this dissertation which I now submit for examination for the award of MSc in Computing (Knowledge Management), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the test of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Dublin Institute of Technology and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institute's guidelines for ethics in research.

Signed

Date: 28th April 2017

Abstract

Due to its persistence spam remains as one of the biggest problems facing users and suppliers of email communication services. Machine learning techniques have been very successful at preventing many spam mails from arriving in user mailboxes, however they still account for over 50% of all emails sent. Despite this relative success the economic cost of spam has been estimated as high as $50 billion in 2005 (Ferris, Jennings, & Williams, 2005) and more recently at $20 billion (Rao & Reiley, 2012) so spam can still be considered a considerable problem.

In essence a spam email is a commercial communication trying to entice the receiver to take some positive action. This project uses the text from emails and creates personality insight and language tone scores through the use of IBM Watsons' Tone Analyzer API. Those scores are used to investigate whether the language used in emails can be transformed into useful features that can be used to correctly classify them as spam or genuine emails. And during the course of this investigation a range of machine learning techniques are applied.

Results from this experiment found that where just the personality insight and language tone features are used in the model some promising results with one dataset were shown. However over all datasets results were inconclusive with this model. Furthermore it was found that in a model where these features were used in combination with a normalised term-frequency feature-set no real improvement in the classification performance was shown.

Key words: Spam, classification, filtering, natural language processing, personality insights, tone analysis

Acknowledgements

I would like to express my sincere thanks to my supervisor, Sarah Jane Delany, for all her help and guidance during the dissertation process. I also want to thank my wife, parents and in-laws for all the support and encouragement you have given to me though-out the course of my studies.

Table of Contents

Declaration of Authorship i

Abstract ii

Acknowledgements iii

Table of Contents iv

Table of Figures viii

Table of Tables ix

Abbreviations x

1 Introduction 1
1.1 Background 1
1.2 Research Project 2
1.2.1 Hypotheses 2
1.3 Research Objectives 2
1.4 Research Methodology and Approach 2
1.5 Scope and Limitations 3
1.6 Outline of Thesis 4
2 Machine Learning and Spam Filtering 5
2.1 Introduction 5
2.2 Spam Definition 5
2.2.1 A Brief History of Spam 6
2.2.2 The Market for Spam 7
2.3 State of the Art 8
2.3.1 Representation 8
2.3.2 Feature Selection 10
2.3.3 Discretisation Techniques 11
2.3.4 Sampling Methods 11
2.3.5 Naïve Bayes Classifiers 12
2.3.6 Naïve Bayes: Main Variations 14
2.3.7 Support Vector Machines 15

2.3.8 Support Vector Machines: Variations 17
2.3.9 Artificial Neural Networks 18
2.3.10 Random Forests 20
2.4 Natural Language Processing in Spam Filtering 22
2.4.1 Sentiment Analysis in Spam Filtering 22
2.5 Conclusion 24
3 Methodology 25
3.1 Introduction 25
3.2 Experimental Approach 25
3.3 Datasets used in Experiments 26
3.3.1 Enron Dataset 26
3.3.2 CSDMC2010 Dataset 27
3.3.3 Spam Assassin Dataset 27
3.4 Feature Selection 28
3.5 Training and Sampling technique 28
3.6 Classification Algorithms and Parameter Settings 29
3.6.1 Algorithms and Parameter Settings for the Baseline Classifier 29
3.6.2 Algorithms and Parameter Range for Personality Insight and Tone Feature Classifiers 30
3.6.3 Parameter Setting for the Combined Classifier 30
3.7 Evaluation Criteria: Average Class Accuracy 31
3.8 Conclusion 32
4 Experiment Design 33
4.1 Introduction 33
4.2 Software Used 33
4.2.1 Python 33
4.2.2 IBM Watson Tone Analyszr API 34
4.2.3 JSON 34
4.3 Data Preparation 35
4.3.1 Normalised Term-Frequency Representation 35
4.3.2 IBM Tone Features 36
4.3.3 Emotional Tone 36
4.3.4 Social Tone 37
4.3.5 Language Tone 39

4.3.6 Creating Tone Features from the IBM Tone Analyzer API 40
4.3.7 Technical Limitations 41
4.4 Experiment Parts.....42
4.4.1 Introduction.....42
4.4.2 Baseline Classifier Design.....42
4.4.3 Baseline Classifier Design Description.....43
4.4.4 Tone Feature Classifier Design.....43
4.4.5 Tone Feature Classifier Design Description 44
4.4.6 Combined Features SVM Classifier Design 44
4.4.7 Combined Features SVM Classifier Description 45
4.5 Conclusion.....45
5 Experimentation and Evaluation.....46
5.1 Introduction 46
5.2 Comparison with a Baseline Model 46
5.2.1 Baseline Model Results 46
5.3 Tone Data Exploration & Analysis.....47
5.3.1 Introduction.....47
5.4 Tone Feature Selection.....59
5.4.1 Feature Selection: Investigation with Enron Data.....60
5.5 Spam Filtering with Personality Insight and Tone Features 61
5.5.1 Experiment Results.....61
5.6 Combined Classifier.....64
5.6.1 Combined Classifier Results 64
5.7 Conclusion.....65
6 Conclusion.....67
6.1 Introduction 67
6.2 Problem Definition and Research Overview 67
6.3 Experimentation, Evaluation and Results 67
6.4 Contribution to the body of Knowledge.....68
6.5 Limitations 69
6.6 Future Works.....69
7 Bibliography 71

Table of Figures

Figure 2-1 Percentage of Spam in all Emails 2006-2016 6
Figure 2-2 Optimal Separating Hyperplane 16
Figure 2-3 Operating mode of SVM allowing margins to be breached 17
Figure 2-4 McCullogh-Pitts Model of Neuron 18
Figure 2-5 Multi-layer feed forward network 19
Figure 2-6 Sigmoid Function for varying slope parameter (a) 19
Figure 4-1 Baseline Classifier Design Diagram 42
Figure 4-2 Tone Feature Based Classifier Design Diagram 43
Figure 4-3 Combined Feature Classifier Design Diagram 44
Figure 5-1 Most Common Words in Enron Ham Class 48
Figure 5-2 Most Common Words in Enron Spam Class 48
Figure 5-3 Most Common Words in CSDMC2010 Ham Class 49
Figure 5-4 Most Common Words in CSDMC2010 Spam Class 49
Figure 5-5 Most Common Words in Spam Assassin Ham Class 50
Figure 5-6 Most Common Words in Spam Assassin Spam Class 50
Figure 5-7 Enron Tone Features: Class-wise split of values 53
Figure 5-8 CSDMC2010 Tone Features: Class-wise split of values 56
Figure 5-9 Spam Assassin Tone Features: Class-wise split of vales 59
Figure 5-10 Sample of "auto" binning outputs for Enron tone features 60
Figure 5-11 Example of Confusion Matrices from application of SVM to Spam Assassin and CSDMC Datasets 63
Figure 5-12 Example of Confusion Matrices from application of Random Forests to Spam Assassin and CSDMC Datasets 63

Table of Tables

Table 2-1 Sentiment analysis of CSDMC emails.....23
Table 2-2 Comparison of original results with results from Polarity Classifiers 24
Table 3-1 Composition of the six Enron benchmark datasets. 26
Table 3-2 Summary of contents of Spam Assassin Dataset.....27
Table 3-3 Parameter Search Ranges for SVM Classifiers 30
Table 3-4 Parameter Search Ranges for Random Forests Classifiers.....30
Table 4-1 Description of Emotional Tone Features.....37
Table 4-2 Description of Social Tone Features.....39
Table 4-3 Description of Language Tone Features.....39
Table 4-4 Emails above technical limit of IBM Tone Analyzer.....42
Table 5-1 Baseline Classifier Average Class Accuracy Results.....46
Table 5-2 Class Split of Mean and Standard deviations of tone features: Enron.....51
Table 5-3 Class Split of Mean and Standard deviations of tone features: CSDMC 54
Table 5-4 Class Split of Mean and Standard deviations of tone features: Spam Assassin 57
Table 5-5 Feature Ranking by Mutual Information and Chi-square applied to the Enron Personality Insight and Tone Features 61
Table 5-6 Personality Insight and Tone Feature Classifier Accuracy Results for all datasets.....62
Table 5-7 Combined Classifier Results.....65

Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
ARPNET	Advanced Research Projects Agency Network
BoW	Bag of Words
CSDMC	CyberSecurity Data Mining Competition
IBM	International Business Machines Corporation
IG	Information Gain
IP	Internet Protocol
IT	Information Technology
NB	Naïve Bayes
NLP	Natural Language Processing
RBF	Radial Basis Function
RF	Random Forests
SVC	Support Vector Classifier
SVM	Support Vector Machine
UBE	Unsolicited Bulk Email
UCE	Unsolicited Commercial Email

1 Introduction

1.1 Background

Spam is a serious problem imposing an economic cost estimated at $20 billion in 2010 (Rao & Reiley, 2012). This cost is based on wasted time for email recipients, the cost of server hardware to handle the volume of spam and the opportunity cost of building spam prevention software. Due to the sheer volume of spam emails machine learning techniques are applied to try to classify them as spam or genuine (ham) in order to direct them to the email recipients' inbox or not. This is generally referred to as spam filtering, detection or classification. Research in this area has shown that supervised machine learning algorithms can be successful in spam filtering ranging from comparatively straight-forward Naïve Bayes classifiers to more complex classifiers such as artificial neural networks (ANN).

Advances in natural language processing, such as sentiment analysis, provide an ability to derive new features from email datasets, namely from the body text of emails, and investigate if these new features can help in spam detection. Recent publications (Ezpeleta, Zurutuza, & Hidalgo, 2016), (Gee, 2003) and (Gansterer, Janecek, & Neumayer, 2008) have used sentiment analysis to help identify spam based on the message content. Based on this research and the intuitive concept that "*authors have textual finger prints*" (Bhowmick & Hazarika, 2016) then it could be possible to identify spam by using features that represent the sentiment as well as the writer as evidenced by their writing style.

This experiment looks to test that intuition as an investigation into spam filtering using personality insight and language tone scores as classifying features. The personality insight and language tone scores will be obtained by using the IBM Watson Tone Analyzer Application Programming Interface (API). To get these scores the body text of an email is submitted to the API, which analyses the text and then returns thirteen different personality insight and language tone scores. The score data will be explored to see if the language of spam emails results in insight and tone scores that are different to the insight and tone scores returned for ham emails. The experiment will then investigate if there is value in using these scores as features to classify emails as spam or ham.

1.2 Research Project

The aim of this project is to investigate if personality insight and language tone features, which are derived from email text content, can improve the classification accuracy of a spam classifier.

1.2.1 Hypotheses

The performance of spam classifiers that use personality insight and language tone scores as features, evaluated by average class accuracy, will be used to test the hypotheses below:

H_0 - Features derived from personality insights and language tone analysis extracted from email text content ***do not*** provide a statistically relevant performance improvement of the classification accuracy of a spam classifier

H_1 - Features derived from personality insights and language tone analysis extracted from email text content ***do*** provide a statistically relevant performance improvement of the classification accuracy of a spam classifier

1.3 Research Objectives

1. Gain in-depth knowledge about spam, machine learning techniques for spam filtering, natural language processing (NLP), sentiment analysis, and application programming interfaces (APIs).
2. Gain practical experience in designing and deploying machine learning algorithms as a classification experiment.
3. Design, test and implement models for spam filtering that use features derived from email body text
4. Design an experiment to evaluate these models and prove one of the hypotheses above.
5. Discuss the relevance of the result, make recommendations for future areas of work and add positively to the body of knowledge in the domain areas outlined above.

1.4 Research Methodology and Approach

The research methodology that will be followed in this project will be empirical evaluation. This will involve investigation and experimentation on features derived from the body text of emails. The emails used come from publicly available email corpuses. The features will be derived using standard text representation techniques and the IBM Watson Tone Analyzer.

The experiment run in this project will involve building classification models that use the derived features to classify the emails as spam or ham. Machine learning techniques will be deployed in the models as these have been shown to be the most practical and successful method of spam filtering. There is a significant body of research on the use of machine learning techniques in this domain and this provides a methodological framework to guide and evaluate research projects.

1.5 Scope and Limitations

The scope of this thesis will be to research and present the main techniques used in spam filtering and to design an experiment to test the hypotheses above. The experiment will be concerned with building a spam classification model that includes personality insight and language tone scores as features to classify emails as spam or ham. The aim will be to evaluate if the inclusion of these features improves the model, with respect to the average class accuracy obtained in classifying emails.

To achieve this a baseline classifier will be built in order to set a level of accuracy against which classifiers that use personality and tone features will be compared. Two classifiers using personality and tone features will be built for this comparison. The first will solely use the personality and tone features while the second will use a combination of the feature-set used in the baseline classifier as well as the personality and tone features.

If the accuracy of the classifiers that use the personality and tone features is better than the baseline classifier than these new features can be considered as a valuable addition to a spam classification model.

A limitation of this work is that the datasets used in this experiment are publically available email corpuses and the experiments are run in an off-line, *in vitro* scenario rather than an *in vivo* scenario mimicking "*an online filter on a mail account that receives feeds of email over time*" (Fawcett, 2003). The *in vitro* scenario has been chosen because it is not known whether the personality insight and tone analysis features are useful in spam filtering. To construct an *in vivo* scenario is more appropriate for when more is known from experimentation whether a feature is useful in filtering spam but it is not known if it remains so in a live environment.

1.6 Outline of Thesis

The remaining chapters in this thesis are arranged as follows:

- Chapter 2 will discuss the literature within the domain of machine learning approaches to spam filtering that are relevant to this project. The chapter firstly covers a brief history of spam, the spam environment and the spam market. Secondly the most common machine learning techniques and algorithms that are used in spam filtering will be described. Followed by an overview natural language processing and sentiment analysis and how these areas have been applied in spam filtering.
- Chapter 3 discusses the methodology of the experiments that will be used for this research project. The sections in this chapter will cover the experimental approach and methodology followed, the datasets used in the experiments as well as the measurement used to evaluate the experiments.
- Chapter 4 outlines the design of the experiments which are the subject of this research, it can be divided into three sections. The first section outlines the software used to develop the experimental framework. The second section discusses the data preparation for the normalised term-frequency feature-sets used in the baseline classifiers as well as how the personality insight and tone features are created. Finally the third section outlines the end to end flows of the three experiments mentioned in section 1.5.
- Chapter 5 covers the results of the experiments that have been run to investigate if the personality insight and tone features, created using the IBM Watson Tone Analyzer, can be used to perform spam filtering. This chapter focuses on the presentation of results from models implemented in the three parts of the overall experiment.
- Chapter 6 concludes the thesis by reviewing the research questions and objectives in the context of the findings from the research conducted. Contributions to the body of knowledge from this work and limitations thereof will be discussed as well as potential areas of future research.

2 Machine Learning and Spam Filtering

2.1 Introduction

In this chapter the research literature within the domain of machine learning approaches to spam filtering, that are relevant to this project, will be discussed. The chapter comprises of three main sections. Firstly a definition for spam will be provided followed by a brief history of spam. Also the spam environment and market will be expanded upon to give some insight into the size of the problem as well as the complexity and persistence of spam.

Secondly the most common machine learning techniques and algorithms that are used in spam filtering will be described. Namely text representation, feature selection techniques, discretization and with respect to algorithms - Naïve Bayes, Support Vector Machines (SVM), Artificial Neural Networks (ANN) and Random Forests.

In the third section natural language processing (NLP) and sentiment analysis will be introduced followed by a discussion on how these areas have been applied in spam filtering. As part of this section a recent paper that has used sentiment scores to improve spam filtering will be summarised. This paper is the concept which this project is looking to advance upon. A similar structure to this paper will be followed but this project will investigate the applicability of personality insight and tone features returned by the IBM Tone Analyzer API.

2.2 Spam Definition

Spam is the common term for unwanted emails, though more technically spam is defined as Unsolicited Bulk Email (UBE) or Unsolicited Commercial Email (UCE). Here the key distinction between spam and marketing is the unsolicited element i.e. an email "*sent to an account by a person unacquainted with the recipient*" (Fawcett, 2003). The first email to be considered spam was sent on May 3rd 1978 to about 400 ARPANET users and it was an open invitation to upcoming computer hardware demonstrations[1]. Since then they have become a daily nuisance for the end users of any email service. The current volume of global email traffic that is spam is estimated to be in the region of 53%[2] – 58%[3] down from a peak of 88% in 2010 (Rao & Reiley, 2012) though still well above its estimated proportion of 10% in 1998 (Goodman *et al.*, 2007).

[1] (Fletcher, 2009)
[2] (Symantec, 2015)
[3] (Kaspersky Lab, 2017)

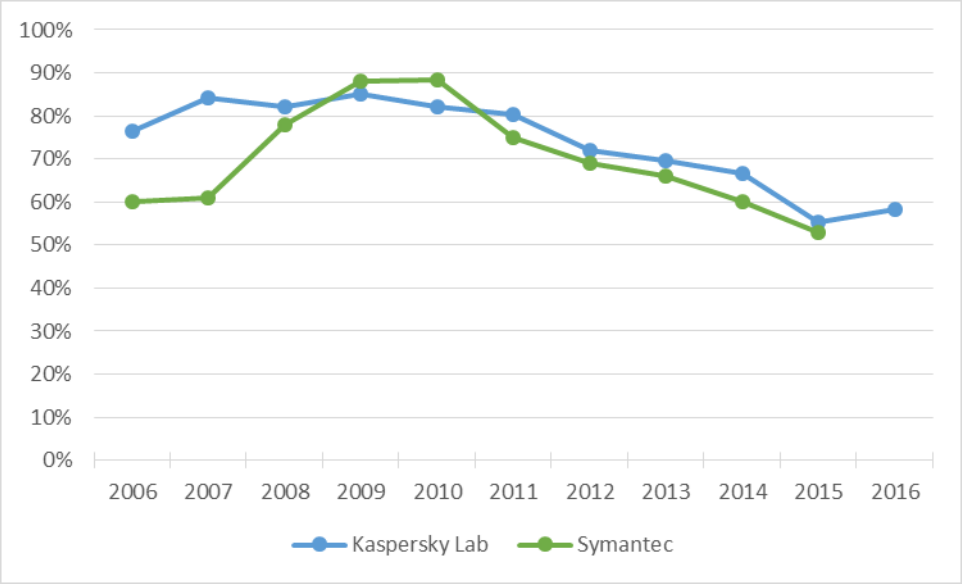


Figure 2-1 Percentage of Spam in all Emails 2006-2016[4]

2.2.1 A Brief History of Spam

In the mid-1990's the first commercial applications for bulk mailing started to be sold (Rao & Reiley, 2012) and from that point on the adversarial, "cat and mouse" game between email service providers and agents that send spam, i.e. "*spammers*", began. Initial tools that email service providers used to block spam incorporated rule or heuristic based filtering (Bhowmick & Hazarika, 2016), IP authentication protocols, static blacklisting IPs of email servers that were known sources of spam and machine learning techniques (Rao & Reiley, 2012).

For each of these blocking tools spammers came up with methods of bypassing them such as misspelling words to bypass set rules or heuristics (e.g. "Viagra" as "Vi@gra") or inserting passages of text from books or news articles to distract from the spam-like content and make the emails harder to classify. IP blacklisting, despite being somewhat blunt, was the most effective of these tools to block spam (Goodman, Cormack, & Heckerman, 2007). The spammers' response to this effective tool would prompt a significant change in the spamming / spam filtering landscape with the wide-scale adoption of spamming botnets in the early 2000's.

A spamming botnet can be defined as a distributed network of compromised computers which are infected by a piece of malicious software (Rao & Reiley, 2012) (Thonnard & Dacier, 2011).

[4] Source: Compiled from Kaspersky Security Bulletin "Spam Evolution" 2006-2016 (where 2006 data relates to Russia only) and Symantec "Internet Security Threat Report" 2006, 2007, 2009-2015 as well as Symantec "The State of Spam" Monthly Report, October 2008.

This malicious software will tie a computer, a.k.a. "*bot*", to an overall "*command and control*" network which sends instructions to many bots for them to send out bulk spam emails.

This now means that spam emails are being sent from a multitude of bots each with their own IP address. Furthermore due to the fact that internet service providers utilise Dynamic Host Control Protocol to assign IP addresses the IP addresses of these bots can change over time. This makes it much harder for email service providers and spam blocking software to use static lists of blacklisted IP addresses to block spam. By the late 2000's botnets became the *de facto* method of sending spam emails, by 2010 it was estimated that "*botnets accounted for 85% of all spam*" (Symantec, 2010).

As with any scenario where there are adaptive adversaries by the latter half of the 2000's co-ordinated attempts by law enforcement agencies, IT companies and IT security firms to take down the command and control network of spam botnets became more common place. The co-ordinated take down of the Rustock botnet by Microsoft, Pfizer and FireEye in March 2011 being a well-known and documented example (Microsoft, 2011)

2.2.2 The Market for Spam

In the online market-place for fake pharmaceuticals, luxury goods (e.g. watches) and software spam is the advertisement at one end of a value chain that links a buyer to a seller. This value chain is globally distributed and involves a number of players and 3rd parties to support it (Levchenko, et al., 2011). The key point of note about this value chain is that the entities that manage the sale and distribution of the products, i.e. merchants, are generally different to the entity that manages the botnet, i.e. the spammer, a.k.a. "*Botmaster*". Botmasters manage botnets – they acquire compromised computers then sell spamming bandwidth and online shop-front support to merchants, as opposed to a situation where a merchant advertises the sale of their own goods using their own spam botnet. Merchants on the other hand deal with the logistics of payment for, sourcing and delivery of goods (Goodman, Cormack, & Heckerman, 2007) (Levchenko, et al., 2011). Spam is usually sold through highly restricted online forums, where members are granted access only where they have been vetted by active or trusted users of the forum.

In their 2011 paper Stone-Gross, Holz, Stringhini, & Vigna managed to obtain a copy of one such forum – Spamdot.biz. From their analysis they describe two methods of selling spam. For each method the basic premise of the botmaster getting a percentage of a completed sale applies.

The first method is "spam as a service" where a botmaster will run a campaign for a merchant based on a payment per million emails sent. The cost per million will vary depending on the quality of the email addresses used and the quality of the bots sending the mails. For example email addresses that belong to free email services like Hotmail or Gmail can be half the price of other email addresses (Stone-Gross, Holz, Stringhini, & Vigna, 2011) primarily because free email services have advanced spam filters. In relation to bot quality more expensive bots will be "clean", i.e. not blacklisted and therefore is less likely to have the mails they send blocked by a spam filter as well as residing in the US or Europe where they can have good broadband connection (Stone-Gross, Holz, Stringhini, & Vigna, 2011).

The second method of selling spam involves the spammer renting out a portion of the bot to a 3rd party who obtains email addresses and arranges the spamming campaigns themselves. This allows that 3rd party to send emails at much higher rate and thereby reduce the cost per million emails sent. In the case of the Cutwail botnet to facilitate this the botmaster provided a web interface with a variety of features that allowed the user to define the parameters of their email campaign such as email headers, sending address, mail content. It even went as far as allowing users to run their campaign against a widely used spam filter in order to adjust the mail parameters to avoid being classified as spam (Stone-Gross, Holz, Stringhini, & Vigna, 2011).

2.3 State of the Art

The following section will discuss machine learning techniques and common algorithms applied in spam filtering, namely naïve Bayes, and its variations, as well as Support Vector Machines, Artificial Neural Networks and Random Forests.

2.3.1 Representation

A key element of spam filtering is how an email is represented to the classifier. The following will discuss some of the most popular representation methods. They will be discussed in the context of the body text of emails although the subject and header fields are also valuable features that can be considered (Guzella & Caminhas, 2009).

The "*bag-of-words"* (BoW) or vector-space model is one of the most popular and successful representation methods. To represent an email as a "*bag-of-words"* a vocabulary or dictionary is created, usually from the training data, of all the words that occur in those emails and the representation is related to that (Eberhardt, 2015). Each email becomes an *N*-dimensional feature vector $\vec{x} = [x_1, x_2 \dots x_n]$ where x_1 represents a word in the email. Commonly this is a

representation of zeros and ones where if a word is in the email it is represented by "1" and if it is absent it is represented by "0". Subsequently regardless of how many times a word appears in an email only its presence or absence is represented (Subramaniam, Jalab, & Taqa, 2010), (Guzella & Caminhas, 2009).

Other variants on the BoW model are character and word n-gram models. Word n-grams attempt take into account that some words generally appear together, such as "buy now", and furthermore that some combinations of words are more likely to appear in spam than in ham. For a given piece of text i.e. "special offer price" word 2-gram sequences are "*obtained through the application of a sliding window"* (Guzella & Caminhas, 2009) so the above text would be represented as |special offer|,|offer price|. This approach however can create very large dictionaries (Kanaris, Kanaris, Houvardas, & Stamatatos, 2007).

Character n-grams work in much the same way as word n-grams but applied to sequences of characters. Using the same example as above a character 2-gram would be represented at |sp|, |pe|, |ec|, |ci|, |ia|, |al|….|ce|. Kanaris, Kanaris, Houvardas, & Stamatatos, 2007 identify a number of advantages of this method. They are robust to mis-spelling, in that a mis-spelled word can share many of the same n-grams with the correct spelling, and they can be language independent. Futhermore although breaking words into character n-grams does create a large dictionary there are less character combinations than there are word combinations, this translates to fewer character combinations with a zero frequency.

Term-frequency is another common method where for a vector $\vec{x} = [x_1, x_2 \dots x_n]$ x_1 is the number of times a word w_1 appears in that an email and it is common for the feature vector to be normalised to the unit length (Drucker, Wu, & Vapnik, 1999). An extention of this method is *tf-idf* (term-frequency-inverse document frequency). The document frequency is calculated as the number of times a word appears in the overall set of documents that the vocabulary or dictionary are based off. The inverse document frequency is calculated as per below;

$$IDF(w_i) = \log(\frac{|D|}{DF(w_i)})$$

Where:

- |D| is the number of documents
- DF(w_i) is the number of times w_i appears in the overall set of documents

The premise of this method is that a "*term which occurs in many documents is not a good discriminator, and should be given less weight than one which occurs in few documents"* (Robertson, 2004). This method gives a measure of significance of a word in identifying the class of an email in the context of the overall feature-set (Subramaniam, Jalab, & Taqa, 2010).

2.3.2 Feature Selection

In general when trying to fit a model reducing the number of features can help with removing noise such as redundant or irrelevant features and thereby increase the performance of the classifier or keep its performance stable for a lower number of features (Kelleher, MacNamee, & D'arcy, 2015). The "*bag-of-words*" type representation methods can lead to a high dimension feature space so feature selection is usually applied in spam filtering to reduce the size of this feature space. There are three main categories of feature selection method. Firstly there are "filter" methods that use the statistical relationship between a feature and the target variable. "*These methods determine a score for each [feature and the features] with the highest scores are selected"* (Guzella & Caminhas, 2009) though non-statistical methods such as stop-word removal and lemmatisation can be included here. Secondly there are "wrapper" methods which employ machine learning models to test the performance of a model with a given subset of features. The feature-set that results in the best performing model is selected (Kelleher, MacNamee, & D'arcy, 2015). Thirdly there are "embedded" methods that "perform variable selection in the process of training" (Guyon & Elisseeff, 2003). In spam filtering it is the "filter" methods that are most commonly used and the remainder of this section focuses on the statistical methods of information gain, also referred to as mutual information, and Chi-square (X^2).

Information gain is the most commonly used of the filter methods (Guzella & Caminhas, 2009). Information gain is based on the Shannon entropy model. Based on this model a dataset of emails could be said to have low entropy if a random email was picked and it had high probability of being a particular class – spam or ham. Information gain measures in terms of "bits" the information gained for categorising a document based on knowing the presence or absence of a word in that document (Subramaniam, Jalab, & Taqa, 2010).

Chi-square (X^2) is another statistical filter method used for feature selection. This measure compares the observed occurrence of a feature against the expected occurrence of that feature. Applied to feature selection this method looks at the relationship between the expected and

observed occurrences of a word (feature) in an email and determines whether that email is spam or ham (Yerazunis, Chhabra, Siefkes, Assis, & Gunopulos, 2005).

2.3.3 Discretisation Techniques

Discretisation, also known as "binning" or "coarse classification", can be defined as transforming "*numerical attributes into discrete or nominal attributes with a finite number of intervals, obtaining a non-overlapping partition of a continuous domain*" (Garcia, Luengo, Sáez, Lopez, & Herrera, 2013). Two of the most common techniques are "equal-width" and "equal-frequency" binning. In both of these techniques the number of bins are set *a priori* and without any evaluation measure applied to the attributes. These are very straight-forward techniques though they do have downsides in terms of a loss of detail in the data, however that loss can be restricted by the appropriate selection of bins to divide the continuous values into (Kwak & Choi, 2002).

In "equal-width" the bins are set to cover a range of values, though this can mean that some bins have very few values whereas others can have the majority of values. With "equal-frequency" binning the attribute values are sorted by ascending value and an equal number of attributes are placed in each bin. This results in bins that can cover varying ranges of values (Kelleher, MacNamee, & D'arcy, 2015).

There are a wide variety of discretisation techniques with varying degrees of complexity. Techniques like MLDP and ID3 are referred to as "dynamic" techniques as they work alongside the machine learning algorithm as it is building the model and they make binning decisions based on the data made available by the algorithm. Other techniques like "chi-merge" or "chi-square" discretisation use statistical evaluation of the correlation measurements among attributes to decide on the bin ranges (Garcia, Luengo, Sáez, Lopez, & Herrera, 2013).

2.3.4 Sampling Methods

As part of evaluating a model it is important to test how that model will perform on data that it has not seen. This testing can help avoid the scenario where model becomes too complex and fitted to the nuances of data it is trained on, a.k.a. "over-fitting" (Kelleher, MacNamee, & D'arcy, 2015). A common and simple approach to mitigate against over-fitting is "hold-out sampling" whereby the available training data is split into two, or three, smaller data sets that are each representative of the overall dataset.

The model is then trained using one portion of the data, the training set, and then its performance after training is tested against the other portion, i.e. the test set. Often a dataset is split into three to incorporate a "validation set" which is used after the model has been trained and before testing. This validation set is used to adjust and test the different parameters of the algorithm in order to improve the performance. Hold-out sampling is the most straight-forward sampling technique but is best suited to large datasets where using only a portion of the data for training is still enough data to appropriately train the algorithm.

Where the available training dataset is small then other techniques exist such as k-fold cross validation and leave one out cross validation which maximise usage of the data for training and testing of the algorithm. The idea of these techniques is that each example in the dataset is used for both training and test data (Russel & Norvig, 2009).

In k-fold cross validation the full available dataset is divided into k equal parts, or folds, and k separate experiments are performed. In each experiment one of the folds is held out as the test set and the remaining k-1 folds are used as the training data. The model is trained on the training data and then the applicable performance measures are calculated using the test set. This is run k times and an overall performance is calculated based on aggregating the results of the k tests. Leave one out cross validation is an extreme form of k-fold cross validation where the dataset is split into k folds where k is equal to the number of examples in the dataset. Here the test set is just one example and the training set is comprised of all of the other examples.

2.3.5 Naïve Bayes Classifiers

The Naïve Bayes classifier is one of the most common machine learning algorithms applied to spam filtering. This classifier and its variants have been shown to be successful in detecting spam since the mid to late 1990's with papers from Sahami, Dumais, Heckerman, & Horvitz, 1998 and Androutsopoulos, Koutsias, Chandrinos, Paliouras, & Spyropoulos, 2000 being some of the foundational works from that time. These type of classifiers have been implemented both in commercial and free spam filters, SpamBayes being an obvious example, due in part to "*their simplicity, which makes them easy to implement…and their accuracy which in spam filtering is comparable to that of more elaborate learning algorithms*" (Metsis, Androutsopoulos, & Paliouras, 2006)

In the spam filtering domain a Naïve Bayes classifier learns to classify an email as spam or ham via supervised learning where it is provided a training set of emails that are already labelled with their target values. In the broadest sense when attempting to classify emails by the text

within them each email is represented as a word feature vector x = {x_1, x_2,..x_n}, with its corresponding target label T, where T ϵ {spam, ham}.

To classify a new email the Naïve Bayes classifier compares the words in the new email against the words from the pre-labelled emails from the training set. It then classifies the new email as whichever class has the higher probability of those words occurring within it. This is calculated via the below equation (Mitchell, 1997).

$$T = \underset{tspam \in T}{\operatorname{argmax}} P\left(t_{spam} \mid x_1, x_2 \dots x_n\right)$$

Using Bayes Theorem this is rewritten as:

$$T = \underset{tspam \in T}{\operatorname{argmax}} \frac{P\left(x_1, x_2 \dots x_n \mid t_{spam}\right) P(t_{\text{spam}})}{P(x_1, x_2 \dots x_n)} = \underset{tspam \in T}{\operatorname{argmax}} P(x_1, x_2 \dots x_n \mid t_{\text{spam}}) P(t_{\text{spam}})$$

In practice calculating P(x_1, x_2… x_n | t $_{\text{spam}}$) is never possible as it requires a "*dataset that is big enough to sufficiently cover all the possible combinations of the feature values*" (Kelleher, MacNamee, & D'arcy, 2015) so it at this point that the naïve assumption is applied.

The naïve assumption assumes all features represented in a word feature vector are conditionally independent. This assumption is made even though its understood that a language is structured therefore some words will have a high likelihood of being followed or preceded by other words, such as "orange" and "juice" (Bhowmick & Hazarika, 2016). Yet even with this naïve assumption Naïve Bayes classifiers still perform well in classification tasks.

The assumption here "*is that given the target value of the instance, the probability of observing the conjunction x_1, x_2… x_n is just the product of the probabilities for the individual attributes: P(x_1, x_2… x_n | t $_{spam}$) = $\prod_{i=1}^{N} P(x_n|t_{spam})$*" (Mitchell, 1997). This can then be substituted into the equation above and be re-written as:

$$T_{SPAM} = \underset{t_{spam} \in T}{\operatorname{argmax}} P(t_{spam}) \prod_{i=1}^{N} P(x_n|t_{spam})$$

Where T_{SPAM} is a scenario in the context of spam filtering where the classifier gives that result for a new unclassified email, the same equation holds for ham or any classification task.

2.3.6 Naïve Bayes: Main Variations

There are a number of variations of the Naïve Bayes classifier that are common in spam literature. The main difference between these variations is in the feature representation. The most common variations – Multinomial and Multivariate Bernoulli Naïve Bayes – are discussed below.

In the Multivariate Bernoulli method of Naïve Bayes for each instance (email) each feature (word) is treated individually but is represented as a Boolean value as discussed earlier. By contrast the Multinomial method treats an instance (email) as a feature vector where each feature (t_i) represents the number of occurrences of a word in (x_1) the email (d). This is referred to as "term-frequency". Each spam email then has the probability $p(t_i \mid C_{spam})$ for each (t_i). In this instance the $P(x_1, x_2 \ldots x_n \mid C_{spam})$ is the multinomial distribution:

$$p(\vec{x} \mid C_{spam}) = p(|d|) \,.\, |d|! \prod_{i=1}^{m} \frac{p(t_i \mid C_{spam}) x^t}{x_i\,!}$$

Using a Laplacean prior each $p(t_i \mid c)$ is estimated by counting the number of t_i in all training documents.

$$p(t|c) = \frac{1 + N_{t,c}}{m + N_c}$$

Where $N_{t,c}$ is the number of times a particular term appears in the training set and $N_c$is the total number of words in the training set.

An assumption of this model is that the number of words in an instance is independent of the class of that instance. Despite this apparent over simplistic assumption, i.e. that it is less likely to receive a lengthy spam email than to receive a ham email of the same length (Metsis, Androutsopoulos, & Paliouras, 2006), a number of studies have found that the multinomial method out-performs the multivariate method (Schneider, 2004). This is not necessarily due to the richer information available when using a term-frequency. Schneider, 2004 has shown that the difference is due to how the models handle negative evidence i.e. words that are not present in an instance. In Multivariate Bernoulli naïve Bayes a word feature, x_1 that is present has a class probability of $P(x_1 \mid c_i)$ and its absence a probability of $1 - P(x_1 \mid c_i)$ so the probability of an instance being one class or another can be as much determined by the words that are not there

as the words that are. Whereas in multinomial naïve Bayes the probability representation of an instance is based on the words that are in the instance only and how often they occur.

The advantages of Naïve Bayes classifiers are that they are good general performers and for their simplicity they perform well in comparison with other more complex classifiers (Guzella & Caminhas, 2009). In classification they take account of every word rather than on a limited sub-set therefore it does not make "*premature classifications of the email"* (Eberhardt, 2015).

A naïve Bayes classifier must have labelled training data in order to learn but it is quite simple to train by comparison to other models. All that is needed are the prior probabilities of the target classes and the conditional probability of the features given the target class(Kelleher, MacNamee, & D'arcy, 2015).

A disadvantage of naïve Bayes classifiers particularly in spam filtering is that they are quite susceptible to "*Bayesian poisoning*" whereby spammers will "*include irrelevant text passages, such as excerpts of news stories that are common in legitimate conversations*" (Rao & Reiley, 2012). This has the effect of tricking the classifier into mis-classifying a spam email as ham.

2.3.7 Support Vector Machines

Support Vector Machines (SVM) is a machine learning technique that was developed in the mid-1990s and is widely used because it generalises well (Russel & Norvig, 2009). It can be described as an error based learning model that is conceptually similar to regression models though it is trained differently (Kelleher, MacNamee, & D'arcy, 2015).

In training a support vector machine the aim is to find a decision boundary, or hyperplane, that separates the target variables. The training examples, spam and ham instances, exist either side of the separating hyperplane and the perpendicular distance between these examples and the hyperplane is known as the margin. The optimal separating hyperplane is one that gives the maximum margin between the positive (spam) and negative (ham) instances (Yu & Xu, 2008). These instances that are closest to the hyperplane "*are called support vectors because they hold up the separating plane*" (Russel & Norvig, 2009).

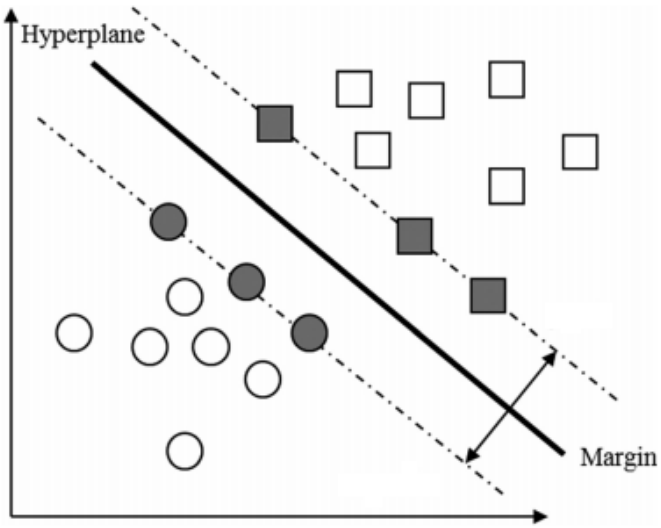


Figure 2-2 Optimal Separating Hyperplane
(Du, Liu, & Lifeng, 2015)

The description so far implies that the data can be linearly separated, which is not normally the case in reality. The introduction of kernel functions into SVMs, known as the "kernelling trick" allows non-linear data to be separated. "*Often, data that are not linearly separable in the original input space are easily separable in the higher-dimensional space…if you look at a set of points from enough directions you'll find a way to line them up. The high-dimensional linear separator is actually nonlinear…when mapped back to the original input space [and] can correspond to arbitrarily wiggly, nonlinear decision boundaries between the positive and negative examples.*" (Russel & Norvig, 2009). In SVM implementations there are four commonly used kernel functions – linear, polynomial, radial basis function (RBF) and sigmoid. Each of these have their own specific parameters to be tuned in order to find an optimal separating hyperplane.

A final concept within SVM to discuss is the C-value or "slack variable" this parameter of SVM allows the margins either side of the hyperplane to be violated (Yu & Xu, 2008). This introduces a trade-off between some level of misclassification and a smooth hyperplane (decision boundary). A low C-value allows more misclassification but "*allows training examples to exist in between*" the margins (Drucker, Wu, & Vapnik, 1999). Whereas a high C-value allows the hyperplane to adapt to the more of the support vectors and so correctly classify more of the training examples.

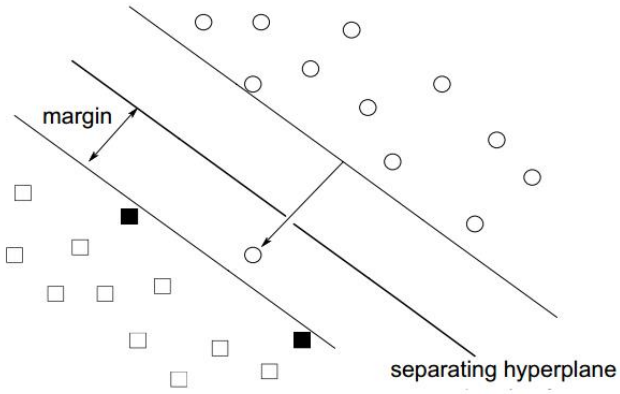


Figure 2-3 Operating mode of SVM allowing margins to be breached (Yu & Xu, 2008)

SVMs have become such a useful tool for machine learning tasks because they have been shown to generalise well and are regarded as quick to train where the number of training examples is not very large and the only parameter to tune is the C value (Drucker, Wu, & Vapnik, 1999). They are "*robust to overfitting*" (Kelleher, MacNamee, & D'arcy, 2015) and are relatively resilient to class imbalance. This resilience stems from the fact that SVMs look to separate the target classes, so it is the support vectors that are the most important instances. New instances that are added to the dataset which are behind the hyperplane of either class do not influence the hyperplane (Drucker, Wu, & Vapnik, 1999).

SVMs can be slow to train however where there is a very large number of training examples and execution can also be slow where data is noisy, not easily separated and complex kernels have to be used to find a higher dimensional space where the examples can be linearly separated (Drucker, Wu, & Vapnik, 1999), (Kelleher, MacNamee, & D'arcy, 2015).

2.3.8 Support Vector Machines: Variations

Two SVM variations that can be used for classification are discussed in this section - LinearSVC and NuSVC. The LinearSVC variation as described in (Hsieh, Chang, Lin, Keerthi, & Sundararajan, 2008) and (Fan, Chang, Hsieh, Wang, & Lin, 2008) is a variation of SVM that improves training speed for large datasets by using dual co-ordinated decent method for finding the optimal separating hyperplane. Furthermore this variation also attempts to improve training speed by solving the optimisation problem by choosing variables randomly rather than in

sequence. This is followed as research has shown that "*solving sub-problems in a random order may give faster convergence*" (Hsieh, Chang, Lin, Keerthi, & Sundararajan, 2008).

NuSVC is a further variation introduced by Schölkopf, Smola, Williamson, & Bartlett, which reparameterises the C value the value **ν** for classification tasks *"such that*

i. ***ν*** *is an upper bound on the fraction of margin errors.*
ii. ***ν*** *is a lower bound on the fraction of Support Vectors"*

The C-value in SVM reflects a choice between a smooth decision boundary and misclassification. Setting a C-value is normally done by testing different values in a range, though there is no upper bound on what that range should be so setting C is somewhat unintuitive. The NuSVC reparameterises C as **ν** which has a set range of 0.0 to 1.0 which translates into the above boundaries and makes setting the parameter more intuitive.

2.3.9 Artificial Neural Networks

Artificial Neural Networks (ANN) are a mathematical representation of the biological neural networks that exist in the human brain. They attempt to emulate the sensory input / response output activity of the brain by following the message sending communication patterns that occur between neurons (Subramaniam, Jalab, & Taqa, 2010).

The standard model of representing neurons in ANN is the McCullogh-Pitts (M-P) Model, shown below. It is made up of three parts which are generally referred to as the input, hidden and output layers.

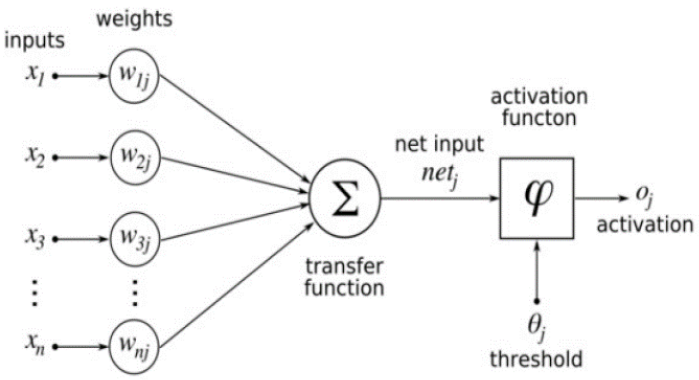


Figure 2-4 McCullogh-Pitts Model of Neuron

(Subramaniam, Jalab, & Taqa, 2010)

Signals are passed through the nodes at the input layer and weights are applied to the signals (see figure 2-4) prior to passing it onto the hidden layer (Zhou, 2012). The hidden layer contains

a "*transfer function"* which sums the weighted signal values and compares them to some threshold or bias. Where the summed signal is above the threshold it is passed through an "*activation function*" to an output layer to generate an output signal (Zhou, 2012), (Haykin, 1999).

In the context of spam filtering the input signal is a feature vector representing an email that is passed into the input layer and the output layer generates a signal, i.e. 0 or 1, to classify the email as ham or spam. In spam filtering the common forms of ANN used are single or multi layered feed forward network with a sigmoid activation function (Subramaniam, Jalab, & Taqa, 2010) (Yu & Xu, 2008)

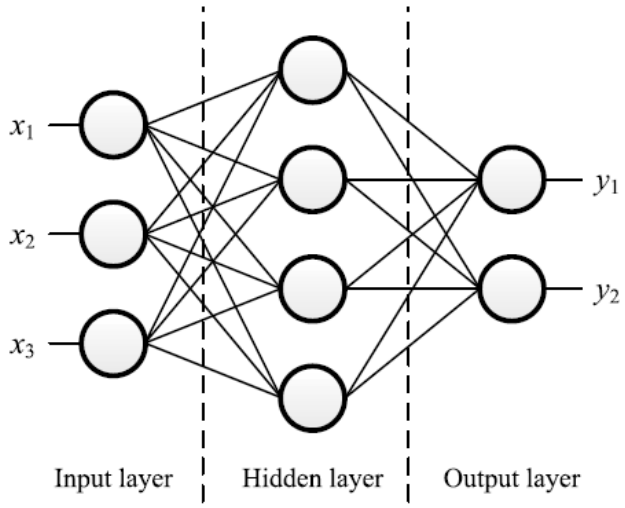


Figure 2-5 Multi-layer feed forward network

(Zhou, 2012)

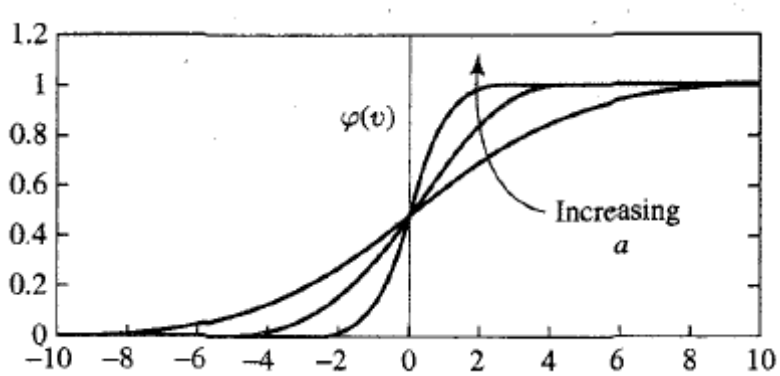


Figure 2-6 Sigmoid Function for varying slope parameter (a)

(Haykin, 1999)

Early application of ANNs in content based spam filtering produced results that outperformed a Naïve Bayes classifier but did not generalise well (Clark, Koprinska, & Poon, 2003). However later applications (Wang, Jones, & Pan, 2006) and (Tzortzis & Likas, 2007) were able to show ANNs outperforming Naïve Bayes and SVM algorithms across multiple datasets. A later study used ANN to compare the classification of emails using non-content features, derived from the delivery information of an email, against an ANN using text features. In this experiment the non-content based model achieved better results (Wu, 2009).

2.3.10 Random Forests

An ensemble model is type of model that uses the aggregate outputs of a set of models to predict or classify a target variable. Ensemble models can be very accurate even if the individual models in the ensemble perform poorly at classification or prediction (Kelleher, MacNamee, & D'arcy, 2015). Two popular methods for creating an ensemble model are boosting and bagging. It is the bagging method that is of relevance for explaining Random Forests.

In the bagging ensemble method the individual models are trained on a randomized sample of the dataset. By using sampling with replacement a training set is made that is the same size as the initial dataset, however it contains duplicate instances and is also missing some instances. This type of dataset is also known as a "bootstrap sample".

Random Forests can be described as an ensemble model that uses decision trees as the individual models and extends the bagging method by incorporating randomized feature selection. "*During the construction of a component decision tree, at each step of split selection, RF first randomly selects a subset of features, and then carries out the conventional split selection procedure within the selected feature subset.* (Zhou, 2012).

Liaw & Wiener, 2002 outline the process of setting up a Random Forests classifier in 3 broad steps, which puts the above into practical stages:

1. Create N_{tree} bootstrap samples from the initial dataset.
2. Train each N_{tree} bootstrap samples on a decision tree without pruning. At each node select a random subset of features and choose the best feature to split on from that sub-set.
3. Use majority voting from the outputs of the N_{tree} decision trees to make a classification.

The parameters to be considered as part of the above are the number of decision trees to include in the model (N_{tree}), the number of features to randomly select at each split in the decision trees and also how to measure the purity of the resulting splits with respect to the classes. In selecting the number of trees to include a balance needs to be found between the tendency of the generalisation error "*ultimately approaches zero as* [the number of trees] *approaches zero"* (Zhou, 2012) and the increase in training time for an increasing number of decision trees in a model. For identifying an appropriate number of features the literature suggests that the logarithm of the number of features in the dataset is used (Breiman, 2001). The most common methods of measuring the impurity of a split are entropy and the Gini Index (Kelleher, MacNamee, & D'arcy, 2015).

Breiman, 2001, identified two key benefits of using Random Forests. Firstly due to the Strong Law of Large Numbers as more trees are added to a Random Forests classifier they tend not to over-fit. Secondly as a result of using a bootstrap sample for each classifier there remains a portion of the training data that remains "out-of-bag". This portion of the data can be used to test the classifier and therefore it can remove the need for a separate hold-out test set to be set aside (Breiman, 2001), which is useful when the amount of available data is small.

There has been research in to the application of Random Forests in spam filtering. Koprinska, Poon, Clark, & Chan, 2007 used Random Forests in a supervised learning experiment with two public datasets (LingSpam & PU1 corpuses). While Random Forests performed better than Naïve Bayes on these datasets they did not perform well when they were tested with other datasets. DeBarr & Wechsler, 2009 proposed a three layered model for spam filtering where clustering is used to label emails for training A Random Forests classifier is then trained off these and used to classify emails and finally an active learning component is used to refine the Random Forests classifier. Using this model the authors achieved better performance on classification than a Naïve Bayes, SVM and k-Nearest Neighbour base classifiers. Also Lee, Kim, Kim, & Park, 2010 successfully applied spam filtering with Random Forests. Their study focused on identifying the most important features and then optimising the parameters of the Random Forests classifier based on the features selected.

2.4 Natural Language Processing in Spam Filtering

In the domain of computer science Natural Language Processing (NLP) is a discipline within Artificial Intelligence (AI). The below definition of implies that link with its overall aim of "*achieving human-like language processing*".

"*Natural language processing is a range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of particular tasks or applications*." (Liddy, 2001). Or in a more straight-forward way as "*how computers can be used to understand and manipulate natural language text or speech to do useful things*" (Chowdhury, 2003)

Within NLP there are four approaches to analysing language: symbolic, statistical, connectionist and hybrid. It is the "statistical" approach to natural language processing that is used in spam filtering. These approaches use "*various mathematical techniques and often use large text corpora to develop approximate generalized models of linguistic phenomena based on actual examples of these phenomena provided by the text corpora with adding significant linguistic or world knowledge*" (Liddy, 2001). Parsing, Part-of-speech tagging, speech recognition are examples of the application of the statistical approach which are used as part of this investigation.

2.4.1 Sentiment Analysis in Spam Filtering

Sentiment analysis or opinion mining is a subject area within NLP that "*deals with the computational treatment of opinion, sentiment and subjectivity in text*" (Pang & Lee, 2008). It has been shown to be successful in identifying spam-like content in areas such as online review spam (Lau, *et al*., 2011) and also in email (Ezpeleta, Zurutuza, & Hidalgo, 2016). In their 2016 paper Ezpeleta, Zurutuza, & Hidalgo "*looked to validate the assumption that a spam message, being a commercial communication, the semantics of its content should be shaped with a positive meaning*"

Using the CSDMC 2010 spam corpus their experiment focused only on content based filtering and they experimented with improving on Naïve Bayes classifiers by including sentiment polarity scores in the feature-set.

In the first part of the experiment the authors applied seven variants of naïve Bayes classifiers with a range of parameter settings on the dataset. The purpose of this was to both set a baseline

level of accuracy to improve upon and also to identify the best performing classifiers to use in the third part of the experiment. In that third part they tested whether the inclusion of sentiment polarity scores improved the performance of those ten classifiers.

In the second part of their experiment the authors chose two freely available sentiment classifiers *SentiWordNet* and *TextBlob* to get sentiment polarity scores. Four versions of sentiment classifiers (described below table) are then applied to the CSDMC2010 dataset in order to assign a positive or negative sentiment polarity to the emails. From the below table it can be seen that the sentiment classifiers classify spam messages as more positive than ham messages.

	SentiWordNet				**TextBlob**			
	Adjectives Only		**Adjectives Plus**		**TB_005**		**TB_01**	
	Positive %	**Negative %**	**Positive %**	**Negative %**	**Positive %**	**Negative %**	**Positive %**	**Negative %**
Spam	66	31	68	31	76	24	62	37
Ham	37	62	37	62	66	34	48	51

Table 2-1 Sentiment analysis of CSDMC emails

(Ezpeleta, Zurutuza, & Hidalgo, 2016)

1. SentiWordNet Adjective: Adjectives are used to get the sentiment polarity. Emails can be given a sentiment of zero or are classed as positive or negative.
2. SentiWordNet AdjectivePlus: Here the emails given a zero sentiment are classed as positive.
3. TextBlob_005: The number refers to the threshold score above which an email is classed as positive (0.005).
4. TextBlob_01: As above but with a threshold of 0.01

For the third part of the experiment four versions of the dataset are created by including the polarity scores from each of the above sentiment classifiers. Then each dataset is tested against the ten best NB classifiers.

	Accuracy %				
Naïve Bayes Classifier	**Baseline Bayes**	**SentiWordNet Adjective Only**	**SentiWordNet Adjective Plus**	**TextBlob_005**	**TextBlob_01**
1	99.1451	**99.1682**	99.1451	99.122	99.122
2	99.122	99.0989	99.0989	99.0989	**99.2144**
3	99.122	99.0989	99.0989	99.0989	**99.2144**
4	99.122	99.0989	99.0989	99.0989	**99.2144**
5	99.0527	99.0989	99.0989	99.0989	**99.2144**
6	99.0527	**99.122**	**99.122**	**99.1682**	**99.1451**
7	99.0527	**99.122**	**99.122**	**99.1682**	**99.1451**
8	99.0527	**99.122**	**99.122**	**99.1682**	**99.1451**
9	99.0527	**99.122**	**99.122**	**99.1682**	**99.1451**
10	99.0296	**99.122**	**99.122**	**99.1682**	99.296

Table 2-2 Comparison of original results with results from Polarity Classifiers

(Ezpeleta, Zurutuza, & Hidalgo, 2016)

What the results show is that, when compared to the baseline accuracy, the accuracy of the best NB classifier is improved with the inclusion of sentiment polarity scores from the *SentiWordNet_Adjective* classifier. Secondly there is a general improvement in all the classifiers when sentiment polarity is included. In the case of the NB with *TextBlob_01* 8 out of 10 classifiers are improved and 5 out of 10 for the others. These results show that an emails sentiment polarity can be used to help detect spam and improve a spam classifiers performance with respect to accuracy.

2.5 Conclusion

In this chapter an outline of the spam domain and application of machine learning with in it has been given. The most common machine learning techniques and algorithms applied in spam filtering have been described. Natural language processing and sentiment analysis have been introduced in order to give context to research that has shown that sentiment features can be used to improve spam filtering. A summary of this research has been provided and it has given a framework which this project will follow to investigate if similar features can be used for the same purpose. The following chapters will expand this framework into the methodology and experiment design of this project.

3 Methodology

3.1 Introduction

This chapter will discuss the methodology of the experiments that will be used for this research project. The aim of this project is to investigate whether personality insight and tone features, derived from the text content of emails, can be used to classify those emails as ham or spam. The sections within this chapter will cover the experimental approach and methodology followed, the datasets used in the experiments as well as the measurement used to evaluate the experiments.

3.2 Experimental Approach

The overall experiment comprises of three parts. The first part relates to the "baseline" classifier which is used to set a baseline performance against which other classifiers that use the tone features will be judged against. The classifiers tested to find a baseline are a multinomial Naïve Bayes classifier and 3 versions of a Support Vector Machine (SVM) classifier.

In the second part two classifiers – a Random Forests and SVM - are experimented with to investigate if the personality insight and tone features, gathered using the IBM Tone Analyzer, can be used *on their own* to classify emails as ham or spam. Since the features returned from the IBM Tone Analyzer are continuous values it allows algorithms beyond Naïve Bayes and Support Vector Machines to be experimented with. A Random Forests classifier is a general purpose classifier which has been successfully implemented in a variety of scenarios and can be implemented here against the personality insight and tone features. In this part of the experiment investigations are carried out on feature selection and optimising the classifiers by using a grid search of a range of parameter settings.

In the third part of the experiment the personality insight and tone features are combined with the features used in by the baseline classifier. The classifier used is a SVM. This part will be referred to as the "combined" classifier.

In each of these experiment parts each time a classifier, with the related parameter settings and feature selection technique, is tested the experiments are run 6 times and the results averaged to provide the results reported.

3.3 Datasets used in Experiments

In this experiment three well known public spam corpuses have been used. These are the Enron, SpamAssassin and CSDMC2010 datasets. Each dataset has been taken from the same website – www.csmining.org and as of writing all datasets are available on that site.

3.3.1 Enron Dataset

The Enron Dataset used in this experiment is a sub-set taken from the dataset created by Metsis, Androutsopoulos, & Paliouras from their 2006 paper. Out of the larger Enron dataset made up of emails from 158 Enron employees the authors took the ham email collections of 6 employees and paired them with spam messages from four different sources (two of which are combined to form 1 below) to make 6 smaller datasets:

1. The SpamAssassin corpus with spam from the Honeypot project, referred to as (SH)
2. The spam collection of Bruce Guenter, (BG)
3. The spam collection of Georgios Paliouras, (GP)

The below table, adapted from Metsis, Androutsopoulos, & Paliouras, 2006, shows the pairing and composition of the dataset.

Dataset Name	Ham Source: Employee Mailbox	Spam Source	Ham:Spam Ratio	Ham Time Period	Spam Time Period
Enron1	Farmer – d	GP	3672:1500	12/99 – 01/02	12/03 – 09/05
Enron2	Kaminski-v	SH	4361:1496	12/99 – 05/01	05/01 – 07/05
Enron3	Kitchen-l	BG	4012:1500	02/01 – 02/02	08/04 – 07/05
Enron4	Wiliams-w3	GP	1500:4500	04/01 – 02/02	12/03 – 09/05
Enron5	Beck-s	SH	1500:3675	01/00 – 05/01	05/01 – 07/05
Enron6	Lokay-m	BG	1500:4500	06/00 – 03/02	08/04 – 07/05

Table 3-1 Composition of the six Enron benchmark datasets.

Firstly in order to maintain independence of the datasets used in this experiment datasets Enron2 and Enron5 have not been used because the spam source for those two Enron datasets, SpamAssassin, is used in its own right as a dataset for this experiment. Secondly the authors do not explicitly say there is no duplication of spam mails where two ham mailboxes are paired

with the same spam source i.e. Enron1 and Enron4 and so on. For this reason in this experiment of the 6 Enron datasets the ham mails from Enron1, Enron3, Enron4, Enron6 are used with only the spam from Enron 1 and Enron3. This is to avoid duplication of spam emails in the overall datasets used. This results in a combined dataset of 13,684 emails with a ham:spam ratio of 10684:3000 or 22% spam.

3.3.2 CSDMC2010 Dataset

This public dataset was created for a data mining competition and is made up of 4327 emails with a ham: spam ratio of 2949:1378 or 32% spam. Some limited details on the sources of the emails are available online stating that they are sourced from messages posted to public fora or from public mail lists with some data from public corpuses and mails from non-Spam trap sources.

3.3.3 Spam Assassin Dataset

This public dataset was collated in 2002 and contains 6047 messages with a ham:spam split of 4150:1897 or 31%. The messages that make up the dataset are a mix of messages that were posted to public fora, sent to and from the provider of the dataset and mails that originated as newsletters from public news web sites. They range in the level of how "*spam-like*" they are. The below table outlines the different message groups with any applicable descriptions from the author.

Message Group	No. of Messages	Provider comments
Easy Ham	2500	Contains messages that are "*typically easy to differentiate from spam as they do not have spam like signatures such as HTML*"
Easy Ham 2	1400	-
Hard Ham	250	Contains "*messages which are closer in many respects to typical spam: use of HTML, unusual HTML markup, coloured text, "spammish-sounding" phrases etc.*"
Spam	500	All messages "*received from non-spam-trap sources.*"
Spam 2	1397	-

Table 3-2 Summary of contents of Spam Assassin Dataset[5]

[5] (Mason, 2006)

In this experiment two methods of representation will be used. In the baseline classifier emails from all three datasets will be represented as a normalised term-frequency. The personality insight and tone features will be returned as continuous features so these will be discretised. These will be expanded on in the data preparation section in Chapter 5.

3.4 Feature Selection

Feature selection is applied in the second part of the overall experiment which investigates whether personality insights and tone features, on their own, can be used in spam filtering. The Chi-square and mutual information techniques are used to rank the features. The experiments on the Enron dataset are then run using the best 5 and 9 ranked features from both techniques as well as running tests using all 13 features. The results of the tests on the Enron dataset, which are detailed further in chapter 5, showed that the models improved as more features were included. Based on those results the remaining tests in this part of the overall experiment are either run using the best 9 ranked features by chi-square or all of the features. No feature selection is applied during the first part of the overall experiment where the baseline classifiers are set nor in the third part of the experiment where both types of feature are combined.

3.5 Training and Sampling technique

For the experiments in this research both hold-out and k-fold cross validation sampling techniques are used. The hold-out sampling technique is used for the baseline classifier primarily because the purpose of that experiment is to set a performance level to judge further classifiers against. Therefore as a relatively straight-forward experiment where no optimisation of parameters is performed hold-out sampling is sufficient. Secondly the datasets used are relatively large and splitting the datasets into subsets still gives enough data to train and test the models. Here all three datasets are split with a training set to test set ratio of 80:20.

For the experiments where only the personality insight and tone features are used by the classifiers 6-fold cross validation will be used. In this experiment a range of parameters are tested to optimise the classifier performance, a validation set is required for this and cross validation facilitates this. Overall the datasets will be split with an 80:20 ratio and 6-fold cross validation will be used on the training set to identify the best parameter settings that are to be applied during classification of the test set. This same approach is followed in the final experiment where the normalised term-frequency features are combined with the personality

and tone features. This use of cross validation on the training set equates to a training: validation: test split of 65:15:20.

For both methods used across all experiments stratified sampling is used to maintain the ham to spam ratio of the overall dataset in the training, validation and test sets created.

3.6 Classification Algorithms and Parameter Settings

In this section the algorithms and related parameter settings used for the three experiments will be discussed. For both the experiment using personality insight and tone features only and the combined experiment a search through a range of parameter settings is performed to find the optimal parameter settings for the algorithms.

3.6.1 Algorithms and Parameter Settings for the Baseline Classifier

Four different classification algorithms are used in this experiment these are a multinomial Naïve Bayes, a standard SVM as well as the LinearSVC and NuSVC variants. Representing the emails as a normalised term-frequency can make the feature-set quite large, so in this part of the experiment the LinearSVC will be implemented to take advantage of its improved training speed. The NuSVC is used to take advantage of the re-parameterisation of the C-value. The bounded range of the **ν**-value allows confirmation of the C-value settings for the other SVCs as judged by the similarity of the accuracy of the NuSVC results with the other classifiers.

No parameter optimisation is performed as part of the baseline classifier experiment so the settings are simply outlined as follows. For the implementation of the multinomial NB classifier the smoothing parameter "alpha" is set to 1. This is the default setting however preliminary testing of the experiment showed that this parameter made little difference to the end accuracy when the range {0.1, 0.2…0.9, 1} was used.

The second parameter set for this classifier is that pre-defined prior probabilities of the dataset are not passed to the classifier, the prior probabilities are learned/adjusted from the underlying data. This was chosen because the experiment is run 6 times and the full dataset is randomly shuffled prior to being split into training and hold out test set. Therefore it is more appropriate to learn the prior probabilities from the shuffled training set used in each iteration rather than passing the prior probabilities of the full dataset before it is shuffled and split.

For the SVM classifiers the main parameter settings are the kernel and the C-value/ **ν**-value. When using SVM classifiers for text analysis the kernel is set to 'linear' for the standard SVC and NuSVC algorithm and it is a set configuration for the Linear SVC algorithm. The C-value selected for standard and linear version of the SVM algorithms is 1.0, this was deemed to be a good starting point with a view to changing the value if the performance was poor. For the NuSVC version **ν**-value is set to 0.05, as above this is deemed a good starting point.

3.6.2 Algorithms and Parameter Range for Personality Insight and Tone Feature Classifiers

In these experiments SVM and Random Forests classifiers are used. To find the optimal parameter setting for the SVM classifier a search through a range of parameter settings is performed. The ranges of parameter values that are tried with the standard SVM classifier are shown in the below matrix. This allows the classifier to be trained on 60 different SVM configurations to find the optimal settings. In this experiment with the SVM classifier the features are numeric values, this allows additional kernel types to be experimented with so the sigmoid and radial basis function kernels are included in this experiment.

C Value	Kernel	Gamma
0.1, 1, 10, 100	Linear	N/A
0.1, 1, 10, 100	Sigmoid	1e-3,1e-2, 1e-1, 1, 1e1, 1e2, 1e3
0.1, 1, 10, 100	Radial Basis Function	1e-3,1e-2, 1e-1, 1, 1e1, 1e2, 1e3

Table 3-3 Parameter Search Ranges for SVM Classifiers

For identifying the best parameters to use with the Random Forests algorithm the below matrix of parameter values was searched through.

No. of trees in random forests	Minimum Sample Split for Internal Nodes	Minimum Samples in a Leaf Node	Criterion
10, 20, 50	2, 3, 10	1	Gini
10, 20, 50	2, 3, 10	1	Entropy

Table 3-4 Parameter Search Ranges for Random Forests Classifiers

3.6.3 Parameter Setting for the Combined Classifier

For the combined experiment only the SVM classifier is used. This classifier was chosen because it was the best performing classifier in the baseline experiment. It was also chosen

because there is a mix of numerical and text based features in the combined feature-set which better suited to SVM. In setting the parameters for that classifier the kernel is set to 'linear', as in the baseline classifier, and only different values for the C-values are experimented with. The range for C-values tested is {0.1, 1, 10, 100}. The various C-values are experimented with using the validation data across all three datasets before selecting one C-value that is set for the classifier when running it against the unseen test data set from each of the three datasets.

3.7 Evaluation Criteria: Average Class Accuracy

For spam datasets the main issue with evaluation is the imbalance in the classes between spam and ham, while this imbalance can vary it is generally the case that for any publicly available datasets the proportion of ham emails will be much higher than the proportion of spam emails. It is therefore possible to mask the true performance of a classifier by selecting the wrong performance measure. For example if a classifier classifies everything as the majority class it might score well if the standard classification accuracy measure is used.

$$Classification\ Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Where:

- TP is true positives
- TN is true negatives
- FP is false positives
- FN is false negatives

To account for the fact that a models poor performance can be masked in this way the average class accuracy will be used.

$$Average\ Class\ Accuracy = \frac{1}{|classes(t)|} \sum_{c \in classes(t)} recall_c$$

Where:

- classes(t) is the set of classes that the target feature, t, can assume
- |classes(t)| is the size of this set, in this case two - spam and ham
- $recall_c$ refers to the recall achieved by a model for class c

To illustrate how using this measure can lead to an incorrect conclusion on a classifiers performance take the below confusion matrices as example results from two different spam classifiers – A and B.

Classifier A			
		Prediction	
		Spam	**Ham**
Actual	**Spam**	810	0
	Ham	81	9

Classifier B			
		Prediction	
		Spam	**Ham**
Actual	**Spam**	630	180
	Ham	18	72

Using the standard classification accuracy measure the accuracy of Classifier A is 91% and of Classifier B is 78%. Based on these results Classifier A would be deemed the better performing classifier even though it classed 90% of the ham examples as spam. The good performance on identification of spam messages completely hides the poor performance on classifying ham emails correctly due to the imbalance in the classes.

If average class accuracy is calculated for these confusion matrices then the results of the two models are 55% for Classifier A and 78% for Classifier B which would lead to a different conclusion, i.e. that Classifier B is the better performer.

3.8 Conclusion

In this chapter the experiment methodology has been discussed, setting out how the experiments will be run, the data that will be used, the techniques that will be employed and the metric by which performance will be judged. The following chapter will expand further on the design and implementation stages of the experiments.

4 Experiment Design

4.1 Introduction

This chapter will detail the design of the experiments which are the subject of this research, it can be divided into three sections. The first section outlines the software used to develop the experimental framework. The second section discusses the data preparation for the normalised term-frequency feature-sets used in the baseline classifiers as well as how the personality insight and tone features are created.

Finally the third section outlines the end to end flows of the experiments, three experiment flows will be described. The first is the "baseline" classifier which is used to set a baseline level of performance against which the performance of the classifiers that utilise the personality insight and tone features can be judged. The second experiment flow concerns classifiers that *only* use the new features to classify emails as spam or ham. The third experiment flow covers the "combined" classifier i.e. where both normalised term-frequency features and personality insight and tone features are used.

4.2 Software Used

4.2.1 Python

The Python language version 2.7 was used to implement the entirety of this experimental framework. Python, along with R, are the two most is widely used and sought after languages in data science[6] [7].Python's popularity stems initially from its flexibility with respect to data manipulation or "wrangling" followed by its range of packages for data analysis, machine learning and visualisation. In addition to this it is open source, easily accessible and supported by a wide range of online courses and communities. Python v2.7 was chosen at the outset over Python v3.0 because many books, online courses, tutorials and community resources are still written in the context of v2.7. Furthermore v3.0 does not have the range of packages that v2.7 has as many of the more bespoke packages have not been migrated to v3.0.

The key Python packages used as part of this experiment are Email, Pandas, JSON and NumPy which are used for the manipulation of raw and partially pre-processed emails into lists and tables of data that are basis of the analysis and experimentation to be performed. NLTK (the

[6] (Piatetsky, 2016)
[7] (Puget, 2016)

Natural Language ToolKit) is used for the pre-processing of email body text from the various email datasets into feature-sets so that the data can be used by the different classification algorithms.

The scikit-learn package primarily provides the implementations of the different machine learning techniques and algorithms used. There is some overlap of NLTK and scikit-learn for this purpose in the "baseline" classifier as NLTK provides an API into scikit-learn and this linkage is used for this classifier. For the personality insight and tone feature classifiers algorithms from scikit-learn are solely used for the classification task as well as for feature selection, parameter tuning and some production of result metrics. Finally MatPlotLib is used for the visualisations.

For this experiment Python is run through the Anaconda Navigator platform where the integrated development environment Spyder v3 was used to write and implement all of the code.

4.2.2 IBM Watson Tone Analyszr API

As part of IBM's Watson Developer Cloud they offer a Tone Analyzer that can be accessed via an API call, this tone analyser is the source of the personality insight and tone features used in the experiment. Once credentials are established with IBM then a string of text can be sent to the Tone Analyser and a JSON object (see below) is returned with thirteen numerical tone features split into 3 tone categories that reflect the tone scores for that string of text. This can be done on a sentence by sentence basis or for the entire block of text sent. For simplicity the tone features gathered for analysis in this experiment relate to the entire block of text in an email not the individual sentences.

4.2.3 JSON

JSON (JavaScript Object Notation) is a human-readable format for transferring data that is designed to be easily parsed by machines. While it is JavaScript based it is language agnostic and uses conventions that are familiar to programmers of C, Java, and Python etc. [8]. The structure of the JSON object returned from the IBM Tone Analyzer is a collection of nested {name:value} pairs, i.e. the tone feature and its numerical value. This will be discussed in more detail below.

[8] (JSON, n.d.)

4.3 Data Preparation

In this section the data preparation and representation steps applied to the three datasets will be discussed and the section can be divided into two parts. The steps required to transform the raw emails into normalised term-frequency features will be covered first followed by the steps followed to turn the same emails into personality insight and tone features. Further detail and background on the personality insight and tone features will also be provided as well as the discretisation method applied to those features.

4.3.1 Normalised Term-Frequency Representation

Not all information in an email is useful for classifying it as spam or ham. In their 2009 paper Guzella and Caminhas outline four pre-processing steps which are common to spam filters, these *"can be grouped into:*

(1) Tokenization, which extracts the words in the message body;
(2) Lemmatization, reducing words to their root forms (e.g. "extracting" to "extract");
(3) Stop-word removal, eliminating some words that often occur in many messages (e.g. "to", "a", "for");
(4) Representation, which converts the set of words present in the message to a specific format required by the machine learning algorithm used."

These pre-processing steps have the dual purpose of firstly removing noisy data that do not have a classification value and secondly converting the relevant information into a form that the classifier can comprehend, e.g. a Boolean vector of word occurrences in order to discern whether an email is ham or spam.

In this instance the item being represented is an email which is made up of two parts - the header and the body. The header comprises of fields such as the subject, the senders name, sending date, an email ID, routing information etc. While this information can be useful in classifying an email in its own right the focus of this experiment is on the text data, i.e. the body content of the email, so header information is ignored in all experiments.

The pre-processing steps used to prepare the three data-sets for the baseline classifier follow the four steps above. Only the data preparation for the Enron dataset differs in that HTML tags did not need to be removed from the email body text for this dataset. After the first three steps the string of text is then converted to a normalised term-frequency representation.

Representation by normalised term-frequency means that for each email the number of times a word appears is counted. To normalise the feature-set the term-frequency of the highest appearing word is set equal to one, by dividing its term-frequency by itself, and every other word in the term-frequency of that email is also divided by that same value. This representation is used for the multinomial Naïve Bayes and the SVM classifier variations used in the experiments.

4.3.2 IBM Tone Features

IBM describe their Tone Analyzer as a "*service [that] uses linguistic analysis to detect three types of tones from written text: emotions, social tendencies, and writing style*"[9]. The Tone Analyzer returns thirteen numeric features under the three types of tone categories mentioned.

The Tone Analyzer is based on research from the psycho-linguistics field. The ideas explored in this field are that "*the words we use in daily life reflect who we are and the social relationships we are in. Language is the most common and reliable way for people to translate their internal thoughts and emotions into a form that other can understand. Words and language, then, are the very stuff of psychology and communication. They are the medium by which cognitive, personality, clinical and social psychologists attempt to understand human beings."* (Tausczik, 2010)

4.3.3 Emotional Tone

This element uses natural language processing concepts and ensemble machine learning techniques to create its scores. The text provided is represented as features such as n-grams and sentiment polarity and these features are used by an ensemble classifier to derive the "*types of emotions and feelings that people express in text*"[10] and return emotional scores from the text.

The scores range from 0.0 – 1.0 and reflect the likelihood of the emotion being perceived in the text. Scores less than 0.5 reflect a low likelihood of the emotion being perceived whereas scores over 0.75 reflects a high likelihood. A low score is still of value in this experiment as it indicates a lack of a feature so all values within the range are used in the models that use these features.

[9] (IBM Watson Developer Cloud, 2017)
[10] (IBM Watson Developer Cloud, 2017)

Emotional Tone Feature	Description / High value score indication
Joy	Joy or happiness has shades of enjoyment, satisfaction and pleasure.
Fear	A response to danger or negative stimulus. A survival mechanism with reactions ranging from mild caution to extreme phobia.
Sadness	Indicates a feeling of loss and disadvantage. A person who is quiet, less energetic and withdrawn, may be showing signs of sadness.
Disgust	An emotional response of revulsion to something considered offensive, revolting or unpleasant.
Anger	Evoked due to injustice, conflict, humiliation, negligence or betrayal. If anger is active, the individual attacks the target, verbally or physically, if passive the person silently sulks and feels tension and hostility.

Table 4-1 Description of Emotional Tone Features[11]

4.3.4 Social Tone

In the Tone Analyzer the social tone features are based on the "Big Five Personality Model" or the "Five Factor Model (FFM)". This model is based on foundational work from the early 1960s by, Ernest Tupes, Raymond Christal and Warren Norman who identified the five features outlined below. Though they used the term "dependability" instead of "conscientiousness". Since then the model and these five traits are a well-established and accepted representation of personality traits.

The IBM Tone Analyzer provides scores for the five traits in the range 0.0 to 1.0 but the scores should be viewed more as a reflection of a spectrum than a score of intensity or "how much" of the tone is present. Where one end of the spectrum, e.g. more than 0.75, indicates a set of traits associated with how that tone would be defined while scores less than 0.5 would indicate the other end of the spectrum or how the opposite of that tone would be defined.

[11] (IBM Watson Developer Cloud, 2017)

Social Tone Feature	Description	Low value score indicates	High value score indicates
Openness	The extent a person is open to experience a variety of activities.	More likely to be perceived as no-nonsense, straightforward, blunt, or preferring tradition and the obvious over the complex, ambiguous, and subtle.	More likely to be perceived as intellectual, curious, emotionally-aware, imaginative, willing to try new things, appreciating beauty, or open to change.
Conscientiousness	The tendency to act in an organized or thoughtful way.	More likely to be perceived as spontaneous, laid-back, reckless, unmethodical, remiss, or disorganized.	More likely to be perceived as disciplined, dutiful, achievement-striving, confident, driven, or organized.
Extraversion	The tendency to seek stimulation in the company of others.	More likely to be perceived as independent, timid, introverted, restrained, boring, or dreary.	More likely to be perceived as engaging, seeking attention, needy, assertive, outgoing, sociable, cheerful, excitement-seeking, or busy.
Agreeableness	The tendency to be compassionate and cooperative towards others.	More likely to be perceived as selfish, uncaring, uncooperative, self-interested, confrontational, sceptical, or arrogant.	More likely to be perceived as caring, sympathetic, cooperative, compromising, trustworthy, or humble.

Emotional Range (Neuroticism)	The extent a person's emotion is sensitive to the environment.	More likely to be perceived as calm, bland, content, relaxed, unconcerned, or careful.	More likely to be perceived as concerned, frustrated, angry, passionate, upset, stressed, insecure, or impulsive.

Table 4-2 Description of Social Tone Features[12]

4.3.5 Language Tone

These tone features describe a writing style and the tone is identified and scored by a linguistic analysis based classifier. The training data used by this classifier was created by taking approximately a thousand sentences for each feature and crowd sourcing the manual labelling on a platform called CrowdFlower. Only platform users with high approval ratings from the platform could vote and the majority vote out of 5 votes was used for the label. The scores range from 0.0 to 1.0 and like the emotional tone features the score is more like an intensity level where a low value score indicates little or no evidence of the tone. Again a low score is still of value in this experiment so all values are used.

Language tone	Description	High value score indicates
Analytical	A person's reasoning and analytical attitude about things.	More likely to be perceived as intellectual, rational, systematic, emotionless, or impersonal.
Confidence	A person's degree of certainty.	More likely to be perceived as assured, collected, hopeful, or egotistical.
Tentative	A person's degree of inhibition.	More likely to be perceived as questionable, doubtful, or debatable.

Table 4-3 Description of Language Tone Features[13]

[12] (IBM Watson Developer Cloud, 2017)
[13] (IBM Watson Developer Cloud, 2017)

4.3.6 Creating Tone Features from the IBM Tone Analyzer API

The process of converting the body content of each email in the datasets into tone features involved extracting the body content and sending it as a string to the IBM Tone Analyzer. After that the personality insight and language tone scores are parsed from the returned JSON object for that string and finally those values are appended into a dataframe for analysis.

The first steps of for creating the personality insight and tone features are similar to those for creating the earlier feature vectors in that the text is extracted from the raw emails though no tokenisation etc. is performed. As an example, below is an email string from the Enron dataset:

> *"people are getting rich using this system ! now it ' s your turn ! we ' ve cracked the code and will show you . . . .this is the only system that does everything for you , so you can make money . . . . . .because your success is . . . completely automated ! let me show you how ! click here"*

After this is sent to the IBM Tone Analyzer the below JSON object is returned:


```
{document_tone:
{tone_categories:
[
{category_id: emotion_tone, tones:
        [
        {tone_name: Anger, score: 0.114473, tone_id: anger},
        {tone_name: Disgust, score: 0.084797, tone_id: disgust},
        {tone_name: Fear, score: 0.109672, tone_id: fear},
        {tone_name: Joy, score: 0.465742, tone_id: joy},
        {tone_name: Sadness, score: 0.256469, tone_id: sadness}
        ],
category_name: Emotion Tone},
{category_id: language_tone, tones:
        [
        {tone_name: Analytical, score: 0.659839, tone_id: analytical},
        {tone_name: Confident, score: 0.739476, tone_id: confident},
        {tone_name: Tentative, score: 0.0, tone_id: tentative}
```

],
category_name: Language Tone},
{category_id: social_tone, tones:
[
{tone_name: Openness, score: 0.343598, tone_id: openness_big5},
{tone_name: Conscientiousness, score: 0.921018, tone_id: conscientiousness_big5},
{tone_name: Extraversion, score: 0.941085, tone_id: extraversion_big5},
{tone_name: Agreeableness, score: 0.363024, tone_id: agreeableness_big5},
{tone_name: Emotional Range, score: 0.944578, tone_id: emotional_range_big5}
],
category_name: Social Tone}]}}

From this JSON object a parsing routine populated a dataframe containing email ID, email text and label with the personality insight and tone features for each email in the dataset and so the new feature-set for analysis and classification is created.

4.3.7 Technical Limitations

The IBM Tone Analyzer has a number of input constraints that are problematic for datasets of the size used in this experiment. Firstly the API applies "throttling" meaning that a user cannot send over strings of text one directly after the other without any breaks in the flow. Where the API identifies a continuous flow it simply stops sending back the JSON object and sends an error message.

In order to stem the flow of strings sent to the API each dataset had to be separated into blocks of 500 emails before the body content from the emails in each block could be sent. This created a pause in the flow of emails sent and allowed the API call to run for the size of the datasets used in this experiment.

Secondly the API returned an error where the string size was greater than 128KB, so where the body content was greater than this amount it was not sent and zeros were applied as tone features. An alternative option could have been to split the larger string into sections of less than 128KB and then average the parsed tone scores for each feature prior to appending them to the dataframe. This relatively complex operation was not pursued as the number of emails was too small to warrant the additional complexity.

	Enron	CSDMC	SpamAssassin
Total Emails	13684	4327	6047
Size > 128KB	60	32	57
% of Total	0.4%	0.7%	0.9%

Table 4-4 Emails above technical limit of IBM Tone Analyzer

4.4 Experiment Parts

4.4.1 Introduction

This overall experiment is composed of three parts. In the first part a "baseline classifier" is constructed to set a level that a classifier using tone features must outperform to be of value. In the second part personality insight and tone features are investigated to see if they can be used on their own to filter spam, this is referred to as the "Tone Feature" classifier. Finally in the third part the personality insight and tone features are combined with the feature-set used in the baseline classifier to investigate if a classifier can use the combined features to outperform the baseline classifier.

4.4.2 Baseline Classifier Design

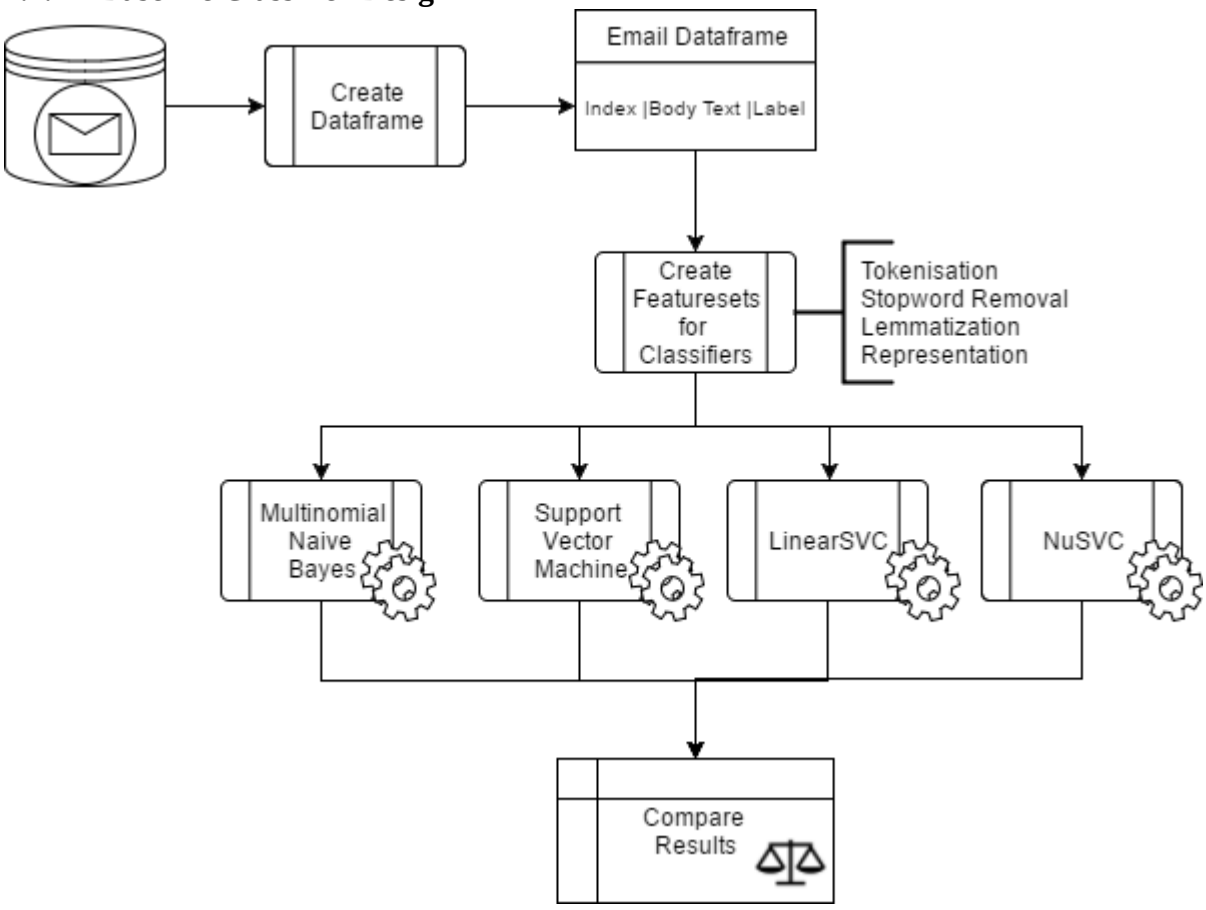


Figure 4-1 Baseline Classifier Design Diagram

4.4.3 Baseline Classifier Design Description

As a high-level description the baseline classifier consists of four stages:

1. The body content of each email in the dataset is extracted and loaded into a dataframe consisting of the text and class label for that email.
2. The feature-set is created by applying the pre-processing steps to represent the email in a format the classifiers can use.
3. The classifiers selected for this experiment are trained on the feature-set to classify the examples in the test set as ham or spam.
4. The classification results are evaluated using a consistent measure so the performance of the classifiers can be compared. The classifier which performs best is chosen as the baseline classifier.

4.4.4 Tone Feature Classifier Design

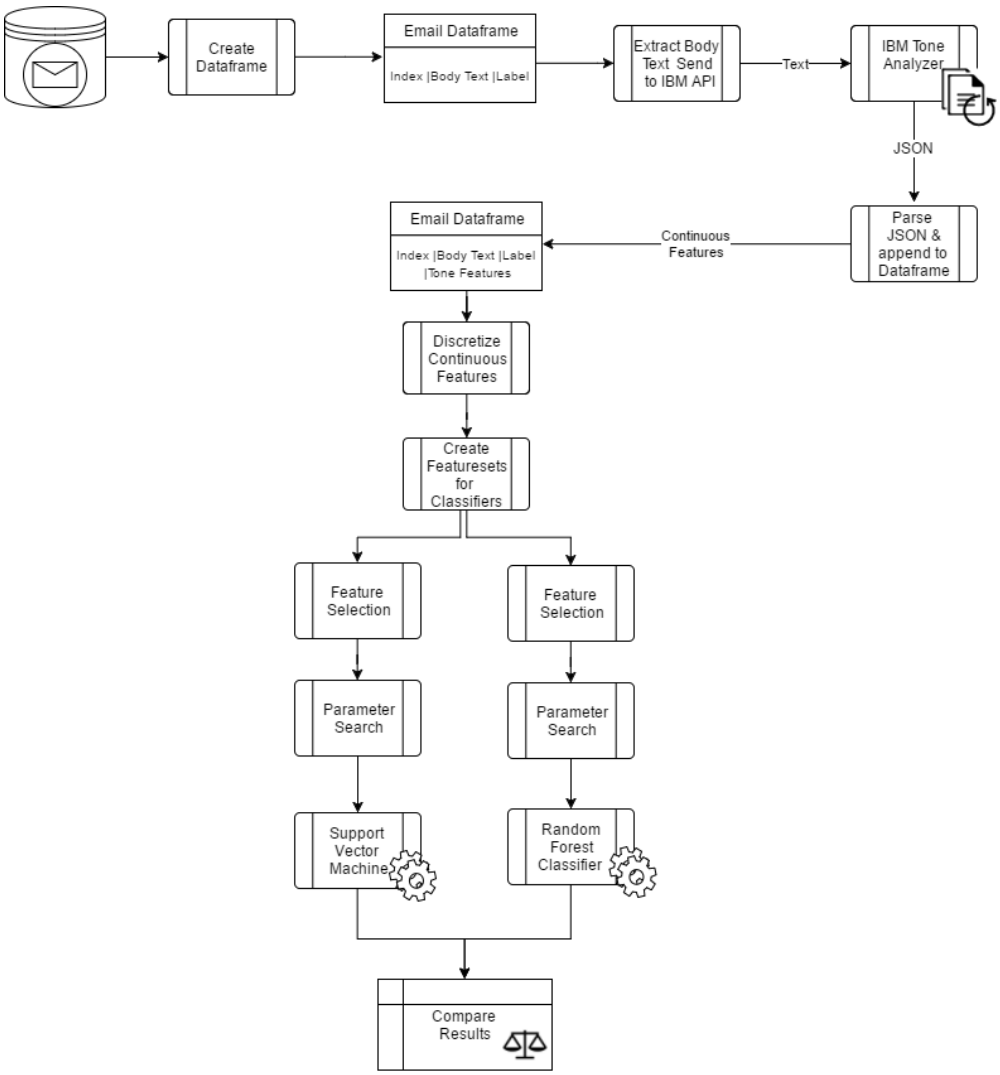


Figure 4-2 Tone Feature Based Classifier Design Diagram

4.4.5 Tone Feature Classifier Design Description

The tone feature classifier replicates many of the same steps as the baseline classifier, overall there are five stages in this process:

1. As per the baseline classifier text is extracted and labelled.
2. The personality insight and language tone features are created. Each block of text is sent to the IBM Tone Analyzer. The personality insight and language tone scores are parsed out of the returned JSON object and appended to the dataframe of labelled text. Lastly the continuous personality and insight tone scores are discretised to make them categorical features completing the data preparation and representation.
3. Two feature selection methods are applied to attempt to identify the most useful features for classification.
4. The SVM and Random Forests classifiers are trained. During training a grid search through a range of parameter values is performed to find the best parameter settings for these classifiers.
5. After the two classifiers have been trained they are tested and their performance is compared to the baseline classifier.

4.4.6 Combined Features SVM Classifier Design

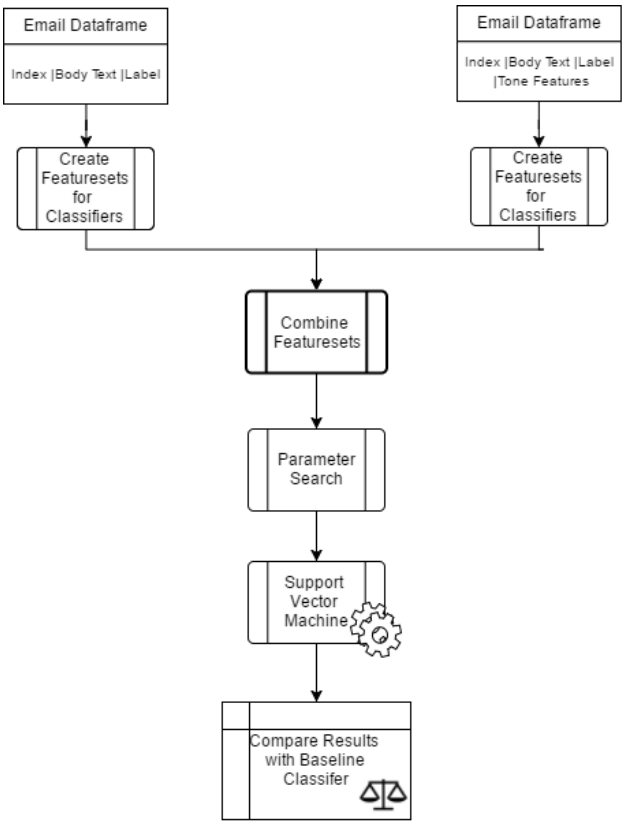


Figure 4-3 Combined Feature Classifier Design Diagram

4.4.7 Combined Features SVM Classifier Description

For the third part of the experiment the combined feature-set is used with the standard SVM classifier to investigate if any improvement on the baseline classifiers performance can be made.

1. The data preparation steps are the same, the only difference here is that the two feature-sets are combined.
2. No feature selection is performed before the model is trained. Following training a search through a range of parameter values is performed using the validation set to find the best parameter settings for this classifier.
3. Once the best parameter settings for the classifier are found it is run against the test sets and the performance compared to the baseline classifier.

4.5 Conclusion

In this chapter three topics have been detailed. Firstly the software used to develop the experimental framework was discussed. Secondly the preparation of the datasets has been detailed, starting with the representation of emails as normalised term frequencies for the baseline classifiers. Following this the features provided by the IBM Watson Tone Analyzer and how the returned scores were turned into a feature-set to be used in the experiments was explained. Finally the design of the three experiments that will be implemented for this project have been described.

5 Experimentation and Evaluation

5.1 Introduction

The purpose of this chapter is to present the results of the three parts of the experiment to investigate if the personality insight and tone features, created using the IBM Watson Tone Analyzer, can improve spam filtering. This is the main element of this chapter though also discussed is the data exploration of the new features values for the two classes.

5.2 Comparison with a Baseline Model

If the aim of any machine learning experiment is to investigate whether a change in features or algorithms improves a model by some measure, then the purposes of the baseline model is to provide a reference to compare the "improved" model against. As discussed earlier two of the most commonly used algorithms in spam classification are Naïve Bayes and Support Vector Machines. The three datasets used in this experiment will be tested against a Naïve Bayes classifier and three SVM classifier variants – the standard, LinearSVC and NuSVC classifiers.

5.2.1 Baseline Model Results

The results for each of the classifier and dataset combinations below represent the average accuracy result over the six iterations of the experiments.

Classifier	Enron	Spam Assassin	CSDMC 2010
Multinomial Naïve Bayes	91.21	87.44	86.34
Support Vector Classifier	97.51	**96.75**	**96.61**
Linear SVC	**97.76**	96.35	95.54
NuSVC	97.45	96.60	96.43

Table 5-1 Baseline Classifier Average Class Accuracy Results

The general approach to training classifiers is to run the algorithm on the training set and then optimise the parameters on a validation set prior to running against the test set (Drucker, Wu, & Vapnik, 1999). In this instance all the classifiers performed well with respect to accuracy on the training sets. In particular the NuSVC variation performs similarly to the other SVM variants indicating that the C-values set for the other classifiers are adequate. Furthermore because the training results were acceptable and the aim of this experiment is to set a baseline level of performance no further optimisation was required. In this instance the trained

classifiers are run against the test sets and the results above show that the classifiers have performed well and generalised well across the three datasets.

Over the 3 datasets the SVM classifiers perform better than the Naïve Bayes classifier on average class accuracy. The standard variation of SVM performs best on two out of the three datasets though there is very little difference between the results of the three SVM variations within or between the datasets. Therefore it is difficult to choose one variation of the algorithm over another.

If the personality insight and tone features are to prove indicative of spam or ham then the standard support vector classifier will be used in a model where the normalised term-frequency features and new features are combined. It will, however, be difficult to improve upon the already high average class accuracy results.

5.3 Tone Data Exploration & Analysis

5.3.1 Introduction

The first step in investigating if tone features derived from the body text can be used for spam filtering is to get a sense of the data. Through visualisations and standard statistics an initial investigation of the tone data for each of the three data sets is undertaken.

Since the IBM Watson Tone Analyzer will be looking at the text of the emails and returning personality insight and language tone scores the most common words in each class were visualised to get a sense of the vocabulary of each class.

These visualisations show that the most common words in the Enron ham and spam classes are different, however there are no obvious words or adjectives in either visualisation that imply a difference in tone. By creating similar visualisations for the other datasets a difference in the vocabularies is clear.

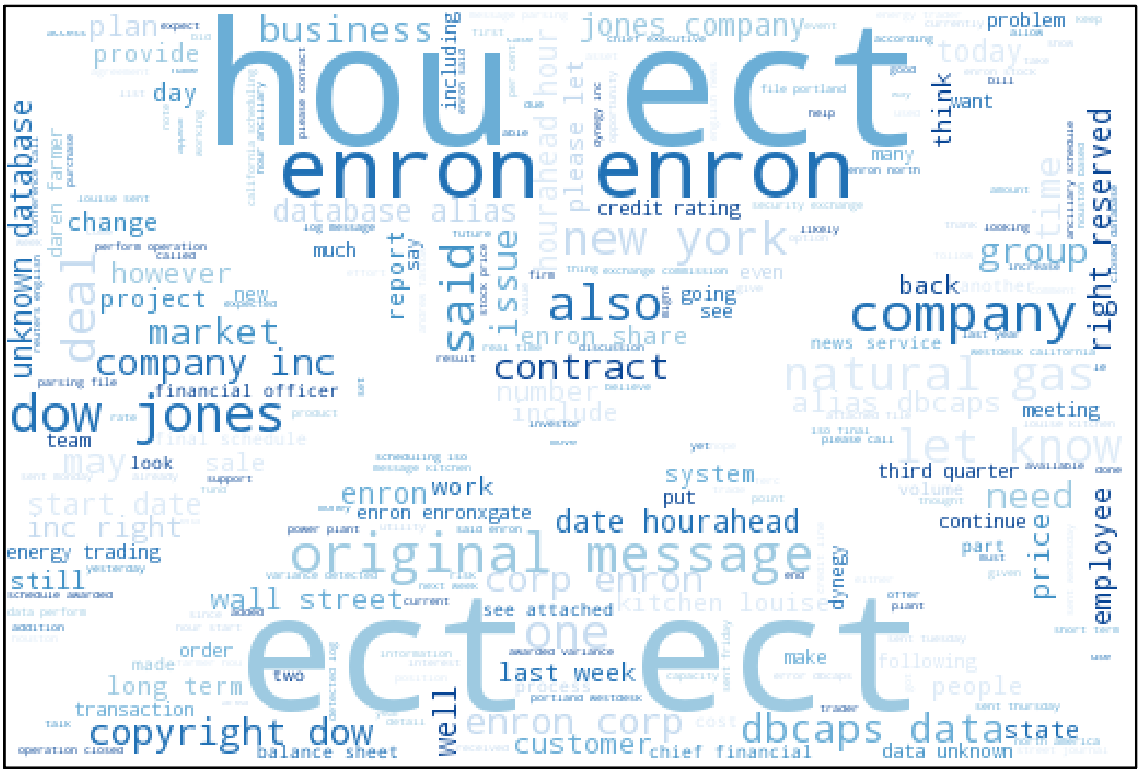


Figure 5-1 Most Common Words in Enron Ham Class

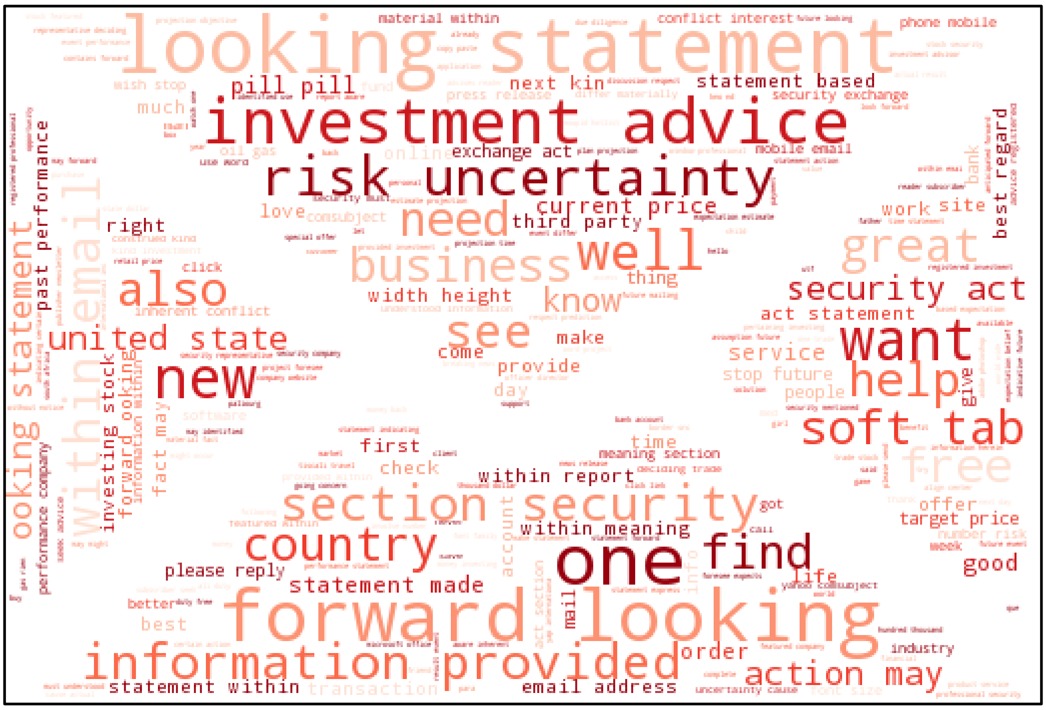


Figure 5-2 Most Common Words in Enron Spam Class

Figure 5-3 Most Common Words in CSDMC2010 Ham Class

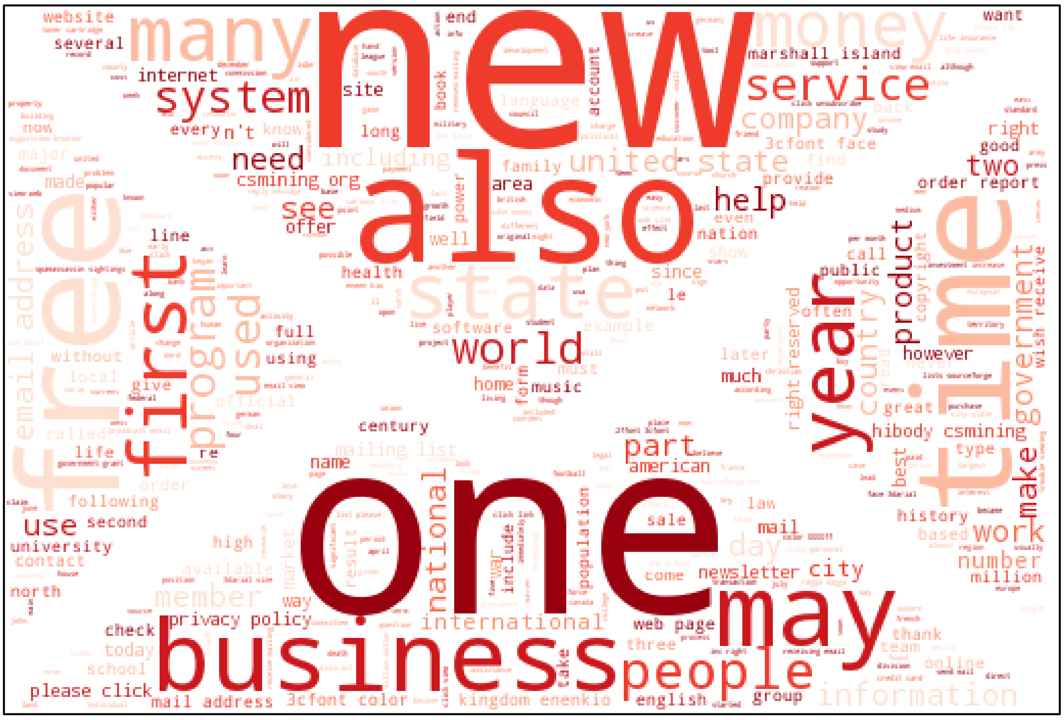


Figure 5-4 Most Common Words in CSDMC2010 Spam Class

Figure 5-5 Most Common Words in Spam Assassin Ham Class

Figure 5-6 Most Common Words in Spam Assassin Spam Class

While a difference in vocabularies is shown it is not yet clear if this will translate into a difference in the personality insight and language tone scores. The following sections will investigate the scores returned to understand if a difference between the classes is visible.

In the below three tables, one for each of the datasets, the mean and standard deviations (S.D) for the thirteen personality insight and tone features are presented. These statistics are presented for spam and ham emails separately. Additionally the differences between those spam and ham statistics are presented. For these figures the darker shades indicate that the differences are lowest, i.e. where the values of the mean and standard deviations of a tone feature are very similar for the spam and ham instances.

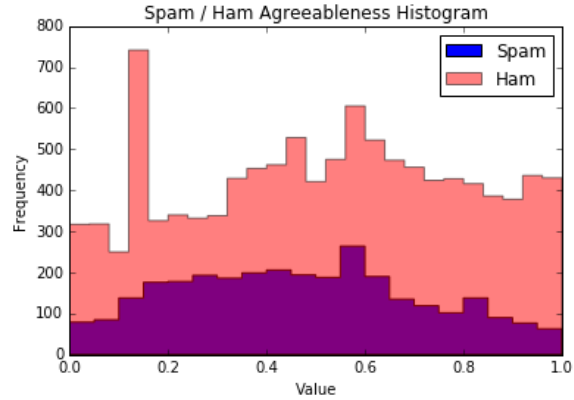
Looking at the features for the Enron dataset in table 5.2 some inferences can be drawn from differences between the ham and spam statistics. For example for the tone features like Anger and Fear, in the emotional tone category, there is little difference between the range and spread of values across the ham and spam examples. This suggests they are unlikely to have any classification value. Conversely for features of the social tone category, such as Conscientiousness and Extraversion, the statistics suggest that these features may have classification value because there are larger differences in the mean and standard deviation figures.

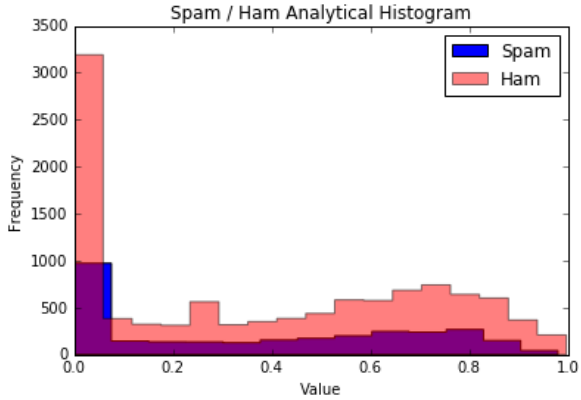
Enron Dataset														
		Emotional Tone					Language Tone			Social Tone				
		Anger	Disgust	Fear	Joy	Sadness	Analytical	Confident	Tentative	Openness	Conscientiousness	Extraversion	Agreeableness	Emotional Range
Ham	count	10684	10684	10684	10684	10684	10684	10684	10684	10684	10684	10684	10684	10684
	mean	0.205	0.131	0.165	0.400	0.344	0.392	0.089	0.296	0.355	0.501	0.507	0.515	0.445
	S.D.	0.133	0.109	0.118	0.161	0.144	0.327	0.225	0.334	0.252	0.283	0.294	0.276	0.285
Spam	count	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000
	mean	0.202	0.145	0.164	0.427	0.314	0.369	0.161	0.233	0.305	0.440	0.439	0.471	0.456
	S.D.	0.135	0.127	0.126	0.166	0.167	0.317	0.286	0.304	0.250	0.305	0.265	0.244	0.317
Diff.	count	13684	13684	13684	13684	13684	13684	13684	13684	13684	13684	13684	13684	13684
	mean	0.003	0.013	0.001	0.027	0.030	0.022	0.072	0.062	0.050	0.061	0.068	0.044	0.012
	S.D.	0.002	0.018	0.008	0.005	0.023	0.010	0.061	0.029	0.002	0.022	0.029	0.032	0.031

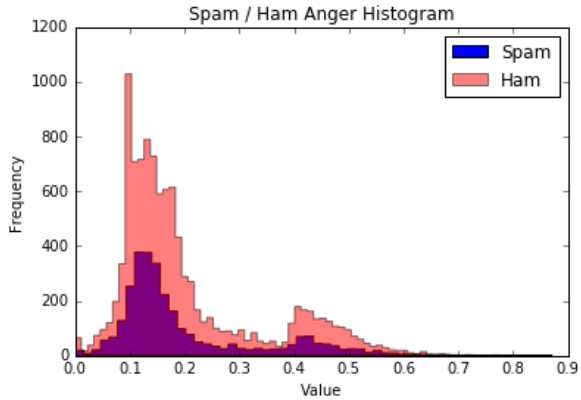
Table 5-2 Class Split of Mean and Standard deviations of tone features: Enron

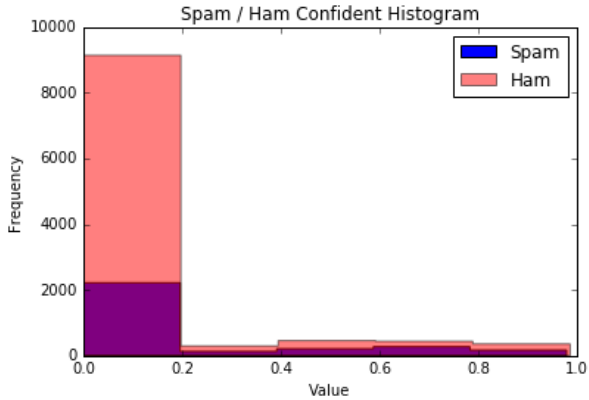
When these same tone features are plotted as histograms, bearing in mind that the rough split of examples is 70:30 ham to spam, it gives a similar visual representation as the above tables. This supports the view that there is a similarity between many of the values of the personality

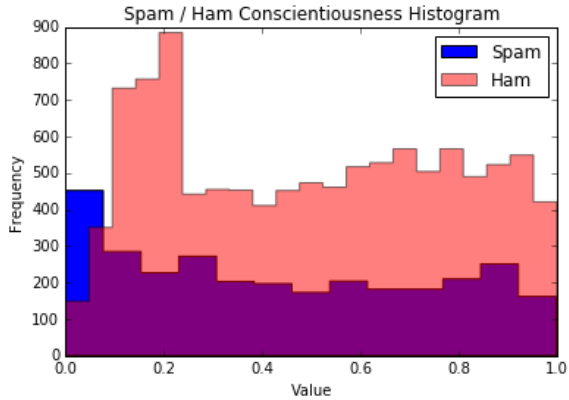
insight and tone features in both classes though some differences are visible, i.e. Conscientiousness and Extraversion are examples of this.

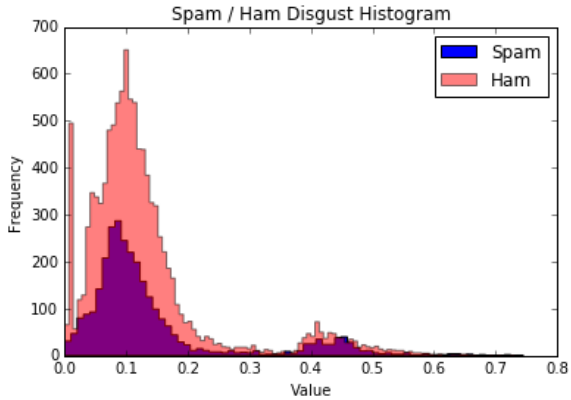


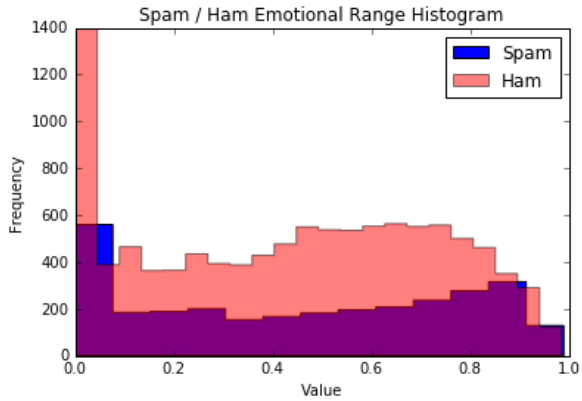


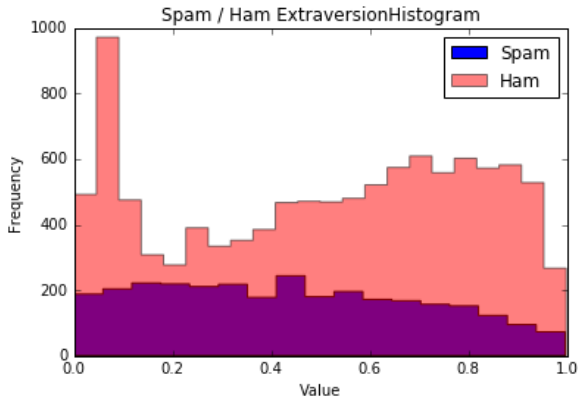












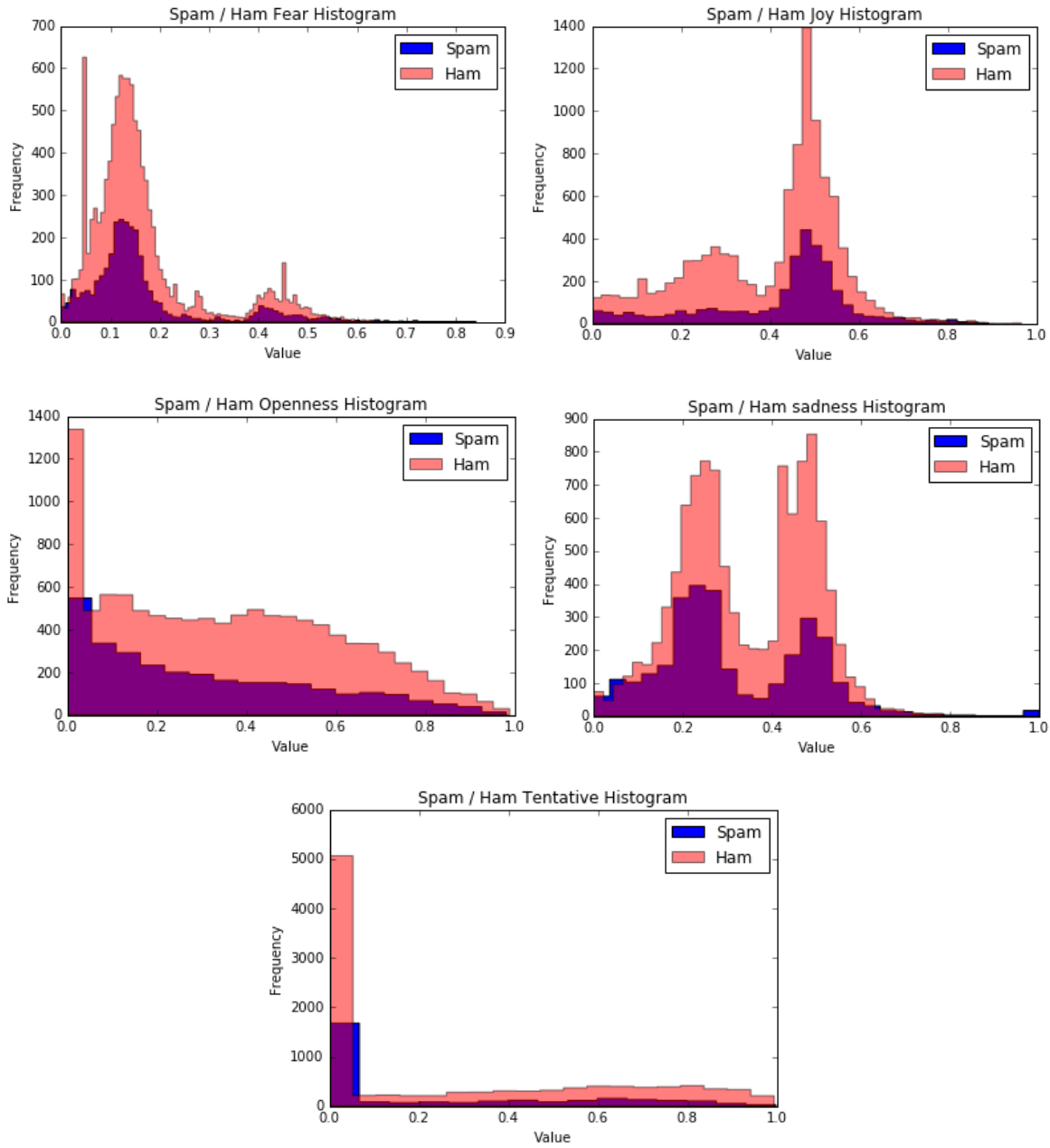


Figure 5-7 Enron Tone Features: Class-wise split of values

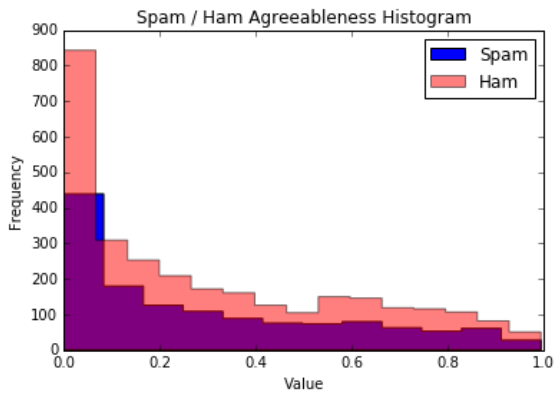
A similar review of the CSDMC2010 dataset displays a similar pattern with respect to the statistics calculated. In the below table it can be seen that the differences between the mean and standard deviations for the ham and spam data are very close together across all of the features. The range of differences between the ham and spam mean figures is 0.001 to 0.012, the same range for the Enron features is 0.001 to 0.068. For the ham and spam standard deviations the range of differences is 0 to 0.009, or the Enron figures this range is 0.002 to

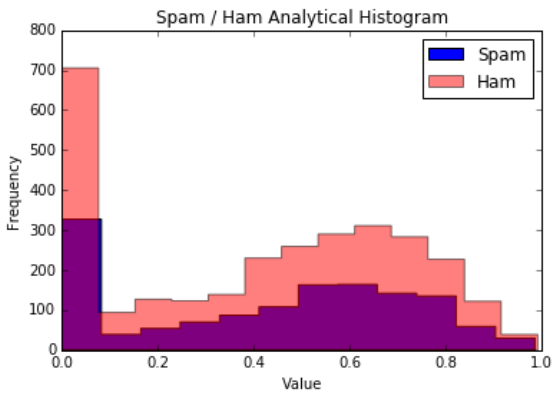
0.032. Furthermore of particular note there are three features, Joy, Openness and Extraversion, where the difference between the ham and spam standard deviations is 0.

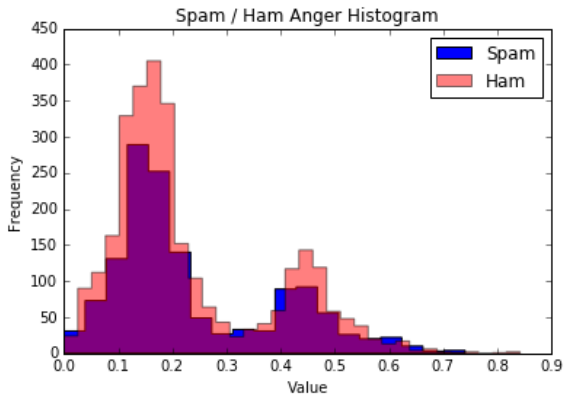
CSDMC 2010 Dataset														
		Emotional Tone					Language Tone			Social Tone				
		Anger	Disgust	Fear	Joy	Sadness	Analytical	Confident	Tentative	Openness	Conscientiousness	Extraversion	Agreeableness	Emotional Range
Ham	count	2949	2949	2949	2949	2949	2949	2949	2949	2949	2949	2949	2949	2949
	mean	0.231	0.124	0.185	0.404	0.341	0.422	0.078	0.392	0.411	0.343	0.381	0.309	0.539
	S.D.	0.148	0.106	0.125	0.180	0.175	0.296	0.221	0.341	0.265	0.276	0.269	0.286	0.268
Spam	count	1378	1378	1378	1378	1378	1378	1378	1378	1378	1378	1378	1378	1378
	mean	0.238	0.118	0.188	0.403	0.343	0.429	0.072	0.380	0.402	0.339	0.384	0.300	0.543
	S.D.	0.153	0.101	0.129	0.180	0.178	0.295	0.212	0.335	0.265	0.280	0.269	0.281	0.266
Differences	count	4327	4327	4327	4327	4327	4327	4327	4327	4327	4327	4327	4327	4327
	mean	0.007	0.006	0.003	0.001	0.002	0.007	0.006	0.012	0.008	0.004	0.004	0.009	0.005
	S.D	0.005	0.005	0.004	0.000	0.003	0.002	0.009	0.005	0.000	0.004	0.000	0.005	0.002

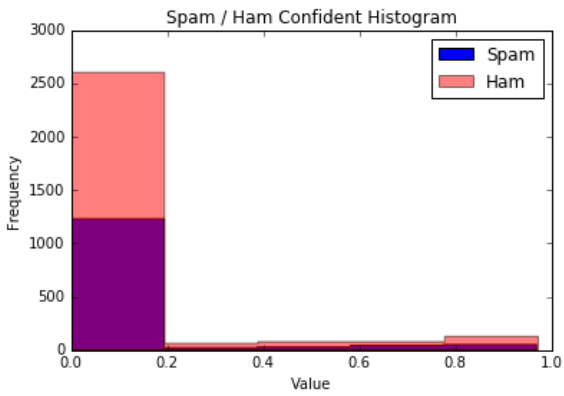
Table 5-3 Class Split of Mean and Standard deviations of tone features: CSDMC

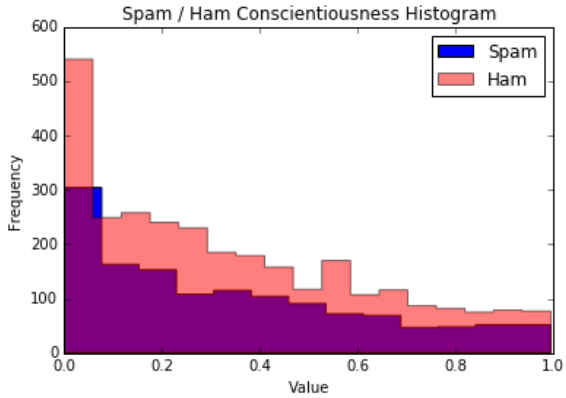
This similarity is visible when the features are graphed by class in the below histograms

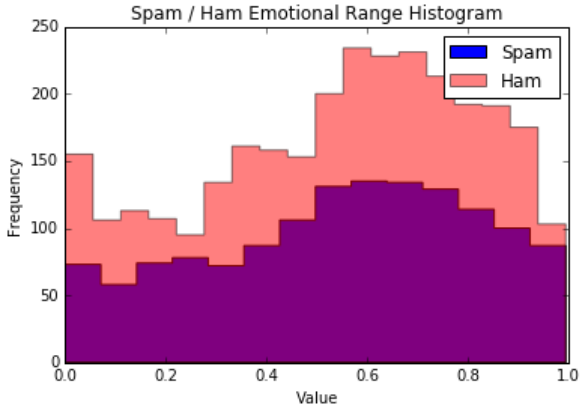


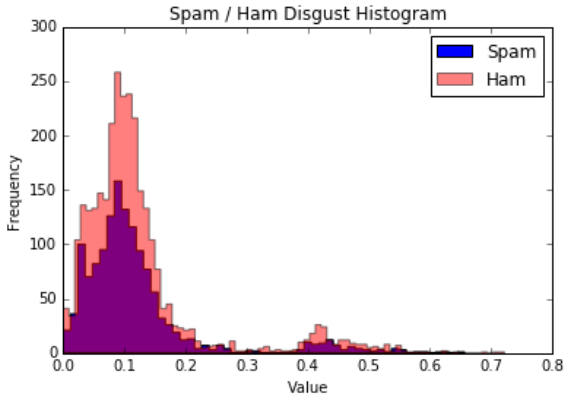


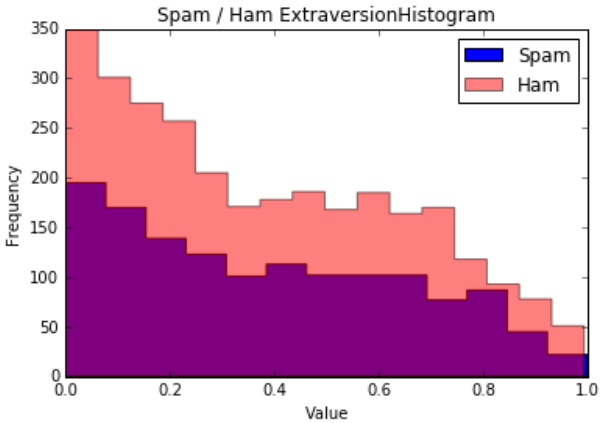

Spam / Ham Anger Histogram
Spam
Ham
Frequency
Value

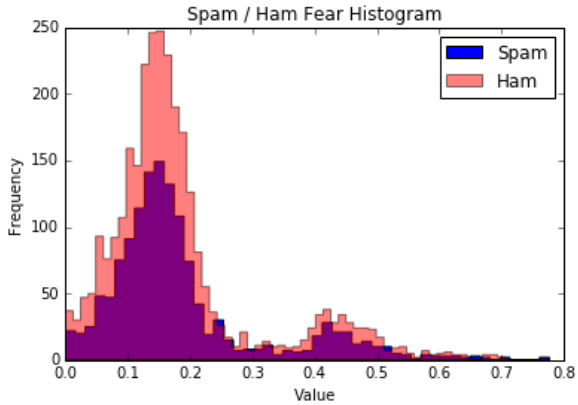

Spam / Ham Confident Histogram
Spam
Ham
Frequency
Value

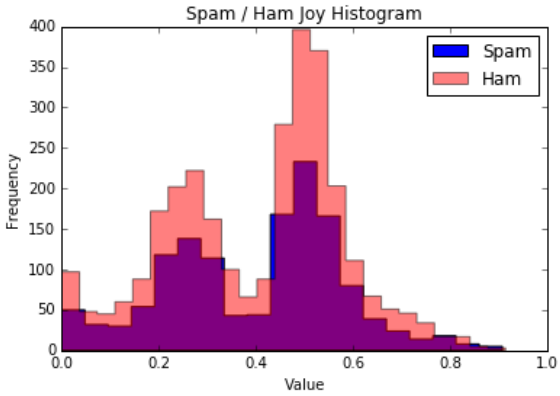

Spam / Ham Conscientiousness Histogram
Spam
Ham
Frequency
Value


Spam / Ham Emotional Range Histogram
Spam
Ham
Frequency
Value


Spam / Ham Disgust Histogram
Spam
Ham
Frequency
Value


Spam / Ham ExtraversionHistogram
Spam
Ham
Frequency
Value


Spam / Ham Fear Histogram
Spam
Ham
Frequency
Value


Spam / Ham Joy Histogram
Spam
Ham
Frequency
Value

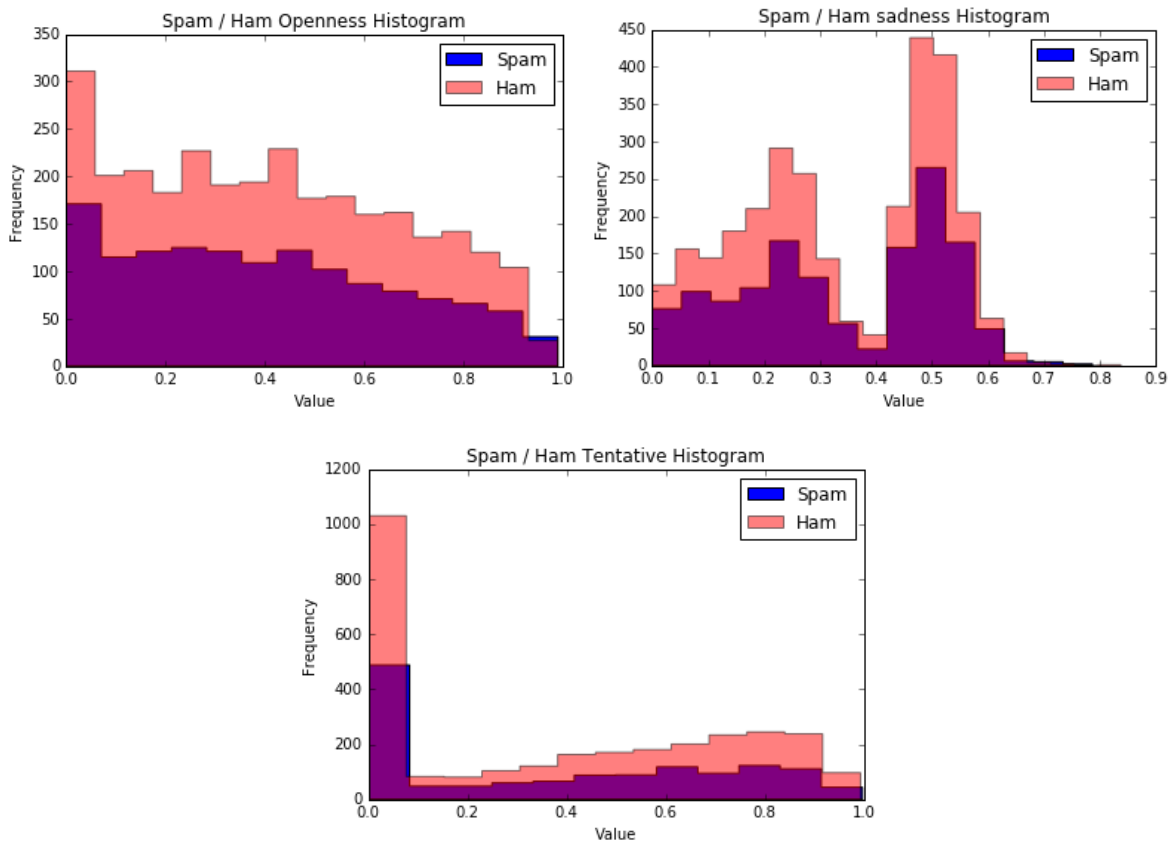


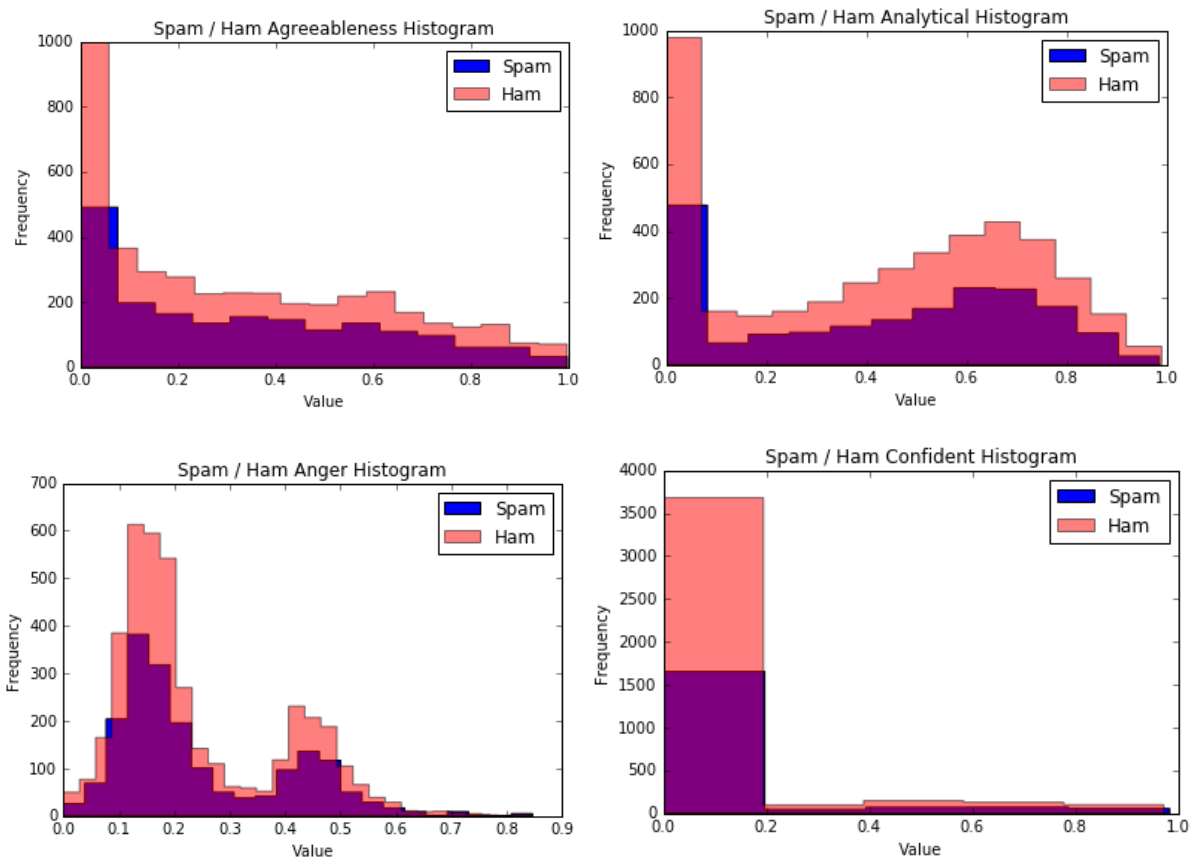
Figure 5-8 CSDMC2010 Tone Features: Class-wise split of values

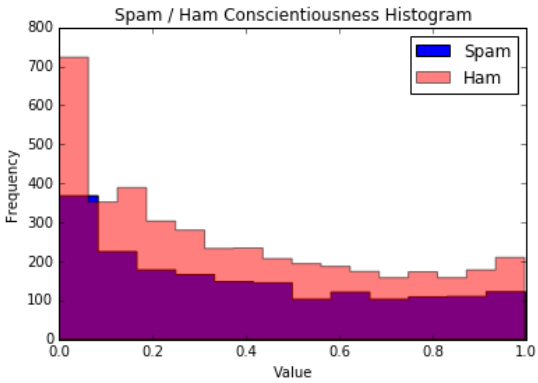
The same table of figures for the Spam Assassin Dataset shows a similar picture to the other two datasets, though the Spam Assassin dataset bears a closer resemblance to the CSDMC2010 dataset than to the Enron dataset. This is particularly evident in that the differences between the mean and standard deviations of the ham and spam classes are small e.g. the differences in standard deviation for Sadness and Conscientiousness being 0.

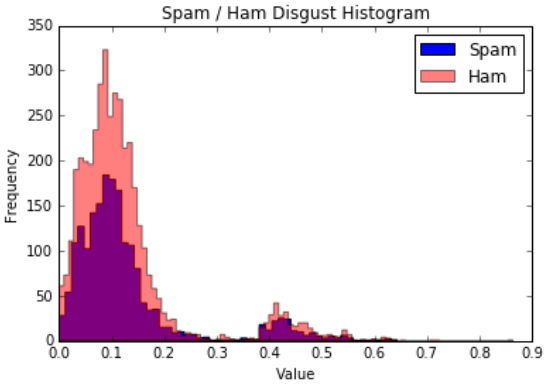
Spam Assassin Dataset														
		Emotional Tone					Language Tone			Social Tone				
		Anger	Disgust	Fear	Joy	Sadness	Analytical	Confident	Tentative	Openness	Conscientiousness	Extraversion	Agreeableness	Emotional Range
Ham	count	4153	4153	4153	4153	4153	4153	4153	4153	4153	4153	4153	4153	4153
	mean	0.241	0.127	0.190	0.410	0.331	0.421	0.069	0.404	0.437	0.384	0.424	0.330	0.580
	std	0.150	0.110	0.132	0.179	0.183	0.298	0.197	0.342	0.258	0.303	0.270	0.282	0.278
Spam	count	1898	1898	1898	1898	1898	1898	1898	1898	1898	1898	1898	1898	1898
	mean	0.245	0.128	0.187	0.409	0.319	0.417	0.080	0.384	0.425	0.394	0.434	0.331	0.578
	std	0.153	0.113	0.131	0.188	0.183	0.299	0.215	0.344	0.262	0.303	0.271	0.276	0.282
Differences	count	6051	6051	6051	6051	6051	6051	6051	6051	6051	6051	6051	6051	6051
	mean	0.004	0.002	0.003	0.000	0.012	0.003	0.011	0.020	0.012	0.010	0.010	0.000	0.002
	std	0.003	0.003	0.001	0.008	0.000	0.001	0.018	0.002	0.004	0.000	0.001	0.006	0.005

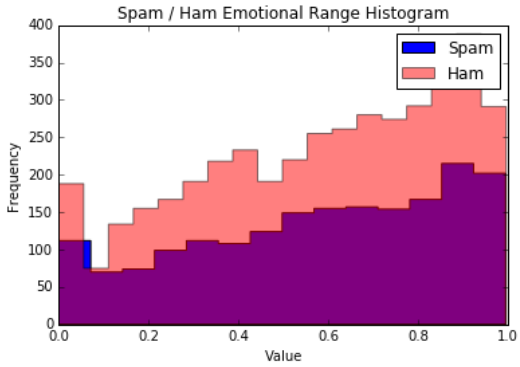
Table 5-4 Class Split of Mean and Standard deviations of tone features: Spam Assassin

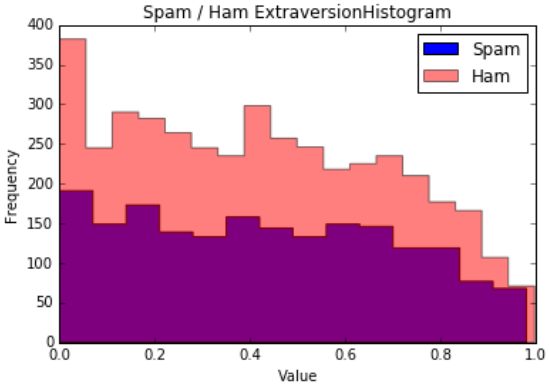
The similarity of the values is re-enforced when the ham and spam feature values are visualised.

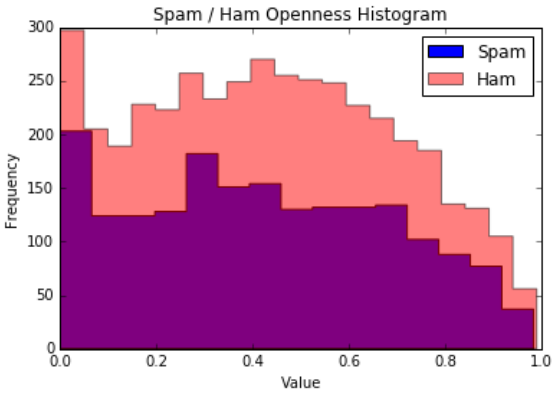


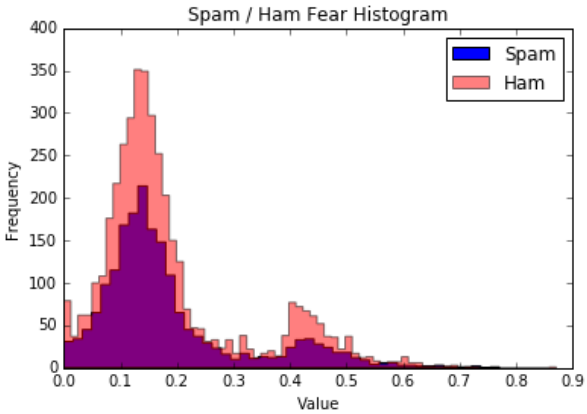

Spam / Ham Conscientiousness Histogram
Spam
Ham
Frequency
Value


Spam / Ham Disgust Histogram
Spam
Ham
Frequency
Value


Spam / Ham Emotional Range Histogram
Spam
Ham
Frequency
Value


Spam / Ham ExtraversionHistogram
Spam
Ham
Frequency
Value


Spam / Ham Openness Histogram
Spam
Ham
Frequency
Value


Spam / Ham Fear Histogram
Spam
Ham
Frequency
Value

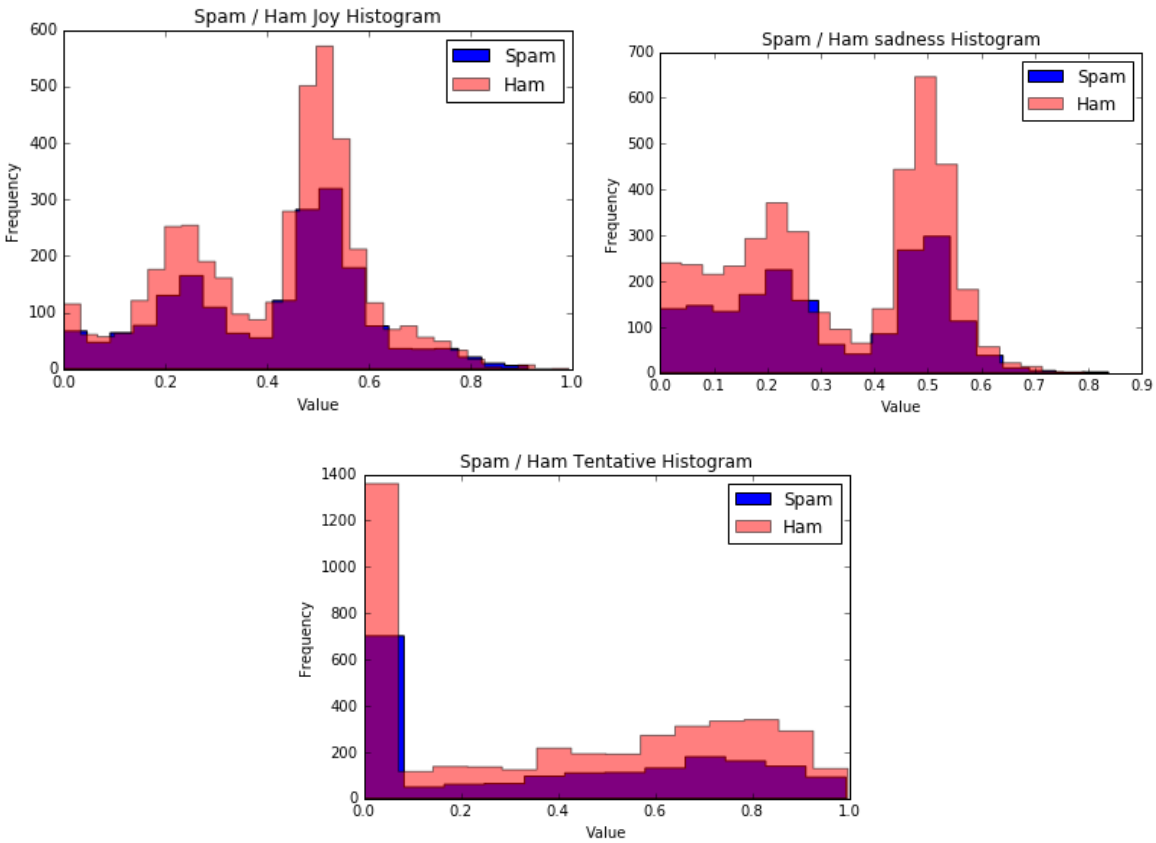


Figure 5-9 Spam Assassin Tone Features: Class-wise split of vales

This initial exploration simply looks at the distribution of the values for the different tone features and does not explore any statistical relationship between the tone features and the target variable. The following section will look at this in more detail.

5.4 Tone Feature Selection

The section above suggests an overall similarity in the distributions of the features for ham and spam emails, however further investigation is required to find out if there are certain features that are of more value than others in classifying an email as ham or spam. This will be investigated using two methods of feature selection common in spam filtering – Mutual Information and Chi-square.

These methods are used to find categorical features that are related to a categorical target. Since the tone feature data is continuous it must be discretized. In fact this has already been done to a degree in visualising the tone features as histograms above. The 'auto' setting for deciding the number of bins for each graph uses two methods for determining the number of bins, Sturges and Freedman Diaconis Estimator. The method which has the highest number of equal width bins is selected. This can be seen in the graphs below where the application has chosen different

bin ranges to suit the data. The bin ranges now need to be converted to a single value. A common method, which is the method followed in this experiment, is to take the mid-point of the bin and treat that as a category value.

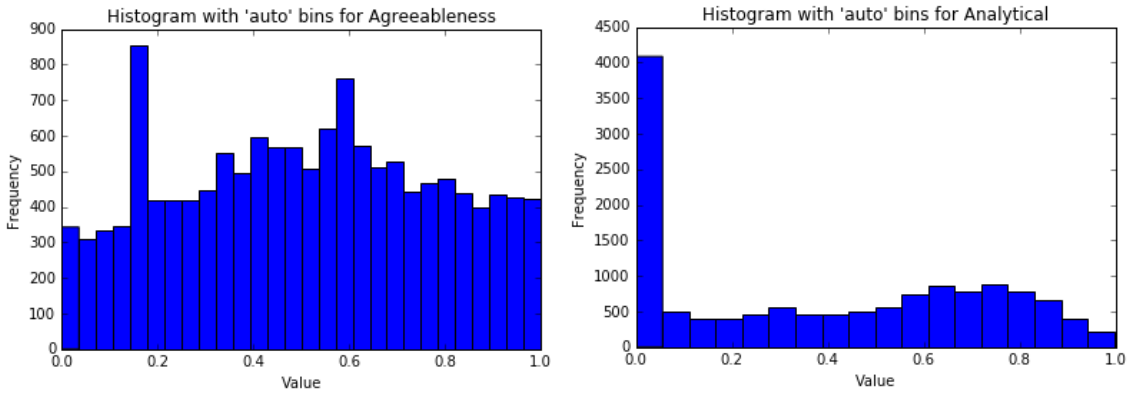


Figure 5-10 Sample of "auto" binning outputs for Enron tone features

5.4.1 Feature Selection: Investigation with Enron Data

The Chi-square and Mutual Information techniques used are relevant for classification problems and provide a score for each feature based on how that feature relates to the target variable. The features are ranked by that score to determine which features are best placed to improve the classification algorithm.

The below table shows the Chi-square and Mutual Information scores for the 13 tone features in the Enron dataset. Common across the two measures is that the features of 'Conscientiousness' and 'Extraversion' rank highly suggesting that they have a value in classifying emails. Whereas 'Anger, and 'Analytical' rank towards the bottom suggesting they have little value in classifying emails. This is broadly in line with the distributions of the feature values. Besides 'Joy', which broadly ranks in the middle, none of the other features rank consistently across the measures to merit further comment.

Feature	MI Score		Feature	Chi-square Score
Conscientiousness	**0.031**		Confident	92.079
Extraversion	**0.015**		Tentative	25.703
Sadness	0.014		**Extraversion**	**18.004**
Joy	**0.009**		**Conscientiousness**	**14.099**
Agreeableness	0.009		Openness	13.572
Confident	0.009		Agreeableness	7.140
Fear	0.007		Sadness	4.594
Emotional Range	0.006		**Joy**	**3.330**
Analytical	**0.005**		Disgust	2.378
Disgust	0.004		**Analytical**	**2.347**
Tentative	0.004		Emotional Range	0.803
Openness	0.004		**Anger**	**0.211**
Anger	**0.003**		Fear	0.018

Table 5-5 Feature Ranking by Mutual Information and Chi-square applied to the Enron Personality Insight and Tone Features

In this part of the experiment the classification value of the personality insight and tone features on their own is being investigated. For the tests using the Enron data the tests using the SVM classifier are run using the five and nine best ranked features under both measures. The results of those tests with respect to feature selection are applied to the remainder of the tests in this part of the experiment.

5.5 Spam Filtering with Personality Insight and Tone Features

In the following section the results of the second part of the experiment – spam filtering with the personality insight and tone features only – are presented. The purpose of this part of the experiment is to investigate firstly if these features can be used to classify spam, secondly if feature selection techniques can improve that classification and thirdly what the best parameter settings for the classifiers are. Overall ten different combinations of classifier, feature selection technique and parameter settings are trained and tested.

5.5.1 Experiment Results

The below table presents the average class accuracy results of each combination of classifier and techniques for all three datasets. In the below MI refers to Mutual Information, F to the number of features used by the classifier, K to the kernel type used by the SVM classifier, C to the c-value and Crit. to the criterion used by the Random Forests classifier to measure the quality of a split.

Classifier Details				Dataset		
No.	**Classifier**	**Feature Selection Technique**	**Parameter Settings**	**Enron**	**CSDMC 2010**	**Spam Assassin**
1	SVM	MI, F = 5	K = RBF, C = 10	59.11	-	-
2	SVM	MI, F = 9	K = RBF, C = 1	64.13	-	-
3	SVM	Chi2, F = 5	K = RBF, C = 1	57.05	-	-
4	SVM	Chi2, F = 9	K = RBF, C = 1	65.93	-	-
5	**SVM**	**F=13**	**K = RBF, C = 1**	**68.08**	**-**	**-**
6	SVM	Chi2, F = 9	K = RBF, C = 10	-	50.55	50.13
7	SVM	F=13	K = Sigmoid, C = 0.1	-	50.00	-
8	SVM	F=13	K = Linear, C=1	-	-	50.00
9	RF	Chi2, F = 9	Crit = Entropy	65.20	50.18	50.87
10	**RF**	**F=13**	**Crit = Entropy**	**69.76**	49.40	51.12

Table 5-6 Personality Insight and Tone Feature Classifier Accuracy Results for all datasets

The first obvious result of this experiment is that none of the results are close to the baseline results, all of which were above 96% accuracy. Only the tests with the Enron dataset show any indication that the personality insight and tone features can be used to classify emails as spam or ham. In these experiments average class accuracy results of 68% and 69.7% are achieved using SVM and RF classifiers respectively. However, the average class accuracy results for tests on both the CSDMC2010 and SpamAssassin datasets are all close to 50% accuracy, meaning that these models are mostly classifying all emails by the majority class – ham. Where SVM is used with all 13 features 100% of the emails in tests with both datasets are classified as ham.

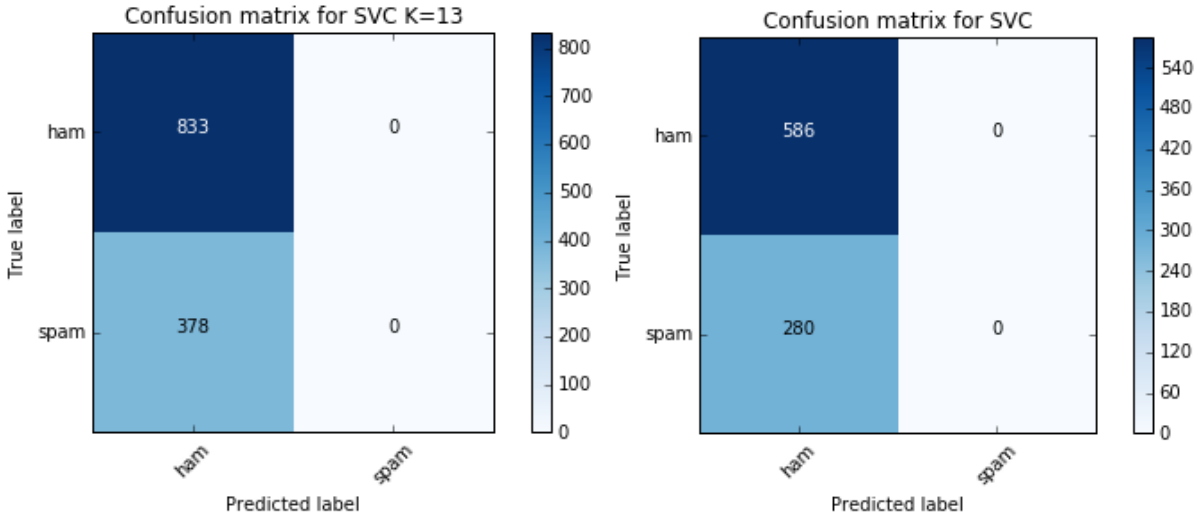


Figure 5-11 Example of Confusion Matrices from application of SVM to Spam Assassin and CSDMC Datasets

In all other tests the classifiers would classify a small number of spam emails correctly but would also misclassify a similar number of ham emails. It is the success or failure of classification on these few instances causes the minor variations, above or below 50%, in the accuracy results for these datasets.

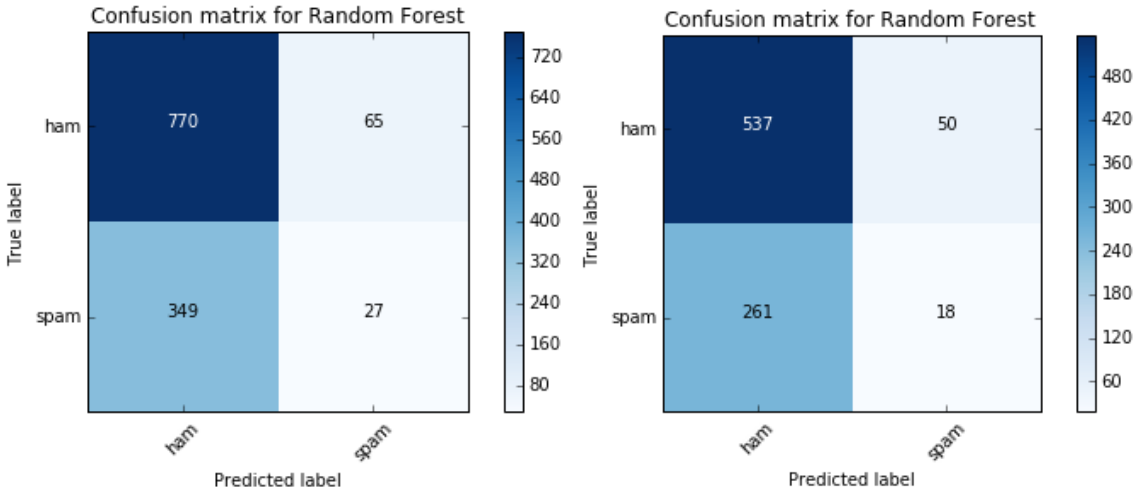


Figure 5-12 Example of Confusion Matrices from application of Random Forests to Spam Assassin and CSDMC Datasets

The second main result of this experiment is that feature selection does not improve any of the classifiers. The results of the tests with the Enron data showed that the best results were obtained when all features were included in the model. The tests on the Enron dataset using SVM and both feature selection techniques show that as more features are added to the model

the performance improves by ~10% overall. The Random Forests results with the Enron dataset again show that using all features is better. So as not to discount feature selection completely based on the accuracy results from these tests the best performing technique is applied to the remaining experiments with the other datasets, i.e. selecting the best nine features via Chi-square. The results in relation to feature selection with the CSDMC2010 and SpamAssassin datasets, however, are inconclusive. Where feature selection was used with SVM it marginally improved the model and the opposite occurred where feature selection was used with Random Forests. The results are deemed inconclusive because in both cases the mixed results are in the context of poor overall classification results achieved by these models. This result is in line with the conclusions made by Drucker, Wu, & Vapnik in their 1999 paper in that finding the best features can take a long time and also for decision tree and SVM classifiers too many features does not degrade performance.

Overall the results of these experiments are inconclusive with respect to whether these new features can be used to classify emails as spam or ham. This is due to the fact that the results of the experiments using the different datasets differ. The results from Enron dataset suggest that these features can be useful in classifying emails as ham or spam, whereas the results using the CSDMC2010 and Spam Assassin datasets show that these features have not been useful in filtering spam emails.

5.6 Combined Classifier

This section will discuss the third part of the overall experiment where the personality insight and tone features are combined with the normalised term-frequency features from the baseline classifier and this combined set of features is used for spam filtering. The purpose of this experiment is to establish if these features can improve the classification performance of the classifier above the baseline classifier performance.

5.6.1 Combined Classifier Results

The results of this experiment across the three datasets are shown below alongside the equivalent results from the baseline classifier experiment. On review of the accuracy results below from the baseline and combined classifiers it can be seen that the results across all of the datasets are quite close together, the range from best to worse accuracy score is less than 1%.

Classifier Details			Dataset		
Type	**Classifier**	**Parameter Settings**	**Enron**	**CSDMC2010**	**SpamAssassin**
Combined	SVM	K = Linear, C = 1	97.28	97.00	97.02
Baseline	SVM	K = Linear, C = 1	97.51	96.75	96.61
		Difference	**-0.24**	**0.25**	**0.41**

Table 5-7 Combined Classifier Results

Furthermore the differences between the combined and baseline classifiers for each dataset are also small. Two of the three datasets showed a slight improvement in classification accuracy, with the third showing a slight decrease. The largest increase is seen on the SpamAssassin dataset at 0.41%. This would suggest that the inclusion of the personality insight and tone features does not make any real difference to the performance of the classifier.

5.7 Conclusion

In this chapter the overall experiment undertaken for this project and the related results have been presented. In the first part of this overall experiment a baseline classifier was run on the three datasets to set a level of accuracy performance against which the performance of the experiments using personality insight and tone features can be judged. This part of the experiment identified the model best performing as the standard SVM algorithm.

In preparing for the experiments where personality insight and tone features would be used to classify emails data exploration was performed on these features. This exploration showed that there was a large degree of similarity between the feature values for spam and ham emails. In the second part of the experiments the results of both the SVM and Random Forests models showed that on two of the three datasets these features were not useful in classifying emails, in these cases the models classified email based on the majority class - ham. Only in tests with the Enron dataset could these features be used to classify emails and the highest accuracy performance achieved was 69.76% using Random Forests.

Feature selection using the statistical methods of Chi-square and mutual information was investigated on the Enron dataset. The results on the Enron dataset showed that the models generally performed better where all features were included and the remaining experiments supported this.

In the third part of the overall experiment the personality insight and tone features were combined with the normalised term-frequency features and an SVM classifier was run on this combined feature-set. The accuracy results of this experiment were very similar to the results

of the baseline classifier indicating that the inclusion of the new features had no real impact to the accuracy of the model.

The following chapter will discuss the conclusions from this project and include some hypotheses around why personality insight and tone features were able to classify emails with the Enron dataset and not the other datasets.

6 Conclusion

6.1 Introduction

This chapter will conclude the thesis by reviewing the research questions and objectives in the context of the findings from the research conducted. Contributions to the body of knowledge from this work and limitations thereof will be discussed and potential areas of future research will also be put forward.

6.2 Problem Definition and Research Overview

The purpose of this work was to investigate if personality insight and tone features, which were created from the body content of emails through the use of the IBM Watson Tone Analyzer API, could be used to improve a spam classifier. As part of this investigation research into the general spam domain was conducted as well as research into literature on the state of the art in spam filtering techniques and classification algorithms. Based on this a set of experiments were designed to investigate the stated purpose of the work.

The following objectives have been achieved as part of this research:

1. Gain in-depth knowledge about spam, machine learning techniques for spam filtering, natural language processing (NLP), sentiment analysis, and application programming interfaces (APIs).
2. Gain practical experience in designing and deploying machine learning algorithms as a classification experiment.
3. Design, test and implement models for spam filtering that use features derived from email body text.
4. Design an experiment to evaluate these models and prove one of the hypotheses set in Chapter 1.
5. Discuss the relevance of the result, make recommendations for future areas of work and add positively to the body of knowledge in the domain areas outlined above.

6.3 Experimentation, Evaluation and Results

This thesis aimed to create new features from the body content of emails and use those new features to improve the accuracy performance of a spam filter. After setting a baseline level of accuracy to judge the performance of a classifier using these features against two further

experiments were carried out. Firstly where only the new features were used to try and filter spam and secondly where the new features were combined with normalised term-frequency features. In the first of these experiments with the new features only the experiments with the Enron dataset returned results that suggested that these features could be used to filter spam. Results from experiments on the other two datasets, CSDMC2010 and SpamAssassin, showed that the classifiers based their decisions on the majority class.

There are some potential causes for the differences seen above. Firstly the Enron dataset is larger than the other two datasets, by roughly twice and three times for the Spam Assassin and CSDMC2010 datasets respectively. Accepting that generally better quality training data is preferred to more training data it could still be the case that the Enron classifiers have more data to learn from during training and so were better able to classify unseen data as a result of that. Secondly the timeframe the datasets were gathered in is different. The Spam Assassin and CSDMC2010 datasets are gathered in 2002 and 2010 respectively however the Enron Dataset has its ham gathered from 1999 to 2002 while its spam is gathered from 2003 to 2005. It could be that the language used (and therefore the feature values returned) differs enough for it to be easier to separate ham and spam emails. This is in line with the fact that is a clear difference in the distribution of the ham and spam personality insight and tone feature values for the Enron dataset by comparison to the Spam Assassin and CSDMC datasets. For the latter two datasets the mean and standard deviation values are very similar for the ham and spam classes.

In the third part of the experiment– the combined experiment – the results were similar to the baseline classifier results. One classifier performed worse and two performed better but all the improvements in accuracy were by less than 0.5%. Based on the results of these experiments it is concluded that these features were not able to improve a spam classifier above the baseline performance level and this the null hypotheses is supported.

6.4 Contribution to the body of Knowledge

Worthwhile contributions from this research are:

- This researched showed how new features for spam filtering can be created through the use of a 3rd party API tool – the IBM Watson Tone Analyzer
- The value of personality insight and tone features for spam filtering was investigated

- Feature selection was not successful in improving the models which used these new features. Results improved when all 13 returned features are used in classification, although this improvement was limited to the Enron dataset.
- The research showed inconclusive results on whether these features, on their own, were useful in detecting spam. Only one of the three datasets returned results that suggested they could be useful in spam filtering.
- When these new features are included in a combined model they did not improve that model above the baseline performance set by a significant amount, thus supporting the null hypotheses.

6.5 Limitations

This investigation was limited to the IBM Watson Tone Analyzer, using another type of sentiment enhanced classifier could have given further context as to whether these type of features are useful in general rather than whether the features returned from the *IBM Watson Tone Analyzer* are useful for spam filtering.

The API used for this research has constraints with respect to the number of requests made of it within a specific time-frame and is also constrained in relation to the size of an individual request. These constraints would need to be resolved for such an API to be useful in an online setting due to the volume of emails that a spam filter would classify on an ongoing basis.

6.6 Future Works

A simple discretization method was used to transform the continuous feature variables returned from the API into categorical features. It would be worthwhile to investigate if using a more complex discretization method that is reflective of the classes would improve the classification accuracy. A more complex classifier might be better suited to finding patterns and relationships between these features and the classes. Artificial Neural Networks have been used in spam filtering successfully but have not been tested as part of this work so could be a worthwhile avenue of investigation.

This research was not carried out in an online setting so it did not cater for any of the more complex issues faced by a spam filter in the real world, e.g. it would not account for concept drift in emails. It is possible that sentiment, personality insight and language tone features would be more beneficial in a dynamic changing environment over a static one. Particularly if

it is logical that most spam email content is written by a relatively small group of people[14] and that writers can have “textual fingerprints” (Bhowmick & Hazarika, 2016).

[14] (Harris, 2003)

7 Bibliography

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., Paliouras, G., & Spyropoulos, C. D. (2000). An evaluation of naive bayesian anti-spam filtering. *arXiv preprint cs/0006013.*

Bhowmick, A., & Hazarika, S. M. (2016). Machine Learning for E-mail Spam Filtering: Review, Techniques and Trends. *arXiv preprint arXiv*, 1606.01042.

Breiman, L. (2001). Random forests. *Machine learning, 45*(1), 5-32.

Chowdhury, G. G. (2003). Natural language processing. *Annual review of information science and technology, 37*(1), 51-89.

Clark, J., Koprinska, I., & Poon, J. (2003). A neural network based approach to automated email classification. *Proceedings IEEE/WIC International Conference on Web Intelligence* (pp. 702-705). IEEE.

DeBarr, D., & Wechsler, H. (2009). Spam detection using clustering, random forests, and active learning. *Sixth Conference on Email and Anti-Spam. Mountain View, California.*

Drucker, H., Wu, D., & Vapnik, V. N. (1999). Support vector machines for spam categorization. *IEEE Transactions on Neural networks, 10*(5), 1048-1054.

Du, S., Liu, C., & Lifeng, X. (2015). A selective multiclass support vector machine ensemble classifier for engineering surface classification using high definition metrology. *Journal of Manufacturing Science and Engineering, 137*(1), 011003.

Eberhardt, J. J. (2015). Bayesian spam detection. *Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal, 2*(1), 2.

Ezpeleta, E., Zurutuza, U., & Hidalgo, J. M. (2016). Does sentiment analysis help in bayesian spam filtering? *International Conference on Hybrid Artificial Intelligence Systems* (pp. (pp. 79-90).). Springer International Publishing.

Fan, R. E., Chang, K. W., Hsieh, C. J., Wang, X. R., & Lin, C. J. (2008, August). LIBLINEAR: A library for large linear classification. *Journal of machine learning research, 9*, 1871-1874.

Fawcett, T. (2003). In vivo spam filtering: a challenge problem for KDD. *ACM SIGKDD Explorations Newsletter, 5*(2), 140-148.

Ferris, D., Jennings, R., & Williams. (2005). *The Global Economic Impact of Spam.* Ferris Research.

Fletcher, D. (2009, November 2). *A brief history of Spam*. Retrieved from Time Magasine: http://content.time.com/time/business/article/0,8599,1933796,00.html

Gansterer, W. N., Janecek, A. G., & Neumayer, R. (2008). Spam filtering based on latent semantic indexing. *Survey of Text Mining II*, 165-183.

Garcia, S., Luengo, J., Sáez, J. A., Lopez, V., & Herrera, F. (2013). A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. *IEEE Transactions on Knowledge and Data Engineering, 25*(4), 734-750.

Gee, K. R. (2003). Using latent semantic indexing to filter spam. *Proceedings of the 2003 ACM symposium on Applied computing* (pp. 460-464). ACM.

Goodman, J., Cormack, G. V., & Heckerman, D. (2007). Spam and the ongoing battle for the inbox. *Communications of the ACM, 50*(2), 24-33.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research, 3*(March), 1157-1182.

Guzella, T. S., & Caminhas, W. M. (2009). A review of machine learning approaches to spam filtering. *Expert Systems with Applications, 36*(7), 10206-10222.

Harris, P. (2003, June 1). *Sick of spam? Then blame Alan Ralsky. He emails a billion of them a day*. Retrieved April 28, 2017, from www.theguardian.com: https://www.theguardian.com/technology/2003/jun/01/spam.theobserver

Haykin, S. (1999). *Neural Networks A Comprehensive Foundation* (2nd ed.). Pearson.

Hsieh, C. J., Chang, K. W., Lin, C. J., Keerthi, S. S., & Sundararajan, S. (2008). A dual coordinate descent method for large-scale linear SVM. *Proceedings of the 25th international conference on Machine learning* (pp. 408-415). ACM.

IBM Watson Developer Cloud. (2017, April 5). *Overview of the Tone Analyzer service*. Retrieved April 5, 2017, from www.ibm.com: https://www.ibm.com/watson/developercloud/doc/tone-analyzer/index.html

IBM Watson Developer Cloud. (2017, April 5). *Understand your tone score*. Retrieved April 5, 2017, from IBM Watson Developer Cloud Website: https://www.ibm.com/watson/developercloud/doc/tone-analyzer/understand-tone.html

IBM Watson Developer Cloud. (2017, April 25). *Understand your tone score.* Retrieved from IBM Watson Developer Cloud: https://www.ibm.com/watson/developercloud/doc/tone-analyzer/understand-tone.html

JSON. (n.d.). *Introducing JSON*. Retrieved from json.org: http://www.json.org/

Kanaris, I., Kanaris, K., Houvardas, I., & Stamatatos, E. (2007). Words versus character n-grams for anti-spam filtering. *International Journal on Artificial Intelligence Tools, 16*(06), 1047-1067.

Kaspersky Lab. (2017, February 20). *Spam and phishing in 2016*. Retrieved from securelist.com: https://securelist.com/analysis/kaspersky-security-bulletin/77483/kaspersky-security-bulletin-spam-and-phishing-in-2016/

Kelleher, J., MacNamee, B., & D'arcy, A. (2015). *Fundamentals of Machine Learning for Predictive Data Analytics.* MIT Press.

Koprinska, I., Poon, J., Clark, J., & Chan, J. (2007). Learning to classify e-mail. *Information Sciences, 177*(10), 2167-2187.

Kwak, N., & Choi, C. H. (2002). Input feature selection for classification problems. *IEEE Transactions on Neural Networks, 13*(1), 143-159.

Lau, R. Y., Liao, S. Y., Kwok, R. C., Xu, K., Xia, Y., & Li, Y. (2011). Text mining and probabilistic language modeling for online review spam detecting. *ACM Transactions on Management Information Systems, 2*(4), 1-30.

Lee, S. M., Kim, D. S., Kim, J. H., & Park, J. S. (2010). Spam detection using feature selection and parameters optimization. *2010 International Conference on Complex Intelligent and Software Intensive Systems (CISIS)* (pp. 883-888). IEEE.

Levchenko, K., Pitsillidis, A., Chachra, N., Enright, B., Félegyházi, M., Grier, C., . . . McCoy, D. (2011, May). Click trajectories: End-to-end analysis of the spam value chain. *IEEE Symposium on Security and Privacy (SP)* (pp. 431-446). IEEE.

Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. *R news, 2*(3), pp. 18-22.

Liddy, E. D. (2001). Natural language processing. In *Encyclopedia of Library and Information Science* (2nd Ed ed.). Marcel Decker, Inc.

Mason, J. (2006). *REVISION HISTORY OF THIS CORPUS*. Retrieved April 28, 2017, from spamassassin.apache.org: http://spamassassin.apache.org/old/publiccorpus/readme.html

Metsis, V., Androutsopoulos, I., & Paliouras, G. (2006, July). Spam filtering with naive bayes-which naive bayes?. *CEAS, 17*, 28-69.

Microsoft. (2011). *Microsoft Security Intelligence Report Special Edition Battling the Rustock Threat.* Microsoft.

Mitchell, T. (1997). *Machine Learning.* McGraw-Hill.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval, 21*(1-2), 1-135.

Piatetsky, G. (2016, Jun). *R, Python Duel As Top Analytics, Data Science software – KDnuggets 2016 Software Poll Results*. Retrieved from kdnuggets.com: http://www.kdnuggets.com/2016/06/r-python-top-analytics-data-mining-data-science-software.html

Puget, J. F. (2016, December 19). *The Most Popular Language For Machine Learning Is ...* Retrieved from ibm.com: https://www.ibm.com/developerworks/community/blogs/jfp/entry/What_Language_Is_Best_For_Machine_Learning_And_Data_Science?lang=en

Rao, J. M., & Reiley, D. H. (2012). The economics of spam. *The Journal of Economic Perspectives, 26*(3), 87-110.

Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation, 60*(5), 503-520.

Russel, S., & Norvig, P. (2009). *Artificial Intelligence A Modern Approach* (3rd ed.). Pearson.

Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998, July). A Bayesian approach to filtering junk e-mail. . *Learning for Text Categorization : Papers from the 1998 workshop , 62*, 98-105.

Schneider, K. M. (2004). On word frequency information and negative evidence in Naive Bayes text classification. *Advances in Natural Language Processing* , 474-485.

Schölkopf, B., Smola, A. J., Williamson, R. C., & Bartlett, P. L. (2000). New support vector algorithms. *Neural computation, 12*(5), 1207-1245.

Stone-Gross, B., Holz, T., Stringhini, G., & Vigna, G. (2011). The Underground Economy of Spam: A Botmaster's Perspective of Coordinating Large-Scale Spam Campaigns. *LEET, 11*, 4-4.

Subramaniam, T., Jalab, H. A., & Taqa, A. Y. (2010). Overview of textual anti-spam filtering techniques. *International Journal of Physical Sciences, 5*(12), 1869-1882.

Symantec. (2010). *Global Internet Security Threat Report Trends for 2009.* Symantec.

Symantec. (2015). *Internet Security Threat Report.* Symantec.

Tausczik, Y. R. (2010). The Psychological Meaning of Words: LIWC adn Computerized Text Analysis Methods. *Journal of Language and Social Psychology, 29*(1), 24-54.

Thonnard, O., & Dacier, M. (2011). A strategic analysis of spam botnets operations. *In Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference* (pp. 162-171). ACM.

Tzortzis, G., & Likas, A. (2007, October). Deep belief networks for spam filtering. *Tools with Artificial Intelligence, 2*, 306-309.

Wang, B., Jones, G. J., & Pan, W. (2006). Using online linear classifiers to filter spam emails. *Pattern analysis and applications, 9*(4), 339-351.

Wu, C.-H. (2009). Behavior based spam detection using a hybrid method of rule-based techniques and neural networks. *Expert Systems with Applications, 36*(3), 4321-4330.

Yerazunis, W. S., Chhabra, S., Siefkes, C., Assis, F., & Gunopulos, D. (2005). A unified model of spam filtration. *Proceedings of the MIT Spam Conference.* Cambridge, MA, USA.

Yu, B., & Xu, Z. B. (2008). A comparative study for content-based dynamic spam classification using four machine learning algorithms. *Knowledge-Based Systems, 21*(4), 355-362.

Zhou, Z. (2012). *Ensemble Methods Foundations and Algorithms.* CRC Press .