Comparative Study of Features Space Reduction Techniques for Spam Detection

By

Nouman Azam

1242 (MS-5)

Supervised by

Dr. Amir Hanif Dar

Thesis committee

Brig. Dr Muhammad Younas Javed

Dr. Azad A Saddiqui

Dr. Shaleeza Sohail

Submitted in partial fulfillment of the requirements for the Degree of Masters in Computer Science

Department of Computer Engineering
College of Electrical and Mechanical Engineering
National University of Sciences and Technology

Abstract

The increased usage of the internet in the recent past has turned email as the most widely used medium for communication. Its wide usage soon attracted a lot of companies to advertise their products on the medium thus starting non ending streams of spams. The growing volume of spam emails has increased the demand for accurate and efficient spam solutions. Many spam solutions have been proposed in the recent past[1]. The one which we addresses in this research has achieved wide spread popularity[2]. It treats spam detection as a simple two class document classification problem, the solution of which will consist of classification algorithm coupled with dimensionality reduction method as document classification tasks are driven by high dimensionality. Classification in reduced dimensionality will help us improving the performance in terms of accuracy and will have lesser computational time and storage requirements.

Earlier work in the feature reduction on the domain of document classification concentrated on multiple class problems [8] [9]. The contribution of this research is the comparison of different dimensionality reduction methods on a two class problem. Performances of the techniques were measured as a function of features set size. There were two sub objectives that were also addressed. Firstly, to find the best size of the features set for every feature reduction technique that will do a good job of classification. Secondly to compare different feature reduction techniques performances under different classifiers so that advances towards finding the best couple of classification algorithm and dimensionality reduction technique can be made. Eight different feature reduction techniques were compared and analyzed with three classifiers i.e. Nearest Neighbor, Weighted Nearest Neighbor and Naïve Bayesian. The techniques showed quite promising results in certain cases, even at as low as features set of size 10. Latent Semantic Indexing was found to have best performance at lower features set while Mutual Information and CHI Square techniques showed acceptable performance at lower features sets and have best performance at higher features set sizes.

[1] http://www.templetons.com/brad/spam/spamsol.html
[2] http://en.wikipedia.org/wiki/Bayesian_spam_filtering

Acknowledgements

One of the great pleasures of writing a thesis is acknowledging the efforts of many people whose names may not appear on the cover, but whose hard work, cooperation, friendship, and understanding were crucial to the production of the thesis.

The report that you are holding is the result of many people's dedication. I am gratefully thankful to my supervisor Dr. Amir Hanif Dar for encouraging my work at each step of my thesis and guiding me towards the right and realistic goals. Without his suggestions and guidance I would not been able to complete my thesis in efficient and timely manner.

Special thanks to Dr. Yerazunis (chair of Spam Conference MIT,USA 2007) for giving me information about the current corpora's and to Mr Georgios Paliouras (National Centre for Scientific Research Demokritos Paraskevi, Athens, Greece) for giving out definitions of some useful terms.

Thanks to my friend Samiullah Marwat for helping me in the preprocessing of the Ling Spam corpura before being used for experimentation purposes. Thanks to Mr. Abdul Baqi for providing me with softwares that I needed and to Mr. Mehmood Alam khan for providing me with some useful books regarding my thesis topic.

This thesis would not have been possible without the love and support of my parents. They were my first teachers and inspired in me a love of learning and a natural curiosity. Last and most importantly, I thank my brother Dr. Shahid Azam. He provides me with the faith and confidence to endure and enjoy it all and to guide me towards the conducting of research and its methodology.

Table of Contents

Table of Tables

Table of Figures

List of Abbreviations

NN	Nearest Neighbor
MI	Mutual information
OR	Odds Ratio
DF	Document Frequency
TF	Term Frequency
TFIDF	Term Frequency inverse of Document Frequency
LSI	Latent Semantic Indexing
LDA	Linear Discriminant Analysis
SP	Spam Precision
SR	Spam Recall

CHAPTER – 1

INTRODUCTION

1.1. Introduction

Spam is an emergent problem of the internet users. The recent increases in the spam rate had caused a great concern among the internet community. Many solutions to deal with the problem had been suggested on the technical and non technical sides. This chapter introduces the problems that the internet users are currently facing due to spam. Study of the currently deployed or suggested solutions is also presented with their corresponding advantages and disadvantages. Automatic spam filtering solution is also introduced and its superiority over the existing solutions is shown. Finally, brief introduction of document classification problem in the context of spam email filtering is also provided with the main challenges that the problem faces.

1.2. Aims

We address the problem of spam as a simple two class document classification problem where the main goal is to filter out or separate spam from non spam. Since document classification tasks are driven by high dimensionality so selecting most discriminating features for improving accuracy is one of the main objectives and my thesis work concentrates on this task. Classification in the reduced dimensions will not only save us time but also will have lesser memory requirements

The primary aim of this thesis is to concentrate on different dimensionality reduction techniques and to compare their performances on the domain of spam email detection. A number of pre classified emails were processed with the techniques to see which one is most successful and under what size of features set. Secondary aim of the thesis is to work towards finding the best couple of dimensionality reduction technique and classification algorithm. Though a hard job as hundreds and thousands of such couples exits but this work can be considered as a step towards that goal.

Success was mainly measured as a function of accuracy. Accuracy of a technique is its ability to correctly identify an unknown instance of email. Success was also investigated as a function of feature set size. Best accuracies and corresponding feature set sizes was determined for this purpose. Other measures that were considered are Spam Recall and Spam Precision.

For testing purpose we tested the dimensionality reduction techniques using the K-Nearest Neighbor algorithm for varying values of K. The data after the preprocessing was represented using the TFIDF with lengths normalized and TF not normalized.

1.3. The Problem of Spam

The recent explosion in research work and human knowledge is made possible through internet. The increased internet usage has turned email to be the most widely used source for communication worldwide. Its popularity is due to its simplicity, fastness, reliability and easy access. With a single click you can communicate your message to any person any where in the world at any time. Due to advantages mentioned above, especially the cost factor, many people use it for commercial advertisement purposes causing unwanted emails at user inboxes.

An email that a user does not want to have in his inbox is known as Spam. The recent increase in the spam rate has converted it as an emergent problem of the internet users. In 2002, statistic revealed that 40% of all incoming emails were spam[1]. In 2003, it rose to 50%[2] and it increased dramatically to 96% according to BBC News in 2006[3.] Table 1-1 will provide an idea about the problems that we are facing due to Spam.

Spam emails come from variety of organizations and people with variety of motives. Most of them can be very annoying. Imagine you are doing a serous work at office and you receive an email of porn while you are waiting for an email of your Boss. Table 1-2 summarizes the percentage of spam with the categories.

There are so many problems that have arisen from Spam. Firstly, it wastes the network resources of organizations as a lot bandwidth is wasted in downloading spam emails from the inbox. Most of the organizations pay for the internet and network resources, so it costs them significant amount. Secondly, spam emails can cause serious problems for Personal

[1] http://zdnet.com.com/2100-1106-955842.html
[2] http://zdnet.com.com/2100-1105_2-1019528.html
[3] http://news.bbc.co.uk/2/hi/technology/5219554.stm

Computer users in the absence of antivirus solutions installed. Thirdly, its wastage of time for employees, resulting in lesser productivity and thus effecting the over all performance

Table 1-1: Statistics of Spam

Daily Spam emails sent	12.4 billion
Daily Spam received per person	6
Annual Spam received per person	2,200
Spam cost to all non-corporation Internet users	$255 million
Spam cost to all U.S. Corporations in 2002	$8.9 billion
Email address changes due to Spam	16%
Annual Spam in 1,000 employee company	2.1 million
Users who reply to Spam email	28%
Users who purchased from Spam email	8%
Corporate email that is considered Spam	15-20%

of employees and companies and finally pornographic spams can cause concerns for parents if there children have access to the email.

Table 1-2: Categories of Spam

Products	25%
Financial	20%
Adult	19%
Scams	9%
Health	7%
Internet	7%
Leisure	6%
Spiritual	4%
Other	3%

1.4. The Spam Solutions

Many people have suggested different kinds of solutions to the spam problem. Some of them have been implemented with quite success while others which are mostly non technological solutions provide good attractive ideas with lots of hurdles to implement. In the following subsections a brief overview of these solutions has been discussed.

1.4.1. Non Technological solutions

In order to deal with the problem of spam some solutions were proposed based on the reaction of the receipts. Here we will mention few of them. The basic nature of these solutions is that they do not use any technological tools to address the problem rather they demand's the users and companies to take actions which will terrify people from sending spam. Another important feature of these solutions is that they are most proactive in nature. They can achieve high popularity in the organizations whose most part of available bandwidth is wasted on downloading spam emails. If proper awareness and devotion is created on the side of email users then these suggested solutions can have very good results.

1.4.1.1. Recipient Revolt

This solution suggests that on reception of any spam the user will react with anger in emails and in physical world. This solution helped significantly to scare more legitimate companies to keep themselves away from using junk E-mail and forced the ISPs to change policies. Some of the advantages from this solution are

- Forcing ISPs to change policies.
- Legit companies will be afraid to spam resulting in removal of email ids from their contacts.
- If it gains momentum then it will be having a nice positive feedback. The fewer spams the more effort can be spent on punishing them.

Some of the disadvantage of the solution

- Burden on ISPs for handling valid and invalid complaints.
- Authentication of complaints so that complaints are checked that they are against the right person.

- As spammers hide their identities, it will cause some people to block all mails from unknown persons. The result will be hurdles and limited range of communication.

1.4.1.2. Customer Revolt

Most of the spams contain advertisements of different sorts from companies. To deal with it this solution suggests that companies to which the users submit their data should be forced to disclose what they will do with that data and should stick to what ever they claim. There should be proper publishing of policies on the web pages, mentioning the purpose of data gathering. The disadvantages of this solution are

- There may be false complaints
- Burden of separating valid from invalid complaints.

1.4.1.3. Vigilante Attack

This solution suggests that spam addresses should be deal with anger and should be treated with mail bombs and denial of service attacks. Though it will make spammers to think before sending spam but sometimes an innocent might be a victim claiming that he is spammer. Some of the disadvantages of this solution are.

- Identification of spammer is very important for this kind of solution which is hard job.
- The results of this solution might be nasty in some cases and unethical mostly.

1.4.1.4. Hide your address

This solution includes using two emails addresses. One email address is used to receive all of the emails. The user then scans the emails and those found valid are forwarded to the second email address. The second email address is disclosed only to known persons and is never publicized on the internet. It suffers from the disadvantage following disadvantages

- Hard job of maintaining couple of email addresses.
- Infact no significant work is done regarding stopping of spams.
- Telling all of your contacts not to give your email address to any one and not to publicize it.

1.4.1.5. Contract-Law and limiting trial accounts

This solution requires an agreement between the user and the organization which provide the email facility. The user should sign a proper agreement before get the registration. Sufficient information should be gathered regarding the user to know his identity. The account should be on trail basis. After passing the trail successfully i.e. without being reported to have send spam, his account will get registered fully. If found violating the laws at any stage, his account will be abundant and should be punished. While this solution looks quite attractive but the big hurdle in its implementation is the disclosure of people's identity without their will to the organizations which might not be acceptable to many users.

1.4.2. Technical Solutions

The technical solutions are mostly reactive in nature i.e. once the spam is present at the user account then techniques are used to eliminate the spam. These solutions do not force spammers not to spam rather they work towards making the job of spammers hard. As more and more, the internet community learns about the problems of spams, the more proactive technical solutions we can expect. At the present moment, researchers have not concentrated on the proactive solutions greatly. In the preceding subsections a brief over view of technical solutions are presented.

1.4.2.1. Domain filters

Mailers programs are configured so that they only accept mails from specific domains. Emails whose domains are not mentioned will not be received. This way a lot of spam is blocked. The major disadvantages are

- Spammers will start using the valid domains.
- Communication range is narrowed down.

1.4.2.2. Blacklisting

It filters out unknown addresses and maintains databases of known abusers thus eliminating mails from them. Servers are placed in distributed manner which constantly monitors the communication of users and try to figure out spammers and their sites. Though it can be help full in some cases but again innocent users might be caught as spammers. Some of the disadvantages are

- Overhead of maintaining the database about the spammers
- Constant updation of the databases and retrieval of information from the distributed database about the spammers.
- It's hard to associate an email user with an email id. A user changing his email id will not be recognized thus outdating the database.

1.4.2.3. White list Filters

Mailer programs are configured to learn all contacts of a user and allow mails from those contacts only. Mails from strangers are put into other folders thus eliminating the chances of spam to be present at user inbox folder. Some of the advantages of this solution are
Almost no spam at user inbox since one is receiving mails at inbox from known contacts only. It can be used in combination with other tools (automatic filtering, stamps etc.). Disadvantages of the solution are

- Configuration of the mailer programs to learn about contacts of the users.
- If contacts email id changes, mailer program will not know about that, thus will eliminate that contacts mails from the user inbox.
- New parties mail might be delayed as they are not directly visible to the user because of not being present at the inbox.
- Overall it will suffer from the limited range of communication and hurdles in communication.

1.4.2.4. Rules based

Spam emails are examined by an expert and then efforts are made to find word or phrasal relationships between email instances and it corresponding class. The relationships thus define are called rules. Many rules are combined in this way to make up the spam detecting solution. Certain weights will also be assigned to rules based on their utility towards the class definition. An unknown instance will be thus classified based on the absence or presence of certain predefined rules along with their weights in the email.

The disadvantage of this solution is the requirement of human expert. Furthermore rules might be outdated due to the spammer's knowledge about the solution, thus changing the nature of the spams, which will lead to different relationship between textual email contents and its corresponding class. In such a challenging environment the needs of human expert will always be required to constantly update the system so that to cater for new kinds of spam.

1.5. Draw Backs in Solutions and an Alternative

All the solutions explained above have generally three major draw backs.

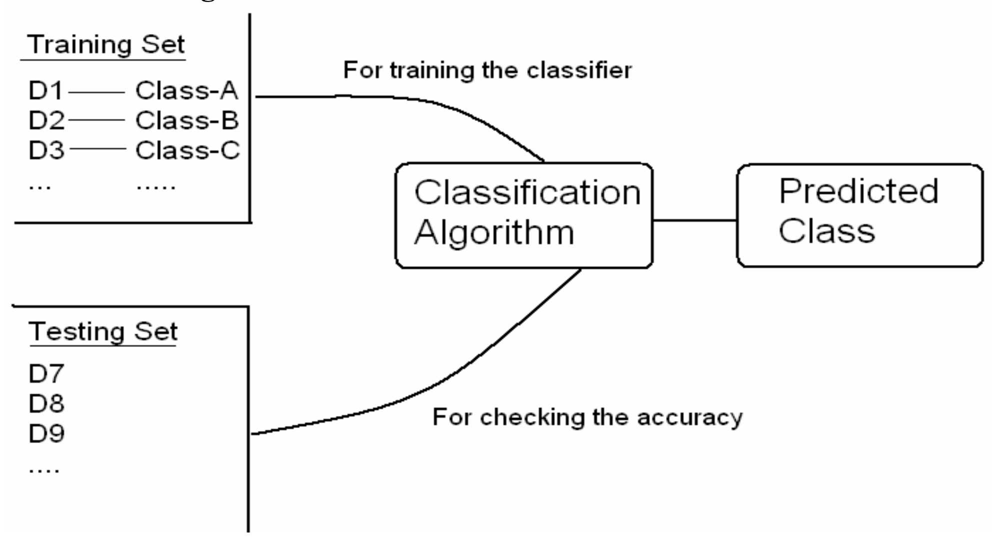
- Limited range of communication.
- Implementation hazards on users and companies sides.
- Expensive human resources requirements.

There exist an alternative to these solutions known as automated spam filtering that will minimize the three drawbacks. The solution uses machine learning algorithms to learn from the previous data and then given an unknown instance it tries to predict its class from previously learned patterns. The benefit of this solution is that it will update its self and will learn automatically about new kinds of spam with minimum user input. The solution will treat the problem of spam detection as an instance of document classification problem. In the preceding section a brief overview of the solutions is given.

1.6. Spam as a Document Classification Problem

Automated spam filtering can be considered as a simple instance of document classification. In document classification problems we have two sets of documents. The first document set has a predefined class and is known as the training set of documents. This document set is used by the classifier to learn patterns in the data. The second documents set do not have the class labels with it and is used for the testing purpose. These documents set constitute all examples from the real world which will be given as input to the classification algorithm to classify later on. The problem of spam detection is necessarily the same with as that of document classification with two classes i.e. Spam and Legitimate The job of our filtering process (learning algorithm or classification) is to take emails as inputs and tries to learn about patterns that will represents different classes. Once the learning is done, then given an unknown instance of email it should be able to filter out spam with high accuracy. The Problem of document classification has been illustrated in Figure 1.1.

Figure 1-1: Problem of Document Classification



Document classification has a wide range of application and is fundamental task in information retrieval. As more and more textual information is available on the internet its effective and fast retrieval is very important. Treating every web page as a document consisting of text will reduce the problem to document classification. Document classification is also used in organizing document for digital libraries. Other applications involve indexing, searching, web sites filtering, and hierarchical categorization of web pages.

1.7. Research Areas in Document Classification

Detailed research in the field of document classification has revealed the following areas of concern [1].

High Dimensionality

A faithful representation of a document that is based on a sequence of words implies high dimensionality since the number of distinct words in Document can be very large even if a document is of moderate size. Dimensionality reduction methods (will be discussed in chapter-3) are used to deal with this problem.

Statistical sparseness

High dimensional data are inherently sparse. Although the number of possible features (words) can be very large, each single document usually includes only a small fraction of them. Stemming algorithms can be used to reduced the sparseness of the data.

Domain knowledge

Since documents are given in natural language, it appears that linguistic studies should help in discovering their inner structure, and therefore in understanding their meaning. Help of domain experts are usually used to sort out this problem.

Multi Labeling

In the multi-labeled version of document classification, a document can belong to multiple classes simultaneously. In our case of spam detection we can say an email which is controversial and is considered as spam and legitimate at the same time. In case where each document has only a single label we say that the categorization is uni-labeled document classification problem.

1.8. Previous Work

There is rich literature on spam email in the context of document classification and on text retrieval. Here we will mention only those which are related to our work. The research work can be divided into two classes i.e. classification algorithms and feature space reduction techniques. Regarding the classification algorithms here is a brief summary. Naïve Bayesian approach was applied on the domain for the first time in [2] with phrasal and domain features. Memory based approach and its comparison with the naïve Bayesian has been discussed in [3]. Both of the classifiers achieve high accuracies and outperforming the traditional key-word based filtering. Support vector machine has been discussed in [4] with both textual and image based emails. AdaBoost Boosting algorithm is reported, outperforming Naïve Bayesian and Decision Trees methods in [5]. Common vector approach is discussed in [6] [7].

All of the above mentioned approaches do a good job of classification but that's not enough. We require the same job done with lesser computation complexity with a hope of increased accuracy. Many dimensionality reduction techniques have been investigated so far on the domain of text classification. Here is a brief summary. Mutual information has been discussed in [2] [3] and Latent semantic indexing in [6] [8]. Other methods such as CHI, Information Gain, and Document Frequency have been discussed on the domain of document classification in [9]. Clustering has been researched in [8] and Linear Discriminant Analysis in [10].

1.9. Summary

This chapter introduces the problems that the internet users are currently facing due to spam. Many solutions that have been implemented to deal with the problem have been discussed in detail with their advantages and disadvantages. The disadvantages that the existing solutions face and an alternative that minimizes those i.e. automatic spam filtering is introduced. Discussion on automatic spam email filtering as a two class document classification problem is presented. Finally the research areas that are present in document classification is also presented briefly.

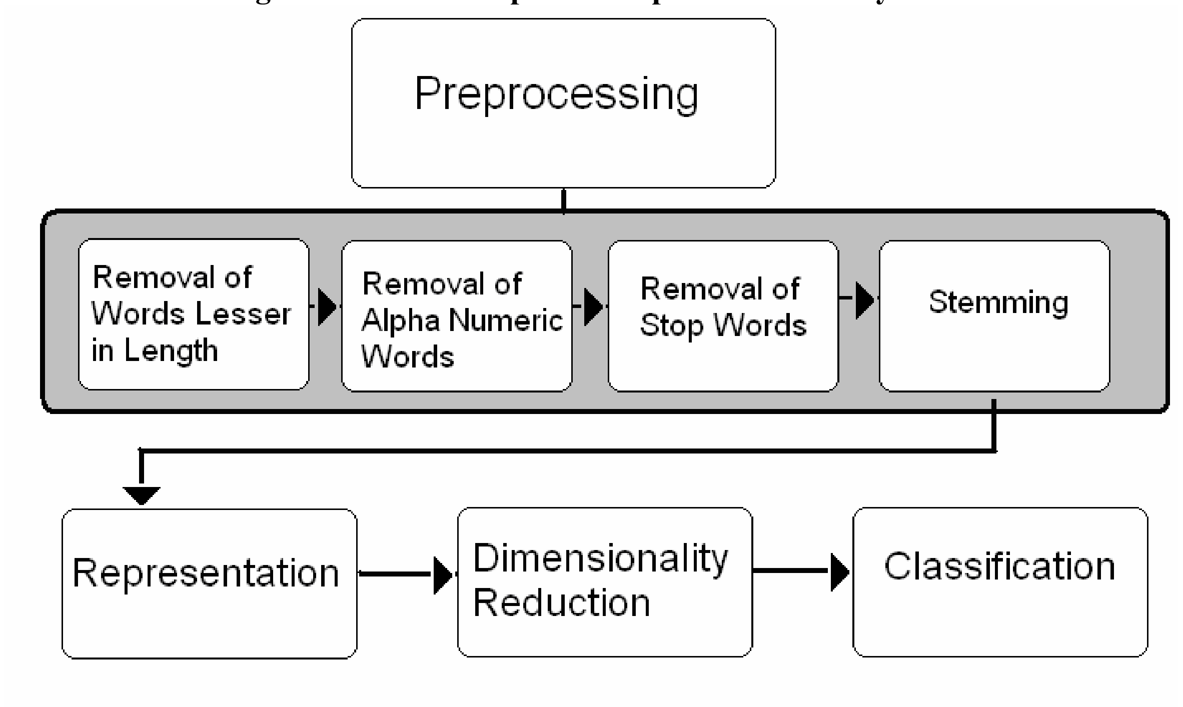
CHAPTER-2

DESIGN OF THE SYSTEM

2.1. Introduction

This chapter the basic algorithm and steps that comprise automatic spam email detection system. Most of the steps are similar to that of any text processing application. Text processing tasks are driven by huge textual information. For relevant information retrieval the textual data must be first passed through preprocessing steps. The preprocessed data is then represented in numeric form using suitable representation. The data represented is usually in very high dimensions. So dimensionality reduction techniques are used to sort out the features and select most suitable and relevant ones. The reduced data is then passed onto classifier to learn the patterns in the data. The main steps of the system are shown in Figure 2.1

Figure 2-1: Main Steps in the Spam Detection System



2.2. Spam Detection Algorithm

The spam detection algorithm used in our research work is shown below.

Spam Detection Algorithm

1. N = Number of email instances in the dataset.
2. M = Number of testing examples.

```
3. for I = 1 to N
    A. Pick up email instance I.
    B. Remove all the words that have length lesser than or equal to 2 from I.
    C. Remove the list of stop words from email I.
    D. Perform Stemming on email instance I.
    E. find list of unique words in the email instance and add that list to global
       list of unique words.
    F. Update the global list of unique words to reflect the unique words of the
       entire data set.
4. Represent the data using one of the weighting methods.
5. Apply the required dimensionality reduction technique.
6. Arrange the data into set of training and testing examples.
7. for J = 1 to M
    A. Pick up the testing example number J.
    B. Use the classifier to classify the example.
    C. Store the results of classification. i.e its accuracy and other evaluation
    measures.
8. Take an average of the evaluation measures to reflect the performance over the
   entire set of testing examples.
```

2.3. Preprocessing

In text retrieval tasks the preprocessing of the textual information is very critical and important. Main objective of text preprocessing is to remove data which do not give useful information regarding the class of the document. Furthermore we also want to remove data that is redundant. Most widely preprocessing steps in the textual retrieval tasks are removing of stop words and performing stemming to reduce the vocabulary. In addition to these two steps we also removed the words that have length lesser than or equal to two. Next we are going to describe the preprocessing steps in detail.

2.3.1. Removal of Words Lesser in Length

Investigation of English vocabulary shows that almost all such words whose length are lesser than or equal to two contains no useful information regarding class of the document. Examples includes a, is, an, of, to, as, on etc. though there are words which have length of three and are useless like the, for, was, etc but removing all such words will cost us loosing some words that are very useful in our domain, like sex, see, sir, fre (often fre is used instead of free to deceive the automatic learning filter).

All email of the data set were passed through a filter which removed the words that have length lesser than or equal to two. This removed bundle of words from the corpus that were useless and reduced the size of the corpus to great extend.

2.3.2. Removal of Alpha Numeric Words

There were many words found in the corpus that were alpha numeric. Removal of those terms was important as they do not keep on repeating in the corpus and they are just added in the emails to deceive the filter so that our classifier fails to find patterns in the given email. Some of the important characteristics of the alpha numeric words found were

They do not keep on repeating in the email instances. In this sense they can be considered as unique terms. They are present in large numbers in the corpus and adding them to our features set will have drastic increase in the features set size with little of information.

Counting the number of alpha numeric words in subject line or in the entire email might be helpful as spams are reported to contain large number of alpha numeric words [2]. So a single feature containing the number of alpha numeric words in an email might be helpful.

2.3.3. Removal of Stop Words

In information textual retrieval there are words that do not carry any useful information and hence are ignored during indexing and searching. Stop terms definition in context of internet search engines is ‘words that is so common on the Internet that search engines ignore them. E.g. homepage, home page, www, Web, Web page, the, of, that, is and, to, etc[1]'. In terms of database searches it is defined as 'words that databases will not search for[2]'. In general and for document classification tasks we consider them as words intended to provide structure of the language rather than the content and mostly include pronouns, prepositions and conjunctions. [13]. Two sets of experiments were performed. List of stop words in both of the experiments were different. In the first set of experiments the list of stop words contains 30 words. The list is shows in figure 2.2.

Figure 2-2: Stop Word List for Experiment Set 1

then, there, that, which, the, those, now, when, which, was, were, been, had, have, has, will, subject, here, they, them, may, can, for, such, and, are, but, not, with, your.

In the second set of experiment 571 stop words were used. The list was used in smart system [12] and was obtained from [11]. Some of the stop words from this list are given in table 2.3.

Figure 2-3: Some Stop Words Used in Experiment Set 2

alone, anyways, along, anywhere, able, already, apart, about, also, appear, above, although, appreciate, according, always, appropriate, between, be, beyond, became, both, because brief, become, but, becomes, by, becoming, before

Results with the second list of 571 words revealed better performances than the first set. So later on all of the experiments were conducted using the second list and the first list was deleted and not used any more.

2.3.4. Stemming

The second main preprocessing tasks applied in textual information retrieval tasks is the stemming. It can be defined as 'an algorithm developed to reduce a search query to its stem or root form, in other words, variations of particular words such as past tense and plural and singular usage are taken into account when performing a search, For example, applies, applying & applied matches apply[3]'. In the context of searching it can be defined as

[1] www.pro-seo.com/glossary.html
[2] www.methodist.edu/library/guides/libraryvocab.htm
[3] www.pr3.co.uk/seo/seo-glossary.php

"expansion of searches to include plural forms and other word variations[4]". In the context of document classification we can define it to be a process of representing words and its variants with its root. We used the porter stemming algorithms described in [14]. Implementation of porter's algorithm in Matlab was downloaded from [33]. Figure 2.4 shows some examples of the words after being stemmed with porter's algorithm.

After performing stemming the preprocessing of data is completed. The original 9.02 Mega Bytes of our corpus was reduced to about 4.5 Mega Bytes after the preprocessing steps mentioned.

Figure 2-4: Few Examples of Words with their Stems

Words	Stem
ponies	poni
caress	caress
cats	cat
feed	fe
agreed	agre
plastered	plaster
motoring	motor
sing	sing
conflated	conflat
troubling	troubl
sized	size
hopping	hop
tanned	tan
falling	fall

2.4. Representation of Data

The Next main task was the representation of data. The data representation step is needed because it's very hard to do computations with the textual data. The representation should be such that it should reveal the actual statistics of the textual data. Data representation should be in a manner so that the actual statistics of the textual data is converted to proper numbers. Furthermore it should facilitate the classification tasks and should be simple enough to implement.

[4] members.optusnet.com.au/~webindexing/Webbook2Ed/glossary.htm

The representation schemes considered in thesis were based on words statistic. There are some representations schemes suggested in [2] which work with the hand made phrasal statistics also. We used the words statistics due to its simplicity and secondly as the instances of emails changes then using the predefined phrases would not have that much of the effect on accuracy. It should be noted that the words statistics completely ignores the context in which the word is used. It rather looks for just the occurrences of words in the email instances and forms those statistics as the basis for prediction. Considering the context of words require many natural language processing tasks to be performed and will increase the complexity of solution. Further more non contextual words statistics have been used over the years on document classification tasks with acceptable performances. That's why we also used the non contextual representation of words. Next we will describe different representation schemes that have been used in the textual processing tasks.

2.4.1. Term Weighting Methods

Consider each email instance as a column vector D, whose values are weights assigned to terms based on the statistics in the email instance and in the entire corpus. D = (w_1, w_2 w_3 w_4 w_5,....., w_n).Where $\boldsymbol{w_i}$ is the weight of *ith* term (feature or word) of document d. combining the whole email instances in a single table will take the form as shown in table 2.1. The table is known as term document matrix. The dimensions of the table are M×N where ***M*** equals the number of distinct features and ***N*** equal the number of email instances. Each element $\boldsymbol{a_{ij}}$ of the term-document matrix represents the degree of relationship between term ***i*** and email instance ***j*** by means of one of the term weighting schemes described in the section latter.

Table 2-1: Representation of Data in Tabular Form

	Email #1	Email #2	Email #3	…….
Feature #1	W_{11}	W_{12}	W_{13}	…….
Feature #2	W_{21}	W_{22}	W_{23}	…….
Feature #3	W_{33}	W_{32}	W_{33}	…….
…………..	…….	…….	…….	…….

The traditional term weighting approach to document classification so far has been using representation in a word-based input space i.e. as a vector in some high dimensional space where each dimension corresponds to a word.[1].

There exist many term weighting methods which will calculate the weight for term differently. These weighting approaches are based mostly on following observations [15].

- The relevance of a word to the class of an email is proportional to the number of times it appears in the email.
- The discriminating power of a word between emails is less, if it appears in most of the emails in the emails collection. In other words, terms which are present in lesser number of emails are more discriminative.

Comparative study of different term weighting approaches in automatic text retrieval is presented by Salton and Buckley in [16]. Before defining each of the term weighting methods individually we define few terms first to make the understanding easier.

$\boldsymbol{tf_{ij}}$ as the frequency of term $\boldsymbol{i}$ in document $\boldsymbol{j}$, $\boldsymbol{N}$ as the total number of documents or emails in the corpus, $\boldsymbol{n_i}$ as the number of documents in the corpus where term $\boldsymbol{i}$ appears and $\boldsymbol{M}$ as the number of terms in the document collection (after stop words removal and stemming).

2.4.1.1. Boolean Weighting

It is the simplest of the term weighting methods where all the data is represented using Boolean values. Mathematically it can be represented as

$$Boolean\_W_{ij} = \begin{pmatrix} 1 & \text{if } tf_{ij} > 0 \\ 0 & \text{otherwise} \end{pmatrix}$$

A term will get a weight of 0 in email j if it is not present otherwise it will get a weight of 1. Boolean weighting makes the computation easy but do not consider the actual statistics of terms in the emails that's why it does not achieve as high accuracy as some of the other weighting methods does.

2.4.1.2. Term frequency

Also widely know as bag of words weighting and vector space model. It is also relatively simple weighting which counts the number of occurrences of term in an email. Mathematically it can represented as

$$Term\_Frequency\_W_{ij} = tf_{ij}$$

We performed few experiments with this weighting until we discovered the TFIDF with lengths normalized later on.

2.4.1.3. Term Frequency with Lengths Normalized

In order to cope with documents of different lengths a variant of term frequency is introduced. Here every weight of a term will be divided by the total number of terms frequencies in the email instance. Mathematically it can be represented as

$$TF\_Normalized\_W_{ij} = \frac{tf_{ij}}{\sum_{k=1}^{M} tf_{kj}}$$

2.4.1.4. Term Frequency inverse document frequency

This is the most widely used weighting scheme. Term frequency and Boolean weighting do not take the global statistics of the term into account. As already established that those terms whose presence is in lesser number of emails can discriminate well between the classes. TFIDF representation takes this property coupled with term frequency to define a new weighting which can be expressed mathematically as

$$TFIDF\_W_{ij} = tf_{ij} \times log\left(N/n_i\right)$$

2.4.1.5. Term Frequency inverse document frequency with lengths Normalized

To account for the documents of different lengths the weights obtained from the TFIDF are normalized. Mathematically the normalized version can be expressed as

$$TFIDF\_Normallized\_W_{ij} = \frac{tf_{ij} \times log\left(N/n_i\right)}{\sum_{k=1}^{M} tf_{kj} \times log\left(N/n_k\right)}$$

TFIDF with lengths normalized is reported to perform better than the others in [16]. So we used this representation for most part of our experimentation.

2.5. Dimensionality Reduction

Dimensionality reduction can be defined as "It is mapping of a multidimensional space into a space of fewer dimensions. It is sometimes the case that analysis such as regression or classification can be carried out in the reduced space more accurately than in the original space"[5]. The data represented with any of the weighting method is in huge dimensions. It is because of the fact that every email instance was represented in terms of words and unique words in the entire data set were found out to be over forty thousands. The result of such representation will over fourty thousands weights to represents a single instance of an email. Computation in such a huge dimensionality will be very hard and inefficient to classify emails at real time. So feature space reduction methods needs to be used. The objectives of feature space reduction methods will be to reduce the dimensionality at lower cost of information loss and accuracy. It will help us to select those features that will discriminate well between the classes and have reduced time and storage requirements. Our research work is concentrated on this task. Comparison of different dimensionality reduction techniques were carried out on a publicly available corpora to sort out the best in terms of accuracy and other evaluation measures.

2.6. Classification

The email instances represented in the reduced dimensions will be provided as inputs to the classification algorithm. A classification algorithm can be defined as "A predictive model that attempts to describe one column (the label) in terms of others (the attributes). A classifier is constructed from data where the label is known, and may be later applied to

[5] en.wikipedia.org/wiki/Dimensionality_reduction

predict label values for new data where the label is unknown. Internally, a classifier is an algorithm or mathematical formula that predicts one discrete value for each input row”[6]. In mathematical terms we can define classification as “Mapping from a (discrete or continuous) feature space X to a discrete set of labels Y”[7]. In Simple terms classification is a task of learning data patterns that are present in the data from the previous known instances and associating those data patterns with the classes. Later on when given an unknown instance it will search for data patterns and thus will predict the class based on the absence or presence of data patterns

Classification algorithms can be divided into three classes based upon the origin from which they evolve [17].

2.6.1. Statistical

The algorithms in this class can be further subdivided into two classes. The first classes of algorithms are those which are derived and based on the fisher’s earlier work on linear discriminant analysis. The second classes of algorithms are those which are based on the joint probability of features distribution which in turn provide rules for classification.

The widely used assumption behinds these algorithms are that they will be used by statisticians and will require some human intervention in variable selection and over all structuring of the problem. Examples of this class include linear discriminant analysis and naïve Bayesian classifiers.

2.6.2. Machine learning

Classification algorithms in this class are those which encompass automatic computing procedures based on logical operations that will learn a task from a series of examples. The classification tasks here are mostly automatic and require minimum human intervention. Some of the most famous algorithms in this class are decision-tree approaches, inductive logic procedures and genetic algorithm. The main characteristic features of algorithms in

[6] www.purpleinsight.com/downloads/docs/visualizer_tutorial/glossary/go01.html
[7] en.wikipedia.org/wiki/Classifier_(mathematics)

this class are that they aim to generate classifying expressions simple enough to be understood by humans.

2.6.3. Neural networks

Neural network is an emerging branch of artificial intelligence inspired from the working of human brain. Algorithms that came under this heading consist of layers of interconnected nodes, (also known as learning entities) each producing a non-linear function of its input. The input of a node may be outputs of some other nodes or may be a direct form of the input data. The final output of the whole system is defined to be output of some predefined nodes. In this way the input data is passed through several nodes to produce the final output. Examples of classification algorithms belonging to this class are support vector machines and perceptron.

2.7. Summary

The chapters discussed the main steps that were required in the spam email detection system. The steps can be divided into three basic modules. The first one was preprocessing which involves removal of redundant data from the data set and then representing the data with numeric values. The preprocessing module involves stemming, removal of short length words, removal of stop words and representation using suitable weighting method. As the representation of data for textual application is in very high dimensions so the second module consists of dimensionality reduction techniques. The third module is used for the classification purposes and consists of classification algorithms. A brief overview of the classification algorithms classes was at the end of the chapter.

CHAPTER –3

DIMENSIONALITY REDUCTION TECHNIQUES

3.1. Introduction

This chapter is intended to introduce and describe the problem of dimensionality reduction, discusses the curse of dimensionality and illustrates three different classes of dimensionality reduction techniques. The first class of these techniques is based on probabilistic model of the data and comprises of mutual information, CHI statistic, Odds Ratio, Information Gain. The second class uses statistics of terms in the corpus. Techniques belonging to this class are Document Frequency, Term Frequency and Mean of Term Frequency inverse of Document Frequency. Finally the third class is based on transformation of the data and contains Principal Component Analysis and Linear Discriminant Analysis. Detailed description of every method with mathematical form is also provided. Furthermore different classes of dimensionality reduction techniques are also introduced.

3.2. Problem of Dimensionality Reduction

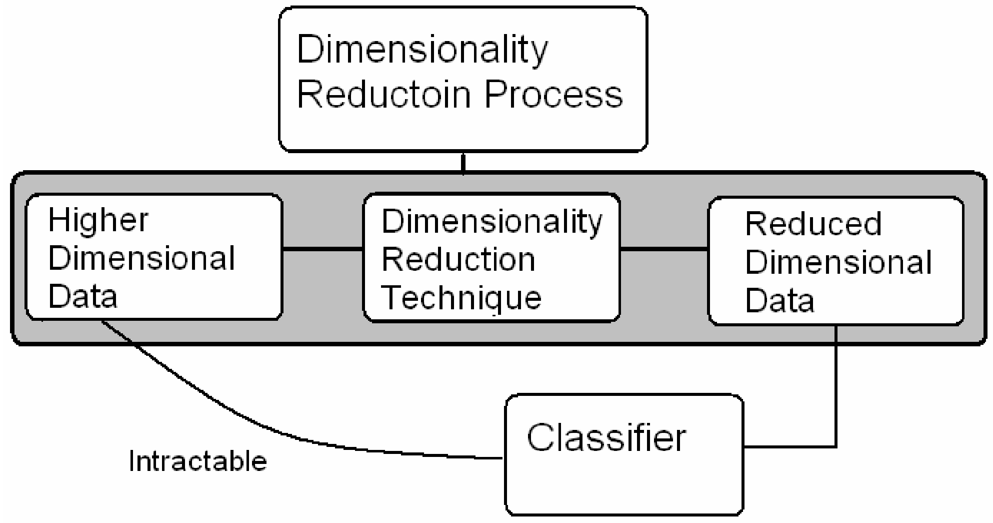
Consider an email application in which a system processes email instances consisting of collection of real values in the form of vector. The system can only be effective if the dimension of each individual vector is not too high. The problem of dimensionality reduction appears when the data are in fact of a higher dimension than tolerated level of dimensions. In the real time applications like email where the responses are required to be made very quickly, representation of an email in over thousands of features would be an unacceptable solution. Therefore dimensionality reduction techniques are called for with the following objectives.

- Reduce noise in document representation
- Understand the structure of the data
- Improve classification
- Improve the computational efficiency.

In mathematical terms, the problem we are investigating can be stated as follows: given the P dimensional random variable $X = (X_1, X_2, X_{3,}....... X_p)$, find a lower dimensional representation, $S = (S_1, S_2, S_{3,} .... S_p)$ with $k < p$, that captures the content in the original data, according to some criterion. The components of S are sometimes called the hidden components [18].

The goal of any dimensional reduction technique is to reduce the dimensions while keeping as much of the original information as possible so that to minimize the effect of dimensionality reduction on the overall accuracy of the system. The data represented in the reduced dimensions will be feed into the classification algorithm. Figure 3.1 summarizes this situation, showing the dimensionality reduction as a preprocessing stage before classification.

Figure 3-1: Dimensionality Reduction Process



The problems of dimensionality reduction can be roughly divided into three classes [19].

3.2.1. Hard Dimensionality Reduction Problems

Problems in this class have data of dimensionality ranging from hundreds to thousands of components (features), and usually a drastic reduction is sought. The components are often repeated measures of a certain magnitude in different points of space or in different instants of time. In this class we would find pattern recognition and classification problems involving images (e.g. face recognition, character recognition etc.) or speech (e.g auditory models). Document classification also belongs to this class.

3.2.2. Soft Dimensionality Reduction Problems

Problems of this class have data which is not very high-dimensional (less than a few tens of components), and the reduction not very drastic. Typically, the components are observed or measured values of different variables, which have a straightforward interpretation. Most statistical analyses in fields like social sciences, psychology, etc. fall in this class.

3.2.3. Visualization problems

Here data does not have normally very high dimension in absolute terms, but we need to reduce it to 2 or 3 in order to plot it. Several representation techniques [20] exist that allow visualizing of up to about 5-dimensional data sets.

3.3. Curse of dimensionality

Curse of dimensionality mentioned in [21] refers to the fact that some problems are very difficult to compute due to large number of features and the process of finding the solution becomes unacceptable because it exceeds available computing resources. This phenomenon in information retrieval domain has been first mentioned in [22].

Data in the high dimensional space is inherently sparse. So to estimate any parameter, we must have many samples to achieve a reasonable accuracy level but by increasing the sample size the vector space will increase approximalty more then exponential with the number of samples. This will severely restrict possible applications because the resulting computer power demand is too high and heavily restricts a potential set of solutions [8]. The same phenomena can be observed in the document classification tasks. Where the feature set size grows with the increase of the sample set size. This is due to the method that is used to extract features from the corpora. In all such applications facing this problem, dimensionality reduction methods are called for and used with great success.

3.4. Classes of Dimensionality Reduction Techniques

From the two separate perspectives of usage of class and the selection of features we can divide the dimensionality reduction techniques into the following four groups.

3.4.1. Supervised Dimensionality Reduction Techniques

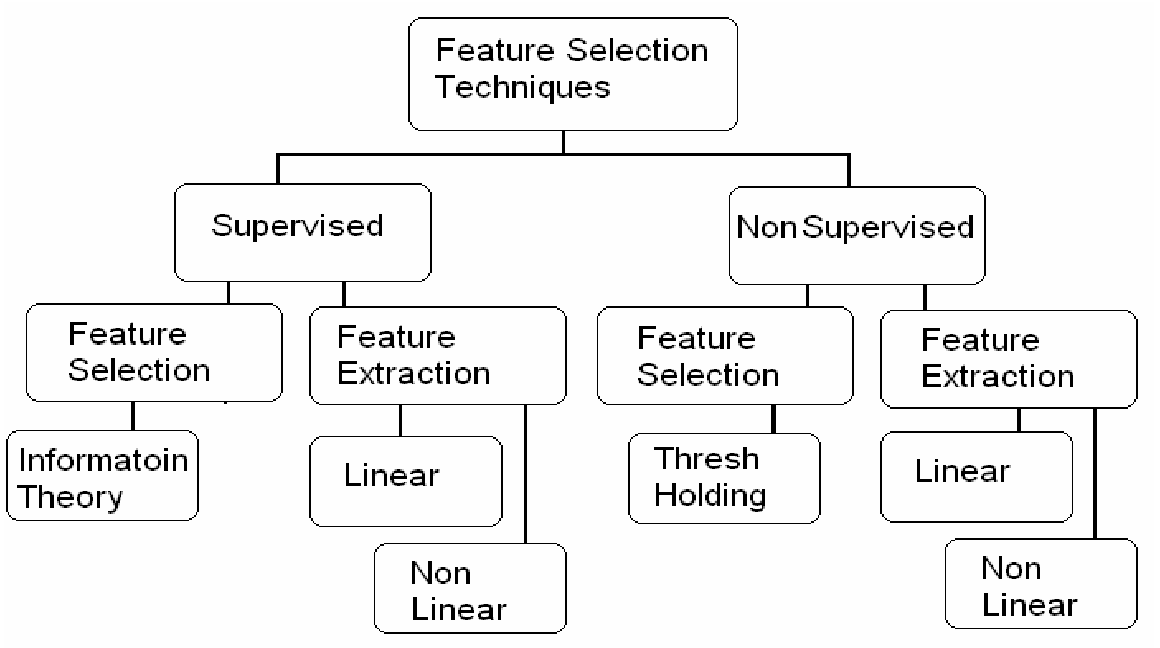
Those techniques which use the class information will come under this group. Examples include Mutual information, Information gain, Odds Ratio etc. They can be further classified as feature selection techniques or feature extraction techniques. Mostly the techniques that lie under this class are probability based and use the features and class joint probabilities to determine the usefulness of a feature.

3.4.2. Unsupervised Dimensionality Reduction Technique

Reduction techniques which do not consider the class information will constitute this group. Examples include Principal Component Analysis, Linear Discriminant Analysis, Document Frequency Thresh holding etc.

Regarding the selection of component (features) and processing on the data we can divide the reduction techniques into the following two groups [8].

Figure 3.2: Taxonomy of Reduction Techniques



3.4.3. Feature Selection

Also widely known as feature reduction. Techniques in this class represent the data using a subset of the original features, thus reducing the dimensions of the data. E.g. Document Frequency Thresh holding, Mutual Information etc. feature selection methods can be further classed as information based and thresh holding based. Information based techniques uses the probabilities information of classes and terms to decide the usefulness of a term. Thresh holding based techniques looks generally for frequencies of words in the corpus and then simply discarding those having lesser frequency then some predefined thresh hold. Thresh holding techniques are the most simple and fastest techniques. The important characteristic of thresh holding based techniques is that they completely ignore the existence of other features present [8].

3.4.4. Feature Extraction

Techniques here work onto transform original data and then represent the transformed data using lesser dimensions e.g. Latent Semantic Indexing, Independent component analysis etc. Features extraction is also known as feature generation. It can be further divided into two sub classes i.e. linear and non-linear. Linear dimensionality reduction techniques are those in which each transformed features can be represented as linear combination of the original features while in Non-linear the transformed features can not be represented in terms of the original features.

3.5. Dimensionality Reduction Techniques

In this section we are going to describe the dimensionality reduction techniques that were investigated and later on implemented. The techniques that were considered are divided into three classes i.e. Probability Based, Statistics Terms Based and Transformation Based. All the techniques have been tried to be represented in mathematical form. Before going further we will define few terms that will be used in the definitions.

tf_{ij} as the frequency of term i in document j, N as the total number of documents or emails in the corpus, n_i as the number of documents in the corpus where term i appears and M as the number of unique terms in the document collection (after stop words removal and stemming). ***C*** = 0 will stands for spam class and ***C*** = 1 for legitimate class. Similarly ***P(0)*** will mean the probability of class spam and ***P(1)*** will denote the probability of class legitimate. Lastly t_k = 0 will mean that the term k is not present while t_k = 1 means its presence.

3.5.1. Probability Based

Techniques in this category measures the relationship between terms and categories based upon the probabilities of terms in specific classes. All the techniques uses the joint probabilities and marginal probabilities of terms and classes. It should be noted that techniques in this category do not work with the TFIDF normalized data. They rather work with the Boolean form of the data. So the actual representation of the data that was set to TFIDF normalized is ignored. Another important feature of techniques from this class is that they completely ignore the dependence of features on one another i.e. they assume independence between the features.

3.5.1.1. Mutual Information

Mutual Information is one of the most widely used feature selection methods. It has been investigated on the domain of text classification in [9] and on the domain of spam email detection in [2] [3]. The mathematical form of Mutual Information corresponding to a term t_k for spam detection problem can be expressed mathematically as.

$$MI(t_k) = \sum_{C=0}^{1} \sum_{t_k=0}^{1} P(t_k, c) \times \log\left(\frac{P(t_k, c)}{P(t_k) \times P(c)}\right)$$

Where ***P(t***$P(t_k,c)$ is the joint probability distribution of term ***k*** with class ***c*** and $P(t_k)$ and ***P(c)*** are the marginal probabilities of term ***k*** and class ***c*** respectively.

Mutual information measures the information that a term and class share. It measures how much knowing one of these variables reduces our uncertainty about the other. It is seen from the above equation that for terms having an equal conditional probability, rare terms will have a higher *MI* value than common terms [9]. So, MI technique has the drawback that *MI* values are not comparable among terms with large frequency gaps.

If a term and its class are independent, then knowing a term does not give any information about its class and vice versa, so their mutual information is zero. In terms of entropy we can define mutual information as

$$MI(t_k) = H(t_k) - H(t_k \mid c)$$

Which can be interpreted as the information that ***c*** tells us about ***t_k*** is the reduction in uncertainty about ***t_k*** due to the knowledge of ***c***.

MI Scores for all of features were calculated this way and then sorted in descending order to select top scoring features.

3.5.1.2. Information Gain

Information Gain has been investigated on the domain of text categorization in [9] [23] and is some what similar to Mutual information. Information gain measures the number of bits of information gained for category prediction when the presence or absence of a term in an email is known. Information Gain for each unique term ***t_k*** can be calculated as follows [9].

$$IG(t_k) = -\sum_{C=0}^{1} P(c) \times log(P(c)) + P(t_k)\sum_{C=0}^{1} P(c \mid t_k) \times log(P(c \mid t_k)) +$$
$$P(\overline{t_k})\sum_{C=0}^{1} P(c \mid \overline{t_k}) \times log P(c \mid \overline{t_k})$$

Where ***$P(t_k)$, P(c)*** are the prior probabilities of term ***t_k*** and class ***c*** respectively and ***$P(c|t_k)$*** is the conditional probability between class ***c*** and term ***t_k***.

In terms of entropy, Information Gain can be seen as the decrease in entropy when the feature is given verses absent [13]. IG Scores for all of the terms were calculated and sorted in descending order to select the top scoring terms.

3.5.1.3. CHI-Square Statistic

The CHI-Square statistic measures the degree of dependence between a certain term and a certain category. In other words, it measures to what degree a certain term is indicative of membership or non-membership of a document in a certain category [24]. It has been applied to the domain of document classification in [9]. For a specific class ***c*** € (0,1). The CHI-Square Statistic for a term $\boldsymbol{t_k}$ can be found out mathematically as [25]

$$CHI(t_k,c) = \frac{N\left(P(t_k,c).P(\overline{t_k},\overline{c}) - P(t_k,\overline{c}).P(\overline{t_k},c)\right)}{P(t_k).P(\overline{t_k}).P(c).P(\overline{c})}$$

The CHI-Square Statistic for term $\boldsymbol{t_k}$ in both the classes were calculated this way and then averaged as follows [9]

$$CHI_{AVG}(t_k) = \sum_{C=0}^{1} P(c) \times CHI(t_k,c)$$

All the terms were sorted in descending order based on their $\boldsymbol{CHI_{AVG}}$ scores to select the top scoring terms.

3.5.1.4. Odds Ratio

Odds ratio is another probability based method of calculating the degree of relationship between a class and a term. Odds Ratio is applied on document classification in [26] and can be calculated for term $\boldsymbol{t_k}$ in a class ***c*** € (0, 1) as

$$OR(t_k,c) = \frac{P(t_k \mid c).(1 - P(t_k \mid \overline{c}))}{(1 - P(t_k \mid c)).P(t_k \mid \overline{c})}$$

The Odds Ratio for term $\boldsymbol{t_k}$ in both the classes were calculated this way and then averaged as follows.

$$OR_{AVG}(t_k) = \sum_{C=0}^{1} P(c) \times OR(t_k,c)$$

All the terms were sorted in descending order based on their $\boldsymbol{OR_{AVG}}$ scores to select the top scoring terms.

3.5.2. Statistics of Terms Based

These techniques are based on two different kinds of term statistics.

- The frequency of their presence or absence in instances.
- Their frequency in the individual instances.

Techniques belonging to this group are the most simple and fast in execution. They are approximalty linear in the number of documents of the training corpus. [9]

3.5.2.1. Term Frequency

Terms Frequency of term can be defined as the overall frequency of a term in the entire corpus i.e. in the entire email instances. To calculate the TF score, frequencies of terms in individual emails were first calculated and then all the frequencies of a term in the entire set of emails were added to find the TF Score for a particular term $\boldsymbol{t_k}$. Mathematically it can be expressed as

$$TF(t_k) = \sum_{j=1}^{N} tf_{kj}$$

Terms having less TF Score will be eliminated and those having high score will be selected.

3.5.2.2. Document Frequency

Document Frequency has been investigated in [9] [25]. Document Frequency Score for a term is the number of document from the entire corpus in which the term is present at least once. In other words DF Score for a term i is the number of documents in the training set for which the $\boldsymbol{tf_{ij}}$ (frequency of term i in instance j) is greater than of equal to 1. Mathematically it can expressed as

$$DF(t_k) = \sum_{j=1}^{N} T_{kj} \qquad \text{Where} \qquad T_{kj} = \begin{cases} 1 & if\ tf_{kj} \geq 1 \\ 0 & otherwise \end{cases}$$

The basic assumption behind this technique is that rare terms are either non informative classification purposes or not influential in global performance. Improvement in categorization accuracy is also possible if rare terms happen to be noise terms. [9].

3.5.2.3. Mean of Term Frequency Inverse Document Frequency

TFIDF with length normalized was reported to perform better than its counter part weighting techniques in [16] (mentioned earlier in Chapter-2 Section ‘term weighting methods'). Mean TFIDF is based on the same statistic. To calculate Mean TFIDF Score for a term, all the weights of the term in the entire data corpus against different email instances were added and then divided by the total number of instances. Mathematically it can be represented as

$$Mean\_TFIDF(t_k) = \frac{\sum_{j=1}^{N} tf_{kj} \times log\left(N/n_k\right)}{N}$$

The assumption behind this technique was the same as that of TFIDF weighting. i.e. Terms whose frequency are more in a document but are repeated in lesser number of document are good indicative of a class. Terms having lesser score of Mean TFIDF were the first ones to be eliminated.

3.5.3. Transformation Based

Techniques in this category are different from those earlier mentioned in the sense that they transform the original data. As mentioned earlier transformation based techniques falls in the feature extraction category and can be further classed as linear or non-linear. Here we will mention only the linear techniques that were investigated and implemented.

3.5.3.1. Latent semantic indexing

One of the most popular dimensionality reduction technique that gained popularity from the face recognition domain [32]. Also widely known as Principal Component Analysis (PCA) and Karhunen Loeve Transform (KLT). It has been investigated on the domain of spam email in [6] and on document classification in [8].

The algorithm tries to identify patterns in data, and express the data in a way as to highlight their similarities and differences. Since high dimensional data is hard to represent graphically, patterns in data can be hard to find in the absence of graphical representation. PCA can be used as tool for analyzing such data.

The second main advantage of PCA which is of concern here is that once we have found the patterns in the data we can compress it by reducing the number of dimensions without any significant loss in information.

Applying the PCA on a data can be done in using two different method i.e. covariance method and correlation method. We will describe the covariance method as we conducted the experiments with this method. The following are the main steps of the algorithm as described in [27] [28].

Step 1: Organizing Data

The first step in carrying out PCA algorithm on a sample data is to organize the data in a matrix or tabular form comprising of M features (words) and N data observations in such a way that features corresponds to rows and observations corresponds to columns. In this way data will be arranged as a set of N data vectors X_1, X_2 …. X_n with each X_i representing a single grouped observation of the M features. The single grouped observations will be denoted by X. The ultimate goal of running PCA algorithm on a sample data would be to represent each data vectors in terms of L features with L < M.

Step 2: Calculating Empirical Mean

Next we find the empirical mean along each dimension m = 1...M and place the calculated mean values into mean vector ***u*** of dimensions M × 1.

$$u(m) = \frac{1}{N}\sum_{n=1}^{N} X[m,n]$$

Step 3: Calculate Deviations from Mean

Subtract the empirical mean vector **u** from each column of the data matrix **X**. Store mean subtracted data in the *M* × *N* matrix **B**

$$B = X - u \times h$$

Where **h** is a 1 x *N* row vector of all 1's.

Step 4: Find the Covariance Matrix

Find the M × M empirical covariance matrix **C** from matrix **B** as follows.

$$C = \frac{1}{N}\left[B \times B^T\right]$$

Where × is the matrix multiplication.

Step 5: Find Eigen Vectors and Eigen Values of Covariance Matrix

Compute the Eigen values matrix **λ** and Eigen vectors matrix **V** of the covariance matrix **C** so that

$$C \times V = V \times \lambda$$

Step 6: Selecting Top scoring Eigen Values

Obtain the Eigen values from the matrix λ. Arrange them in increasing order and select the top few. In our example we select top L Eigen Values with L < M (typical L's taken were 10, 25, 50, 100, 250 and 500). Also select the Eigen vectors corresponding to the top most Eigen values selected. Make sure to get the correct pairings of the Eigen values and Eigen vectors.

Step 7: Obtaining the Transformed Data

The transformed data with dimensions L is obtained by multiplying the Eigen vectors matrix selected based on the top most Eigen values as follows.

$$Trans\_data = V^T_{Top\_Selected} \times B$$

3.5.3.2. Linear Discriminant Analysis

Originated from the work of fisher in [29], LDA has gain popularity for classification and dimensionality reduction problems. LDA has been applied to the domain of document classification in [23]. To our knowledge it has not been investigated on the domain of spam email yet. The idea behind LDA is maximizes the ratio of between-class variance to the within class variance in any particular data set thereby guaranteeing maximal separability.

The prime difference between LDA and PCA is that PCA does more of feature classification and LDA does data classification. In PCA, the shape and location of the original data sets changes when transformed to a different space whereas LDA doesn't change the location but only tries to provide more class separability and draw a decision region between the given classes. This method also helps to better understand the distribution of the feature data [30].

Data sets can be transformed and test vectors can be classified in the transformed space by two different approaches.

Class Dependent Transformation

This type of approach involves maximizing the ratio of between class variance to within class variance. The main objective is to maximize this ratio so that adequate class separability is obtained. The class-specific type approach involves using two optimizing criteria for transforming the data sets independently.

Class Independent Transformation

This approach involves maximizing the ratio of overall variance to within class variance. This approach uses only one optimizing criterion to transform the data sets and hence all data points irrespective of their class identity are transformed using this transform. In this type of LDA, each class is considered as a separate class against all other classes.

Since we are using the LDA for dimensionality reduction we will be using the class independent transformation. Next we are going to describe the detailed mathematical steps needed for running LDA over sample data.

Step 1: Organize the Data Set

The first step is the same as that of the PCA. The data will be arranged into a data matrix X with dimensions M×N, where M and N corresponds to number of features and data vectors respectively. Then we will split the data matrix into class specific matrices by putting the class specific data vectors into each of them. Now we have two kinds of matrices. One that corresponds to the entire data and second set of those matrices that corresponds to the individual classes.

Step 2: Calculating Means

Next Class specific means will be calculated from the class matrices earlier developed. In our case, two class specific means corresponding to spam and legitimate will be calculated as follows.

$$u_{spam}(m) = \frac{1}{A}\sum_{a=1}^{A} X[m,a]$$

$$u_{Legitimate}(m) = \frac{1}{B}\sum_{b=1}^{B} X[m,b]$$

Where A = number of data vectors corresponding to class Spam and B = number of data vectors corresponding to class Legitimate so that N = A + B.

The mean of the entire data also needed to be calculated and will be calculated as follows.

$$u = P_{Spam} \times u_{Spam} + P_{Legitimate} \times u_{Legitimate}$$

Where P_{spam} and $P_{Legitimate}$ are the prior probabilities of the respective classes.

Step 3: Calculating Within Class Scatter Matrix

Corresponding to all classes we will calculate the covariance or scatter matrix as follows

$$S_W = \sum_{C=0}^{1} P(C)(cov(C))$$

Where ***P(C)*** is the prior probability of the class ***C***. The covariance matrix for a class is calculated as follows

$$cov(C) = (x_C - \mu_C)(x_C - \mu_C)^T$$

x_C is the data matrix corresponding to class C

Step 4: Calculating between Class Scatter Matrix

Next the between class scatter matrix is obtained using the following equation.

$$S_b = \sum_{C=0}^{1} (\mu_C - \mu)(\mu_C - \mu)^T$$

μ is the mean vector of the entire data and μ_C is the class specific mean vector.

Step 5: Eigen Values and Eigen Vector of Criteria matrix

Next we computer the matrix ***T*** as follows.

$$T = S_W^{-1} \times S_b$$

It should be noted that ***T*** captures both of with in class scatter and between class scatter. Eigen values matrix λ and Eigen vectors matrix ***V*** of ***T*** are then computed. Top scoring Eigen values and Eigen vectors pairs are then selected.

Step 6: Obtaining Transformed Data

Transformed data Y in reduced space is obtained from the original data X as

$$Y = V_{Selected} \times X$$

Where $V_{selected}$ are the Eigen Vectors selected based on Eigen Values score.

3.5.4. Combination of Different Techniques

Apart from those techniques mentioned above few of their combinations were also considered. In fact LDA and LSI are hard to implement on the entire corpus with over 40 thousands features. Computation of Eigen vectors for such a huge covariance matrix was almost impossible given the computation resources. That's why LDA and LSI were implemented in conjunction with Mean TFIDF, DF and TF separately to see the performances. Mean TFIDF, DF and TF were also used in combination with MI, CHI and OR to see the effects on accuracy.

3.6. Summary

This chapter explained the problem of dimensionality reduction in general and with the context of document classification tasks. Efficient feature selection is not only important for reducing the complexity of the problem but also improves the accuracy of the system. Two main characteristics of the dimensionality reduction techniques i.e. their ability to work on the actual statistics of the data and their ability to represent relationships between the individual features, were explained in detail. Nine different techniques for dimensionality reduction were explained with their mathematical forms. Different classes of dimensionality reduction were also presented.

CHAPTER – 4

CLASSIFIERS

4.1. Introduction

Classification algorithms have wide range of application in many areas. They are used in medicine for drug trial analysis and MRI data analysis, in finance for share analysis and index prediction, in data communication for signal decoding and error correction, in computer vision for face recognition and pattern recognition applications, in voice recognition, in management for market prediction and uncountable other areas[1].

This chapter is intended to describe two widely used classification algorithms namely: Naïve Bayesian and Nearest Neighbor, which were used for the implementation purposes. Detailed description of each classifier with its background, and mathematical form is also presented. Both of the classifiers have been used on the document classification problem in [1] and on spam email detection in [2] [3]. Their interpretation for the document classification problem is well developed and understood. In addition, detailed description of the classification tasks with their origin is also presented.

4.2. History of classification

The earliest known system of classification is that of Aristotle, who attempted in the 4th century B.C. to group organisms into two classes i.e. plants and animals. The animal class was further divided into blood and bloodless and was also divided into three sub classes according to their movement i.e. walking, flying and swimming. Carolus Linnaeus, Swedish scientist from 18th century modified the Aristotle system by classifying plants and animals according to similarities in form. He divided living things into two kingdoms i.e. plant kingdom and animal kingdom. Furthermore he divided each of the kingdoms into smaller groups called genera and then divided each general into smaller groups called species.

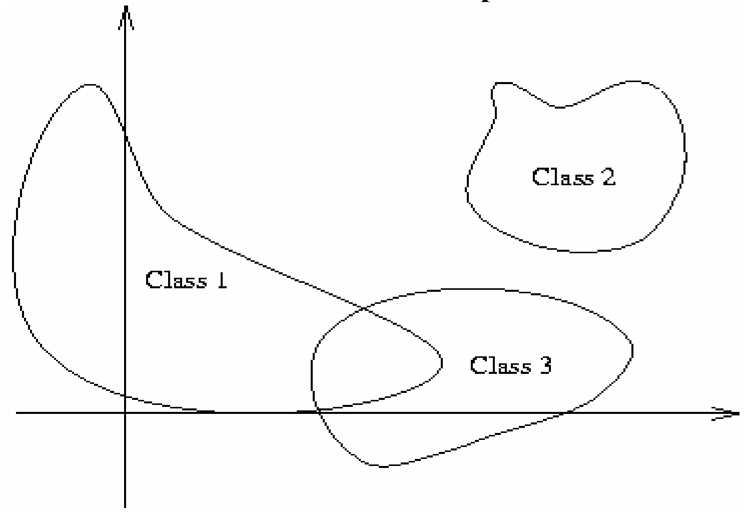
4.3. Problem of classification

A classification problem deals with the association of a given input pattern to one of the distinct classes. Patterns are specified by a number of features so it is natural to think of

[1] http://en.wikipedia.org/wiki/Classifier_(mathematics)

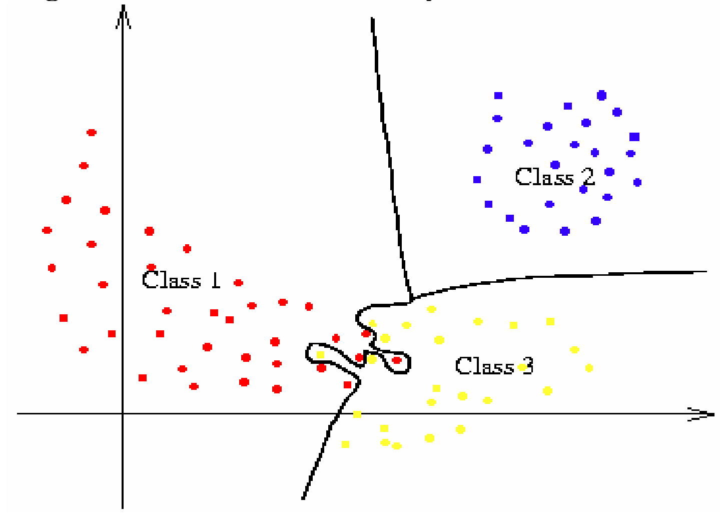
them as d-dimensional vectors, where d is the number of different features. This gives rise to a concept of feature space. Patterns are points in the d-dimensional space and classes are sub-spaces. Classification problem task is to determine which of the regions a given pattern falls into. If classes do not overlap they are said to be separable and, in principle, one can design a decision rule which will successfully classify any input pattern. A decision rule determines a decision boundary which partitions the feature space into regions associated with each class. It represents our best solution to the classification problem. Figure 4-1 illustrates a 2-dimensional feature space with three classes occupying regions of the space. The goal is to design a decision rule which is easy to compute and yields the smallest possible probability of misclassification of input patterns from the feature space.

Figure 4-1: Two Dimensional Feature Space with Three Classes



Our information about the classes is usually derived from some finite sample of patterns with known class affiliations. This sample is called a training set. If we make a decision boundary complex enough every pattern in the training set will be properly classified using the underlying decision rule, even if the distributions of patterns overlap. Classifiers are designed with a purpose of classifying unknown patterns and it is unlikely that an overly complex decision boundary would provide good generalization as it was tuned to perform extremely well on the training set. This is known as over-fitting the training set. Figure 4-2 shows a decision boundary over-fitting a training set distributed according to the classes of the Figure 4-1.

Figure 4-2: Decision boundary between the classes



Many algorithms have been developed which construct this decision boundary. In the next sections we are going to describe few of them in detail.

4.4. Naive Bayesian

Byes is believed to be the first to use probability inductively. He also established a mathematical basis for probability inference. Probability inference is the means of calculating, from the frequency with which an event has occurred in prior trials, the probability that this event will occur in the future. According to this Bayesian view, all quantities are either known or unknown for a person making infernece. Known quantities are defined by their known values while unknown quantities are described by joint probability distribution.

A specific contribution that Thomas Bayes made to the fields of probability and statistics is known as Bayes Theorem. Naïve Bayesian classifier takes the advantage of Bayes theorem which is stated as

$$P(A \mid B) = \frac{P(B \mid A) \times P(A)}{P(B)}$$

It is common to think of Bayes rule in terms of updating our belief about a hypothesis ***A*** in the light of new evidence ***B***. Specifically, our posterior belief ***P(A|B)*** is calculated by multiplying our prior belief ***P(A)*** by the likelihood ***P(B|A)*** that ***B*** will occur if ***A*** is true.

Bayesian filtering was proposed by [2] and later on used by [3]. It gained attention in 2002 when it was described in a paper by Paul Graham [31]. Since then it has become a popular mechanism for spam filtering. Bayesian poisoning is a technique used by spammers in an attempt to degrade the effectiveness of spam filters that rely on Bayesian filtering. A spammer practicing Bayesian poisoning will send out emails with large amounts of legitimate text[2].

The definition of Bayes theorem in the context of spam, says that the probability that an email is spam, given that it has certain words in it, is equal to the probability of finding those certain words in spam email, times the probability that any email is spam, divided by the probability of finding those words in any email.

$$P(Spam \mid Words) = \frac{P(Words \mid Spam) \times P(Spam)}{P(Words)}$$

There are many words that are indicative of an email to be classed as spam or legitimate. These words tends to have particular probabilities of occurring in respective classes e.g. many people will find the word sex in spam emails, but will rarely see it in legitimate emails. The filter is not aware of these probabilities in advance, and must be trained so it can build them up. To train the filter, the user must manually indicate whether a new email is spam or not. Thus for all words in each of the training email, the filter will adjust the probabilities that each word will appear in spam or legitimate email in its database.

Naïve Bayesian classifier is based on Bayes theorem and the theorem of total probability. For an email instance, the probability that it belongs to class $\boldsymbol{C}$ having a Vector of words $\boldsymbol{X = (x_1, x_2, x_3 \ldots\ldots\ldots x_N)}$ is

$$P(C_j \mid X) = \frac{P(C_j) \times P(X \mid C_j)}{\sum_{k=0}^{1} P(C_k) \times P(X \mid C_k)}$$

Where $\boldsymbol{J}$ € (Spam, Legitimate). In practice, the probabilities $\boldsymbol{P(X|C_i)}$ are impossible to estimate without simplifying assumptions, because the possible values of $\boldsymbol{X}$ are too many.

[2] http://en.wikipedia.org/wiki/Bayesian_spam_filtering

The Naive Bayesian classifier assumes that $x_1, x_2, x_3 \ldots\ldots\ldots x_N$ are conditionally independent given the category ***C***, which yields

$$P(C_j \mid X_i) = \frac{P(C_j) \times \prod_{t=1}^{N} P(x_t \mid C_j)}{\sum_{k=0}^{1} P(C_k) \times \prod_{t=1}^{N} P(x_t \mid C_k)}$$

Where N = total number of distinct words in the email instance X_i. So for every email instance class specific probabilities will be calculated and then the one with the highest probability would be the predicted class for that email instance.

4.5. Nearest Neighbor

Nearest Neighbor classifier is the easiest and one of the effective classifier for any task. It has been investigated for spam problem in [3]. The common feature of NN is that it stores all training instances in a memory structure, and uses them directly for classification. The simplest form of memory structure is the multi-dimensional space defined by the attributes in the instance vectors. Each training instance is represented as a point in that space. The classification procedure includes finding the distances of every point from the testing instance vector. The distances are usually found out using the Euclidean formula. The class to which the majority of nearest neighbors belongs is the predicted class for the testing example. Euclidean distance for a testing vector $X = (x_1, x_2, x_3 \ldots\ldots\ldots x_N)$ and training vector $X` = (x`_1, x`_2, x`_3 \ldots\ldots\ldots x`_N)$ can be found out as

$$Euclidean_{Distance}(X, X') = \sqrt{(x_1 - x_1')^2 + (x_2 - x_2')^2 + \ldots\ldots + (x_N - x_N')^2}$$

4.5.1. Weighted Nearest Neighbor

We studied two different weighted methods of the traditional k-NN. In weighted k-NN, the contributions of each of the k-NNs are weighted for each class according to some predefined criteria. Then for each class, the weights assigned by the weighting method are summed to obtain the score of the class for that instance. The class with the highest score is then reported to be the predicted class for that instance. In the preceding section we are going to explain the weighting methods in detail.

4.5.2. Weighting Based on Document Similarity

The weights assigned to each of the k-NN were based on its similarity to the test document. Document similarity between a test document ***X*** and its neighbor ***D*** was carried out by taking the cosine between them.

$$Cos(X,D) = \frac{X.D}{\|X\| \times \|D\|}$$

Class specific scores were then calculated using the following formula.

$$Score(C_j, X) = \sum_{i=1}^{k} Cos(X, D_i) \times Y(D_i, C_j)$$

Where ***k*** equals the number of nearest neighbors that we wish to find. $D_1, D_2 \ldots D_k$ are the neighbors founded while ***j*** € (Spam, Legitimate). All the neighbors were arranged in ascending order of their distances from the test example such that D_1 is the most nearest neighbor D_k the farthest. $Y(D_i, C_j)$ is a function whose value is 1 if D_i belongs to category C_j and 0 otherwise. The test document ***X*** is assigned to the category with the highest score.

4.5.3. Weighting Based on Distance

In this variant of k-NN the weights assigned to each of the neighbors were based on its distance from the test example. Neighbors with the nearest distance from the test example will get higher weights then those that were away from the test example. The weights defined were whole numbers and starts with the value of k for the most nearest neighbor and k-1 for the next nearest neighbor and so on. The weights for the class were then summed up and the class with the highest score is the predicted class for the example.

$$Score(C \ , X) = \sum_{i=1}^{k} W_i \times Y(D_i, C \ )$$

Where $W_i = k - i + 1$ and corresponds to the ith nearest neighbor and ***k*** equals the number of nearest neighbors that we wish to find. $D_1, D_2 \ldots D_k$ are the neighbors founded in descending order of distances from the test example, while C € (Spam, Legitimate). All the nearest neighbors were arranged in ascending order of their distances from the test example such that D_1 is most nearest neighbor and D_k is farthest. ***Y(Di, C)*** is again the same function as mentioned in the above section whose value is 1 if D_i belongs to category C_j and 0 otherwise. The test document ***X*** is assigned to the category with the highest score.

4.6. Summary

Classification tasks found their origin in 400 B.C when Aristotle classified organisms into two classes. Since then classification has found application in wide variety of scientific application. This chapter discussed in detailed the description of classifiers that were used for the experimental purposes. Three widely used and well understood classifiers on the domain of document classification were explained with their mathematical forms. Two variants of k-NN were also discussed in the chapter.

CHAPTER-5

RESULTS AND DISSCUSSION

5.1. Introduction

This chapter contains detailed description of the experiments conducted along with their results. The evaluation measure mainly used was accuracy. Results were also compiled for Spam Recall and Spam Precision, the two widely known measures used in information retrieval tasks. At the end of the chapter observations based on the results were presented.

5.2. Evaluation Measures

We used the evaluation measures that were used in [2] [3]. Let N_{Spam} and N_{Legit} be the total number of spam and legitimate emails in the entire corpus and let $N_{Y\text{-}Z}$ be the number of email instances that were classified as Z but belongs to class Y. where {Y, Z € (spam, legitimate)}.In classification tasks performance is often measured in terms of accuracy and error which can be defined as follows

$$Accuracy = \frac{N_{Spam \to Spam} + N_{Legit \to Legit}}{N_{Spam} + N_{Legit}} \qquad Error = \frac{N_{Legit \to Spam} + N_{Spam \to Legit}}{N_{Spam} + N_{Legit}}$$

In the above formulas both type of errors are treated equally i.e. $N_{Legit\text{-}Spam} = N_{Spam\text{-}Legit}$. However identifying legitimate email as spam is more harmful and costly then identifying spam as legitimate. To reflect this cost difference we introduce a constant **λ** and redefine accuracy and error as

$$WAC = \frac{\lambda . N_{Legit \to Legit} + N_{Spam \to Spam}}{N_{Spam} + N_{Legit}} \qquad ERR = \frac{N_{Legit \to Spam} + N_{Spam \to Legit}}{N_{Spam} + N_{Legit}}$$

The values of **λ** that were used in the experiments were mostly 9 and 99.

We also measured our results in terms of recall and precision which are two widely used techniques used in information retrieval tasks. SP and SR are given as

$$SP = \frac{N_{Spam \to Spam}}{N_{Spam \to Spam} + N_{Legit \to Spam}} \qquad SR = \frac{N_{Spam \to Spam}}{N_{Spam}}$$

It should be noted that SP measures the degree to which the blocked messages are indeed spam while SR measures the percentage of spam messages that the filter manages to block [3].

5.3. Ling Spam Corpus

All the experiments were conducted using the ling spam corpus[1]. The corpus contains 2412 legitimate emails and 481 spam emails. The spam rate of the corpus is about 16.6%. The corpus contains only the textual information i.e. the subject and the email body. HTML headers and other information like attachments etc were discarded to make the processing of emails fast. The corpus is available in four versions. The four versions are defined as follows.

- Without Stop words removal and without Stemming.
- With Stop words removal but without Stemming.
- With Stop words removal and Stemming.
- Without Stop words removal and Stemming.

We used the first version in our experimentations because we were also interested in removing the alpha numeric terms and terms that have length lesser than or equal to 2. Elimination of stop words and performing of stemming were carried out manually later on.

5.4. Experimental setup

We used 10 folds cross validation in our experiments. The ling spam corpus was divided into ten parts and then the experiments were repeated ten times, each time reserving different part for the testing and the remaining nine for the training purposes. All the results were then averaged over the entire set of experiments.

[1] Corpus is available at http://www.iit.demokritos.gr/skel/i-config/downloads/lingspam_public.tar.gz

Two sets of experimental setup were used for the ling Spam corpus. In the first set of experiments the data was represented using the TF representation of the data without normalization. All the dimensionality reduction techniques were tested using the features set of sizes 20, 50, 100, 250 and 500. Only three methods were tested with this setup. The methods were Latent Semantic Indexing, Word Frequency Thresh holding and Mutual Information. MI and LSI were used in combination with Word Frequency Thresh holding i.e. first the features were Thresh holded with Words Frequency of 29 and then LSI and MI were run on the data. Two variants of MI were considered. One that works with the MI Scores of the entire data set and other that works with the MI Scores of individual files. The Thresh holds for the word frequencies used were 332, 583, 1079, 1730 and 2315 (as these correspond to 500, 250, 100, 50 and 20 features respectively). The classifier for these experiments were k-NN with k = 1, 3.

In the second set of experiments the representation of the data was changed to TFIDF with lengths normalized. Here 11 different methods were tested using the features set of sizes 10, 25, 50, 100, 250 and 500. It was very hard to run LSI and LDA over the entire data of over 40 thousands dimensions given the available computation resources. For this reason LSI and LDA were used in combination with DF, TF and Mean TFIDF i.e. DF, TF and Mean TFIDF were first used separately to select the top most 1500 features from the corpus and then LSI and LDA were used to transform the data. Experimental results corresponding to this set of experiments are shown from Tables 4.3 to 4.10. The classifier used for these experiments were k-NN with k equals 3, 5. The same reduction techniques were also compared with weighted k-NN based on the distances from the testing examples described in chapter-3 section 3.6.2.1. The values of k used again were k equal to 3, 5. Tables 4.11 to 17 summarize the results obtained with weighted k-NN.

5.5. Results of Experimental Setup 1

Tables 4.1 and 4.2 summarize the results of experimental setup 1. The list of stop words used in these experiments is presented in Chapter-2, Figure 2-2. Top scoring results in each category are marked as yellow to make the results obvious.

MI Scores based methods performs better with weighted accuracy. The best accuracy that the LSI and thresh holding achieves is almost about 94% while MI feature sets achieves as high as 98.3%. Careful investigation reveals that classification based on MI Scores of entire data set have slightly better results then those computed on individual files. Statistics of SRs of reveals LSI to be the obvious winner. Furthermore Thresh holding proves it self to be a strong competitor for LSI at higher feature set. The remaining two MI Scores based methods fails to impress. It can be seen that the MI Scores based techniques remains stable with their SRs results while the other two methods does not. Results with SP are quite different from that of SR. Here both MI Scores based methods over run the LSI and thresh holding methods. Furthermore, the MI Scores calculated over the entire data set performs better than the ones calculated on the individual files. LSI and thresh holding goes neck to neck but the winner is LSI in terms of numbers. Apart from few exceptions, there is decrease in the SP as the feature set increases for all of the four methods.

Table 5-1: Results of Experimental Setup 1 with k-NN (k = 1)

No Feature	**LSI**				**Thresh holding**				**MI(Entire data set)**				**MI(individual files)**			
	WAC λ=9 %	WAC λ=99 %	SR %	SP %	WAC λ=9 %	WAC λ=99 %	SR %	SP %	WAC λ=9 %	WAC λ=99 %	SR %	SP %	WAC λ=9 %	WAC λ=99 %	SR %	SP %
20	94.0	94.1	90.9	80.5	94.5	94.8	76.4	77.4	97.5	98.0	73.9	89.1	96.4	96.9	74.4	85.6
50	92.7	92.8	90.9	78.4	94.7	94.9	82.6	78.4	97.5	98.0	74.7	89.5	95.4	95.9	73.9	83.9
100	90.7	90.7	91.0	75.0	93.2	93.3	84.3	76.9	96.4	96.8	75.3	87.2	93.9	94.3	73.8	82.0
250	88.5	88.5	85.9	70.9	90.4	90.5	85.2	74.0	93.7	94.1	74.8	83.5	92.8	93.1	76.9	82.4
500	-	-	-	-	91.0	91.1	83.4	73.9	-	-	-	-	-	-	-	-

Table 5-2: Results of Experimental Setup 1 with k-NN (k = 3)

No Feature	LSI (PCA)				Thresh holding				MI(Entire data)				MI(individual File)			
	WAC λ=9 %	WAC λ=99 %	SR %	SP %	WAC λ=9 %	WAC λ=99 %	SR %	SP %	WAC λ=9 %	WAC λ=99 %	SR %	SP %	WAC λ=9 %	WAC λ=99 %	SR %	SP %
20	92.9	92.9	91.1	79.6	94.3	94.6	79.6	78.9	97.5	98.0	73.9	89.1	96.0	96.5	74.6	84.8
50	91.3	91.3	92.4	77.0	93.2	93.4	83.1	76.7	97.8	98.3	74.0	90.7	95.5	95.9	73.7	84.1
100	89.0	89.0	91.9	74.1	91.9	92.1	84.1	77.5	96.3	96.7	74.2	87.4	93.8	94.2	72.8	82.9
250	88.5	88.6	83.8	74.3	90.8	91.0	83.9	75.8	95.1	95.6	73.1	85.8	92.7	93.1	72.5	83.3
500	-	-	-	-	90.0	90.1	83	74	-	-	-	-	-	-	-	-

The important Results of experimental setup 1 can be summarized as

- LSI performs better than thresh holding
- MI has greater accuracy then LSI and Word Frequency Thresh holding.
- MI Scores corresponding to the entire data set performs better than MI Scores calculated over individual files.

5.6. Results of Experimental setup 2

The second experimental setup can be divided into two categories. In the first category the classifier used was simple k-NN while in the second category the classifier used was weighted k-NN. First we will mention the results obtained with simple k-NN

5.6.1. Simple k-Nearest Neighbor

Tables 4-3 to 4-10 summarize the results obtained with simple k-NN.

Table 5-3 Spam Precision with simple k-NN (k = 5)

Features	10	25	50	100	250	500
DF	66.6	76.2	80.4	86.8	91.3	95.7
TF	64.7	79.7	83.9	86.5	94.2	94.6
M_TFIDF	67.6	68.8	84.7	85.6	89.5	97.4
TF_LSI	94.3	97	96	96.9	97.4	97.8
DF_LSI	94.1	96.8	95.9	96.6	97.3	97.3
M_TFIDF_LSI	94.5	97	96.1	96.6	97.1	97.9
DF_LDA	99.5	96.8	94.8	89.5	84.5	74.7
TF_LDA	90.5	63.7	76.8	82.4	75.7	68.4
M_TFIDF_LDA	98.8	94.7	93.4	92.2	86.5	80.2
MI	92.8	97.1	93.8	98.3	99	98.8
CHI	91	97.1	98	98.5	99.2	98
OR	76	76.2	77.2	71.2	73.9	70.2

Table 5-4: Spam Precision with simple k-NN (k = 3)

Features	10	25	50	100	250	500
DF	66.5	74.1	78.1	85	89.6	94.3
TF	65	77.7	81	84.8	91.8	93.3
M_TFIDF	69.8	66.2	81.9	85.6	86	96
TF_LSI	92.9	95.9	95	94.9	95.6	96.1
DF_LSI	92.9	95.6	95	95.1	95.5	96.3
M_TFIDF_LSI	93.2	92.1	94.8	95	95.7	96.4
DF_LDA	99.5	96.8	91.6	84.2	73.1	61.2
TF_LDA	89.3	59	70.4	75.7	64.6	55
M_TFIDF_LDA	98.1	91.6	89.4	87.6	76.7	65
MI	91.3	95.4	91.2	93	98.4	98.7
CHI	91	95.1	97.3	97.6	98.7	98.4
OR	92	93	75.3	71.2	81.7	77.3

Table 5-5: Spam Recall with simple k-NN (k = 5)

Features	10	25	50	100	250	500
DF	63	60.9	53.9	58.6	43.7	35.8
TF	57	60.2	61.4	61.7	48	46.2
M_TFIDF	45.6	53.8	64.8	65.8	54.5	34.7
TF_LSI	91.4	88.1	82	70.6	52.7	33.7
DF_LSI	91.6	88.7	83.6	72.7	53.4	36.8
M_TFIDF_LSI	91.5	88.1	82.3	70.8	52.6	33.2
DF_LDA	88.4	72.4	50.9	37	26.5	18.2
TF_LDA	60.3	31.2	28.5	25	18.7	16.1
M_TFIDF_LDA	87.7	54.7	51.3	41.1	27.5	22.2
MI	70	60.2	58.5	50	38.7	24.7
CHI	74	58.5	48.8	37	28.7	17.8
OR	0	24.7	36.6	45.1	70	68.1

Table 5-6: Spam Recall with simple k-NN (k = 3)

Features	10	25	50	100	250	500
DF	64	64.2	59.4	63.6	50.3	43.1
TF	52.4	65.6	64.4	66.2	54.2	52
M_TFIDF	45.3	56.5	68.2	65.8	60	43.8
TF_LSI	91.9	89.8	85.8	77.6	62.6	46.2
DF_LSI	92	90.3	86.9	79.1	62.8	46.4
M_TFIDF_LSI	91.8	92.6	86	78.4	62.3	44.1
DF_LDA	88.4	72.4	50.9	38.7	29.1	21.7
TF_LDA	60.8	32.2	30.3	26.3	21.5	19.2
M_TFIDF_LDA	87.9	54.8	50.6	41.9	30.1	25
MI	74.3	63.7	63.5	55.5	47.8	34.1
CHI	74	62.9	53.8	45	37.8	26.3
OR	18	19.3	34.4	45.1	67	64.3

Table 5-7: Weighted Accuracy (λ = 9) with simple k-NN (k = 3)

Features	10	25	50	100	250	500
DF	92	94.6	95.4	96.3	97.5	98.1
TF	93.2	95.1	95.6	96	97.9	97.7
M_TFIDF	94.3	91.9	95.7	96.2	94.1	98.4
TF_LSI	98.2	98.9	98.7	98.6	98.5	98.4
DF_LSI	98.2	98.9	98.7	98.7	98.5	98.4
M_TFIDF_LSI	98.3	98.9	98.7	98.6	98.5	98.4
DF_LDA	99.5	98.5	97.9	97.2	96.4	95.5
TF_LDA	97.7	94	95.8	96.7	95.9	95.1
M_TFIDF_LDA	99.4	98	97.7	97.5	96.6	95.5
MI	98	98.5	97.2	97.1	98.6	98.4
CHI	97.9	98.5	98.6	98.5	98.5	98.3
OR	97.8	97.2	96.1	92.2	90.1	90

Table 5-8: Weighted Accuracy (λ = 9) with simple k-NN (k = 5)

Features	10	25	50	100	250	500
DF	92	95.2	96	96.5	97.8	98.2
TF	92.5	95.8	96.2	96	98.2	97.8
M_TFIDF	93.7	92.4	96.2	96.2	95.9	98.3
TF_LSI	98.6	99.1	98.9	98.8	98.6	98.4
DF_LSI	98.5	99.1	98.9	98.8	98.6	98.4
M_TFIDF_LSI	98.6	99.1	98.9	98.8	98.6	98.4
DF_LDA	99.6	98.9	98.3	97.7	97.4	97
TF_LDA	97.7	97.9	96.6	97.3	97	96.8
M_TFIDF_LDA	99.5	98.4	98.2	98	97.5	97.1
MI	98.2	98.7	97.6	98.7	98.5	98.3
CHI	97.9	98.7	98.6	98.5	98.4	98.1
OR	97.8	96.5	95.6	92.2	85.5	83.2

Table 5-9: Weighted Accuracy (λ = 99) with simple k-NN (k = 3)

Features	10	25	50	100	250	500
DF	92.6	95.2	96.2	97	98.5	99.2
TF	94	95.7	96.3	96.6	98.8	98.6
M_TFIDF	95.3	92.6	96.3	96.8	94.8	99.4
TF_LSI	98.4	99.1	99	99	99.2	99.5
DF_LSI	98.4	99	99	99	99.2	99.5
M_TFIDF_LSI	98.4	98.2	98.9	99	99.3	99.5
DF_LDA	99.8	99.4	98.9	98.3	97.7	97
TF_LDA	98.4	95.3	97.1	98.1	97.4	96.7
M_TFIDF_LDA	99.6	98.8	98.7	98.6	98	97
MI	98.4	99.2	97.9	98	99.7	99.7
CHI	98.4	99.2	99.5	99.6	99.7	99.7
OR	99.7	98.8	97.3	93.2	90.6	87.3

Table 5-10: Weighted Accuracy (λ = 99) with simple k-NN (k = 5)

Features	10	25	50	100	250	500
DF	92.6	95.8	96.8	97.3	98.9	99.4
TF	93.3	96.5	96.6	96.7	99.2	98.9
M_TFIDF	94.7	93.2	96.8	96.8	96.7	99.6
TF_LSI	98.7	99.3	99.2	99.4	99.6	99.7
DF_LSI	98.7	99.3	99.2	99.3	99.5	99.6
M_TFIDF_LSI	98.8	99.3	99.2	99.4	99.5	99.7
DF_LDA	99.8	99.4	99.3	98.9	98.8	98.6
TF_LDA	98.6	96.2	98	98.7	98.6	98.4
M_TFIDF_LDA	99.7	99.2	99.1	99.1	98.9	98.6
MI	98.8	99.5	98.4	99.6	99.7	99.7
CHI	98.4	99.5	99.6	99.7	99.8	99.7
OR	99.7	98	96.8	93.2	85.8	83.5

Results of SP show that LSI and LDA perform well at lower feature sets of 25 or lesser. MI and CHI starts improving in performance as the feature set size grows. At features set of size 25 or more MI and CHI outperforms LSI and LDA. DF, TF and Mean TFIDF do not match up the results shown by the others techniques and lacks behind. The highest value of SP against these three techniques is 97.4% but at feature set of size 500 while DF_LDA achieves 99.5% at feature set of size 10 only.

In order to investigate the performances of the methods in detail we took the average of the results for different features set size. Table 4-11 shows the averages taken for the entire sets of features. The top three scoring techniques values were colored yellow to highlight. It can be seen that CHI and MI are the best and LSI remains second but it should be noted that LSI performance over the lesser features set size was very impressive and better than those of CHI and MI. The reason for LSI lacking behind from CHI and MI is its performance degradation over higher features set. DF, TF, Mean TFIDF, LDA and OR remains lower in the 80's and apart from few instances never seems to impress. We can put LSI, CHI, and MI in the first category and the rest in the second category based on the performances.

Table 5-11: Spam Precision for the Entire Sets of Features

	Averages	
Technique	**k = 3**	**k = 5**
DF	82.8	81.2
TF	83.9	82.2
M_TFIDF	82.2	80.9
LSI	96.4	94.8
LDA	85.6	79.3
MI	96.6	94.6
CHI	96.9	96.3
OR	74.1	81.7

Best SR results are shown by LSI as can be seen from Table 4-5 and 4-6. LSI achieves very high SRs (In 90s) at features set of size 10 but after the features set size grows beyond 25 there is drastic decrease. LDA seems to be the second best at lower features set but falls again to drastic levels at higher features set. At higher features set of 250 and 500 OR produces the best of the results over running LDA and LSI. Table 4-11 shows the averages taken for the entire sets of features. LSI produces the best average while Mean TFIDF, TF and DF can fall in the second category. MI, CHI and OR can be put into the third category. Over all the results can be summarized as

- At lower features set LSI over run others.
- At higher features set OR has better results.

Table 5-12: Spam Recall for the Entire Sets of Features

	Average	
Technique	**k = 3**	**k = 5**
DF	57.4	52.6
TF	59.1	55.7
M_TFIDF	56.6	53.2
LSI	75.8	70.1
LDA	43.4	42
MI	56.4	50.3
CHI	49.9	44.1
OR	41.3	44

All the techniques achieve very high scores of accuracy in the 90s. Very few instances were reported to have accuracy below 90. LSI over runs in terms of numbers from its competitors and achieves as high as 99.6 at features set of 10 size only. MI and CHI proves them selves to be very strong competitors and lacks a bit from LSI. Averaging results shows that it’s very hard to choose between CHI, MI and LSI. CHI wins from LSI at higher value of λ i.e. 99 but LSI over runs CHI at lower values of λ i.e. 9. We can place CHI, MI and LSI in the first category, LDA in the second category while DF, TF and Mean TFIDF in the third. Over all the results can be summarized as

- At lower value of λ LSI performs better.
- At higher value of λ CHI and MI performs better.

Table 5-13: Weighted Accuracy (λ = 9) for the Entire Set of Features

	Average	
Technique	**k = 3**	**K = 5**
DF	95.6	95.9
TF	95.9	96
M_TFIDF	95.1	95.4
LSI	98.5	98.7
LDA	96.9	97.8
MI	97.9	98.3
CHI	98.3	98.3
OR	93.9	91.8

Table 5-14: Weighted Accuracy (λ = 99) for the Entire Set of Features

	Average	
Technique	**k = 3**	**K = 5**
DF	96.4	96.8
TF	96.6	96.8
M_TFIDF	95.8	96.3
LSI	98.9	99.2
LDA	98	98.7
MI	98.8	99.2
CHI	99.3	99.4
OR	94.4	92.8

5.6.2. Weighted k-Nearest Neighbor

Tables 4-15 to 4-22 summarize the results obtained with simple k-NN.

Table 5-15: Spam Precision with Weighted k-NN (k = 5)

Features	10	25	50	100	250	500
DF	67.2	75.3	78.8	85.7	90.2	94.6
TF	66.3	78.6	82.2	85.8	92.8	94
M_TFIDF	70.5	68.2	82.8	86.1	87.5	96.3
TF_LSI	93.3	96.2	95.1	95.7	96	96.5
DF_LSI	93.4	96.2	95.3	95.6	95.9	96.6
M_TFIDF_LSI	93.7	96.3	95.1	95.5	96	96.7
DF_LDA	99	96.8	92.9	85.9	76.2	65.5
TF_LDA	89.5	60.3	72.2	78.6	68.4	58.3
M_TFIDF_LDA	98.3	92.8	74.4	88.9	80.6	68.7
MI	91.9	95.9	93.7	94.2	98.4	98.2
CHI	91.6	95.6	97.6	97.8	98.8	98.4
OR	92	93.1	61.2	73.3	79.4	77.7

Table 5-16: Spam Precision with Weighted k-NN (k = 3)

Features	10	25	50	100	250	500
DF	67.2	68.2	70.3	77	81.9	86
TF	61	71	73.6	78.3	83.5	89.2
M_TFIDF	65.3	60.1	74.8	79.4	79.8	90.7
TF_LSI	89.4	92.1	90.2	88.7	88.5	96.5
DF_LSI	89.4	91.8	90.2	89.4	89.7	96.6
M_TFIDF_LSI	89.7	92.1	89.7	88.5	88.6	87.9
DF_LDA	94.5	96.8	78.2	68	52.8	43.3
TF_LDA	81.2	47.6	55.4	58.5	47	38.1
M_TFIDF_LDA	94.9	78.4	90.6	73.2	58.5	45.8
MI	88.2	90.4	88.2	88.9	92.6	91.8
CHI	88.5	91.1	93.4	94.8	92.8	91.5
OR	52	63.8	61.2	58.5	72.9	71.3

Table 5-17: Spam Recall with Weighted k-NN (k = 5)

Features	10	25	50	100	250	500
DF	64.4	65.1	59.3	63.3	49.8	43
TF	52.9	65.7	64.6	66.3	54.2	51.8
M_TFIDF	45.3	56.3	68.6	65.8	59.9	43.4
TF_LSI	92	89.8	85.6	77.4	62.2	45.9
DF_LSI	92	90	87	79	62.2	46.2
M_TFIDF_LSI	92.1	89.3	85.8	78	62.1	43.6
DF_LDA	87.9	72.4	50.8	38.6	27.9	21
TF_LDA	61.1	31.6	29.6	26.4	20.6	17.7
M_TFIDF_LDA	87.9	54.9	58.2	41.2	29.3	24
MI	74	63.7	63.1	55.1	47.1	34
CHI	73.6	62.7	53.4	44.3	37.1	25.8
OR	18	19.6	39.2	46.3	68.8	64.8

Table 5-18: Spam Recall with Weighted k-NN (k = 3)

Features	10	25	50	100	250	500
DF	64.4	75	69.7	72.2	60.7	54.3
TF	58.5	74.5	73.7	75.2	64.6	61.5
M_TFIDF	52.1	64.6	76.6	74.2	69.2	56.3
TF_LSI	93.6	92.6	89.7	86.3	75.3	45.9
DF_LSI	93.6	92.8	90.3	86.4	74.4	46.2
M_TFIDF_LSI	93.4	92.6	89.8	86.7	75	60.9
DF_LDA	89.8	72.4	59.1	49.5	40.9	34.1
TF_LDA	67.2	41.6	39.8	36.8	32.9	29.7
M_TFIDF_LDA	90.2	62.8	50.7	50.6	42	36.3
MI	78.8	71.6	71.5	65.4	59.4	48.1
CHI	78.4	70.1	63	56.3	49.6	39.8
OR	18.5	22	39.2	52.2	75.6	71

Table 5-19: Weighted Accuracy (λ = 9) with Weighted k-NN (k = 3)

Features	10	25	50	100	250	500
DF	92.3	92.3	93.1	94	95.8	96.2
TF	91.5	93.1	93.1	94.3	96	97
M_TFIDF	92.8	89.1	93.2	94.6	92.8	97.7
TF_LSI	97.3	98	97.6	97.2	96.9	98.5
DF_LSI	97.3	98	97.6	97.4	97.5	98.5
M_TFIDF_LSI	97.4	98.1	97.4	97.1	97	97
DF_LDA	98.7	98.9	95.8	94.2	91.5	89.7
TF_LDA	96.2	89.7	92.2	93.4	91.1	89.1
M_TFIDF_LDA	98.8	95.7	97.9	95.1	92.7	90
MI	97.4	97.8	96.7	96.3	97.6	96.9
CHI	97.4	97.9	98.3	98.4	97.6	97.2
OR	97.2	95.7	93.7	89.1	88.1	85.7

Table 5-20: Weighted Accuracy (λ = 9) with weighted k-NN (k = 5)

Features	10	25	50	100	250	500
DF	92.3	94.8	95.6	96.5	97.6	98.2
TF	93.4	95.3	95.9	96.2	98	97.9
M_TFIDF	94.4	92.3	95.9	96.2	94.9	98.4
TF_LSI	98.3	99	98.7	98.7	98.6	98.5
DF_LSI	98.3	99	98.7	98.7	98.6	97.9
M_TFIDF_LSI	98.4	99	98.7	98.7	98.6	98.5
DF_LDA	99.5	98.9	98.1	97.4	96.7	96
TF_LDA	97.7	94.3	96.1	96.9	96.4	95.8
M_TFIDF_LDA	99.4	98.1	95.1	97.6	97	96.1
MI	98.1	98.6	97.8	97.7	98.6	98.4
CHI	98	98.5	98.7	98.5	98.5	98.3
OR	97.8	97.2	93.7	92.3	89.5	87.6

Table 5-21: Weighted Accuracy (λ = 99) with weighted k-NN (k = 3)

Features	10	25	50	100	250	500
DF	92.8	93.6	93.5	94.5	96.5	97
TF	92.2	93.5	93.4	94.7	96.6	97.7
M_TFIDF	93	89.6	93.5	95	93	98.5
TF_LSI	97.4	98.2	97.8	97.4	97.4	99.5
DF_LSI	97.4	98.1	97.8	97.6	97.9	99.5
M_TFIDF_LSI	97.5	98.2	97.6	97.3	97.4	97.7
DF_LDA	98.9	99.4	96.5	95.1	92.5	90.8
TF_LDA	96.8	90.6	93.3	94.6	92.3	90.3
M_TFIDF_LDA	98.9	96.4	98.8	96	93.8	91.1
MI	97.8	98.3	97.2	96.9	98.3	97.9
CHI	97.9	98.5	99	99.2	98.6	98.3
OR	99.1	97.1	94.7	89.9	88.4	86

Table 5-22: Weighted Accuracy (λ = 99) with weighted k-NN (k = 5)

Features	10	25	50	100	250	500
DF	92.8	95.4	96.3	97.1	98.6	99.3
TF	94.2	95.9	96.5	96.8	98.9	98.8
M_TFIDF	95.4	93	96.4	96.8	95.6	99.5
TF_LSI	98.5	99.2	99	99.1	99.3	99.5
DF_LSI	98.5	99.2	99	99.1	99.3	99.5
M_TFIDF_LSI	98.6	99.2	99	99.1	99.3	99.6
DF_LDA	99.7	99.4	99	98.5	98.1	97.6
TF_LDA	98.5	95.6	97.4	98.4	97.9	97.3
M_TFIDF_LDA	99.6	99	95.8	98.8	98.4	97.5
MI	98.6	99.3	98.5	98.6	99.7	99.7
CHI	98.5	99.3	99.6	99.6	99.7	99.7
OR	99.7	98.8	94.7	93.3	89.9	88.1

SR results shows that LSI is the most effective techniques of all by a big margin. There isn't any other result comparable to LSI for smaller features set. The average values of LSI, SR for the entire set of features set is also the most leading ones. OR has few leading values at the higher sets but over all its average isn't that much impressive. LDA shows better results at lower features set but over all lacks consistency.

Table 5-23: Spam Recall for the Entire Sets of Features

	Average	
Technique	**k=3**	**k=5**
DF	66	57.4
TF	68	59.2
M_TFIDF	65.5	56.5
LSI	81.3	75.5
LDA	51.4	43.3
MI	65.8	56.4
CHI	59.5	49.4
OR	46.1	42.7

SP results shows that LSI performs better at lower features set and MI and CHI performs better at higher feature set sizes. The results are compatible with those obtained with simple k-NN. Further more there is an increase in the over all results of SP when we use the higher value of k. The results of CHI and MI increases till features set size increases to 250 but after that the results shows a little degradation.

Table 5-24: Spam Precision for the Entire Sets of Features

	Average	
Technique	**k=3**	**k=5**
DF	75.1	81.9
TF	76.1	83.2
M_TFIDF	75	81.9
LSI	90.4	95.4
LDA	66.7	83.4
MI	90	95.3
CHI	92	96.6
OR	63.2	79.4

The results of weighted accuracy were again very similar to that of simple k-NN. With increasing value of λ from 9 to 99 there is an increase in performance for CHI and MI. LSI performs well not only on lower features sets but also on higher features set sizes. LDA shows good results only at lower features set OR also shows an odd good result also but over lacks consistency. LSI, MI and CHI can be placed in the first category while LDA, TF, DF and Mean TFIDF can be placed in the second category and OR in the third category. Table 4-25 and 4-26 summarizes the averages of the techniques.

Table 5-25: Weighted Accuracy (λ = 9) for the Entire Set of Features

	Average	
Technique	**k=3**	**k=5**
DF	93.9	95.8
TF	94.1	96.1
M_TFIDF	93.3	95.3
LSI	97.5	98.5
LDA	93.9	97.2
MI	97.1	98.2
CHI	97.8	98.4
OR	91.5	93

Table 5-26: Weighted Accuracy (λ = 99) for the Entire Set of Features

	Average	
Technique	**k=3**	**k=5**
DF	94.6	96.5
TF	94.6	96.8
M_TFIDF	93.7	96.1
LSI	97.8	99.1
LDA	94.7	98.1
MI	97.7	99
CHI	98.5	99.4
OR	92.5	94

5.6.3. Simple k-NN versus weighted k-NN

In order to see the effect of changing the classifier from simple nearest neighbor to weighted nearest neighbor we took the average of all of the techniques results at specified features sets. Table 4-27 summarizes the results. SR is improved with weighted k-NN. SP results got worse with weighted k-NN at k = 3 but does not change great deal at higher values of k = 5. With weighted k-NN the accuracy was degraded at k =3 but does not change a great deal at higher values of k = 5 and matches the results produced by simple k-

NN. The immediate conclusion from the results is that weighted k-NN works better at k = 5.

Table 5-27: Averaged Results of Simple k-NN and Weighted k-NN

Spam Precision with k = 3						
Features	**10**	**25**	**50**	**100**	**250**	**500**
KNN	86.7	86	86.7	87.4	87.2	85.6
WKNN	80.1	78.6	79.6	78.6	77.3	77.3
Spam Precision with k = 5						
KNN	85.8	86.7	89.2	90	90.4	89.2
W-KNN	87.2	87.1	85.1	88.5	88.3	86.7
Spam Recall with k = 3						
KNN	70	63.6	61.1	56.9	48.7	38.8
W-KNN	73.2	69.3	67.7	65.9	59.9	48.6
Spam Recall with k = 5						
KNN	70	61.7	58.5	52.9	42.9	32.2
W-KNN	70.1	63.4	62.1	56.8	48.4	38.4
Weighted Accuracy with L = 9, k = 3						
KNN	97	96.9	97.1	96.9	96.7	96.8
W-KNN	96.1	95.3	95.5	95	94.5	94.4
Weighted Accuracy with L = 9, k = 5						
KNN	97	97.4	97.5	97.2	96.8	96.6
W-KNN	97.1	97	96.9	97.1	96.9	96.8
Weighted Accuracy with L = 99, k = 3						
KNN	97.6	97.5	97.9	97.7	97.7	97.7
W-KNN	96.6	95.9	96	95.6	95.2	95.3
Weighted Accuracy with L = 99, k = 5						
KNN	97.6	97.9	98.2	98.1	97.9	97.9
W-KNN	97.7	97.7	97.6	97.9	97.8	98

5.6.4. Naive Bayesian

Table 4-28 to 4-31 summarizes the results obtained with the Naïve Bayesian classifier.

Table 5-28: Weighted Accuracy (λ = 9)

Features	10	25	50	100	250	500
DF	98.2	98.6	98.7	99.2	99.4	99.7
TF	98	98.7	99.2	99.4	99.6	99.8
M_TFIDF	98.2	98.8	99.3	99.4	99.6	99.8
MI	99.7	99.8	99.8	99.8	99.9	99.9
CHI	99.7	99.8	99.8	99.8	99.9	99.9
OR	98.3	98.2	98.2	98.2	98.7	98.8

Table 5-29: Weighted Accuracy (λ =99)

Features	10	25	50	100	250	500
DF	99.8	99.8	99.8	99.9	99.9	99.9
TF	99.8	99.8	99.9	99.9	99.9	99.9
M_TFIDF	99.8	99.8	99.9	99.9	99.9	99.9
MI	99.9	99.9	99.9	99.9	99.9	99.9
CHI	99.9	99.9	99.9	99.9	99.9	99.9
OR	99.8	99.8	99.8	99.8	99.8	99.8

Table 5-30: Spam Recall

Features	10	25	50	100	250	500
DF	17	37	44	65.8	76.2	90.7
TF	11.2	40.6	63.1	72.6	83.5	93
M_TFIDF	19.1	46.5	70	75.3	84.9	92.9
MI	90.3	92.1	94.5	94.5	96	96
CHI	89.5	92.2	93.9	93.9	96	95.8
OR	23.5	18	19.5	19.8	44.2	60.8

Table 5-31: Spam Precision

Features	10	25	50	100	250	500
DF	30	80	82.2	100	100	100
TF	38.8	80	97.7	100	100	100
M_TFIDF	70	84.4	100	100	100	100
MI	100	100	100	100	100	100
CHI	100	100	100	100	100	100
OR	58.8	84.4	97.7	98.8	100	100

It can be seen that there is an over all improvement in the results against all of the features reduction techniques with Naïve Bayesian classifier. The 100 percent of SP was never achieved with the previous classifiers. MI and CHI achieves 100 percent of SP at features set of only 10 while others achieves it with a little greater size of the features set. An improvement in the results of SR can also be seen for MI and CHI. Previously the best results of SR for them were in 70's while with Naïve Bayesian it increased to 90's. The most important results to analyze were from weighted accuracy. With λ = 99 the results of all of the techniques never fall below 99.8% and λ = 9 the lowest value recorded were 98 %. Over all CHI and MI over runs the rest of the techniques with Naïve Bayesian.

5.6.5. Execution times

Tables 5-32 to 5-36 summarizes the results of the execution times taken by different dimensionality algorithms with the classifiers specified.

Table 5-32: Execution Times in Seconds with Simple KNN (K = 3)

Features	10	25	50	100	250	500
DF	0.23	0.34	0.54	1.0	2.32	4.62
TF	0.21	0.31	0.46	0.89	2.29	4.79
M_TFIDF	0.31	0.34	0.53	0.89	2.37	4.65
TF_LSI	0.25	0.35	0.60	1.12	2.65	5.83
DF_LSI	0.23	0.32	0.62	1.07	2.98	5.92
M_TFIDF_LSI	0.28	0.42	0.65	1.28	2.96	5.98
DF_LDA	1.10	1.35	2.25	3.48	7.37	14.60
TF_LDA	1.39	1.37	2.12	3.37	7.81	14.31
M_TFIDF_LDA	1.40	1.42	1.96	3.39	7.71	14.20
MI	0.51	0.39	0.62	1.07	2.35	4.50
CHI	0.20	0.34	0.54	1.06	2.32	4.42
OR	0.18	0.23	0.42	0.79	2.18	4.39

Table 5-33: Execution Times in Seconds with Simple KNN (K = 5)

Features	10	25	50	100	250	500
DF	0.26	0.32	0.50	0.90	2.37	4.53
TF	0.31	0.35	0.54	.95	2.59	4.75
M_TFIDF	0.25	0.31	0.48	0.89	2.31	4.54
TF_LSI	0.26	0.35	0.59	1.17	2.95	6.17
DF_LSI	0.26	0.35	0.60	1.15	3.06	5.96
M_TFIDF_LSI	0.28	0.39	0.60	1.10	2.98	5.90
DF_LDA	1.15	1.40	3.23	3.96	7.82	14.40
TF_LDA	1.2	1.4	2.92	4.04	8.43	14.23
M_TFIDF_LDA	1.2	1.43	1.95	3.95	8.40	14.15
MI	0.25	0.31	0.46	0.87	2.28	4.54
CHI	0.28	0.32	0.51	0.87	2.28	4.84
OR	0.26	0.31	0.51	0.85	2.29	4.56

Table 5-34: Execution Times in Seconds with Weighted KNN (K = 3)

Features	10	25	50	100	250	500
DF	0.25	0.37	0.53	1.03	2.39	4.57
TF	0.25	0.34	0.54	1.07	2.40	4.65
M_TFIDF	0.26	0.39	0.57	0.96	2.46	4.64
TF_LSI	0.28	0.43	0.64	1.14	3.01	6.01
DF_LSI	0.28	0.43	0.65	1.15	3.01	6.01
M_TFIDF_LSI	0.26	0.43	0.65	1.29	3.10	5.92
DF_LDA	1.15	1.51	3.20	4.31	8.01	14.34
TF_LDA	1.10	1.51	3.0	4.20	8.50	14.28
M_TFIDF_LDA	1.10	1.45	2.03	4.18	8.51	14.20
MI	0.25	0.37	0.51	1.06	2.34	4.64
CHI	0.31	0.34	0.57	0.87	2.32	4.92
OR	0.25	0.39	0.51	1.01	2.34	4.50

Table 5-35: Execution Times in Seconds with Weighted KNN (K = 5)

Features	10	25	50	100	250	500
DF	0.26	0.40	0.62	1.03	2.56	4.92
TF	0.25	0.35	0.64	0.93	2.39	4.07
M_TFIDF	0.26	0.31	0.51	0.92	2.40	4.54
TF_LSI	0.25	0.42	0.64	1.15	3.07	6.07
DF_LSI	0.28	0.39	0.62	1.20	3.07	6.04
M_TFIDF_LSI	0.31	0.42	0.64	1.20	3.06	6.00
DF_LDA	1.12	1.50	3.03	4.25	8.14	14.57
TF_LDA	1.10	1.46	3.00	4.12	8.64	14.37
M_TFIDF_LDA	1.20	1.79	2.21	4.17	8.89	14.43
MI	0.23	0.32	0.51	0.93	2.37	4.53
CHI	0.28	0.39	0.51	0.93	2.54	4.84
OR	0.25	0.34	0.59	1.04	2.31	4.57

Table 5-36: Execution Times in Seconds with Naïve Bayesian

Features	10	25	50	100	250	500
DF	0.03	0.04	0.06	0.12	0.25	0.43
TF	0.01	0.03	0.07	0.10	0.23	0.40
M_TFIDF	0.01	0.03	0.06	0.10	0.25	0.43
MI	0.03	0.03	0.04	0.09	0.20	0.39
CHI	0.01	0.03	0.04	0.06	0.18	0.46
OR	0.01	0.04	0.04	0.09	0.18	0.37

The results from the above tables are the learning times of the dimensionality reduction algorithms against the classifiers specified. It should be noted that there are no values in the top three results for LDA and very few times LSI showed good results. All the rest of the methods have approximately similar sort of results. The results with Naïve Bayesian were more promising than that obtained with the other two classifiers.

Table 5-37: Execution Times in Seconds of Dimensionality Reduction Methods

DF	TF	M_TFIDF	LSI	LDA	MI	CHI	OR
0.18	0.2	0.22	97.02	156.89	0.67	0.4	0.35

The results in the tables above (averaged for different values of k) with the execution times of dimensionality reduction methods added to them, gives us the following three graphical results.

Figure 5-1: Simple KNN

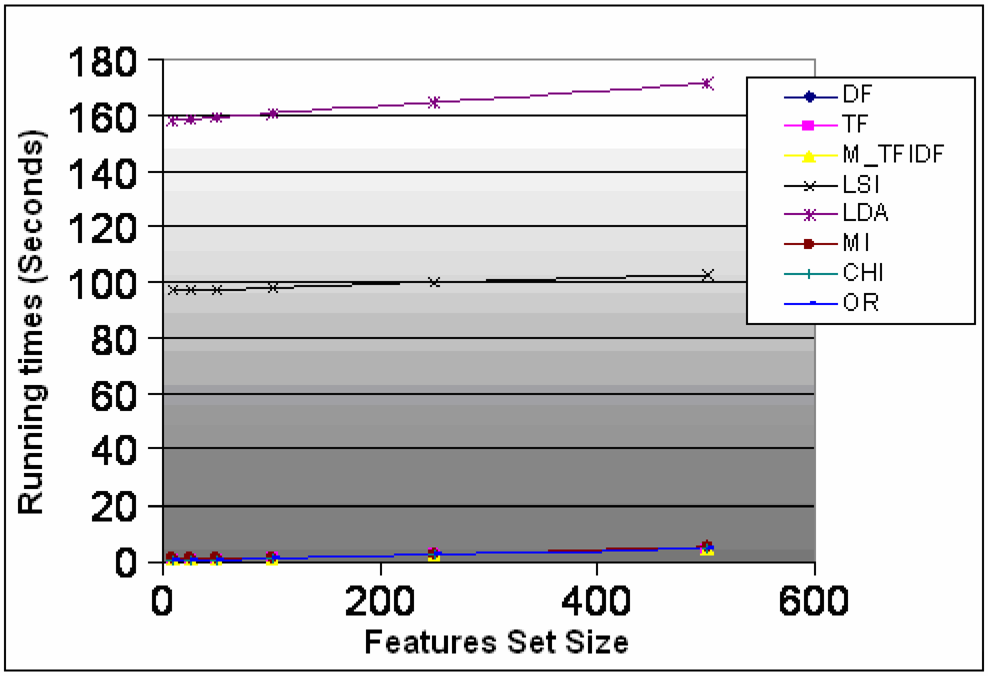


Figure 5-2: Weighted KNN

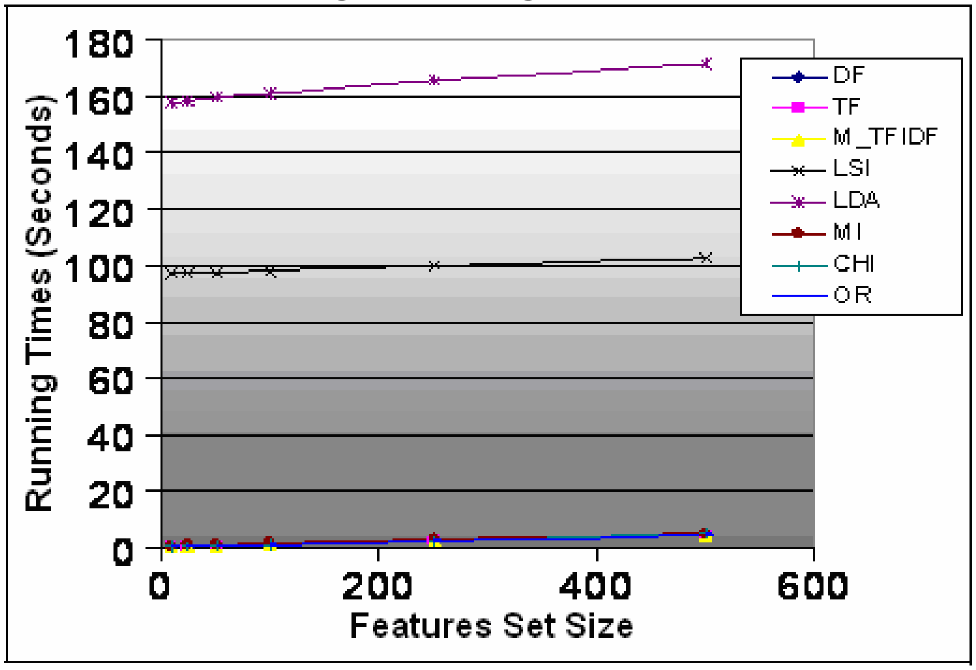
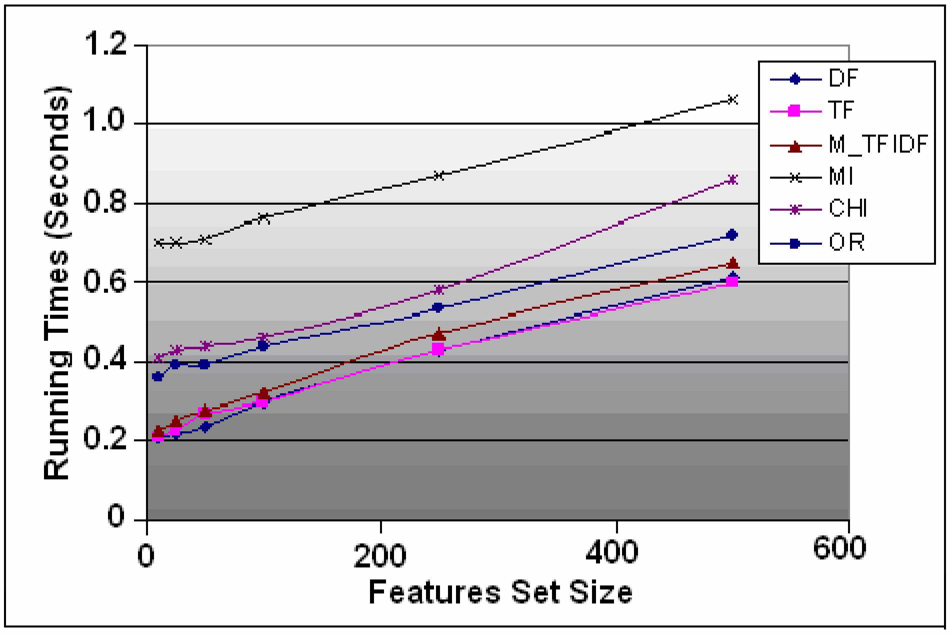


Figure 5-3: Naïve Bayesian



The results can be summarized to the conclusion that LSI and LDA are the most efficient with respect to time while other methods have approximately similar sort of results when we have the size of the document instances in the range of three thousands.

The running times of the preprocessing steps were also of interest. The following table summarizes the results from different steps of the preprocessing.

Table 5-38: Execution Times of Preprocessing Steps

Preprocessing Step	Execution Time in Seconds
words lesser in length than 3	27.34
Alpha Numeric Words	35.31
Stop Words	7023.72
Stemming	571.87

5.7. Discussion

In this section we are going to analyze the results in detail. Furthermore we will also explain the observations that were recorded.

The first thing that can be immediately seen note from the results is the effect of changing the values of k on the results. It can be seen that no matter which classifier we are using, the increase in value of k from 3 to 5 (experiment set 2) or from 1 to 3 (experiment set 1) improves accuracy and SP.

Regarding the features set size and its relationship with accuracy there isn't consistent relationship found. Some of the results were good at lower features set sizes while other techniques have tendency for good results at higher features set sizes. If we keep the reduction technique constant then we might be able to find some relationship but finding a general relationship is almost impossible. The reason for this is that at certain features set size the data representation is very good while at others the representation is not that much good.

The effect of **λ** on accuracy was also worth noticing. The accuracy increase about 1% on average as **λ** increases from 9 to 99. The possible reason is that we have more legitimate examples than spam examples. The accuracy results for features set of size 10 and 25 were quite promising and in most cases better than those provided by that of feature set of size 500 which is great improvement over the original features set. Apart from OR all other techniques achieve their best of SR at features set of size 10. The accuracy of the techniques increases as the data representation changes from TF to TFIDF normalized. IG and MI produce the same sort of results for the spam detection which is two class problem.

One important aspect of CHI and MI is that their corresponding scores can be found out using the Boolean data while LSI works with the original TFIDF normalized data. In order to see the effect of performance on CHI and MI making use of the TFIDF representation we suggested another reduction technique which is combination of Mean TFIDF and CHI and MI. The following are the mathematical description of these techniques.

$$new\_CHI(t_k) = Mean\_TFIDF(t_k) \times CHI(t_k)$$

$$new\_MI(t_k) = Mean\_TFIDF(t_k) \times MI(t_k)$$

Figure 4-1 and 4-2 shows the results of weighted accuracy corresponding to these techniques. It can be seen clearly that results were not improved. After the initial fluctuation in the results the new techniques never over run MI and CHI. It can be concluded that the more detailed and actual statistics of the data i.e. TFIDF normalized does not have significant impact on the performance of MI and CHI. Furthermore by ignoring the dependence between features not only simplifies the computation and complexities of algorithm but also have lesser impact on performance as can be seen from the results of LSI, MI and CHI.

Overall CHI, MI and LSI produces the same sort of results and can be put into the first category while LDA, DF, TF and Mean TFIDF can be put into the second category.

Figure 5-4: Weighted Accuracy for (λ = 9) with Simple k-NN (K = 3)

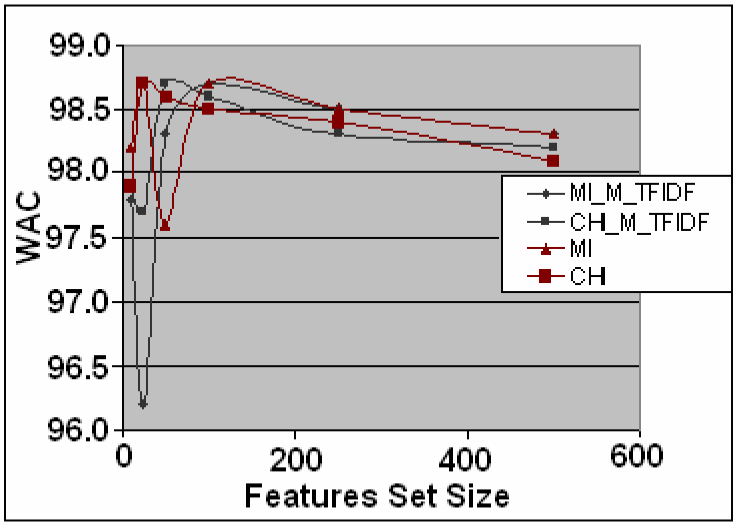
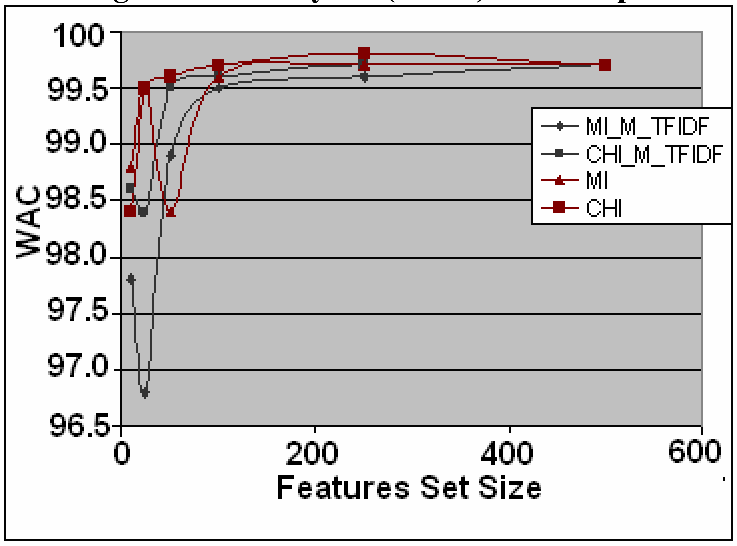


Figure 5-5: Weighted Accuracy for (λ = 99) with Simple k-NN (K = 5)



5.8. Summary

In this chapter the detailed results of the experiments were presented. The results were compiled using three classifiers i.e. K-NN, weighted K-NN and Naïve Bayesian with eight different dimensionality reduction techniques. Evaluation measures used were spam precision, spam recall and accuracy. It has been concluded that naïve Bayesian outperforms its counter parts classifiers and the best dimensionality reduction techniques are LSI, CHI and MI. The combination of probability based techniques with that of TFIDF do not improve the performance.

CHAPTER-6

CONCLUSIONS

6.1. Introduction

This chapter is aimed to provide the key conclusions and future directions for the spam email detection problem. Many observation regarding the changing accuracy as a function of feature set size, change in classifier, change in data representation and changing the values of λ are described in detail. Future directions and problems with the ling spam corpus is also discussed.

6.2. Findings

There were seven major findings of this research

1. The Effect of λ on Accuracy

Since the data set contains about 84% of legitimate emails so correctly identified legitimate emails should always be greater than correctly identifying spams. For this reason increasing the value of λ improves the weighted accuracy results. To study the effects of λ more closely we need to use a data set that contains more percentage of the spams then the one employed in the ling spam corpus.

2. Use of Boolean Weighting

MI and CHI can be computed with the Boolean weighting. Though Boolean weighting which isn't great for classification purpose [16] provides good results for these features reduction methods.

3. Data Representation and Accuracy

The experimental results with data representation of TFIDF normalized weighting were quite better than that of TF weighting.

4. Over all Performance Wise Ranking

The average performance of LSI, MI and CHI over the entire features sets were the best of all with LSI showed better results than MI and CHI at lower features set. OR, Mean TFIDF, TF and DF produced similar sort of results and can be put into the second category.

5. Ranking of classifiers

The best reported classifier was Naïve Bayesian. The second one was simple k-NN and weighted k-NN stood third in the category while LDA was the lowest, performance wise.

6. Best Features Set Size

The question of which features set size is best depends upon the techniques used. The feature extraction based techniques shows good results at lower sizes while others show good results at higher set sizes

7. TFIDF Representation and Performance of Reduction Techniques

There wasn't any relationship found out between the usage of TFIDF normalized data and the performance of the Reduction technique. LDA and Mean TFIDF used TFIDF normalized weighted data and was reported to have lesser performance than MI and CHI which used the Boolean data

6.3. Future Directions

1. Usage of LSI in conjunction with MI and CHI

We have tested LSI with TF, DF and mean TFIDF. Possible improvements in the results are expected to use CHI and MI in conjunction with LSI.

2. Changing the weights of the proposed reduction method

In section 4.8 the new reduction method that were proposed contains equal weights for both of the Mean TFIDF and CHI, MI. Experiments should be conducted by changing the weights of the CHI, MI and Mean TFIDF to see the effect on the performance. We suggest more weights of CHI, MI.

3. Spam rate of the corpus

The corpus spam rate was very low which do not actually represent the current spam rate. More spams should be added to the corpus and many versions should be made available for experimentations to reflect the effect of performance on changing the spam rates

6.4. Summary

We compared eight different feature reduction techniques with few of their combinations on the domain of spam email detection. Only the text of the message body and the subject lines were taken as features. No phrasal or domain specific features were considered. All the techniques showed good results with the data set achieving results of accuracy in the

90’s. The best techniques reported were MI, CHI and LSI with LSI slightly performing better at lower features set making it the ultimate winner. TF, DF, Mean TFIDF and OR showed similar sorts of results and were ranked in the second category. The effect of changing classifier was not conclusive though weighted k-NN showed slightly better results at k equals 5.

REFERENCES

[1]. Ron Bekkerman, “Distributional Clustering Of Words For Text Categorization”, *Masters Thesis*, Israel Institute Of Technology, 2003.
Retrieved from: www.cs.umass.edu/~ronb/papers/thesis.pdf on 06/02/07.

[2]. M.Sahami, S.Dumais, D.Heckerman, and E.Horvitz, “A bayesian approach to filtering junk e-mail”, *In Learning for Text Categorization Papers from the AAAI Workshop*, pages 55–62, 1998.

[3]. I.An-droutsopoulos, J. Koutsias, K.V. Chandrinos, and D. Spyropoulos, “Learning to filter spam email: A comparison of a Naive Bayesian and a memory-based approach”, *In Proceedings of the Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Lyon, France, 1–13, 2000.

[4]. Islam, M. R., Chowdhury, M. U., and Zhou, W, “An Innovative Spam Filtering Model Based on Support Vector Machine”, *In Proceedings of the international Conference on Computational intelligence For Modeling, Control and Automation and international Conference on intelligent Agents, Web Technologies and internet Commerce* Vol-2 (Cimca-Iawtic'06) CIMCA. IEEE Computer Society, Washington, DC, 348-353, 2005.

[5]. X. Carreras and L. Mrquez, “Boosting trees for anti-spam email filtering”, *In Proceedings of RANLP-01, Jth International Conference on Recent Advances in Natural Language Processing*, Tzigov Chark, BG, 2001.

[6]. S Günal, S Ergin, Ö Gerek, “Spam E-mail Recognition by Subspace Analysis”, *International Symposium on Innovations in Intelligent Systems and Applications*, Yýldýz Technical University 2005.

[7]. S Günal, S Ergin, MB Gülmezolu, Ö Gerek, "On Feature Extraction for Spam E-Mail Detection", Page 1. B. Gunsel et al. (Eds.): MRCS 2006, LNCS 4105, pp. 635–642, 2006.

[8]. Fuka, K. and Hanka, R, "Feature Set Reduction for Document Classification Problems", *IJCAI-01 Workshop: Text Learning: Beyond Supervision*, Seattle 2001.

[9]. Yang, Y. Pedersen, J. O, "A comparative study on feature selection in text categorization", *In Proceedings of the 14th International Conference on Machine Learning,* ICML-97. pp. 412-420. 1997.

[10]. K. Torkkola, "Linear Discriminant Analysis in Document Classification", *Proceedings of IEEE ICDM Workshop Text Mining,* 2001.

[11]. ftp://ftp.cs.cornell.edu/pub/smart/ Retrieved on 03/01/07.

[12]. Salton, G., C. Yang, and A. Wong, "A vector-space model for automatic indexing", *Communications of the ACM*, Vol. 18, No. 11, pp. 613–620, 1975.

[13]. Ozgur A, "Supervised and Unsupervised Machine Learning Techniques For Text Document Categorization", *Maters Thesis*, Bogazici University Turkey, 2004.
Retrieved from: www-personal.umich.edu/~ozgur/ozgur-gungor05.pdf.on 04/02/07.

[14]. Porter, M. F., "An algorithm for suffix stripping", Program, Vol. 14, pp. 130–137, 1980.

[15]. Aas, L. and L. Eikvil, "Text categorisation: A survey", 941, Norwegian Computing Center, June 1999.

[16]. Salton, G. and C. Buckley, "Term weighting approaches in automatic text retrieval", i*nformation Processing and Management*, Vol. 24, No. 5, pp. 513–523, 1988.

[17]. D. Michie, D.J. Spiegelhalter, C.C. Taylor, "Machine Learning, Neural and Statistical Classification", 1994. (Book).

[18]. Imola K. Fodor," A survey of dimension reduction techniques", *Technical report*, Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, 2002.

[19]. Carreira-Perpinan, M. A, "Continuous latent variable models for dimensionality reduction and sequential data reconstruction", *PhD thesis*, Department of Computer Science, University of Sheffield, UK, 2001.
Retrived from: http://www.csee.ogi.edu/~miguel/papers/phd-thesis.html on 04/03/07.

[20]. D. W. Scott. "Multivariate Density Estimation. Theory, Practice, and Visualization", *Wiley Series in Probability and Mathematical Statistics,* John Wiley & Sons, New York, London, Sydney, 1992.

[21]. Bellman, R, "Adaptive Control Processes: A Guided Tour", *Princeton University Press,* 1961.

[22]. Koller, D. Sahami, M. "Hierarchically classifying documents using very few words.", *In Proceedings of International Conference on Machine Learning*, pp. 170-178, 1997.

[23]. D. D. Lewis, M Ringuette, "Comparison of two learning algorithms for text categorization". *In Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval.* 1994.

[24]. Spitters, M. "Comparing feature sets for learning text categorization", *Proceedings of RIAO* 2000.

[25]. Gabrilovich, E. Markovitch, S. “Text Categorization with Many Redundant Features: Using Aggressive Feature Selection to Make SVMs Competitive with C4.5” ,*Proceedings of the 21 International Confeence on Machine Learning*, Banff, Canada, 2004.

[26]. Mladenic, D. “Feature subset selection in text learning” In the proceeding of European Conference on Machine Learning pp. 95, 1998.

[27]. csnet.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf retrieved on 25/12/06

[28]. http://en.wikipedia.org/wiki/Karhunen-Lo%C3%A8ve_transform retrieved on 25/12/06

[29]. Fisher, R. A. “ The use of multiple measurements in taxonomic problems.”Annals of Eugenics 7,pp- 179-188, 1936.

[30]. S. Balakrishnama, A. Ganapathiraju. “Linear Discriminant Analysis - A Brief Tutorial”, Institute for Signal and Information Processing Department of Electrical and Computer Engineering Mississippi State University.
Retrieved: lcv.stat.fsu.edu/research/geometrical_representations_of_faces/PAPERS/lda_theory.pdf on 04/01/07.

[31].] P. Graham, “A Plan for Spam”. Webpage
Retrieved from: http://www.paulgraham.com/spam.html on 20/12/06.

[32]. Turk, M and Pentland, A, “Eigenfaces for Recognition”, *Journal of Cognitive Neuroscience*, vol 3, No. 1, 1991.

[33]. http://www.tartarus.org/~martin/PorterStemmer/matlab.txt Retrieved on 20/12/06.