E-mail Filtering System for Nigerian Spam

by

Siti Sarah Mohd Nasiruddin

Dissertation submitted in partial fulfillment of
the requirements for
Bachelor of Technology (Hons)
(Business Information System)

JULY 2005

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

t
LB
2385
.S672
2005

1) Electronic mail systems -- Security measures
2) IT/IS -- Thesis

CERTIFICATION OF APPROVAL

E-Mail Filtering System for Nigerian Spam

By

Siti Sarah Mohd Nasiruddin

(3022)

A project dissertation submitted to the

Information Technology Programme

Universiti Teknologi PETRONAS

In partial fulfillment of the requirement for the

BACHELOR OF TECHNOLOGY (Hons)

(BUSINESS INFORMATION SYSTEM)

Approved by,

[illegible]

....

(Anang Hudaya Muhammad Amin)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

JULY 2005

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have nit been undertaken or done by unspecified sources or persons.

[illegible]

....

SITI SARAH MOHD NASIRUDDIN

ABSTRACT

This project shows about the project details in developing the E-Mail Filtering System specifically in filtering the Nigerian Spam. The main elements in this report consist of introduction, literature review, methodology and result and discussion. The project is developed by focusing on research activities, findings analysis and developing product. This project is developed based on the advancement of Information Technology (IT) system today which is recently growing rapidly. Recent growth in the use of email for communication and the corresponding growth in the volume of email received have made automatic processing of email desirable. Present day solutions to stop spam work by analyzing headers and message text or classifying the mail based on history. This report gives an introduction to machine learning methods for spam filtering especially for Nigerian Spam. An overview of this mail system will fall back on SPAM filters that use "Naive Bayesian Filtering" which is a probabilistic approach to estimate the degree of SPAM.

ACKNOWLEDGEMENTS

Alhamdulillah, praise to Allah SWT as I manage to finish this Final Year Project (FYP). I would like to thank Universiti Teknologi PETRONAS for having given me this wonderful opportunity to involve myself in this final year project (FYP) of such a challenging nature. This project has certainly brought inestimable and invaluable experiences which have never been encountered before. It had become and opportunity for me to enhance my skills in solving problems independently and it has brought chances for me to apply my knowledge and expanding my thought in order to complete this project.

My utmost gratitude and thank goes to my respective supervisor, Mr. Anang Hudaya b. Muhammad Amin for his support, guidance and incessant help given throughout completing this project. I am indebted to my Final Year Project (FYP) for his constant support and encouragement in the study, design and implementation of the project. I thank him for having given generously his time and expertise. His patience and criticism were essential and appreciated. This gratitude also goes to Mr. Mohd Noor Ibrahim and Mr. Nordin Zakaria, FYP Coordinator and the FYP Committee for ensuring this project becomes a successful one.

Finally, I wish to express my deep sense of gratitude to my friends and the non-teaching lecturer who, in one way or the other, helped in the completion of this Final Year Project. Also thanks to my beloved family for their support and encouragement for me to wade through this period. Last but not least, to those who had helped me through thick and thin directly or indirectly, I owe all of you a debt of gratitude beyond measure. Thank you so much and may Allah bless you.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLE

CHAPTER 1
INTRODUCTION

Nowadays, sending and receiving e-mail are common in our daily lives, and people communicate each other all over the world only by one click on the fingertips. There also has been a great deal of interest of late in the problem of automatically detecting and filtering out unsolicited commercial e-mail messages, commonly referred to as *spam* [2]. There are many methods used before in order to filter the spam. This paper describes one method that is using Naïve Bayes Algorithm that is effective in filtering spam. This system will focus on filtering Nigerian Spam, which is now spread all over the world.

1.1 Background of Study

E-mail Filtering System been developed with the support of Naïve Bayes Algorithm, are among the most successful known algorithms for learning to classify text documents. The process of this system with support by Naïve Bayes Algorithm is to classify the common text of Nigerian Spam and find the level of spamicity. In this project, it also comprises the analysis to the Statistical Token Analysis (STA) which analyzing and defining about tokens that will be extracted from the spam e-mails.

There are many methods available in the internet that can filter e-mail from entering the recipient's mailbox. But the Naïve Bayes algorithm is one of the best methods to filter and calculate the level of spamicity based on probabilities. For example, naive Bayes is optimal for learning conjunctions and disjunctions, even though they violate the independence assumption.

Software to combat spam e-mail, until recently, was based on simple keyword filters; if a given term was present in a message's headers and/or body, the mail was labeled as spam. This rapidly became unscalable as each potential spam term had to be manually added to the system, resulting in little time saved for users. Artificial intelligence techniques are an obvious alternative. [9]

Academic Approaches

Zhou et al (2003) proposed using interfaces used in peer-to-peer networks to handle spam. They reported 97 percent accuracy using this technique. This was deemed eyond the scope of this course and will not be investigated here. [9]

From a machine learning perspective, spam filtering is just text categorization, with categories of spam and nonspam (also known as"ham"). Androutsopoulos (2000) reported that Naive Bayes obtained accuracy 1 an order of magnitude higher than Microsoft Outlook's keyword patterns. Drucker et al (1999) compared the performance of Ripper, Rocchio, support vector machines, and boosting. They concluded that either boosting or SVMs were the methods of choice; boosting offered a somewhat higher accuracy but had a much longer training time. [9]

There are some avenues in support vector machines that have not yet been examined for spam, notably transductive SVMs (Joachims, 1999) or "logical" SVMs (Vapnik andWu, 1999). Nonetheless, because of this greater accuracy, it was decided to pursue a boosting-based algorithm. One that has received much attention recently is AdaBoost (Schapire, 2001). Schapire and Singer (2000) showed that in text categorization, AdaBoost can give high levels of stability and accuracy if properly parameterized. Boosting algorithms work over another base algorithm, called the weak learner. The slow performance of Drucker et al (1999) is at least partly due to the choice of C4.5 as the weak learner; though it may have improved accuracy, this is not necessary for a weak learner, not at the cost of performance. Carreras and Marquez (2000) used boosting with simple decision stumps - decision trees with only a single node, with the presence or absence of a single term as a predicate - and were able to obtain with that significantly better accuracy than with Naive Bayes. Counter intuitively, using decision trees up to five levels deep obtained only marginally higher accuracy than stumps, with a performance penalty. [9]

It was therefore hoped that AdaBoost, with decision stumps as its base learner, could be used as the basis for a spam filter. [9]

1.2 Problem Statement

1.2.1 Problem Identification

Author has done research on the internet, which states that advance fee fraud, often also known as the Nigerian money transfer fraud, Nigeria spam or 419 spams after the relevant section of the Nigerian Criminal Code it violates, is a fraudulent scheme to extract money from investors living in rich countries in Europe, Australia, or North America. Although these confidence tricks originated in Nigeria, they have since become a worldwide criminal activity that is conservatively estimated to net billions of dollars a year.

Addresses of recipients are often harvested from Usenet postings or web pages, obtained from databases, or simply guessed by using common names and domains. Spam is sent without permission of the recipients. Constraints about studying the relationship of the Naïve Bayes Algorithm with the system developed [1].

1.2.2 Significant of the Project

The purpose of this project being done is to demonstrate the filtering features of Nigerian Spam. With the program applied by Naïve Bayes Algorithm, the filtering can be done and calculate the level of spamicity so the spam e-mail can be identified and subsequently can be filtered before it goes to the mailbox of the recipient.

1.3 Objectives & Scope of Study

1.3.1 Objective(s)

In order to keep the project more focus and meet the system and user requirements, the author has outlined some objectives that need to be achieved at the end of completing this project. The concentration is more towards the development of the system design in order to produce the high quality final product.

The project objectives are as below:

- To develop an application (filtering system) to detect the Nigerian Spam those enters into the mailing inbox.
- To study Naïve Bayes Algorithm that has been applied in the system database to calculate the level of spamicity.
- To understand the Java Code in Java Forte application which is a tool that develops the system, so that the functionality, method and uses of the codes can be understood.
- To learn various methods in filtering spam e-mails.
- To demonstrate the implementation of e-mail filtering specifically for Nigerian Spam.

Therefore, in order for the objective to be successful, the Naïve Bayes Algorithm that acts as a database of keywords will have to be understood and be applied to the system for classifying the text and calculate the level of spamicity of the Nigerian Spam.

1.3.2 Scope of Study

Technical feasibility is one of the important processes during feasibility studies since the results can enhance the author's understanding and knowledge to construct a proposed system. During this process, the author tried to understand of the possible target hardware, software, project's complexity, experiences with similar system and the environment of the system.

From the author side or research findings, it is towards research on the application of Naïve Bayes Algorithm and understands how it is going to be implemented in the system to filter the Nigerian Spam that entered to the recipient inbox. The analyzing process is more on the application of the algorithm in the system to find the level of spamicity. The author could say that this project is quite challenging and risky since the author is not familiar with several concepts such as Bayesian filtering and also tokenization of strings. As the result, the author tried to gain some knowledge regarding those topics and also discussed with supervisor, Mr. Anang Hudaya Muhammad Amin. In addition, the author tried to study the software requirements, hardware and programming language in Java that related to the project.

The author believed that this research findings and study will help in order to avoid certain challenges and risk such as wrong interpretation of the algorithm itself, misconduct in developing the system project overdue and failure to determine programming languages as well as procedures.

The scope of study are more abroad as the research will be focusing on how to break down the strings into tokens, calculate the ham and spam frequencies, the total frequencies of the spam, and also how the Bayesian algorithm will be applied to the system. On the designing part in Java Forte, the study will comprise on how the functionality of the buttons, fields and coding in order to classify e-mails as spam or non-spam e-mail. The author tried to obtain the output from those sources and modify the coding in the implementation phase.

CHAPTER 2
LITERATURE REVIEW and/or THEORY

Based on the introduction, problem statement and scope of study above, this literature review will give more understanding of my project as well as my project scope. Some of the sources for this chapter are found in the internet which demonstrates the concept of Naïve Bayes Algorithm, tokenization as well as database.

From the literature review on the Internet, the author came to know that there were attempts to create SPAM Filters by purely using Bayesian Filtering Technique [2]. But after all the Bayesian System just relies on probabilistic means and is never hundred percent accurate [2]. To have a clear understanding of the Bayesian System, one has to have a clear idea of conditional probability and Bayes Theorem [2].

Before further explanation of the research findings on Naïve Bayes Algorithm, the author has found some explanation and definition of Bayesian Filtering in the internet.

Bayesian filtering is a technique for identifying incoming e-mail spam. Unlike other filtering techniques that look for spam-identifying words in subject lines and headers a Bayesian filter uses the entire context of an e-mail when it looks for words or character strings that will identify the e-mail as spam. Another difference between a Bayesian filter and other content filters is that a Bayesian filter learns to identify new spam the more it analyzes incoming e-mails. [11]
Bayesian filtering is named for English mathematician Thomas Bayes, who developed a theory of probability inference. Bayesian filtering is predicated on the idea that spam can be filtered out based on the probability that certain words will correctly identify a

piece of e-mail as spam while other words will correctly identify a piece of e-mail as legitimate and wanted. At its most basic level, a Bayesian filter examines a set of e-mails that are known to be spam and a set of e-mails that are known to be legitimate and compares the content in both e-mails in order to build a database of words that will, according to probability, identify, or predict, future e-mails as spam or not. Bayesian filters examine the words in a body of an e-mail, its header information and metadata, word pairs and phrases and even HTML code that can identify, for example, certain colors that can indicate a spam e-mail. [11]

Bayesian filters are adaptable in that the filter can train itself to identify new patterns of spam and can be adapted by the human user to adjust to the user's specific parameters for identifying spam. Bayesian filters also are advantageous because they take the whole context of a message into consideration. For example, not every e-mail with the word "cash" in it is spam, so the filter identifies the probability of an e-mail with the word "cash" being spam based on what other content is in the e-mail. [11]

Proponents of Bayesian filters assert that the filters return less than one percent of false positives. [11]

How do Bayesian anti-spam products work?

All Bayesian spam or junk e-mail filters generally have two things in common: (1) Bayesian filters use previous examples of actual e-mail and spam messages to classify new mail and (2) Bayesian statistics are applied to observations to calculate probabilities. Bayesian junk e-mail filters estimate the likelihood that a message should or should not be blocked based on a wide range of content. This approach differs from rule or list based systems that block e-mail messages that contain a certain combination of stop words or are from a "blacklist" of bad domains. [10]

Almost all Bayesian spam and junk e-mail filters sample and categorize actual e-mail messages. For example, the messages might be sorted into a "spam" category (junk e-mails) and a "ham" category (good e-mails). The Bayesian paradigm does not define how the samples are to be collected, where the messages might come from, or how they are to be categorized. Each company can use its own methodology. For example, some companies may collect their own messages and will build models from them. Other companies may collect information from the user's e-mail folders. [10]

One next step is to statistically analyze the content of the message to identify significant or meaningful characteristics. For example, certain words may be used frequently in good messages, but rarely in junk e-mail messages. (The names of family members or your company's products often are strong good e-mail indicators.) [10]

The key to this step is a process known as tokenization. A token is the smallest unit for which a statistic is collected. A word may be a token. But, some companies may also identify punctuation marks, phrases, e-mail addresses and invisible contents as tokens. In some cases, a word in the body of an e-mail and the same word in the subject line in an email will be considered separate tokens. Tokens may also be weighted to give extra importance to certain types of tokens. Much of the quality of a Bayesian junk e-mail filter and the proper identification of spam will rest on the skill of the designer of the tokenization process. [10]

Each token is assigned a mathematical value based on how strongly it indicates whether a given message is a good message or junk e-mail. A straightforward mathematical formula is used to evaluate all of the statistics (or just the most significant statistics) to produce a probability estimate that a message is either good or junk e-mail. The result is not a categorization, but rather the probability that it is either good or junk. For example, process might compute that the probability that a message is spam is .80 (80%), based on its content. [10]

A characteristic of many Bayesian systems is that there can be a "maybe" category. It is possible to calculate that a given message has a .50 (50%) probability of being junk because it has both strong positive and negative indicators. The identification as “maybe” is under the user’s control, based on the observed probabilities. [10]

Once a message is reviewed and identified as good or junk e-mail, its contents are added to the statistics that make future evaluations more accurate. Many Bayesian junk e-mail and spam filters will allow the user to adjust the threshold for good, junk, and maybe messages. Proper thresholds will minimize mistakes and false alarms. [10]

2.1 Bayes Theorem

Bayes Theorem gives us a method of obtaining the P (A/B) when we are provided with P (B/A) [1].

$$P(H \mid E, c) = \frac{P(H \mid c)\ P(E \mid H, c)}{P(E \mid c)}$$

Figure 2.1 Bayes Theorem

This simple but extremely useful Theorem was derived by Rev. Thomas Bayes and was published in the year 1763 [2].

2.2 Role of Bayes Theorem in Filtering

A Bayesian network is a directed, acyclic graph that compactly represents a probability distribution. Given specific instant x (an assignment of values x1, ..., xn), the Bayesian network allows us to compute the probability P(C=ck | X=x) for each possible class ck [2].

$$P(C = c_k \mid \mathbf{X} = \mathbf{x}) = \frac{P(\mathbf{X} = \mathbf{x} \mid C = c_k) P(C = c_k)}{P(\mathbf{X} = \mathbf{x})}$$

Figure 2.2 Probability 1

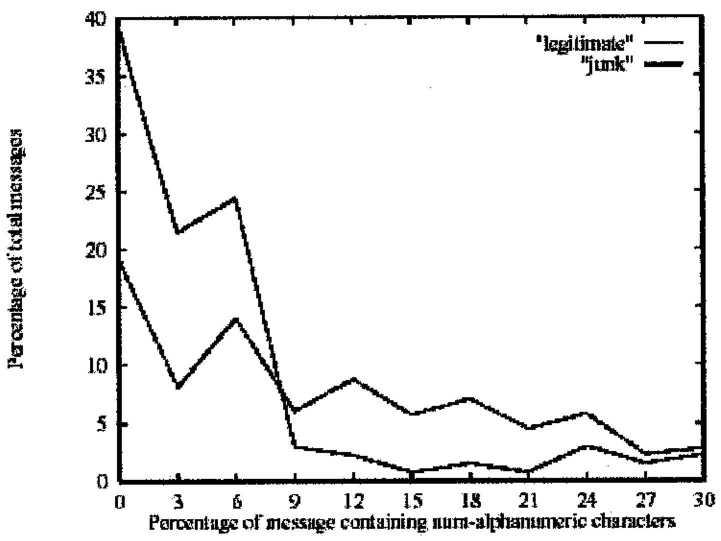
Each feature Xi is conditionally independent of every other feature, given class variable C [2].

$$P(\mathbf{X} = \mathbf{x} \mid C = c_k) = \prod_i P(X_i = x_i \mid C = c_k)$$

Figure 2.3 Probability 2

In the context of text classification, especially junk E-mail filtering; it becomes necessary to represent mail messages as feature vectors so as to make such Bayesian classification method directly applicable. Each dimension of this space corresponding to a given word in the entire corpus of message is seen. Each individual message can then be represented as a binary vector denoting which words are present and absent in the message. With this representation, it becomes straightforward to learn a probabilistic classifier to detect junk mail given a pre-classified set of training message [2].

There are many particular features of E-mail beside just the individual words in the text of a message that provide evidence as to whether a message is junk or not. Examples: phrase "Free Money", over emphasize "!!!!", non-textual feature such as domain type of the message sender. Such additional 10 features can be incorporated into the Bayesian classifiers by adding additional variables denoting the presence or absence of these features into the vector for each message. In this way, various types of evidence about messages can be uniformly incorporated into the classification models and learning algorithms employed need not be modified [2].



Percentages of legitimate and junk E-mail with subjects comprised of varying degrees of non-alphanumeric characters

Figure 2.4 Percentage of legitimate and junk E-Mail

The above graph shows the statistics been obtained from a previous study conducted by Mehran Sahami, Susan Dumais, David Heckerman, Eric Horvitz at the University of Wisconsin Madison [2].

Based on author's findings in the Internet, series of tests perform the Bayesian statistical computations to arrive at the spam probability index called spamicity. By default there are three rules that combine to provide the Bayesian filtering in Praetor. These are explicitly listed for when the ADVANCED filtering mode is set, and implicitly performed if the selection is for the BASIC mode with the pre-configured rule filters. There are three explicit rules that test the Bayesian spamicity value. [5]. Table below shows the three explicit rules:

Spamicity < 0.30

Purpose:	Check the Bayesian spamicity to see if it is less than 0.30 (default).
Action:	**Accept** if the value is less than this good message threshold level.
Default state:	Enabled
False Positive:	No, but it can allow spam to be accepted which represents the false negatives.
Other notes:	

Spamicity > 0.60

Purpose:	Check the Bayesian spamicity to see if it is greater than 0.60 (default).
Action:	**Quarantine** if the value is greater than this bad message (spam) threshold level.
Default state:	Enabled
False Positive:	Low incidence of non-spam messages being rated as spam.
Other notes:	Once you feel confident that this 0.60 threshold has no visible false positives, you may want to change the action to reject.

Spamicity is unsure

Purpose:	Check the Bayesian spamicity to see if it is in the range between 0.30 and 0.60 (default).
Action:	**Accept** if the value is in this unsure range.
Default state:	Enabled
False Positive:	No false positives, but likely to be the source of some false negatives which are spam that is not caught.
Other notes:	If the threshold values are adjusted to your satisfaction, it may be better to change the action to **Quarantine** and deal with the fewer false positives.

Table 1: Three explicit rules that test the Bayesian spamicity value

With the new technologies that we have in the world now, there are many Spam that the user has been receives in their mailbox which lead to problem and also the inbox storage would be always full. For effective Spam control, a new approach is needed. Spammers are innovative constantly coming up with new ideas to defeat Spam controls, so some innovative solutions are needed to combat the problem [6].

Two new technologies that are producing some extremely effective results are Distributed Checksum Clearinghouse (DCC) and Statistical Token Analysis (STA). Both of these technologies require little administration and maintenance and are achieving excellent results by blocking close to 100% of Spam with a very low false positive rate [5]. But for this project, we will only be focusing on filtering Spam using Statistical Token Analysis (STA), which will classify those texts and categorize it in the table.

Below is some explanation about Statistical Token Analysis (STA). Statistical Token Analysis [Graham 2002] detects Spam by analyzing the words and tokens of e-mail messages using frequency analysis and statistical comparison to determine if a message is Spam or a legitimate e-mail. STA's operation is based on maintaining two frequency tables, one for Spam and one for Ham (ham is the colloquial name for legitimate e-mail). These tables are generated by analyzing large numbers of known Spam and Ham messages. The tables list all tokens (words or sequences of characters separated by spaces) with the frequency that these tokens occur in the analyzed Spam or Ham (the Spam table records only the Spam frequency, the Ham table records the Ham frequency) [6]. Basically, the above explanation that is about STA is one of the processes that is involved in segregating all those words in the Spam.

Below figure shows some simplified examples of Spam and Ham frequency tables:

Spam Frequency Table	
debt	80
elimination	79
paycheck	79
scary	75
color=red>	74

Ham Frequency Table	
website	65
order	64
invoice	59
proposal	51
update	43

Table 2: Example of spam and ham frequency table

To analyze a message, it is split into tokens and each token is looked up in both the Spam and Ham frequency tables. A statistical analysis of the set of tokens with the highest Spam frequency score and of the (different) set with the highest Ham frequency score then produces a probability that the message is Spam. The measured probability ranges from 100 (certain to be Spam as all analyzed tokens appear in the Spam table and none appear in the Ham table) to 1 (certain to be Ham). Experience running STA shows that the majority of messages classify strongly as either Spam or Ham with relative few in the mid range of probability. [6]

As an example, the message shown in **Figure 2.6** would classify strongly as Spam as may of its tokens have high frequency scores in the Spam table while none score highly in the Ham table. [6]

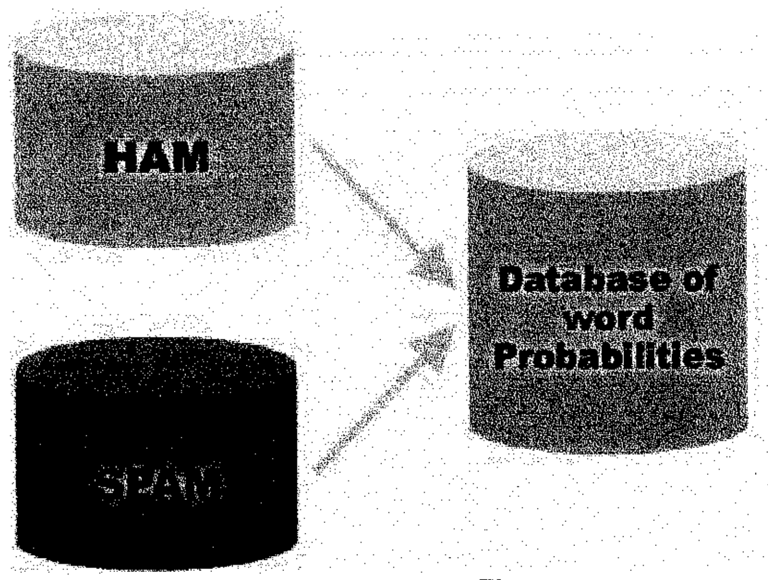
```
We know how scary debt can be.

Our Debt Elimination program can help you become debt free
quickly. Stop harrassing calls from collectors, stop living
from paycheck to  paycheck, and begin to build a stress-free
future.
```

Figure 2.5 Example STA Input

The statistical filter starts the dissection by breaking the message content into a list of unique tokens. A token is a word or any string of identifiable characters, such as dollar amounts and HTML tags. Once a complete list of tokens has been generated for a message, the list will be analyzed. The analysis consists of comparing tokens found in the message to the user and general datasets. The user is searched first, and the general data is searched only when a token cannot be found in the user data. If a token from the message is found in either dataset, the probability score for that token is noted. The probability score for a token indicates the probability that a message containing that token is spam. The overall probability is a statistical score that is calculated over the entire email message. One a complete list of probabilities has been compiled; the probabilities are used to calculate the message spam score. This score reflects the overall probability that the message is spam [7]. This explains on the server end, on how the system which is spam filter works.

Refer to the author in http://www.paulgraham.com/spa,.html, they treat mail as spam if the algorithm above gives it a probability of more than .9 being spam. But in practice it would not matter much where they put this threshold, because few probabilities end up in the middle of the range. Also according to the author of all the approaches to fighting spam, from software to laws, the author believes Bayesian filtering will be the single most effective. But the author also thinks that the more different kinds of anti spam efforts we undertake, the better, because any measure that constraints spammers will tend to make filtering easier. And even within the world of content-based filtering, it will be a good thing if there are many different kinds of software being used simultaneously. The more different filters there are, the harder it will be for spammers to tune spasm to get through them [8]. So what have been understood was, the Bayesian is the effective way and exact approach in fighting spam with support the systematic conduct of e-mail filtering.

The author also found some research on the database that involves in the spam filtering. Before mail can be filtered using this method, the user needs to generate a database with words and tokens (such as the $ sign, IP addresses and domains, and so on), collected from a sample of spam mail and valid mail (referred to as 'ham'). [12]



Creating a word database for the filter

Figure 2.6 Figure of Spam/Ham Database

A probability value is then assigned to each word or token; the probability is based on calculations that take into account how often that word occurs in spam as opposed to legitimate mail (ham). This is done by analyzing the users' outbound mail and by analyzing known spam: All the words and tokens in both pools of mail are analyzed to generate the probability that a particular word points to the mail being spam. This word probability is calculated as follows: If the word "mortgage" occurs in 400 of 3,000 spam mails and in 5 out of 300 legitimate emails, for example, then its spam probability would be 0.8889 (that is, [400/3000] divided by [5/300 + 400/3000]). [12]

Once the ham and spam databases have been created, the word probabilities can be calculated and the filter is ready for use. When a new mail arrives, it is broken down into words and the most relevant words – i.e., those that are most significant in identifying whether the mail is spam or not – are singled out. From these words, the Bayesian filter calculates the probability of the new message being spam or not. If the probability is greater than a threshold, say 0.9, and then the message is classified as spam. [12]

CHAPTER 3
METHODOLOGY/PROJECT WORK

In the earlier phase of this project, the author has decided to use waterfall methodology as the main guidance. Based on the author findings and, a project involves working with a tool to produce documents and system, with the guidance of the requirement, objectives and also strategies

One of the important aspects that determine a good and successful project is to choose the right methodology. For this project, the author has chosen the waterfall methodology as guidance for her to complete this project. For this project, the author decides to come out with her own methodology that suits with the project progression.

3.1 WATERFALL MODEL

The methodology used in this project would be the Waterfall Model.

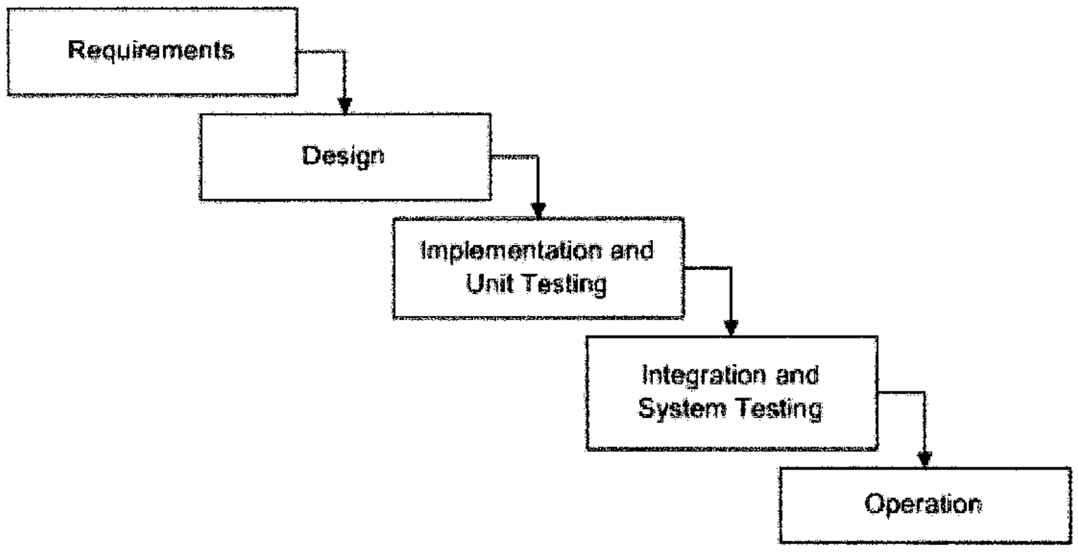

Requirements
Design
Implementation and Unit Testing
Integration and System Testing
Operation

Figure 3.1: Waterfall Model

This is the most common and classic of life cycle models, also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed in its entirety before the next phase can begin. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. Unlike what I mentioned in the general model, phases do not overlap in a waterfall model [4].

Requirements

In this phase, all the requirements are gathered. It is a matter of investigation, scoping and definition of a new system. Requirements analysis is an important, where developer identify the needs or requirements of a client. Subsequently identified the requirements and in the position to the design of solution [4].

Design

System design will be produced from the results of the requirements phase. Developer will have to think about the interface in this phase in which there focus lies. Architecture, including hardware and software, communication, software design is all part of the deliverables of a design phase [4].

Implementation

Code is produced from the deliverables of the design phase during implementation, and this is the longest phase of the software development life cycle. For a developer, this is the main focus of the life cycle because this is where the code is produced. Implementation may overlap with both the design and testing phases [4].

Testing

During testing, the implementation is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. Unit tests and system/acceptance tests are done during this phase. Unit tests act on a specific component of the system, while system tests act on the system as a whole [4].

For this project, the author comes out with her own methodology that adopts some features in waterfall methodology. These features are based on the concepts that have been discussed earlier which is requirements, design, implementation, integration and system testing and operation.

3.2 E-Mail Filtering System for Nigerian Spam Methodology

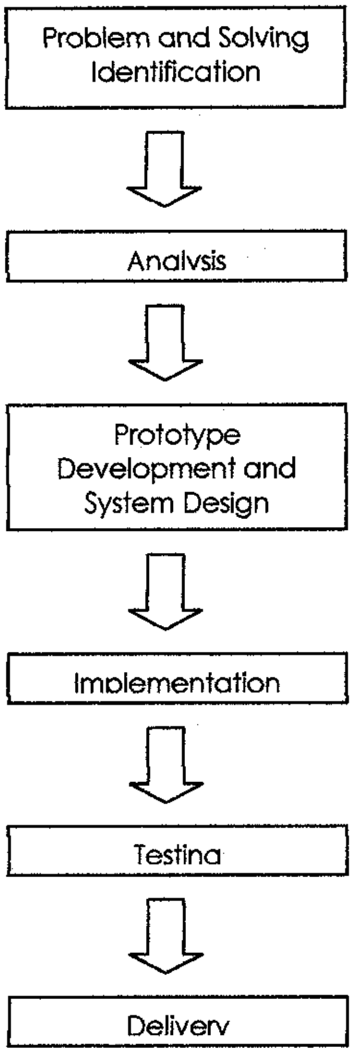

Problem and Solving Identification
Analysis
Prototype Development and System Design
Implementation
Testing
Delivery

Figure 3.2: New Methodology for E-Mail Filtering for Nigerian Spam

From the new methodology, there are aspects that need to be considered in each phase of the methodology. The explanations are described as below:-

Problem and Solving Identification

In the first stage, the author identifies the problem statement in developing this project as well as the problem in the real world. The solution also been identified on how to develop the system and also can the system will solve the problem that occurs currently. This phase also can be considered as a requirement phase where the author does some part of investigation, scoping and definition of the new system as well as the environment. When requirement are identified it will subsequently identified the design of the solution.

Analysis

In this second stage, the author worked with analysis that is based on resources such as internet, books, journals and literature review. The analysis is about the algorithm that is used in the system, the method and flow process of the system, the tools needed to develop the system as well as research on the project. The analysis/research is basically based on the requirement that has been stated in the **Problem and Solving Identification** phase.

Prototype Development and System Design

In the third stage, the prototype development and system design will be produced based on the requirement that has been stated in the first stage. The author plans and sketches the storyboard of the interface, the attributes that involve in the system development, the functionality each of it, decide how the result would be in the real environment. In this stage, the requirement changes only when it is absolutely necessary to the success of the product. This stage is considered and important phase, where there are many important aspects that need to be considering in order developing the system. The design is basically begins with the storyboard sketching, define system functionality, the impact of the system to the real environment as well to the user.

Implementation

In the fourth stage, the development and architecture are frozen and placed under change order control. Means that there is no more architectural changes are allowed unless they are absolutely necessary. During this stage, the author is more focusing on the implementation and the system quality before it is finalized to the final product. After the implementation phase has been done, the system can be only being tested in the **Testing** Phase.

Testing

In the fifth stage, the implementation is tested against the requirement to make sure it is sure that the product is actually solving the needs addressed and gathered during the **Problem and Solving Phase.** During this stage, the system performance and system functionality tests are done in this phase. It is to make sure that the system run smoothly and also the functionality of the system is very well and also it gives impact and meet requirement as stated in the early stage.

Delivery

The final stage is the final product delivery where the implementation stage is completely frozen and only focusing on the quality of the product. The important matter in this stage is to make the final product in high quality which is ready to be used and no critical errors that are allowed in this final product delivery.

3.3 Tools and Other Documents

Besides methodology, the tools and documents that are used throughout this project also been identified. The list of tools and documents used are described as below:

1. Tools

- ❖ Sun One Studio 4, Java Forte and Java 2SDK 1.4.2_03 used in Windows XP Professional / Windows Millennium environment.
- ❖ Sources from World Wide Web documents, books, journals from library.
- ❖ Microsoft Office such as Microsoft Word as document processors and Microsoft Access to built the database.

2. Documents

- ❖ Requirement document that defines the problem of the project. This document also identifies the hardware, software and system requirement.
- ❖ The design of architecture (diagrams), flow process and implementation plan.

3. Coding

- ❖ Object, attributes, data structures, Java Codes that will be used in Java Forte.
- ❖ Naïve Bayes Algorithm that will be applied in the system database to calculate the level of spamicity.

CHAPTER 4
RESULTS & DISCUSSION

4.1 E-MAIL FILTERING SYSTEM FOR NIGERIAN SPAM

For ‘E-Mail Filtering System for Nigerian Spam, there is levels of filtering or can be called as the flow process which can be illustrated as below:

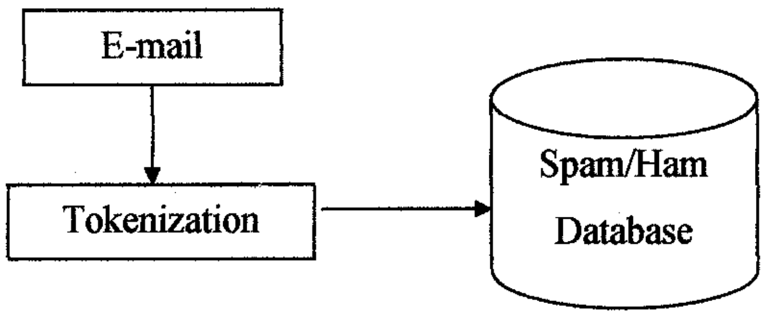


Figure 4.1 Flow Process of E-Mail Filtering for Nigerian Spam

The filtering process begins when the spam e-mails or non-spam e-mail are entered into the system in order to be filtered. The system continues when it tokenize the strings in the e-mail of non-spam e-mails and broke down the strings into words or tokens. The system will first tokenize all the string into tokens (words) and also calculate the frequencies of each word in the sentences or strings which in the other words, calculates how often that word occurs in spam as opposed to legitimate mail (ham).

This process occurs when **"Tokenize"** button are clicked. Tokenization is one of the main processes that involved in this project. For this project, author has decided to do tokenize from strings into tokens (words) by applying Java Codes which is **StringTokenizer** method. This is really required because it will ease the process of comparing those tokens with the existing tokens in the database which is in Microsoft Access.

The main concern from the result of tokenize the strings is how many elements that consists in the strings or sentences and how many frequencies of each tokens (words) in the e-mail. This frequencies that are compared in the databases and there are some conditions that the system can classify the e-mails is spam or not spam. To identify the e-mail is spam or not, Naïve Bayes Algorithm will be applied which it can calculate the level of spamicity based on probabilities. Once the Bayesian filter has the list of tokens in the message, it searches the spam and non-spam token databases for these tokens. These databases of tokens are created and updated whenever the Bayesian filter is "trained" on a new message.

4.2 E-Mail Filtering System for Nigerian Spam Interface

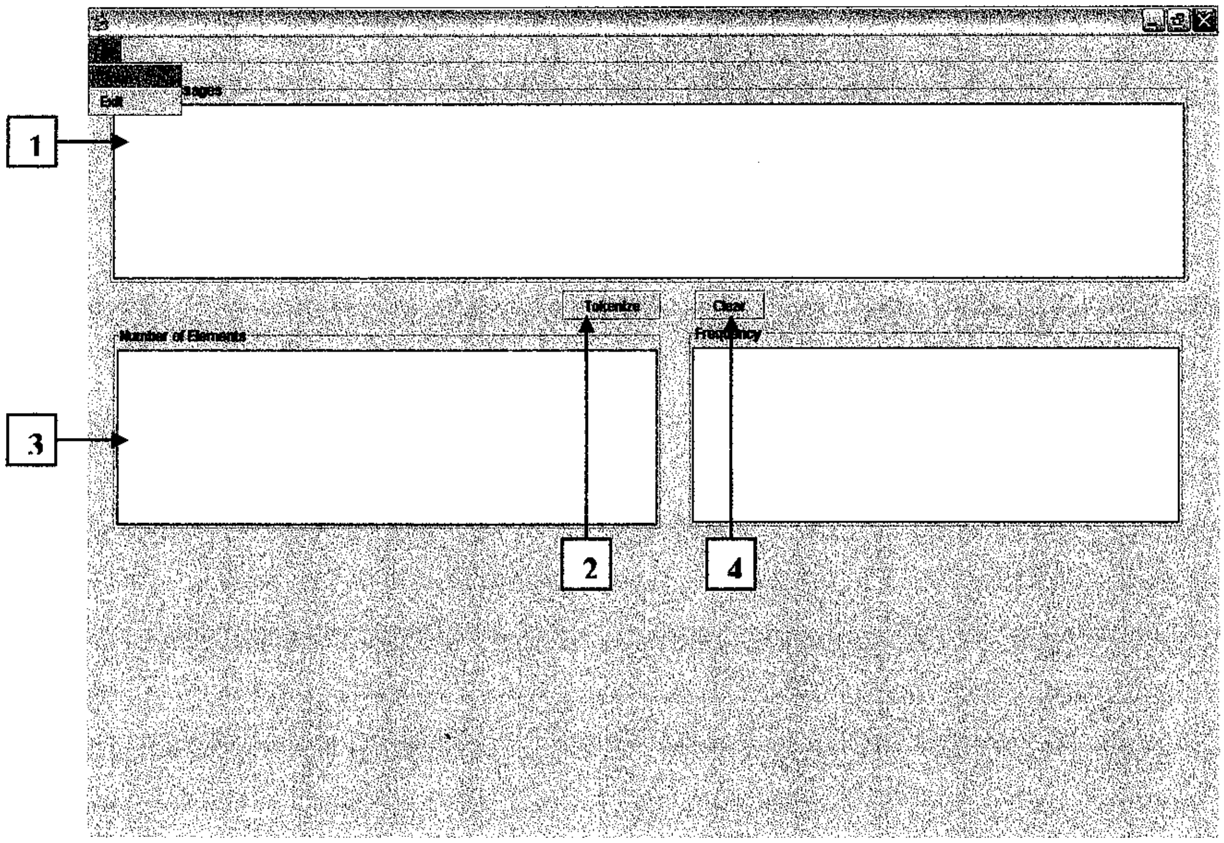
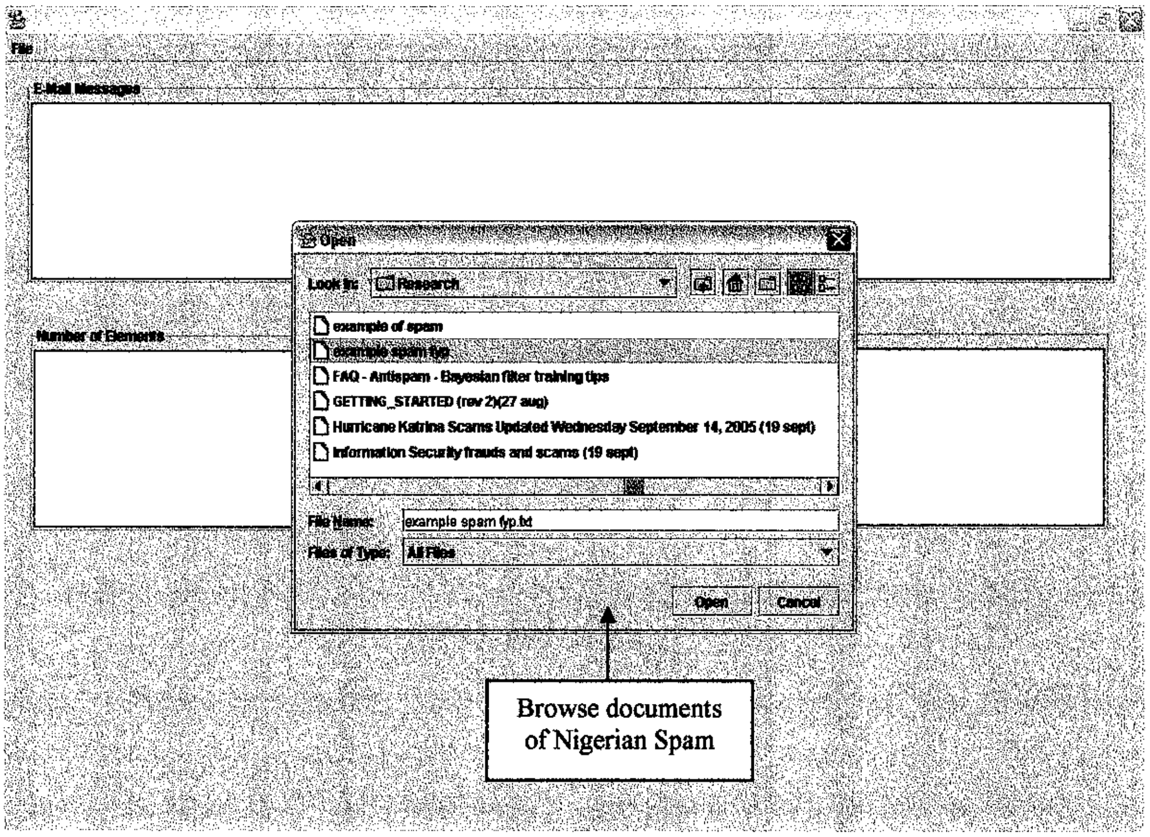


Figure 4.2 System Interface

1. This input text area is the input of the e-mail been open by the user.
2. Clicking on "Tokenize" button will allow the tokenization process, where the string in the sentences in the e-mail at **E-Mail Messages** will be extracted and broke down into tokens.
3. This **Number of Elements** output text area contains the output or the tokenize words from the entered e-mails after clicking "Tokenize" button as explained above.
4. Clicking on "Clear" button will clear the **E-Mail Messages** text area.

5. Clicking on the **File** menu will display **Open** to open documents of Nigerian Spam and **Exit** to exit the system.

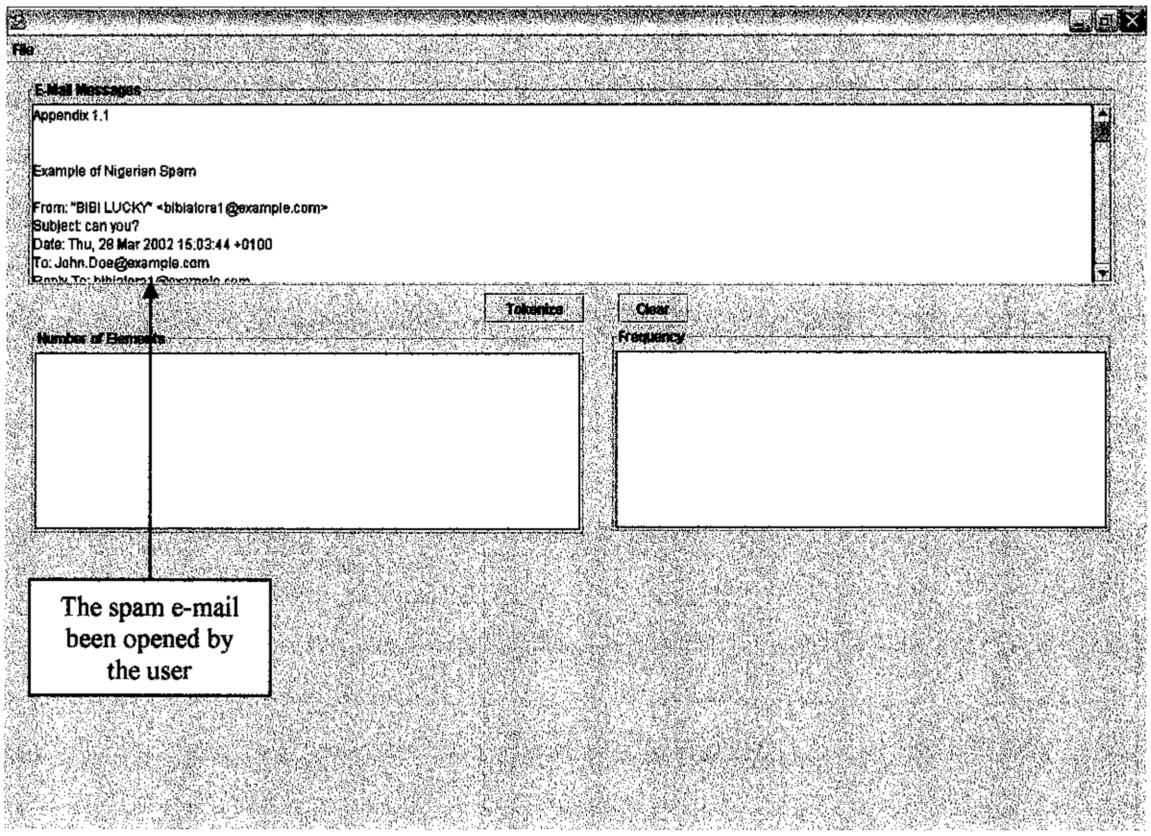


example of spam
example spam fyp
FAQ - Antispam - Bayesian filter training tips
GETTING_STARTED (rev 2)(27 aug)
Hurricane Katrina Scams Updated Wednesday September 14, 2005 (19 sept)
Information Security frauds and scams (19 sept)

File Name: example spam fyp.txt
Files of Type: All Files

Figure 4.3 System interface with functionality 1

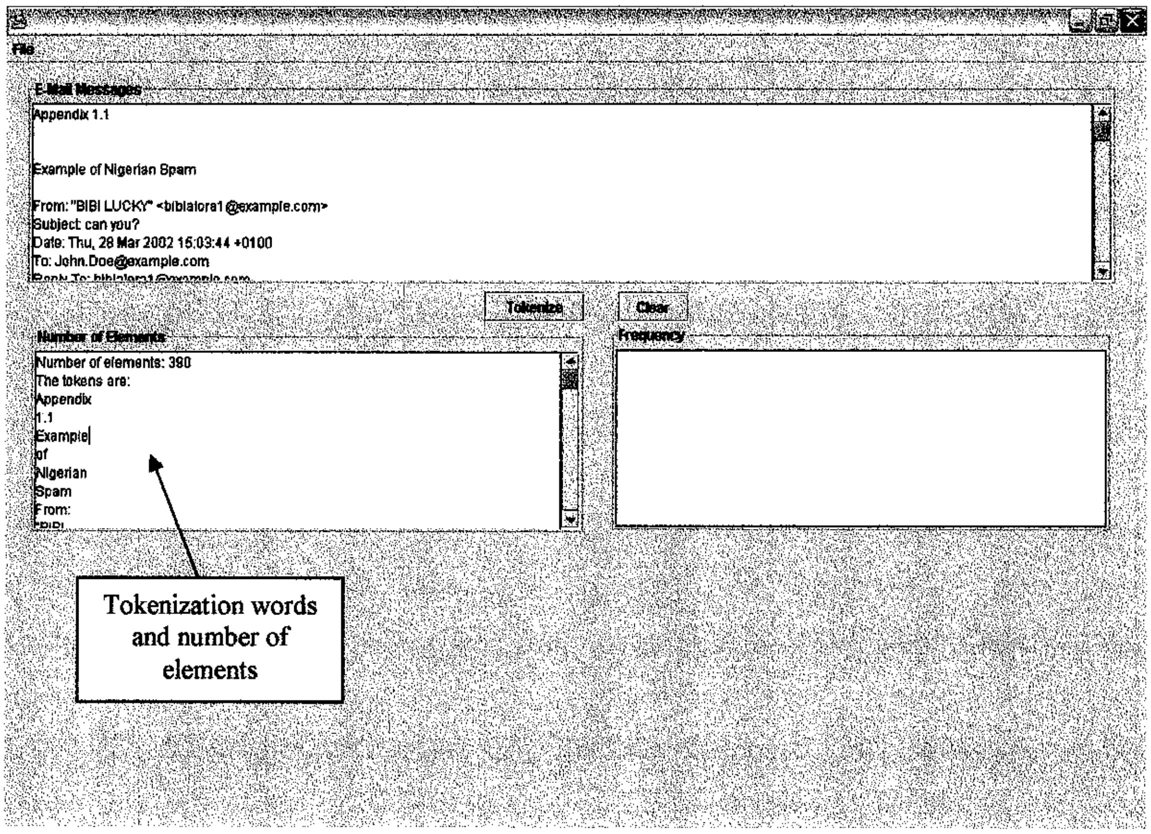
When the user click on File menu and choose Open, it will display the Open window browser enable the user to browse file or documents of Nigerian Spam in the computer local directory.



```
Appendix 1.1


Example of Nigerian Spam

From: "BIBI LUCKY" <bibiaiora1@example.com>
Subject: can you?
Date: Thu, 28 Mar 2002 15:03:44 +0100
To: John.Doe@example.com
Reply-To: bibiaiora1@example.com
```

Figure 4.4 System interface with functionality 2

When the document has been opened by the user using the **Open** menu, it will display on the **E-Mail Messages** text area.



```
Appendix 1.1


Example of Nigerian Spam

From: "BIBI LUCKY" <biblalora1@example.com>
Subject can you?
Date: Thu, 28 Mar 2002 15:03:44 +0100
To: John.Doe@example.com
```

```
Number of elements: 380
The tokens are:
Appendix
1.1
Example
of
Nigerian
Spam
From:
```

Figure 4.5 System interface with functionality 3

When clicked on the **Tokenize** button, it will tokenize the sentences in the **E-Mail Messages** to words/tokens and it will be display on the **Number of Elements** text area and also calculate the number of elements in the sentences.

4.3 Conclusions And Discussion

For this system, there are several functions need to be focused on the future upgrading and maintenance.

- The final product should be able to calculate the frequencies/occurrences of the tokenize words in the Nigerian Spam.
- The system should be able to calculate the level of spamicity of more than one e-mail in one real time, means that assuming that the e-mail enters into the system to be tokenize is more than one.
- The database should be able to store the words/tokens every time it has been trained on when the e-mails are been tokenize by the system.

The idea of e-mail filtering is one of the alternatives to combat fee fraud that widely spread all over the world. Using the Bayesian Algorithm makes the system easier to detect and calculate the level of spamicity and identify it as a spam email or not.

CHAPTER 5
CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

Bayesian filtering goes well with the e-mail filtering system application where in calculating the level of spamicity. The objective of e-mail filtering system is to identify the spam especially Nigerian spam that enters in the recipient inbox and act as middle application to reduce the fee fraud to be happen again and again. It is recommended to have the system in the mailing system now in order to reduce the advance fee fraud of the Nigerian Spam.

As a conclusion, the project will comprise the research about the Naïve Bayes Algorithm that will be implemented in the system, and act as a database of keywords to find the level of spamicity and meant for Nigerian spam. Naïve Bayes algorithm is one of the better filtering technique based on the probabilities.

4.2 Recommendation

Although the author had managed to do the tokenization processes by breaking down the strings into tokens, the system could be expanded more by linking the system with the database and can do comparison with the tokens databases to calculate the level of spamicity. Currently the system only works in Java environment. In addition, the system can be use in the real online environment.

The author hope that this project will be able to be implemented in the real world, so that it can reduce the spam fraud that is now coming to spread all over the world. The author also recommend for the system to delete those spam directly from the inbox of e-mail when it is automatically detected from the system before it enters into the recipient inbox.

REFERENCES

[1] http://en.wikipedia.org/wiki/Advance_fee_fraud

[2] A Simple SPAM Filter for SMTP Using User Authorization and Naïve Bayesian Networks
(http://thor.prohosting.com/~amitc/Source/sagereport.pdf)

[3] Learning Spam: Simple Techniques for Freely-Available Software
(http://nexp.cs.pdx.edu/twiki-sam/pub/PSAM/PsamDocumentation/spam.pdf)

[4] Raymond Lewallen Software Development Life Cycle Models.htm

[5] BAYESIAN FILTERING EXAMPLE - Bayesian Spamicity

[6] Understanding and Controlling Spam: the #1 E-mail threats for 2003
(http://www.swandistribution.com/software/download/SPAM-Jan03.pdf)

[7] http://email.secureserver.net/primer

[8] http://paulgraham.com/spam.html

[9] Using AdaBoost and Decision Stumps to Identify Spam E-mail

[10] How do Bayesian spam filters work
(http://www.inboxer.com/supp_faqbayes.shtml)

[11] What is Bayesian filter - A Word Definition from the Webopedia Computer Dictionary
(http://www.webopedia.com/TERM/B/Bayesian_filter.html)

[12] http://www.gfi.com/whitepapers/why-bayesian-filtering.pdf

APPENDIXES

Appendix 1: Example of Nigerian Spam

```
From: "BIBI LUCKY" <bibialoral@example.com>
Subject: can you?
Date: Thu, 28 Mar 2002 15:03:44 +0100
To: John.Doe@example.com
Reply-To: bibialoral@example.com

Dear Sir,

ASSISTANCE REQUIRED FOR ACQUISITION OF ESTATE

I write to inform you of my desire to acquire estates
or landed properties in your country on behalf of the
Director of Contracts and Finance Allocations of the
Federal Ministry of Works and Housing in Nigeria.

Considering his very strategic and influential
position, he would want the transaction to be as
strictly confidential as possible. He further wants
his identity to remain undisclosed at least for now,
until the completion of the transaction. Hence our
desire to have an overseas agent.

I have therefore been directed to inquire if you would
agree to act as our overseas agent in order to
actualize this transaction.

The deal, in brief, is that the funds with which we
intend to carry out our proposed investments in your
country is presently in a coded account at the
Nigerian Apex Bank (i.e. the Central Bank of Nigeria)
and we need your assistance to transfer the funds to
your country in a convenient bank account that will be
provided by you before we can put the funds into use in
your country. For this, you shall be
considered to have executed a contract for the Federal
Ministry of Works and Housing in Nigeria for which
payment should be effected to you by the Ministry, The
contract sum of which shall run into US$26.4 Million,
of which your share shall be 30% if you agree to be
our overseas agent.

As soon as payment is effected, and the amount
mentioned above is successfully transferred into your
account, we intend to use our own share in acquiring
some estates abroad. For this too you shall also serve
as our agent.

In the light of this, I would like you to forward to
me the following information:

1. Your company name and address if any
2. Your personal fax number
3. Your personal telephone number for easy
communication.

You are requested to communicate your acceptance of
this proposal through my above stated email address
after which we shall discuss in details the modalities
for seeing this transaction through.
```

Your quick response will be highly appreciated. Thank you in anticipation of your cooperation.

Yours faithfully,
BIBI LUCKY.

Appendix 2: Java Codes for Tokenization

```
/*
 * MyTokenization.java
 *
 */


package TokenizationDemo;

import javax.swing.*;
import java.util.*;
import java.awt.*;
import java.awt.event.*;


/**
 *
 * @author  Shasha
 */
public class MyTokenization extends javax.swing.JFrame {

   /** Creates new form MyTokenization */
   public MyTokenization()
   {
         initComponents();
   }


   /** This method is called from within the constructor to
    * initialize the form.
    * WARNING: Do NOT modify this code. The content of this method is
    * always regenerated by the Form Editor.
    */

   private void initComponents() {
      jpTittle = new javax.swing.JPanel();
      jlableTitle = new javax.swing.JLabel();
      jpIO = new javax.swing.JPanel();
      jpInput = new javax.swing.JPanel();
      jScrollPane1 = new javax.swing.JScrollPane();
      jtaInput = new javax.swing.JTextArea();
      jpOutput = new javax.swing.JPanel();
      jScrollPane2 = new javax.swing.JScrollPane();
      jtaOutput = new javax.swing.JTextArea();
      jpButtons = new javax.swing.JPanel();
      jbtTokenize = new javax.swing.JButton();
      jbtExit = new javax.swing.JButton();

      addWindowListener(new java.awt.event.WindowAdapter() {
         public void windowClosing(java.awt.event.WindowEvent evt) {
            exitForm(evt);
         }
      });

      jlableTitle.setText("Welcome To Tokenization Page");
      jpTittle.add(jlableTitle);

      getContentPane().add(jpTittle, java.awt.BorderLayout.NORTH);

      jpIO.setLayout(new java.awt.GridLayout(1, 0));
    jpInput.setLayout(null);

    jpInput.setBorder(new javax.swing.border.TitledBorder("Input Text Area"));
    jtaInput.setName("jtaInput");
    jScrollPane1.setViewportView(jtaInput);

    jpInput.add(jScrollPane1);
    jScrollPane1.setBounds(20, 20, 460, 560);

    jpIO.add(jpInput);

    jpOutput.setLayout(null);

    jpOutput.setBorder(new javax.swing.border.TitledBorder("Output Text Area"));
    jpOutput.setFont(new java.awt.Font("American Classic Extra Bold", 0, 18));
    jtaOutput.setName("jtaOutput");
    jScrollPane2.setViewportView(jtaOutput);

    jpOutput.add(jScrollPane2);
    jScrollPane2.setBounds(10, 20, 480, 560);

    jpIO.add(jpOutput);

    getContentPane().add(jpIO, java.awt.BorderLayout.CENTER);

    jbtTokenize.setText("Tokenize");
    jbtTokenize.setName("jbtTokenize");
    jbtTokenize.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(java.awt.event.ActionEvent evt) {
        jbtTokenizeActionPerformed(evt);
      }
    });

    jpButtons.add(jbtTokenize);

    jbtExit.setText("Exit");
    jbtExit.setName("jbtExit");
    jbtExit.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(java.awt.event.ActionEvent evt) {
        jbtExitActionPerformed(evt);
      }
    });

    jpButtons.add(jbtExit);

    getContentPane().add(jpButtons, java.awt.BorderLayout.SOUTH);

    pack();
  }

  private void jbtExitActionPerformed(java.awt.event.ActionEvent evt) {
    // Add your handling code here:
    System.exit(0);
  }

  private void jbtTokenizeActionPerformed(java.awt.event.ActionEvent evt) {


        String stringToTokenize = jtaInput.getText();
        StringTokenizer tokens = new StringTokenizer( stringToTokenize );
```

```
            jtaOutput.setText( "Number of elements: " + tokens.countTokens() + "\nThe tokens are:\n" );

             while ( tokens.hasMoreTokens() )
             jtaOutput.append( tokens.nextToken() + "\n" );



    }

    /** Exit the Application */
    private void exitForm(java.awt.event.WindowEvent evt) {
        System.exit(0);
    }

    /**
     * @param args the command line arguments
     */
    public void main(String args[]) {
        new MyTokenization().show();
    }


    // Variables declaration - do not modify
    private javax.swing.JButton jbtExit;
    private javax.swing.JTextArea jtaOutput;
    private javax.swing.JTextArea jtaInput;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JPanel jpIO;
    private javax.swing.JPanel jpOutput;
    private javax.swing.JPanel jpInput;
    private javax.swing.JLabel jlableTitle;
    private javax.swing.JPanel jpButtons;
    private javax.swing.JButton jbtTokenize;
    private javax.swing.JPanel jpTittle;
    // End of variables declaration

}
```

Appendix 3: Database Source Codes

```
/*
 * TestJdbcOdbcSha.java
 *
 */

package jdbcdemo;

import java.sql.*;
/**
 *
 * @author  Shasha
 */
public class TestJdbcOdbc {

  public static void main (String [] args)
  throws SQLException, ClassNotFoundException {

    String url = "jdbc:odbc:Tokens";
    String userID = "sha";
    String password = "sha";
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    System.out.println("Driver sun.jdbc.odbc.JdbcOdbcDriverloaded");

    Connection connection = DriverManager.getConnection(url,userID,password);
    System.out.println("Database tokens connected");

    Statement stmt = connection.createStatement();

    try{

      stmt.executeUpdate("create table Frequencies(Tokens varchar(20), Frequency varchar(5))");
      stmt.executeUpdate("insert into Frequencies values ('the', '5')");
      stmt.executeUpdate("insert into Frequencies values ('from', '10')");
    }

    catch (SQLException ex){
      System.out.println("Table Frequencies Already Exist");
    }

    ResultSet rset = stmt.executeQuery("select Tokens, Frequency from Frequencies");

    while (rset.next())
      System.out.println(rset.getString(1) + " " + rset.getString(2) + " ");

    connection.close();
  }

}
```

Appendix 4: Example of Bayesian Filtering

BAYESIAN FILTERING EXAMPLE

Using Bayes' Formula to keep spam out of your Inbox

:ument introduces ormula and provides :pth example of how ian filter can be used fy spam e-mail es. A more general v of Bayesian filtering ined in the :tion to Bayesian t whitepaper, 'e from Process e's website at 'ww.process.com.

BAYES' FORMULA

Thomas Bayes was born in 1702 in London, the son of a minister. After being educated privately, he was ordained a minister like his father and was assigned to a chapel in Tunbridge Wells, 35 miles outside of London. After Bayes' death in 1761, his friend Richard Price discovered his theory of probability in his papers. The theory was published by the Royal Society in 1764.

In basic terms, Bayes' Formula allows us to determine the probability of an event occurring based on the probabilities of two or more independent evidentiary events. Mathematically, the general formula is represented as:

$$P(E_j \mid F) = \frac{P(F \mid E_j)\,P(E_j)}{\sum P(F \mid E_i)\,P(E_i)}$$

Assuming that the variables *a* and *b* are the probabilities of two evidentiary events, the probability would be equal to:

$$\frac{ab}{ab + (1 - a)(1 - b)}$$

For three evidentiary events *a*, *b*, and *c*, the formula expands so the probability is equal to:

$$\frac{abc}{abc + (1 - a)(1 - b)(1 - c)}$$

In this fashion, the formula can be expanded to accommodate any number of evidentiary events.

A PLATINUM EQUITY COMPANY

IMPLE EXAMPLE

ıse that CheapSkies Airlines flights between Boston and New York City are delayed 75% of the time if it's raining. ;uppose that if a flight is scheduled to leave Boston before noon, it's only delayed 10% percent of the time (rain or . If you take a CheapSkies flight from Boston to New York City on a rainy day, and the flight is scheduled to depart : noon, what are the odds your flight will be delayed?

there are only two pieces of evidence to consider (the weather conditions and the scheduled departure time), we can e basic form of Bayes' Formula to solve this problem. The probability that the flight will be delayed on a rainy day or 0.75) is represented by the variable a, and the probability that the flight will be delayed if it's scheduled to leave : noon (10%, or 0.10) is represented by the variable b.

; in Bayes' Formula from above, we see that the probability is equal to:

$$\frac{(0.75)(0.10)}{(0.75)(0.10) + (1 - 0.75)(1 - 0.10)}$$

ıg this equation yields a probability of 0.25, or a 25% chance that your flight will be delayed.

ıportant observation from this example is that we're dealing with independent events – the probability of one event › impact on the other event. In the case of our example, there's a 75% chance the flight will be delayed on a rainy ;gardless of whether or not it's scheduled to leave before noon. The probability of 75% includes both cases where ght leaves before noon, and cases where it doesn't. Likewise, the fact that there's a 10% chance of the flight being :d if it leaves before noon takes into account all flights – not just ones that leave on rainy days.

this concept to filter spam messages is known as naive Bayesian filtering, because we don't take into account the ınships between the various words contained in email messages. While it may certainly be true that a message con- ; all three of the words "clinical", "trial", and "Viagra" is never spam, all the naive Bayesian filter knows is that the "clinical" and "trial" occur mostly in non-spam messages while the word "Viagra" occurs mostly in spam mes-

ıM FILTERING EXAMPLE

real world, applications for Bayes' Formula are messier and more complicated than the contrived example in the ıus section. Following is a complete example of an e-mail message being filtered by a Bayesian filter similar to the cluded in Process Software's PreciseMail Anti-Spam Gateway.

ır example, we're going to use the following "Nigerian spam" message. Note that we're looking at the complete ge – headers and all.

Figure 1: Sample Spam Message

```
ived: from unknown (HELO incamail.com) (209.11.24.18)
 venice.example.com with SMTP; 4 May 2003 14:15:35 -0000
ived: from [10.1.1.27] (HELO app2.incamail.com)
 incamail.com (CommuniGate Pro SMTP 4.0.6)
:h ESMTP id 2217203; Sun, 04 May 2003 10:12:16 -0400
ige-ID: <6549662.1052057538895.JavaMail.tomcat@app2.incamail.com>
: BUMA SARO WIWA <bsarowiwa@incamail.com>
)sarowiwa@incamail.com
ect: URGENT ASSISTANCE PLEAse
-Version: 1.0
ent-Type: text/plain; charset=us-ascii
ent-Transfer-Encoding: 7bit
iority: 3
fix: INBOX
: Sun, 04 May 2003 10:12:16 -0400
ent-Length: 2388

rincess Buma Saro-Wiwa
'ounde avenue YD
 Cameroun.
)wiwa@incamail.com OR b_sarowiwa@yahoo.com.au

 Friend,

: your contact from a directory in a library in one of our international school in my
:ry and my instinct tells me to write you and i feel It will be a great pleasure to
i contact with someone like you.
:, let me introduce myself, my name is PrincessBuma Nene Saro Wiwa Ken. I am 27 years
from a royal family of Ken sarowiwa Kings hence I bear the tittle "PRINCESS" I am
.e and the only duagther of my parents.my father was a royal king of OGONI a promi-
 community in Rivers state Nigeria who was killed through hanging by the order of
 Gen sani Abacha because of his community inheritance which are ( crude oil) that the
I has taken possession of it.
re only two, I and my younger brother KEN SARO WIWA[jnr],after one year death of my
er, my mother died of High Blood preasure (HBP).Meanwhile, we inherited some fortune
orm of cash which I will reveal to you when we get your response.Our old family
ids have been very dishonest with us since the death of our parents, they have duped
f virtually all cash in the banks with different stories and reason. As such we
led to cut off relationship from people around us because we find out that they have
otive to squander what is left. We had to leave Nigeria to stay in neighbuoring
roun republic with the assistance of our family lawyer in Nigeria, we are here now
three years and would like to move out to another continent.I am interested to enter
 strong relation with you as a friend and partner after i have gotten good informa-
 about you on internet.To be frank, we need someone who is kind and sincere that will
st us.
re interested to invest and live in your country therefore, it will be our pleasure
ou can be of help to us by assisting us to handle the investment and planing of our
ine we inherited, to enable us build a new home for safekeeping of our lives.
se let me receive your response urgently.My kindest compliments.

s Faithfully,
ress B. Saro-Wiwa.
)wiwa@incamail.com OR b_sarowiwa@yahoo.com.au

_________________________________

i of spam and email overload?
i FREE 6MB email account at http://www.incamail.com
```

rst thing a Bayesian filter must do is split the message into tokens and build a table of all the tokens it intends to use decision making process. For our sample message, the table would be:

Figure 2: Spam Message Token Table

10.1.1.27	209.11.24.18	abacha	about
account	after	all	and
another	app2.incamail.com	are	around
assist	assistance	assisting	avenue
banks	bear	because	been
bit	blood	brother	bsarowiwa
build	buma	cameroun	can
cash	charset	communigate	community
compliments	contact	content-length	content-type
continent.i	country	crude	cut
dear	death	decided	died
different	directory	dishonest	duagther
duped	email	enable	enter
esmtp	f.g.n	faithfully	family
father	feel	find	for
form	fortune	frank	free
friend	friends	frist	from
gen	get	good	got
gotten	great	had	handle
hanging	has	have	hbp
helo	help	hence	here
high	his	home	http
inbox	incamail.com	information	inheritance
inherited	instinct	interested	international
internet.to	into	introduce	invest
investment	jnr	ken	killed
kind	kindest	king	kings
late	lawyer	leave	left
let	library	like	live
lives	may	meanwhile	mime-version
mother	motive	move	myself
name	need	neighbuoring	nene
new	nigeria	now	off
ogoni	oil	old	one
only	order	our	out
overload	parents	parents.my	partner
people	plain	planing	please
pleasure	possession	preasure	princess
princessbuma	pro	prominent	reason
receive	received	relation	relationship
republic	response	response.our	reveal
rivers	royal	safekeeping	sani
saro	saro-wiwa	sarowiwa	school
since	sincere	single	smtp
some	someone	spam	squander
state	stay	stories	strong
subject	such	sun	taken
tells	text	that	the
therefore	they	three	through
tired	tittle	two	unknown
urgent	urgently.my	us-ascii	venice.example.com
very	virtually	was	what
when	which	who	will
with	wiwa	would	write
www.incamail.com	x-priority	x-suffix	yahoo.com.au
year	years	you	younde
younger	your	yours	

the Bayesian filter has the list of tokens in the message, it searches the spam and non-spam token databases for tokens. These databases of tokens are created and updated whenever the Bayesian filter is "trained" on a new mes-

ken from the message is found in the databases, the Bayesian filter calculates the token's spamicity based on the ing variables:

he frequency of the token in spam messages that the filter has been trained on

he frequency of the token in ham messages that the filter has been trained on

he number of spam messages the filter has been trained on

he number of ham messages the filter has been trained on

lgorithm used to calculate a token's spamicity from these pieces of information is as follows:

```
robability = Token frequency in ham messages / Number of ham messages trained on

 probability = Token frequency in spam messages / Number of spam messages trained on

ither Ham probability or Spam probability are greater than 1.0, set them equal to


icity = Spam probability / (Ham probability + Spam probability)
```

ken has occurred less than 5 times total in both ham and spam messages, the token is assigned a default spamicity . The following example and table use a set of sample token databases generated by live mail feed on a test system cess Software. The Bayesian filter was trained on 19,977 spam messages and 5,141 ham messages.

ample of this algorithm, using the token "after" from the example spam message and frequency values from Figure

```
robability = 1184 / 5141 = 0.230305
 probability = 1134 / 19977 = 0.056765
icity = 0.056765 / (0.056765 + 0.230305) = 0.197740
```

ells us that there's only a 19.8% chance that a message containing the word "after" is a spam message.

ting this process for each of the tokens in our sample message, we get the following frequencies and spamicities:

Figure 3: Spam Message Token Frequency and Spamicity Table

oken	Spam Frequency	Ham Frequency	Spamicity
10.1.1.27	0	0	0.400000
209.11.24.18	0	0	0.400000
bacha	14	2	0.643038
bout	3301	2578	0.247848
ccount	585	563	0.210984
fter	1134	1184	0.197740
ll	9767	3759	0.400717
nd	32109	12353	0.500000
nother	1305	784	0.299898
pp2.incamail.com	0	0	0.400000
re	13555	6130	0.404241
round	433	480	0.188409
ssist	256	46	0.588847
ssistance	386	171	0.367453

isting	6	4	0.278509
nue	70	25	0.418797
ks	238	8	0.884474
r	80	12	0.631763
ause	5114	973	0.574936
n	3233	2036	0.290097
	4296	2292	0.325398
od	383	53	0.650312
ther	171	171	0.403703
rowiwa	0	0	0.400000
ld	3364	576	0.600475
a	0	0	0.400000
eroun	0	0	0.400000
	8083	4568	0.312889
h	1318	49	0.873771
rset	9300	3324	0.418608
munigate	16	61	0.063232
munity	70	76	0.191612
pliments	58	58	0.788651
tact	1552	760	0.344489
tent-length	0	0	0.400000
tent-type	26907	5054	0.504267
tinent.i	0	0	0.400000
ntry	316	62	0.567406
de	19	0	0.990000
	272	199	0.260218
r	752	113	0.631350
ch	118	37	0.450768
ided	205	107	0.330228
d	44	31	0.267542
ferent	593	704	0.178152
ectory	57	401	0.035289
honest	0	0	0.400000
gther	0	0	0.400000
ed	0	0	0.400000
il	13820	2097	0.629081
ple	65	97	0.147084
er	753	139	0.582309
tp	7239	7152	0.265983
.n	0	0	0.400000
thfully	35	0	0.990000
ily	3255	172	0.829646
her	75	38	0.336835
l	2269	299	0.661350
d	2966	854	0.471956
	29946	14355	0.500000
n	2721	258	0.730756
tune	211	16	0.772404
nk	47	85	0.124571
e	13077	948	0.780215
end	456	110	0.516164
ends	1215	181	0.633362
st	0	0	0.400000

n	65251	18549	0.500000
	63	14	0.536620
	10853	2876	0.492677
d	1426	1752	0.173185
	946	998	0.196101
ten	49	35	0.264860
at	1761	556	0.449061
	1202	1709	0.153260
dle	201	103	0.334309
ging	39	51	0.164434
	3661	2693	0.259176
e	11235	7113	0.359958
	0	0	0.400000
o	1855	1473	0.244761
o	2364	1406	0.302014
ce	36	16	0.366699
l	2032	265	0.663674
	815	712	0.227545
e	3510	650	0.581532
o	57485	4233	0.548432
ox	74	91	0.173055
amail.com	0	0	0.400000
ormation	4197	1490	0.420252
eritance	0	0	0.400000
erited	0	5	0.010000
tinct	0	0	0.400000
erested	592	237	0.391291
ernational	1392	165	0.684648
ernet.to	0	0	0.400000
o	1359	1268	0.216187
roduce	53	20	0.405458
est	139	7	0.836338
estment	657	31	0.845059
	0	0	0.400000
	0	0	0.400000
led	10	25	0.093331
d	130	266	0.111720
dest	0	0	0.400000
g	210	117	0.315960
gs	8	24	0.079005
e	181	221	0.174078
ver	31	9	0.469894
ve	141	189	0.161066
:	9847	488	0.838522
	1007	987	0.207959
tary	242	274	0.185197
e	6794	2752	0.388500
e	667	166	0.508366
es	106	47	0.367248
	4255	2102	0.342510
nwhile	3	13	0.056058
e-version	17646	4370	0.509602
ner	76	45	0.302956

ive	0	0	0.400000
e	403	336	0.235861
elf	103	110	0.194178
e	10101	1624	0.615480
d	2714	1813	0.278103
ghbuoring	0	0	0.400000
e	0	0	0.400000
	9051	2191	0.515291
eria	132	2	0.944398
	8920	2034	0.530203
	3061	835	0.485437
ni	0	0	0.400000
	64	42	0.281685
	949	731	0.250427
	8722	2995	0.428388
y	4954	2298	0.356824
er	4442	680	0.627015
	16869	1634	0.726535
	5565	2829	0.336092
cload	0	5	0.010000
ents	119	61	0.334237
ents.my	0	0	0.400000
tner	509	39	0.770574
ple	1808	828	0.359768
in	954	3206	0.071131
ning	0	0	0.400000
ase	11780	2108	0.589846
asure	117	13	0.698442
session	10	9	0.222359
asure	0	0	0.400000
ocess	0	0	0.400000
ocessbuma	0	0	0.400000
	1388	102	0.777873
minent	6	0	0.990000
son	552	487	0.225823
eive	8509	348	0.862871
eived	19967	10164	0.499875
ation	20	3	0.631763
ationship	133	69	0.331570
ublic	34	16	0.353529
ponse	645	311	0.347992
ponse.our	0	0	0.400000
eal	29	3	0.713276
ers	0	0	0.400000
al	168	16	0.729885
ekeeping	10	0	0.990000
i	0	0	0.400000
o	0	0	0.400000
o-wiwa	0	0	0.400000
owiwa	0	0	0.400000
ool	313	68	0.542239
te	299	854	0.082654
tere	22	0	0.990000

ingle	229	372	0.136755
mtp	2374	1702	0.264140
ome	1981	2262	0.183924
omeone	728	517	0.265988
pam	1167	956	0.239049
quander	0	0	0.400000
tate	929	467	0.338597
tay	453	201	0.367084
tories	112	44	0.395793
trong	10357	154	0.945377
ubject	22169	10497	0.500000
uch	1026	848	0.237435
un	2608	1611	0.294089
aken	382	122	0.446225
ells	11	29	0.088933
ext	19009	4012	0.549410
hat	10559	9075	0.345789
he	34475	16621	0.500000
herefore	117	122	0.197946
hey	2319	2640	0.184376
hree	607	245	0.389346
hrough	4241	758	0.590138
ired	227	128	0.313369
ittle	0	0	0.400000
wo	775	940	0.175036
nknown	2667	695	0.496866
rgent	93	31	0.435678
rgently.my	0	0	0.400000
s-ascii	665	1891	0.082989
enice.example.com	0	0	0.400000
ery	1173	980	0.235490
irtually	136	18	0.660371
as	3573	4367	0.173933
hat	3050	3548	0.181150
hen	2404	2614	0.191378
hich	1200	2132	0.126521
ho	2041	1183	0.307476
ill	9749	4255	0.370922
ith	39458	15761	0.500000
iwa	0	0	0.400000
ould	6023	3296	0.319851
rite	903	329	0.413948
ww.incamail.com	0	0	0.400000
-priority	11524	852	0.776826
-suffix	0	0	0.400000
ahoo.com.au	0	0	0.400000
ear	1096	421	0.401182
ears	1397	503	0.416820
ou	40273	9606	0.500000
ounde	0	0	0.400000
ounger	250	4	0.941466
our	31926	4534	0.531370
ours	682	75	0.700611

hat the filter has calculated the spamicity value for each token in the message, it needs to choose 15 tokens that will ıgged into the Bayesian formula to calculate the message's overall spamicity. Using a subset of the tokens in the ıge enhances the Bayesian filter's performance, especially when dealing with large messages.

implementations of Bayesian filters chose the 15 tokens that had the most extreme values (i.e. the 15 tokens whose was furthest from the neutral value of 0.5). Spammers have started including words that they're fairly sure will ı low spamicity, such as "congresswoman" and "umbrella", in their messages in an attempt to circumvent this sys- As a result, the Bayesian filter included in Process Software's PreciseMail Anti-Spam Gateway uses a sampling thm based on standards of deviation to choose the 15 tokens fed to the Bayesian formula.

ır sample message, the 15 tokens chosen by the Bayesian filter are:

Figure 4: Token Subset Used in Bayesian Formula

Token	Spamicity
account	0.210984
after	0.197740
crude	0.990000
faithfully	0.990000
good	0.173185
inherited	0.010000
invest	0.836338
investment	0.845059
let	0.207959
overload	0.010000
prominent	0.990000
receive	0.862871
safekeeping	0.990000
sincere	0.990000
therefore	0.197946

the Bayesian filter has selected 15 tokens, it plugs their spamicity values into Bayes' formula, as shown below. 15 different values, this gets a little bit messy on paper.) For our sample message, the probability of the message spam is:

```
(0.210984)(0.197740)(0.990000)(0.990000)(0.173185)(0.010000)(0.836338)(0.845059)
    (0.207959)(0.010000)(0.990000)(0.862871)(0.990000)(0.990000)(0.197946)

(0.210984)(0.197740)(0.990000)(0.990000)(0.173185)(0.010000)(0.836338)(0.845059)
   (0.207959)(0.010000)(0.990000)(0.862871)(0.990000)(0.990000)(0.197946) +
    (1 - 0.210984)(1 - 0.197740)(1 - 0.990000)(1 - 0.990000)(1 - 0.173185)
    (1 - 0.010000)(1 - 0.836338)(1 - 0.845059)(1 - 0.207959)(1 - 0.010000)
    (1 - 0.990000)(1 - 0.862871)(1 - 0.990000)(1 - 0.990000)(1 - 0.197946)
```

quation simplifies to:

```
          0.000000017249220883574410361053715216318

             0.000000017249334195201446371086
```

ıg this equation yields a probability of 0.999993, or a 99.9993% chance that the message is spam. If this message ɜent to an email server protected by PreciseMail Anti-Spam Gateway, it would be quarantined, discarded, or tagged m based on the options chosen by the systems administrator.

ɪian filtering is one method used by Process Software's PreciseMail Anti-Spam Gateway to keep junk out of your Inbox. For more information on Bayesian filtering, including an in-depth example, visit the ɜss Software website at http://www.process.com/. A free demonstration of PreciseMail Anti-Spam Gateway ɔ available from the Process Software website, so you can try Bayesian filtering on your email server.

PROCESS™
SOFTWARE

A PLATINUM EQUITY COMPANY

U.S.A.: (800)722-7770 • International: (508)879-6994 • Fax: (508)879-0042
E-mail: info@process.com • Web: http://www.process.com

Appendix 5: Example of Spam E-Mail and Database

Sample Email

From: lonely_prince@mymail.com
To: myself@yourmail.com
Subject: Need Help

Dear Friend

I got your email from a respective source. Let me introduce myself, my name is Prince Horo, I am a son of King Huru. I need your help in keeping a secret. If you could help me, I will reward you with a sumptuous amount of money. Please reply to me as soon as possible.

Regards

Prince Horo

List of Tokens extracted from the Email

a	help	King	of	source
am	help	let	please	sumptuous
amount	help	lonely_prince	possible	to
as	Horo	me	Prince	will
could	Huru	money	reply	with
dear	i	my	respective	you
email	if	mymail	reward	your
friend	introduce	myself	secret	yourmail
from	is	name	son	
got	keeping	need	soon	

Spam/Ham Database

Token	***Ham Frequency***	***Spam Frequency***	***Total Frequency***
a	5	4	9
am	6	6	12
amount	2	8	10
as	4	3	7
could	6	7	13
dear	5	5	10
email	4	7	11
friend	4	3	7
from	6	3	9
got	7	5	12
help	8	5	13
help	6	4	10
help	3	3	6
Horo	0	1	1
Huru	0	1	1
i	3	5	8
if	4	2	6
introduce	8	6	14
is	8	7	15
keeping	7	3	10
King	5	2	7
let	4	4	8
lonely_prince	5	1	6
me	6	3	9
money	7	11	18
my	6	6	12
mymail	6	3	9
myself	3	4	7
name	7	6	13
need	5	5	10
of	3	4	7
please	8	7	15
possible	9	4	13
Prince	4	2	6
reply	2	4	6
respective	6	3	9
reward	2	9	11
secret	8	7	15
son	5	2	7
soon	4	10	14
source	3	8	11
sumptuous	3	9	12
to	9	5	14
will	5	7	12
with	4	6	10
you	7	3	10

your	2	4	6
yourmail	5	6	11

Number of Ham Messages trained:	***100***
Number of Spam Messages trained:	***100***

could	46	57	103	0.46	0.57	0.553398058
source	36	45	81	0.36	0.45	0.555555556
will	34	44	78	0.34	0.44	0.564102564
keeping	53	78	131	0.53	0.78	0.595419847
reply	23	34	57	0.23	0.34	0.596491228
amount	56	88	144	0.56	0.88	0.611111111
a	25	40	65	0.25	0.4	0.615384615
help	33	54	87	0.33	0.54	0.620689655
my	22	36	58	0.22	0.36	0.620689655
am	32	63	95	0.32	0.63	0.663157895
soon	39	100	139	0.39	1	0.71942446
dear	23	66	89	0.23	0.66	0.741573034
King	24	72	96	0.24	0.72	0.75
money	33	131	164	0.33	1	0.751879699
let	23	83	106	0.23	0.83	0.783018868

Conditions:	1. If total frequency of token < 5, default spamicity of 0.4 is assigned
	2. If either ham probability or spam probability greater than 1.0, set them to 1.0

Token	***Ham Frequency***	***Spam Frequency***	***Total Frequency***	***Ham Probability***	***Spam Probability***	***Spamicity***
lonely_prince	23	1	24	0.23	0.01	0.041666667
if	87	32	119	0.87	0.32	0.268907563
secret	88	34	122	0.88	0.34	0.278688525
mymail	54	21	75	0.54	0.21	0.28
friend	54	22	76	0.54	0.22	0.289473684
introduce	76	33	109	0.76	0.33	0.302752294
myself	77	34	111	0.77	0.34	0.306306306
got	76	34	110	0.76	0.34	0.309090909
help	89	45	134	0.89	0.45	0.335820896
need	87	44	131	0.87	0.44	0.335877863
your	62	34	96	0.62	0.34	0.354166667
i	76	45	121	0.76	0.45	0.371900826
son	67	42	109	0.67	0.42	0.385321101
from	54	34	88	0.54	0.34	0.386363636
Horo	0	1	1	0	0.01	0.4
Huru	0	1	1	0	0.01	0.4
email	45	32	77	0.45	0.32	0.415584416
as	44	32	76	0.44	0.32	0.421052632
of	67	54	121	0.67	0.54	0.446280992
is	67	56	123	0.67	0.56	0.455284553
you	37	33	70	0.37	0.33	0.471428571
reward	66	59	125	0.66	0.59	0.472
possible	45	43	88	0.45	0.43	0.488636364
me	44	43	87	0.44	0.43	0.494252874
Prince	43	43	86	0.43	0.43	0.5
to	56	56	112	0.56	0.56	0.5
name	55	56	111	0.55	0.56	0.504504505
yourmail	55	56	111	0.55	0.56	0.504504505
please	65	67	132	0.65	0.67	0.507575758
respective	32	34	66	0.32	0.34	0.515151515
help	21	23	44	0.21	0.23	0.522727273

RESULTS

Token	***Spamicity***	***1 - Spamicity***
lonely_prince	0.041666667	0.958333333
if	0.268907563	0.731092437
secret	0.278688525	0.721311475
mymail	0.28	0.72
friend	0.289473684	0.710526316
introduce	0.302752294	0.697247706
myself	0.306306306	0.693693694
got	0.309090909	0.690909091
help	0.335820896	0.664179104
need	0.335877863	0.664122137
soon	0.71942446	0.28057554
dear	0.741573034	0.258426966
King	0.75	0.25
money	0.751879699	0.248120301
let	0.783018868	0.216981132

1.9276E-07   3.71916E-05

Following the Naïve Bayes Theorem

The probability of the email being spam is:	***0.005156172***
Percentage %:	***0.515617157***