Mining Mailing Lists for Content

by

Mario A. Harik

B.E. Computer and Communications Engineering
American University of Beirut, 2002

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

MASTER OF ENGINEERING in CIVIL AND ENVIRONMENTAL ENGINEERING

at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUNE 2003

© 2003 Mario A. Harik. All rights reserved.

The author hereby grants to MIT permission to reproduce
and to distribute publicly paper and electronic
copies of this thesis document in whole or in part.

Signature of Author: _____
Department of Civil and Environmental Engineering
May 9, 2003

Certified by: _____
John Williams
Associate Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by: _____
Oral Buyukozturk
Professor of Civil and Environmental Engineering
Chairman, Departmental Committee on Graduate Studies

BARKER

MASSACHUSETTS INSTITUTE
OF TECHNOLOGY
JUN 02 2003
LIBRARIES

Mining Mailing Lists for Content

by

Mario A. Harik

Submitted to the Department of Civil and Environmental Engineering
on May 9, 2003
in partial fulfillment of the requirements for
the degree of Master of Engineering in Civil and Environmental Engineering

ABSTRACT

In large decentralized institutions such as MIT, finding information about events and activities on a campus-wide basis can be a strenuous task. This is mainly due to the ephemeral nature of events and the inability to impose a centralized information system to all event organizers and target audiences. For the purpose of advertising events, Email is the communication medium of choice. In particular, there is a wide-spread use of electronic mailing lists to publicize events and activities. These can be used as a valuable source for information mining.

This dissertation will propose two mining architectures to find category-specific event announcements broadcasted on public MIT mailing lists. At the center of these mining systems is a text classifier that groups Emails based on their textual content. Classification is followed by information extraction where labeled data, such as the event date, is identified and stored along with the Email content in a searchable database. The first architecture is based on a probabilistic classification method, namely naïve-Bayes while the second uses a rules-based classifier. A case implementation, FreeFood@MIT, was implemented to expose the results of these classification schemes and is used as a benchmark for recommendations.

Thesis Supervisor: John Williams
Title: Associate Professor of Civil and Environmental Engineering

ACKNOWLEDGEMENTS

I dedicate this thesis to my parents, Adel and Roxanne, for their love, guidance and constant support; to my wonderful brothers, Mel and Marc, for standing by my side.

I thank the following people for their valuable contribution:

John Williams, for being an encouraging thesis advisor.

Eric Adams, Georges Kocur and Kevin Amaratunga for helping me throughout my stay at MIT.

Nadim Chehade, for his unwavering support.

Hani Harik, for valuable guidance and advice.

My family and friends for their care and affection.

And all the other wonderful people I met at MIT for their friendship and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1
Introduction

1.1 Motivation and Problem Description

In large academic institutions such as MIT, finding about a certain event or activity can be extremely hard. This is due to decentralization where each department, research group or social club has a limited reach in terms of spreading information on an institute-wide basis. For example, this shortcoming can be seen through the relative isolation of students within their departments. It is rare for them to hear about an event occurring on the "other side of campus".

A communication medium that facilitates the dissemination of information in such environments is Email. In the rare occurrences where people hear about events not affiliated to their surroundings, it is primarily due to Email. However, receiving Emails publicizing an event is based on how well events organizers advertise it and does not allow people to search for events. Although there are other approaches for searching for specific types of activities or events such as dedicated websites (i.e. events.mit.edu) or particular newsgroups (i.e. mit.lcs.announce or mit.lcs.seminar), these have the major flaw of assuming that both target audience and event organizers know about and will be using them. This becomes an unrealistic assumption in environments with a large number of diversified communities.

The problem is well identified; people at MIT cannot search for events based on their type or content. For example, if a student knows that there is a certain lecture of interest somewhere around campus; he doesn't have a way to search for it. In answering this concern, a key requirement is to stipulate minimum effort from both people who want to publicize an event and those who are interested in finding about it. As far as end users are concerned, they want to be able to search for an event based on its type (i.e. seminar, lecture, dinner...) regardless of its affiliations. On the other hand, event

organizers cannot be involved in the process of gathering event information since this brings us to the drawbacks of dedicated approaches.

To address this problem, this thesis will investigate searching Emails to find about events. However, Email is a targeted communications medium where both senders and recipients need to be identified which makes it hard to actually use it as a source of information. The special use of Email to drive topic-specific discussion through mailing lists is a more accessible data channel. In this case, recipients are identified as the Emails subscribed to the mailing list. And subscribing any Email to a mailing list can make it a point of access to the broadcasted information.

Information broadcasted on mailing lists includes much more than just events and activities. This thesis will only cover one approach to formulating useful content out of this extracted information; providing people at MIT with a type or category specific events and activities search engine for mailing lists.

1.2 The Proposed Approach

The volume of information broadcasted on accessible MIT mailing lists can reach remarkable levels (10 GB per day) as will be seen in chapter 2. Therefore, a way to filter information based on our application's need is essential since end users cannot be expected to search through large databases of Emails. For this reason, the use of text classification techniques and information extraction is imperative. These would limit content exposed to end users based on the specific category of information they are looking for. The proposed system to find events and activities from mailing lists can be described as follows:

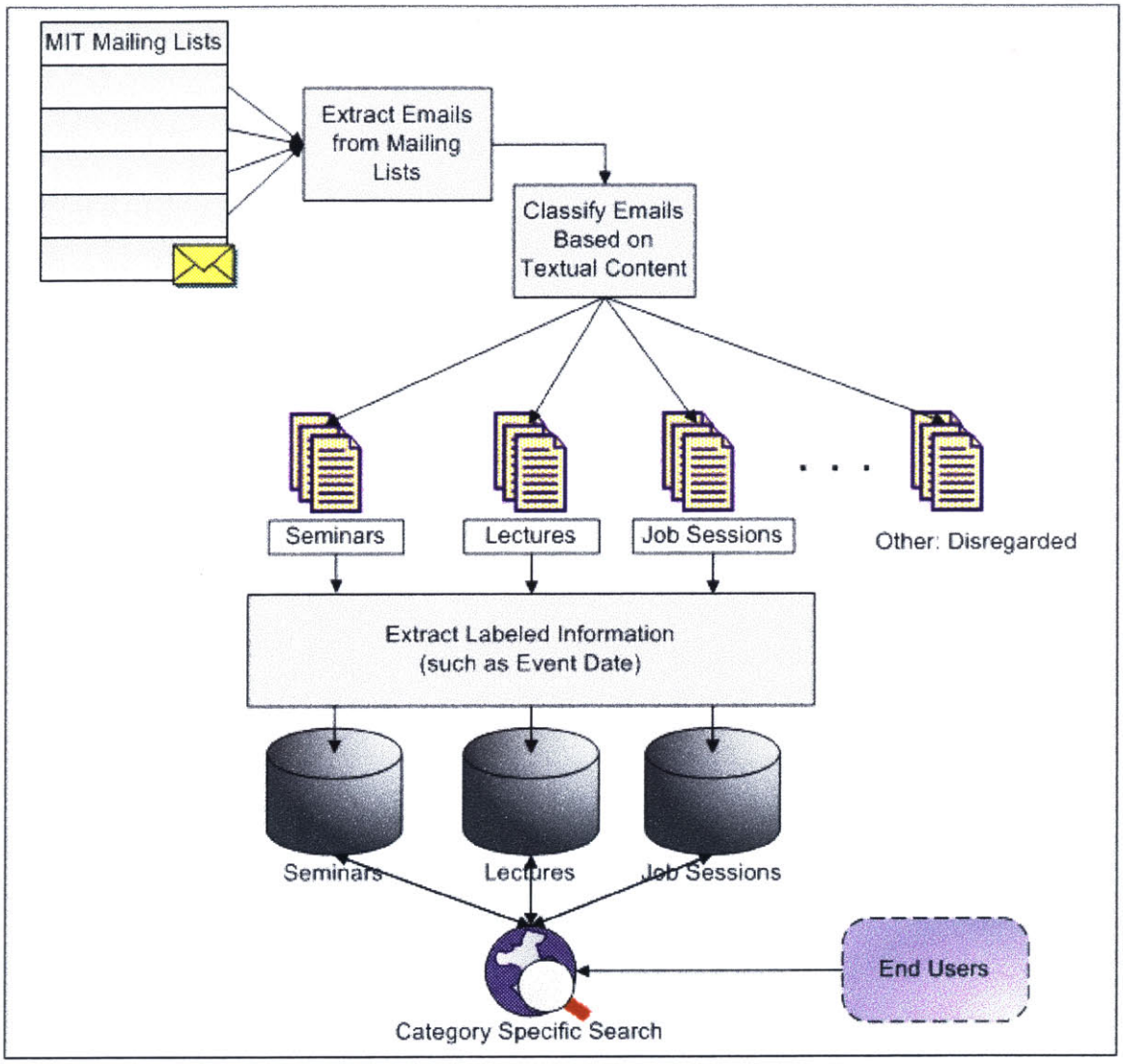


Figure 1 - General mining approach to finding events

This mining approach consists of retrieving Emails from accessible public mailing lists, classifying them based on their content, extracting labeled information that would assist the end users in looking for events and storing them into a searchable database.

Based on this approach, two architectures are going to be exposed in this thesis. The first one is identified by a rules-based classifier and the second by a Bayesian classifier.

1.3 Chapter Discussions

Chapter 2 will introduce mailing lists at MIT and how they can be used as a source of information. Chapter 3 will cover text classification and delve into detailing the two approaches that are considered in the mining architectures: rules-based and naïve-Bayes classification. Chapter 4 will detail the two mining architectures and illustrate the processes involved in transforming broadcasted Email into clustered and searchable content. It will also detail an information extraction scheme. Chapter 5 will cover a case implementation of these architectures that filters a specific type of events labeled as free food events. These are events or activities that have free food servings associated with them. The results of both classification and information extraction schemes will be exposed giving an idea of both accuracy and usability of such a mining system. Finally, a brief conclusion will summarize the thesis recommending one of the two proposed architectures and presenting possible future work.

Chapter 2
Mailing Lists as a Source of Information

This chapter will first introduce mailing lists and will give an insight on the volume of their broadcasted information. It will then cover MIT's mailing lists and how the public ones can be used as a real-time source of data rich in events to mine.

2.1 Using Mailing Lists as a Source of Information

Mailing lists are services where email messages sent to the list are forwarded to everyone that is subscribed to the list. These lists normally cover topic-specific discussions. For example, all the clubs at MIT have mailing lists for their club members to communicate between each other. Mailing lists can be interactive where subscribers can send messages to the entire list or distribution-only where only the list owner can send messages to the list. In both types, the intended receivers are the Emails that are subscribed to the mailing list. Subscribing to a mailing list is dependent on the used mailing list management software. For example, registering to a mailing list provided by LISTSERV (software that provides Email list distribution services) can be done by sending an email with the word "SUBSCRIBE" in its body. Subscribing to a mailing list might require the approval of the list owner. These lists are normally referred to as private lists.

Since the aim of this thesis is to build a system that exposes events broadcasted over mailing lists in a certain institution or community, we must first find a way to access all the messages exchanged through these lists. Creating an Email that our system regularly checks, and subscribing it to a significant number of mailing lists can be the answer.

There are two intuitive concerns in this process. First, we need to find a significant number of mailing lists in the community where we want to discover and expose events.

This can become complex in case we have many mailing list providers. Second, subscribing to a large set of mailing lists can become a lengthy process if we have to request the list administrator's approval for each list. For this purpose, it becomes obvious that private lists cannot be considered as valid sources of information since we need to automate the process of subscribing the system's Email to mailing lists.

2.2 MIT Mailing Lists and Statistics

There are two main Email list distribution services at MIT. The first one is provided by Athena, MIT's "UNIX-based campus-wide academic computing facility" ([27]), and the second is L-Soft's Listserv ([30]) running on mitvma.mit.edu. There are many other Email list management software hosted on different servers all around campus, but the former are the most used by far.

Mailing lists on both systems are divided into two groups: public and private. As mentioned previously, adding an Email to a private list requires the approval of the list administrator. Besides the manual intervention that is involved in adding an Email to such a list, it is unlikely for a private list owner to contribute to a system that might make the messages broadcasted on his list publicly accessible. For this reason, the focus is only on MIT's public mailing lists.

In order to add our system's Email to all the mailing lists on Athena and Listserv, the first step is to get a listing of all the public mailing lists on both.

In Athena's case, there are a few programs that allow us to get information, subscribe and unsubscribe from lists. These are *http://web.mit.edu/moira/*, a web interface to manage Athena's lists, a shell program called *blanche* and another one called *listmaint* ([28]). Although all of these programs have options to display all the public mailing lists at MIT, none is able to create a clear dump of the lists to file. However, a textual listing of those is available on Athena at the following location: */mit/info/public-mailing-lists*.

As for Listserv, the only way to communicate with it is through Email. A simple email containing the word *lists* sent to *listserv@listserv.mit.edu* returns all the names and descriptions of public mailing lists. A listing of these is available in Appendix B.

The number of public mailing lists at MIT is astonishing. Athena and Listserv host 4550 and 415 public mailing lists respectively. If our event miner's Email was subscribed to all of them, the average size of an Email being 18,500 bytes ([29]) and considering only one Email is sent per day to any public mailing list on average, 87.6Mb of data would be received in our server's Email inbox daily. However, L-Soft's Internet mailing lists statistics ([31]) record an average of 113 messages per day per mailing list. This would yield 9.66 Gb of daily traffic to our server's inbox. These numbers were not verified since the case implementations of this thesis used a static pool of Emails for testing the mining architectures.

2.3 Accessing the Information on Public MIT Mailing Lists

With such a large number of public mailing lists at MIT, subscribing an Email to all of them can be problematic if not automated. Fortunately, both systems can be handled in a batch manner. Let's assume the Email associated with the system is *event@freefood.mit.edu.*

In order to subscribe and Email to Athena's public mailing lists, we can use the following command on any Athena terminal on campus:

> **blanche** *listname* **–add** *event@freefood.mit.edu*

Being able to add the Email on a UNIX shell command and having the list of all *listnames* in a text file allows us to add an email to all the public mailing lists using the following Perl script:

```
#!/usr/bin/perl

open(EMAILS, "./Athena_Public_Lists") || die "Error reading from file: $!";
while(<EMAILS>) {
chomp($_);
system("blanche ".$_." -add event@freefood.mit.edu");
}
close(EMAILS);
```

Table 1 - Perl script to subscribe an Email to all Athena public mailing lists

The same process can be applied to Listserv, but instead of running a shell command, the following Email should be sent to *listserv@mitvma.mit.edu*:

To: *listserv@listserv.mit.edu*
From: *event@freefood.mit.edu*
Subject:

subscribe *listname*

Table 2 - Subscribe an Email to Listserv

As in Athena's case, this email can be sent with *listname* being read from a file using a similar Perl script as the one described above.

After subscribing the Email *event@freefood.mit.edu* to all of Athena's and Listserv's public mailing lists, the event mining system will only have to check this Email to read the messages broadcasted over all the public mailing lists. These Emails are then considered as input documents to the first component of the events mining system, the classification engine.

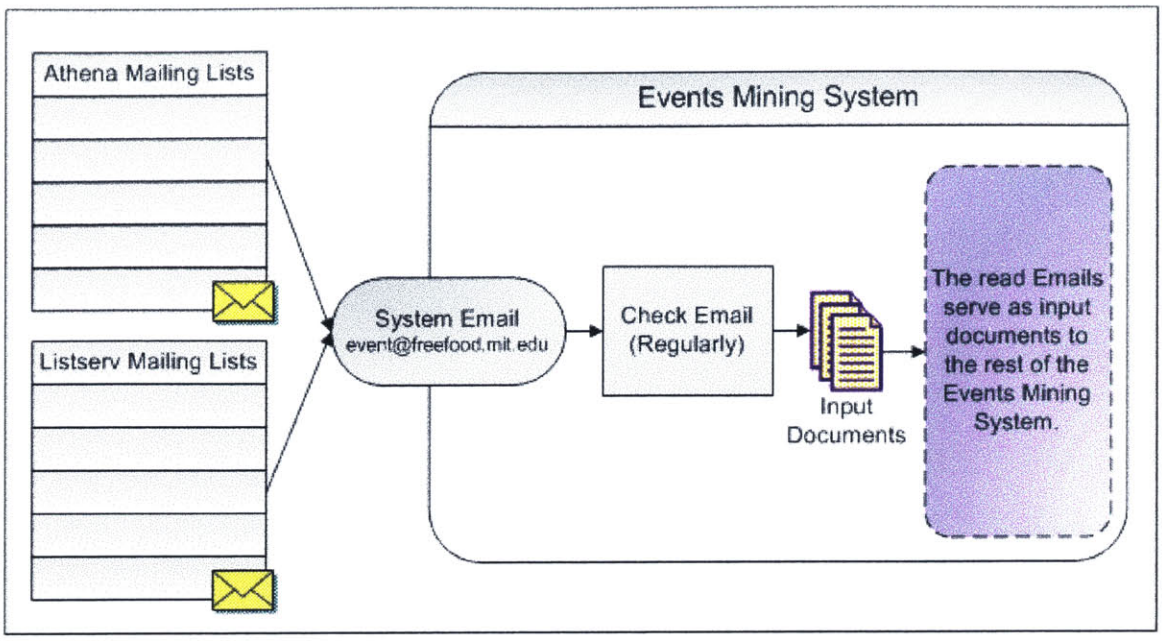


Figure 2 - Emails as input documents to the events mining system

Chapter 3
Text Classification

With the large amounts of emails to be classified, as described in chapter 2, it is essential to have an automated way for performing text classification. For this purpose, the use of machine learning techniques for automatic text classification is essential. This chapter first gives an introduction to text classification and then delves into detailing each of the classification algorithms that will be later used in the proposed mining approaches (chapter 4).

3.1 Definition and Terminology

Text classification is the task of automatically assigning free text input documents to a predefined set of classes. A class can be viewed as a semantic category that groups a set of documents having certain features or properties in common. For example, in email classification, a simple scheme would be to group emails by whether they contain information about seminars or not. In general, classification can result in a document being assigned to exactly one, multiple or no classes. In the case implementation covered by this thesis, we will use a specific type of classification known as information filtering or binary classification where each document is classified as belonging to a certain class or not.

The text classification problem can be described as follows:

Assume a set of documents D and a fixed set of classes $C = \{c_1, c_2, ..., c_k\}$ which implies a disjoint, exhaustive partition of D. Text classification is a mapping $h : D \rightarrow C$, from the document space into the set of classes ([11]).

With ever increasing amounts of information, and with the fixed speed at which humans can read and analyze it, it becomes essential to find ways for automating the

classification task. Using machine learning techniques to handle this activity is one of the most promising approaches. The idea behind machine learning is to be able to label a new document based on a set of previously labeled or training set of documents. The outcome is obviously a prediction that the classifier makes based on its training set. This problem is known as supervised learning and it can be described as follows:

Assume a set of n labeled training documents $D' = \{d_1, d_2, ..., d_n\} \subset D$ and a fixed set of classes $C = \{c_1, c_2, ..., c_k\}$. Let $t : D \rightarrow C$ be the target function that assigns each training document $d \in D'$ its true class label $t(d)$. The objective of the learning task is to induce a classifier represented by the mapping $h : D \rightarrow C$ from D', which approximates the target function t well with respect to a given effectiveness measure ([11]).

Such a classifier is a two phase process. The first is a learning phase where supervised learning is applied to construct the classifier. The second is the classification phase where each new, previously unseen, document is classified or has its label predicted.

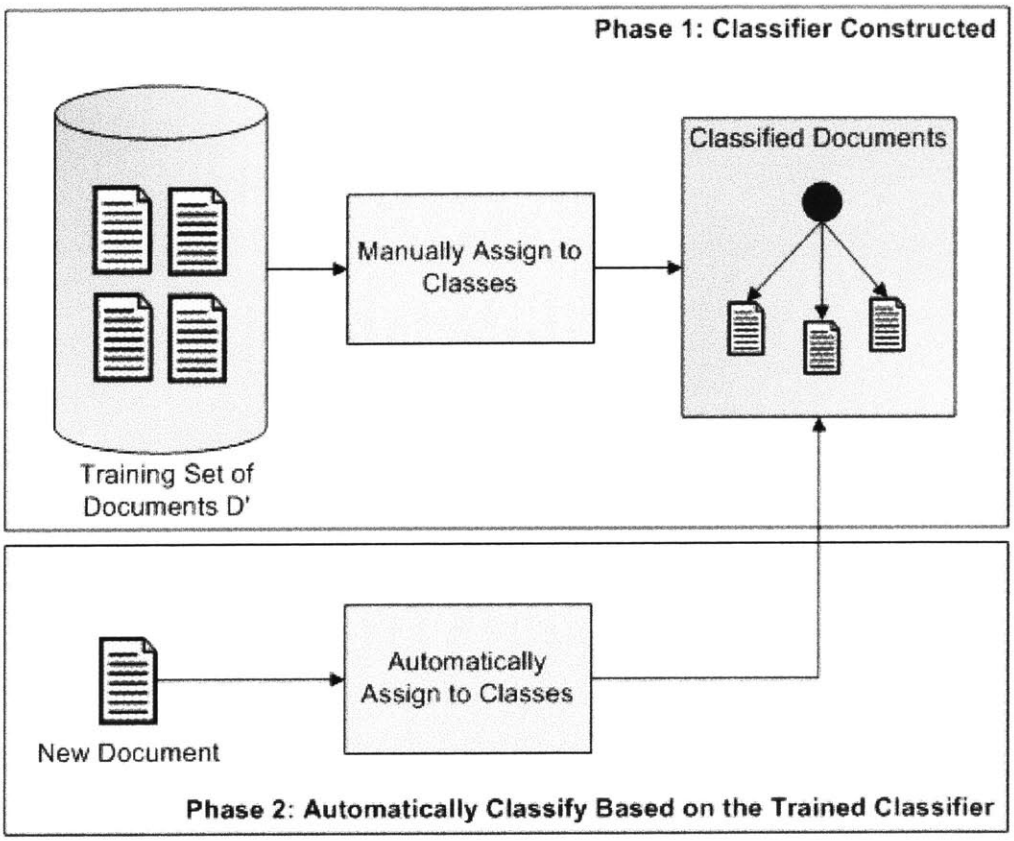


Figure 3 - The two phases of automatic classification

It is worth mentioning that the first, building, phase of the classifier is preceded by a pre-processing phase that transforms text into a format appropriate as input for machine learning algorithms. Many techniques are used to represent the information contained in the documents to be classified in a suitable format. In the case of probabilistic classification discussed later in this chapter, we will be using the vector space model to represent emails in a way suitable to construct and use a Bayesian classifier.

A lot of efforts have been versed into using machine learning algorithms to perform text classification. In summary, these algorithms learn to classify new documents based on their textual content after being trained with a set of manually classified documents. Applied to emails, algorithms of this kind have been used to thread e-mail ([12]), classify e-mail into folders ([3]) and filter junk mail ([18]).

In the following two sections, we will describe both Bayesian and rules-based classification schemes that were used in our proposed mining architectures to classify emails sent to mailing lists.

3.2 Bayesian Classification

This section will detail a Bayesian classifier based on the one proposed in [19].

3.2.1 Introduction to Bayesian Classification

Bayesian classification is a form of probabilistic classification that uses the formalism of Bayesian networks. The idea behind probabilistic classification is that our model of the world is represented as a probability distribution over the space of possible states of the world. Such states can be identified as a set of random variables where each state represents a certain assignment of this set. For the sake of conciseness, the benefits of probabilistic classification compared to other methods won't be enumerated. These are well described in [16].

As depicted in [16], probability theory can be used as a foundation for common sense reasoning by finding a way to represent a complete probability distribution over possible world histories. Since a Bayesian network models the causal structure of a nondeterministic process, it is well suited to represent this probability distribution.

As a primer to Bayesian classification, we will first explain how Bayesian networks model these causal relationships. A Bayesian network is a directed graph with nodes and edges. The nodes can be described as events and the edges as the causal relationship. The following example, taken from [13], illustrates a simple Bayesian Network.

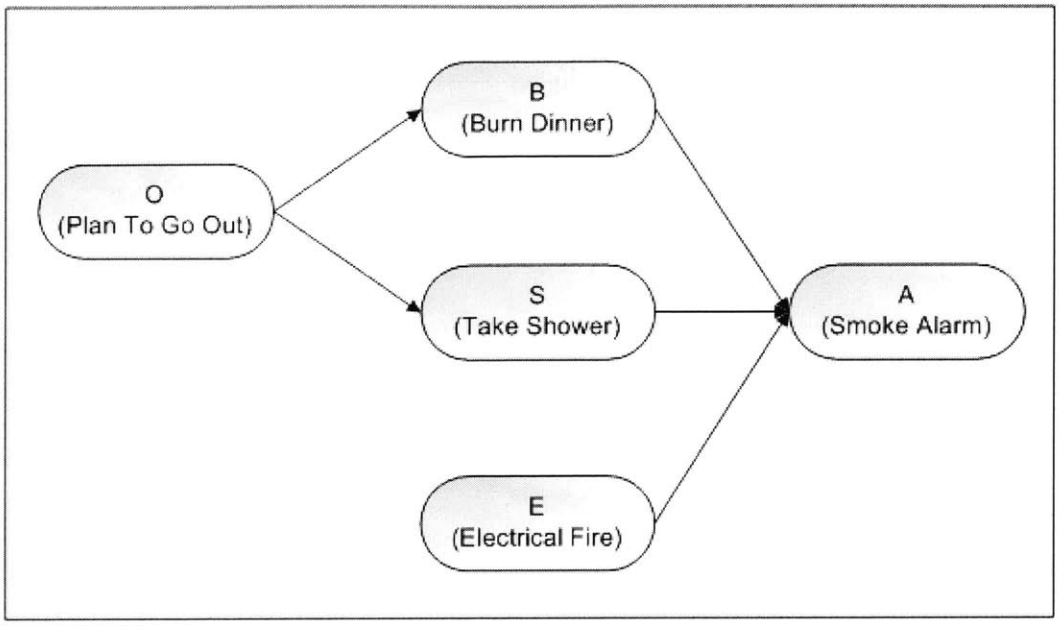


Figure 4 - A simple Bayesian network

This Bayesian network illustrates how different events are inter-related in terms of one influencing the other. For example, taking a shower (S) might trigger the smoke alarm (A) due to steam. Another event that might influence the occurrence of (A) is burning a dinner (B). In this scenario, we can see how each event can be a random variable that can be either true or false (i.e. the smoke alarm goes on or not). The simultaneous assignment of the random variables represented by this network is in fact a probability distribution function. This network models the constraints on this distribution. The "*true*" value of any variable represented in this network is influenced by the truth of the variables having a directed edge pointing to it. In our case we can see how the triggering of the smoke alarm (A=true) is dependent on the truth of (B), (S) and (E). We can describe this by having the conditional probabilities $P(A/B)$, $P(A/S)$ and $P(A/E)$. Also note that this graph suggests an independence between the events (O) and (E) (meaning $P(O \wedge E) = P(O) * P(E)$), however it does not imply that (B) is independent of (S).

We can now define a Bayesian network as follows:

A Bayesian network is a directed acyclic graph where each node is a random variable X with a finite set of possible values $\{x_1,...,x_n\}$ and each node X is associated probability matrix M^X which gives the probability that $X = x$ for each possible value X and each assignment of values to the variables at the origin of arcs leading to X. The notation $M^X_{x/y_1,...,y_k}$ is used to represent the entry in the matrix M^X giving the probabilities $P(X = x / Y_1 = y_1 \wedge Y_2 = y_2 \wedge ... \wedge Y_k = y_k)$ where $Y_1,...,Y_k$ are all the nodes at the origin of arcs coming into the node X ([13]).

In order to understand this notation, and going back to the smoke alarm network, the matrix M^B associated with node B has four elements representing the four possible assignments of (B) and (O) (since (O) is the only node that has a directed edge to (B)). These can be written as $P(B = T / O = F)$ (or $M^B_{T/F}$), $P(B = T / O = T)$ (or $M^B_{T/T}$), $P(B = F / O = F)$ (or $M^B_{F/F}$) and $P(B = F / O = T)$ (or $M^B_{F/T}$). Also note that in this matrix, $M^B_{T/F} + M^B_{T/T} = 1$.

Later in this chapter and based on this Bayesian network modeling of common sense reasoning, we will detail how, in the case of text classification a Bayesian network can be formulated.

3.2.2 Document Representation and Pre-Processing

In order to be able to apply machine learning algorithms to the task of text classification, we need to represent the text documents in a manner suitable for input to these algorithms. In the case of Bayesian classification, documents should be represented in a way to fit probabilistic reasoning.

For the purpose of classifying emails, we chose to employ a vector space representation of the input documents. This representation consists of casting textual

documents as vectors with a very high dimension. However, working in high dimensional space can limit robustness and computational speed of probabilistic classification models. For this reason, techniques such as stop words removal or zipf's law were used to reduce dimensionality.

From Document to Vector Space

The vector space representation, described in [20], consists of having each document identified by a numerical or Boolean vector. The dimensions in this vector correspond to a set of tokens or features that are formulated out of our entire corpus of documents. The tokens can be words, a combination of words or any set of characters that we define as token in our tokenization scheme. In simpler terms, if we define tokens to be the "space" separated words in a certain document, our vector's dimension components, or feature set, will be all the words present in our set of documents.

The values attributed to each dimension can vary with different vector space representations. For example, we can have a numerical vector representation where each value corresponds to the number of times the given token appears in the document. This assignment is dubbed simple term frequency representation. Another representation would be the Boolean vector one where the values are set to "*one*" if the token is present in the document, and "*zero*" if it is not. Note that in this case, the number of times the token appears in the document is not reflected in the representation. One might think that this would lead to decreased classifier performance and accuracy due to missing part of the information provided by the document's content. However, as shown by [23], a comparison between the Boolean and simple term frequencies vector representations yielded little difference in results with the former being a lot easier to implement. For this purpose, the Bayesian classification used by the subsequently proposed mining architectures will apply Boolean vector representation for its documents.

Vector space representation can thus be defined as follows:

Having a textual document $d \in D$, the representation of d is the vector $X(d) = \vec{x} = (x_1,...,x_n)^T$, where each dimension corresponds to a distinct token in the document collection and x_i denotes the weight or value of the i[th] term. The set of these n index terms, $V = \{t_1,...,t_n\}$, is referred to as the feature set. In the case of Boolean vector representation, $x_i \in \{0,1\} \forall i$ [11].

To better visualize this concept, the following figure shows a corpus of three text documents and their corresponding vector representations. In this case, tokens are the "*space*" separated words present in the documents.

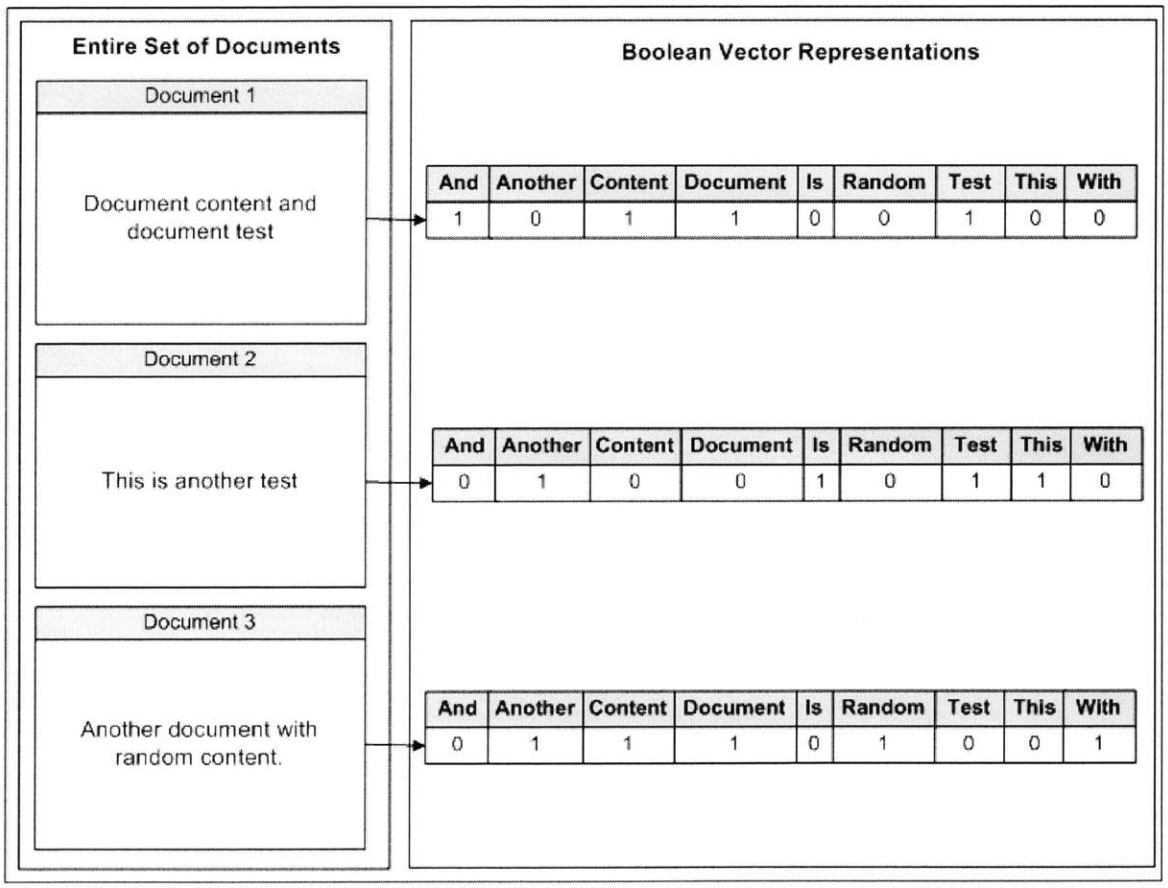


Figure 5 - Boolean vector representation of a set of textual documents

Tokenizing the Input Document

As mentioned previously, tokenizing a document is the first step to move from textual content to a vector format suitable as input to machine learning algorithms. The process of parsing documents for tokens will build our initial feature set and will determine the vector representation of any new document fed for classification. Tokenization can greatly increase the performance of the classifier by taking context specific content into account. For example, in the case of email classification, a word present in the subject of the message may have a greater importance than the same word appearing in the body. Although we will look into these context-specific features in a later section of this chapter, we need to define the basic operations that our parser will undertake in order to reach the vector representation needed for classification.

The classifier covered by this thesis uses emails as input documents. Emails are differentiated from text-only documents in two aspects. The first one is the content organization where text is divided into Email Date, From Email, To Email, Subject and Body. And the second is the fact that Emails can include HTML for text formatting. For this purpose, the tokenization scheme should account for these differentiations, and a clear decision should be made on which specific features to include in our feature set.

There is always a tradeoff between how much to tokenize compared to the added accuracy we are getting out of it. In the case implementation, we tried to limit the feature set and not depend on attributes that appear in a portion of the input documents. Since only a part of the emails had HTML formatting, we opted not to include this information in the tokenization process. Having this in mind, the following parsing procedure was used to extract the set of tokens out of input documents:

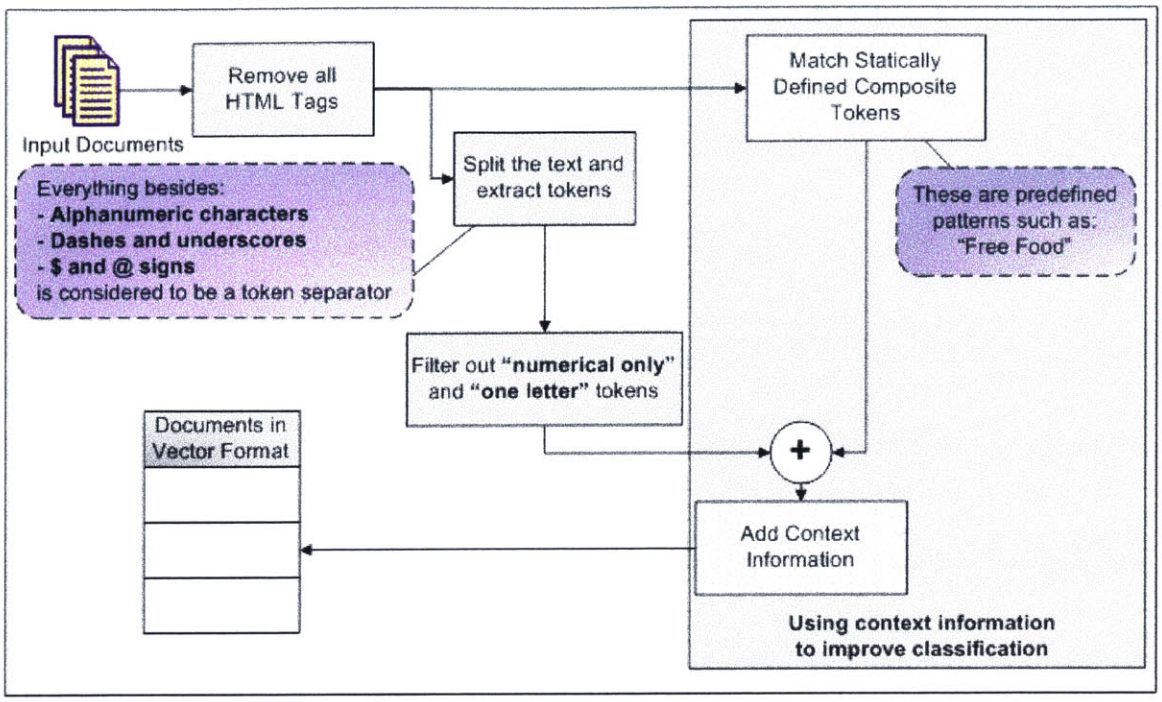


Figure 6 - The tokenization process

In summary, this process first involves cleaning the input documents from all the HTML tags and changing all the characters to lower case. All alphanumerical characters, dashes, underscores, "*dollar*" and "*at*" signs are considered part of tokens and the document is split over the remaining characters. In parallel, the document is matched for predefined patterns formulating tokens that are specific to our classification task. Tokens made of one letter or that consist of numerical characters only are then dropped. And finally, the tokens are added to a single hash table and context information is appended to differentiate the occurrence of a token in different parts of the Email or input document.

Please note that the elaboration on the context-specific features will be exposed later in this chapter.

Reducing Dimensionality

In the vector space representation of documents, dimensions in the order of 10^3 to 10^5 ([19]) can be reached for a relatively small corpus of documents. Reducing this feature set can have many benefits in the case of naïve Bayesian classification (discussed in the next section). As described by [18], reducing dimensionality helps provide explicit control over the model variance resulting from estimating many parameters. And in the case of naïve Bayesian classification, it helps attenuate the effect of the assumed independence between features in the vector space.

The classification scheme presented in this thesis uses mainly two techniques to reduce dimensionality. These are stop word removal and zipf's law.

Stop Word Removal

Stop word removal is a simple step that consists of removing all the words that contribute little to the semantic meaning of the document. An example of the set of words that the classifier covered by this thesis considers as stop words is:

An	be	Each	if	last	near
That	about	But	else	in	late
No	the	All	by	is	like
They	most	For	it	of	to
And	did	From	into	many	often
Are	do	further	much	on	with
As	down	More	once	which	at
During	get	Just	must	or	whether

Table 3 - Sample list of stop words

Features Selector (Zipf's Law)

Zipf's law ([25]) states that, on average, the n^{th} most frequent word will occur $\frac{1}{n}$ times the frequency of the most frequent word in a corpus. This means that most of the words present in the corpus appear rather infrequently. Since rare words might have excessive discriminatory value, removing them will both reduce dimensionality and improve classification. For this reason, in the classifier covered by this thesis, all words that appear three times or less in the entire corpus are removed from the feature set. This part of documents pre-processing is identified as the features selector.

There are many other ways to reduce dimensionality and possibly improve performance (i.e. stemming, inverse document frequency ...). However, in the work presented in this thesis, only these two methods were used.

Summary of the Pre-Processing Steps

The following diagram illustrates all the steps taken to prepare the input documents for classification:

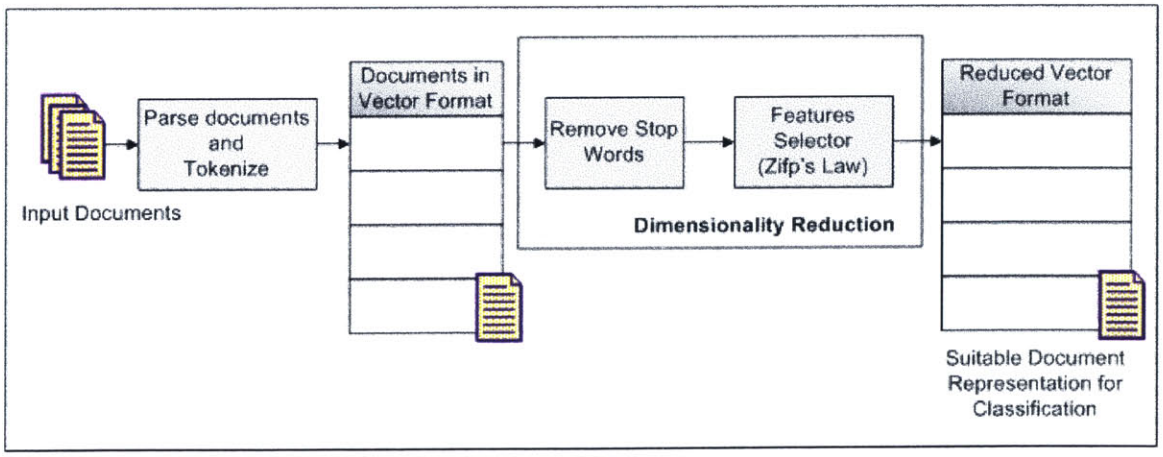


Figure 7 - Pre-processing steps for adequate document representation

3.2.3 The Naïve-Bayes Classification

Bayesian classification is a Bayesian network applied to a classification task. This network has a node C for the class variable where:

$$C_k \in \{\text{"Belongs to Class } \kappa\text{", "Does not belong to Class } \kappa\text{"}\} \forall\ k \quad \textbf{(3.1)}$$

We will refer to *"Belongs to Class κ"* as κ and to *"Does not belong to Class κ"* as $!\kappa$.

As detailed previously, each document can be represented in the form of a high-dimensional vector $\vec{x} = (x_1,...,x_n)$ where $x_1,...,x_n$ are the values of the attributes $X_1,...,X_n$. These attributes or features, are the ones that remained after dimensionality reduction. They are represented in the Bayesian network as a set of nodes X_i. Since we are set on using Boolean vector representation, the values x_i attributed to these nodes are bound to:

$$x_i \in \{0,1\} \forall\ i \quad \textbf{(3.2)}$$

A realistic network representation of the classification would be:

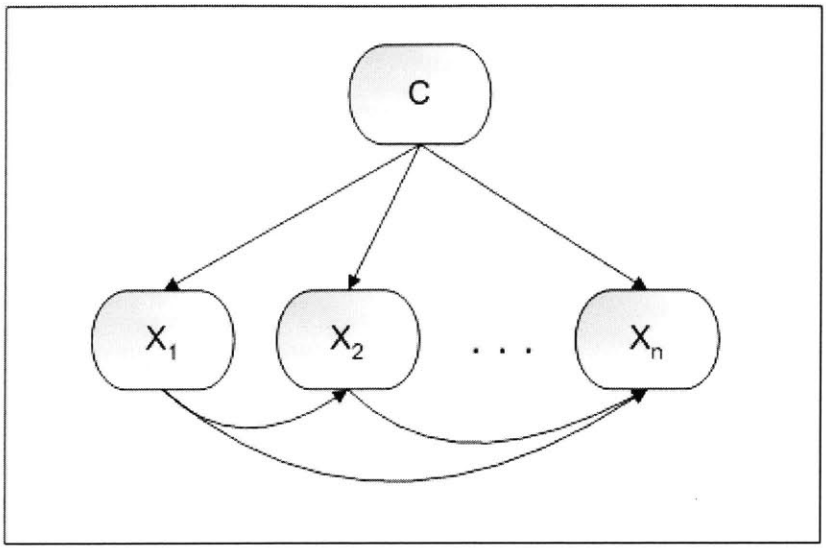


Figure 8 - Network representing a realistic Bayesian classifier

This network models the dependence of each of the features of a certain document on a given class as well as the limited dependencies between the features.

From Bayes' theorem, given a document d represented by the vector $\vec{x} = (x_1,...,x_n)$, the probability for d to belong to a certain class $C_k = c \in \{\kappa, !\kappa\}$ is:

$$P\left(C = c / \overrightarrow{X} = \vec{x}\right) = \frac{P(\overrightarrow{X} = \vec{x} / C = c) * P(C = c)}{P(\overrightarrow{X} = \vec{x})} \quad \textbf{(3.3)}$$

The theorem of total probability yields:

$$P(\overrightarrow{X} = \vec{x}) = \sum_k P(\overrightarrow{X} = \vec{x} / C = C_k) * P(C = C_k) \quad \textbf{(3.4)}$$

This transforms equation 3.3 into:

$$P\left(C = c / \vec{X} = \vec{x}\right) = \frac{P(\vec{X} = \vec{x} / C = c) * P(C = c)}{\sum_k P(\vec{X} = \vec{x} / C = C_k) * P(C = C_k)} \quad \textbf{(3.5)}$$

In this equation, computing $P(\vec{X} = \vec{x} / C = c)$ can be extremely difficult without assuming independence between the features X_i of $\vec{X}$. The independence of these features is what makes the classifier a Naïve-Bayesian one ([8]). This assumption can be described in this updated Bayesian network:

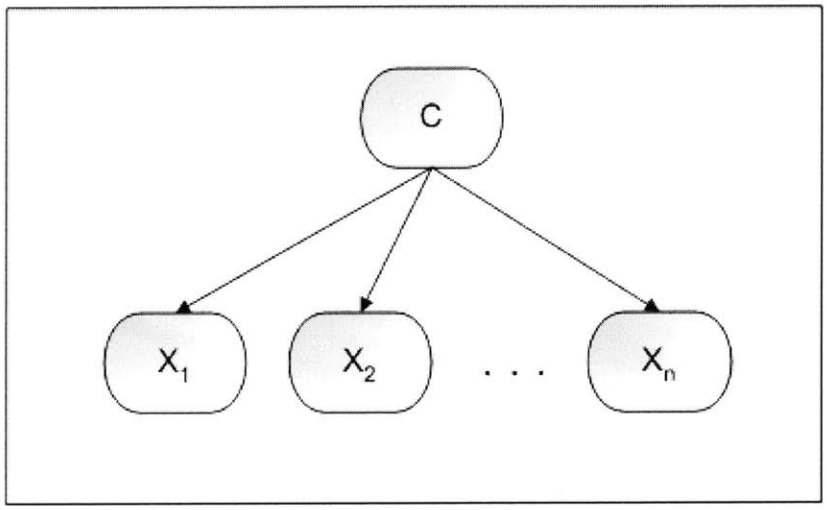


Figure 9 - Network representing a Naïve-Bayes Classifier

With this assumption, we can now compute $P(\vec{X} = \vec{x} / C = c)$ as:

$$P\left(\vec{X} = \vec{x} / C = c\right) = \prod_i P(X_i = x_i / C = c) \quad \textbf{(3.6)}$$

Both (3.5) and (3.6) lead to:

$$P\left(C = c / \vec{X} = \vec{x}\right) = \frac{\left(\prod_i P(X_i = x_i / C = c)\right) * P(C = c)}{\sum_k \left(\prod_i P(X_i = x_i / C = C_k)\right) * P(C = C_k)} \quad \textbf{(3.7)}$$

Which is the probability of a certain document d represented by the vector $\vec{x}$ to belong to a class c. Note that in this equation, $P(C = c)$ and $P(X_i = x_i / C = c)$ can be computed as term frequency ratios from the training set of documents. For example, for a given token A:

$$P(X_i = A / C = \kappa) = \frac{N_{A/\kappa}}{N_\kappa} \quad \textbf{(3.8)}$$

Where N_k is the total number of tokens in the class κ and $N_{A/\kappa}$ is the total number of occurrences of token A in the documents manually labeled as belonging to κ.

We can now decide whether each document belongs to κ or $!\kappa$ by comparing $P(\vec{X} = \vec{x} / C = \kappa)$ and $P(\vec{X} = \vec{x} / C = !\kappa)$. In this case, having only two possible values for the class node, for any given document:

$$P(\vec{X} = \vec{x} / C = \kappa) + P(\vec{X} = \vec{x} / C = !\kappa) = 1 \quad \textbf{(3.9)}$$

For the purpose of our application, the gravity of erroneously classifying a document is evenly distributed over our classes. Along with equation (3.9), this translates into the following conditions for classification:

If: $P(\vec{X} = \vec{x} / C = \kappa) > \alpha$ (where $0 < \alpha < 1$ is the threshold for classification)
the document represented by $\vec{x}$ is classified as κ
Else: the document represented by $\vec{x}$ is classified as $!\kappa$

Table 4 - Conditions for classification

The threshold α can be changed according to the proportion of documents belonging to κ and those that don't belong to κ. In the case implementation α was initially set to 0.5 then changed to higher values to improve accuracy.

3.2.4 Context Specific Tokenization

Despite the fact that the independence assumption is over simplistic, Naïve-Bayes classification performs surprisingly well as shown by ([10] and [6]). However, the fact still remains that we are missing part of the information that is provided by the document. One approach to attenuate the effects of assumed independence and increase accuracy is to add context to the tokenization process.

The first way to add context information is to manually defined patterns that can have a substantial effect on the class of the document. For example, in the case of FreeFood@MIT, having the pattern "Free Food" included in the list of features or tokens can influence classification much more than only relying on "Free" and "Food" being treated as separate tokens. This would improve the vector space representation by adding yet another hand-crafted, meaningful dimension to our classification task. Note that these patterns are relevant to the specific classification task.

Since we are using Emails as documents to classify. It makes sense, for the sake of increasing accuracy, to use the specific structure of Emails as an added source of information. As stated previously in this chapter, having a given word or token in the Subject of the Email can yield much higher influence than having the same word in the

Body. For this reason, it makes sense to include the logical partition in which tokens were recorded to create a new set of tokens. The way this was implemented in the work covered by this thesis is to explicitly add the location where the token was found if it was different than the Body.

For example, if the term "food" was found in the Subject of the Email, the token representing it would be "Subject:food". If the same term was found in the Body, it would be represented by the token "food". Another example are the To and From Email fields. These are tokenized as "From:email@address" or "To:email@address".

3.3 Rules-Based Classification

The rules-based classification covered by this thesis consists of a set of rules that are hand crafted to fit the needs of the case implementation, FreeFood@MIT. Therefore, we will only briefly cover this type of classification.

3.3.1 Introduction to Rules-Based Classification

Rules-based classification is based on a simple principle: each classifier is identified by a set of keyword-spotting rules. If all the keywords in a rule are found in a certain document the conclusion is drawn. As described by [2], these set of rules can be interpreted as a disjunction of conjunctions. For example, a document d is considered to be in class κ (i.e. seminar) if and only if:

(word "seminar" appear in d) OR
(word "speaker" AND word "location" appear in d) OR
(word "location" AND word "time" AND word "speaker" appear in d) OR
.
.
.
(word "presents" AND word "abstract" appear in d) OR
(word "speaker" AND word "abstract" appear in d) OR
(word "lecture" appear in d)

Table 5 - Sample rules set for rules-based classification

Similar to Bayesian classification where learning plays a major role, rules-based classification can use a training set to build its rule set automatically. For example, a program called RIPPER, described in [2] and [4], can be used to obtain keyword-spotting rules. The way RIPPER works is that it keeps on greedily adding rules to an empty rule set until all positive examples are covered. The training data is first split into a "growing set" and a "pruning set". The former set is used to greedily grow the rule set, and the latter is used to simplify the formed rule by greedily deleting conditions so as to improve the rule's performance. Cohen reported that the rules generated by RIPPER have similar accuracy as manually generated rules.

3.3.2 Implementation in the Proposed Mining Approaches

For our case implementation, we didn't use any algorithm to build the rules from a training set. We rather crafted a rule set manually. However, in [3], we can see that RIPPER can be effectively used for email classification. In this case, rules not only work on the keywords or terms present in a document but also on their location in the document. For example, here is a set of handcrafted rules that are used in the context of emails: An email d is considered to be in class κ (i.e. freefood) if and only if:

(word "food" in field "Subject:" of d) OR
(word "food" AND word "free" in d) OR
(word "pizza" in d) OR
.
.
.
(word "free-food@mit.edu" in "To:" field of d) OR
(word "bertucci" in d)

Table 6 - Sample rules set for rules-based classification of emails

The tokenization techniques used for Bayesian classification can also be used in the case of rules-based classification. However, it is much simpler in this case since there is no assumed independence between the tokens in the document. Tokenization is thus limited to the first phase where the set of features or keywords are extracted.

Next section will describe the common performance measures that will be used to evaluate both approaches to mine events from MIT mailing lists.

3.4 Classification Performance Evaluation

In order to compare the performance of each of the two considered classification approaches, we need to define the common evaluation measures that will be used.

Lets consider we have two possible outputs for our classifier and these are whether the document is labeled as belonging or not to a class κ. The framework for performance evaluation is set by a corpus of manually sorted documents. These are divided into a training set and a testing set. The testing set's automatic classification is compared to the manual classification to compute specific performance attributes such as accuracy or error.

The following table illustrates the partitioning of our test corpus of documents after classification:

	Documents belonging to κ	**Documents not belonging to κ**
Documents Classified as Belonging to κ	$N_{\kappa \to \kappa}$	$N_{!\kappa \to \kappa}$
Documents Classified as not Belonging to κ	$N_{\kappa \to !\kappa}$	$N_{!\kappa \to !\kappa}$

Table 7 - Performance evaluation measures

The top header represents what we consider as "real class belonging" of the test documents. This belonging is known beforehand through the manual classification we performed on these documents.

The symbol $N_{\kappa \to !k}$ can be read as the number of documents that belong to κ but that have been classified as not belonging to κ (or belonging to $!\kappa$).

The total number of test documents that were automatically classified to measure performance is defined as:

$$N = N_{\kappa \to \kappa} + N_{!\kappa \to !\kappa} + N_{\kappa \to !\kappa} + N_{!\kappa \to \kappa} \quad \textbf{(3.10)}$$

We can now define the following performance measures:

- **Recall** is the ratio of documents correctly classified as belonging to κ over all the documents that actually belong to κ:

$$R = \frac{N_{\kappa \to \kappa}}{N_{\kappa \to \kappa} + N_{\kappa \to !\kappa}} \quad (\text{If } N_{\kappa\to\kappa} + N_{\kappa\to!\kappa} > 0) \textbf{ (3.11)}$$

- **Precision** is the ratio of documents correctly classified as belonging to κ over all the document that were classified as belonging to κ:

$$P = \frac{N_{\kappa \to \kappa}}{N_{\kappa \to \kappa} + N_{!\kappa \to \kappa}} \quad (\text{If } N_{\kappa\to\kappa} + N_{!\kappa\to\kappa} > 0) \textbf{ (3.12)}$$

- **Fallout** is the ratio of documents erroneously classified as belonging to κ over all the document that actually do not belong to κ:

$$F = \frac{N_{!\kappa \to \kappa}}{N_{!\kappa \to \kappa} + N_{!\kappa \to !\kappa}} \quad (\text{If } N_{!\kappa\to\kappa} + N_{!\kappa\to!\kappa} > 0) \textbf{ (3.13)}$$

- **Accuracy Rate** is the ratio of documents correctly classified over all the test documents:

$$AR = \frac{N_{\kappa \to \kappa} + N_{!\kappa \to !\kappa}}{N} \textbf{ (3.14)}$$

- **Error Rate** is the ratio of documents erroneously classified over all the test documents:

$$ER = \frac{N_{\kappa \to !\kappa} + N_{!\kappa \to \kappa}}{N} \textbf{ (3.15)}$$

These performance evaluation measures can be placed in an evaluation table in the following format:

	Recall (R)	Precision (P)	Fallout (F)	Accuracy (AR)	Error (ER)
Class K	98.3%	86.9%	1.9%	98%	1.9%
Class $!K$	98%	99.7%	0.8%	98%	1.9%

Table 8 - Evaluation table with sample values

From this table, the most important figures are the accuracy and error rates. These would give us clear standpoints for general performance evaluation. The other values can be used in case we consider the gravity of misclassification as uneven between the classes.

The results of the classification approaches as well as their interpretation are the subject of chapter 5.

Chapter 4
The Proposed Mining Approaches

As described in the introduction, the purpose of this thesis is to describe a system that mines MIT mailing lists for events. After illustrating how to retrieve the messages broadcasted over public mailing lists in chapter 2 and depicting the classification methods that can be used to filter events in chapter 3, this chapter will expose the entire framework of the mining system. Two architectures are the subject of this chapter. The first one is centered on the rules-based classification method, and the second on the Bayesian classification method.

4.1 The Rules-Based Proposed Approach

4.1.1 Overview of this Approach

The purpose of the architectures proposed in this chapter is to mine the messages broadcasted over public MIT mailing lists for events. This is done through classification or information filtering as described in the previous chapter. Such a mining architecture can be used as a topic-specific mailing lists search engine where, for example, a user can search for all the seminars publicized by Email at MIT. In other terms, this architecture is supposed to retrieve the messages from the mailing lists, classify them, retrieve event related information such as the date and store the relevant messages in a database. This database is then exposed through a web interface that provides searching capabilities to end users. In the case of this first proposal, the core of the system is the rules-based classification method described in Chapter 3. The following two sections will detail this architecture and its various components and provide a guided walkthrough of how it works.

4.1.2 Proposed System Architecture

From mailing list Emails to a searchable database of topic-specific events lay many components. Some of these were described in the previous chapters. These can be summarized as follows for this rules-based approach:

Public Emails Retrieval

The first component of this mining architecture is to retrieve the Emails sent over public MIT mailing lists. These are considered to be a real-time source of information. As described in chapter 2, this component consists of having the system regularly check a specific Email that is automatically added to all the public mailing lists. The retrieved documents are then treated as unit inputs to the classification component.

Rules-based Classification

After retrieving the Emails broadcasted on the mailing lists, the system filters them based on a topic-specific categorization such as whether the Emails describe a seminar or not. This component uses rules-based classification where emails are treated as documents. After doing some pre-processing on the documents such as cleaning their HTML content and performing tokenization, they are matched for specific keywords based on a set of rules as described in chapter 3. Notice that in the case implementation covered by this thesis; these sets were hand-crafted to fit the topic-specific classification. Methods to build rules set automatically, such as the one described in [2] and [4], were not used. The Emails that do adhere to the classification criteria of this component are then passed to the information extraction engine.

Information Extraction

This component handles extracting labeled information from the documents that made it through classification. Labeled information is highly dependent on the topic that

this architecture would be handling. In the case of seminars, retrieving the *date, time, location, speaker* and *subject* is the type of needed labeling. However since this architecture deals with exposing any kind of events, the only considered information was the event date. Therefore, this component reads each document and tries to estimate the correct date of the event the document or Email describes. After information extraction, the Email is stored in a database along with the labeled data to be exposed to the end users who want to search for topic-specific events.

Web Interface

After the broadcasted emails are labeled and stored in a database, the final component of this architecture is to expose the regularly updated database of events to the end user. This web interface implements searching functionalities where users can search by keyword the entire contents of the stored Emails as well as the labeled data. For example, since the event date is estimated by the system, the user can search for all the topic-specific events that are scheduled for a specific date. Furthermore, the user can give feedback on the event stating whether it is recommended, not recommended or is not related to the topic of classification. In this case, if an event amasses a consistent amount of error feedbacks, it is cleared from the database. This mining architecture does not improve classification based on feedback since the rules set are defined manually.

System Architecture

The framework in which all these components fit can be described as follows:

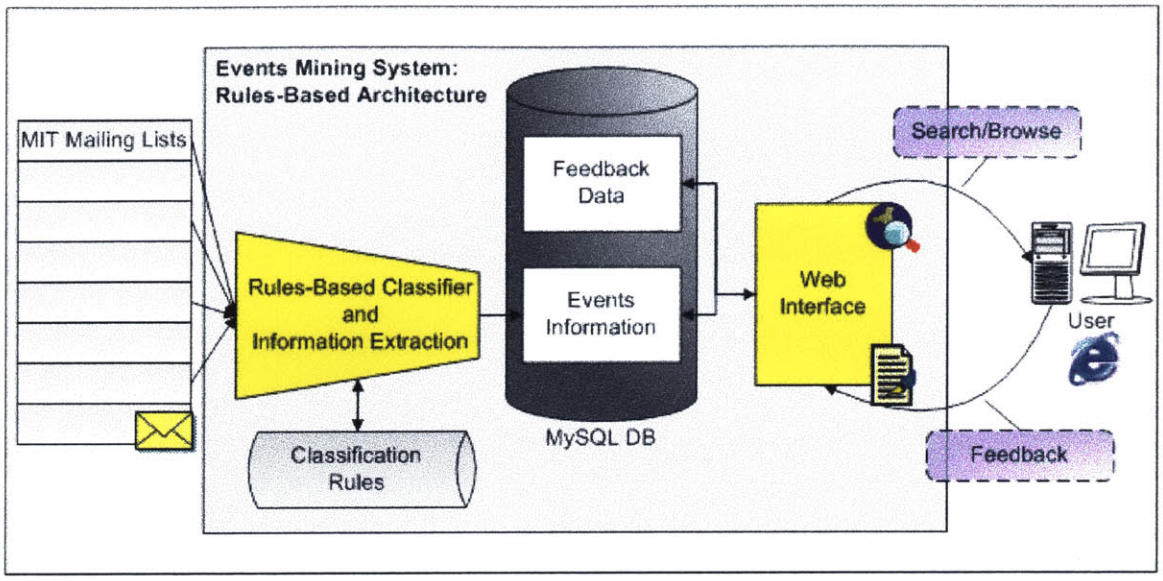


Figure 10 - Rules-based system architecture

This system chart illustrates the entire framework of this mining architecture. The next section will detail the system processes involved in this architecture.

4.1.3 System Operation

The main classification process involved in this system is described as follows:

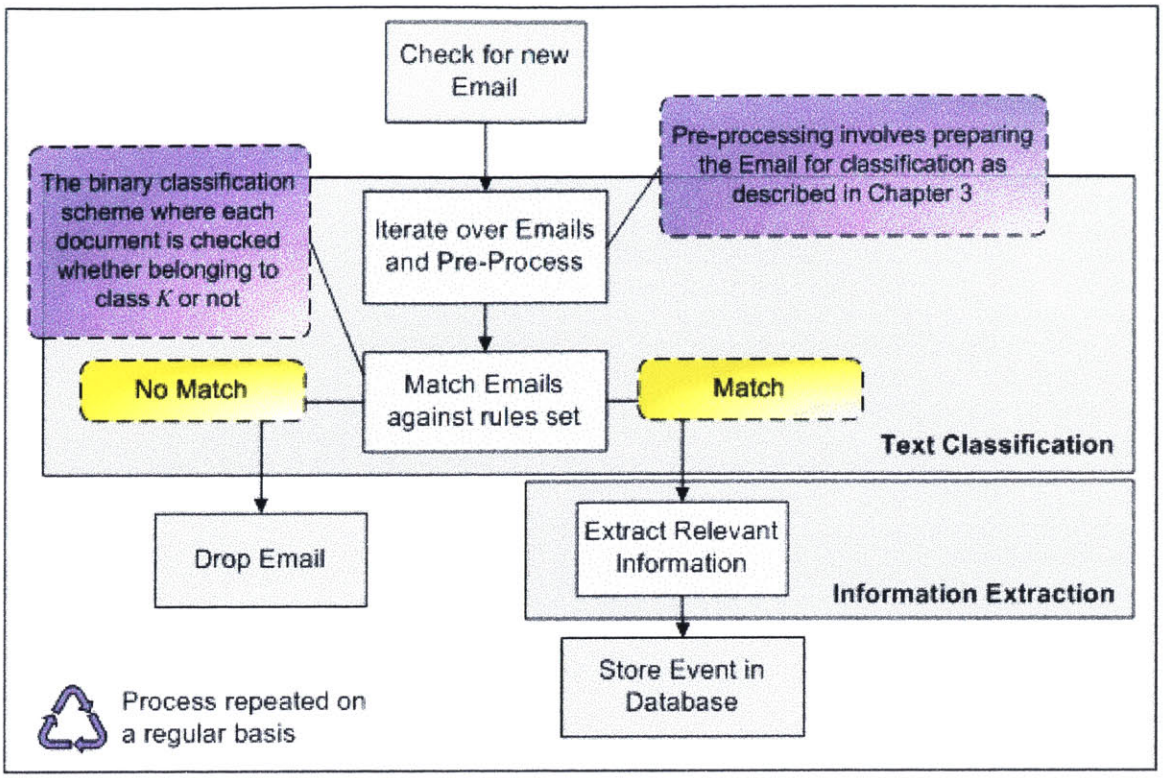


Figure 11 - Rules-based classification process

Putting it in words, the system first checks for new Emails sent to the public MIT mailing lists. Then it iterates over them, removes all HTML content, tokenizes if need be and feeds the prepared documents to the classifier. Based on the rules set, the Email gets classified as belonging to the topic-specific class of events or not. If it belongs to this class, the date of the event is estimated and all the information is stored in the database. This system process is repeated on a regular basis. In our case implementation, this process was modeled by a batch script that was scheduled to run daily on the server.

The other process involved in this architecture is the user process. This one is too simple to be modeled by a flowchart. It consists of the user searching/browsing for events and giving feedback.

4.2 The Bayesian Proposed Approach

4.2.1 Overview of this Approach

This approach is identical to the rules-based in many aspects. The difference lies in the core classification engine. The Bayesian based architecture has at its center a naïve Bayes classifier that is trained on a set of previously labeled Emails. These Emails constitute the training set.

Since the accuracy of the Bayesian classifier is highly influenced by the size of our training corpus ([21]), it is essential to have a large number of labeled Emails pertaining to our topic-specific classification task. For example, if the events mining system is to be used to find seminars, we should have a large number of emails that we manually identified as seminars. Since the system can be replicated to support many kinds of events, this can become an expensive drawback. Having to manually classify thousands of Emails for each new topic we want the system to cover is not an easy task.

In these regards, this architecture uses the user feedback to improve classification by regularly updating the training set. For example, if a newly received Email was classified as describing a seminar and users consistently report an error feedback, the training set gets eventually updated by adding this Email to the category of not describing a seminar. Alternatively, if users consistently report a non-erroneous feedback (recommended or non-recommended), the Email gets labeled as describing a seminar. Using this process makes the system's accuracy in detecting topic-specific events increase over time with more user feedbacks being accrued. And it also reduces the need to have a large initial training set which can be handicapping to build.

4.2.2 Proposed System Architecture

This system architecture is differentiated by three components from the one described in the previous section. The first one is the naïve Bayes classifier that is now

constructed using supervised learning as described in chapter 3. The second is the training engine that constructs the classifier. And the third is the feedback analyzer that updates the classifier's training corpus to improve accuracy over time.

Naïve Bayes Classifier

The naïve Bayes classifier used in this architecture is the one described in chapter 3. It includes all the pre-processing steps such as the vector space representation of documents and the features set reduction. It is constructed based on an initial training corpus of manually labeled Emails. Unlike the previous architecture where the classifier was manually built through a specific set of rules pertaining to a certain topic of events, this architecture supports an easy replication process. By replication we mean that the same system can be used to support many classes of events such as seminars, lectures, free food... This process can be conducted by providing the system with a new database of manually labeled Emails. The constructed naïve Bayes classifier is represented by a table in the system database with the probabilities of every token in the feature set. In order for the classifier to compute the probability of a new Email to belong to a given class, it uses these class dependent token probabilities.

Training Engine

The task of the training engine is to populate the database with the class dependent probabilities of tokens, or in other terms, it constructs the Bayesian classifier. The probabilities are computed through token frequency ratios from the training set as described in chapter 3.

Feedback Analyzer

The feedback analyzer is the system component that updates the training set from user feedbacks to increase the accuracy of the classifier over time. This increase of accuracy will be well shown in the results of our case implementation, Freefood@MIT.

The way the feedback analyzer works is straightforward. It compares the number of times an event was reported as recommended, not recommended or being erroneously classified as belonging to the topic of searched events. Considering E is the number of times an Email is reported as an error, R the number of times it was recommended, ER the number of times it was not recommended and γ a feedback threshold that indicates the minimum acceptable number of feedbacks, the following condition must apply for an Email to be incorporated in the training corpus:

$$E + R + ER > \gamma \quad \textbf{(4.1)}$$

This condition states that if an Email doesn't have at least γ feedbacks, it cannot be used as a training document. An appropriate value for γ depends on how many users access the system, a value somewhere around ten can be appropriate. The second condition is whether to label the Email as belonging to the specific topic of events or not. Considering λ is the threshold number of error feedbacks compared to non-error feedbacks, if the following conditions apply:

$$\left(\frac{E}{R + ER} > \lambda\right) \text{ or } (R + ER = 0) \quad \textbf{(4.2)}$$

Then the Email is labeled as not belonging to the class of events in the training set and is removed from the events database since it was erroneously classified. An appropriate value of λ can be anywhere above 1. In our case implementation, it was chosen to be 1.5, or in other terms, if an Email was reported 60% of the time as being a classification error then it is removed from the database and labeled accordingly in the training set.

System Architecture

The other components of this architecture such as information extraction or public Emails retrieval are described in the rules-based architecture and remained the same. The Bayesian classification framework for the events mining system can be described as follows:

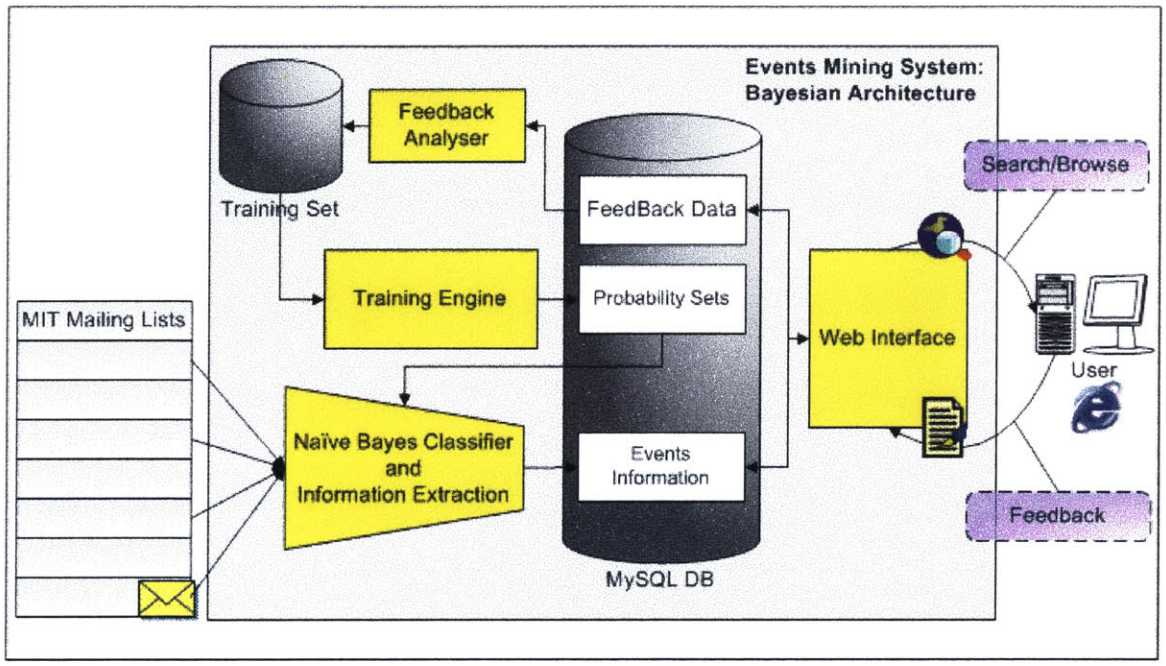


Figure 12 - Bayesian system architecture

There are three main system processes associated with this architecture: the training process, the classification process and the feedback process. They will be detailed in the next section.

4.2.3 System Operations

As mentioned previously, the Bayesian classifier is constructed through supervised learning. The following process, dubbed training process, defines how the classifier is built from the training set of Emails:

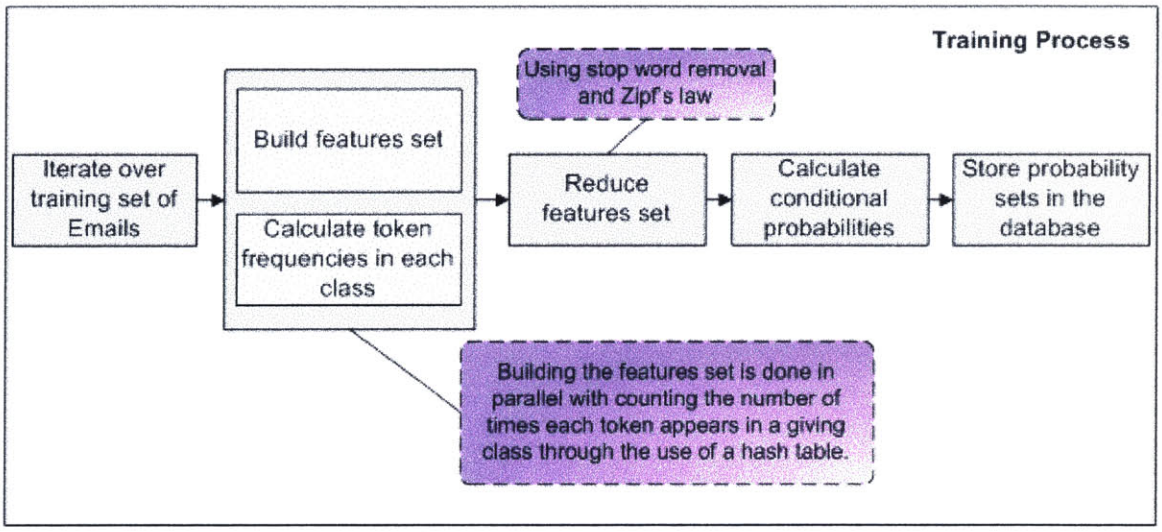


Figure 13 - Training the Bayesian classifier

After the classifier is built from a set of manually classified Emails, the system can start classifying new Emails sent over MIT public mailing lists. The classification process can be defined as follows:

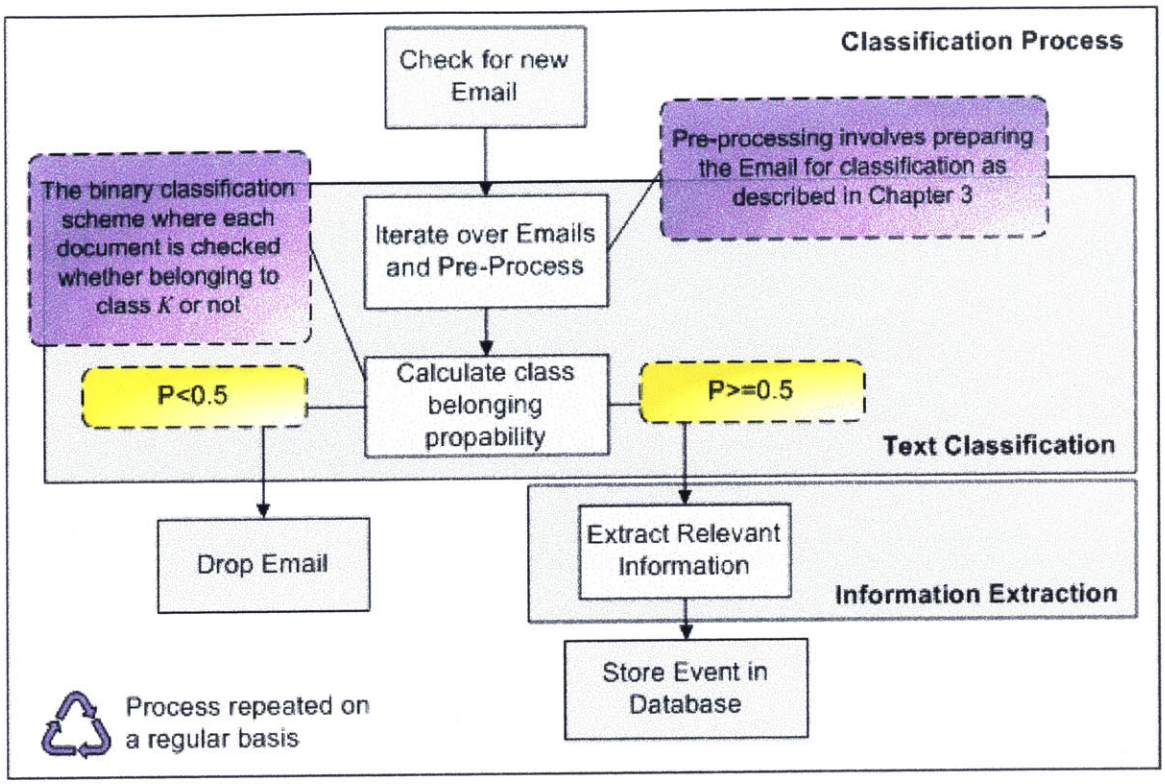


Figure 14 - Bayesian classification process

As in the rules-based approach, this process is modeled in a batch script that is repeatedly executed in order to always update the database of events with the newest Emails broadcasted over the mailing lists.

Finally, the process associated with the feedback analyzer is separate from the main classification process. This process reconstructs the Bayesian classifier by updating the training set based on user feedback and re-executes the training process described above.

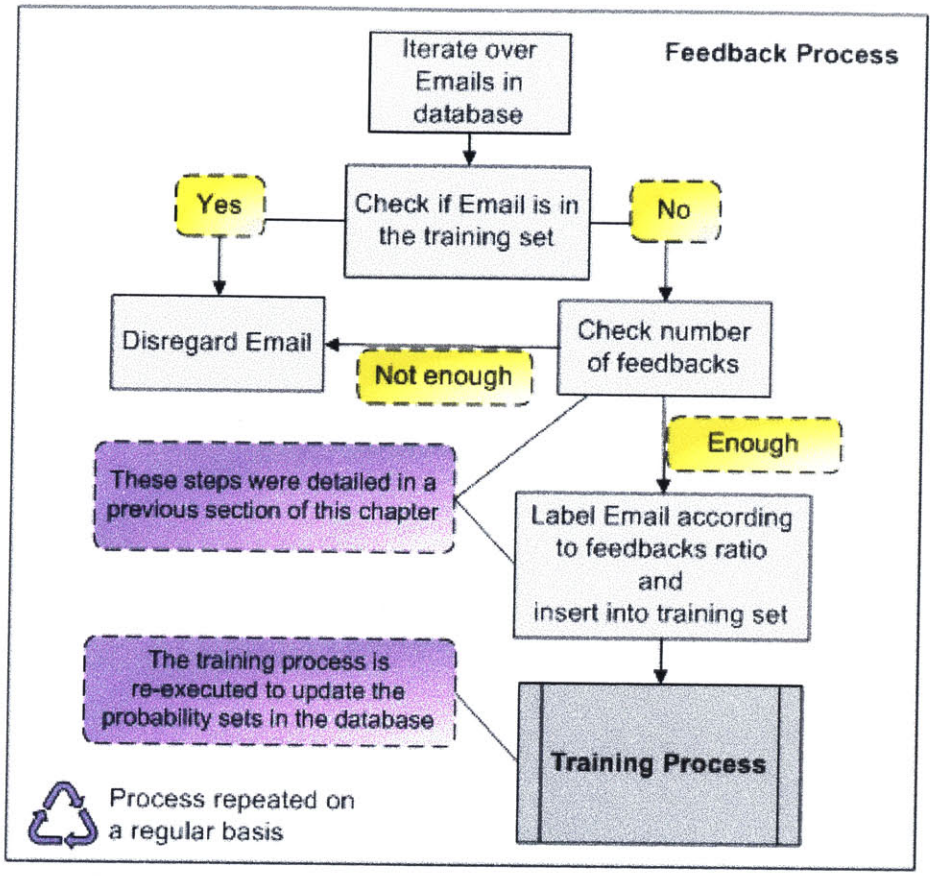


Figure 15 - Feedback process to improve classification accuracy

This process is also executed regularly in the form of a batch script. However, it is not as crucial as the classification process. It can be repeated on larger intervals, for example weekly instead of daily.

4.3 Information Extraction

The information extraction portion of the mining system hasn't been detailed previously due to its rather limited functionality. Since information extraction is highly dependent on the type of events being classified, only event date matching is implemented. This provides users with the ability to search events by date.

Matching event dates is based on a set of conditions and a set of regular expressions that captures every possible date format. The following table illustrates a sample list of dates matched by the information extraction component:

String	Matched Date
11/23/02	11/23/2002
11-23-2002	11/23/2002
23/11/2002	11/23/2002
23-11-02	1/23/2002
jan 3rd, 2002	1/3/2002
mar 2	3/2/2003
5th of april	4/5/2003
2nd of june 2003	6/2/2003
july 23, 2005	7/23/2005
6 dec 02	12/6/2002
nov 25, 2001	11/25/2001
12 13 2002	12/13/2002
7 october	10/7/2003
8th jan	1/8/2003
may 3rd	5/3/2003
aug 1st , 2001	8/1/2001
Tue, 24 Sep 2002	9/24/2002
november the 2nd, 2001	11/2/2001
mar the 1st	3/1/2003
september the 25th	9/25/2003

Table 9 - Sample list of dates matched through information extraction

Here is the procedure followed by the information extraction component to estimate the date of an event described in an Email:

Each incoming Email is matched for all the dates in its Subject and Body. If there is a date in the subject, it is considered to be the event date regardless of the other dates present in the Body (if any). In case there is no date in both Subject and Body, the date the Email was sent (available in the header) is assumed as the event date. In case there are multiple dates in the Body, and none of the above applies, the furthest away date is assumed as the event date.

As we will see in chapter 5, this relatively simple information extraction procedure yielded decent results with an acceptable accuracy rate. Accuracy refers to the portion of events that had their correct date matched.

Chapter 5
Performance & Results of a Case Implementation: FreeFood@MIT

This chapter will cover a case implementation of the events mining system. Both architectures described in the previous chapter were used and the results as well as their interpretation will be exposed. The evaluation of performance of the text classifiers will be done in accordance to the measures described in section 3.4.

5.1 Overview of FreeFood@MIT

The case implementation covered by this thesis consists of finding events that have free food servings associated with them on the MIT campus. For example, if a certain seminar or lecture is held at lunch time, and if free food will be served for people attending this event, then it becomes a target for our class of free food events.

In summary, this project is a mailing list miner that searches MIT public mailing lists for free food events through classification. As previously described, associated with FreeFood@MIT is an email (*event@freefood.mit.edu*) that is added to the list of MIT public mailing lists on both Athena and Listserv (~5000 lists). On a regular basis, and in the form of a batch process, the system checks this email and dumps the messages to file. Through a classification process (rules-based or Bayesian), the emails that seem to be food related are stored in the system's database. These emails are also matched for an event date for sorting and added searching functionality.

Accessing the results of FreeFood@MIT's classification of Emails is done through a web interface that provides a search tool for the database. Since the system is based on an automated process where misclassification can occur, a recommendation system is provided to account for system errors and inform users about inaccuracies through

feedback. In the case of the Bayesian classification architecture described in the previous chapter, this feedback is also used to improve the accuracy of the classifier.

Figure 16 - FreeFood@MIT logo

This project implements both rules-based and Bayesian architectures and the obtained results will be used to compare the proposed methods.

5.2 Technologies and Implementation

The current version of the system is Linux based and is entirely written in Perl. It consists of different shell scripts that match all the processes of the previously described architectures. The database was implemented using MySql. There are two web interfaces involved in the system; one that provides searching and feedback for users who want to use the system and another that provides an easy way to manually classify a set of unlabeled Emails that can be used as the training set of the Bayesian classifier.

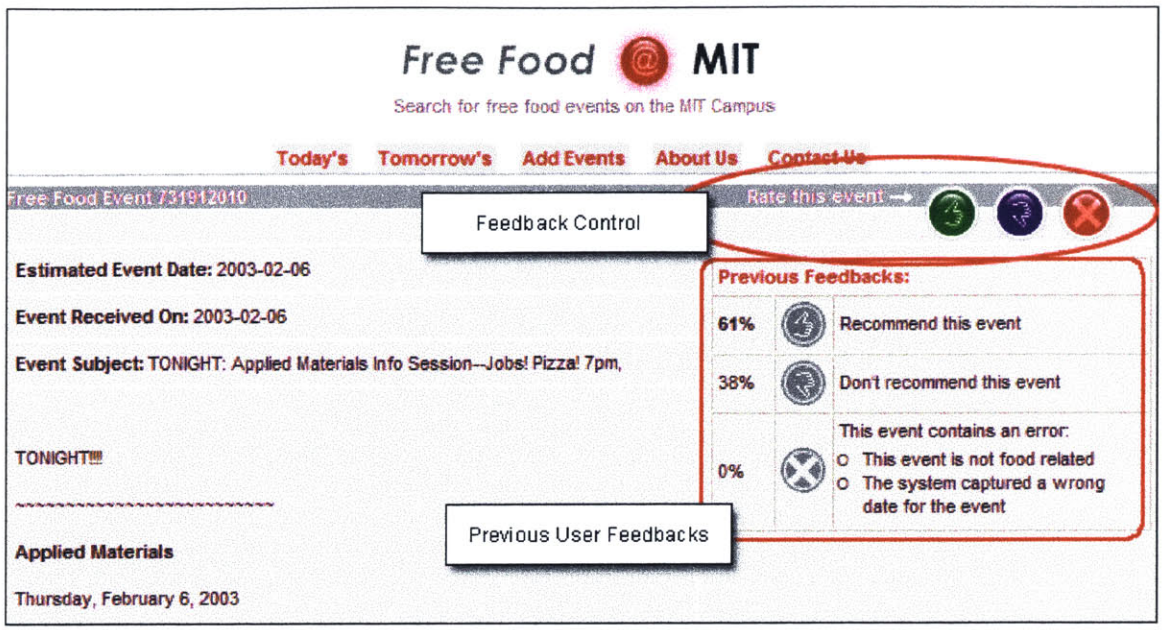


Figure 17 - User feedback in event view

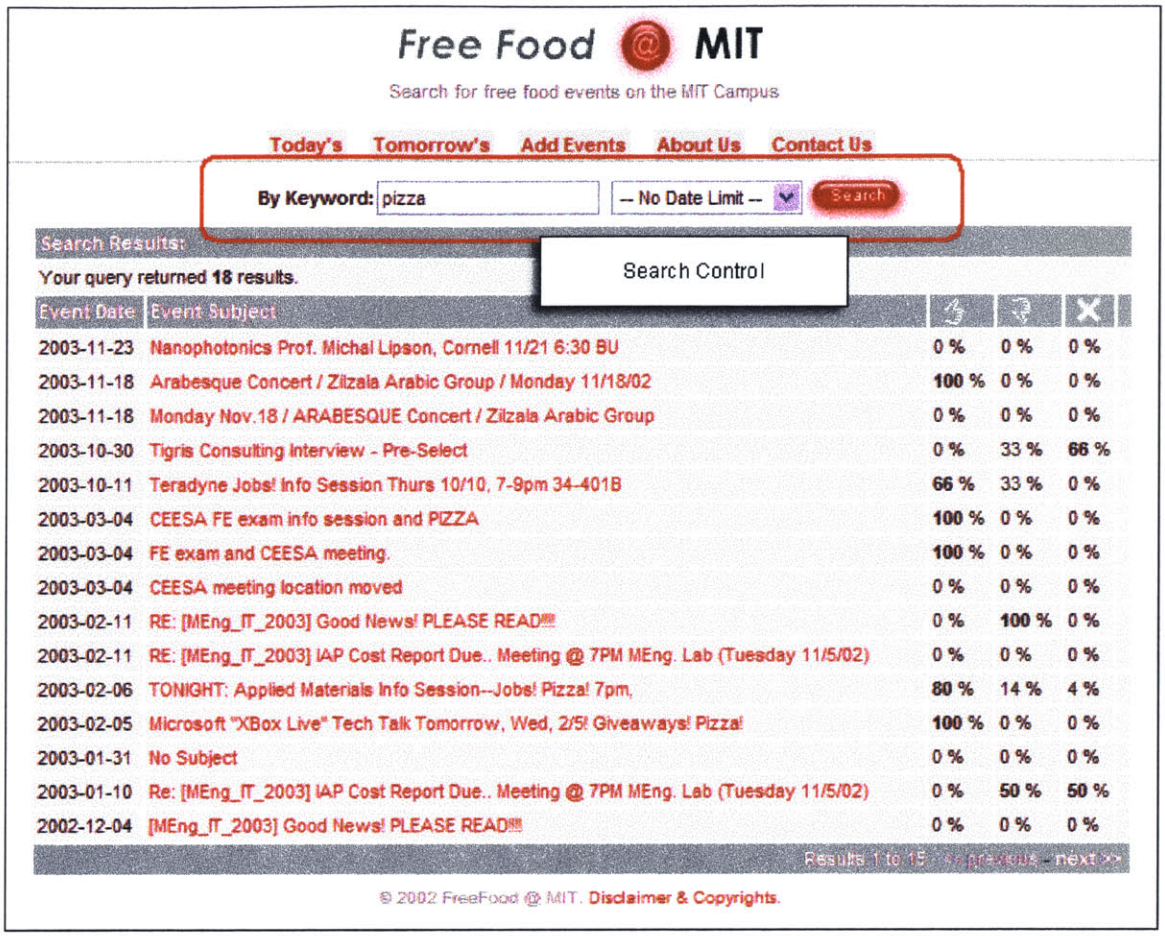


Figure 18 - Web interface searching

The system was tested with a corpus of 1876 Emails. These were primarily gathered from personal Email and classified using the manual classification web interface. This corpus was manually classified as follows:

	In class Freefood	Not in class Freefood	Total
Emails in corpus:	238	1638	1876

Table 10 - Freefood@MIT's testing corpus

In other terms, out of the 1876 Emails in the test corpus, 238 describe a free food event and 1638 do not. Free food events represent 12.68% of the total testing corpus.

5.3 Performance of the Rules-Based Architecture

Since the rules-based classifier has a static set of rules that fit the specific needs of classifying Emails on the criteria of whether they belong to the class Freefood or not, the entire testing corpus was used to test the performance of the classifier. Although the set of rules were changed throughout the course of this research, the current performance values represent the optimal results that we were able to get from a set of hand-crafted rules.

The classification of the testing corpus resulted in the following:

	Documents belonging to Freefood	**Documents not belonging to Freefood**
Documents Classified as Belonging to Freefood	197	62
Documents Classified as not Belonging to Freefood	41	1576
Totals	238	1638

Table 11 - Results of the rules-based classification

These numbers result in the following evaluation table (refer to section 3.4):

	Recall (R)	**Precision (P)**	**Fallout (F)**	**Accuracy (AR)**	**Error (ER)**
Class Freefood	82.7%	76.0%	3.8%	94.5%	5.5%

Table 12 - Rules-based classification evaluation results

The first thing to note in this case implementation is that there is a big discrepancy between the number of Emails that belong to the class Freefood and those that don't. Since the events mining system will always have a large source of input Emails and these will get massively filtered to keep the events pertaining to a certain category, a class disparity will always be present. The proposed architectures should account for this disparity and be evaluated in accordance.

Under this case, the seemingly high value of 94.5% of the accuracy is not as important as having a large precision rate. The precision rate describes the portion of the events that are now stored in the database and that actually describe a Free food event. This precision of 76% is barely acceptable, since it means that in 24% of the cases, users will fall on a misclassified Email while searching for a free food event.

The second most important evaluation measure is the recall in this case. Recall is the portion of Emails actually describing a free food event and that are properly classified by the system. This means that in these results, 82.7% of the Emails actually representing a free food event were captured by the system.

Overall, the results of this classification scheme are acceptable since they removed the majority of the non-class related set of Emails while keeping a decent amount of Emails pertaining to the events we are looking for. Also note that the performance measures that involve the total number of Emails such as error and accuracy rates are not as important as the others since the relatively small number of Emails belonging to our target class get diluted when compared to the overall number of received Emails. For this reason, an accuracy of 94.5% would not mean that the classification process performs well.

5.4 Performance of the Bayesian Architecture

In the case of the Bayesian classifier, the testing corpus of documents needs to be split into a training set and a verification set. The latter will be used to calculate the evaluation measures. The Former will be used to build our classifier.

The testing corpus was divided with the ratio 70/30% between training and verification corpuses. This yields the following segmentation of the corpus:

	Training Corpus		Verification Corpus		Total
	Freefood	Not Freefood	Freefood	Not Freefood	
Emails in corpus:	167	1147	71	491	1876

Table 13 - Bayesian classification corpus

After training the naïve Bayes classifier as described in chapter 3 and computing the individual token probabilities, the verification corpus was classified to evaluate the performance of this approach. The following results were obtained:

	Documents belonging to Freefood	Documents not belonging to Freefood
Documents Classified as Belonging to Freefood	65	7
Documents Classified as not Belonging to Freefood	6	484
Totals	71	491

Table 14 - Results of the Bayesian classification

These classification numbers result in the following performance evaluation table:

	Recall (R)	Precision (P)	Fallout (F)	Accuracy (AR)	Error (ER)
Class Freefood	91.5%	90.3%	1.4%	97.7%	2.3%

Table 15 - Bayesian classification evaluation results

In this case, we recorded better performance compared to the rules-based approach in regards of all parameters. The 90.3% precision is adequate for the sake of our application where a mere 9.7% of the stored Emails are not pertinent to our aimed class. The recall of 91.5% yields that a small portion of the Emails that actually belong to our class of events didn't make it through classification.

However these results are the optimal reached after a long series of tests with corpus partitioning, probability decision threshold (ranged from 0.5 to 0.9) and feature set alternations. For example, adding context specific information to our set of tokens improved performance figures dramatically. This was done using the methods described in chapter 3 where, for example, new phrasal tokens such as "Free Food" were added.

Another important aspect of improving performance was enlarging the training set. For the sake of experiment, we tried dividing the testing corpus into 10% training and 90% verification which led to a meager 35% precision. We then used the feedback loop and the feedback analyzer process to increase the size of our training corpus. This led to astonishing results proving that having a feedback loop is a valid approach to increase performance.

5.5 Performance of Information Extraction

The information extraction component of the mining architectures revolved around estimating event dates based on their content. Since this estimation is used to sort events and serves as a prime search parameter for users, high accuracy rates are essential.

For the sake of testing date extraction, we used the entire set of free food Emails. From the manual classification, these summed up to 238 as previously declared. From these 238 free food events, 217 had their correct date estimated using the procedure described in section 4.3. This yields an accuracy of approximately 91.2%.

This attained accuracy was considered to be suitable for our case application. By searching and testing the database for free food events, the 8.8% wrong date estimations did not have a significant impact on usability.

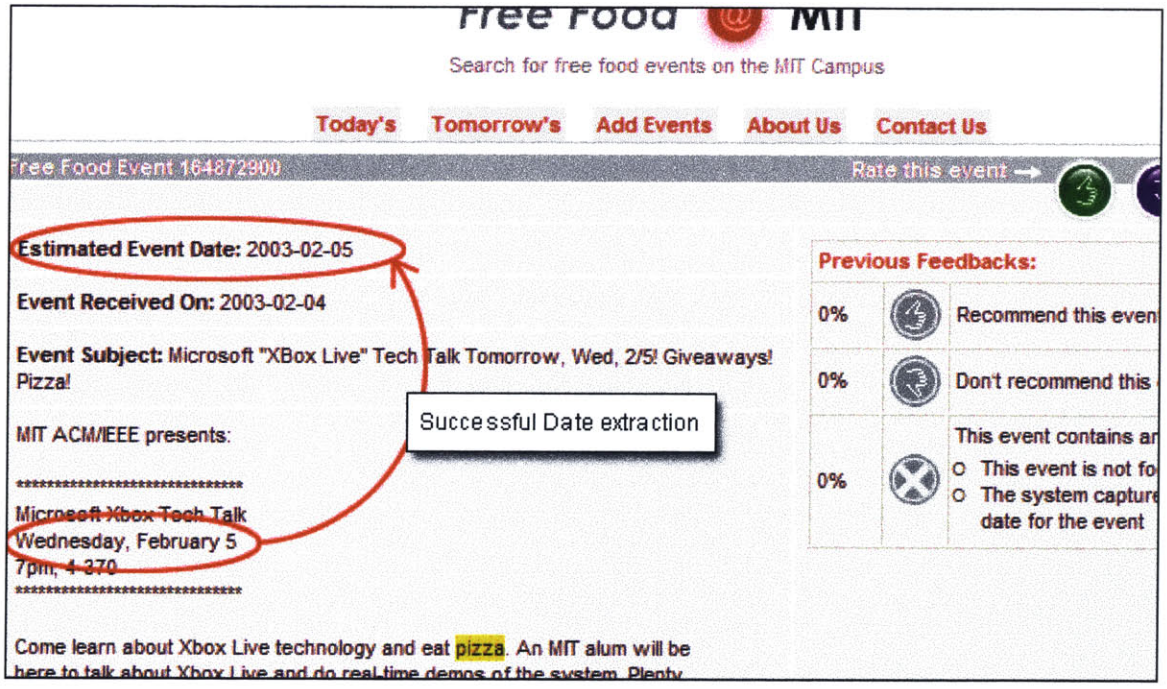


Figure 19 - Sample date extraction from FreeFood@MIT

Chapter 6
Conclusion

6.1 Summary and Contributions

The mining architectures and the case implementation presented in this thesis proved that mining mailing lists for specific content has tremendous benefits. In the case of FreeFood@MIT, out of 1876 Emails, 238 (around 12%) are pertinent to our classification task. As depicted in chapter 2, there are more than 5000 mailing lists at MIT and an expected 5000 to 500000 messages can be expected per day. With such large amounts of broadcasted information, searching for a free food event doesn't have to involve searching the entire archive of received Emails. The proposed mining architectures attend to this issue by providing a reduced set that contains the specific content we are looking for with a certain precision and recall. In both implementations, precision figures ranged between 76% and 90%, and recall figures between 82% and 91%. These numbers yield that at least 76% of the Emails we are searching are truly free food events. Compare this worst case precision to having to search a database with only 12% of the events matching our search criteria. The minimum recall value of 82% means that we are missing 18% of the broadcasted free food events to public mailing lists. With such a large source to begin with, the worst case scenario of omitting 18% of the events is still acceptable.

Between the two architectures, it is clear that the Bayesian approach outperformed the rules-based one. This is due to the fact that our set of rules was constructed manually, thus potentially missing attributes pertaining to our classification task. During the course of this research, it was noted that the Bayesian classification scheme reacted better to Emails that belong to our target class but have a different structure from the typical ones. Another benefit is the influence of the training corpus' size on accuracy. Consuming user feedbacks to increase the size of this corpus and thus improving accuracy becomes a great answer to the high costs of labeling Emails for training. The

only drawback of this probabilistic approach is that the naïve-Bayes classifier achieved leaps in performance by including context specific information in its features set. This might be harder to implement in case we have a multi-class classification scheme since we would have to build a specific classifier for each binary classification task.

The contribution of this thesis lies on two levels. The first major contribution is identifying mailing lists as a potential source of information and using it to provide topic specific search engines. The benefits of these are that they use Emails as their input and not regular web pages. In the case of finding events and activities, using web searches is inadequate since rare are the events that are publicized through a web site. We can see how searching broadcasted Emails brings a new dimension to searching real-time information. The second contribution is identifying two mining architectures that are specifically designed to extract category specific Emails from a large input set and comparing them. The user feedback loop that helps improve classification accuracy is a novel approach to deal with the high costs of labeling a training set.

6.2 Future Work

This thesis represents a preliminary study on how to extract topic-specific content from public mailing lists. With such a rich source of information, there is a lot more to be done. As a primer, this document identified mailing lists as potentially searchable and proved how information aggregation and filtering can be useful to people in large decentralized institutions such as MIT. Subsequent steps to harness this information can be numerous. The first that comes to mind is to grow from information filtering or binary classification to a multi-class classification scheme where all the Emails broadcasted on mailing lists would be classified in different ways to create a clustered document space of Emails thus starting a new bread of mailing lists search engines.

BIBILIOGRAPHY

[1] Charniak E (1993), *Statistical Language Learning*, MIT Press

[2] Cohen W (1995), *Fast Effective Rule Induction*, In Machine Learning: Proceedings of the 12th International Conference, Morgan Kaufmann

[3] Cohen W (1996), *Learning Rules that Classify E-mails*, In AAAI Spring Symposium on Machine Learning for Information Access

[4] Cohen W (1996), *Learning with Set-valued Features*, In proceedings of AAAI-96

[5] Cover TM, Thomas JA (1991), *Elements of Information Theory*, Wiley

[6] Domingos P, Pazzani M (1996), *Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier*, Proceedings of the 13th International Conference on Machine Learning: pp. 105–112

[7] Fung R, Del Favero B (1995), *Applying Bayesian Networks to Information Retrieval*, Communications of the ACM 38(3): pp. 42-48

[8] Good IJ (1965), *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*, MIT Press

[9] Hall RJ (1998), *How to Avoid Unwanted Email*, Communications of the ACM, 41(3): pp. 88–95

[10] Langley P, Wayne I, Thompson K (1992), *An Analysis of Bayesian Classifiers*, Proceedings of the 10th National Conference on AI: pp. 223–228

[11] Lanquillon C (2001), *Enhancing Text Classification to Improve Information Filtering*, Doctoral Dissertation, Otto-von-Guericke-Universität Magdeburg

[12] Lewis D, Knowles KA (1997), *Threading Electronic Mail: A Preliminary Study*, Information Processing and Management, 33(2): pp. 209–217

[13] McAllester D (1993), *Bayesian Networks*, Lecture Notes for 6.824, Artificial Intelligence, MIT

[14] Mitchell TM (1997), *Machine Learning*, McGraw Hill, 1997

[15] Pazzani MJ (1995), *Searching for Dependencies in Bayesian Classifiers*, In Proceedings of the 5th International Workshop on Artificial Intelligence and Statistics, D. Fisher and

[16] Pearl J (1988), *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kauffmann

[17] Sahami M (1996), *Learning Limited Dependence Bayesian Classifiers*, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press: pp. 335-338

[18] Sahami M, Dumais S, Heckerman D, Horvitz E (1998), *A Bayesian Approach to Filtering Junk E-mail*, In Learning for Text Categorization – Papers from the AAAI Workshop: pp. 55–62

[19] Sahami M (1998), *Using Machine Learning to Improve Data Access*, PhD Thesis, Stanford University

[20] Salton G, Wong A, Yang CS (1975), *A Vector Space Model for Automatic Indexing*, Communications of the ACM 18: pp. 613-620

[21] Valiant L (1984), *A Theory of the Learnable*, Communications of the ACM, 27(11): pp. 1134-1142

[22] Van Rijsbergen CJ (1979), *Information Retrieval*, Butterworths

[23] Yang Y, Chute CG (1994), *An Example-based Mapping Method for Text Categorization and Retrieval*, Transactions of Office Information Systems 12(3), Special Issue on Text Categorization

[24] Yang Y, Pedersen J (1997), *Feature Selection in Statistical Learning of Text Categorization*, In Machine Learning: Proceedings of the 14th International Conference, Morgan Kaufmann: pp.412-420

[25] Zipf GK (1949), *Human Behavior and the Principle of Least Effort*, Addison-Wesley

Online References

[26] MIT Encyclopedia of Cognitive Science: Bayesian Networks (MIT), *http://cognet.mit.edu/MITECS/Entry/pearl.html*

[27] MIT Information Systems: Athena Computing Facility (MIT), *http://web.mit.edu/olh/Welcome/intro.html#what_is_athena*

[28] MIT Information Systems: How to create and edit MAILING LISTS and GROUPS (MIT), *http://web.mit.edu/answers/accounts/accounts_listmaint.html*

[29] How Much Information: Email and Mailing Lists (University of California, Berkeley), *http://www.sims.berkeley.edu/research/projects/how-much-info/internet.html*

[30] L-Soft: Listserv, *http://www.lsoft.com/products/default.asp?item=listserv*

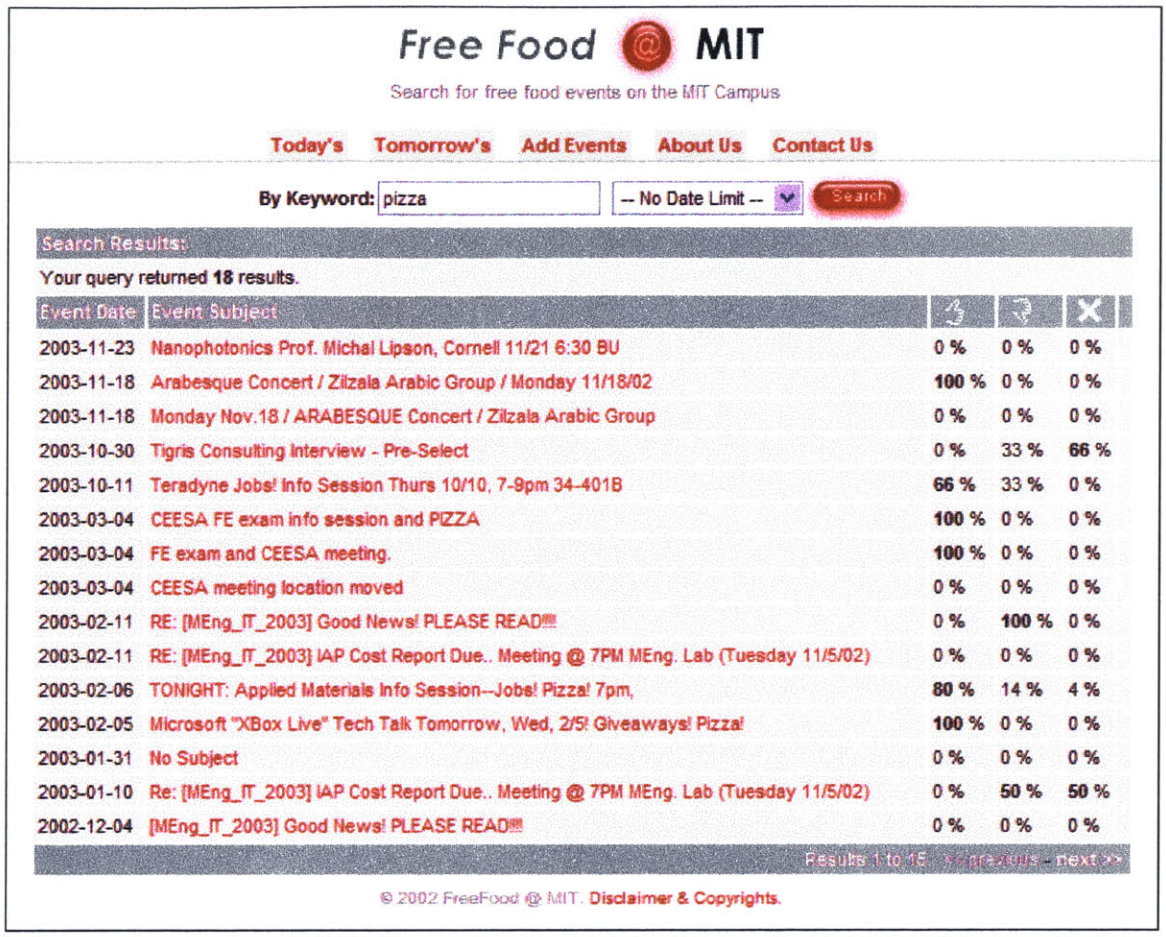
[31] L-Soft: Listserv Statistics, *http://www.lsoft.com/news/default.asp?item=statistics*

APPENDICES

Appendix A: Screenshots of FreeFood@MIT

This appendix contains screenshots from FreeFood@MIT. These are taken from both the web interface and the batch scripts that check for and classify incoming emails.

Figure 20 - The index page of FreeFood@MIT



Free Food @ MIT

Search for free food events on the MIT Campus

Today's Tomorrow's Add Events About Us Contact Us

By Keyword: pizza -- No Date Limit -- Search

Search Results:

Your query returned **18** results.

Event Date	Event Subject			
2003-11-23	Nanophotonics Prof. Michal Lipson, Cornell 11/21 6:30 BU	0 %	0 %	0 %
2003-11-18	Arabesque Concert / Zilzala Arabic Group / Monday 11/18/02	100 %	0 %	0 %
2003-11-18	Monday Nov.18 / ARABESQUE Concert / Zilzala Arabic Group	0 %	0 %	0 %
2003-10-30	Tigris Consulting Interview - Pre-Select	0 %	33 %	66 %
2003-10-11	Teradyne Jobs! Info Session Thurs 10/10, 7-9pm 34-401B	66 %	33 %	0 %
2003-03-04	CEESA FE exam info session and PIZZA	100 %	0 %	0 %
2003-03-04	FE exam and CEESA meeting.	100 %	0 %	0 %
2003-03-04	CEESA meeting location moved	0 %	0 %	0 %
2003-02-11	RE: [MEng_IT_2003] Good News! PLEASE READ!!!	0 %	100 %	0 %
2003-02-11	RE: [MEng_IT_2003] IAP Cost Report Due.. Meeting @ 7PM MEng. Lab (Tuesday 11/5/02)	0 %	0 %	0 %
2003-02-06	TONIGHT: Applied Materials Info Session--Jobs! Pizza! 7pm,	80 %	14 %	4 %
2003-02-05	Microsoft "XBox Live" Tech Talk Tomorrow, Wed, 2/5! Giveaways! Pizza!	100 %	0 %	0 %
2003-01-31	No Subject	0 %	0 %	0 %
2003-01-10	Re: [MEng_IT_2003] IAP Cost Report Due.. Meeting @ 7PM MEng. Lab (Tuesday 11/5/02)	0 %	50 %	50 %
2002-12-04	[MEng_IT_2003] Good News! PLEASE READ!!!	0 %	0 %	0 %

Results 1 to 15 << previous - next >>

© 2002 FreeFood @ MIT. Disclaimer & Copyrights.

Figure 21 - Sample free food search results

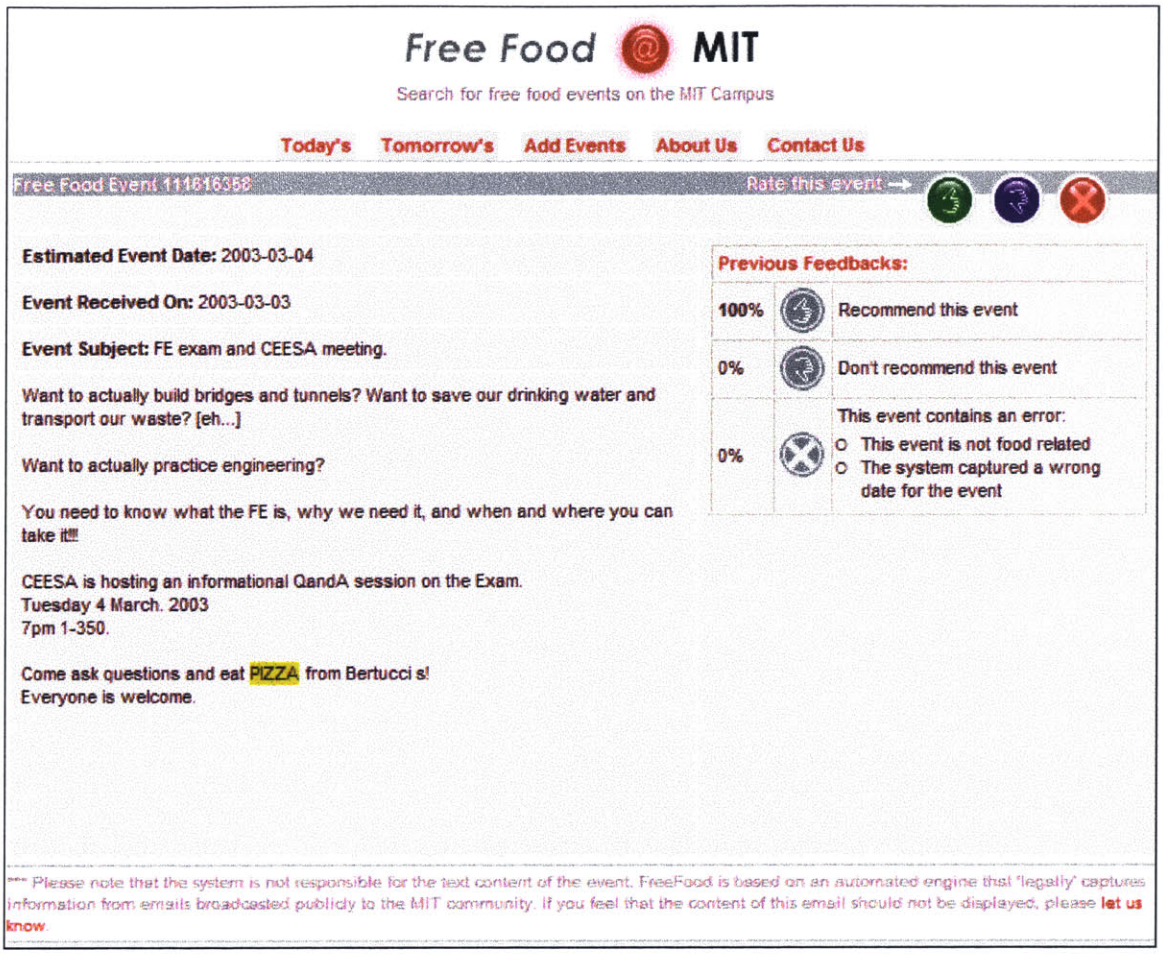


Figure 22 - Sample event view

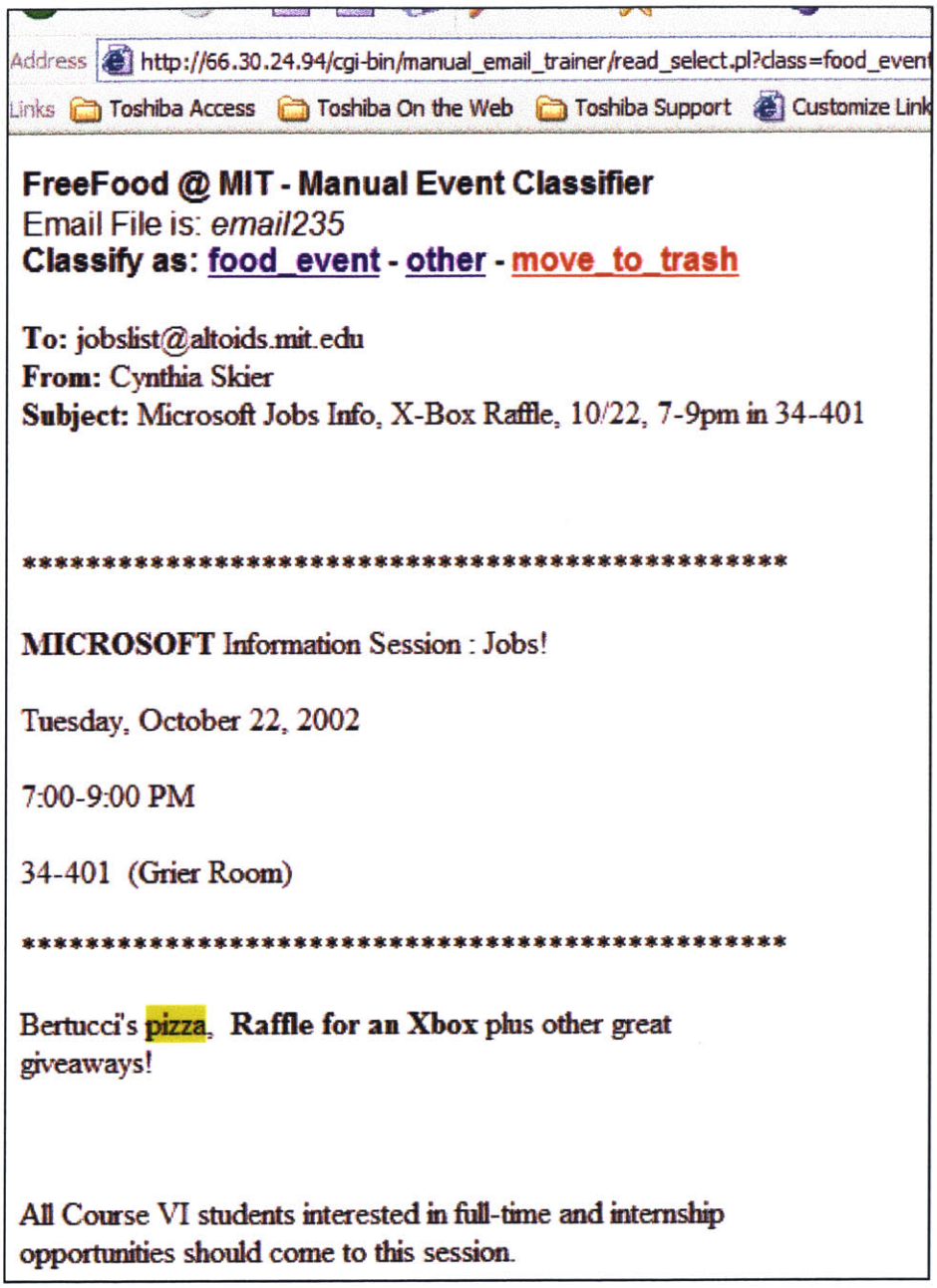


Figure 23 - Manual email classifier

```
[lkscol00:marioh]/marioh/FreeFood@MIT/bin 34# ./train_bayes.pl
Bayesian Trainer

Connecting to database [66.30.24.94/freefood]

Verifying training corpus:
Freefood: 167 - Not Freefood: 1147 - Total: 1314

Calculating probabilities and building feature set:
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Training complete
[lkscol00:marioh]/marioh/FreeFood@MIT/bin 35#
```

Figure 24 - train_bayes.pl : Script that models the training process

```
[lkscol00:marioh]/marioh/FreeFood@MIT/bin 44# ./classify_bayes
Bayesian Classification Process

Checking for new e-mail: [562 new e-mails] found

Connecting to database [66.30.24.94/freefood]

Classifying:
.....
.....
.....
.....
.....
.....
.....
..
[72 e-mails] classified as free food

Estimating dates and storing in database:
.....

Classification complete
[lkscol00:marioh]/marioh/FreeFood@MIT/bin 45#
```

Figure 25 - classify_bayes: Batch script that models the Bayesian classification process

```
[lkscol00:marioh]/marioh/FreeFood@MIT/bin 49# ./feeback_analyzer
Feedback Process

Connecting to database [66.30.24.94/freefood]

Checking events for feedback treshold:
[12 e-mails] added to training corpus
Freefood: 10 - Not Freefood: 2
2 Events cleared from database as misclassified

Re-train classifier:
Bayesian Trainer

Connecting to database [66.30.24.94/freefood]

Verifying training corpus:
Freefood: 167 - Not Freefood: 1147 - Total: 1314

Calculating probabilities and building feature set:
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Training complete


Feedback process complete
[lkscol00:marioh]/marioh/FreeFood@MIT/bin 50#
```

Figure 26 - feedback_analyzer: Batch script that models the feedback process

```
[lkscol00:marioh]/marioh/FreeFood@MIT/bin 56# ./classify_rules
Rules-Based Classification Process

Checking for new e-mail: [1876 new e-mails] found

Connecting to database [66.30.24.94/freefood]

Classifying:
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

[238 e-mails] classified as free food

Estimating dates and storing in database:
.....
.....

Classification complete
[lkscol00:marioh]/marioh/FreeFood@MIT/bin 57#
```

Figure 27 - classify_rules: Batch script that models the rules-based classification process

Appendix B: List of Listserv Public Mailing Lists at MIT

This is the list of Listserv public mailing lists with their description.

List Name	List Description
iberia	Asociacion de Espanioles en Boston
mitaah-announce	MIT Atheists; Agnostics; and Humanists Announcement list
rsa	MIT Romanian Student Association
rsa-boston	MIT Romanian Stu. Assn. - Boston events
rsamit	Romanian Student Association - MIT members
ACAFEN-L	ACAFEN-L Academic Study of Fandom
ACE-NNWL	Mass. Network ACE--National Network Women Leaders
ACRLNECB	ACRL/NEC Board
ACRM-L	SPG/ACRM Team
ACSP-INT	ACSP Spec Comm: Globalization & Planning Ed
ACTIVITY	Alumni/ae Association Email List--Activities
ADADBA-L	MIT Adabase Database Adminstrator and Programmers
ADDSOP	Alumni Donor Development Schools Operations Group
ADDSPLAN	ADDS System - Planning Discussion List
ADDSTRAT	ADDS Client/Server Strategy Group
ADMALL-L	Entire Admissions Office
ADMIS-DR	Admissions Disaster Recovery list
AEPI-GAM	AEPi Board Games Mailing List
AFRICANS	MIT African Students Association
AFS-SEL	AFS SocioEconomics Discussion List
AGS-WSC	Alliance for Global Sustainability World Student Community
AIDFOLKS	Student Financial Services discussion group
ALG-COMB	Preprint Announcements in Algebraic Combinatorics
ALUMGRAD	MIT Grad-school Alumni announcements
AMPOL-L	MIT Amer. Pol. Papers-Conference Announce
ANZCLUB	MIT Australia New Zealand Club
APE-L	Discussion list for Aperture users in O&S
APHI05-A	MIT Alpha Phi '05 Asians
ARCH-HTC	Histsory; Theory & Criticism Section
ASCEND	Ascend Replacement Project
ASDFCM-L	ASD/Facilities Management Issues
ASDP-L	African State and Democracy Project
ASH_BAD	Ash_Bad Mailing List
ASHA	Funders of NGO educational projects in India
ASHDOWNA	MIT Ashdown House Announcements List
ATO-OFFI	MIT ATO Officer Mailing List
AWP-TALK	America's War on Poverty Talk
BAJANS	A Barbadian Mailing List
BALLRM-M	Moderated Discussion List for Ballroom and Swing Dancing
BALLROOM	Discussion of Any Aspect of Ballroom and Swing Dancing
BAR-LTE	MIT Barker Library Local Tech Experts
BC223	MIT Burton Conner Suite 223
BDASH02	MIT SWE BeaverDash 2002 Participation Info
BDC-LIST	MIT Ballroom Dance Club's Announcement List
BEAVDASH	MIT SWE BeaverDash 2002
BEIJING	MISTI 2001 Beijing Interns
BFC-FANS	Fans of the Boston Demons - Events
BFC-TEAM	Boston Demons Football Club discussion and information
BIO-GRAD	MIT Biology Graduate Students
BIO-SRS	MIT Dept of Biology Sponsored Research Staff
BIOADMIN	Biology Administrative Staff
BIOCOMP	MIT Biology Department Computer Help
BIOJRFAC	MIT Biology Department Junior Faculty
BIOLOGY	MIT Biology Department EMail list

BIOMEDIA	Biology Dept. Kitchen and Media Requests
BIOMGS	Biology; All Merck Fellows
BIOMPI	Biology Principal Investigators with Merck Affiliation
BIOSAFTY	A Biosafety Discussion List
BIOSTAFF	Biology Support and Admin. Staff
BIOWEEK	MIT This Week in Biology
BKUP-MIN	The TSM Announcement List
BLACCM	Boston Serbian Community Events and Discussion
BLAKMAIL	MIT Black Student Union
BNCT	Boron Neutron Capture Therapy Discussion List
BOS-GER	Boston-Area German-Language Events
BOST-REV	Boston Review Newswire
BRAINTMR	Brain Tumor Research/Support
BRUHA	Joint Email for Roommates at Dickenson-St. Apt.
CAOPAY-L	CAO Users Group at MIT
CAOPROG	CAO Programmers Group at MIT
CAOTECH	Tech Support within MIT CAO
CAOUSERS	CAO Users Group at MIT
CARD-REQ	Access-Card Team
CARDREQ	Access-Card Team
CASH-SAP	SAP A/R Cashier's Project at M.I.T.
CASTAC-L	Comm of Anthropology of Sci; Tech; and Computers
CEPAT-L	Running group for Singapore Students' Society
CHAPIN-L	Chapin-L Foro electronico de la tematica chapina 1995-2002
CHIPAPER	Working Paper on Consumer Health Informatics; Production Team
CMI-BSBL	CMI-Baseball - MIT Students at Cambridge
CMI-CRKT	CMI-Cricket - Cambridge Students at MIT
CMROMERO	Cesar Romero and Friends
COFAID-L	Section-568 Financial-Aid Discussion Group
COLL-ORG	DUSP Fall Colloquium - Organizers
CORE-NE	NikoNiko Net - Japanese Language Educators
CPS-SCI	MIT Working Group for CPS Science
CS-GROUP	Communicating in Cyberspace Group
CSCVB	MIT CSC Volleyball Mailing List
CSEVEGO	MIT Hungarian Students' Association
CSU-L	Cambridge Sports Union(running club) Discussion
CTPSTUDY	MIT Center for Theoretical Physics Study Group
CUA-SEM	Center for Ultracold Atoms seminar announcements
CUR-L	Computer Usage Report List
CYBER21W	MIT 21W.785: Communicating in Cyberspace
CYPRUS	CYPRUS LIST
C3PO	MIT ECAP Student Mailing List
DAPER-FI	MIT DAPER Facilities Information
DAWG	MIT DUSP Activist (Planarchist) Working Group
DB-L	MIT Database Services Maintenance and Installations
DBLDAY-L	MIT Varsity Baseball Team
DESIGN99	Design Content Institute 99-00 MIT-Cambridge
DHOLI	MIT Diwali Night Group
DIAL-UP	Dial-Up Software Advisory Group
DIGLAB	Digital Instruction lab in libraries
DISCUS-L	List on keeping/raising Discus fish
DNE-L	Dance New England On_Line Discussions
DOST-DOC	MIT O&S Documentation discussion
DOST-HID	MIT DOST Hardware Installations and Deinstallations
DSPC-ANN	MIT DSpace Project Announcements
DTDSUMMR	MIT Delta Tau Delta summer boarders 2001
DTSPG-L	DTS - SPG Discussion List
DUSPCC-L	MIT Urban Studies Dept. Commun. Comm.
DUSPSUMR	MIT Course-11 Summer Activities and Events
DYNBBALL	Dynasty Basketball
EBIZNEWS	Friends of the MIT Center for eBusiness

EC-RES	MIT East Campus Dormitory Residents
ECAT2DEV	MIT ECAT2 Developers List
EH-ALL	Edgerton House Announcements & Events
EMIT	e-MIT Entrepreneurship Digest
ERGO	Discussion list for Ergo (an MIT RadCaps publication)
ERROR404	Error 404 - Friends of Adam Powell
ERT-LEAD	MIT Emergency-Response Team Leaders
ESP-NEWS	MIT ESP-News
ESP-2003	MIT ESP-News-2003
ESP-2004	MIT ESP-News-2004
ESP-2005	MIT ESP-News-2005
ESP-2006	MIT ESP-News-2006
ESP-2007	MIT ESP-News-2007
ESP-2008	MIT ESP-News-2008
ESP-2009	MIT ESP-News-2009
ESP-2010	MIT ESP-News-2010
EVENTS-L	Event Planners at MIT
EXPEND-L	Expendables - Consortium of Colleges and Universities
FAMILIES	News from the MIT Family Resource Center
FAS-3A04	MIT FAS 3.A04 Physical Metallurgy
FASSAC	CSS Services and Standards Committee
FEARTRST	Fear and Trust
FEMINIST	ALA Feminist Task Force Discussion List
FESTIVAL	Culture Festival
FIBROM-L	FIBROM-L Fibromyalgia Discussion Group
FISHFOLK	Fisheries Social Science Network
FOHLEN	Borussia Moenchengladbach
FULB98UK	Fulbright 1998 UK Fellows in USA
GIFT_01	Sloan Class of 2001 Gift Committee
GLUTTON	People visiting Dim Sum; Anna's Taqueria; etc.
GNAHELPW	Globewide Network Academy Help Wanted Ads
GNAMAIL	Globewide Network Academy Mailing List Coordination
GSC-ANNO	MIT Graduate Student Council Announcements
GSC-EXE	MIT Graduate Student Council Executive Committee
GSC-REQ	MIT Graduate Student Council E-mail Administrators
GSC-TEXT	MIT Graduate-Student Council Text Announcements
GSVC	MIT Graduate Student Volunteer Corps
GULFTALK	A Gulf of Maine Discussion List
GWIS	Alpha Omega Chapter - Graduate Women in Science
GZBALL	Gizmoball Group - MIT 6.170; Fall '01
HAUSLAB	Users of the Haus Lab; MIT RLE (26-465)
HAWKEYES	Communications in Cyberspace Group List
HCA-ANNO	MIT Housing&Community Affairs Announcements
HEALTHEV	Health Evaluation Informatics Discussion
HEUMC-L	Harvard-Epworth U. Meth. Church Announce
HKN-WEB	HKN Web Mailing List
HTC-ACAD	History; Theory & Criticism - Academic notices
HTCFORUM	History; Theory & Criticism - Forums & lectures
IBBL	Interbaronial Buffens League
IEEE-CS	Meeting Announcements for the Boston IEEE Computer Society
IGBP-WDG	IGBP-Webmasters Discussion Group
IHS97	Independence High School Class of 1997
IMBBALL2	MIT Course-2 C-League IM Basketball Team
IMHOCKEY	IM Hockey teams at MIT
IMUPDATE	Image and Meaning Announcements
INFO-COL	Colombian-related events in the Boston area
INT2004	International Class of 2004 discussion list
INT2005	MIT International Class of 2005 Discussion
IS-TQM	Discussion of TQM implementation in IS
ISACA-L	Information Systems Audit and Control Association List
ITID-ED	Editors-in-Chief of the ITID Journal

JEFF-L	List O' Jeffs
JHORNE	Jed's E-mail list
JOURNALS	The MIT Press Journals Staff List
JPNETLEC	JP NET Lecture Series List
JRLIUSHA	MIT 21W.785 Project Group
JUGUSERS	MIT Java Users Group
K-MBA-03	Korean Students at MIT Sloan Class 2003
K-TV	KTV Entertainment - $50K Team
K-12SD	MIT System Dynamics K-12 Discussion
KAPA-L	Korean News Around the World
KENLUG	Kenya Linux User Group Admin List
KNH-L	New Publications on MySocialNetwork.Net
KS-INFO	KS-Info
KS06	MIT Kappa Sigma Class of 2006
LAEC-L	Metro LA Educational Counselors
LEM	Lutheran-Episcopal Ministry at MIT
LEMSC	MIT Lutheran-Episcopal Ministry Steering Committee
LEONENET	A Discussion of Sierra Leonean Issues
LFOPROG	LFO Programmers Group at MIT
LINUXPCC	Linux for PowerPC Port
LOGPROFS	The Worldwide Logistics Professor List
MACA-L	Massachusetts Association of Crime Analysis
MACPCI-L	Macintosh PCI Discussion List
MAINZ05	FSV Mainz 05
MAPWNEWS	News pertaining to postdocs at MIT
MAPWTALK	Discussion group for postdocs at MIT
MARSROVS	Model-based Autonomy & Robotic Systems Rover
MASS-RES	Massachusetts ResNet Coordinators
MCCNEWS	MIT Computer Connection News
MCCORM-L	McCormick Hall Residents - General Announcements
MEDFIELD	Medfield MA High School Alumni Assoc
MGRSMTG	IS Managers Meeting
MG18-03	MacGregor 18.03 Study Group
MG24-04	MacGregor 24.04 Study Group
MG3-091	MacGregor 3.091 Study Group
MG8-012	MacGregor 8.012 Study Group
MIS-TEAM	Members of Physical Plant's MIS Group
MIT-BKUP	MIT Central File-Backup Service news
MIT-EWEB	MIT Users of EnterpriseWeb/VM
MIT-GCF	MIT Graduate Christian Fellowship announcements
MIT-GEMS	Grants for Education in Marine Science
MIT-GR	MIT-Greece Program discussion list
MIT-HOTF	MIT Homes of the Future Mailing List
MIT-IO	MIT Industrial Organization Seminar
MIT-ISA	International Student Association of MIT
MIT-News	News of interest to the MIT community
MIT-RAP	MIT Rap Radio Discussion
MIT-SSP	MIT Security Studies Contact Address
MIT-Talk	Talk about MIT; for MIT; by MIT.
MIT-TV-L	MIT Cable Television Schedule
MITADM-L	Admissions Office Staff
MITBIO	MIT Biology Seminar Colloquium
MITCAM	MITCAM Users Group at MIT
MITCF	MIT Communications Forum Subscribers
MITCSSA	MIT CSSA Mailing List
MITCSSAL	MIT CSSA Mailing List - Local
MITEFNYC	MIT Enterprise Forum of New York City
MITERS	MIT Electronic Research Society Discussion
MITES-L	MIT MITE2S Alumni Mailing List
MITESA-L	MITVMA/C Upgrade to VM/ESA Discussion List
MITEZT-L	MIT Community - EasyTrieve-Plus Info Exchange

MITHAS	MITHAS Concert Announcements
MITIRLIB	MIT Industrial Relations Library
MITISO-L	MIT International Students Info List
MITNA	MIT Nautical Association Announcements
MITNEWS	MIT News Office Newsletter
MITP-L	MIT Press Staff Discussion List
MITPSS	Messages about MIT's Microsoft PSS agreement
MITRAVEL	Friends of the MIT Alumni Travel Program
MITRECYC	MIT Recycling Information
MITTRADV	TechReview Advt'g - Updates; Rates; Deadlines; Offers
MITVIRUS	MIT Virus Notification Service
MITVMWWW	I/T Service Process Webmasters
MOST	MIT Organization of Serbian sTudents
MOSTNEWS	MIT Organization of Serbian sTudents Event News Only
MSEKGSAS	MIT DMSE Korean Grad Stud Assoc of Singles
MSST-L	Macintosh Server Service Team Project
MSWDHLPR	MIT Microsoft Word User Group
NAMELESS	The Nameless Coffee House Mailing List
NAMEPRIZ	MIT Advance Naming Team - Prize List
NANTEERS	MIT Nantucketeers and others interested
NATYA02	Culture Show 2002; MIT Natya Tilang Thillana Participants
NAT2CHAT	Natural V2 Issues and Information Forum
NAVY-ULT	MIT Navy ROTC IM Ultimate Team
NEPAL-03	MIT CivEnvEng M.Eng - NEPAL group for 2003
NETV-L	NETVIEW Installation Project
NIMWF	National Initiative for Minority Women Faculty - Discussion List
NOEL-L	A testing list - not for subscription
NO6PARTY	The No6 Club Party List
NSFAWARD	Electronic Award Notices from NSF
NTUCE-92	National Taiwan University Civil Engineering '92 List
NZPOLICY	The Harvard/MIT NZ Policy Network
NZPOP-L	New Zealand Popular Music List
OCEANF-L	Ocean Farmers of America Forum
OKAZIYA	Offers/requests to carry mail/docs/etc to/from Russia
OKIDESIG	OKI architecture design group
OMEGA01	Order of Omega - '01 pledge class
ONRAWARD	Electronic Award Notices from ONR
OPEN-CUR	MIT Open-Curriculum Discussion & Announcements
ORIGAMI	Origami List
OS	IS Operations & Systems Staff
OS-ADMIN	IS Operations & Systems - Administration
OS-FM	IS Operations & Systems - Facility Management
OS-MGR	IS Operations & Systems - Managers
OS-UA	IS Operations & Systems - User Accounts
OSP-TECH	OSP Awards Technical Team
OTG-TECH	MIT OTG - SUMMIT development
PAW-AC	Palestine Awareness Week Action Committee
PHRJMAIL	MIT Program on Human Rights and Justice
PHYSFAC	Physics Faculty
PHYSGS	Physics Graduate Students
PI-2002	Project Interphase Student List
PI02-TAS	Project Interphase 2002 Staff List
POOL-HRS	Weekly Schedule of MIT Alumni Pool Hours
PORTUGAL	Portuguese Student Association
PPST00	MIT Prog in Polymer Sci&Tech - students from 2000
PRASTFAM	The Whole Praster Family
PROJ-CON	Project-Contact Volunteers
PSSTATD	Production Control Daily Status Report
PSSTATW	Production Control Weekly Status Report
QIP-SEM	MIT Quantum Info-Processing Seminar Announcements
QLOOK-L	Quick Look Discussion List

QUANT-01	Freshmen Advising Seminar - Our Quantum World
RACE-MED	Race and/in the History of Medicine
RACE-SCI	Race and/in the History of Science
RACE-TEC	Race and/in the History of Technology
RADGRAD	List for Graduates of the RAD class
RAFFLES	A Moderated Informational List for Rafflesians
RAID-L	Bug/Problem/Suggestion Processes
RAMIT-L	Russian Club at MIT Discussion List
ROOFTOPS	Wireless community data networks
RSA-Forum	MIT RSA Forum
RUSHPRO	MIT IFC Rush Promotions Committee; 1997
RX-ANTH	Community for the Anthropology of Pharmaceuticals
R3-MAINT	SAP R3 Maintenance Team
SABMAG	Honda Sabre and Magna Motorcycle Owners
SAKLUNCH	Physical Oceanography Sack-Lunch Seminars
SANDESH	MIT SANGAM student-group newsletter
SANG-ADS	SANGAM student group at MIT: Advertisements
SANG-GEN	SANGAM student group at MIT: Events and Announcements
SANGMOON	Boston SangMoon alumni
SAP-HWS	SAP implementation for MIT Hardware Services
SAP-R3-L	SAP-R3-L List auto-responder
SAPUSERS	SAPUSERS WebMeeting
SAPWEB-L	MIT SAP Web Interface Development Team
SCHNAM-L	MITVMA/MITVMC Schedule Notification List
SCUBA	MIT SCUBA Club
SCW-L	Supercritical Water Research and Applications
SDODIST	SDODIST
SEAGRANT	MIT Sea Grant Staff List
SEASON	SEASON Mailing List
SEF-LIST	Science and Engineering Faculty Discussion
SEQUEST	MIT Carbon-Sequestration Info Network
SGLTEAMS	SGL Team Owners
SIGUNION	Sigma Chi Reunion List
SIM-ULT	Simmons Hall Pickup Ultimate Frisbee Announce
SMANETOP	Network technical staff at MIT; NTU; NUS
SOCIAL-L	Socialist Club at MIT Discussion List
SPE-EXEC	MIT Sigma Phi Epsilon Executive Committee
SPEC-LAB	MIT Chemistry Department Instrument Facility News
SPES-DES	Parties to the MIT SPES IT Project
SPGMIN-L	VM System Services Team meeting minutes
SPOUSES	Spouses & Partners @ MIT discussion list
SQLDBA-L	MIT SQL Administrators/Consultants
SQLUSR-L	SQL/DS Users at MIT
ST-USP	Special Topics - MIT Urban Studies Fall'01
STEP2SD	Second Step San Diego Discussion
STIHI	Discussion of (mostly) Russian poetry
SWEATS98	MIT SweatShirt Company 1998
SYLLOGOS	Hellenic Students' Association of MIT
T+CFORUM	MIT Technology and Culture Forum
T+D-IDEA	Training & Development Idea-Bank
TANGO-A	Announcements of Argentine Tango Events
TANGO-L	Discussion of Any Aspect of the Argentine Tango
TCCM	MIT Tech Catholic Community Announcements
TDC-Deadwood	MIT TDC Deadwood (Alumni)
TDC-Local-Deadwood	MIT TDC Boston-area Deadwood (Alumni)
TDQM	Total Data-Quality Management
TEAM6	Project team for MIT subject 2.74
TECHDIVE	MIT SCUBA Technical Diver Announcements
TECHREV	Technology Review Magazine-upcoming articles
TECSUN-L	SUN Computer Technical Users List
THE_AM	Solid-state physics using the AM book

THETAXI	Discussion List
TLO-SW	MIT TLO Software Requests
TOAST-L	Toastmasters @ MIT
TOUCH	Social touch rugby players at MIT
TPP2001	MIT Tech & Policy Prog - new students 2001
TPR-ANNC	Telecomm Policy Roundtable - Northeast - Announcement List
TPR-NE	Telecomm Policy Roundtable - Northeast
TPR-XX	TELECOMM POLICY ROUNDTABLE - NORTHEAST ADMINISTRATION LIST
TRAIN-L	Training & Development Planning Team
TRANSNTL	US-Asia Transnational Culture Discussion
TRI-ADMN	MIT Triathlon Club administration & announcements
TSS-ISIG	TSS Intermodal Interest Group
TSTOFF-L	Officers of Toastmasters @ MIT
UNDERLIB	The Underground Libraries List
US-FOOTY	Discussing footy in the US
VALMERAS	Valmeras's mailing list
VAXSYM	VAX System Users Group at MIT
VIKINGCF	XRF Viking Group
VTEAMS-L	Virtual Teams Research Discussion
WEBPUB-L	Web Publishers and Designers at MIT
WELLESLEY-EVENTS	Wellesley-College party & event announcements
WESS-CML	WESS College & Medium-sized Libraries Group
WESS-ROM	WESS Romance Language Discussion Group
WFILES	Archive files for the WRITERS list.
WH-ALL	MIT Warehouse-Apts Residents (NW30 Grad Res)
WRITERS	WRITERS
WSCD	Women's Studies Collection Development List
WWOW-L	Wild Women on Wheels
X-FRENCH	Crossroads Faire French Embassy
X-INDIAN	Crossroads Faire Indian Embassy
X-IRISH	Crossroads Faire Irish Group
X-ITALY	Crossroads Faire Italian Group
X-MEAST	Crossroads Faire Middle Eastern Group
X-RUSSIA	Crossroads Faire Russian Group
X-SPAIN	Crossroads Faire Spanish Group
X-VIKING	Crossroads Faire Viking Group
YIPINFO	Modeling Industrial Materials - UCSB; Jan 96
1011	MIT Class 1.011 Spring 2001
15PEARL	Tenants at 15 Pearl Street; Cambridge
16070PUB	MIT 16.070 Public course discussion
1635-T1	MIT 16.35 Team-1 communication list
1801A	MIT 18.01A/18.02A Class Notices
1802	MIT 18.02 Class Notices
1802A	MIT 18.01A/18.02A Class Notices
1803	MIT 18.03 Class Notices
18100	MIT 18.00A Class Notices
2S02B	Raffles Junior College 2S02B List
2WEC-PGM	WESS Paris 2004 Program Committee
2003UA	MIT Class of 2003 Announcements
3A18	MIT Freshman Seminar 3A18
3132000	MIT Course 3.13 mailing list
6BABYLON	6th Babylon Team for MIT 50K Competition
6777INFO	MIT 6.777 Class info and announcements
77TONOW	1977-style punk rock & related music
970GRP3	MIT 9.70 Social Psychology Study Group 3