Financial Market Predictions using Web Mining Approaches

by

Ma Yao

A Thesis Submitted to
The Hong Kong University of Science and Technology
in Partial Fulfillment of the Requirements for
the Degree of Master of Philosophy
in Computer Science and Engineering

July 2009, Hong Kong

Authorization Page

I hereby declare that I am the sole author of the thesis.

I authorize the Hong Kong University of Science and Technology to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the Hong Kong University of Science and Technology to reproduce the thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

_____

Ma Yao

Financial Market Predictions using Web Mining Approaches

By

Ma Yao

This is to certify that I have examined the above MPhil thesis
and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by
the thesis examination committee have been made.

_____

Prof. David Rossiter, Supervisor

_____

Prof. Mounir HAMDI, Head of Department

Computer Science and Engineering

July 2009, Hong Kong

To my parents and friends

Acknowledgments

A special thank you goes to my thesis supervisor Prof. David Rossiter for his guidance and great patience during my master's studies.

Thanks also go to the committee members of my thesis, Prof. Lei Chen and Prof. Jogesh K. Muppala who took the time to read my thesis and offer valuable comments.

Thanks also go to the office staff in the Computer Science and Engineering department. They are always helpful, friendly and patient especially Ms. Connie Lau and Mr. Isaac Ma.

Thanks also go to two fellow students, Mr. Gibson Lam and Mr. Kwan Shun Tsui, who offered me much kind help while the thesis was being written.

Finally, I would like to thank my parents for their love and support over these many years.

Ma Yao

The Hong Kong University of Science and Technology

July 2009

Table of Contents

List of Figures

List of Tables

List of Formulas and Equations

Financial Market Predictions using Web Mining Approaches

by

Ma Yao

Computer Science and Engineering

The Hong Kong of Science and Technology

Abstract

There has been a lot of research on the application of data mining and knowledge discovery technologies into financial market prediction area. However, most of the existing research focused on mining structured or numeric data such as financial reports, historical quotes, etc. Another kind of data source – unstructured data such as financial news articles, comments on financial markets by experts, etc., which is usually of a much higher availability, seems to be neglected due to their inconvenience to be represented as numeric feature vectors for further applying data mining algorithms. With text preprocessing (document representation) technologies, this thesis makes use of this kind of data, specifically financial news articles, to

apply data mining in financial market predictions.

A web-based system has been developed for this purpose. It retrieves financial news articles from the internet periodically and using text mining techniques to categorize those articles into different categories according to their expected effects on the market behaviors, then the results will be compared with the real market data. The system allows the users to select different algorithms for each phase of the text mining process, so that the results for different combinations of algorithms can be compared and the best one can be selected by observing the results. This combination of algorithms can be applied to do financial market prediction in the future.

The text mining process has three phases totally, keyword extraction, keyword weighting and classification. Keyword extraction is to extract a keyword list from a corpus, according to the ability of each word to distinguish the category of a document from others. The system has implemented the following keyword extraction algorithms: document frequency threshold, entropy method, information gain, gain ratio, chi-square statistic and mutual information. Keyword weighting is to transform a document into a numeric feature vector according to a keyword list generated previously. Each word in the keyword list will be assigned a weight according to the number of occurrence of this keyword in a document. The system has implemented Boolean weighting, term frequency (TF) weighting, term frequency times inverse document frequency weighting (TFxIDF), LTC weighting and TFC weighting methods. For classification algorithms, this system has implemented

Navie Bayes Classifier and Support Vector Machines, but the experiment is focused on the former classifier.

The system collected news articles and market data and was tested to compare different algorithms for each phase. As there are a huge amount of combinations of different algorithms, we adopted a greedy approach to find out the optimized combination. Particularly, we vary the algorithms or parameters for one phase of the text mining, and fix all the others. Then by observing the results, select the best algorithm/parameter and assume that it is also the global optimized algorithm for this phase no matter how the algorithms/parameters of other phases vary. The results are presented and analyzed in this thesis for selecting the best combination of algorithms for text mining in financial market prediction.

CHAPTER 1

INTRODUCTION

1.1 Thesis Motivation and Objective

There has been a lot of research on the application of data mining or knowledge discovery in financial market predictions. In those research, various data mining techniques are applied to predict stock price trends, index values, currency exchange rates, volatilities etc. However, most of the studies [2][3][4][5] in this area are based on numeric and structured data, such as historical price quotes, financial statements, interest and tax rates, or some other quantifiable figures. Research on mining textual and unstructured information like news, recommendation and comments from experts, postings from online forums and chat rooms, personal blogs and so on seems to be only an emerging area of study [1]. However, available textual information on the internet is usually of huge quantity and is much more informative than purely numeric information. For example, one can identify not only the market behavior from a piece of financial news reasons but also understand why the market behaves this way [11]. The purpose of this thesis is to explore this emerging research area, and to give comprehensive experiments and comparisons on applications of various textual data mining techniques on financial market predictions.

1.2 General Architecture of Market Prediction by Text Mining Approaches

By reviewing the past research in this area, it can be summarized that a text mining

system for financial market prediction usually follows a common pattern. The process can be generally divided into a learning phase, where historical texts and time series of historical market quotes, such as stock prices, index values etc., are collected to train a classifier, and an operational phase, where newly published articles are input to the previously developed classifier to predict the future trends or other features of the market. Figure 1 and Figure 2 show the major steps of the two phases respectively.

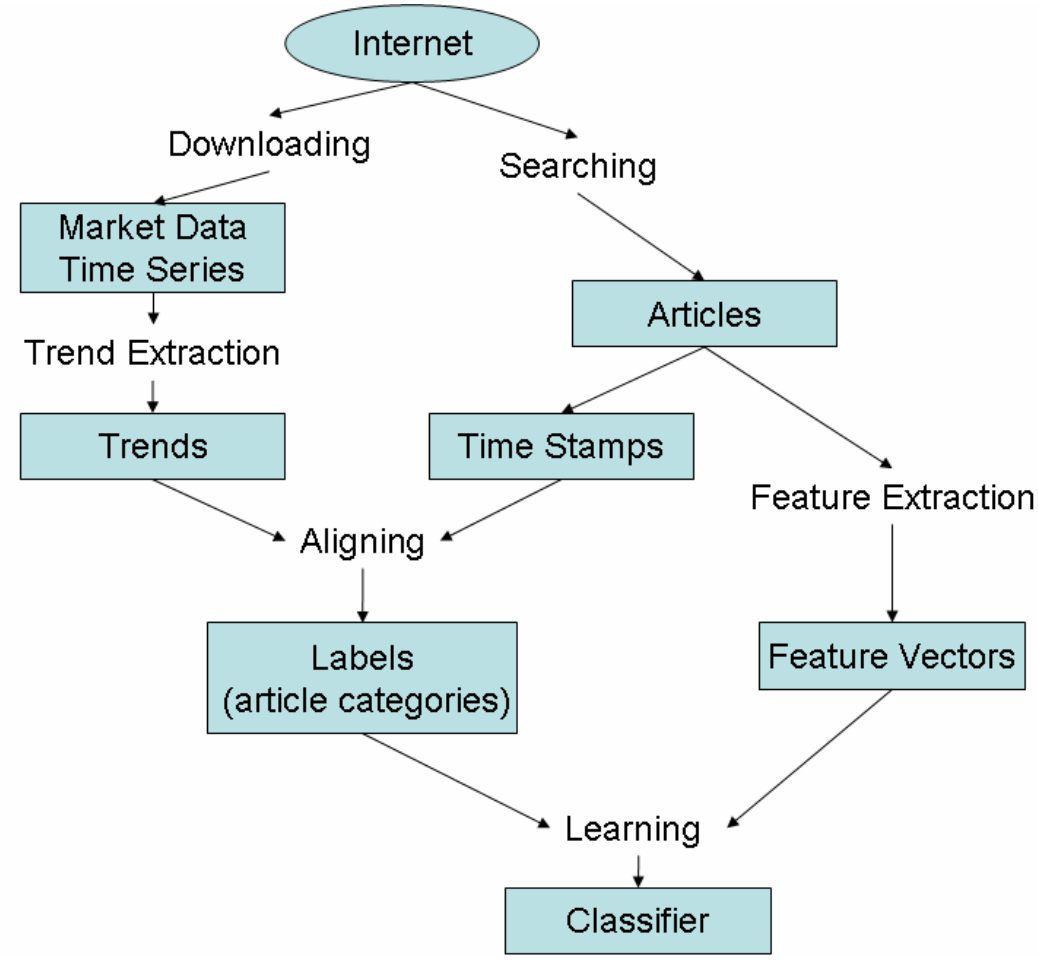

Figure 1: General Model – Learning Phase

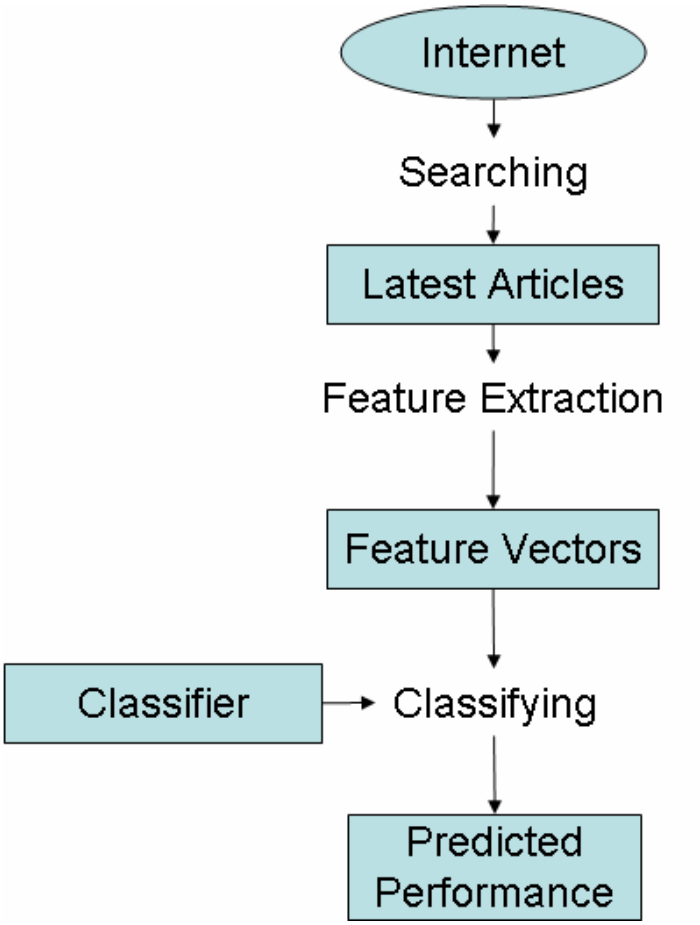

Figure 2: General Model - Operational Phase

As shown in Figure 1, in the learning phase, texts and time series are all retrieved from the internet, or the pre-stored database. The first step to process the text is called labeling. The purpose of this step is to categorize the articles according to the effect that the article has to the market, e.g. either rising, steady, or falling [11]. This step can be done either manually by human experts [13], in which way the leftmost part of Figure 1 (including downloading, time series, trend extraction, trends and aligning) is no longer required, or automatically by the following two steps.

1) Extract the trends of the historical market behavior

Depending on the horizontal resolution of the time series, this step can be as

simple as considering only the closing prices of consecutive trading days [11][13][17], or as difficult as doing some segmentation of the time series with a piecewise linear regression [12][16].

2) Align the articles to the trends

Articles are then aligned to a specific segment of the time series according to their time stamps. They are then labeled with the trends of their respective segments. The alignment can be either delayed, meaning each article is aligned to a segment which is a little bit later than the publication time, or immediate, meaning each article is aligned to a segment which is exactly corresponding to the time that the article is published. Mittermayer et al. [1] did not explicitly compare this feature among those prototypes. The supporters of the former alignment method argue that a newly published article takes time to have impact to the market. However, as the developers of most of the prototypes [16][18] emphasize the Efficient Market Hypothesis (EMH), they believe that the later alignment method is generally preferred. Interestingly, an important inference of EMH states that market behavior is unpredictable, so all the research works on market prediction, not only those based on textual but also numeric data mining, should be based on doubt of the EMH to some extent.

Another issue worth noting is that some of the prototypes do not predict the features represented in the time domain, such as stock prices, index values, exchange rates

[15] etc., but other properties of the market, like volatility. If this is the case, the leftmost part of Figure 1 will be slightly different accordingly. For example, if the objective is to predict the volatility, the “trends extraction” and “trends” in Figure 1 will be replaced with “volatility calculation” and “volatility ranges”.

Text pre-processing is the most essential part for text mining. It is also the most interesting part as it has the greatest variety in selecting the technologies to achieve this procedure. And this procedure is contained in both the learning phase and the operational phase. The purpose of this procedure is to extract the features of each article for classifier training and classifying in the later steps of learning phase and operational phase respectively. All the eight prototypes studied by Mittermayer et al. [1] are all using keywords, or word tuples (combinations of keywords), as the features of the articles. They count the number of occurrences of each keyword, and transferred the numbers into a weight by the application of some algorithm. Some of the prototypes generated the keywords manually by domain experts [11][13][15], whereas some prototypes do it automatically [12][14][16] or semi-automatically [17]. And the weight generation methods are also different. The number of keywords being studied also varies from one prototype to another. The currently existing techniques of these two steps will be discussed in detail in Section 2.2.

The final step of the learning phase is to train the classifier. The labels and features obtained from earlier steps constitute the training set. A specific learning technology is applied to train a classifier, the mapping rules from features to labels, to category the newly published articles, and then accordingly predict the future behavior of the

market. Among the eight prototypes that Mittermayer et al. [1] studied, Naïve Bayes [11][12][14], decision rules [13][15], and Support Vector Machines (SVM) [16][18] learning technologies are adopted.

1.3 Thesis Organization

Chapter 2 gives a general review on the previous related work done on text mining applications on financial market prediction as well as on text mining techniques, including both text preprocessing techniques and text categorization algorithms.

In Chapter 3, the text mining system for financial market prediction for this thesis is described in details, including its mechanisms, architecture, components, technologies, programming languages, code structure and functionalities. Readers will know how the system retrieves articles and market data from the internet, how it stores and analyzes each article, how the classifier is trained to predict market behavior, how the prediction and analysis results are displayed, etc. by the system.

The way how the system is tested is discussed in Chapter 4. For each phase of the text mining process, the algorithms attempted are listed. The chapter also presented what underlying market values are selected and the selection of the sources for training data.

In Chapter 5, the results of our experiments are presented. We analyze the results by comparing all results given by different combinations of algorithms and compares which combination gives the best result for market prediction purpose.

Finally, the conclusions are drawn and some future work and research directions are

suggested in Chapter 6.

CHAPTER 2

PREVIOUS WORK

2.1 Previous Research on Text Mining on Financial Market Prediction

Although text mining for financial market prediction is an emerging area as discussed above, there are indeed some previous research in this area can be found. This section will present a general survey of currently existing research in this emerging area and compare the techniques that they applied in different steps of the architecture mentioned previously in Section 1.2.

Mittermayer et al. [1] did a comprehensive survey in this area, which studied eight prototypes [11][12][13]…[18] developed for text mining in financial market predictions. Their survey report gave summaries and comments for each of the prototype and made comparisons between them in terms of their objectives, involved markets, adopted techniques, performances etc, as shown in Table 1. However, there is still some research [19][20][21]…[23] which is not included in their survey report.

The survey report presented in this section will be based on the one done by Mittermayer et al. [1] and do some further discussions on the similarities and differences of the prototypes and attempt to find out some deficiencies of existing works.

	Prototype 3.1.	Prototype 3.2.	Prototype 3.3.	Prototype 3.4.	Prototype 3.5.	Prototype 3.6.	Prototype 3.7.	Prototype 3.8.
Prototype idea								
Aims to forecast…	price trends	price trends	volatilities	price trends	price trends	price trends	volatilities	price trends
Underlying	equity index	single stock	single stock	single stock	exchange rate	single stock	single stock	single stock
Forecasting horizon	24 hours	1 hour	N/A	1 hour	3 hours	1 hour	N/A	15 minutes
Text mining parameter								
Feature definition	manually	automated	manually	automated	manually	automated	automated	semi-automated
Number of features	423	N/A	145	1000	400	N/A	200	85
Feature granularity	tuple (words)	terms	tuple (terms)	single words	tuple (words)	single words	single words	tuple (terms)
Primary classifier	Naïve Bayes	Naïve Bayes	decision rules	Naïve Bayes	decision rules	linear SVM	regression	polynomial SVM
Number of categories	3	5	39	3	3	5 (training: 3)	2	4 (training: 3)
Input data								
Information age	2 - 15 hours	0 hours	0 - 24 hours	0 hours	0 - 2 hours	0 hours	0 hours	0 hours
Text analyzed	headline, body	headline, body	headline	headline, body	headline	headline, body	headline, body	headline, body
Labeling	automated	automated	manually	automated	automated	automated	automated	automated
Price frequency	daily close	10 min.	daily close	10 min.	60 min.	intraday	daily close	15 sec.
Test								
Period investigated	1997 - 1998	1999 - 2000	2001 - 2002	2001 - 2002	1993	2002 - 2003	1999 - 2002	2002
Training/Test split	3 months rolling	3 / 1.5 months	8 / 5 months	5.5 / 2 months	1 month rolling	6 / 1 month(s)	cross validation (90% / 10%)	cross validation (90% / 10%)
Prototype vs. random	44% vs. 33%	N/A	N/A	40% vs. 33%	50% vs. 33%	N/A	61% vs. 50%	45% vs. 33%
Roundtrips per year	< 600	> 100'000	(200)	< 6000	N/A	N/A	N/A	< 500
Profit per roundtrip as reported	13 bps	23 bps	(first phase: 10 bps)	10 bps	N/A	N/A	N/A	29 bps
Market	DJIA, Nikkei, FTSE, HS, ST	127 stocks (USA)	constituents Russell 3000	constituents DJIA	USD/DEM and USD/JPY	614 stocks (Hong Kong)	constituents DAX100	constituents S&P500

Table 1. Comparison of Main Properties of the Prototypes.

Table 1: Comparison of Main Properties of the Prototypes
(Directly imported from [1])

Each of the steps of the general architecture described above contains a certain range of selections. All these will provide us a clue to find out a new research direction. The following part of this chapter will discuss them one by one.

2.1.1 Objectives

2.1.1.1 Properties to Predict

Most of the existing works are aimed to predict the price or index value trends. The system that Seo et al. [13] developed performed rather poor and focused on forecasting the volatility of the market instead [1]. In terms of application, predicting the volatility is also desirable because this property can also give us useful information. For example, in Black-Schole model, option prices can be derived from several parameters including volatility, which is the hardest one to obtain. Other properties of markets, such as volumes, may also be areas worth exploring, but they are now found to be deficiencies in currently existing research.

2.1.1.2 Market Types

Most of the models studies stock prices or index values, with the following a few exceptions. Peramunetilleke et al [15] examines currency exchange rates. Yu et al. [19] studied commodity (crude oil) prices [19] whereas Lua et al. [20] studied utility (electricity) prices [20]. The research of derivative markets like futures, options or exotics is not yet found, and might become a new research direction.

2.1.2 Text Sources

2.1.2.1 Selection Ranges

Nearly all the past models consider only news articles or even only news headlines [12][15] to be their sources of text. Other sources like postings on related forums and chat rooms, comments, recommendations from experts or portfolio managers and companies' financial statements has not been adopted so far.

2.1.2.2 Source Acquisition Methods

Wüthrich et al. [11] downloads articles for selection from indicated news sources containing financial analysis reports and information about what happened on world's stock, currency and bond markets. Some other models manually downloaded the articles first and then feed them to the classifier. Anyway, they are all replying on a static website to get news articles. This is considered to be improved as market participates will not only react according to the articles from those sources indicated by those models. Instead, they will browse randomly on the internet or other media. So a dynamic downloading scheme is desired in the future works. This may involve a crawler to search for information on the web, and then some mechanism need to be added to identify whether the obtained article is related to the market we are observing or not.

2.1.3 Time Series

2.1.3.1 Number of Time Series

It is noticeable that Fung et al. [16] examined multiple time series obtained from different stocks, and then tried to find out the correlations between them. For each

stock, they selected its “potential related stocks”, whose behavior this stock may affect, and when doing trend prediction, they will also take the news related to those stocks into account.

2.1.3.2 Horizontal Resolutions

Mittermayer et al. [1] compared this parameter among all eight prototypes they studied. This information is shown in the “price frequency” row in Table 1. For those prototypes that has price frequencies more often than “daily close”, they need to apply some techniques to extract the trends as described in the previous chapter.

2.1.4 Text Pre-Processing (Feature extraction) Technologies

The prototypes found by this survey are all based on keywords or key word tuples (a sequence of words) to convert text to feature vectors. The diversity is that in some models, the keywords are generated automatically while in the others, they are selected manually by domain experts. And the numbers of keywords among those models are also different. Another diversity is that they weight each keywords by different approaches. Non-keyword based feature extraction method, such as semantic prosody [10] techniques, is not yet found so far. A comprehensive study of popular text pre-processing technologies is presented in Section 2.2.

2.1.5 Learning Technologies

Naïve Bayes, Support Vector Machine, Rule Induction and Regression are respectively adopted by the eight prototypes. This information can also be found in Table 1. The major technologies for learning and classifying will be discussed in

Chapter 2.3.

2.2 Approaches for Text Pre-processing

The objective of text preprocessing is to transform a text-based document into a numeric feature vector in order to apply standard AI algorithms. This process is also called “document indexing” (p.83, [25]), “document representing” (p.68, [26]). After text preprocessing, the following steps of text mining will be the same as those of general data mining problems. This step is essential to the quality of text mining.

There are two major phases for text preprocessing, keyword selection and keyword weighting. The two phases are relatively independent and we can select one approach for each phase without considering too much of our approach selection of the other phase.

2.2.1 Keyword Selection

Not every word in a document has significance for text mining purposes. The most obvious examples are those which we called “stopping words”, such as *for*, *to*, *an*, *it*. A keyword will be selected either manually according to human knowledge about the specific domain that is being studied, or automatically by some statistical or probabilistic algorithms. Later, each keyword will be corresponding to an element of the future feature vector. So basically, the purpose of keyword selection is to reduce the dimensionality of the feature vector representing the documents.

Here is a summary of keyword selection methods.

1) Manual Selection

As keyword selection is a one time work rather than a day-to-day operation, it is possible to get it done by human rather by computers. This process requires the domain knowledge of the specific area that is being studied.

2) Automatic Selection

Most of the automatic selection approaches are based on some evaluation of each word in the whole corpus to be analyzed. In the selection process, each word will be assessed by some statistical methods or formulas and a numeric scalar value will be returned representing the "qualification" of the word to be used as a keyword. Then a specific threshold is set according to the expected size of the keyword list and those words getting a value higher than the threshold will be selected as keywords.

i) Unsupervised Methods

- Document Frequency Threshold

This is almost the simplest way of automatic keyword selection methods. It simply compares the document frequency of each word in the vocabulary. Document frequency of each word means in how many documents this word appears, divided by the total number documents in the whole corpus.

- Entropy Method (sec.2.1.2 of [27])

In Formula 1, $W(t)$ measures the entropy of each word in the vocabulary. Entropy, to some extent, represents how well a word is suited to separate documents.

$$W(t) = 1 + \frac{1}{\log |D|} \sum_{d \in D} P(d,t) \log P(d,t) \ldots\ldots$$

1)

, where

$$P(d,t) = \frac{tf(d,t)}{\sum_{l=1}^{n} tf(d_l,t)} \ldots\ldots$$

2)

To select a number of keywords according to the entropy value calculated by Formula 1, we can apply a simple greedy algorithm (sec.2.1.2 of [27]), as follows.

- Outer Loop
  - Initialize all documents as “unmarked”
  - Inner Loop: Until all documents are marked
    - Find the next unmarked document d_i
    - T = the term with the highest entropy
    - Add T to the keyword list
    - If desired number of keywords has been obtained, terminate the algorithm
    - For each document d_j containing T, mark d_j
- Return the keyword list obtained

ii) Supervised Methods

The major purpose of text mining is to classify the documents into several categories. If the training document set is provided with their expected categories, this information can be utilized to improve the quality of keyword selection.

For most of the evaluation functions in this sector, these probabilities usually need to be evaluated as the input of the function. (p.84 of [25])

- $P(t_k, c_i)$ indicates the probability that, for a random document x, term t_k occurs in *x* and *x* belongs to category c_i.
- $P(\overline{t_k}, c_i)$ indicates the probability that, for a random document x, term t_k does not occur in *x* and *x* belongs to category c_i.
- $P(t_k, \overline{c_i})$ indicates the probability that, for a random document x, term t_k occurs in *x* and *x* does not belong to category c_i.
- $P(\overline{t_k}, \overline{c_i})$ indicates the probability that, for a random document *x*, term t_k does not occur in *x* and *x* does not belong to category c_i.

These probabilities could usually be estimated by the following measurements.

- *A* indicates the number of documents that contains *t* and belongs to category *c*.
- *B* indicates the number of documents that contains *t* and does not belong to category *c*.
- C indicates the number of documents that does not contain *t* and belongs to category *c*.
- D indicates the number of documents that does not contain *t* and does not belong to category *c*.
- *N* indicates the total number of documents in the corpus.

So the probabilities can be estimated using Formula Set 3.

$$\begin{cases} P(t,c) \approx \dfrac{A}{N} \\ P(t,\bar{c}) \approx \dfrac{B}{N} \\ P(\bar{t},c) \approx \dfrac{C}{N} \\ P(\bar{t},\bar{c}) \approx \dfrac{D}{N} \end{cases} \cdots\cdots \quad 3)$$

Usually, supervised evaluation functions of a term depend on a specific category (in $f(t_k, c_i)$ form). It gives a value representing how the term can distinguish documents belonging to this category from those not belonging to this category. However, for the keyword selection purpose, we need to "assess the value of a term in a 'global', category-independent sense, a 'globalization' technique is applied so as to extract a global score from the $f_{glob}(t_k)$ from the $f(t_k, c_i)$ scores relative to the individual categories." (p.85 of [25]). There are 3 common formulas for this globalization purpose (p.85 of [25]):

1) Sum

$$f_{glob}(t_k) = \sum_{i=1}^{m} f(t_k, c_i) \quad 4)$$

2) Weighted Sum

$$f_{glob}(t_k) = \sum_{i=1}^{m} P(c_i) f(t_k, c_i) \quad 5)$$

3) Maximum

$$f_{glob}(t_k) = \max_{i=1}^{m} f(t_k, c_i) \tag{6}$$

The supervised evaluation functions are summarized as follows.

1) Information Gain (p.85 of [25], [28], [29])

$$IG(t_k, c_i) = \sum_{c \in \{c_i, \overline{c_i}\}} \sum_{t \in \{t_k, \overline{t_k}\}} P(t, c) \log \frac{P(t, c)}{P(t)P(c)} \tag{7}$$

2) Chi-Square Statistic (p.85 of [25], [28], [29])

$$\chi^2(t_k, c_i) = \frac{N \times (AD - CB)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)} \tag{8}$$

Or

$$\chi^2(t_k, c_i) = \frac{[P(t_k, c_i)P(\overline{t_k}, \overline{c_i}) - P(t_k, \overline{c_i})P(\overline{t_k}, c_i)]^2}{P(t_k) \times P(\overline{t_k}) \times P(c_i) \times P(\overline{c_i})} \tag{9}$$

3) Mutual Information [28][38][39][40]

$$I(t_k, c_i) = \log \frac{P(t_k, c_i)}{P(t_k) \times P(c_i)} \tag{10}$$

4) Gain Ratio (p.85 of [25])

$$GR(t_k, c_i) = \frac{IG(t_k, c_i)}{-\sum_{c \in \{c_i, \overline{c_i}\}} \Pr(c) \log \Pr(c)} \tag{11}$$

5) Term Strength [27][36][37][38]

$$s(t) = \Pr(t \in y \mid t \in x) \qquad (12)$$

iii) Alternative methods for dimensionality reduction

Another way to reduce the dimensionality of the feature vectors are by combining or transforming the original features into another smaller set of features (p.69 of [26]). This can be done by making use of polysemy, homonymy and synonymy. The following two approaches are found to achieve this objective.

- LSI: Latent Semantic Indexing (sec.2.3.2 of [29], [31], [32])
- Filtering, Lemmatization and Stemming (sec.2.1.1 of [27])

2.2.2 Term Weighting

The value (denoted as a_{ik}) of the element of a keyword t_i in a document d_k is determined by different term weighting algorithms. In the following summary of term weighting algorithms, we denote f_{ik} to be the term frequency of a term t_i in the document d_k, which means the number of occurrence of term t_i in the document d_k divided by the total number of words in the document d_k. We denote n_i to be the number of documents that contains keyword t_i. We denote N as the total number of documents in the corpus. So $\frac{n_i}{N}$ means the percentage of documents containing term t_i among all documents in the corpus, which we call document frequency.

1) Boolean Weighting (sec.2.2 of [29])

This is the simplest method. It assigns the weight to be 1 if the word appears in the document, 0 if not. The weight of a term can be represented in term of the term frequency by Formula 13.

$$a_{ik} = \begin{cases} 1....if\ f_{ik} > 0 \\ 0...otherwise \end{cases} \quad (13)$$

2) Term Frequency (sec. 2.2 of [29])

This method considers the term frequency as the sole factor for weighting a term. The weight of a term is simply given by Formula 14.

$$a_{ik} = f_{ik} \quad (14)$$

3) Term Frequency Times Inverse Document Frequency (sec.2.2 of [29], [30])

This is the most popularly used term weighting method. The basic idea is that a term is much more important if it only appears in a smaller number of documents in a corpus. For example, the terms such as “today”, “great” are much less important because it appears in almost every document, it may even occupy a relatively large proportion of each document. So when weighting a term, the term frequency should be offset by the frequency of the term in the corpus, namely, document frequency. The weight of a term with term frequency f_{ik} and a document frequency $\frac{n_i}{N}$ is given by Formula 15.

$$a_{ik} = f_{ik} \log(1/\frac{n_i}{N})$$

15)

4) TFC-Weighting (sec.2.2 of [29], [39])

This method considers the length of document into account. Assume there are totally *M* terms in the vocabulary, the weight of a term can be represented by Formula 16.

$$a_{ik} = \frac{f_{ik} \log(\frac{N}{n_i})}{\sqrt{\sum_{j=1}^{M} [f_{jk} \log(\frac{N}{n_j})]^2}}$$

16)

5) LTC-Weighting (sec.2.2 of [29], [40])

This is similar to the previous method. Logarithm is used to replace the raw word frequency. By this method, the weight of a term is given by Formula 17.

$$a_{ik} = \frac{f_{ik} \log(\frac{N}{n_i})}{\sqrt{\sum_{j=1}^{M} [\log(f_{jk} + 1.0) \log(\frac{N}{n_j})]^2}}$$

17)

6) Entropy Weighting (sec.2.2 of [29], [41])

This method is based on information theory. The weight of a term is given by Formula 18.

$$a_{ik} = \log(f_{jk} + 1.0) \times \left(1 + \frac{1}{\log(N)} \sum_{j=1}^{N} [\frac{f_{ij}}{n_i} \log(\frac{f_{ij}}{n_i})]\right)$$

18)

, where

$$\frac{1}{\log(N)}\sum_{j=1}^{N}[\frac{f_{ij}}{n_i}\log(\frac{f_{ij}}{n_i})] \quad 19)$$

is the average uncertainty or entropy of word *i*. This quantity is -1 if the word is equally distributed over all documents and 0 if the word occurs in only one document.

2.3 Text Categorization Approaches

The final step of text mining is to classify the feature vectors transformed from text documents into different categories. As this step is the same as common data mining algorithms, and the quality of text mining is mainly related to the selection of the former two steps, we will not discuss this section in details. Popular classification algorithms are as follows.

- Naïve Bayes Classifier
- Neural Networks
- Support Vector Machines

CHAPTER 3

SYSTEM DESCRIPTION

3.1 System Objective

The “Financial Market Forecast System” (FMFS) is developed by the author of this thesis to predict financial market behaviors by text mining approaches for both experimental (research) and practical (business) purposes.

FMFS directly gives the result of its prediction for today’s market behavior, so it can be referred by the users from the business to do trading. Users can also see the confidence for today’s prediction by FMFS, so that they will have an idea about how much they can trust FMFS’s prediction.

FMFS provides a high flexibility for the researchers to configure the algorithms, approaches, training data ranges, etc. of each phase of the whole text mining process. It also allows the users to see the technical data, system details, and results and evaluation chart and figures. This made FMFS very helpful for research purpose, so that the users can make comparisons between all available algorithms and approaches, and see which one or combination is the most suitable for financial market prediction purpose.

3.2 System Mechanisms

Basically FMFS also follows the general architecture described in the first chapter of

this thesis. It searches, downloads and accumulates articles from the major financial news websites, such as Bloomberg, CNN, BBC, etc., periodically. It also acquires the real market data from the Internet. Then FMFS aligns each article to a market trend according to the time when the article is published and the time when the market trend really took place. This information combined together will be used to trained a classifier using AI technologies, and then when a new article is published, it classifies the article using the previously trained classifier, to predict future market behavior.

FMFS comprises several components, which are introduced in the following section. In the back-end, it includes a spider, an article downloader, and a market data updater (Section 3.3.1 – 3.3.3). The other components are in the front-end, so the algorithms and approaches used by those components can be configured by the users.

3.3 System Components

3.3.1 Spider

This part is used for collect the URLs of the newly published news articles from several predetermined sources (financial news websites). The source information is stored in an xml file, which comprises the id and the root URL of each source and the URL pattern of the article pages of each source. The URL patterns are described using regular expressions. This part is a server side web application. Once a specific HTTP request is sent to the server, it will do the following for each source stored in the xml file:

1) Get the root page of the source by the root URL, such as “www.bloomberg.com”.

2) Search for all the hyperlinks in the root page, collect those that match the URL pattern of this source.

3) For each link found, check whether this link already exists in the links.xml file. If not, add it into the file, and mark this link as “unprocessed” (meaning that it is to be processed by the following part of FMFS).

The architecture of the spider component is shown in Figure 3.

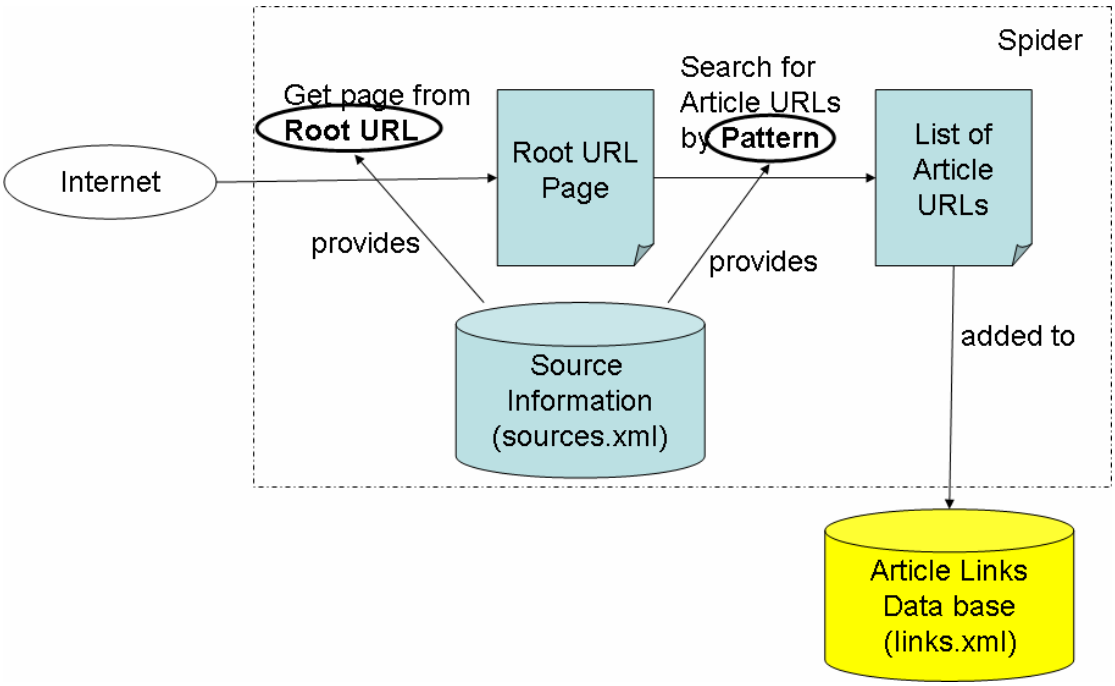

Figure 3: Architecture of Spider

3.3.2 Article Downloading and Processing

The links that are stored in links.xml will be processed by this component. For each link that is found by the spider part, the web page of the link will be downloaded from the Internet. Then the actual article text will be extracted from the HTML source file, this involves filter out the HTML tags, scripting codes, and other information such as

advertisement, hyperlinks, etc. The text is saved in a txt file named by the ID of the article. In addition, the time stamp that the article is published is also extracted from the HTML source code. The ID, source, and the time stamp of each article will be stored in articles.xml, which is called “text information database”. If the whole process is successful, the flag to represent whether this link has been processed or not in links.xml file will be marked as “processed”, so that next time this component will not process the link once again.

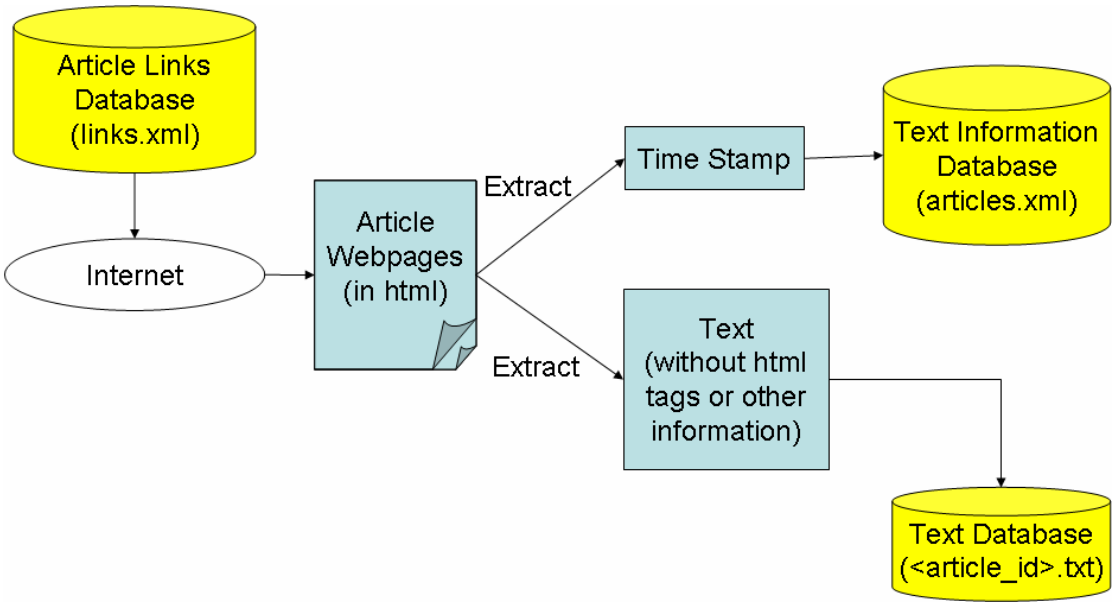

Figure 4: Architecture of Article Downloading and Processing

3.3.3 Market Information Updating

The real market data will be downloaded automatically from a pre-configured financial website and will be stored in .csv format every day when a new closing value is available. For each trading day, the date, the opening value, daily high, daily low, closing value and the volume will be stored.

3.3.4 Keyword Extraction

Keyword extraction methods are divided into two types, supervised methods and unsupervised methods. The detailed approaches have already been introduced in the previous chapter. In this system, the keywords, called keyword set, are all extracted from the existing articles that the system has already accumulated, and then they are stored in a text file. The users can generate a keyword list for future text processing at any time, and choose an approach from those described in the previous chapter. User can also configure the size of the keyword set and the articles that to be used to extract keywords.

The unsupervised and supervised keyword extraction processes are shown by in Figure 5 and Figure 6 respectively.

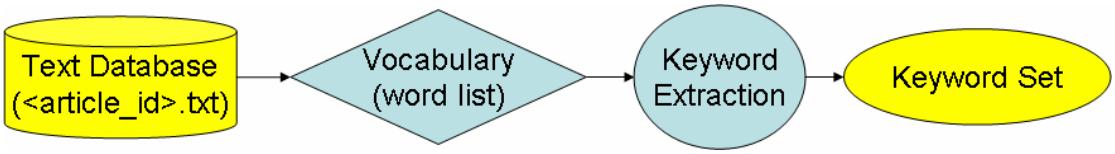

Figure 5: Architecture of Unsupervised Keyword Extraction Methods

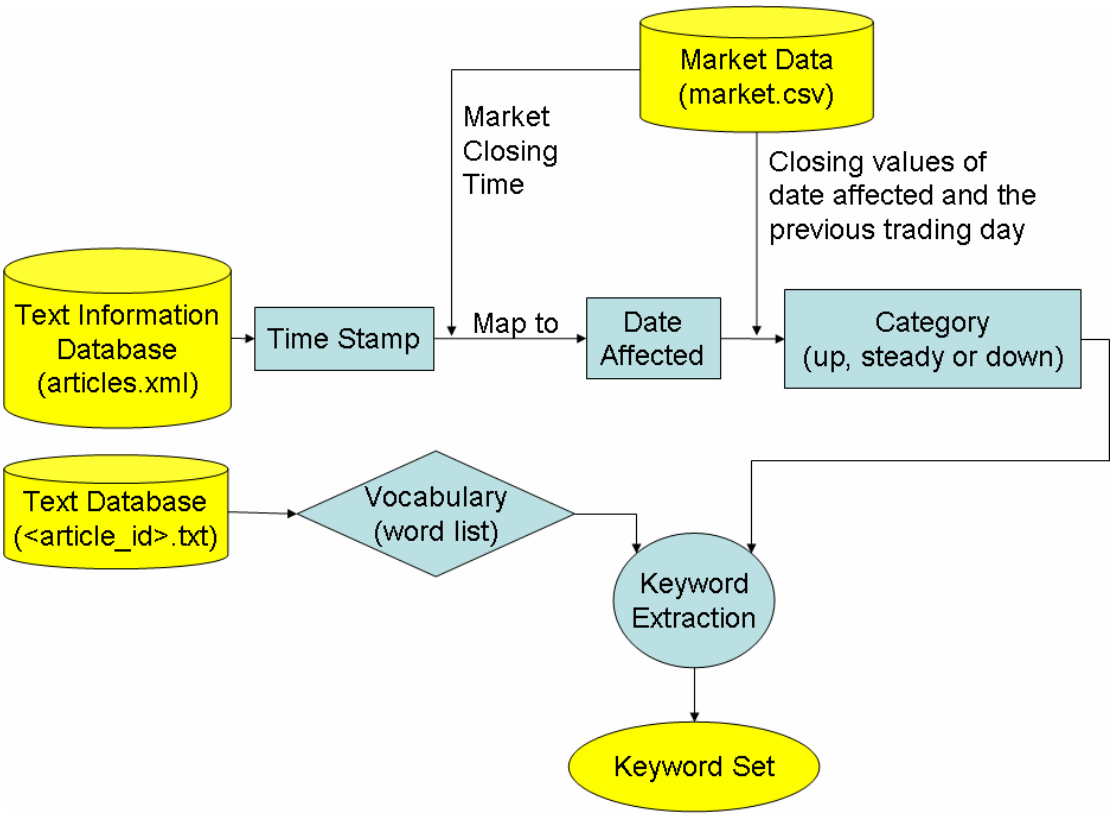

Figure 6: Architecture of Supervised Keyword Extraction Methods

3.3.5 Classifier Training

User can train a classifier at any time for future text processing and choose the pre-extracted keyword list used for text preprocessing, term weighting algorithms, classification algorithms, and training data set.

During the training process, first, the time stamp of each article in the training set is used to determine which trading day that this article is related to. The date is the first closing time of the market after the article is published. For example, if the article is published at 15:00, 3rd March and the market closes at 16:00 every day, the system will assume the article has affected the market behavior of 3rd March. If the article is published at 21:00, 4th March, and 5th March is not a trading day, the system will

assume the article has affected the market behavior of 6th March. Then market closing value of the date that the article has affected will be compared to the closing value of the previous trading day. According to the ratio of the two values, a category (either up, steady, or down) will be assigned to the article. For each article, a feature vector is also extracted by a combination of keyword list and term weighting algorithms that the user has specified. With those feature vectors and its corresponding categories, a user specified supervised classification algorithm is applied to train the classifier.

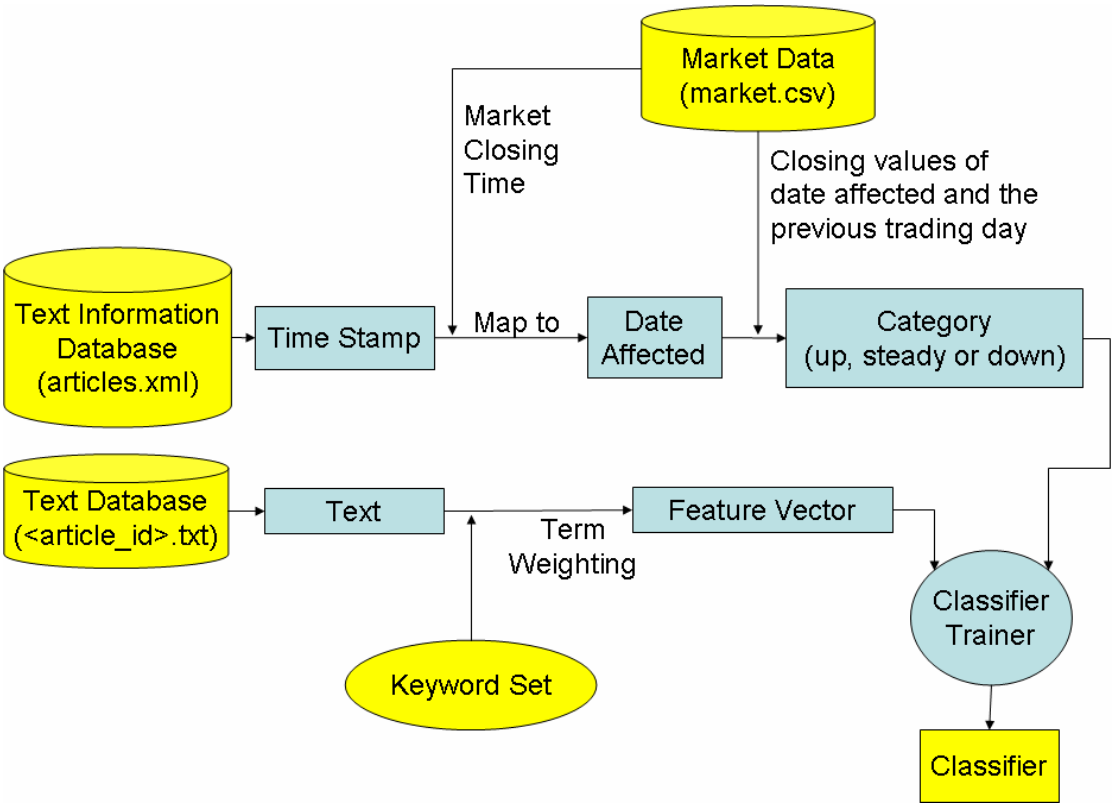

Figure 7: Architecture of Classifier Training

3.3.6 Classifier Operation

Once a classifier is trained, the remaining articles (those not used for training data) are classified by the previously trained classifier. The classification result will be

compared to the real market data. A summary of classification results will be generated in forms of figures and charts. The confidence value of the classification of each article is also shown to the users.

The architecture of this component can be described by Figure 8.

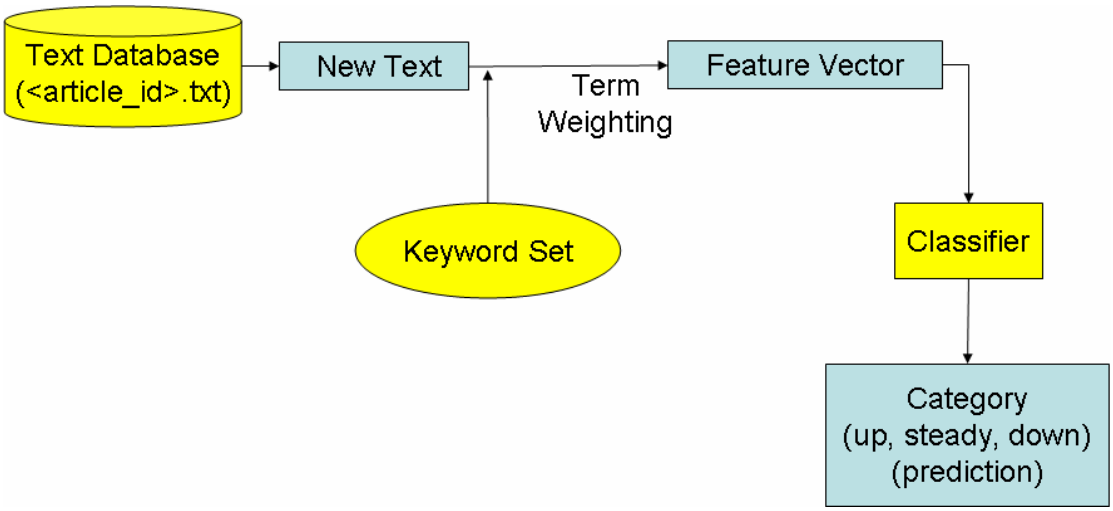

Figure 8: Architecture of Classifier Operation

3.4 System Architecture and Implementation

The components described in the previous section show how FMFS is conceptually constituted, which does not infer how the system is actually implemented. In fact, FMFS is composed of three relatively independent subsystems. This section will present how the three subsystems implement the function of all components, and integrate them into one single system.

3.4.1 Spider Component

The spider component of the system was developed from an existing web-application called SASSLO, using JavaScript and PHP. The links for newly published articles

will be added to the links.xml file once a specific HTTP request is sent to the server. In this system, the request is sent by the following component described by the following section.

3.4.2 Information Retriever

Information Retriever is a standalone Java application performs periodical tasks including retrieving and doing initial process of newly published articles from the Internet for the system, and updating the latest market data. This application implemented the components described in Section 3.3.2 and Section 3.3.3, namely "article downloading and processing" and "market data updating" jointly. Once the application starts running, 3 schedulers will be set up to periodically trigger the following 3 jobs respectively.

1) Send an HTTP request to the spider as described in the previous section.

2) Trigger the article downloading and processing component.

3) Trigger the market data updating component.

The first job is triggered at the 00 min of each hour, and five minutes later, the second job is triggered. Five minutes is a sufficient time for the first job to finish. The third job is triggered every day.

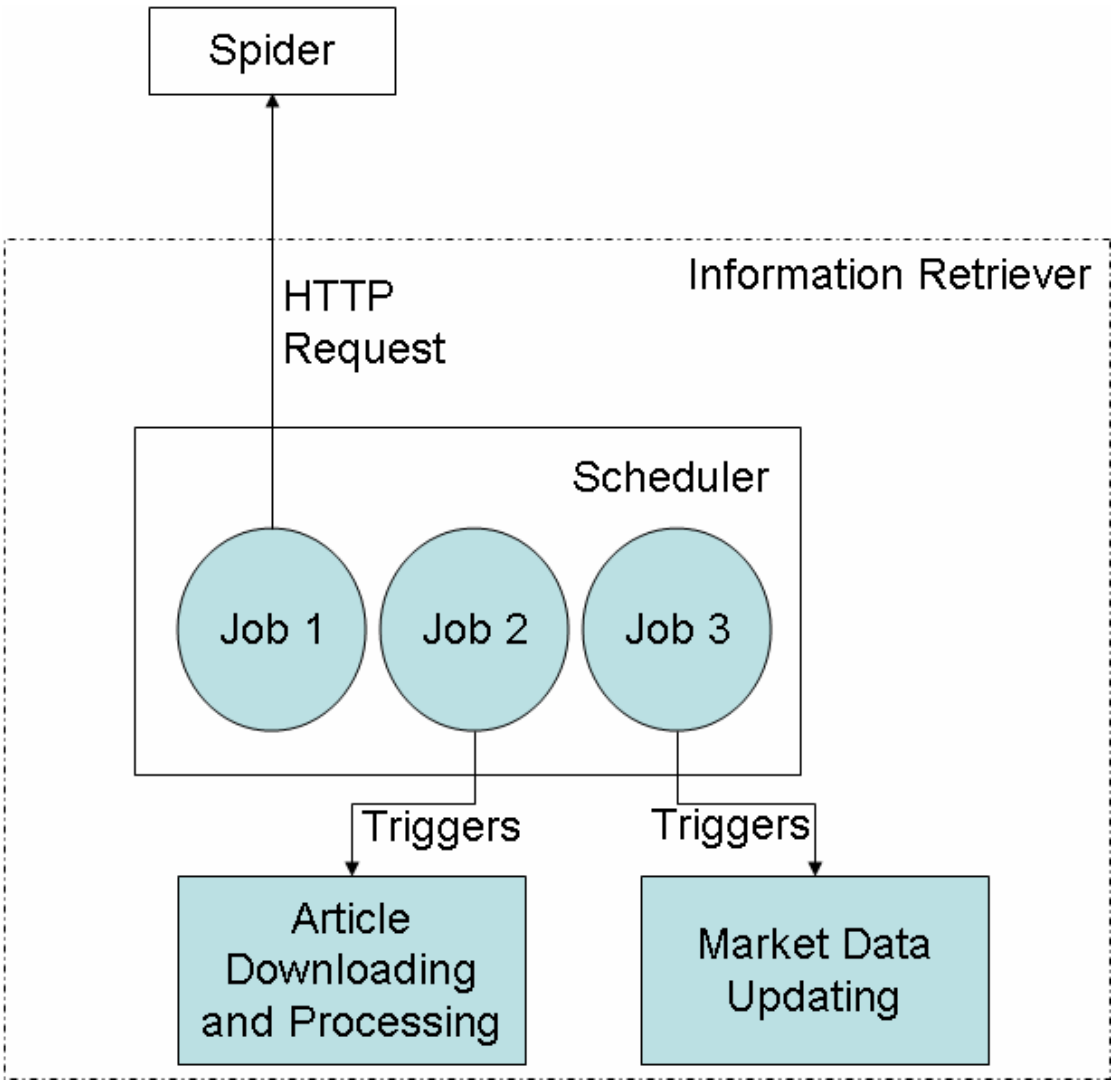

Figure 9: Architecture of the Information Retriever

3.4.3 Other Components

The other components of the system include keyword extraction, keyword weighting, classifier, (both training and operational phase), and the user interface.

This is the part that handles user interaction. It is a web-based application developed using Java Servlet Technology. Users can choose the approaches of keyword extraction, keyword weighting, select training data, classification algorithms and see classification (prediction) results through a web interface. Users can also see historical

market data, keyword list, and the parameters of classifiers. For each article, the article content, time stamp, feature vector, classification result (if it is not used for training data), and its confidence are also visible to the users.

CHAPTER 4

SYSTEM TEST

4.1 Fixed Configurations

Once the system is developed, we have conducted a comprehensive experiment to test and compare all algorithms that we have implemented. However, the following settings are fixed and will not be tuned during our experiments.

4.1.1 Categorization Scheme

As described in above chapters, the articles will be categorized according to their influences to the market. In our experiments, we define three categories as the scheme shown in Table 2.

Category	Definition (Influence to the market)
Market Down	Market value decreased by more than 1% on the day when the news is published compared with last closing value.
Market Steady	Market value varies less than 1% on the day when the news is published compared with last closing value.
Market Up	Market value increased by more than 1% on the day when the news is published compared with last closing value.

Table 2: Categorization Scheme

The reason why ‘1%’ is adopted is because this threshold can divide the trading days that we studied into three categories roughly of the same size. Table 3 shows the different thresholds and their effects on the distributions of the trading days in the three categories.

Threshold	Days of market up	Days of market steady	Days of market down
0.5%	38.92%	20.20%	40.89%
1%	29.56%	36.95%	33.50%
2%	19.70%	61.08%	19.21%

Table 3: Effects of Different Thresholds on Category Distribution

4.1.2 Market Data

In our experiments, the market data what we are studying is **Dow Jones Industrial Average**. That means the articles will be classified according to the daily trends of this index value. To calculate the daily trend, we divided the daily change of closing values of this day and the previous day by the closing value of the previous trading day, and then map the trend to a category according to the table in the above section.

4.1.3 Stop Word List

Before processing each article, we will filter out all words that does not contain meanings, such as, this, that, the etc. Those words will not be extracted as keyword or used to calculate term frequency and document frequency when weighting keywords.

The stop word technique is adopted by many search engines, and the stop word lists used by different search engines do not vary much. In our system, we used the one accessible by link (http://www.webconfs.com/stop-words.php). The word list can be found in Appendix A.

4.2 Configurable Parameters

To test and compare the performances of the algorithms, we need to select the algorithms and tune the parameters. In this section, we have listed all the configurations that we tuned and their possible effects towards the testing results.

4.2.1 Training Data Set

In our system, the user can select any subset of the articles that the system has downloaded for training data. To get balanced results, we usually select similar number of articles from each category.

As the articles are classified according to the real market data of the day when they are published, the articles published on the same day always belong to the same category, so a convenient way to select the training data is to select all articles published on same number of days for each category as training data.

Small training data set may not be able to well reflect the nature of relationships of the articles and the market. However, the performance may become worse again as the training data set is going too large. So it is reasonable to tune the size of the training data set until we get the optimized result.

4.2.2 Keyword Extraction Algorithms

We have compared four supervised keyword extraction algorithms (Information Gain, Gain Ratio, Chi-square statistics, and Mutual Information) and one unsupervised keyword extraction algorithms (Entropy). Before generating a keyword list, we can select one of these five algorithms.

Except for choosing the keyword extraction algorithms, we can also configure two parameters when extracting the keyword lists. One is the document frequency threshold, and the other is the keyword list size. These two parameters will also influence the result. We have done a comparison of different parameter settings and generated charts to present the effect to the result of the parameters.

4.2.3 Keyword Weighting Algorithms

In this system, we can select five keyword weighting algorithms. Ordered from the simplest to the most complicated, they are Boolean Weighting, Term Frequency (TF), Term Frequency times Inverse Document Frequency (TFxIDF), TFC Weighting, and LTC Weighting.

If we use Naïve Bayes Classifiers, there are theoretically no difference between using term frequency weighting and TFxIDF weighting, as for an element (keyword) of all vectors (articles), their values will be divided by the same denominator (the logarithm of the document frequency of this keyword) when converting TF to TFxIDF, and in Naïve Bayes Classification, all elements are independent, so such change of weighting method will not affect the classification results.

4.2.4 Classification Algorithms

We have implemented Naïve Bayes Classifier and Support Vector Machines. Our experiment is focused on Naïve Bayes Classifier.

4.3 Testing Procedures

4.3.1 Accumulating Articles

The news articles are periodically downloaded from the major financial news websites every hour. The three sources we have chosen are Bloomberg, CNN Business News and BBC Business News. But by observation, we found that Bloomberg is the one which updates most frequently and give us the largest number of pieces of news.

We had the system started regularly accumulated articles since 10 Feb 2009. We typically get around 200 articles per weekday and around 70 articles per day on weekends. However, the system has experienced three outages since it is started. The details are shown in Table 4

Outage	Dates Affected	Reason	Resolution
1	10 Apr 2009 to 21 Apr 2009	links.xml file exceeding the size limit of the PHP engine.	Leverage the Java Servlet to check whether a link has been downloaded or not, so the entries which have been downloaded in links.xml can be removed. This essentially

			reduced the size of links.xml
2	4 May 2009 to 12 May 2009	CNN changed the format of the news web pages. It caused an exception when running the code to extract the article text from the web pages.	Removing CNN from the sources. As CNN did not contribute to the article database too much anyway, this resolution is appropriate.
3	3 Jun 2009 to 10 Jun 2009	Bloomberg changed its news link patterns.	Updated the link pattern in the spider component.

Table 4: System Outages during Article Accumulation

All these three outages resulted in a huge reduction of the number of articles accumulated on the days affected. Fortunately, no outages caused irreversible damages to the articles accumulated previously.

To date, we have accumulated 16719 articles totally, with the published dates ranging from 12 Dec 2008 to 21 Jun 2009.

4.3.2 Selecting Training Data Sets

We have selected three sets of articles for training data, each of them is with different amount of articles. By this way we can study how the size of the training data can influence the testing results. When selecting the training data sets, we used the

strategy mentioned in Section 4.2.1. The details of the three sets are shown in Table 5

Name of the training data set	**Articles included (by date)**	**Size of Training Data Set**
set1	Market Down: 10 Feb 2009, 17 Feb 2009 Market Steady: 11 Feb 2009, 18 Feb 2009 Market Up: 17 Mar 2009, 18 Mar 2009 Total: 6 trading days	1098
set2	Market Down: 10 Feb 2009, 13 Feb 2009, 17 Feb 2009, 5 Mar 2009 Market Steady: 11 Feb 2009, 12 Feb 2009, 18 Feb 2009, 3 Mar 2009 Market Up: 17 Mar 2009, 18 Mar 2009, 25 Mar 2009, 26 Mar 2009 Total: 12 trading days	2306
set3	Market Down: 10 Feb 2009, 13 Feb 2009, 16 Feb 2009, 17 Feb 2009,	4465

	19 Feb 2009, 20 Feb 2009, 23 Feb 2009, 25 Feb 2009 Market Steady: 11 Feb 2009, 12 Feb 2009, 18 Feb 2009, 03 Mar 2009, 06 Mar 2009, 11 Mar 2009, 13 Mar 2009, 16 Mar 2009 Market Up: 24 Feb 2009, 04 Mar 2009, 10 Mar 2009, 12 Mar 2009, 17 Mar 2009, 18 Mar 2009, 25 Mar 2009, 26 Mar 2009 Total: 24 trading days	

Table 5: Training Data Sets

4.3.3 Generate Keyword Lists and Train Classifiers

To train a classifier, we need to generate a keyword list, and select a training data set, a keyword weighting method and a classification algorithm. Each of these except the classification algorithm has 4 to 5 choices, so totally we have more than 100 combinations, which results in more than 100 classifiers (predictors). It will be a huge amount of work to simply train and compare the performance of all those classifiers.

In our experiments, we used another strategy to do the testing and the comparison. First we assume the setting of one parameter or algorithm selection of one step is relatively independent from the others in terms of performance. For example, if Entropy method is the best algorithm for keyword extraction, then it will be the best algorithm for all combinations of parameter settings of the other parameters. By this

assumption, we can vary one or a small set of parameters first, and fix all the others, and then select the optimized setting. Then we fix the setting to the optimized one and tune other parameters. In this way we can get the best combination of algorithm selections and parameter settings that this greedy approach can give.

At the start of the experiment, we use several assumed values, like the keyword list size is 512, and the document frequency threshold is 0.05, etc. This is because we need to fix something to a random but reasonable value first, and stick to one parameter to tune, or there will be too many combinations to test. After tuning one parameter, we can come back to tune the parameters that is previously fixed.

The exact procedure that we adopted is as follows.

1) Select the best training data set and the best combination of keyword extraction and keyword weight methods:

   Set the document frequency threshold to 0.05, and the keyword list size to 512. Train a classifier for each combination of training data set, keyword extraction algorithm, keyword weighting method selection. Observe the results and select the best combination.

2) Observe how the performance varies with the document frequency threshold. Select the best document frequency threshold:

   Set the keyword extraction algorithm to the one that selected in the previous step, and the keyword list size to 300. Generate a new keyword list for each document frequency threshold ranging from 0.01 to 0.1 (adding 0.01 each time). Train a

classifier for each keyword list with other parameters set to values according to the results of previous steps. Observe the results to see how the performance varies, and select the best document frequency threshold

3) Observe how the performance varies with the size of the keyword list. Select the best keyword list size:

   Similar to last step, but set the document frequency threshold to the one selected in the previous step, and vary the sizes of the keyword lists. Observe how the result varies and select the best keyword list size.

4.3.4 Observing Testing Results

Once a classifier is trained, we can classify all non-training articles, which we also call testing articles. We will observe the classification results and compare them with the real market data, and then extract the following measurements from the comparison results for further performance evaluation purpose.

i) "Classification Matrix": For each testing article, we will classify it by the classifier we trained, and get its expected category according to the daily trend of the real market when the article was published. Among the articles with each expected category, we can count the number of articles that were classified into each category. Therefore, we can get a 3x3 matrix as shown in Figure 10. E.g. the element a21 represent the number of articles that were published on a day that the market is going steady, but were classified as market down by our classifiers. This is the most direct measurements that we can get and is the foundation of all

the following derived measurements.

a11	a12	a13
a21	a22	a23
a31	a32	a33

Figure 10: Format of a “Classification Matrix”

We can also classify trading days rather than articles, by calculating the number of articles belonging to each category published on this day, and taking the category with the majority articles as the category for this day. With the real market situation of this day, we can get another classification matrix. We call these two matrices **classifications matrix by articles** and **classifications matrix by date** respectively.

ii) **Rate of correct classifications:** For one article, if its expected category matches the category that this article is classified by the trained classifier, then the classification of this article is considered as a correct classification. Otherwise, it is considered as an incorrect classification. The rate of correct classification is simply defined as the ratio of the number of correct classifications to the number of total classifications. If we define this measurement using the classification matrix that we defined above, this rate will be the trace (summation of all diagonal elements) divided by the summation of all elements of the matrix. If we make use of the symbols shown in Figure 10, the rate of correct classifications

can be represented as Formula 20.

$$RoCC = \frac{a11 + a22 + a33}{a11 + a12 + a13 + a21 + a22 + a23 + a31 + a32 + a33} \quad 20)$$

In the same way as the classification matrix, this measurement can also be defined by date rather by article, which is even more important when doing actual trading.

iii) **Rate of reverse classifications:** Sometimes the classification inaccuracy may be due to the inappropriate threshold settings to distinguish between the three categories of "market up", "market steady", and "market down". However, if a classifier classified an article as “market down”, but the expected category is “market up”, then we can be sure that this problem must be due to the classifier itself rather than the threshold settings. So another measurement should be the rate of classifications that were classified as opposite categories compared with their expected categories. This number can be represented as Formula 21.

$$RoRC = \frac{a13 + a31}{a11 + a13 + a31 + a33} \quad 21)$$

4.3.5 Result Comparisons

For each classifier we trained, we can get a set of the above measurements. For our experiment, we only compare the measurements by article rather than by date, as theoretically, this way is more accurate, as there are quite a different number of articles published on each day, so the measurement by date is much more unstable and

less meaningful. Once the comparison is done, we can select the best parameter settings and algorithm selections according to the procedure described previously in Section 4.3.3.

The results of the experiments are discussed in the next chapter.

CHAPTER 5

TEST RESULTS AND ANALYSIS

In this chapter, we will conduct our experiments according to the steps described in Section 4.3.3. And then we will present the results and do some analyses.

5.1 Rate of Correct Classifications

5.1.1 Step 1 – Select the Best Training Data Set, keyword extraction algorithms and keyword weighting algorithms.

i) Purpose of this step

In this step, we fix the document threshold to 0.05, then we test all combinations for each training data set, keyword extraction algorithm and keyword weighting algorithm. Finally we select the best combinations for the following steps of our experiment.

ii) Testing results

Training data set: set1

	ig_05_512	gr_05_512	cs_05_512	mi_05_512	entropy_05_512
Boolean	0.3337	0.3347	0.3538	0.3091	0.3270
TF	0.3266	0.3268	0.3197	0.3375	0.3292

TFxIDF	0.3266	0.3268	0.3197	0.3375	0.3292
LTC	0.3270	0.3268	0.3226	0.3315	0.3233
TFC	0.3271	0.3268	0.3226	0.3309	0.3228

Table 6: Testing Results for Training Data Set 1, DF=0.05

The result can be represented as Figure 11.

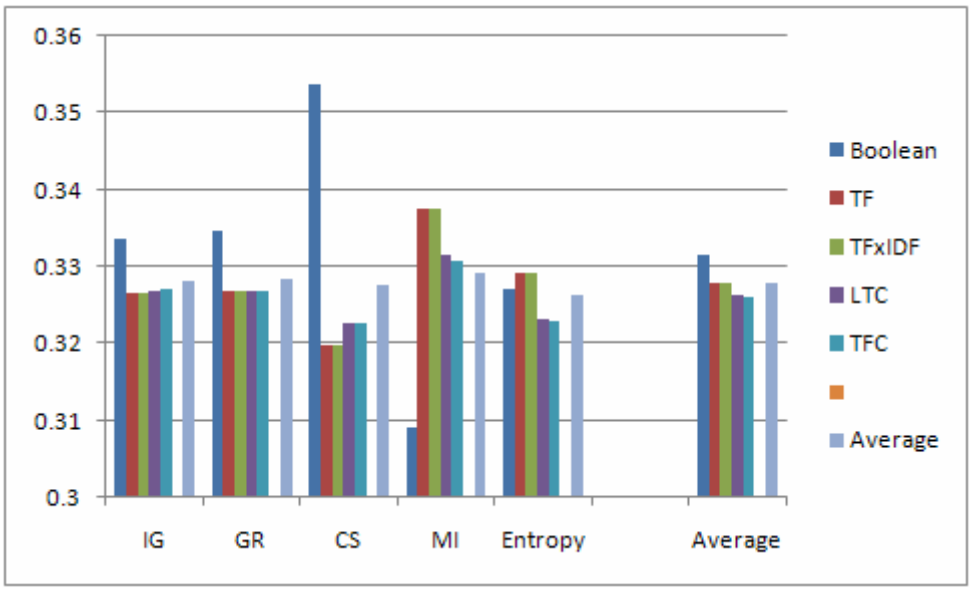

Figure 11: Testing results for DF=0.05, training set 1

Training data set: set2

	Ig_05_512	gr_05_512	cs_05_512	mi_05_512	entropy_05_512
Boolean	0.3181	0.3199	0.3340	0.3047	0.3203
TF	0.3286	0.3270	0.3198	0.3268	0.3206

TFxIDF	0.3286	0.3270	0.3198	0.3268	0.3206
LTC	0.3205	0.3214	0.3294	0.3190	0.3172
TFC	0.3202	0.3209	0.3288	0.3198	0.3169

Table 7: Testing Results for Training Data Set 2, DF=0.05

The result can be represented as Figure 12.

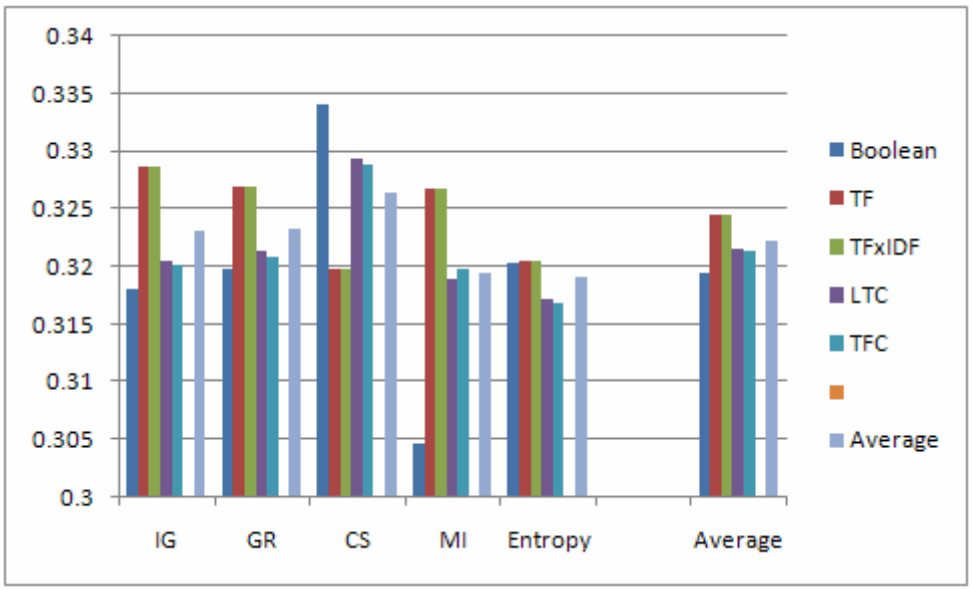

Figure 12: Testing results for DF=0.05, training set 2

Training data set: set3:

	Ig_05_512	gr_05_512	cs_05_512	mi_05_512	entropy_05_512
Boolean	0.3459	0.3457	0.3694	0.2944	0.3119
TF	0.3394	0.3387	0.3364	0.3382	0.3400

TFxIDF	0.3394	0.3387	0.3364	0.3382	0.3400
LTC	0.3304	0.3306	0.3295	0.3227	0.3334
TFC	0.3306	0.3306	0.3299	0.3226	0.3335

Table 8: Testing Results for Training Data Set 3, DF=0.05

The result can be represented as Figure 13.

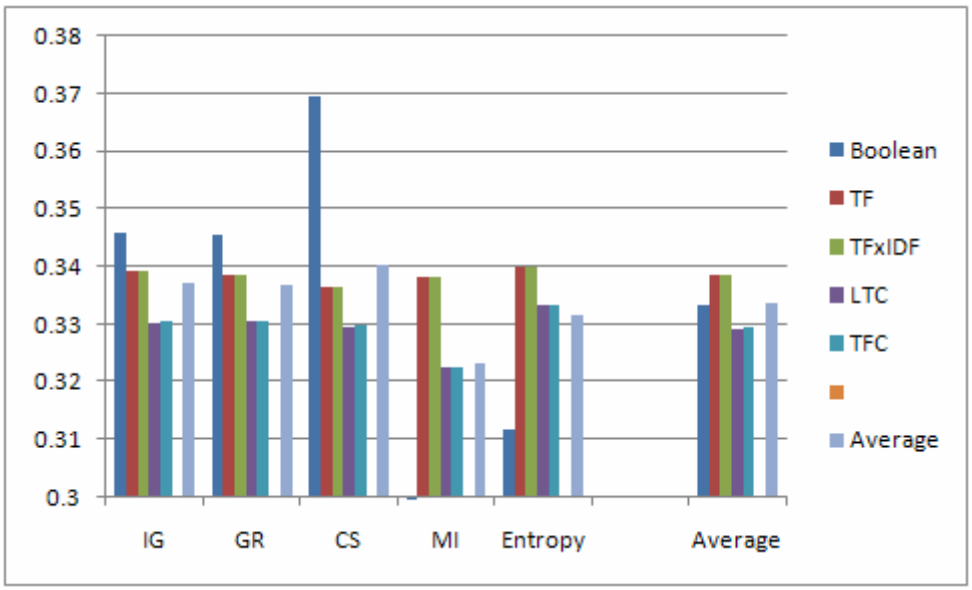

Figure 13: Testing results for DF=0.05, training set 3

iii) Result analysis

Figure 14 and Figure 15 are plotted to facilitate us to compare the performance for each training data set, keyword extraction method and keyword weighting method. In Figure 14, each group of columns represents a keyword extraction algorithm and a column represents the average performance for all keyword weighting methods of a training data set. The final group represents the average performance for all keyword extraction algorithms for a training data set. In Figure 15, the concept is similar except

the columns are grouped by keyword weighting algorithms rather than keyword extraction algorithms.

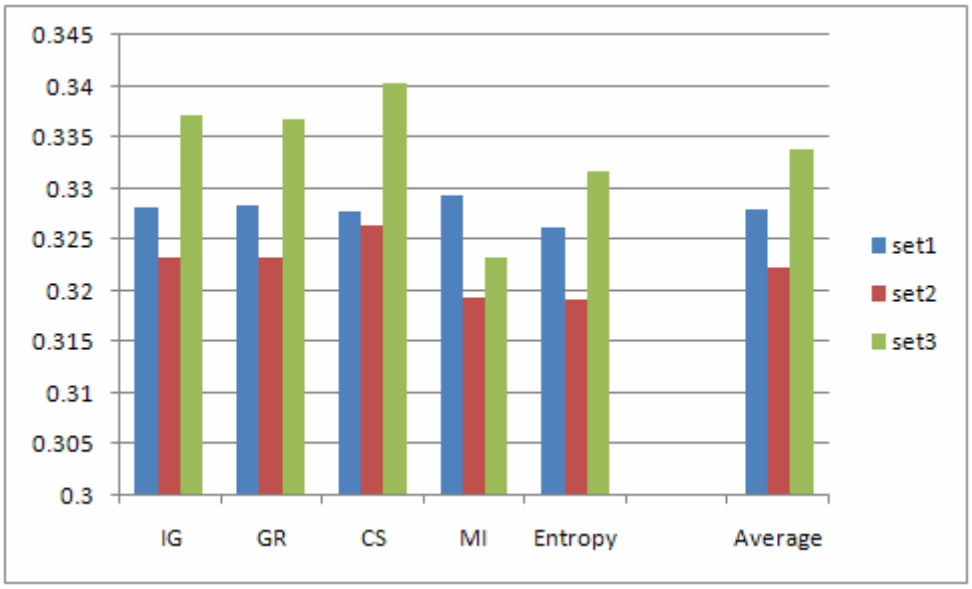

Figure 14: Performance against training data sets
(grouped by keyword extraction methods)

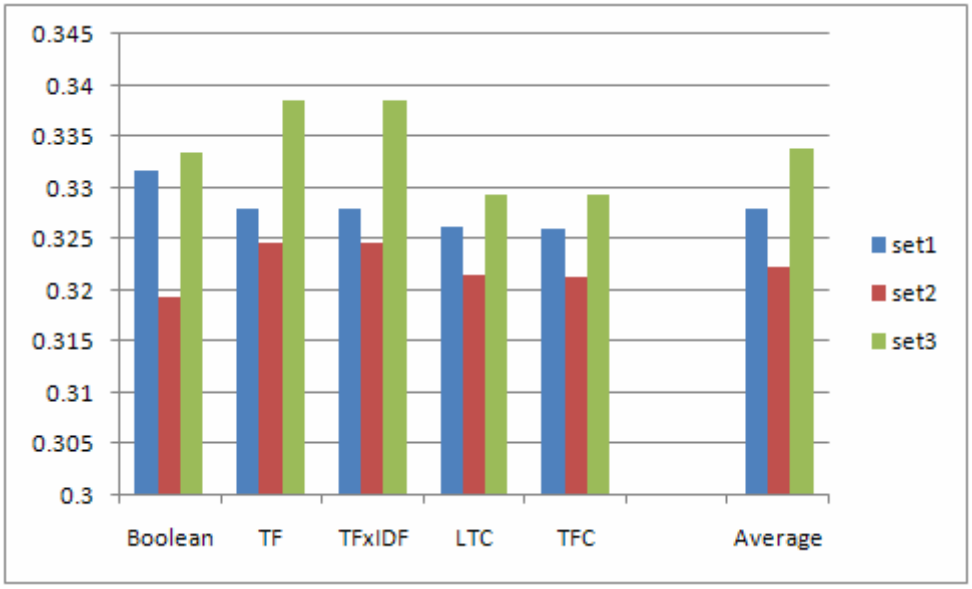

Figure 15: Performance against training data sets
(grouped by keyword weighting algorithms)

By observing Figure 14 and Figure 15, we can easily see that Training Data Set

named “set3” always has the best performance. We will use this training data set to conduct the second step of our experiment.

By observing Figure 14, we can find Chi-square Statistics are the best algorithm for keyword extraction method.

By observing Figure 15, we can find out even sometimes Boolean weighting can give us much better performance, but it is very unstable. So TF and TFxIDF are the best algorithms for keyword weighting.

5.1.2 Step 2 – Observe how the performance varies with the document frequency threshold. Select the best document frequency threshold

i) Purpose of this step

As we have already selected the best training data set and the combinations for keyword extraction and keyword weighting methods, we fix these two algorithms and tune the document frequency threshold. And see how the performance varies with this parameter.

ii) Testing results

Document Frequency Threshold	Rate of Correct Classifications
0.01	0.3322
0.02	0.3273
0.03	0.3398
0.04	0.3439
0.05	0.3503
0.06	0.3478
0.07	0.3416

0.08	0.3390
0.09	0.3383
0.1	0.3375

Table 9: Performance against document frequency threshold

The result can be represented as Figure 16.

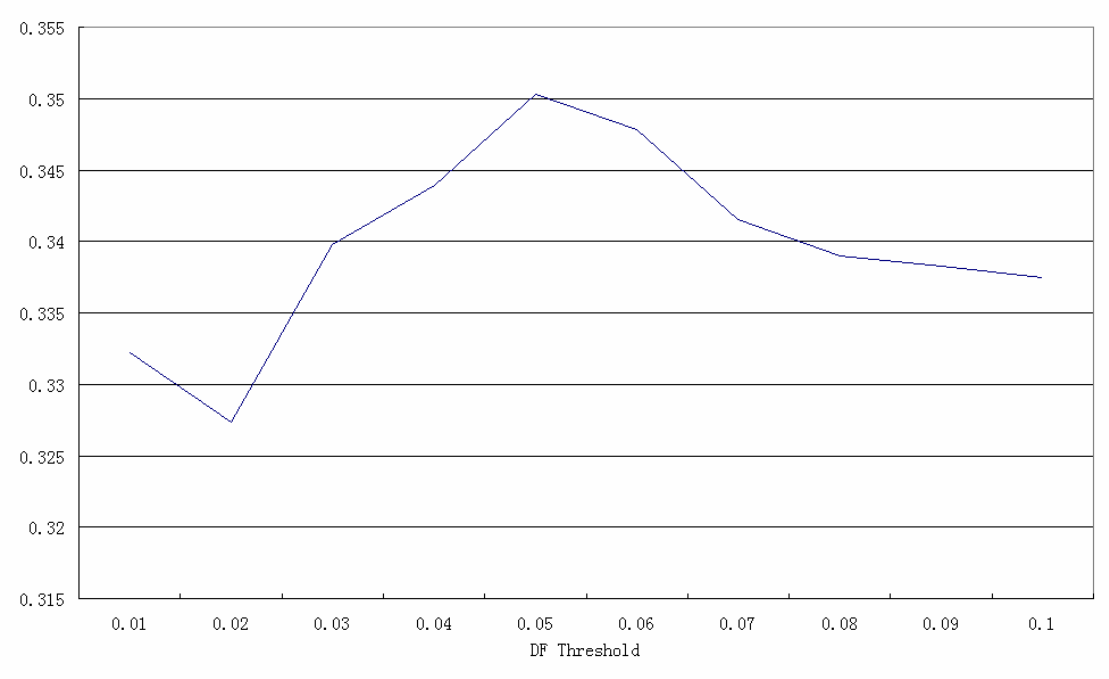

Figure 16: Performance against document frequency threshold

iii) Result analysis

According to the figure, we can easily find out that the optimal value for document frequency threshold is 0.05.

5.1.3 Step 3 – Observe how the performance varies with the size of the keyword list. Select the best keyword list size

i) Purpose of this step.

In this step, we will set the document frequency threshold to the above optimal

value, 0.05. Then we will tune the keyword list size, to see how the performance varies with this parameter.

ii) Testing results

Size of the Keyword List	Rate of Correct Classifications
300	0.35029
350	0.35061
400	0.34318
450	0.34424
500	0.34351
550	0.34318
600	0.33829
650	0.33918
700	0.33992

Table 10: Performance against size of keyword list

The result can be represented as Figure 17.

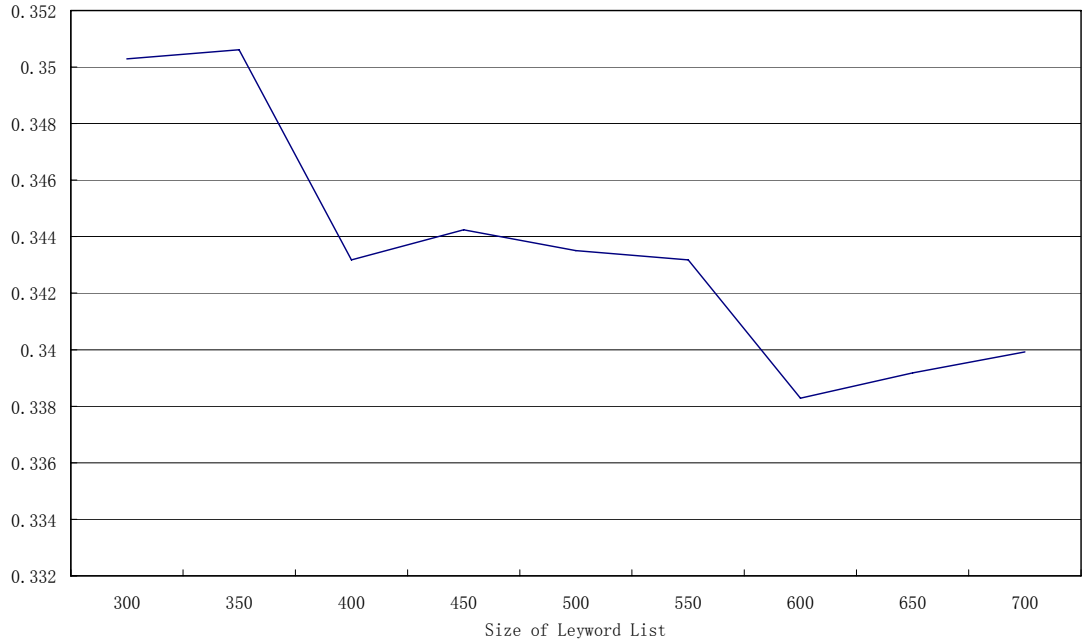

Figure 17: Performance against size of keyword list

iii) Result analysis

By observing the results, we can find out that the performance is generally getting worse as the size of the keyword lists is increasing.

5.2 Rate of Reverse Classifications

i) Introduction

We make use of the result from the last section to set the parameters for the experiment of this section. We set the document threshold to 0.02, training data set to "set3", and test all combinations for each keyword extraction algorithm and keyword weighting algorithm.

ii) Testing results

	ig_02_512	gr_02_512	cs_02_512	mi_02_512	entropy_02_512
boolean	0.5289	0.5250	0.4352	0.5606	0.5006
Tf	0.4886	0.4891	0.4689	0.4913	0.4867
tfxidf	0.4886	0.4891	0.4689	0.4913	0.4867
ltc	0.4886	0.5092	0.4756	0.5077	0.5154
tfc	0.5061	0.5080	0.4762	0.5067	0.5143

Table 11: Testing Results for Rate of Reverse Classifications

iii) Result analysis

Figure 18 presents the rate of reverse classifications for different keyword extraction and keyword weighting algorithms. Different from other figures in this chapter, a large number represents a worse performance as we always want to minimize reverse classifications.

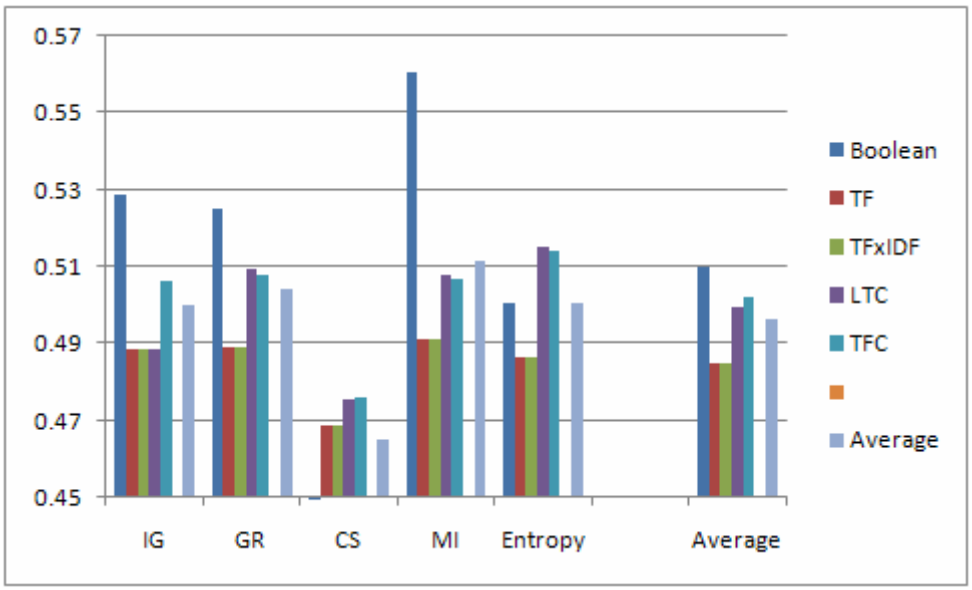

Figure 18: Rate of reverse classifications for all keyword weighting and keyword extraction algorithms

CHAPTER 6

CONCLUSION AND FUTURE WORK

By observing the testing results presented in last chapter, we can get the following conclusions. We will also try to explain the reasons when presenting our discovery.

6.1 Training Data Size

By observing the result, Training Data Set named “set3” (with the largest training data size) has the best performance. This means that most of the time increasing the training data size can better off the performance of the classifiers.

6.2 Keyword List

6.2.1 Similarity of Information Gain and Gain Ratio

No matter by observing the testing result or by directly observing the keyword list generated, we can all conclude the keyword list generated by Information Gain or Gain Ratio are very similar.

6.2.2 Document Frequency Threshold

We found that the optimal value for the document frequency threshold is 0.05. Low values of document frequency threshold may cause a lot of articles having a feature vector with most of the elements equaling to 0. High values will result that keywords are selected mostly according to the document frequency thresholds, but the original evaluation function will become less important.

6.2.3 Size of Keyword Lists

We found that the performance is generally getting worse as the size of keyword list is decreasing. This shows that the most distinguishable words are much more important in classification than those less distinguishable ones. But generally speaking, the performance does not vary so obviously with this parameter comparing with other parameters.

6.2.4 Best Keyword Extraction Method

By comparing, the best keyword extraction algorithm is Chi-square Statistics.

6.3 Keyword Weighting Algorithms

6.3.1 Boolean Weighting

We found that the performance of Boolean Weighting method is unstable, so it is not a good weighting method for financial market prediction problem.

6.3.2 Term Frequency and TFxIDF weighting

When we are using Naïve Bayes Classifiers, the result for Term Frequency Weighting and TFxIDF Weighting are exactly the same if the other parameters are the same. This verifies the inference that we proposed in Section 4.2.3

6.3.3 Best Keyword Weighting Algorithm

By observing the values, we found that TFxIDF/TF has the best performance among all weighting methods.

6.4 Future Work

This section we suggest some works that this thesis can be extended and some future research directions.

6.4.1 Values to Predict

In this thesis, we studied the index Dow Jones Industrial Average only. However, the financial news articles are influencing all kinds of market values like other indices and stock prices. We may observe more index values in the future to study the different influences that financial news articles had to the different market values.

Except predicting the index values, we can also predict other properties of this financial product such as volume, volatility, etc.

6.4.2 Categorization Schemes

In this thesis, we have a single categorization scheme. In the future we can vary it by either adding or deleting the categories, e.g. two categories, four categories, or adjusting the margins to distinguish each category.

6.4.3 Relation Analysis

As it is hard to judge whether a news article is related to the market value that we are studying or not, we simply assume all news articles published in a financial news web sites are related to the index that we are studied. In the future, we can apply some methods so that the articles so that unrelated articles will be removed. This will definitely improve the performance of our system.

6.4.4 Word Stemming

In our thesis we are only doing a stop words removal before doing text preprocessing, but not stemming. This means that, currently, “word” and “words” will be considered as different words. However, there should be little or no difference in influence on the market of these two words. Therefore, stemming will also enhance the performance of the system.

6.4.5 Stop Word List Customization

As the most of the stop word lists are adopted by search engines who are mainly doing text mining for general indexing purpose, it is very possible that we can get better results if we can customize the stop word list to make it more suitable to do financial related text mining.

6.4.6 Simulation Trading

As this system is ultimately for trading purpose, we should test it by doing simulation trading using our system and consider the return as an important measurement to test our system.

BIBLIOGRAPHY

[1] Marc-André Mittermayer and Gerhard F. Knolmayer, “Text Mining Systems for Market Response to News: A Survey”, Institute of Information Systems University of Bern.

http://www.ie.iwi.unibe.ch/publikationen/berichte/resource/WP-184.pdf

[2] H. Braun, “Predicting stock market behavior through rule induction: an application of the learning-from example approach”, Decision Sciences, vol. 18, no. 3.

[3] R. Gencay, “Non-linear prediction of security returns with moving average rules”, Journal of Forecasting, vol. 15,110.3, pp. 165-174, 1996.

[4] Steven H. Kim and Se Hak Chun, “Graded forecasting using an array of bipolar predictions: application of probabilistic neural networks to a stock market index”, In: International Journal of Forecasting 14 (1998) 323–337.

[5] Y.-F. Wang, “Predicting stock price using fuzzy grey prediction system”, In: Expert Systems with Applications 22 (2002) 33-39.

[6] Indranil Bose and radha K. Mahapatra, “Business data mining - a machine learning perspective”, In: Information & Management 39 (2001) 211-225.

[7] Dongsong Zhang and Lina Zhou, “Discovering Golden Nuggets: Data Mining in Financial Application”, In: IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews, vol. 34, No. 4, November

2004.

[8] Wesley S. Chan, “Stock price reaction to news and no-news: drift and reversal after headlines”, In: Journal of Financial Economics 70 (2003) 223–260.

[9] Fabrizio Sebastiani, “Machine Learning in Automated Text Categorization”, Consiglio Nazionale delle Ricerche, Italy.

[10] Richard Xiao and Tony McEnery, “Collocation, Semantic Prosody, and Near Synonymy: A Cross-Linguistic Perspective”, In: Applied Linguistics 27/1: 103–129.

[11] V. Cho, B. Wüthrich and J. Zhang, “Text Processing for Classification”, In: Journal of Computational Intelligence in Finance 7 (1999) 2, pp. 6-22.

[12] V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen and J. Allan, “Mining of Concurrent Text and Time Series”, In: Proceedings 6th ACM SIGKDD Int. Conference on Knowledge Discovery and Data Mining, Boston 2000, pp. 37-44.

[13] Y. Seo, J.A. Giampapa and K. Sycara, “Financial News Analysis for Intelligent Portfolio Management”, Technical Report CMU-RI-TR-04-04, Robotics Institute, Carnegie Mellon University, Pittsburgh.

http://www.ri.cmu.edu/pub_files/pub4/seo_young_woo_2004_2/seo_young_woo_2004_2.pdf

[14] G. Gidófalvi, “Using News Articles to Predict Stock Price Movements”, Project Report, Department of Computer Science and Engineering, University of

California, San Diego.

http://www-cse.ucsd.edu/users/elkan/254spring01/gidofalvirep.pdf

[15] D. Peramunetilleke and R. K. Wong, “Currency Exchange Rate Forecasting from News Headlines”, In: Proceedings 13th Australasian Database Conference, Melbourne 2002, pp.131-139.

[16] G. P. C. Fung, J. X. Yu, and W. Lam, “Stock Prediction: Integrating Text Mining Approach Using Real-time News”, In: Proceedings IEEE Int. Conference on Computational Intelligence for Financial Engineering, Hong Kong 2003, pp. 395-402.

[17] A. Schulz, M. Spiliopoulou and K. Winkler, “Kursrelevanzprognose von Ad-hoc-Meldungen: Text Mining wider die Informationsüberlastung im Mobile Banking”, In: W. Uhr, W. Esswein and E. Schoop(eds.): Wirtschaftsinformatik 2003, Physica, Heidelberg 2003, pp. 181-200.

[18] M.-A. Mittermayer, “Forecasting Intraday Stock Price Trends with Text Mining Techniques”, In: Proceedings 37th Annual Hawaii Int, Conference on System Sciences (HICSS), Big Island 2004, p. 64.

[19] Lean Yu, Shouyang Wang and K. K. Lai, “A Rough-Set-Refined Text Mining Approach for Crude Oil Market Tendency Forecasting”, In: International Journal of Knowledge and Systems Sciences Vol. 2, No. 1, March 2005.

[20] Xin Lua, Zhao Yang Dong and Xue Li, “Electricity market price spike forecast

with data mining techniques", In: Electric Power Systems Research 73 (2005) 19–29.

[21] Vincent Cho and Beat Wüthrich, "Combining Forecasts from Multiple Textual Data Sources", The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, Springer Berlin, 1999, pp.174-179

[22] Pegah Falinouss, "Stock Trend Prediction Using News Articles - A Text Mining Approach", Master's thesis of Industriell ekonomi och samhällsvetenskap, 2007.

[23] Robert P. Schumaker and Hsinchun Chen, "Textual Analysis of Stock Market Prediction Using Financial News Articles", Artificial Intelligence Lab, Department of Management Information Systems The University of Arizona, Tucson, Arizona 85721, USA.

[24] Michael W. Berry, "Survey of Text Mining – Clustering, Classification, and Retrieval", Springer, 2004.

[25] Spiros Sirmakessis, "Text Mining and its Applications", Springer, 2003

[26] Ronen Feldman and James Sanger, "The Text Mining Handbook – Advanced Approaches in Analyzing Unstructured Data", Cambridge, 2007

[27] Andreas Hotho., Andreas Nurnberger, Gerhard Paaß and Fraunhofer AiS, "A Brief Survey of Text Mining", May 13, 2005.

[28] Yiming Yang and Jan O. Pedersen, "A Comparative Study on Feature Selection", Machine Learning, International Workshop and Conference, 1997.

[29] Kjersti Aas and Line Eikvil, “Text Categorisation: A Survey”, June 1999.

[30] Franca Debole and Fabrizio Sebastiani, “Supervised Term Weighting for Automated Text Categorization”,

[31] G. Salton and M. J. McGill, “An Introduction to Modern Information Retrieval”, McGraw-Hill. 1983.

[32] M. W. Berry, S. T. Dumais and G. W. O’Brien, “Using linear algebra for intelligent information retrieval”, SIAM Review, Vol. 37, pp. 573-595, December 1995.

[33] S. Deerwester, S. Dumais, G. Furnas., T. Landauer and R. Harshman, “Indexing by latent Semantic Analysis”, Journal of the American Society for Information Science, Vol. 41, No. 6, pp.391-407. 1990.

[34] R. Fano, “Transmission of Information”, MIT Press, Cambridge, MA, 1961.

[35] Kenneth Ward Church and Patrick Hanks, “Word association norms, mutual information and lexicography”, In Proceedings of ACL 27, pages 76-83, Vancouver, Canada, 1989.

[36] E. Wiener, J. O. Pedersen, A. S. Weigend, “A neural network approach to topic spotting”, In Proceedings of the fourth annual symposium on document analysis and information retrieval (SDAIR ‘95), 1995.

[37] J. W. Wilbur, K. Sirotkin, “The automatic identification of stop words”, Journal of Information Science 18:45-55, 1992.

[38] Y. Yang, “Noise reduction in a statistical approach to text categorization”, In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR ‘95), pages 256-263. 1995.

[39] Y. Yang and W. J. Wilbur, “Using corpus statistics to remove redundant words in text categorization”, In Journal of American Society for Information Science, 1996.

[40] G. Salton and C. Buckley, “Term weighting approaches in automatic text retrieval”, Information Processing and Management, Vol.24, No. 5, pp.513-523, 1988.

[41] C. Buckley, G.Salton, J. Allan and A. Singhal, “Automatic Query Expansion Using SMARTL TREC 3”, In Proceeding of the 3rd Text Retrieval Conference, NIST, 1994.

[42] S. T. Dunais, “Improving the retrieval information from external sources”, Behavior Research Methods, Instruments and Computers, Vol. 23, No. 2, pp. 229-236, 1991.

Appendix A

Stop Word List

Appendix A is the stop word list we are using for our system. It is quoted from http://www.webconfs.com/stop-words.php.

	did	inside	others	thereupon
a	didn	insofar	otherwise	these
able	different	instead	ought	they
about	directly	into	oughtn	thing
above	do	inward	our	things
abroad	does	is	ours	think
according	doesn	isn	ourselves	third
accordingly	doing	it	out	thirty
across	don	its	outside	this
actually	done	itself	over	thorough
adj	down	j	overall	thoroughly
after	downwards	just	own	those
afterwards	during	k	p	though
again	e	keep	particular	three
against	each	keeps	particularly	through
ago	edu	kept	past	throughout
ahead	eg	know	per	thru
ain	eight	known	perhaps	thus
all	eighty	knows	placed	till
allow	either	l	please	to
allows	else	last	plus	together
almost	elsewhere	lately	possible	too
alone	end	later	presumably	took
along	ending	latter	probably	toward
alongside	enough	latterly	provided	towards
already	entirely	least	provides	tried
also	especially	less	q	tries
although	et	lest	que	truly
always	etc	let	quite	try
am	even	like	qv	trying
amid	ever	liked	r	twice
amidst	evermore	likely	rather	two
among	every	likewise	rd	u
amongst	everybody	little	re	un

an	everyone	ll	really	under
and	everything	look	reasonably	underneath
another	everywhere	looking	recent	undoing
any	ex	looks	recently	unfortunately
anybody	exactly	low	regarding	unless
anyhow	example	lower	regardless	unlike
anyone	except	ltd	regards	unlikely
anything	f	m	relatively	until
anyway	fairly	made	respectively	unto
anyways	far	mainly	right	up
anywhere	farther	make	round	upon
apart	few	makes	s	upwards
appear	fewer	many	said	us
appreciate	fifth	may	same	use
appropriate	first	maybe	saw	used
are	five	mayn	say	useful
aren	followed	me	saying	uses
around	following	mean	says	using
as	follows	meantime	second	usually
aside	for	meanwhile	secondly	v
ask	forever	merely	see	value
asking	former	might	seeing	various
associated	formerly	mightn	seem	ve
at	forth	mine	seemed	versus
available	forward	minus	seeming	very
away	found	miss	seems	via
awfully	four	mon	seen	viz
b	from	more	self	vs
back	further	moreover	selves	w
backward	furthermore	most	sensible	want
backwards	g	mostly	sent	wants
be	get	mr	serious	was
became	gets	mrs	seriously	wasn
because	getting	much	seven	way
become	given	must	several	we
becomes	gives	mustn	shall	welcome
becoming	go	my	shan	well
been	goes	myself	she	went
before	going	n	should	were
beforehand	gone	name	shouldn	weren
begin	got	namely	since	what
behind	gotten	nd	six	whatever

being	greetings	near	so	when
believe	h	nearly	some	whence
below	had	necessary	somebody	whenever
beside	hadn	need	someday	where
besides	half	needn	somehow	whereafter
best	happens	needs	someone	whereas
better	hardly	neither	something	whereby
between	has	never	sometime	wherein
beyond	hasn	neverf	sometimes	whereupon
both	have	neverless	somewhat	wherever
brief	haven	nevertheless	somewhere	whether
but	having	new	soon	which
by	he	next	sorry	whichever
c	hello	nine	specified	while
came	help	ninety	specify	whilst
can	hence	no	specifying	whither
cannot	her	nobody	still	who
cant	here	non	sub	whoever
caption	hereafter	none	such	whole
cause	hereby	nonetheless	sup	whom
causes	herein	noone	sure	whomever
certain	hereupon	nor	t	whose
certainly	hers	normally	take	why
changes	herself	not	taken	will
clearly	hi	nothing	taking	willing
co	him	notwithstanding	tell	wish
com	himself	novel	tends	with
come	his	now	th	within
comes	hither	nowhere	than	without
concerning	hopefully	o	thank	won
consequently	how	obviously	thanks	wonder
consider	howbeit	of	thanx	would
considering	however	off	that	wouldn
contain	hundred	often	thats	x
containing	i	oh	the	y
contains	ie	ok	their	yes
corresponding	if	okay	theirs	yet
could	ignored	old	them	you
couldn	immediate	on	themselves	your
course	in	once	then	yours
currently	inasmuch	one	thence	yourself
d	inc	ones	there	yourselves

dare	indeed	only	thereafter	z
daren	indicate	onto	thereby	zero
definitely	indicated	opposite	therefore	
described	indicates	or	therein	
despite	inner	other	theres	