Yahoo! for Amazon: Opinion Extraction from Small Talk on the Web*

Sanjiv R. Das
Department of Finance
Santa Clara University
Email: srdas@scu.edu

Mike Y. Chen
Computer Science Division
University of California, Berkeley
Email: mikechen@cs.berkeley.edu

August 5, 2001

*Comments welcome. Thanks to Robert Wilensky and Sridhar Rajagopalan for helpful comments and guidance. Many thanks to Priya Raghubir for reclassifying the training set used in this study, and for many useful suggestions. Thanks to Raman Uppal for many helpful comments. We are also grateful to Vikas Agarwal, Chris Brooks, Yuk-Shee Chan, David Gibson, David Levine, Asis Martinez-Jerez, Ajit Ranade, Mark Rubinstein, Peter Tufano, Shiv Vaithyanathan and seminar participants at Northwestern University, UC Berkeley-EECS, London Business School, Multinational Finance Conference, Italy and the Asia Pacific Finance Association Meetings, Bangkok for helpful discussions and insights. Danny Tom and Jason Waddle were instrumental in delivering insights into this paper through joint work on alternative techniques via support vector machines. We owe a special debt to the creative environments at UC Berkeley's CS Division and Haas School, where this work was begun. The first author gratefully acknowledges support from the Price Waterhouse Cooper's Risk Institute, the Dean Witter Foundation, and a Research Grant from Santa Clara University. Please address all correspondence to Professor Sanjiv Das, Dean Witter Fellow & Associate Professor, Santa Clara University, Leavey School of Business, Dept of Finance, 208 Kenna Hall, Santa Clara, CA 95053-0388. Email: srdas@scu.edu.

Yahoo! for Amazon: Opinion Extraction from Small Talk on the Web

Abstract

The internet has made it feasible to tap a continuous stream of public sentiment from the world wide web, quite literally permitting one to "feel the pulse" of any issue under consideration. We present a methodology for real time sentiment extraction in the domain of finance. With the advent of the web, there has been a sharp increase in the influence of individuals on the stock market via web-based trading and the posting of sentiment to stock message boards. While it is important to capture this "sentiment" of small investors, as yet, no index of sentiment has been compiled. This paper comprises (a) a technology for extracting small investor sentiment from web sources to create an index, and (b) illustrative applications of the methodology. We make use of computerized natural language and statistical algorithms for the automated classification of messages posted on the web. We design a suite of classification algorithms, each of different theoretical content, with a view to characterizing the sentiment of any single posting to a message board. The use of multiple methods allows imposition of voting rules in the classification process. It also enables elimination of "fuzzy" messages which are better off uninterpreted. A majority rule across algorithms vastly improves classification accuracy, but also leads to a natural increase in the number of messages classified as "fuzzy". The classifier achieves an accuracy of 62% (versus a random classification accuracy of 33%), and compares favorably against human agreement on message classification, which was 72%. The technology is computationally efficient, allowing the access and interpretations of thousands of messages within minutes. Our illustrative applications show evidence of a strong link between market movements and sentiment. Based on approximately 25,000 messages for the last quarter of 2000, we found evidence that sentiment is based on stock movements.

Contents

1 Introduction

> "Language is itself the collective art of expression, a summary of thousands upon thousands of individual intuitions. The individual gets lost in the collective creation, but his personal expression has left some trace in a certain give and flexibility that are inherent in all collective works of the human spirit" – Edward Sapir, cited in "Society of Mind" by Marvin Minsky.

The "new economy" has resulted in greater participation in the markets by small investors. The sentiment of these investors comprises an influencing force on market sentiment. There are also noticeable effects on trading volume and the price volatility of stocks. The web offers a new approach to accessing and analyzing public sentiment on the stock markets. Large institutions have always expressed their sentiment on stocks via published analyst forecasts. Now, the advent of stock chat boards enables small investors to express their sentiments too, frequently and forcefully.

The internet has fostered a new age in which it is easy to communicate sentiment. It has resulted in major changes in the way sentiments are developed, modified, diffused and expressed. While financial markets are just one case in point, the web has been used as a medium for polling in fields such as voting behavior, consumer purchases, political views and others (see Godes and Mayzlin [2001], Urban [2001], Lam and Myers [2001] for examples). No matter what the realm, accessing, summarizing and aggregating the vast amount of sentiment posted to web sites is a gigantic task. In this article, we present some new approaches to interpreting and analyzing the sentiment expressed in stock market web forums. The initial results presented in this paper appear promising for the technology.[1]

[1] We suggest that there are two standards for success here. (a) The ability of the technology to access and rapidly process sentiment (the sentiment extraction objective). (b) The value this sentiment has in terms of the ability to predict stock returns (the profitability objective), and its prognosis for the efficiency of financial markets. This paper is primarily concerned with the former objective. However, we do investigate the latter too, and find in favor of market efficiency.

It is said that differences of opinion make a market. Estimates suggest that 15% of NASDAQ volume now comes from day trading by retail players. The ratio of small investors to institutional traders has undergone substantial change over the past few years. Choi, Laibson and Metrick [2000] analyze the impact of a web-based trading channel on the trading activity in corporate 401(k) plans, and find that the "web effect" is very large: trading frequency doubles, and portfolio turnover rises by over 50 percent, when investors are permitted to use the web as the information and transaction channel. In many ways, stock market sentiment, formerly the domain of experienced Wall Street analysts, is now being generated by small investors, using web chat rooms and message boards as media.

In contrast to older approaches such as investor questionnaires, sentiment extraction from web postings is new. It constitutes a *real time* approach to sentiment polling, as opposed to traditional *point-in-time* methods. It may be more efficient, and timely as well. Yet, it may suffer from biases less likely with more controlled polling experiments, and allow feedback information loops that may contaminate information from the poll.

With so many investors turning to the internet as the medium of access to the stock market, the volume of information flow on the web has accelerated. Stock market message boards have become an extremely active forum for investors to debate the fortunes of a stock. The medium is catching on, particularly as it offers one of the lowest cost sources of information. For example, in the case of Amazon Inc., there were cumulatively 70,000 messages by the end of 1998 on Yahoo's message board, and this had grown to about 320,000 messages by the end of 2000. There are almost 8000 stocks for which message board activity exists, and about 5-10 major message board providers. Without doubt, the amount of discussion is staggering. The message flow comprises, at the same time, valuable insights, market sentiment, manipulative behavior, and reactions to other sources of news. Message boards have attracted the attention of investors, corporate management, and of course, regulators.[2]

[2]Das, Martinez-Jerez and Tufano [2000] present an empirical picture of the regularities found in messages posted to stock boards.

The recent case of Emulex Corp highlights the sensitivity of the web as an sentiment channel. Emulex's stock declined 62% when an anonymous, false news item on the web claimed reduced earnings and the resignation of the CEO. The Securities Exchange Commission (SEC) promptly apprehended the perpetrator, and the CEO of Emulex issued the following statement, a testimony to both, the magnitude of the impact of electronic news forums on stock values, and the commitment of the SEC to keeping this sentiment channel free and fair:

> "Since the events of last Friday [August 25, 2000], we have noted increasingly widespread concern over the vulnerability of the financial markets due to fraudulent acts of this nature. Clearly, the huge impact in our stock that resulted from this incident has demonstrated the high regard and trust that the public has for financial news services. We believe that the public's trust is well founded and the stability of financial markets worldwide remains critically dependent on the continued role in disseminating accurate information. While there will always be challenges in completely safeguarding the integrity of electronic information, in the aftermath of this incident, we have been gratified by the promise of increased vigilance and scrutiny by financial news services worldwide."

We are in the early stages of understanding the impact this new(s) channel has on trading behavior, information flow, regulation, market volatility, institutional design, and market efficiency. The web makes available the largest data set of public sentiment that has ever existed. This data is noisy, unformatted, complex, and is maximally heterogeneous, being generated by a potentially infinite number of sources! It is humanly impossible to process all this information. Our goal is to develop and test a methodology for accessing and understanding this new sentiment channel. This paper presents algorithms to process and analyze this data, and builds a "sentiment" index that aggregates the sentiments of many small investors.

We use statistical and natural language processing (NLP) techniques to to elicit emotive sentiment from a posted message, i.e. determine whether the sentiment is an optimistic or pessimistic one (amongst other emotive distinctions). There are many other settings in which this objective arises: political sentiment polls, consumer research, medical discussions, economic news announcements, magazine editorials. However, we chose financial sentiment as the focus of the methodology because of the vast amount of related data, such as stock prices, trading volume, etc, which enable validation of our "sentiment" index. To do so, we implement a set of algorithms, some language-dependent, others not, using varied parsing and statistical approaches. The methodology used here has antecedents in the text classification literature (see Koller and Sahami [1997] and Chakrabarti, Dom, Agrawal and Raghavan [1998]). These papers classify textual content into natural hierarchies, and is a popular approach employed by web search engines. Our technology attempts to classify the *emotive* content of text, rather than its *factual* content, a different, and more complex problem.

The vastness of message board sentiment is staggering. For the major stock message boards, active stocks experience message postings once every 3-5 minutes. This makes for a rich source of trading sentiment. In order to keep the analysis in this paper focused, we used only messages from Yahoo's message board. While Yahoo is the most popular message board, accounting for close to 90% of posting volume, there are other major boards, notably Motley Fool, Silicon Investor, and Raging Bull. The volume of messages is huge, and if all stocks were taken together, the overwhelming amount of data would challenge the performance of any algorithm.

There is growing evidence that chat board sentiment influences and is influenced by the stock market. Wysocki [1998] reports that variations in daily message posting volume are related to news and earnings announcements, and this relationship is statistically significant. Posting volume is highest for firms with extreme past returns, low book to market values, high price-earnings ratios, high analyst following and low institutional holdings. While Wysocki's work is based purely on message counts, we go further and determine the emotive content of each message. Incorporating statistical language techniques, we provide a new approach

to explore the relationship of internet sentiment to stock returns.

The internet has enabled disparate, yet reliable sources of corporate sentiment. The "whisper" number, an aggregate of informal earnings forecasts self-reported by individual investors is now watched extensively by market participants, large and small.[3] Bagnoli, Beneish and Watts [1999] examined the predictive validity of whisper forecasts, and found them to be superior to those of First Call (Wall Street) analysts. Moreover, a trading strategy based on whispers yielded better than risk-adjusted profits (the interested reader may find it worthwhile to visit the URL: `www.whispernumber.com`). The whisper is a poll number, and the number of votes in these polls is typically small (10-50). The methodology in this paper, which allows inference based on thousands of messages, effectively expands poll volumes by immense factors and gathers a greater quantity of information.

Our methodology is able to analyse thousands of messages quickly, and return a threshold of accuracy comparable to human agreement levels. Moreover, the sentiment index results in intuitive correlations with financial time series. The results presented in the paper (a) validate the algorithms used, and (b) show that message board postings are reactive and not predictive of stock price movements, which is supportive of market efficiency. Tumarkin and Whitelaw [2001] also find similar results using self-reported sentiments on the Raging Bull message board. While only about 1 in 10 messages posted also contains a self-reported sentiment, Our paper goes further in that it analyzes each message for sentiment, allowing greater aggregation of information. In addition, our paper does not take the stated sentiment of the poster as given; instead, our algorithms analyze the textual content for sentiment directly.

The scheme of the paper is as follows. Section 2 provides an introduction to the technology used for message classification, and Section 3 describes the data. The classification models are in Sections 4. Results are presented in Section 5. Conclusions and extensions are in

[3]Whispers are forecasts of the quarterly earnings of a firm posted to the web by individuals in a voluntary manner. The simple average of these forecasts is presented on the whisper web page, along with the corresponding forecast from First Call, which is an aggregate of the sentiment of Wall Street analysts.

Section 6.

2 Model Structure

Chat board messages are classified by our algorithms into one of 3 types: bullish (optimistic), bearish (pessimistic) and neutral (comprising either spam messages or messages that are neither bullish or bearish). At the outset of any time period, we initialize the sentiment index to zero, and then increment it by 1 whenever a bullish message is posted. Likewise, we decrement the index by 1 when a bearish message is posted. This provides a time-series of sentiment for any stock we choose to analyze. We gathered data real time over different sub-periods and stocks in the year 2000, and provide an illustrative application of the methodology.

Five distinct algorithms, each with different conceptual underpinnings, are used to classify each message. They comprise a blend of language features such as parts of speech tagging, and more traditional statistical methods.[4] Before initiating classification, the algorithms are tuned on a training corpus. A small subset of pre-classified messages is used for training the algorithms.[5] The algorithms "learn" sentiment classification rules from the pre-classified data set, and then apply these rules out-of-sample. A simple majority of the five rules is required before a message is finally classified, else it is discarded. This *voting* approach results in a better signal to noise ratio for extracting sentiment.

[4] This paper complements recent work in Chen, Das, Tom and Waddle [1999], where Bayesian analysis and support vector machines were used to attempt classification. Support vector machines are optimization methods that classify content. See the papers by Vapnik [1995], Vapnik and Lerner [1963], Vapnik and Chervonenkis [1964], Joachims [1999] for a review. These approaches are computationally intensive and are often run on parallel processors. Moreover, they have been used for more than 30 years, and the technology is well-developed. In this paper we did not employ support vector machines, choosing to focus on purely analytic techniques that did not require optimization methods in the interests of computational efficiency.

[5] The training corpus is kept deliberately small to avoid over-fitting, which is a common ailment of text classification algorithms. The training set is described in more detail in the sequel.

2.1 Model Schematic

Figure 1 presents the flowchart for the methodology. We begin by using a web scraper program to download messages from the world-wide web. These messages are then (i) cleansed by parsing out HTML tags, (ii) expanding abbreviations, and (iii) undertaking some preliminary grammar processing, in particular, tagging negation effects in sentences for later inference. These processed messages are then fed to five separate classification algorithms to categorize them as buy, sell or null types.

There are 3 databases that support the classification algorithms. First, the algorithms access the CUOVALD English dictionary, which comprises base language data. Second, the algorithms use a "lexicon" which is a hand-picked collection of finance words (such as bull, bear, uptick, value, buy, pressure, etc). These words form the variables for statistical inference undertaken by the algorithms. The third database is the "grammar" or the training corpus. It proxies for a base set of rules used in the statistical analysis. The following subsections provide more details on the lexicon and grammar, as well as a description of message preprocessing.

2.2 The Lexicon

Words are the heart of any language inference system, and in a specialized domain, this is even more so. In the words of F.C. Bartlett,

> "Words ... can indicate the qualitative and relational features of a situation in their general aspect just as directly as, and perhaps even more satisfactorily than, they can describe its particular individuality, This is, in fact, what gives to language its intimate relation to thought processes."

The sentiment classification model relies on a lexicon of "discriminant" words. This lexicon was designed using domain knowledge and statistical methods. A discriminant function

was used to statistically detect which words in the training corpus were good candidates for classifier usage. The details of the discriminant function are provided in Section 4.3.

The features of the lexicon are as follows:

1. These words are hand-selected based on a reading of several thousand messages.

2. The lexicon may be completely user-specified, allowing the methodology to be tailored to individual preference. For example, if the user is only interested in messages that relate to IPOs, a lexicon containing mostly IPO-related words may be designed. (The grammar, i.e. the training set would also be correspondingly tagged).

3. For each word in the lexicon, we tag it with a "base" value, i.e. the category in which it usually appears. For example, the word "sell" would be naturally likely to appear in messages of type SELL, and we tag "sell" with base value 1. If the word is of BUY type, we tag it with value 3, and NULL words are tagged 0.[6] Every time a new word is added to the lexicon, the user is required to make a judgment on the base type.

4. Each word is also "expanded", i.e. appears in the lexicon in all its forms, so that across forms, the word is treated as one word. This process is analogous to stemming words, except that we exhaustively enumerate all forms of the word rather than stem them.[7]

5. Each word is also entered with its "negation" counterpart, i.e. the sense in which the word would appear if it were negated. Negation is detected during preprocessing (described later) and is used to flag portions of sentences that would be reversed in meaning.

An example of a lexical entry along with its base value, expansion and negation is provided below:

[6]These tag values seem odd, but are used in the algorithms; the numbers are an implementation detail, and may vary across algorithms. There is no special reason for the choice of the numbers used.

[7]Stemming is the process of mapping a word to its root word. For example, the root of "buying" is "buy".

```
3 favorable favorite favorites favoring favored
1 favorable__n favorite__n favorites__n favoring__n favored__n
```

All forms of the word appear in the same line of the lexicon. As can be seen, a tag is attached to each negated word in the second line above. The default classification value (the "base" value) is specified at the beginning of the line for each lexical item (i.e. a 0, 1 or 3).

The current size of the lexicon is approximately 300 distinct words. Ongoing, incremental analysis results in a continuous stream of additions to the word set.

Based on the training corpus, we can compute the *discriminant value* of each item in the lexicon (as in [1998]). This value describes the power of the lexical item in differentiating message types. For example, the word "`buy`" is likely to be a strong discriminator, since it would be suggestive of positive sentiment. The user's goal is to populate the lexicon with words of high discriminant value, and this is where the application of domain expertise is valuable.

2.3 The Grammar

A grammar may be defined as a set of functions or rules applied to the lexicon to extract sentiment. Correspondences between word sets, language features and classification types comprise the grammar. The training corpus is one simple form of a grammar. This set of messages, once hand-tagged, may be thought of as a set of rules that indicate the classification of other messages. One way to approach classification of any message is to search the grammar for a rule that may be applied to the message. For example, a distance function under a carefully chosen metric may be used to identify the applicable rule. Intuitively speaking, any message is explored for affinity to a rule, and depending on its properties, it may take on the classification character of the rule.

We may think of the grammar as Roger Schank did, i.e. it is a "conceptual processor". With stock market messages, the language is cryptic, and the grammar rules must work together so as to make sense of the "thought bullets" posted to the web. Schank states this

particularly well: "People do not usually state all the parts of a given thought that they are trying to communicate because the speaker tries to be brief and leaves out assumed or inessential information. The conceptual processor searches for a given type of information in a sentence or a larger unit of discourse that will fill the needed slot." Our algorithms combine grammar rules and lexical items to achieve automated classification.

Before applying the lexicon-grammar based algorithms, each message is preprocessed to enable cleaner interpretation. First, we carry out "HTML Cleanup", which removes all HTML tags from the body of the message as these often occur concatenated to lexical items of interest. Examples of some of these tags are: `<BR>,<p>,&quot`, etc. Second, we expand abbreviations to their full form, making the representation of phrases with abbreviated words common across the message. For example, the word "`ain't`" is replaced with "`are not`", "`it's`" is replaced with "`it is`", etc. Finally, we handle negation words. Whenever a negation word appears in a sentence, it usually causes the meaning of the sentence to be the opposite of that without the negation. For example, the sentence "`It is not a bullish market`" actually means the opposite of a bull market. Words such as "`not`", "`never`", "`no`", etc., serve to reverse meaning. We handle negation by detecting these words and then tagging the rest of the words in the sentence after the negation word with markers. These markers appear as "`--n`" concatenated to the end of the word. The lexicon is designed to contain words with negation markers, so that their presence in a message will be correctly interpreted. These three parsers deliver a clean set of messages for classification.

3 Data

Sentiment extraction using our automated algorithm comprises two steps: (i) training and (ii) classification. The algorithm is trained on a small data set and then implemented out-of-sample. We collected both training data and testing data to illustrate the methodology.

3.1 Training and Testing Corpus

Members of the more active stock message boards typically post about 200 messages a day to the board for a single stock.[8] These messages were downloaded using a "crawler" program, which is called a "web-scraper". Messages were downloaded from the biggest message board site, i.e. Yahoo. The crawler program makes it possible to download several thousand messages in a few minutes.

We created an intraday mixed sample via a real-time download of contemporaneous messages and stock prices for about 8 stocks during the last two months of 2000. Each day we obtained a file of all the posted messages and the concurrent stock price when the message was posted. Since this exercise was run in real time, there were periods for which some boards were technically inaccessible.[9] After accounting for these problems, we were able to collect a total of about 100 stock-days, amounting to well over 25,000 messages. This panel data set is used to illustrate intraday phase lags between the sentiment index and the stock time series.

Downloaded messages are fed to the classification algorithms. The algorithms are initially trained using a portion of the data, which we designate as the "training set", typically of size 300-500 messages. The number of messages is deliberately kept small so as to (a) assess whether the methods are amenable to a minimal amount of training, (b) allow for an extension to the approach where the training corpus is allowed to be time-varying, and (c) to prevent overfitting of the data, which is a common ailment in text classification algorithms, leading to poor out-of-sample performance.

All messages in the training set were hand-classified into three categories:

- **Buy**: this rating was assigned if the message indicated positive (i.e buy) sentiment.
- **Sell**: this was assigned if the message indicated negative sentiment.

[8] In qualification, it must be stated that there are about 200 active message boards. Many stocks do not see the posting volume that we observe on (say) Amazon's message board.

[9] This is caused by server rejection, machine downtime, network failure, message board repair, etc.

- **Neutral/Spam**: this rating applies when the message is neither buy nor sell, and may be simply a neutral message or a nonsense message, i.e. spam.

An examination of sample messages illustrates the high degree of ambiguity faced in the classification process. Some postings are clear and easy to classify such as the following pair of conflicting messages (the first is rated a buy and the second a sell, and we retain some of the fields in the original data file, so as to fix location of the message):

A positive message:

Yahoo AMZN 195006 12/19/99 3:26 pm The fact is..... The value of the company increases because the leader (Bezos) is identified as a commodity with a vision for what the future may hold. He will now be a public figure until the day he dies. That is value.

A negative message:

Yahoo AMZN 195007 12/19/99 3:30 pm Is it famous on infamous? A commodity dumped below cost without profit, I agree. Bezos had a chance to make a profit without sales tax and couldn't do it. The future looks grim here.

On the other hand, there are other messages that are somewhat ambiguous and often lead to two people differing in their categorizations of the message. For example, the following message is not easy to classify, and would most likely receive a null classification.

An ambiguous message:

Yahoo AAPL 100127 4/14/00 8:18 pm

fundamental model tells 6 month target.

Last night over dinner I talked with the famous Professor (Nobel Prize Winner) about how leading academics and Wall Street practitioners think about fundamental/economic analyses of stocks. Many professional apply these analyse to figure out the near to mid-term (such as 6month) price targets for stocks. His view is that,even though most stocks' near term price movements are difficult to predict no matter what kind of analysis you apply, on average, if a stock is too far away from it's intrinsic economic/fundamental value, its price tends to come back closer to the fair value.

Or we may need to analyze a message with an analogy, wherein the meaning would be considerably less obvious to a computer than to a human being, since external context is required. For example, the following message might well be classified as negative by an algorithm, when a human would perceive it to be positive. Hence, messages tend to vary widely in classification difficulty.

A message requiring context:

Yahoo AMZN 195016 12/19/99 4:01 pm

You're missing this Sonny, the same way the cynics pronounced that "Gone with the Wind" would be a total bust.

Such messages also lead to inconsistent classification across human subjects. Asking a second person to classify the training corpus revealed an ambiguity level of around 28 percent, i.e. the percentage of message classifications on which two humans disagreed. This analysis may be extended to testing agreement among many humans, and in the limit n humans. There is likely to be a natural percentage of messages on which all humans agree, some sort of "non-dissension limit."

Our data includes auxiliary information on the English language. To exploit parts-of-speech usage in messages, a dictionary was used to detect adjectives and adverbs for

the classifier algorithms. This dictionary is called CUVOALD (Computer Usable Version of the Oxford Advanced Learner's Dictionary).[10] This dictionary contains parts-of-speech tagging information and appropriate program logic was written to access this dictionary while analyzing messages for grammatical information.

In the next two sections, we present (a) an overview of the architecture of the methodology, and (b) details of the classification algorithms.

4 Classification Models

Each classifier algorithm relies on a different approach to message interpretation. Some of them are language-independent, and some are not. Each one comprises an intuitive approach. In addition, they are all analytical, and do not require any optimization, hence they are computationally efficient, making feasible the processing of huge volumes of data in real time. We describe each one in turn.

4.1 Naive Classifier [NC]

This algorithm is based on a word count of positive and negative connotation words. It is the simplest and most intuitive of the algorithms. Each word is parsed through the lexicon, and assigned a value $(-1, 0, +1)$ based on the default value (sell, null, buy) in the lexicon. If the net word count crosses a given threshold, we can classify it as a buy or sell, else it is treated as neutral. This algorithm constitutes a baseline approach to the sentiment extraction problem.

[10] The dictionary was downloaded from Birkbeck College, University of London. It is the creation of Roger Mitton of the Computer Science Department. It contains about 70,000 words, and covers most of the commonly used words in the English language. Informal tests of the dictionary showed that about 80-90 percent of the words in a message were found in the dictionary.

4.2 Vector Distance Classifier [VDC]

This algorithm treats each message as a word vector in D-dimensional space (where D is the size of the lexicon). Every word in the dictionary is indexed by its alphabetical order of appearance in the dictionary. Each message vector M is a representation of the word frequency in the message for every word in the dictionary. It contains mostly zeros, i.e. is a sparse vector, as most of the words in the dictionary do not appear in the message. A few of the elements of the vector contain positive integers denoting the number of times a word appears in the message. If any of the lexical items are phrases, then a count is also taken of the number of phrases in the message. Hence the word model may be expanded to a word and phrase model as required.

First, each hand-tagged message (or grammar rule) in the training corpus (grammar) is converted into a vector G, and hence occupies a location in D-dimensional Euclidian space. Next, each new message is classified by comparison to the cluster of pretrained vectors in this space. The angle θ between the message vector (M) and the vectors in the grammar (G) provides a measure of proximity.

$$\cos(\theta) = \frac{M.G}{|M|.|G|} \in [0, 1] \tag{1}$$

Each message is assigned the classification of the grammar rule with which it has the lowest angle. Variations on this theme are possible using sets of top-n closest rules. Note that $\cos(\theta) \in [0, 1]$, hence the vector distance classifier provides a measure of proximity in the form of percentage closeness. When the angle is small, $\cos(\theta)$ is closer to 1.

4.3 Discriminant-Based Classifier [DBC]

Words in the lexicon have differential importance for classification purposes. Some words, such as "buy" may be more indicative of sentiment than words such as "position". Using the training corpus, we can compute a measure of the discriminating ability of each word in our lexicon.

We compute Fisher's discriminant function for each word (see Chakrabarti, Dom, Agrawal and Raghavan [1998] for usage in another context). Let the set C denote the categories for our messages. In our case, $C = \{null, sell, buy\}$. The mean score (average number of times word w appears in a message of category i) of each word for each category is denoted μ_i, where i indexes category (we suppress all subscripts for w to simplify exposition here). Let messages be indexed by j. The number of times word w appears in a message j of category i is denoted m_{ij}. Let n_i be the number of times word w appears in category i. The discriminant formula for each word is:

$$F(w) = \frac{\frac{1}{|C|}\sum_{i \neq k}(\mu_i - \mu_k)^2}{\sum_i \frac{1}{n_i}\sum_j (m_{ij} - \mu_i)^2}, \quad \forall w. \tag{2}$$

This equation assigns a score $F(w)$ to each word w which is the ratio of the across-class (class i vs class k) variance to the average of within-class (class $i \in C$) variances. The larger the ratio, the greater the discriminating power of word w in the lexicon. Any word, which maximizes across-class variation and minimizes within-class variation is a good discriminant. As an extreme example, consider the case where each buy message contains the word "`bullish`" exactly two times. Also, this word never appears in a message of another class. Here, the across-class variation is very high, and the within class variation is zero. Thus, $F(w = \texttt{bullish}) = \infty$, i.e. the Fisher discriminant is infinitely powerful in discriminating buy messages from the others. Such a perfect discriminant word is rare in practice, but is the holy grail of this method. If found, classification only involves searching for this word w and basing classification on it.

The DBC classifier works in two steps. First, the discriminant value is computed for each word in the lexicon based on the chosen grammar. Second, a discriminant weighted score is calculated for each message, where buy words are assigned a positive sign and sell words a negative sign, each word being assigned its discriminant score. The sum total gives a score which may turn out to be negative or positive. Depending on whether the total discriminant for the message is within given threshold ranges, we assign a classification of buy, sell or null.

4.4 Adjective-Adverb Phrase Classifier [AAPC]

This classifier is based on the assumption that adjectives and adverbs emphasize sentiment and require greater weight in the classification process. In this algorithm, we employ parts of speech information to add discriminating value. Our goal is to focus on the more emphatic parts of the message. We wrote program logic for a parts of speech "tagger" which, in conjunction with the CUVOALD dictionary, searches for noun phrases containing adjectives or adverbs, i.e. in its simplest form, this would be an adjective-noun pair. Whenever this is detected, we form a "triplet", which consists of the adjective or adverb and the two words immediately following or preceding it in the message. This triplet usually contains meaningful interpretive information because it contains the adjective or adverb, both of which are parts of speech that add emphasis to the phrase in which they are embedded. Using this simple heuristic to identify possibly significant phrases, we then submit each phrase for lexical-grammar analysis, obtaining either a buy or sell token for the phrase. The net score of buy and sell tokens is then used to classify the message.

4.5 Bayesian Classifier [BC]

The Bayesian classifier relies on a multivariate application of Bayes' theorem (see Mitchell [1997], Neal [1996], Koller and Sahami (KS) [1997], and Chakrabarti, Dom, Agrawal and Raghavan (CDAR) [1998]). Recently, it has been used for web search algorithms, for classifying web content into communities, and in classifying pages on internet portals. KS [1997] and CDAR [1998] report that a hierarchical Bayesian classifier developed by them is able to mimic Yahoo's portal classification scheme to around 86% accuracy. Our model here is an adaptation of this technology for stock market sentiment.[11]

[11] Koller and Sahami develop a hierarchical model, designed to mimic Yahoo's indexing scheme. Hence their model has many categories and is more complex. On the other hand, their classifier was not discriminating emotive content, but factual content, which is arguably more amenable to the use of statistical techniques. Our task is complicated by the fact the messages contain sentiments, not facts, which are usually harder to interpret. Chen, Das, Tom and Waddle [1999] used an initial version of the Bayesian classifier in earlier work

The model consists of three components: (i) lexical words[12], (ii) messages or text, and (iii) categories (bullish, bearish, spam or neutral). Therefore, we get a word-message-class (w, m, c) model. The words w are the specially chosen discriminant words, that reside in the lexicon. The messages m are the target of study in the model. The categories or classes c are the types of messages defined by the model.

The Bayesian classifier works off word-based probabilities, and is thus indifferent to the structure of the language. Since it is language-independent, it has wide applicability. In particular, the method enables investigation of message boards in other financial markets, where the underlying language may not be English.

Our notation is as follows. The total number of categories or classes is $C(=3)$, $c_i, i = 1...C$. Each message is written as $m_j, j = 1...M$. We define M_i as the total number of messages per class i, and $\sum_{i=1}^{C} M_i = M$. Lexical words (w) are indexed by k, and the total number of words is T (as opposed to the total number of words in the dictionary, D). The set of lexical words is $F = \{w_k\}_{k=1}^{T}$. These are used to drive message classification. We use discriminant analysis (described in the previous section) to determine a good set of words $\{w_k\}$.

Let $n(m, w) \equiv n(m_j, w_k)$ be the total number of times word w_k appears in message m_j.[13] Hence, using the lexicon as a base, we maintain a count of the number of times each lexical item appears in every message in the training data set. This leads naturally to the variable $n(m)$, the total number of words in message m including duplicates. This is a simple sum, $n(m_j) = \sum_{k=1}^{T} n(m_j, w_k)$.

An important quantity is the frequency with which a word appears in a message class.

on stock message classification, coupled with support vector machine technology. The reader may obtain details of the hierarchical scheme by referring to the technical descriptions in [1997] and [1998].

[12]Recall that the lexicon contains "key words" from the finance domain which are used to determine the sense of the message.

[13]We simplify notation by suppressing subscripts as far as possible. The reader will be able to infer this from the context.

Hence, $n(c, w)$ is the number of times word w appears in all $m \in c$. This is

$$n(c_i, w_k) = \sum_{m_j \in c_i} n(m_j, w_k) \tag{3}$$

This measure has a corresponding probability: $\theta(c_i, w_k)$ is the probability with which word w appears in all messages m in class c:

$$\theta(c, w) = \frac{\sum_{m_j \in c_i} n(m_j, w_k)}{\sum_{m_j \in c_i} \sum_k n(m_j, w_k)} = \frac{n(c_i, w_k)}{n(c_i)} \tag{4}$$

We require that $\theta(c_i, w_k) \neq 0$, $\forall c_i, w_k$. Hence, an adjustment is made to equation (4) via Laplace's formula which is

$$\theta(c_i, w_k) = \frac{n(c_i, w_k) + 1}{n(c_i) + K}.$$

where K is the size of the set of lexical words. $\theta(c_i, w_k)$ is unbiased and efficient. If $n(c_i, w_k) = 0$ and $n(c_i) = 0, \forall k$, then every word is equiprobable, i.e. $\frac{1}{K}$. We now have the required variables to compute the conditional probability of a message j in category i, i.e. $\Pr[m_j|c_i]$:

$$\begin{aligned} \Pr[m_j|c_i] &= \left(\begin{array}{c} n(m_j) \\ \{n(m_j, w_k)\} \end{array} \right) \prod_{k=1}^{T} \theta(c_i, w_k)^{n(m_j, w_k)} \\ &= \frac{n(m_j)!}{n(m_j, w_1)! \times n(m_j, w_2)! \times ... \times n(m_j, w_T)!} \times \prod_{k=1}^{T} \theta(c_i, w_k)^{n(m_j, w_k)}. \end{aligned} \tag{5}$$

$\Pr[c_i]$ is the proportion of messages in the posterior (training corpus) classified into class c_i.

The classification goal is to compute the most probable class c_i given any message m_j. Therefore, using the previously computed values of $\Pr[m_j|c_i]$ and $\Pr[c_i]$, we obtain the following conditional probability (applying Bayes' theorem):

$$\Pr[c_i|m_j] = \frac{\Pr[m_j|c_i].\Pr[c_i]}{\sum_{i=1}^{C} \Pr[m_j|c_i].\Pr[c_i]}. \tag{6}$$

For each message, equation (6) delivers three posterior probabilities, $\Pr[c_i|m_j], \forall i$, one for each message category. The category with the highest probability is assigned to the message. The Bayesian classifier is easy to compute since no optimization is required. It has also been shown to be highly useful in related domains (see Koller and Sahami [1997],

Chakrabarti, Dom, Agrawal and Raghavan [1998]). The probabilities are computable in deterministic time. In fact, none of the five classifier methods used here entail issues of numerical convergence. Given the huge size of the data sets involved, this is an important consideration of the overall algorithm design.

4.6 Voting amongst Classifiers

The voting approach exploits the results of the NC, VDC, DBC, AAPC and BC methods. A voting scheme attempts to make a classification decision based on the intuition that there is available information that is not exploited when classifiers are used in isolation, instead of in conjunction.

We experimented with three voting schemes. First, we worked with a simple majority rule, i.e. 3 of 5 classifiers should agree on the message type. Second, we raised this threshold to requiring agreement amongst 4 of the 5 classifiers. Third, we only classified the messages when we obtained consensus, i.e. all 5 agreed. There is an inverse relation between the accuracy of classification and the number of messages classified. As we go from simple majority to consensus, the accuracy of classification improves dramatically, but the number of messages classified falls as well. Since the lack of consensus indicates the degree of ambiguity of the message, imposing an unwillingness to classify ambiguous messages ensures that the algorithm improves in accuracy, as it only looks at more interpretable messages.

4.7 Message Ambiguity

The grammar is created by hand-tagging the sentiment category for each message in the training set. Human classifiers may disagree on the classification of such messages. Correspondingly, this tends to drive up the classification error rate for each individual algorithm. It is often better to abstain from classification than to classify incorrectly, and we inject this cautious approach into our algorithm through the use of voting rules, whereby decisions are taken by a committee of the five different algorithms. Looking for majority or consensus

amongst the classifiers will increase classification accuracy, if messages that do not lead to consensus are not classified. The accuracy of attempted classification should increase, *but only if* the theoretical ideas embedded in the classifiers are valid. Otherwise, if the classifiers were generating only random classifications, we would actually see them obtaining erroneous consensus. In our tests, we found that voting schemes resulted in sharp improvements in classification accuracy, providing evidence of consistency across the algorithms.

Messages in chat rooms are often highly ambiguous, implying that quite often, two people would be likely to disagree on the classification. In order to gauge the extent of this, a re-classification of the training corpus was undertaken by a second human subject. Of the 374 training messages, and 64 test messages, the two human subjects agreed on on the classification of only 72.46% of the messages.

We may like to think of the mismatch percentage of 27.54% (100.00-72.46) as the "ambiguity coefficient" of the message boards. A more stable version of this coefficient would be one obtained from many (say n) human subjects, for reasonably large n (approximately $n \sim 10$), where the agreement percentage is based on the consensus of all n people. This might well result in an ambiguity coefficient a little higher than from just a few subjects. It is also intuitive that as we increase n, the ambiguity coefficient will first rise rapidly and then taper off to an asymptote, as there will be a core set of messages on which there can be little disagreement. Hence there are two benchmarks of algorithm performance. One is perfect performance, i.e. a comparison with 100% accuracy rates, and the second is the human benchmark, i.e. an "agreement" coefficient, equivalent to 100 minus the ambiguity coefficient. Of course, the worst-case benchmark, i.e. random classification may also be used. With three sentiment categories, this benchmark is 33%.

4.8 Algorithm Performance

Any analysis of results of the methodology must be based on simple performance metrics. The natural metric here is the accuracy of *attempted* classification. The percentage accuracy

is:

$$\text{Accuracy}(\%) = \frac{\text{No of correct classifications}}{\text{No of attempted classifications}}$$

Therefore, if we have 50 test messages, discard 10 as ambiguous and classify correctly 20 of the remaining messages, the accuracy is 50%.

For each method, including the voting schemes, across all messages, we computed the classification accuracy. The results are presented in Table 1. A naive scheme would result in a classification accuracy of 33%, since there are 3 categories of messages. The rate at which classification accuracy increases as greater consensus requirements are imposed provides a justification for the use of voting-based methods.

The last two columns in the table describe the inverse relationship between attempted classification and classification accuracy. The simple majority scheme (3-votes) delivers 62% accuracy while achieving an attempted classification rate of 80% which is encouraging when compared with the human agreement coefficient of around 72%. We decided to use the 3-vote scheme in further analysis using much more detailed data.

Most of the rules do sufficiently well, returning accuracy rates above 50%. The Bayesian classifier does best, speaking to the encouraging results in the literature on probabilistic language analysis (see Charniak [1993]). The most interesting results come from the adjective-adverb phrase classifier (AAPC). The use of triplets containing adjectives or adverbs appears to pick up sentiment to a degree of accuracy comparable to the more formal Bayesian classifier. This occurs even though a very simple parts-of-speech tagging approach is employed. This possibly predicates the use of language based classification rules in addition to statistical techniques. Intuitively, the structure of language used must contain clues as to the sentiment of the writer. Emotion translates into expression, and is reflected in the parts of speech used.

In addition to the five stand alone algorithms, voting schemes allowed for higher accuracy rates. Classification was attempted with k-voting schemes, where $k = 2, 3, 4, 5$, i.e. ranging from bare majority to complete consensus. When $k = 2$, there is no improvement in classifi-

cation, but for $k > 2$, accuracy ramps up. Overall, $k = 3$ seems to be a fair balance between accuracy and classification rates. It provided an accuracy percentage of 62% (versus a naive level of 33%).

5 Illustrative Applications of the Methodology

The time series of classified messages for any stock provides a sentiment index by inter-temporal aggregation of positive and negative messages. We examine the relationship of the extracted sentiment series to stock prices. At the outset, a visual relationship between the stock graph and the sentiment graph provides casual validation of the methodology.

We examine data at high frequency (i.e. intraday). For the last quarter of 2000, we ran our message analysis programs on a real time basis on the following stock boards at Yahoo: AAPL (Apple Computer), AMZN (Amazon Inc), CDNW (CD Now), CSCO (Cisco Systems), DELL (Dell Computer), EBAY, ITWO (i2 Technologies), MSFT (Microsoft Corp), and YHOO (Yahoo). For every stock we recorded the message and the contemporaneous stock price in a separate file for each day for which we were able to access the board continuously for all trading hours.[14] Hence we collected about 100 stock+day combinations. This number may have been higher if we had run the programs for a longer period, or if the downtimes (from server breakdowns, board rejections, network failures, etc) had been less frequent. Nevertheless, the roughly 100 files we obtained contained more than 25,000 messages.

For each stock-day combination we classified the messages and used the time series of messages to create the sentiment index. Starting the index at zero at the beginning of the day, we update it with each message, adding 1 for positive messages and subtracting one for pessimistic ones. This provides a time series of sentiment for the 24 hour period. In Figures 2-7, we present plots representing the variety of relationships between the stock price and

[14] The time stamp of stock price quotes was corrected if the "real-time" source was time-lagged. In some cases, there was a 15 minute delay in quote posting on Yahoo boards. We made the required correction to ensure the synchronicity of sentiment and stock quotes.

our algorithm-generated sentiment index. All the plots show a close visual relation between stock price and sentiment, leading to subsequent analysis of the lead-lag relationship between stock prices and sentiment.

A variety of plot shapes was evidenced. We state here a few examples to demonstrate the different types of relationships we observed between the stock price series and sentiment. For example, on some days, message board sentiment responds almost contemporaneously to stock price movements. The plot for Apple Computer on 18th October, 2000 (Figure 2) reveals that the sentiment index reacts very quickly to a sharp drop in Apple's stock price. In contrast, on October 20, 2000, Apple Computer's stock (Figure 3) reflects the built-up positive sentiment from before the opening of trading. In Figure 4, on 7th December, 2000, Amazon's sentiment shows conflicting lead-lags, since in some portions of the plot, sentiment appears to lead, and in others it lags the stock price move. On other days, such as for Amazon on 11th December, 2000 (Figure 5), there appears to be almost no relationship between sentiment and stock price. Figure 6 for Dell Computer on 9th November, shows that sentiment was a precursor to stock price change, which also appears to be the case in Figure 7, the graph of Dell Computer on 13th November, 2000. These plots are indicative of the striking relationship between the sentiment index and the traded stock price, which is the focus of analysis in the following subsections.

Figures 2-7 visually demonstrate that sentiment tracks the stock price. Moreover the figures offer an alternative mode of representing a "stream" of sentiment data. The density of sentiment arrival may be inferred from the thickness of the plotted sentiment line on the graph. This becomes clear from a plot of the sentiment index for Apple Computer Inc. around a day on which the company announced an adverse earnings report. On September 28, 2000 Apple Computer revealed reduced earnings. The announcement was made at 4pm Eastern time. From the graphs (see Figure 8) for the Apple sentiment index, it is apparent that the message boards had not anticipated the announcement. A day earlier, on September 27, the sentiment index tracked up and then down, but revealed no persistent direction (the thin line shows low sentiment volume). A day later, when the announcement was made, just

after 4pm, the sentiment index crashed and continued to take a beating for the next few days. The public announcement created an immense volume spurt on the message board (notice that the plot gets substantially thicker). Between 4pm and midnight on September 28, there were more than 2000 messages posted to the Yahoo board. There was clearly plenty of disagreement too, as the net index was -150, which is the net of many buy and sell messages. Figure 8 shows the sentiment index for the 3 days around the event, i.e the day before, of and after the event. We see that the boards are swift to react, but are not clairvoyant with regards to stock behavior. The reaction tends to be quite persistent: the pattern after 4pm on September 28 extends well into September 29.

Undoubtedly, major news events are sentiment-forming. This illustration also informally validates our methodology, since the impact on our sentiment index is consistent, both in sign and size. A natural extension is to apply the methodology to detecting lead or lag relationships between sentiment and markets. We illustrate that this is feasible by conducting two analyses. Our first analysis explores the lead or lag at which the correlation between the stock and sentiment is maximized. The second approach uses a "phase-lag" measure to determine whether sentiment leads stock price movements. Our goal is to demonstrate the value of extracting an sentiment index from web sources.

5.1 Correlation Analysis

For the correlation analysis, we computed the correlation between stock price and sentiment when both series were contemporaneous (zero lag), and for lags -5 hours (sentiment leads stock price) to +5 hours (sentiment lags stock price) hours at intervals of 5 minutes each. The lags are numbered from -60 to +60, each representing a 5-minute block. Hence, for each stock-day we computed 121 correlation numbers. We recorded the lag or lead at which correlation was maximum. After this was done for all sets in the data, we plotted the histogram of the results. This is presented in Figure 9. It is apparent from the figure that a majority of times the stock price leads the sentiment index, predicating that small investor

sentiment on message boards is reactive rather than predictive. The average number of lags is 10.04, i.e. message board sentiment reflects the stock price change with an average delay of 50 minutes. However, the histogram shows that there are also many occasions on which the sentiment index leads the stock price. Therefore, one cannot conclusively infer that there is no information in the sentiment index. A sharper analysis is presented in the next section.

5.2 Phase-Lag Analysis

In this section, we develop a new approach to comparing graphs for the purpose of determining a phase-lag relationship. We design an algorithm that detects the predominant pattern in both, the stock price and the sentiment graphs, and then determines the extent to which this pattern materialized first in either graph.

The pattern recognition algorithm is based on a small and simple set of graph patterns. Figure 10 displays the 8 canonical patterns which we ascribe to any graph. For example, the "min-max" pattern is said to be present in a graph if the graph begins at its minimum value, and ends at its maximum value. The "up-down" graph is one where the maximum and minimum are not the end-points of the graph, and the maximum value comes before the minimum. The "down-max" graph is one where the graph ends at its maximum, but the minimum is not an end-point. The other five types are self-explanatory and are depicted in Figure 10.

Each graph may have from 1 to 3 predominant swings. For example, in the "max-min" graph the main change is a downward one, while in the "down-up" graph it could be either the up or down swing. For each stock price graph, we examine the matching sentiment graph and assess the number of hours by which the predominant pattern reflects a prediction of or reaction to the predominant pattern in the stock graph. If the graph pair contains more than one pattern match, then both are assessed for the lead-lag computation. Since public information available before trading will be reflected in the sentiment index, but not in the stock graph, we only looked for matching patterns after the opening of the trading day.

For each stock-day pair of stock and sentiment graphs, we determine the amount of lead or lag. The results are plotted in the histogram in Figure 11, which is a frequency distribution of phase-lags (in units of hours). In contrast to the correlation results in Figure 9, the phase-lag analysis produces much sharper results. The pattern-matching algorithm indicates poor predictive ability in the sentiment index, as can be seen from the fact that there are very few instances when the sentiment graph led the stock graph. This demonstrates that message postings appear to be largely reactive in nature. In fact, sentiment levels react strongly to stock price changes. On average, the message boards lag by 0.92 hours, i.e. approximately 55 minutes. The sharper results in Figure 11 may arise since, by looking only at predominant patterns, the algorithm detects lead-lag relationships *conditional* on an information change or stock move of reasonable size. The results indicate that information surprises were just that, i.e. they were not based on informed prior sentiment.

6 Conclusions

We presented a methodology for extracting small investor sentiment from stock chatboards. We believe that this approach offers a powerful complement to other methods of public sentiment extraction. The technology described in this paper is simple to implement, and easy to extend. The illustrative applications we developed offer validation, and demonstrate some ways in which the methodology may be applied.

There are many extensions of this work. First, the obvious thing to do is amass a data set comprising small investor sentiment for a large number of stocks (e.g. all the stocks in the S&P 500 index), and then examine the cross-sectional properties of sentiment. The current paper developed only illustrative applications of a time series nature, not the cross-section. Since this paper is primarily methodological, a large cross-sectional study exceeds its scope, and is better handled as a full-blown extension. Apart from cross-sectional analysis, the algorithms in this paper may be used to further many other types of research. First, there is a limited understanding of how tech stocks trade – with these algorithms, we could expand

the inference domain to cover not just sentiment, but also to look at the revealed trading behavior of small investors. Second, herding is a strong feature of volatile markets. Our algorithms will enable investigation of such market microstructure issues. Third, regulators are concerned that there may be a degree of market manipulation that goes undetected amongst the millions of messages posted to message boards every day. Large firms pay third party service providers to manually monitor their message boards to stymie manipulation. Our algorithms would enable better monitoring of market activity. Fourth, the sentiment index may be useful for testing theories in the domain of behavioral finance. Correspondingly, there are interesting possibilities for language research in computer science. First, there is a vast amount of data. Second, the data has known correlations with financial variables such as stock market indicators, trading volume, etc, so that inference may be checked extensively using financial variables as proxies. Third, the emotive classification problem is known to be very hard, yet it yields to several statistical techniques. This ameliorates the difficulty of the problem somewhat. Fourth, while NLP researchers have looked at word disambiguation in some detail, this research is an attempt to examine text disambiguation. Finally, little NLP research is aimed at determining the emotive content of text – this is now possible with the enormous amount of data that may be thrown at the problem. Finally, understanding the linkage between message board activity and the use of electronic trading may help in furthering our knowledge of investor behavior.[15]

[15]The natural extension is to apply the methodology in domains other than finance. There are several other internet sentiment forums other than those pertaining to stock markets, and the methodology may easily be extended to other fields of research. To do so, no modification needs to be made to the algorithms. However, a new lexicon and grammar need to be created, since these are domain-specific. The lexicon is a specialized domain dictionary, hence a user with domain expertise is required to generate one, but the effort takes only a few hours. The grammar is created by hand-tagging an appropriate training data set, which also requires minimal time.

References

[1999] Bagnoli, M., M. D. Beneish, and Susan G. Watts (1999). Whisper forecasts of Quarterly Earnings per Share, forthcoming, *Journal of Accounting and Economics*, v28(1), 1999.

[1998] Chakrabarti, S., B. Dom, R. Agrawal, and P. Raghavan. (1998). Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies, *The VLDB Journal*, Springer-Verlag.

[1998] Chakrabarti, S., B. Dom, P. Indyk. (1998). Enhanced hypertext categorization using hyperlinks, SIGMOD ACM, 1998.

[1993] Charniak, E. (1993). Statistical Language Learning, MIT Press, Cambridge, Massachusetts.

[1999] Chen, M., S. Das, D. Tom, and J. Waddle (1999). On Small Investor Information: Feature Selection and Categorization of Stock Message Board Postings, working paper, UC Berkeley.

[2000] Choi, J., D. Laibson, and A. Metrick (2000). "Does the Internet Increase Trading? Evidence from Investor Behavior in 401(k) Plans," NBER Working Paper No. W7878.

[2000] Das, S., A. Martinez-Jerez, and P. Tufano (2000). "e-Information: Preliminary Findings," working paper, Harvard Business School.

[2001] Godes, D., and D. Mayzlin (2001). Using Online Conversations to Forecast New Product Sales, working paper, MIT.

[1999] Joachims, T (1999). Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, B. Scholkopf and C. Burges and A. Smola (ed.), MIT-Press.

[1997] Koller, D., and M. Sahami (1997). Hierarchically classifying documents using very few words, International Conference on Machine Learning, vol 14, Morgan-Kaufmann, San Mateo, California.

[2001] Lam, S.L., and J. Myers (2001). Dimensions of Web Site Personas, working paper, UC Berkeley.

[1997] Mitchell, Tom (1997). Machine Learning, McGraw-Hill.

[1996] Neal, R.(1996) Bayesian Learning for Neural-Networks, Lecture Notes in Statistics, v118, Springer-Verlag.

[1998] Smola, A.J., and Scholkopf, B (1998). A Tutorial on Support Vector Regression, NeuroCOLT2 Technical Report, ESPIRIT Working Group in Neural and Computational Learning II.

[2001] Tumarkin, R., and R. Whitelaw (2001). News or Noise? Internet Postings and Stock Prices, *Financial Analysts Journal*, v57(3), 41-51.

[2001] Urban, G. (2001). "Uncovering Needs for New Products on the Internet," working paper, MIT.

[1963] Vapnik, V, and A. Lerner (1963). Pattern Recognition using Generalized Portrait Method, *Automation and Remote Control*, v24.

[1964] Vapnik, V. and Chervonenkis (1964). On the Uniform Convergence of Relative Frequencies of Events to their Probabilities, *Theory of Probability and its Applications*, v16(2), 264-280.

[1995] Vapnik, V (1995). The Nature of Statistical Learning Theory, Springer-Verlag, New York.

[1998] Wysocki, Peter (1998). Cheap Talk on the Web: The Determinants of Postings on Stock Message Boards, working paper No.98025, University of Michigan Business School.

Figure 1: **Schematic of the Algorithms used for Sentiment Extraction**

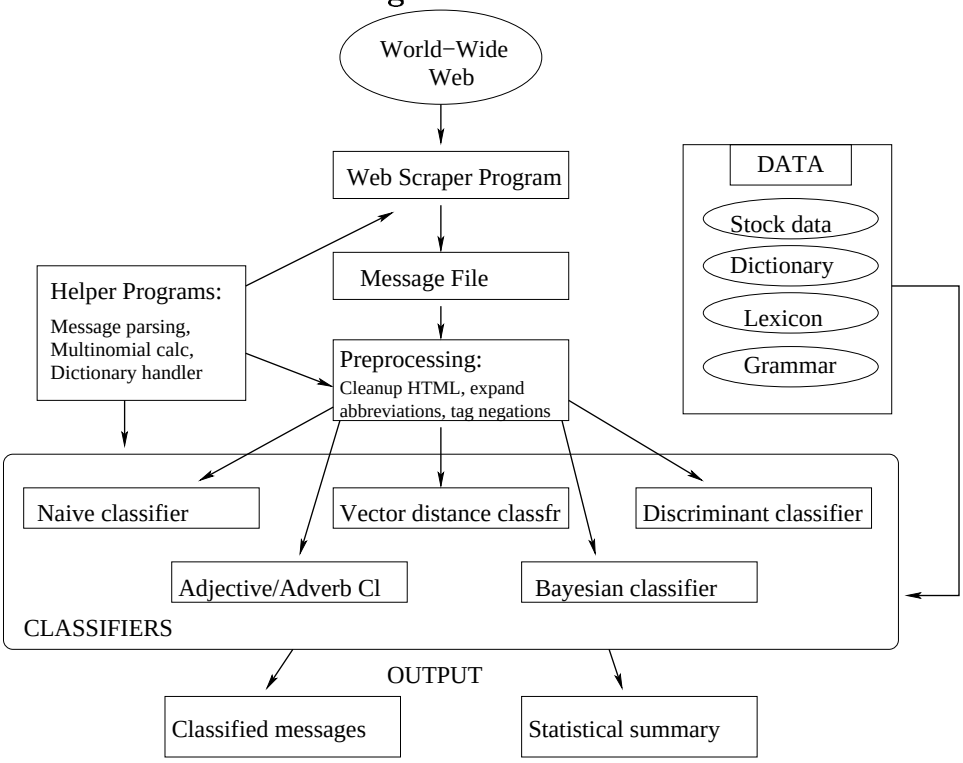


Table 1: **Algorithm Performance**

This table presents the results of the five different language algorithms used for classifying stock market messages. Four voting rules were also implemented based on the five algorithms. When using voting algorithms, if the threshold number of votes is not achieved, the message is not classified. and column (4) in the table reports the number of attempted classifications. The percentage accuracy based on attempted classifications is reported in column (5).

(1) Method No	(2) Classification Method	(3) Correct Messages	(4) Attempted Messages	(5) Percent Accuracy
1	Naive (NC)	33	64	51.56%
2	Vector-Distance (VDC)	27	64	42.19
3	Discriminant (DBC)	32	64	50.00
4	Adj-Noun (AAPC)	33	64	51.56
5	Bayes (BC)	34	64	53.13
6	2-votes	34	64	53.13
7	3-votes	32	52	61.54
8	4-votes	17	20	85.00
9	5-votes	5	5	100.00

Figure 2: **Apple Computer, 18-October-2000**
The two plots below depict the stock price and sentiment for the 24 hours of the day. Each point corresponds to the arrival of a message on the stock board. The stock price graph is usually flat in the region outside regular trading hours. The stock price is contemporaneously collected whenever a message arrives on the stock board.

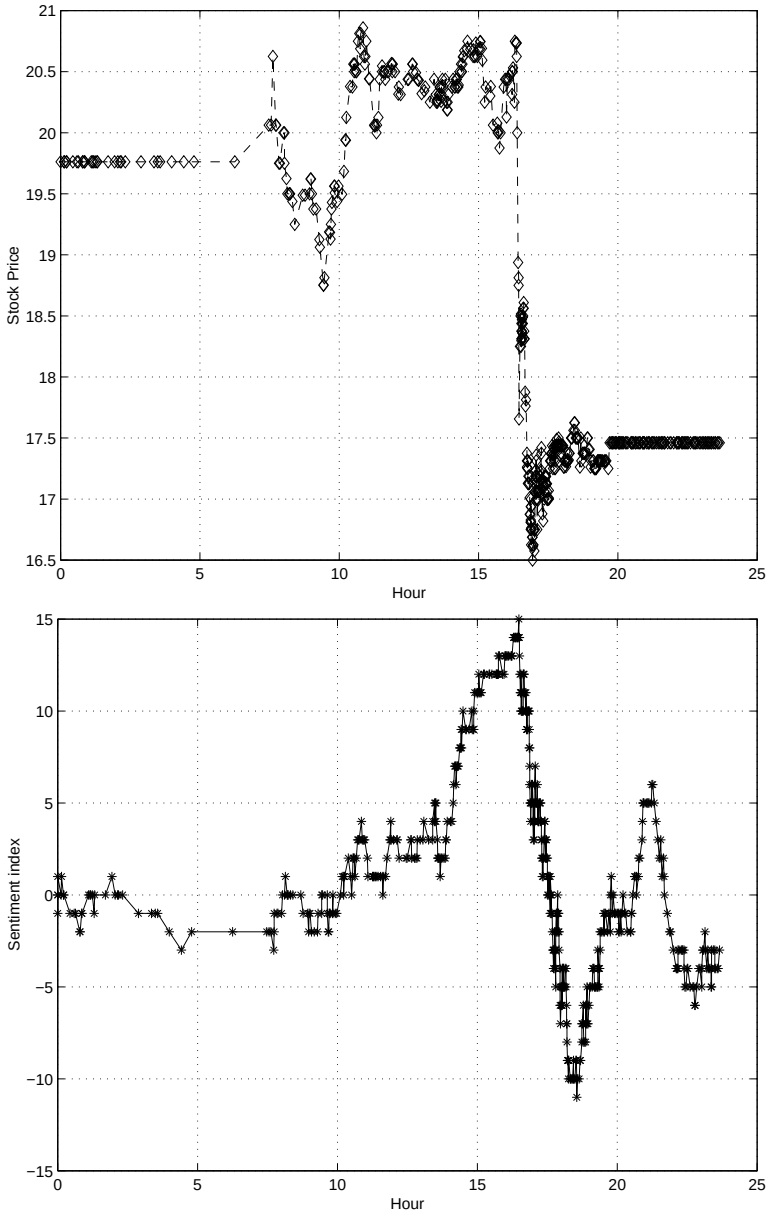


Figure 3: **Apple Computer, 20-October-2000**
The two plots below depict the stock price and sentiment for the first 16 hours of the day. Each point corresponds to the arrival of a message on the stock board. The stock price graph is usually flat in the region outside regular trading hours. The stock price is contemporaneously collected whenever a message arrives on the stock board.

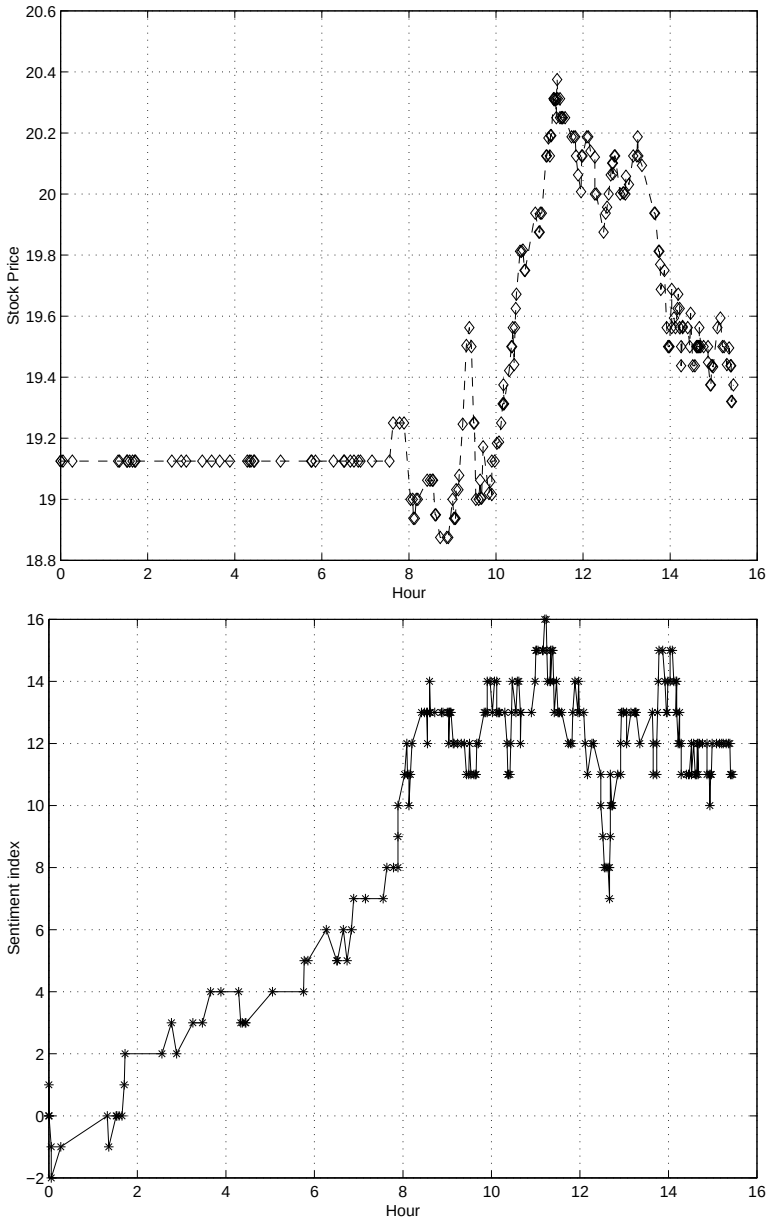


Figure 4: **Amazon, Inc., 07-December-2000**
The two plots below depict the stock price and sentiment for the 24 hours of the day. Each point corresponds to the arrival of a message on the stock board. The stock price graph is usually flat in the region outside regular trading hours. The stock price is contemporaneously collected whenever a message arrives on the stock board.

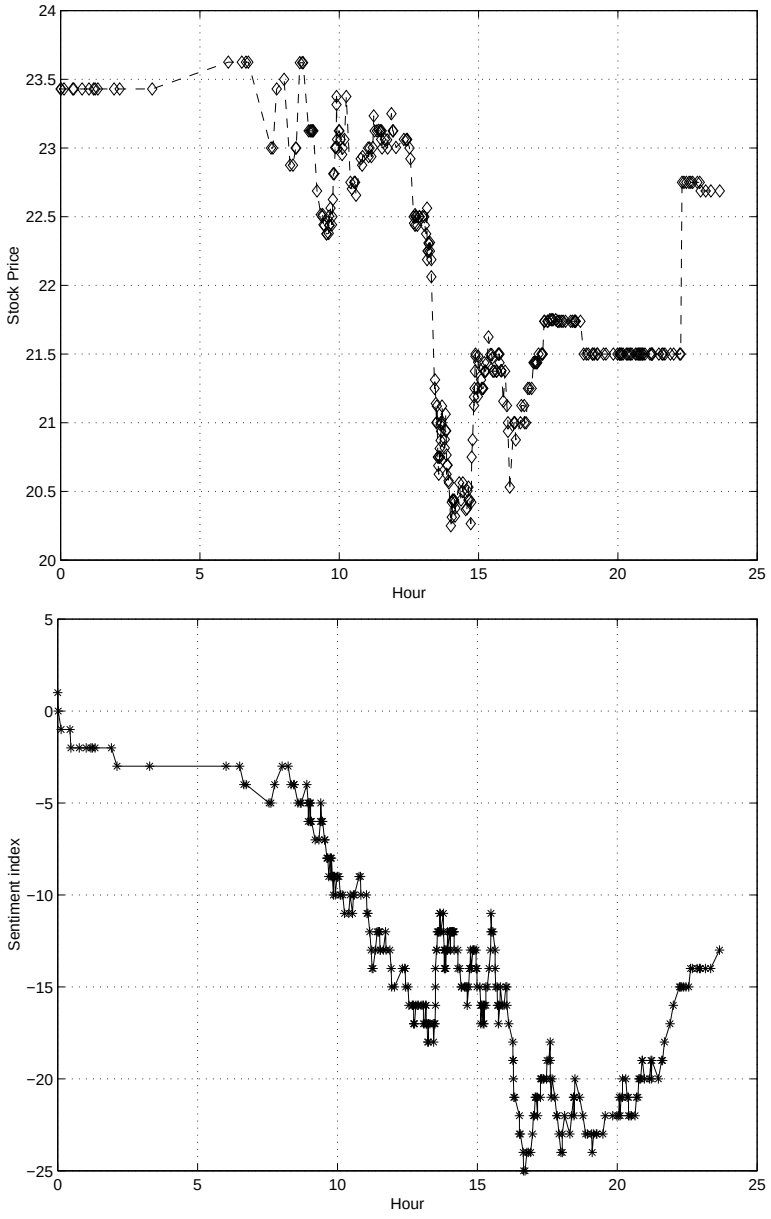


Figure 5: **Amazon, Inc., 11-December-2000**
The two plots below depict the stock price and sentiment for the 24 hours of the day. Each point corresponds to the arrival of a message on the stock board. The stock price graph is usually flat in the region outside regular trading hours. The stock price is contemporaneously collected whenever a message arrives on the stock board.

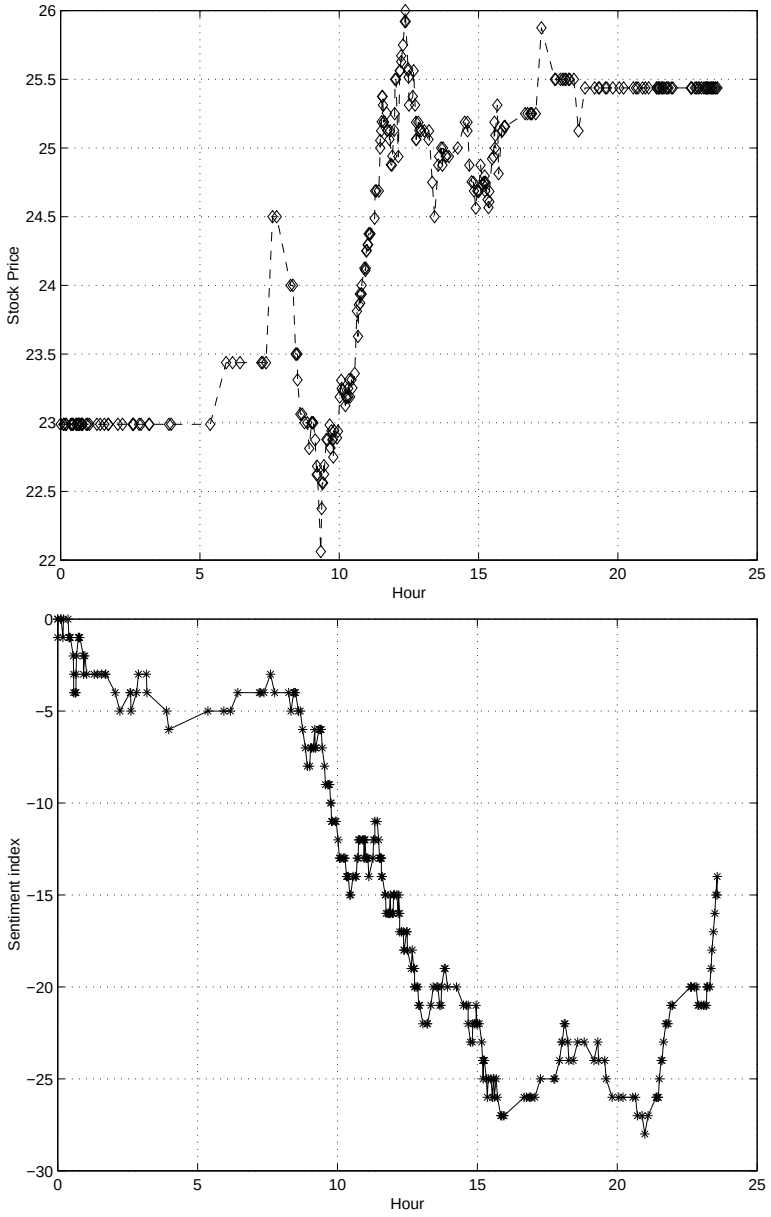


Figure 6: **Dell Computer, 09-November-2000**
The two plots below depict the stock price and sentiment for the 24 hours of the day. Each point corresponds to the arrival of a message on the stock board. The stock price graph is usually flat in the region outside regular trading hours. The stock price is contemporaneously collected whenever a message arrives on the stock board.

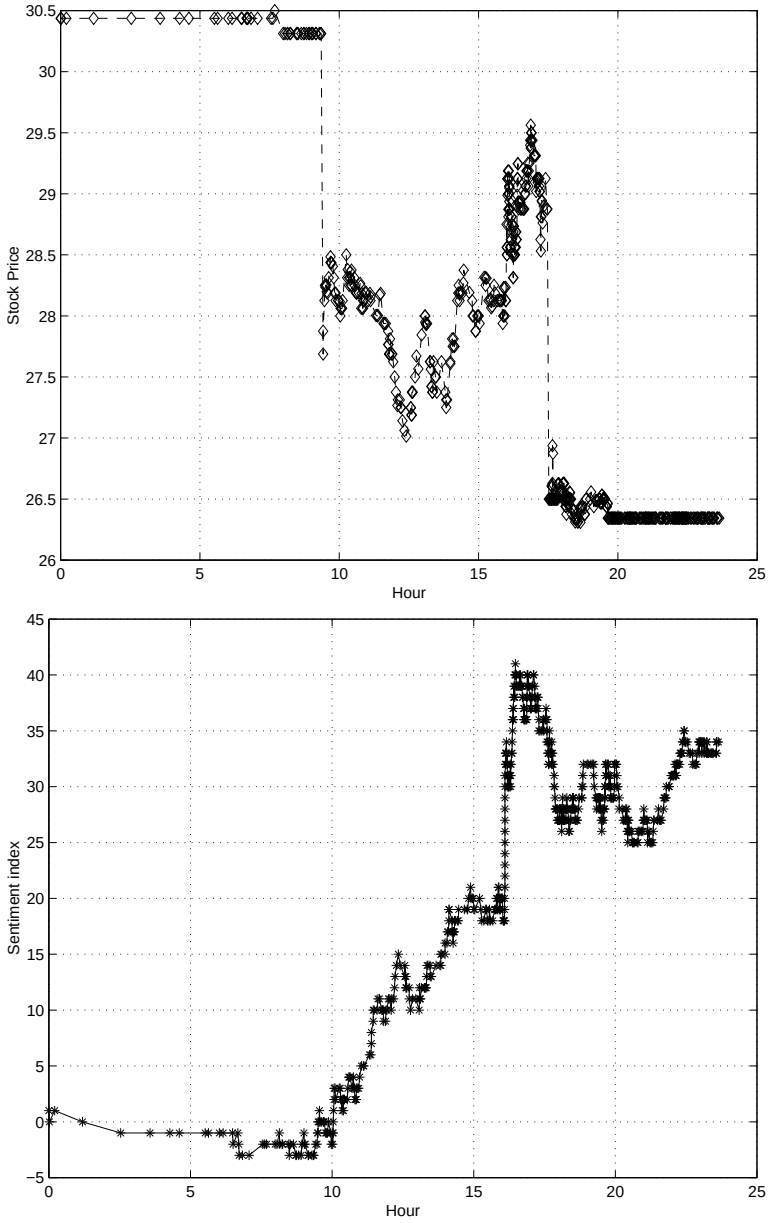


Figure 7: **Dell Computer, 13-November-2000**
The two plots below depict the stock price and sentiment for the 24 hours of the day. Each point corresponds to the arrival of a message on the stock board. The stock price graph is usually flat in the region outside regular trading hours. The stock price is contemporaneously collected whenever a message arrives on the stock board.

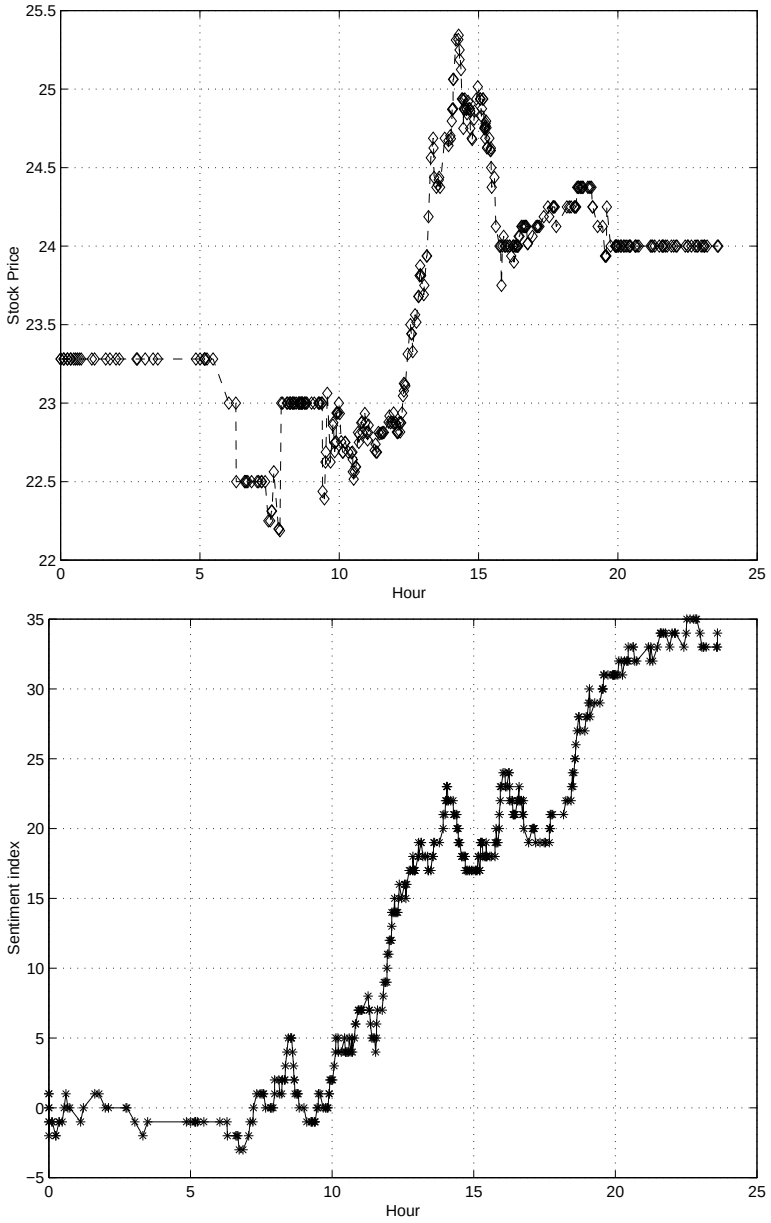


Figure 8: **Apple Computer, Event Effects and Sentiment** The three plots below depict the sentiment index of Apple Computer for September 27, 28, 29, 2000. The earnings announcement was made a little after 4pm on September 28, and the immediate and persistent reaction of sentiment is evident from the plots.

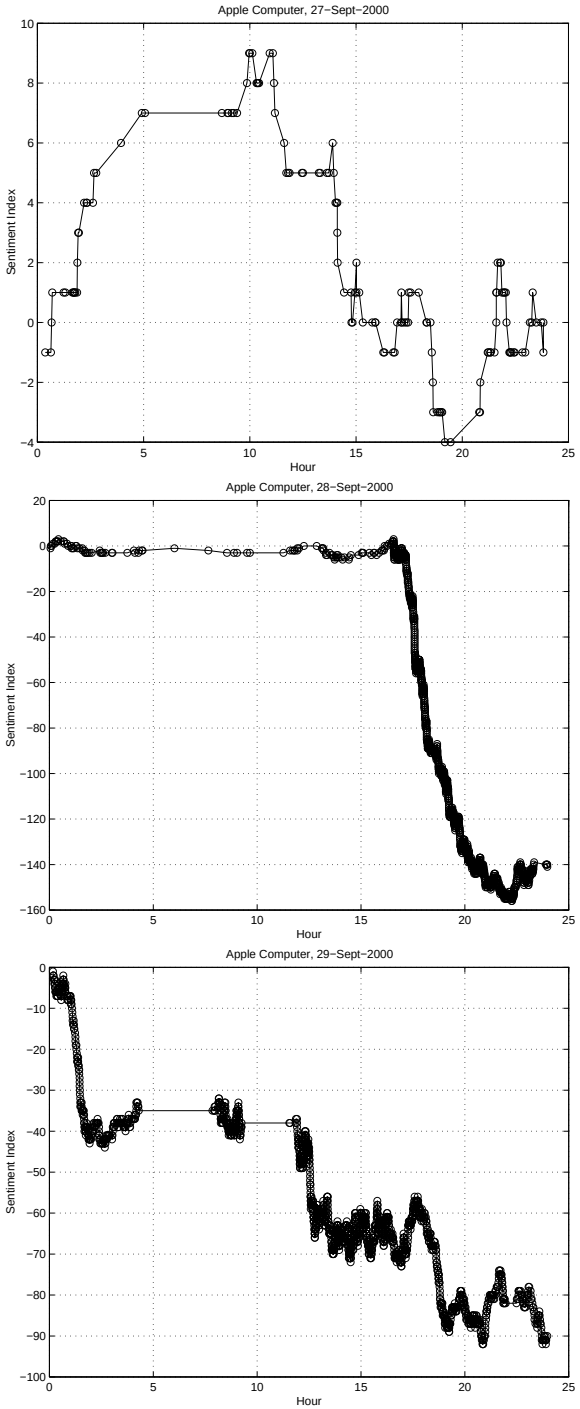


Figure 9: **Correlation based lag**
This figure presents a correlation analysis of the lead-lag relation between sentiment and the stock price. For each stock-day, the correlation between stock price and sentiment is computed when both series are contemporaneous (zero lag), and for lags -5 hours (sentiment leads stock price) to +5 hours (sentiment lags stock price) hours at intervals of 5 minutes each. The lags are numbered from -60 to +60, each representing a 5-minute block. Hence, for each stock-day we computed 121 correlation numbers. The the lag or lead at which correlation was maximum is reported in the histogram for all stock-day combinations in the data from the last two months of 2000.

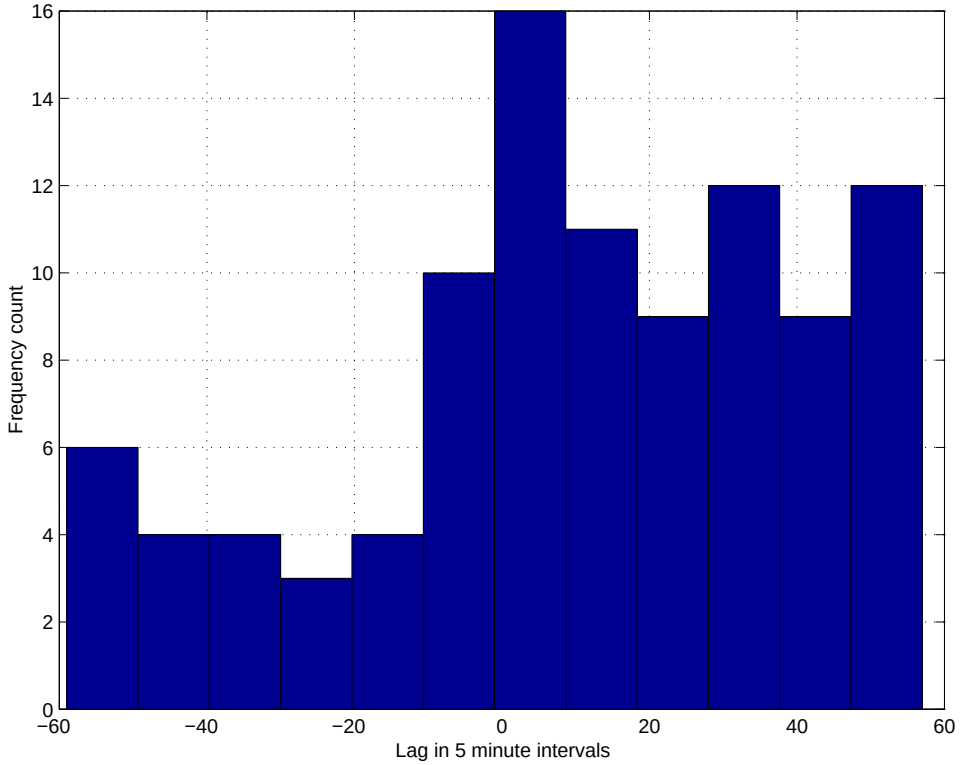
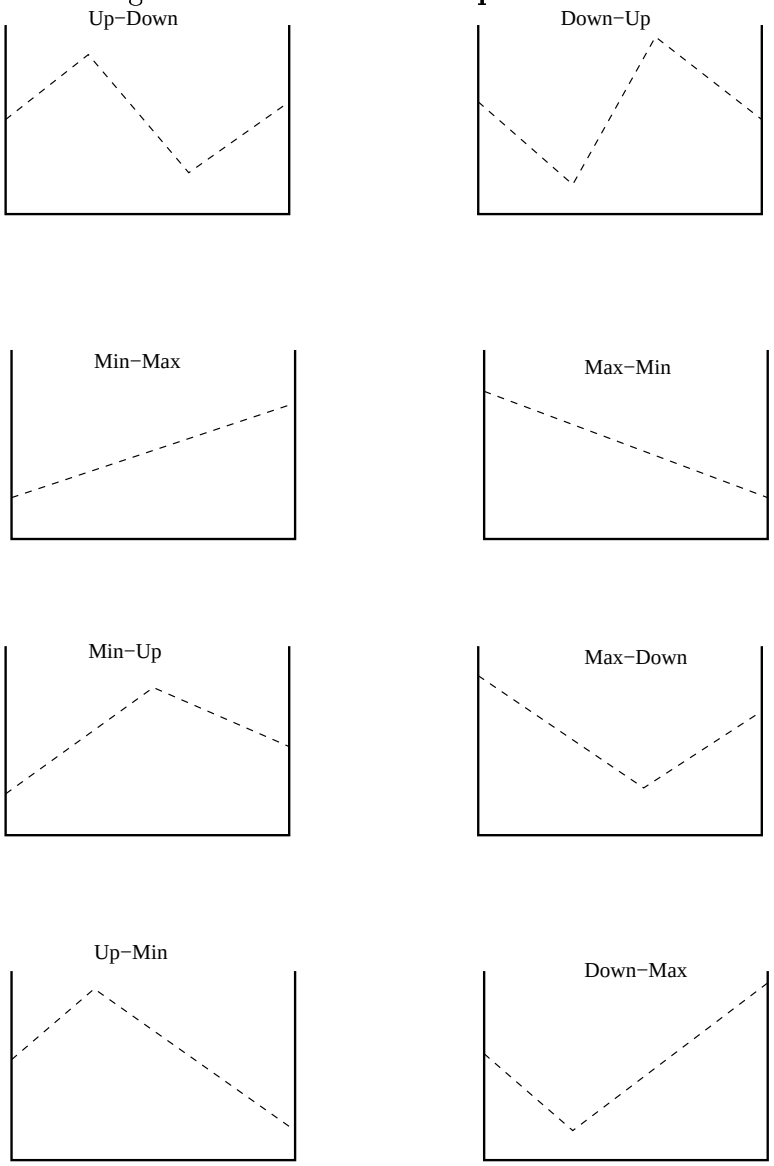


Figure 10: **Canonical Graph Patterns**



This figure depicts the eight canonical graph patterns that are used for comparing the stock graph with the sentiment graph. The graphs are based on the idea that by treating the start and end points and the maximum and minimum values as key features, we get exactly eight possible graph types.

Figure 11: **Trading Phase-based Lag**
This figure presents the results of the lead-lag analysis obtained from a pattern-matching algorithm. Lead-lags are reported in hours based on a comparison of the stock and sentiment graphs for major directional changes, based on patterns arising from the simple set in Figure 9. Positive lags indicate that the stock price leads the sentiment index; vice-versa for negative lags.