ABSTRACT

Investment institutions have sophisticated methods and software for the systematic analysis of stocks, but ordinary individuals investing on their own behalf do not have access to the same tools. We first describe the field of stock analysis and some of the methods used to predict stock movements, in particular three areas: fundamental analysis, which seeks to establish the value of a stock based on the attributes of the underlying company; technical analysis, which examines a stock’s price and volume movements; and sentiment analysis, the systematic examination of attitudes surrounding a stock. While investment institutions have spent considerable resources on developing software to automate all of these analyses, their complexity means that retail (non-institutional) investors typically lack the knowledge, time, and capital to do the same. This creates an unfair situation in the stock market, where institutions are able to more quickly identify and exploit profit opportunities, while retail investors lag behind and never realize the same levels of profit.

We then propose Phinance Analytics, a system to address the gap between institutional and retail investors. A system that performs a hybrid combination of technical, fundamental, and sentiment analysis, exposed to users through a convenient web interface, allows individuals access to some portion of the same types of tools that investment institutions have at their disposal. The system provides the user with a score for each analytical component outlined above, combined with a simple natural language description of the reasons for the score, and an overall recommendation for what the user should do with the stock. This information can be used as part of a user’s research into a stock that the user is already interested in, or to quickly identify which stocks may be interesting investment opportunities to consider. The user interacts with the system through a simple-to-navigate, intuitive website.

1. INTRODUCTION

The field of stock analysis is well explored, and many tools have been created to gain insight into the true value of a stock. Some of these methods are based on a company’s financials and management, and thus belong to the category of fundamental analysis, or the analysis of the company’s economic fundamentals. Other analysis methods focus on the price and quantity movements of the stock, and are classified under the umbrella of technical analysis. More recently, many institutions are also developing sentiment analysis tools to improve the efficiency of market research. These tools programmatically scrape and read text from the Web or other sources and determine its positivity or negativity to capture a picture of investor confidence in a stock.

To implement all of these diverse analysis tools and successfully identify profitable stocks, a large amount of time and knowledge is required. These requirements are easily met by institutional investors, such as large Wall Street firms with scores of employees and capital, but are mostly out of reach for retail investors, who are individuals investing on their own behalf. Large firms also have the means to purchase analytic tools, such as Bloomberg Terminals [9], which cost tens of thousands of dollars annually and are therefore once again not accessible to many retail investors. The large imbalance in the quality of the tools available for stock analysis between institutional and retail investors has allowed institutions to reap the profits of the economic recovery since 2009 while retail investors, those hardest hit by the recession, have often been unable to realize similar gains.

A few companies, such as E-Trade[1] and Scottrade[2] provide retail investors with charts and data to perform their own analysis and valuations. While the data itself is similar to what is available to institutions, these services leave retail investors to process and analyze this data, even though many retail investors do not have the requisite time or knowledge to adequately do so. To make sense of the data, retail investors require automated stock analysis tools employing methods similar to some of those employed by institutions.

We propose a new system, Phinance Analytics, that aggregates, analyzes and presents multiple kinds of stock analyses in a simple manner. The user has the ability to view sentiment, fundamental, and technical analysis scores for stocks individually or by entire sector from a website, www.phinanceanalytics.com. For each score that the system generates, it also presents a simple description of the reasons why the score was assigned.

Each type of analysis that the system presents is the result of an algorithm that computes a rating from 0 - 100 indicating the strength of a buy or sell recommendation (0 is a strong sell, 100 a strong buy). The technical analysis algorithm detects technical chart patterns in the recent price history, the sentiment analysis algorithm classifies tweets from Twitter according to their positivity to determine overall sentiment about a stock, and the fundamental analysis algorithm compares financial metrics of stocks to the remainder of the sector to determine whether a particular stock offers high value for its price.

---

Chris Holt  
tholtc@seas.upenn.edu  
Univ. of Pennsylvania  
Philadelphia, PA

Eugene Yarovoii  
eyarovoii@seas.upenn.edu  
Univ. of Pennsylvania  
Philadelphia, PA

Fahim Abouelfadl  
fahim@seas.upenn.edu  
Univ. of Pennsylvania  
Philadelphia, PA

---

Institutional Analytics for Retail Investors

Dept. of CIS - Senior Design 2014-2015

Advisor: Aaron Roth (aaroth@cis.upenn.edu).
The analysis provided by our system is intended to aid users in performing research on stocks that they already own or are already considering purchasing, as well as a tool to efficiently discover potentially profitable stock picks of which they were previously not aware. It is not intended for the user to trade solely based on the generated analyses without further research and deeper understanding of the causes underlying positive analysis scores. However, we do present an analysis of the advice provided by the system.

2. BACKGROUND

2.1 Fundamental Analysis

Stock analysis has existed as long as stocks themselves, but no analytic system gained widespread popularity until Benjamin Graham created a system called value investing, in the early 20th century. Outlined in his books [11] and [12], the value analysis system is based upon the principle that ownership of a stock is equivalent to ownership in the company, and thus the price movements of a stock are not as important as the underlying financial health of the company itself. Information about the finances of a company is not difficult to find: in the United States, pursuant to regulations set forth by the Securities and Exchange Commission (SEC), all publicly traded companies are required to submit publicly available financial documents, such as balance sheets, income statements, cash flow statements, and annual reports, so that shareholders can assess the financial state of the company. These documents contain critical financial information such as revenue, expenses, assets, growth forecasts, liabilities, and annual dividends. In addition to these quantitative measures, investors take into account many qualitative features such as company management, company vision, and intellectual property, all of which can affect a company’s profitability.

Graham proposed using such measures to calculate an intrinsic value, or true value, of the company, which could be divided by the total number of shares to calculate the intrinsic value of each share [12]. This value is then compared against the current stock price, and if the margin of safety, or difference between the stock price and the intrinsic value, is large enough, then the stock is a good value, and should be bought; in the long run, the market will realize the intrinsic value of the company [12].

2.2 Sentiment Analysis

The intrinsic value of a stock can be difficult to determine reliably; in part because one needs to estimate how the profits of the company will vary over time. Future growth depends on somewhat subjective and difficult-to-judge factors such as the quality of the company's management and business strategy. Sentiment analysis approaches the problem of determining what stock trades to make from a different angle. It is well-understood among investors that the attitude towards a stock can affect its price. There are several mechanisms by which this can occur: for example, attitudes could be indicative of a growing realization that the company is incorrectly valued by the market, the market could be reacting to news about the company or data recently come to light, or prices could simply be affected by the fact that positive attitudes increase demand and hence the price for the stock, while negative attitudes have the opposite effect. Sentiment may thus reflect news, investor confidence, and the general attitude towards a company in the market, so it stands to reason that analyzing sentiment about a company could be useful for stock analysis. While this type of analysis is performed naturally thousands of times daily by investors who simply react to news and analyst ratings, any given investor can only read a relatively small amount of articles in a fixed period of time. A program using natural language processing (NLP) techniques to read text and determine its positivity would be able to read many thousands of times the number of articles and make decisions based on an enormous volume of data.

In general, the goal of a sentiment analysis program is to accurately label a sentence or document with a label indicating its positivity, and thus sentiment analysis is a type of text classification task. It is typically performed by extracting some set of features from a sentence or document, and then training a classifier on some number of manually-labeled examples to distinguish between instances with different positivity. Some of the key differences in sentiment analysis techniques consist of which features are extracted for use in the classifier, the type of classifier used, and the range of possible outputs the classifier can produce. Some sentiment analysis algorithms use regression instead of classification, producing a score in a continuous range (e.g. [10]).

2.3 Technical Analysis

Another popular analytic system is technical analysis, or the prediction of stock price based on price and volume movements correlated with past trends and industry indicators. A book by Schabacker [19] is an example of early work in technical analysis; Bassetti et al. [18] presents a much more modern treatment. Murphy [17] is studied by many technical analysts today. Technical analysis follows the rule of the efficient market hypothesis, or the belief that stocks always trade at their intrinsic value, and the only way to profit on the stock market is to play the market swings, which are based on price and volume movements. As technical analysis has progressed, many types of patterns, such as head and shoulders or rectangle tops, have emerged in addition to the standard price and volume calculations [18]. These patterns help to identify possible trend reversals or continuations, and thus may be useful in identifying profit opportunities.

As technical analysis has developed, many institutions have turned towards automated short-term trading to gain large profits. These systems perform technical analysis across the market and automatically buy or sell stocks based on generated indicators. These systems can execute thousands of trades per second, which emphasizes enormous numbers of small-profit trades, rather than occasional, highly profitable trades. As these systems need to work as close to real-time as possible, they usually connect directly to the stock exchanges themselves. These types of automated trading systems currently account for two-thirds of all stock trades on major exchanges and will continue to increase as more institutions trend towards high frequency trading.

3. RELATED WORK

Technical analysis methods have been studied in depth and textbooks exist on the subject (e.g. [17]). Because some technical analysis involves looking at charts and analyzing visual patterns, some of the literature has not always described everything in rigorous, algorithmic form. A good
translation of many of the most common patterns into precise formulas and algorithms is given by Lo et al. [15], which also cites a large number of studies and publications that have provided support for the notion that detecting "visual" patterns in stock prices can lead to profitable trades. We follow the algorithmic definitions set forth by Lo et al. for the part of our technical analysis that relates to detecting chart patterns commonly used by financial analysts.

In fundamental analysis, the main question is how to determine a company’s proper valuation. The literature is vast on this subject, and many textbooks exist, with one by McKinsey & Company [7] commonly used in business schools. Comparatively few sources, however, give models that are presented algorithmically and that are easy to immediately translate into code. Blakeslee et al. [4] give one such presentation. Athanassakos [3] also gives some formulas in a compact form useful for translating into code. These sources give guidance as to the most relevant factors to consider when performing fundamental analysis; we have chosen a simplified set of factors based on some of this information.

For non-experts in the natural language processing domain to be able to more easily leverage the state-of-the-art in academic research, the Stanford Natural Language Processing Group makes available an open source NLP project with many libraries to perform NLP tasks, called Stanford CoreNLP [16]. We use several components of this library, most notably the sentiment analysis module, which is an implementation of a recently published approach by Socher et al. titled Recursive Deep Models for Semantic Compositional Over a Sentiment Treebank [20]. The approach uses the idea of parsing the sentence into a syntax tree and assigning a sentiment to each node in the syntax tree in a bottom-up recursive manner. The leaf nodes of the tree are individual tokens; the sentiment assigned to a leaf node is the general sentiment associated with the token when interpreted outside of any context (for example, "great" represents a highly positive sentiment, while "terrible" represents a highly negative sentiment). Each internal node then holds the sentiment of the portion of the sentence represented by the subtree rooted at that node. The manner in which an internal node's sentiment depends on its descendants is learned by a neural network.

The idea of using sentiment analysis specifically for the purpose of predicting financial markets, as we do in this paper, is not new. Bollen et al. [6] used sentiment analysis of Twitter posts (tweets) to evaluate market sentiment and found that certain mood indicators within tweets were correlated with the performance of the Dow Jones Industrial Average three days later, lending added credibility to the notion that sentiment analysis can have predictive power. Our sentiment analysis pipeline uses Twitter as the primary source of data. Das and Chen [8] provide another example of extracting market sentiment from text on the Web, this time from stock discussion boards, lending further credence to the idea that examining sentiment about a stock can reveal useful information. While we did not extract sentiment data from sources other than Twitter, this direction holds promise for future work.

An important part of our system also involves presenting an elegant user interface. Applications such as ours are particularly susceptible to clutter because they consist of many tools and are prone to feature creep. Since the central concept of this project is to present a web application that is accessible to ordinary investors investing on their own behalf, we should have a very strong bias towards simplicity. For this, we referred to a classic book by Krug [14] on the topic of creating simple and intuitive interfaces. Johnson [13] provides some additional useful guidance.

4. SYSTEM MODEL & USER INTERFACE

Our system is comprised of distinct modules that are each responsible for one particular type of data processing or analysis, as well as components to combine the results from the individual modules and present them together. Figure 1 consists of a diagram outlining the interaction between the different modules, and the major components are summarized below.

4.1 Data Retrieval Module

The data retrieval module uses a combination of several APIs to retrieve or scrape data and save it in a database for later use. There are a few kinds of data that are required.
by the analysis modules.

To enable fundamental analysis, detailed financial statement data by ticker symbol is retrieved from the Web and stored in a database. This includes income statements, cash flow statements, and balance sheets for each stock ticker symbol that will be analyzed by the system. Not all of the numerous fields that appear on these statements are required for fundamental analysis, but the system captures this information as completely as possible to enable re-processing in any fundamental analysis module is tweaked at a later time to incorporate more factors. Additionally, daily stock price data is also captured to compute financial ratios such as price-to-earnings. Daily stock price data includes the daily low, high, open, close, and volume of trades.

To enable richer technical analysis, the module downloads intraday stock data. This information consists of price and volume information on a fairly granular level (on the order of minutes) over the course of each day. This information can be important for generating some types of technical analysis trading signals. However, because our system is geared at presenting data to users for their own decision making and not towards automated trading, our current analysis modules make use of only the daily closing price data because such information is all that is needed to identify multi-day trends. The intraday information could generate additional signals, but these would need to be acted upon immediately rather than presented to the user for consideration.

Another component extracts recent human-generated content for sentiment analysis. This content could be extracted from anywhere where there is a significant amount of content being written by humans on a frequent, ongoing basis that expresses opinions about stocks. For each supported data source, there is a component of the module to acquire that particular data source, scraping the information from the Web in the necessary manner, downloading it via a web service, or retrieving it from a file or other source. The information obtained is stored in a database and marked with the stock ticker symbol to which it pertains and the date of the content.

More concretely, in our current implementation, we use Twitter as our source of data. As detailed further in the System Implementation section, we download tweets that discuss a particular stock and store the tweets in the database for use by the sentiment module. However, the implementation could be expanded following the framework described above to capture other sources of data such as opinions expressed on stock discussion boards and articles by analysts and writers (e.g. articles on Bloomberg).

Once the raw data is captured, it is stored in a SQL database from where it can be accessed by the analytical modules.

### 4.2 Fundamental Analysis Module

This module is responsible for performing the fundamental analysis calculations. The module depends on information that is stored in the SQL database – more specifically the balance sheets, cash flow statements, and income statements of companies downloaded by the Data Retrieval Module. Based on the financial data contained in these documents, the module evaluates the favorability of key financial ratios for the stock, both in their own right, and compared to the rest of the sector. Comparing a stock’s financial ratios and fundamental metrics to the rest of the sector is important because it is often difficult to make judgments about the favorability of financial ratios in isolation. We will see why this is so after presenting some of the techniques for fundamental analysis.

With some simplifications, the total intrinsic value of a company should be the value of all the company’s current assets, minus the company’s current liabilities, plus the sum of all the future expected discounted cash flows [7]. A discounted cash flow consists of future profits that should be multiplied by a discount factor (a value between 0 and 1) that decreases (becomes a progressively more significant discount) with how far away in the future the cash is expected to be produced by the company. Let \( V \) be the intrinsic value of a company, \( A \) denote the company’s current assets, \( L \) denote the company’s current liabilities, \( d(t) \) be the discount factor at time \( t \), \( c(t) \) be the amount of cash generated at time \( t \), and \( T \) the set of all times at which cash is generated. Then,

\[
V = A - L + \sum_{t \in T} c(t)d(t)
\]  

(1)

Subject to \( d(t_1) < d(t_2) \) if \( t_1 < t_2 \) and \( \forall t, 0 \leq d(t) \leq 1 \). Importantly, the discount factor is a strictly decreasing function of time.

A common simplifying assumption for the discount factor is that the discount factor should decrease exponentially with the amount of time that elapses from the current time forwards. It is justified in the medium-term where we do not anticipate huge economic upheavals and assume the economic climate remains fairly stable. The exponential discount represents several different factors, all of which grow exponentially over time. It accounts for inflation, the time value of money (money now is more valuable than money in the future, because money possessed now can be reinvested), and the repeated risk each year that some event will happen that will cause the company to stop producing cash. This last factor has an exponential form if we assume that a company will still produce cash some number \( K \) years later if it has produced cash all the years before that, and each year there is some probability of failure.

Once we assume an exponentially decreasing discount factor, we can rewrite Equation 1 as follows, where \( t_0 \) is the current time, at which the discount factor is taken to equal 1, and \( \alpha \) (0 ≤ \( \alpha \) < 1) is the exponential discount factor:

\[
V = A - L + \sum_{t \in T} c(t) \cdot \alpha^{t-t_0}
\]  

(2)

If we assume we are dealing exclusively with large-cap companies (i.e. those companies that have a large market capitalization, defined as the market value of all shares), these companies control large portions of their respective markets and therefore do not grow at very aggressive rates in the same manner that is possible for small companies or startups. Much of retail investing in the stock market does deal with very-large cap companies; investors are often not familiar with smaller companies, and very small companies are often not publicly traded on the stock exchange. If we make the simplifying assumption that such established companies have a constant yearly cash flow \( Y \), and we consider yearly discounted cash flows out infinitely far into the future, then the equation becomes:
A constant yearly cash flow may not seem like a reasonable assumption, but note that the equation retains the same form if $Y$ grows at some exponential rate that is smaller than the rate at which the discount factor shrinks; this alteration would simply cause an adjustment to the $\alpha$ factor. Since growth increases $\alpha$ and hence the valuation, a company exhibiting a good quarterly growth rate should receive a more favorable valuation.

In the case that $A$ and $L$ are small (the company does not have excessive assets or liabilities) in comparison to $V$, we can approximate the equation by $V = \frac{1}{1-\alpha} Y$. Letting $S$ be the number of shares and $P$ the price of a share, and $E$ the earnings per share, we can rewrite the equation as $P \cdot S = E \cdot S \cdot \frac{1}{1-\alpha}$ or $P/E = \frac{1}{1-\alpha}$ if the company is correctly valued. Based on that, we may believe the company has strong fundamentals if its actual price-to-earnings ratio is better.

Economic sectors may vary in their risk-to-reward profile. The companies in some sectors, due to the nature of how they conduct business, have cash flows that are very safe for a long period of time, whereas other sectors, such as the technology sector for example, have large companies rise and fall relatively frequently, making some of these companies less safe investments in relative terms. The choice of $\alpha$ to properly value a company closely depends on the risk involved in each company. Estimating the risk in a particular company is a difficult task and depends on many factors, but one possibility is to use the average risk for the sector under the assumption that companies that compete in a similar space sometimes have comparable risks. If the correct value of $\alpha$ is represented relatively well by the average $P/E$ ratio in a sector (that is, the average company in the sector is correctly valued), the stocks that have better $P/E$ than the sector average can be considered to have strong fundamentals.

We also take into account debt-to-equity ratio for companies. High debt-to-equity can drive a company into bankruptcy, but one possibility is to use the average risk for the sector and hence the valuation, a company exhibiting a good quarterly growth rate should receive a more favorable valuation.

$V = A - L + Y \cdot \frac{1}{1 - \alpha}$  

4.3 Technical Analysis Module

This module's role is to generate trading signals based on technical analysis. The data required for this module is the daily price data that was extracted by the Data Retrieval Module. The high-level workflow of this module is that it first extracts the necessary raw price data from the database. Then, for each day for which technical analysis has not yet been done, the system examines a sliding window of price history, and applies an algorithm to detect each type of technical signal known to the system. If any technical signals were detected, an indicator to buy or sell the stock is generated, depending on which signals were found. The algorithms we implemented for chart signal detection are the main algorithms discussed in Lo et al. [15]. While many books have been written on technical analysis (e.g. [17]), Lo et al. provides one of the clearest translations into algorithms of the criteria for identifying some of the most popular chart signals, criteria that other sources often state with a lack of precision.

Lo et al. does not discuss, however, any methods for how one should trade based on technical chart signals. Their paper limits itself to giving algorithms for detecting technical patterns and an analysis of whether historical returns are statistically independent of each of the patterns; evidence is found for some patterns in favor of rejecting the null hypothesis that daily returns are independent of the pattern. In order to implement buy/sell recommendations based on technical patterns, we extend the work in Lo et al. with a method for making trades based on detected chart signals; the Results section provides some support in favor of the approach’s effectiveness.

We first look at the technical signal detection algorithms. Pursuant to the method of Lo et al., technical signals can be seen as sequences of extrema (local maxima and minima) in price data that satisfy certain conditions. As such, a technical signal occurs over many days, and what is really being detected is the culmination of the signal; that is, the occurrence of the last extremum necessary to meet the conditions of a particular technical signal. Once the technical pattern is complete, price movements are expected to follow depending on which pattern was observed. Therefore, the logic to detect technical signals in a given stock’s prices is to examine, for each date, whether it is a local extremum, and if so, whether it completes a sequence of extrema that constitutes a technical signal.

One choice that has to be made is the length of price history that must be used. We have chosen to look at sliding windows of 50 days because many long-term technical signals can require a timeframe of this duration to develop. However, making the window longer than this results in many spurious signals being detected. Lo et al. discusses the issue of window size in depth, though they do not use the same value for this parameter. Our parameter of 50 days should be similar in intent to Lo et al.’s 38 days, however, because their analysis only used data from days where the stock exchange was open. In our rough estimation, 38 trading days is approximately equal to 50 total days. The analysis is not highly sensitive to small changes in this parameter, hence we allowed ourselves some rounding. The start of the window is 46 days before the given date; the end of the window is 3 days after the given date. These 3 days are to allow time for the detection date to be properly smoothed and ensure that no false extrema are detected; this also means that a
smoothed price

s
vided by. Otherwise, there could be a small bias to the price
cause it is being applied on a discrete sample, the weights in
The Gaussian function given above is normalized, but be-
cause it is being applied on a discrete sample, the weights in
Using the Gaussian shown above as a convolution, the
smoothed price \( s(t) \) for a time \( t \) has the following values,
where \( w(i, t) \) is the weight of the raw price at time \( i \) in
the calculation of the smoothed price for time \( t \):

\[
w(i, t) = e^{-\frac{(i-t)^2}{2\sigma^2}} \tag{6}
\]

4.4 Sentiment Analysis Module
This module employs sentiment analysis to detect the
sentiment expressed about a stock. It uses the human-
generated content captured by the data retrieval module,
which is already tagged with information about the stock
symbol and date to which the content pertains. For a given
stock and date on which the sentiment module gauges the
sentiment, the relevant data is retrieved from the database
and passed into the sentiment analysis algorithm.
This logic assumes that sentiment is short-term; more
specifically, the exact assumption is that the best estimate of
the sentiment pertaining to a stock on a given date is the
sentiment expressed by content on the same date, with-
out taking content from past dates into account. The intu-
ition behind this assumption is first that the market reacts
fairly quickly to news, and second that if there have been
recent events that have positively or negatively affected the
sentiment about a stock, the sentiment at the present time
should already capture that information to the extent that
such information continues to be relevant; there is therefore
no need to consider content from earlier dates to determine
signals that they did, we use only the following ones to make
recommendations, because the other signals did not generate
profitable trades during back-testing:
1. Head and shoulders - Sell recommendation
2. Inverted head and shoulders - Buy recommendation
3. Double top - Sell recommendation
4. Double bottom - Buy recommendation
5. Broadening Top - Sell recommendation
6. Broadening Bottom - Buy recommendation
7. Triangle Top - Buy recommendation
8. Triangle Bottom - Sell recommendation

We added sell or buy recommendations to the above sig-
nals in accordance with the technical analysis textbooks
cited in Related Work.

Once the technical signals are detected, the technical score is
computed as follows. The base score starts at 50. It is then
modified by 25 multiplied by a signal multiplier (be-
 tween 0 and 1) and 25 multiplied by a recent performance
rating (between 0 and 1), giving a possible minimum of 0
and maximum of 100. There are 3 cases to consider:
1. If a signal is detected that day, it is automatically given
a 25 point boost for a buy signal or a 25 point decline
for a sell signal.
2. If there is no signal that day, but there was a signal
within the last 5 trading periods, the base score is mod-
ified by +/- 25 \( \cdot (6 - d) \) where the sign depends on
whether the signal is a buy signal or a sell signal, and
\( d \) is the number of trading periods since the last signal.
The score is then modified by a further 25 multiplied
by the percent change in price from the last signal.
3. If there was no signal that day, and no recent signal,
the base score is modified by 25 multiplied percent
change in price from 10 trading periods earlier.

Lo et al. discuss, before detecting extrema and scan-
ning the extrema for signals, it is important to first smooth
the data to remove local noise. Since stock prices almost
always contain small, short-term price fluctuations which
often have the appearance of random walks, many highly
local extrema will be detected if the data is not smoothed.
These extrema are in some sense "false"; they do not reflect
long-term trends. It is therefore important to smooth the
data substantially so that any extrema still remaining after
smoothing accurately represent the culmination of longer-
term price trends.
The smooth price at time \( t \) is computed by taking a weighted
average of the raw prices in a neighborhood close to \( t \). That
is, we can conceptualize that the raw prices between time
\( t - \delta \) and \( t + \delta \) for some reasonable \( \delta \) will be part of the
calculation for the smooth price at time \( t \). Following Lo
et al. we choose a Gaussian convolution for the weights in
the weighted average calculation. The Gaussian function is
given by the equation:

\[
G(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \tag{4}
\]

where \( \sigma \) is the standard deviation of the Gaussian, and \( x \)
is the distance from the mean of zero. The choice of \( \sigma \) depends
on the amount of smoothing one wishes to perform, which
in turn depends on the noisiness of the data. We discuss our
choice of smoothing in the System Implementation.
The Gaussian function given above is normalized, but be-
cause it is being applied on a discrete sample, the weights in
the weighted average calculation should be summed and di-
vided by. Otherwise, there could be a small bias to the price
smoothing that would adjust all prices slightly upwards or
downwards; this is clearly undesirable because it gives a false
perception of a stock’s value.
Using the Gaussian shown above as a convolution, the
smoothed price \( s(t) \) for a time \( t \) has the following values,
letting the raw (unsmoothed) prices with respect to \( t \) be
represented by \( r(t) \):

\[
s(t) = \frac{\sum_{i=0}^{i=N} w(i, t) \cdot r(i)}{\sum_{i=0}^{i=N} w(i, t)} \tag{5}
\]

\[
G(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \tag{4}
\]

where \( \sigma \) is the standard deviation of the Gaussian, and \( x \)
is the distance from the mean of zero. The choice of \( \sigma \) depends
on the amount of smoothing one wishes to perform, which
in turn depends on the noisiness of the data. We discuss our
choice of smoothing in the System Implementation.
The Gaussian function given above is normalized, but be-
cause it is being applied on a discrete sample, the weights in
the weighted average calculation should be summed and di-
vided by. Otherwise, there could be a small bias to the price
smoothing that would adjust all prices slightly upwards or
downwards; this is clearly undesirable because it gives a false
perception of a stock’s value.
Using the Gaussian shown above as a convolution, the
smoothed price \( s(t) \) for a time \( t \) has the following values,
letting the raw (unsmoothed) prices with respect to \( t \) be
represented by \( r(t) \):

\[
s(t) = \frac{\sum_{i=0}^{i=N} w(i, t) \cdot r(i)}{\sum_{i=0}^{i=N} w(i, t)} \tag{5}
\]

\[
w(i, t) = e^{-\frac{(i-t)^2}{2\sigma^2}} \tag{6}
\]

4.4 Sentiment Analysis Module
This module employs sentiment analysis to detect the
sentiment expressed about a stock. It uses the human-
generated content captured by the data retrieval module,
which is already tagged with information about the stock
symbol and date to which the content pertains. For a given
stock and date on which the sentiment module gauges the
sentiment, the relevant data is retrieved from the database
and passed into the sentiment analysis algorithm.
This logic assumes that sentiment is short-term; more
specifically, the exact assumption is that the best estimate of
the sentiment pertaining to a stock on a given date is the
sentiment expressed by content on the same date, with-
out taking content from past dates into account. The intu-
ition behind this assumption is first that the market reacts
fairly quickly to news, and second that if there have been
recent events that have positively or negatively affected the
sentiment about a stock, the sentiment at the present time
should already capture that information to the extent that
such information continues to be relevant; there is therefore
no need to consider content from earlier dates to determine
signals that they did, we use only the following ones to make
recommendations, because the other signals did not generate
profitable trades during back-testing:
1. Head and shoulders - Sell recommendation
2. Inverted head and shoulders - Buy recommendation
3. Double top - Sell recommendation
4. Double bottom - Buy recommendation
5. Broadening Top - Sell recommendation
6. Broadening Bottom - Buy recommendation
7. Triangle Top - Buy recommendation
8. Triangle Bottom - Sell recommendation

We added sell or buy recommendations to the above sig-
nals in accordance with the technical analysis textbooks
cited in Related Work.

Once the technical signals are detected, the technical score is
computed as follows. The base score starts at 50. It is then
modified by 25 multiplied by a signal multiplier (be-
 tween 0 and 1) and 25 multiplied by a recent performance
rating (between 0 and 1), giving a possible minimum of 0
and maximum of 100. There are 3 cases to consider:
1. If a signal is detected that day, it is automatically given
a 25 point boost for a buy signal or a 25 point decline
for a sell signal.
2. If there is no signal that day, but there was a signal
within the last 5 trading periods, the base score is mod-
ified by +/- 25 \( \cdot (6 - d) \) where the sign depends on
whether the signal is a buy signal or a sell signal, and
\( d \) is the number of trading periods since the last signal.
The score is then modified by a further 25 multiplied
by the percent change in price from the last signal.
3. If there was no signal that day, and no recent signal,
the base score is modified by 25 multiplied percent
change in price from 10 trading periods earlier.
the sentiment. This assumption seems to be particularly appropriate to content that is generated with high frequency, such as the Twitter data that we used in our implementation. While most or all sentiment is short-term, it could make sense in future work to consider data from past dates to some extent, for example for articles that are updated once every few days.

The sentiment analysis algorithm is designed to have three parts. First, the sentiment evaluator lexes, splits the data into sentences, and determines the sentiment of each sentence. This component can have any implementation, as long as it is capable of splitting a file of text into sentences and extracting the sentiment of each sentence. In our case, we use the algorithms implemented in the Stanford CoreNLP library [16] to carry out these tasks, as Stanford CoreNLP contains implementations of many natural language processing algorithms that have very competitive performance. For its sentiment classification module, Stanford CoreNLP uses a classifier due to Socher et al. [20]. The approach uses the idea of parsing the sentence into a syntax tree and assigning a sentiment to each node in the syntax tree in a bottom-up recursive manner, inferring how to combine subtree sentiments with a neural network.

The second portion of the sentiment analysis algorithm consists of a sentiment aggregator that combines the sentiment results from all the sentences that have been classified by the sentiment evaluator into an overall sentiment score for the stock symbol and date pair. This task is accomplished by taking a weighted combination of the sentence scores:

$$ V = \frac{\sum_{s \in S} w(s)v(s)}{\sum_{s \in S} w(s)} $$

Where $V$ is the overall score, $S$ is the set of all sentences pertaining to the same stock and date, $w(s)$ is the weight of a given sentence $s$, and $v(s)$ is the sentiment value of a sentence $s$. The sentiment evaluator returns a sentiment classification label (for example, StanfordNLP returns a sentiment in the set "VeryNegative", "Negative", "Neutral", "Positive", "VeryPositive"). The sentiment aggregator would convert such scores to a 0-100 scale to match the range used by all other analysis modules; the exact values to use for this conversion depend on the sentiment evaluator. For the sentence weights, one possibility is to use a uniform weight of 1 for each sentence. Another possibility is to use a weighting scheme that assigns larger weights to longer sentences, under the assumption that they are likely more significant. The best weighting and label-to-score conversion formulas depend on the data and the properties of the sentiment evaluator component; we discuss our implementation based on our experience with Stanford CoreNLP in the System Implementation section.

During its execution, the sentiment aggregator also calculates some additional information, such as the variance and histogram of the sentence sentiments it encountered. This information is stored together with the overall score.

The third and final component of the module, the description generator, is responsible for generating natural-language descriptions for the ratings, based on the information obtained from the sentiment aggregator. This component uses a rule-based system to fill in natural-language sentence templates with values from the analysis. This information serves to permit the user to make better use of the data as a result of having more detailed information. For example, when sentiment is positive for a stock, the user may wish to know whether it is uniformly positive, or whether there is a large amount of positive sentiment mixed with some negative sentiment. The former scenario may be the result of a general positive perception of the company; the latter could be the result of a controversy where the company appears on average in a positive light.

Here is an example of a description generated by the description generator: "The general opinion about this stock is neutral (52/100 sentiment rating) with a moderate standard deviation (20 rating points). The overall sentiment distribution was Positive: 12 %, Neutral: 87 %, Negative: 1 %".

Once the sentiment scores and overall descriptions have been generated, they are stored in the SQL database from which the website backend loads its data for display to users.

### 4.5 Summarizer

The Fundamental Analysis Module, the Technical Analysis Module, and the Sentiment Extraction Module all store scores on a 0 to 100 scale indicating whether the stock is to be sold or bought. Once all three modules have run, the summarizer is responsible for determining which of these scores and trading signals is relevant to the user. Currently, this is done simply by running a database script to average the three ratings and link the data from each module to the overall score, which is then stored in the appropriate database table to allow the score to appear on the website. An interesting direction for future work would be to weigh the analyses based on the preferences of the user. This way, users who do not believe or do not wish to consider certain kinds of analyses could exclude those analyses from consideration when receiving an overall stock recommendation, or users could have the website weigh their preferred type of analysis more heavily.

### 4.6 Website

The website hosts a web application that presents all of the relevant data for viewing by the user. It is the only way in which the user interacts with our system, since the system is intended for lay users. Through the website, the user can search for stocks and view their analyses, accessing the information generated by the above modules.

## 5. System Implementation

### 5.1 Languages and Technologies

For writing the code for the modules, the data retrieval, fundamental analysis, and technical analysis modules are implemented in Python 2.7 to take advantage of its dynamically-typed nature and ability to easily run scripts. Although Python 3 has been released, many external libraries have not been updated to use Python 3 yet, so Python 2.7 was chosen due to compatibility considerations. Additionally, creating connections from Python code to MySQL is made simple through the python-mysqldb library, allowing us to store any data we require. Integrating many of the other APIs we wish to use, such as the Twitter API to obtain our data for sentiment analysis, also proved easy with Python since Python wrappers already existed. The relatively lower computational speed of Python in comparison to compiled
l languages like C is not currently a concern because our runtime is dominated by the time taken to load data from, and store data to, the database. If the databases were distributed and scaled out, this operation would become slower still in comparison to local calculations.

The sentiment analysis module was implemented in Java because Stanford CoreNLP is written as a Java API. It was simplest to use the same language to interface with it. Since our top-level scripts that run all the modules for each day are in Python, a script was created to furnish the inputs to our sentiment analysis module as files, from where the sentiment analysis module picks up the files, processes them, and then writes the outputs to another set of files from where the top-level scripts read the ratings.

5.2 Data Storage

We chose to store our data in a MySQL database. MySQL is a simple database solution that has the advantages of being readily understood by developers and achieving a decent tradeoff between expressiveness and performance. The main disadvantage of MySQL is that it involves all the data being stored on a single machine. This sufficed for our purposes; however, we did not scale the system to the full range of stocks available on stock exchanges throughout the world.

If the system needed to scale further, it would not be too difficult to implement a system where the data is sharded between multiple machines (and therefore databases). The sharding could be done based on any number of criteria, with sharding based on the stock symbol being simplest, since data for each individual stock is usually examined separately by our system (and combined by the web application in the end).

We chose MySQL Community Server, because it has the advantage of providing easy server management through MySQL Workbench. MySQL Community Server makes it simple to update a database in real-time and then obtain dumps of the schema, so that the final database schema used in our testing can then be captured and checked in to the source depot to become the database schema used for installing the project from scratch.

5.3 Fundamental Analysis

At first we used the Yahoo! Query Language (YQL) API to retrieve detailed quarterly financial information as published by publicly traded companies. We used the yahoofinance, balancesheet, yahoofinance.incomestatement, and yahoofinance.cashflow tables to obtain financial statements. For price data, we used the yahoofinance.historicaldata (long-term price data) and yahoofinance.quotes (shorter-term price data) tables. While we continue to use quote and historical price data from YQL, we experienced a problem with the financial statements when YQL suddenly failed to continue to supply that data. Upon closer examination, it was determined that this occurred because, surprisingly, the API operated by scraping a Yahoo webpage containing the information. To rectify the problem, we implemented our own scraper capable of parsing a portion of the same data. This type of problem is somewhat representative of the difficulty of finding reliable and free-of-charge APIs to retrieve financial data.

While we implemented tools to scrape some of the same data, the difference in data formats has led to our implementation of the fundamental analysis algorithm described under System Model to be currently incomplete.

5.4 Technical Analysis

The price data for technical analysis is retrieved from the YQL API described above. The only data required for the currently implemented indicators is end-of-day price data over a long period of time. While the technical analysis module examines signals over a relatively short timeframe of 50 days, data from 2007 onwards was downloaded to adequately backtest the system.

We implemented the algorithm described for technical analysis in the System Model section and generate technical analysis scores based on it, as discussed in the Results section.

To determine the best smoothing factor to use for the data, we back-tested the analysis for a variety of potential smoothing factors. In general, we found that a smoothing factor of 1 produces reasonable results. Lo et al. [15] recommends using a method called cross-validation to determine the best smoothing factor. This method involves finding the value of $\sigma$ that would result in the least squared error when the smoothed observations are used to predict each price based on the weighted averages of the neighboring prices, in accordance with the Gaussian weighed averages. Lo et al. reported that this method produced curves that were too smooth for them, and they multiplied the result of the cross-validation by an arbitrary scaling factor to ensure data retained sufficient granularity. We found that on the contrary, cross-validation resulted in very little smoothing. In our final analysis of the results, we consider a variety of possible smoothing factors. A larger smoothing factor represents in some sense the extent to which the user wants to make more long-term trades and detect more long-term signals. The more smoothing is applied, the fewer extrema the smoothed data has, and the more the detected patterns represent long-term trends.

5.5 Sentiment Analysis

Our implementation uses Twitter as the source of sentiment data. We reason that Twitter should be effective in representing sentiment because it has such high traffic and users are constantly engaged in sharing their opinions on the platform. Twitter also has a fairly feature-rich and easy-to-use API for retrieving Twitter posts (tweets). For most other sources of news or opinions, the data would need to be scraped from web pages, which is a far more fragile implementation.

The portion of the data retrieval module that provides Twitter search capability can issue arbitrary queries to the Twitter API. When searching for tweets related to stocks, we issue queries of the form "$STOCKSYMBOL". For example, the query "$AAPL" would return all tweets with the "AAPL" tag. On Twitter, this is the equivalent of a "hash-tag" for stock ticker symbols; investors who frequently discuss stocks on Twitter would know to use such symbols in their tweets.

Tweets from more reliable and well-established sources are a better indicator of public opinion considering the average person tends to base their views on people of authority. The module inherently takes this into account through the collection of retweets when collecting the raw data. A tweet that has been retweeted several times will contribute more towards the overall sentiment score in comparison to a lone
Once the raw data has been obtained, automated data cleaning is performed on the tweets to remove portions that are difficult for sentiment analysis tools to comprehend. Hashtags (e.g., "#mystockpick"), stock hash tags (e.g., "$AAPL"), content containing long lists of stock ticker symbols (e.g., "$MSFT, $AAPL, $GOOG, $AMZN, $INTC"), URLs, and tags that indicate a tweet is in reply to a user (e.g., "@UserName") are removed. Sentiment analysis is still not considered to be a fully solved problem, and even advanced sentiment analysis algorithms such as the one supplied by Stanford CoreNLP have significant error rates even on well-formed, concise sentences. The tendency of Twitter posts to contain spam, tags, and URLs, sometimes in seemingly haphazard places, make it very difficult for just about any sentiment analysis algorithm to operate properly; even obtaining each sentence’s parse tree, a key preliminary step of Stanford CoreNLP’s sentiment analysis, may present considerable difficulty if the data is not cleaned.

In terms of the best weighting scheme to use in the sentiment aggregation component, after some testing of our sentiment analysis module on small pieces of sample text (such as a few Yelp reviews for example), we used the formula $w(s) = \sqrt{L(s)}$ where $L(s)$ is the length of a sentence $s$ in characters. The intuition behind this formula is that it provides a middle ground between uniform weighing of sentences, which assigns too much weight to short, sometimes one-word fragments or sentences that often contain no meaningful sentiment value, and weighing each sentence proportionally to its length, which causes small sentences to carry very little weight. The square-root formula was chosen before any significant back-testing on the stock sentiment data, and not modified afterwards. The approach is admittedly somewhat ad-hoc, since there are other possible functions whose growth rate is strictly between $\Theta(1)$ and $\Theta(n)$, and refining the choice of weighing scheme is a possible topic for future work.

5.6 Website

The website, www.phinanceanalytics.com, is optimized for both mobile and desktop platforms to ensure a good viewing experience on a variety of devices.

The user can interact with the website in a number of different ways. The first way for the user to interact with the website is to search for a specific stock symbol and receive a report containing each sentence’s parse tree, a key preliminary step of Stanford CoreNLP have significant error rates even on well-formed, concise sentences. The tendency of Twitter posts to contain spam, tags, and URLs, sometimes in seemingly haphazard places, make it very difficult for just about any sentiment analysis algorithm to operate properly; even obtaining each sentence’s parse tree, a key preliminary step of Stanford CoreNLP’s sentiment analysis, may present considerable difficulty if the data is not cleaned.

In terms of the best weighting scheme to use in the sentiment aggregation component, after some testing of our sentiment analysis module on small pieces of sample text (such as a few Yelp reviews for example), we used the formula $w(s) = \sqrt{L(s)}$ where $L(s)$ is the length of a sentence $s$ in characters. The intuition behind this formula is that it provides a middle ground between uniform weighing of sentences, which assigns too much weight to short, sometimes one-word fragments or sentences that often contain no meaningful sentiment value, and weighing each sentence proportionally to its length, which causes small sentences to carry very little weight. The square-root formula was chosen before any significant back-testing on the stock sentiment data, and not modified afterwards. The approach is admittedly somewhat ad-hoc, since there are other possible functions whose growth rate is strictly between $\Theta(1)$ and $\Theta(n)$, and refining the choice of weighing scheme is a possible topic for future work.

The second way for the user to interact with the website is to choose a specific form of analysis out of the three analyses conducted and view the summaries and scores pertaining to that form of analysis alone. Figure 3 shows an example of such an interaction through a screenshot of the Sentiment Analysis Page. In this view, the user can see the sentiment score associated with several companies’ stocks. Upon clicking on a specific company, the website will present the user with a graph of the company’s score in the selected form of analysis as a function of time, along with a summary that explains why the company received the calculated score.
to go to the Trading Signals Page and select a sector to be analyzed. The purpose of having this mode of view available to the user is to provide the user with the top performing stocks according to the system’s metrics within the given sector. Figure 4 demonstrates this type of interaction using scores calculated for the technology sector. If the user is interested in investing in the technology sector, this view provides a good starting point, as the user can compile a short list of the top performing stocks within the sector and conduct further research on those specific companies.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Recommendation</th>
<th>Sentiment Score</th>
<th>Fundamental Score</th>
<th>Technical Score</th>
<th>Signal Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAPL</td>
<td>Buy</td>
<td>50</td>
<td>23</td>
<td>52</td>
<td>2009-04-29</td>
</tr>
<tr>
<td>NVIDIA</td>
<td>Buy</td>
<td>90</td>
<td>41</td>
<td>44</td>
<td>2009-04-29</td>
</tr>
<tr>
<td>BP</td>
<td>Sell</td>
<td>48</td>
<td>37</td>
<td>46</td>
<td>2010-01-01</td>
</tr>
<tr>
<td>CNI</td>
<td>Buy</td>
<td>40</td>
<td>27</td>
<td>81</td>
<td>2010-01-01</td>
</tr>
<tr>
<td>AIG</td>
<td>Hold</td>
<td>45</td>
<td>54</td>
<td>62</td>
<td>2009-04-29</td>
</tr>
<tr>
<td>MSFT</td>
<td>Hold</td>
<td>50</td>
<td>73</td>
<td>48</td>
<td>2010-01-01</td>
</tr>
</tbody>
</table>

Figure 4: Technology Sector

The website is hosted on an Amazon AWS EC2 instance, and pulls the data from a MySQL database hosted on an Amazon AWS RDS instance. It is structured as a modular design based on two separate Maven projects, one for our database access, and one for our web access. The database project, written in Java and encased in the Spring Framework, houses the code related to data insertion, retrieval, or modification, using Hibernate for object-relational mapping. This allows for a very flexible structure enabling the database layer to change, while preserving a public interface that allows the web project to query data without needing to know how that data is retrieved. The web layer houses all code necessary to process web requests and retrieve the appropriate data through the public interfaces provided by the database layer. The web layer, also written in Java and wrapped in the Spring Framework, allows for simple authentication and authorization using the Spring Security Framework. In this manner, we have the ability to implement customizations of the web experience to each user’s preferences and saved settings. Additionally, this two-tiered structure allows us to change the front-end display of the website, saving time and increasing availability. For deployment, the web layer is compiled into a WAR file, and deployed to a Wildfly 8.2 server instance running on EC2, allowing public access to the website.

6. RESULTS

To test the profitability of our individual modules, as well as our overall recommendations, we implemented a simple buy/sell strategy. We buy one share upon our algorithms indicating that a trade is profitable, and then sell that stock on the next day that our algorithms indicate that the stock is no longer profitable. If we do not have a stock to sell, the sell order is ignored; this simulates the buying habits of casual retail investors who mostly buy and sell stocks and do not short sell or execute more complex trades as often. The results for our sentiment analysis and fundamental analysis were inconclusive due to several factors. Due to limitations with the Twitter Search API, we were able to get tweets only for a relatively short range of time. Since users cannot buy or sell over the course of a weekend, both buy and sell dates would need to be exclusive of weekend days, which reduced the number of buy and sell transactions recommended by the system. Finally, despite all the data cleansing we performed on tweets, the initial quality of tweet content was very low. It would seem that, perhaps unlike in the days of older work such as Bollen et al. [6], Twitter has become increasingly full of spam. Inspection of a sample of the tweets downloaded that contained stock ticker hashtags such as "$AAPL" shows that many of the messages are spam, generic messages promoting many stocks, and in some cases are messages only weakly related to the stock. More sophisticated methods of filtering tweets to eliminate spam messages may be required in order to effectively leverage Twitter for sentiment analysis. Another approach would be to leverage sentiment sources other than Twitter.

For the technical analysis, we back tested our algorithm on data from Jan 1, 2007 until Apr 28, 2015 in order to determine the performance of our modules. Due to API limitations and recent IPO dates, GOOG data runs from Mar 27, 2014, FB from May 18, 2012, and LNKD from May 19, 2011.

The results for our chart pattern detector analysis are dependent upon the smoothing factor used, and as such, we show the top ten most profitable smoothing factors per stock in Table 1. Not all stocks have ten profitable smoothing factors, and some do not have ten smoothing factors that lead to any trading. Focusing on the top few best performing smoothing factors per stock, the returns are very high across the board. To show the performance in an economic downturn, Table 2 displays the data from Table 1 that occurs between Jan 1, 2007, and Dec 31, 2009, which spans the official timing of the Great Recession. Our returns are still very strong against the economic downturn, and far exceed the market returns during the same period. Another important factor to consider is the profit per trade, due to the high trading fees charged by outlets available to retail investors. With every stock, we can find at least one very manageable profit per trade ratio that would enable even modest retail investors to turn a profit in spite of the trading fees. To illustrate this example, Tables 3 and 4 show our same trading algorithm, but buying 10 shares at once, instead of only a single share, in the recession, and overall, respectively. With a budget of under $1,500, well within the means of a retail investor, it is possible to create profits in excess of the standard $7-8 trading fees most retail outlets will offer. Additionally, the best few smoothing values per stock also have tremendous profitable trade ratios. Looking at solely the chart pattern performance, independent of stock, Tables 5 and 6 show the performance of each chart pattern overall, and in the recession, respectively. Every chart pattern we detect, except for triangle tops in the recession, generates profitable trades. All together, these results present a strong case that our chart detection algorithms, although simple, are effective at generating profits.

Because some of the referenced tables are large and include data for each of 25 stocks that were analyzed, we have published the data for Table N at www.phinanceanalytics.com/data/tableN.
7. FUTURE WORK

The sentiment extraction module can be improved and extended in three main ways. Although the Stanford CoreNLP API is a state-of-the-art tool used to extract and examine sentiment, it has one large drawback that stems from the fact that the training data used is not based on reviews of stocks. This training data results in the API being particularly effective at determining the sentiment related to media, and arts and culture. Due to the nature of linguistics, certain words and combinations of words can have a positive meaning in one context, and a negative meaning in another context. In order to ensure that the sentiment extraction is as accurate and reliable as possible for inputs pertaining to stocks, it would be useful to feed a set of stock-related inputs as the training data to the CoreNLP API. It can be important to train sentiment analysis algorithms on training sets containing words from the domain, because otherwise words and phrases like “bear market”, “bullish”, or “rally” might be judged as neutral by sentiment classifiers that have not encountered them in the training corpus.

Furthermore, this module could be extended to determine which entities are the object of the sentiment when reading sentences, in order to produce multifaceted sentiment analysis that can distinguish between different aspects of the sentiment expressed towards a stock. In other words, the module could be used to gauge sentiment about a company’s management, revenues, profits, products, and more. Google Product Search employs this type of sentiment analysis. For example, it shows what reviewers have said about a camera’s lens, resolution, ease of use, etc.

Blair-Goldensohn et al. [5] discuss a sentiment analysis system for product reviews with similar capabilities. Such an extension would greatly strengthen the user’s ability to conduct further research on specific aspects of a company, enabling them to make better informed investment decisions.

Moreover, the sentiment extraction module relies on tweets as a source of input, but does not handle other forms of input such as news articles, journal articles, blog posts, and other mediums. As discussed in the Results section, while Tweets may provide some indication of the public’s sentiment towards a specific stock, there was generally too much spam on Twitter to produce reliable sentiment analysis results that would result in profits. Capturing data from higher-quality sources would mitigate much of the problem.

In addition to improving the analytical modules associated with the system, the user experience on the website can be further enhanced in several ways. Users already have the ability to create accounts on the website. This customization by user can be furthered by creating a page that provides the user with stocks they may be interested in based on their investment or viewing history. One way this could be implemented is to provide users with well-performing stocks that have been viewed by users with a similar viewing or investment history. Another possible direction for improving the website would be to provide more detailed plots in some sections. For example, the technical analysis page could show users where technical indicators occurred on a stock price graph. A page could also be added that would show all the analysis scores on one graph; this may be useful for comparing and contrasting analysis methods.

Finally, the platform as a whole can be extended to take full advantage of the technical analysis conducted. As it stands, the platform provides users with a tool that helps users make investment decisions on a day to day basis. The true potential of investing based on technical analysis is unlocked when the user partakes in automated trading. The system currently has the ability to conduct technical analysis that could in principle be used for automated trading by collecting intra-day price and volume values for stocks.

Connecting this website with an automated trading platform would make the website more useful and valuable, but would definitely raise ethical issues pertaining to financial loss due to too much dependence on a single source of investment information.

8. ETHICS

Since the website is intended to help inform users’ financial investments, the main ethical issue that arises is the potential to harm users financially should the recommendations be poor or the analyses unreliable. It is worth noting that the website does not allow the user to invest directly through the platform, nor do we encourage investors to base their decisions entirely on the generated indicators. We have attempted to make clear to users that Phinance Analytics is meant to merely serve as a starting point for research into potential investments, and all trading signals and scores should be received by the user with a certain degree of skepticism, as there may be several conflicting views over the future movement of a stock price.

The purpose of the website is to help users narrow down their investment options and find a small number of companies to further investigate, rather than being burdened by the abundance of noise and excess information available on the web. There are often thousands of companies in a sector. While large investment institutions will devote large amounts of manpower to analyzing all of the notable companies in a sector, the average person investing on their own behalf cannot reasonably attempt to do the same. As such, Phinance Analytics generates trading signals to inform a user about where interesting profit opportunities may lie; further research and investigation by the user is expected in order to determine whether the opportunity is in fact as good as it appears from the available metrics.

Experienced investors are aware that no investment strategy, platform, or fund manager can ever be impervious to loss, and they act accordingly by diversifying their portfolios and conducting ample research before reaching an investment decision. The true ethical dilemma thus lies with the inexperienced investor who views the platform as a “get rich quick” scheme, or as a shortcut to investment. Such an investor would probably begin by making small investments that are consistent with the recommendations provided by the website. If these investments prove to be profitable, the investor might incorrectly assume that they have discovered a tool that is guaranteed to turn a profit, and would therefore make more bold investments that could result in large losses. In order to avoid such an ethical issue and to ensure the financial well-being of the users, users should frequently be encouraged to conduct further research, and the website can connect them to other platforms that can help them in this endeavor. Another way to help combat this issue is to improve upon the summary and explanation aspect of the platform to ensure that the user is not taking the numbers and results at face value, but is instead using the tool in the intended manner, as an educational and research tool that can help them quickly discover potentially profitable stock picks and find.
metrics on stocks they already hold or are already considering purchasing.

A second ethical issue that arises pertains to the sentiment extraction module. The module relies on the analysis of tweets, which are meant to provide a sample for the general population’s opinion on a certain company or stock. It is possible for the tweets and input data to be manipulated through the use of a tweet bot, or spam tweets that are intended to lead people to believe that the sentiment associated with the company or stock is more positive than the reality. One way to mitigate such issues is to extend the sentiment extraction module in the manner described in the Future Works section; collect input data from outside sources in addition to Twitter, and provide a breakdown of the sentiment score according to the aspects of the company (management, products, etc) that are viewed positively or negatively.

9. CONCLUSIONS

Phinance Analytics is a tool that is intended to empower the retail investor by providing them with user-friendly analyses that are easy to understand and that require minimal time, effort, and technical knowledge on the part of the user. The platform is geared towards automating much of the data collection, analysis, and aggregation that would otherwise be done manually. Despite the platform’s attempt to be as comprehensive as possible, it is not meant to be used as the sole investment research tool by a retail investor, but rather as a solid first step that can expose retail investors to the most profitable investment opportunities possible. If used properly, the tool can reveal profitable opportunities for its user base, while further research and due diligence can help mitigate any losses that may result due to an overly optimistic analysis by the system.

10. REFERENCES