Financial Analysis Using a Distributed System

By:

Douglas Brandt

Senior Project

Computer Engineering Department,

California Polytechnic State University, San Luis Obispo

June 2012

Abstract

In recent years, the amount of data being generated has caused a big data revolution. Suddenly, companies are finding new ways to gather and analyze massive amounts of data. One of the interesting applications of this new big data analysis is the application to finance. Trillions of dollars are traded in the market each day. Analyzing and trying to profit from these trades is of major concern to hedge funds, pension funds, banks, and many high net worth individuals. This paper analyzes ways to try to outperform the market, while also setting up an infrastructure which will allow future development at a much reduced cost and effort. Beating the market is no easy task and in the end, all three strategies that were attempted underperformed the market by more than 50%, clearly indicating no arbitrage opportunities.

Contents

List of Figures and Tables

Acknowledgements

I would like to thank my advisor, Dr. John Clements, for all of his advice and suggestions throughout this project. With his keen interest in both computer science and finance, it has been a pleasure working with him.

I would also like to thank my family, Mom and Matt, for all of their love, support and encouragement in pursuing all of my goals in life.

Introduction

Today's highly competitive and software-driven black boxes are the new way that Wall Street operates. The data generated on any given day, across all publicly-traded stocks can be upwards of one terabyte of data (Rogers). This massive amount of data must be analyzed in a very rapid and efficient way in order to derive predictions faster than the competition. The analysis of data this large can thankfully be managed by use of a distributed system that will process the various tasks in parallel. Other uses of distributed computing are finding their way into financial analysis. Some such applications are insurance companies or banks which repeatedly calculate their risk exposure in various sectors or hedge funds that use highly optimized computers to perform high frequency trading.

Background

A few years ago, Google released a paper regarding its new paradigm for the way it handles the massive amount of information that it has acquired (Dean, Ghemawat). The concept was called Map-Reduce and since the papers publication, it has been steadily gaining popularity among companies with massive amounts of data. Google's implementation has been kept proprietary, but an open-source project called Hadoop was created to satisfy the needs of other companies with massive amounts of data such as EMC, Facebook, and IBM to name a few.

Hadoop is currently implemented in Java and it being hosted as an Apache project. It was developed as open-source software for reliable, scalable and distributed computing. Hadoop is broken into two parts. The first is a distributed file system and the second is the Map-Reduce programming model. The Hadoop Distributed File System (HDFS) is used to replicate and distribute the data around a cluster of computers. If one or several of the computers fails for any reason, as long as one computer remains with the data, more copies of the data can be distributed throughout the cluster. This helps to ensure than data is always available even in the event of machine failures, rack failures, or even data-center failures.

Map-Reduce is a framework that is typically used for processing large data sets. Its innovative concept in distributed computing involves moving the computation closer to the data rather than the other way around. Map-Reduce works by having the programmer implement both a map task and a reduce task. The map task is used as the first stage of the processing where the input is used to produce intermediate key/value pairs. These key/value pairs are then optionally combined, partitioned, and then passed to the reduce function whereby a final processing step is performed and then output.

As for the financial analysis aspect, much research has been done in trying to outperform the market. If an algorithm were developed which would consistently outperform the market, this could realize the developer or company a substantial sum of money. There are many strategies which people use to try to outperform the market. One such strategy is through the use of technical analysis. Technical Analysis is a security analysis discipline for analyzing statistics generated by market activity, such as past prices and volume (Investopedia).

One of the other more prominent investment analysis strategies that is often used in stock valuation is fundamental analysis which is concerned with the underlying fundamentals of the company. This data usually comes from the company's quarterly report filings with the SEC and other public announcements.

Requirements

This project was broken up into several smaller phases resulting in an end result which would provide functionality to test various financial analysis algorithms quickly and without much difficulty.

The first goal of this project was to successfully set up and administer a basic Hadoop cluster using commodity machines.

The second goal to the success of the project was to design and implement a stock data crawler.

After all of the necessary data was acquired, a thorough analysis of the data was performed. Various and rather simple statistical measures were sought out from the data. Such examples include time-based moving averages for individual stocks and the variance/standard deviation for individual stocks.

Building on the success of the implementation of various statistical parameters, arbitrage opportunities will be sought after for past data and will be used to make predictions about future pricing and trends of stocks.

Design

Phase 1 – Build Cluster

The setup of the Hadoop cluster was done with several objectives in mind. First, I wanted the experience of setting up a cluster. I felt that doing so would help both with understanding the overall design picture as well as with future endeavors and interests that I might have. Also, having complete access and control over the computers allowed configuration and tuning to set each one up with exactly what it needed, while not having extra programs running. Due to the

typical replication rate for data, three for Hadoop systems, I wanted to have three or more nodes in the cluster.

Phase 2 – Stock Data Crawler

Prior to this project's start, I had previously created a similar but more generic web crawler. For this project, I basically took my previous implementation and made it more specific, targeting specifically querying stock data. The data that was queried came from Yahoo Finance. I intentionally did not optimize this portion of the project; actually I slowed it down considerably for two main reasons. First, I didn't want Yahoo to reject or deny me service, so I decided that waiting 3 seconds between queries wouldn't overwork any of their servers. Also, because the data that was being queried was only end-of-the-day data points, as long as the entire query finished within 1 day, the data would be considered useful.

Phase 3 - Simple Statistical Analysis

For this phase of the project, several statistical measurements were generated from the data. For instance, I wanted to have access to measurements such as the daily high and low values for the stock, the daily open and close values for the stock, the standard deviation and variance for a given time period and lastly, a moving average statistic. To help fit within the Map-Reduce framework, the Welford method was used to calculate the standard deviation and variance. This method allowed for a pipelined approach whereby the map output was only processed once in the reduce task while still generating an acceptable and accurate result for the standard deviation and variance values.

Phase 4 - Arbitrage Strategies

The largest portion of time put forth towards the completion of this project was spent at this stage. Several trading strategies were tested. The first was a moving-average based design which had both a longer and shorter moving average. The longer moving average was adjusted during testing, and a shorter moving average was set to 10 trading days. The moving-average based algorithm is stated as follows (Gleason):

1. Buy when the closing price is over the longer moving average
2. Sell when the shorter moving average goes below the longer moving average.

Two other strategies that were tested were derived from research done by Dr. Andrew Lo, a professor at The Massachusetts Institute of Technology (MIT) and prominent figure in the field of quantitative finance. The two strategies/patterns are known as the head-and-shoulders pattern and the inverse head-and-shoulders pattern. The two algorithms for identifying these patterns are as follows (Lo):

Let $E_1, E_2, \ldots, E_n$, represent the n local extrema for time $t_1, t_2, \ldots, t_n$, where $t_1 < t_2 < t_n$. Extrema is defined as an extreme whereby the slope of the stated point is zero and opposite directions on either side (ie. positive to negative or negative to positive). Then both the head-and-shoulders

(HS) and inverse head-and-shoulders (IHS) patterns are characterized by a sequence of five consecutive local extrema $E_1,\ldots, E_5$ such that

$$HS = \begin{cases} E_1 \text{ is a maximum} \\ E_3 > E_1, E_3 > E_5 \\ E_1 \text{ and } E_5 \text{ are within } 1.5 \text{ percent of their average} \\ E_2 \text{ and } E_4 \text{ are within } 1.5 \text{ percent of their average} \end{cases}$$

$$IHS = \begin{cases} E_1 \text{ is a minimum} \\ E_3 < E_1, E_3 < E_5 \\ E_1 \text{ and } E_5 \text{ are within } 1.5 \text{ percent of their average} \\ E_2 \text{ and } E_4 \text{ are within } 1.5 \text{ percent of their average} \end{cases}$$

These signals were used to determine when to buy or sell a particular stock. If the pattern was found, it was bought (or sold) at the last data point indicating when the best possibility for profit was and then sold 5 days after the end of the moving average.

Implementation

Phase 1 - Build Cluster

Prior to starting the project, I read a few books relating to Hadoop, its uses, and its setup. Thus, setting up the cluster to run as several individual machines was fairly easy and straightforward. Unfortunately, a few minor problems did occur while trying to connect the various computers together into a truly distributed computing cluster. These errors were associated with the formatting of the cluster and were corrected by restarting the cluster and reformatting individual machines. Luckily, at this point, I didn't have any sensitive data in the Hadoop Distributed File System and didn't have to worry about data loss due to the reformatting.

Phase 2 – Stock Data Crawler

Having created a prior stock data crawler using the C programming language, I decided to implement it using Java so that all of the core stages of this project would be implemented in the same programming language. By gathering all of the stock symbols prior to downloading the data, I was able to make sure that none of the stocks were queried more than once as well as have the ability to generate accurate estimates of completion times for the total download.

Phase 3 - Simple Statistical Analysis

Of all of the relevant statistical values desired, several were actually provided in the Yahoo finance query. The only extra steps were passing them through the map into the reduce tasks so that they could be accessed. As for the moving average, an example was found that provided a

simple moving average using Hadoop (Patterson). The example was adapted slightly, but for the most part, it was similar to what I was looking at building.

Phase 4 - Arbitrage Strategies

The head and shoulders pattern was much harder to implement than I had anticipated. Fitting the pattern into the Map-Reduce framework took some creative thinking. An example demonstrating the two algorithms, both head-and-shoulders and inverse head-and-shoulders, is shown below. These figures were generated with real data.

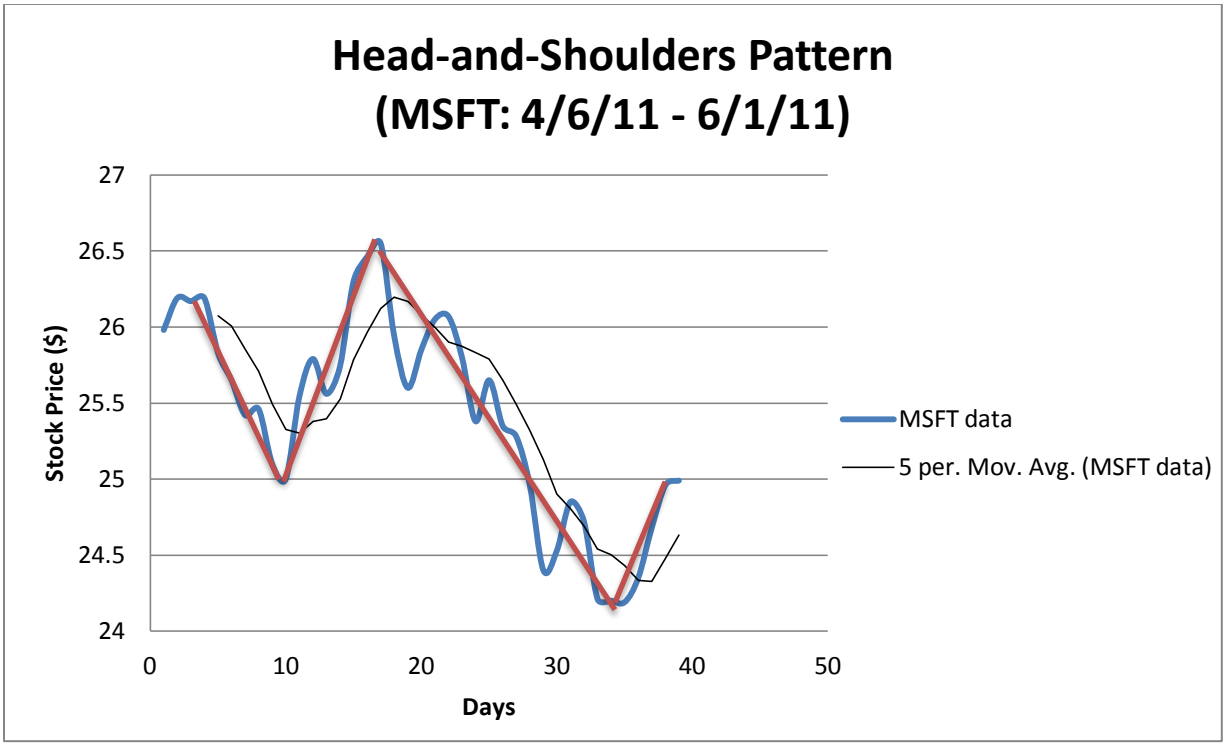

Figure 1 – Head-and-Shoulders Pattern with Real Data

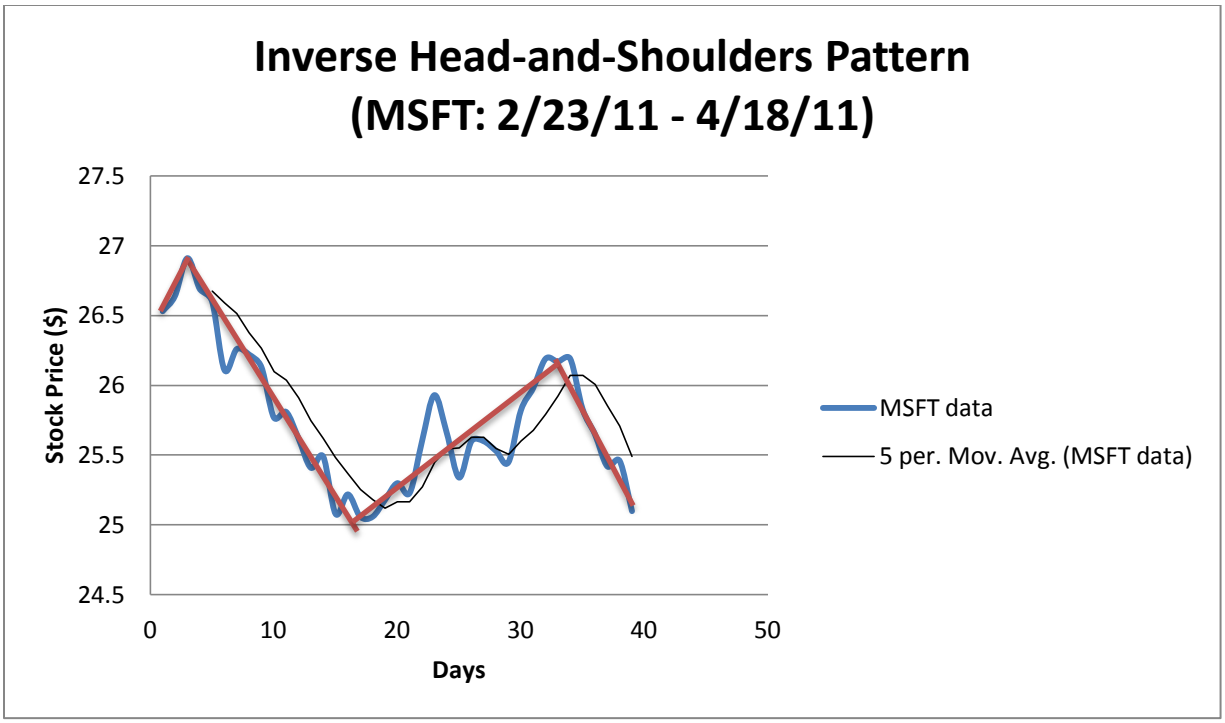

Figure 2 - Inverse Head-and-Shoulders Pattern with Real Data

Results

With both limited processing capabilities per machine and only three machines in the cluster, significant speed improvements were not realized. A modest increase was noticed, roughly 50% speedup, but not directly proportional to the number of the machines in the cluster.

The stock crawler worked great; the total download time for the entire market's data was just over four hours. This long time period, as discussed previously, was intentionally set. It was robust and handled errors such as misspellings in stock symbols, timeouts or other common networking and querying problems.

There are several facts which surprised me with the results relating to trying to beat the market. First, I was surprised at how high the market average was for this given data set. Using a buy and hold strategy, a $10,000 investment in each stock in the market would have returned just over $131,000 in roughly 14 years. This equates to a compound interest rate of 20.2%. Another surprise derived from the results was the extremely poor performance of the head-and-shoulders and inverse head-and-shoulders patterns. As shown in Table 1 below, only one out of six different pattern variations broke even (not counting inflation).

Table 1 - Trading Simulation Results

Moving Average Days	Moving Average – Long/Short strategy	Head-and-Shoulders Pattern	Inverse Head-and-Shoulders Pattern
40	$38,382	$11,350	$3,927
20	$36,198	$4,785	$2,686
15	$32,731	$-15,043	$2,881

Future Work

As for future work, I feel that the steps taken throughout this senior project are merely a proof of concept. Much work can be extended or built upon the existing success of this project. Some enhancements and future ideas that one might decide to pursue based on this project are:

1. Moving the cluster to the cloud. For example, by moving the cluster to a service such as Amazon's EC2, better machines with more available hardware resources would be available. This option wasn't pursued due to costs associated with such usages and because of the availability of commodity machines already possessed.
2. Enhancing the stock data crawler to automatically query and add the day's stock information to the existing data sets would be a large improvement. Currently, an entire run is necessary to re-download all of the data which can amount to a significant amount of time, considering that there are over five thousand stocks.
3. More, or other, technical analysis strategies could be explored. Adding other strategies is actually quite simple. With the basics of the framework having already been setup, final strategies that were being implemented and tested required fewer than 3 hours worth of development.
4. Other extension of technical analysis could be analyzed such as the performance of sectors or classes of stocks. Do small cap stocks outperform large cap? Are technology or commodity-based companies more volatile? These somewhat simple but very important questions could be answered with simple extensions to the existing system.
5. Extend the use of the system toward a more high frequency trading platform. During high frequency trading, small inefficiencies in the market are calculated and then the corresponding stocks are bought (or sold) to correct the market and profit a small amount in the process. Because high frequency trading occurs so often, these small gains (or losses) can quickly become magnified for the trading firm. A distributed system for this kind of calculation would be perfect because each stock or particular strategy could have

a dedicated computer which would look for the opportunity and then send a buy (or sell) signal to a master machine that would actually perform the trade.

Conclusion

The several technical analysis strategies that were implemented and tested were not successful at beating the market. Actually, they were far from it. From the derived results, buying index funds that would track the market would have provided far superior portfolio returns. This lackluster performance can partially be attributed to having only analyzed well-known strategies that many individuals are trying to profit from. If such patterns were ever found, or if new ones are found, the individuals who create them would obviously take advantage of the strategy such that it would no longer exist in the market and it would make them a significant sum of money.

References

Dean, Jeff, and Sanjay Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters." *Google Research Publication: MapReduce*. Google Inc., 1 Dec. 2004. Web. 29 May 2012. <http://research.google.com/archive/mapreduce.html>.

Gleason, Tom. "Moving Average Timing Systems." *Moving Average Timing Systems*. Southwest Ranch Financial, LLC, 1 Jan. 2003. Web. 16 Jan. 2012. <http://www.gleasonreport.com/documents/ma-systems/movingavg.htm>.

Lo, Andrew W. and H. Mamasky. Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation, with A.W. Lo and H. Mamaysky, Journal of Finance 55, 1705-1770, 2000.

Noll, Michael G. "Running Hadoop on Ubuntu Linux (Multi-Node Cluster)." Michael G. Noll. 14 Mar. 2012. Web. 28 Dec. 2011. <http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-multi-node-cluster/>.

Osler, Carol Lee. Identifying Noise Traders: The Head-and-Shoulders Pattern in U.S. Equities. New York, NY: Federal Reserve Bank of New York, 1998. Print.

Patterson, Josh. "Simple Moving Average, Secondary Sort, and MapReduce (Part 3)." Cloudera. 11 Apr. 2011. Web. 16 Jan. 2012. <http://www.cloudera.com/blog/2011/04/simple-moving-average-secondary-sort-and-mapreduce-part-3/>.

Ponzo, Peter, M. Kishinevsky, and M. Higgs. "Downloading Yahoo Data." Yahoo Data Download. Web. 19 Dec. 2011. <http://www.gummy-stuff.org/Yahoo-data.htm>.

Rogers, Shawn. "Big Data Is Scaling BI and Analytics." *Information Management Magazine Article*. Information Management and SourceMedia, Inc., 1 Sept. 2011. Web. 22 May 2012. <http://www.information-management.com/issues/21_5/big-data-is-scaling-bi-and-analytics-10021093-1.html>.

"Technical Analysis." *Investopedia*. Investopedia ULC, 1 Jan. 2012. Web. 22 May 2012. <http://www.investopedia.com/terms/t/technicalanalysis.asp>.

White, Tom. Hadoop: The Definitive Guide. 2nd ed. Sebastopol: O'Reilly, 2010. Print.