Investment Portfolio Performance Evaluation

Jay Patel
(jayp@seas.upenn.edu)
Faculty Advisor: James Gee

April 18, 2008
Abstract

Most people who contract with financial managers do not have much understanding of how much benefit paid support provides them in terms of investment return. Similarly, most people who manage their own portfolios do not know how well they are doing with respect to the rest of the financial market – they simply define a successful portfolio by a positive return on investment. The goal of this project is to provide an elegant, simple interface for people to compare different financial instruments offered by various financial managers. The key point here is ease of use. The ideal website will have an interface which is accessible and intelligible to the large majority of users who have little to no financial knowledge. The broader goal of this project is to not just compare instruments offered by financial managers, but to also compare these instruments to a user’s own portfolio.

Perhaps the most interesting part of this project is the social networking aspect and potential. Along with their own portfolios, users could also see some of the contents of other user’s portfolios. The amount of information available from this service only increases as more users utilize the website. This information can be aggregated to provide interesting analytics not currently available. For example, we can see which stocks are most popular among users, along with the popularity of certain sectors and industries. We can also see which stocks among those in users’ portfolios performed best as well as those which lost the most value. We can also rank users based on their portfolio performance. This lets users examine the trading patterns of the top performing users and portfolios and compare them to their own. Through these unique analytics, we can obtain a new kind of data – the investment patterns of the average consumer.

However, due to the timeframe of this project, it is not feasible to have a meaningful dataset of users to fully explore the social networking aspects. For this reason, the focus will be on individual investors and their portfolios. Portfolio performance will be examined from both a short and long term viewpoint. Performance will also be examined on the industry and sector level. Again, to maintain the tenets of clean interface and ease of use, initial comparison will strictly be on net return. Unlike beta values or moving averages for example, net return is a value even novice investors can look at and easily comprehend.

Related Work

There are a few websites today that provide similar services to those of my project. One such example is Morningstar (http://www.morningstar.com). Morningstar provides data on more than 250,000 investment offerings in the form of stocks, mutual funds, and ETF’s (exchange traded funds). The company also provides a unique Morningstar Rating service, Morningstar Style Box, and other proprietary measures for investors to evaluate their holdings. Furthermore, the website provides financial articles to give users general advice. However, many of their rankings are limited to paying, premium customers. The primary service this website provides is its own investment advice, which comes from Morningstar’s researchers and proprietary evaluation techniques. Morningstar does not make use of member information for investment evaluation. This project, on the other hand, strictly focuses on user input for investment techniques. The goal of the project is not to tell investors what to invest, but rather to let investors experiment with their own theories and strategies.
Another website which provides a wealth of financial tools is Yahoo! Finance (http://finance.yahoo.com). This website provides a wide array of financial information, including news, blogs, and message boards. They also provide articles and profiles of companies, as well as analyst opinions and research reports. While all of this information is useful to an investor, Yahoo! Finance seems to fit a niche separate from my project. Yahoo’s website seems primarily useful for investors seeking information on a specific stock or company. Although there is a wealth of information specific to a particular stock, the website has considerably less information about the overall market or larger indicators. The main feature Yahoo! Finance has that makes it stand out from other financial websites is its charting capabilities. Yahoo! provides dynamic, full screen charts that are fully customizable. The charts feature index comparisons as well comparisons to other stocks. They also provide access to a number of technical indicators. Though this is a very useful tool for the advanced investor, for novices most of the advanced features would not come into use – most novice investors do not care about Bollinger Bands or Money Flow Indexes. While I plan to display charts on my website, I intend to keep the charts very basic so as to prevent users from getting lost in the technical details of analysis. While advanced options like Yahoo! Finance are open users who want to broaden their financial expertise, the simple, easy to digest interface of my website will be a useful tool for casual investors.

There has been a great amount of research done on portfolio optimization. While there are many proprietary algorithms for this process that use machine learning or neural networks, one of the basic investment strategies is having a constant rebalanced portfolio. Much as the name implies, this strategy revolves around keeping the distribution of asset in a portfolio the same over time. However, this method is exponential in the number of stocks being considered for the portfolio. To optimize portfolios more efficiently, Kalai and Vempala have used an implementation of Cover’s Universal algorithm to find portfolios that are comparable in performance to those generated by the constant rebalance strategy. The main benefit of this algorithm is that it runs in polynomial time in the number of stocks. Time permitting, the final addition to this project will be a portfolio optimization algorithm along the lines of the algorithm presented by Kalai and Vempala.

The key difference between this project and previous work is the intended audience. While other websites that evaluate different investment options exist, they are usually targeted towards users with some amount of financial knowledge. The goal with this project is to provide a platform that any investor can utilize, regardless of financial knowledge or experience. This project also promisingly fits into the Web 2.0 mentality: the information housed by the service is collaborative and increases as the amount of users increases. To this end, the layout and ease of use will be integral to the popularity and concomitant success of this project.

**Technical Approach**

This project is very data intensive - for each of this website’s members, user information and portfolio contents must be stored on the database. Furthermore, daily price information for each financial instrument will also need to be stored, as well as specific categorical information about each stock. Due to space/computational limitations, I have decided to limit the storage of historical quotes to about two years. My system will not store any stock information prior to
January 1, 2006. For the database component of my project, I used Microsoft SQL Server 2005. I initially set up a server using MySQL, but after starting to program the website, I noticed ASP.NET provided much better integration for MS SQL (as to be expected since these are both Microsoft technologies). Using the IDE, I was able to visually create an XML schema to map my database’s tables and generate adapters to edit and view data. The end result was that I was able to insert, delete, and make changes to database content via my website without having to write out the low level commands for each action. The IDE handled the implementation of low level details such as table adapters, connection pools, and concurrency issues. Due to the nature of my project, my first challenge was to design a suitable database schema to represent the data. The basic schema can be seen below in Figure 1 (only some of the tables I created are shown in the diagram):

Figure 1: Database Schema
Most of the relations in the database schema are fairly straightforward. Basic user information is stored in the table user, with the primary key being username. In stock_info, various information is stored for each stock, including ticker, name, sector, industry, and percent gains over various time periods, ranging from five day to overall. The information specific to each portfolio is stored in port_info. This includes username, port name, and percent gains over the same time periods as in stock_info. Each row in this table is unique on user name and portfolio name, which allows different users to have a portfolio with the same name. The breakdown for each portfolio is stored in port_content. This table shows the tickers that are in a portfolio and in their percentage amount. The stock quotes for each ticker are stored in the table quote. Finally, to store benchmark information, the table IndexMember can store which tickers are in a given benchmark.

The most difficult part of this project was developing a good method for ranking financial instruments and normalizing existing rankings to provide a single method of comparison among all financial investments. Once we have normalized rankings for different instruments, we have a means of easily comparing, for example, the performance of the Dow Jones or S & P 500 to the client's own portfolio. After a suitable normalization scheme is decided upon, the rest of the project is fairly straightforward. The ranking scheme I am using is fairly simple – different instruments are compared on relative rate of return.

The second major piece of this project is some connection to the stock market. This consists of two components: current stock prices, and historical stock prices. While there are a number of free online resources that provide access to current stock quotes, access to historical stock quotes typically requires a monthly subscription fee. One such example is Strikelron – a fee-based web service which provides an API for direct access to current quotes as well as historical data. One free alternative that offers both kinds of prices is MarketWatch (http://www.marketwatch.com), a web resource provided by Dow Jones. However, unlike most fee-based services which provide an elegant, easy to use API that can be directly invoked by prospective applications, the free MarketWatch service provides data in text format meant for consumption by an end user. For this reason, the data provided by MarketWatch must be parsed using some kind of text scraping script.

However, instead of using MarketWatch, I decided to retrieve my stock quote information from Yahoo! Finance. This website proved to have a number of advantages over MarketWatch. The first was the fact that Yahoo! Finance provided historical quotes in a clean, easy to read and load into database .csv (comma separated value) format. Yahoo! Finance also provided uniform information on industry, sector, and index membership of stocks. On MarketWatch, this information was not always guaranteed to be available and the format was less structured and therefore more difficult to parse. Lastly, Yahoo! Finance provided more information (again in .csv format) if I ever wanted to use an advanced method for financial instrument rating. The extra information I could access included things such as moving average, price earnings ratio, short ratio, market capitalization, volume, etc.

In order to access Yahoo’s stock quotes, I created a simple Yahoo! Finance API. The API consists of two methods – one to retrieve broader stock information for a ticker (industry, sector, etc.) and one to retrieve stock quotes between a given date and the current date. To lower the
overhead on Yahoo’s servers, my API only asks for any type of information once. After the first time information is requested, it is stored on my own server. Any subsequent requests for that information will be directed to my database instead of Yahoo. For example, with stock quotes, when stock quotes are requested, first the API examines when the quotes for the ticker were last updated in the local database. Then, the API retrieves all stock quotes for the given ticker between the last updated date and the current date. However, if this project were ever to advance, it would ideally need a more direct (and immediate) connection to stock information. Nevertheless, scraping financial data from Yahoo! Finance is adequate for a proof of concept demonstration.

The next major component of this project is the website itself. The front-end needed to be aesthetically pleasing and have a well-designed user interface. Ease of use is crucial for user retention. After comparing different frameworks, I decided to use ASP.NET with a C# backend. To gain access to a new programming technology called LINQ, I used Visual Studio 2008. LINQ (Language Integrated Query) adds native querying syntax to .NET programming languages. Although similar in syntax to SQL, a key difference with LINQ is the ability to directly map database content to collections. For my project, this is particularly useful when accessing historical stock quotes for graphs. To mitigate the need for full page refreshes every time something is added or removed from a portfolio, I used an AJAX wrapper on certain components. This highlights one of my main reasons for choosing the .NET framework – my project uses ASP, C#, LINQ, XML, and JavaScript all working together seamlessly.

The layout of the website is fairly sparse – the goal was to go for a utilitarian feel. As seen in Figure 2, each page has a title bar and a side navigation panel with links to the website’s different functions. The rest of the page comprises the body, where each page’s content is shown. This scheme is utilized using ASP.NET’s concept of master pages. Each web page can inherit a master page, which can contain style information as well any code that would be consistent throughout different pages. Each webpage in this project inherits from a master page with the basic layout. The content for each page is placed in a content placeholder tag. The benefits of this approach are twofold: there is less overhead on the server when clients make page requests and the code itself is more efficient as common elements such as the title navigation bars do not have to be redefined for each web page.
In terms of functionality, the first web page of note is the Manage Portfolio section. On this web page, users can add or remove stocks from their portfolios, as well as change their percentage in the portfolio. This page also provides an interface to add, remove, and rename portfolios. The basic form that allows these actions can be seen below:

![Figure 3: The form used to manage portfolio contents.](image)

The next, and perhaps one of the most important pages, is Portfolio Return. On this page, the user is shown the contents of his portfolio (including percentage composition), how each individual stock performed, and how the portfolio as a whole performed. This information is displayed in both tabular and graphical form. There is also a control to choose different lengths of time for comparison, ranging from five days to over two years.
Figure 4:  The Portfolio Return page.

The Industry Breakdown page seen below in Figure 5 shows the diversification of the portfolio on an Industry level.  Like the Portfolio Return page, the top of the page consists of a table showing the contents of the user’s current portfolio.  Below this table, a pie chart is expresses the Industry breakdown of the portfolio.  This helps users easily determine if their portfolio has any particular area of saturation.
The Sector Breakdown page looks almost exactly like the Industry Breakdown page, with the only difference being what values are used for calculation. Although I did not have time to fully implement it, one of my goals for the project was to have both breakdown pages suggest stocks to the user based on how they performed relative to other stocks in the same industry or sector. However, due to the nature of the calculation and how stock information is retrieved, with a sparse amount of users it would be difficult to obtain useful suggestions.

**Conclusion**

The most difficult and unanticipated challenge of my project was my lack of prior web programming experience. With my knowledge of data modeling I was able to design a solid
database schema. Even the Yahoo! Finance API, which I expected to be one of the most difficult aspects of the project, was fairly straightforward. However, the design and programming on the front end proved to be by far the most difficult and time-consuming part of my project. I explored countless numbers of web technologies while deciding what would work best for my project. At times it felt as if I spent more time learning new web technologies than I did focusing on the topic of my project. In retrospect, a better approach would have been to use something simple, such as just PHP and SQL, and flesh out my project. Once basic functionality was in place, I could have moved on to more complicated technologies for greater efficiency and a more enjoyable user experience. However, all of the skills I learned will undoubtedly serve me in the future – the web has truly become the standard platform for modern applications.

Overall, I feel a strong sense of accomplishment in my work on this project over the past two semesters. While as a final product this project is far from complete, with my limited resources and lack of prior web design experience, I developed a product that I would want to use to evaluate my own investment strategies. As a person who fits the target demographic of my website, this fact alone leads to a great deal of satisfaction. With a little more time and work, this project could become a truly powerful investment evaluation tool. In the future, I aim to further refine this project and eventually deploy it.

**Milestones/Timetable**

1. Learn AJAX/Ruby, SQL, and basics of database design.
2. Develop database with tables needed to store financial information.
3. Develop algorithm to evaluate performance of a financial instrument.
4. Create website with preliminary user interface.
5. Create a text scraping script to fetch historical and current stock quotes from Yahoo! Finance.
6. Add ability to compare different financial instruments. Show top performers, etc, on user interface.
7. Provide support on website for multiple users.
8. Add user information to database.
9. Add in features to compare different user’s portfolios as well as user ranking.
10. Improve user interface
11. Increase efficiency of stock quote retrieval algorithms.
Related Work:


This website provides access to over 70 web services; include those which provide current and historical stock quotes. Again, none of the free web services provided by this website provided the information needed for my project. This was another website I used as a starting point for research on what options were available to me for stock quote retrieval.


This article is published in an editor reviewed journal which only accepts original, unpublished research in the field of machine learning. The authors discuss a common portfolio optimization strategy known as Constant Portfolio Rebalancing (CPR). However, all known implementations of this algorithm are exponential in the number of stocks considered for the portfolio. These authors introduce a new algorithm which generates portfolios competitive in performance to CPR portfolios in polynomial time. While not the most current work in this field, the article and problem set described by this article fits nicely within the scope of my project.


This is the company website for MarketWatch. MarketWatch provides up to date information about stocks, as well as news articles. This service is provided by Dow Jones and intended for people interested in financial news and stock quotes. This resource was considered as a source of stock quotes, but after further research another source was chosen.


This is the company website for Morningstar. Initially as a journal and now in website form, Morningstar has been a provider of independent investment research for over twenty years. Although most of their information is reserved for premium members, the free sections have access to news and stock quotes. Morningstar is known for their proprietary ranking system, and is particularly well known for its information on mutual funds. The Morningstar website is a product that is similar to my project, but the main difference is that Morningstar provides ratings based on their objective investment
research (an active approach). With my project, I hope to rate investment options with a quantitative approach.


This website serves a source of information on mashups and new web API’s. While it does not provide products of its own, it has many reviews and links to existing mashups and the API’s they use. This is a particularly good source for developers who are looking to use an existing web API. It also provides reviews of current mashups that have been submitted by developers. For my project, I used this website as a starting point to see if there were any financial API’s available for me to use. Unfortunately, all of the financial API’s I found either had limited information or required an expensive subscription fee.


This is the company website for StrikelRon. StrikelRon’s primary product is an API and web service that provides financial data. StrikelRon’s services easily integrate into applications and the web, and provide access to current and historical stock quotes. However, the main downside to StrikelRon is that it is a fee based service. Since most of the data can already be found on other websites, I decided to scrape my data from another website rather than pay for StrikelRon’s premium service.


Yahoo! Finance is a source of financial advice and data, as well as news. Although Yahoo! is primarily known for their search engine, their finance website is a very reliable source for financial information and stock quotes. After much research I decided to use Yahoo!’s site for stock sector and industry information. I also later decided to use Yahoo! Finance as my source of current and historical stock quotes.