Department of Computer Science and Engineering
UNIVERSITY OF COLORADO
DENVER | ANSCHUTZ MEDICAL CAMPUS

Final Project Report

Stock Market Simulation

Team Members: **Kuilin Wang, Qian Mai**
CSCI 5800-001 Simulation
Computer Science and Engineering
University of Colorado Denver

Contents

1. Abstract

This project is about simulating the stock market exchange for a period of time in the future. The basic technique we used for generating the future price is Monte Carlo method. Based on the generated future price, we used a trading strategy to determine when to buy and sell the stocks. The whole project was written in Python. We also used Matlab and Excel to generate future indices of the NASDAQ.

2. Motivation and Problem

A stock market is s an institution where humans and computers buy and sell shares of companies. A stock market simulator is A program or application that attempts to reproduce or duplicate some or all features of a live stock market on a computer so that a player may practice trading stocks without financial risk.

Compare to the car wash, stock market needs minimal amount of physical effort to play with, but we can make much more money through it.

Like most people, we curious about how a stock index changes and how to decide the buy-time and sell-time. That became the problem popped up in our head.

3. Background Knowledge and Introduction

3.1 Monte Carlo Method

Monte Carlo methods are used in finance and mathematical finance to value and analyze (complex) instruments, portfolios and investments by simulating the various sources of uncertainty affecting their value, and then determining their average value over the range of resultant outcomes. This is usually done by help of stochastic asset models. The advantage of Monte Carlo methods over other techniques increases as the dimensions (sources of uncertainty) of the problem increase.

3.2 Relative Strength Index (RSI)

The relative strength index (RSI) is a technical indicator used in the analysis of financial markets. It is intended to chart the current and historical strength or weakness of a stock or market based on the closing prices of a recent trading period. The indicator should not be confused with relative strength.

The RSI is classified as a momentum oscillator, measuring the velocity and magnitude of directional price movements. Momentum is the rate of the rise or fall in price. The RSI computes momentum as the ratio of higher closes to lower closes: stocks which have had more or stronger positive changes have a higher RSI than stocks which have had more or stronger negative changes.

The RSI is most typically used on a 14-day timeframe, measured on a scale from 0 to 100, with high and low levels marked at 70 and 30, respectively. Shorter or longer timeframes are used for alternately shorter or longer outlooks. More extreme high and low levels—80 and 20, or 90 and 10—occur less frequently but indicate stronger momentum.

4. Implementation

4.1 Data Collection

One way to get NASDAQ historical stock raw data is from Yahoo website. We can select data in a specific period and download it as a spreadsheet.

Figure 4.1.1 Data Collection

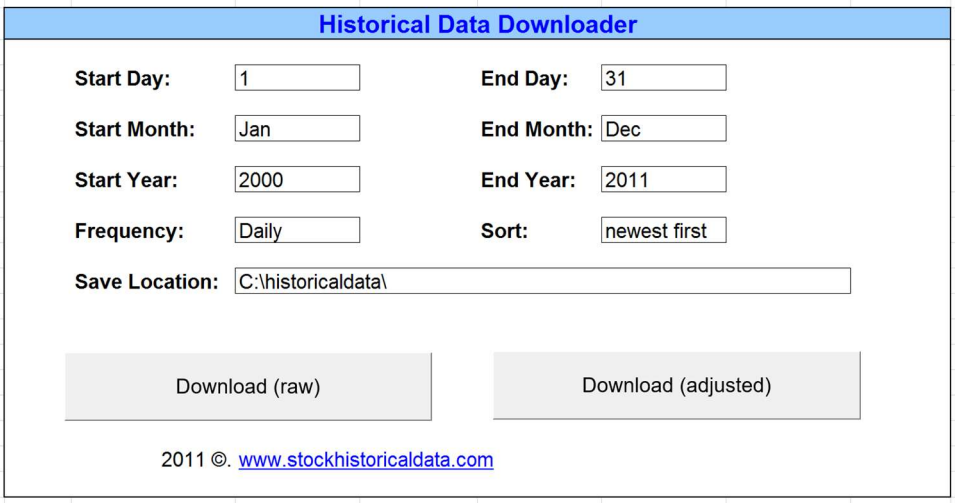

Figure 4.1.2 Data Collection

4.2 Matlab Approach

Using Matlab, we can transfer data from Excel spreadsheet to Matlab chart.

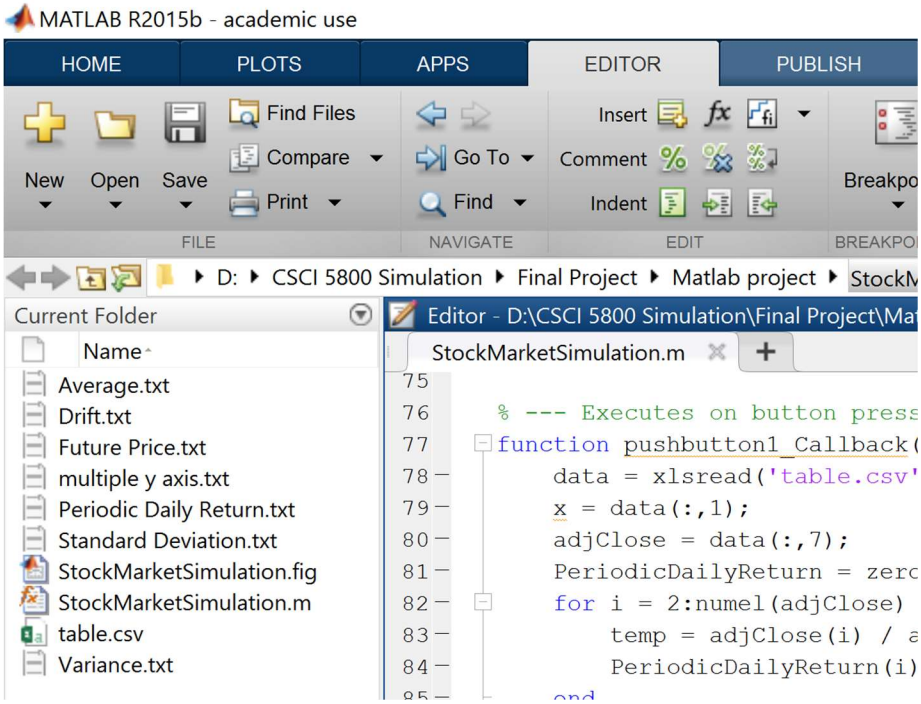

Figure 4.2.1 Matlab Approach

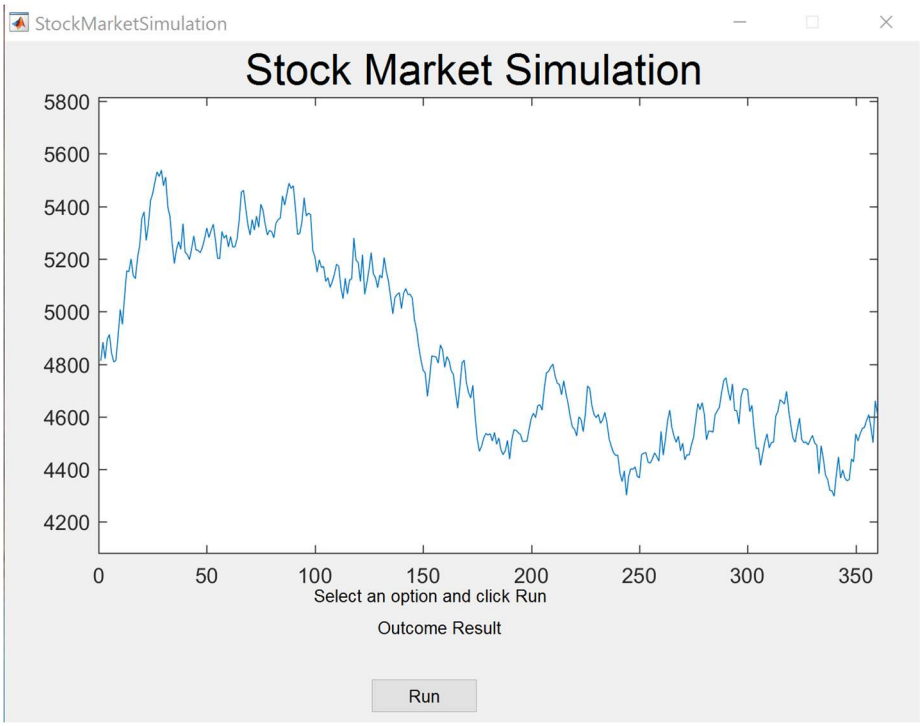

Figure 4.2.2 Matlab Approach

4.3 PyCharm Approach

PyCharm is an Integrated Development Environment used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems.

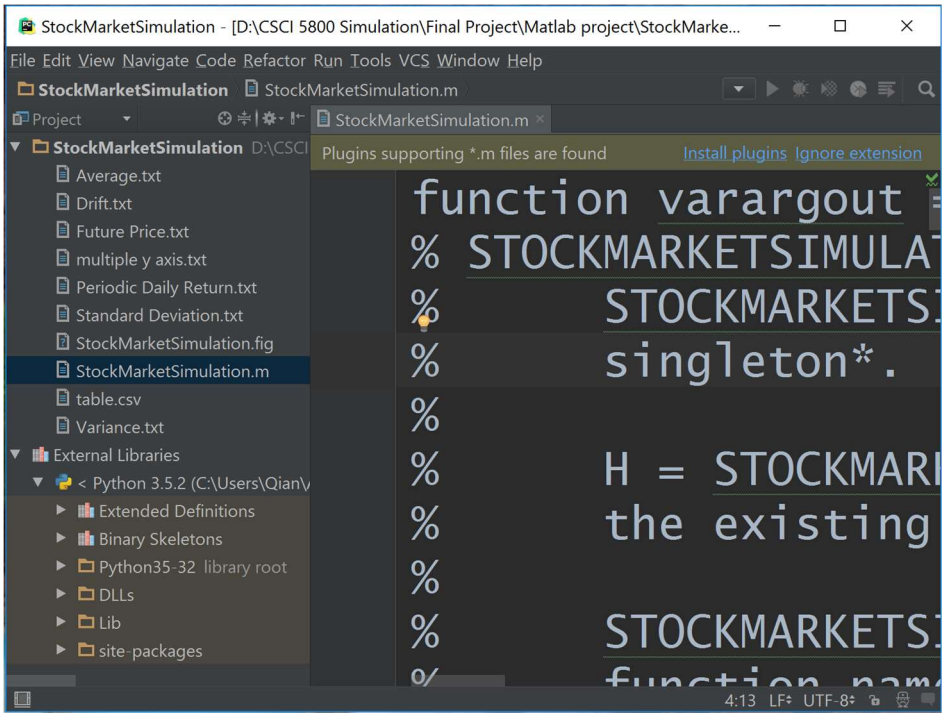

Figure 4.3 PyCharm Approach

4.4 Visual Studio Approach

Python Tools for Visual Studio is a free, open source plugin that turns Visual Studio into a Python IDE. It supports CPython, IronPython, editing, browsing, IntelliSense, mixed Python/C++ debugging, remote Linux/MacOS debugging, profiling, IPython, and web development with Django and other frameworks.

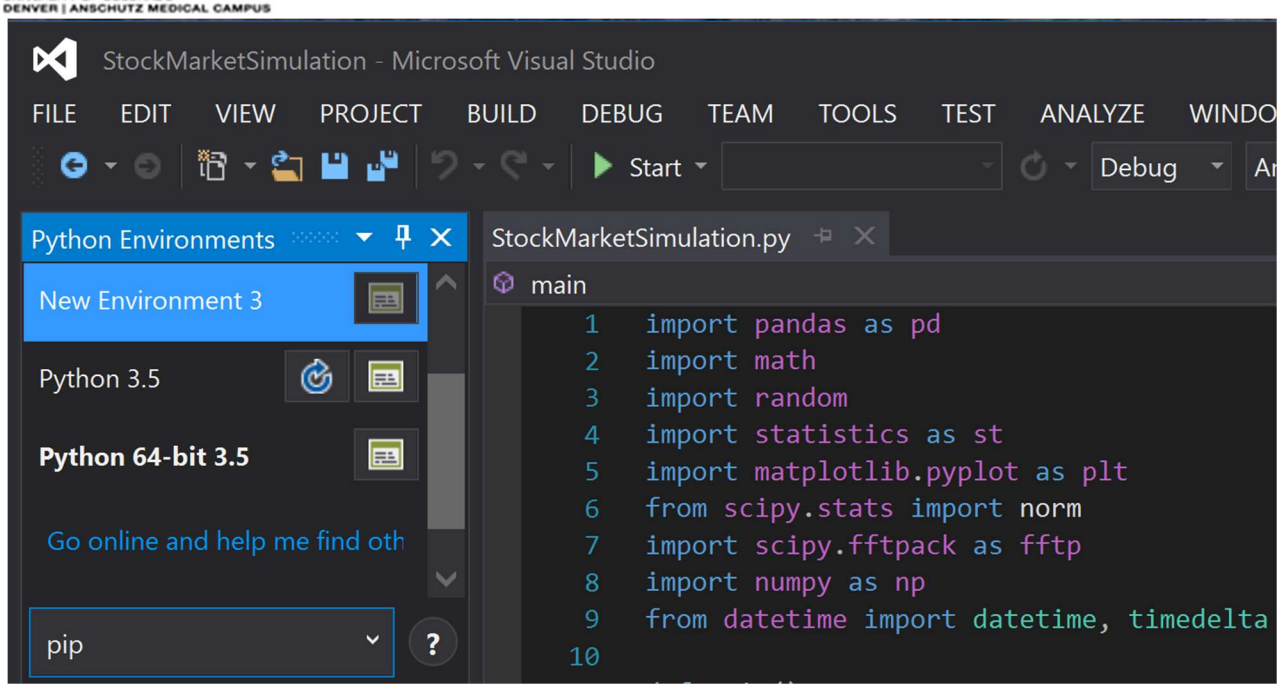

Figure 4.4 Visual Studio Approach

4.5 Monte Carlo

We made an Excel table as follow.

Date	Open	High	Low	Close	Volume	Adj Close	Periodic Daily Return		Future Price
7/1/2016	4837.180176	4880.169922	4837.169922	4862.569824	1745130000	4862.569824	0.004100863123		4862.569824
6/30/2016	4793.75	4843.109863	4774.52002	4842.669922	2171180000	4842.669922	0.01318257498	Average	4810.132477
6/29/2016	4732.930176	4787.589844	4732.339844	4779.25	2116550000	4779.25	0.01845238182	0.000057199222	4823.634631
									4841.307937
								Variance	4916.390486
								0.000126442378	4851.057966
									4802.342078
								Std Dev	4848.766986
								0.01124466001	4880.054224
									4878.868326
								Drift	4781.958466
								-0.00000602196	4753.686087
									4722.23934
									4705.281307
									4782.015523
									4729.544902
									4826.951155
									4772.449751
									4694.140423
									4771.631747
									4798.03479
									4814.978516
									4869.397011

Figure 4.5 Monte Carlo Demonstration by Excel

4.6 Future Prices and Trading Volume

We used one-year historical data of NASDAQ adjusted close index to calculate periodic daily return values. $periodic\ daily\ return\ values = ln\left(\frac{current\ index}{previous\ index}\right)$. So if there are n days indices, then it will be $n - 1$ periodic daily return values. We also calculated the $average$, $variance$ and $standard\ deviation$ of periodic daily return values, and used $average - \left(\frac{variance}{2}\right)$ to get the $drift$. The last step, we calculated $previous\ ind$ $(drift + standard\ deviation \times NORMSINV(RAND(\ ,1)))$ iteratively to get the future price in a random pattern.

4.7 Trading Principal

RSI was used to decide the buy-time and sell-time. To calculate the RSI, we need to get the total amount of increase and the total amount of decrease. Then

$$RSI6 = \left(\frac{total\ amount\ of\ increase\ in\ 6\ days}{total\ amount\ of\ decrease\ in\ 6\ days\ +\ total\ amount\ of\ increase\ in\ 6\ days}\right) RSI12$$

$$= \left(\frac{total\ amount\ of\ increase\ in\ 12\ days}{total\ amount\ of\ decrease\ in\ 12\ days\ +\ total\ amount\ of\ increase\ in\ 12\ days}\right)$$

Because the RSI6 changes faster than the RSI12, RSI6 up-crossing RSI12 shows the trend of increase and RSI6 down-crossing RSI12 shows the trend of decrease. We buy in after up-crossing and sell out after down-crossing.

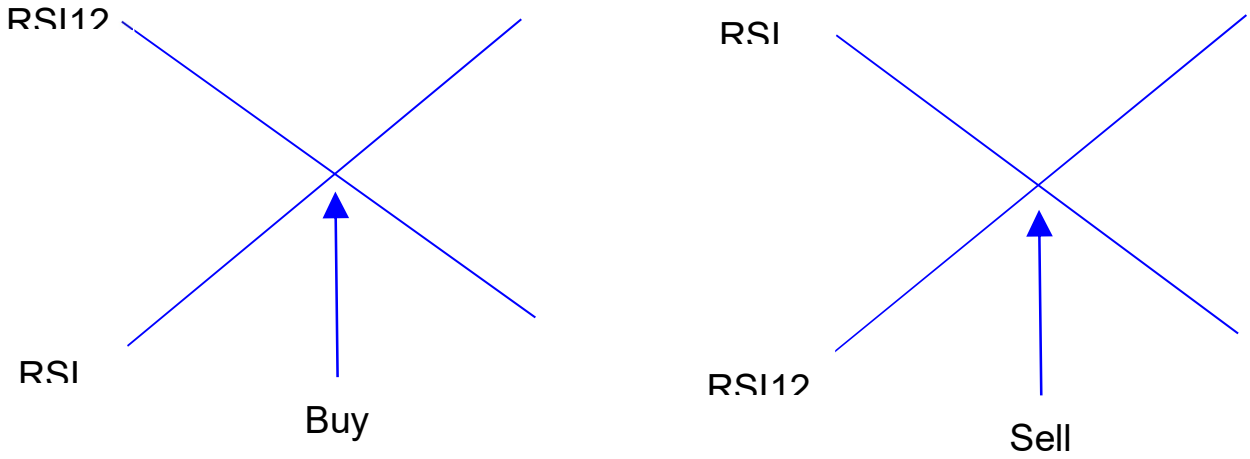

Figure 4.7 Buying and Selling Points

5. Results

5.1 Priceline and Trading Volume

The main price chart only displays daily price variation (blue bold line), but also includes original referral price line (black bold line), highest and lowest period record (green and red thin lines).

The secondary trading volume bar chart is according to two adjacent prices difference. The bar height reflects market fluctuation level as well. Red bars represent upward market trends while green bars indicate downward trends.

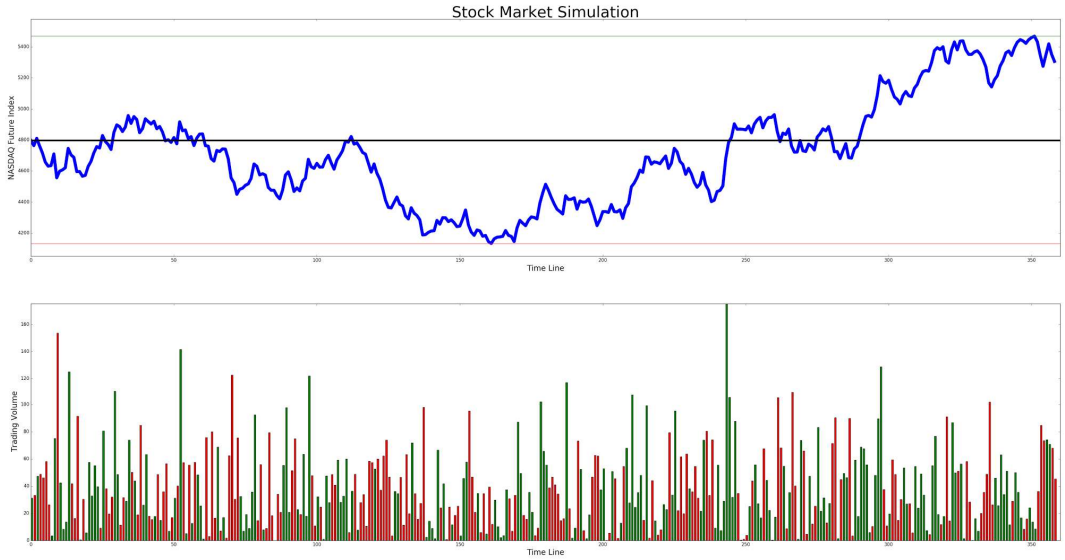

Figure 5.1 Priceline and Trading Volume Bar Chart

5.2 Relative Strength Index(RSI) Chart

The percentages from accumulation of 6 days and 12 days are various. Due to longer time range, the line of 12 days' RSI (magenta line) is more stable than the 6 days' RSI (cyan line). The transaction will be triggered every time that two lines intersect. All exchange points are mark with "buy" or "sell" tag.

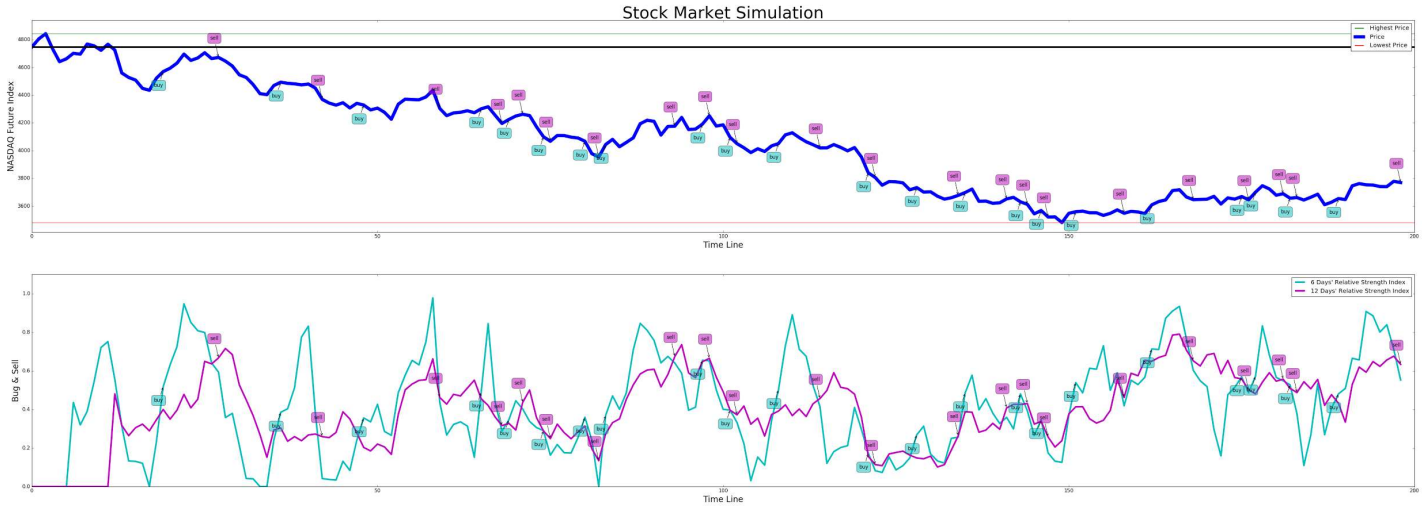

Figure 5.2 Relative Strength Index Chart

5.3 Profit Balance

To simplify the balance, a transaction amount is only one share. Once a sell transaction finish, a new profit comes out with the subtraction of the pair prices.

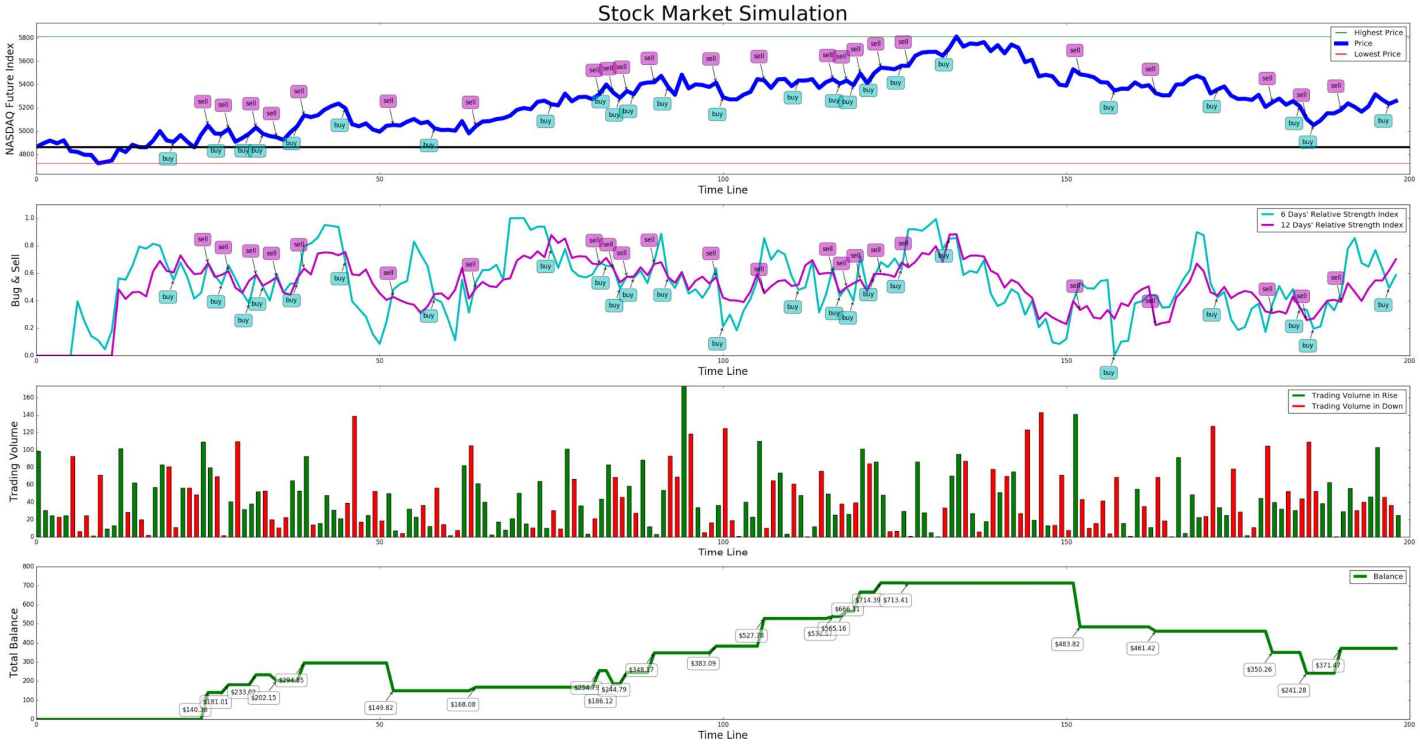

Figure 5.3 Profit Balance Chart

5.4 Optimization for Exchange Decision

Since there shouldn't be so many exchanges during 200 days. We import a new percentage index to compare the buying price and the selling price. The selling exchange will happen if it meets the price gap condition, otherwise keep holding the share. Besides, we set a rule that a buy and a sell as a pair of transaction, so that the next purchase shall wait for the previous sell done.

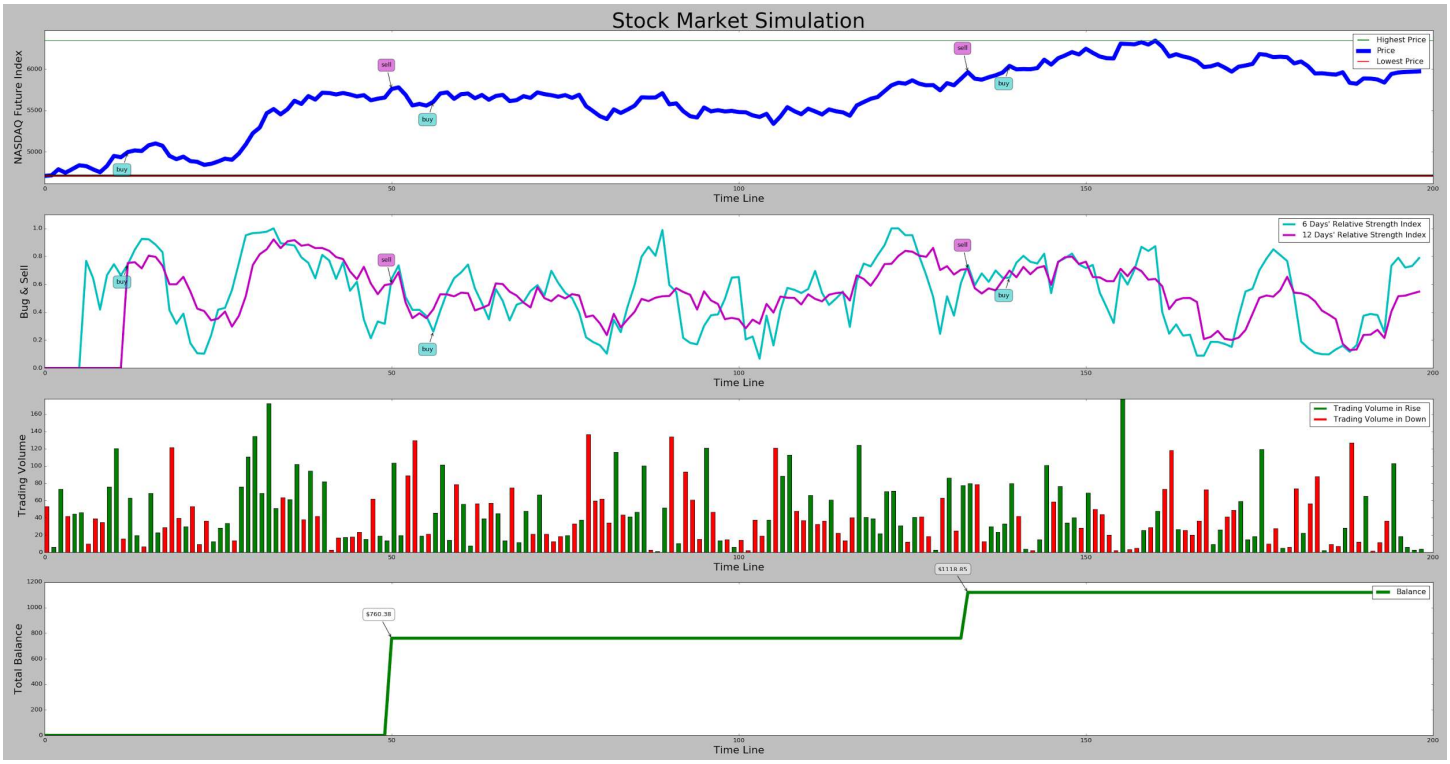

Figure 5.4 Stock Market Optimized Chart

5.5 Volatility Animation

To demonstrate daily stock price volatility and exchange actions in real time dynamically, the program generates new data and refresh charts every couple seconds, which is more reasonable and closed to the actual stock market.

Figure 5.5 Stock Market Dynamic Demonstration

6. Evaluation

6.1 Profit Evaluation in Different Times

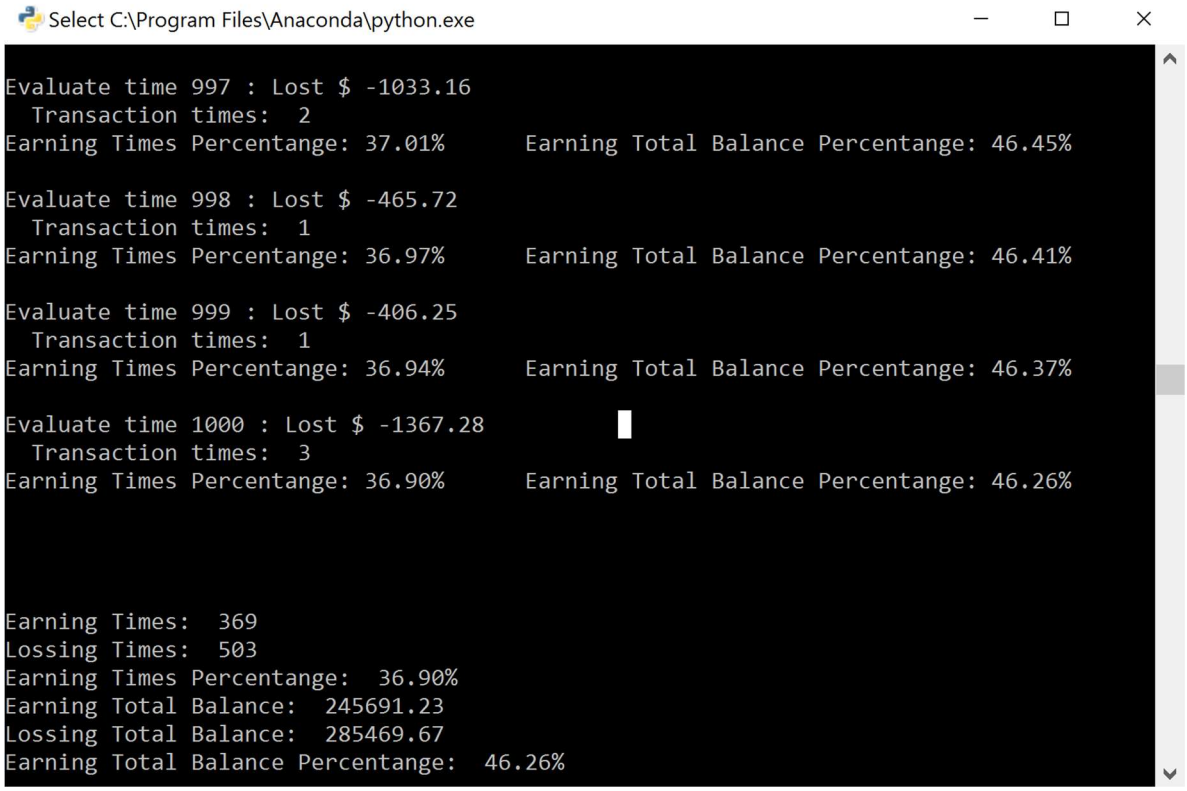

Figure 6.1.1 Stock Prediction Evaluation Real Time Outcome

Base on the adjusted Relative Strength Index strategy, the average transaction times is approximatively 2 times, which leads some of evaluations zero transaction. We consider those results are tie (yellow portion in the pie charts).

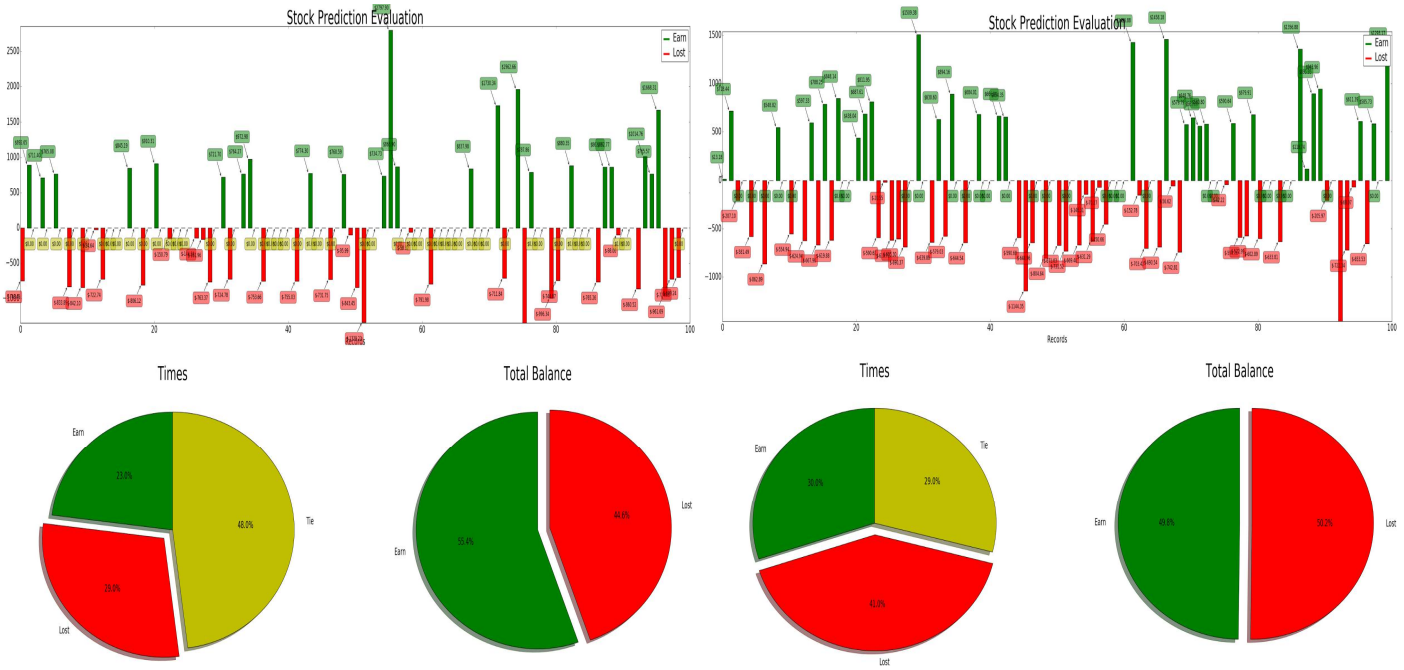

Figure 6.1.2 Stock Prediction Evaluation in 100 Times

This evaluation program is high-effective. It can run 100 times prediction less than 30 seconds. We only two typical results for demonstration. In above pie charts, green slides represent earning outcome while red slides are loss results. From the left partial figure, in the 55.4% chance the strategy is earning base on the total balances in spite of the win times is less than the loss times. The right partial figure shows, on the contrary, more than 50% are deficit.

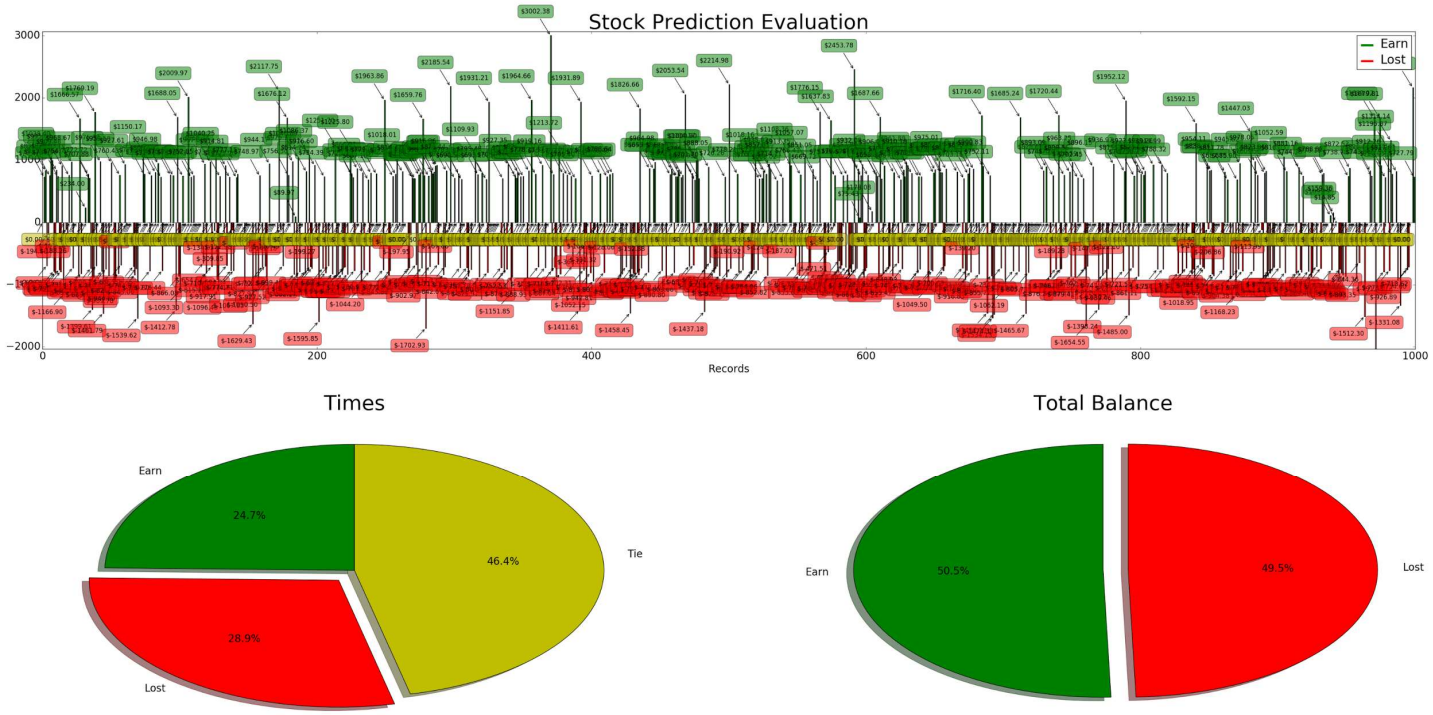

Figure 6.1.3 Stock Prediction Evaluation in 1000 Times

In 1000 times evaluations, the earn and loss are more closed to theoretical value 50% respectively.

6.2 Profit Evaluation Using Two Different Strategies

	if rsi6[k] > rsi12[k] and rsi12[k - 1] > rsi6[k - 1] and isHolding == False: BUY if rsi6[k] < rsi12[k] and rsi12[k - 1] < rsi6[k - 1] and isHolding and np.abs(buy - FuturePrices[k]) / buy > sellGap : SELL					
	100		200		1000	
sellGap	Earn Times/Total Times	Earn Balance/(Earn+Loss)	Earn Times/Total Times	Earn Balance/(Earn+Loss)	Earn Times/Total Times	Earn Balance/(Earn+Loss)
0.0%	50.00%	48.93%	47.50%	45.75%	43.10%	42.85%
1.0%	48.00%	42.22%	47.50%	49.29%	47.50%	47.22%
2.0%	51.00%	47.63%	49.00%	43.71%	43.30%	43.39%
5.0%	42.00%	43.13%	45.00%	44.48%	44.30%	46.10%
7.5%	41.00%	51.16%	41.00%	43.51%	40.20%	43.94%
10.0%	31.00%	45.06%	36.00%	45.26%	33.60%	45.27%

Figure 6.2.1 Stock Prediction Evaluation using strategy 1

	if rsi6[k] < rsi12[k] and rsi12[k - 1] < rsi6[k - 1] and isHolding == False: BUY if rsi6[k] > rsi12[k] and rsi12[k - 1] > rsi6[k - 1] and isHolding and np.abs(buy - FuturePrices[k]) / buy > sellGap : SELL					
	100		200		1000	
sellGap	Earn Times/Total Times	Earn Balance/(Earn+Loss)	Earn Times/Total Times	Earn Balance/(Earn+Loss)	Earn Times/Total Times	Earn Balance/(Earn+Loss)
0.0%	53.00%	50.16%	45.50%	45.30%	48.00%	48.67%
1.0%	38.50%	41.03%	53.00%	56.45%	46.00%	44.27%
2.0%	47.00%	43.00%	43.50%	46.91%	43.80%	45.11%
5.0%	47.00%	44.75%	43.00%	49.45%	45.10%	45.24%
7.5%	51.00%	54.78%	40.50%	44.16%	38.90%	43.84%
10.0%	38.00%	53.34%	33.50%	44.67%	34.40%	47.09%

Figure 6.2.2 Stock Prediction Evaluation using strategy 2

To fully analyze those two strategies, we select buy/sell gap percentage from 0% to 10%, and run it at 100, 200, 1000 times respectively. The Earn/Total Balance are around 40% and 55%. Neither of them is superior to each other.

7. Conclusions

Stock market simulation is powerful and promising. We have successfully implemented Monte Carlo Method to our future prices generator and Relative Strength Index to our price predicted organism. Thanks to the power python and numpy library, we have created a basic but informative chart interface with prices analysis and stock exchange features. With this platform, as a good start point, all aspects factors of financial information can be imported to rich virtual environment and bring user more authentic experience.

Stock price prediction is complicated. From the further predicted strategy evaluation, unfortunately, neither of two RSI strategies guarantee the profit. We modified the trading strategy several times and analyzed the profit distribution one by one. And then we found out they turn out to be equal, negative and positive profits were nearly 50:50 distributed. This result bothered us quite a bit.

After this study, one idea came up to our mind. There is no way to make a profit just by looking at the chart if it's truly randomized. But in the real world, the fluctuation of the indices is not random. There must be some reasons behind it, may be the good news or bad news of that company, may be the impact of the global market. There are so many reasons that we cannot fully cover, but by using big data analysis, we can still find a way to calculate the probabilities of going up or going down of those indices.

This project is over, but the motivation of making a usable market predictor still continues. Currently financial simulators have left out dramatic manual repetitive computation. In the near future, artificial intelligence will make huge changes to the financial system.

Reference

[1] Jackel, P. (2001). Monte Carlo methods in finance. *Stochastic Dynamics*, *3*, 3-2.
[2] Fournié, E., Lasry, J. M., Lebuchoux, J., Lions, P. L., & Touzi, N. (1999). Applications of Malliavin calculus to Monte Carlo methods in finance. *Finance and Stochastics*, *3*(4), 391-412.
[3] John J. Murphy (2009). The Visual Investor: How to Spot Market Trends (2nd ed.). John Wiley and Sons. p. 100. ISBN 9780470382059.
[4] "RSI". StockCharts. StockCharts.com. Retrieved 29 June 2016.
[5] Herman, Kahn; Harris, Theodore, E. (1951). *"Estimation of particle transmission by random sampling"*. Natl. Bur. Stand. Appl. Math. Ser. 12: 27–30.
[6] Kolokoltsov, Vassili (2010). Nonlinear Markov processes. Cambridge Univ. Press. p. 375.
[7] Kroese, D. P.; Taimre, T.; Botev, Z. I. (2011). *Handbook of Monte Carlo Methods*. John Wiley & Sons.