US Historical Stock Data Analysis

Leung Wing Yan (MB95524), Lam Tek Hong (MB95559), Joseph Yen (YB87412)

In the current age, many people have considered investing in stock and others have traded stocks through mobile applications as a simple buy and sell button. However, it is important that comprehensive trading strategies and technical analysis methods should be used to maximize profit. Through this project, we will try to prepare a program to produce technical analysis such as the Golden and Death Cross in Simple Moving Average Lines, Bollinger Band as well as Regression (Linear & Polynomial), in order to find the buying and selling signals and make a profit through graphic analysis. There is a practical justification to pursue this topic.

In recent years, Macau had initiatives to expand its entertainment portfolio as well as reduce its reliance on gambling. One industry that is growing there is the financial industry. In 2019, A stock trading exchange was announced to be in development as well as the entrance of Ant Financial, a subsidiary of Alibaba has expanded to Macau as a virtual service. Therefore, our team has decided to focus on financial analysis as a relatable and practical topic for our group.

**Data Source**

We chose the “US Historical Stock Prices with Earnings Database” from Kaggle for practicing and learning.

The database contains the data of historical US stock prices from AMEX, Nasdaq and NYSE markets for more than 7000 stocks during the last 20 years, which is from Nasdaq, Yahoo Finance, Zacks, Alpha Vantage. In addition, the size of the database is approximately 2GB.

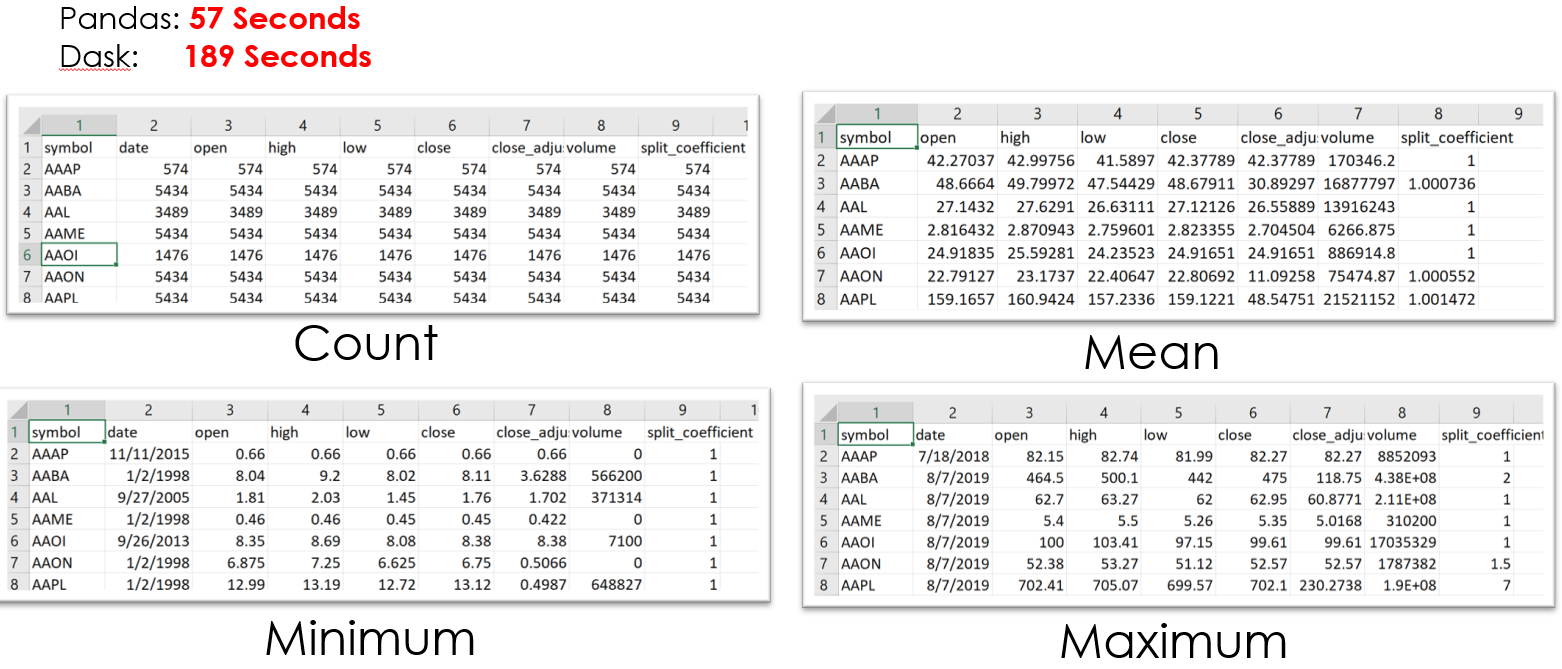
There are 4 files in the database and we used the “stock\_prices\_latest” file of the database whose size is about 1.7GB for analysis and practice. The file consists of totally 2.1 million rows x 9 columns data of the highest, lowest, close, and adjusted close price, etc. of 7091 stocks data in the US from Jan 2 1998 to Aug 15 2019.

**Data Analysis**

1. **Data exploring**

Due to the large size of the data, it leads to be overrun memory for our laptops when exploring the data with the Pandas Library. The describe function required a minimum of 3.6 gigabytes of memory to work. Consequently, we decided to use the Dask Library to solve this problem. Dask Library produces the same result and offloads data to hard disk instead of RAM without the need to close applications to free up memory.

Although Dask is a library designed for large data handling and scalability, it leads to another problem where the calculation of count, mean, minimum, and maximum of the data would be relatively slower to process than the calculation with the Pandas Library as shown. The same calculation tasks would increase from approximately 57 seconds with the Pandas Library to approximately 189s with the Dask Library) for the same calculation. The results for count, mean, minimum, and maximum are shown in figure 1.



*Figure 1*

1. **Data cleaning, sampling and selection**

From the describe analysis, we found that some of the price values were negative, which is impossible for stock. As a result, we recoded all negative stock pricing values to be 0. During the process of trying various analysis tools, we resampled only 5% of the dataset for testing and debugging. For the actual analysis on the whole dataset, we kept only the columns of ‘time’, ‘symbol’ and ‘close\_adjusted’ by writing them to a csv file. Then we can read this smaller csv file for later analysis. As for analysis targets, the popular FAANG Stocks (namely Facebook, Amazon, Apple, Netflix and Google) were chosen. In addition, users can also input their interested stock to see the analysis results.

1. **Stock Analysis**

**Golden and Death Cross**

With the data, we decided to use the 50-day and 200-day Moving Average Lines for analysis. With this, we can visually find out the extreme fluctuations in stock price trend through the graph in order to evaluate opportunities to buy or sell each stock. A Golden Cross is formed when the 50-day Moving Average Line goes beyond the 200-day Moving Average Line. When informed, we can consider buying the stock. The Death Cross is formed when the 50-day Moving Average Line falls below the 200-day Moving Average Line. With this information, we can then consider selling the stock.

We made a function to locate the areas where the cross occurs by:

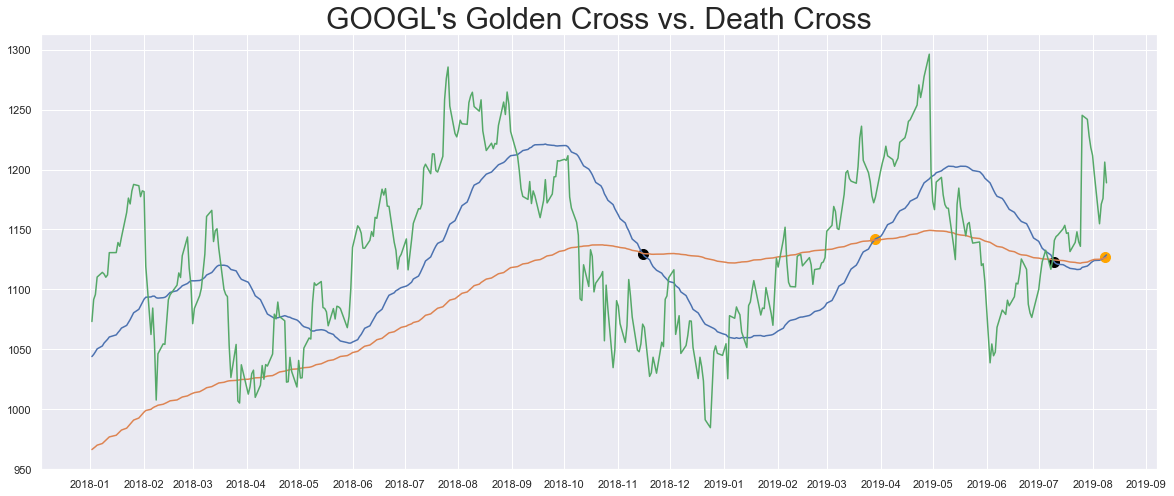
* Transferring the two lines data into a numpy array can speed up calculation.
* Subtraction function to distinguish which line is on top and which is below.
* Shift the subtraction result by 1 place, then multiply them, only at the points of crossing have negative values.
* Determine whether it is Golden Cross or Death Cross according to the subtraction sign
* Extract those numpy positions to two list “Golden” and “Death”. Then produce two subsets of a dataframe of a moving average line, which can be used to plot the cross dots.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Subtraction** | **+** | **+** | **-** | **-** | **-** | **+** | **+** | **-** | **-** |
| **Shift** | **0** | **+** | **+** | **-** | **-** | **-** | **+** | **+** | **-** |
| **Multiply** | **0** | **+** | **-** | **+** | **+** | **-** | **+** | **-** | **+** |

*Figure 2* Logic of finding the cross

For plotting, we have merged the original stock line, the 50-day moving average line, the 200-day moving average, into a Golden Cross list and the Death Cross. In this process, we had to spend quite some trial and error on the merge and join between series and dataframes. From this we learn that it is important to name the generated series for later dataframe uses. Afterwards, we then customize the plotting on all 5 columns, and especially to show Golden Cross and Death Cross in big dots with no line. Despite putting in ‘label’ parameters, we couldn’t get labels for each line to show on graph. Finally, for interactivity: the graph is plotted in the time frame between the input start time and the last day available in the dataset.

We observed that for the shorter time frame, shorter days for moving average lines may be more suitable for detecting the rise and fall of the trend, otherwise, they are quite lagging behind the changes.

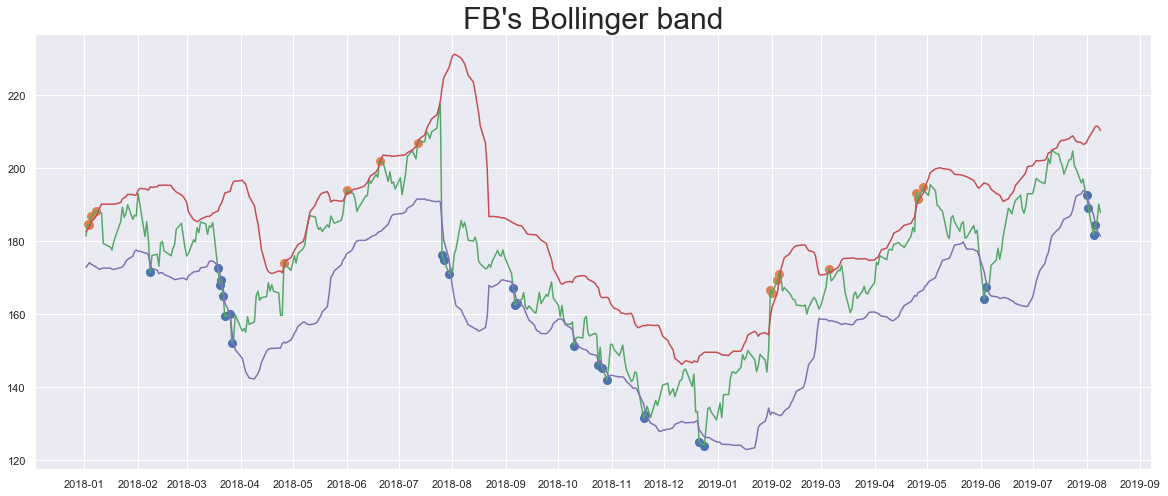


*Figure 3* Result graph sample of Golden Cross and Death Cross

**Bollinger Band**

In addition to the Golden and Death Cross analysis, we included another analysis known as the Bollinger Band. Bollinger Bands are lines whose upper and lower bands are +/- a certain standard deviation from the stock’s 20-day Moving Average Line. With this we can find out the possible change points of the stock price trend through the graph, in order to grasp the opportunities for buying or selling. When the stock price goes beyond the upper line of Bollinger Bands, it signals that a stock could potentially rise. we may consider buying such stock. When the stock price falls below the lower line, it could signal further decline. we may consider selling the stock instead as shown in Figure 4. To get alerts, we used numpy function “where” to return the position list of the data points exceeding the band. Then we can obtain the subsets of the stock line for plotting with the lists. We also implemented an input function to design a time frame for the Bollinger Band. The end date will be the latest available data. The 5 columns are merged with common axis of ‘date’ for plotting.

Finally, we observed that the number of STD should be set wisely. A small number will result in more frequent triggers of alert than desired.

****

*Figure 4* Result graph sample of Bollinger band analysis.

**Regression (Linear & Polynomial)**

In addition to the two analysis techniques, we wished to perform the regression on the Golden/Death cross as well as the Bollinger band to predict the indicators, but we found multiple problems with applying regression techniques on time series data. The first problem was that the regression would only work on integers and floating point numbers. It would not run with time data. To rectify the problem, the dates were converted into “ticks” starting from 0. It would then increase to the last date. With this method, the regression was successful.

This produced another problem, which is that the regression predictor is now tied to the ticks instead of the dates. Graphically, it would be visually confusing when the stock prices are plotted without the date numbers. Many methods were tried, including toordinal and fromordinal conversion but they failed as well. The most feasible solution was to plot two axes, one with the date labels, and the other with the day count. By hiding the day count to the upper chart and showing the date axis, we can then present an understandable regression chart.

**Challenges encountered and Lessons learned**

**Large File Processing:**  
The data size was larger than we were used to, which was over 1.7 gigabytes in size. Initial sampling attempts failed due to the amount of memory that our laptops have as well as the amount of memory that other programs such as chrome consume. Our solution was to use the library Dask for initial analysis of the entire dataset. Many of the functions were similar to that of Pandas. However, there are functions that would normally work with pandas fail. Much time was spent on learning and debugging the Dask scripts.

**Graphing Problem:**

At first, we were unable to generate the Golden/Death Crosses on the same position with the crosses of Moving Average Lines in the same graph. As shown in Figure 5. The solution was to merge the Golden/Death Crosses data to the data frame of the stock price line and Moving Average Lines, so that the Golden/Death Crosses shown on the right position, as shown in Figure 6. Graphing was also problematic with regression as well.

|  |  |
| --- | --- |
| *Figure 5* | *Figure 6* |

**Conclusion**

We have managed to recreate two basic financial analysis tools to apply to a historical stock dataset. We made it flexible to tune the parameters such as time frame, days of moving averages and the width of the Bollinger Band, which is essential for coping with different interests of the users. Financial analysis was complicated and our work is still far from mature to apply on actual finance decision making. The results of the various analysis tools, including the moving averages and Bollinger bands, should be viewed in bulk and considered altogether. It is highly not recommended to use any single tool as buy or sell signal. For regression, the initial programming was not difficult to accomplish. However, we learned that linear and polynomial regression functions were not suitable with them, it conflicts with the graphing function. ARIMA regression would have been reliable for our project as it handles time series more accurately and can be plotted without issue.

We have gained a lot through the project. Not only consolidating the relative basic python knowledge such as Pandas, Matplotlib Library, etc. that we have learned in the course, but also we further learned on the project about the graphic compiling through Matplotlib Library of python and the application of some other libraries such as Dask, Scipy and Scikit-learn Libraries for the issues of out of memory and regression, etc. We also understand the importance of teamwork and communication when dealing with the challenges, which benefits us for the course’s study in the future.