Technical Report: Final Project
EECE 2560: Fundamentals of Engineering Algorithms

Ryder Paulson and Kevin Li
Department of Electrical and Computer Engineering
Northeastern University
paulson.r@northeastern.edu
li.kevin4@northeastern.edu

December 5, 2024

Contents

1 Project Scope

The aim of this project is to design a system for simulating and forecast the stock market. It will have 3 main components

- Represent the stock market, with classes representing individual shares and tickers with their normal functions.
- An algorithm to forecast the future values of a stock based on its past history.
- Create plots of ticker values as well as their forecast.

The expected outcomes include a functioning desktop-based system, proper documentation, and a final project report.

2 Project Plan

2.1 Timeline

The overall timeline for the project is divided into phases:

- **Week 1 (October 7 - October 13)**: Define project scope, establish team roles, and outline skills/tools.
- **Week 2 (October 14 - October 20)**: Begin development, set up the project repository, and start coding basic system functionalities. By the end of the week make final decision on whether we continue using Qt.
- **Week 3 (October 21 - October 27)**: By the end of the week finish the basic stock functionality. Starting exploring whether to import data or hardcode it.
- **Week 4 (October 28 - November 3)**: Work on importing stock data and begin working on forecasting algorithm. Evaluate whether we should important stock data using an API or just hardcode in values.
- **Week 5 (November 4 - November 10)**: Finalize the forecasting algorithm and begin working on the graphing functionality.
- **Week 6 (November 11 - November 17)**: Finalize the graphing functionality and do the technical report and the PowerPoint presentation.
- **Week 7 (November 18 - November 28)**: Final presentation, report submission, and project closure.

2.2 Milestones

Key milestones include:

- Project Scope and Plan (October 7).
- GitHub Repository Setup and Initial Development (October 10).
- Stock functionality completion (October 28).
- Finalize graphing (November 10).
- Finalize forecasting algorithm (November 17).
- Final Presentation and Report Submission (November 28).

2.3 Team Roles

- **Ryder:** Algorithm Development and MATLAB.
- **Kevin:** Data Import and MATLAB.

3 Methodology

3.1 Equations

The algorithm uses multiple formula that are easier to understand when represented in mathematical form rather than code form. Those have been included here for reference later.

Equation for standard deviation: $\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2}$

Equation for forecasted price: $f = p - 2 * \frac{d}{\sigma} * v * binomial(\sigma, 0.5)$

- f - forecasted price
- p - previous price
- d - deviation
- v - volatility

3.2 Pseudocode and Complexity Analysis

The following is the pseudocode for the broad approach to the regression algorithm.

Algorithm 1: Regression Overview

```
Input: data, n, *company;
Reverse(data);
normalizeXAxis();
standardInterval();
calculateRegressionVariables();
calculateCoefficient();
calculateConstantTerm();
forecastMeanLine();
calculateVolatility();
standardDeviation();
forecastFutureValues();
```

The algorithm begins by pre-processing the data. The import functions imports it in reverse order so the data is reversed within the regression. The normalizeXAxis() function subtracts the first x value from every time value. This makes it so that the data starts at x = 0. This is essential for the regression to be correct. The standard interval function calculates the standard interval which refers to the average difference between two data points. This is approximately a day but also includes weekends and holidays when markets are not open. That means the actual value is over a days time. This value is used to forecast future values. The regression then calculates all of the variables that are necessary to calculate the coefficient and constant of the linear regression. These consist of various combinations of the sum of x and y data points. The regression then calculates the variables of the regression and forecasts the mean line out the amount of standard intervals the user chooses. All of these functions progress linearly through the data. This means the time complexity of this part of the algorithm is $O(n)$.

The algorithm then calculates the unique variables which are used when forecasting future values. The first one of these variables is the volatility. The volatility is the average rate of change of the price. It is used to scale the rate at which the predicted values approach the mean line. Next, the algorithm calculates the standard deviation (σ). Finally, it enters the forecastFutureValues() function which calculates the forecasted values. The pseudocode for this function is shown below.

Algorithm 2: Forecast Future Values

Input: placeholder;
Output: placeholder;
SharePrice sp;
for $i = 0$ *to* $i < n$ **do**
  $sp.stdt \leftarrow meanLine[size - 1 + i].stdt$;
  $deviation \leftarrow previousPrice - meanLine[size - 1 + i].price$;
  $change \leftarrow 2 * deviation \div \sigma * volatility$;
  $sp.price \leftarrow previousPrice - addRandomness(change)$;
  $forecast.append(sp)$;
end

In forecastFutureValues(), a placeholder SharePrice is made to store new values. For each of the forecasted values it calculates the deviation, or difference between the previous value and the mean line. It then calculates the initial change which will then be randomized using the addRandomness function below.

Algorithm 3: Randomizer

Input: placeholder;
Output: placeholder;
binomialDistribution binomial(σ, 0.5);
bernoulliDistribution bernoulli(0.5);
$price \leftarrow price * binomial()$;
$upOrDown = bernoulli()$;
if $upOrDownis1$ **then**
  $return\ -1 * price$;
end
$return\ price$;

The addRandomness() function begins by generating an object representing a binomial distribution with σ trials with a 50% chance of success. it also creates a bernoulli distribution centered at 0.5. It scales the price by a value received from the binomial distribution. This was added to the algorithm because a function that just approaches the mean line is not a good representation of actual stock values. In reality, the amount of variables making up the price of a stock are so plentiful that in the short term the movement of a stock is essentially random. The bernoulli distribution was added so that 50% of the time the price goes up and 50% of the time it goes down. This is only used as an estimation of what the stock fluctuation would look like. If there are multiple value in the row that skew the forecasted price one direction, the non-random part of the algorithm should work to scale it back towards the mean line.

The overall time complexity of the algorithm is $O(n + m)$, where n is the size of the input data and m is the number of values to be forecasted. The time complexity is linear because every step of the pre-forecasting algorithm is linear according to the input data and every aspect of the forecasting portion is linear according to the numbers of values being forecasted.

3.3 Data Input & Output

Historical data was obtained from marketwatch.com and downloaded in CSV format. The file contained information about the history of a companies stock price spanning one year. The fstream library was then used to handle file I/O. For each day, the date and closing price was parsed into a SharePrice object, A vector of SharePrices was then stored in a Company class. After the algorithm is finished forecasting values, it outputs them to a CSV file (output.csv) and calls a MATLAB script which plots from the CSV file.

4 Results

Reference Appendix B for a collection of graphical outputs. The company for which these forecasts originate is in the title of the plot. Note that, because the algorithm has a randomizing element, the same collection of data can receive varying forecasts. Additionally, some outlier outputs have been included to demonstrate some of the flaws of the algorithm.

5 Discussion

As shown in Figures 1, 2, and 4, this program is able to generate fairly accurate predictions. However due to random values, some forecasts can result in significant outliers. This is apparent in Figures 3 and 5. In these cases, as the program predicts prices further into the future, the outliers become more significant and the predicted values deviate more and more from the trend line. This highlights a limitation of the program: while randomness is necessary to replicate the unpredictable nature of stock prices, the lack of constraints or corrective mechanisms sometimes leads to extreme deviations.

6 Conclusion

This project successfully implemented a program for forecasting stock prices using historical data and a combination of regression and randomization techniques. The generated plots demonstrate that the algorithm is capable of forecasting values that align with historical trends. However, due to the generation of random values, some of these plots depict significant outliers. This is especially apparent the further into the future the program is asked to forecast. Potential solutions may include some sort of filtering/weighting system or the inclusion of additional factors such as historical volatility trends or web scraping for trending headlines related to a company. With these additions and further refinements, this project could potentially be a very useful tool for predicting stock prices.

7 References

Bachelier, Louis, et al. *Louis Bachelier's Theory of Speculation: The Origins of Modern Finance.* Princeton: Princeton University Press, 2006.

Exante Data. "This is the 6th post in a series..." *Twitter,* 14 July 2023, https://x.com/ExanteData/status/1679788970262102020

"How Can You Calculate Linear Regression?" *Voxco,* www.voxco.com/blog/how-can-you-calculate-linear-regression/.

Muller, Derek. "The Trillion Dollar Equation." *YouTube,* Veritasium, 27 Feb. 2024, www.youtube.com/watch?v=A5w-dEgIU1M. Accessed 28 Feb. 2024.

NIST/SEMATECH e-Handbook of Statistical Methods, http://www.itl.nist.gov/div898/handbook/, 17 November 2024.

"Regression Analysis and the Best Fitting Line Using C++." *GeeksforGeeks*, 28 July 2022, www.geeksforgeeks.org/regression-analysis-and-the-best-fitting-line-using-c/.

Weiss, Angela. "US-ECONOMY-NYSE," *Getty Images,* 31 Jan. 2024, www.gettyimages.com/detail/news-photo/exterior-view-of-the-new-york-stock-exchange-at-wall-street-news-photo/1968238334?adppopup=true.

A Appendix A: Code

All relevant code for this project is available on GitHub:
`https://github.com/RyderPaulson/EECE2560_Final_Project`

B Appendix B: Figures

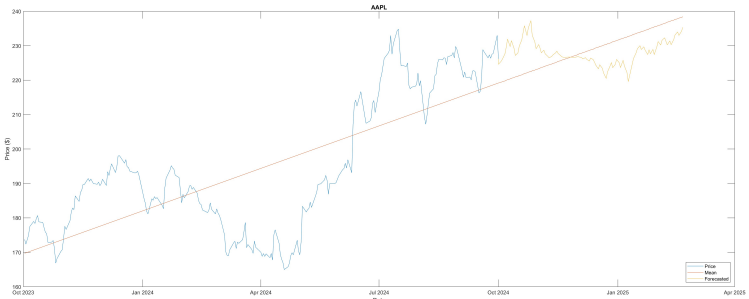

Figure 1: A plot with a realistic prediction of the future price of Apple stock.

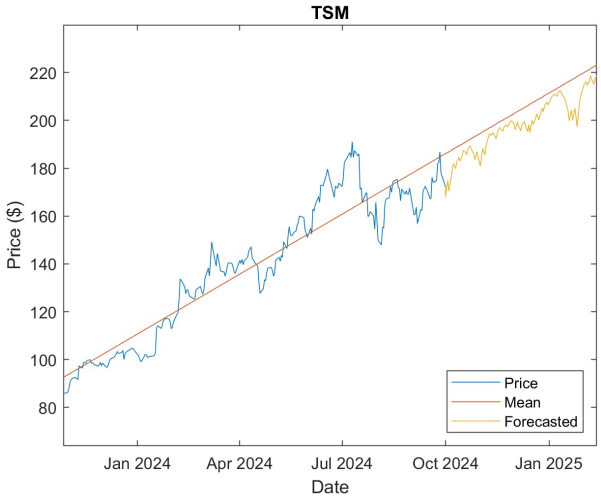

Figure 2: Stock forecast or TSMC. This output is a realistic prediction of what the stock's value may be.

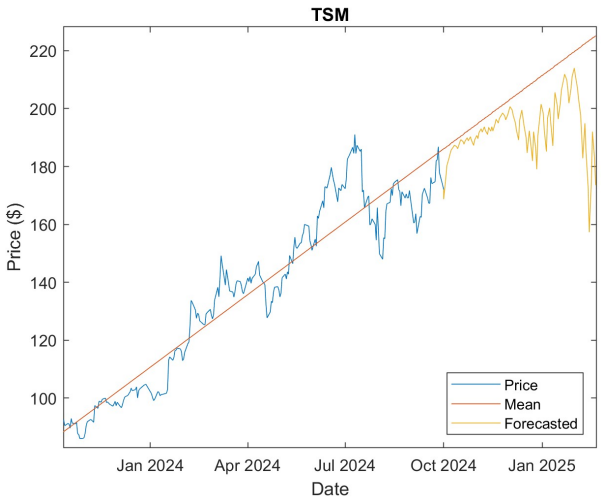

Figure 3: Stock forecast for TSMC. This is an example of an outlier. The value deviates from what would be expected towards the end of the forecast.

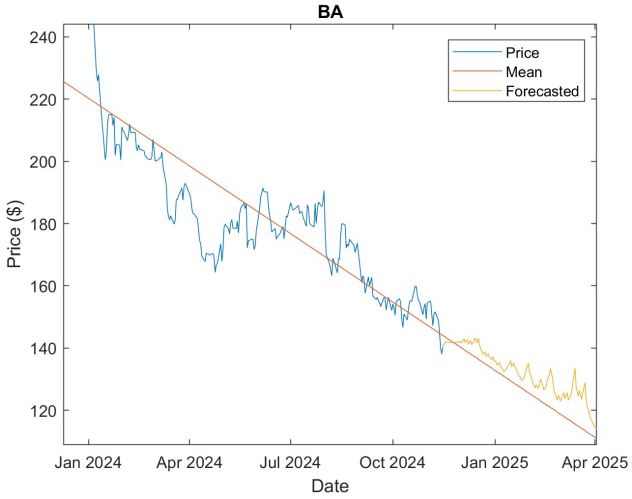

Figure 4: A realistic prediction for the stock value of Boeing. This example demonstrates that the algorithm is able to produce good results for a bearish stock.

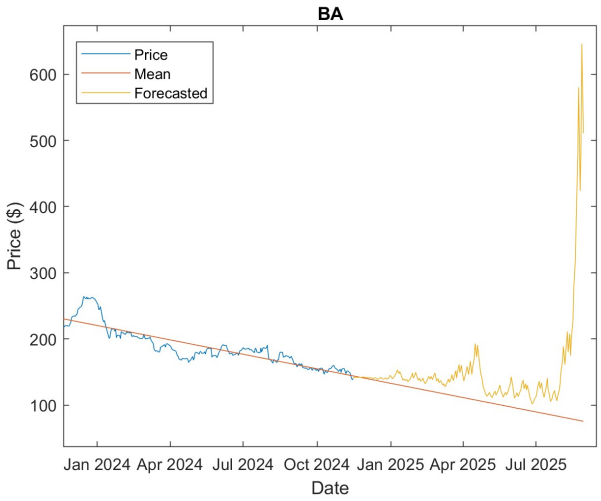

Figure 5: This is a significant outlier example. The algorithm deviates wildly off of what would be expected of the stock price.