Linear Prediction with Real Data

Owen Sowatzke

Georgia Institute of Technology
School of Electrical and Computer Engineering

ECE 4271
Spring 2021

Introduction

The goal of this project was to design and evaluate linear predictors using MATLAB. Specifically, this project examined linear predictors in applications involving the stock market and Ethereum block difficulty. This document details the results of each of these analyses and is divided into two sections that investigate prediction of the stock market and Ethereum block difficulty respectively.

Part I

The linear prediction of the stock market that follows parallels exercises in the Buck et al. textbook. 91 years of weekly stock market data from Oct. 1, 1928 to Feb. 11, 2019 is used to design linear predictors and then analyze the resulting predictions.

(a)

The Dow Jones Industrial Average Data is plotted on a linear and semi-logarithmic scale in Figures 1 and 2 respectively.

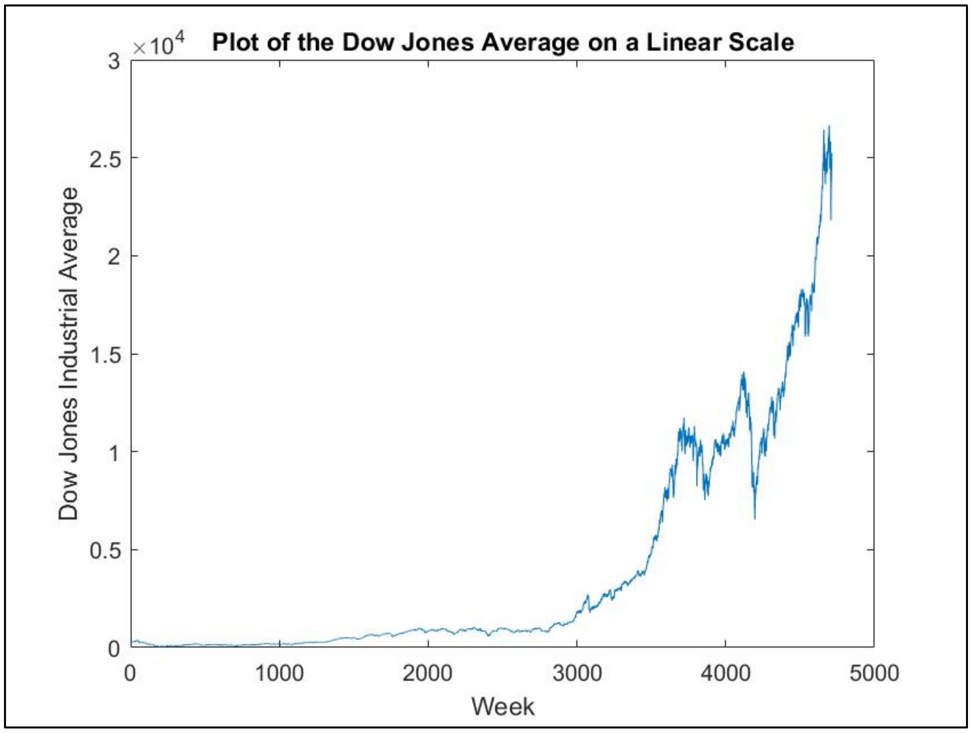

Figure 1 – Plot of the Dow Jones Industrial Average on a Linear Scale.

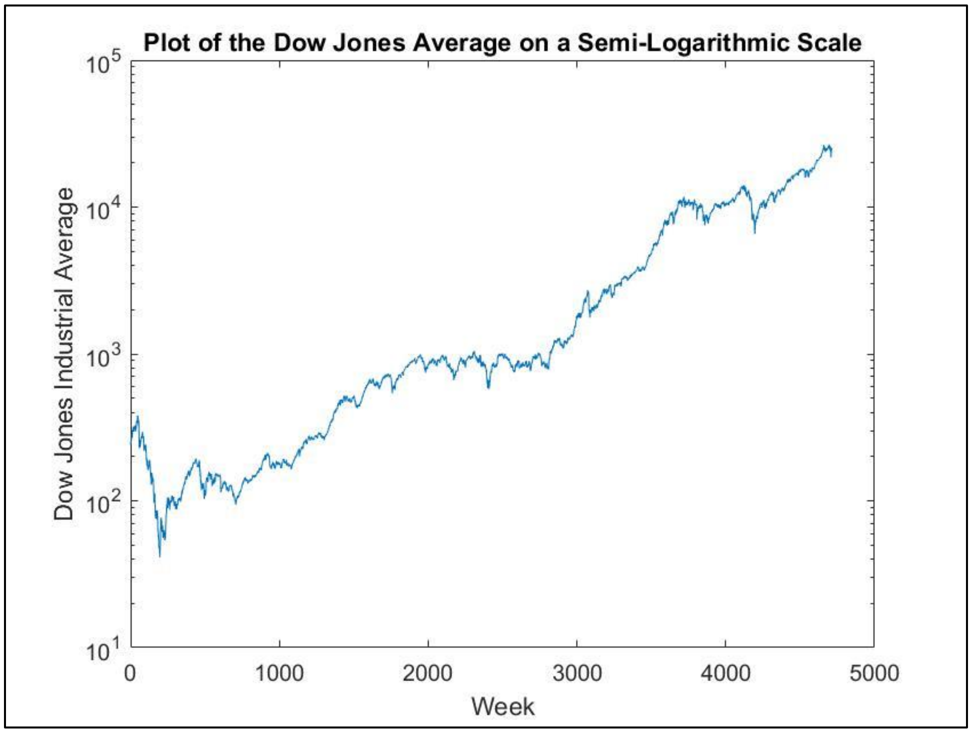

Figure 2 - Plot of the Dow Jones Industrial Average on a Semi-Logarithmic Scale.

Assuming that we start with \$1000 and invest all our money in the DJIA, we would have \$104,196.93 at the end of investment interval (4714 weeks). If we decided to instead put all our money in the bank at %3 APR, it would take 8056 weeks to make the same amount of money. If we wanted to make the same amount of money over the same time interval, we would need a 5.13% APR.

(b)

Using $p = 3$ and $N = 520$, we can solve for the vector a that minimizes the inner product: $e' * e$. Using the MATLAB's \ operator, we get the following results:

$$a = -X\backslash x = \begin{bmatrix} 0.0268 \\ 0.0938 \\ -1.1183 \end{bmatrix}$$

(c)

Using $xhat1 = -X * a$ and the $filter$ command, we can predict values for the first decade of decade. Figure 3 displays the predictions on the same set of axes as the actual weekly average.

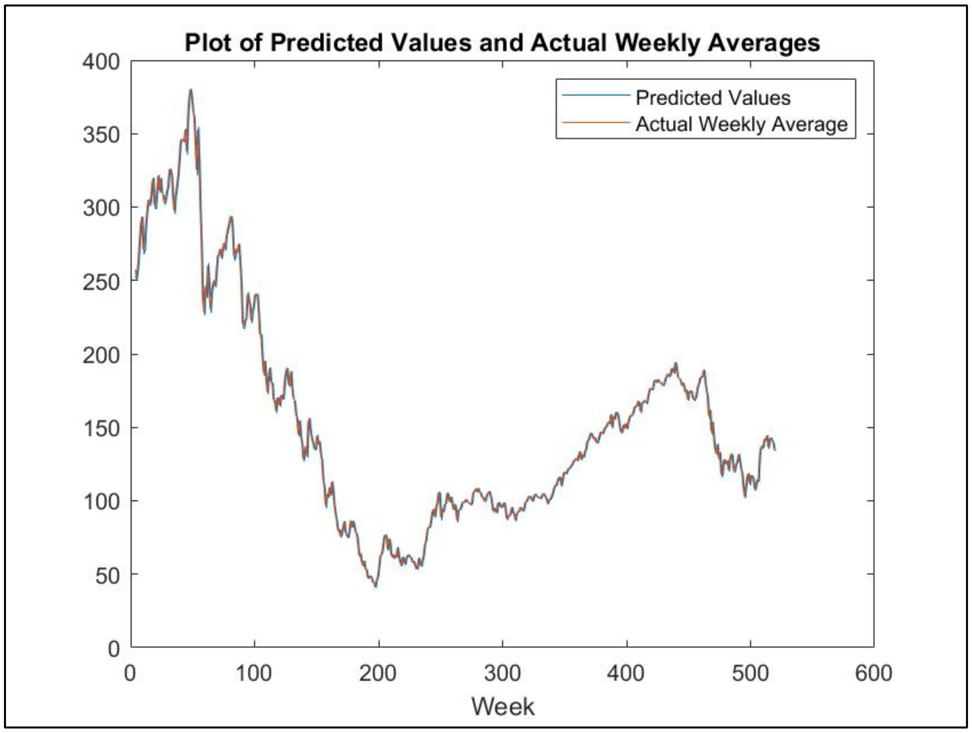

Figure 3 – Plot of the Predicted and Actual Weekly DJIA Data over the First Decade.

Using $e = x + X * a$, we can calculate the total squared error according to the following formula: $E = e' * e$. This gives us the following result:

$$E = 23{,}638.06$$

To check the results, we can compute e according to the following formula: $e = x - xhat2$. Then, we can calculate E using the formula above. Doing so, gives us the following result:

$$E = 23{,}638.06$$

(d)

In Figure 4, we plot the total squared prediction error as a function of p for $p = 1, \dots, 10$.

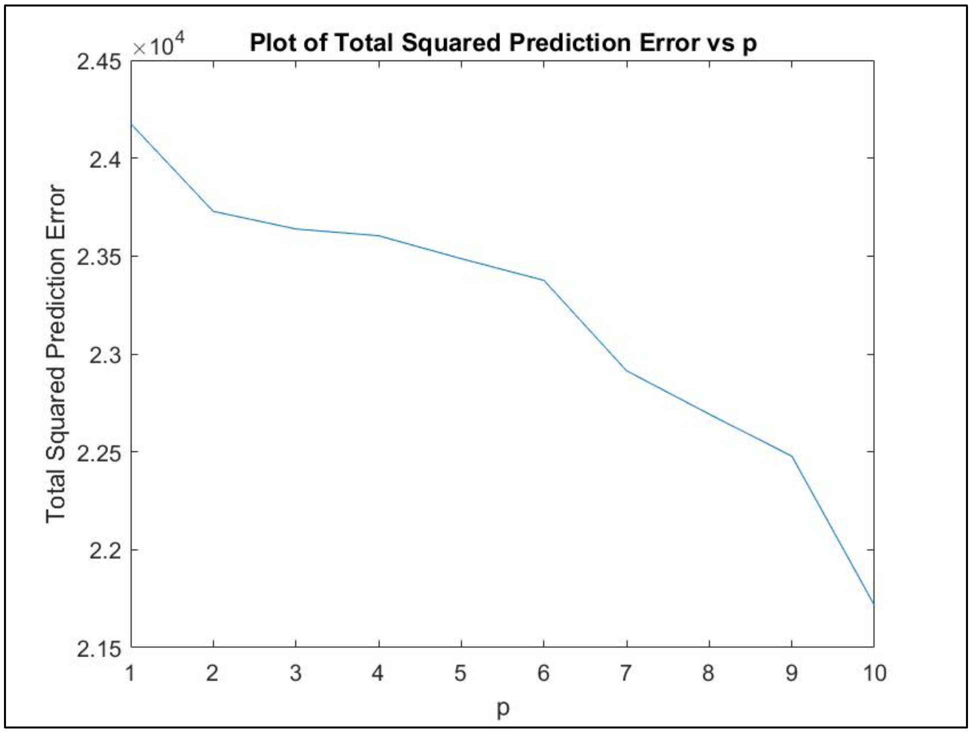

Figure 4 – Plot of the Total Squared Error vs p for $p = 1, \dots, 10$.

There appears to be a "knee" at $p = 2$. However, the total squared prediction error drops steeply starting at $p = 6$. Therefore, there is not a value of p after which the decrease in prediction error is negligible. As such, we choose $p = 10$ because it minimizes the prediction error.

(e)

If we have $1000 at the start of the p-th week and make 520 trading decisions, the upper bound to how much we could make is $4,700,565.98 - $1000 = $4,699,565.98. One lower bound to how much we can make is the amount we can make if all the money is invested in the bank. If this is the case, we can make $1349.74 - $1000 = $349.74. Another lower bound to how much we can make is the amount we can make if all the money is invested in the DJIA. If this is the case, we can make $544.44 - $1000 = - $455.56. Using the linear predictor, we can make $1422.65 - $1000 = $422.65. In order to make this same amount of money in the bank, an APR of 3.53% is needed.

(f)

Using the linear predictor coefficients derived in (e), we can use the same prediction strategy on the most recent decade of data. The upper bound to how much we can make is $104,798.92 - $1000 = $103,798.92. By investing the $1000 in the bank, we can determine one of the lower bounds, which is $1349.74 - $1000 = $349.74. If we instead invest the $1000 in the stock market, we can determine the other lower bound, which is $3521.28 - $1000 = $2521.28. Using our linear predictor, we can make $2265.53 - $1000 = $1265.53. This is equivalent to an APR of 8.18%.

Part II

In this part, we use the autocorrelation method to predict Ethereum block difficulty. Specifically, we use daily data from July 30, 2015 to February 11, 2019 to make predictions and then analyze the resulting predictions.

(a)

In part (a), we use block difficulty data from July 30, 2015 to December 31, 2015 to predict difficulty from January 1, 2016 to June 30, 2016.

(i)

First using $P = 2$, we can use the lpc command to determine the linear predictor coefficients:

$$a = \begin{bmatrix} -1.0378 \\ 0.0442 \end{bmatrix}$$

In Figure 5, we plot the predicted difficulty and real difficulty on the same set of axes.

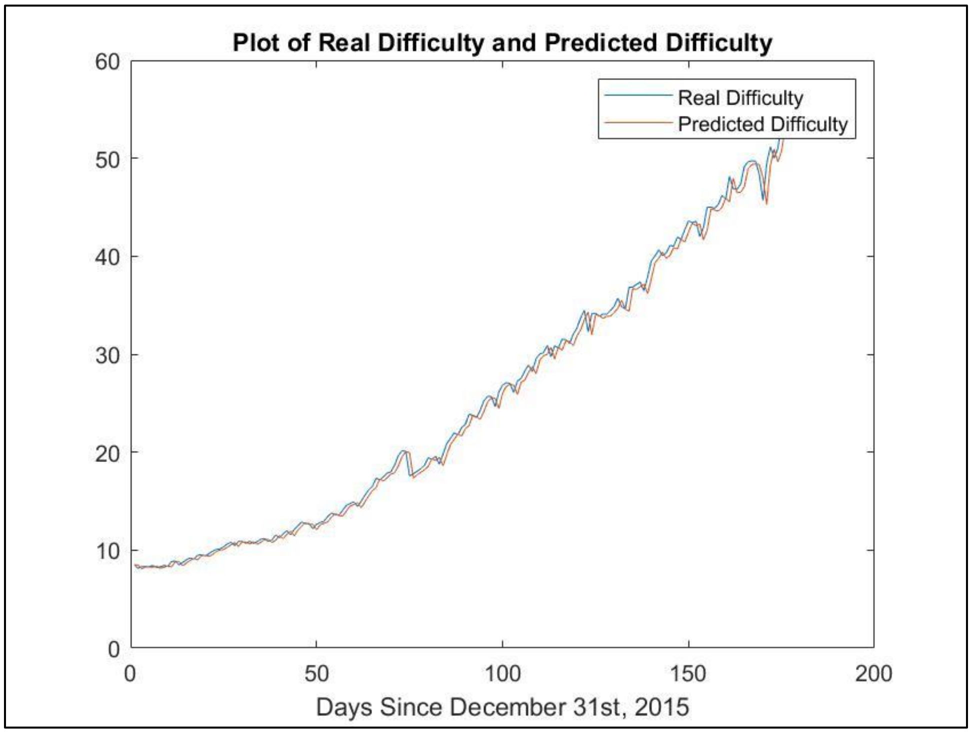

Figure 5 – Plot of the Predicted Difficulty and Real Difficulty from January 1, 2016 to June 30, 2016.

(ii)

The least squares error E is defined according to the following formula:

$$E = \sum_{n=0}^{L+P-1} e^2[n] = \sum_{n=0}^{L+P-1} (x[n] - \hat{x}[n])^2$$

where $x[n]$ is the training data sequence. In Figure 6, we choose $P = [2:4:50]$ and plot the least squares error E versus p.

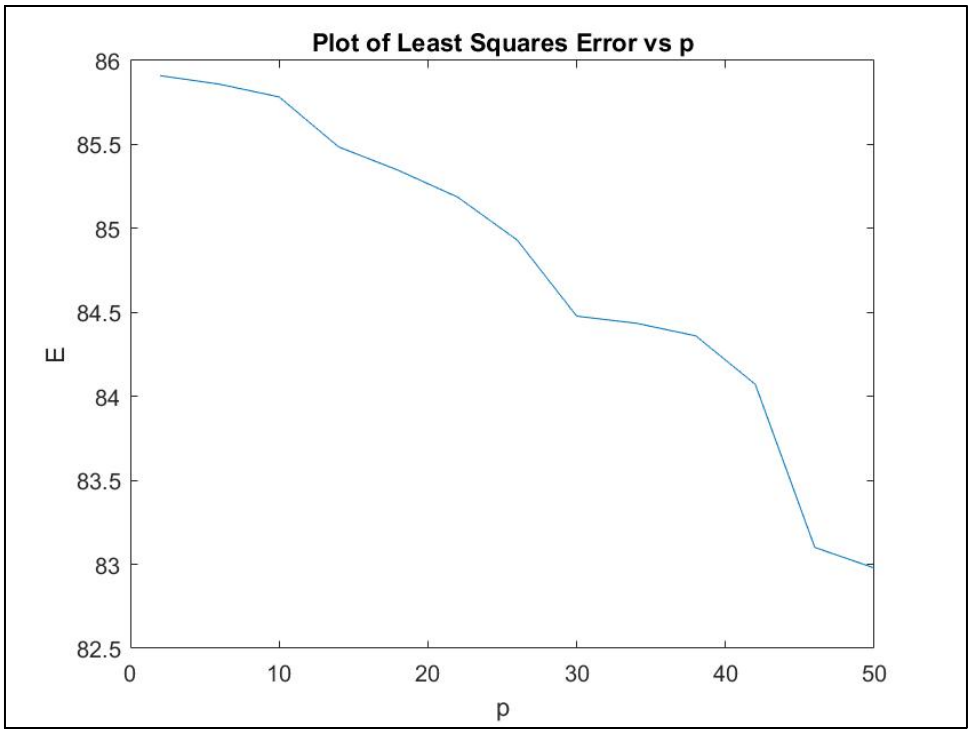

Figure 6 – Plot of the Least Squares Error vs p.

(iii)

The average predicted error is calculated according to the following formula:

$$(\sum(\text{predicted difficulty} - \text{real difficulty})^2)/(\text{total number of predicted days})$$

In Figure 7, we plot the average predicted error for $P = [2:4:50]$.

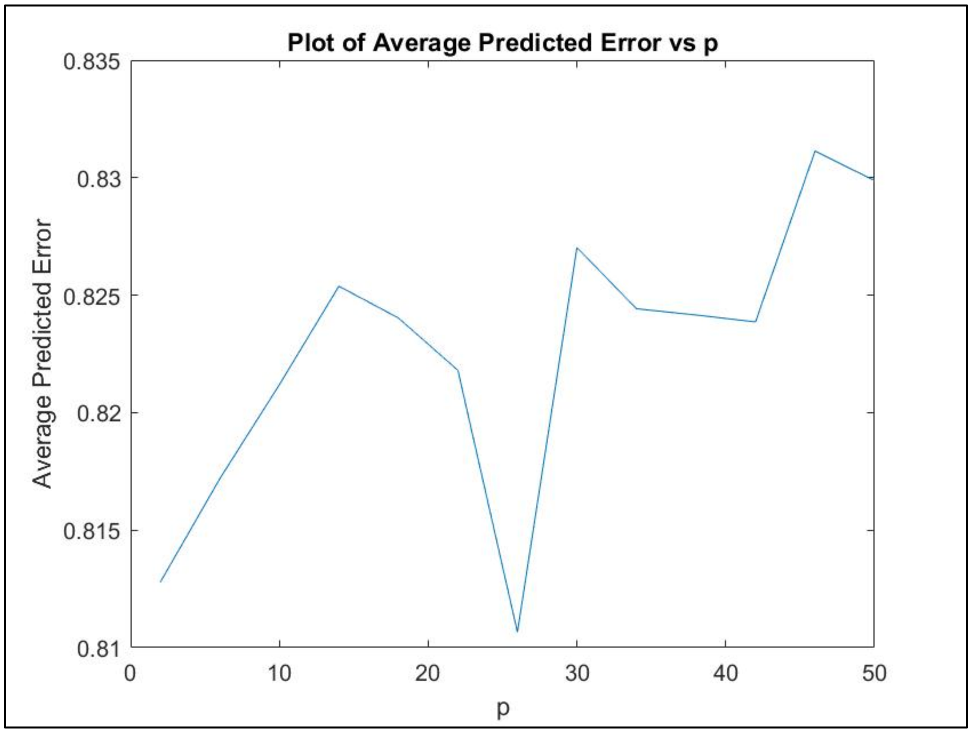

Figure 7 – Plot of the Average Predicted Error vs p.

Comparing Figure 6 and Figure 7, we note that the plots are different from one another. This occurs because the least squares error is determined from the 2015 training data, and the average predicted error is determined using the 2016 predicted data. The least squares error approaches zero as p increases. This makes sense because the number of unknowns approaches the number of equations. On the other hand, note that the average predicted error does not follow the same trend. This occurs because the 2015 and 2016 data are different from one another. Too many coefficients can weight the predicted data off potentially irrelevant data, and too few coefficients prevents us from training our predictor well.

(b)

In part (b), we use block difficulty data from January 1, 2016 to December 31, 2016 as training data to predict difficulty over various date ranges. Then we compute the average predicted error for each date range.

(i)

In part (i), we use the predictor to predict difficulty from January 1, 2017 to December 31, 2017. In Figure 8, the predicted block difficulty is plotted on the same set of axes as the actual block difficulty.

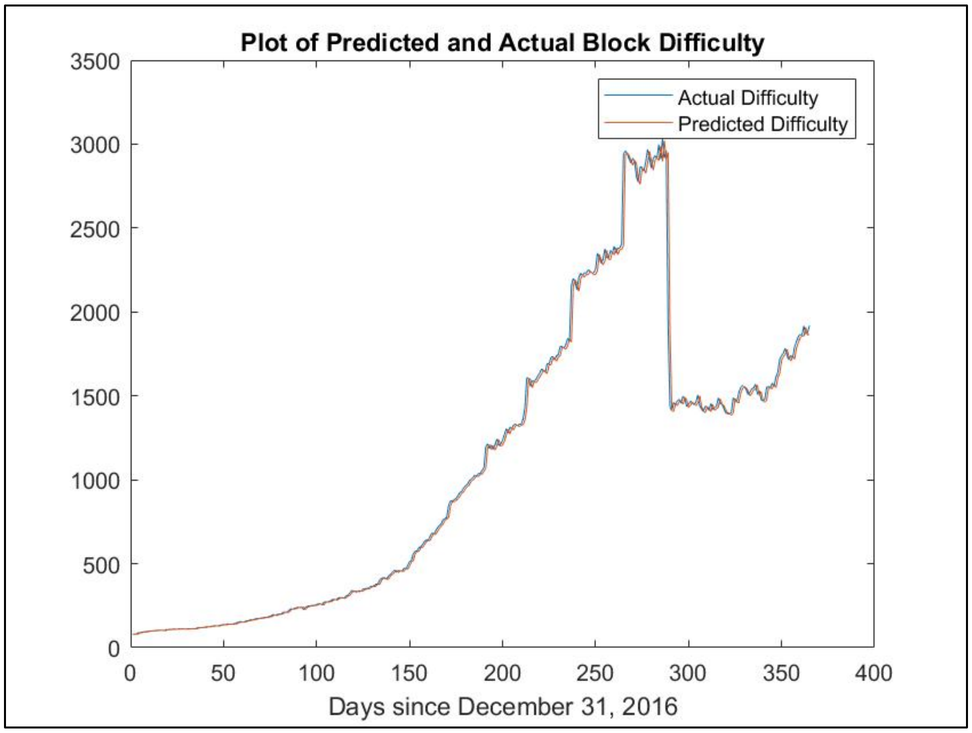

Figure 8 – Plot of the Predicted Block Difficulty and Actual Block Difficulty from January 1, 2017 to December 31, 2017.

(ii)

In part (ii), we use the predictor to predict difficulty from January 1, 2018 to December 31, 2018. In Figure 9, the predicted block difficulty for the 2018 data is plotted on the same set of axes as the actual block difficulty.

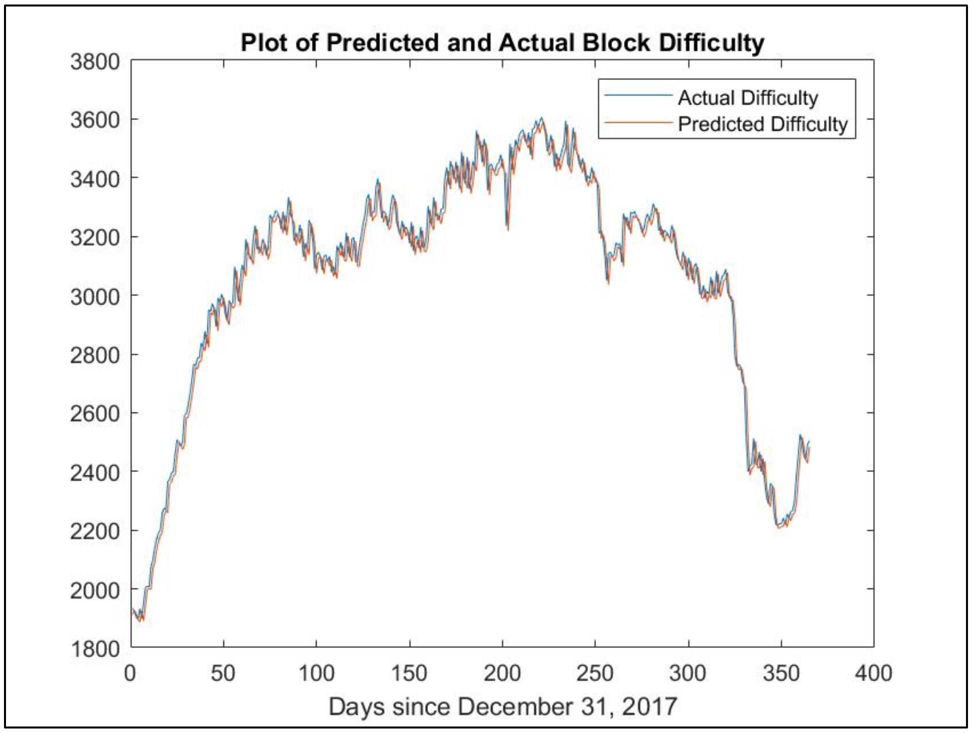

Figure 9 - Plot of the Predicted Block Difficulty and Actual Block Difficulty from January 1, 2018 to December 31, 2018.

(iii)

In part (iii), we compute the average predicted error between the actual and predicted block difficulty for the 2017 and 2018 data respectively. Using the definition of averaged predicted error, we determine that the average predicted error is 5531.24 for the 2017 predictions and 3049.19 for the 2018 predictions. The average predicted error is greater for the 2017 predictions because the predictor has not been trained for the steep drop, as observed in Figure 8.

(c)

In part (c), we predict the difficulty data from January 1, 2018 to June 30, 2018 using a subset of the previous data.

(i)

In part (i), we use the previous year (365 days) of data to train our predictor. Using this predictor, we plot the predicted difficulty vs the actual block difficulty from January 1, 2018 to June 30, 2018. This plot is displayed in Figure 10.

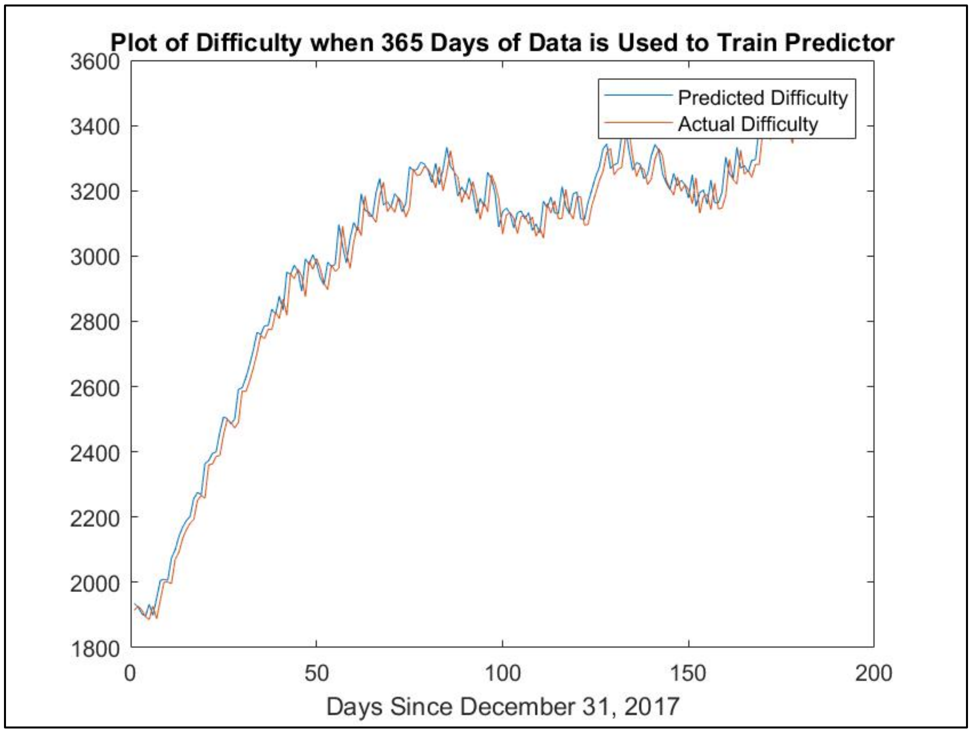

Figure 10 – Plot of the Predicted Difficulty and Actual Difficulty when 365 days of data is used to train the predictor.

The average predicted error using 365 days of training data is 3012.30.

(ii)

In part (ii), we use the previous 6 months (180 days) of data to train our predictor. Using this predictor, we plot the predicted difficulty vs the actual block difficulty from January 1, 2018 to June 30, 2018. This plot is displayed in Figure 11.

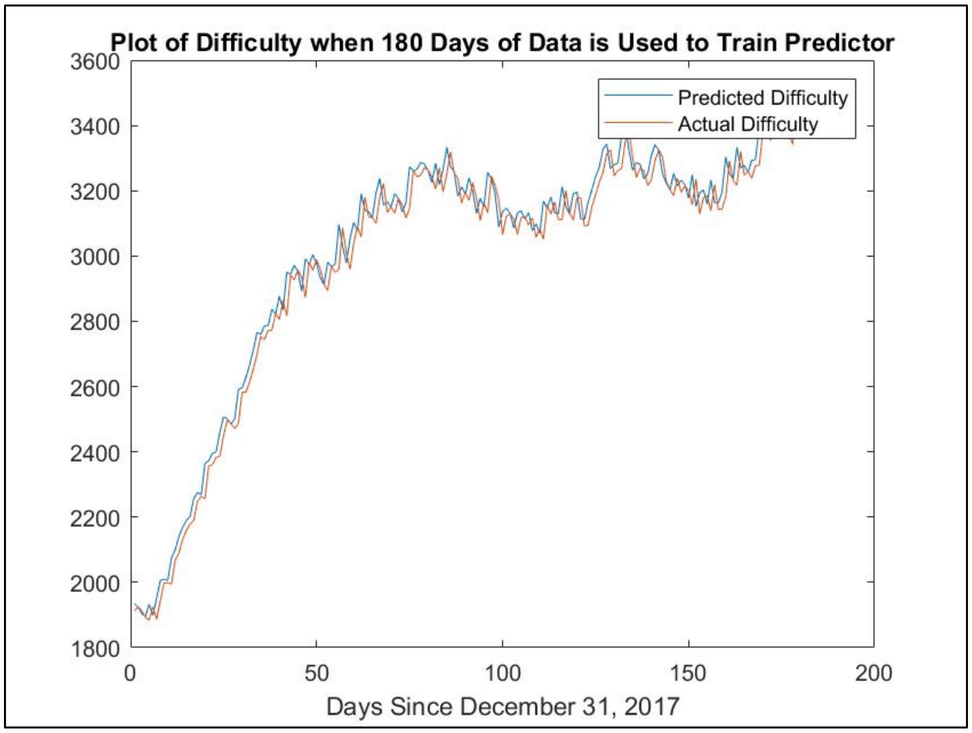

Figure 11 – Plot of the Predicted Difficulty and Actual Difficulty when 180 days of data is used to train the predictor.

The average predicted error using 365 days of training data is 3135.98.

(iii)

In part (iii), we use the previous month (30 days) of data to train our predictor. Using this predictor, we plot the predicted difficulty vs the actual block difficulty from January 1, 2018 to June 30, 2018. This plot is displayed in Figure 12.

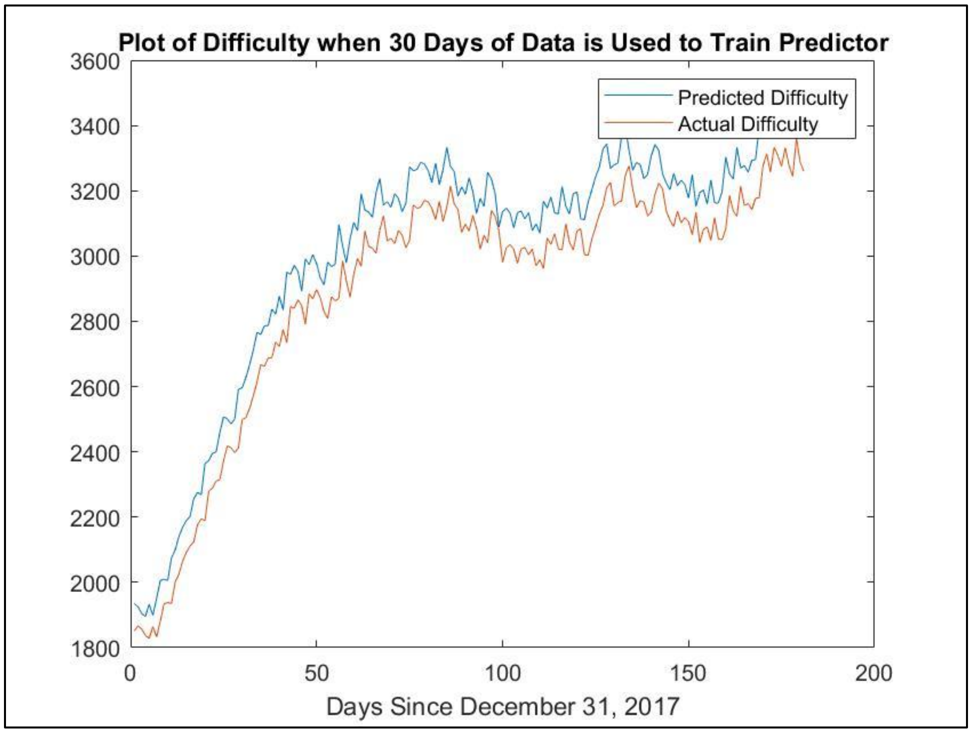

Figure 12 – Plot of the Predicted Difficulty and Actual Difficulty when 30 days of data is used to train the predictor.

The average predicted error using 365 days of training data is 15578.56.

Comparing the average predicted error in all three cases, the average predicted error is smallest when the most days of training data is used. Note that the average predicted error increases more going from a 6-month to a 1-month training data set than going from a 12-month to a 6-month training data set. In other words, the average predicted error increases most substantially as the training data set gets "very small."

Conclusion

The previous sections investigated how linear predictors could be used to predict stock market values and block difficulty values. In the case of the DJIA predictor, we found that the linear predictor could be used to make investing choices that increased gain, while minimizing the risks of investing in the stock market. In both prediction cases, we observed how the predicted error decreased with increasing values of p. When using the linear predictor to predict block difficulty, we observed a "knee" or best choice for p. Another conclusion we can draw from the block difficulty predictor is that a larger set of training data improves the predictor results as observed in part 2 (c).