# Predictive Modeling for Buying and Selling Bitcoin

Andrew Struthers[1] Evan Nordin [2]

Central Washington University

## Abstract

Investable assets are frequently targeted by the general public in an attempt to turn some money into more money. One of the most volatile investable assets is Bitcoin. In September 2017, the value of one Bitcoin reached $\$20,000$ before crashing to around $\$6,000$ in December 2017. Many investors are attracted to Bitcoin due to this volatility. In this project, we will be attempting to use numerical modeling methods to analyze the price of Bitcoin over time. The goal of modeling the price of Bitcoin will be to form a system of predicting when it will rise and fall. This will allow us to write software that automates the buying and selling of Bitcoin to maximize the value of our trades.

## Numerical Modeling Methods

We will be using a few different types of numerical methods with the goal of creating a system that can accurately predict when the price of Bitcoin will rise and when it will fall. The primary method we are using is called numerical differentiation. This method allows us to find the rate of change of a dataset without needing a symbolic function. The second method we are using is creating various interpolating polynomials, using the Lagrange method and Spline method. While interpolating polynomials aren't particularly useful at predicting behaviour outside of our dataset, they provide information about what happens between distinct points in our data.

## Dataset

We gathered the price of Bitcoin from 5/1/2022 at 8:00 am to 5/7/2022 at 4:00 am on two hour intervals for our dataset, resulting in the following graph:

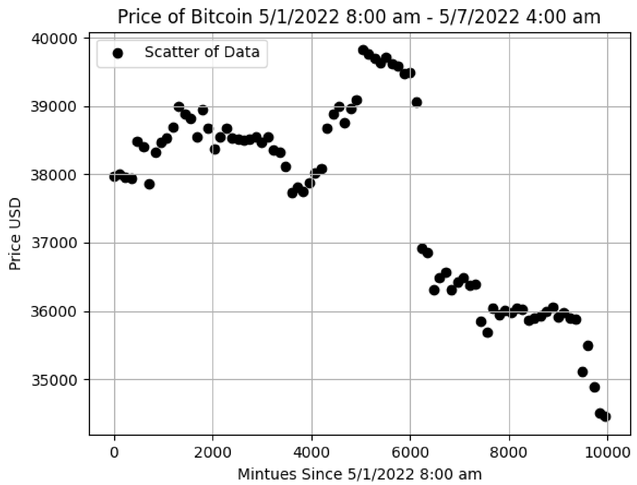


Figure 1. Price of Bitcoin

As we can see on the graph, the price of Bitcoin was anywhere between nearly $\$40,000$ and almost $\$34,000$. This volatility is what attracts investors bold enough to take the risk of investing.

## Interpolation

An interpolation is a function that must exactly equal each point in the data. For example, if we had the point $(240, 37, 950.89)$, any interpolating polynomial will go through that point.

### Lagrange

The equation for an $n^{th}$ degree Lagrange polynomial can be found below.

$$P(x) = \frac{(x-x_1)(x-x_2)...(x-x_n)}{(x_0-x_1)(x_0-x_2)...(x_0-x_n)}y_0 + ... + \frac{(x-x_0)(x-x_0)...(x-x_{n-1})}{(x_n-x_1)(x_n-x_1)...(x_n-x_{n-1})}y_n$$

Our data contains 84 points, so we would need a polynomial with 84 terms. This would result in an equation that at most behaves like $f(x) = x^{83}$, which would go to $\pm\infty$ almost instantly after the end of our data. Since the price of Bitcoin is unlikely to spontaneously go to $\infty$, this polynomial is not useful for predicting the future value.

### Cubic Spline

Spline interpolation is different than Lagrange interpolation mainly because instead of one equation that interpolates all the data, there are $n-1$ cubic polynomials that each interpolate only two points. For our dataset, we would get $83$ different cubic polynomials that interpolate our data. This method doesn't help us predict the behaviour outside of the dataset either, unfortunately, but it did spark the idea that we might be able to look at the rate of change of the data and predict whether the price will rise or fall in the short term future.

## Numerical Differentiation

Using Taylor's Method and algebra, we can derive three formulas for calculating the derivative of a function from just function values. This is useful for calculating the rate of change of data when no symbolic function can be found. The three equations below represent the Five Point Forward Difference, Five Point Midpoint, and Five Point Backward Difference equations for calculating the first derivative of a function.

$$f'(x) = \frac{-25f(x) + 48f(x+1) - 36f(x+2) + 16f(x+3) - 3f(x+4)}{12h} \quad (1)$$

$$f'(x) = \frac{f(x-2) - 8f(x-1) + 8f(x+1) - f(x+2)}{12h} \quad (2)$$

$$f'(x) = \frac{-25f(x) + 48f(x-1) - 36f(x-2) + 16f(x-3) - 3f(x-4)}{-12h} \quad (3)$$

Using these equation, we can plug in our data and find the rate of change of the price of Bitcoin at any known point. Below is a subsection of the calculated derivatives for our data on the interval of $x = 3000$ to $x = 4080$:

| Time (Minutes) | Price of Bitcoin | First Derivative | Second Derivative |
|---|---|---|---|
| 3000 | 38464.2 | 0.126944 | -0.00375075 |
| 3120 | 38554.2 | -1.75724 | -0.0157015 |
| 3240 | 38350.3 | 3.16472 | 0.0410163 |
| 3360 | 38322.9 | -5.82652 | -0.074927 |
| 3480 | 38112.6 | 3.77713 | 0.0800304 |
| 3600 | 37727.9 | -5.24359 | -0.0751727 |
| 3720 | 37811 | -3.14029 | 0.0175275 |
| 3840 | 37748 | 5.74478 | 0.0740422 |
| 3960 | 37886.6 | -5.45559 | -0.0933364 |
| 4080 | 38026.2 | 4.96343 | 0.0868252 |

Table 1. Derivative Calculations

## Results and Predictions

Using these methods, we developed a strategy to predict when to buy or sell Bitcoin throughout the week. We noticed that if the derivative of the price went from negative to positive, the price was at a valley and was likely to go up in the short term future. Likewise, if the derivative went from positive to negative, the price was likely at a peak and was about to drop in the short term. We also factored in the magnitude of the change between positive and negative or vice versa, to come up with a tolerance level. If the derivative went from negative to positive with a high enough magnitude, then we would expect the price to continue going up into the future, and we should buy. Likewise, if the derivative went from positive to negative with a high enough magnitude, we expected the price to continue to fall.

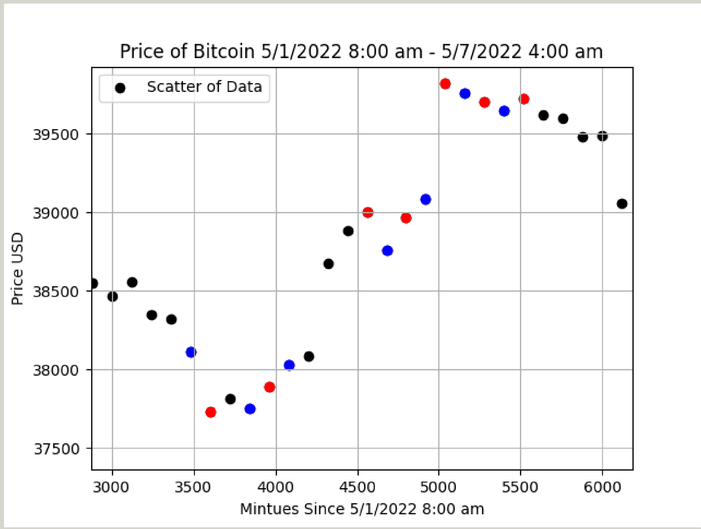


Figure 2. Spline Approximation of Data

The figure above is just a subset of our data, zoomed in for clarity. The blue dots represent the times the code simulated a "Buy" of one Bitcoin, and the red dots are the times the code simulated a "Sell". We can see that the transactions between around $x = 3750$ and $x = 5000$ all made money, since one Bitcoin was bought at a lower price than it sold for. This algorithm isn't perfect, however, because the first set of transactions near $x = 3500$ and all transactions after the sell at $x = 5000$ sold for a loss. Over the span of the entire week, with the code simulating transactions with Bitcoin according to our method, we would have made a total of approximately $\$1,500$, which is surprising because in that same time frame Bitcoin fell approximately $\$3,500$.

## References

[1] R.L. Burden and J.D. Faires. *Numerical Analysis*. Cengage Learning, 2010.

[2] J.F. Epperson. *An Introduction to Numerical Methods and Analysis*. Wiley, 2013.