Bitcoin Trade Advisor and Market Predictor

James Marini

December 2024

Abstract

Bitcoin as a type of cryptocurrency that changes daily based on news trends, the global economy, and most importantly people's buying and selling of the coin. At the time of writing this, one bitcoin is worth roughly 96,000 USD, but the price of the coin was not always this high. The coin's worth grew linearly over the past 10 years, which is the data that we used to analyze and create our tool.

1 Objective

We originally aimed to make a tool that could model Bitcoin prices on given days based on historical data, and possibly even data from news sources that day that might impact the price of the coin. We also wanted to create a logic that would tell a user to prepare to buy, sell, or hold their investments in the coin depending on what that price might be.

We were able to find a CSV file online that contained daily information about Bitcoin (open price, close price, high, low, percent change, volume, etc.) for whatever time range we seemed fit. For proof of concept and conservation of computational resources, we chose a time range of 10 years, which encapsulates the linearity of the growth of Bitcoin over time.

2 Techniques and Tools

The entirety of our project lives in Python, as it is regarded as one of the best tools to use to deal with machine learning algorithms. Although

we did not cover machine learning in depth during our class time, only a basic knowledge of splitting data into learning and testing sets is required to understand how prices were predicted [1]. The package "sklearn" allows users to split a Pandas DataFrame into a "training set" and a "testing set" and set a threshold for what percentage of the data goes into which set. We did a 50/50 split here when predicting our prices.

We first imported the historical CSV file into Python and then organized it appropriately for our analysis and predictions. This required us to change some string fields to "float" and "datetime" variables to be used in the predicting and computing of the moving average.

We then implemented an algorithm to compute this moving average depending on the users input of how many days they want the average to compute over. If the price of Bitcoin on a given day is significantly less or greater than this moving average, then the user will be advised to buy or sell their coin balance, respectively. This was done using basic for loops and if statements, and a user input of these buying and selling percent differences are required to be used in this logic.

Moving onto the modeling, we used a "LinearRegression()" model from sklearn to predict the prices randomly throughout the dataset. The dataset was split randomly into two groups; a training group and a testing group. The training group took half of the dataset (and it's features/variables) and learned from the trends that lead to the price. The testing set had its prices removed, stored into a different Pandas series variable, and had new prices predicted based on the same features that the training set learned off of. The accuracy of the model's ability to predict is determined via an R^2 value, which can be simply derived in Equation 1:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{1}$$

After predicting prices, we implemented the same logic that advised a user on when to buy and sell on this new data set and analyzed our results in the same way.

3 Observations and Results

We saw an R^2 value of 0.999 when predicting the prices, which is alarmingly high at a first glance. This data is extremely linear when looking at

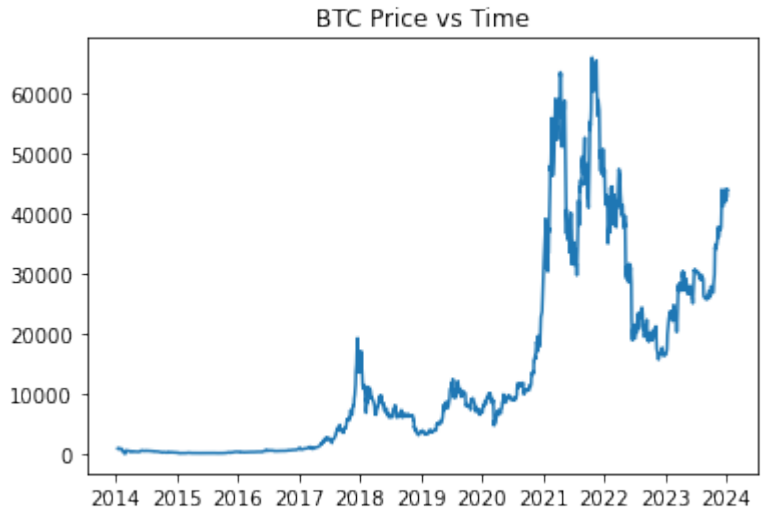

Figure 1: Price of Bitcoin over the ten year period, not a very pretty curve, but we can still see linear patterns over short time periods.

how the price changes on the daily. Over a large period of time though, the graph doesn't have such an easy function to fit to, as seen in Figure 1.

Looking by hand we can see the prices match up to the original file pretty well in addition to this high R^2. We also take note of the code making good decisions on when to buy and sell Bitcoin based on its daily price compared to the 'MAD'. This holds true for both the predicted values as well as the original data.

The code does make some mistakes, in both datasets, but it often corrects itself. At times where the market bears, the code quickly realizes it and sells the user's Bitcoin to protect them from suffering an even greater loss.

For a greater explanation and overview of how we determined the efficiency of our code, please see our YouTube video posted [2] where we go through some examples of different dates throughout the two data sets.

4 Conclusion

Our bot can not only advise a user when to buy or sell their Bitcoin, but can also predict what the price of Bitcoin may be on a day-to-day basis and further advice the user on what to do on a given day. This is proven by

the bots ability to predict prices for days that were removed from the data randomly over the ten year interval.

Had we had more computational resources, we may have been able to mine text from internet articles that may impact how buyers and traders may feel about the market, further impacting a price prediction. In addition, we may have been able to make a friendlier user interface for inputting threshold values and other information, and also a prettier way of seeing the results for a given day.

For now though, we believe this is a good way of proving our concept and we believe we have accomplished all of our original goals and exceeded our own expectations.

Acknowledgements

Allen Li derived the original algorithm for determining whether a user should buy or sell their coin, while James Marini came up with the model predictions and implementing the original algorithm to work on the new predicted prices. They both collaborated on importing the data set and finding it on the internet as well.

References

[1] GeeksforGeeks, "How to do train test split using sklearn in Python," June 27, 2022.

[2] Allen Li, YouTube "Allen Li and James Marini AMS 325 Final Project Video" `https://www.youtube.com/watch?v=bJkX3_xcKTo`