Reinforcement Learning for Crypto-Currency Arbitrage Bot

Allen Hosler
Stanford University
450 Serra Mall
ahosler@stanford.edu

Jaisel Sandhu
Stanford University
450 Serra Mall
jsandhu@stanford.edu

Abstract

This paper establishes a model to optimize capital invested in an automated crypto-currency Arbitrage bot. This method involves first dynamically estimating the maximum difference between two exchanges, noted herein as X1, and then investing some percentage of available capital based on the policy taken. The estimation will be done using a Naïve Bayesian Model, and the Optimal Policy will be found using Q-Learning. The model was able to provide a no-risk weekly return of 9% on the test week.

1. Introduction

With the recent rise in mainstream media coverage of Bitcoin and other crypto-currencies, there has been significant uncertainty in the value, validity, and future of these crypto-currencies. These currencies have had unprecedented swings, such as Ripple, which increased over 7,500% in 11 weeks, just to crash to 50% of its peak value one week later. It would appear that just about the only thing one is certain of, is the uncertainty of these currencies. Which is exactly what this model aims to exploit.

When extreme uncertainty exists about a commodity, there's often extreme volatility. When several markets are unsure of what the "true" value of a commodity is, they often produce different valuations throughout time relative to each other. Arbitrage is a process of finding where these differentials exist, and buying the commodity on the less expensive market, to sell it on the more expensive market for a profit. In the case of crypto-currencies, the markets are the exchanges: GDAX, Bitfinex, Bitstamp, and Poloneix to name a few, and the commodities are the currencies themselves: Bitcoin, Litecoin, Ethereum, etc.

The aim of this paper is to produce an arbitrage bot, which will discover these differences in crypto-currency prices across different exchanges, and exploit them. Then, given a finite amount of capital, find the optimal policy for investing that capital across the various arbitrage opportunities.

Due to the complexity and democratized-nature of the blockchain technology supporting these currencies, it could take over an hour to verify purchases, and even longer to exchange currency between 2 market places. Seeing as the average arbitrage opportunity only lasts about an hour, buying commodity C1 from market M1, and trying to sell C1 directly to market M2 will not always work in the limited time. Thus, to avoid exchanging commodities between markets, we will instead buy C1 from the lower market and short C2 on the higher market. Then we don't care what happens to the actual price of C1 or C2 during this interval, so long as they eventually converge within some epsilon.

Thus, theoretically, this arbitrage bot will be completely risk-free, with returns determined by how optimal the chosen policy is.

2. Related Work

The method discussed in this paper is a completely original approach to optimizing an arbitrage bot. A lot of other work has been referenced in the field of Reinforcement Learning, but none that address the same problem exactly. Resources that were referenced include: Statistical Arbitrage in Financial Stocks, and Energy Storage Arbitrage in Real-Time Markets via Reinforcement Learning, Machine Learning in Statistical Arbitrage.

2.1. Statistical Arbitrage in Financial Stocks:

John-Paul Meyers' research into Statistical Arbitrage for Financial Markets provided insight into the effectiveness of prediction [7]. The model proposed was complex, containing Kalman Filtering and Neural Networks. Ultimately it did a decent job of predicting the optimal time to buy and sell for the dataset it was given, but missed out on several opportunities when estimates were to high or too low to take full advantage.

This paper will build on part of the approach taken here by applying Reinforcement Learning to determine the optimal policy for buying and shorting positions provided a reasonable estimate of the max value.

2.2. Energy Storage Arbitrage:

A Paper published by Hao Wang and Baosen Zhang titled "Energy Storage Arbitrage in Real-Time Markets via Reinforcement Learning" explores representing arbitrage as an MDP and methods used for discretizing the state space. [8] The authors suggest Q-Learning provided the best results for policy finding of their arbitrage problem. This paper will use Q-Learning as well to find it's optimal policy.

2.3. Machine Learning in Statistical Arbitrage:

The model introduced by Xing Fu and Avinash Patra in their paper "Machine Learning in Statistical Arbitrage," involves taking in financial time-series data and performing Principal Component Analysis to find indicative signals of the market changing. Their analysis was an important baseline in the field of Statistical Arbitrage. [5]

3. Dataset

Financial data is among the best documented in the world, with records of stock changes every minute for the last 50 years. However, obtaining this data can be expensive, and gathering enough data to create a meaningful model is either very difficult or very expensive. Figure 1 below shows the real time price differential of Bitcoin between GDAX and Bitstamp on 12/9/17. [3]

Figure 1: Different Prices for Bitstamp and GDAX on 12/9/17

Luckily for crypto-currencies, bitcoincharts.com [2] contains minute-by-minute data from 2009, the conception of Bitcoin, to date for nearly 100 different exchanges. Then, using the Python Selenium Package, we wrote a script to copy all of this data onto a massive CSV file.

In Figure2 below we plot the differences in Bitcoin price minute-to-minute of two exchanges: GDAX and Bitstamp, from October 1st, 2017 to December 1st, 2017.

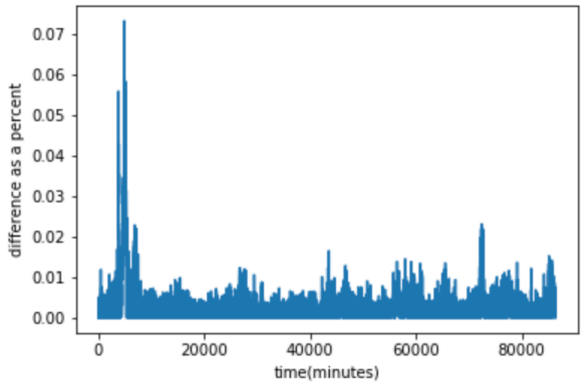

Figure 2: Percent Difference Between GDAX and Bitstamp from 10/1/17 to 12/1/17

While there are several variables that one could compare even within one minute of data (i.e. open price, closing price, high or low price), this paper examines the difference in weighted prices, which is simply the average price over that minute weighted by the volume traded at that price. It is our belief that this will be the best estimator of the price for that minute.

For training throughout this paper, we will use the difference in GDAX and Bitstamp from 12/1/16 to 12/1/17, which is a vector of 381,600 differentials.

4. Methods

There are several sub-problems within this problem of: determining the optimal way to allocate your capital into risk free arbitrage opportunities. This paper breaks down the problem into the three following sections: 1) Creating the software for the actual bot; 2) Given you are at a nonzero differential, estimating the peak differential you will achieve before reaching zero again; 3) Finding the optimal investment policy given your estimation in part two.

Altogether, the algorithm will look as shown below:

```
Capital ← constant amount willing to invest
Profit ← 0
Eps ← total cost of fees
While(True):
        Cap_used = 0
        X ← percent difference between exchanges
        While( X > Eps):
                X1 ← Get_X_max_Estimate(Volume, Price, X1_log)
                Policy ← Get_Optimal_Policy(X, X1, Cap_used)
                Profit, Cap_used = Execute_Arbitrage(Policy, X, X1, Cap_used, Profit)
Return Profit
```

Figure 3: Algorithm For Arbitrage Bot

4.1. Creating the Arbitrage Bot

Unlike High Frequency Trading, this program does not operate on a second-to-second timescale; there is no need to. Thus, it can afford to update the estimates of X1 (the max percentage differential between the exchanges) and the investment policy at every time step.

This program requires real-time data, and real-time interactions with the exchanges. For the naïve implementation described in this paper, the Python Selenium Package with Firefox Web driver will perform well. Though, for a more robust design, one should consider other languages as well.

4.2. Estimating Peak Differential

Provided two exchanges, with varying prices in time, the problem arises of estimating what the maximum price differential is likely to be before the differential hits zero again.

Simply plotting the X1 values (the max percentage differential between the exchanges), (Figure 2) without any prior knowledge one might expect them to resemble the right half of the normal curve, but instead we get Figure 3. The distribution more closely resembles a skewed Poison, with other distributions confounded within it.

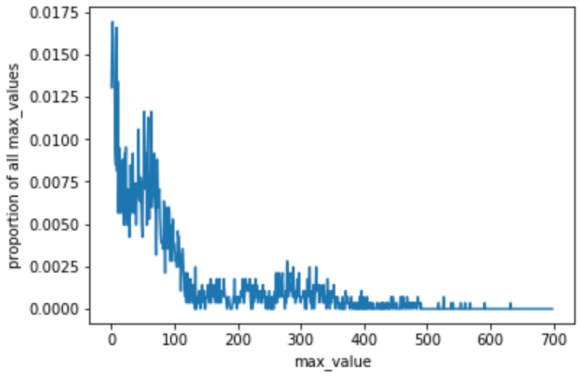

Figure 3: The Distribution of X1's

In an attempt to predict this X1 value, this paper uses a Naïve Bayesian Model, which will create a Maximum Likelihood Estimate of X1, given the current X (discretized into buckets by 5%), and a measure of variability (discretized into 5 buckets).

This variability metric comes from the exponential decay of standard deviation of the differentials data, over the previous year. This equation is represented below:

$$\text{Variability} = \sum_{i=1}^{365} \{a^i * \text{sd}(\text{diff}[i : (i+1440)])\}$$

Where: sd(x) is the standard deviation of the time series x; a is a tunable parameter € [0,1] of the exponential decay; and 1440 comes from the number of minutes in a day.

For better estimates, it is suggested that readers reference the paper mention herein section 2.1. While a Neural Network or Support Vector Machine may offer better estimates, such methods were not explored in this paper.

4.3. Optimal Policy Finding

Given the estimate of the max differential from 4.2, X1, what is the optimal price to perform the arbitrage? If it were known with 100% certainty that the differential would hit the estimate X1, the optimal policy would be trivial: invest 100% of capital available at X1. However, X1 is only an estimate, and if the differential only reaches X1-0.0001, no capital will be invested, and thus no profit will be made. Thus this is likely not the optimal policy.

Framing this problem as a Markov Decision Process allows for an opportunity to find this policy that optimizes profit. The State Space will be discretized as a grid of the percent of (X1) currently achieved, by the percent of capital currently invested. In this state space one can only move right, and some unknown distribution will determine the up and down movements, as shown in Figure 4 below. Figure 4 shows and example with 5 buckets, however the actual model contains 100 bucket, discretized into one percent intervals along the X- and Y-axis.

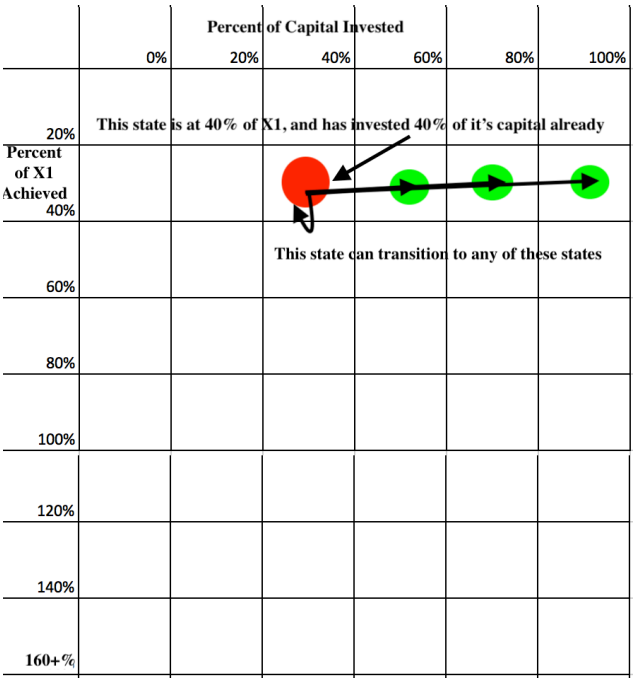

Figure 4: The State Space of the MDP with 5 buckets

The Action Space, as depicted above, is moving right by anywhere from 0 to k values, where k is dependent on the current state (or more specifically, on the percent of capital invested). The Transition Probabilities are 0 for any movement to the left, or off of the grid, and need to be determined for any other movement.

The Reward Space is a simple function of a' and s'. For the example in Figure 4, the rewards for each of the possible actions a0, a1, a2, a3, given the state s = (40%, 40%) are 0, (0.2*Capital + 0.4*X1), (0.4*Capital + 0.4*X1), and (0.6*Capital + 0.4*X1) respectively. This model assumes that Capital is nonnegative, and X1 is the absolute value of the differentials.

This paper then explored applying Q-Learning and SARSA-Lambda to achieve an optimal policy in the described MDP.

Some of the concerns of this model are the limitations. Because the model is for a stationary MDP, it does not take into account the amount of time spent at a certain threshold, or therefore the rate of range of the percent of X1. This information will ideally be included in the dynamic estimate of X1.

5. Experiments and Results

At the end of this modeling, the most salient result of the project is straightforward: Percent Return. This model is only effective if it can generate a significant amount of profit for the capital invested. To determine the percent return in the real world, this model was applied to the data collected from 12/1/17 to 12/8/17. The model was able to return 9% over the week, risk free.

This result should be taken with a grain of salt, however, as the transaction fees were only estimated, and the week of 12/1/17 to 12/8/17 was one of the most volatile weeks in the history of Bitcoin, growing from $15,000 USD to $19,000 USD, then back down to $15,000 USD. While crypto-currencies are volatile, historically they have not been this volatile. [4]

5.1. Estimation of Maximum Differential Results

The Naïve Bayesian Model was likely not the optimal solution, as the Maximum Likelihood Estimates were not significantly different enough to generate any sort of meaningful difference. Ultimately, this aspect of the model wants to predict the outliers, a rather difficult task from a statistics perspective.

However, it appears likely that models producing better estimates exist. With the wealth of knowledge available, and long periods of time for calculation, the Kalman Filter to Neural Network approximation discussed in Meyers [7] would make an excellent estimator.

5.2. Optimal Policy Results

The MDP structure presented was slightly unconventional, but turned out to be rather successful when implemented. The model did better as more data was added and the number of buckets (discretized chunks of the percent of X1) decreased to around 10. Going further, making the bucket sizes exponential may help the Q-Learning converge faster, as the model adopted was a nearly exponential assignment of capital.

Q-Learning appeared to work well, however moving forward, one may try experiment with other algorithms to determine the optimal policy such as Monte Carlo Tree Search, which will increase the sampling, to increase the models confidence in its X1 estimates.

Finally, finding a better way to model the fees may improve the policy. The model was written such that it wouldn't execute a trade unless it was greater than the cost of fees (which was typically about 0.1-0.2%).

6. Conclusion

6.1. Pitfalls of Model

This model, as with most time series or forecasting models, struggles to predict outliers in future data given historical data [1] [6]. Theoretically, that should not matter, as there is no significant risk associated with the

arbitrage scheme established. In actuality, however, models, especially bots, fail tragically all of the time and create crisis.

6.2. Usage of Model

This model should be used with extreme caution and supervision. It was built to perform in a specific way, and if inputs vary slightly, a lot of money could be lost due to some obscure bug. For this reason, this model has never been tested in the real world, only tested in a simulation with real world data.

6.3. Further Exploration

There are three big progressions that must be made in this model. Firstly, determining a better way to estimate X1. Several ideas have been put forward herein for this purpose. Alternatively, estimating Xt1, the average time to the maximum differential, may prove easier and more significant. The MDP aspect would only change slightly to have the state space by the percent of (current time / estimated time to maximum) X percent capital invested.

Secondly, allowing the MDP to move right and left in the state space, corresponding to the bot liquidating more capital from its investments. The Reward scheme would need to be more complex to account for the fees of this procedure.

Finally, this model could have a second estimated parameter: X2, which determines the lower threshold for when to sell. This would serve to address circumstances in which differentials oscillate between 0.8% and 0.2%, but rarely hit 0%. Allowing for this lower threshold to be dynamic as well will optimize this problem further.

Acknowledgments

Special thanks to the team bitcoincharts.com for publishing their data online for the general public.

References

[1] Arratia, Argimiro. “Forecasting Financial Time Series with Machine Learning Models and Twitter Data.”

[2] “Bitcoin Charts.” *Bitcoincharts*, 8 Dec. 2017, bitcoincharts.com/.

[3] “Cryptowatch.” *Cryptowatch*, cryptowat.ch/.

[4] “Fees Schedule.” *Bitfinex.com*, 2013.

[5] Fu, Xing, and Avinash Patra. “Machine Learning in Statistical Arbitrage.” *Http://cs229.Stanford.edu/*.

[6] Lee, Jae Won. “STOCK PRICE PREDICTION USING REINFORCEMENT LEARNING .”*Http://Plaza.ufl.edu/*, 2001.

[7] Meyer, John Paul. “Statistical Arbitrage in Financial Stocks.”*Www.engineering.unsw.edu.au*.

[8] Wang, Hao, and Baosen Zhang. “Energy Storage Arbitrage in Real-Time Markets Via Reinforcement Learning.” *Arxiv.org*, 8 Nov. 2017.