VWMA Trading

Ahmed El Aaradi
Matthieu Green

Table of Contents

Introduction

For this project we were given the tweeter feed and order books for five cryptocurrencies on five different exchanges for a timer period spanning six days from February 28th 2022 to March 5th 2022 included. In this paper we will analyze the available data before building an algorithmic trading strategy. We will overview various possible strategies like trading on bid-ask volumes imbalances or making a machine learning model on the tweets to determine sentiment and make trades accordingly. Then we will detail our final algorithm.

Ultimately, we are looking to produce a profit generating strategy once transaction costs are considered. Our final model isn't a high frequency trading algorithm based on order book information, but rather directional trading based on a single signal computed from two Volume Weighted Moving Averages (VWMA) with different window sizes. We will also discuss the scalability and the limitations of our strategy, especially during periods where the prices move sideways.

Files Description

Main:

- backtest.ipynb: code where we test the strategy. You can choose the exchange and the asset.
- vwma_model.ipynb: code to view charts on the VWMA strategy

Order Book Viewing:

- animated_order_book.py: run it to see the animated order book over time
- data_viewer.ipynb: code plotting interesting moments in the order book
- order_books_stats.ipynb: code to compute various statistics for each exchange

Fees:

- fees.py: code to build the table of the best fees depending on the notional and whether it is taker or maker fees

Tweets:

- tweets.ipynb: code to process and clean the tweets
- model_tweets.ipynb: quick logistic regression on tweets
- tweets_stemmed_hashtags_no_stopwords.csv: CSV of the processed tweets

Alternative strategies:

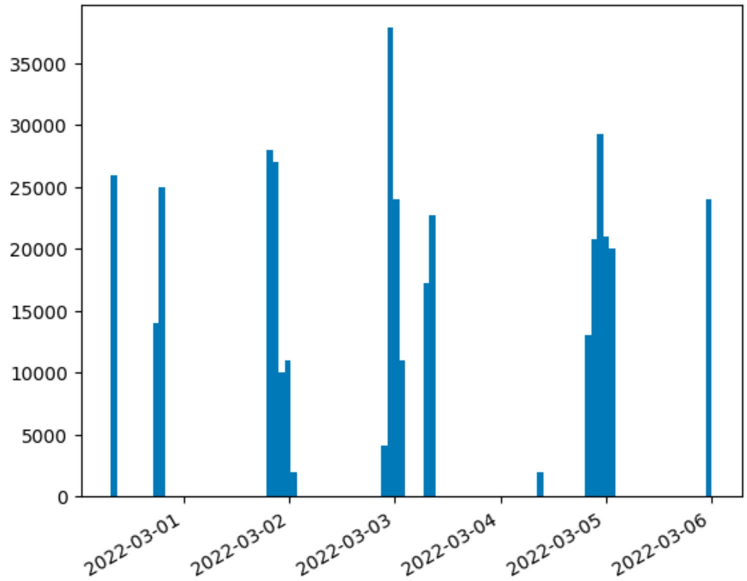
- Trading_strategies.ipynb

I. Data Observation

A. The tweets

There are several ways that tweeter feeds can be used as a signal for algorithmic trading. Accounts followed by millions of users might have significant influence on cryptocurrencies' prices. Indeed, such assets are easily accessible to retail investors that trade on sentiment which makes them more likely to purchase or sell cryptos depending on what public figures think of them. However, tweets have also gathered the attention of more sophisticated investors for exactly that reason. A trading algorithm that enters a position right after a tweet from above mentioned accounts would benefit from the subsequent orders of retail investors seeking to join the rally or exit before the bear market. However, although these strategies can be very profitable, such events remain rare. Nonetheless, tweeter feeds can also cast a light on the overall sentiment. Analyzing what is being said in crypto-related tweets, regardless of account followers, it could turn into an interesting indicator of the wider trend.

Let’s take a closer look at the data before we start. We first noticed that the feed isn’t in continuous time throughout the period. In fact, we only have feeds for a few hours per day.



Histogram of tweeter feeds over the six-day period

This becomes an issue if we need to keep our positions opened until the next signal, which might come several hours later. There are two options: the first one is that we use the overnight and morning tweets as signals for the daily sentiment. For instance, if the tweets from March 1st to March 2nd are leaning towards positive returns, we would enter a long position that we revert if the tweets from the next day indicate a fall in crypto prices ahead. This could be combined with a stop loss and take profit to better manage risk during the time when our positions are open. Another strategy could be to trade at a given frequency (say 5 min for instance), hoping that tweets from the prior period would signal the direction of prices in the next one.

However, tweet-related strategies require some cleaning of the data prior to any machine learning algorithm. Indeed, tweeter is a common place to promote crypto giveaways, new launches, easy money etc. All of which are considered noise as they are independent from the underlying performance of cryptocurrencies. After filtering them out, we significantly decrease the number of relevant tweets, from 389k to just over 156k. Although such filtering is necessary to ensure good model accuracy, we have much less data to train on. This might not be an issue if we decide to trade based on public figures’ tweets. Unfortunately, we have observed no events, during the six-day period, where crypto prices react to any of the 44 tweets from the 9 accounts having more than 1M followers after the filtering.

Testing a quick model

We decided to test the accuracy of a model based on a Logistic regression of the vectorized words appearing in the 156k tweets. We first stemmed the words and removed any stop words to reduce the number of features. We proceeded to resample the order book for ETH on Binance by 1min frequencies and match them with the timestamps of the tweeter feed. We then computed the returns of ETH on each period and converted it to binary, 1 being a positive return. We now have a classification problem and will use the tweets from period $t-1$ to predict the returns on period t. The results are as follows:

TRAIN	0	1		VALIDATION	0	1
0	187	0		0	97	90

1	0	187		1	82	102
Accuracy	1.0			Accuracy	0. 5363881401617251	

As we can see, once we resample the data into a given frequency, even as small as 1 minute, we must train our model on 374 observations which makes us question the statistical significance of this model and poses a challenge to statistical inference and robustness. The accuracy is slightly better than random, but we need to bear in mind that a trading algorithm on a 1min frequency will make hundreds of trades. Entering and closing a position means we must pay a taker fee twice. When we predict accurately the performance of the underlying, we hope that the positive returns more than compensate for the fees. However, when we are wrong, we will bear the fees as well as the negative returns.

Just for interpretation purpose, we decided to try it with a 10-minute frequency, to reduce the number of trades hence the fees.

TRAIN	0	1		VALIDATION	0	1
0	42	0		0	31	11
1	0	25		1	18	7
Accuracy	1.0			Accuracy	0. 5671641791044776	

The accuracy on the validation set jumps to 56.7%. This could be linked to the increased number of words per observation as more tweets are joined together over the longer frequency. It could also mean that tweets have a better predictive power for the next period spanning 10 minutes rather than 1 minute. However, as predicted, the number of observations falls to just 67 for the train set which is not enough to be statistically robust.

B. The Exchanges

Let's look at the different exchanges, analyzing the fees but also statistics on the order books such as bid-ask price differences and volumes. We will use Ethereum as a basis of comparison between each one, as our final model will trade on ETH. It is important, however, to note that our final algorithm is meant to work with any underlying (crypto but also equity), without any further calibration needed. Furthermore, we will focus on taker fees, as our model only makes market orders.

When it comes to fees, Bitmex is the undeniable winner, no matter the notional traded or the type of order placed. It does come at a cost though. As we can see from Table 1 in the appendix, the average bid-ask price spread for ETH is 0.343, the highest spread out of all the exchanges. For our model and a starting capital of 1,000 USD, performs worse on Bitmex. Does it mean that a higher bid-ask spread comes at a greater cost than slightly higher fees? The answer isn't straightforward, and we will discuss this later. However, there is another issue with Bitmex and it's the sheer volume on its order books. Although it ranks among the less liquid exchanges out there, their order book suggests nearly 10 times more volumes than the rest. There are reasons to doubt the validity of these figures which are probably the main reason why our algorithm fairs so poorly on Bitmex.

The next best exchange for taker fees, regardless of notional, is FTX. Ideally, we want to build a trading algorithm that can be used tomorrow in real market conditions, and although FTX still existed around the time when our sample is taken from, we decided not to include it as a viable option given the circumstances. We are therefore left with Binance, Coinbase or Kraken. As we can see from Table 2 in the appendix, Binance has the third lowest taker fees after Bitmex and FTX which is why we have chosen it.

As we will see later, the row count in Table 1 is also worth mentioning. Binance has just over 0.7 million rows (i.e. timestamps) over the course of one day whereas Coinbase and Kraken both have

14.8M and 2.3M respectively. This ultimately translates into smaller time intervals between order book snapshots. This is particularly important when computing the traded volume and traded price.

C. The Order Books

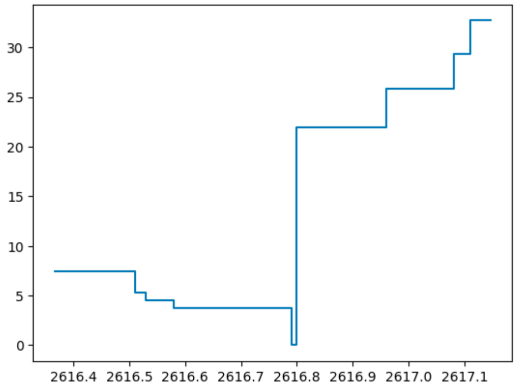
Each timestamp gives us a snapshot of the limit orders placed on both sides, the bids and asks which represent buying and selling orders respectively. For each price the volume is expressed as a fraction of one unit of Ethereum:

$$Asks[0].amount = \frac{Quantity\ (USD)}{Asks[0].price}$$

To view the order book, it is common practice to plot it as steps, with the cumulative volume of the bids on the left and of the asks on the right. Indeed, buying pressure (left) will try to move the prices upwards whereas the selling pressure will act as an exact opposite. They are separated by the mid-price:

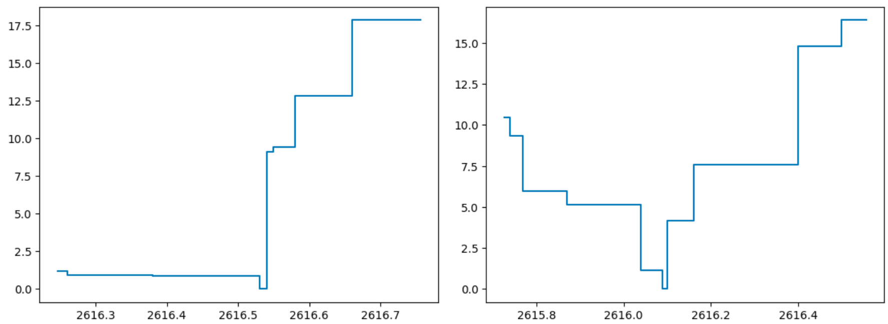
$$Mid - Price = \frac{Asks[0].price + Bids[0].price}{2}$$

Let's take a look at our first snapshot, on February 28th :



Binance Order Book snapshot for ETH at 2022-02-28 00:00:01.975576

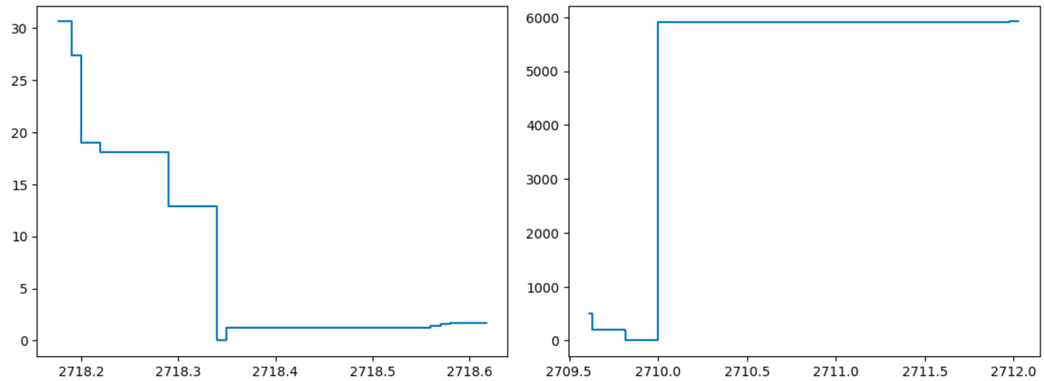
As we can see, the mid-price is set at 2616.795 USDT, the bid-ask spread is only 0.01 USDT and there is an important selling pressure with a volume of 18.1477 at 2616.8 USDT versus a volume of 3.7203 at 2616.79 USDT on the bid side. Another observation worth mentioning on this snapshot is the difference between the first bid price and the second closest bid at 2616.79 - 2616.58 = 0.21, adding just 0.7679 to the volume. On the other hand, the price spread between the two first asking prices is smaller, at 2616.96 - 2616.8 = 0.16. Clearly, the selling pressure is taking the lead. As expected the price later falls :



Binance Order Book snapshot for ETH at 2022-02-28 00:00:02.172 and 2022-02-28 00:00:02.272

This might be a very good algorithmic trading strategy. As we've seen on the previous example, the mid-price fell from 0.7 (or 0.02%) USDT in less than 300ms. However, on the next timestamp we see the price goes up 0.375 USDT, it is therefore important to close the position at the right time. Furthermore, once fees of entering in a position and exiting are considered (0.1% for binance for instance), this profitable trade turns into a loss (it is important to note that there are other examples for which this strategy generates profit, especially when considering wider time windows).

In other cases, the imbalance between bids and asks might be misleading altogether. Take a look at these two consecutive snapshots:



Binance Order Book snapshot for ETH at 2022-03-04 01:09:13.184 and 2022-03-04 01:09:13.284

They were taken at 100ms interval, during which ETH drops by a whopping 18 USDT. At first, the buying power being the dominant force, one might enter a long position, hoping for the prices to shoot up. Unfortunately, a massive selling order (market order that turns into a 2710 USDT limit order) wipes out the entire observable bid volumes and even more. It then turns into a resistance level with more than 6000 ETH waiting to sell at this new level.

Out of the 4,213,301 timestamps that we have for ETH on Binance, there are 91,755 instances of big orders. We define "big" orders as events where the traded volume from one period to the next one is larger than the observable volumes on either the bids or the asks depending on the side of the trade. This is roughly 2.17% that could come at a great cost as they are market orders not initially observable on the order book that override the imbalances as seen in above example. It is, once again, important to note that data with smaller time intervals could solve this issue. Indeed, with Kraken, which has 11,039,808 timestamps for the six-day period, only shows 2,316 orders with bigger volumes than we can observe in the order book.

D. Limitations of the Order Books

The last example reminded us (i) that we don't have the full spectrum of the order book and (ii) the disruptive nature of a big market order. Both combined become a big issue when trying to compute the traded volume and traded price at each instance, which we will get into in the next section. However, (i) alone has many consequences. One might argue that an order book depth of 5 is enough to compute a reliable imbalance metric, which indeed is mostly driven by the volumes closest to the mid-price. But there is also a psychological aspect to order books. Seeing a massive volume at levels that are a little bit further from the mid-price gives an overall market sentiment. This could allow us to have a strategy that focuses on the wider trend, holds positions longer, and identifies levels of resistance and support. All of which would be beneficial to an algorithmic trading strategy as it would reduce the number of trades (hence the fees) and signal when to close or revert positions. Having less depth in our order book forces us to take positions on the very short term. Imbalances at this level might be good indicators on the direction of the price in the next milliseconds but not more. Furthermore, lower depth also means that volumes can change very fast, and imbalances might shift a few times in a matter of seconds. In the absence of transaction costs, such a strategy might be extremely profitable but even when trading on Bitmex with 0.05% taker fees it would be hard to break even and once the bid-ask spread is considered, you will most likely be losing money.

A shallow order book might also send the wrong signals for strategies tracking order book volumes and momentum. Let's imagine that there is a big buying market order that completely shifts the order book to the right. Suddenly, we might observe significant volumes appearing on the sell side and volumes on the buy side fading. This does not suggest the buying momentum is over. The selling orders were probably already there, just not on our observable window. On the other hand, the buy side might seem weak, but it means it is still building up at these new price levels. This new imbalance favors the sellers but if momentum has built up, we will observe more buying market orders of investors that want to join the rally and even if the buying limit orders take time to appear, the market orders can still consume and break the resistance level.

One last point that is important to mention although less relevant for an order book depth of 5 is spoofing. For example, investors wanting to buy at a lower price might place a big selling limit order to give a false impression that the sellers are building up momentum in hope that prices fall to the desired price. However, as mentioned, such activity rarely happens on price levels so close to the mid-price as there would be a non negligeable risk of the order being executed.

II. Trading Strategies: trial and errors

Now that we have talked about the data structure, first intuitions and the orderbook & its limitations, we will now discuss the various trading strategies that we have tried but that we ended up not choosing as our main model, either because the strategy does not yield any remarkable returns, or maybe that it still need a lot of additions and corrections. After working with the given order book and observing volume and price changes, we noticed various price behaviors that directed us towards specific strategies, some related to price and others more related to volume and order book imbalances.

Now, before discussing the implementation and results of the trading strategies, it is crucial to address an important question - how should we manage our starting capital? There are various approaches to handling capital, such as starting the strategy with all the capital and seeing how it affects our capital. We did not take this approach.
Instead, we opted to divide the entire data period into smaller periods. And the initial capital is equally distributed among all these smaller periods: 1000 USD / number of periods. This approach allows us to deal with a relatively small amount of data at a time, which will in turn allow for faster computations. This approach was also adopted since the strategies do not contain any risk management characteristic: no stop loss or exit strategy in place. By subdividing the data into smaller periods (and the capital into smaller portions), we can effectively test our trading strategy to see if it's applicable or not. To obtain the overall PnL of the strategy, we sum all the profits and losses over six days.

It is worth noting that the trading strategies were primarily designed for high-frequency trading and were created using one-hour datasets. The large quantity of data available made one-hour datasets sufficient for tweaking the strategies before testing them on the remaining data. During the testing phase, it was crucial to determine if the strategies were viable or if they were merely returning positive results due to overfitting bias.

Therefore, the strategies were constructed in a way that made them as general and scalable as possible to avoid overfitting. While this approach is more time-consuming, it is crucial to ensure that the trading strategies can deliver consistent profits in real-world scenarios. By adopting a more rigorous and cautious approach, we can minimize the risks and improve the overall effectiveness of the trading strategies.

A. Price-based Strategies:

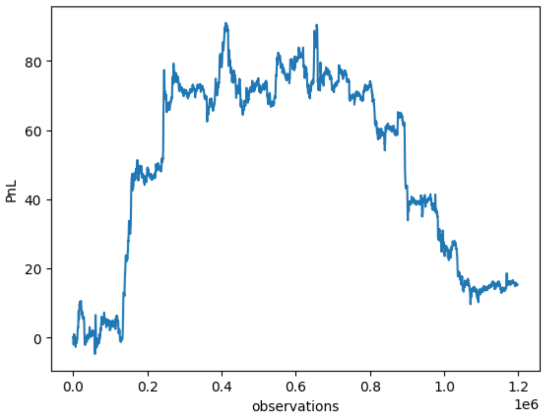
The first idea we had was as simple as they come: we expect the price to decrease when reaching the previous $n_{observations}$' max price. We start by computing the mid-price as well as its max value within a fixed sliding range of observations. The function max_to_short computes the returns of such a strategy using the following formula:

$$(bids[0].price_{t+timeperiod} \,/ asks[0].price_t) - 1) \;\mathbb{1}_{\left\{\left|\max_{[t,t-\max\_sliding\_period]}(\text{midprice}) - midprice_t\right| < \varepsilon\right\}}$$

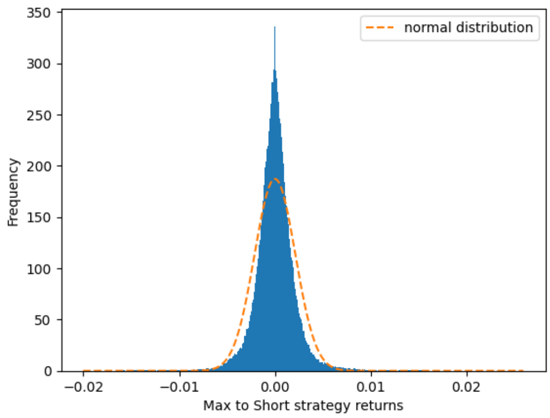
The main assumption behind this (which will later prove to be not true) is that the price will decrease after getting closer to the sliding period's max price. Therefore, we proceeded with the strategy and used the huge amount of data to our advantage:

➔ Assuming this strategy is valid, if we were to open a position at every observation where the condition above is fulfilled and compute the average PnL of all the opened positions throughout the entire data and across various currencies, then we should be able to have a robust and solid idea on the returns generated by this strategy.

This graph represents the strategy returns over the entire 6-day period applied to Ethereum. All the opened positions were taken into account (up to one position per observation depending on the period). Each position was given a 1$ capital and we cumulated the PnL of all the positions to obtain the graph on the left.



PnL on ETH over the entire data.



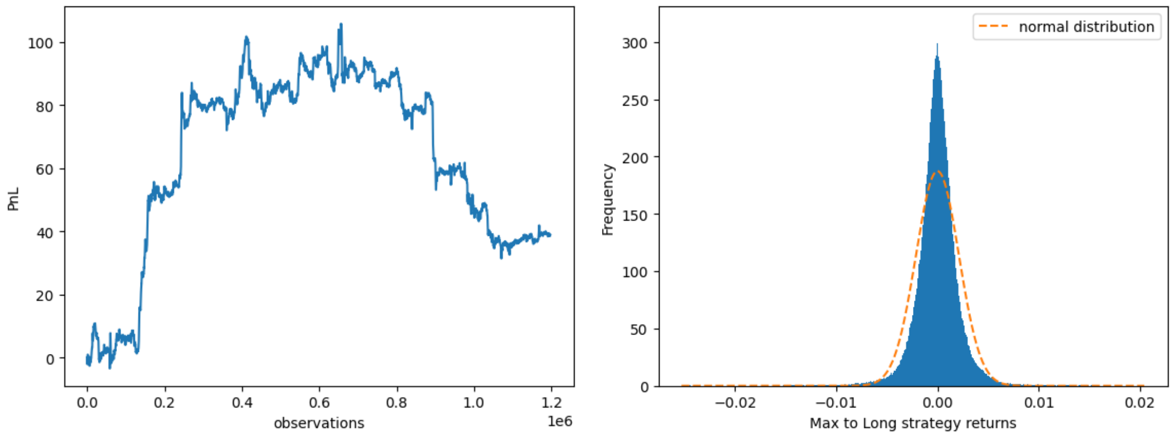
Distribution the returns on ETH over the entire data

The returns of this strategy turned out to be very limited (given the number of required transactions), centered around 0 and with a slight negative skewness (around 0.16). Despite the appearance, these returns do not follow a normal distribution (JB-test: 4913354). And when testing the significance of these returns, we find that they're significantly different from 0 with 95% certainty.

Distribution of max_to_short strategy returns on ETH over the entire data.

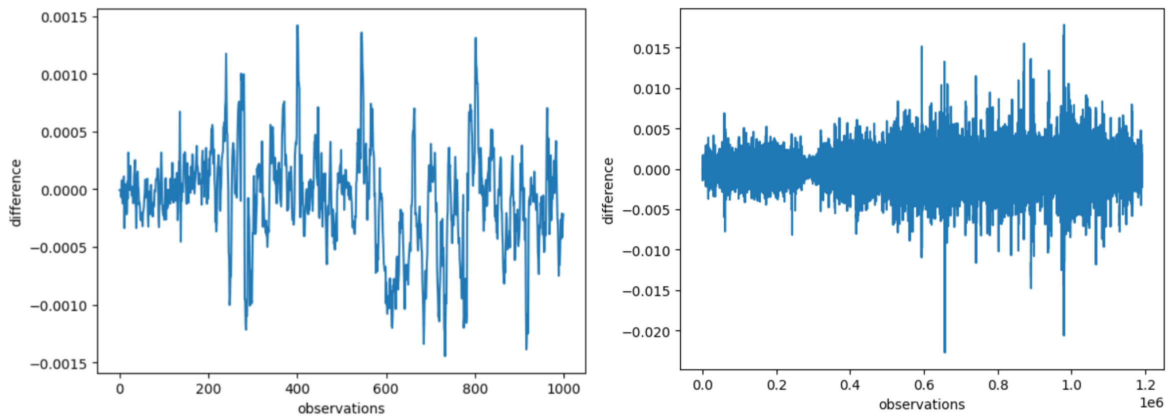
We get somewhat the same results with the opposite strategy max_to_long: open long position when we get close to the period's max price.

$$(asks[0].price_{t+timeperiod} / \,bids[0].price_t) - 1) \;\mathbb{1}_{\left\{\left|\max_{\left[t,t-\max_{sliding}\ \text{period}\right]}(\text{midprice}) - midprice_t\right| < \varepsilon\right\}}$$



PnL and distribution of max_to_long strategy returns on ETH over the entire data.

This is very surprising! At a first glance we see no apparent difference between the two graphs for the different strategies, however, once we plot the differences between the two, we obtain the following:



Difference between Max_to_Short and Max_to_Long strategies:
first 1000 observations(left) and entire data (right)

We do end up with some differences in these strategies, however, these differences are very small considering the nature and the formulas behind these strategies. This poses the question of why we have such similarities.

B. Sentiment-based Strategies:

Sticking with the same strategies, we decided to use the tweet signal we developed earlier to signal the choice of opening a long or a short position each time the condition is fulfilled. As explained earlier, it's important to note that the tweet data is very limited (only a couple of hours per day).

We obtain from the tweet signal model a set of 1s and 0s that represents the market sentiment over each period (1: returns>0, 0: returns<0). After resampling the data with respect to the time and date, we end up with the daily signal. This signal will therefore be used to give us an idea of the market sentiment over each day. A neutral signal will have a value of 0.5, but slight divergence from this value will indicate a market sentiment leaning more towards positive or negative expected returns.

2022-02-28	0.419355
2022-03-01	0.490909
2022-03-02	0.521739
2022-03-03	0.409836
2022-03-04	0.410714
2022-03-05	0.522388

Tweet Signal Results for ETH

Usually, the median is a more robust indicator of the "middle value" since it isn't biased, however in this case using the mean here does not add any bias to the final estimation since the values as only 0s and 1s with no presence of outliers in this set.

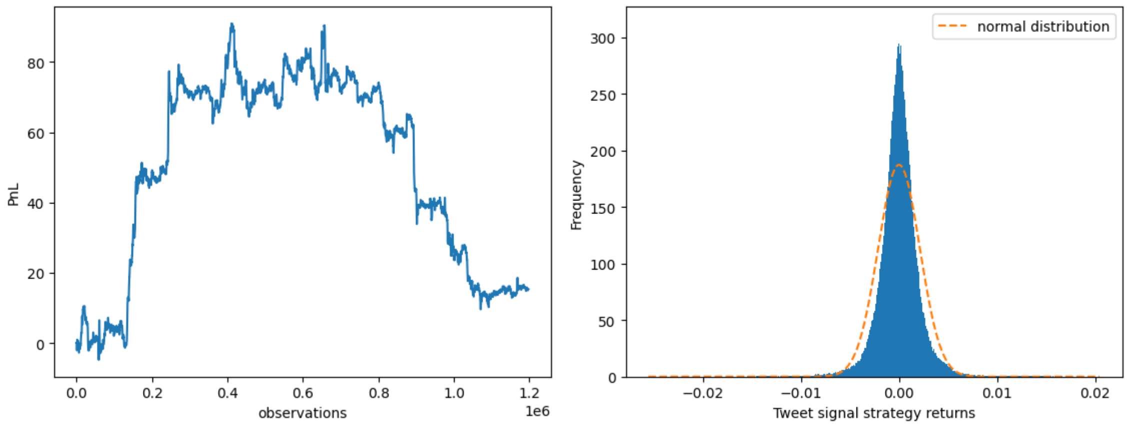
And so, when we observe a market sentiment leaning more towards negative expected returns (i.e., signal <0.5), we expect that the price will mostly likely decrease when it gets closer to the previous period's max price, and vice versa:

$$\text{Returns}(0,\text{T}) = \sum_{i=1}^{nb\_days} Max\ to\ long_i\ \mathbb{1}_{\{Tweet\ signal_i > 0.5\}} + Max\ to\ short_i\ \mathbb{1}_{\{Tweet\ signal_i < 0.5\}}$$

Again, the objective here is to find the average position return over all the data, in order to answer the following:

- Does the strategy work? How are the returns?
- Is the returns distribution well known? If close to 0, are the returns significant?

One weakness of this strategy is that it depends strongly on the tweet signal. If it was to use the max_to_long strategy when there is clearly a bearish trend, the strategy will cause us a lot of losses. And since we only have a measly 56% accuracy on the predictions, we don't expect any positive returns.



PnL and distribution of tweet signal strategy returns on ETH over the entire data.

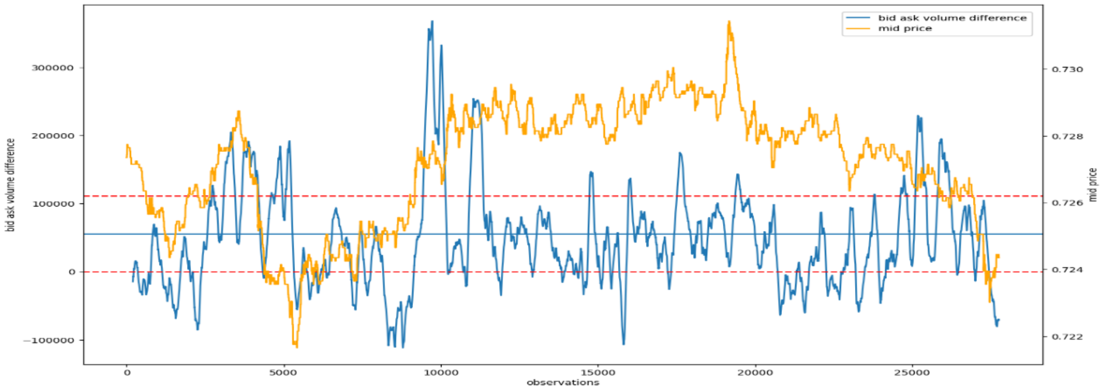
These results are surprising once again, since we barely see any differences between these graphs and those for each strategy. However, after further examination of the results, we can see that there are some differences, first in the number of opened positions and second in the actual returns. However, it can be understandable since both strategies returned somewhat the same returns (ref. price-based strategies).

To summarize, does the strategy work? Not so much in this case, there is close to no difference between the results given by this strategy and those of the strategies that it uses. The returns are centered around 0 with a slight skewness, but we do not know the distribution of such returns (J-B test: 43029). Since the returns are quite close to 0, we test for the significance of these returns, and we find that they're not statistically significant from 0 at 95% certainty.

C. Orderbook-based Strategies:

The second strategy we worked on is directly related to the order book and the imbalances we can witness in it. The main assumption is when we observe a big volume differential between the bid and ask sides, then the price is likely to move up or down within a set time frame.

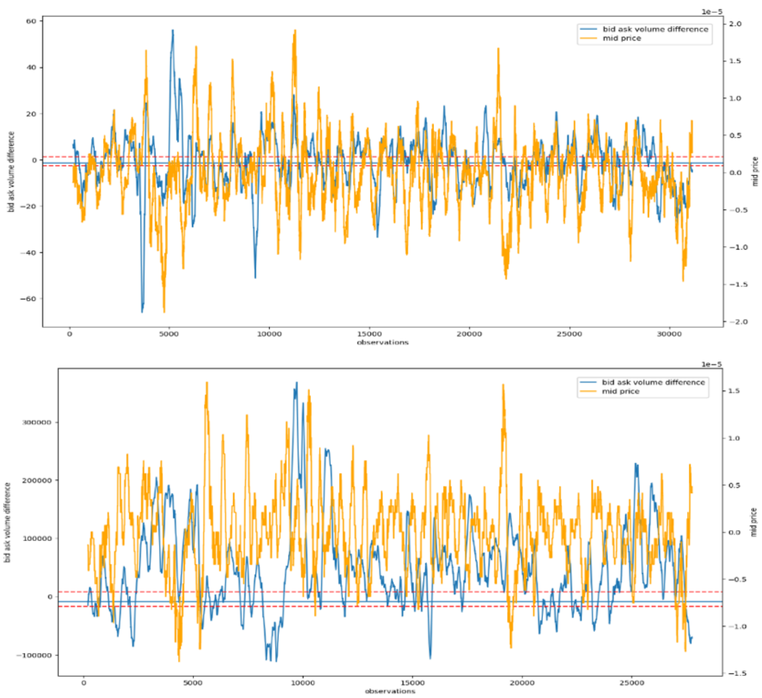
$$bid\ ask\ volume\ difference = \sum_i bids[i].amounts - \sum_i asks[i].amounts$$



XRP mid-price and bid-ask volume difference 28/02/2022
13:00:07.473 - 13:59:59.892

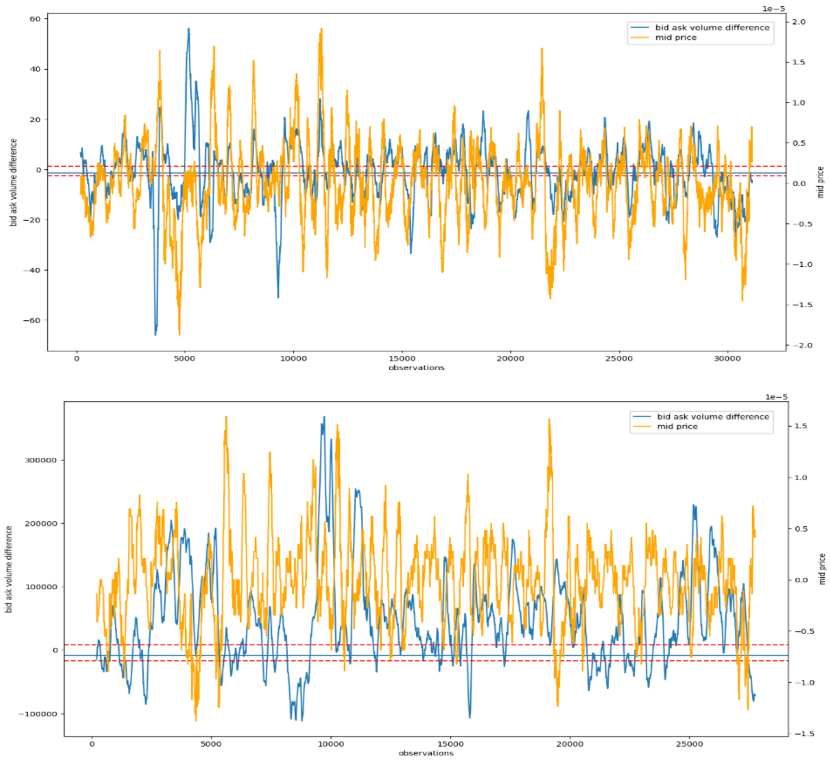
The graph above represents the first order differential between the ask and bid volumes (blue curve), the horizontal lines represent the average of the differential over the first 3000 observations along with a chosen interval around the mean, and finally, the mid-price in the yellow curve. The idea is to be able to find an interval which will contain most of the movement of the differential between the bid and ask volumes, so that once the differential is out of that fixed interval, we can open a long or short position accordingly. The first intuition would be to use the average. Sadly, it doesn't yield good results for all currencies (ref. graph below).

Given the nature of the data, the difference's mean is usually around 0. But as we can see from the graph below, the mean on its own is not enough to have good interval that will encompass most of the variations in the differential. We can also see from the graphs below that the volume differential is very volatile and has a very wide range (in the case of the ripple, volume differential ranges between -100000 and 350000 units).



Bid ask volume differences vs mid-price returns ETH (above) & XRP (below) 28/02/2022 13:00:07.473 - 13:59:59.892. Used the mean for the blue horizontal line and [– mean, 2 x mean] for the interval.

This is why we use the median. The median is an unbiased estimator that will help us have a concrete idea on the scale of this data, without being impacted by the outliers. And thus, we will use the median for our bounds around the mean.
However, the mean as we said earlier is not always helpful. One way to correct for this would be to apply the mean to the absolute value of the differences. And so, it will be necessary to explore the data to know whether or not the differences are centered around the mean or the mean of the absolute value of the differences to then construct a correct interval which should be representative of most variations in volume (most of the noise).



Bid ask volume differences vs mid-price returns ETH (above) & XRP (below) 28/02/2022 13:00:07.473 - 13:59:59.892. Used the mean for the blue horizontal line and [– mean, 2 x mean] for the interval.

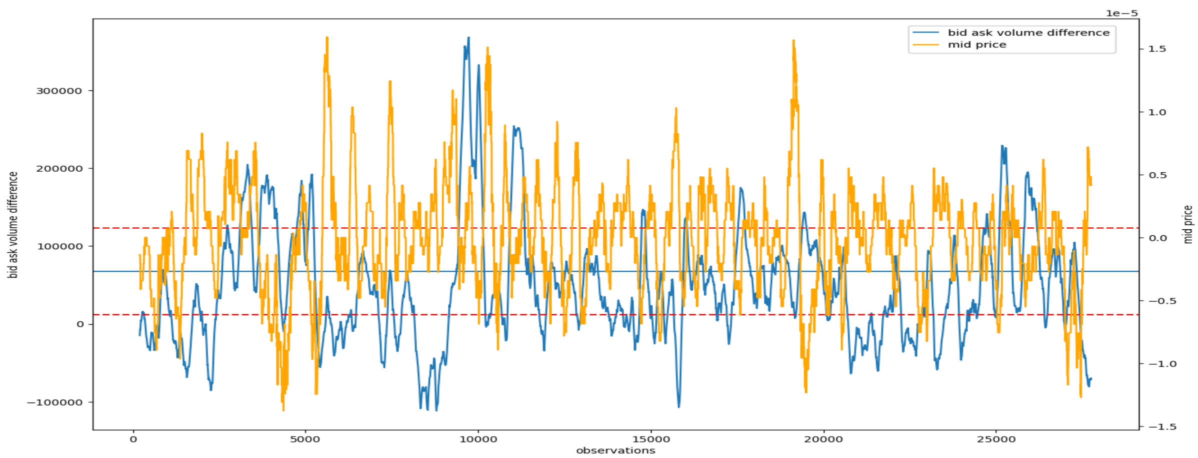
Applying the following intervals:

$$variation\ interval \coloneqq \left|Average_{0,3000}(bid\ ask\ vol\ diff) - Median_{0,3000}(bid\ ask\ vol\ diff)\right| \leq 0$$

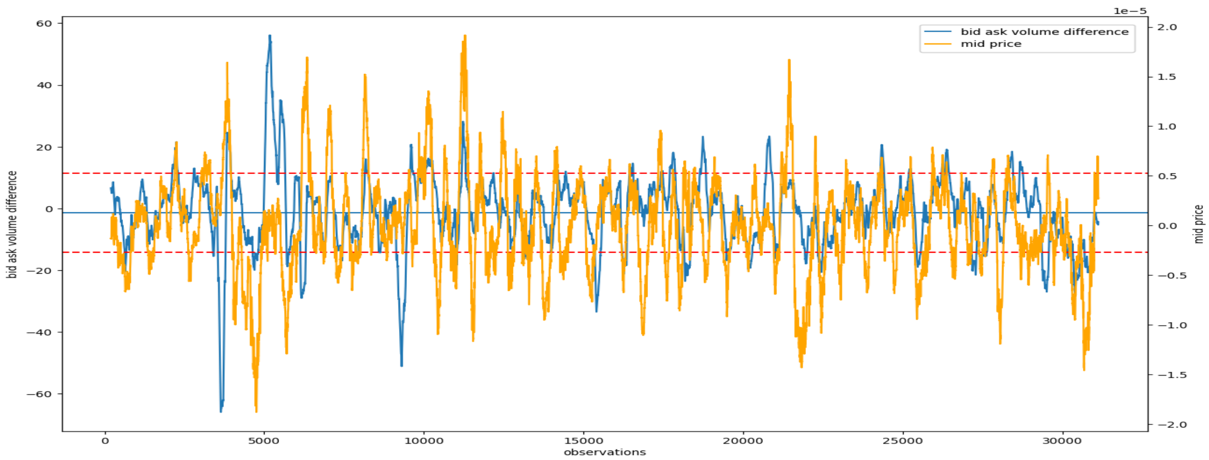
or

$$variation\ interval \coloneqq \left|Average_{0,3000}(|bid\ ask\ vol\ diff|) - Median_{0,3000}(bid\ ask\ vol\ diff)\right| \leq 0$$

We obtain the following results for Ethereum and Ripple:

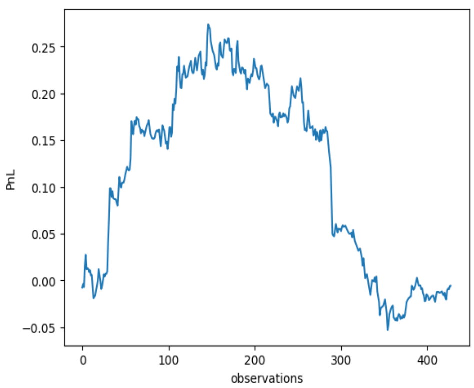


ETH bid-ask volume differences vs mid-price returns 28/02/2022 13:00:07.473 - 13:59:59.892



XRP bid-ask volume differences vs mid-price returns 28/02/2022 13:00:07.473 - 13:59:59.892.

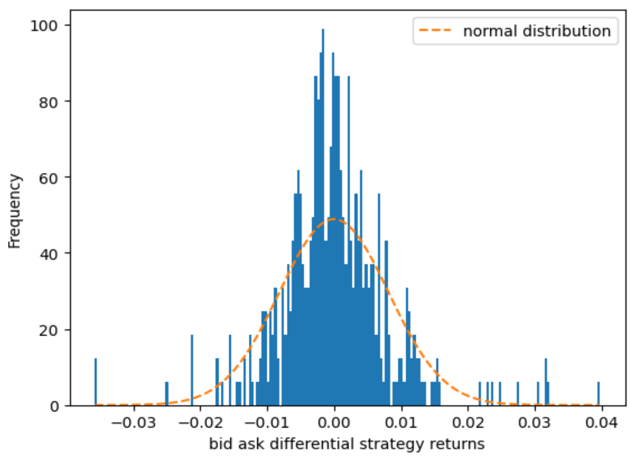
The average value, lower and upper bounds will be computed using the first tenth of the dataset. Then we check for each observation if we are within the set bounds of our model. Once the volume differential gets out of the interval, we open one position for a set time window and store the resulting PnL once the time frame ends. This will allow us to only be long or short at a time and correctly follow the signal we get from the bid ask volume differential.



PnL of the orderbook based strategy on ETH over the 6-days – timeperiod

We then proceed to test our strategy on Ethereum. We proceed a bit differently this time regarding the capital allocation since we have at most one position per set time window. Here we distribute the capital as: 1000 USD / (max number of positions per day * 6). So, for a set time window of 10000 observations, we will have at most around 100 positions per day, which gives us a starting capital around 1.7 USD for each position.

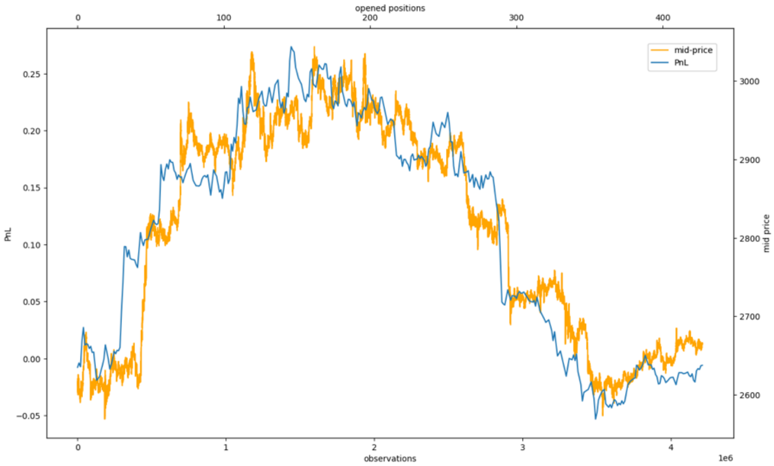
This graph represents the strategy PnL over the entire 6-day period on Ethereum. We have at most one position per fixed time frame (here: 10000 obs per time frame). Each position was given 1.7 USD capital as explained earlier and we cumulated the PnL of all the positions to obtain the graph on the left.



Distribution of the PnL of the orderbook based strategy returns on ETH over the 6-days – timeperiod = 10000, starting capital for each position at 1.7 USD

In this strategy, the returns are quite interesting, especially since we're not having that many open positions (428 positions). Still, we have not yet considered the impact of fees. Given the structure of the strategy (we'll have at most 400 open positions throughout these 6 days) and the limited capital of 1000$, we should expect a limited decrease in the observed returns: 2.23 USD in this specific case.
The returns distribution is also quite interesting: it is the closest we had till now to a gaussian distribution (J-B test: 383) with a slight positive skew around 0.49. And when testing for the significance of the returns, we find that the returns are not significantly different from 0 with 95% certainty.

Finally, surprisingly enough, the PnL observed above, is not that much different from the actual price movements of the underlying (here Ethereum):



One can even think, looking at this graph, that we're dealing with a long only position. This might be due to the low number of positions and since we're keeping them open for 10000 observations. We'll be dealing with the limitations of such an approach in the next subsection.
We do notice some slightly "better performance" at times, but we will need to implement an exit strategy to be able to profit from such opportunities.

Limits of the trading strategies:

Throughout these experiments, we entertained some assumptions and implemented these strategies in specific ways that can have some important limitations or that can hinder the performance of the strategies and thus not paint a complete picture of how useful and profitable they can be. We will therefore expand on some specific points regarding our approach and explain why it can have a bad impact on the strategy returns as well as how to improve it / a better approach to it.

First and foremost, in multiple facets of our strategies, we fixed specific time windows, whether it be for representing how long a position will stay open, or to represent the sliding window used to compute the maximum mid-price. Setting a fixed time frame has pros and cons. First, it provides a clear time frame for the position to stay open, which can help with risk management and planning. Second, it also allows us to not spend any more computing time/power by checking other conditions or due to having a more complex model. And finally, it is not impacted by the market fluctuations. However, this last "advantage" can be very costly in some market conditions. In these specific market conditions, since the time period does not account for important events or news or changes in market trends, having such a strategy can lead to unexpected losses or missed opportunities. Moreover, given the unpredictable nature of the markets, the choice of using a fixed time period can be costly. And finally, the model will yield results specific only to the chosen time frame, one small change in the value will most likely return different results.

Second, the choice of capital allocation shows that the strategy is incomplete. Having to subdivide our capital to the number of positions we're planning to open conveys mistrust in our strategy and urges the need to implement an exit strategy (specific conditions or stop loss) in order to have a more correct and usable strategy.

Third, in the chosen strategies, we either went with a statistical approach: opening as many positions as possible when the conditions are met, or with a more restrictive approach: opening one position every time the conditions are met for a set time window. However, the approaches still lack an important side: the choice of exactly when to get in (for the case of the statistical approach) and exactly when to exit the position.

➔ The statistical approach allows us to test for the validity of the strategy and to get an overall idea on the performance of such strategy within the market data. This approach, however, must consider the fees for its results to be give us some insight on the actual performance of such a strategy. Since we did not limit the number of positions and with a starting capital of 1000 USD, a small number of 10 positions per hour will cause the starting capital for each position to be 0.7 USD. Using this information, we find the fees to cause a decrease of around 5.25 USD or a return loss of 0.525%. This means that our returns first need to be positive and above 0.525% just to be able to pay for the participation fees.

➔ The restrictive approach allows for at most one position per time period. As explained above, this approach can cost us quite a bit of opportunities. Even though we have a starting point (when the condition is met) we do not have an exit point. An exit point strategy can contain a stop loss, a take profit and even a specific condition we'd have set. This is even more relevant since we use this approach within the scope of orderbook imbalances. A correct approach would be to add a feature which tracks the changes in the orderbook imbalances and at each trend change imply a closing of the opened position and eventually of another one.

III. The Strategy

A. Traded Price and Volumes

In this section we will discuss how we computed the traded prices and volumes for each timestamp, which is a crucial step for our final model. We need to compute the actual volume that has been traded from one timestamp to the next one and the relevant Volume Weighted Average Price. However, we make a first assumption that there are no cancelled orders between timestamps. Let's take the simplest example where the bid and ask prices haven't moved but only the amount:

bids[1].amount	bids[1].price	bids[0].amount	bids[0].price	asks[0].price	asks[0].amount	asks[1].price	asks[1].amount
1.042	2726.85	1.7846	2726.93	2726.94	14.7489	2727.1	3.23
1.042	2726.85	0.5454	2726.93	2726.94	16.2689	2727.1	3.23

As we can see, the amount on the bid side at 2,726.93 USDT has fallen from 1.7846 to 0.5454 whereas the amount on the ask side has gone up to 16.2689 from 14.7489. The most likely scenario is that 1.2392 has been traded at the bid price of 2,726.93 USDT. We therefore have our traded price and volume between the two timestamps. But what if new bid orders have been passed at that price during the period? We could imagine that an investor added another 0.5 ETH at 2,726.93 USDT between the two timestamps. In which case the actual traded volume would be 1.7846 + 0.5 – 0.5454 = 1.7392. We could also imagine a more unlikely scenario where someone cancelled their limit order, say 0.5 again. Then the actual traded volume would be 0.7392. As we will see next, when orders consume the amount of several levels of bid (rep. ask) prices, this issue is extended to all the relevant bid (resp. ask) amounts.

Now let's imagine that there is enough traded volume to shift the prices upwards. In other words, buying market orders with volumes greater that the available amount on the first ask price. We will denote the ask amount as AA and the ask price as AP:

$$Traded\ Volume\ (t) = \sum_{0}^{4} (AA_{t-1}^{i} - AA_{t}^{0} * \mathbb{1}_{AP_{t-1}^{i}=AP_{t}^{0}}) * \mathbb{1}_{AP_{t-1}^{i} \leq AP_{t}^{0}}$$

$$Traded\ Price\ (t) = \frac{\sum_{0}^{4} AP_{t-1}^{i} * [(AA_{t-1}^{i} - AA_{t}^{0} * \mathbb{1}_{AP_{t-1}^{i}=AP_{t}^{0}}) * \mathbb{1}_{AP_{t-1}^{i} \leq AP_{t}^{0}}]}{Traded\ Volume\ (t)}$$

$$AA_{t}^{i} = asks[i].amount_{t}$$
$$AP_{t}^{i} = asks[i].price_{t}$$

We therefore iterate through the various asking prices of the previous timestamp that are lower or equal to the new $ask[0].price$ and sum their respective amounts.

asks[0]. price	asks[0]. amount	asks[1]. price	asks[1]. amount	asks[2]. price	asks[2]. amount	asks[3]. price	asks[3]. amount	asks[4]. price	asks[4]. amount
2726.92	1.947	2726.94	0.5607	2727.1	4.13	2727.22	4.1256	2727.24	4.8339
2727.22	1.1	2727.24	1.1545	2727.25	0.0455	2727.27	3.092	2727.36	0.367

As we can see the asking prices have increased, with the first ask price going from 2,726.92 USDT to 2,727.22 USDT which means that a buy market order was passed.

$$Traded\ Volume = 1.947 + 0.5607 + 4.13 + (4.1256 - 1.1) = 9.6633\ ETH$$

$$Traded\ Price = \frac{1.947 * 2726.92 + 0.5607 * 2726.94 + 4.13 * 2727.1 + (4.1256 - 1.1) * 2727.22}{9.6633}$$
$$= 2727.092\ USDT$$

As mentioned above, it is important to note that unobservable order flows could change these results. If there is a big order with volumes that are greater than the cumulated amount observable, we cannot precisely compute the traded price and volume and will compute an estimate from available data. The formula to compute the traded price and volume when a sell market order is passed is similar:

$$Traded\ Volume\ (t) = \sum_{0}^{4} (BA_{t-1}^{i} - BA_{t}^{0} * \mathbb{1}_{BP_{t-1}^{i}=BP_{t}^{0}}) * \mathbb{1}_{BP_{t-1}^{i} \geq BP_{t}^{0}}$$

$$Traded\ Price\ (t) = \frac{\sum_{0}^{4} BP_{t-1}^{i} * [(BA_{t-1}^{i} - BA_{t}^{0} * \mathbb{1}_{BP_{t-1}^{i}=BP_{t}^{0}}) * \mathbb{1}_{BP_{t-1}^{i} \geq BP_{t}^{0}}]}{Traded\ Volume\ (t)}$$

$$BA_{t}^{i} = bids[i].amount_{t}$$
$$BP_{t}^{i} = bids[i].price_{t}$$

In some rare instances, there might be market orders to buy and sell between two timestamps. This is easily identifiable as it results in a greater bid-ask spread. Indeed, the selling market order will shift the bid prices downwards and the buying market order will shift ask prices upwards.

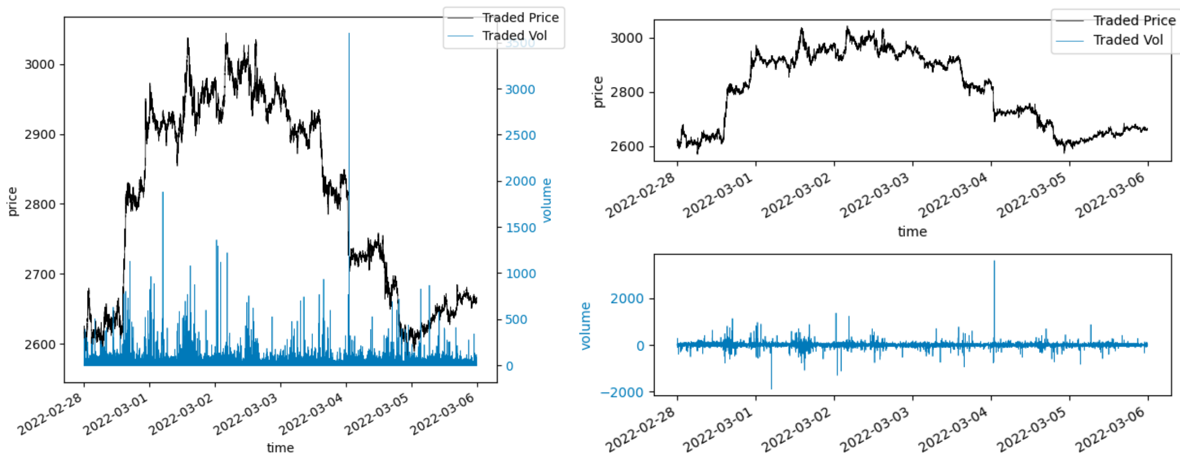
bids[2]. amount	bids[2]. price	bids[1]. amount	bids[1]. price	bids[0]. amount	bids[0]. price	asks[0]. price	asks[0]. amount	asks[1]. price	asks[1]. amount	asks[2]. price	asks[2]. amount
0.7424	2720.47	0.2038	2720.52	1.1	2720.62	2720.63	2.65	2720.81	0.244	2720.86	3.2694
1.4463	2720.28	0.6043	2720.34	0.1548	2720.47	2720.81	0.244	2720.86	3.2694	2720.88	0.425

Here, the $bids[0].price$ has gone down suggesting a selling market order and the $asks[0].price$ has gone up, implying a buying market order. The bid-ask spread soars to 0.34 from 0.01. When computing the traded price and traded volume, we must consider both sides:

$$Traded\ Volume = \underbrace{\big(1.1 + 0.2038 + (0.7424 - 0.1548)\big)}_{bid\ amounts\ executed} + \underbrace{\big(2.65 + (0.244 - 0.244)\big)}_{ask\ amounts\ executed} = 4.5414$$

$$Traded\ Price = \frac{\overbrace{1.1 * 2720.62 + 0.2038 * 2720.52 + 0.5876 * 2720.47}^{bids} + \overbrace{2.65 * 2720.63}^{asks}}{Traded\ Volume}$$
$$= 2720.6\ USDT$$

We can now plot the evolution of the traded price along with the traded volume for ETH over the six-day period:

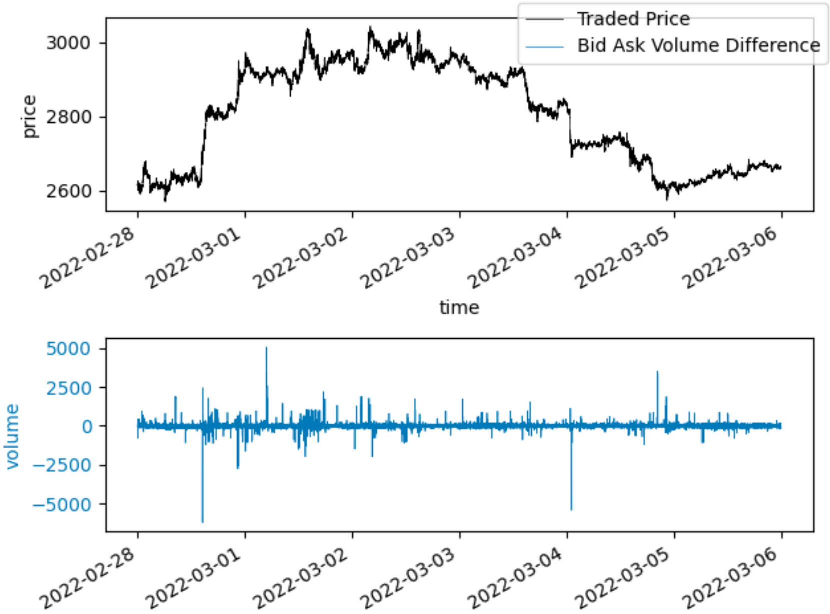


ETH traded price and volume over the six-day period

We previously discussed how volumes can send mixed signals. Analyzing above charts also suggests so. Indeed, the first chart shows a huge volume spike on March 24th 2022 when the price of ETH drops from roughly 2,800 USDT to 2,710 USDT. We have already seen this exact moment on the order books in the section on the order books. With the first graph only, one could misinterpret the spike as sell side orders putting downward pressure on prices. However, after taking a closer look, one can notice that this spike happens right after the price drop.

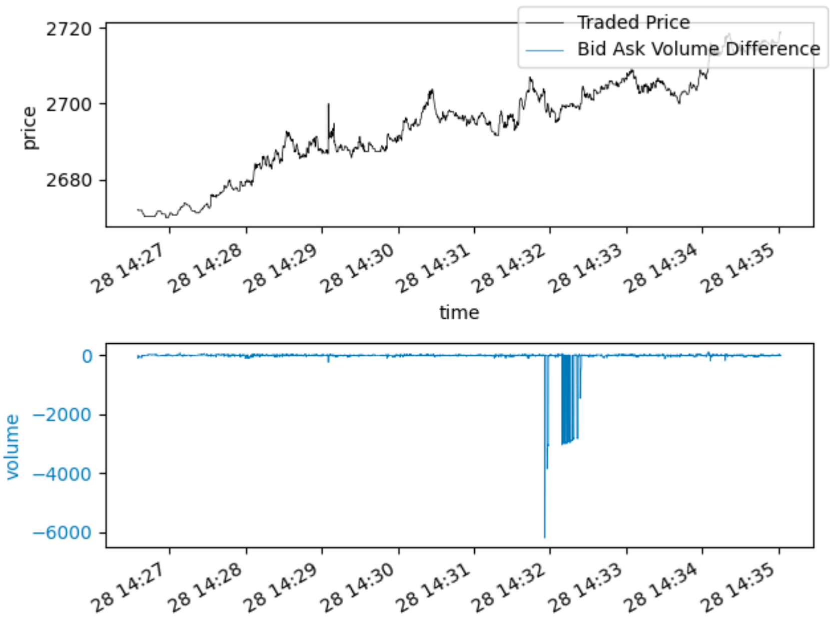
In the second graph, traded volumes are positive when it is a buying order (prices shift upwards and ask amounts get consumed) and negative otherwise. We can clearly see that this spike is a positive volume which suggests buying power. This is counter intuitive. The usual scenario would probably go as follows: the price drops suddenly and reaches a support level where the price moves sideways until the next breakthrough. However, this would imply huge volumes on the bid side and the selling power consuming this amount, struggling to get passed that price.

The actual scenario is very different. As mentioned previously, this is probably a huge selling order with a limit price of 2,710 USDT. All of the buying orders between 2,800 USDT and 2,710 USDT get executed explaining the sudden drop and then stagnates at 2,710 USDT where roughly 6,000 ETH are still waiting to be sold. After such a drop, buyers believing ETH is now trading at a discount jump in and pass market orders that get executed at 2,710 USDT. The ask amount at that level falls from 6,000 ETH to 2,313 ETH which explains the positive volume spike. Perhaps this can be confirmed when looking at the difference between bid and ask total volumes.



ETH traded price and bid-ask volume difference over the six-day period

The equivalent spike on the bid-ask volume difference is negative. Which suggests a significant imbalance in favor of the sell side. However, this metric can also send mixed signals. Although it helps understand the spike on March 4th, the negative spike on February 28th seems counterintuitive. The price jumps from roughly 2,600 USDT to 2,800 USDT but the bid-ask volume difference suggests an imbalance in favor of the sell side. Upon taking a closer look, this is exactly the "usual" scenario described above. The price increases from 2,600 USDT to 2,700 USDT where it reaches a resistance level characterized with huge selling volumes. However, the upward momentum is so strong that the buying power breaks that resistance level in less than 400ms.



ETH traded price and bid-ask volume difference 28/02/2022

These examples come to show how important it is to understand how momentum works and the limitations of an order book of depth 5. These are exceptions that a model should take into consideration. This imbalance, without context, could be interpreted as a sudden selling force that I likely to put significant downward pressure to the price when in fact it is just a resistance level that already existed on the order book but not observable as it was outside our window.

A strategy based on volumes and order book imbalances should therefore deal with above mentioned exceptions all while trading at a frequency that bears smaller fees than the profit generated from the trades. We have decided to develop a strategy that uses traded prices and traded volumes to detect and capture the wider trend.

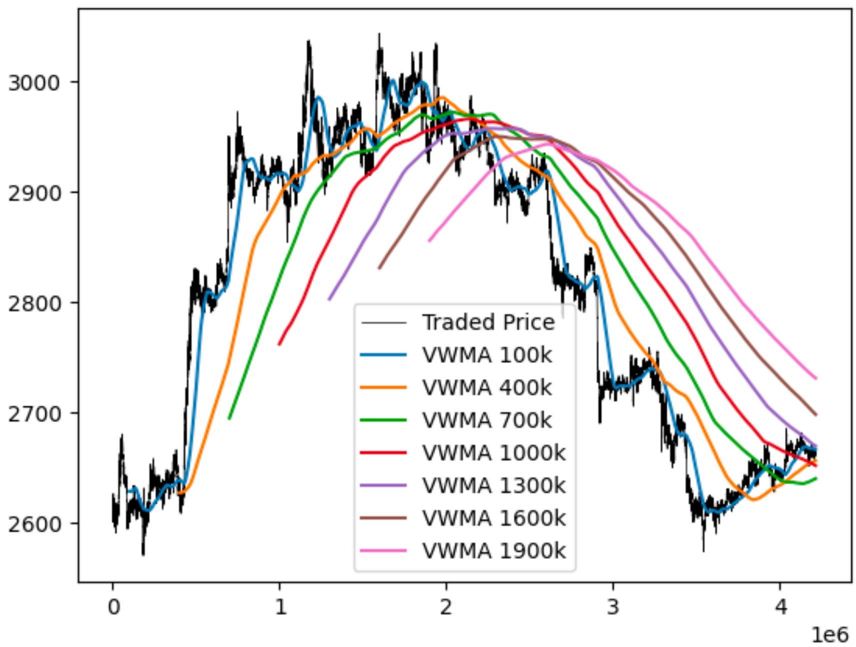
B. The Model

Our model uses Volume Weighted Moving Averages (VWMA) as a signal. It has the advantage of being a simple and easy to implement strategy. The more volatile the underlying the better the strategy works. Let's first introduce the VWMA formula:

$$VWMA(t, f) = \frac{\sum_{t-f}^{t} V_t * P_t}{\sum_{t-f}^{t} V_t}$$

$$f = window\ size$$
$$V_t = Traded\ Volume\ at\ time\ t$$
$$P_t = Volume\ Weighted\ Average\ Price\ at\ time\ t$$

The wider the window on which we compute the VWMA, the less sensitive it will be to the price. We can now plot the evolution of the underlying's price with different VWMAs:



ETH price evolution and VWMAs

Each VWMA is above graph have been computed with different time windows f. These time windows are expressed as the number of timestamps used to compute the VWMA. In Binance, 100,000 timestamps cover roughly 3 hours and 12 minutes. As we can see, the bigger f, the less sensitive the VWMA to the underlying price evolution. Moreover, the greater f, the smaller the timeframe on which we can compute the VWMA. The red curve is computed over a rolling window of 1 million timestamps which explains why it only starts after 1 million periods.

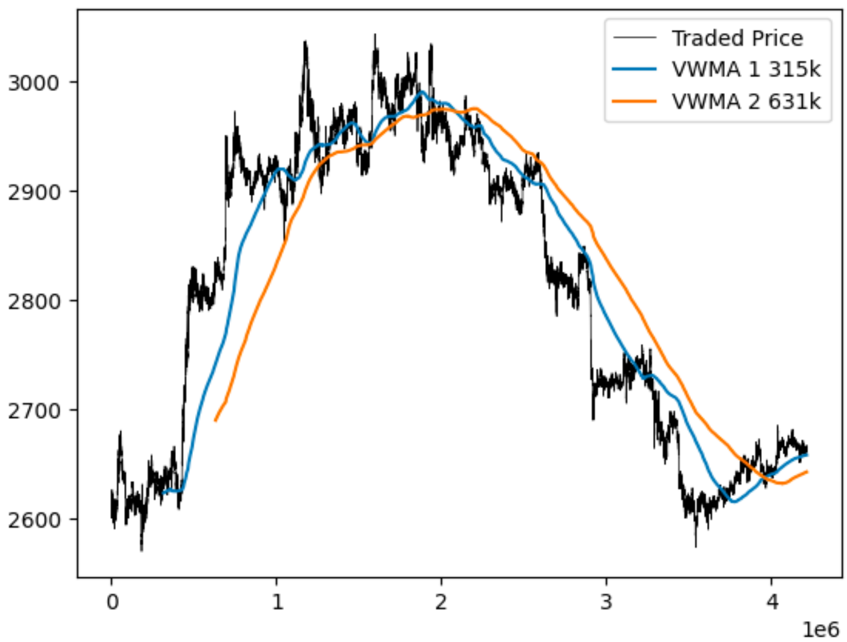
For this strategy, we need to choose 2 VWMAs that will act as a signal whenever they cross each other. Let's denote by $VWMA_1$ the VWMA with the smaller time window of the two, and $VWMA_2$ the one with the larger time window. The former will follow the price evolution closer, whereas the latter will be less sensitive to the price evolution. The strategy is simple and can be summarized as such:

$$\begin{cases} Long\ if\ VWMA_2 < VWMA_1 \\ Short\ if\ VWMA_2 > VWMA_1 \end{cases}$$

Therefore, if $VWMA_2$ is below $VWMA_1$ we have a long position. We will revert the position and go short the next time that both VWMAs meet and cross. Hence, when $VWMA_2$ goes above $VWMA_1$ we will switch from long to short.

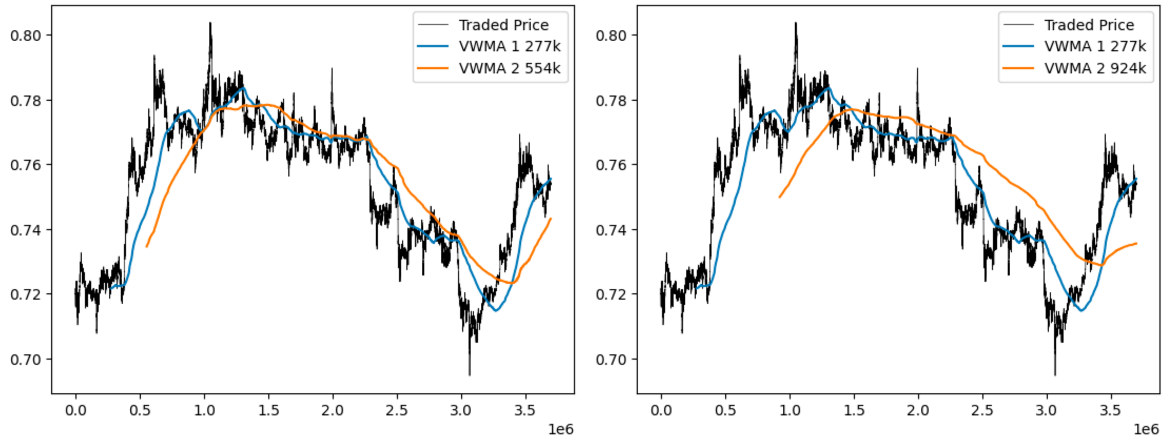
When choosing both time windows for the VWMAs, there are several factors to take into consideration. There is a balance to find. Indeed, a larger the window for $VWMA_2$ will allow us to capture a smaller portion of the underlying trend. Let's take the orange line as the reference for $VWMA_1$ (with $f_1 = 400k$). The first VWMA to cross the orange line is unsurprisingly the green line, which is the next VWMA with the closest time window f_2 to f_1. Hence, they will be the first ones to indicate a trend reversal. On the other hand, the pink line with $f_2 = 1.9M$ is the last VWMA to cross the orange line which means we will capture les of the upcoming downward trend.

This, however, does not imply that we should systematically pick VWMAs with smaller difference between time windows. Indeed, the downside to picking VWMAs that are closer to each other is that they will cross more frequently, especially when the prices move sideways. Fortunately for ETH on the six-day period, this is not the case. In fact, that is the perfect scenario for our strategy, the price is volatile and goes in very clear directions, first up then down and up again. However, as we will see later on, this is not the case for Ripple which is more indecisive. Last but not least, we need to bear in mind the lag between the start of the price timeseries and the relevant VWMAs. As mentioned, a wider time window f implies that we need to wait longer in the beginning before we see our signal. This might be an issue in this case where we only trade over a six-day period but we have to bear in mind that this isn't an issue when we will implement the strategy in the real world as we will use historical data, and will already have both VWMAs computed at the very beginning.



ETH price and relevant VWMAs

In this example, the first VWMA is computed on a time window $f_1 = 0.45\ days$ and the second one with $f_2 = 0.9\ days$. They cross twice, once at 2,937 USDT indicating a trend reversal from long to short and a second time at 2,633 USDT where we switch again to a long position. We are capturing a big part of the trends and benefit from clear price directions. This is the ideal scenario for a trading strategy on VWMA. Now let's look at the same strategy on the Ripple.



Ripple price and relevant VWMAs

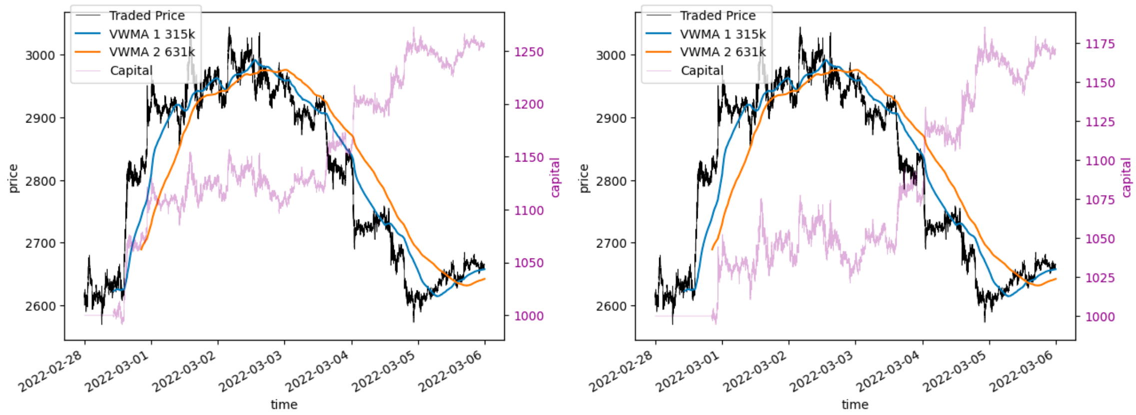
On the lefts we have $f_1 = 0.45\ days$ and $f_2 = 0.9\ days$. As we can see, the price is more indecisive, and the trends are more difficult to identity. In this scenario, the VWMAs cross a total of 7 times. More notably, they cross multiple times when the price oscillates between 0.76 and 0.78. Our strategy is therefore less adapted to periods of greater uncertainty.

Choosing $f_2 = 1\ day$ would render the chart on the right-hand side. The $VWMA_2$ is now less sensitive to price movements and follows the prices with more distance. Both VWMAs now only cross three times, rendering a much better result. However, it is important, once again, to note that we will enter trends with greater lag as we can see from the last trend reversal. Indeed, the price changes direction suddenly and jumps back up but the two VWMAs only cross halfway through the upward trend.

C. Backtest

We will now proceed to backtesting our strategy. To ensure we make no forward bias, we will go through each timestamp as if it were live data feed. All computations are used using periods prior to the current one. We wanted to develop an algorithmic strategy that can be used with different exchanges and underlyings. The window sizes chosen for $VWMA_1$ and $VWMA_2$ will be fixed otherwise an over calibration will lead to overfitting. We will test the performance of the algorithm over the six-day period considering the taker fees of each exchange.

The first position is determined by the direction of $VWMA_1$. An upward slope will indicate that we should start long and vice versa. You will notice that all underlyings start with a significant upward trend. If we wait for the $VWMA_2$ to appear we will indeed miss most of the trend. Although this might sound like cheating, if we had historical data before February 28th, we would have both VWMAs to start with, and they would probably cross around the time when the first VWMA is computed. Of course, we will also compute the performance with the first position depending solely on the relative position of $VWMA_2$ compared to $VWMA_1$ when it first appears.

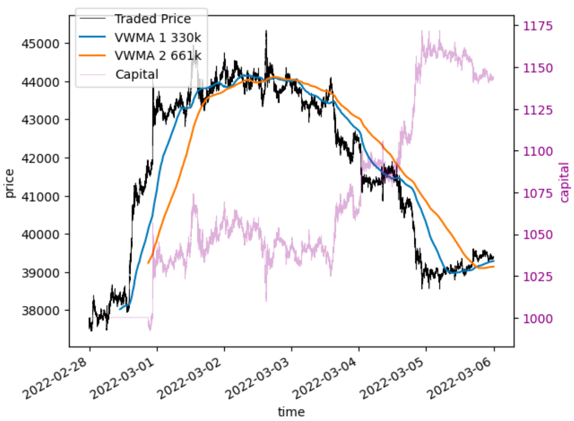


ETH price with relevant VWMAs and Capital evolution

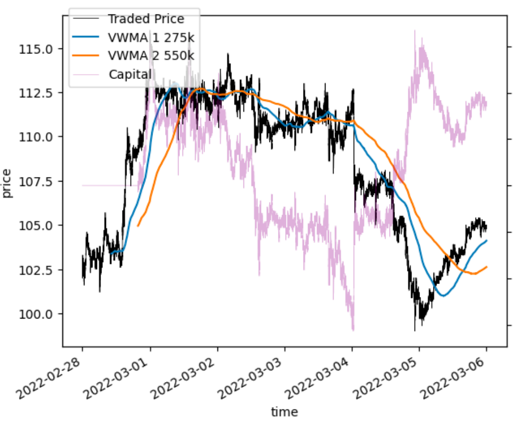
The window sizes to compute the moving averages has been set to 0.45 and 0.9 days. In the first scenario we enter a position as soon as we get the first $VWMA_1$ results. Given that the slope of the $VWMA_1$ curve is positive, we are long. Overall the strategy works very well for ETH during that six-day period with the capital growing from 1,000 USD to 1,256 USD.

On the right, we have the same strategy but different entry point. Indeed, we enter a position when the first $VWMA_2$ is computed, compare its relative position with $VWMA_1$ and trade accordingly. The gains are understandingly lower but still significant as our capital reaches 1,170 USD at the end of the period.

We will now test the strategy on several cryptocurrencies. We will only enter in the first position once we get the values for $VWMA_2$. Please note that the number of periods used to compute the VWMAs will change between cryptocurrencies and exchanges as the time frequency of the observations changes and 0.45 days might be 315k periods for ETH on binance but 330k for Bitcoin on Binance.



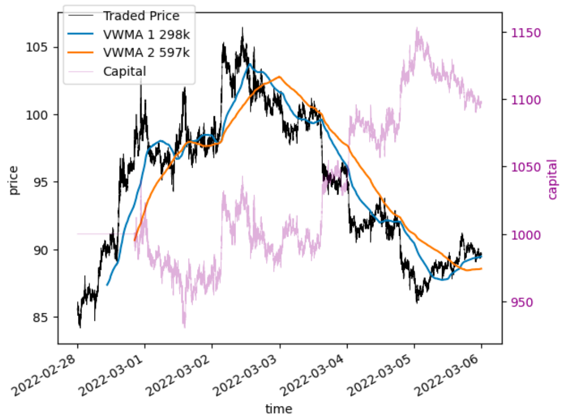
Bitcoin 1143.98



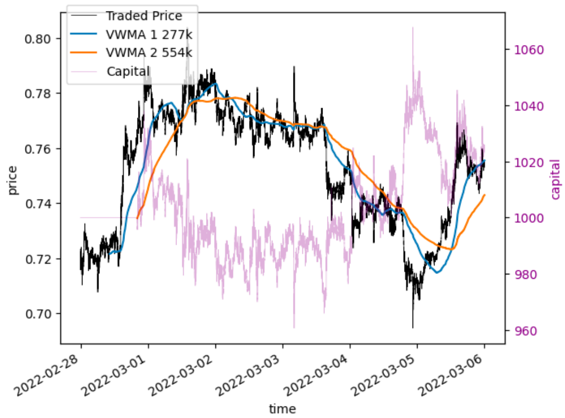
Litecoin 1035.21

	ETH	Bitcoin	Litecoin	Solana	Ripple
Binance	1170.73	1143.98	1035.21	1097.013	1024.6

Table of the capital at the end of the period for different assets



Solana 1097.013



Ripple 1024.6

As we can see from above charts, our algorithm works on any cryptocurrency, but the performance varies greatly from one to another. The evolution of our capital when trading on Litecoin is very volatile although it ends up being slightly profitable. We can also note that the algorithm works best for cryptos with clearly identifiable trends like ETH, BTC and Solana and fairs worse when prices stagnate around a level like Litecoin and Ripple.

We mentioned that our strategy should work on any exchange. Given we have a starting capital of 1,000 USD, we don't have to worry about liquidity too much, especially when trading Ethereum. Of course, we will consider each specific taker fee. As we will see from below table, the returns are very similar from one exchange to another. The major difference will lie on the fees, the executed prices of each trade and the accuracy of the computation of the traded prices and volumes. Indeed, as mentioned above, the smaller the time gap between timestamps, the smaller the probability of seeing one market order wipe out more than the observable volume on our order book. Recall that binance has the smallest time frequency, hence the whopping 91,755 instances where we couldn't precisely compute the traded price and volume.

	Binance	Kraken	FTX	Bitmex
Capital	1170.73	1178.09	1160.66	1144.15
Big orders	91,755 (2.17%)	2,316 (0.02%)	6,752 (0.09%)	1,851 (0.02%)

Table of ending capital on ETH for several exchanges

The average return across these 4 exchanges is 16.34% over the six-day period. Unfortunately, we couldn't run the algorithm on Coinbase due to the sheer size of the data, struggling even to load the order book for a single day. As we can see from the table, Bitmex, although it has the lowest fees, renders the worst returns of them all and by quite a margin. Indeed, 14.4% returns compared with 17% returns on Binance is a significant gap. Let's take a closer look at the trade log for both:

Index	Timestamp	Side	VWAP	Vol
631997	28/02/2022 20:27:12	buy	2822.1	0.35399171
2086039	02/03/2022 19:49:24	sell	2936.83	0.70727543
3957195	05/03/2022 13:48:29	buy	2633.74	0.87855211

Binance trade log on ETH

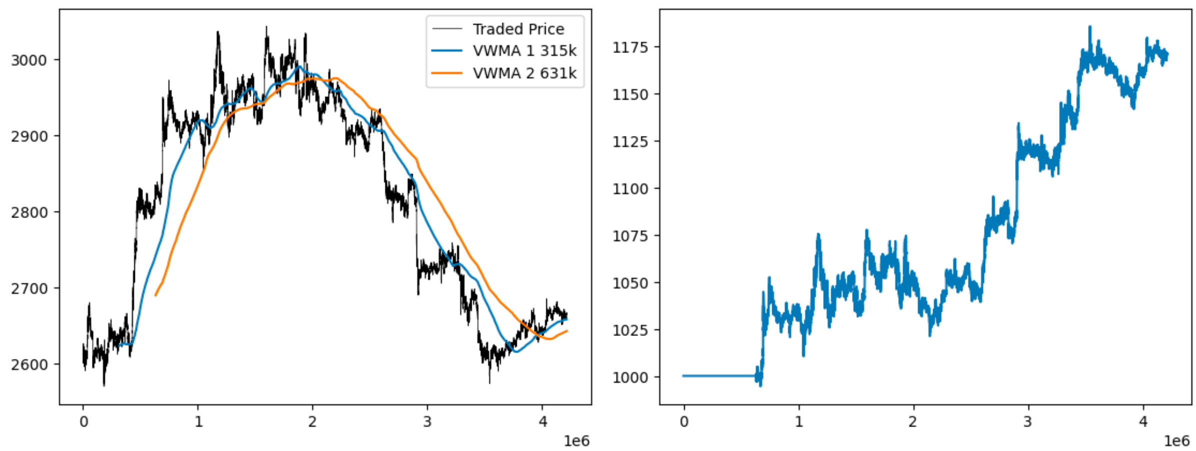
Index	Timestamp	Side	VWAP	Vol
1658658	28/02/2022 16:50:52	buy	2811.45	0.3555105
6195474	02/03/2022 18:40:51	sell	2935.5	0.71078654

10925395	05/03/2022 19:49:12	buy	2671.45	0.85782795

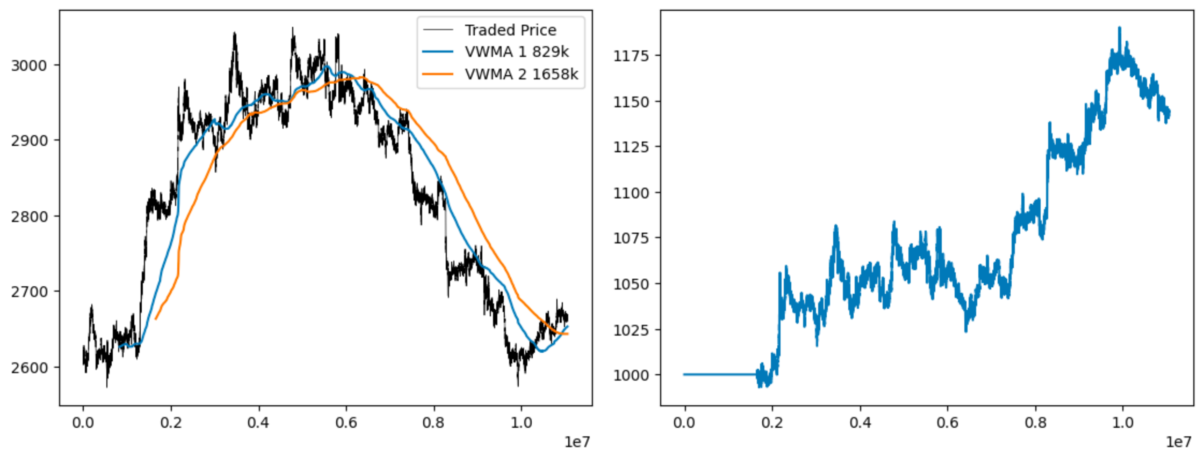
Bitmex trade log on ETH

At first the trade logs benefit Bitmex. Indeed, we see that we execute the first trade to buy at 2,811 USD which is lower than 2,822.1 USDT on Binance (assuming a 1:1 peg USD/USDT). Then, both sell at around 2,936 USD, at which point the trade on Bitmex shows greater volume at 0.71 ETH sold against 0.7 ETH on Binance. Recall that going from long to short and vice versa requires the algorithm to sell or buy twice the open position's quantity. It all comes down to the last trade, where there is a huge difference in the executed price. Indeed, we buy at 2,634 USDT in Binance versus 2,671 USD on Bitmex. Here is where most of the worst performance for Bitmex comes from. As we can also see, the time at which the trades take place are different for both exchanges.

This would point towards a discrepancy in the computation of the VWMAs, which don't appear to be crossing at the same points in time. The difference might come from the fact that Bitmex has smaller time gaps between observations which reduces the number of instances of "big orders" hence the computation of the traded price and volume (and VWMAs). In fact, Bitmex only shows 1,851 such instances or 0.02% against 2.17% when using Binance's order book. However, Kraken's rate of big order instances is just as low as Bitmex's and doesn't appear to have the same issue in terms of performance, it even beats Binance's.

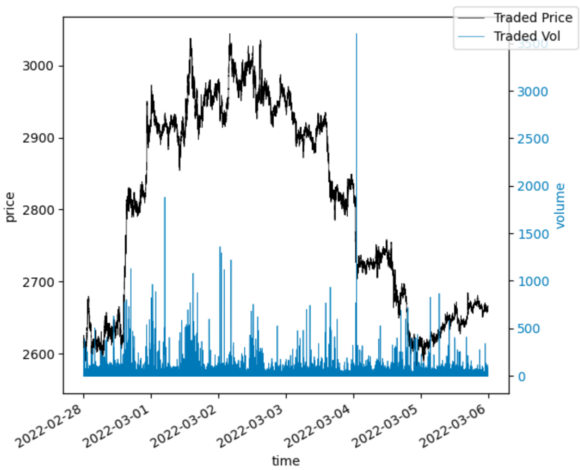


ETH and capital evolution on Binance

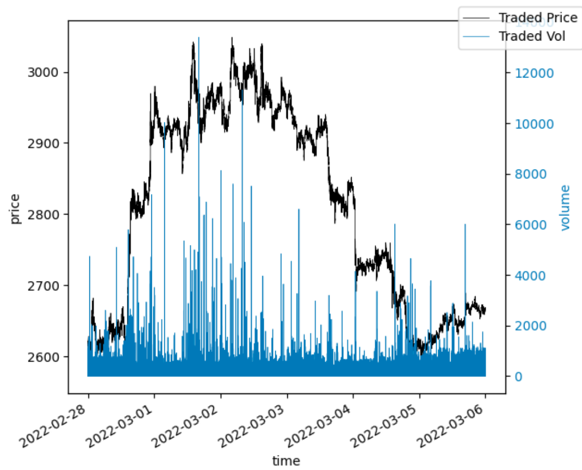


ETH and capital evolution on Bitmex

As shown in the graphs, the main difference comes from the computation of the VWMAs which comes most certainly from the traded volume.



ETH on Binance



ETH on Bitmex

The cumulated traded volume adds up to 7,233,554 ETH over the six-day period on Binance against 67,441,202 ETH for Bitmex. Such volume discrepancies have already been noted at the beginning of this report where we were wary regarding the validity of these figures. One way to overcome such issue could be to compute the signals from Binance's order book and trade on Bitmex to benefit from lower fees. However, the impact of transaction costs on such a strategy with very few trades is negligeable.

IV. Enhancements

A. Scalability

We want to see how the algorithm would perform with greater initial capital invested. Although it benefits from possible lower fees, it also suffers from worse execution prices. A larger volume passed on a market order will probably outsize the available volumes at the first bis and ask prices which implies reaching deeper in the order book to be fully executed. Furthermore, we might not have enough order book depth to execute the totality of our orders. However, we will discuss how we can deal with scaling the model.

We will test the performance with different initial capitals however it is very important to note that backtesting is limited. Indeed, when we have big volumes to trade, each instance when we make a trade might alter completely the order book, in which case the following timestamps should consider the shift in prices caused by our trades. Unfortunately, we don't know how the order book would have evolved had we made these trades in real life. It is therefore important to bear in mind that our trades, regardless of the size and possible shift in prices that they cause, will not be reflected on the rest of the order book. When testing for 10,000 USD initial capital, the return was nearly the same as for 1,000 USD, at around 17.07%. When testing with 100,000 USD, that return falls ever so slightly to 17.064%. That is probably because Binance fees are still the same for all three notionals and our algorithm makes only 3 trades in that period so the worse execution price for bigger notionals is poorly reflected on the returns. However, any initial capital of 100,000 USD or above cannot be backtested. Indeed, for every trade, there isn't enough order book depth to fully execute the full amount. Together with an impossibility to alter the future course of the order flow, it has no meaning testing such big notionals with the available market data.

		1K	10K	100K	1M
First Trade	VWAP	2822.1	2822.1	2822.17988	2822.17988
	Vol	0.35399171	3.53991708	26.4457	26.4457
	Executed	Fully	Fully	Partially	Partially
Second Trade	VWAP	2936.83	2936.83	2936.79388	2936.79388
	Vol	0.70727543	7.07275433	17.6733	17.6733
	Executed	Fully	Fully	Partially	Partially
Third Trade	VWAP	2633.74	2633.74	2633.83147	2633.83147
	Vol	0.87855211	8.78552112	20.5865	20.5865
	Executed	Fully	Fully	Partially	Partially

Trade Log for different initial Capitals

As we can see, the trade logs are identical for 100k and 1M USD starting capitals which suggests that there isn't enough order book depth to fully execute the volumes.

To reduce the impact of too big orders, we could trade on various exchanges simultaneously. Trading the maximum optimal volume of less liquid but cheaper exchanges like Bitmex and FTX and then executing the rest on more liquid exchanges such as Binance and Coinbase.

B. Improvements

There is still a lot of room for improvement. Having more data to backtest the strategy would probably show the limits of the strategy. As mentioned previously, the VWMA risk crossing several times in consecutive manner when prices swing sideways hence bearing the transaction costs as well as any loss from the trades themselves. However, even then, the VWMAs will most likely cross around similar price levels which means the magnitude of the loss incurred from those consecutive trades shouldn't be too big.

Furthermore, the traded prices and volumes being at the heart of the strategy, we should think of ways to improve the precision of their computation hence of the VWMAs too. There are two obvious improvements that can be done in this regard: (i) increasing the depth of the order book and (ii) combining all the order books into a single meta book. The former will solve the issue raised from big market orders that wipeout the entirety of the volume observable with the current depth and more that isn't picked up by the model. In such instances, we would clearly identify all the executed prices with their respective volumes and compute the actual VWAP. The meta order book would provide us with the complete trade flow for any given asset.

Another possible extension of this strategy would imply dynamic windows for the VWMAs computations depending on the underlying's volatility. Less volatility implies that the price oscillates around a mean value, i.e. moving sideways, in which case we might want to increase the window sizes for the VWMAs to be less sensitive to price movements hence crossing less. More volatility suggests the price moves in one direction and decreasing the window sizes would allow us to capture a greater portion of the trends. Such dynamic calibration could depend on a given period prior to the current instant.

Finally, for risk management purposes it might be interesting to add stop losses and take profits. Ideally, we would set fixed stop losses, say -1% on a given trade and dynamic take profits. To capture as much profit possible from a trade, stop losses could be set at $\pm x\%$ of the trading price at each given time interval. We could imagine a take profit updated every 5 minutes to be set at $\pm 1\%$ of the trading price at that moment. Then, if the price keeps moving in the same direction over the next 5 minutes, the position stays open, and the take profit is updated again with the new trading price at the end of the period. However, if the price moves in the opposite direction during those 5 minutes, we exit the position and wait for the next signal.

Appendix

Asset	Metrics	Binance	Coinbase	FTX	Bitmex	Kraken
ethereum	row_count	741653	14826721	1519761	2294588	2020165
ethereum	bid_ask_spread	0.02278494	0.27915559	0.1644752	0.34304184	0.26929792
ethereum	avg_diff_ask_0_1	0.11866598	0.0785504	0.16650184	0.32226672	0.18408822
ethereum	avg_diff_ask_1_2	0.07480719	0.07393368	0.16406514	0.20755096	0.19361273
ethereum	avg_diff_ask_2_3	0.06499331	0.06887904	0.15178137	0.19385994	0.14948824
ethereum	avg_diff_ask_3_4	0.05657267	0.06688697	0.14311079	0.18088877	0.12804158
ethereum	avg_diff_bid_0_1	0.12232459	0.08561403	0.17188472	0.32030445	0.24289787
ethereum	avg_diff_bid_1_2	0.07638868	0.07796437	0.16343037	0.20100343	0.1921781
ethereum	avg_diff_bid_2_3	0.06686015	0.07463128	0.14915319	0.17902998	0.13951963
ethereum	avg_diff_bid_3_4	0.05784448	0.07277232	0.13932559	0.16053431	0.11846467
ethereum	avg_vol_ask_0	7.70957154	1.37050509	8.0978648	89.4562819	22.4602742
ethereum	avg_vol_ask_1	2.52494123	1.34831395	5.5245285	74.8194399	9.77656698
ethereum	avg_vol_ask_2	3.06567296	1.50244583	6.52614036	86.9494062	7.51614717
ethereum	avg_vol_ask_3	3.19217343	1.76073788	7.50046678	94.6437692	6.48457828
ethereum	avg_vol_ask_4	3.34010843	2.04114157	8.65890261	109.705235	5.98752267
ethereum	avg_vol_bid_0	6.5394394	1.22115748	9.20552612	127.230915	29.7357333
ethereum	avg_vol_bid_1	2.49310556	1.21477502	5.90720933	94.9786454	11.7744878
ethereum	avg_vol_bid_2	2.95268054	1.32834073	6.76633796	119.167211	8.83616323
ethereum	avg_vol_bid_3	3.13081145	1.55900362	7.68693269	123.182223	7.86057291
ethereum	avg_vol_bid_4	3.18221996	1.82107928	8.91283563	141.465868	7.36031293

Table 1: Summary order book statistics for ETH on various exchanges for February 28th 2022

Notional	Taker Fees	Taker Exch	Maker Fees	Maker Exch
10 000.00	0.001	binance	0.001	binance
50 000.00	0.001	binance	0.001	binance
100 000.00	0.001	binance	0.001	binance
250 000.00	0.001	binance	0.001	binance
500 000.00	0.001	binance	0.001	kraken
1 000 000.00	0.001	binance	0.0008	kraken
2 500 000.00	0.001	binance	0.0006	kraken
5 000 000.00	0.001	binance	0.0004	kraken
10 000 000.00	0.001	binance	0.0002	kraken
20 000 000.00	0.001	binance	0.0002	kraken
100 000 000.00	0.001	binance	0.0002	kraken
120 000 000.00	0.001	binance	0.0002	kraken
200 000 000.00	0.0009	binance	0.0002	kraken
300 000 000.00	0.0008	binance	0.0002	kraken

500 000 000.00	0.0008	binance	0	coinbase
1 000 000 000.00	0.0007	binance	0	coinbase
2 500 000 000.00	0.0006	binance	0	coinbase
5 000 000 000.00	0.0005	binance	0	coinbase

Table 2: Best fees between Binance, Coinbase & Kraken