Orderbook Delta price reaction research

Table of Contents

Orderbook Delta defenition

The orderbook delta is calculate by the difference between the sum of the bid and ask orders at a certain depth. Formular:

$$\Delta_{x\%} = \sum_{i=1}^{x\%} \Delta_i$$

- $\Delta_{x\%}$ is the sum of the orderbook delta for the last $x\%$ of the orderbook.
- Δ_i is the orderbook delta for the i-th level of the orderbook.

Outlier Detection Using Mean and Standard Deviation (Z-Score Based Outlier Detection)

Normal Range

What I want to test is how price reacts to anomalous orderbook delta movements, particularly in scenarios where unrealistic or clearly outlying values are detected. In cryptocurrency markets, such inefficiencies can be caused by various events, one example is liquidation events that interact with passive demand order stacked zones. During these events, the orderbook delta exhibits significant increases, providing a clear signal of market stress. This research will focus on understanding the relationship between rapid delta movements and how price reacts after these events.

My Hypothesis

- I expext realized volatility to increase after an outlier is detected.
- I expext a return to the mean after a strong outlier is detected.

Normal Range

$$\mu(\Delta) \pm 2\sigma(\Delta)$$

This means most data points (about 95% if normally distributed) are expected to lie within this range.

Outlier Condition

A value is considered an outlier if:

$$\Delta < \mu(\Delta) - 2\sigma(\Delta) \quad \text{or} \quad \Delta > \mu(\Delta) + 2\sigma(\Delta) \tag{1}$$

This is a simple Z-score based outlier detection.

- Δ - Orderbook Delta Depth with a certain depth I will test on: $\Delta_{1\%}$ $\Delta_{2.5\%}$ $\Delta_{5\%}$ from Coinbase (BTC/USD)
- This basically means we take a delta of the Bid and Ask orders which are in a range of x% from the current price.
- $\mu(\Delta)$ — Mean of the last 1440 values of Δ before time t
- $\sigma(\Delta)$ - Standard deviation over the last 1440 Δ values before time t

Idea behind

- This method assumes data is roughly normally distributed.
- Using 2σ captures approximately 95% of data points under a normal distribution.
- You can adjust the multiplier (e.g., 3σ) for stricter or looser thresholds.

Future Plans

- Test on more data
- use rolling windows (e.g. 1 day or 1 week) for local context.
- Compare sensitivity with +- 1.5σ or +-2.5σ $\rightarrow$ optimize for best results

Measuring Volatility After Price Outlier Detection

$$r_t = \frac{P_t - P_{t-1}}{P_{t-1}} \tag{2}$$

Dictionary of Terms

- P_t Asset price at time t.
- $r_t = \frac{P_t - P_{t-1}}{P_{t-1}}$ – 1-minute price return at time t.
- $\sigma_t^{(15)}$ – Realized volatility: the standard deviation of the next 15 one-minute returns,

$$\sigma_t^{(15)} = \sqrt{\frac{1}{14}\sum_{i=1}^{15}\left(r_{t+i} - \bar{r}_t\right)^2}, \quad \bar{r}_t = \frac{1}{15}\sum_{i=1}^{15} r_{t+i}. \tag{3}$$

  aligned so that at time t it measures volatility over $t+1$ to $t+15$.

In Python code

```
import pandas as pd
df = pd.read_csv(file_path)
df.set_index('timestamp', inplace=True)
#Compute 1-min return of delta_5

df['r_t'] = df['price'].pct_change().fillna(0)



#compute rolling std of the future 15 min window

window = 15

#rolling on r_t, then shift forward so index t hold vol of t+1...t+15
df['future_vol_15] = (
    df['r_t']
    .rolling(window=window)
    .std()
    .shift(-window)
)
```

Statistical evidence

Once an outlier is detected (1) inside of the Orderbook Δ, we calculate the15-minute ahead realized volatility using Equation: (3)

if a Δ_t values is flagged as an outlier (1) we record

$$\sigma_t^{(15)} \;=\; \sqrt{\frac{1}{14}\sum_{i=1}^{15}\big(r_{t+i}-\bar{r}_t\big)^2},$$

We then form two samples over our full dataset which during this test includes 104 957 one minutes intervals of P and Orderbook Δ:

$$\mathcal{S}_{\text{out}} \;=\; \{\sigma_t^{(15)} : t \text{ is an outlier}\}, \quad \mathcal{S}_{\text{non}} \;=\; \{\sigma_t^{(15)} : t \text{ is not an outlier}\}.$$

Sample mean results:

$$\overline{\sigma}_{\text{out}}^{(15)} = 0.0006244, \qquad \overline{\sigma}_{\text{non}}^{(15)} = 0.0005138,$$

This concludes an increase of r_t of roughly 21.5%

To check Statistical evidence

- a two-sample *t*-test (unequal variances), which yields

$$T = 24.72, \quad p = 4.79 \times 10^{-132},$$

- a Mann–Whitney *U*-test, which returns

$$p = 4.02 \times 10^{-157}.$$

Optimising for best Z-Score thresold for outliers

As state inside of (1) we use a Z-Score thresold of 2 to detect outliers. I now want to see if by any chance there is a better Thresold value

To compare the outliers Volatility with the non outliers volatility I will use the following formular:

$$U(z) = \frac{\overline{\sigma}_{\text{out}}^{(15)}}{\overline{\sigma}_{\text{non}}^{(15)}} \tag{4}$$

First I run an optimization for the thresholds of the Z-Score to find the best thresold value for the outliers on a 45 days dataset. After that I compare the result with a 107880 minutes dataset. Where I also run an optimization for the thresholds of the Z-Score to find the best thresold value for the outliers.

Top 3 z-values with largest volatility uplift 69811-minutes sample

z	N_{out}	$U(z)$ (%)	Mann–Whitney p
3.8	221	+58.36	1.913×10^{-51}
3.9	166	+61.99	4.227×10^{-41}
4.0	117	+58.36	8.228×10^{-28}

Same z-values on extended dataset 107880-minutes sample

z	N_{out}	$U(z)$ (%)	Mann–Whitney p
3.8	644	+51.73	2.549×10^{-77}
3.9	561	+53.28	6.150×10^{-88}
4.0	479	+50.90	5.517×10^{-59}

Top 3 z-values with largest volatility uplift 107880-minutes sample

z	N_{out}	$U(z)$ (%)	Mann–Whitney p
4.8	214	+65.10	6.475×10^{-33}
4.9	197	+66.81	3.693×10^{-29}
5.0	181	+70.53	6.599×10^{-28}

Measuring avearge return after price outlier detection

Formulars

Once a Δ_t Outlier is detected we calculate the 15-min forward return of BTC/USD price

$$\mathrm{Ret}_t^{(15)} = \frac{P_{t+15} - P_t}{P_t} \tag{4}$$

We then differentiate between a bullish and a bearish outlier. Which is already defined (1)

$$\overline{\mathrm{Ret}}_{\mathrm{bull}}^{(15)} = \frac{1}{|\mathcal{T}_{\mathrm{bull}}|} \sum_{t \in \mathcal{T}_{\mathrm{bull}}} \mathrm{Ret}_t^{(15)} \tag{7}$$

$$\overline{\mathrm{Ret}}_{\mathrm{bear}}^{(15)} = \frac{1}{|\mathcal{T}_{\mathrm{bear}}|} \sum_{t \in \mathcal{T}_{\mathrm{bear}}} \mathrm{Ret}_t^{(15)} \tag{8}$$

Dictionary of Terms

- Price at a certain time: P_t
- 15-min forward return: $\mathrm{Ret}_t^{(15)}$

```
#compute 15-min forward return of BTC/USD price
```

1 Combining indicators for strategy

Underlying strategy Bias

Every single parameter has to fight to be implemented into my strategy. To get some kind of filter since we are working with an asset which has clear trends and isn't stationary we need to do some trend identification. I'll call it the underlying bias. Some simple examples are for a bias are:

- $\Delta_{5\%} < 0$ (more passive demand than supply)
- $EMA_{\mathrm{n}} > P_t$ (EMA is an exponential average of $P_t - n$, $P - T$ is the price at time t)
- $EMA_{\mathrm{n}} > EMA_{\mathrm{x}}$ (Crossing of two EMAs with different time periods n and x)

But I want to get some clear trend identification where we divide into the three different categories:

Ideas I want to test on:

- Market Structure defenition (Higher Highs and Higher Lows)
- Realtive strenght index on Price and $\Delta_{x\%}$

Finding an edge

No clear path

Finding an edge is a very hard task. There is no clear path to success. You kinda have to try by trail and error every error could be a step further but also a potential path into a dead end.

Sources: TRDR This platform allow you to use different kind of metrics on different Timeseries datasets (BTC/USD Price, orderbookdelta and Open Interest

2 Comparison of Delta Indicators on 39,694-minute sample

Period	$N_{signals}$	$U(z)$ (%)	T-statistic p
Delta2.5 Strategy (N_{total} = 9,182)			
$5m$	$9,182$	$+52.03$	1.851×10^{-41}
$60m$	$9,182$	$+55.94$	3.537×10^{-4}
$360m$	$9,182$	$+59.20$	12.524×10^{0}
Delta5 Strategy (N_{total} = 12,234)			
$5m$	$12,234$	$+51.38$	3.382×10^{-7}
$60m$	$12,234$	$+53.44$	4.010×10^{-1}
$360m$	$12,234$	$+57.04$	13.617×10^{0}

Table 1: Comparison of Delta2.5 vs Delta5 strategies across different time periods

- $N_{signals}$ represents number of trading signals
- $U(z)$ represents positive return ratio
- T-statistic p represents statistical significance

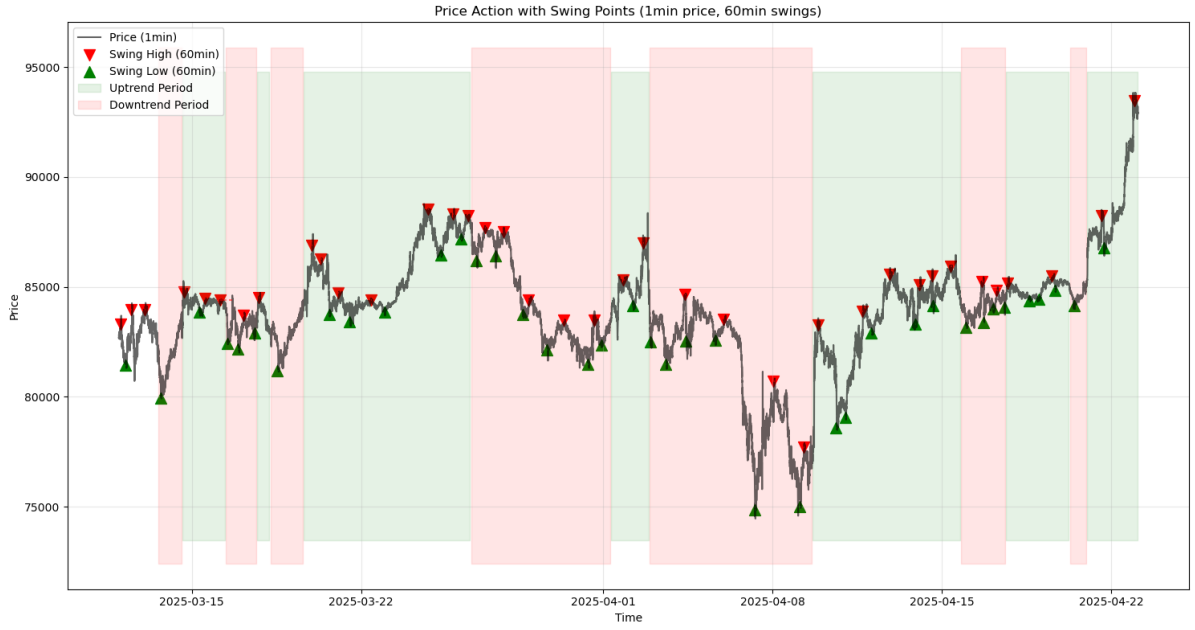

Figure 1: 1 min interval price with 60 min swing points and a n value of 8

Swing Point Bias Visualization

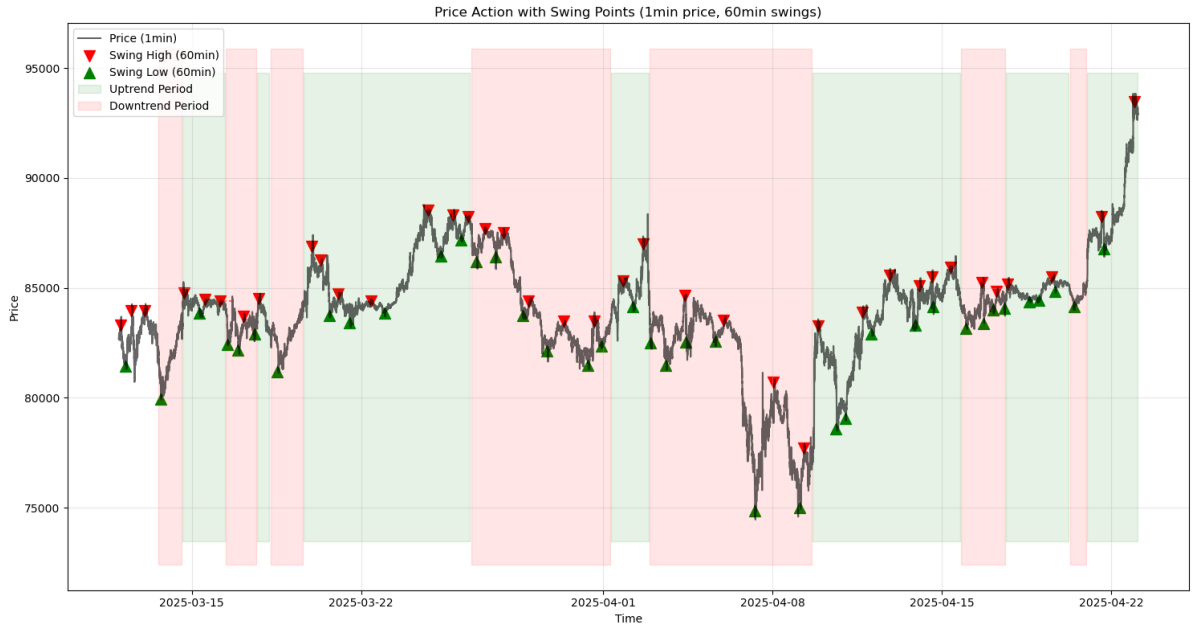

Figure 2: Price Action with Swing Points (1min price, 60min swings) showing trend periods

This visualization demonstrates how swing points are identified on a 1-minute price chart using a 60-minute window for swing point detection. The green and red shaded areas represent uptrend and downtrend periods respectively, while red triangles mark swing highs and green triangles mark swing lows.

Combining Indicators

Here I visulised the swing points, the EMA spread and the 100 outliers with the highest Z-Score in the same plot.

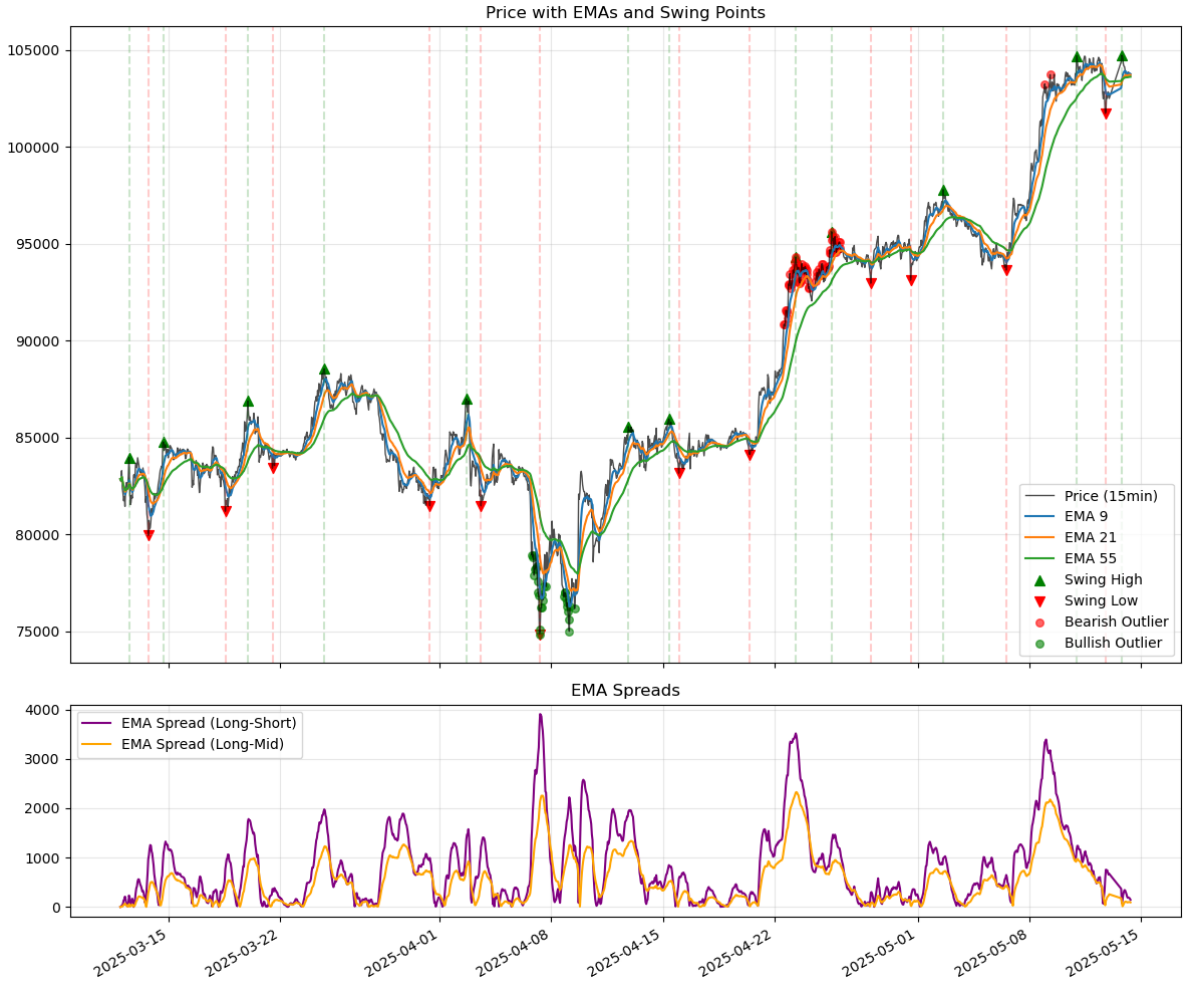

Figure 3: combined indicators png

1

[1]Chart made with Matplotlib and Seaborn

Mapping Orderbook delta EMA with standard deviation shifing

Formulars

$$EMA_t = \alpha \cdot \Delta_t + (1 - \alpha) \cdot EMA_{t-1} \tag{5}$$

Dictionary of Terms

- EMA_t - Exponential Moving Average at time t
- Δ_t - Orderbook delta at time t
- α - Smoothing factor
- $\alpha = \frac{2}{n+1}$ where n is the period of the EMA
- $\sigma(\Delta)$ - Standard deviation of the orderbook delta with a

We'll now shift the EMA by the standard deviation of the orderbook delta

From outliers to strategy

Progress description

I had a pretty hard time going from developing a logic for the outliers of the Delta. Finding that volatility increases after outliers was a good first find which didn't take long to find. But finding some a defenition condtion is meet directional bias was very difficult. By directional bias I mean a price direction which is not random after some kind of condition. It was pretty clear from the begging on that the outliers by there selfe won't offer any kind of directional bias

An outlier is defined as:

- (1). $\Delta < \mu(\Delta) - 2\sigma(\Delta) \quad \text{or} \quad \Delta > \mu(\Delta) + 2\sigma(\Delta)$

Then since we Bitcoin clearly is not a stationary asset I needed to find some kind of trend identification. I found that the swing points are a good indicator for this. We determined a trend by using looking at swingpoints. A swing point is a local extrema of a certain period.

Where did I start my research

```
#compute swing points
swing_points = swing_points(df['price'], period=n)
```

Parameters of the swing_points function:

- n is the lookback period
- $df['price']$ is the price column of the dataframe (Which is a timeseries)
- P_t is a value of $df['price']$ at time t

We basically got through the time series dataset and look back n P_t values. The highest and lowest points inside of that specific lookback period are the swing points. After that we detetermine if price is making higher highs or lower lows. We check this by storeing the last swing points and wait till we are eighter making a higher high or a lower low.

Finding an edge

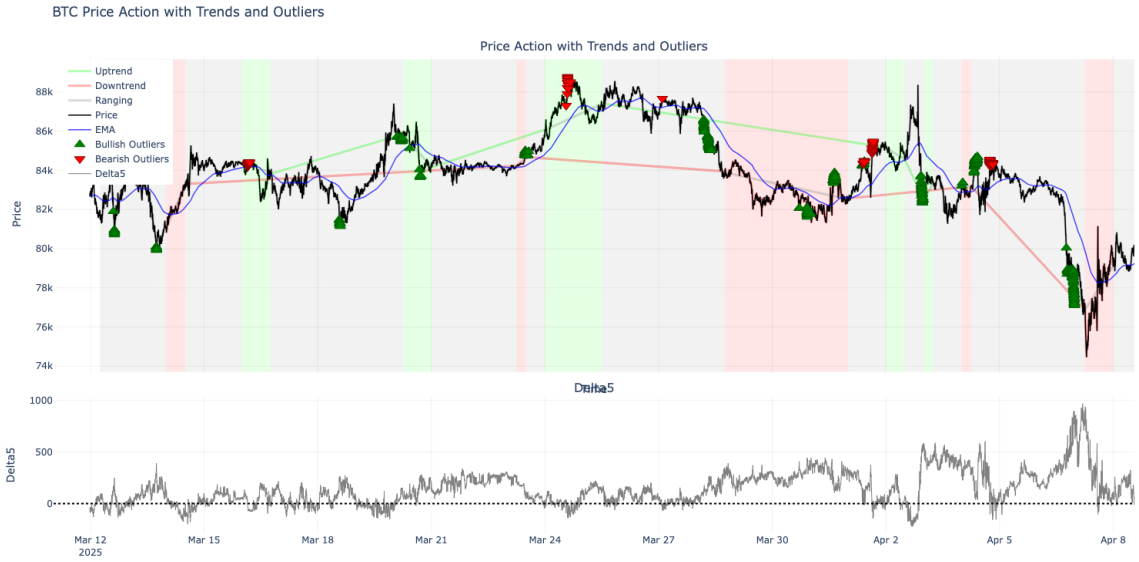

Figure 4: Visualization of Δ_t outliers and swing points

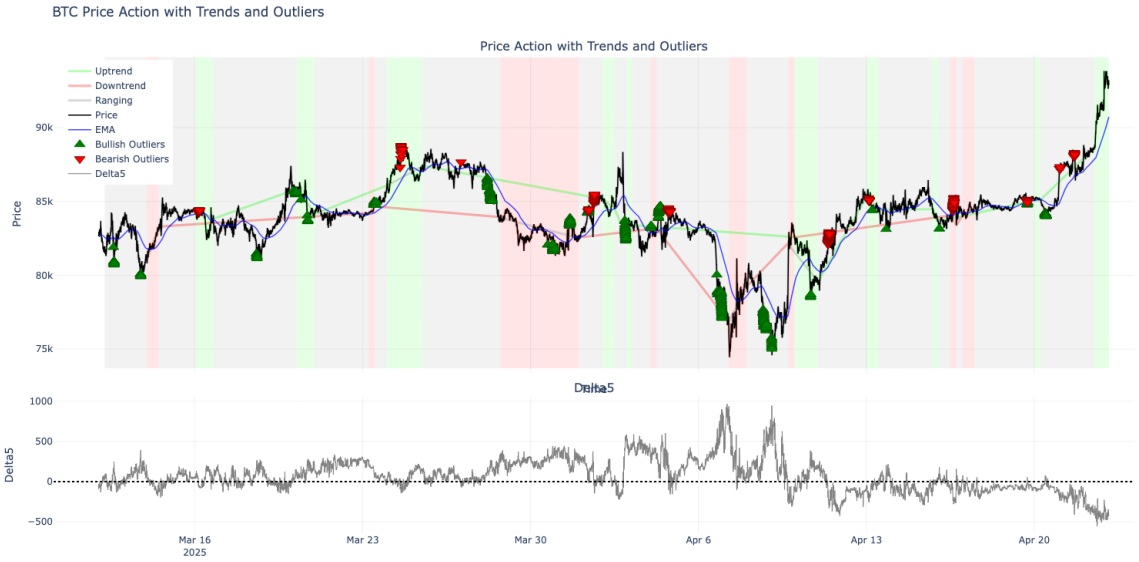

Figure 5: Visualization of Δ_t outliers and swing points

After plotting different kind of parts of Timeseries Data sets I created I was pretty sure that somewhere I would be finding an edge since the outliers often where at good entry points for a strategy. Only problem was I wasn't able to get any statistical proof ot this. After finding a Github Repo which used Vectorbt to test 1000 strategies at the same time

I tried a similar approach and tested different conditions to see if there was some kind of edge.

Monte Carlo Testing

First of I have to explain how I search for a strategy and how it nearly killed my computer. We start by defining a different kind of sample sizes of all the Δ_t outliers time periods after the outlier detection. We want to know the return after our condition meets and then check in after z minutes after that

```
sampling_percentages = [0.1, 0.2, 0.6, 0.8, 1.0]
holding_periods = [60, 120, 240, 360]
```

The sampling percentage definies how many of the existing outliers we use and test the returns on. So let's say we have x amount of Δ_t Outliers. We then randomly select z % amount of outliers and test the returns on them. We do this 1000 times and then take the average return of all the 1000 tests. The second step is to test each sampling percentage on the different holding periods. So let's say we have a sampling percentage of z % and a holding period of y minutes. We then take the z % amount of outliers and test the returns on them after y minutes. This test is basically copied from the Vectorbt Github Respository. I just changed the code to fit my needs and added some extra information returns. My code gives back the average return, the Sharpe, expectancy and mean Z-Score.

- **Sharpe Ratio**: The Sharpe ratio is a measure of the risk-adjusted return of a strategy. It is calculated by dividing the average return of the strategy by the standard deviation of the returns.
- Calculated by $\frac{\overline{r}}{\sigma}$ Where $\overline{r}$ is the average return and σ is the standard deviation of the returns.
- **Expectancy**: The expectancy is a measure of the profitability of a strategy. It is calculated by dividing the average return of the strategy by the average loss of the strategy.
- Calculated by $\frac{\overline{r}}{\overline{l}}$ Where $\overline{r}$ is the average return and $\overline{l}$ is the average loss.
- **Mean Z-Score**: The mean Z-Score is a measure of the average Z-Score of the strategy. It is calculated by taking the average of the Z-Scores of the strategy.
- Calculated by $\frac{1}{n}\sum_{i=1}^{n} z_i$ Where z_i is the Z-Score of the strategy at time i and n is the number of Z-Scores.
- **Average Return**: The average return is a measure of the average return of the strategy. It is calculated by taking the average of the returns of the strategy.

- Calculated by $\frac{1}{n}\sum_{i=1}^{n} r_i$ Where r_i is the return of the strategy at time i and n is the number of returns.

Now it is time to test some strategies. The first strategy I want to test is pretty intuitive. Measure the average return of bullish outliers inside of an uptrend.

spacing

label A outlier is bullish if the the orderbook Δ_t is two standard deviations above the mean of the last 1440 Δ_t values and bearish if it is two standard deviations below the mean of the last 1440 Δ_t values.

Now when we backtest a strategy we have to have a few things in mind. First of all we are backtesting on on historical data and if we just use different kind of entry conditions we might just change the entry conditions till we have a good end results. This would be overfitting and not work on future data. In order to prevent overfitting we have to use a holdout sample. I have one dataset on which we test our entry conditions to see if they even have potential.

Table 2: Worth a out of sample test

Sharpe Ratio	Average Return
1.5	0.005

If these conditions are met for a condition I test on some out of sample data. (Out of sample data just means that we test on new data) Most of the time things end up not even passing the first out of sample test and lose their potential instantly. If not I have another out of sample test and if that one is passed we are ready to do some more monte carlo testing and look how potential equity curves look like. And to make things even worse we can assume that the sharpen ratio will decrease by atleast 20% compared to the backtest simulation.

Developing a strategy

sources

How good or random is your trading

Testing 1000 strategies at the same time (via Monte Carlo)

A lot of my ideas come from thiy guys Tweets, to research into this