Qishi Crypto

Runmin Zhang[1], Jimo Zhang[2*], Junwei Zhang

March 11, 2019

Abstract

This is a manuscript template for Data Descriptor submissions to Qishi Club.

1 Introduction

Cryptocurrencies, a class of digital assets built on block-chain technology, have experienced a great rise during past several years. Unlike traditional financial instruments, cryptocurrencies are trading on hundreds of unregulated exchanges everyday, leading to distributed liquidities, dispersed prices, and precious arbitrage opportunities. In this project, we aim to understand liquidity of cryptocurrencies. Based on market data provided by ZYO Capital (referred as Company) and data fetched from different exchanges, We investigate multiple metrics and models to quantify different aspects of liquidity. We also explored trading strategies based on simple language model. In particular, we deliver an end-to-end pipeline which covers raw-data fetching from exchanges and online calculation and visualization of market liquidity.

2 Data Description

2.1 From Company

The datasets provided by Company include three types of tables, depth data, quote data, and trade data. The screenshots of sample data are listed below.

The depth data (Fig 9 are the orderbook snapshots per second. They include timestamp of exchanges, trading symbols, and price/size information of best 20 ask/bid levels.

	time_exchange	symbol_id	bid1_price	bid1_size	ask1_price	...	ask19_size	bid20_price	bid20_size	ask20_price	ask20_size
5210	2018-01-01 01:26:50	BINANCE_SPOT_BTC_USDT	13544.02	0.208934	13568.28	...	0.142500	13492.59	1.00000	13620.73	0.000080
24281	2018-01-01 06:44:41	BINANCE_SPOT_BTC_USDT	13719.99	0.001008	13722.00	...	0.030000	13669.99	0.03000	13756.62	0.008824
51224	2018-01-01 14:13:44	BINANCE_SPOT_BTC_USDT	12825.00	0.452617	12827.59	...	0.365767	12763.33	0.05000	12897.95	1.000000
81260	2018-01-01 22:34:20	BINANCE_SPOT_BTC_USDT	13538.90	0.088050	13541.00	...	0.086597	13506.46	0.01650	13598.00	0.003575
83664	2018-01-01 23:14:24	BINANCE_SPOT_BTC_USDT	13415.21	0.074500	13425.15	...	0.011000	13351.00	0.16165	13474.96	0.050000

Figure 1: Depth data from Company

Trade data records actual trades happened at each exchanges. Each record includes price, size, trading directions, time stamps when exchanges broadcast trading messages and time stamps when data vendor received message.

	price	size	symbol_id	taker_side	time_coinapi	time_exchange	uuid
14154	13315.19	0.179077	BINANCE_SPOT_BTC_USDT	BUY	2018-01-01T02:53:46.6720000Z	2018-01-01 02:53:46.672	9d3ce5db-8e4a-4bad-b05a-23c6554764e5
15437	13380.00	0.000514	BINANCE_SPOT_BTC_USDT	BUY	2018-01-01T03:12:58.6670000Z	2018-01-01 03:12:58.667	e1d78ec6-fee1-412f-934d-566c92fcc6d1
90562	13439.63	0.036002	BINANCE_SPOT_BTC_USDT	SELL	2018-01-01T20:14:38.5790000Z	2018-01-01 20:14:38.579	bd9960f6-a1ae-442a-aca3-a561b2ad94ed
97090	13498.03	0.079609	BINANCE_SPOT_BTC_USDT	SELL	2018-01-01T21:45:00.7190000Z	2018-01-01 21:45:00.719	8309703b-5fec-4b14-8caa-f4cef6b91457
97996	13460.00	0.326889	BINANCE_SPOT_BTC_USDT	SELL	2018-01-01T22:02:33.2450000Z	2018-01-01 22:02:33.245	b5ab840a-896b-452e-b6bf-267d06816870

Figure 2: Trade data from Company

Quote data records changes of best prices/sizes in orderbooks. Similar to trade data, quote data also include two time stamps: one corresponds to the time when exchanges broadcast messages, and the other one records when data vendor receives messages.

	ask_price	ask_size	bid_price	bid_size	symbol_id	time_coinapi	time_exchange
11344	13396.03	0.222961	13370.01	0.289000	BINANCE_SPOT_BTC_USDT	2018-01-01T02:51:07.1826733Z	2018-01-01 02:51:06.239617
20176	13606.90	0.085190	13593.27	1.651465	BINANCE_SPOT_BTC_USDT	2018-01-01T05:00:46.5148541Z	2018-01-01 05:00:45.155653
21140	13627.30	0.010000	13627.29	1.262043	BINANCE_SPOT_BTC_USDT	2018-01-01T05:13:01.8634268Z	2018-01-01 05:13:00.023665
41756	13513.60	0.049000	13496.51	0.510300	BINANCE_SPOT_BTC_USDT	2018-01-01T10:21:23.6677566Z	2018-01-01 10:21:22.739622
50230	13197.66	0.030000	13146.33	0.150000	BINANCE_SPOT_BTC_USDT	2018-01-01T12:30:08.3640146Z	2018-01-01 12:30:07.874332

Figure 3: Quote data from Company

2.2 Fetching From Exchanges(CCXT)

CCXT [7] is a JavaScript / Python / PHP cryptocurrency trading API with support for more than 130 bitcoin/altcoin exchanges, which provide an unified and friendly interface for engineers to accelerate the project process.

In this paper, we choose the Python version to implement the demo trading system.

2.2.1 Installing CCXT

The installation method of CCXT:

Python

ccxt algotrading library in PyPI

```
pip install ccxt
```

```
import ccxt
print(ccxt.exchanges) # print a list of all available exchange classes
```

The library supports concurrent asynchronous mode with asyncio and async/await in Python 3.5.3+

```
import ccxt.async_support as ccxt # link against the asynchronous version of ccxt
```

Figure 4: CCXT installation

In this paper, we provide an example to obtain the realtime orderbook from Binance [8] and Okex [9].
We utilize the uniform API to call various platforms, yet the return result format may be different among them, which means we need more effort to clean data and integrate the data format for futher research.

The demo source code of obtaining the realtime exchange data from Binance:

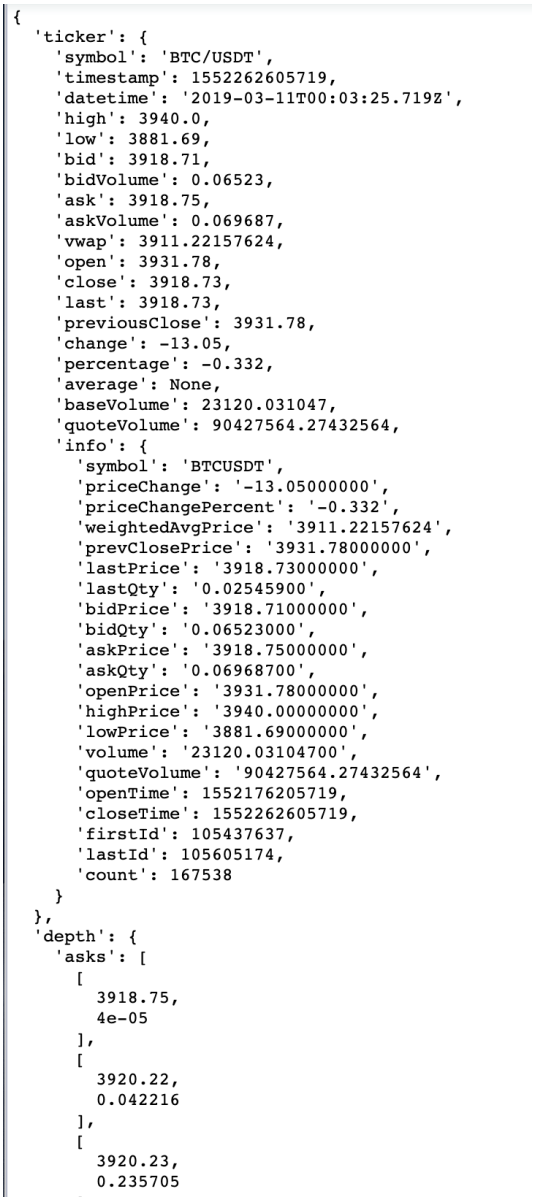
```
def getData(exchange, symbol):
    data = {}
    tickerInfo = exchange.fetch_ticker(symbol)
    depth = {}
    exchange_depth = exchange.fetch_order_book(symbol)
    asks = exchange_depth.get('asks')[:20]
    bids = exchange_depth.get('bids')[:20]
    depth['asks'] = asks
    depth['bids'] = bids

    data['ticker'] = tickerInfo
    data['depth'] = depth

    return data
```

Figure 5: Fetch realtime orderbook and ticker information

The return result data description is:



```
{
  'ticker': {
    'symbol': 'BTC/USDT',
    'timestamp': 1552262605719,
    'datetime': '2019-03-11T00:03:25.719Z',
    'high': 3940.0,
    'low': 3881.69,
    'bid': 3918.71,
    'bidVolume': 0.06523,
    'ask': 3918.75,
    'askVolume': 0.069687,
    'vwap': 3911.22157624,
    'open': 3931.78,
    'close': 3918.73,
    'last': 3918.73,
    'previousClose': 3931.78,
    'change': -13.05,
    'percentage': -0.332,
    'average': None,
    'baseVolume': 23120.031047,
    'quoteVolume': 90427564.27432564,
    'info': {
      'symbol': 'BTCUSDT',
      'priceChange': '-13.05000000',
      'priceChangePercent': '-0.332',
      'weightedAvgPrice': '3911.22157624',
      'prevClosePrice': '3931.78000000',
      'lastPrice': '3918.73000000',
      'lastQty': '0.02545900',
      'bidPrice': '3918.71000000',
      'bidQty': '0.06523000',
      'askPrice': '3918.75000000',
      'askQty': '0.06968700',
      'openPrice': '3931.78000000',
      'highPrice': '3940.00000000',
      'lowPrice': '3881.69000000',
      'volume': '23120.03104700',
      'quoteVolume': '90427564.27432564',
      'openTime': 1552176205719,
      'closeTime': 1552262605719,
      'firstId': 105437637,
      'lastId': 105605174,
      'count': 167538
    }
  },
  'depth': {
    'asks': [
      [
        3918.75,
        4e-05
      ],
      [
        3920.22,
        0.042216
      ],
      [
        3920.23,
        0.235705
```


Figure 6: Orderbook and ticker data description

CCXT also supports fetching the history information, for example ohlc(Open-high-low-close chart) data. the API is

```python
def retry_fetch_ohlcv(exchange, max_retries, symbol, timeframe, since, limit):
    num_retries = 0
    try:
        num_retries += 1
        ohlcv = exchange.fetch_ohlcv(symbol, timeframe, since, limit)
        return ohlcv
    except Exception:
        if num_retries > max_retries:
            raise  # Exception('Failed to fetch', timeframe, symbol, 'OHLCV in', max_retries, 'attempts')
```

Figure 7: OHLC API

2.2.2 Data Store

After the fetch data, we utilize the database MySQL and MongoDB [10] to store the realtime data for further research, for example data cleaning, feature engineering or machine learning model.

For example, the table schema for ohlc data:

```sql
CREATE TABLE IF NOT EXISTS binanceohlcv
(
   id            INTEGER auto_increment,
   timestamp     VARCHAR(40) NOT NULL,
   open_price    FLOAT NOT NULL,
   highest_price FLOAT NOT NULL,
   lowest_price  FLOAT NOT NULL,
   closing_price FLOAT NOT NULL,
   volume        FLOAT,
   PRIMARY KEY (id)
);
```

Figure 8: MySql ohlcv table schema

2.3 Data Visualization

Considering the data visualization, we choose plotly [11] Plotly, also known by its URL, Plot.ly, is a technical computing company headquartered in Montreal, Quebec, that develops online data analytics and visualization tools. Plotly provides online graphing, analytics, and statistics tools for individuals and collaboration, as well as scientific graphing libraries for Python, R, MATLAB, Perl, Julia, Arduino, and REST.

The benefits come from keeping the development language consistency and reducing the workload to future maintenance. The programmers can focus on the business logic and implementing them with Python, yet the CSS, HTML and javascript issues are transparent for them.

Figure 9: The dashboard of local trading system

3 Liquidity Metrics

3.1 Common Metrics

Since liquidity itself is not a well-defined quantity, we have to use multiple metrics to describe different aspects of liquidity. In this section, we introduce several metrics that are commonly used in stock markets. Since the intrinsic trading mechanism of cryptocurrencies is similar to equities, we would like to generalize metrics to equity markets into cryptocurreny markets. We can divide liquidity metrics into two categories, i.e., univariate metrics and multivariate metrics in terms of numbers of input variables. In our project, we focus on market impact. Market impacts is a good metric to characterize market slippages due to large orders. Since liquity of cryptocurrencies are limited, any large orders may cause significant market slippages, which bring in additional trading costs. Therefore, we believe that understanding and modeling market impact are a crucial step in the development of trading strategies. In the appendix, we also present other metrics commonly used in markets.

3.2 Modeling Market Impacts

3.2.1 Main models

The main model we used are linear models inspired by two commercial available models – Bloomberg market impact model and JP Morgan market impact model.

- Bloombeg market impact model:

$$MI = \frac{1}{2}\frac{S}{P} + \sqrt{\frac{\sigma^2/3}{250}}\sqrt{\frac{V}{0.3EDV}}$$

  where S is the spread, P is the mid price, σ is the volatiliy, V is the trading volume, and EDV is the expected daily volume. In this equation, the first term can be regarded as spread costs, while the second term can be regarded as the costs from permanent price impacts.

- JP Morgan market impact model

$$MI = \frac{5}{100}I + 1.4\frac{95}{100}\frac{V}{EPV}I$$

where $I = 0.187\sqrt{\frac{V}{EDV}}\sigma^2$, V is the trading volume, and σ is the volatility.

By investigating the two commercial models, we believe that coefficients in front of each term are strongly correlated to assets and markets. To generalize these two models to cryptocurrency markets, we fit the market impacts with linear regressions based on the following equations

$$MI = f\left(\frac{S}{P}, \sqrt{\sigma^2/3}, \sqrt{\frac{V}{EDV}}\right)$$

and

$$MI = g\left(I, \frac{V}{EDV}\right)$$

Here, f and g are all linear functions. The training targets are market impacts calculated from order book:

$$MI(V) = \frac{p_A(V) - p_B(V)}{p_M(V)}$$

which is the enlarged relative spread after a certain volume V is fulfilled. In our training process, we calcualte market impacts with different trading volume V, and we use corresponding V as features in functions f and g during the training process.

3.2.2 Challenger Models

Apart from linear models, we also tried other possible models with different features and different training targets, such as random forests or gradient boosting trees. We tried to apply these models on current and historical depth data to predict future market impacts. However, due to the limits of objective functions used in our model(MSE), we failed to capture the temporal results from these two model. More work is needed to extract better features and more appropriate loss function.

4 Trading

We introduced a bitcoin trading strategy inspired from statistical language modeling. When linguists try to identify a paragraph, i.e., choose a set of languages with maximum likelihood, they first partition the sentences into words, then evaluate the likelihoods of different candidate sentences based on the historical language database, and eventually pick the candidate with the largest likelihood. When trading bitcoins, similar philosophy could be applied. First, we symbolize the historical price data, and convert it into a series of price directions. Then, based on the historical price directions of the past time interval, we could evaluate the conditional probabilities of the two potential price directions (i.e., up or down) in the future respectively. Finally, the direction with higher conditional probability will be chosen as we prediction for the future.

In our bitcoin trading strategy, we take the mid prices of historical bid-one prices and ask-one prices at second level, and denote the series as p_1, p_1, p_2, ..., p_n. Then we covert the historical mid prices into a series of price direction symbols. Denote the price direction symbols as s_1, s_2, s_3, ..., s_n, where $s_i = 1$ if $p_i > p_(i-1)$, else s_i= -1.

Our objective is to predict the price direction (up or down) of the next second. To do this, we need to calculate the conditional probability of either price direction based on the price direction series of the past time interval, and make our prediction by picking the one with larger probability.

The two conditional probabilities could be evaluated as below:

$$\begin{aligned} p(up) &= p(s_i = 1 | s_{i-n+1}, s_{i-n+2}, ..., s_{i-1}) \\ &= \frac{p(s_{i-n+1}, s_{i-n+2}, ..., s_{i-1}, 1)}{p(s_{i-n+1}, s_{i-n+2}, ..., s_{i-1})} \end{aligned}$$

$$\begin{aligned} p(down) &= p(s_i = -1 | s_{i-n+1}, s_{i-n+2}, ..., s_{i-1}) \\ &= \frac{p(s_{i-n+1}, s_{i-n+2}, ..., s_{i-1}, -1)}{p(s_{i-n+1}, s_{i-n+2}, ..., s_{i-1})} \end{aligned}$$

The probability of given historical price direction series above are evaluated via their frequencies within our whole historical price direction series.

$$p(s_1, s_2, ..., s_n) = \frac{\sharp(s_1, s_2, ..., s_n)}{\sum \sharp(s_1, s_2, ..., s_n)}$$

After the above evaluations, we could predict price at the next second will go up if p(up) > p(down), it will go down if p(up) < p(down), and make no decision if they are the same.

Based on the above calculation, we developed a trading strategy as below.

- When we are holding no position
  - If higher price is predicted for the next second, we take long position.
  - If lower price is predicted for the next second, we take short position.
- When we are holding long positions
  - If higher price is predicted for the next second, we take no action.
  - If lower price is predicted for the next second, we close all long positions, and enter short position.
- When we are holding short positions
  - If higher price is predicted for the next second, we close all short positions, enter long position.
  - If lower price is predicted for the next second, we take no action.

In all the above three scenarios, if the calculated probability of higher price equals to that of lower price, we take no action.

Please note that there is one parameter which will heavily affect the performance of our trading strategy: length of rolling window when evaluating frequencies of the historical price direction series. If it is too short, then the price direction series don't contain enough historical price information, our strategy tend to under-fit our market data; if it is too long, then our strategy tends to over-fit our market data, meanwhile the chance that during prediction a given series not being stored in our "price direction database" will be larger. After various tests, we eventually pick the rolling window of 6 as a trade-off between the above two scenarios.

Besides, we take small trading sizes and make sure they are within the volume of bid-one sizes (when taking long positions) or the volume of ask-one sizes (when taking short positions). We assume the trading infrastructure could allow us to take our target position in the high frequency trading market.

We back tested our strategy via the January 1st 2018 Binance data. By using the first 80% of the data as training data and the remaining as test data, as well as setting rolling window length as 6, we eventually achieved positive profit.

5 Implementation and Results

5.1 System Structure

5.2 Results

Since the data flow directly fetched from exchanges are not stable, we present our results based on depth data on 2018.10.01 from Company. In Figure **??** and 11, we present the comparison between actual markets impacts and theoretical market impacts predicted by the two models respectively. As we can see, both models are almost unbiased predictors of market impacts, and they can be used as theoretical benchmark or trading signals of market impact.

5.2.1 Market Impacts

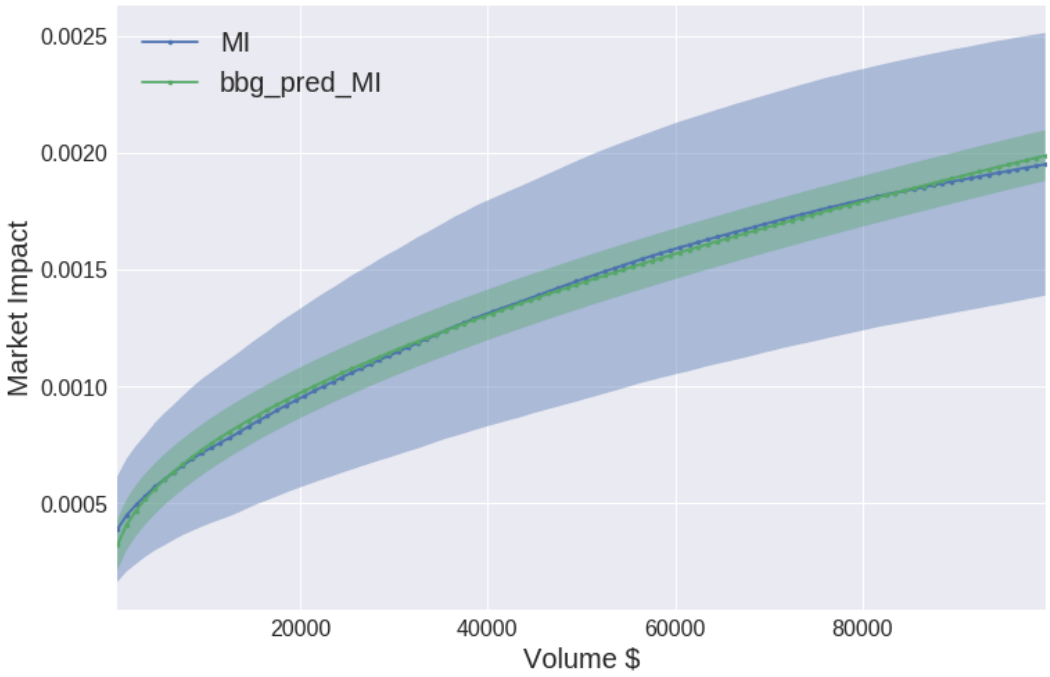


Figure 10: Comparison between actual market impacts and market impacts predicted by Bloomberg-based model. The blue dotted line is the average actual market impacts with different trading volumes, and the green dotted line is the average theoretical market impacts predicted by Bloomberg based model. The blue band and the green band denote data ranges $[\mu_{MI} - \sigma, \mu_{MI} + \sigma]$.

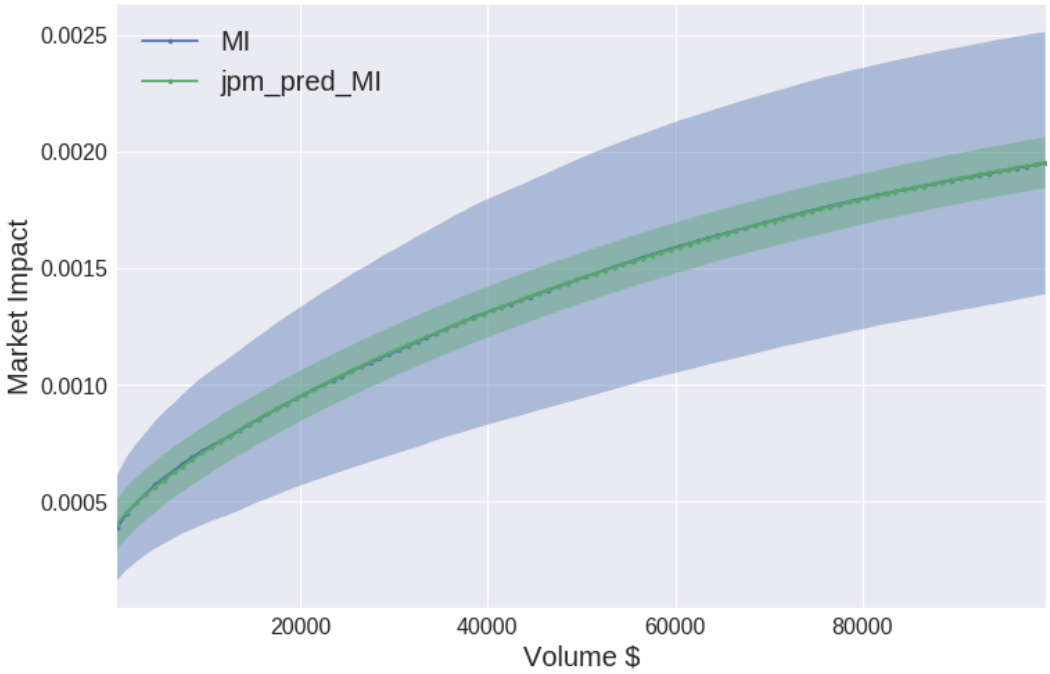


Figure 11: Comparison between actual market impacts and market impoacts predicted by JP Morgan-based model. The blue dotted line is the average actual market impacts with different trading volumes, and the green dotted line is the average theoretical market impacts predicted by Bloomberg based model. The blue band and the green band denote data ranges $[\mu_{MI} - \sigma, \mu_{MI} + \sigma]$.

5.2.2 Dashboard

We also developed an interactive dash board based on Python. In Figure 12, the screenshot of our dash board. The dash board supports:

- Display current order books. The orderbook information is fetched from exchanges and displayed. Multiple diagrams, such as line diagrams, candle-sticks diagram can be plotted with different frequencies.
- Present headlines. Top headline news related to bitcoin are fetched from News API, and are displayed on the bottom-left part of the dash board.
- Paper-trading tracker. A paper-trading simulator is also included, which tracks current open positions and PnL.

Figure 12: Screenshot of the dashboard.

6 Conclusion and Future Work

We developed an end-to-end pipeline to evaluate liqudity of cryptocurrencies. Started from an open source library `ccxt`, we built a data fetcher which obtains level-2 data of current order books from exchanges. After we obtained updated data, two models were created to generate theoretical market impacts of different cryptocurrencies. As an unbiased estimator, the theoretical market impact is a great metric to characterize market liquidities, and it will be useful in developments of any liquidity-based strategies. We also explored the possibility to trade cryptocurrencies with simple language models. A strategy was proposed, and preliminary backtests showed great potential. Finally, a interactive dashed is delivered, which integrate a visualizer of level-2 data from order books, a panel of top head lines of bitcoin news, and a paper-trading simulator.

Due to time constraints, our work so far is still preliminary, and future work are required to optimize the whole system. First, we would like to explore much more models and features apart from current linear models, and we would like to transform the current models as “liquidity calibrators" to “liquidity predictors". Besides, rigorous statistical tests should be applied validate our models. Also, optimization should be carried out on our platform. For example, we would like to build a memory-based database to store order book to substitute current disk-based solutions. Last but not least, more tests should be done to test the statistical significance and profitability of our trading strategy.

7 Appendix: Common Metrics

7.1 Univariate Metrics

Volume-based Metrics

- Trading volume:

$$Q_t = \sum_{t=1}^{N_t} q_i$$

Q_t	trading volume from time t-1 to time t
N_t	number of trades between t-1 and t
q_i	number of shares in trade i

- Turnover:

$$V_t = \sum_{i=1}^{N_t} p_i * q_i$$

V_t	turn over in terms of dollar
p_i	trading volume from time t-1 to time t
N_t	number of trades between t-1 and t
q_i	number of shares in trade i

- Depth:

$$D_t = q_t^A + q_t^B$$

OR

$$D_t = (q_t^A + q_t^B)/2$$

D_t	market depth at time t
q_t^A, q_t^B	volumes on the best ask/bid levels in the order book

- Log Depth:

$$Dlog_t = ln(q_t^A) + ln(q_t^B) = ln(q_t^A * q_t^B)$$

q_t^A, q_t^B	Best ask/bid sizes

- Dollar Depth

$$D\$_t = (q_t^A * p_t^A + q_t^B * p_t^B)/2$$

$D\$_t$	dollar dpeth
p_t^A, p_t^B	Best ask/bid prices
q_t^A, q_t^B	Best ask/bid sizes

Time-related Liquidity Metrics

- Trade Frequency NT_t: the total number of trades between time $t-1$ and t.
- Order Frequency NO_t: the total number of new orders(quotes) inserted into the order book between time time $t-1$ and t

Spread-Related Liquidity Measures

- Absolute Spread:

$$Sabs_t = p_t^A - p_t^B$$

We can also use log scale:

$$LogSabs_t = log(p_t^A - p_t^B)$$

- Relative Spread:

$$Srel_t = (p_t^A - p_t^B)/p_t^M$$

or

$$Srel_t = (p_t^A - p_t^B)/p_t^T$$

or relative spread of log prices

$$Srellog_t = ln(p_t^A) - ln(p_t^B)$$

p_t^A, p_t^B	Best ask/bid prices
p_t^M	Mid price at time t
p_t^T	Last price at time t

- Effective Spread

$$Seff_t = |p_t - p_t^M|$$

p_t	Last trade price at time t
p_t^M	Mid price at time t

- Relative Effective Spread

$$Seff_t = |p_t - p_t^M|/p_t$$

OR

$$Seff_t = |p_t - p_t^M|/p_t^M$$

p_t	Last trade price at time t
p_t^M	Mid price at time t

7.2 Multi-Dimensional Liquidity Metrics

Spread-related metrics in the numerator; volume-related metrics in the denominator

- Quote Slope

$$QS_t = Sabs_t/Dlog_t$$

$Sabs_t$	Absolute spread t
$Dlog_t$	Log septh$t$

- Log Quote Slope

$$LogQS_t = Srellog_t / Dlog_t$$

$Srellog_t$	Relative spread of log prices t
$Dlog_t$	Log septh$t$

- Adjusted Log Quote Slope

$$LogQSadj_t = LogQS_t * (1 + |ln(q_t^B / q_t^A)|)$$

- Composite Liquidity

$$CL_t = Srel_t / D\$_t$$

$Srel_t$	Relative spreads at time t
$D\$_t$	Dollar depth at time t

Market Ratios

- Liquidity Ratios:

$$LR1_t = V_t / |r_t|$$

$$LR2_t = LR_t / NS$$

$$LR3_t = \sum_{i=1}^{N} |dp_i^T| / N_t$$

V_t	$V_t = \sum_{i=1}^{N} p_i * q_i$, dollar volume from time $t-1$ to t
$r\$_t$	returns from time $t-1$ to time t
NS	nets shares of stocks excluding shares owned by the firm
dp_i^T	changes of trade prices between time $t-1$ and t

- Flow Ratios

$$FR_t = V_t / WT_t$$

V_t	$V_t = \sum_{i=1}^{N} p_i * q_i$, dollar volume from time $t-1$ to t
WT_t	average waiting time between trades

- Order Ratio

$$OR_t = |q_t^B - q_t^A| / V_t$$

V_t	$V_t = \sum_{i=1}^{N} p_i * q_i$, dollar volume from time $t-1$ to t
q_t^A, q_t^B	sizes of best ask/bid quotes

References

[1] Califano, A., Butte, A. J., Friend, S., Ideker, T. & Schadt, E. Leveraging models of cell regulation and GWAS data in integrative network-based association studies. *Nat. Genet.* **44**, 841–847 (2012).

[2] Wang, R. *et al.* PRIDE Inspector: a tool to visualize and validate MS proteomics data. *Nat. Biotechnol* **30**, 135–137 (2012).

[3] Zhang, Q-L., Chen, J-Y., Lin, L-B., Wang, F., Guo, J., Deng, X-Y. Characterization of ladybird Henosepilachna vigintioctopunctata transcriptomes across various life stages. *figshare* https://doi.org/10.6084/m9.figshare.c.4064768.v3 (2018).

[4] *NCBI Sequence Read Archive* http://identifiers.org/ncbi/insdc.sra:SRP121625 (2017).

[5] Barbosa, P., Usie, A. and Ramos, A. M. Quercus suber isolate HL8, whole genome shotgun sequencing project. *GenBank* http://identifiers.org/ncbi/insdc:PKMF00000000 (2018).

[6] *DNA Data Bank of Japan* http://trace.ddbj.nig.ac.jp/DRASearch/submission?acc=DRA004814 (2016).

[7] *CCXT* https://github.com/ccxt.

[8] *BINANCE* https://www.binance.com/en.

[9] *OKEX* https://www.okex.com/lang/en-US/

[10] *MongoDB* https://www.mongodb.com/

[11] *plotly* https://github.com/plotly/dash-web-trader