TU-EV005 Financial Engineering III: Machine Learning in Finance
Ruth Kaila
Spring 2022

Assignment for the 5 credit version and the 6 credit version of the course

- You can work alone or in groups of 2 to 3 persons
- For any questions, contact Ruth

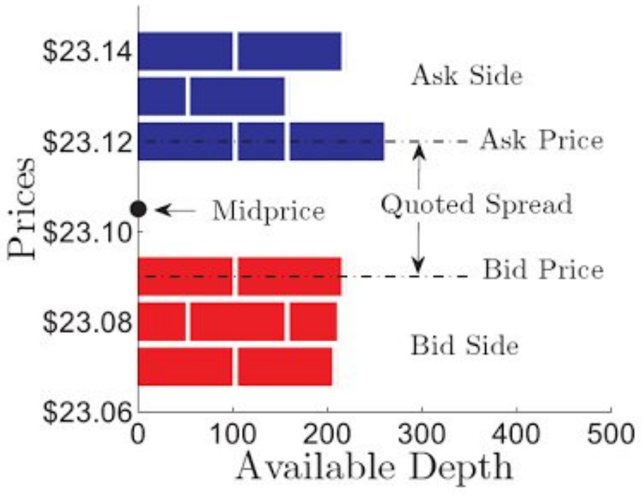
Data:
The data you will be working with is cryptocurrency derivatives tick data. More specifically, BitMEX *perpetual swap* tick data (top-of-book & trades). BitMEX has made this data publicly available (url: https://public.bitmex.com/) in .csv format. This link works at least with Chrome. It does not work with Safari.

In this study, our focus will be in three different large-tick swap contracts:

- **XBTUSD**
- **ETHUSD**
- **BCHUSD**

These contracts have relatively large minimum price increments and, therefore, exhibit interesting price dynamics.

The listed website contains two different data sets, namely quotes and trades. The quote data contains best bid- and ask-prices. These prices represent the most competitive prices at which one would be able to buy (ask) or sell (bid) these derivatives contracts at each given point in time ("top-of-book"). The trade data contains the actual executed buy and sell transactions ("trades").



Research problems:
Using the BitMEX data and machine learning methods of your choice, try to find (i) *the best forecasting model* and (ii) *set of predictive features* for all, or some, of the following quantities:

- Change in the best bid-price $\Delta P^{b}_{t,t+dt}$,
- Change in the best ask-price $\Delta P^{a}_{t,t+dt}$,
- Probability $\Pr(\Delta P^{b}_{t,t+dt} < 0)$ that the best bid-price ticks down,
- Probability $\Pr(\Delta P^{a}_{t,t+dt} > 0)$ that the best ask-price ticks up,
- Size of the best-bid queue $Q^{b}_{t,t+dt}$,
- Size of the best-ask queue $Q^{a}_{t,t+dt}$,
- Number of executed sell trades $N^{sell}_{t,t+dt}$, and
- Number of executed buy trades $N^{buy}_{t,t+dt}$,

- **Challenge**: number of executed sell trades $\widehat{N}^{sell}_{t,\hat{t}}$ *before* the bid-price changes the next time, and
- **Challenge**: number of executed buy trades $\widehat{N}^{buy}_{t,\hat{t}}$ *before* the ask-price changes the next time.

Forecasting horizon: In this study, we are interested in forecasting the previously listed quantities for the next, say, 100 – 5000 milliseconds. That is, given a vector of statistical features $\boldsymbol{\theta}_t$ that I can observe "now" at time t, I am interested in building a forecasting model $f(\boldsymbol{\theta}_t)$, for example, for the number of buy trades that will "lift" the sell limit orders resting at the ask price during the next 500 milliseconds.

Change: In this study, a change in price is defined as the signed (positive or negative) number of ticks:

$$\Delta P^{b}_{t,t+dt} = (P^{b}_{t+dt} - P^{b}_{t})/TickSize$$

The *tick size* represents the amount the price may change. It is worth noting that it might be different for each contract. You should be able to find the tick size information from the contract specifications listed at the exchange website.

Some (old) ideas for feature engineering and forecasting:

- Avoid naïve standardization of predictors, i.e. avoid making strong assumptions about non-stationarity.
- Do not use price *levels* as predictors: prices might be dramatically different tomorrow and hence your model will be useless!

- Try to use *derived variables* which are *computed from* the data.

  Example: an exponentially smoothed buy-pressure is obtained by dividing the size of the best bid-price queue by the size of best ask-queue:

$$BuyPressure_t = EWMA\left(\frac{Q_t^b}{Q_t^b+Q_t^a}, \alpha = 0.95\right)$$

The buy-pressure may be used to compute a sort of weighted-mid-price:

$$WeightedMid_t = BuyPressure_t \times P_t^a + (1 - BuyPressure_t) \times P_t^b$$

The weighted-mid-price may then be used to compute micro-price-difference, i.e. the difference between weighted mid-price and naïve mid-price:

$$MicroPriceAdjustment_t = WMid_t - \frac{1}{2}(P_t^a + P_t^b)$$

One could then use the microprice adjustment to forecast future price changes:

$$\Delta P_{t,t+dt}^b = \gamma_1 \times MicroPriceAdjustment_t + \cdots + \epsilon_t$$

- Try to use features across different contracts. For example, it might be that features computed from ETHUSD help to predict the probability of down tick in the best bid-price in XBTUDS etc.

About ML techniques:

- Try to minimize the complexity of your modeling mathematics. Linear approximation is usually much faster to compute than non-linear modeling (linear regression vs. deep neural networks).
- Try to minimize the number of predictors in your model.
- If you are doing classification, check your training set for class imbalance. If you have imbalanced data, try to use frequency-based weighting and try to avoid random sampling techniques.
- Be creative!

About programming:

- Remember unit-testing, even when working with data. Especially if you write OOP.
- Fit your model to a test data set to make sure your code / library works as expected before fitting to noisy real-world data.
- If you are working with Python, a library called *vaex* might turn out to be helpful if you are running out of memory.
- If you are doing brute force stuff in Python, try to use *numba*.

Write a 5-7 page report with the following content:

1. Introduction on the theoretical background of the research problem (HFT; limit-order-book)
2. Research problem
3. Methods used with justifications of the choices
4. Data and research setting
5. Results with analysis
6. Conclusions

(line spacing 1.5; you can write in English or Finnish; you can write a shorter report if you work alone and a longer if you are working in a group.)