Crypto Trading System

Group 14:

Ken Chen, Nicole Zhao, Yuqin Wang, Linxia Li, Xinyu Guo

Crypto Market

There are 10+ liquid crypto exchanges (Binance, Coinbase, FTX, Huobi, OKX…) and hundreds of trading instruments . Cryptocurrency arbitrage strategies are based on low latency, which uses algorithmic trading to react to market events faster than the competition and increase profitability of trades.

Project Vision

We want to introduce a **real-time, low-latency** python-based trading system for **cryptocurrency arbitrage strategies**, including efficiently download historical data, record live data, train predictive model, and run model in real-time.

NYU

Project Achievements

- Build a crypto trading system that could be used to trade **arbitrage trading strategies** using Python.
- Connect to **OKX** crypto exchange based on REST/WebSocket API.
- Build functions such as data recording, signal/position calculation.
- Apply **predictive models** to trading strategies.
- Control the internal latency to the level of **7ms** (from receiving data to sending orders).
- Extra functions: strategy monitor UI, database management, risk management, order management

NYU

Data Collection

- **Minute bar data**
  - BTC/BCH/ETH data 1/1/2020 - 4/22/2022, total 1,212,300 lines.
  - Multi-threading with 5 threads improves the most with 1,568s.

time	ticker	open	high	low	close	volume
1/1/2020 00:00	BTC-USDT	7220	7220	7219.5	7219.5	2.7594
1/1/2020 00:01	BTC-USDT	7219.6	7222.9	7219.5	7222.9	2.841
1/1/2020 00:02	BTC-USDT	7223	7224.8	7222.9	7224.8	3.68631
1/1/2020 00:03	BTC-USDT	7224.8	7226.1	7224.8	7226	1.74307
1/1/2020 00:04	BTC-USDT	7226.1	7226.4	7225.7	7225.7	4.91684
1/1/2020 00:05	BTC-USDT	7225.8	7225.8	7225.3	7225.3	8.8941

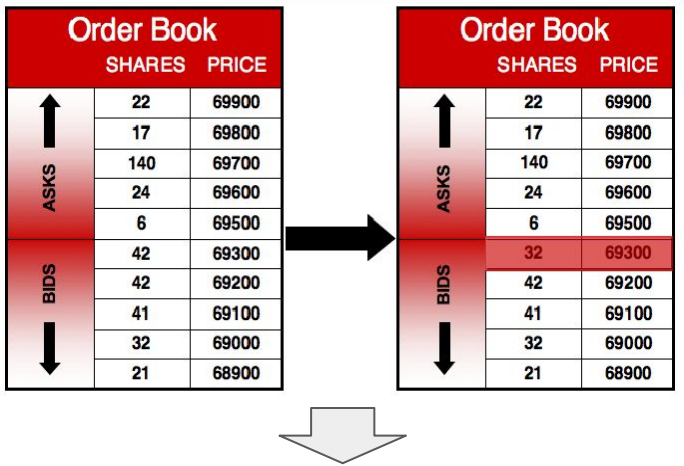
- **Tick data**
  - BTC/BCH/ETH's spot, futures and swap contract data 4/19/2022 - 4/22/2022, total 2,791,623 lines.
  - An recorder function to register market data.
  - Single thread implementation + SQLite database is used to store the live tick data.

Method	# of Threads/Processes	Time Cost (s)
Theoretical optima		**1212**
No speed up		3846
Multi-threading	3	2628
Multi-threading	5	**1568**
Multi-processing	3	2811
Multi-processing	5	1705

NYU

Features Selection

- **Order book:** buy and sell orders for a specific instrument organized by price level at a snapshot



$$Price\ Ratio = \frac{Mid\ Price(Spot)}{Mid\ Price(Swap)}$$

$$Order\ Imbalance\ Ratio = \frac{(bid\ volume - ask\ volume)}{(bid\ volume + ask\ volume)}$$

Notation	Name
x_1	$Relative\ spread\ spot$
x_2	$Relative\ spread\ swap$
x_3	$Order\ imbalance\ ratio\ spot$
x_4	$Order\ imbalance\ ratio\ swap$
x_5	$\Delta PriceRatio\ (\%)\ for\ the\ past\ 10S$
x_6	$\Delta PriceRatio\ (\%)\ for\ the\ past\ 20S$
x_7	$\Delta PriceRatio\ (\%)\ for\ the\ past\ 60S$
y	$(30\ Seconds) Change\ of\ future\ PriceRatio$

NYU

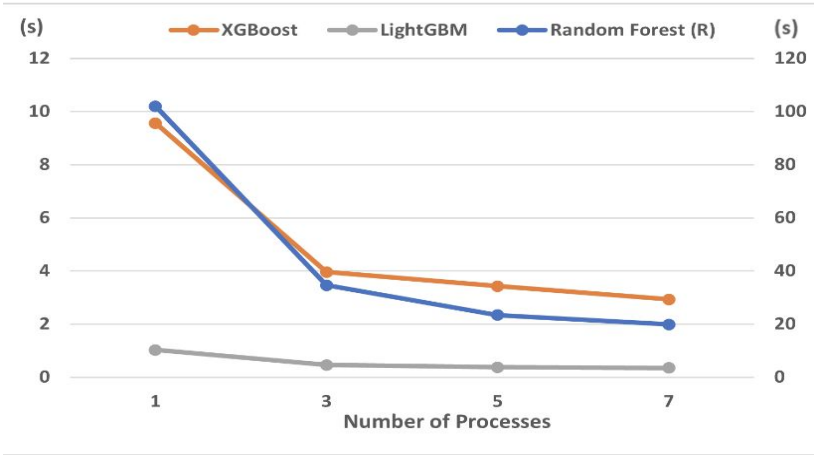
Model Training Result & Optimization

- **Evaluation**
  - R2: total price movements explained by model
  - IC: Information coefficient**,** shows how closely the analyst's financial forecasts match actual financial results; higher the better

- **Optimization Methods Implemented**
  - Multi-threading
  - Multi-processing

  From 1 core to 7 cores:
  - running speed of RF is increased by 413%
  - running speed of XGBoost is increased by 226%

Model	# of Cores	R^2 Score	IC	Time (s)
Linear Reg	1	0.188	0.489	0.049
Ridge Reg	1	0.229	0.513	0.015
Random Forest	1	0.261	0.521	101.95
Random Forest	7	0.260	0.521	19.884
XGBoost	1	0.291	0.539	9.564
XGBoost	7	0.291	0.539	2.927
LGBM	1	0.132	0.386	1.022
LGBM	7	0.132	0.386	0.350



NYU

Trading System

- **Build system**

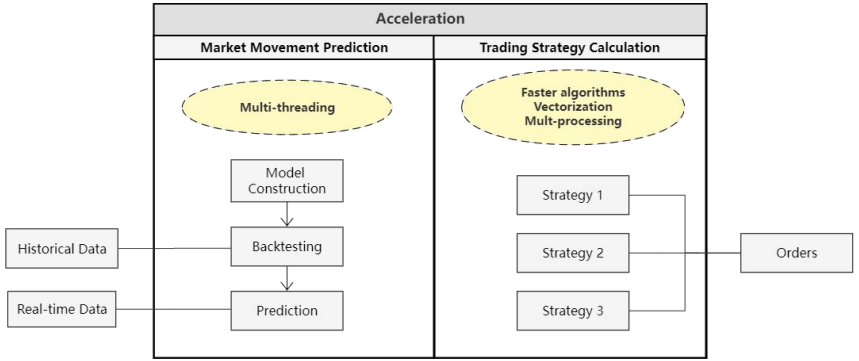
  We build our trading system based on VeighNa, a python-based open-sourced trading system used to trade Chinese futures market.

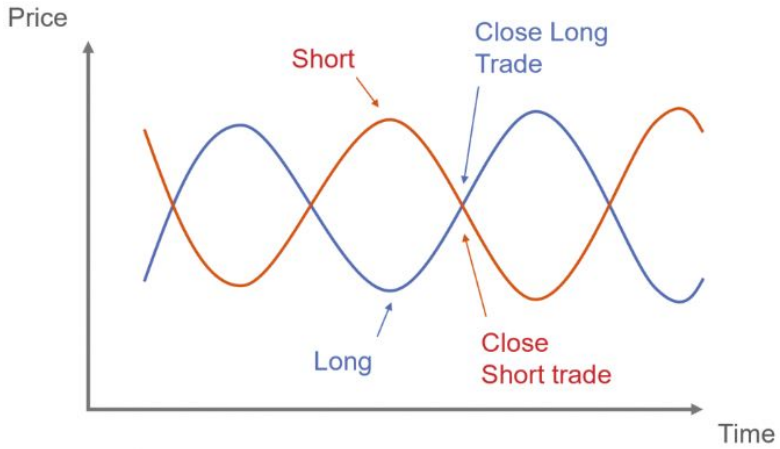
- **Add applications**

  We create our own applications including access to the OKX exchanges, data downloader, and trading strategy.

- **Load model**

  The predictive models are loaded in the trading system so we can predict the future price movements. Based on the prediction from the model (alpha), we can set a open limit parameter to decide when we will make the trades.





NYU

Trading System Latency

- **Latency Evaluation: Tick-to-Trade**

  Tick-to-Trade is the time interval between receiving a market tick data and processing the buy or sell order.

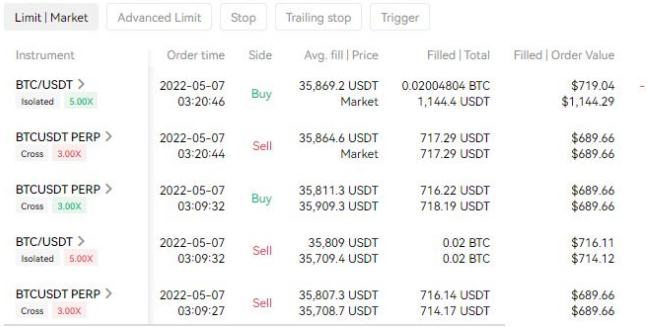
- **Implementation Plans**

  (1) pandas DataFrame

  (2) NumPy array

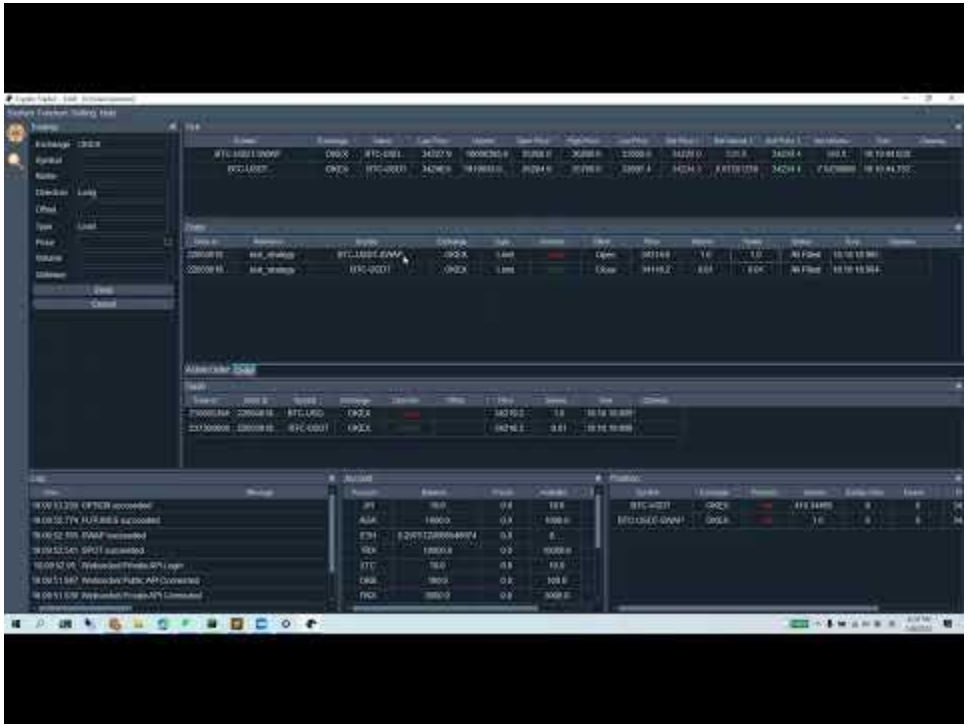
  (3) Improved algorithms and built-in data structures including deque and hash-tables.

Model	Mean (ms)	Std (ms)
Plan 1: DataFrame	15.05	3.75
Plan 2: NumpyArray	10.33	3.07
Plan 3: Built-in data structures with improved algorithms	**7.05**	**2.06**
- Update variables	0.02	0.01
- Model predict	7.19	2.04
- Strategy and Send orders	0.04	0.02



NYU

Trading System Demo



https://www.youtube.com/watch?v=MSD3RArbnnQ

NYU