REAL-TIME TRADING SYSTEM

Implementation of different trading strategies in a real-time system

Peter Eusebio, Julian Kleindiek, Markus Wehr

June 9th, 2020

Agenda

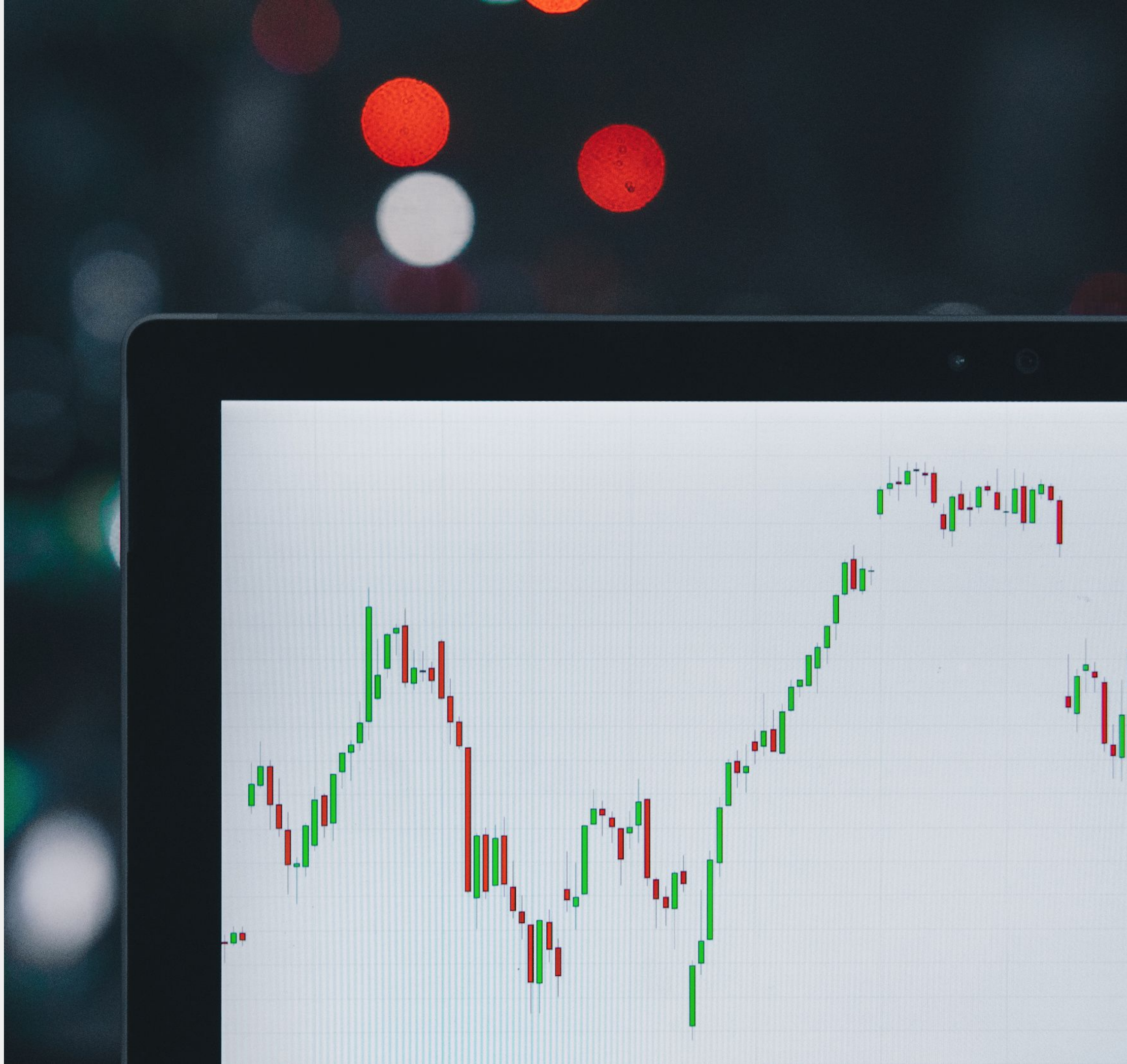
Project Introduction

System Design

Trading Strategies

Comparison of Results

Future Work

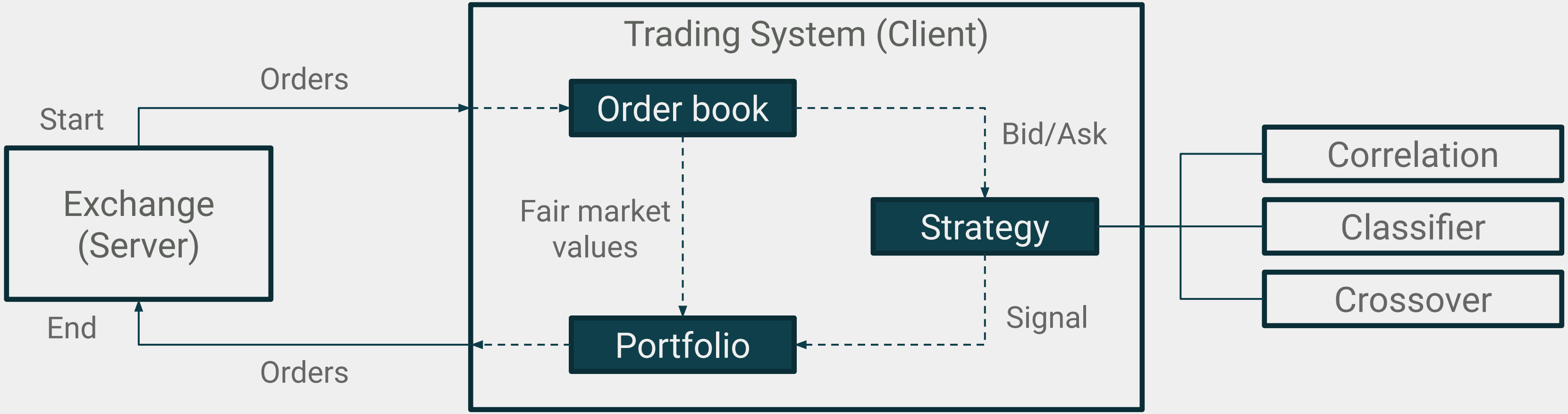


Business Problem

- Build a real-time trading system that is capable of interacting with a fictitious stock market
- Trading system receives market updates from a server and makes decisions based on the following different strategies
- The goal of each strategy is to maximize the return on the invested money

System Design

Overview



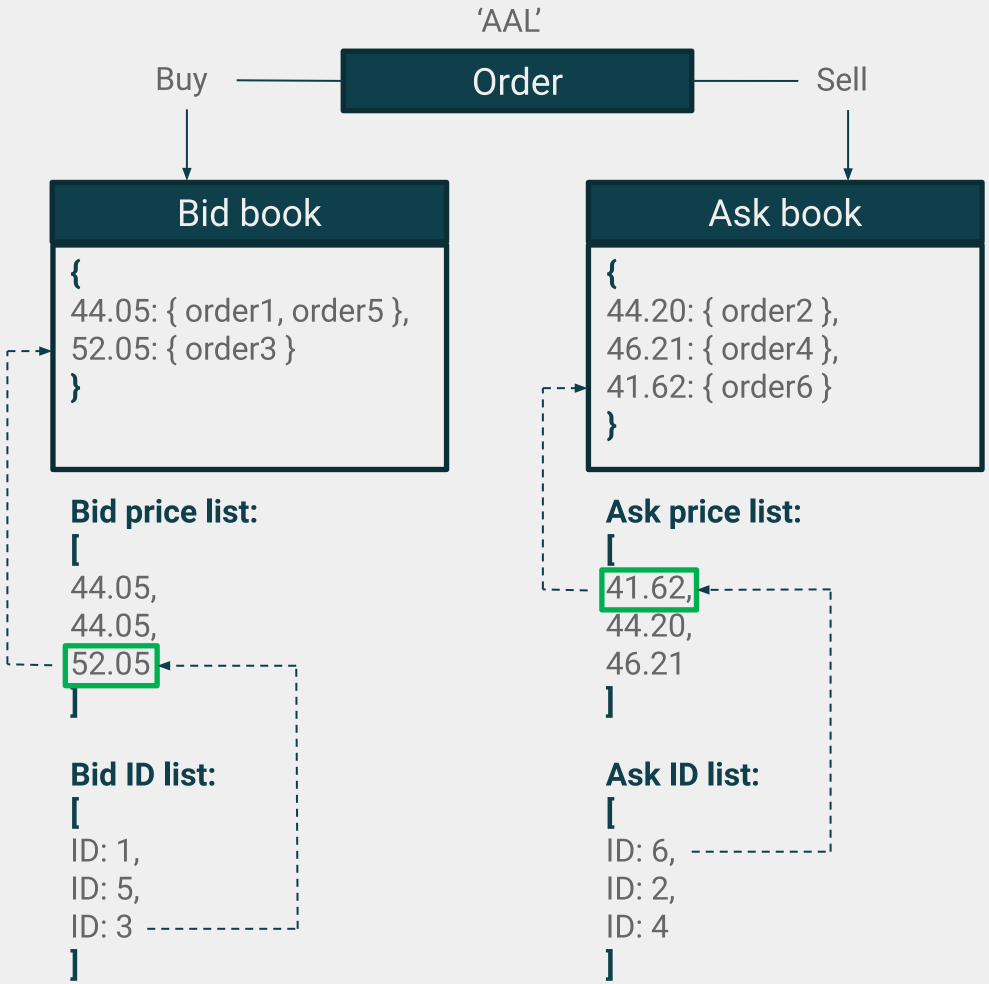
Execution time: ~ 0.601 sec from start to end

Order book: Dictionary where key equals symbol and item equals class object (bid/ask book)

Portfolio: Holdings with updated market values, quantity, cash position

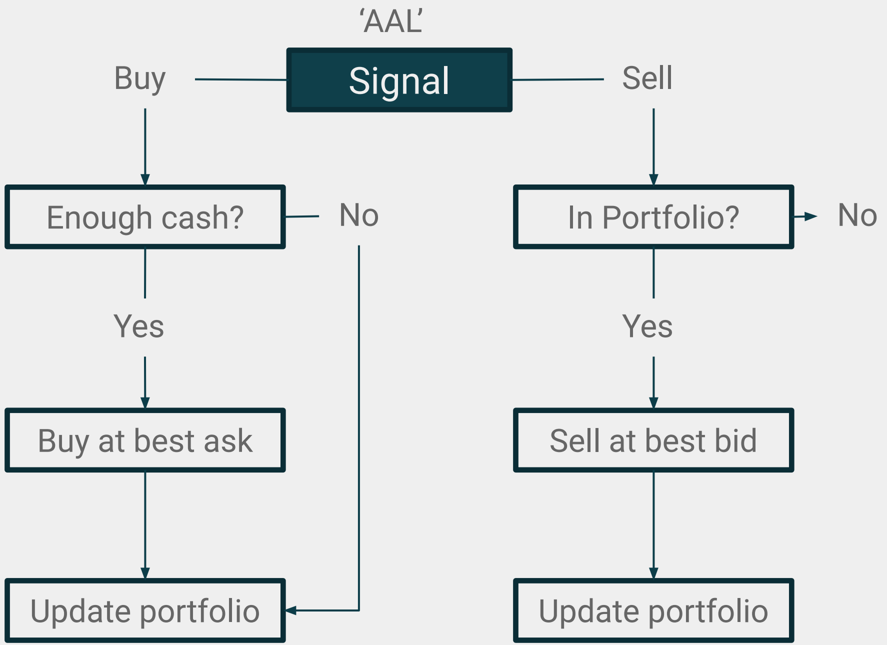
Order book

- New orders are stored in bid or ask book with price as key
- Sorted price lists are maintained as pointers to the dictionary
- ID lists are maintained as pointers to the price list
- ID lists are in same order as price lists, so books can be queried by ID also (i.e. for price modifications)



Portfolio

- Checks whether there is enough cash or stocks in the portfolio
- If yes, buys pre-determined amount of shares (Buy)
- If yes, sells all shares in portfolio (Sell)
- Updates quantity, cash position and market value of position (avg. best bid and best ask)
- If no, update market values, if in portfolio (Buy)
- If no, do nothing (Sell)



Trading Strategies

Feature Engineering

1. Create dataframe from each symbol
2. Extract features from each symbol
3. Recombine data from each symbol into single train dataset

Features:

Dummy:

- Action
- Side
- Exchange

With two windows, lengths 5 and 10, compute:

- Standard deviation
- Mean
- Difference

For:

- Quantity
- Price
- News

Classifier

1. Create dataframe for each symbol
2. Calculate response for each row
3. Combine with extracted features
4. Train classifier on first 80% of data, test on last 20%

Defining the response:

1. Calculate percent change on price for consecutive orders of the same stock, ignoring side/action
2. If change > 5%, correct prediction is “Buy”
3. If -5% < change < 5%, correct prediction is “Hold”
4. If change < -5%, correct prediction is “Sell”

Training the classifier:

1. Extract features from each symbol for every row after minimum window is met (10 rows)
2. Predict “Buy,” “Hold,” or “Sell” using those features as a multiclass problem
3. Best performance on gradient boosting classifier
   a. 92% accuracy
   b. 96% F1 score for “Hold”
   c. 77% F1 score for “Buy” and “Sell”

Crossover Strategy

- Short (5 orders) and long (10 orders) rolling mean price
    - diff = short - long
- “Buy” when diff > 5%
- “Hold” when -5% < diff <5%
- “Sell” when diff < -5%

https://www.investopedia.com/articles/active-trading/052014/how-use-moving-average-buy-stocks.asp

Rolling dataframe for classifier and crossover strategies

- Create dict with keys = symbols, values = empty dataframes
- Append new orders to dataframe corresponding to each symbol
- Hold before dataframe reaches length = 10
- Once length = 10, extract features
- Run classifier or crossover strategy on last row
- Make each trading strategy a class with a handle_market_order method and a rolling_df_dict instance attribute

Index	Symbol	Action	Price	Exchange	Side	Recommendation
1	‘AAPL’	A	44.08	2	B	“Hold”

2	‘AAPL’	A	47.45	1	B	“Hold”

Index	Symbol	Action	Price	Exchange	Side	Recommendation
1	‘AAPL’	A	44.08	2	B	“Hold”
...	...	...	...	...	...	...
10	‘AAPL”	A	73.22	3	S	“Sell”

Index	Symbol	Action	Price	Exchange	Side	Recommendation
2	‘AAPL’	A	47.45	1	B	“Hold”
...	...	...	...	...	...	...
11	‘AAPL”	A	71.92	2	S	“Sell”

Correlation strategy

Pairs included in the strategy	Correlation coefficient
(‘BMRN’, ‘GOOGL’)	0.657
('CSCO', 'ISRG')	0.475
('CTXS', 'INTC')	0.440
('CERN', 'CTRP')	0.436
('ADI', 'DISCK'),	0.427
('ALXN', 'FAST')	0.400

Initial exploration:

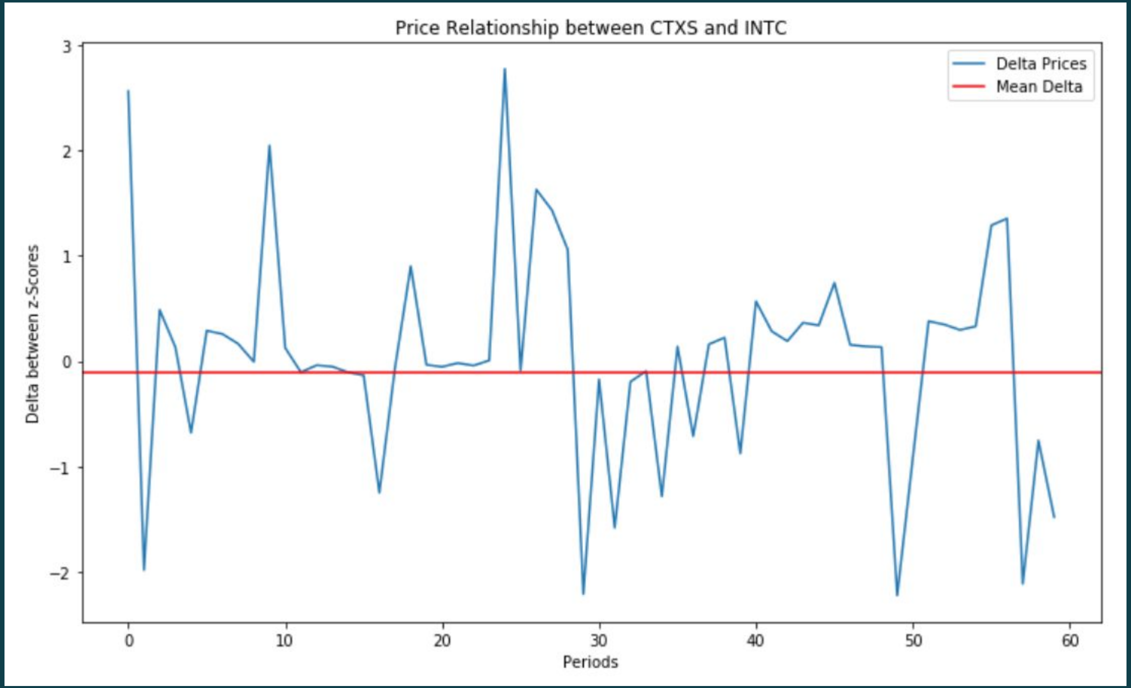
- Calculate returns for each stock for the given periods
- Determine correlation coefficients between the returns of the stocks
- Identify pairs with highest correlation

Real-time implementation:

- Use window of past 10 prices to calculate the z-score of the prices for the incoming stock and its partner
- Calculate delta between the z-score prices for the pair of the incoming stock
- Send buy/sell/hold signal based on a predefined threshold for the delta between the z-score prices

Fine-tuning of the correlation strategy

- Tested different thresholds and 1/-1 yields the best results
- Delta >1: “Buy” INTC and “Sell” CTXS as CTXS increases while INTC doesn't or INTC decreases while CTXS doesn't
- Delta <-1: Decreasing delta: “Sell” INTC and “Buy” CTXS as CTXS decreases while INTC doesn't or INTC increases while CTXS doesn't
- 1 > delta > -1: “Hold”



Comparison of Results

Our strategies come with different pros and cons

Strategy	Return	Max frequency	Insights
Cross-Over	**42.2%** initial cash: $100k trx amount: 15	**10 order/sec**	• High frequency, low volume strategy • Data storage: pandas dataframes 10 rows
Classification	**262.6%** initial cash: $100k trx amount: 10	**3.3 order/sec**	• High frequency, low volume strategy • Data storage: 10 rows of 9 columns in pandas dataframe • Feature creation and prediction slows down execution time
Correlation	**75.1%** initial cash: $100k trx amount: 200	**100 order/sec**	• Low frequency, high volume strategy • Data storage: 10 prices in a dictionary of deques by symbol; no other data required

Future Work

Moving forward we want to make a few improvements

- **Closed loop:** Create a feedback loop of the transactions made by our system back into our order book
- **Multiple exchanges:** Make the trading system more fine grained by acknowledging different exchanges and having separate order books for each
- **Transaction fees:** Consider transaction fees when calculating the return on investment of a strategy
- **Threading:** Implement threading to be able to process multiple incoming orders
- **Cloud computing:** Improve runtime using cloud technology