Data Science Case Specification

1 Overview

You will trade four symbols, $Stock1, Stock2, Stock3, Stock4$, which are ETFs on related baskets of stocks. As a new trading firm with a slow connection, by default you will receive price data on these securities one timestamp delayed. Another trading firm has decided to auction off rights to their connection to the exchange which gives them access to up-to-date price data. You will need to determine how much a faster connection is worth to you, and how to execute on the price data available to you in order to maximize trading expectancy.

Coming up with a successful strategy for this case might involve:

1. Performing statistical analysis to deduce the relationships between the symbol returns.
2. Determining the most profitable way to trade on that information.
3. Examining the additional value of having the symbol price data one time step in advance and using that to participate in the per-round auction.

2 Case Information

2.1 Summary

The case consists of 4 rounds, each of which consists of 1000 timesteps. At each timestep of every round, you (or rather, your Python bot) will have access to the price data for the preceding timestep. If you won the auction for a particular round, you will additionally have access to the price data for the *current* timestep. From this data, you will submit the quantities of each symbol you wish to hold for that timestep, subject to position limits.

At the end of each round, you'll receive a .csv file that contains the price history of the symbols for that round.

Additionally, for each round, there will be an auction where participants can submit a sealed bid to the firm with the fast connection. The top 12 bids will be considered winners of the auction and will pay out at the price of the 13th highest bid. In the following round, each auction winner receives up-to-date price data for the symbols they won the auction for.

2.2 Transaction costs

There is a transaction cost associated with trading each symbol. The transaction cost will be levied as a fraction of the volume traded (in dollars).

Symbol	Transaction Cost
Stock 1	0.0005
Stock 2	0.0010
Stock 3	0.0015
Stock 4	0.0020

This means that if you buy or sell $10000 of stock 1, you will have to pay a $5 transaction cost.

2.3 Round details

There will be 4 rounds.

2.3.1 Round 0

For round 0, there will only be Stock1 and Stock2 available. You have about 40 minutes to prepare for this round.

2.3.2 Round 1

For round 1, there will only be Stock1 and Stock2 available. You have about 20 minutes to prepare for this round.

2.3.3 Round 2

For round 2, all symbols will be available. You have about 20 minutes to prepare for this round.

2.3.4 Round 3

Round 3 proceeds exactly as round 2. You have about 10 minutes to prepare for this round.

2.4 Data details

At the start of every round, you will be provided with a `train.csv` file, containing the data for the stocks available for that round. For round 1, only Stock1 and Stock2 are available. `train.csv` may look something like:

Stock1_Delay	Stock2_Delay	Stock1	Stock2
49.72	101.23	50.00	101.65
50.00	101.65	50.42	101.42
50.42	101.42	50.64	102.01

Each row represents a time step. The last two columns represents the data that you will receive only if you won the auction, whereas everybody has access first two columns. Based on these data, you should decide whether to buy or sell Stock1 or Stock2 (it is also okay to not trade at all).

2.5 Input

When we test your strategy, we will grade a `Trader` object on the timesteps for the given round. The starter code we provide initializes the class as follows:

```
# Place any imports you need here!
# Helpful packages may include numpy, pandas, and sklearn.

class Trader:
    def __init__(self):
        self.team_id = 0 # TODO: CHANGE TO YOUR TEAM'S PASSWORD.
        # Add any additional info you want

    def MakeTrades(self, time, stock_prices):
        """
        Grader will call this once per timestep to determine your buys/sells.
        Args:
            time: int
            stock_prices: dict[string -> float]
        Returns:
            trades: dict[string -> float] of your trades (quantity) for this timestep.
                Positive is buy/long and negative is sell/short.
        """

        trades = {}

        # TODO: PICK HOW TO MAKE TRADES.
        trades['Stock1'] = 1000
        if 'Stock2' in stock_prices:
            if stock_prices['Stock2'] > 123:
                trades['Stock2'] = 1000
            else:
                trades['Stock2'] = -1000

        return trades
```

You should feel free to make changes as necessary. We will only interact with your bot via the `MakeTrades` method.

2.6 Output

Your trader bot should contain the following methods:

- `MakeTrades(time, stock_prices)`: outputs how much of each symbol you choose to purchase in given round.
  - `time`: an integer argument indicating the current timestep.
  - `stock prices`: a dictionary argument with symbols as keys corresponding to the delayed price information for each symbol at a given timestep, in addition to keys for the current price information if you've won the auction, and float values which will be taken from the price data in the symbols for that round. For instance, in round 1 you will certainly have keys `Stock1_Delay`, `Stock2_Delay`, and you will in addition have `Stock1`, `Stock` if you won their respective auctions.

  This will output a dictionary of floats in each symbol `"Stock1","Stock2"` `,Stock3","Stock4"` corresponding to how much of each symbol you choose to trade at a given timestep. Your outputs keys must exactly align with the available stocks for a given round. For round 1 and 2 it must contain keys `"Stock1","Stock2"`, and for rounds 3 and 4 it must in addition contain keys `"Stock3","Stock4"`.

2.7 Position Limits

At each timestep, you are allowed to hold a position of maximum 1,000,000 USD across the four instruments. In order to manage risk, the maximum position you can hold in any particular instrument is capped at 600,000 USD. You may hold short (negative) or long (positive) positions as you see fit—these limits apply to the sum of the absolute values of your signed positions.

If your desired position for a particular timestep exceeds any of these limits, it will be automatically truncated.

3 Auction

Before each round, there will be an auction for extra information to be given in that round. The auctioned information is as follows: for each stock, at every given timestep, the firm provides you the up-to-date price information. This information may be helpful in improving execution and prediction for the various symbols.

3.1 Auction Format

There will be one opportunity to obtain contemporaneous timestep information each round. At the start of each round, teams may submit their bids through the submission website at the same time that they submit their bot.

The payment structure is as follows: for each stock, the teams with the 12 highest bids will each pay a price equal to the 13th highest bid. This amount will be subtracted from your PNL that round. Ties will be broken arbitrarily. **You may submit non-integer bids, but all bids must be non-negative**!

3.2 Scoring

There will be four rounds of submissions as described in the section on rounds. Your overall score is the sum of the PnLs over all the timesteps in all four rounds, subtracted by the amount you pay in the four sets of auctions.

The PnL per timestep for each symbol is the price of the symbol at the next time minus the price at the current time step, multiplied by the position you are holding at the current timestep. Essentially, this means that your position is liquidated from one time step to the next, and you will always have a maximum of $1,000,000$ USD to trade. There are no spreads, no other traders on the market, and no other elements of market micro structure you need to consider in this case.

Note: Due to the structure of the case, you should expect for your PNL per time step to be on the order of hundreds or thousands. The symbol and hedge fund price characteristics won't change from round to round—the data is generated consecutively and artificially partitioned into rounds.

Having multiple rounds of submissions allows you to modify and improve your code in between rounds. You may also decide to update your auction bids after seeing the results of the previous round. Having four rounds might seem overwhelming, but don't let that bother you. Rounds are spaced further apart in the beginning to allow you more time to iterate on your strategy as data comes in. It's totally viable for you to not significantly change your bot between submissions for the later rounds and use that time to perform more sophisticated statistical analysis that might lead to a more substantial revision at a later round.

3.3 Technical Requirements

Your submission will be run on our AWS server using Python 3.7.9 and will be able to import NumPy (v1.21.2), SciPy (v1.7.1), Pandas (v1.3.2), Scikit-learn (v0.24.2), and Statsmodels (v0.12.2). You should not assume that any other packages are available, nor should you try to use any other packages. **Attempts at abusing our system will result in immediate disqualification. This includes any attempts at reading or writing files.**

Your code is expected to run quickly. If your code takes more than 10 seconds to execute the 1000 timesteps, it may not run to completion. In such a situation, our system does its best to count all timesteps executed before timing out, but we make no guarantees.

Upon code submission, you will be provided feedback on your in-sample PNL for that round's available symbols, as well as any errors your code incurred. Please be sure every dictionary you return contains every symbol available.

3.4 Submission

All of your bots and bids will be submitted via the following link:

`http://3.95.7.226:<port>/submit`

Your port is determined as follows:

- trader1 - trader16: port 8080
- trader17 - trader 32: port 8081
- trader33 - trader 48: port 8082
- trader49 - trader 64: port 8083

You will be using the password we emailed out. Be careful to enter it correctly. Once you have submitted your code, you can go to the website below to check that your code uploaded properly. Your latest bot/bid submission is the only one that will be considered. Rounds are zero-indexed, so the round number will be 0 at first, and the last round will be round 3.

You will use the same website to download your team's information (e.g. csvs containing hedge fund information and price data, and PNL and auction information). To do so, use this URL with the correct substitutions:

`http://3.95.7.226:<port>/results/<team_password>/round/<round_id>/`

Once again, be sure to substitute your team password in correctly, and remember to zero-index the round number.