HKU QIDS 2023 Quantitative Investment Competition

Instruction for the Second-Round Competition

Dear Candidates,

Congratulations on your success in moving forward to our second-round competition! This second-round competition is an extension of the task in the first round, and we have 25 teams in total to join it. Please read this document carefully to better understand the task and rules. Should you have any enquiry about anything in this document, feel free to email hkuqids@outlook.com.

1. Details for the Second-Round Competition

1.1 Target

Our task in this round is to develop an investment strategy which can generate a good performance on the new test datasets. More specifically, you are provided historical datasets from d1 to d1700. You can do whatever back tests you like to do on the historical datasets to test your model performance, generate your decision rules, explore methods for portfolio allocation, and eventually develop a complete investment strategy.

After confirming all details of your investment strategy, you will **write down your strategy into Python code** and submit via the second-round API, which will then run your code with the test datasets and output the overall performance in the test period.

You may find your good modelling work in the first-round competition on Kaggle useful in producing signals as your decision rule inputs.

1.2 Outputs

Different from the first-round competition on Kaggle, in the second-round competition **you will need to output your investment decisions and corresponding portfolio allocation on each investment in each trading day on the test datasets**. The portfolio allocation is expressed by percentage, which means the percentage of your total capital in this investment on this trading day. As is previously stated, we only consider a two-day holding horizon. Here we assume that you split all your capital equally into two parts, and in each day you invest in one part to long the selected investment and close the position after two days.

Also, to better simulate the market, **we set a transaction fee of 0.04 percent when you open a position and another transaction fee of 0.20 percent when you close a position**. If your strategy choose to open and close a position with the same investment at the same moment, the transaction fee will be offset correspondingly.

Here is an example that can help you to better understand the logic: Assume the test set length is from d1701 to d1703. On d1701p50 you choose to invest 50% of available capital

into investment 1 and the rest 50% into investment 2. Your output on d1701 should be pd.Series([0.5, 0.5, 0, 0, 0, ⋯]) with the length be 54. On d1702p50 you choose to invest 50% of available capital into investment 1 and the rest into investment 2 and 3 equally. Your output on d1702 should be pd.Series([0.5, 0.25, 0.25, 0, 0, ⋯]). On d1703p50 you choose to invest 25% of available capital into investment 2 and 50% of available capital into investment 4. Your output on d1703 should be pd.Series([0, 0.25, 0, 0.5, 0, ⋯]).

Let's further assume that the daily returns of investment 1, 2, 3, 4 are:

	investment 1	investment 2	investment 3	investment 4
d1701p50-d1702p50 (d1702)	0.01	0.04	-0.03	0.1
d1702p50-d1703p50 (d1703)	0.02	0	-0.02	0.1
d1703p50-d1704p50 (d1704)	0	0.08	-0.1	0.1
d1704p50-d1705p50 (d1705)	-0.01	0.06	0	-0.1

You will then receive your daily return performance as follows:

	portfolio return	explanation
d1701p50-d1702p50 (d1702)	0.5 * (0.01 * 0.5 + 0.04 * 0.5) – 0.0004 * 0.5 * (0.5 + 0.5)	half capital in the market; half capital open position
d1702p50-d1703p50 (d1703)	0.5 * (0.02 * 0.5 + 0 * 0.5) + 0.5 * (0.02 * 0.5 + 0 * 0.25 – 0.02 * 0.25) – 0.0020 * 0.5 * (0.5 + **0.5 - 0.25**) – 0.0004 * 0.5 * (0.5 + 0.25 + 0.25)	all capital in the market; half capital open position; half capital close position; **some close position fee for investment 2 offset**
d1703p50-d1704p50 (d1704)	0.5 * (0 * 0.5 + 0.08 * 0.25 – 0.1 * 0.25) + 0.5 * (0.08 * 0.25 + 0.1 * 0.5) – 0.0020 * 0.5 * (0.5 + 0.25 + 0.25) – 0.0004 * 0.5 * (**0.25 – 0.25** + 0.5)	87.5% capital in the market; 37.5% capital open position; half capital close position; **all open position fee for investment 2 offset**
d1704p50-d1705p50 (d1705)	0.5 * (0.06 * 0.25 - 0.1 * 0.5) – 0.0020 * 0.5 * (0.25 + 0.5)	37.5% capital in the market; 37.5% capital close position

Your scoring will then be based on the daily portfolio return series on the test period. **Please be noticed that, the sum of the output for each day must not exceed 1. Also, the range for each element in the output for each day must be in [0.01, 1]**, which means that short-selling is not allowed, and the percentage of capital in one investment should not be too small. **The amount of non-zero numbers in the output for one trading day is viewed as**

the number of investment selected for that day.

1.3 Scoring

After each successful submission, our system will score and output overall performance of your strategy, which includes the **cumulative return** (Absolute Profitability, the bigger the better), **Sharpe ratio** (Risk-adjusted Profitability, the bigger the better), **maximum drawdown** (Extreme Risk Measurement, the smaller the better) and **median amount of investment selected per trading day** (Capital Capacity, the bigger the better). With all successful submissions, you need to manually select ONE submission which will be put into the overall ranking. **For each aspect, your strategy will have a ranking in all teams' strategies (from 1 to 25), and your score in this aspect will be calculated by 10 – 0.2 * (n - 1), where n stands for your ranking.**

After obtained your scores in all four aspects, your final score will be calculated by:
(0.4 * Absolute Profitability Score + 0.3 * Risk-adjusted Profitability Score + 0.2 * Extreme Risk Measurement Score + 0.1 * Capital Capacity Score) * (1 - Penalty)

About Penalty: Please be noticed that, in order to control the phenomenon of test dataset overfitting, which is very commonly seen in the first-round competition on Kaggle, **we allow up to 7 successful submissions with no penalty. After seven successful submission, any successful submissions will lead to a 6 percent score deduction as a penalty of test dataset overfitting. And the maximum of successful submission is 12**, which means the maximum penalty is 30% for any team that has 12 successful submissions.

After the second-round competition finish, the top 10 teams will be selected to move forward to the final-round presentation.

2. Data Introduction

The train dataset structure is exactly the same as that in the first-round competition on Kaggle. The only difference is that we provide the test datasets in the first-round competition on Kaggle as the train datasets in this second-round competition.

For the test dataset feed in the API, it will also be the same as that in the first-round competition on Kaggle. Here are two examples of what the structures of "market_df" and "fundamental_df" will be like in the "get_decisions(market_df, fundamental_df)" function:

```
In [42]: market_df
Out[42]:
          date_time     open    close  ...     volume         money  date
4587300    s0d1700p1  35.3823  35.1759  ...  1281100.0  4.504193e+07  1700
4587301    s1d1700p1  84.7232  84.9781  ...    90400.0  7.685102e+06  1700
4587302    s2d1700p1  82.7205  83.1210  ...   191448.0  1.590485e+07  1700
4587303    s3d1700p1  15.3182  15.5245  ...   824900.0  1.271619e+07  1700
4587304    s4d1700p1  17.0539  17.2845  ...  1229790.0  2.108100e+07  1700
...              ...      ...      ...  ...        ...           ...   ...
4589995  s49d1700p50  27.6868  27.6868  ...   329087.0  9.110941e+06  1700
4589996  s50d1700p50  28.9370  28.9127  ...    59517.0  1.720829e+06  1700
4589997  s51d1700p50   9.9896   9.9896  ...    87100.0  8.700919e+05  1700
4589998  s52d1700p50  71.9055  71.9662  ...   126493.0  9.103476e+06  1700
4589999  s53d1700p50  16.1193  16.1071  ...    74900.0  1.205762e+06  1700

[2700 rows x 8 columns]

In [43]: market_df.columns
Out[43]: Index(['date_time', 'open', 'close', 'high', 'low', 'volume', 'money', 'date'], dtype='object')
```

```
In [48]: fundamental_df.head(10)
Out[48]:
      date_time  turnoverRatio  transactionAmount  ...       ps       pcf  date
91746   s0d1700         4.3319            89757.0  ...   3.5318   35.4291  1700
91747   s1d1700         0.5391            23155.0  ...   6.3305  -90.4791  1700
91748   s2d1700         0.5276            24381.0  ...   9.7966  244.4131  1700
91749   s3d1700         2.5263            21168.0  ...   0.5485    4.1089  1700
91750   s4d1700         1.4400            54454.0  ...   4.6359   70.8454  1700
91751   s5d1700         2.6516             7538.0  ...   4.3510  -24.6549  1700
91752   s6d1700         1.8556            51535.0  ...   0.3210  -48.2615  1700
91753   s7d1700         0.2005             6690.0  ...  13.0369   15.8168  1700
91754   s8d1700         1.4409            18700.0  ...   5.5814   70.4798  1700
91755   s9d1700         0.8934            19399.0  ...   0.9684    4.4606  1700

[10 rows x 9 columns]

In [49]: fundamental_df.shape
Out[49]: (54, 9)

In [50]: fundamental_df.columns
Out[50]:
Index(['date_time', 'turnoverRatio', 'transactionAmount', 'pe_ttm', 'pe', 'pb',
       'ps', 'pcf', 'date'],
      dtype='object')
```

You may find your good notebook work in the first-round competition on Kaggle useful in fulfilling the code structure requirement.

3. How to Use the API?

We have provided the code framework and sample code to you. **Please refer to "strategy_framework.py" to better understand the requirements on your code structure.** Please refer to the "code_examples" folder to read the sample code which can successfully run on the API and generate an overall performance. **Please be noticed that you MUST rename your strategy code into "strategy.py", and you MUST submit a file named "requirements.txt", in which please write down any package (the name must be the same as those in the "pip install XXX" syntax, not those in the "import XXX" syntax) that your code need to install.** Our system will automatically check and install the packages for your code if necessary. **Please refer to "submit.py" to better understand how to submit your code to the API.**

You will have received your access_token which will help you submit your code to the API and view your strategy performance via
http://competition.hkuqids.com:8880/ui/submissions?group_id=G000&access_token=kkkkk
Please replace "G000" with "G" + group_id and "kkkkk" with access_token.

Your code will be run automatically after the submission. **Please be noticed that you CANNOT submit another code while you have already submitted one version of code and the result generation process has not finished.** For each day you can have up to five successful submissions. After you have reached this limit, you cannot make any submission today. If any error is raised when running your code, the result generation process will be terminated and the error message will be sent to you by email to the email address of your team. If your code run successfully, you will receive the overall performance information, as well as your strategy's cumulative return graph, by email to the email address of your team.

Your code will be run in an offline environment. For the resource constraint, every submission is allowed to use up to 6 cores of CPU and 16G RAM, or otherwise your code will be terminated and an error will be raised. You code should run for no more than 6 hours, or otherwise your code will be terminated and an exceed-time error will be raised. Our system allows up to five code running at the same time, based on the first-come-first-serve method. Other codes will be put in a queue and wait until more computation quotas become available.

4. Logics Beneath the API

After you submit your code, the system will first check and install the packages if necessary and check if there is any prohibited code in your notebook (including "os" package, "raise" syntax, and any other syntax that try to directly call commands towards the underlying operation system). After passing the checking procedure, the result generation system will feed the market data and fundamental data of the test datasets day by day into the "get_decisions(market_df, fundamental_df)" function, and store the outputs for each day. After walking through all the test datasets, the system will calculate your strategy performance based on all decisions and the corresponding investment returns. The results will then be stored in the database and published on the dashboard.

5. Suggestions to Get A Good Score

1. Make sure your strategy contains clear economic logic when facing unclear market conditions.
2. Make sure your strategy can perform robustly on the train datasets. Test dataset overfitting becomes very hard and expensive to implement now.
3. The result generation process do takes time, so do not be the deadline fighter!
4. Write responsible code! All code trying to violate the rules or cause computation recourse hazards will be viewed invalid, and the competition host reserve all rights to cancel the eligibility of the relevant teams when necessary.

6. Timeline

The second-round competition submission deadline is set at 2023.4.15 23:59 UTC+8. Our competition host members will then check the results and send out invitations for the final-round presentation at 2023.4.20 00:00 UTC+8.

An information session will be held on 2023.3.28 15:00-15:30 UTC+8. You may join the information session via the following Zoom link:
https://hku.zoom.us/j/9513418561?pwd=NUIjSytYa0l4VFVpVnlHVVIyWUNjUT09

The final-round presentation is expected to be held at 2023.4.29. More detailed information about the final-round presentation will be provided with the invitations.

GOOD LUCK CANDIDATES!

HKU Quantitative Investment and Data Science Society
2023-03-19