Name: Brandon Tan Jun Da
Email: brandontan.2019@business.smu.edu.sg

Writeup on Skills Challenge – March 2021

Way of Work

I started by reading the paper, getting a baseline understanding of what it was trying to achieve. After which, I jumped to Section 3.1.1 and noted down the approaches and started to brainstorm by writing out how the flow might go.

Once the flow was decided, I went to the GitHub repository of the previous cohort's closed pull request history to take note of what were the expectations required of participants. In particular, I noted aarondeb's work (as it was noted as an impressive piece of work) and the comments.

Here are the insights that I gained:

1. Grammar is important for docstring and comments.
2. Follow the existing docstring used by the Hudson and Thames team. I believe it is sphinx.
3. Modularity was important, and that "function/method to more or less fit on a single screen, else one should consider writing helper functions".

With the insights in mind, I created a jupyter notebook and started prototyping the various approaches. I opted for using a notebook as I felt that it would allow me to rapidly iterate my code. For each approach, I looked at what I had written out earlier and matched it to the various appropriate Numpy functions to speed up the code. Once I had gotten a concrete sensing of what each function might entail, I drew up the process workflow diagram (can be found on page 3) and locked in the overall architecture of the partner selection framework.

With the framework and code ready, I began to port them over to a python file and implemented the partner selection framework. My IDE of choice was VSCode, as I have been coding on it for quite some time, and thought that a sudden switch to Pycharm might be a tad abrupt.

To save time on docstring formatting, I installed the docstring helper extension and switched it to sphinx docstring format. To meet PEP8 requirements, I uploaded my code onto PEP8online.com to check for code issues and fixed it (except for line too long errors).

Design Ideas

Although the paper outlined that for every given security, there should be 3 potential partners, my implementation opted to allow users to choose how many potential partners they would like. It also allows users to scope down the number of top correlated securities.

Here are the design philosophies that I have kept in mind while designing the module for the end-user, which are inspired by Jakob Nielson's 10 general principles for interaction design.

1. Visibility of module status/information
   a. Usage of tqdm for loops
   b. Function to print descriptive statistics
2. User control and freedom
   a. User freedom to decide on how many partners
   b. User freedom to decide on the scope of correlated securities
3. Minimalist design
   a. Parameters are only there if they are needed

With regards to the source code of the partner framework, my main aim was to keep it modular and readable. I will share one example for each aim.

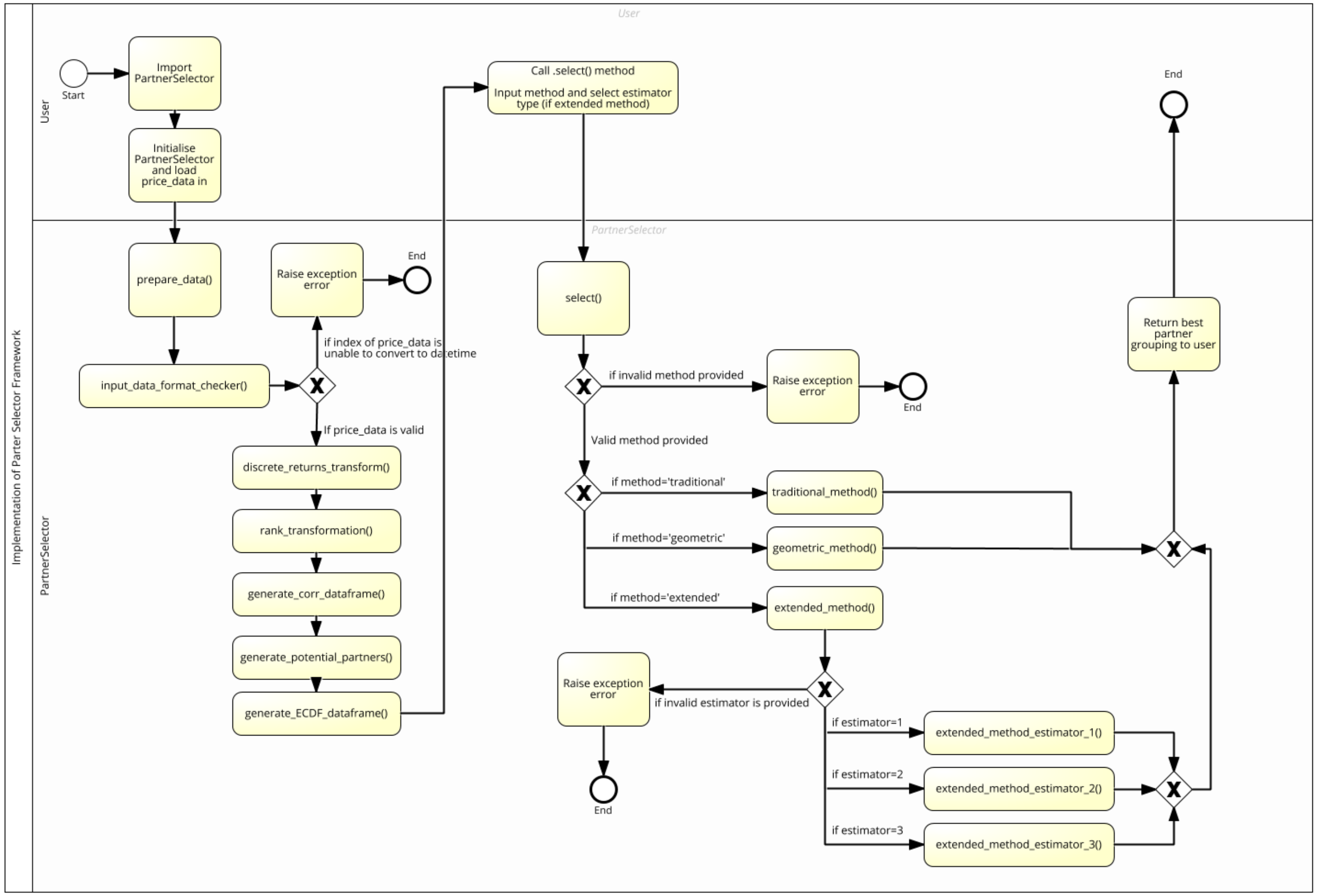
Modular

For the private function to prepare the dataset that was loaded by users, I opted to implement it all separately. Concretely, getting daily discrete returns, performing rank transformation and getting the correlation matrix were all separate private functions. These were then packaged together into "_prepare_data()" that would call all these functions.

Readable

Under the extended approach – estimator 3, I was considering using np.einsum for the double summation. I decided against it in the end, as the np.einsum syntax was not straightforward and if someone wanted to optimise the source code, it might produce unnecessary difficulties for them. The result was that I used two for loops and dot products to mirror the formula as closely as possible.

Partner Selection Framework Process Workflow Diagram



Implementation of Parter Selector Framework
User
Start
Import PartnerSelector
Initialise PartnerSelector and load price_data in
Call .select() method
Input method and select estimator type (if extended method)
End
PartnerSelector
prepare_data()
Raise exception error
End
if index of price_data is unable to convert to datetime
input_data_format_checker()
If price_data is valid
discrete_returns_transform()
rank_transformation()
generate_corr_dataframe()
generate_potential_partners()
generate_ECDF_dataframe()
select()
if invalid method provided
Raise exception error
End
Valid method provided
if method='traditional'
traditional_method()
if method='geometric'
geometric_method()
if method='extended'
extended_method()
Return best partner grouping to user
Raise exception error
if invalid estimator is provided
End
if estimator=1
extended_method_estimator_1()
if estimator=2
extended_method_estimator_2()
if estimator=3
extended_method_estimator_3()

Learnings
I found that the strict enforcement of documentation, PEP8 and commenting made me more aware of the shortcomings of my existing coding habits. Before this challenge, my coding was more free-styled as all I had to do was to deploy endpoints and as long as the flask app worked, no one would say anything.

I also enjoyed implementing the tasked paper and seeing the ideas come to life. I think that the skills challenge provided me with a good glimpse into what will be required of researchers and it gave me an appetiser of this career path that I am pursuing.

Identified Areas for improvement

Coding

1. The code for extended and geometric method could probably be further optimised, as they currently take very long to run. I should dive deeper into algorithms and learn to write more efficient code.

2. Under the extended method, I implemented the various estimations as different functions for modularity. However, they have repeated code between them. I was wondering if it would be better practice to maintain modularity or reduce repeated code.

3. The skills challenge requested for great visualisations, but I am not too sure what great visualisations are available as the intended outcome for the module was to present the end-user with the best Q (quadruple, in cases of n=4). I am keen to learn more about this aspect.

Math

I thought that I was quite fortunate that for the three approaches, they were relatively straight forward. Coming from a Bachelors in Business Management with a major in quantitative finance, I find myself being required to take more business modules and thus lacking in mathematical skills. I have attempted to catch up by watching MIT videos on linear algebra and taking courses on machine learning and deep learning (Andrew Ng).

I still have to further build my knowledge on statistics and conduct some self-studying in my free time, as I couldn't fully grasp copulas on my first reading.

Overall, this has been a wonderful skills challenge and I would like to thank the team for organising it.