Project 2: Manipulating data in Pandas

Introduction

Effectiveness in Finance research and industry necessitates a facility with data aggregation, combination, selection, and manipulation. In the previous project, you are asked to write general code to assemble a data set from different sources, and developed foundational skills of data acquisition, cleaning, and merging. In this assignment, we will help you develop those skills further by mimicking the initial steps required to implement an event study. You will combine data from multiple CSV files into a single table, calculate individual and abnormal stock returns, and compute returns for a portfolio of stocks.

The main goal of this assessment is to create data frames with the returns and abnormal returns of several stocks. This task can be broken down into a series of intermediary steps:

- Create a data frame containing the stock price data for multiple companies.
- Calculate returns for each of these companies.
- Subtract the market returns from each individual stock returns to generate abnormal returns.

You will then compare the performance of these companies over a pre-determined period.

The Source Files

All required files are included in a zip archive with the following structure:

```
<ZIPFILE>
|__ project2/
|   |
|   |__ __init__.py
|   |__ config.py
|   |__ project_desc.pdf
|   |__ zid_project2.py
|   |
|   |___data/
|   |   |   <many csv files here>
```

where

- `project2/` represents the main folder containing all the project files.
- `zid_project2.py` contains the functions you need to write for this project. This is the only file you need to submit.
- `project_desc.pdf` is the PDF version of this document.
- `data/`: This is the sub-directory where all the data files for this files are stored. Inside this folder you will find many files. Each `<tic>_prc.csv` contains stock price data for the ticker `<tic>`. These CSV files include the column names in a header row of text. In addition, this folder contains a file called `ff_daily.csv`, which includes market returns.

- `config.py` is the configuration module for this package. **You should not modify this file**.

Instructions

Important: This project is a group project. Do not exchange complete or partial codes with students from **other** groups. Please do not post any project related questions in public online forums.

Again, please do not post any project related question in a public online forum, including the discussion board in ED. We will deduct marks from your project if you violate this important rule.

Preparing the files for this project

1. Copy the `project2` folder into the `toolkit` project folder. Afterwards, your `toolkit` folder will look like:

```
toolkit/                      <- PyCharm project folder
|   ...
|__ project2/                 <- New folder
|   |
|   |__ __init__.py
|   |__ config.py
|   |__ project_desc.pdf
|   |__ zid_project2.py       <- This is the only file you need to edit
|   |
|   |___data/
|   |   |   <many csv files here>
|   ...
|__ toolkit_config.py         <- Your toolkit_config.py (required)
|   ...
```

2. Unless explicitly stated below, **do not change any variable, import statement, function, or parameter names in the `project2` module.**

How to complete this project

This project has eight parts, which should be completed in sequence. Only parts 2 to 8 will be marked. You can find the number of marks for each part at the end of this document. Each part is described in detail in the next section.

Overview:

- Part 1: Read the documentation for the following methods:
  - pandas.DataFrame.mean
  - pandas.Series.add
  - pandas.Series.prod
  - pandas.Series.dropna
- Parts 2 to 7: Complete the functions in `zid_project2.py`. See the step-by-step instructions below.
- Part 8: Answer the described in Part 8 below by setting the value of the relevant variables in `zid_project2.py` (i.e., `Q1_ANSWER`, `Q2_ANSWER`, etc...)

How to submit the project and the peer review form

Please make sure each *group* submits one project and each *student* submits a peer review form:

1. Each **group** should submit a single `zid_projec2.py` file with the completed version of `zid_project2.py`. Your team should choose one member who will be responsible for submitting the project. There is no need to tell us in advance who the team representative will be.
    - The group representative should copy and paste the entire contents of the `zid_project2.py` module to ED. The `zid_project2.py` is the only file the group representative will need to submit.
    - **Please make sure only one team member submits the group project.** If multiple members of the same group submit, we will only consider the last submission by any student belonging to the group.
    - Remember to press "Mark" to submit your project. Your project will not be submitted until you do so.
2. Each **student** must also complete an individual peer review form: The peer review form will be available in Moodle (not ED). We will make an announcement when the form is ready to be completed. This form is to be completed individually by each student.

To make sure this process is clear to everyone, here is an example. Suppose a group consists of three students `A`, `B`, and `C`. The group decides that student `A` will submit the `zid_project2.py` file.

- Student `A` will submit the file through ED. Student `A` will copy the content of the completed `zid_project2.py` file, navigate to the "Submit your codes here" slide in ED, and then paste the code. After that, student `A` will press "Mark" to submit the code.
- Students `B` and `C` should ignore the "Submit your codes here" slide in ED.
- Students `A`, `B`, and `C` will then separately complete their individual peer review forms in Moodle (not ED).

Completing the `zid_project2.py` module

After setting up your PyCharm development environment with the project files (see instructions above), modify the `zid_project2.py` module by following the steps below, in sequence.

Part 1: Read the relevant documentation

Read the documentation for the following methods:

- pandas.DataFrame.mean (note the parameter `axis` which will indicate if the mean will be computed column-wise or row-wise)
- pandas.Series.add
- pandas.Series.prod
- pandas.Series.dropna

Part 2: Include a statement to import the `config` module

Open the `config.py` file included in this project in PyCharm. **Please do not modify the `config.py` module.** Note that this file includes many constants (e.g., `DATADIR`, `FF_CSV`, `TICMAP`) and a function called `standardise_colnames`. You should not modify the file `config.py`.

The `config.py` module must be imported by the `zid_project2.py` module. This is because the `zid_project2.py` module needs access to the constants and the `standardise_colnames` function defined in `config.py`.

Complete the import portion of the `zid_project2.py` module by creating a new import statement. This statement should import the module `config.py` which is part of the zip file provided to you. Your import statement must:

- Take into account that the `config.py` module is inside the `project2` package.
- Import the `config.py` module using "cfg" as an alias (so, "as cfg")

Part 3: Complete the `read_prc_csv` function

Complete the indicated part of the function `read_prc_csv.` so it produces the data frame described in the *docstring*.

You can test this function by calling the `_test_read_prc_csv` test function.

Part 4: Complete the `mk_prc_df` function

Complete the indicated part of the function `mk_prc_df.` Make sure that this function returns a data frame as described in the *docstring*. Remember that the sample period for the *docstring* example may be different than that in your specific source data.

You can test this function by calling the `_test_mk_prc_df` function.

Part 5: Complete the `mk_ret_df` function

Complete the indicated part of the function `mk_ret_df.` Make sure that this function returns a data frame with the same format as described in the *docstring*.

You can test this function by calling the `_test_mk_ret_df` test function.

Part 6: Complete the `mk_aret_df` function

Complete the indicated part of the function `mk_aret_df.` Make sure that this function returns a data frame with the same format as described in the *docstring*.

You can test this function by calling the `_test_mk_aret_df` test function.

Part 7: Complete the auxiliary functions

Complete the following auxiliary functions following the instructions specified in their docstrings:

- `get_avg`: Calculates the average value of a column for a given year.
- `get_ew_rets`: Calculates the returns on an equally-weighted portfolio of stocks.
- `get_ann_ret`: Calculate the annualised return for a given period.

You can test these functions by calling appropriate test functions.

Part 8: Answer a few questions

For this part of this project, you should answer the questions below. Your answers should be included in the `zid_project2.py` module. For example, answer `Q1` by setting the value of `Q1_ANSWER` in the `zid_project2.py` file.

All your answers should be strings. If they represent a number, include 4 decimal places (e.g., `'1.1234'`). When marking this part of the project, we will ignore string capitalization (i.e., lowercase vs uppercase characters).

- Q1: Which stock in your sample has the highest average daily return for the year 2020 (ignoring missing values)? The sample should include all tickers included in the dictionary `config.TICMAP`. Your answer should include the ticker for this stock (as a string).
- Q2: What is the annualised return for the EW portfolio of all your stocks in the `config.TICMAP` dictionary from the beginning of 2010 to the end of 2020?
- Q3: What is the annualised daily return for the period from 2010 to 2020 for the stock with the highest average return in 2020 (the one you identified in the first question above)?
- Q4: What is the annualised daily ABNORMAL return for the period from 2010 to 2020 for the stock with the highest average return in 2020 (the one you identified in the first question Q1 above)? Abnormal returns are calculated by subtracting the market return from the individual stock return.

Important:

- The file `zid_project2.py` contains placeholders for your answers.
- You should replace the relevant variables in `zid_project2.py` file with your answers. For instance, your answer to Q1 should be included in the variable `Q1_ANSWER`.
- You can create a separate module (you can call it `main.py` if you want) and then use the functions defined above to answer the questions below.
  HOWEVER, THE ONLY MODULE YOU SHOULD SUBMIT IS zid_project2.py.
- All your answers should be strings. If they represent a number, include 4 decimal places.
- Here is an example of how to answer the questions below. Consider the following question:

  > Q0: Which ticker included in config.TICMAP starts with the letter "C"? Q0_ANSWER = '?'

  You should replace the '?' with the correct answer:

  > Q0_ANSWER = 'CSCO'

Administrative Guidelines and Hints

We will enfore the following:

1. This assessment must be completed in groups but you should not cooperate with students from other groups. Failure to do so may result in a full loss of marks.
2. Late submissions are allowed, but will be penalised following the guidelines described in the course outline.

Hints

Your code should be portable, working in a variety of settings. For example, we should be able to run your codes in different computers using different operating systems. We should also be able to import and run your code from other modules.

The following hints should help you correct any portability mistakes:

1. The contents of your `zid_project2.py` module **must not contain any direct reference to folders in your computer.** In other words, you must use the variables in the `config.py` and the `os` module to create path variables.
2. When writing functions in the file `zid_project2.py`:
   - Do not modify the function names or the parameters.

- Only modify the parts indicated by the `"<COMPLETE THIS PART>"` tag.
- You should not import any other module (with the exception of `config` as described in Part 1.)

3. **Only** submit the `zid_project2.py` module. Make sure your code works with this module only. No other modules can be submitted.

How we will mark your assessment

The following parts of this assessment will be marked. This project is worth a total of 100 marks.

- Part 2: Importing the config module (3 marks)
- Part 3: Complete the read_prc_csv function (10 marks)
- Part 4: Complete the mk_prc_df function (15 marks)
- Part 5: Complete the mk_ret_df function (15 marks)
- Part 6: Complete the mk_aret_df function (15 marks)
- Part 7: Auxiliary functions:
  - `get_avg` function (10 marks)
  - `get_ew_rets` function (10 marks)
  - `get_ann_ret` function (10 marks)
- Part 8: Each question is worth 3 marks (for a total of 12 marks)