Project 2: Manipulating data in Pandas

Introduction

Effectiveness in Finance research and industry necessitates a facility with data aggregation, combination, selection, and manipulation. This assignment helps you develop those skills 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:

```
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 do not need to modify this file.

Instructions

Important: This project is to be completed individually. Do not exchange complete or partial codes with other students. Please do not post any of 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/
|    ...
|__ toolkit_config.py
|    ...
|__ project2/
|    |
|    |__ __init__.py
|    |__ config.py
|    |__ project_desc.pdf
|    |__ zid_project2.py
|    |
|    |___data/
|    |    |    <many csv files here>
```

2. Unless explicitly stated below, **do not change any variable, import statement, function, or parameter names in the `zid_project2.py` 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: Create a new module (you can call it anything you want, for instance `project2.main.py`). Then use the functions you created in `zid_project2.py` to answer the questions in Part 8 below.
- After you have completed the functions in the `zid_project2.py` module in PyCharm and answered the questions in Part 8, copy and paste the entire contents of this module to ED. The `zid_project2.py` is the only file you will need to submit to complete this assessment.
- Press "Mark" to submit your project. Your project will not be submitted until you do so.

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

For this assessment, 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.py` module

Open the `config.py` file included in this project in PyCharm. 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`.

To get access these constants and the `standardise_colnames` function, the `config.py` module needs to be imported first.

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. Make sure you import this module using the "cfg" alias (so, "as cfg").

After including this import statement, you can use the test function `_test_cfg` to make sure the config module is being imported correctly and tho check the location of the files for this project.

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 following questions:

- 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.
- 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 individually. Failure to complete the assignment on your own 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. It should be sufficient to copy your code from PyCharm to *Ed* for submission. If your code works on your computer, but not on *Ed*, then you have not made your code portable. 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)