M07-HW-2

July 27, 2023

1 Metadata

```
Course:  DS 5100
Module:  07 Python Classes
Topic:   HW 07 Stock Class
Author:  R.C. Alvarado
Date:    7 July 2023
```

2 Student Info

- Name: Lindley Slipetz
- Net UD: ddj6tu
- URL of this file in GitHub: https://github.com/sliplr19/DS5100-ddj6tu/blob/main/lessons/M07/M07-HW-2.ipynb

3 Instructions

In your **private course repo on Rivanna**, use this Jupyter notebook and the data file described to write code that performs the tasks below.

Save your notebook in the `M07` directory.

Remember to add and commit these files to your repo.

Then push your commits to your repo on GitHib.

Be sure to fill out the **Student Info** block above.

To submit your homework, save the notebook as a PDF and upload it to GradeScope, following the instructions.

TOTAL POINTS: 12

4 Overview

In this assignment you will define a class and use it to perform the requested tasks.

Before answering the questions, read the market data from `apple_data.csv` into a Pandas dataframe. The file is in the HW for this module in the course repo.

5 Setting Up

[2]:
```
import pandas as pd
import numpy as np
```

6 Prepare the Data

Read in the dataset from the attached file `apple_data.csv` using `pd.read_csv()`.

[4]:
```
apple = pd.read_csv("apple_data.csv")
apple.head().T
```

[4]:
```
                    0           1           2           3           4
date       2020-01-02  2020-01-03  2020-01-06  2020-01-07  2020-01-08
adj_close  298.829956  295.924713  298.282715  296.879883  301.655548
```

7 Task 1

(5 PTS)

Define a class with these features:

Class Name: `Stock`

Attributes: - `ticker`: a string to hold the stock symbol - `sector`: a string to hold the sector name - `prices`: a dataframe to hold the prices for the stock

Methods: - `print_sector` to just print out the sector string. - `get_row_count` to count the number of rows in the price dataframe. Set an attribute "price_records" equal to this count. -`__init__` to build objects. Initialize with the three attribute values passed to the constructor.

[17]:
```
class Stock:

    def __init__(self, sector, prices, ticker):
        self.sector = sector
        self.prices = prices
        self.ticker = ticker

    def print_sector(self):
       print(self.sector)

    def get_row_count(self):
        self.price_records = self.prices.shape[0]
        return(self.price_records)
```

8 Task 2

(1 PT)

Create an instance of your class with the these initial values: - `ticker`: 'AAPL' - `sector`: 'technology' - `prices`: *the imported price dataframe*

Then Use the dot operator to print the stock's ticker.

```
[18]: stockint = Stock('technology', apple, 'APPL')
      stockint.ticker
```

```
[18]: 'APPL'
```

9 Task 3

(1 PT)

Use the `print_sector()` method to print the sector.

```
[19]: stockint.print_sector()
```

```
technology
```

10 Task 4

(2 PTS)

Use the `get_row_count()` method to compute the number of price records and set price_records.

Use the dot operator to access the stock's price_records, printing the result.

```
[20]: stockint.get_row_count()
```

```
[20]: 135
```

```
[21]: stockint.price_records
```

```
[21]: 135
```

11 Task 5

(1 PT)

Add a new column called `'month'` to the `prices` attribute and put the month number there.

Hint: You can use `.apply()` with a lambda function to split the month string and keep the second element.

```
[37]: import calendar as cal
      import locale
```

```
stockint.prices['date'] = pd.to_datetime(stockint.prices['date'])
stockint.prices['month'] = stockint.prices['date'].dt.month_name()
```

12 Task 6

(1 PT)

Use `.groupby()` to compute the mean `adj_close` by month. Save your result is a dataframe, not a series.

[42]:
```
mean_dat = stockint.prices.groupby('month')['adj_close'].mean().to_frame()
mean_dat
```

[42]:
```
            adj_close
month
April      271.650839
February   310.271843
January    310.337596
July       378.385999
June       345.806360
March      261.735581
May        309.785164
```

[40]:
```
# Another method
# my_stock.prices.groupby('month').agg({'adj_close':'mean'})
```

13 Task 7

(1 PT)

Plot the mean `adj_close` by month using a simple line plot.

[ ]:
```

```