553.688 Computing for Applied Mathematics
Fall 2023
Final Assignment - Form 4 Filings

When certain executive employees of a publicly traded US company buy or sell shares of stock in their company, they are required to file a form with the US Securities and Exchange Commission (SEC) detailing the nature of the transaction. These filings, which are referred to as Form 4 filings, must be submitted within 2 business days after the date of a transaction. In this assignment, will be

This final assignment will consist of 3 parts:

- **In Part 1** you are tasked with writing a function that will create a pandas data frame to work with from the data made available to you. **This part of the assignment must be completed by Sunday December 10th. There will be no exceptions to this because on Monday December 11th, you will be provided with a correct version of the data frame.** [1]
- **In Part 2** you are tasked with performing some analysis of the data using the data frame from Part 1. This part of the assignment is due on Tuesday December 19th at noon.
- **In Part 3** You will be assigned a training dataset (with response variable included) and a test dataset (with response variable excluded) and you will be asked to produce predictions for the test dataset.

Important Reminder

When you work on your assignment, you should always write your own code. You should not share your code with anyone in the class. Any copying of code is considered plagiarism and a form of academic misconduct. Your work will be carefully checked and evidence of violating the rules will be followed up with potentially serious consequences.

Data

The PFE filings have been downloaded from the SEC site and are available to you as a zip file using this link:
`https://www.ams.jhu.edu/~dan/Form4Filings/PFE.zip`

[1]This assignment is posted early so that you can and should get started on it early. If you wait until the last minute and then get sick and don't complete this first part in time you will get no sympathy since you should have exercised better time management

You should download this file and unzip it in some location of your computer I will refer to as **basefolder**. In basefolder, you will see 4,750 subfolders:
0000078003-02-00031
0000078003-03-00034
⋮
Each of the 4,750 subfolders contains a single file called "full-submission.txt" which is a filing on behalf of one *owner*.

Part 1

Your first task is to write a function called **CreateDataFrame** that takes as input a string giving the path to a folder so that when the function is called, you will pass it the string representing the basefolder where you extracted the zip file to as the function argument. Your function should output a dataframe.

The data making up the dataframe should be extracted from the filings as follows:

- Each file/filing may or may not contain an XML *ownership document*. If the file contains such a document, it will always be defined as the text that starts with an `<ownershipDocument>` tag and ends with an `</ownershipDocument>` tag.
- For each file that does contain an XML ownershipDocument you should extract the text making up that ownership document as a string, and do further extraction of data needed from that document using the **xml.etree.ElementTree** package as described in the Jupyter notebook ("XML and Element Tree.ipynb") that was provided in Lecture 18. You are required to use this package to carry out the tasks!
- Each ownership document can describe so-called *derivative transactions* and *non-derivative transactions.* We are only interested in non-derivative transactions. All derivative transactions should be ignored.
- Some of the ownership documents do not contain `rptOwnerName` tags. These documents should be ignored.
- Each ownership document can describe multiple non-derivative transactions. Your dataframe should contain a row for every non-derivative transaction found in an XML ownership document.
  - non-derivative transactions will always be described in material appearing between a `<nonDerivativeTransaction>` and a `</nonDerivativeTransaction>` tag
  - your data frame should contain the following columns with the following information for each nonderivative transaction

* **Folder**: the folder name in which the filing appears e.g. "000078003-02-00031".
* **OwnerName**: found between the `rptOwnerName` opening and closing tags (there should only be one of these - see above).
* **IsDir**: an indicator (0/1) as to whether the owner is a company director (see tag `reportingOwnerRelationship`).
* **IsOff**: an indicator (0/1) as to whether the owner is a company officer (see tag `reportingOwnerRelationship`).
* **IsTen**: an indicator (0/1) as to whether the owner is a ten percent owner (see tag `reportingOwnerRelationship`).
* **SecTitle**: the security title, which appears between `<securityTitle>` and `</securityTitle>` tags.
* **TransDate**: the transaction date, which appears between `<transactionDate>` and `</transactionDate>` tags.
* **Shares**: the number of shares traded, which appears between `<transactionShares>` and `</tranactionShares>` tags.
* **PPS**: the price per share for the shares traded, which appears between `<transactionPricePerShare>` and `</tranactionPricePerShare>` tags.
* **ADCode**: a code A or D indicating whethe the shares were acquired or disposed of `<transactionAcquiredDisposedCode>` and `</tranactionAcquireDisposedCode>` tags.
* **SharesAfter**: the number of shares owned following the transaction, which appears between `<postTransactionAmounts>` and `</postTransactionAmounts>` using opening and closing `sharesOwnedFollowingTransaction` codes.
* **DIOwner**: a code (I or D) indicating whether the ownership involved is indirect or direct, which appears between `<ownershipNature>` and `</ownershipNature>` tages using opening and closing `directOrIndirectOwnership` tags.

The output of your function should be an N $\times 12$ pandas data frame where N is the number of non derivative transactions found in all of the ownership documents.

Part 1 requires 2 submissions:

- **Part 1A:** a Jupyter notebook in which you are to provide your CreateDataFrame function code.
- **Part 1B:** a csv file obtained by writing the data frame produced by the function to a file using the `to_csv(...,index=False)` data frame method

Part 2:

For Part 2 of the assignment, you are tasked with doing various things with the data frame from Part 1. It is **strongly recommended** that you begin working on Part 2 as soon as you have finished with Part 1. Once the correct version data frame is released it should be easy to work on that even if you started with you own version. This part will require that you put code in multiple cells in a **Jupyter notebook provided in Canvas** and upload the notebook.

Part 3: For Part 3 of the assignment, you will be sent an email with a link to two comma delimited files related to Form 4 filings: a training dataset and a test dataset. Your dataset is the only one you should look at. It is different from the dataset of other students and

- you should not share data with other students, and
- you should not discuss with other students how you made your predictions.

Here is a description of the datasets:

- The training dataset has the following variables included:
  - `TRANS_DATE`: date ranging from 1/1/2013 through 9/29/2013 with 500 dates missing
  - `ASHARES`: total number of shares reported as acquired on the `TRANS_DATE`
  - `TRANS_PRICEPERSHARE`: average price of shares acquired or disposed of on the `TRANS_DATE`
  - DSHARES: total number of shares reported as disposed of on the `TRANS_DATE`
- The test dataset has data for the 500 dates missing in the training datase and the same variables except that `DSHARES` has been removed

Your task in this part is to

- use the training dataset to build a model for predicting the variable `DSHARES` using the other available variables
- use your prediction model to predict the `DSHARES` variable for all 500 observations in the test dataset.
- predict the performance of your predictions

Prediction criteria

- If $\widehat{DSHARES}_i$ denotes your predicted value of $DSHARES_i$ then the quality of your `DSHARES` predictions will be evaluated based on the mean absolute error of your **log** predictions, i.e. you should aim to minimize

$$M = \frac{1}{500}\sum_{i=1}^{500} |\log(1 + DSHARES_i) - \log(1 + \widehat{DSHARES}_i)|$$

- To predict the performance of your predictions, you are asked to provide an estimate of M

How these datasets were produced

For each student, I started with data for a random set of companies (the companies are unique to each student and can exhibit different behaviors from dataset to dataset) and I compiled the data by date based on filings for those companies. I randomly selected 500 dates to remove to create the test data (dates unique to each student). So I am in possession of the actual value of DSHARES associated with dates in your test dataset. Consequently, I will be able to determine the value of M you are trying to estimate. **IMPORTANT: Due to the nature of the datasets, it is highly unlikely that a model fitted on one particular student's dataset will produce good predictions on another student's dataset.**

Part 3 Submission

This part requres 2 items for submission:

- **Part 3A** a comma delimited file with two columns, a heading with `TRANS_DATE` and `DSHARES`, and 500 rows of predictions - the `TRANS_DATE` column should contain the same dates as the ones in your test dataset
- **Part 3B** a Jupyter notebook (provided in Canvas) with the code with all of the work you did to get answers in part 3 - a cell will be provided for you to report your prediction of M.