**Large data set stats & profiling**

**Overview:**

A large data set contains over 10 years of price history of 4000+ securities. However, the data set has gaps. We want to write an efficient Python script to profile the data set to create the gap statistics.

**Input file:**

A Pandas data frame is saved as Python pickle file (filename: px.xz). You can download it at the following link.

<https://drive.google.com/open?id=1XVeOoBXMr9LXIlyxeIqolM80JbXioGBW>

It contains the large data set. Two columns are present in the data frame.

* “dt”: Date
* “bbgid”: security identifier (composite FIGI)

**Helpful hints:**

* **Multiple data gaps exist for the same security**. We want to detect all of them.
* **We only care about the data gaps between the earliest date and the latest date** of a security whose data are present in the given data set.
  + Each security has its own earliest and latest record date.
    - If a security become publicly traded (i.e., IPO’ed) after the earliest date in the data set, then it is OK for the data to be missing before the IPO (since there are no data prior).
    - If a security is delisted before the latest date in the data set, then it is OK for the data to be missing after the latest available record date of that security.
* **Please write the python script in a way that can minimize the processing time**

**Reference output:**

You can view the reference output at the URL below. It’s an Excel spreadsheet.

<https://drive.google.com/open?id=1cPZ8uNZriM5PlpVvhAuD7gZkv1eD-oHK>

The reference output has the following columns

* “start”: start date of a data gap
* “end”: end date of a data gap
* “bbgid”: security identifier
* “length”: number of days of the data gap

The reference output should be sorted by the following fields:

* Length: descending
* Bbgid: ascending
* Start: ascending

Reference output only needs to show the first 1,000 records.

**Requirement:**

* Write a Python 3 script to re-produce the reference output.
* Execution time is key. Less than 10 mins is preferred.
  + Our reference solution takes about 75 seconds to complete on an Intel i7-5557U mobile CPU PC running Windows 10.
* Open book: You are welcome to search for help online.
* Must complete the task independently.
* Coding style matters. Please comment your code with clear intention.

**Extra credit:**

**Sample code skeleton:**

**# load modules**

**import pandas as pd**

**import time**

**# log start time**

**start\_time = time.time()**

**# load input data set (Python pickle file)**

**df = pd.read\_pickle(r'<path>\px.xz') # replace <path> with proper file path**

**# USER CODE**

**…**

**# export result to Excel**

**stats.iloc[0:1000].to\_excel(r'<path>\px\_stats.xlsx') # replace <path> with proper file path**

**# show execution time**

**print("--- %s seconds ---" % (time.time() - start\_time))**