CSE 4334/5334 Programming Assignment P3

Fall 2018

Due: 11:59pm Central Time, Friday, December 14, 2018

The instructions of this assignment are written in an .ipynb file. You can use the following commands to install the Jupyter notebook viewer. "pip" is a command for installing Python packages. You are required to use "Python 3.6.x" (any version of Python equal to or greater than version Python 3.6.0) in this project.

```
pip install jupyter
```

To run the Jupyter notebook viewer, use the following command:

```
jupyter notebook P3.ipynb
```

The above command will start a webservice at http://localhost:8888/ (http://localhost:8888/) and display the instructions in the '.ipynb' file.

Requirements

- This assignment must be done individually. You must implement the whole assignment by yourself. Academic dishonety will have serious consequences.
- You can discuss topics related to the assignment with your fellow students. But you are not allowed to discuss/share your solution and code.

Assignment Files

All the files for this assignment can be downloaded from Blackboard ("Course Materials" > "Programming Assignments" > "Programming Assignment 3 (P3)" > "Attached Files").

1. This instruction file itself "P3.ipynb"
2. Data file "1991-2004-nba.dat"
3. A skeleton of code "P3_skeleton.py"
4. Grading rubrics "rubrics_P3.txt"

Programming Language

1. We will test your code under the particular version of Python 3.6.x. So make sure you develop your code using the same version.
2. You are free to use anything from the Python Standard Library that comes with Python 3.6.x (https://docs.python.org/3.6/library/ (https://docs.python.org/3.6/library/)).
3. **Other than the Python Standard Library, you are NOT allowed to use any non-standard Python package.**

Finding Prominent Streaks in NBA Boxscores

1. Description of Task

You code should accomplish the following tasks:

(1) **Read** the text file 1991-2004-nba.dat. In this file, each line is the boxscore of an NBA player in an NBA game. The records are already sorted by players and then by dates, i.e., all the boxscores of a player are in consecutive rows, in chronological order.

You only need to use the first column (player ID) and the last column (number of points by the player in a game) of this file.

We provide a skeleton file "P3_skeleton.py" to you. It already has the code for reading from the data file.

(2) **Compute** the prominent streaks in this dataset. For each player, the sequence has all the points the player made in his games. There are multiple sequences, one for each player. Each prominent streak is a sub-sequence of a player's sequence. It must not be dominated by any other streak in any player's sequence.

For the concepts and algorithms of prominent streaks, refer to our lecture vidoes, slides, as well as the following two papers.

Xiao Jiang, Chengkai Li, Ping Luo, Min Wang, and Yong Yu. Prominent Streak Discovery in Sequence Data. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 1280-1288, San Diego, California, USA, August 2011. (http://ranger.uta.edu/~cli/pubs/2011/streak-kdd11-jllwy-jun11.pdf (http://ranger.uta.edu/~cli/pubs/2011/streak-kdd11-jllwy-jun11.pdf))

Gensheng Zhang, Xiao Jiang, Ping Luo, Min Wang, and Chengkai Li. Discovering General Prominent Streaks in Sequence Data. In ACM Transactions on Knowledge Discovery from Data (TKDD), 8(2): 9:1-9:37, June 2014. (http://ranger.uta.edu/~cli/pubs/2014/streak-tkdd-jllwy-sept13.pdf (http://ranger.uta.edu/~cli/pubs/2014/streak-tkdd-jllwy-sept13.pdf))

2. Expected Output

Your code should return the following results. Each prominent streak should be displayed in the format of (player ID, beginning index of the streak, length of the streak, minimum value in the streak). The beginning index provides the position of the left end of the streak. In a player's seequence, the index of the first element is 0.

Note that the prominent streaks in the output can be in any order, depending on the particular way an implementation finds the prominent streaks.

In a typical run, our solution code finishes computing prominent streaks in less than half a second on our computer. (Reading the data file itself takes 20-30 seconds.) When we test your code on the same computer, your code is expected to achieve the same efficiency, in order to be considered efficiently implemented. Note that an implementation of the brute-force baseline method took several mminutes to finish.

In [ ]:

```
Reading the data file takes  22.284526109695435  seconds.
Computing prominent streaks takes  0.49631690979003906  seconds.
[('BRYANKO01', 457, 4, 42), ('BRYANKO01', 457, 9, 40), ('BRYANKO01', 453
, 13, 35), ('BRYANKO01', 453, 14, 32), ('BRYANKO01', 453, 16, 30), ('IVE
RSAL01', 305, 27, 26), ('IVERSAL01', 554, 2, 51), ('IVERSAL01', 550, 45,
 21), ('IVERSAL01', 550, 57, 20), ('JACKSJI01', 0, 1185, 0), ('JAMISAN0
1', 107, 2, 51), ('JORDAMI01', 196, 17, 27), ('MALONKA01', 176, 159, 14
), ('MALONKA01', 72, 263, 13), ('MALONKA01', 72, 357, 12), ('MALONKA01',
 482, 96, 17), ('MALONKA01', 459, 119, 16), ('MALONKA01', 430, 149, 15),
 ('MALONKA01', 72, 527, 11), ('MALONKA01', 24, 575, 10), ('MALONKA01', 2
4, 758, 7), ('MALONKA01', 0, 866, 2), ('MCGRATR01', 380, 34, 24), ('ONEA
SH01', 373, 74, 19), ('ONEASH01', 353, 94, 18), ('ONEASH01', 0, 858, 6),
 ('ROBINDA01', 229, 1, 71), ('STOCKJO01', 0, 932, 1)]
```

3. What to Submit

You are required to submit a single .py file of your code. You are expected to use the code in P3_skeleton.py. You algorithm should be in function ''prominent_streaks(sequences)''. You have the freedom to define any other functions that you deem necessary. You shouldn't change any existing code in the file.

4. Grading Rubrics

Your program will be evaluated on correctness, efficiency, and code quality.

Make sure to thoroughly understand the grading rubrics in file "rubrics_P3.txt".

In [ ]: