STATS 607

ASSIGNMENT 2

PART I

The first part of this assignment involves reading a network traffic log file and conducting a small analysis on it. The raw file is in pcap format, which is a standard format for storing tcp transaction data. These logs are used (among other purposes) to identify network attacks such as denial of service (DOS) attacks. The diversity of IP addresses sending traffic over a short period of time is one indicator of whether an attack might be underway.

The source data file we will use for this exercise (maccdc2012_00016.pcap.gz) can be obtained here (this is optional). To convert the uncompressed pcap file to text you could run the following command on a Unix like shell: 'tcpdump -nnr [your_file_name] > [output_file_name]', but I have done that work for you. You will be working directly with the file 'maccdc2012_00016.txt.gz'.

The analysis should be based on a subset of the lines in the file that correspond to events of interest to us. The format for the lines of interest is:

[time] IP [address] > [address]: …

The "…" represents additional information that we will not use. You can count both IP addresses as contributing to the count during the minute that the transaction took place.

Bonus points will be given if you use and explain the use of a generator function to solve this problem. You can remind yourself about iterators here (very related) and read about generator functions here.

Questions:

1.1 Write a script to calculate the number of distinct IP addresses appearing within each minute. (Tip: use the 'gzip' module for the purpose).

1.2 Calculate the 10th, 25th, 75th, and 90th percentiles of the values obtained in 1.

1.3 Calculate the mean number of distinct times that each IP address appears within a minute of log data.

1.4 Calculate the 10th, 25th, 75th, and 90th percentiles of the values obtained in 3.

1.5 (Optional) If you have used a generator function to solve the above, explain what you think the advantages are in using it. Submit your response in a separate text file named 'generation_explanation.txt'.

You should complete this part of the assignment using only core Python and the standard library (do not use Numpy or Pandas).

PART II

The FDA Adverse Event Reporting System (FAERS) is a database that contains adverse event reports, medication error reports and product quality complaints resulting in adverse events that were submitted to FDA. Healthcare professionals, consumers, and manufacturers submit reports to FAERS. FDA receives voluntary reports directly from healthcare professionals (such as physicians, pharmacists, nurses and others) and consumers (such as patients, family members, lawyers and others). Healthcare professionals and consumers may also report to the products' manufacturers. If a manufacturer receives a report from a healthcare professional or consumer, it is required to send the report to FDA as specified by regulations. See more about this here. FAERS data is available to the public and can be accessed via the FAERS dashboard. I've used this dashboard to collect seriousness of the effect reports from 1968 to 2018 ('adverseCountsFinal.txt'). The four columns of the data correspond to 'Year', 'Serious', 'Death', 'Non-serious' reports, respectively. 'Serious' indicates that one or more of the following outcomes, excluding death, were documented in the report: hospitalization, life-threatening, disability, congenital anomaly, required intervention, and/or serious outcome. 'Non-Serious' is used for outcomes which were not documented as 'Serious' or 'Death'.

Questions:

2.1 Load the text file into a Numpy array data structure.

2.2 Since 1968, how many 'Serious', 'Death' and 'Non-Serious' side effects have been reported?

2.3 How many side effects have been reported per year?

2.4 Which year has seen the highest number of reports for each 'Serious', 'Death' and 'Non-Serious' side effect?

2.5 What type of side effect has been reported the most for each year?

2.6 Normalize the side effect counts by the total number of side effects per year. The outcome should be bi-dimensional, but does not necessarily includes the year.

2.7 Use matplotlib to create a 'stackplot' of the data.

You should complete this part of the assignment using Numpy and no explicit for-loops (do not use Pandas). Except for 2.7, the outcome of all these questions should be a Numpy array data structure.

Make sure you have the provided files ('maccdc2012_00016.txt.gz', 'adverseCountsFinal.txt') for the assignment in your current working directory before starting to solve the problems. Your main scripts (one for each part of this assignment) should be called 'test_assignment2_part[1 or 2]_[your name].py'. Remember, your scripts should properly demonstrate outcomes.

Use exceptions and create your own functions for all parts of this assignment as you see fit (you should need it). Define the functions in a separate module (file) named 'assignment2_[your name].py'. For example, my module containing function definitions would be called 'assignment2_Marcio.py'. Your code, obviously including functions, should be well documented. For functions, state the input and output variables as well as their type. For example, see the following docstring:

```
“””
Input:
  a: np.array
  b: string

Output:
  out: tuple
“””
```

Note that your final solution should consist of two or three .py files and one optional text file: one main script for each of the parts of the assignment; one potential module file you created with the definition of your functions; and one optional text file with the explanation for the use of a generator function in Part I of this assignment.

You have until October 7th to upload your solutions via canvas.

Good Luck!