HW 1

CS 216, Everything Data, Spring 2020

DUE: Monday Jan. 27 by 4:40 pm (class time)

This homework assignment is in five parts. First, you will get some hands on experience and tutorials with regular expressions, Pandas, and OpenRefine. We strongly reccomend that you go through these exercises and tutorials in Parts 1 through 3 first before moving on to the problems in Parts 4 and 5. You can include all of your answers (either text written in Markdown cells or code written in code cells) within this notebook. You will then convert your notebook to a .pdf and a .py file to submit to gradescope (submission instructions are included at the bottom).

Please take note of the course collaboration policy (https://sites.duke.edu/compsci216s2020/policies/). You may work alone or with a single partner. If you work with a partner, you may not split up the assignment; you should work together in-person or complete parts independently and come together to discuss your solutions. In either case, you are individually responsible for your work, and should understand everything in your submission.

Name and NetID

Write your name and netID (along with your partner if you have one) in the Markdown cell below.

Eugene Kim (yk171), Nathan Kim (njk24)

Part 1: Regular Expressions Practice

As discussed in class, regular expressions are a formal language for specifying strings of characters. That can be very helpful when wrangling and cleaning data, so you should start by practicing to get some basic familiarity with regular expressions. Review the materials from lecture, then go to regex101.com (https://regex101.com). You should see an interactive console that allows you to practice regular expressions (feel free to switch to the Python flavor in the bottom left if you prefer; note that the regular expressions themselves are the same regardless).

Play around with the console until you are comfortable, then go to the quiz and complete the first six tasks. In each, you will be asked to write a regular expression. You can submit and get feedback until you get it right.

What to turn in

For part 1, write the six regular expressions that solve the six tasks in the quiz. Write your answers in the markdown cell below. Don't worry about how short your expressions are, just try to solve the tasks. Please clearly number and delineate the answers to different tasks.

Part 1 Answer

1. /\b[Ww][Oo][Rr][Dd]\b/ 2. s/\bi\b/I/g 3./[BCDFGHJKLMNPQRSTVWXYZ]/g 4./[0-9]+/g 5./\s{4,}/g 6.s/(.)\1{2}/\1/g

Part 2: Pandas Tutorial

As discussed in class, Pandas is a powerful library for dealing with structured data in Python, and you can handle a lot of data wrangling and cleaning using Pandas data structures and Python operations. This tutorial is intended as a brief introduction to using Pandas.

1. First, go to pandas.pydata.org/pandas-docs/stable (https://pandas.pydata.org/pandas-docs/stable/index.html) and read the section titled 10 minutes to Pandas. Note that this page also gives the official documentation for any other functions in Pandas you might need to use.
2. (optional) If you would like to use Pandas instead of OpenRefine for the later parts of the assignment, you may want to read more about how to use it to wrangle and clean data. You can check out chapter 4 on Pandas from Python Data Science Handbook (https://jakevdp.github.io/PythonDataScienceHandbook/), which comes with Jupyter notebooks to follow along in examples. For an even shorter and more specific tutorial, try this data cleaning article (https://realpython.com/python-data-cleaning-numpy-pandas/), which also has a linked Jupyter notebook where you can follow along. If you prefer to use a graphical interface for wrangling and cleaning data in this assignment, feel free to skip this second step and go straight to the OpenRefine tutorial in Part 3.

What to turn in

Nothing to submit for part 2.

Part 3: OpenRefine Tutorial

Data often comes along messy. Either people make mistakes entering and collecting data, or the data you got is in the wrong format for what you want to do with it. OpenRefine (http://openrefine.org/) is a powerful data wrangling tool that was built to deal with these kinds of problems and to bring data into the shape you need, all inside of a graphical user interface designed to make it more accessible. For completing the subsequent parts of the assignment, you may wish to use OpenRefine instead of Python and Pandas. We particularly reccomend that you use OpenRefine if you are still getting used to programming in Python. Regardless, in this part you should try out OpenRefine to add an additional tool to your data cleaning toolbox.

1. First, visit the OpenRefine website and watch the short videos on exploring data and cleaning & transforming data. These will introduce you to the OpenRefine design and functionality.
2. Next, install OpenRefine. Visit http://openrefine.org/download.html (http://openrefine.org/download.html) and download the kit for your operating system.
3. Finally, complete the short step by step OpenRefine tutorial included in the box folder for this assignment (just open the .html file to view the tutorial in your web browser). (This tutorial originally came from http://unurl.org/cbclean (http://unurl.org/cbclean). It has been cleaned up and modified a bit.)
4. (optional) If you would like even more tutorials on using OpenRefine, feel free to browse the extensive curated list of tutorials maintained by the developers: https://github.com/OpenRefine/OpenRefine/wiki/External-Resources (https://github.com/OpenRefine/OpenRefine/wiki/External-Resources).

What to turn in

Nothing to submit for part 3.

Part 4: Cleaning up system logs

Now that you have experienced regular expressions, Pandas, and OpenRefine, it's time to use your tools to clean some data. In this part you will use OpenRefine or Python (your choice) to clean and analyze system logs. You can find a csv of the data inside the box folder for this homework assignment; it is titled 'monitor.csv'. Answer the following questions:

1. Identify at least two problems with this dataset. For each, explain intuitively (in English, not code) what you will do in data cleaning to fix the problem and make the dataset more usable.
2. What is the average and median running time of the program? Provide the code you use to clean the dataset and answer the question. If you are using OpenRefine, simply write the sequence of steps you took in OpenRefine explicitly.
3. What is the average and median memory of the program? Provide the code you use to clean the dataset and answer the question. If you are using OpenRefine, simply write the sequence of steps you took in OpenRefine explicitly.

What to turn in

Type your answer to each of the three enumerated questions above. Write in the markdown cell below, but feel free to add additional cells (Markdown or Code) as needed to solve the problem. Please clearly number and delineate the answers to different questions.

1. The first thing we did was go to each column, press edit cells, then common transforms, and made sure that there was not trailing white spaces. The "System Mem: 3Gb" column had all cells with these spaces. The next problem is the names of the column "System Time 23 second", "System Mem: 3Gb" and "414Mb". This should not be specific to a value(It was a specific value so when we opened the project we checked the "column names" box which added row "0" to the data).The names should be changed to "System Time", "System Memory" and "Mb". And then in each of the cells we would take out the starting line "System Time:" and "System Mem:" and "Mb" to clean up the data. Another thing we would do is use the toNumber function to convert the cells of the "Mb" to numbers. Another problem is the system times. Some of them are given by seconds or minutes. we would convert all of them from seconds to minutes. 2. In order to change all of the system times to seconds, we filtered the cells with minutes by clicking edit cells in the "System Time" column, transform, then using toNumber to make them a number and then multiplied them by 60. Then we used replace(" second","") and replace(" minute","") to get rid of the units in the data. Now all the times are in seconds. Then we exported the data into a CSV to use pandas and numpy. Below you can see that we read the file and used the mean and median function. The mean is 1172.581 seconds and the median is 1380 seconds. 3. We made an assumption that the Mb column should be added to the Gb column. First we replaced both columns to remove the "Gb" and "Mb" units and made them into numbers using .toNumber. From here, we did "Edit column"->"Join columns...", selecting the system mem and Mb columns from our data. We also included a separator of "." in order to replicate the format of common data displays. At this point, the two columns have been combined and we ran the .toNumber function once again to have a final, numerical memory column. We then used the same procedure as question 2 and the code is below. The median is 2.505Gb and the mean is 2.87Gb.

In [1]:
```
import numpy as np
```

In [2]:
```
import pandas as pd
```

In [3]:
```
mn = pd.read_csv("C:/Users/ek99k/Downloads/monitor-csv (1).csv")
```

In [4]:
```
mn['System Time (seconds)'].agg('median')
```

Out[4]: 1380.0

In [5]:
```
mn['System Time (seconds)'].agg('mean')
```

Out[5]: 1172.581

In [6]:
```
mn['System Memory'].agg('mean')
```

Out[6]: 2.8698389999999994

In [7]:
```
mn['System Memory'].agg('median')
```

Out[7]: 2.505

Part 4 Answer

Part 5: Wrangling Congressional Member Listing

govtrack.us (https://www.google.com/url?q=http://www.govtrack.us/&sa=D&ust=1516309781268000&usg=AFQjCNHmEaudoYkhtS2Du47GafoIeXdJYw) contains a wealth of data on the U.S. Congress (that is, both the house and senate). Along with partners, they curate a publicly available dataset containing, among other things, information on all current members of the U.S. Congress: https://github.com/unitedstates/congress-legislators (https://github.com/unitedstates/congress-legislators). Pull the legislators-current file from the repository. It is available in different formats: the master branch maintains it as a .yaml file, but if you scroll down it is also available as a .json or .csv; the choice is up to you. Using this dataset (and no other sources; you will not get credit for pulling stats from online without showing any work), answer the following questions.

For each question, provide the code you use to answer the questions. If you are using OpenRefine, simply write the sequence of steps you took in OpenRefine explicitly.

1. What percentage of the current U.S. congress is identified as female?
2. What percentage of current members were at least 60 years old as of January 16, 2020?
3. Members of the U.S. Congress are either serving a term in the senate or the house. What is the average age of members of the senate? What is the average age of members of the house?
4. For each possible party affiliation (R for Republicans, D for Democrats, and I for independents), list the number of members with that affiliation.

What to turn in

Type your answer to each of the four enumerated questions above. Be sure to provide the code (or OpenRefine instructions) you use. Write in the markdown cell below, but feel free to add additional cells (Markdown or Code) as needed to solve the problem. Please clearly number and delineate the answers to different questions.

1. First we sorted the gender column alphabetically. Since "F" is before "M", we went to the last cell with "F" to find the count of females. Then we divided that by the total number of congressmen. There were 131 females out of 536 congressmen. That is 24.4% off congress.
2. We first unsorted the gender column. Then we sorted the birthday column through sort->text. This sorted the data by birthday. They have to be born before January 16, 1960. We went to the last birthday before january 16, 1960 which was January 11, 1960. This was in row 269 so we calculated 269/536 = 50.2%.
3. The average age of the Senate is 64.23 and the average age of the House of Representations is 58.96. We found this by first finding all the ages of all congressmen by subtracting their birthyear from 2020. We found their birthyear by indexing the first 4 characters of their birthday and then using the .toNumber() function. From there, we filtered out two csv files based on whether the congressman was in the House of Representatives or the Senate. For each file, we calculated the mean of eac groups' respective ages by using the aggregate and mean functions.
4. We simply used a text filter for the "party" column and found that there are 281 Democrats, 252 Republicans and 3 independent.

In [8]:
```
mn1 = pd.read_csv("C:/Users/ek99k/Downloads/legislatorscurrentcsvHW1 (2).csv")
```

In [9]:
```
pd.read_csv("C:/Users/ek99k/Downloads/legislatorscurrentcsvHW1 (2).csv")
```

Out[9]:

	last_name	first_name	middle_name	suffix	nickname	full_name	birthday	BirthdayDay	B
0	Brown	Sherrod	NaN	NaN	NaN	Sherrod Brown	1952-11-09	9	
1	Cantwell	Maria	NaN	NaN	NaN	Maria Cantwell	1958-10-13	13	
2	Cardin	Benjamin	L.	NaN	NaN	Benjamin L. Cardin	1943-10-05	5	
3	Carper	Thomas	Richard	NaN	NaN	Thomas R. Carper	1947-01-23	23	
4	Casey	Robert	P.	Jr.	Bob	Robert P. Casey, Jr.	1960-04-13	13	
...	...	...	...	...	...	...	...	...	
531	Golden	Jared	Forrest	NaN	NaN	Jared F. Golden	1982-07-25	25	
532	Keller	Fred	NaN	NaN	NaN	Fred Keller	1965-10-23	23	
533	Bishop	Dan	NaN	NaN	NaN	Dan Bishop	1964-07-01	1	
534	Murphy	Gregory	Francis	NaN	NaN	Gregory F. Murphy	1963-03-05	5	
535	Loeffler	Kelly	NaN	NaN	NaN	Kelly Loeffler	1970-11-27	27	

536 rows × 37 columns

In [10]:
```
mn2 = pd.read_csv("C:/Users/ek99k/Downloads/legislatorscurrentcsvHW1 (3).csv")
```

In [11]:
```
pd.read_csv("C:/Users/ek99k/Downloads/legislatorscurrentcsvHW1 (3).csv")
```

Out[11]:

	last_name	first_name	middle_name	suffix	nickname	full_name	birthday	Age	BirthdayDa
0	Brown	Sherrod	NaN	NaN	NaN	Sherrod Brown	1952-11-09	68	
1	Cantwell	Maria	NaN	NaN	NaN	Maria Cantwell	1958-10-13	62	1
2	Cardin	Benjamin	L.	NaN	NaN	Benjamin L. Cardin	1943-10-05	77	
3	Carper	Thomas	Richard	NaN	NaN	Thomas R. Carper	1947-01-23	73	2
4	Casey	Robert	P.	Jr.	Bob	Robert P. Casey, Jr.	1960-04-13	60	1
...	...	...	...	...	...	...	...	...	
95	Braun	Mike	NaN	NaN	NaN	Mike Braun	1954-03-24	66	2
96	Hawley	Joshua	NaN	NaN	NaN	Josh Hawley	1979-12-31	41	3
97	Romney	Mitt	NaN	NaN	NaN	Mitt Romney	1947-03-12	73	1
98	McSally	Martha	NaN	NaN	NaN	Martha McSally	1966-03-22	54	2
99	Loeffler	Kelly	NaN	NaN	NaN	Kelly Loeffler	1970-11-27	50	2

100 rows × 38 columns

In [12]:
```
mn2['Age'].agg('mean')
```

Out[12]:
```
64.23
```

In [13]:
```
mn3 = pd.read_csv("C:/Users/ek99k/Downloads/legislatorscurrentcsvHW1 (4).csv")
```

In [14]:
```
pd.read_csv("C:/Users/ek99k/Downloads/legislatorscurrentcsvHW1 (4).csv")
```

Out[14]:

	last_name	first_name	middle_name	suffix	nickname	full_name	birthday	Age	BirthdayD
0	Aderholt	Robert	B.	NaN	NaN	Robert B. Aderholt	1965-07-22	55	
1	Amash	Justin	NaN	NaN	NaN	Justin Amash	1980-04-18	40	
2	Bass	Karen	NaN	NaN	NaN	Karen Bass	1953-10-03	67	
3	Bilirakis	Gus	M.	NaN	NaN	Gus M. Bilirakis	1963-02-08	57	
4	Bishop	Rob	NaN	NaN	NaN	Rob Bishop	1951-07-13	69	
...	...	...	...	...	...	...	...	...	
431	Miller	Carol	D.	NaN	NaN	Carol D. Miller	1950-11-04	70	
432	Golden	Jared	Forrest	NaN	NaN	Jared F. Golden	1982-07-25	38	
433	Keller	Fred	NaN	NaN	NaN	Fred Keller	1965-10-23	55	
434	Bishop	Dan	NaN	NaN	NaN	Dan Bishop	1964-07-01	56	
435	Murphy	Gregory	Francis	NaN	NaN	Gregory F. Murphy	1963-03-05	57	

436 rows × 38 columns

In [15]:
```
mn3['Age'].agg('mean')
```

Out[15]:
```
58.96330275229358
```

Submitting HW 1

1. Double check that you have written all of your answers along with your supporting work in this notebook. Make sure you save the complete notebook.
2. Double check that your entire notebook runs correctly and generates the expected output. To do so, you can simply select Kernel -> Restart and Run All.
3. You will download two versions of your notebook to submit, a .pdf and a .py. To create a PDF, we reccomend that you select File --> Download as --> HTML (.html). Open the downloaded .html file; it should open in your web broser. Double check that it looks like your notebook, then print a .pdf using your web browser (you should be able to select to print to a pdf on most major web browsers and operating systems). Check your .pdf for readability: If some long cells are being cut off, go back to your notebook and split them into multiple smaller cells. To get the .py file from your notebook, simply select File -> Download as -> Python (.py).
4. Upload the .pdf to gradescope under hw1 report and the .py to gradescope under hw1 code. If you work with a partner, only submit one document for both of you, but be sure to add your partner using the group feature on gradescope (https://www.gradescope.com/help#help-center-item-student-group-members).