Introduction to Big Data - Practice 4 (Data Manipulation II)

For each question, make sure you not only just “write down” the Python codes but also “explain the answer with your own language”. All answers without explanation will not be accepted.

Problem

< Question 1 – User Consumption data>

Import ‘UserConsumption.csv’ to a data frame variable called *UserConsump*. This data is about user’s OTT consumption in 2019. Here, the users are classified in one of three groups, Low (0), Medium (1) and High Consumption (2). Write down Python codes that corresponds to the following questions:

(1-1, 5 pts) Explore the data set and answer the following question. 1) What is the number of rows and columns? 2) What does each column stand for? Explain.

(1-2, 5 pts) Which variable can be used as an identifier? Write down Python codes that support your answer.

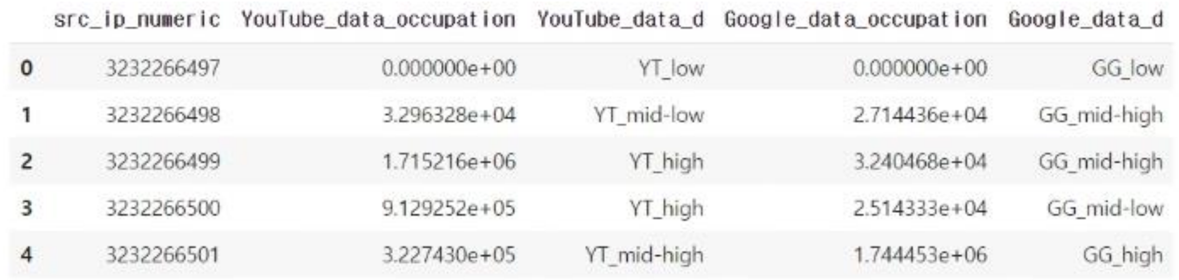
(1-3, 5 pts) Are there any missing values? Which variable can be used as an identifier? Write down Python codes that support your answer.

(1-4, 10 pts) There are so many zeros in this data set. Write down Python codes that count the total number of zeros. Do you think those zeros are acceptable? Explain.

(1-5, 15 pts) We want to check the frequency of Google and Youtube’s data consumption. Here we want to see four categories of low (0%<= x <= 25%), mid-low (25% < x <= 50%), mid-high (50% < x <= 75%), and high (75% < x <=100%). Create new variable called *UserConsump_ed* (which is the same to *UserConsump*) and write down Python codes that produces the new columns called “YouTube_data_d” and “Google_data_d” (Hint: use numpy.percentile() & pandas.cut()). The expected result is shown below.

Expected Result

```
>>> UserConsump_ed[['src_ip_numeric','YouTube_data_occupation', 'YouTube_data_d',
'Google_data_occupation', 'Google_data_d']].head()
```

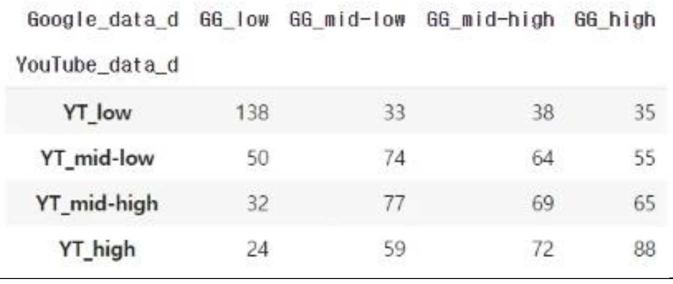


```
   src_ip_numeric  YouTube_data_occupation YouTube_data_d  Google_data_occupation Google_data_d
0      3232266497             0.000000e+00         YT_low            0.000000e+00        GG_low
1      3232266498             3.296328e+04     YT_mid-low            2.714436e+04   GG_mid-high
2      3232266499             1.715216e+06        YT_high            3.240468e+04   GG_mid-high
3      3232266500             9.129252e+05        YT_high            2.514333e+04    GG_mid-low
4      3232266501             3.227430e+05    YT_mid-high            1.744453e+06       GG_high
```

(1-6, 10 pts) The new variables that we created from (1-2), “YouTube_data_d” and “Google_data_d”, can tell us the user’s YouTube and Google data consumption type. With these two variables, a frequency table (or pair matrix) can be used to see the relationship between two. Write down Python codes that produce this table and explain the meaning of the numbers and what you can notice from this. The expected result is shown below (Hint: use pandas.crosstab).

Expected Result

```
>>> ???
```

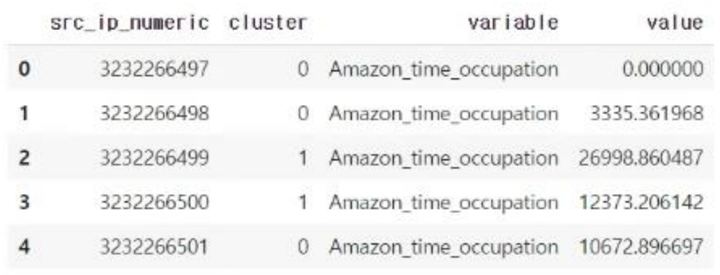


```
                Google_data_d  GG_low  GG_mid-low  GG_mid-high  GG_high
YouTube_data_d
        YT_low                    138          33           38       35
    YT_mid-low                     50          74           64       55
   YT_mid-high                     32          77           69       65
       YT_high                     24          59           72       88
```

(1-7, 10 pts) As you noticed, the current data format of *UserConsump_ed* is "wide". Convert this into a long format and create a variable called *UserConsump_long* as shown below. As we no longer need the new variables, "YouTube_data_d" and "Google_data_d", remove them.

Expected Result

```
>>> UserConsump_long.head()
```

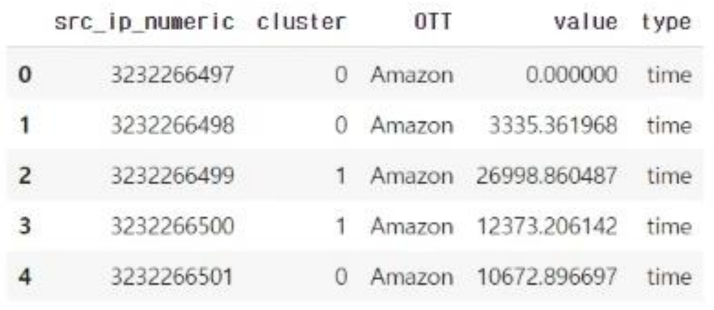


```
   src_ip_numeric  cluster               variable         value
0      3232266497        0  Amazon_time_occupation      0.000000
1      3232266498        0  Amazon_time_occupation   3335.361968
2      3232266499        1  Amazon_time_occupation  26998.860487
3      3232266500        1  Amazon_time_occupation  12373.206142
4      3232266501        0  Amazon_time_occupation  10672.896697
```

(1-8, 15 pts) In *UserConsump_long*, there are many elements included in "variable". Since these elements contain both the name of OTT service and type of occupation, it is difficult for us to separate them. In this respect, do the following task. 1) create a variable called "type", which indicates whether it is about "time" or "data" (Hint: Use numpy.where() & .str.contains()). 2) Convert the values of "variable" to have only the name of OTT (Hint: Use .str.replace()). 3) Rename "variable" to "OTT". Below is the updated *UserConsump_long*.

Expected Result

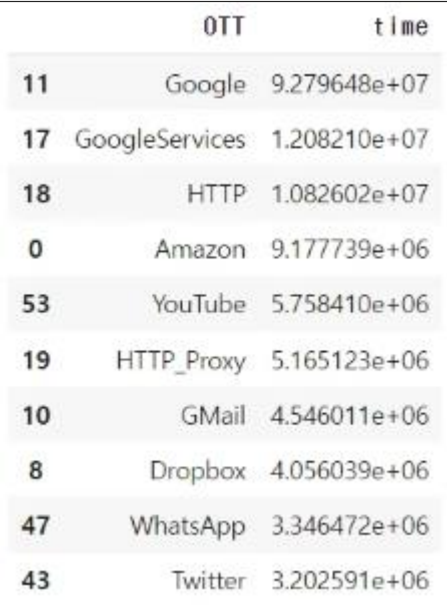
```
>>> UserConsump_long.head()
```



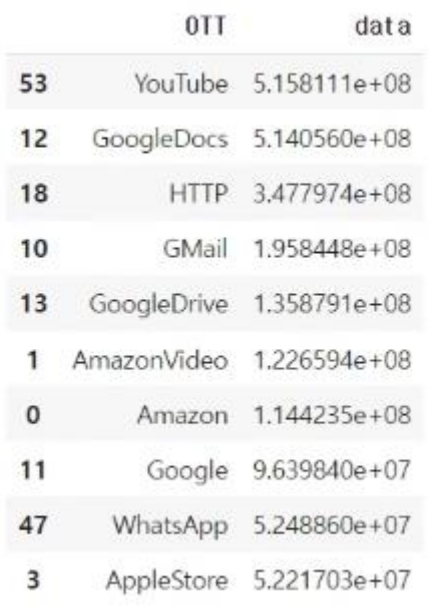
```
   src_ip_numeric  cluster     OTT         value  type
0      3232266497        0  Amazon      0.000000  time
1      3232266498        0  Amazon   3335.361968  time
2      3232266499        1  Amazon  26998.860487  time
3      3232266500        1  Amazon  12373.206142  time
4      3232266501        0  Amazon  10672.896697  time
```

(1-9, 10 pts) Write down Python codes that answer the following questions: 1) What are Top 10 OTT services that users consume the time mostly? 2) What are Top 10 OTT services that users consume the data mostly?

Expected Result	
(1)	(2)



	OTT	time
11	Google	9.279648e+07
17	GoogleServices	1.208210e+07
18	HTTP	1.082602e+07
0	Amazon	9.177739e+06
53	YouTube	5.758410e+06
19	HTTP_Proxy	5.165123e+06
10	GMail	4.546011e+06
8	Dropbox	4.056039e+06
47	WhatsApp	3.346472e+06
43	Twitter	3.202591e+06



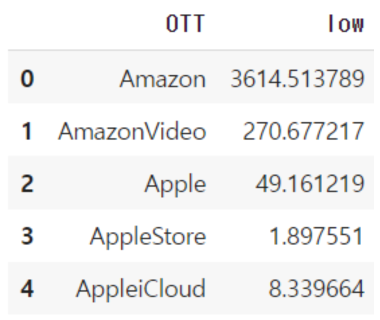
	OTT	data
53	YouTube	5.158111e+08
12	GoogleDocs	5.140560e+08
18	HTTP	3.477974e+08
10	GMail	1.958448e+08
13	GoogleDrive	1.358791e+08
1	AmazonVideo	1.226594e+08
0	Amazon	1.144235e+08
11	Google	9.639840e+07
47	WhatsApp	5.248860e+07
3	AppleStore	5.221703e+07

(1-10, 15 pts) Compare the average time consumption between “low” consumption group and “high” consumption group. To do so, (1) Create a table that shows average time consumption of “low” group (*UserConsump_long_low*) and “high” groups (name it *UserConsump_long_high*). (2) Calculate average the overall average time consumption for low and high groups. Knowing the fact the time is measured in seconds, compare those two with the unit of minutes.

Expected Result

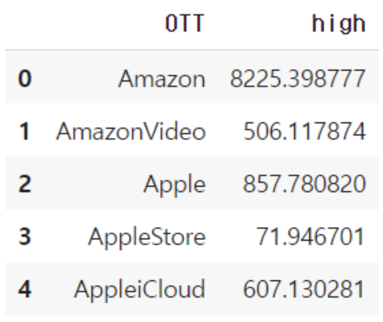
(1)

```
>>> UserConsump_long_low.head()
```



	OTT	low
0	Amazon	3614.513789
1	AmazonVideo	270.677217
2	Apple	49.161219
3	AppleStore	1.897551
4	AppleiCloud	8.339664

```
>>> UserConsump_long_high.head()
```



	OTT	high
0	Amazon	8225.398777
1	AmazonVideo	506.117874
2	Apple	857.780820
3	AppleStore	71.946701
4	AppleiCloud	607.130281

(2)

```
>>> ???
18.064625776703085
>>> ???
38.3012019369619
```