Untitled

February 11, 2022

Name: Mohammad Aziz, Student ID: 20192233, Module: Geospatial Data Analysis

[176]:
```
import pandas
import numpy
```

1 TASK 1

[174]:
```
# Import the dataset, World GDP.csv and store as a Pandas data frame. Perform␣
 ↪some basic operations
# on this data frame such as, reading data, making changes in the data frame,␣
 ↪saving data into desired#
# format, and filtering
```

[91]:
```
df=pandas.read_csv("World GDP.csv")
```

[92]:
```
df=df.dropna()
```

[93]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 115 entries, 7 to 263
Data columns (total 64 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Country Name    115 non-null    object
 1   Country Code    115 non-null    object
 2   Indicator Name  115 non-null    object
 3   Indicator Code  115 non-null    object
 4   1960            115 non-null    float64
 5   1961            115 non-null    float64
 6   1962            115 non-null    float64
 7   1963            115 non-null    float64
 8   1964            115 non-null    float64
 9   1965            115 non-null    float64
 10  1966            115 non-null    float64
 11  1967            115 non-null    float64
 12  1968            115 non-null    float64
```

```
13  1969            115 non-null    float64
14  1970            115 non-null    float64
15  1971            115 non-null    float64
16  1972            115 non-null    float64
17  1973            115 non-null    float64
18  1974            115 non-null    float64
19  1975            115 non-null    float64
20  1976            115 non-null    float64
21  1977            115 non-null    float64
22  1978            115 non-null    float64
23  1979            115 non-null    float64
24  1980            115 non-null    float64
25  1981            115 non-null    float64
26  1982            115 non-null    float64
27  1983            115 non-null    float64
28  1984            115 non-null    float64
29  1985            115 non-null    float64
30  1986            115 non-null    float64
31  1987            115 non-null    float64
32  1988            115 non-null    float64
33  1989            115 non-null    float64
34  1990            115 non-null    float64
35  1991            115 non-null    float64
36  1992            115 non-null    float64
37  1993            115 non-null    float64
38  1994            115 non-null    float64
39  1995            115 non-null    float64
40  1996            115 non-null    float64
41  1997            115 non-null    float64
42  1998            115 non-null    float64
43  1999            115 non-null    float64
44  2000            115 non-null    float64
45  2001            115 non-null    float64
46  2002            115 non-null    float64
47  2003            115 non-null    float64
48  2004            115 non-null    float64
49  2005            115 non-null    float64
50  2006            115 non-null    float64
51  2007            115 non-null    float64
52  2008            115 non-null    float64
53  2009            115 non-null    float64
54  2010            115 non-null    float64
55  2011            115 non-null    float64
56  2012            115 non-null    float64
57  2013            115 non-null    float64
58  2014            115 non-null    float64
59  2015            115 non-null    float64
60  2016            115 non-null    float64
```

```
 61   2017             115 non-null    float64
 62   2018             115 non-null    float64
 63   2019             115 non-null    float64
dtypes: float64(60), object(4)
memory usage: 58.4+ KB
```

[94]:
```
#we don't have any null values
```

[ ]:
```

```

2 TASK 2

[120]:
```
# Choose any five countries of your choice and make a new data frame containing␣
 ↪only five countries
# and their GDP data (from 1990 until 2019).
```

[121]:
```
df_countries=df[df.columns[~df.columns.isin(['Country Code','Indicator Name',␣
 ↪"Indicator Code"])]]
```

[126]:
```
df_5_countries=df_countries.drop(df_countries.iloc[:, 1:31], axis=1).sample(5)
```

[131]:
```
df_5_countries
```

[131]:
```
      Country Name          1990          1991          1992          1993  \
202     South Asia  6.620550e+11  6.743800e+11  7.128640e+11  7.444140e+11
21    Bahamas, The  7.606466e+09  7.288380e+09  7.009557e+09  7.031133e+09
92          Guyana  1.112850e+09  1.180265e+09  1.271829e+09  1.375802e+09
171          Niger  4.119519e+09  4.113445e+09  4.196032e+09  4.214267e+09
231       Thailand  1.416110e+11  1.537300e+11  1.661570e+11  1.798680e+11

              1994          1995          1996          1997          1998  \
202   7.897990e+11  8.450050e+11  9.032710e+11  9.377250e+11  9.905490e+11
21    7.252537e+09  7.570108e+09  7.889834e+09  8.052541e+09  8.432338e+09
92    1.493192e+09  1.568321e+09  1.693089e+09  1.797733e+09  1.767547e+09
171   4.286575e+09  4.384687e+09  4.401116e+09  4.466897e+09  4.907794e+09
231   1.942520e+11  2.100260e+11  2.218970e+11  2.157870e+11  1.993130e+11

      …          2010          2011          2012          2013  \
202   …  2.060780e+12  2.166660e+12  2.285860e+12  2.425020e+12
21    …  1.009576e+10  1.015764e+10  1.047119e+10  1.016207e+10
92    …  2.273225e+09  2.391343e+09  2.517521e+09  2.643843e+09
171   …  7.792421e+09  7.976693e+09  8.822215e+09  9.313235e+09
231   …  3.411050e+11  3.439710e+11  3.688840e+11  3.787970e+11

              2014          2015          2016          2017          2018  \
202   2.594590e+12  2.788670e+12  3.005590e+12  3.210830e+12  3.406640e+12
```

```
21   1.023692e+10  1.029809e+10  1.034405e+10  1.035094e+10  1.051301e+10
92   2.746906e+09  2.830820e+09  2.926027e+09  2.987474e+09  3.109960e+09
171  9.924596e+09  1.035808e+10  1.094731e+10  1.149444e+10  1.229958e+10
231  3.825260e+11  3.945140e+11  4.080430e+11  4.246350e+11  4.422610e+11

             2019
202  3.571270e+12
21   1.070217e+10
92   3.256001e+09
171  1.301647e+10
231  4.527510e+11

[5 rows x 31 columns]
```

```
[ ]:
```

3 TASK 3

```
[136]: # Perform z-score standardisation on the GDP data (1990-2019) of all the five countries.
```

```
[132]: z_score=df_5_countries.drop("Country Name", axis=1)
```

```
[134]: z_score=(z_score-z_score.mean())/z_score.std()
```

```
[135]: z_score
```

```
[135]:         1990      1991      1992      1993      1994      1995      1996  \
202  1.749447  1.743744  1.741479  1.737728  1.735753  1.734525  1.735772
21  -0.546120 -0.554046 -0.558021 -0.562620 -0.564972 -0.566386 -0.565211
92  -0.568897 -0.575086 -0.576713 -0.580263 -0.581905 -0.582876 -0.581136
171 -0.558351 -0.564982 -0.567187 -0.571408 -0.573692 -0.575138 -0.574177
231 -0.076081 -0.049630 -0.039558 -0.023437 -0.015184 -0.010124 -0.015248

        1997      1998      1999  …      2010      2011      2012  \
202  1.742500  1.753830  1.755960  …  1.765176  1.767125  1.766377
21  -0.558051 -0.544154 -0.540951  … -0.531123 -0.527951 -0.529585
92  -0.573529 -0.559749 -0.556599  … -0.539883 -0.536216 -0.537611
171 -0.566924 -0.552401 -0.549910  … -0.533702 -0.530272 -0.531249
231 -0.043995 -0.097526 -0.108499  … -0.160468 -0.172687 -0.167932

        2013      2014      2015      2016      2017      2018      2019
202  1.767821  1.770186  1.771696  1.773082  1.773898  1.774456  1.775146
21  -0.527566 -0.523417 -0.520647 -0.518019 -0.516481 -0.515446 -0.514012
92  -0.534712 -0.530064 -0.526808 -0.523693 -0.521750 -0.520437 -0.518799
171 -0.528373 -0.523694 -0.520598 -0.517557 -0.515663 -0.514241 -0.512524
```

```
231 -0.177169 -0.193012 -0.203643 -0.213814 -0.220004 -0.224332 -0.229810

[5 rows x 30 columns]
```

[ ]:

```

```

4 TASK 4

[137]:

```
# Import the COVID 19 dataset, total_cases.csv as Pandas data frame and print␣
 ↪the bottom five rows
# of the data.
```

[138]:

```
df1=pandas.read_csv("total_cases.csv")
```

[145]:

```
df1.tail(5)
```

[145]:

```
           date     World  Afghanistan  Albania  Algeria  Andorra   Angola  \
325  2020-11-20  57030619      44133.0  30623.0  71652.0   6066.0  13922.0
326  2020-11-21  57710848      44365.0  31459.0  72755.0   6142.0  14134.0
327  2020-11-22  58278092      44519.0  32196.0  73774.0   6207.0  14413.0
328  2020-11-23  58794150      44771.0  32761.0  74862.0   6256.0  14493.0
329  2020-11-24  59307493      45017.0  33556.0  75867.0   6304.0  14493.0

     Anguilla  Antigua and Barbuda  Argentina  …  Uzbekistan  Vanuatu  \
325       3.0                139.0  1349434.0  …     71071.0      1.0
326       3.0                139.0  1359026.0  …     71280.0      1.0
327       3.0                139.0  1366169.0  …     71280.0      1.0
328       4.0                139.0  1370350.0  …     71617.0      1.0
329       4.0                139.0  1374348.0  …     71847.0      1.0

     Vatican  Venezuela  Vietnam  Wallis and Futuna  Western Sahara   Yemen  \
325     26.0    98665.0   1304.0                2.0            766.0  2086.0
326     26.0    98665.0   1305.0                2.0            766.0  2090.0
327     26.0    99017.0   1306.0                2.0            766.0  2093.0
328     26.0    99835.0   1306.0                2.0            766.0  2099.0
329     26.0   100143.0   1312.0                2.0            766.0  2107.0

      Zambia  Zimbabwe
325  17350.0    9046.0
326  17373.0    9120.0
327  17394.0    9172.0
328  17424.0    9220.0
329  17454.0    9308.0

[5 rows x 216 columns]
```

```
[ ]:
```

5 TASK 5

```
[140]: # Print the columns corresponding to the United Kingdom, United States, and␣
        ↪Sweden
```

```
[156]: df1[df1.columns[df1.columns.isin(["United Kingdom", "United States",␣
        ↪"Sweden"])]].sample(5)
```

```
[156]:        Sweden  United Kingdom  United States
       298  110992.0        830998.0      8493669.0
       74      771.0          1766.0         2174.0
       246   84396.0        337168.0      6075652.0
       215   76940.0        303952.0      4620444.0
       121   20169.0        167150.0      1039909.0
```

```
[ ]:
```

6 TASK 6

```
[157]: # Find the first quartile, second quartile, third quartile, and mean of COVID␣
        ↪19 cases for the United
       # Kingdom and the United States. (HINT - You can use an inbuilt function in␣
        ↪Pandas)
```

```
[169]: df1[["United Kingdom", "United States"]].quantile([.25, .50, .75])
```

```
[169]:       United Kingdom  United States
       0.25        98133.50       368196.0
       0.50       282770.50      2312302.0
       0.75       357408.25      6300671.0
```

```
[173]: round(df1[["United Kingdom", "United States"]].mean(), 2)
```

```
[173]: United Kingdom     337147.98
       United States     3571053.69
       dtype: float64
```

```
[171]: # Or alternative and easy method
```

```
[170]: df1[["United Kingdom", "United States"]].describe()
```

```
[170]:        United Kingdom  United States
       count    2.980000e+02   3.090000e+02
```

```
mean     3.371480e+05   3.571054e+06
std      3.460354e+05   3.414074e+06
min      2.000000e+00   1.000000e+00
25%      9.813350e+04   3.681960e+05
50%      2.827705e+05   2.312302e+06
75%      3.574082e+05   6.300671e+06
max      1.527495e+06   1.242087e+07
```

[ ]:

[ ]: