Lab 3

March 24, 2020

[1]:
```
import pandas as pd
```

0.1 Load the dataset from https://www.kaggle.com/sudalairajkumar/novel-corona-virus-2019-dataset

[101]:
```
data = pd.read_csv('covid_19_data.csv')
```

[101]:
```
          SNo ObservationDate Province/State  Country/Region  \
0           1      01/22/2020          Anhui  Mainland China
1           2      01/22/2020        Beijing  Mainland China
2           3      01/22/2020      Chongqing  Mainland China
3           4      01/22/2020         Fujian  Mainland China
4           5      01/22/2020          Gansu  Mainland China
…         …               …              …               …
11336   11337      03/23/2020            NaN      Uzbekistan
11337   11338      03/23/2020            NaN       Venezuela
11338   11339      03/23/2020            NaN         Vietnam
11339   11340      03/23/2020            NaN          Zambia
11340   11341      03/23/2020            NaN        Zimbabwe

                Last Update  Confirmed  Deaths  Recovered
0           1/22/2020 17:00        1.0     0.0        0.0
1           1/22/2020 17:00       14.0     0.0        0.0
2           1/22/2020 17:00        6.0     0.0        0.0
3           1/22/2020 17:00        1.0     0.0        0.0
4           1/22/2020 17:00        0.0     0.0        0.0
…                       …        …       …          …
11336   2020-03-23 23:19:21       46.0     0.0        0.0
11337   2020-03-23 23:19:21       77.0     0.0       15.0
11338   2020-03-23 23:19:21      123.0     0.0       17.0
11339   2020-03-23 23:19:21        3.0     0.0        0.0
11340   2020-03-23 23:19:21        3.0     1.0        0.0

[11341 rows x 8 columns]
```

0.1.1 Visualize first ten records

description of the task

[102]:
```
data.head(20)
```

[102]:

	SNo	ObservationDate	Province/State	Country/Region	Last Update	\
0	1	01/22/2020	Anhui	Mainland China	1/22/2020 17:00	
1	2	01/22/2020	Beijing	Mainland China	1/22/2020 17:00	
2	3	01/22/2020	Chongqing	Mainland China	1/22/2020 17:00	
3	4	01/22/2020	Fujian	Mainland China	1/22/2020 17:00	
4	5	01/22/2020	Gansu	Mainland China	1/22/2020 17:00	
5	6	01/22/2020	Guangdong	Mainland China	1/22/2020 17:00	
6	7	01/22/2020	Guangxi	Mainland China	1/22/2020 17:00	
7	8	01/22/2020	Guizhou	Mainland China	1/22/2020 17:00	
8	9	01/22/2020	Hainan	Mainland China	1/22/2020 17:00	
9	10	01/22/2020	Hebei	Mainland China	1/22/2020 17:00	
10	11	01/22/2020	Heilongjiang	Mainland China	1/22/2020 17:00	
11	12	01/22/2020	Henan	Mainland China	1/22/2020 17:00	
12	13	01/22/2020	Hong Kong	Hong Kong	1/22/2020 17:00	
13	14	01/22/2020	Hubei	Mainland China	1/22/2020 17:00	
14	15	01/22/2020	Hunan	Mainland China	1/22/2020 17:00	
15	16	01/22/2020	Inner Mongolia	Mainland China	1/22/2020 17:00	
16	17	01/22/2020	Jiangsu	Mainland China	1/22/2020 17:00	
17	18	01/22/2020	Jiangxi	Mainland China	1/22/2020 17:00	
18	19	01/22/2020	Jilin	Mainland China	1/22/2020 17:00	
19	20	01/22/2020	Liaoning	Mainland China	1/22/2020 17:00	

	Confirmed	Deaths	Recovered
0	1.0	0.0	0.0
1	14.0	0.0	0.0
2	6.0	0.0	0.0
3	1.0	0.0	0.0
4	0.0	0.0	0.0
5	26.0	0.0	0.0
6	2.0	0.0	0.0
7	1.0	0.0	0.0
8	4.0	0.0	0.0
9	1.0	0.0	0.0
10	0.0	0.0	0.0
11	5.0	0.0	0.0
12	0.0	0.0	0.0
13	444.0	17.0	28.0
14	4.0	0.0	0.0
15	0.0	0.0	0.0
16	1.0	0.0	0.0
17	2.0	0.0	0.0

```
18        0.0     0.0        0.0
19        2.0     0.0        0.0
```

0.1.2 Visualize last ten records

[7]:
```
data.tail(10)
```

[7]:
```
          SNo ObservationDate Province/State        Country/Region  \
11331  11332      03/23/2020            NaN                Uganda
11332  11333      03/23/2020            NaN               Ukraine
11333  11334      03/23/2020            NaN  United Arab Emirates
11334  11335      03/23/2020            NaN                    UK
11335  11336      03/23/2020            NaN               Uruguay
11336  11337      03/23/2020            NaN            Uzbekistan
11337  11338      03/23/2020            NaN             Venezuela
11338  11339      03/23/2020            NaN               Vietnam
11339  11340      03/23/2020            NaN                Zambia
11340  11341      03/23/2020            NaN              Zimbabwe

                Last Update  Confirmed  Deaths  Recovered
11331  2020-03-23 23:19:21        9.0     0.0        0.0
11332  2020-03-23 23:19:21       73.0     3.0        1.0
11333  2020-03-23 23:19:21      198.0     2.0       41.0
11334  2020-03-23 23:19:21     6650.0   335.0      135.0
11335  2020-03-23 23:19:21      158.0     0.0        0.0
11336  2020-03-23 23:19:21       46.0     0.0        0.0
11337  2020-03-23 23:19:21       77.0     0.0       15.0
11338  2020-03-23 23:19:21      123.0     0.0       17.0
11339  2020-03-23 23:19:21        3.0     0.0        0.0
11340  2020-03-23 23:19:21        3.0     1.0        0.0
```

0.1.3 Number of rows

[9]:
```
data.shape[0]
```

[9]:
```
11341
```

0.1.4 Number of columns

[10]:
```
data.shape[1]
```

[10]:
```
8
```

0.1.5 Let's see the column names

[11]:
```
data.columns
```

[11]:
```
Index(['SNo', 'ObservationDate', 'Province/State', 'Country/Region',
       'Last Update', 'Confirmed', 'Deaths', 'Recovered'],
      dtype='object')
```

0.1.6 And now the rows

[13]:
```
data.index
```

[13]:
```
RangeIndex(start=0, stop=11341, step=1)
```

0.1.7 Data Types

[14]:
```
data.dtypes
```

[14]:
```
SNo                 int64
ObservationDate    object
Province/State     object
Country/Region     object
Last Update        object
Confirmed         float64
Deaths            float64
Recovered         float64
dtype: object
```

0.1.8 Distinct Regions

[20]:
```
data['Country/Region'].unique()
```

[20]:
```
array(['Mainland China', 'Hong Kong', 'Macau', 'Taiwan', 'US', 'Japan',
       'Thailand', 'South Korea', 'Singapore', 'Philippines', 'Malaysia',
       'Vietnam', 'Australia', 'Mexico', 'Brazil', 'Colombia', 'France',
       'Nepal', 'Canada', 'Cambodia', 'Sri Lanka', 'Ivory Coast',
       'Germany', 'Finland', 'United Arab Emirates', 'India', 'Italy',
       'UK', 'Russia', 'Sweden', 'Spain', 'Belgium', 'Others', 'Egypt',
       'Iran', 'Israel', 'Lebanon', 'Iraq', 'Oman', 'Afghanistan',
       'Bahrain', 'Kuwait', 'Austria', 'Algeria', 'Croatia',
       'Switzerland', 'Pakistan', 'Georgia', 'Greece', 'North Macedonia',
       'Norway', 'Romania', 'Denmark', 'Estonia', 'Netherlands',
       'San Marino', ' Azerbaijan', 'Belarus', 'Iceland', 'Lithuania',
```

```
'New Zealand', 'Nigeria', 'North Ireland', 'Ireland', 'Luxembourg',
'Monaco', 'Qatar', 'Ecuador', 'Azerbaijan', 'Czech Republic',
'Armenia', 'Dominican Republic', 'Indonesia', 'Portugal',
'Andorra', 'Latvia', 'Morocco', 'Saudi Arabia', 'Senegal',
'Argentina', 'Chile', 'Jordan', 'Ukraine', 'Saint Barthelemy',
'Hungary', 'Faroe Islands', 'Gibraltar', 'Liechtenstein', 'Poland',
'Tunisia', 'Palestine', 'Bosnia and Herzegovina', 'Slovenia',
'South Africa', 'Bhutan', 'Cameroon', 'Costa Rica', 'Peru',
'Serbia', 'Slovakia', 'Togo', 'Vatican City', 'French Guiana',
'Malta', 'Martinique', 'Republic of Ireland', 'Bulgaria',
'Maldives', 'Bangladesh', 'Moldova', 'Paraguay', 'Albania',
'Cyprus', 'St. Martin', 'Brunei', 'occupied Palestinian territory',
"('St. Martin',)", 'Burkina Faso', 'Channel Islands', 'Holy See',
'Mongolia', 'Panama', 'Bolivia', 'Honduras', 'Congo (Kinshasa)',
'Jamaica', 'Reunion', 'Turkey', 'Cuba', 'Guyana', 'Kazakhstan',
'Cayman Islands', 'Guadeloupe', 'Ethiopia', 'Sudan', 'Guinea',
'Antigua and Barbuda', 'Aruba', 'Kenya', 'Uruguay', 'Ghana',
'Jersey', 'Namibia', 'Seychelles', 'Trinidad and Tobago',
'Venezuela', 'Curacao', 'Eswatini', 'Gabon', 'Guatemala',
'Guernsey', 'Mauritania', 'Rwanda', 'Saint Lucia',
'Saint Vincent and the Grenadines', 'Suriname', 'Kosovo',
'Central African Republic', 'Congo (Brazzaville)',
'Equatorial Guinea', 'Uzbekistan', 'Guam', 'Puerto Rico', 'Benin',
'Greenland', 'Liberia', 'Mayotte', 'Republic of the Congo',
'Somalia', 'Tanzania', 'The Bahamas', 'Barbados', 'Montenegro',
'The Gambia', 'Kyrgyzstan', 'Mauritius', 'Zambia', 'Djibouti',
'Gambia, The', 'Bahamas, The', 'Chad', 'El Salvador', 'Fiji',
'Nicaragua', 'Madagascar', 'Haiti', 'Angola', 'Cabo Verde',
'Niger', 'Papua New Guinea', 'Zimbabwe', 'Cape Verde',
'East Timor', 'Eritrea', 'Uganda', 'Dominica', 'Grenada',
'Mozambique', 'Syria', 'Timor-Leste', 'Bahamas', 'Belize',
'Gambia'], dtype=object)
```

0.1.9 Distinct number of Regions

[21]:
```
data['Country/Region'].nunique()
```

[21]: 203

0.1.10 Number of records for each Region (TOP)

[31]:
```
data['Country/Region'].value_counts().head()
```

[31]:
```
US                4796
Mainland China    1920
Australia          331
Canada             266
France             136
Name: Country/Region, dtype: int64
```

0.1.11 Number of records for each Region (Tail)

[32]:
```
data['Country/Region'].value_counts().tail()
```

[32]:
```
Channel Islands        1
 Azerbaijan            1
Republic of Ireland    1
St. Martin             1
North Ireland          1
Name: Country/Region, dtype: int64
```

0.1.12 Basic stats (Numeric ony)

[24]:
```
data.describe()
```

[24]:

	SNo	Confirmed	Deaths	Recovered
count	11341.000000	11341.000000	11341.000000	11341.000000
mean	5671.000000	491.569615	17.524116	174.971078
std	3274.009036	4295.461678	204.529848	2334.484509
min	1.000000	0.000000	0.000000	0.000000
25%	2836.000000	1.000000	0.000000	0.000000
50%	5671.000000	5.000000	0.000000	0.000000
75%	8506.000000	71.000000	0.000000	2.000000
max	11341.000000	67800.000000	6077.000000	59882.000000

0.1.13 Basic stats (all)

[27]:
```
data.describe(include='all')
```

[27]:

	SNo	ObservationDate	Province/State	Country/Region
count	11341.000000	11341	7746	11341
unique	NaN	62	286	203
top	NaN	03/23/2020	Texas	US
freq	NaN	3415	267	4796
mean	5671.000000	NaN	NaN	NaN
std	3274.009036	NaN	NaN	NaN

```
min         1.000000             NaN            NaN            NaN
25%      2836.000000             NaN            NaN            NaN
50%      5671.000000             NaN            NaN            NaN
75%      8506.000000             NaN            NaN            NaN
max     11341.000000             NaN            NaN            NaN

                Last Update     Confirmed        Deaths     Recovered
count                 11341  11341.000000  11341.000000  11341.000000
unique                 1898           NaN           NaN           NaN
top     2020-03-23 23:19:34           NaN           NaN           NaN
freq                   3179           NaN           NaN           NaN
mean                    NaN    491.569615     17.524116    174.971078
std                     NaN   4295.461678    204.529848   2334.484509
min                     NaN      0.000000      0.000000      0.000000
25%                     NaN      1.000000      0.000000      0.000000
50%                     NaN      5.000000      0.000000      0.000000
75%                     NaN     71.000000      0.000000      2.000000
max                     NaN  67800.000000   6077.000000  59882.000000
```

0.1.14 Basic Stats for Confirmed

[44]:
```
data.Confirmed.describe()
```

[44]:
```
count    11341.000000
mean       491.569615
std       4295.461678
min          0.000000
25%          1.000000
50%          5.000000
75%         71.000000
max      67800.000000
Name: Confirmed, dtype: float64
```

0.2 Focus on Italy

[46]:
```
data_it=data[data['Country/Region'] == 'Italy']
data_it.head()
```

[46]:
```
      SNo ObservationDate Province/State Country/Region          Last Update  \
480   481      01/31/2020            NaN          Italy      1/31/2020 23:59
539   540      02/01/2020            NaN          Italy       1/31/2020 8:15
608   609      02/02/2020            NaN          Italy  2020-01-31T08:15:53
675   676      02/03/2020            NaN          Italy  2020-01-31T08:15:53
743   744      02/04/2020            NaN          Italy  2020-01-31T08:15:53
```

```
     Confirmed  Deaths  Recovered
480        2.0     0.0        0.0
539        2.0     0.0        0.0
608        2.0     0.0        0.0
675        2.0     0.0        0.0
743        2.0     0.0        0.0
```

0.2.1 Basic Stats on Italy

[49]:
```
data_it.describe()
```

[49]:
```
                SNo     Confirmed       Deaths    Recovered
count     53.000000     53.000000    53.000000    53.000000
mean    3124.113208   9485.811321   755.396226   969.094340
std     2303.588836  16786.028262  1498.717944  1857.606196
min      481.000000      2.000000     0.000000     0.000000
25%     1391.000000      3.000000     0.000000     0.000000
50%     2394.000000    453.000000    12.000000     3.000000
75%     4515.000000  10149.000000   631.000000   724.000000
max    11255.000000  63927.000000  6077.000000  7432.000000
```

0.2.2 Project only columns of interest

[50]:
```
data_it_f=data_it[['ObservationDate','Confirmed', 'Deaths', 'Recovered']]

data_it_f.head()
```

[50]:
```
    ObservationDate  Confirmed  Deaths  Recovered
480      01/31/2020        2.0     0.0        0.0
539      02/01/2020        2.0     0.0        0.0
608      02/02/2020        2.0     0.0        0.0
675      02/03/2020        2.0     0.0        0.0
743      02/04/2020        2.0     0.0        0.0
```

0.2.3 Plot some graphix for fast visualization

[54]:
```
import matplotlib.pyplot as plt
```

[103]:
```
data_it_f.index=data_it_f['ObservationDate']
data_it_f.head()
```

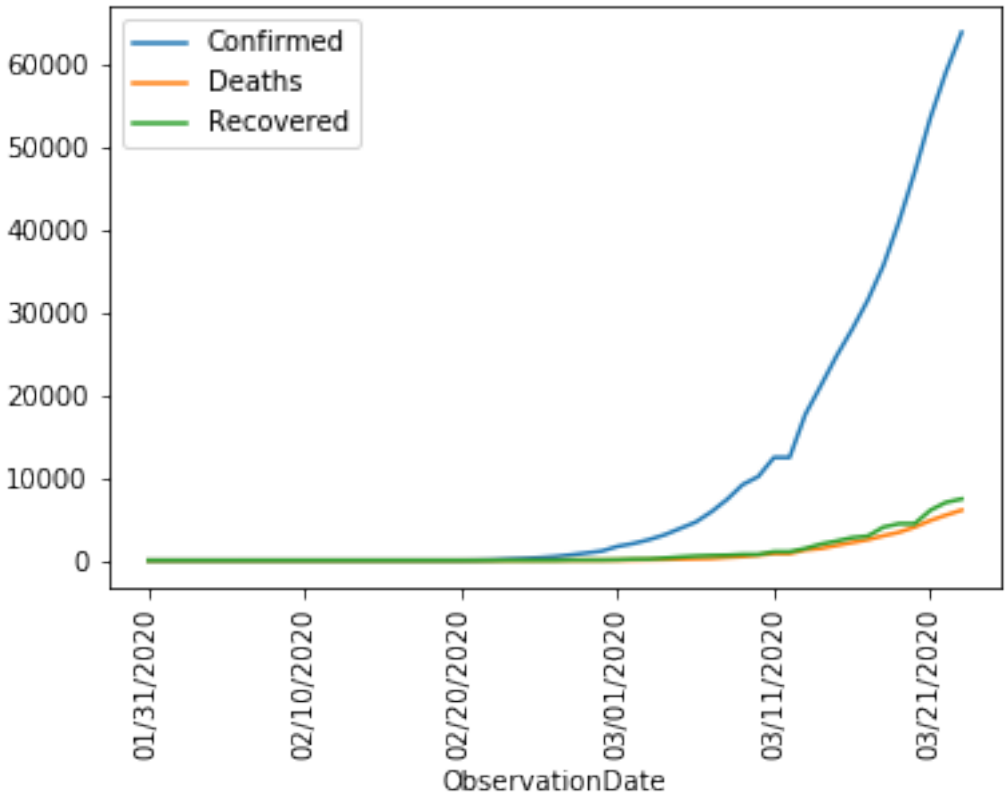
[103]:

ObservationDate	ObservationDate	Confirmed	Deaths	Recovered
01/31/2020	01/31/2020	2.0	0.0	0.0
02/01/2020	02/01/2020	2.0	0.0	0.0
02/02/2020	02/02/2020	2.0	0.0	0.0
02/03/2020	02/03/2020	2.0	0.0	0.0
02/04/2020	02/04/2020	2.0	0.0	0.0

[105]:
```
data_it_f.plot()

plt.xticks(rotation=90)
```

[105]:
```
(array([-10.,   0.,  10.,  20.,  30.,  40.,  50.,  60.]),
 <a list of 8 Text xticklabel objects>)
```



0.2.4 Transform Data to support further analysis (add columns in this case)

[96]:
```
data_it_modified=data_it_f.copy()


data_it_modified['month'] = pd.DatetimeIndex(data_it['ObservationDate']).month
data_it_modified['day'] = pd.DatetimeIndex(data_it['ObservationDate']).day
data_it_modified.head()
```

[96]:

ObservationDate	ObservationDate	Confirmed	Deaths	Recovered	month	day
01/31/2020	01/31/2020	2.0	0.0	0.0	1	31
02/01/2020	02/01/2020	2.0	0.0	0.0	2	1
02/02/2020	02/02/2020	2.0	0.0	0.0	2	2
02/03/2020	02/03/2020	2.0	0.0	0.0	2	3
02/04/2020	02/04/2020	2.0	0.0	0.0	2	4

0.2.5 Transform Data to support further analysis (change index type and compute days of months)

[98]:
```
data_it_modified.index=pd.to_datetime(data_it_modified.index)
data_it_modified.index.days_in_month
```

[98]:
```
Int64Index([31, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29,
            29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 31, 31, 31, 31,
            31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31,
            31, 31],
           dtype='int64', name='ObservationDate')
```

0.2.6 Filter data on the need

[99]:
```
data_it_modified[data_it_modified['day'].isin(data_it_modified.index.
↪days_in_month)]
```

[99]:

ObservationDate	ObservationDate	Confirmed	Deaths	Recovered	month	day
2020-01-31	01/31/2020	2.0	0.0	0.0	1	31
2020-02-29	02/29/2020	1128.0	29.0	46.0	2	29

0.2.7 Use groupby operation to improve analysis (what is the last day of current month?)

[108]:
```
data_it_modified.groupby('month').aggregate({'day':'max'})
```

[108]:

month	day
1	31
2	29
3	23