covid19 data analysis notebook

March 8, 2025

0.1 # Welcome to Covid19 Data Analysis Notebook

0.1.1 Let's Import the modules

[52]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
print('Modules are imported.')
```

```
Modules are imported.
```

0.2 Task 2

0.2.1 Task 2.1: importing covid19 dataset

importing "Covid19_Confirmed_dataset.csv" from "./Dataset" folder.

[53]:
```
corona_dataset_csv = pd.read_csv("Datasets/covid19_Confirmed_dataset.csv")
corona_dataset_csv.head(10)
```

[53]:

	Province/State	Country/Region	Lat	Long	\
0	NaN	Afghanistan	33.0000	65.0000	
1	NaN	Albania	41.1533	20.1683	
2	NaN	Algeria	28.0339	1.6596	
3	NaN	Andorra	42.5063	1.5218	
4	NaN	Angola	-11.2027	17.8739	
5	NaN	Antigua and Barbuda	17.0608	-61.7964	
6	NaN	Argentina	-38.4161	-63.6167	
7	NaN	Armenia	40.0691	45.0382	
8	Australian Capital Territory	Australia	-35.4735	149.0124	
9	New South Wales	Australia	-33.8688	151.2093	

	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	…	4/21/20	\
0	0	0	0	0	0	0	…	1092	
1	0	0	0	0	0	0	…	609	
2	0	0	0	0	0	0	…	2811	
3	0	0	0	0	0	0	…	717	
4	0	0	0	0	0	0	…	24	

```
5         0         0         0         0         0         0  …        23
6         0         0         0         0         0         0  …      3031
7         0         0         0         0         0         0  …      1401
8         0         0         0         0         0         0  …       104
9         0         0         0         0         3         4  …      2969

   4/22/20  4/23/20  4/24/20  4/25/20  4/26/20  4/27/20  4/28/20  4/29/20  \
0     1176     1279     1351     1463     1531     1703     1828     1939
1      634      663      678      712      726      736      750      766
2     2910     3007     3127     3256     3382     3517     3649     3848
3      723      723      731      738      738      743      743      743
4       25       25       25       25       26       27       27       27
5       24       24       24       24       24       24       24       24
6     3144     3435     3607     3780     3892     4003     4127     4285
7     1473     1523     1596     1677     1746     1808     1867     1932
8      104      104      105      106      106      106      106      106
9     2971     2976     2982     2994     3002     3004     3016     3016

   4/30/20
0     2171
1      773
2     4006
3      745
4       27
5       24
6     4428
7     2066
8      106
9     3025

[10 rows x 104 columns]
```

Let's check the shape of the dataframe

[54]:
```
corona_dataset_csv.shape
```

[54]:
```
(266, 104)
```

0.2.2 Task 2.2: Delete the useless columns

[59]:
```
corona_dataset_csv.drop(["Lat","Long"], axis=1,inplace=True)
```

[63]:
```
corona_dataset_csv.head(10)
```

[63]:
```
                Province/State       Country/Region  1/22/20  1/23/20  \
0                          NaN          Afghanistan        0        0
1                          NaN              Albania        0        0
```

```
2                          NaN              Algeria        0        0
3                          NaN              Andorra        0        0
4                          NaN               Angola        0        0
5                          NaN  Antigua and Barbuda        0        0
6                          NaN            Argentina        0        0
7                          NaN              Armenia        0        0
8  Australian Capital Territory            Australia        0        0
9              New South Wales            Australia        0        0

   1/24/20  1/25/20  1/26/20  1/27/20  1/28/20  1/29/20  …  4/21/20  \
0        0        0        0        0        0        0  …     1092
1        0        0        0        0        0        0  …      609
2        0        0        0        0        0        0  …     2811
3        0        0        0        0        0        0  …      717
4        0        0        0        0        0        0  …       24
5        0        0        0        0        0        0  …       23
6        0        0        0        0        0        0  …     3031
7        0        0        0        0        0        0  …     1401
8        0        0        0        0        0        0  …      104
9        0        0        3        4        4        4  …     2969

   4/22/20  4/23/20  4/24/20  4/25/20  4/26/20  4/27/20  4/28/20  4/29/20  \
0     1176     1279     1351     1463     1531     1703     1828     1939
1      634      663      678      712      726      736      750      766
2     2910     3007     3127     3256     3382     3517     3649     3848
3      723      723      731      738      738      743      743      743
4       25       25       25       25       26       27       27       27
5       24       24       24       24       24       24       24       24
6     3144     3435     3607     3780     3892     4003     4127     4285
7     1473     1523     1596     1677     1746     1808     1867     1932
8      104      104      105      106      106      106      106      106
9     2971     2976     2982     2994     3002     3004     3016     3016

   4/30/20
0     2171
1      773
2     4006
3      745
4       27
5       24
6     4428
7     2066
8      106
9     3025

[10 rows x 102 columns]
```

0.2.3 Task 2.3: Aggregating the rows by the country

[64]:
```
corona_dataset_aggregated = corona_dataset_csv.groupby("Country/Region").sum()
```

[65]:
```
corona_dataset_aggregated.head(10)
```

[65]:

Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	\
Afghanistan	0	0	0	0	0	0	
Albania	0	0	0	0	0	0	
Algeria	0	0	0	0	0	0	
Andorra	0	0	0	0	0	0	
Angola	0	0	0	0	0	0	
Antigua and Barbuda	0	0	0	0	0	0	
Argentina	0	0	0	0	0	0	
Armenia	0	0	0	0	0	0	
Australia	0	0	0	0	4	5	
Austria	0	0	0	0	0	0	

Country/Region	1/28/20	1/29/20	1/30/20	1/31/20	…	4/21/20	\
Afghanistan	0	0	0	0	…	1092	
Albania	0	0	0	0	…	609	
Algeria	0	0	0	0	…	2811	
Andorra	0	0	0	0	…	717	
Angola	0	0	0	0	…	24	
Antigua and Barbuda	0	0	0	0	…	23	
Argentina	0	0	0	0	…	3031	
Armenia	0	0	0	0	…	1401	
Australia	5	6	9	9	…	6645	
Austria	0	0	0	0	…	14873	

Country/Region	4/22/20	4/23/20	4/24/20	4/25/20	4/26/20	4/27/20	\
Afghanistan	1176	1279	1351	1463	1531	1703	
Albania	634	663	678	712	726	736	
Algeria	2910	3007	3127	3256	3382	3517	
Andorra	723	723	731	738	738	743	
Angola	25	25	25	25	26	27	
Antigua and Barbuda	24	24	24	24	24	24	
Argentina	3144	3435	3607	3780	3892	4003	
Armenia	1473	1523	1596	1677	1746	1808	
Australia	6652	6662	6677	6694	6714	6721	
Austria	14925	15002	15071	15148	15225	15274	

Country/Region	4/28/20	4/29/20	4/30/20

```
Afghanistan             1828     1939     2171
Albania                  750      766      773
Algeria                 3649     3848     4006
Andorra                  743      743      745
Angola                    27       27       27
Antigua and Barbuda       24       24       24
Argentina               4127     4285     4428
Armenia                 1867     1932     2066
Australia               6744     6752     6766
Austria                15357    15402    15452

[10 rows x 100 columns]
```

[16]:
```
corona_dataset_aggregated.shape
```

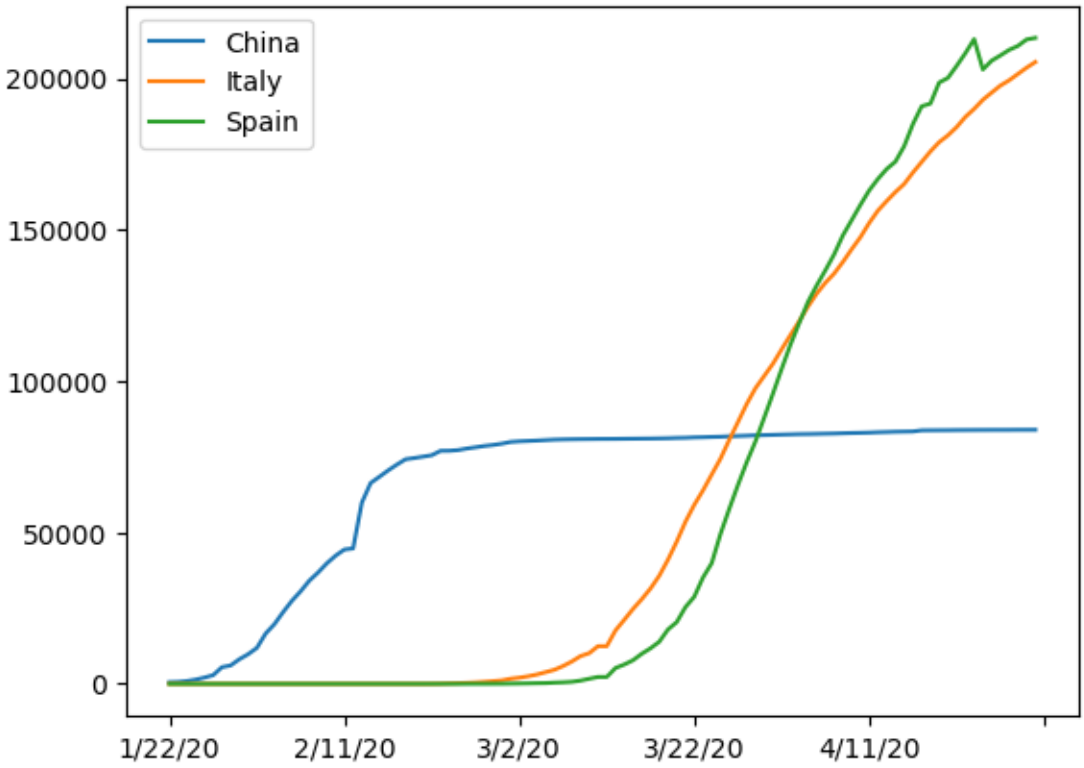
[16]:
```
(187, 102)
```

0.2.4 Task 2.4: Visualizing data related to a country for example China

visualization always helps for better understanding of our data.

[71]:
```
corona_dataset_aggregated.loc["China"].plot()
corona_dataset_aggregated.loc["Italy"].plot()
corona_dataset_aggregated.loc["Spain"].plot()
plt.legend()
```

[71]:
```
<matplotlib.legend.Legend at 0x7968b9c7fa60>
```

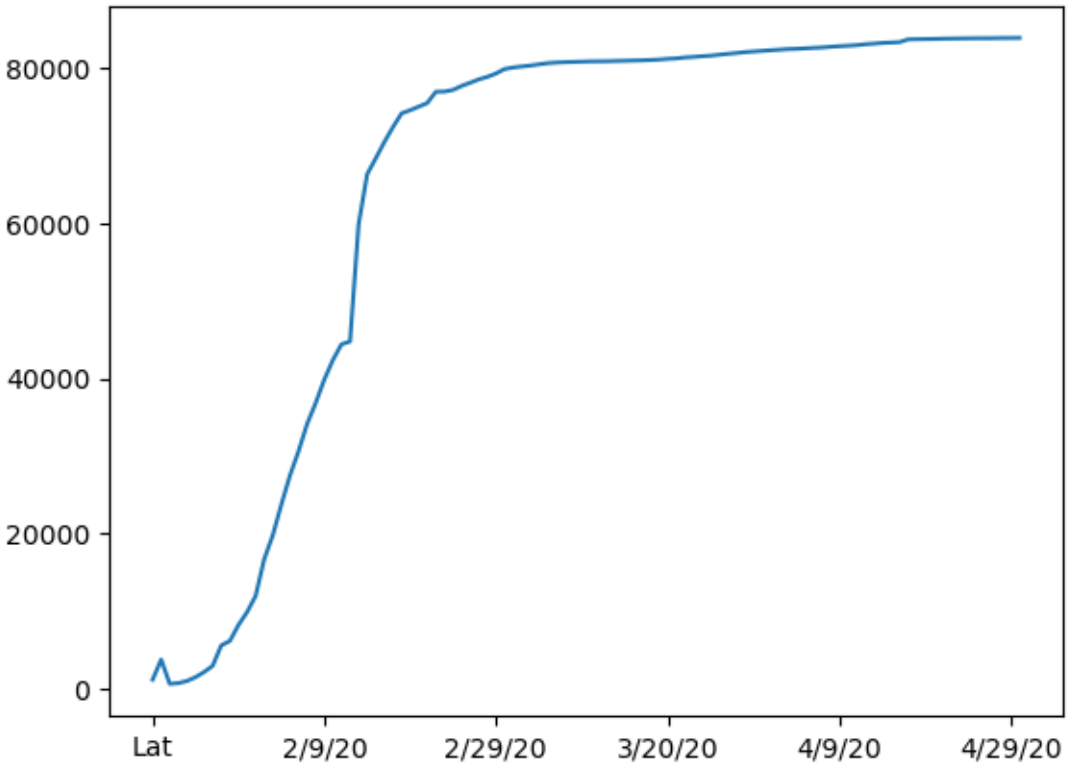


0.2.5 Task3: Calculating a good measure

we need to find a good measure reperestend as a number, describing the spread of the virus in a country.

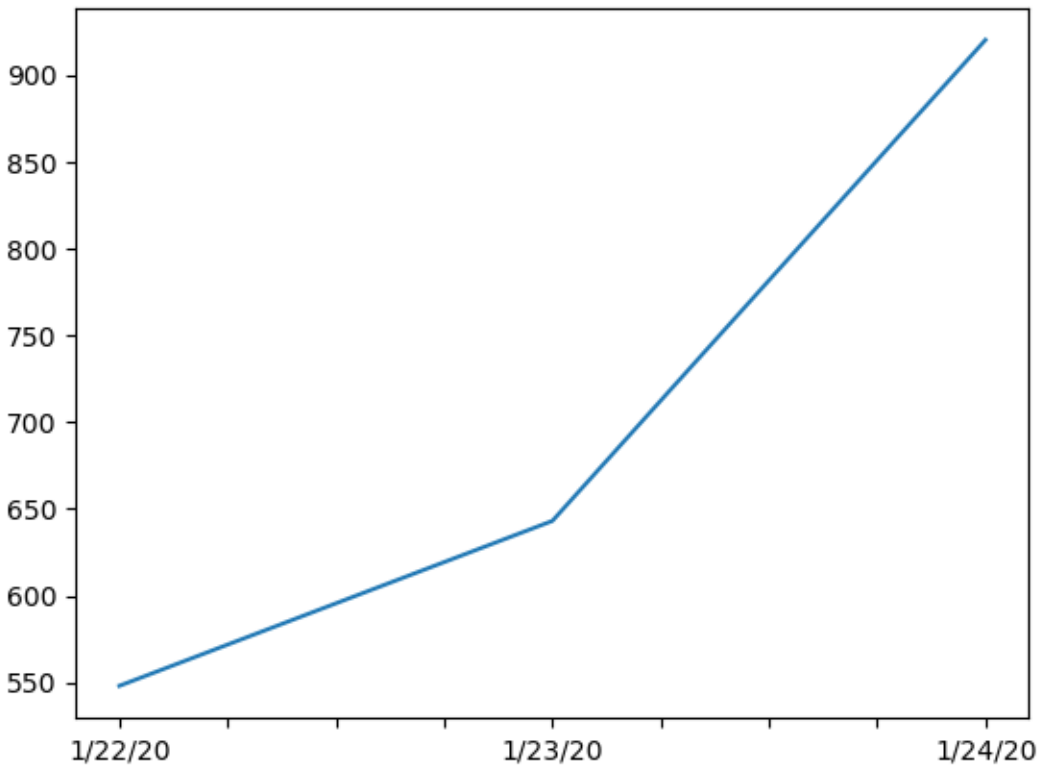
[22]:

[22]: <AxesSubplot: >



[72]:
```
corona_dataset_aggregated.loc["China"][:3].plot()
```

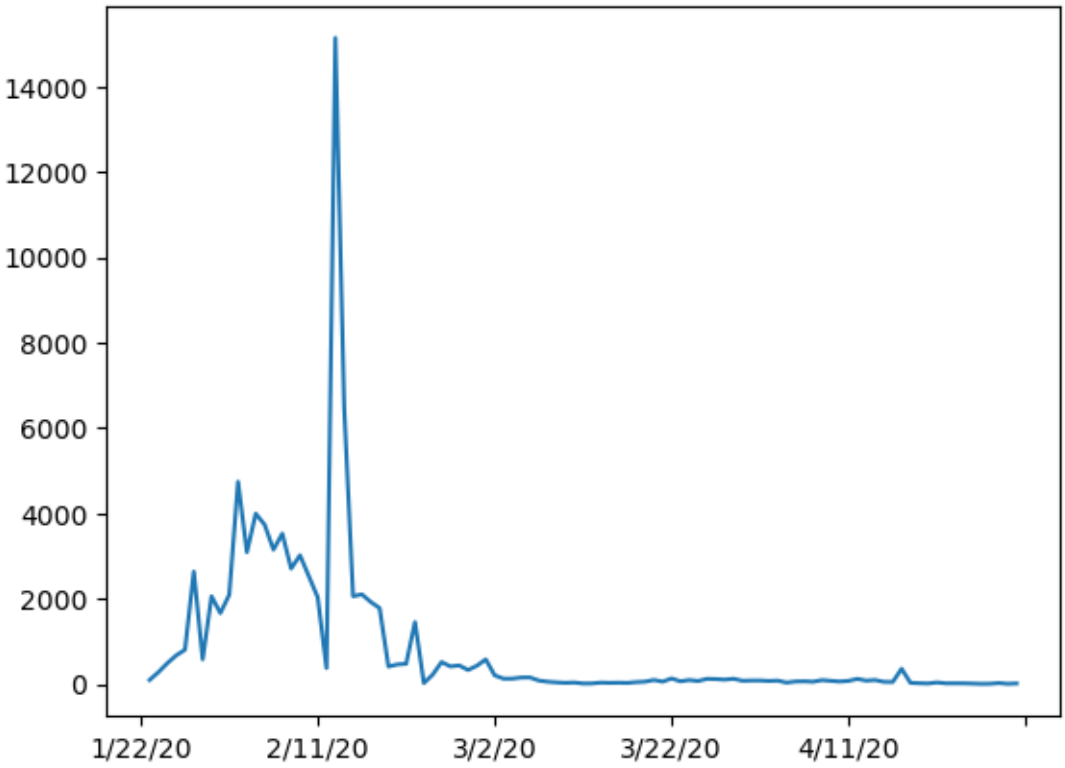
[72]: <AxesSubplot: >



0.2.6 task 3.1: caculating the first derivative of the curve

[74]:
```
corona_dataset_aggregated.loc["China"].diff().plot()
```

[74]: <AxesSubplot: >



0.2.7 task 3.2: find maxmimum infection rate for China

[75]:
```
corona_dataset_aggregated.loc["China"].diff().max()
```

[75]: 15136.0

[76]:
```
corona_dataset_aggregated.loc["Italy"].diff().max()
```

[76]: 6557.0

[77]:
```
corona_dataset_aggregated.loc["Spain"].diff().max()
```

[77]: 9630.0

0.2.8 Task 3.3: find maximum infection rate for all of the countries.

[83]:
```
countries = list(corona_dataset_aggregated.index)
max_infection_rates = []
for c in countries :
    max_infection_rates.append(corona_dataset_aggregated.loc[c].diff().max())
corona_dataset_aggregated["max_infection_rate"] = max_infection_rates
```

[85]:
```
corona_dataset_aggregated.head()
```

[85]:

Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	\
Afghanistan	0	0	0	0	0	0	0	
Albania	0	0	0	0	0	0	0	
Algeria	0	0	0	0	0	0	0	
Andorra	0	0	0	0	0	0	0	
Angola	0	0	0	0	0	0	0	

Country/Region	1/29/20	1/30/20	1/31/20	…	4/22/20	4/23/20	4/24/20	\
Afghanistan	0	0	0	…	1176	1279	1351	
Albania	0	0	0	…	634	663	678	
Algeria	0	0	0	…	2910	3007	3127	
Andorra	0	0	0	…	723	723	731	
Angola	0	0	0	…	25	25	25	

Country/Region	4/25/20	4/26/20	4/27/20	4/28/20	4/29/20	4/30/20	\
Afghanistan	1463	1531	1703	1828	1939	2171	
Albania	712	726	736	750	766	773	
Algeria	3256	3382	3517	3649	3848	4006	
Andorra	738	738	743	743	743	745	
Angola	25	26	27	27	27	27	

Country/Region	max_infection_rate
Afghanistan	232.0
Albania	34.0
Algeria	199.0
Andorra	43.0
Angola	5.0

[5 rows x 101 columns]

0.2.9 Task 3.4: create a new dataframe with only needed column

[87]:
```
corona_data = pd.DataFrame(corona_dataset_aggregated["max_infection_rate"])
```

[88]:
```
corona_data.head()
```

[88]:

Country/Region	max_infection_rate
Afghanistan	232.0
Albania	34.0
Algeria	199.0

```
Andorra                          43.0
Angola                            5.0
```

0.2.10 Task4:

- Importing the WorldHappinessReport.csv dataset
- selecting needed columns for our analysis
- join the datasets
- calculate the correlations as the result of our analysis

0.2.11 Task 4.1 : importing the dataset

[91]:
```
happiness_report_csv= pd.read_csv("Datasets/worldwide_happiness_report.csv")
```

[93]:
```
happiness_report_csv.head()
```

[93]:
```
   Overall rank Country or region  Score  GDP per capita  Social support  \
0             1           Finland  7.769           1.340           1.587
1             2           Denmark  7.600           1.383           1.573
2             3            Norway  7.554           1.488           1.582
3             4           Iceland  7.494           1.380           1.624
4             5       Netherlands  7.488           1.396           1.522

   Healthy life expectancy  Freedom to make life choices  Generosity  \
0                    0.986                         0.596       0.153
1                    0.996                         0.592       0.252
2                    1.028                         0.603       0.271
3                    1.026                         0.591       0.354
4                    0.999                         0.557       0.322

   Perceptions of corruption
0                      0.393
1                      0.410
2                      0.341
3                      0.118
4                      0.298
```

0.2.12 Task 4.2: let's drop the useless columns

[96]:
```
useless_cols = ["Overall rank","Score","Generosity","Perceptions of corruption"]
```

[97]:
```
happiness_report_csv.drop(useless_cols,axis=1,inplace=True)
happiness_report_csv.head()
```

[97]:
```
  Country or region  GDP per capita  Social support  Healthy life expectancy  \
0           Finland           1.340           1.587                    0.986
1           Denmark           1.383           1.573                    0.996
```

```
2            Norway           1.488           1.582                    1.028
3           Iceland           1.380           1.624                    1.026
4       Netherlands           1.396           1.522                    0.999

   Freedom to make life choices
0                         0.596
1                         0.592
2                         0.603
3                         0.591
4                         0.557
```

0.2.13 Task 4.3: changing the indices of the dataframe

[99]:
```
happiness_report_csv.set_index("Country or region", inplace=True)
```

[100]:
```
happiness_report_csv.head()
```

[100]:
```
                  GDP per capita  Social support  Healthy life expectancy  \
Country or region
Finland                     1.340           1.587                    0.986
Denmark                     1.383           1.573                    0.996
Norway                      1.488           1.582                    1.028
Iceland                     1.380           1.624                    1.026
Netherlands                 1.396           1.522                    0.999

                  Freedom to make life choices
Country or region
Finland                                   0.596
Denmark                                   0.592
Norway                                    0.603
Iceland                                   0.591
Netherlands                               0.557
```

0.2.14 Task4.4: now let's join two dataset we have prepared

Corona Dataset :

[101]:
```
corona_data.head()
```

[101]:
```
               max_infection_rate
Country/Region
Afghanistan                232.0
Albania                     34.0
Algeria                    199.0
Andorra                     43.0
Angola                       5.0
```

[102]:
```
corona_data.shape
```

[102]:
```
(187, 1)
```

wolrd happiness report Dataset :

[103]:
```
happiness_report_csv.head()
```

[103]:
```
                    GDP per capita  Social support  Healthy life expectancy  \
Country or region
Finland                      1.340           1.587                    0.986
Denmark                      1.383           1.573                    0.996
Norway                       1.488           1.582                    1.028
Iceland                      1.380           1.624                    1.026
Netherlands                  1.396           1.522                    0.999

                    Freedom to make life choices
Country or region
Finland                                    0.596
Denmark                                    0.592
Norway                                     0.603
Iceland                                    0.591
Netherlands                                0.557
```

[104]:
```
happiness_report_csv.shape
```

[104]:
```
(156, 4)
```

0.2.15 Task 4.5: correlation matrix

[106]:
```
data=corona_data.join(happiness_report_csv, how="inner")
data.head()
```

[106]:
```
             max_infection_rate  GDP per capita  Social support  \
Afghanistan               232.0           0.350           0.517
Albania                    34.0           0.947           0.848
Algeria                   199.0           1.002           1.160
Argentina                 291.0           1.092           1.432
Armenia                   134.0           0.850           1.055

             Healthy life expectancy  Freedom to make life choices
Afghanistan                    0.361                         0.000
Albania                        0.874                         0.383
Algeria                        0.785                         0.086
Argentina                      0.881                         0.471
Armenia                        0.815                         0.283
```

0.2.16 Task 5: Visualization of the results

our Analysis is not finished unless we visualize the results in terms figures and graphs so that everyone can understand what you get out of our analysis

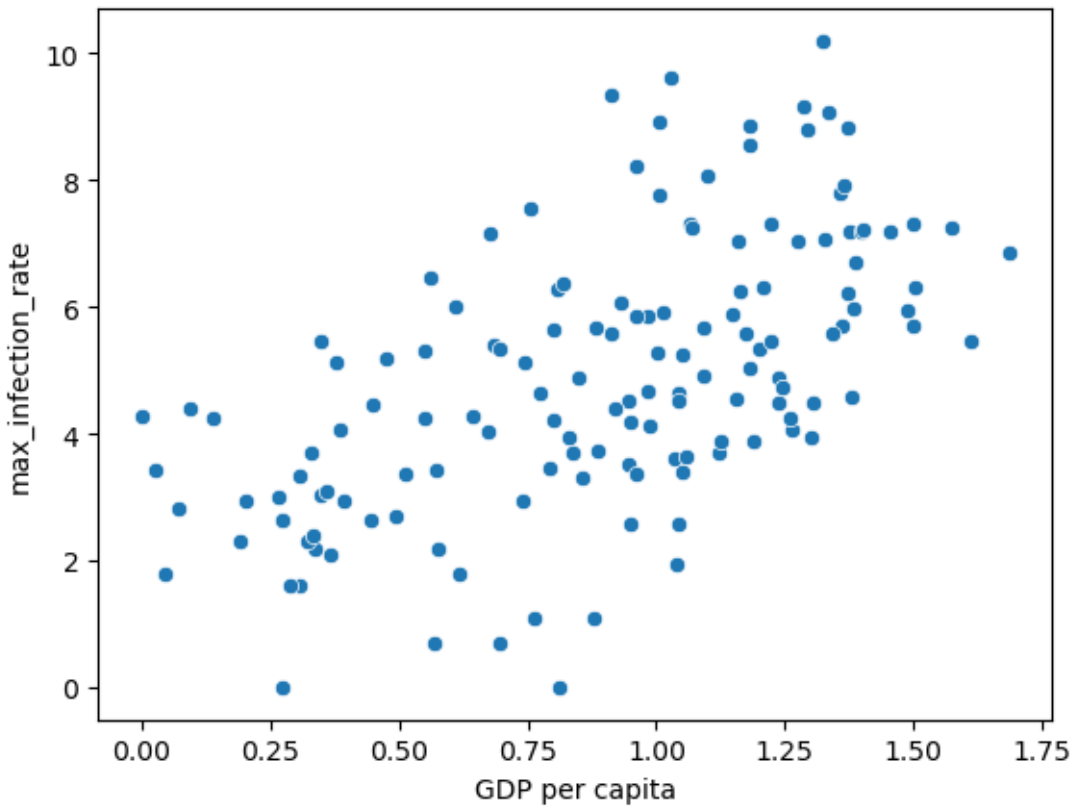
[108]:
```
data.corr()
```

[108]:

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
max_infection_rate	1.000000	0.250118	0.191958	0.289263	0.078196
GDP per capita	0.250118	1.000000	0.759468	0.863062	0.394603
Social support	0.191958	0.759468	1.000000	0.765286	0.456246
Healthy life expectancy	0.289263	0.863062	0.765286	1.000000	0.427892
Freedom to make life choices	0.078196	0.394603	0.456246	0.427892	1.000000

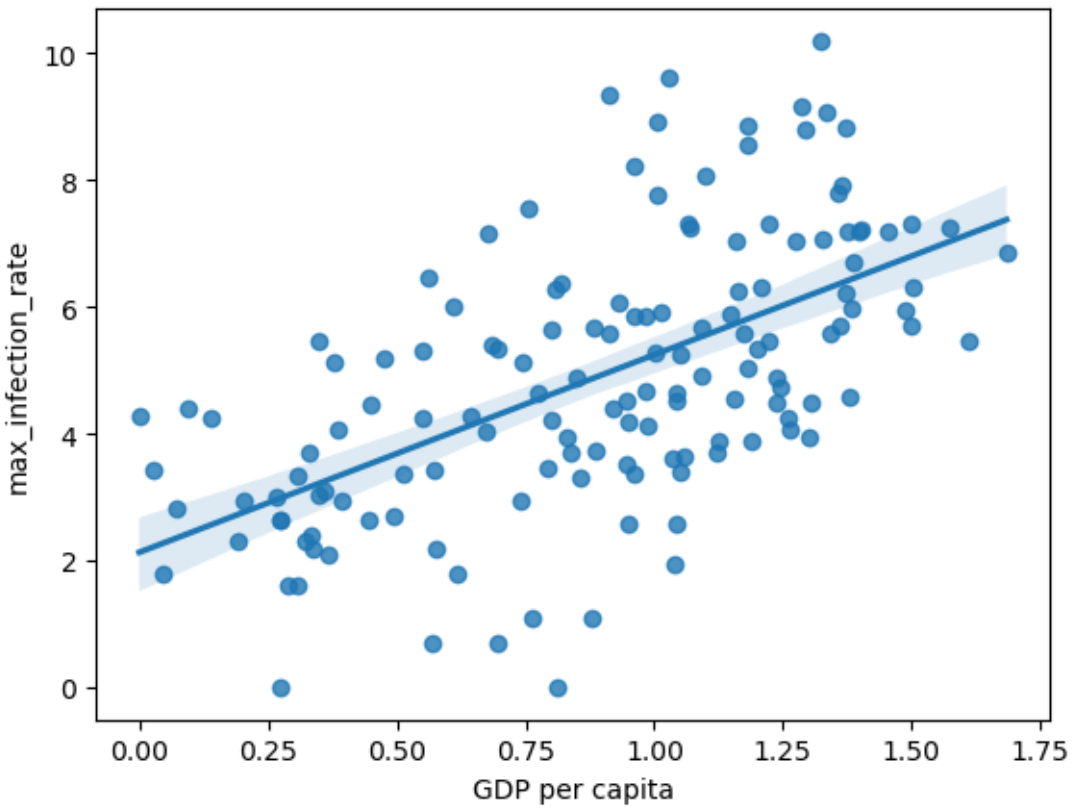
0.2.17 Task 5.1: Plotting GDP vs maximum Infection rate

[119]:
```
x = data["GDP per capita"]
y = data["max_infection_rate"]
sns.scatterplot(x=x,y=np.log(y))
plt.show()
```



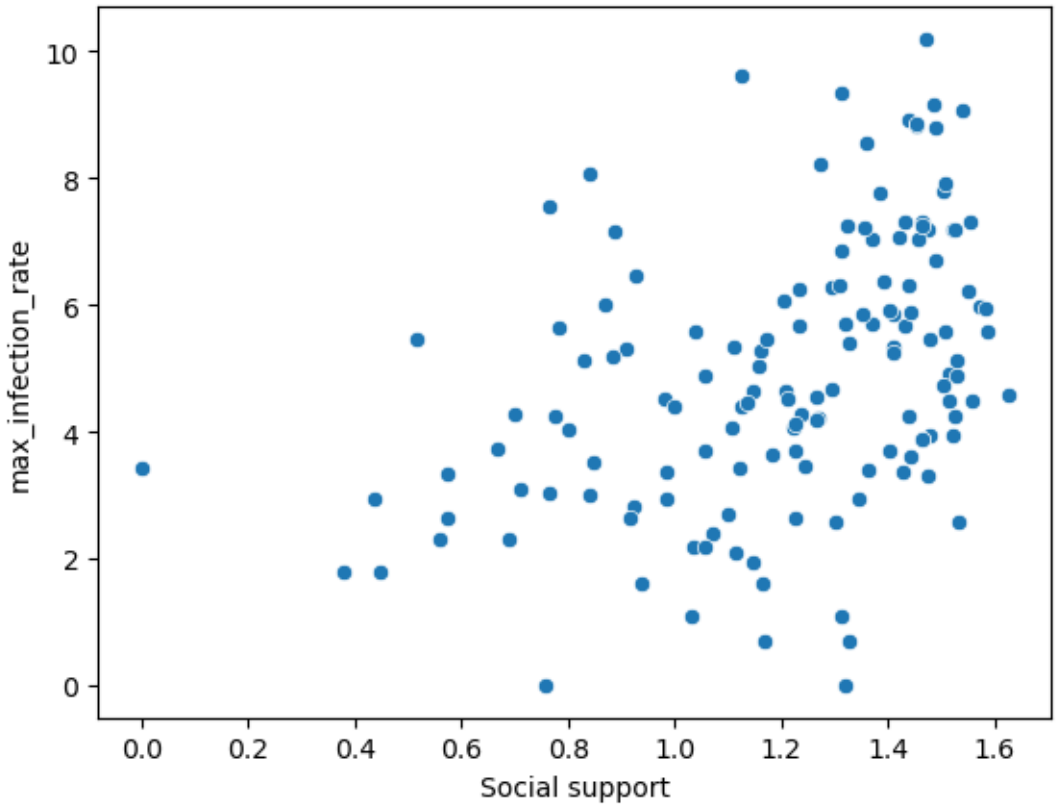
[122]:
```
sns.regplot(x=x, y=np.log(y))
```

[122]: <AxesSubplot: xlabel='GDP per capita', ylabel='max_infection_rate'>



0.2.18 Task 5.2: Plotting Social support vs maximum Infection rate

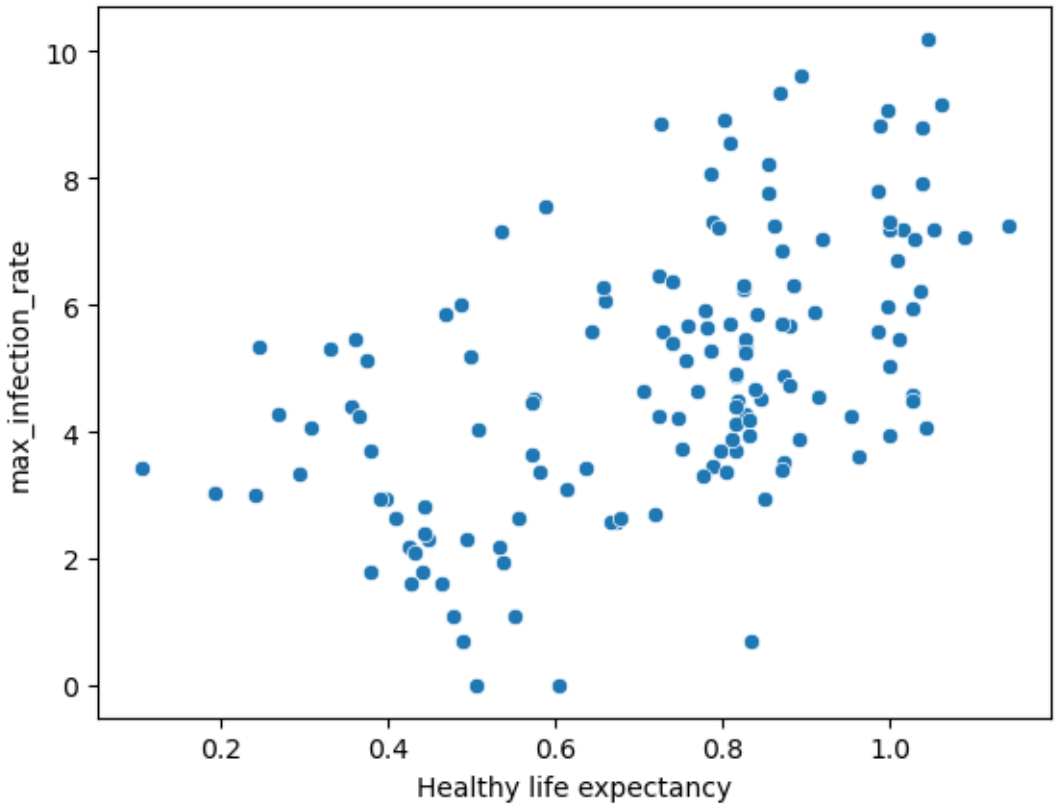
[123]:
```
x = data["Social support"]
y = data["max_infection_rate"]
sns.scatterplot(x=x,y=np.log(y))
plt.show()
```



[ ]:

0.2.19 Task 5.3: Plotting Healthy life expectancy vs maximum Infection rate

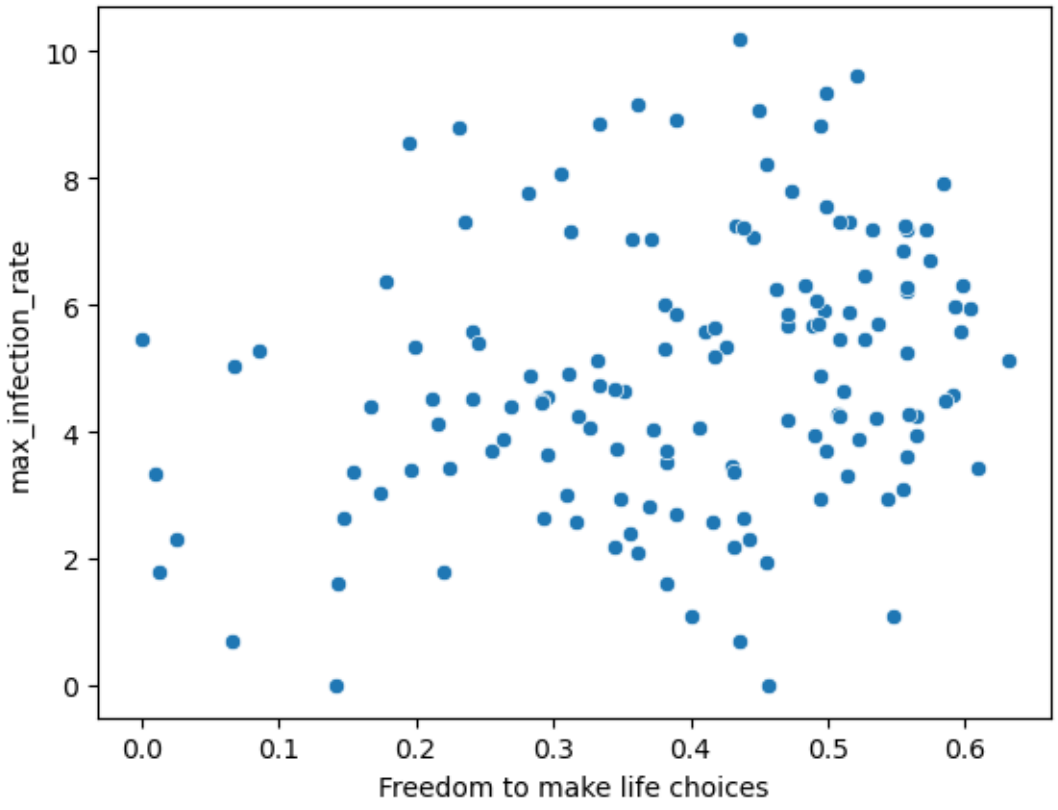
[124]:
```
x = data["Healthy life expectancy"]
y = data["max_infection_rate"]
sns.scatterplot(x=x,y=np.log(y))
plt.show()
```



[ ]:

0.2.20 Task 5.4: Plotting Freedom to make life choices vs maximum Infection rate

[126]:
```
x = data["Freedom to make life choices"]
y = data["max_infection_rate"]
sns.scatterplot(x=x,y=np.log(y))
plt.show()
```


max_infection_rate
Freedom to make life choices
0
2
4
6
8
10
0.0
0.1
0.2
0.3
0.4
0.5
0.6

[ ]: