Welcome to Covid19 Data Analysis Notebook

Let's Import the modules

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
print('Modules are imported.')
```

```
Modules are imported.
```

2

2.1: importing covid19 dataset

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

In [2]:
```
c_df = pd.read_csv("Datasets/covid19_Confirmed_dataset.csv")
```

Let's check the shape of the dataframe

In [3]:
```
c_df.shape
```

Out[3]:
```
(266, 104)
```

In [4]:
```
c_df.head(10)
```

Out[4]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/2
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	
1	NaN	Albania	41.1533	20.1683	0	0	0	
2	NaN	Algeria	28.0339	1.6596	0	0	0	
3	NaN	Andorra	42.5063	1.5218	0	0	0	
4	NaN	Angola	-11.2027	17.8739	0	0	0	
5	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	
7	NaN	Armenia	40.0691	45.0382	0	0	0	
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	

10 rows × 104 columns

2.2: Delete the useless columns

In [5]:
```
df = c_df.drop(['Lat','Long'], axis = 1)
```

In [6]:
```
df.head(10)
```

Out[6]:

	Province/State	Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27
0	NaN	Afghanistan	0	0	0	0	0	
1	NaN	Albania	0	0	0	0	0	
2	NaN	Algeria	0	0	0	0	0	
3	NaN	Andorra	0	0	0	0	0	
4	NaN	Angola	0	0	0	0	0	
5	NaN	Antigua and Barbuda	0	0	0	0	0	
6	NaN	Argentina	0	0	0	0	0	
7	NaN	Armenia	0	0	0	0	0	
8	Australian Capital Territory	Australia	0	0	0	0	0	
9	New South Wales	Australia	0	0	0	0	3	

10 rows × 102 columns

2.3: Aggregating the rows by the country

In [7]:
```
df.columns
```

Out[7]:
```
Index(['Province/State', 'Country/Region', '1/22/20', '1/23/20', '1/24/20',
       '1/25/20', '1/26/20', '1/27/20', '1/28/20', '1/29/20',
       ...
       '4/21/20', '4/22/20', '4/23/20', '4/24/20', '4/25/20', '4/26/20',
       '4/27/20', '4/28/20', '4/29/20', '4/30/20'],
      dtype='object', length=102)
```

In [8]:
```
df_aggregated = df.groupby('Country/Region').sum()
```

In [9]:
```
df_aggregated
```

Out[9]:

Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29
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	
...	...	...	...	...	...	...	...	
West Bank and Gaza	0	0	0	0	0	0	0	
Western Sahara	0	0	0	0	0	0	0	
Yemen	0	0	0	0	0	0	0	
Zambia	0	0	0	0	0	0	0	
Zimbabwe	0	0	0	0	0	0	0	

187 rows × 100 columns

2.4: Visualizing data related to a country for example China

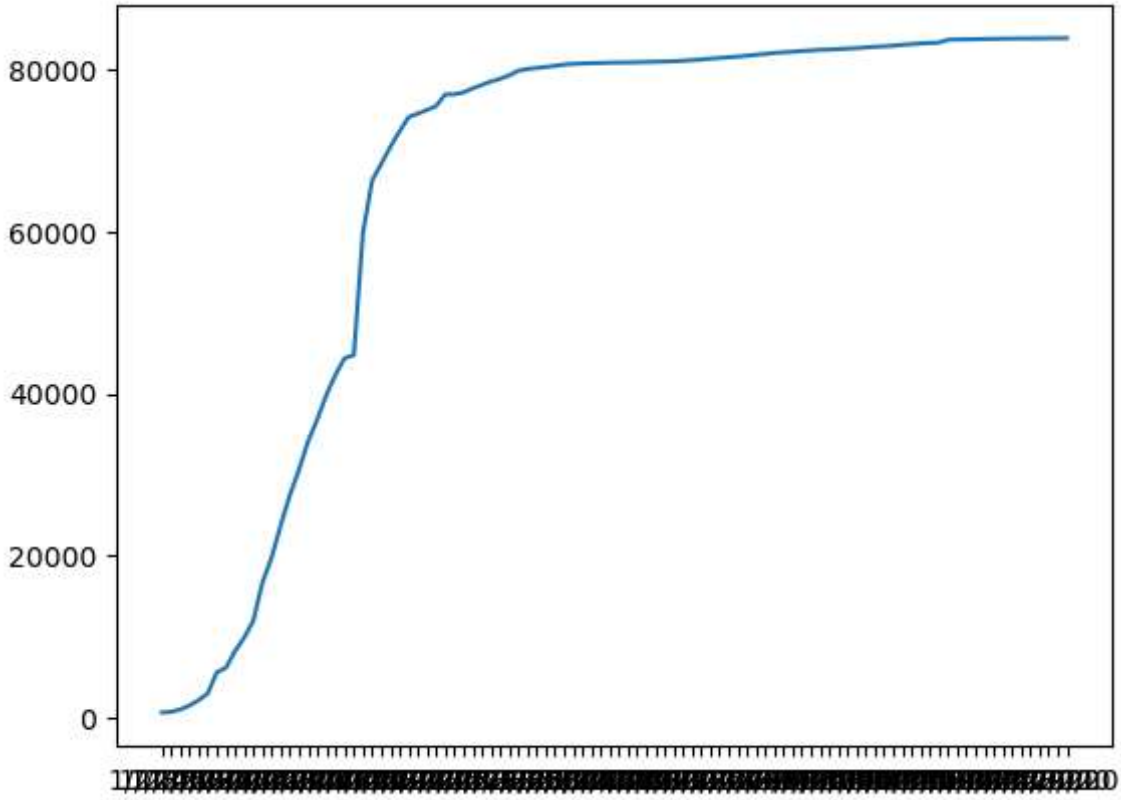
visualization always helps for better understanding of our data.

In [10]:
```
df_aggregated.loc["China"]
```

Out[10]:
```
1/22/20      548
1/23/20      643
1/24/20      920
1/25/20     1406
1/26/20     2075
           ...
4/26/20    83912
4/27/20    83918
4/28/20    83940
4/29/20    83944
4/30/20    83956
Name: China, Length: 100, dtype: int64
```

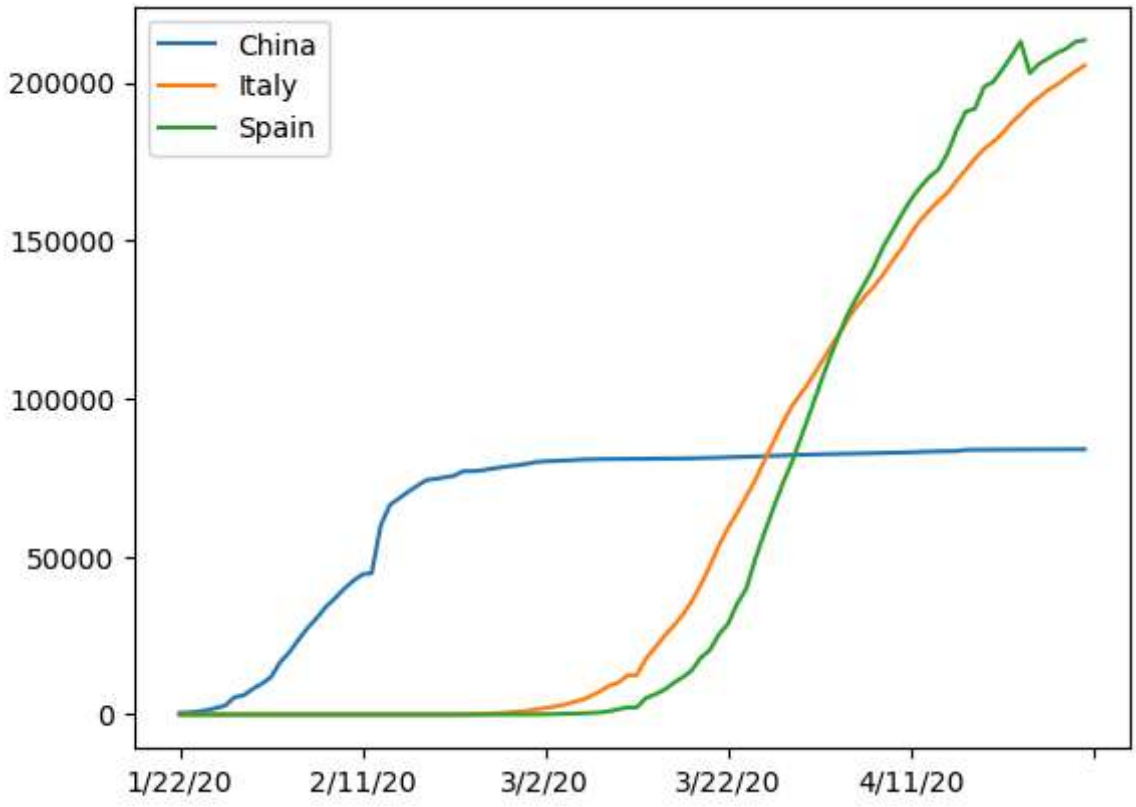
In [11]:
```
plt.plot(df_aggregated.loc["China"])
```

Out[11]:
```
[<matplotlib.lines.Line2D at 0x70a074c27220>]
```



In [12]:
```
df_aggregated.loc["China"].plot()
df_aggregated.loc["Italy"].plot()
df_aggregated.loc["Spain"].plot()
plt.legend()
```

Out[12]: <matplotlib.legend.Legend at 0x70a06c813190>

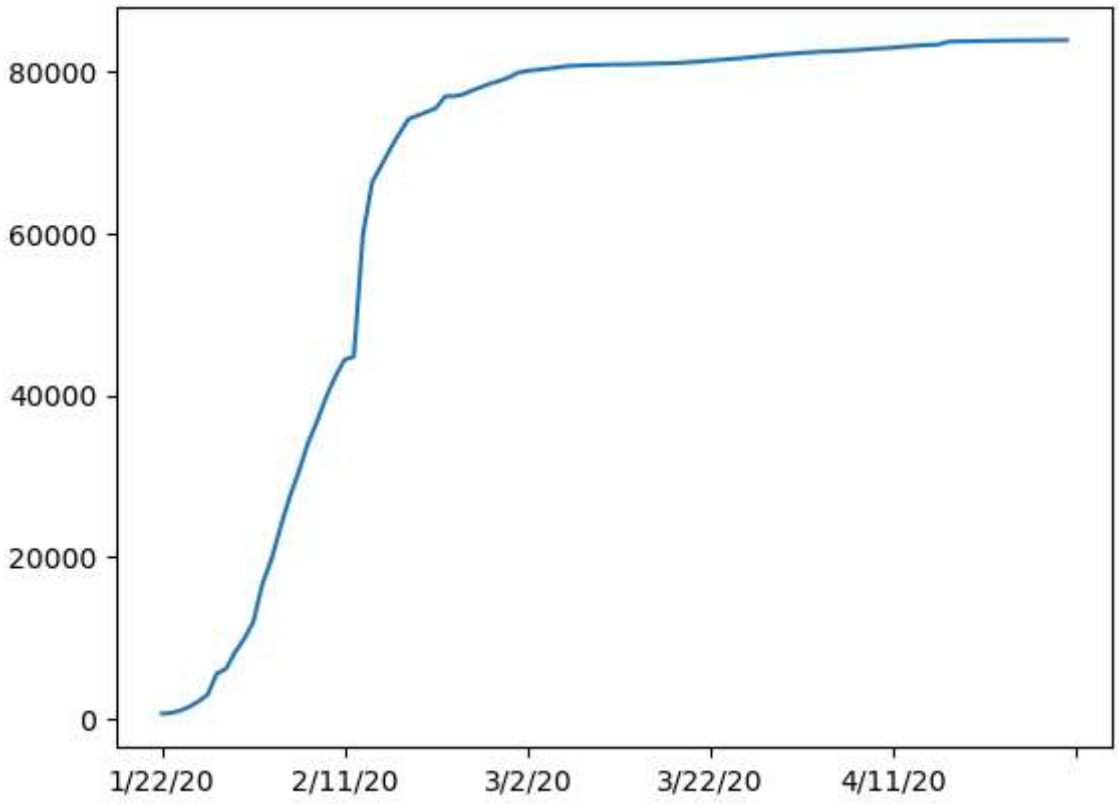


3: Calculating a good measure

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

In [13]:
```
df_aggregated.loc['China'].plot()
```

Out[13]: <AxesSubplot: >



In [14]:
```
df.describe()
```

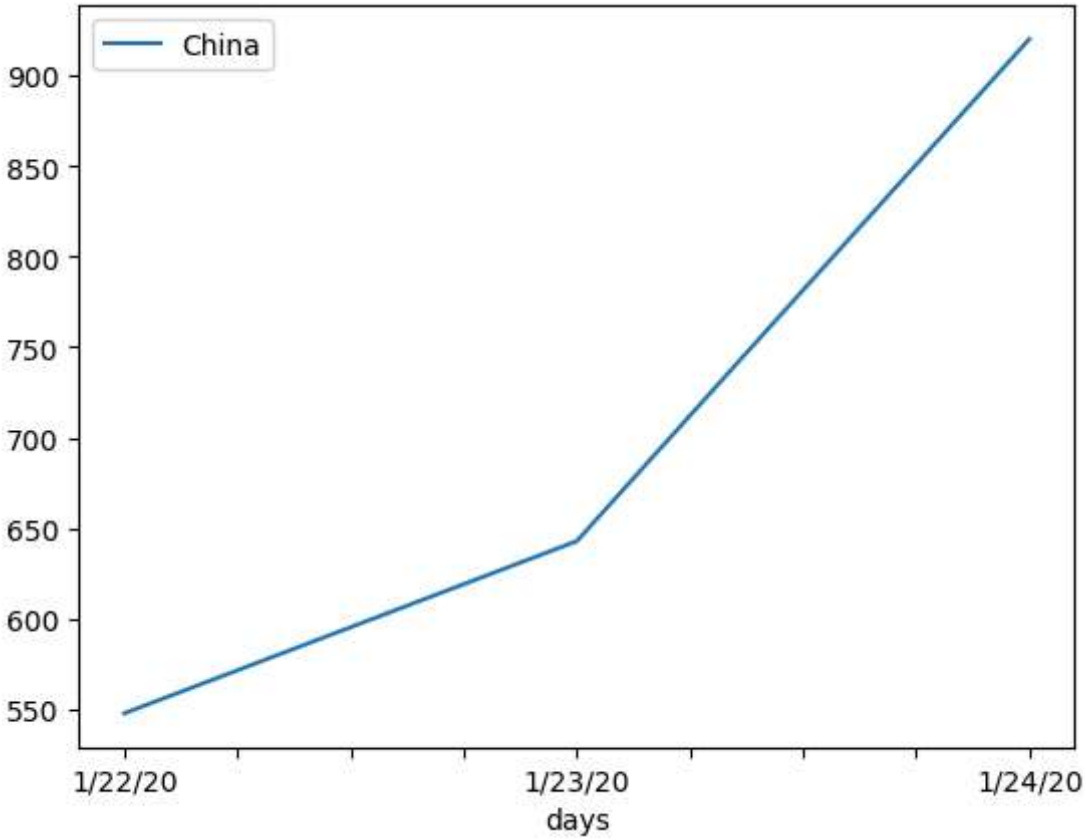
Out[14]:

	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/
count	266.000000	266.000000	266.000000	266.000000	266.000000	266.000000	266.0
mean	2.086466	2.458647	3.537594	5.390977	7.962406	11.003759	20.9
std	27.279200	27.377862	34.083035	47.434934	66.289178	89.313757	219.1
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
50%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
75%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
max	444.000000	444.000000	549.000000	761.000000	1058.000000	1423.000000	3554.0

8 rows × 100 columns

In [15]:
```
df_aggregated.loc['China'][:3].plot()
plt.legend()
plt.xlabel("days")
```

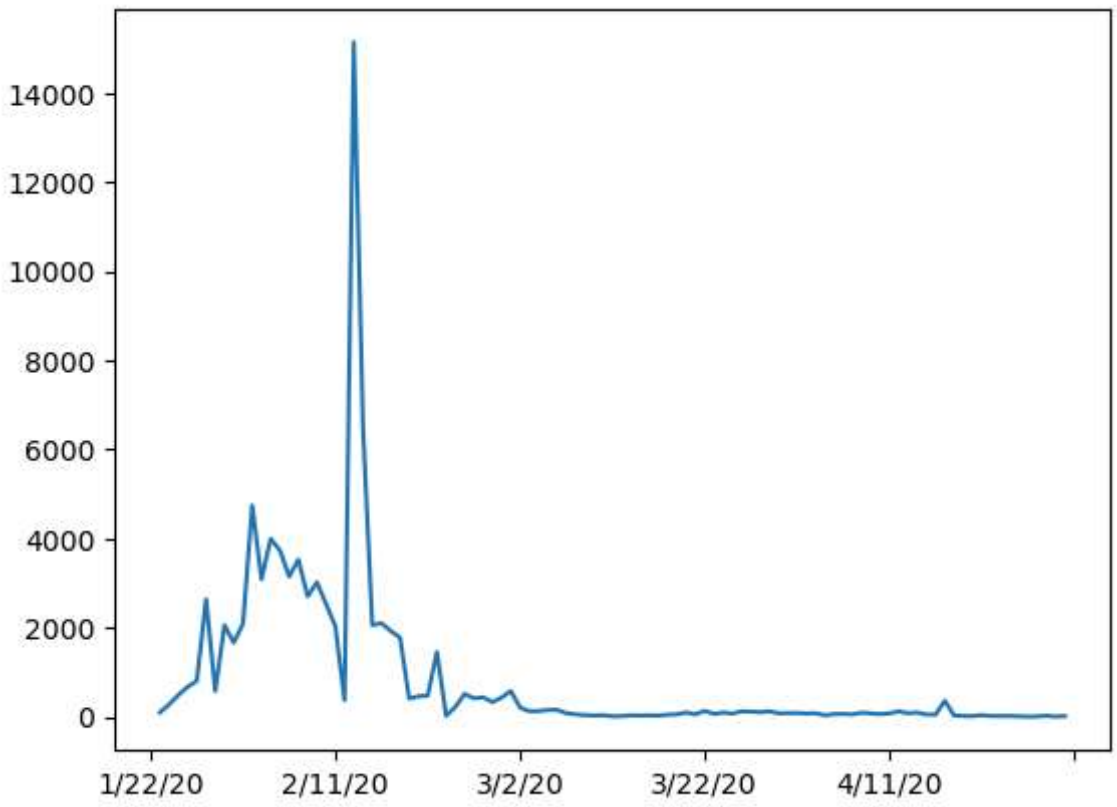
Out[15]: Text(0.5, 0, 'days')



3.1: caculating the first derivative of the curve

In [16]:
```
df_aggregated.loc['China'].diff().plot()
```

Out[16]: <AxesSubplot: >



In [17]:
```
# df_aggregated.loc['China']
```

3.2: find maxmimum infection rate for China

In [18]:
```
df_aggregated.loc['China'].diff().max()
```

Out[18]: 15136.0

In [19]:
```
df_aggregated.loc['Italy'].diff().max()
```

Out[19]: 6557.0

In [20]:
```
df_aggregated.loc['Spain'].diff().max()
```

Out[20]: 9630.0

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

In [21]:
```
countries = list(df_aggregated.index)
```

In [22]:
```
countries
```

Out[22]:
```
['Afghanistan',
 'Albania',
 'Algeria',
 'Andorra',
 'Angola',
 'Antigua and Barbuda',
 'Argentina',
 'Armenia',
 'Australia',
 'Austria',
 'Azerbaijan',
 'Bahamas',
 'Bahrain',
 'Bangladesh',
 'Barbados',
 'Belarus',
 'Belgium',
 'Belize',
 'Benin',
 'Bhutan',
 'Bolivia',
 'Bosnia and Herzegovina',
 'Botswana',
 'Brazil',
 'Brunei',
 'Bulgaria',
 'Burkina Faso',
 'Burma',
 'Burundi',
 'Cabo Verde',
 'Cambodia',
 'Cameroon',
 'Canada',
 'Central African Republic',
 'Chad',
 'Chile',
 'China',
 'Colombia',
 'Comoros',
 'Congo (Brazzaville)',
 'Congo (Kinshasa)',
 'Costa Rica',
 "Cote d'Ivoire",
 'Croatia',
 'Cuba',
 'Cyprus',
 'Czechia',
 'Denmark',
 'Diamond Princess',
 'Djibouti',
 'Dominica',
 'Dominican Republic',
 'Ecuador',
 'Egypt',
 'El Salvador',
 'Equatorial Guinea',
 'Eritrea',
 'Estonia',
 'Eswatini',
 'Ethiopia',
```

```
'Fiji',
'Finland',
'France',
'Gabon',
'Gambia',
'Georgia',
'Germany',
'Ghana',
'Greece',
'Grenada',
'Guatemala',
'Guinea',
'Guinea-Bissau',
'Guyana',
'Haiti',
'Holy See',
'Honduras',
'Hungary',
'Iceland',
'India',
'Indonesia',
'Iran',
'Iraq',
'Ireland',
'Israel',
'Italy',
'Jamaica',
'Japan',
'Jordan',
'Kazakhstan',
'Kenya',
'Korea, South',
'Kosovo',
'Kuwait',
'Kyrgyzstan',
'Laos',
'Latvia',
'Lebanon',
'Liberia',
'Libya',
'Liechtenstein',
'Lithuania',
'Luxembourg',
'MS Zaandam',
'Madagascar',
'Malawi',
'Malaysia',
'Maldives',
'Mali',
'Malta',
'Mauritania',
'Mauritius',
'Mexico',
'Moldova',
'Monaco',
'Mongolia',
'Montenegro',
'Morocco',
'Mozambique',
'Namibia',
```

```
'Nepal',
'Netherlands',
'New Zealand',
'Nicaragua',
'Niger',
'Nigeria',
'North Macedonia',
'Norway',
'Oman',
'Pakistan',
'Panama',
'Papua New Guinea',
'Paraguay',
'Peru',
'Philippines',
'Poland',
'Portugal',
'Qatar',
'Romania',
'Russia',
'Rwanda',
'Saint Kitts and Nevis',
'Saint Lucia',
'Saint Vincent and the Grenadines',
'San Marino',
'Sao Tome and Principe',
'Saudi Arabia',
'Senegal',
'Serbia',
'Seychelles',
'Sierra Leone',
'Singapore',
'Slovakia',
'Slovenia',
'Somalia',
'South Africa',
'South Sudan',
'Spain',
'Sri Lanka',
'Sudan',
'Suriname',
'Sweden',
'Switzerland',
'Syria',
'Taiwan*',
'Tajikistan',
'Tanzania',
'Thailand',
'Timor-Leste',
'Togo',
'Trinidad and Tobago',
'Tunisia',
'Turkey',
'US',
'Uganda',
'Ukraine',
'United Arab Emirates',
'United Kingdom',
'Uruguay',
'Uzbekistan',
```

```
 'Venezuela',
 'Vietnam',
 'West Bank and Gaza',
 'Western Sahara',
 'Yemen',
 'Zambia',
 'Zimbabwe']
```

In [23]:
```
max_infection_rates= []
for i in (countries):
    max_infection_rates.append(df_aggregated.loc[i].diff().max())
print(max_infection_rates)
```

```
[232.0, 34.0, 199.0, 43.0, 5.0, 6.0, 291.0, 134.0, 497.0, 1321.0, 105.0, 7.0, 30
1.0, 641.0, 12.0, 1485.0, 2454.0, 4.0, 19.0, 1.0, 104.0, 92.0, 7.0, 7502.0, 26.0,
137.0, 41.0, 21.0, 6.0, 45.0, 31.0, 203.0, 2778.0, 31.0, 21.0, 1138.0, 15136.0, 3
53.0, 1.0, 57.0, 81.0, 37.0, 113.0, 96.0, 63.0, 58.0, 381.0, 391.0, 99.0, 156.0,
5.0, 371.0, 11536.0, 269.0, 32.0, 130.0, 7.0, 134.0, 20.0, 9.0, 5.0, 267.0, 2684
9.0, 38.0, 5.0, 42.0, 6933.0, 403.0, 156.0, 6.0, 68.0, 167.0, 132.0, 12.0, 10.0,
3.0, 72.0, 210.0, 99.0, 1893.0, 436.0, 3186.0, 91.0, 1515.0, 1131.0, 6557.0, 52.
0, 1161.0, 40.0, 264.0, 29.0, 851.0, 289.0, 300.0, 69.0, 3.0, 48.0, 61.0, 17.0, 1
3.0, 21.0, 90.0, 234.0, 7.0, 14.0, 10.0, 235.0, 190.0, 58.0, 52.0, 2.0, 41.0, 142
5.0, 222.0, 12.0, 13.0, 30.0, 281.0, 19.0, 3.0, 14.0, 1346.0, 89.0, 2.0, 69.0, 20
8.0, 107.0, 386.0, 144.0, 1292.0, 357.0, 5.0, 27.0, 3683.0, 538.0, 545.0, 1516.0,
957.0, 523.0, 7099.0, 22.0, 5.0, 6.0, 4.0, 54.0, 6.0, 1351.0, 87.0, 2379.0, 2.0,
20.0, 1426.0, 114.0, 70.0, 73.0, 354.0, 28.0, 9630.0, 65.0, 67.0, 3.0, 812.0, 132
1.0, 6.0, 27.0, 15.0, 181.0, 188.0, 10.0, 14.0, 40.0, 82.0, 5138.0, 36188.0, 11.
0, 578.0, 552.0, 8733.0, 48.0, 167.0, 29.0, 19.0, 66.0, 4.0, 5.0, 9.0, 8.0]
```

In [24]:
```
max(max_infection_rates)
```

Out[24]:
```
36188.0
```

In [25]:
```
df_aggregated['max_infection_rates'] = max_infection_rates
```

In [26]:
```
df_aggregated
```

Out[26]:

Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29
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	
...	...	...	...	...	...	...	...	
West Bank and Gaza	0	0	0	0	0	0	0	
Western Sahara	0	0	0	0	0	0	0	
Yemen	0	0	0	0	0	0	0	
Zambia	0	0	0	0	0	0	0	
Zimbabwe	0	0	0	0	0	0	0	

187 rows × 101 columns

3.4: create a new dataframe with only needed column

In [27]:
```
corona_data = pd.DataFrame(df_aggregated.max_infection_rates)
```

In [28]:
```
corona_data
```

Out[28]:

Country/Region	max_infection_rates
Afghanistan	232.0
Albania	34.0
Algeria	199.0
Andorra	43.0
Angola	5.0
...	...
West Bank and Gaza	66.0
Western Sahara	4.0
Yemen	5.0
Zambia	9.0
Zimbabwe	8.0

187 rows × 1 columns

4:

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

4.1 : importing the dataset

In [29]:
```
happy_report_csv = pd.read_csv("Datasets/worldwide_happiness_report.csv")
```

In [30]:
```
happy_report_csv.head()
```

Out[30]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perc cor
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	

4.2: let's drop the useless columns

In [31]:
```
happy_report_csv.columns
```

Out[31]:
```
Index(['Overall rank', 'Country or region', 'Score', 'GDP per capita',
       'Social support', 'Healthy life expectancy',
       'Freedom to make life choices', 'Generosity',
       'Perceptions of corruption'],
      dtype='object')
```

In [32]:
```
useless_columns = ['Overall rank','Score'
                   , 'Generosity',
                     'Perceptions of corruption']
happy_report = happy_report_csv.drop(useless_columns , axis =1 )
happy_report
```

Out[32]:

	Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
0	Finland	1.340	1.587	0.986	0.596
1	Denmark	1.383	1.573	0.996	0.592
2	Norway	1.488	1.582	1.028	0.603
3	Iceland	1.380	1.624	1.026	0.591
4	Netherlands	1.396	1.522	0.999	0.557
...	...	...	...	...	...
151	Rwanda	0.359	0.711	0.614	0.555
152	Tanzania	0.476	0.885	0.499	0.417
153	Afghanistan	0.350	0.517	0.361	0.000
154	Central African Republic	0.026	0.000	0.105	0.225
155	South Sudan	0.306	0.575	0.295	0.010

156 rows × 5 columns

4.3: changing the indices of the dataframe

In [33]:
```
happy_report.set_index("Country or region", inplace = True)
```

In [34]:
```
happy_report.head()
```

Out[34]:

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592
Norway	1.488	1.582	1.028	0.603
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557

4.4: now let's join two dataset we have prepared

Corona Dataset :

In [35]:
```
corona_data
```

Out[35]:

Country/Region	max_infection_rates
Afghanistan	232.0
Albania	34.0
Algeria	199.0
Andorra	43.0
Angola	5.0
...	...
West Bank and Gaza	66.0
Western Sahara	4.0
Yemen	5.0
Zambia	9.0
Zimbabwe	8.0

187 rows × 1 columns

In [36]:
```
corona_data.shape
```

Out[36]:
```
(187, 1)
```

In [37]:
```
happy_report.head()
```

Out[37]:

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592
Norway	1.488	1.582	1.028	0.603
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557

In [38]:
```
happy_report.shape
```

Out[38]: (156, 4)

In [39]:
```
data = corona_data.join(happy_report,how ="inner")
```

In [40]:
```
# %whos
```

In [41]:
```
data.head()
```

Out[41]:

	max_infection_rates	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Afghanistan	232.0	0.350	0.517	0.361	0.000
Albania	34.0	0.947	0.848	0.874	0.383
Algeria	199.0	1.002	1.160	0.785	0.086
Argentina	291.0	1.092	1.432	0.881	0.471
Armenia	134.0	0.850	1.055	0.815	0.283

wolrd happiness report Dataset :

In [42]:
```
data.head()
```

Out[42]:

	max_infection_rates	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Afghanistan	232.0	0.350	0.517	0.361	0.000
Albania	34.0	0.947	0.848	0.874	0.383
Algeria	199.0	1.002	1.160	0.785	0.086
Argentina	291.0	1.092	1.432	0.881	0.471
Armenia	134.0	0.850	1.055	0.815	0.283

4.5: correlation matrix

In [43]:
```
data.corr()
```

Out[43]:

	max_infection_rates	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
max_infection_rates	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

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

In [44]:
```
data.head()
```

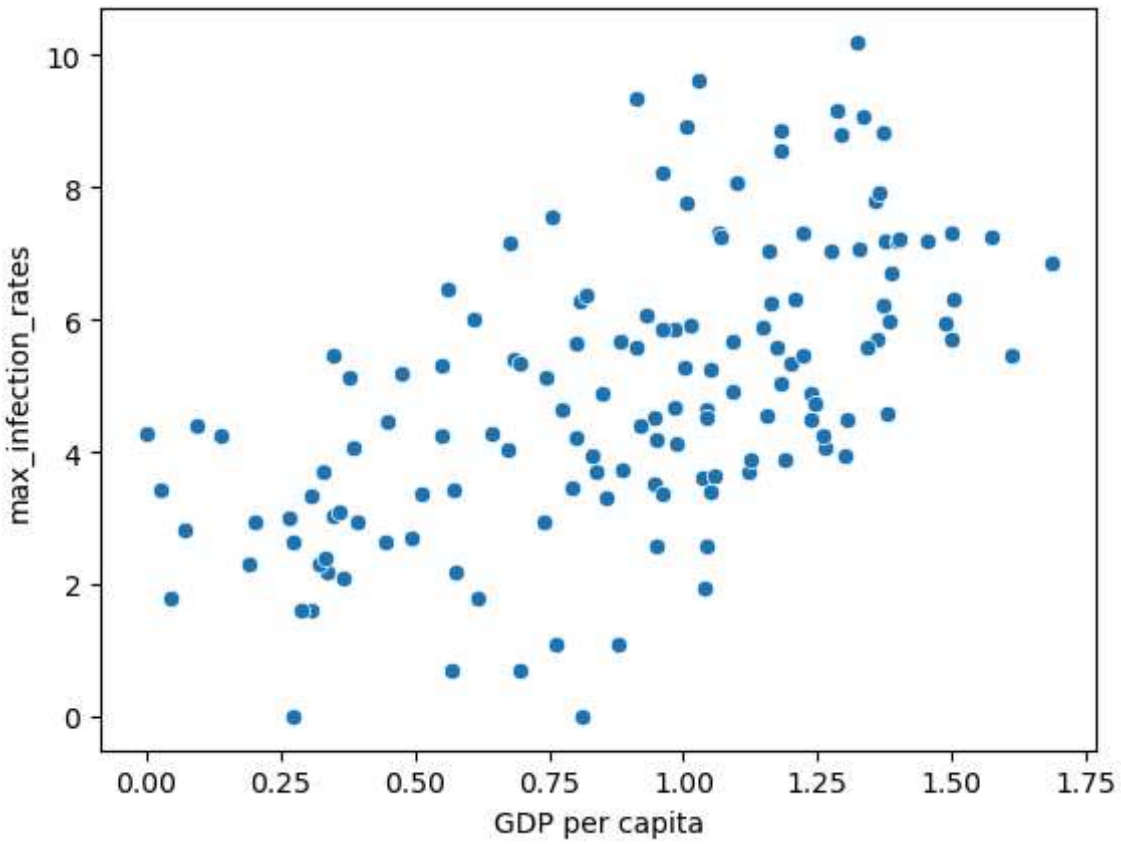
Out[44]:

	max_infection_rates	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Afghanistan	232.0	0.350	0.517	0.361	0.000
Albania	34.0	0.947	0.848	0.874	0.383
Algeria	199.0	1.002	1.160	0.785	0.086
Argentina	291.0	1.092	1.432	0.881	0.471
Armenia	134.0	0.850	1.055	0.815	0.283

5.1: Plotting GDP vs maximum Infection rate

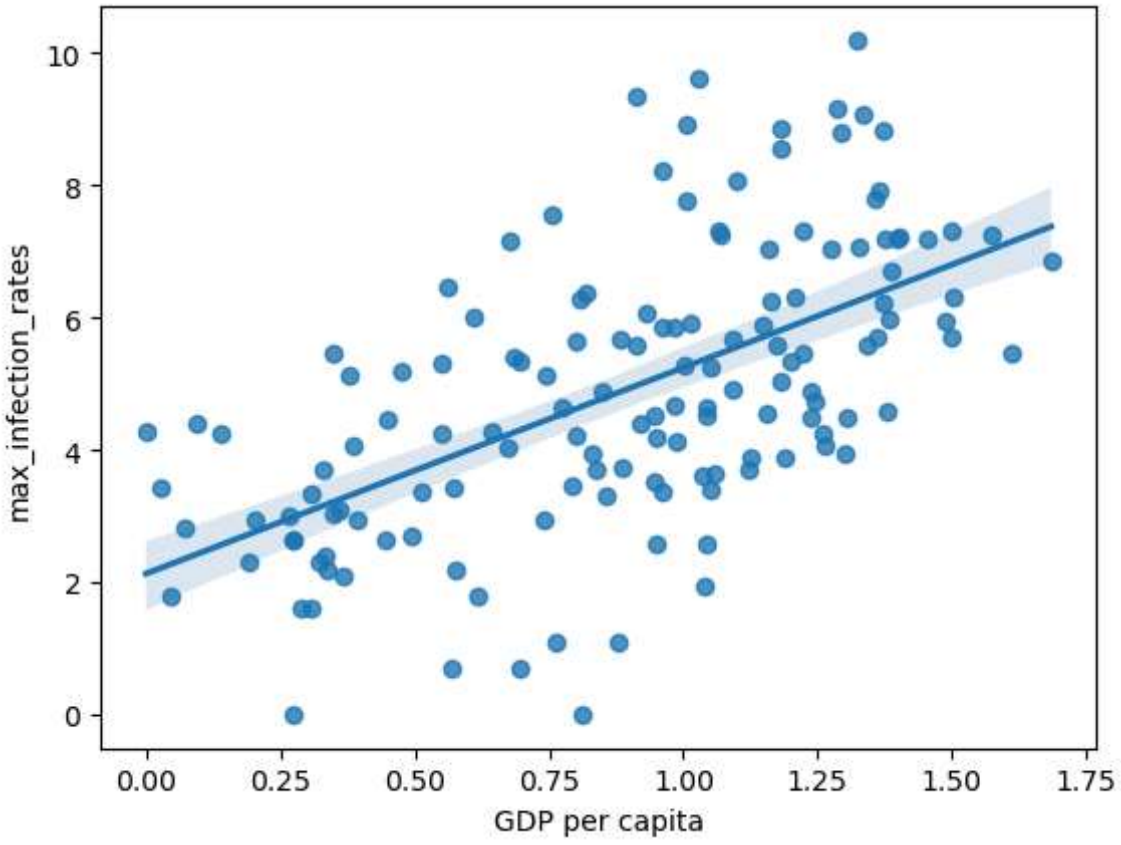
In [45]:
```
x = data["GDP per capita"]
y = data["max_infection_rates"]
sns.scatterplot(x=x, y=np.log(y))
```

Out[45]:
```
<AxesSubplot: xlabel='GDP per capita', ylabel='max_infection_rates'>
```



In [46]:
```
sns.regplot(x=x,y=np.log(y))
```

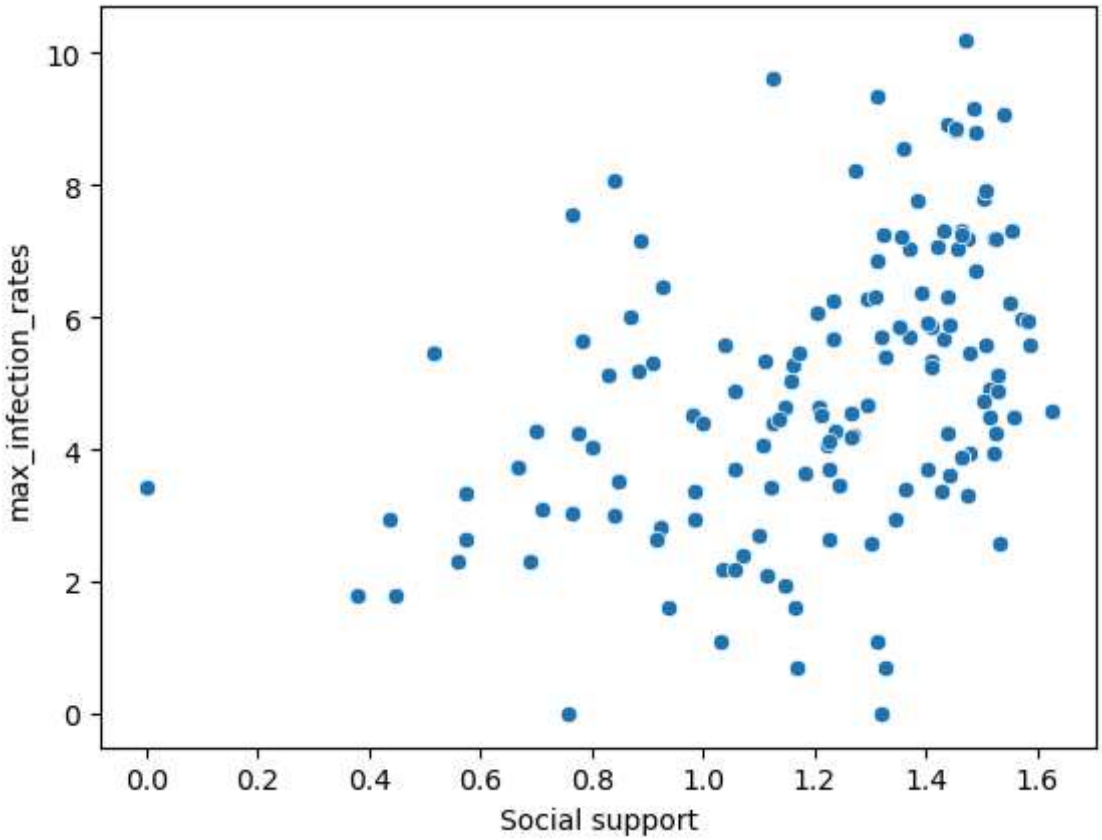
Out[46]: <AxesSubplot: xlabel='GDP per capita', ylabel='max_infection_rates'>



5.2: Plotting Social support vs maximum Infection rate

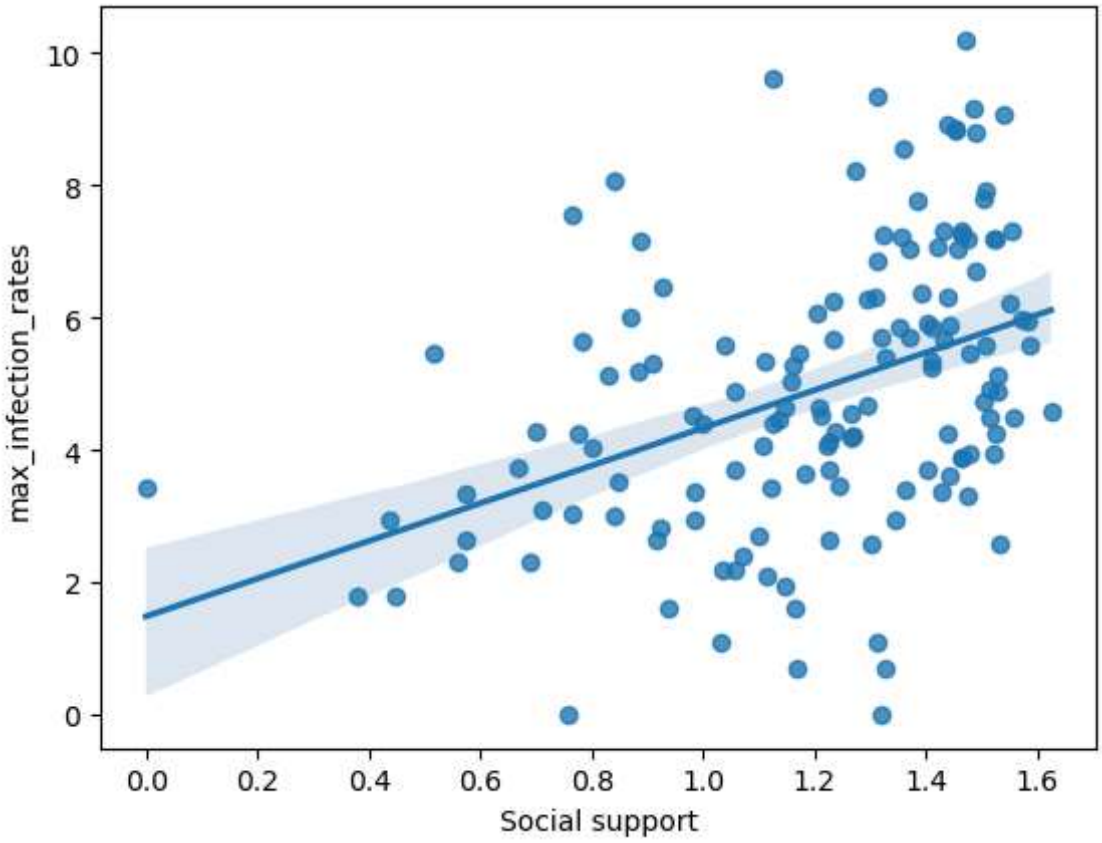
In [47]:
```
y = data["max_infection_rates"]
x = data["Social support"]
sns.scatterplot(x=x ,y=np.log(y))
```

Out[47]: <AxesSubplot: xlabel='Social support', ylabel='max_infection_rates'>



In [48]:
```
sns.regplot(x=x,y=np.log(y))
```

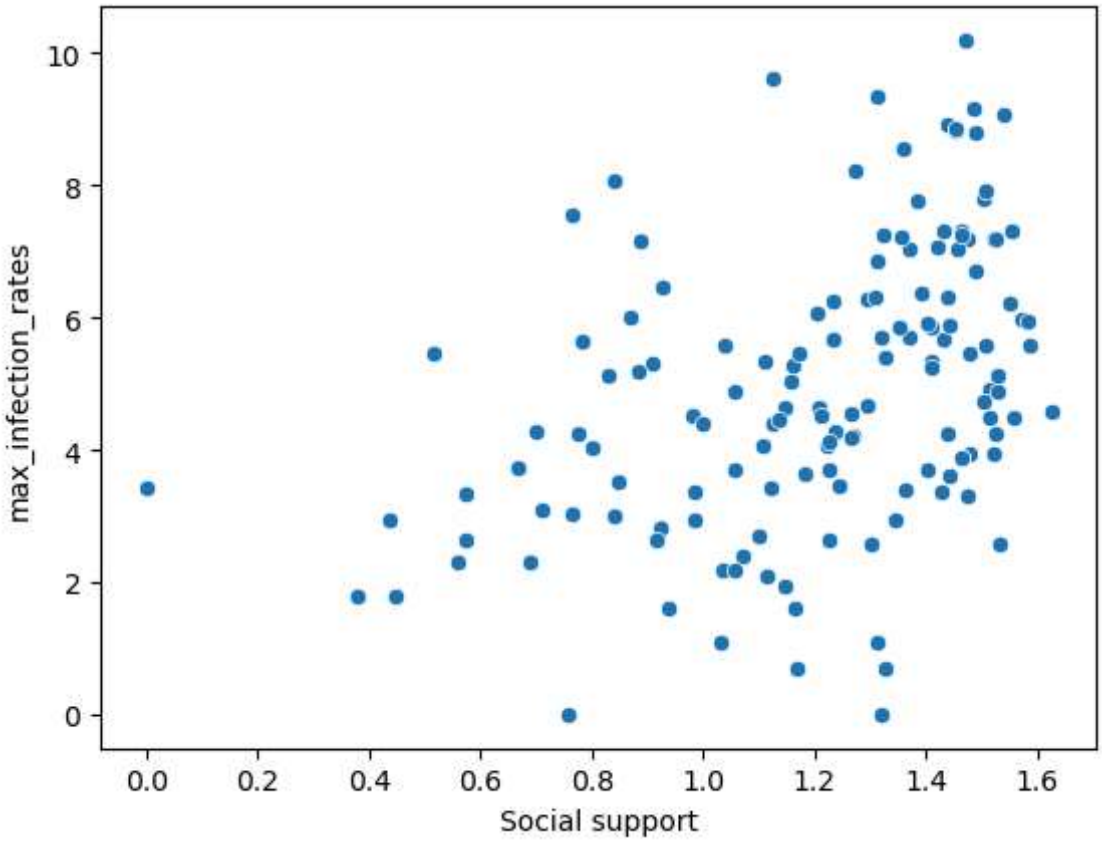
Out[48]: <AxesSubplot: xlabel='Social support', ylabel='max_infection_rates'>



5.3: Plotting Healthy life expectancy vs maximum Infection rate

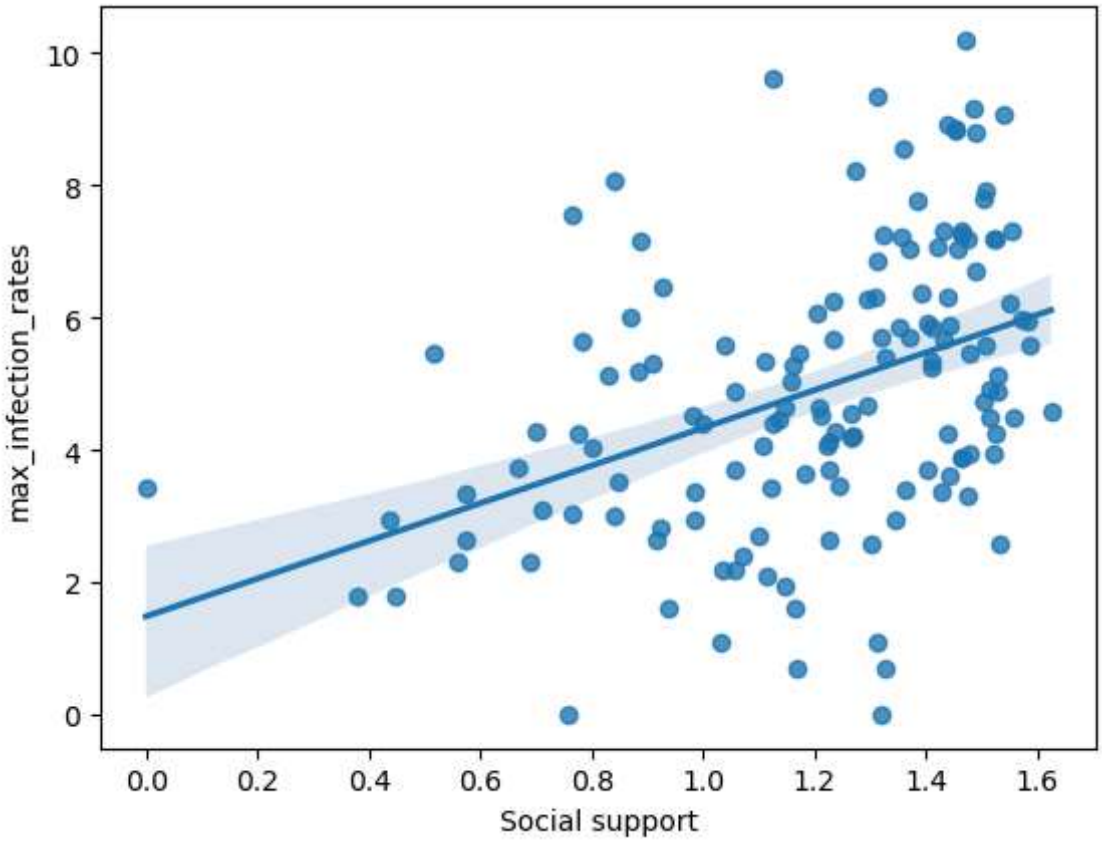
In [49]:
```
y = data["max_infection_rates"]
x = data["Social support"]
sns.scatterplot(x=x ,y=np.log(y))
```

Out[49]:
```
<AxesSubplot: xlabel='Social support', ylabel='max_infection_rates'>
```



In [50]:
```
sns.regplot(x=x,y=np.log(y))
```

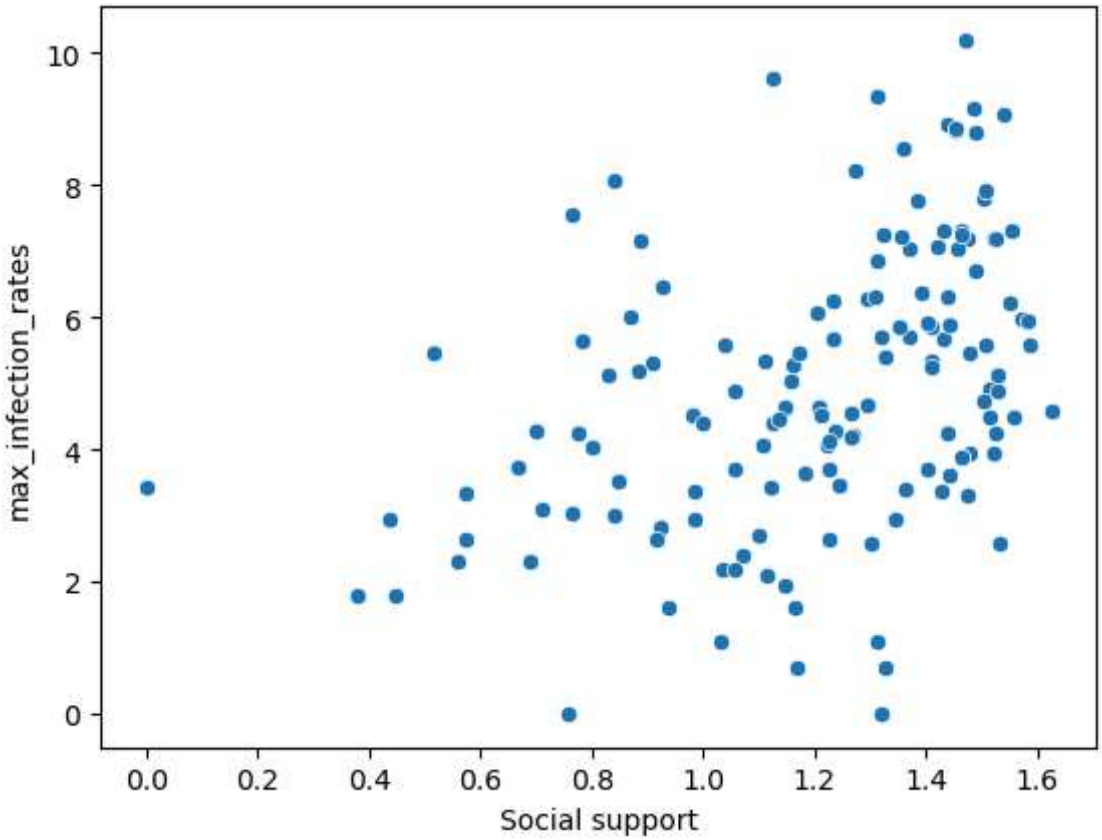
Out[50]:
```
<AxesSubplot: xlabel='Social support', ylabel='max_infection_rates'>
```



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

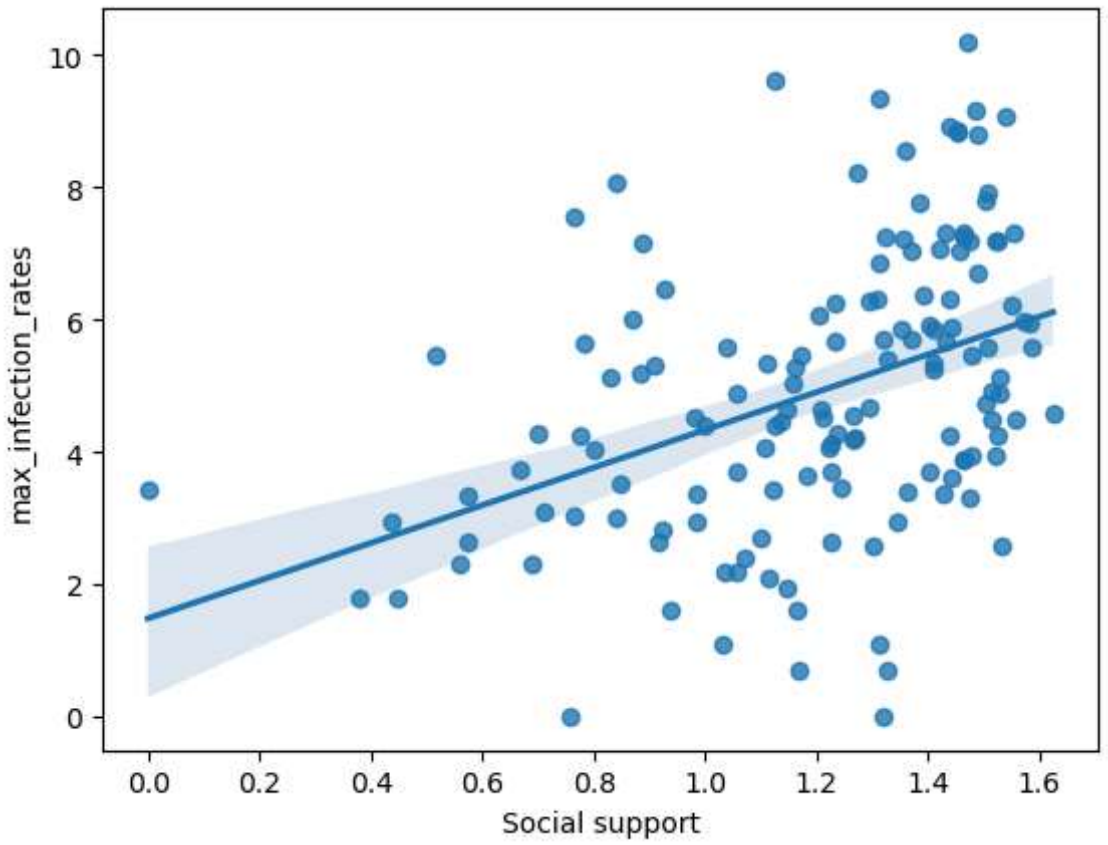
In [51]:
```
y = data["max_infection_rates"]
x = data["Social support"]
sns.scatterplot(x=x ,y=np.log(y))
```

Out[51]: <AxesSubplot: xlabel='Social support', ylabel='max_infection_rates'>



In [52]:
```
sns.regplot(x=x,y=np.log(y))
```

Out[52]: <AxesSubplot: xlabel='Social support', ylabel='max_infection_rates'>



In [53]:
```
covid_death_csv = pd.read_csv("Dataset for practice/covid19_deaths_dataset.csv")
```

In [54]:
```
covid_death_csv
```

Out[54]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/2
0	NaN	Afghanistan	33.000000	65.000000	0	0	
1	NaN	Albania	41.153300	20.168300	0	0	
2	NaN	Algeria	28.033900	1.659600	0	0	
3	NaN	Andorra	42.506300	1.521800	0	0	
4	NaN	Angola	-11.202700	17.873900	0	0	
...	...	...	...	...	...	...	
261	NaN	Western Sahara	24.215500	-12.885800	0	0	
262	NaN	Sao Tome and Principe	0.186360	6.613081	0	0	
263	NaN	Yemen	15.552727	48.516388	0	0	
264	NaN	Comoros	-11.645500	43.333300	0	0	
265	NaN	Tajikistan	38.861034	71.276093	0	0	

266 rows × 104 columns

In [66]:
```
covid_death_csv.drop(["Lat","Long","Province/State"], axis= 1 ,inplace = True)
```

In [68]:
```
covid_death_aggregated = covid_death_csv.groupby("Country/Region").sum()
```

In [71]:
```
covid_death_aggregated
```

Out[71]:

Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29
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	
...	...	...	...	...	...	...	...	
West Bank and Gaza	0	0	0	0	0	0	0	
Western Sahara	0	0	0	0	0	0	0	
Yemen	0	0	0	0	0	0	0	
Zambia	0	0	0	0	0	0	0	
Zimbabwe	0	0	0	0	0	0	0	

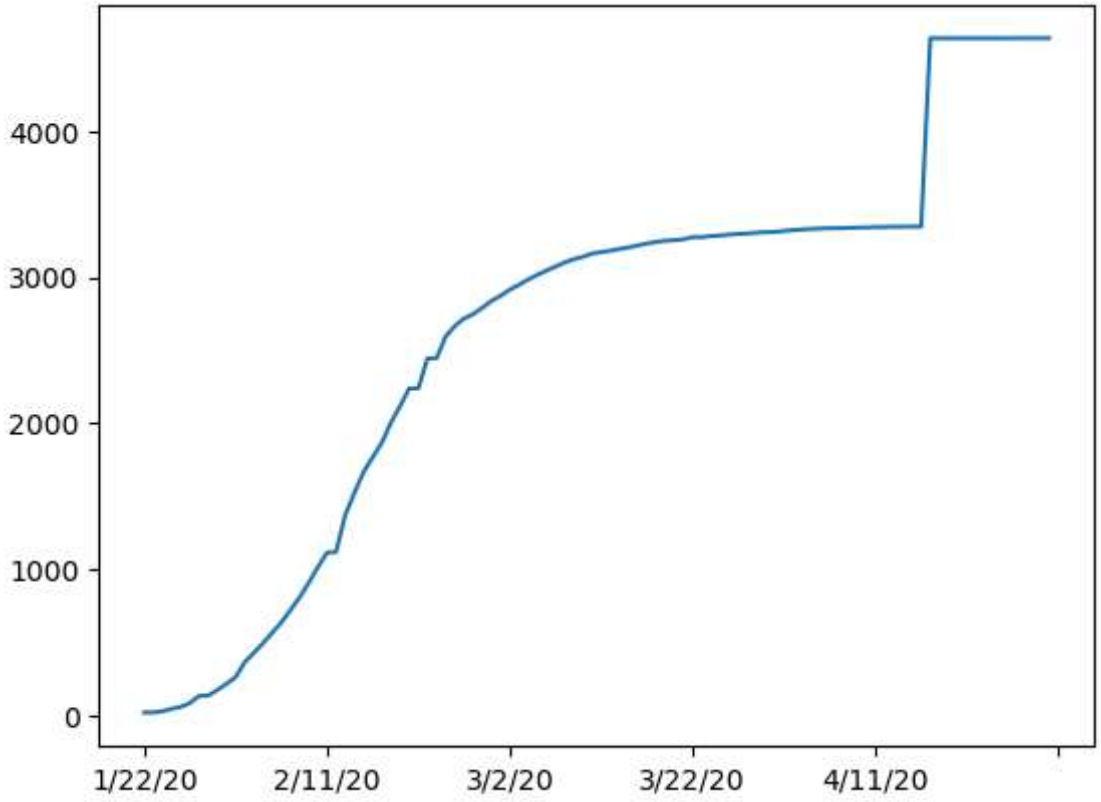
187 rows × 100 columns

In [72]:
```
covid_death_aggregated.loc["China"]
```

Out[72]:
```
1/22/20      17
1/23/20      18
1/24/20      26
1/25/20      42
1/26/20      56
            ...
4/26/20    4637
4/27/20    4637
4/28/20    4637
4/29/20    4637
4/30/20    4637
Name: China, Length: 100, dtype: int64
```

In [73]:
```
covid_death_aggregated.loc["China"].plot()
```

Out[73]:
```
<AxesSubplot: >
```

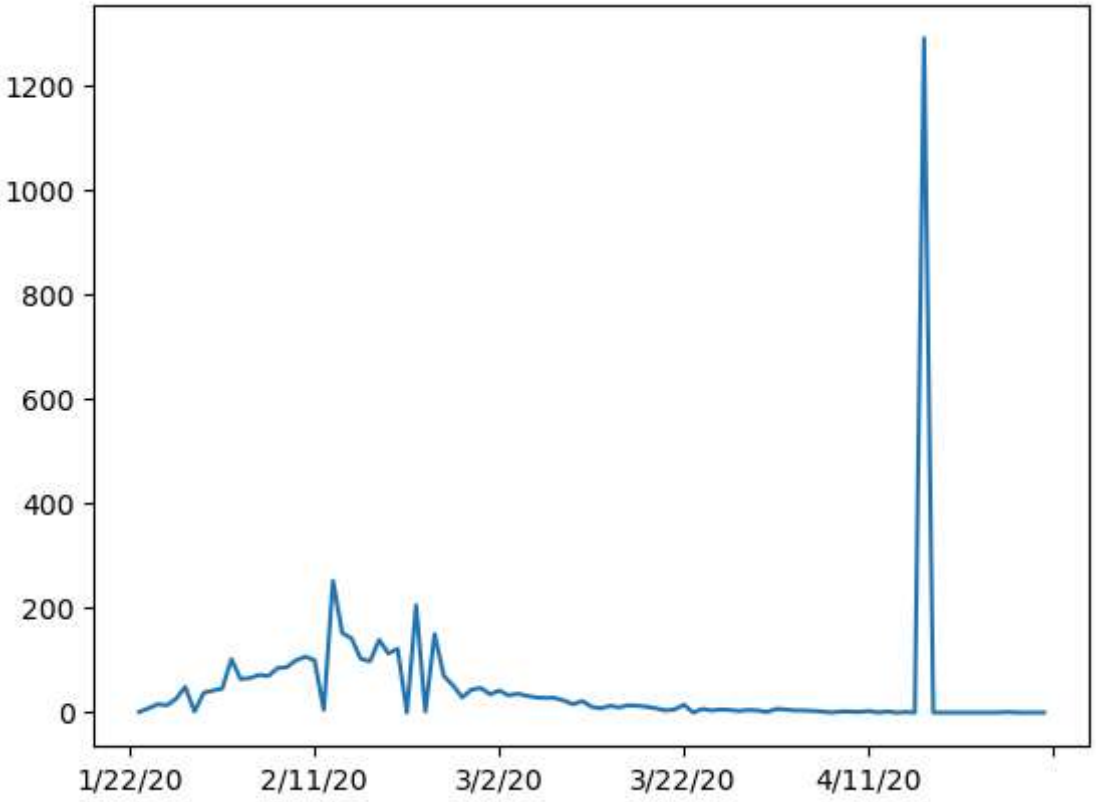


In [74]:
```
covid_death_aggregated.loc["China"].diff()
```

Out[74]:
```
1/22/20     NaN
1/23/20     1.0
1/24/20     8.0
1/25/20    16.0
1/26/20    14.0
           ...
4/26/20     1.0
4/27/20     0.0
4/28/20     0.0
4/29/20     0.0
4/30/20     0.0
Name: China, Length: 100, dtype: float64
```

In [75]:
```
covid_death_aggregated.loc["China"].diff().plot()
```

Out[75]:
```
<AxesSubplot: >
```



In [77]:
```
covid_death_aggregated.loc["China"].diff().max()
```

Out[77]: 1290.0

In [81]:
```
covid_death_aggregated.loc["Afghanistan"].diff().max()
```

Out[81]: 7.0

In [78]:
```
countries
```

Out[78]:
```
['Afghanistan',
 'Albania',
 'Algeria',
 'Andorra',
 'Angola',
 'Antigua and Barbuda',
 'Argentina',
 'Armenia',
 'Australia',
 'Austria',
 'Azerbaijan',
 'Bahamas',
 'Bahrain',
 'Bangladesh',
 'Barbados',
 'Belarus',
 'Belgium',
 'Belize',
 'Benin',
 'Bhutan',
 'Bolivia',
 'Bosnia and Herzegovina',
 'Botswana',
 'Brazil',
 'Brunei',
 'Bulgaria',
 'Burkina Faso',
 'Burma',
 'Burundi',
 'Cabo Verde',
 'Cambodia',
 'Cameroon',
 'Canada',
 'Central African Republic',
 'Chad',
 'Chile',
 'China',
 'Colombia',
 'Comoros',
 'Congo (Brazzaville)',
 'Congo (Kinshasa)',
 'Costa Rica',
 "Cote d'Ivoire",
 'Croatia',
 'Cuba',
 'Cyprus',
 'Czechia',
 'Denmark',
 'Diamond Princess',
 'Djibouti',
 'Dominica',
 'Dominican Republic',
 'Ecuador',
 'Egypt',
 'El Salvador',
 'Equatorial Guinea',
 'Eritrea',
 'Estonia',
 'Eswatini',
 'Ethiopia',
```

```
'Fiji',
'Finland',
'France',
'Gabon',
'Gambia',
'Georgia',
'Germany',
'Ghana',
'Greece',
'Grenada',
'Guatemala',
'Guinea',
'Guinea-Bissau',
'Guyana',
'Haiti',
'Holy See',
'Honduras',
'Hungary',
'Iceland',
'India',
'Indonesia',
'Iran',
'Iraq',
'Ireland',
'Israel',
'Italy',
'Jamaica',
'Japan',
'Jordan',
'Kazakhstan',
'Kenya',
'Korea, South',
'Kosovo',
'Kuwait',
'Kyrgyzstan',
'Laos',
'Latvia',
'Lebanon',
'Liberia',
'Libya',
'Liechtenstein',
'Lithuania',
'Luxembourg',
'MS Zaandam',
'Madagascar',
'Malawi',
'Malaysia',
'Maldives',
'Mali',
'Malta',
'Mauritania',
'Mauritius',
'Mexico',
'Moldova',
'Monaco',
'Mongolia',
'Montenegro',
'Morocco',
'Mozambique',
'Namibia',
```

```
'Nepal',
'Netherlands',
'New Zealand',
'Nicaragua',
'Niger',
'Nigeria',
'North Macedonia',
'Norway',
'Oman',
'Pakistan',
'Panama',
'Papua New Guinea',
'Paraguay',
'Peru',
'Philippines',
'Poland',
'Portugal',
'Qatar',
'Romania',
'Russia',
'Rwanda',
'Saint Kitts and Nevis',
'Saint Lucia',
'Saint Vincent and the Grenadines',
'San Marino',
'Sao Tome and Principe',
'Saudi Arabia',
'Senegal',
'Serbia',
'Seychelles',
'Sierra Leone',
'Singapore',
'Slovakia',
'Slovenia',
'Somalia',
'South Africa',
'South Sudan',
'Spain',
'Sri Lanka',
'Sudan',
'Suriname',
'Sweden',
'Switzerland',
'Syria',
'Taiwan*',
'Tajikistan',
'Tanzania',
'Thailand',
'Timor-Leste',
'Togo',
'Trinidad and Tobago',
'Tunisia',
'Turkey',
'US',
'Uganda',
'Ukraine',
'United Arab Emirates',
'United Kingdom',
'Uruguay',
'Uzbekistan',
```

```
 'Venezuela',
 'Vietnam',
 'West Bank and Gaza',
 'Western Sahara',
 'Yemen',
 'Zambia',
 'Zimbabwe']
```

In [83]:
```
max_death_rate = []
for c in countries:
    max_death_rate.append(covid_death_aggregated.loc[c].diff().max())
print(max_death_rate)
```

```
[7.0, 4.0, 30.0, 4.0, 2.0, 1.0, 13.0, 3.0, 8.0, 30.0, 3.0, 3.0, 1.0, 15.0, 1.0,
5.0, 496.0, 1.0, 1.0, 0.0, 6.0, 6.0, 1.0, 493.0, 1.0, 6.0, 4.0, 2.0, 1.0, 1.0, 0.
0, 20.0, 251.0, 0.0, 3.0, 13.0, 1290.0, 26.0, 0.0, 3.0, 5.0, 1.0, 4.0, 8.0, 6.0,
2.0, 18.0, 22.0, 2.0, 1.0, 0.0, 38.0, 208.0, 22.0, 1.0, 1.0, 0.0, 6.0, 1.0, 2.0,
0.0, 43.0, 1440.0, 1.0, 1.0, 1.0, 510.0, 5.0, 10.0, 0.0, 3.0, 2.0, 1.0, 2.0, 2.0,
0.0, 8.0, 23.0, 5.0, 75.0, 60.0, 158.0, 7.0, 220.0, 13.0, 919.0, 2.0, 47.0, 2.0,
5.0, 2.0, 11.0, 10.0, 4.0, 3.0, 0.0, 4.0, 2.0, 4.0, 1.0, 1.0, 6.0, 8.0, 2.0, 0.0,
1.0, 8.0, 1.0, 4.0, 1.0, 1.0, 2.0, 163.0, 10.0, 2.0, 0.0, 1.0, 12.0, 0.0, 0.0, 0.
0, 234.0, 4.0, 1.0, 4.0, 7.0, 6.0, 16.0, 2.0, 42.0, 11.0, 0.0, 2.0, 108.0, 50.0,
40.0, 37.0, 2.0, 34.0, 105.0, 0.0, 0.0, 0.0, 0.0, 6.0, 0.0, 9.0, 2.0, 54.0, 0.0,
3.0, 2.0, 4.0, 6.0, 8.0, 14.0, 0.0, 961.0, 1.0, 4.0, 1.0, 185.0, 75.0, 1.0, 3.0,
0.0, 6.0, 4.0, 0.0, 2.0, 2.0, 4.0, 127.0, 2612.0, 0.0, 19.0, 9.0, 1172.0, 2.0, 1.
0, 6.0, 0.0, 1.0, 0.0, 2.0, 1.0, 1.0]
```

In [84]:
```
covid_death_aggregated['max_death_rate'] = max_death_rate
```

In [86]:
```
covid_death = pd.DataFrame(covid_death_aggregated.max_death_rate)
```

In [87]:
```
covid_death
```

Out[87]:

Country/Region	max_death_rate
Afghanistan	7.0
Albania	4.0
Algeria	30.0
Andorra	4.0
Angola	2.0
...	...
West Bank and Gaza	1.0
Western Sahara	0.0
Yemen	2.0
Zambia	1.0
Zimbabwe	1.0

187 rows × 1 columns

In [91]:
```
covid_death.shape
```

Out[91]: (187, 1)

In [89]:
```
%who
```

```
c        c_df     corona_data     countries       covid_death     covid_death_aggr
egated   covid_death_csv          data     df
df_aggregated     happy_report    happy_report_csv        i       max_death_rate
max_infection_rates       np      pd      plt
sns      useless_columns          x       y
```

In [90]:
```
happy_report
```

Out[90]:

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592
Norway	1.488	1.582	1.028	0.603
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557
...	...	...	...	...
Rwanda	0.359	0.711	0.614	0.555
Tanzania	0.476	0.885	0.499	0.417
Afghanistan	0.350	0.517	0.361	0.000
Central African Republic	0.026	0.000	0.105	0.225
South Sudan	0.306	0.575	0.295	0.010

156 rows × 4 columns

In [93]:
```
happy_report.shape
```

Out[93]: (156, 4)

In [94]:
```
covid_death.join(happy_report,how=("inner"))
```

Out[94]:

	max_death_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Afghanistan	7.0	0.350	0.517	0.361	0.000
Albania	4.0	0.947	0.848	0.874	0.383
Algeria	30.0	1.002	1.160	0.785	0.086
Argentina	13.0	1.092	1.432	0.881	0.471
Armenia	3.0	0.850	1.055	0.815	0.283
...	...	...	...	...	...
Venezuela	6.0	0.960	1.427	0.805	0.154
Vietnam	0.0	0.741	1.346	0.851	0.543
Yemen	2.0	0.287	1.163	0.463	0.143
Zambia	1.0	0.578	1.058	0.426	0.431
Zimbabwe	1.0	0.366	1.114	0.433	0.361

143 rows × 5 columns

In [95]:
```
corona_death = covid_death.join(happy_report,how=("inner"))
```

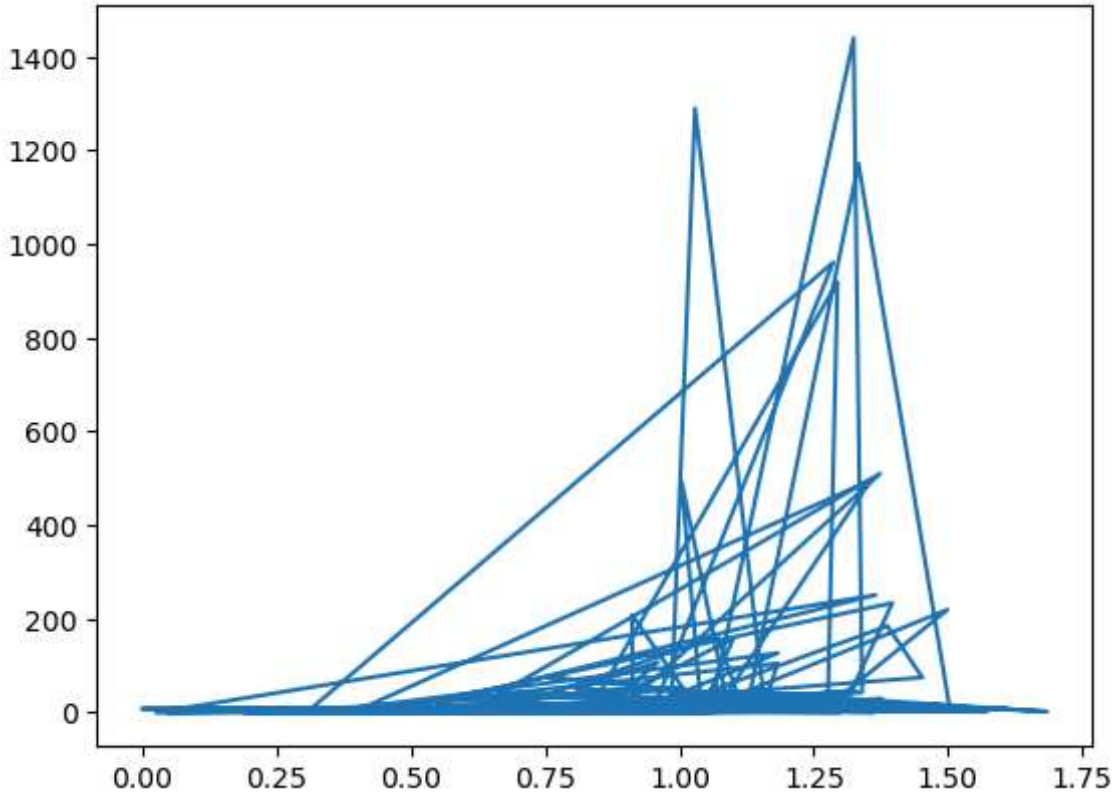
In [96]:
```
corona_death.corr()
```

Out[96]:

	max_death_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
max_death_rate	1.000000	0.259893	0.204148	0.309666	0.080166
GDP per capita	0.259893	1.000000	0.759468	0.863062	0.394603
Social support	0.204148	0.759468	1.000000	0.765286	0.456246
Healthy life expectancy	0.309666	0.863062	0.765286	1.000000	0.427892
Freedom to make life choices	0.080166	0.394603	0.456246	0.427892	1.000000

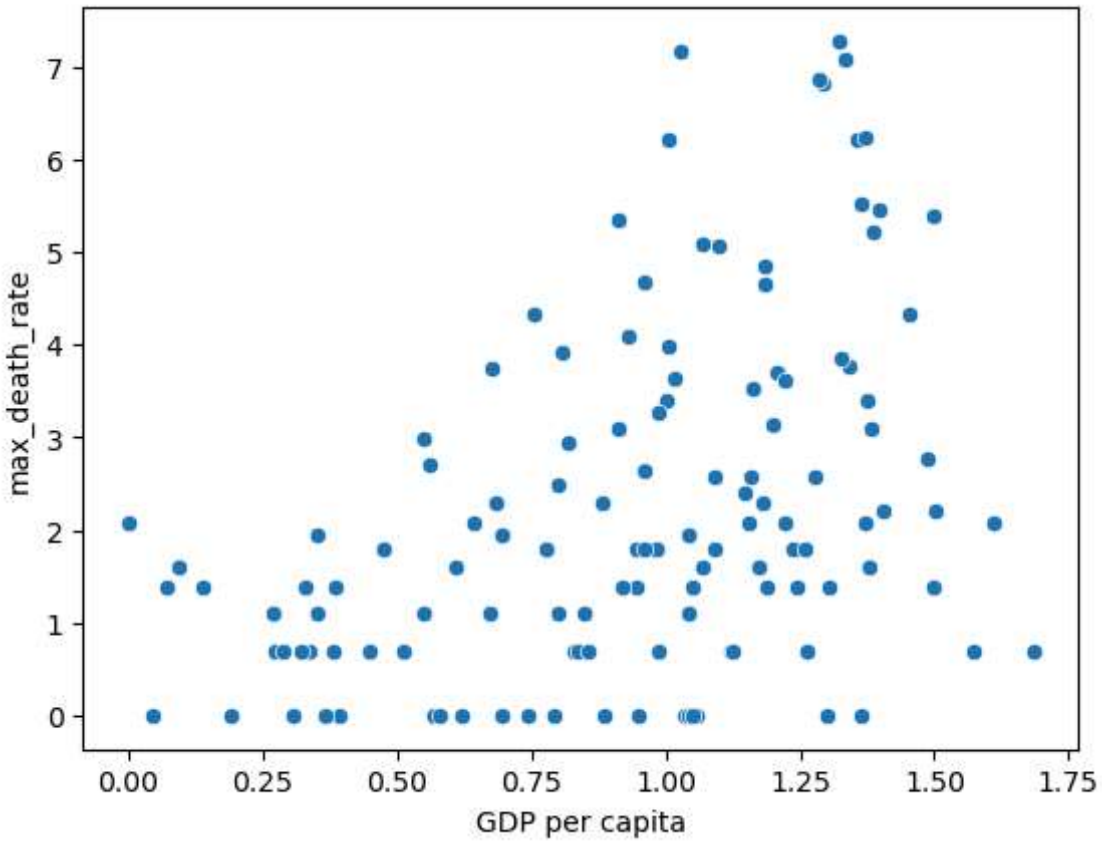
In [98]:
```
x = corona_death["GDP per capita"]
y = corona_death.max_death_rate
plt.plot(x,y)
```

Out[98]:
```
[<matplotlib.lines.Line2D at 0x70a069408220>]
```



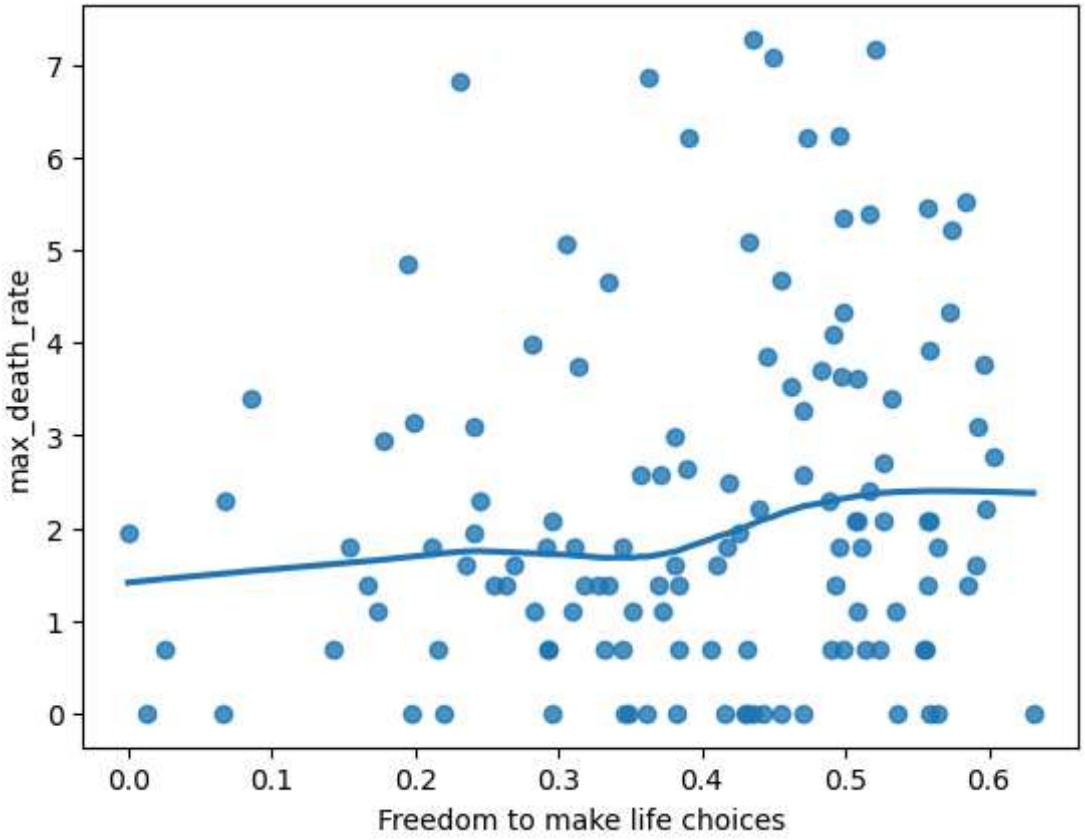
In [103…
```
x = corona_death["GDP per capita"]
y = corona_death["max_death_rate"]
sns.scatterplot(x=x,y=np.log(y))
```

Out[103… <AxesSubplot: xlabel='GDP per capita', ylabel='max_death_rate'>



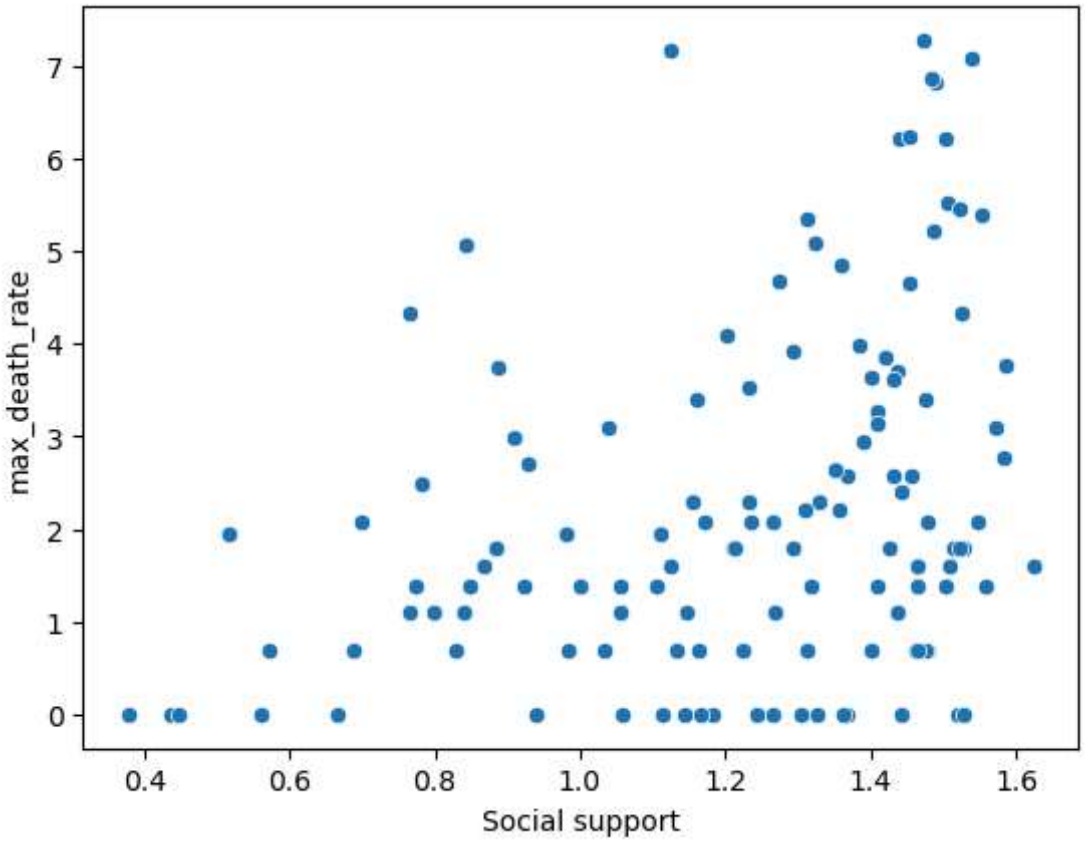
In [126…
```
sns.regplot(x=x,y=np.log(y), lowess=True)
```

Out[126… <AxesSubplot: xlabel='Freedom to make life choices', ylabel='max_death_rate'>



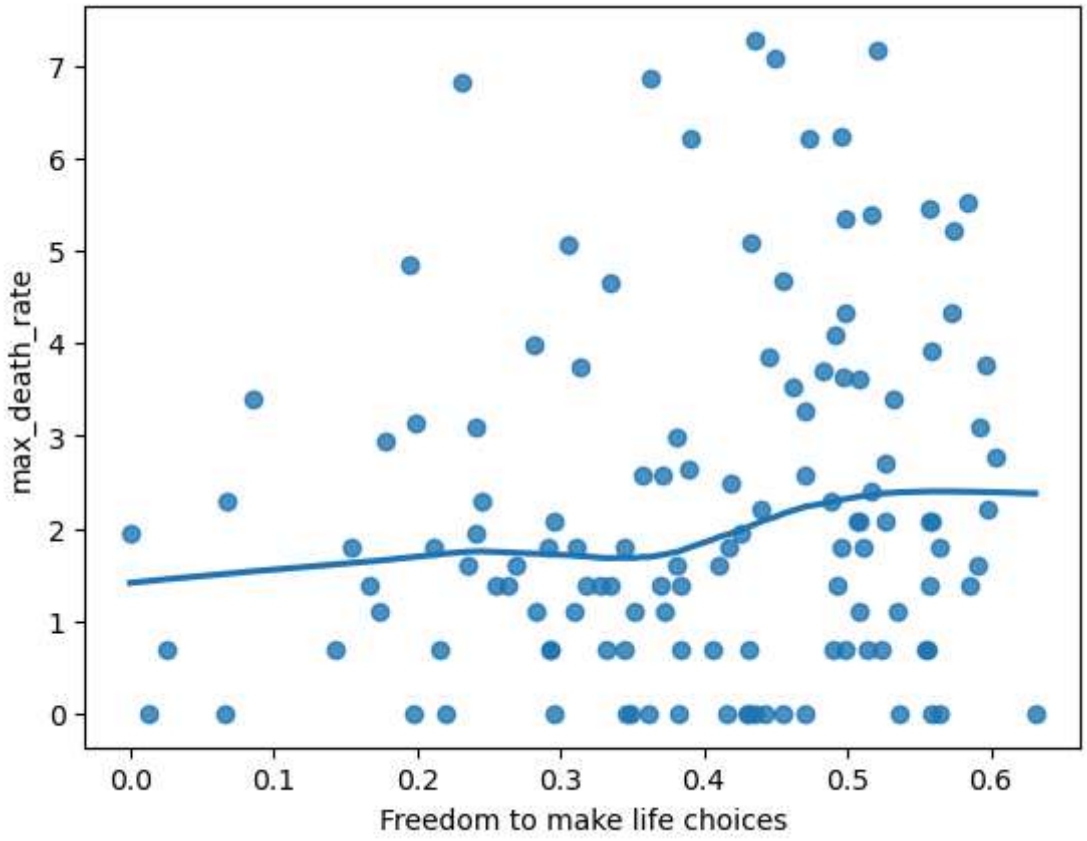
In [107…
```
x = corona_death["Social support"]
y = corona_death["max_death_rate"]
sns.scatterplot(x=x,y=np.log(y))
```

Out[107… <AxesSubplot: xlabel='Social support', ylabel='max_death_rate'>



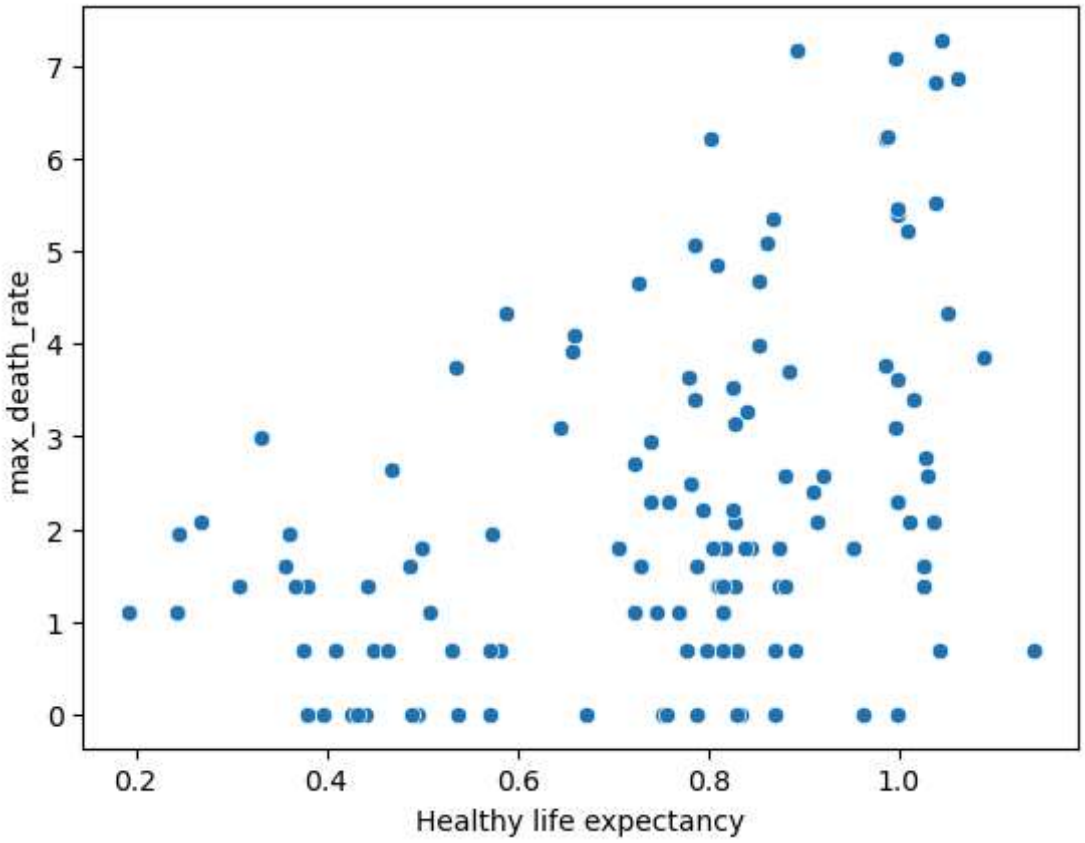
In [127…
```
sns.regplot(x=x,y=np.log(y),lowess=True)
```

Out[127…
```
<AxesSubplot: xlabel='Freedom to make life choices', ylabel='max_death_rate'>
```



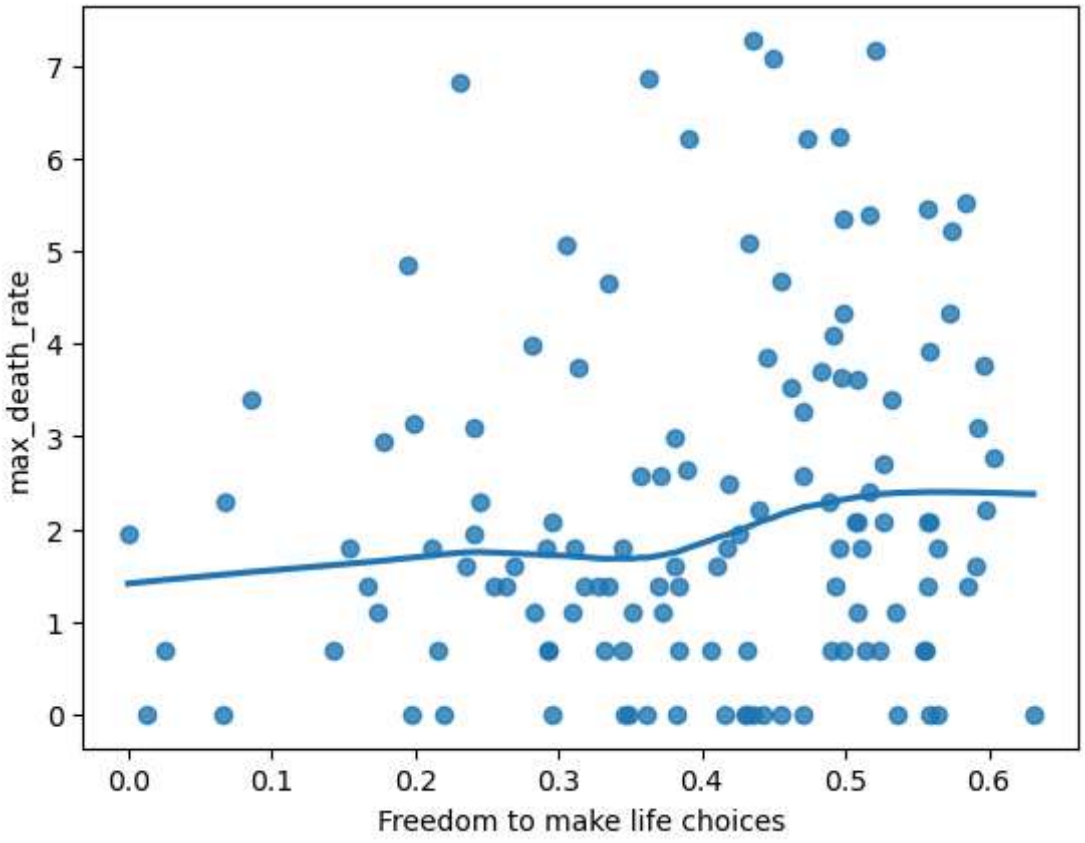
In [115…
```
x = corona_death["Healthy life expectancy"]
y = corona_death["max_death_rate"]
sns.scatterplot(x=x,y=np.log(y))
```

Out[115…
```
<AxesSubplot: xlabel='Healthy life expectancy', ylabel='max_death_rate'>
```



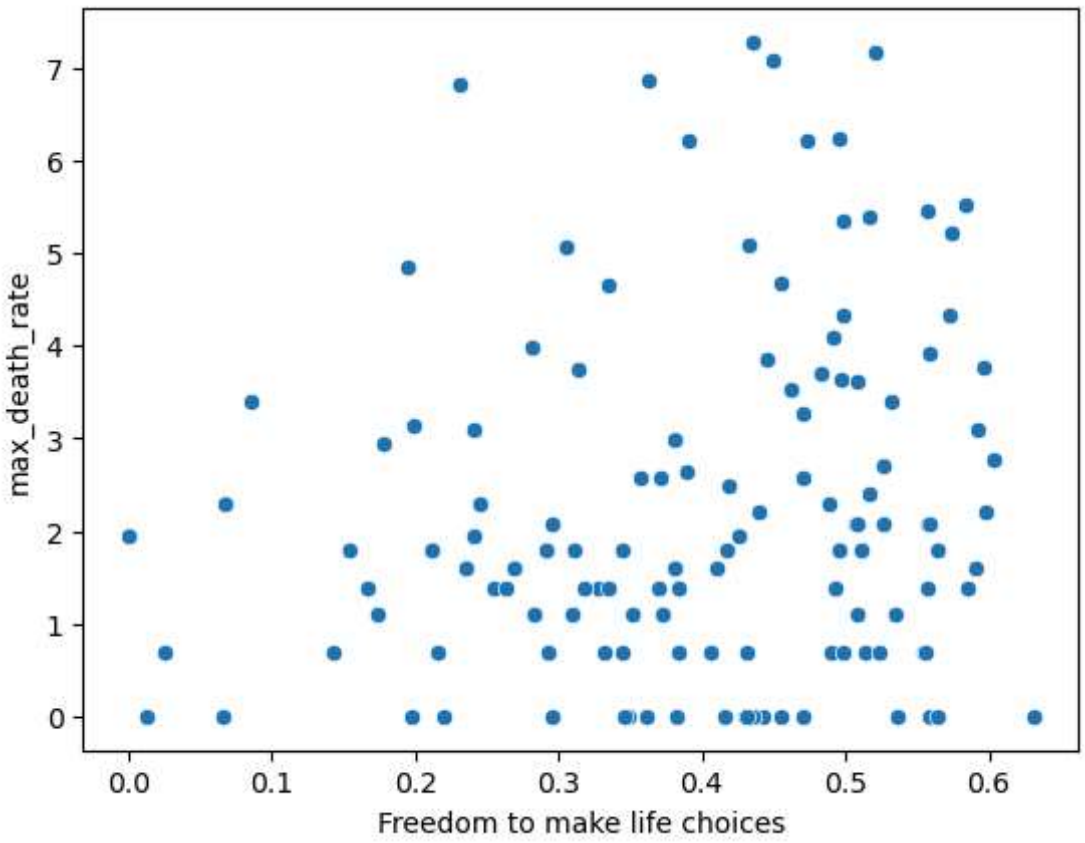
In [128…
```
sns.regplot(x=x,y=np.log(y),lowess=True)
```

Out[128…
```
<AxesSubplot: xlabel='Freedom to make life choices', ylabel='max_death_rate'>
```



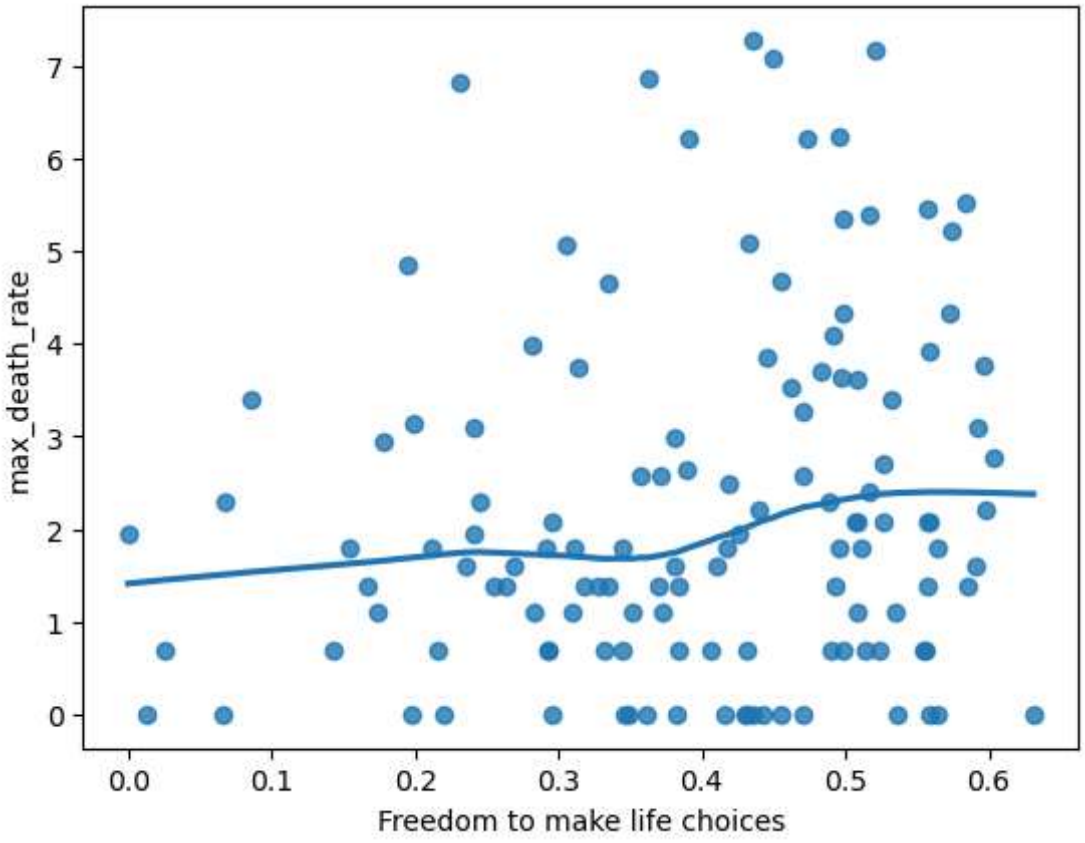
In [116…
```
x = corona_death["Freedom to make life choices"]
y = corona_death["max_death_rate"]
sns.scatterplot(x=x,y=np.log(y))
```

Out[116… <AxesSubplot: xlabel='Freedom to make life choices', ylabel='max_death_rate'>



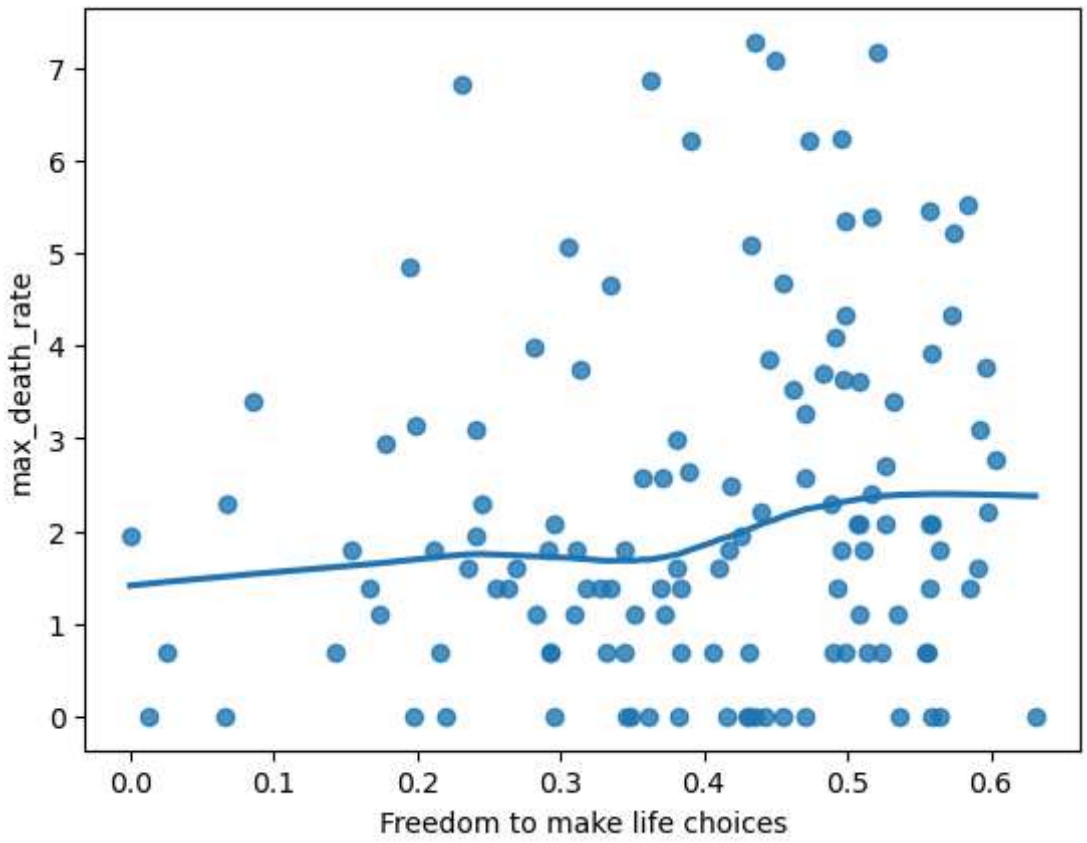
In [129…
```
sns.regplot(x=x,y=np.log(y),lowess=True)
```

Out[129… <AxesSubplot: xlabel='Freedom to make life choices', ylabel='max_death_rate'>



In [123…
```
sns.regplot(x=x, y=np.log(y), lowess=True)
```

Out[123… <AxesSubplot: xlabel='Freedom to make life choices', ylabel='max_death_rate'>



In [ ]: