Welcome to Covid19 Data Analysis Notebook

Let's Import the modules

In [120]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

print('Modules are imported.')
```

```
Modules are imported.
```

Task 2

Task 2.1: importing covid19 dataset

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

In [121]:
```
corona_dataset_csv = pd.read_csv('Dataset/covid19_Confirmed_dataset.csv')
corona_dataset_csv.head(10)
```

Out[121]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	4/21/20	4/22/20	4/23/20	4/24/20	4/25/20	4/26/2(
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0	0	...	1092	1176	1279	1351	1463	153
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0	0	...	609	634	663	678	712	72
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0	0	...	2811	2910	3007	3127	3256	338
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0	0	...	717	723	723	731	738	73
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0	0	...	24	25	25	25	25	2
5	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	0	0	...	23	24	24	24	24	2
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	0	0	0	...	3031	3144	3435	3607	3780	389
7	NaN	Armenia	40.0691	45.0382	0	0	0	0	0	0	...	1401	1473	1523	1596	1677	174
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	0	0	0	...	104	104	104	105	106	10
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	0	3	4	...	2969	2971	2976	2982	2994	300

10 rows × 104 columns

Let's check the shape of the dataframe

In [122]:
```
corona_dataset_csv.shape
```

Out[122]: `(266, 104)`

Task 2.2: Delete the useless columns

In [123]:
```
corona_dataset_csv.drop(['Lat','Long'],axis=1,inplace=True)
```

In [124]:
```
corona_dataset_csv.head(10)
```

Out[124]:

	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
0	NaN	Afghanistan	0	0	0	0	0	0	0	0	...	1092	1176	1279	1351	1463	1531
1	NaN	Albania	0	0	0	0	0	0	0	0	...	609	634	663	678	712	726
2	NaN	Algeria	0	0	0	0	0	0	0	0	...	2811	2910	3007	3127	3256	3382
3	NaN	Andorra	0	0	0	0	0	0	0	0	...	717	723	723	731	738	738
4	NaN	Angola	0	0	0	0	0	0	0	0	...	24	25	25	25	25	26
5	NaN	Antigua and Barbuda	0	0	0	0	0	0	0	0	...	23	24	24	24	24	24
6	NaN	Argentina	0	0	0	0	0	0	0	0	...	3031	3144	3435	3607	3780	3892
7	NaN	Armenia	0	0	0	0	0	0	0	0	...	1401	1473	1523	1596	1677	1746
8	Australian Capital Territory	Australia	0	0	0	0	0	0	0	0	...	104	104	104	105	106	106
9	New South Wales	Australia	0	0	0	0	3	4	4	4	...	2969	2971	2976	2982	2994	3002

10 rows × 102 columns

Task 2.3: Aggregating the rows by the country

In [125]:
```
corona_dataset_aggregated = corona_dataset_csv.groupby("Country/Region").sum()
```

In [126]:
```
corona_dataset_aggregated.head(10)
```

Out[126]:

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	1/30/20	1/31/20	...	4/21/20	4/22/20	4/23/20	4/24/20	4/25/20	4/26/20	4/
Afghanistan	0	0	0	0	0	0	0	0	0	0	...	1092	1176	1279	1351	1463	1531	
Albania	0	0	0	0	0	0	0	0	0	0	...	609	634	663	678	712	726	
Algeria	0	0	0	0	0	0	0	0	0	0	...	2811	2910	3007	3127	3256	3382	
Andorra	0	0	0	0	0	0	0	0	0	0	...	717	723	723	731	738	738	
Angola	0	0	0	0	0	0	0	0	0	0	...	24	25	25	25	25	26	
Antigua and Barbuda	0	0	0	0	0	0	0	0	0	0	...	23	24	24	24	24	24	
Argentina	0	0	0	0	0	0	0	0	0	0	...	3031	3144	3435	3607	3780	3892	
Armenia	0	0	0	0	0	0	0	0	0	0	...	1401	1473	1523	1596	1677	1746	
Australia	0	0	0	0	4	5	5	6	9	9	...	6645	6652	6662	6677	6694	6714	
Austria	0	0	0	0	0	0	0	0	0	0	...	14873	14925	15002	15071	15148	15225	

10 rows × 100 columns

In [127]:
```
corona_dataset_aggregated.shape
```

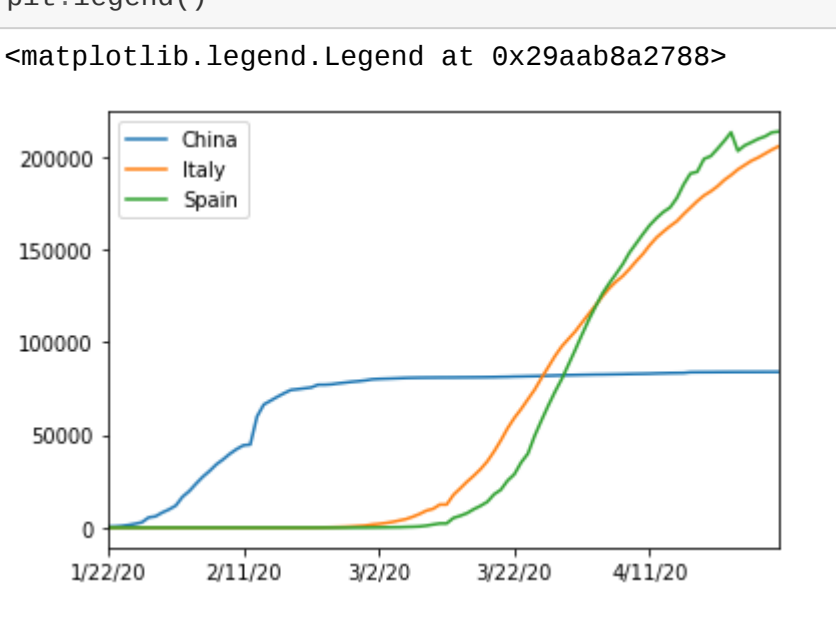
Out[127]: `(187, 100)`

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

visualization always helps for better understanding of our data.

In [128]:
```
corona_dataset_aggregated.loc['China'].plot()
corona_dataset_aggregated.loc['Italy'].plot()
corona_dataset_aggregated.loc['Spain'].plot()
plt.legend()
```

Out[128]: `<matplotlib.legend.Legend at 0x29aab8a2788>`

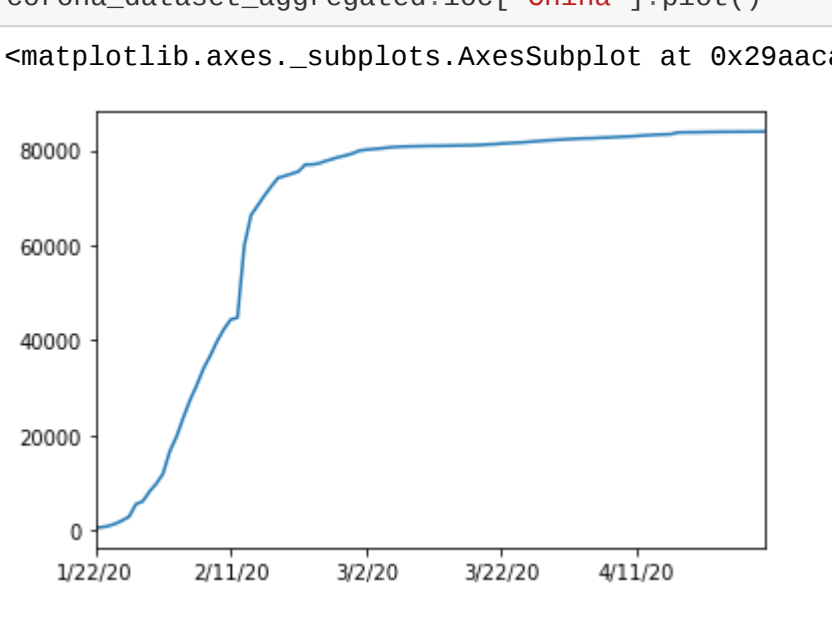


Task3: Calculating a good measure

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

In [129]:
```
corona_dataset_aggregated.loc['China'].plot()
```

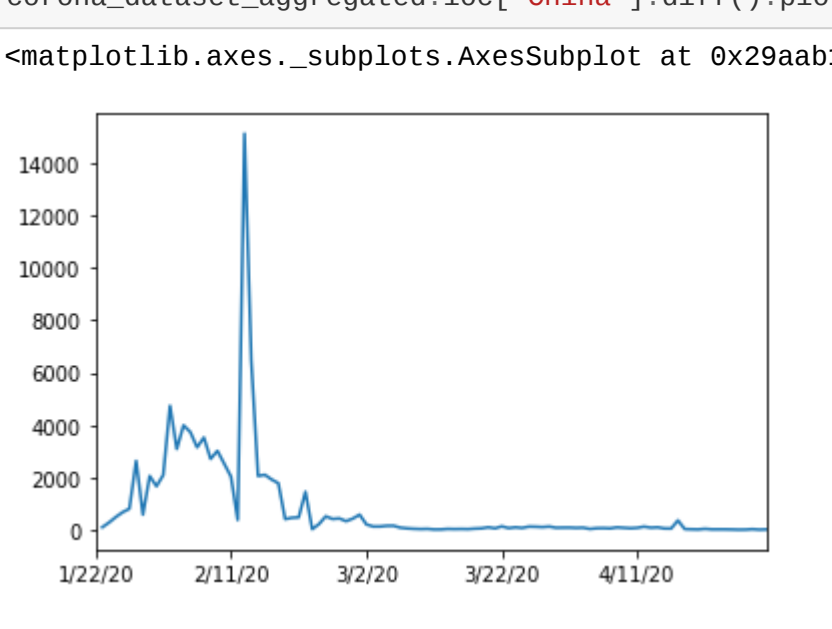
Out[129]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aacac1ec8>`



task 3.1: caculating the first derivative of the curve

In [130]:
```
corona_dataset_aggregated.loc['China'].diff().plot()
```

Out[130]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aab141908>`



task 3.2: find maxmimum infection rate for China

In [131]:
```
corona_dataset_aggregated.loc['China'].diff().max()
```

Out[131]: `15136.0`

In [132]:
```
corona_dataset_aggregated.loc['Italy'].diff().max()
```

Out[132]: `6557.0`

In [133]:
```
corona_dataset_aggregated.loc['Spain'].diff().max()
```

Out[133]: `9630.0`

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

In [134]:
```
countries = list(corona_dataset_aggregated.index)
max_infection_rates = []
for country in countries :
    max_infection_rates.append(corona_dataset_aggregated.loc[country].diff().max())
corona_dataset_aggregated['max infection rate'] = max_infection_rates
```

In [135]:
```
corona_dataset_aggregated.head()
```

Out[135]:

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	1/30/20	1/31/20	...	4/22/20	4/23/20	4/24/20	4/25/20	4/26/20	4/27/20	4/
Afghanistan	0	0	0	0	0	0	0	0	0	0	...	1176	1279	1351	1463	1531	1703	
Albania	0	0	0	0	0	0	0	0	0	0	...	634	663	678	712	726	736	
Algeria	0	0	0	0	0	0	0	0	0	0	...	2910	3007	3127	3256	3382	3517	
Andorra	0	0	0	0	0	0	0	0	0	0	...	723	723	731	738	738	743	
Angola	0	0	0	0	0	0	0	0	0	0	...	25	25	25	25	26	27	

5 rows × 101 columns

Task 3.4: create a new dataframe with only needed column

In [136]:
```
corona_data = pd.DataFrame(corona_dataset_aggregated['max infection rate'])
```

In [137]:
```
corona_data.head()
```

Out[137]:

Country/Region	max infection rate
Afghanistan	232.0
Albania	34.0
Algeria	199.0
Andorra	43.0
Angola	5.0

Task4:

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

Task 4.1 : importing the dataset

In [138]:
```
world_happiness_report = pd.read_csv("Dataset/worldwide_happiness_report.csv")
world_happiness_report.head()
```

Out[138]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298

In [139]:
```
world_happiness_report.shape
```

Out[139]: `(156, 9)`

Task 4.2: let's drop the useless columns

In [140]:
```
columns_to_dropped = ['Overall rank','Score','Generosity','Perceptions of corruption']
world_happiness_report.drop(columns_to_dropped,axis=1 , inplace=True)
```

In [141]:
```
world_happiness_report.head()
```

Out[141]:

	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

Task 4.3: changing the indices of the dataframe

In [142]:
```
world_happiness_report.set_index(['Country or region'],inplace=True)
world_happiness_report.head()
```

Out[142]:

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

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

Corona Dataset :

In [143]:
```
corona_data.head()
```

Out[143]:

Country/Region	max infection rate
Afghanistan	232.0
Albania	34.0
Algeria	199.0
Andorra	43.0
Angola	5.0

wolrd happiness report Dataset :

In [144]:
```
world_happiness_report.head()
```

Out[144]:

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 [145]:
```
data = world_happiness_report.join(corona_data).copy()
data.head()
```

Out[145]:

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	max infection rate
Finland	1.340	1.587	0.986	0.596	267.0
Denmark	1.383	1.573	0.996	0.592	391.0
Norway	1.488	1.582	1.028	0.603	386.0
Iceland	1.380	1.624	1.026	0.591	99.0
Netherlands	1.396	1.522	0.999	0.557	1346.0

Task 4.5: correlation matrix

In [146]:
```
data.corr()
# it is representing the currelation between every two columns of our dataset
```

Out[146]:

	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	max infection rate
GDP per capita	1.000000	0.754906	0.835462	0.379079	0.250118
Social support	0.754906	1.000000	0.719009	0.447333	0.191958
Healthy life expectancy	0.835462	0.719009	1.000000	0.390395	0.289263
Freedom to make life choices	0.379079	0.447333	0.390395	1.000000	0.078196
max infection rate	0.250118	0.191958	0.289263	0.078196	1.000000

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

In [147]:
```
data.head()
```

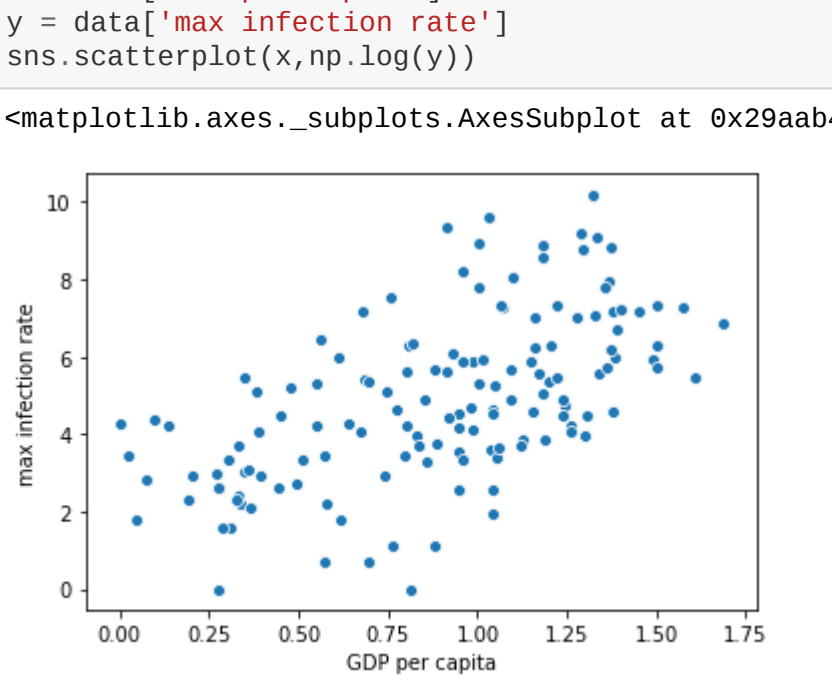
Out[147]:

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	max infection rate
Finland	1.340	1.587	0.986	0.596	267.0
Denmark	1.383	1.573	0.996	0.592	391.0
Norway	1.488	1.582	1.028	0.603	386.0
Iceland	1.380	1.624	1.026	0.591	99.0
Netherlands	1.396	1.522	0.999	0.557	1346.0

Task 5.1: Plotting GDP vs maximum Infection rate

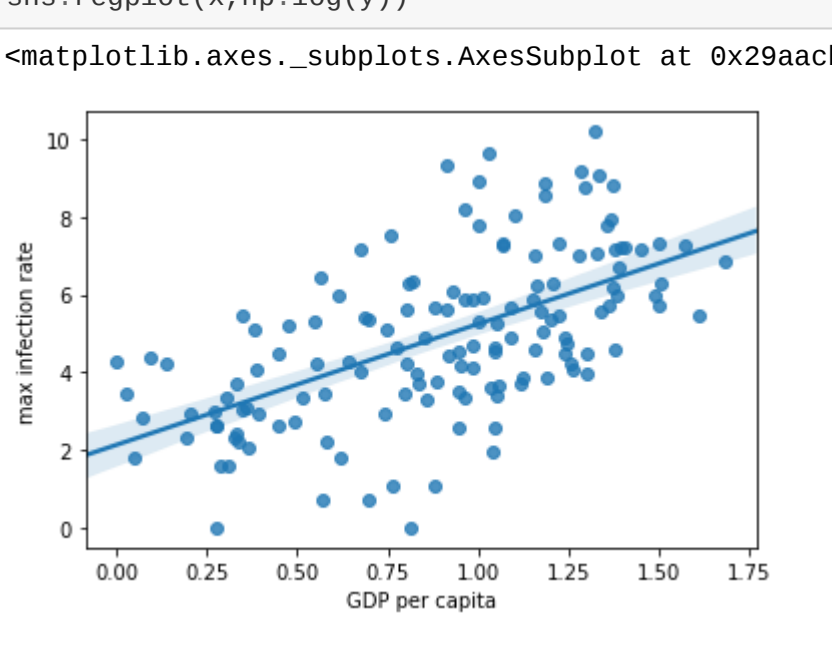
In [148]:
```
x = data['GDP per capita']
y = data['max infection rate']
sns.scatterplot(x,np.log(y))
```

Out[148]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aab4fcd88>`



In [149]:
```
sns.regplot(x,np.log(y))
```

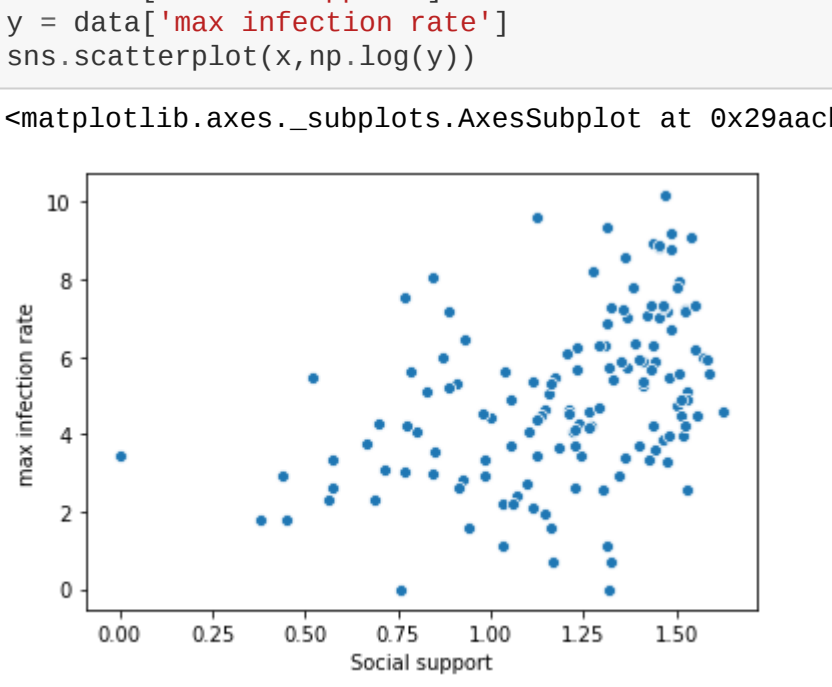
Out[149]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aacb83788>`



Task 5.2: Plotting Social support vs maximum Infection rate

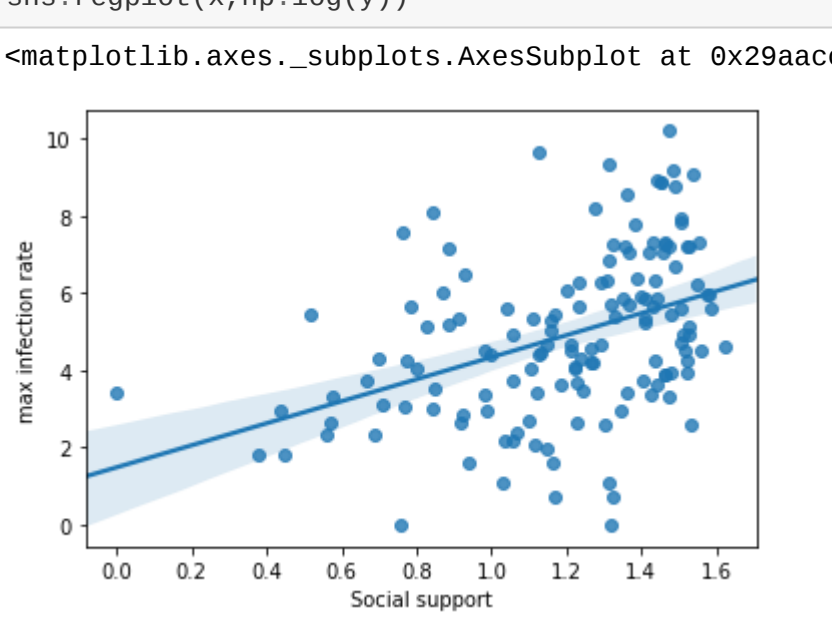
In [150]:
```
x = data['Social support']
y = data['max infection rate']
sns.scatterplot(x,np.log(y))
```

Out[150]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aacbe6b08>`



In [151]:
```
sns.regplot(x,np.log(y))
```

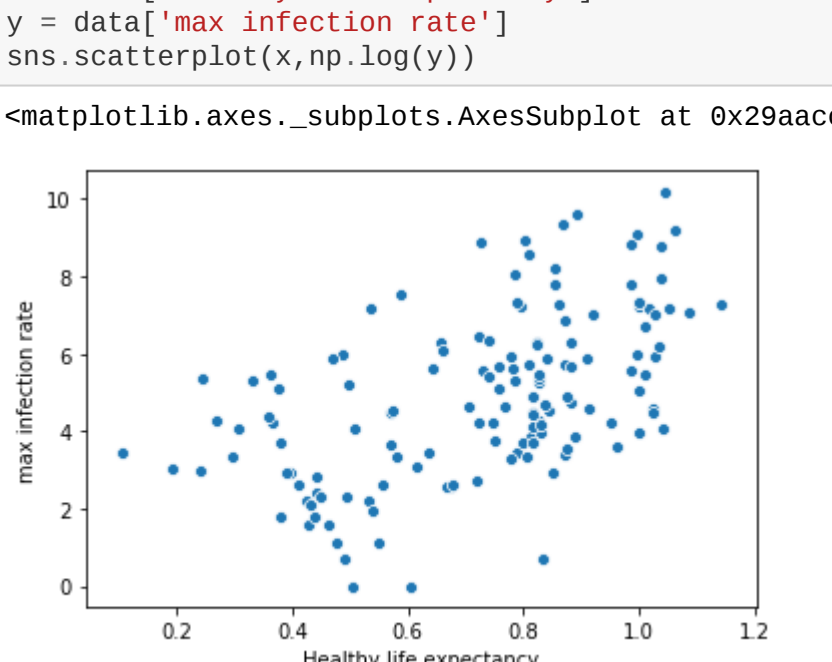
Out[151]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aacc527c8>`



Task 5.3: Plotting Healthy life expectancy vs maximum Infection rate

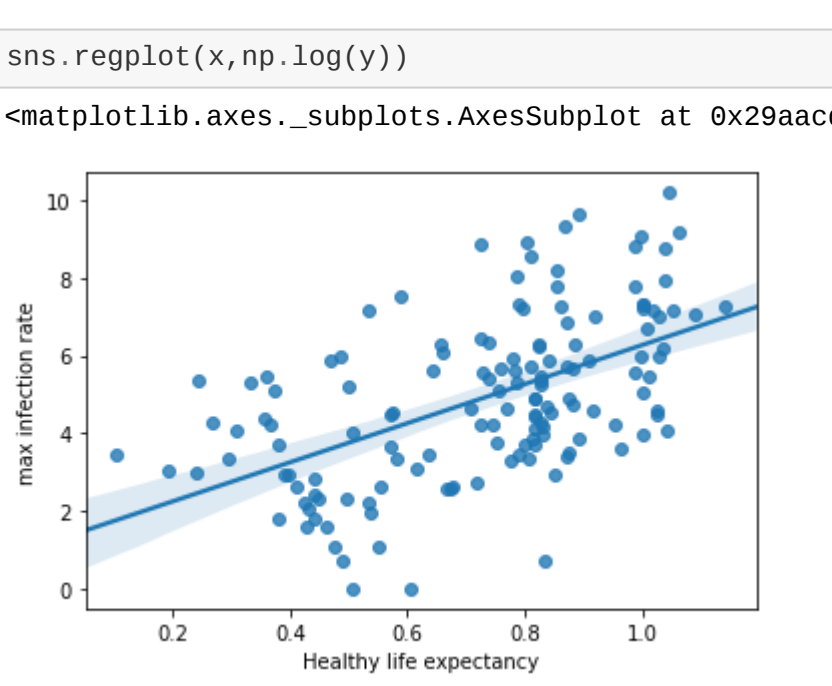
In [152]:
```
x = data['Healthy life expectancy']
y = data['max infection rate']
sns.scatterplot(x,np.log(y))
```

Out[152]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aaccab448>`



In [153]:
```
sns.regplot(x,np.log(y))
```

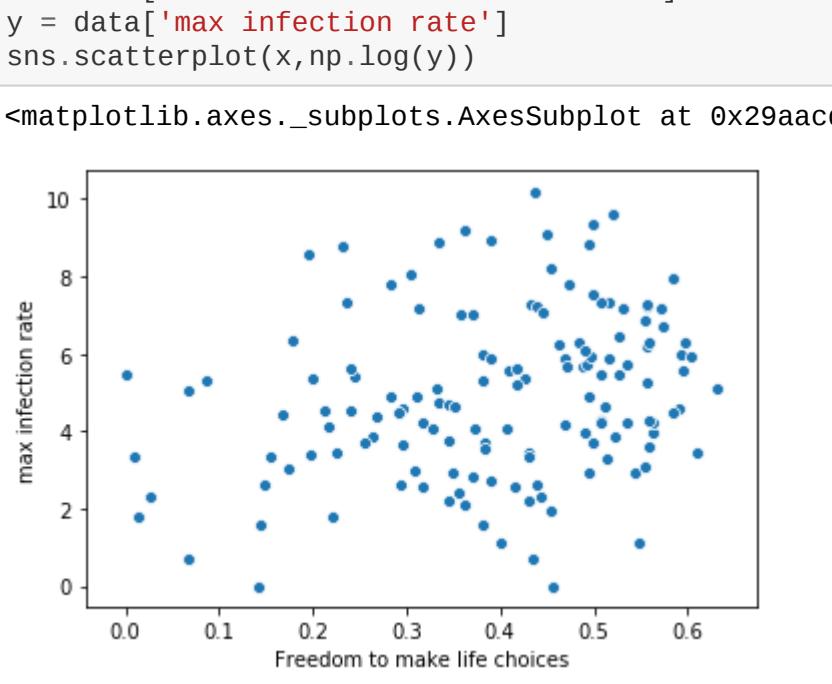
Out[153]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aacd1d448>`



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

In [154]:
```
x = data['Freedom to make life choices']
y = data['max infection rate']
sns.scatterplot(x,np.log(y))
```

Out[154]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aacd856c8>`



In [156]:
```
sns.regplot(x,np.log(y))
```

Out[156]: `<matplotlib.axes._subplots.AxesSubplot at 0x29aace5e748>`