Welcome to Covid19 Data Analysis Notebook

Let's Import the modules

In [39]:

```
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 [40]:

```
corona_dataset_csv = pd.read_csv("Datasets/covid19_Confirmed_dataset.csv")
corona_dataset_csv.head(10)
```

Out[40]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/2
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	
1	NaN	Albania	41.1533	20.1683	0	0	0	0	
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	
5	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	0	
7	NaN	Armenia	40.0691	45.0382	0	0	0	0	
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	0	
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	0	

10 rows × 104 columns

Let's check the shape of the dataframe

In [41]:

```
corona_dataset_csv.shape
```

Out[41]:
```
(266, 104)
```

Task 2.2: Delete the useless columns

In [42]:
```
df = corona_dataset_csv.drop(["Lat","Long"],axis=1, inplace=True)
```

In [ ]:
```

```

Task 2.3: Aggregating the rows by the country

In [43]:
```
corona_dataset_aggregated = corona_dataset_csv.groupby("Country/Region").sum()
```

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

Out[44]:

	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
Country/Region									
Afghanistan	0	0	0	0	0	0	0	0	0
Albania	0	0	0	0	0	0	0	0	0
Algeria	0	0	0	0	0	0	0	0	0
Andorra	0	0	0	0	0	0	0	0	0
Angola	0	0	0	0	0	0	0	0	0

5 rows × 100 columns

In [45]:
```
corona_dataset_aggregated.shape
```

Out[45]:
```
(187, 100)
```

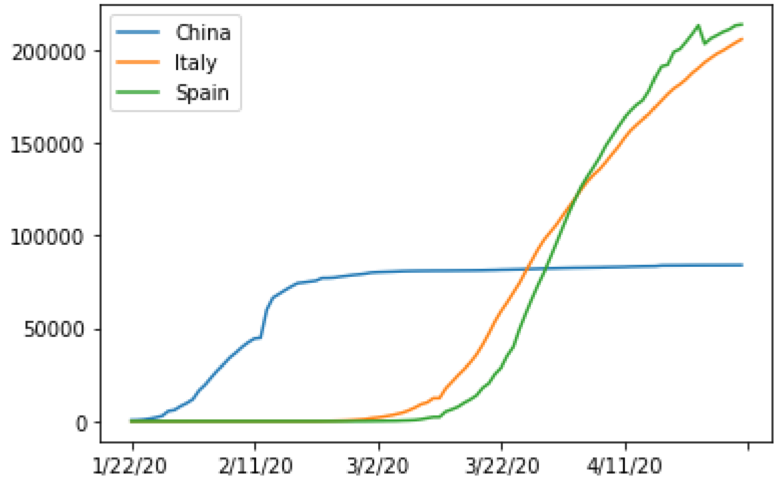
Task 2.4: Visualizing data related to a country for example China

visualization always helps for better understanding of our data.

In [46]:
```
corona_dataset_aggregated.loc["China"].plot()
corona_dataset_aggregated.loc["Italy"].plot()
corona_dataset_aggregated.loc["Spain"].plot()

plt.legend()
```

Out[46]:
```
<matplotlib.legend.Legend at 0x1929970>
```



Task3: Calculating a good measure

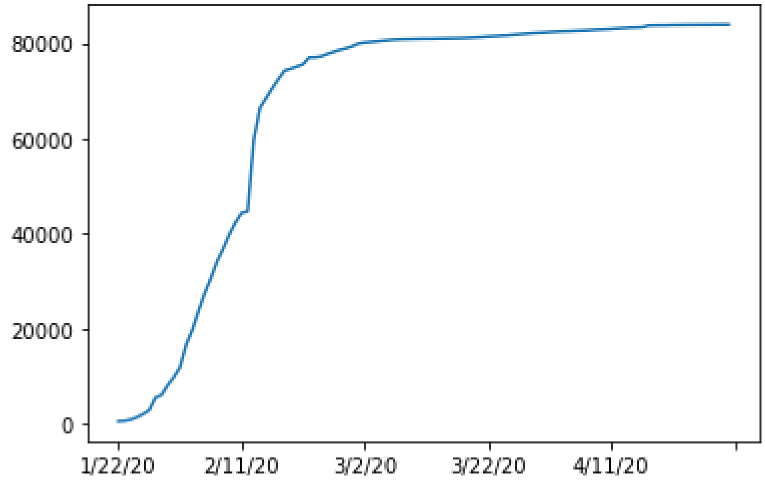
we need to find a good measure reperestend as a number, describing the spread of the virus in a country.

In [47]:

```
corona_dataset_aggregated.loc['China'].plot()
```

Out[47]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1988f40>
```

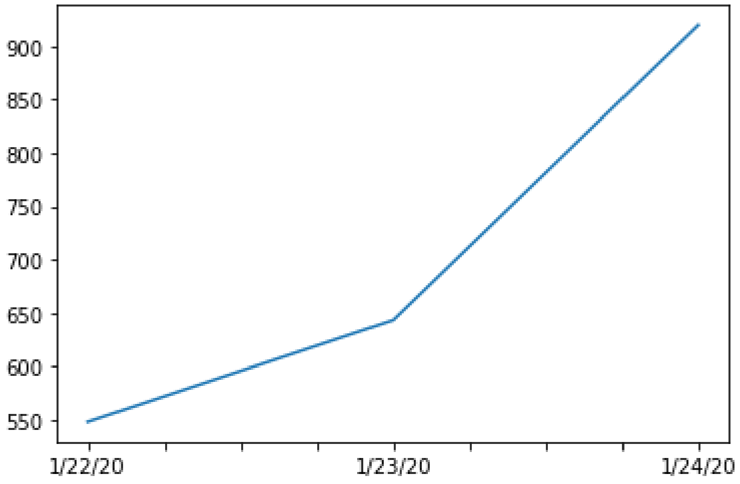


In [48]:

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

Out[48]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x19bae08>
```



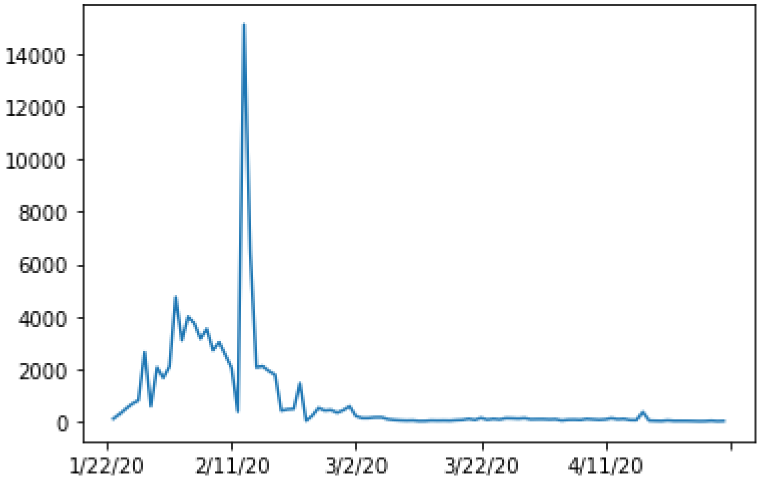
task 3.1: caculating the first derivative of the curve

In [49]:

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

Out[49]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x5572fb8>
```



task 3.2: find maxmimum infection rate for China

In [50]:

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

Out[50]:

```
15136.0
```

In [51]:

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

Out[51]:

```
6557.0
```

In [52]:

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

Out[52]:

```
9630.0
```

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

In [53]:

```
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
```

In [54]:

```
corona_dataset_aggregated.head()
```

Out[54]:

	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
Country/Region									
Afghanistan	0	0	0	0	0	0	0	0	0
Albania	0	0	0	0	0	0	0	0	0
Algeria	0	0	0	0	0	0	0	0	0
Andorra	0	0	0	0	0	0	0	0	0
Angola	0	0	0	0	0	0	0	0	0

5 rows × 101 columns

Task 3.4: create a new dataframe with only needed column

In [55]:

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

In [56]:

```
corona_data.head()
```

Out[56]:

	max_infection_rate
Country/Region	
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 [57]:

```
happiness_report_csv= pd.read_csv("Datasets/worldwide_happiness_report.csv")
```

In [58]:

```
happiness_report_csv.head()
```

Out[58]:

	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

Task 4.2: let's drop the useless columns

In [59]:

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

In [60]:

```
happiness_report_csv.drop(useless_cols,axis=1,inplace=True)
happiness_report_csv.head()
```

Out[60]:

	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 [62]:

```
happiness_report_csv.set_index("Country or region",inplace=True)
```

In [63]:

```
happiness_report_csv.head()
```

Out[63]:

	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Country or region				
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
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 [64]:

```
corona_data.head()
```

Out[64]:

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

In [66]:

```
corona_data.shape
```

Out[66]:

```
(187, 1)
```

wolrd happiness report Dataset :

In [68]:

```
happiness_report_csv.head()
```

Out[68]:

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 [69]:

```
happiness_report_csv.shape
```

Out[69]:

```
(156, 4)
```

In [71]:

```
data = corona_data.join(happiness_report_csv,how="inner")
data.head()
```

Out[71]:

	max_infection_rate	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

Task 4.5: correlation matrix

In [72]:

```
data.corr()
```

Out[72]:

	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

In [ ]:

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 [73]:

```
data.head()
```

Out[73]:

	max_infection_rate	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

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Armenia	134.0	0.850	1.055	0.815	0.283

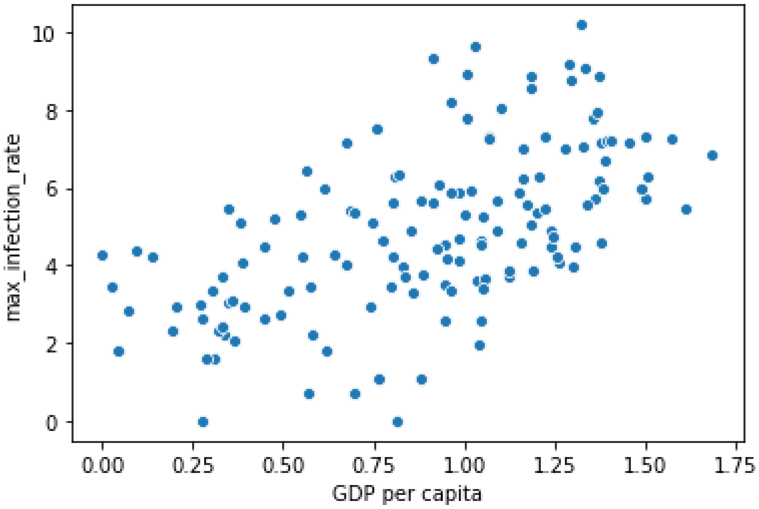
Task 5.1: Plotting GDP vs maximum Infection rate

In [75]:

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

Out[75]:

```
<matplotlib.axes._subplots.AxesSubplot at 0xf9b0520>
```

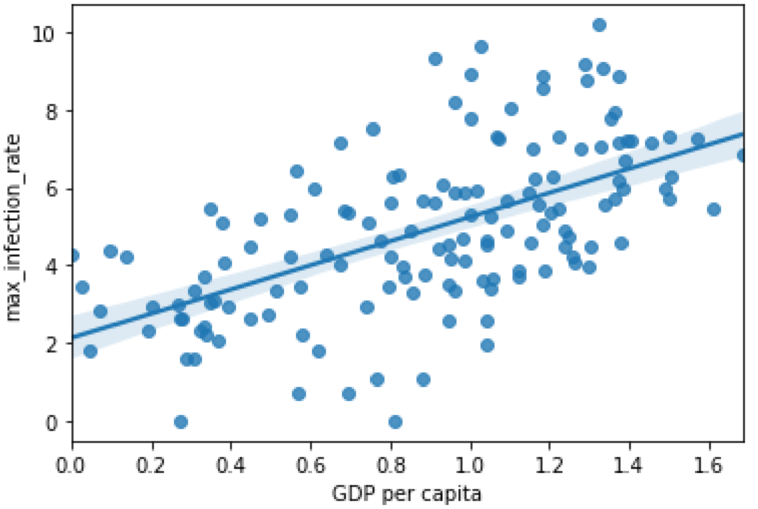


In [76]:

```
sns.regplot(x,np.log(y))
```

Out[76]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x107f0aa8>
```

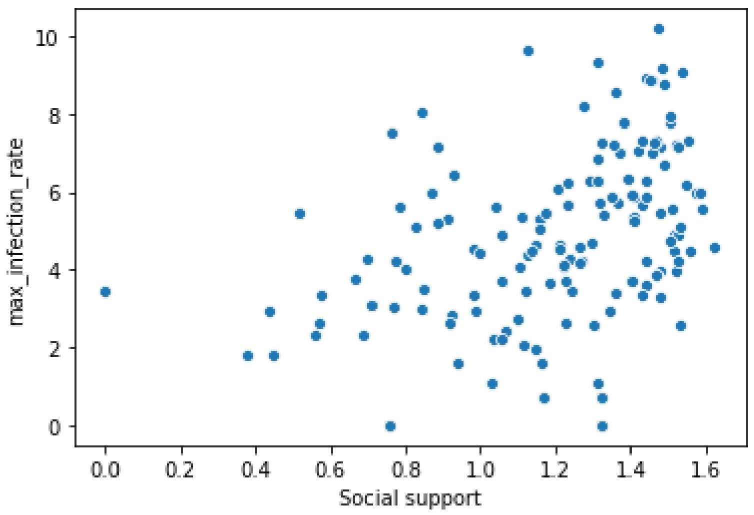


Task 5.2: Plotting Social support vs maximum Infection rate

In [78]:

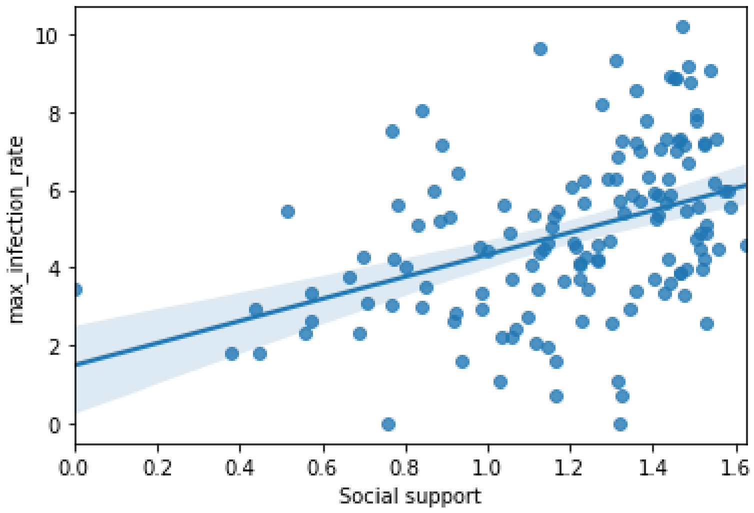
```
x = data ["Social support"]
y = data["max_infection_rate"]
sns.scatterplot(x,np.log(y))
```

```
Out[78]: <matplotlib.axes._subplots.AxesSubplot at 0xfb83280>
```



```
In [79]: sns.regplot(x,np.log(y))
```

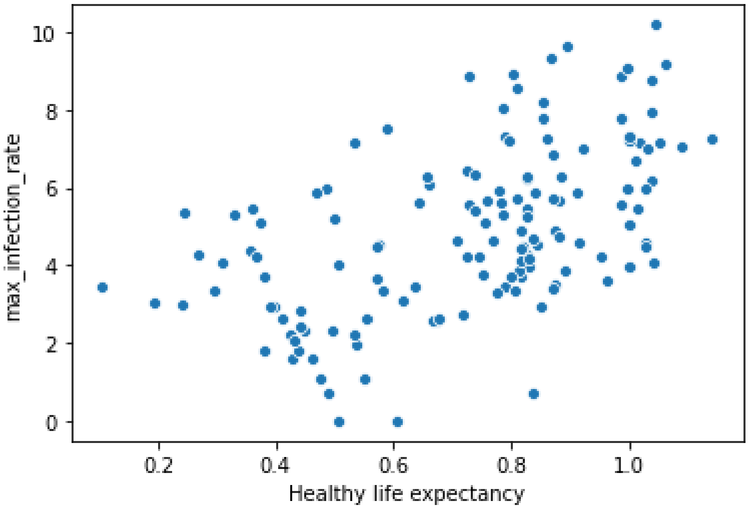
```
Out[79]: <matplotlib.axes._subplots.AxesSubplot at 0xfe087a8>
```



Task 5.3: Plotting Healthy life expectancy vs maximum Infection rate

```
In [80]: x = data ["Healthy life expectancy"]
         y = data["max_infection_rate"]
         sns.scatterplot(x,np.log(y))
```

```
Out[80]: <matplotlib.axes._subplots.AxesSubplot at 0xfe55430>
```

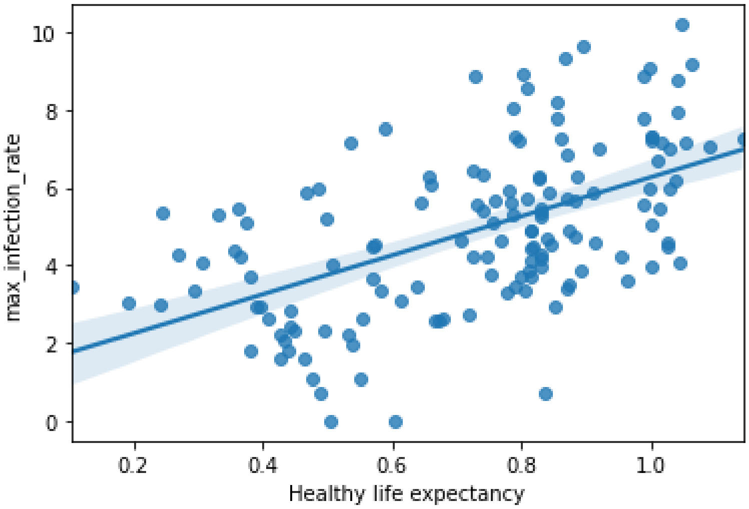


In [81]:

```
sns.regplot(x,np.log(y))
```

Out[81]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1017d598>
```



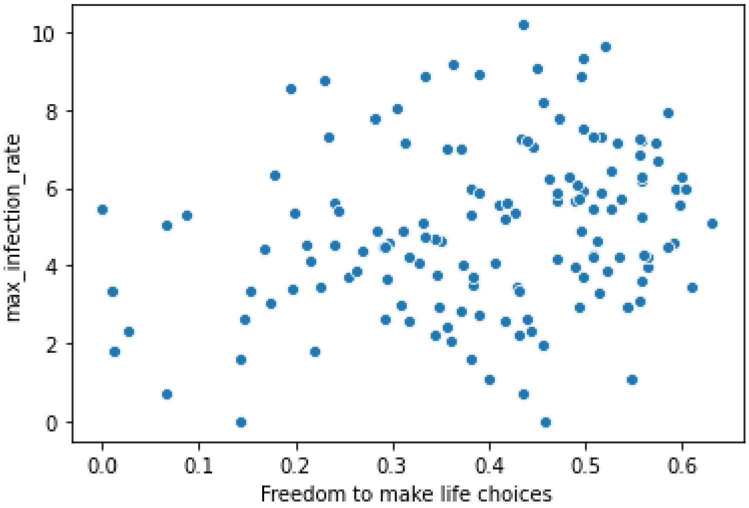
Task 5.4: Plotting Freedom to make life choices vs maximum Infection rate

In [82]:

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

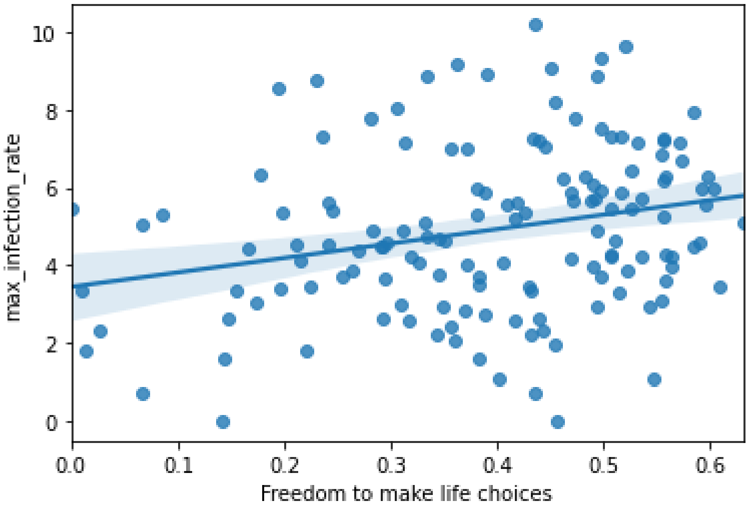
Out[82]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x101b61c0>
```



In [83]:
```
sns.regplot(x,np.log(y))
```

Out[83]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x101db418>
```



In [ ]: