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.
```

Task 2

Task 2.1: importing covid19 dataset

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

In [2]:

```
corona_dataset_csv = pd.read_csv('Dataset/covid19_Confirmed_dataset.csv')
corona_dataset_csv.head(10)
```

Out[2]:

	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	4/22/2
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0	0	...	1092	117
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0	0	...	609	63
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0	0	...	2811	291
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0	0	...	717	72
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0	0	...	24	2
5	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	0	0	...	23	2
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	0	0	0	...	3031	314
7	NaN	Armenia	40.0691	45.0382	0	0	0	0	0	0	...	1401	147
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	0	0	0	...	104	10
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	0	3	4	...	2969	297

10 rows × 104 columns

Let's check the shape of the dataframe

In [3]:

```
corona_dataset_csv.shape
```

Out[3]:

```
(266, 104)
```

Task 2.2: Delete the useless columns

In [4]:

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

In [5]:

```
corona_dataset_csv.head(10)
```

Out[5]:

	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
0	NaN	Afghanistan	0	0	0	0	0	0	0	0	...	1092	1176
1	NaN	Albania	0	0	0	0	0	0	0	0	...	609	634
2	NaN	Algeria	0	0	0	0	0	0	0	0	...	2811	2910
3	NaN	Andorra	0	0	0	0	0	0	0	0	...	717	723
4	NaN	Angola	0	0	0	0	0	0	0	0	...	24	25
5	NaN	Antigua and Barbuda	0	0	0	0	0	0	0	0	...	23	24
6	NaN	Argentina	0	0	0	0	0	0	0	0	...	3031	3144
7	NaN	Armenia	0	0	0	0	0	0	0	0	...	1401	1473
8	Australian Capital Territory	Australia	0	0	0	0	0	0	0	0	...	104	104
9	New South Wales	Australia	0	0	0	0	3	4	4	4	...	2969	2971

10 rows × 102 columns

Task 2.3: Aggregating the rows by the country

In [6]:

```
corona_dataset_aggregated = corona_dataset_csv.groupby("Country/Region").sum()
```

In [7]:

```
corona_dataset_aggregated.head(10)
```

Out[7]:

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
Afghanistan	0	0	0	0	0	0	0	0	0	0	...	1092	1176
Albania	0	0	0	0	0	0	0	0	0	0	...	609	634
Algeria	0	0	0	0	0	0	0	0	0	0	...	2811	2910
Andorra	0	0	0	0	0	0	0	0	0	0	...	717	723
Angola	0	0	0	0	0	0	0	0	0	0	...	24	25
Antigua and Barbuda	0	0	0	0	0	0	0	0	0	0	...	23	24
Argentina	0	0	0	0	0	0	0	0	0	0	...	3031	3144

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
Armenia	0	0	0	0	0	0	0	0	0	0	...	1401	1473
Australia	0	0	0	0	4	5	5	6	9	9	...	6645	6652
Austria	0	0	0	0	0	0	0	0	0	0	...	14873	14925

10 rows × 100 columns

In [8]:

```
corona_dataset_aggregated.shape
```

Out[8]:

```
(187, 100)
```

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

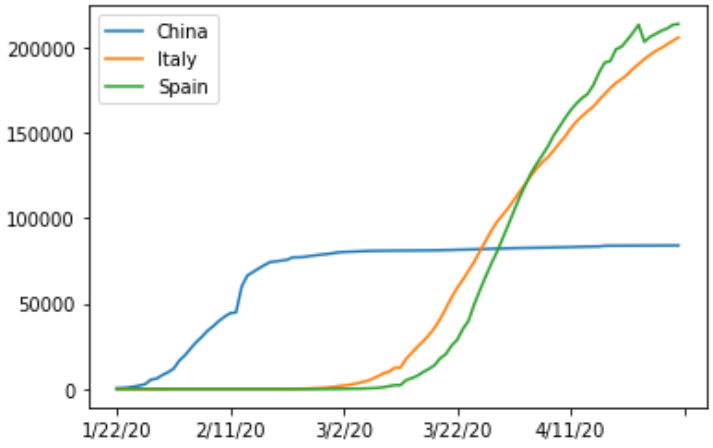
visualization always helps for better understanding of our data.

In [9]:

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

Out[9]:

```
<matplotlib.legend.Legend at 0x1bab41bd780>
```



Task3: Calculating a good measure

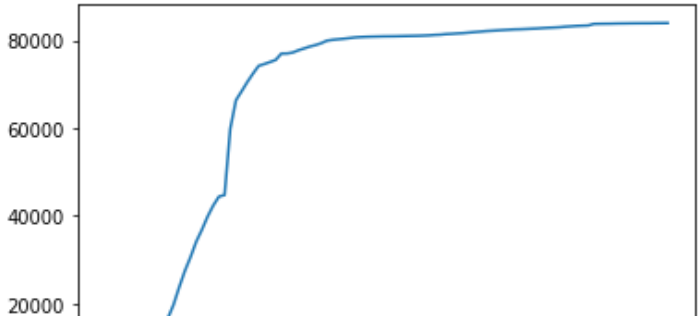
we need to find a good measure reperestend as a number, describing the spread of the virus in a country.

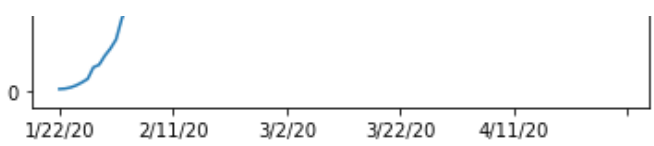
In [10]:

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

Out[10]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab4325550>
```



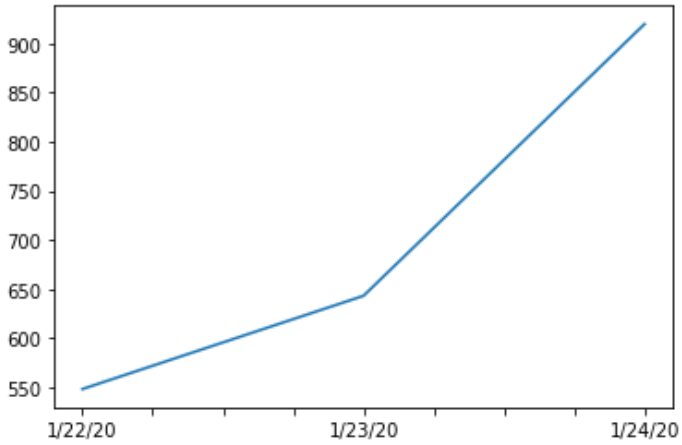


In [11]:

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

Out[11]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab437c4e0>
```



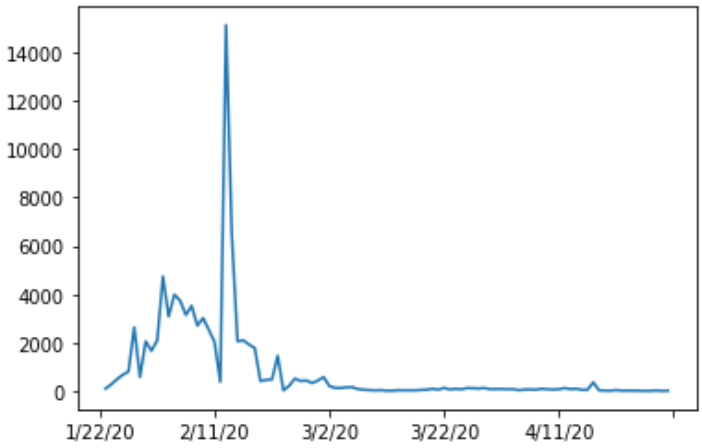
task 3.1: caculating the first derivative of the curve

In [12]:

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

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab43fb6d8>
```



task 3.2: find maxmimum infection rate for China

In [13]:

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

Out[13]:

```
15136.0
```

In [14]:

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

Out[14]:

6557.0

In [15]:

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

Out[15]:

9630.0

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

In [16]:

```
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 [17]:

```
corona_dataset_aggregated.head()
```

Out[17]:

	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
Country/Region													
Afghanistan	0	0	0	0	0	0	0	0	0	0	...	1176	1279
Albania	0	0	0	0	0	0	0	0	0	0	...	634	663
Algeria	0	0	0	0	0	0	0	0	0	0	...	2910	3007
Andorra	0	0	0	0	0	0	0	0	0	0	...	723	723
Angola	0	0	0	0	0	0	0	0	0	0	...	25	25

5 rows × 101 columns

Task 3.4: create a new dataframe with only needed column

In [18]:

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

In [19]:

```
corona_data.head()
```

Out[19]:

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

Task4:

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 [20]:

```
world_happiness_report = pd.read_csv("Dataset/worldwide_happiness_report.csv")
world_happiness_report.head()
```

Out[20]:

	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 [21]:

```
world_happiness_report.shape
```

Out[21]:

```
(156, 9)
```

Task 4.2: let's drop the useless columns

In [22]:

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

In [23]:

```
world_happiness_report.head()
```

Out[23]:

	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

In [24]:

```
world_happiness_report.shape
```

Out[24]:

```
(156, 5)
```

Task 4.3: changing the indices of the dataframe

In [25]:

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

Out[25]:

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 [26]:

```
corona_data.head()
```

Out[26]:

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

In [27]:

```
corona_data.shape
```

Out[27]:

```
(187, 1)
```

World Happiness Report Dataset :

In [28]:

```
world_happiness_report.head()
```

Out[28]:

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

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557

In [29]:

```
world_happiness_report.shape
```

Out[29]:

```
(156, 4)
```

Merging Both Dataset:

In [30]:

```
data = corona_data.join(world_happiness_report, how='inner')
data.head()
```

Out[30]:

	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 [31]:

```
data.corr()
# it is representing the correlation between every two columns of our dataset
```

Out[31]:

	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

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 [32]:

```
data.head()
```

Out[32]:

	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

	max infection rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
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

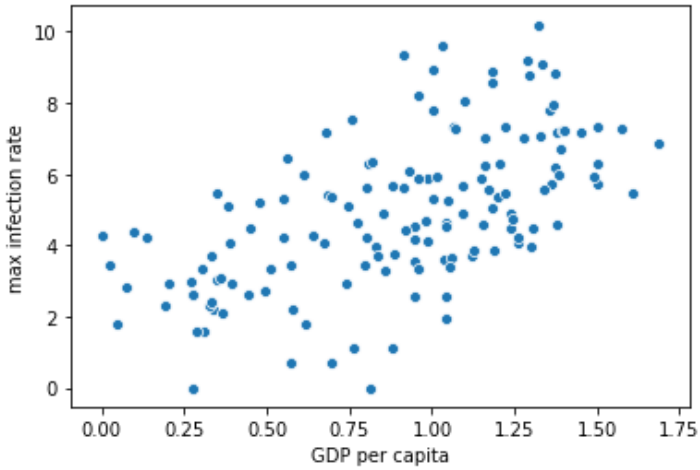
Task 5.1: Plotting GDP vs maximum Infection rate

In [33]:

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

Out[33]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab42fee80>
```

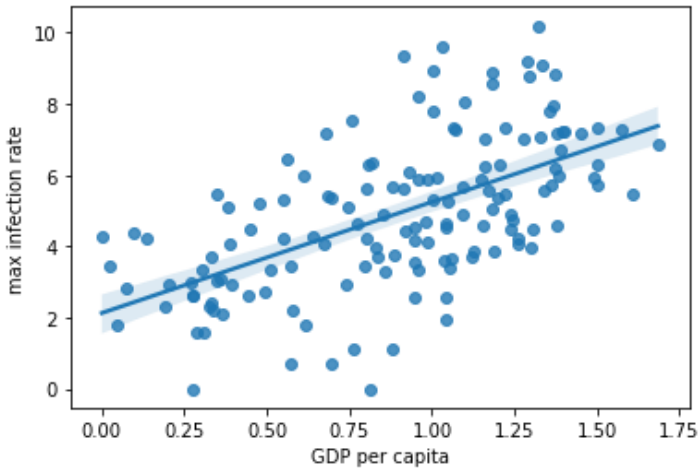


In [34]:

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

Out[34]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab44fba58>
```



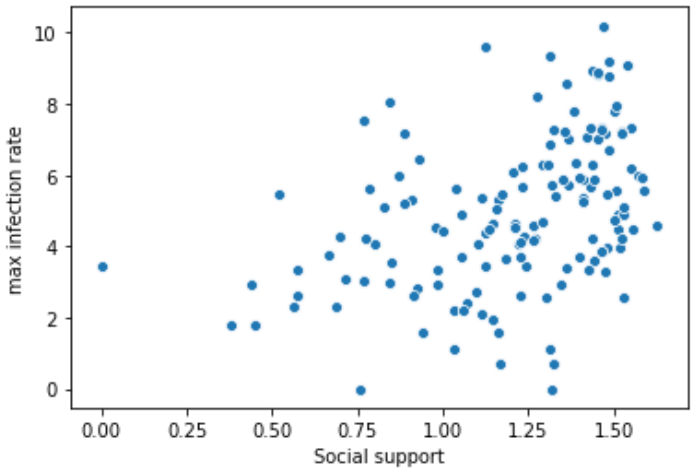
Task 5.2: Plotting Social support vs maximum Infection rate

In [35]:

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

Out[35]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab4c032e8>
```

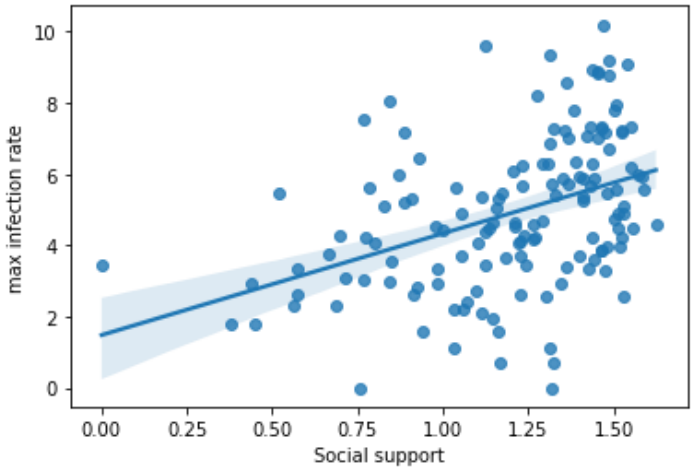


In [36]:

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

Out[36]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab4c03048>
```



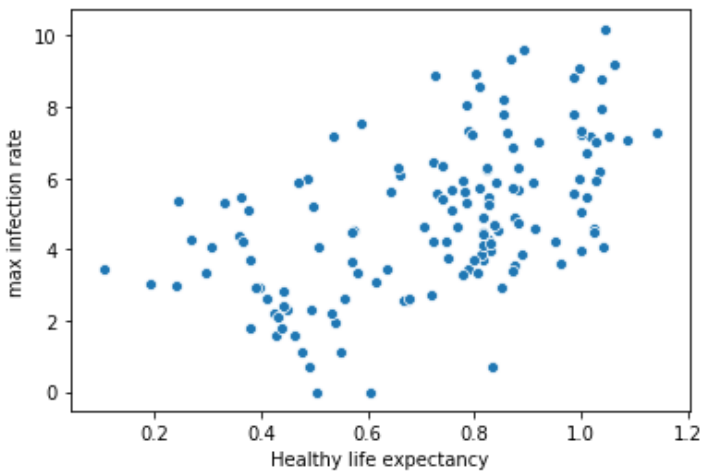
Task 5.3: Plotting Healthy life expectancy vs maximum Infection rate

In [37]:

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

Out[37]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab4cc7d68>
```

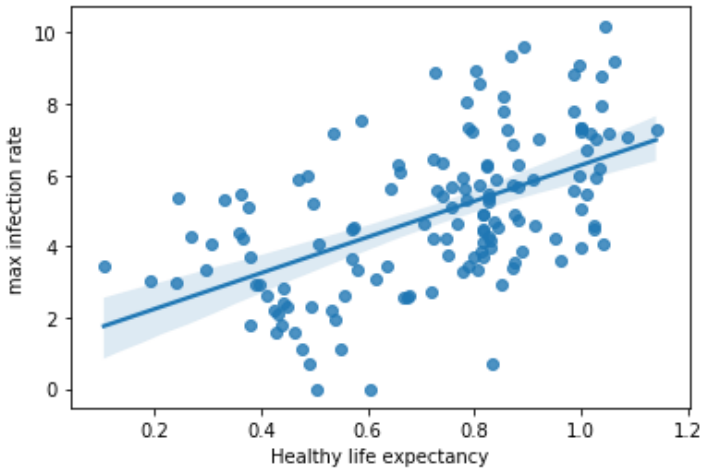


In [38]:

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

Out[38]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab4d17e80>
```



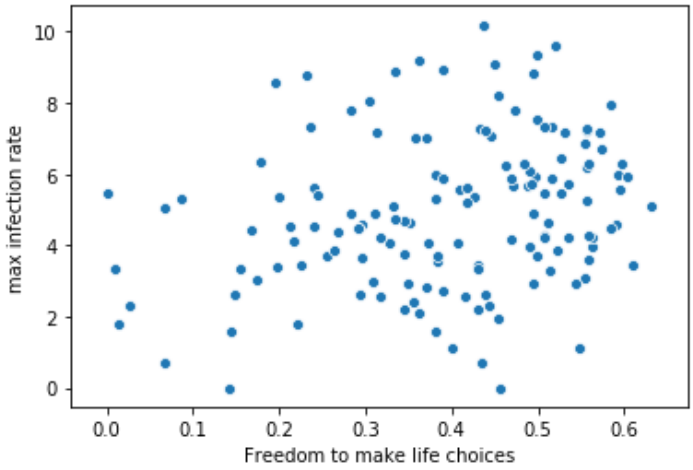
Task 5.4: Plotting Freedom to make life choices vs maximum Infection rate

In [39]:

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

Out[39]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab4d87518>
```

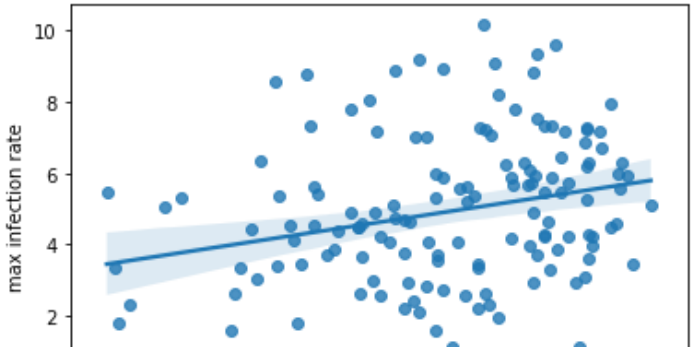


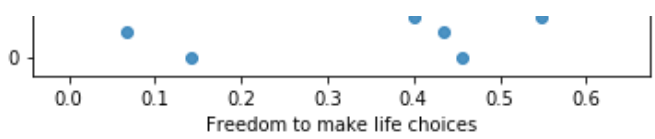
In [40]:

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

Out[40]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bab4dc5080>
```




0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Freedom to make life choices