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('Datasets/covid19_Confirmed_dataset.csv')
corona_dataset_csv.head()
```

Out[2]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0

5 rows × 104 columns

We can clearly see from above data that Data is available from 23/1/2020 to 30/4/2020

Some columns are also not of use so we will drop them(Lat, Long

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]:

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

In [5]:

```
corona_dataset_csv.head()
```

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/
0	NaN	Afghanistan	0	0	0	0	0	0	0	
1	NaN	Albania	0	0	0	0	0	0	0	
2	NaN	Algeria	0	0	0	0	0	0	0	
3	NaN	Andorra	0	0	0	0	0	0	0	
4	NaN	Angola	0	0	0	0	0	0	0	

5 rows × 102 columns

Task 2.3: Aggregating the rows by the country

In [6]:

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

this method will return us an aggregated value

In [7]:

```
df_aggregated.head()
```

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	1/3
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

We can look two data shown above, How index with 0,1,2,3 is changed to Country/Region name

In [8]:

```
df_aggregated.shape
```

Out[8]:

```
(187, 100)
```

This means that we have 187 countries and 100 days data is present

Task 2.4: Visualizing data related to a country for example China, Italy and India

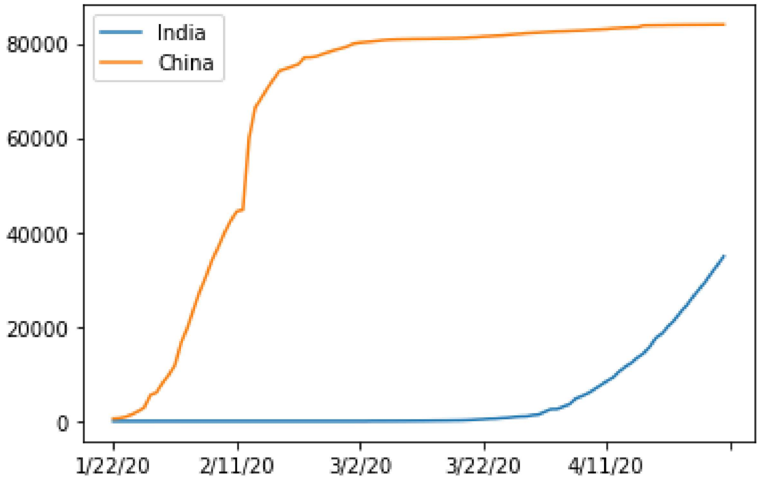
visualization always helps for better understanding of our data.

In [9]:

```
df_aggregated.loc['India'].plot()
df_aggregated.loc["China"].plot()
plt.legend()   #this will show which color belongs to which country
```

Out[9]:

```
<matplotlib.legend.Legend at 0xcbd56a0>
```

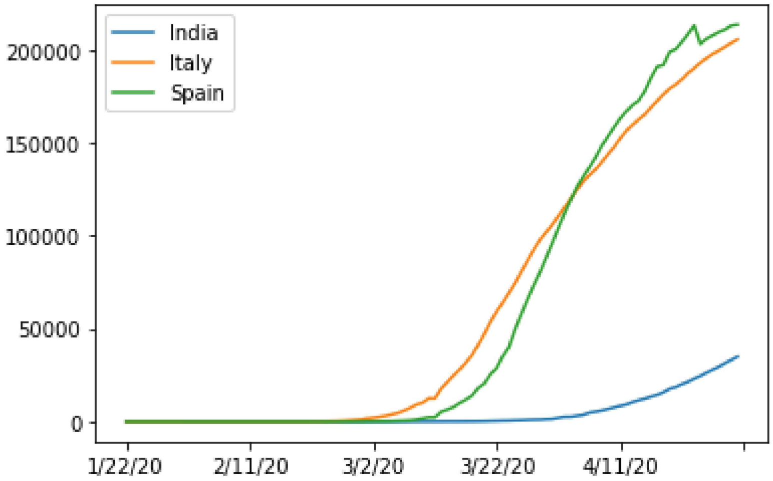


In [10]:

```
df_aggregated.loc["India"].plot()
df_aggregated.loc["Italy"].plot()
df_aggregated.loc["Spain"].plot()
plt.legend()
```

Out[10]:

```
<matplotlib.legend.Legend at 0xcceadd8>
```

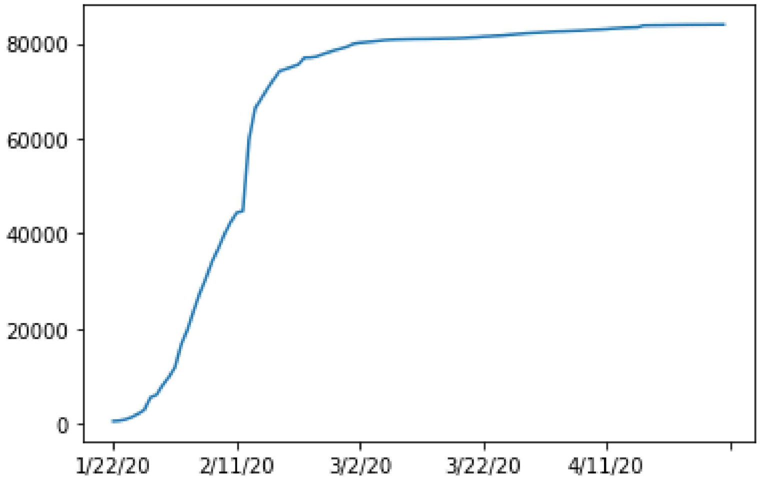


Task3: Calculating a good measure

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

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

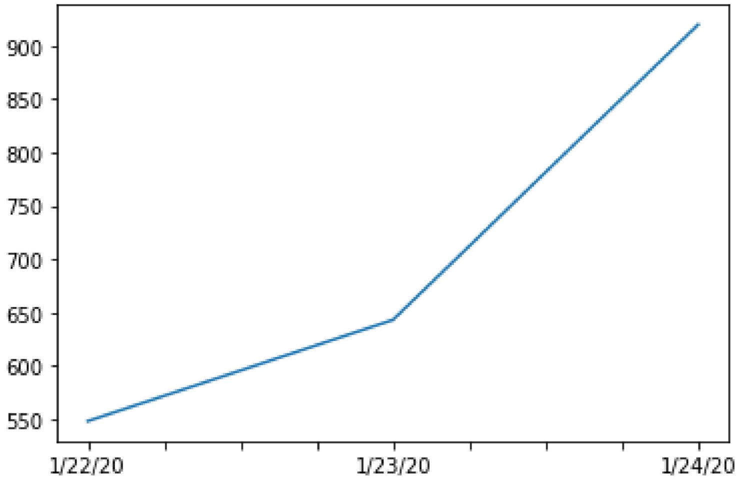
```
Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0xcd30af0>
```



Let we want to see first three days cases in China...

```
In [12]: df_aggregated.loc["China"][ :3].plot()
```

```
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0xcd6c868>
```



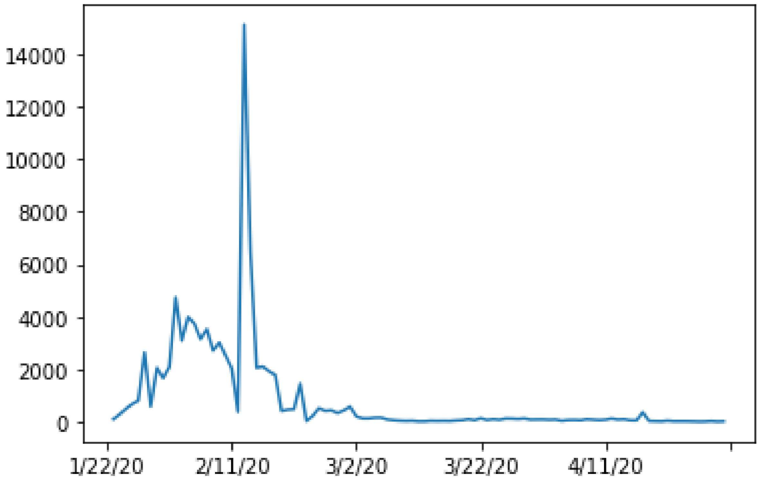
We acn see that in first 24 hrs 'number of cases' jumped from 550 to 650 ie. only 100 new cases. But on next 24 hrs, it jumps from 650 to 900 ie. 250 new cases.

Now we want to find on which day maxm number of cases was recorded. For this we will find FIRST DERIVATIVE

task 3.1: caculating the first derivative of the curve

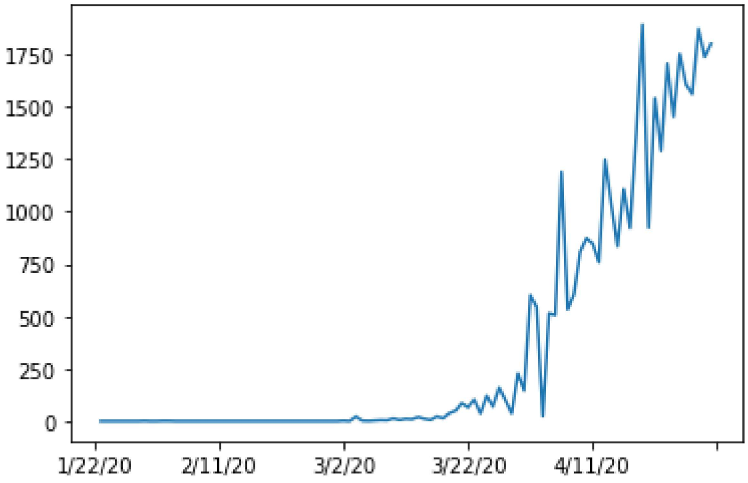
```
In [13]: df_aggregated.loc["China"].diff().plot()

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0xcd9cfe8>
```



```
In [14]: df_aggregated.loc["India"].diff().plot()

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0xcdd7ee0>
```



task 3.2: find maxmimum infection rate for China , Italy and India

```
In [15]: df_aggregated.loc["China"].diff().max()

Out[15]: 15136.0
```

```
In [16]: df_aggregated.loc["Italy"].diff().max()

Out[16]: 6557.0
```

```
In [17]: df_aggregated.loc["India"].diff().max()

Out[17]: 1893.0
```

The maxm number of cases recorded in 24 hrs in **China was 15136** , in **Italy was 6557** and in **India was 1893**

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

In [18]:

```
countries = list(df_aggregated.index)
max_infection_rates = []
for c in countries :
    max_infection_rates.append(df_aggregated.loc[c].diff().max())

# Adding new column "max_infection_rate to dataframe"
df_aggregated['max_inf_rate'] = max_infection_rates
```

In [19]:

```
df_aggregated.head()
```

Out[19]:

	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	1/3
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

Giving a new name to data frame "corona_data"

In [20]:

```
corona_max_df = pd.DataFrame(df_aggregated['max_inf_rate'])
```

In [21]:
```
corona_max_df.head()
```

Out[21]:

	max_inf_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 [22]:
```
happiness_repo_csv = pd.read_csv("DAtasets/worldwide_happiness_report.csv")
```

In [23]:
```
happiness_repo_csv.head()
```

Out[23]:

	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 [24]:
```
useless_cols = ["Overall rank","Score","Generosity","Perceptions of corruption"]
```

In [25]:

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

Out[25]:

	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 [26]:

```
happiness_report_csv=happiness_repo_csv.groupby("Country or region").sum()
```

In [27]:

```
happiness_report_csv.head()
```

Out[27]:

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

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

Corona Dataset :

In [28]:

```
corona_max_df.head()
```

Out[28]:

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

In [29]:

```
corona_max_df.shape
```

Out[29]:

```
(187, 1)
```

wolrd happiness report Dataset :

In [30]:

```
happiness_report_csv.head()
```

Out[30]:

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

In [31]:

```
happiness_report_csv.shape
```

Out[31]:

```
(156, 4)
```

In [32]:

```
# We will do inner join as less rows in second dataframe
```

In [33]:

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

In [34]:

```
data.head()
```

Out[34]:

	max_inf_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 [35]:

```
data.corr()
```

Out[35]:

	max_inf_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
max_inf_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 [37]:

```
data.head()
```

Out[37]:

	max_inf_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

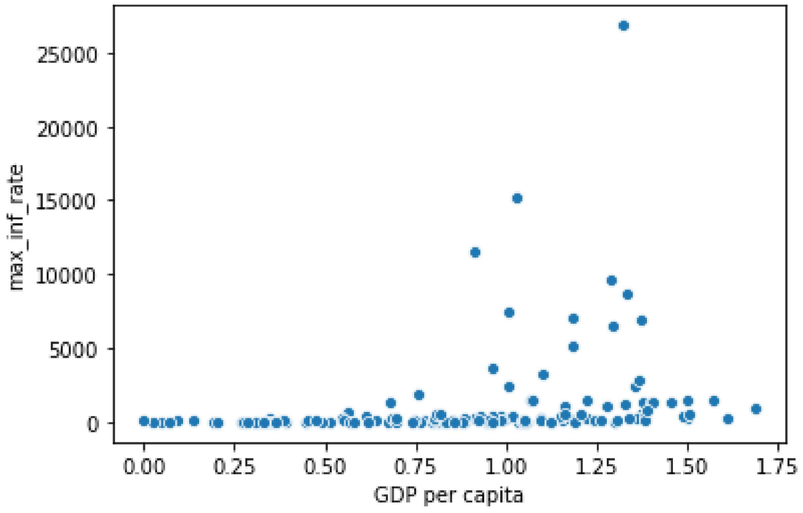
Task 5.1: Plotting GDP vs maximum Infection rate

In [38]:

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

Out[38]:

```
<matplotlib.axes._subplots.AxesSubplot at 0xce10478>
```

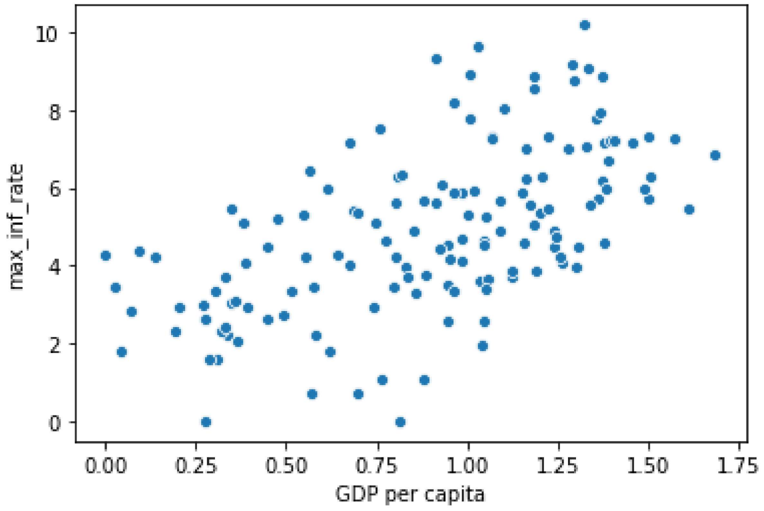


In [39]:

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

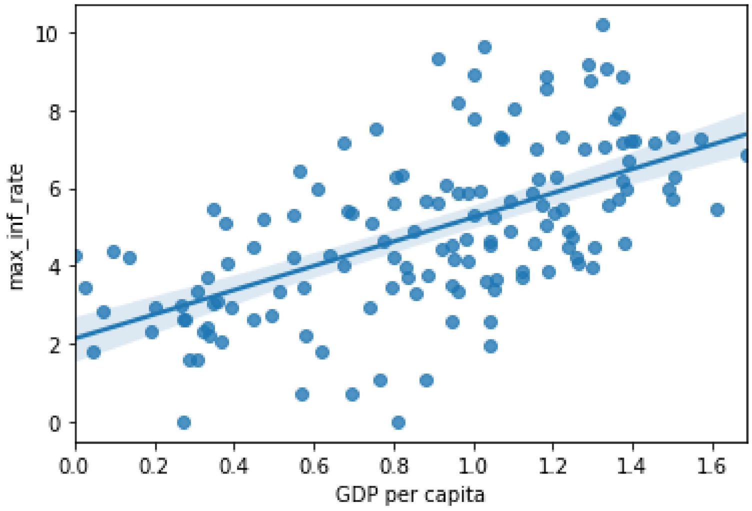
Out[39]:

```
<matplotlib.axes._subplots.AxesSubplot at 0xcbd5418>
```



In [40]:
```
sns.regplot(x,np.log(y))
```

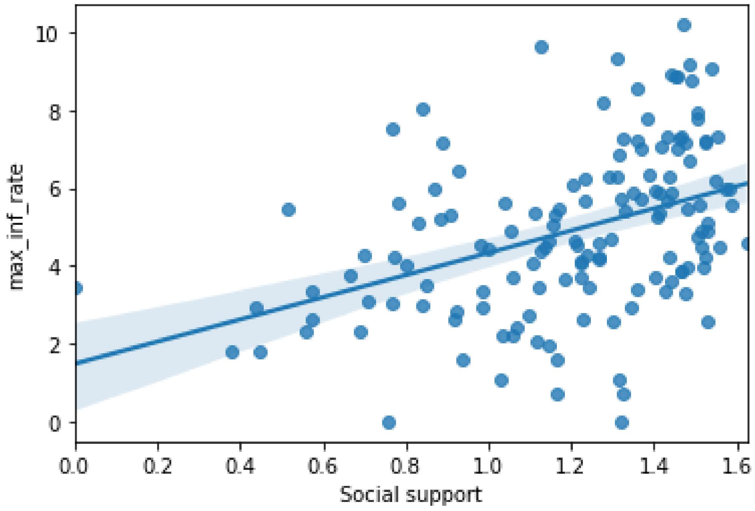
Out[40]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xadfc10>
```



Task 5.2: Plotting Social support vs maximum Infection rate

In [41]:
```
x = data["Social support"]
y = data["max_inf_rate"]
sns.regplot(x,np.log(y))
```

Out[41]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x913640>
```



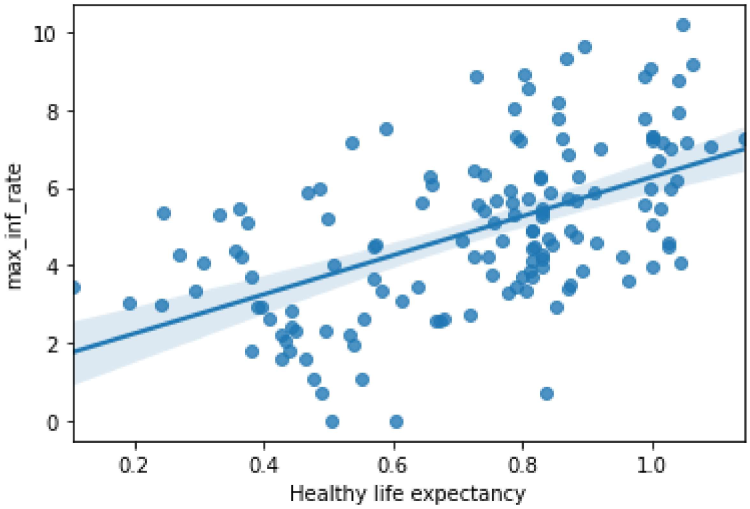
In [ ]:
```

```

Task 5.3: Plotting Healthy life expectancy vs maximum Infection rate

In [43]:
```
x = data["Healthy life expectancy"]
y = data["max_inf_rate"]
sns.regplot(x,np.log(y))
```

Out[43]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x8df328>
```

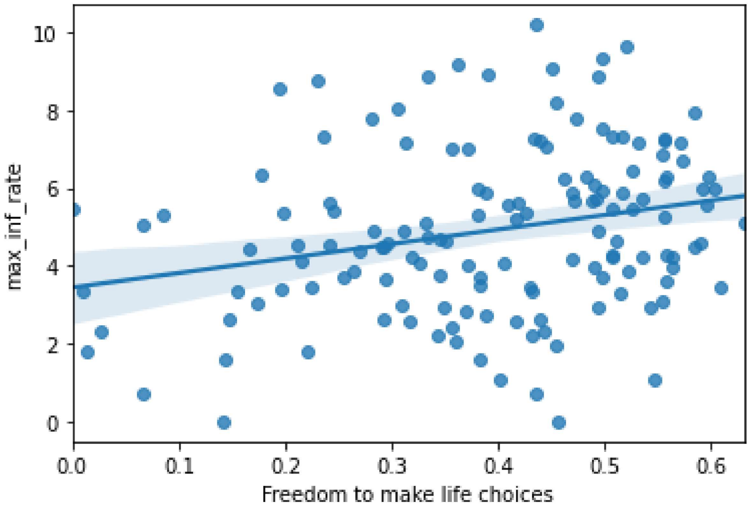


In [ ]:

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

In [45]:
```
x = data["Freedom to make life choices"]
y = data["max_inf_rate"]
sns.regplot(x,np.log(y))
```

Out[45]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xb821d8>
```



In [ ]: