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 [10]:
```
corona_dataset_csv = pd.read_csv("covid19_Confirmed_dataset.csv")
corona_dataset_csv.head(10)
```

Out[10]:

	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	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	0
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	0	0
7	NaN	Armenia	40.0691	45.0382	0	0	0	0	0
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	0	0
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	0	3

10 rows × 104 columns

Let's check the shape of the dataframe

In [11]:
```
corona_dataset_csv.shape
```

Out[11]: (266, 104)

Task 2.2: Delete the useless columns

In [19]:
```
corona_dataset_csv.drop(["Lat", "Long"], axis = 1,inplace=True)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-19-1cde3f253253> in <module>
----> 1 corona_dataset_csv.drop(["Lat", "Long"], axis = 1,inplace=True)

~\anaconda3\lib\site-packages\pandas\core\frame.py in drop(self, labels, axi
s, index, columns, level, inplace, errors)
   3995             level=level,
   3996             inplace=inplace,
-> 3997             errors=errors,
   3998         )
   3999

~\anaconda3\lib\site-packages\pandas\core\generic.py in drop(self, labels, ax
is, index, columns, level, inplace, errors)
   3934         for axis, labels in axes.items():
   3935             if labels is not None:
-> 3936                 obj = obj._drop_axis(labels, axis, level=level, error
s=errors)
   3937
   3938         if inplace:

~\anaconda3\lib\site-packages\pandas\core\generic.py in _drop_axis(self, labe
ls, axis, level, errors)
   3968                 new_axis = axis.drop(labels, level=level, errors=erro
rs)
   3969             else:
-> 3970                 new_axis = axis.drop(labels, errors=errors)
   3971             result = self.reindex(**{axis_name: new_axis})
   3972

~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in drop(self, label
s, errors)
   5016         if mask.any():
   5017             if errors != "ignore":
-> 5018                 raise KeyError(f"{labels[mask]} not found in axis")
   5019             indexer = indexer[~mask]
   5020         return self.delete(indexer)

KeyError: "['Lat' 'Long'] not found in axis"
```

In [21]:
```
corona_dataset_csv.head(10)
```

Out[21]:

	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	NaN	Antigua and Barbuda	0	0	0	0	0	0	0	
6	NaN	Argentina	0	0	0	0	0	0	0	
7	NaN	Armenia	0	0	0	0	0	0	0	
8	Australian Capital Territory	Australia	0	0	0	0	0	0	0	
9	New South Wales	Australia	0	0	0	0	3	4	4	

10 rows × 102 columns

Task 2.3: Aggregating the rows by the country

In [22]:
```
corona_dataset_aggregated=corona_dataset_csv.groupby("Country/Region").sum()
```

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

Out[23]:

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

In [24]:
```
corona_dataset_aggregated.shape
```

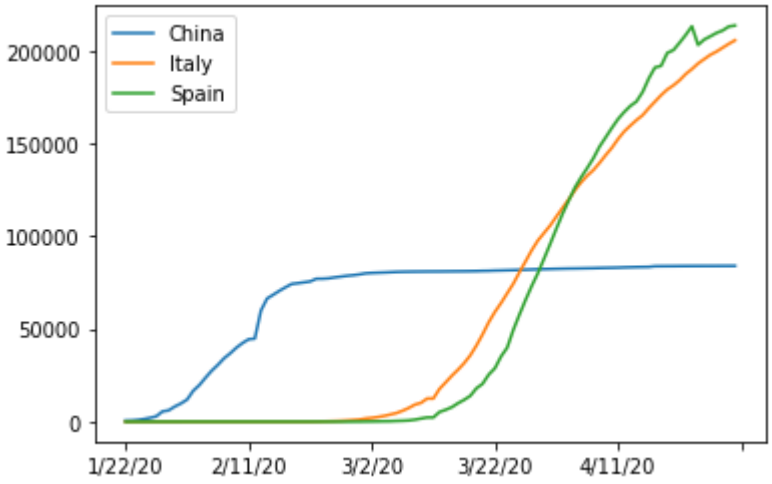
Out[24]:
```
(187, 100)
```

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

visualization always helps for better understanding of our data.

In [27]:
```
corona_dataset_aggregated.loc["China"].plot()
corona_dataset_aggregated.loc['Italy'].plot()
corona_dataset_aggregated.loc['Spain'].plot()
plt.legend()
```

Out[27]: <matplotlib.legend.Legend at 0x211753c0548>

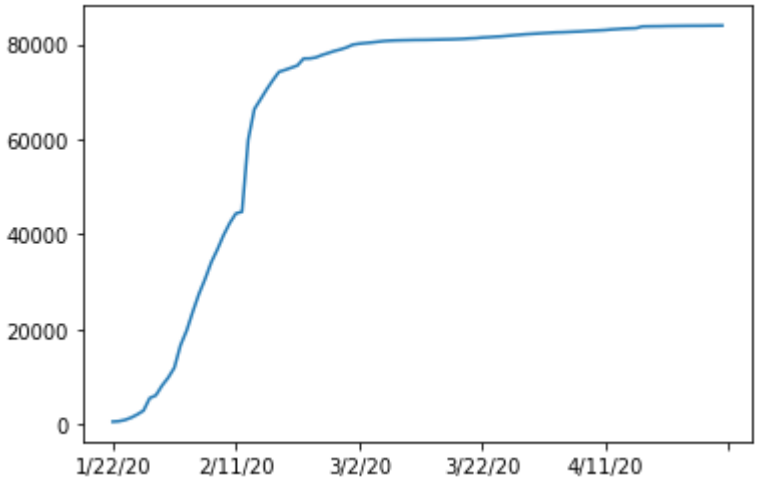


Task3: Calculating a good measure

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

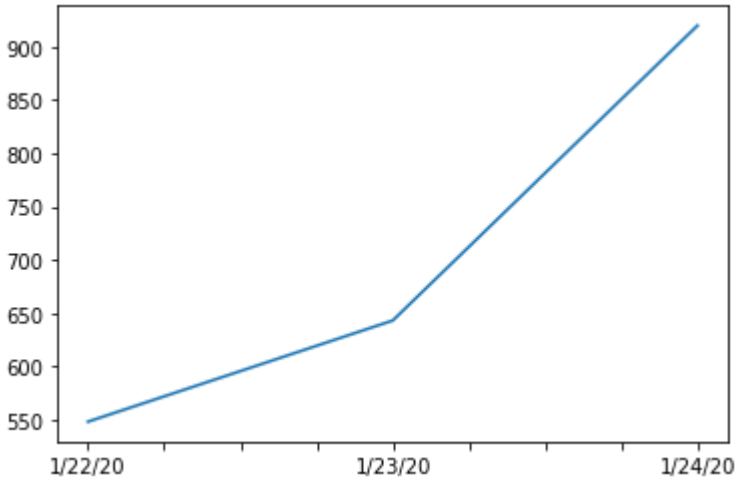
In [28]:
```
corona_dataset_aggregated.loc['China'].plot()
```

Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x21174caa7c8>



In [30]:
```
corona_dataset_aggregated.loc['China'][:3].plot()
```

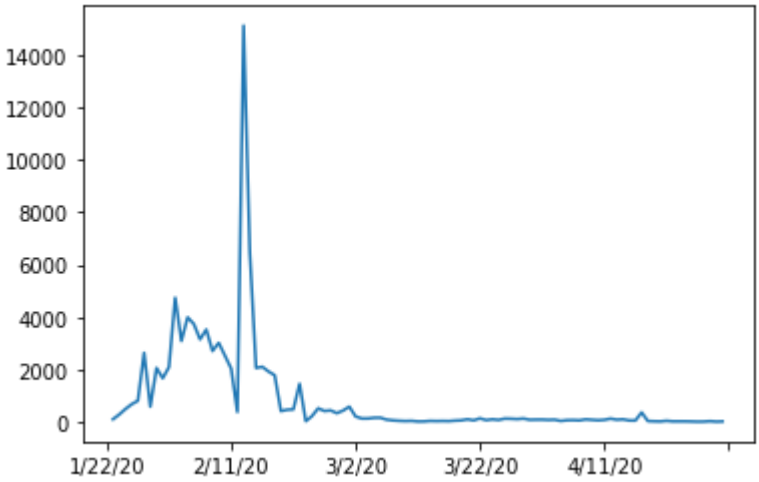
Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x21174f47448>



task 3.1: caculating the first derivative of the curve

In [32]:
```
corona_dataset_aggregated.loc['China'].diff().plot()
```

Out[32]: <matplotlib.axes._subplots.AxesSubplot at 0x21173325608>



task 3.2: find maxmimum infection rate for China

In [33]:
```
corona_dataset_aggregated.loc['China'].diff().max()
```

Out[33]: 15136.0

In [34]:
```
corona_dataset_aggregated.loc['Italy'].diff().max()
```

Out[34]: 6557.0

In [35]:
```
corona_dataset_aggregated.loc['Spain'].diff().max()
```

Out[35]: 9630.0

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

In [39]:
```
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 [40]:
```
corona_dataset_aggregated.head()
```

Out[40]:

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 × 101 columns

Task 3.4: create a new dataframe with only needed column

In [41]:
```
corona_data=pd.DataFrame(corona_dataset_aggregated['max infection rate'])
```

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

Out[42]:

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 [45]:
```
world_happiness_report=pd.read_csv("worldwide_happiness_report.csv")
```

In [46]:
```
world_happiness_report.head()
```

Out[46]:

	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 [47]:
```
world_happiness_report.shape
```

Out[47]:
```
(156, 9)
```

Task 4.2: let's drop the useless columns

In [48]:
```
columns_to_drop=['Overall rank','Score','Generosity','Perceptions of corruptio
n']
world_happiness_report.drop(columns_to_drop,axis=1,inplace=True)
```

In [49]:
```
world_happiness_report.head()
```

Out[49]:

	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 [50]:
```
world_happiness_report.shape
```

Out[50]:
```
(156, 5)
```

Task 4.3: changing the indices of the dataframe

In [51]:
```
world_happiness_report.set_index(['Country or region'],inplace=True)
world_happiness_report.head()
```

Out[51]:

	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
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 [52]:
```
corona_data.head()
```

Out[52]:

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

In [53]:
```
corona_data.shape
```

Out[53]: (187, 1)

wolrd happiness report Dataset :

In [54]:
```
world_happiness_report.head()
```

Out[54]:

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 [56]:
```
world_happiness_report.shape
```

Out[56]: (156, 4)

Merging Both Dataset:

In [57]:
```
data=corona_data.join(world_happiness_report,how='inner')
```

In [66]:
```
data.head()
```

Out[66]:

	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 [67]:
```
data.corr()
```

Out[67]:

	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 [68]:
```
data.head()
```

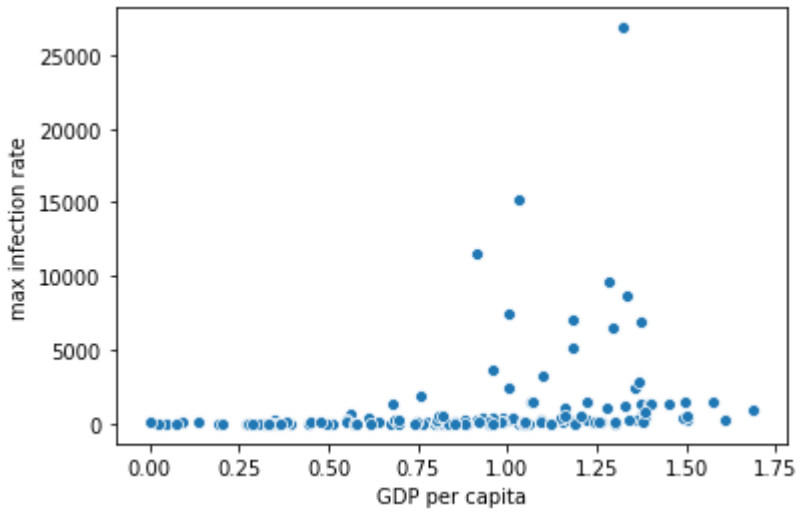
Out[68]:

	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 5.1: Plotting GDP vs maximum Infection rate

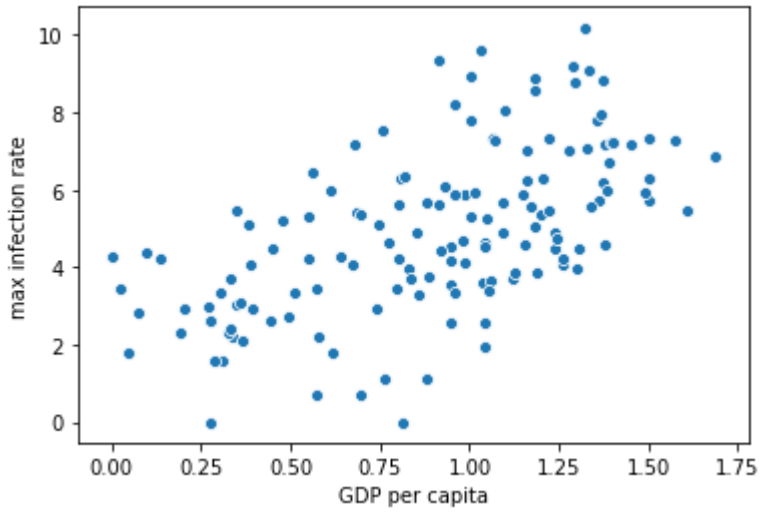
In [70]:
```
x=data["GDP per capita"]
y=data["max infection rate"]
sns.scatterplot(x,y)
```

Out[70]: <matplotlib.axes._subplots.AxesSubplot at 0x211747f3cc8>



In [71]:
```
x=data["GDP per capita"]
y=data["max infection rate"]
sns.scatterplot(x,np.log(y))
```

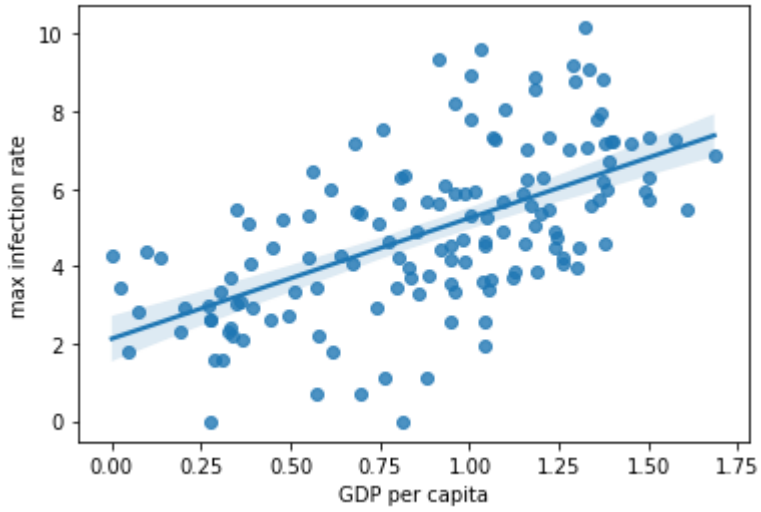
Out[71]: <matplotlib.axes._subplots.AxesSubplot at 0x21174a6f908>



for better visualisation::::

In [72]:
```
sns.regplot(x,np.log(y))
```

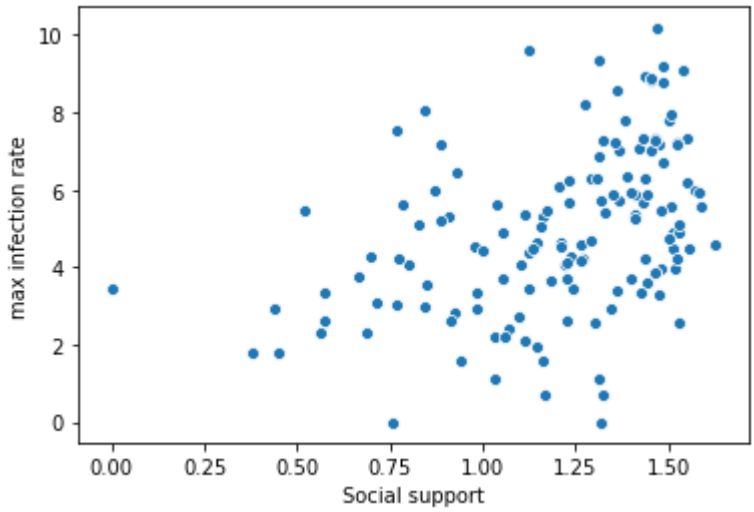
Out[72]: <matplotlib.axes._subplots.AxesSubplot at 0x21175483588>



Task 5.2: Plotting Social support vs maximum Infection rate

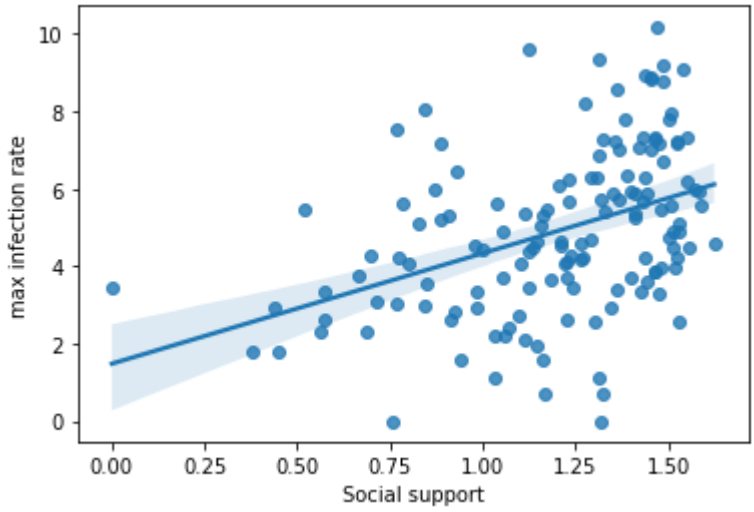
In [77]:
```
x = data['Social support']
y = data['max infection rate']
sns.scatterplot(x, np.log(y))
```

Out[77]: <matplotlib.axes._subplots.AxesSubplot at 0x2117660e088>



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

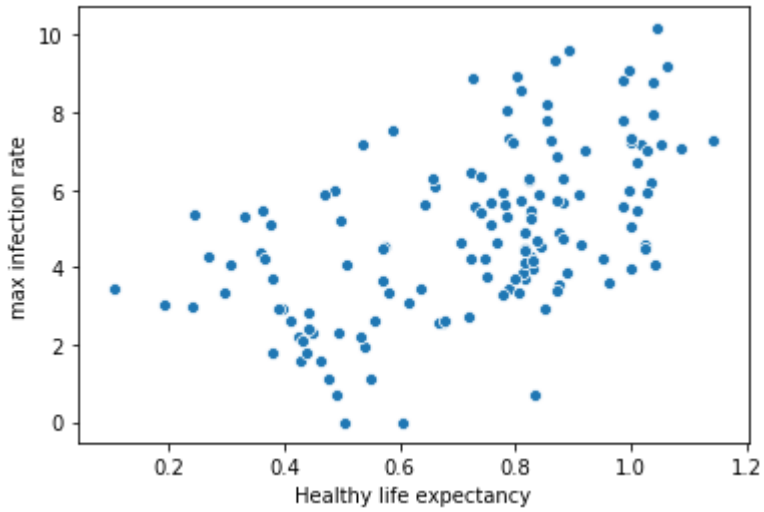
Out[79]: <matplotlib.axes._subplots.AxesSubplot at 0x21176f43dc8>



Task 5.3: Plotting Healthy life expectancy vs maximum Infection rate

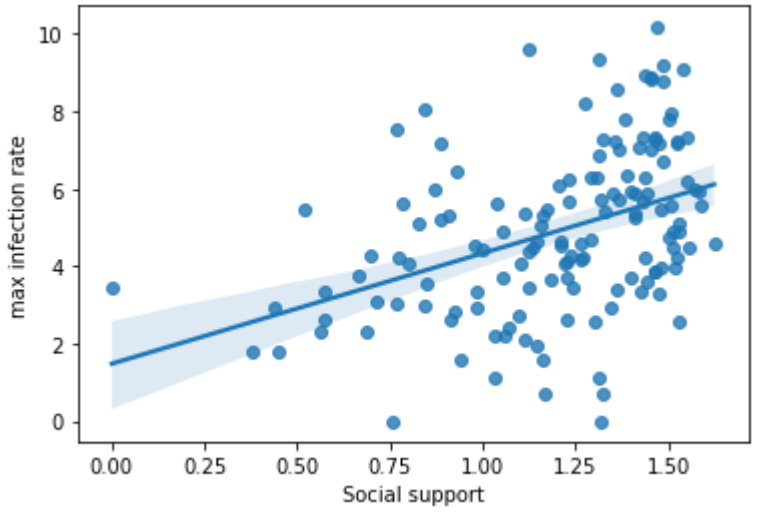
In [76]:
```
x = data['Healthy life expectancy']
y = data['max infection rate']
sns.scatterplot(x, np.log(y))
```

Out[76]: <matplotlib.axes._subplots.AxesSubplot at 0x21171d96388>



In [78]:
```
sns.regplot(x, np.log(y))
```

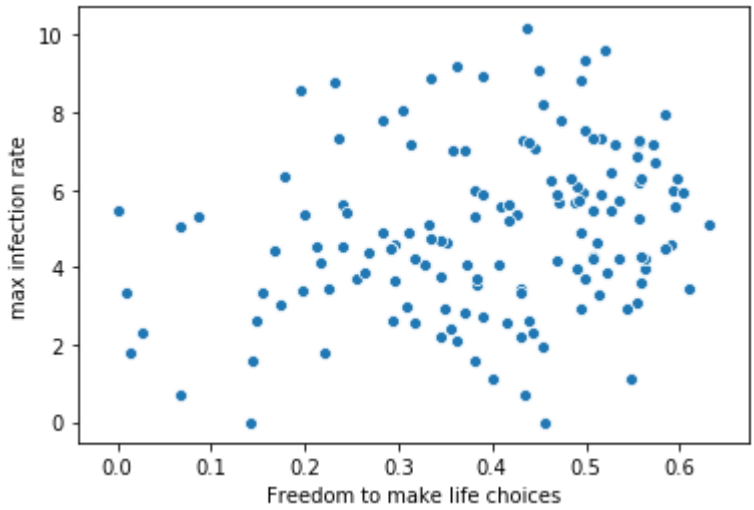
Out[78]: <matplotlib.axes._subplots.AxesSubplot at 0x2117660acc8>



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

In [73]:
```
x = data['Freedom to make life choices']
y = data['max infection rate']
sns.scatterplot(x, np.log(y))
```

Out[73]: <matplotlib.axes._subplots.AxesSubplot at 0x2117654c948>



In [74]:
```
sns.regplot(x, np.log(y))
```

Out[74]: <matplotlib.axes._subplots.AxesSubplot at 0x21177495dc8>