In [19]:
```
# import the libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [20]:
```
# import the dataset
dataset=pd.read_csv("covid19_Confirmed_dataset.csv")
dataset.head()
```

Out[20]:

	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

In [22]:
```
dataset.shape
```

Out[22]:
```
(266, 104)
```

In [23]:
```
# delete the useless columns
df=dataset.drop(["Lat","Long"],axis=1,inplace=True)
```

In [24]:
```
dataset.head()
```

Out[24]:

	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
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

In [25]:
```
# aggregate the rows by the country
corona_dataset_aggregated=dataset.groupby("Country/Region").sum()
```

In [26]:
```
corona_dataset_aggregated.head()
```

Out[26]:

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
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 [27]:
```
corona_dataset_aggregated.shape
```

Out[27]:
```
(187, 100)
```

In [30]:
```
# visualize data related to the country
corona_dataset_aggregated.loc["China"]
```

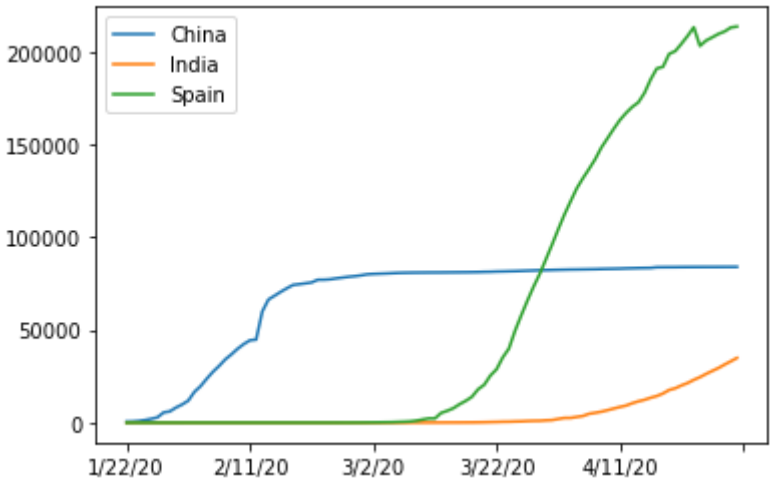
Out[30]:
```
1/22/20      548
1/23/20      643
1/24/20      920
1/25/20     1406
1/26/20     2075
           ...
4/26/20    83912
4/27/20    83918
4/28/20    83940
4/29/20    83944
4/30/20    83956
Name: China, Length: 100, dtype: int64
```

In [31]:
```
corona_dataset_aggregated.loc["China"].plot
```

Out[31]:
```
<pandas.plotting._core.PlotAccessor object at 0x000001EC60C2EC40>
```

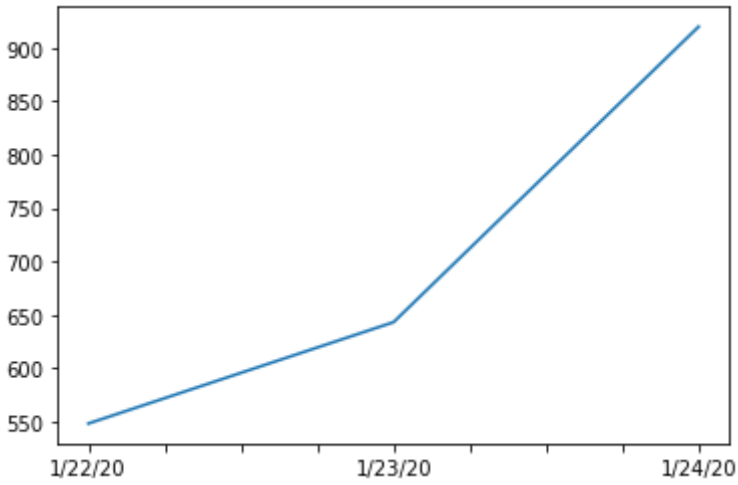
In [35]:
```
# visualize data related to the country
corona_dataset_aggregated.loc["China"].plot()
corona_dataset_aggregated.loc["India"].plot()
corona_dataset_aggregated.loc["Spain"].plot()
plt.legend()
```

Out[35]:
```
<matplotlib.legend.Legend at 0x1ec61694a60>
```



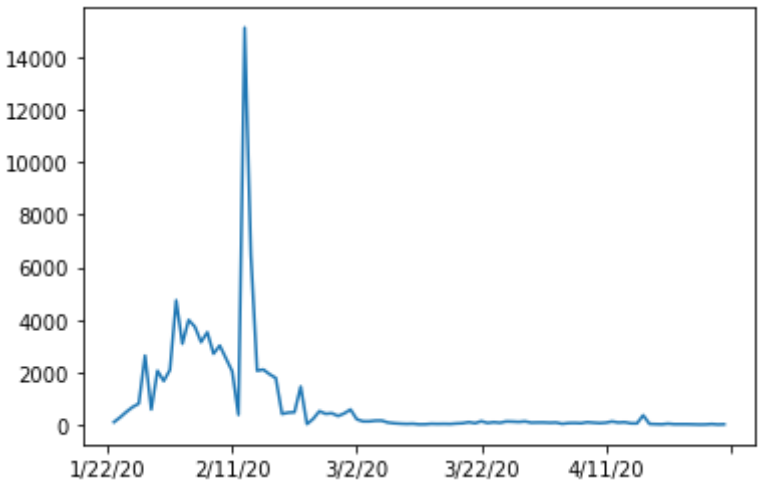
In [38]:
```
# calculate the good measure
corona_dataset_aggregated.loc["China"][:3].plot()
```

Out[38]: <AxesSubplot:>



In [39]:
```
# calculate the first derivative of the curve
corona_dataset_aggregated.loc["China"].diff().plot()
```

Out[39]: <AxesSubplot:>



In [40]:
```
# maximum infection rate
corona_dataset_aggregated.loc["China"].diff().max()
```

Out[40]: 15136.0

In [41]:
```
corona_dataset_aggregated.loc["India"].diff().max()
```

Out[41]: 1893.0

In [42]:
```
corona_dataset_aggregated.loc["Spain"].diff().max()
```

Out[42]: 9630.0

In [44]:
```
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_rates"]=max_infection_rates
```

In [45]:
```
corona_dataset_aggregated
```

Out[45]:

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
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
...	...	...	...	...	...	...	...	...	...
West Bank and Gaza	0	0	0	0	0	0	0	0	0
Western Sahara	0	0	0	0	0	0	0	0	0
Yemen	0	0	0	0	0	0	0	0	0
Zambia	0	0	0	0	0	0	0	0	0
Zimbabwe	0	0	0	0	0	0	0	0	0

187 rows × 101 columns

In [47]:
```
#create a new dataframe
corona_data=pd.DataFrame(corona_dataset_aggregated["Max_infection_rates"])
```

In [48]:
```
corona_data
```

Out[48]:

Country/Region	Max_infection_rates
Afghanistan	232.0

Country/Region	Max_infection_rates
Albania	34.0
Algeria	199.0
Andorra	43.0
Angola	5.0
...	...
West Bank and Gaza	66.0
Western Sahara	4.0
Yemen	5.0
Zambia	9.0
Zimbabwe	8.0

187 rows × 1 columns

In [49]:
```
#importing the dataset 2
happiness_report=pd.read_csv("worldwide_happiness_report.csv")
```

In [51]:
```
happiness_report
```

Out[51]:

	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
...	...	...	...	...	...	...	...	...	...
151	152	Rwanda	3.334	0.359	0.711	0.614	0.555	0.217	0.411
152	153	Tanzania	3.231	0.476	0.885	0.499	0.417	0.276	0.147
153	154	Afghanistan	3.203	0.350	0.517	0.361	0.000	0.158	0.025
154	155	Central African Republic	3.083	0.026	0.000	0.105	0.225	0.235	0.035
155	156	South Sudan	2.853	0.306	0.575	0.295	0.010	0.202	0.091

156 rows × 9 columns

In [52]:
```
#drop the useless columns
useless_col=["Overall rank","Score","Generosity","Perceptions of corruption"]
```

In [54]:
```
happiness_report.drop(useless_col,axis=1,inplace=True)
happiness_report
```

Out[54]:

	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
...	...	...	...	...	...
151	Rwanda	0.359	0.711	0.614	0.555
152	Tanzania	0.476	0.885	0.499	0.417
153	Afghanistan	0.350	0.517	0.361	0.000
154	Central African Republic	0.026	0.000	0.105	0.225
155	South Sudan	0.306	0.575	0.295	0.010

156 rows × 5 columns

In [55]:
```
happiness_report.set_index("Country or region",inplace=True)
happiness_report.head()
```

Out[55]:

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]:
```
# join the dataset
corona_data.shape
```

Out[56]:
```
(187, 1)
```

In [57]:
```
happiness_report.shape
```

Out[57]:
```
(156, 4)
```

In [59]:
```
data=corona_data.join(happiness_report,how="inner")
data
```

Out[59]:

	Max_infection_rates	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
...	...	...	...	...	...
Venezuela	29.0	0.960	1.427	0.805	0.154
Vietnam	19.0	0.741	1.346	0.851	0.543
Yemen	5.0	0.287	1.163	0.463	0.143
Zambia	9.0	0.578	1.058	0.426	0.431
Zimbabwe	8.0	0.366	1.114	0.433	0.361

143 rows × 5 columns

In [60]:
```
data.corr()
```

Out[60]:

	Max_infection_rates	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Max_infection_rates	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 [61]:
```
data
```

Out[61]:

	Max_infection_rates	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

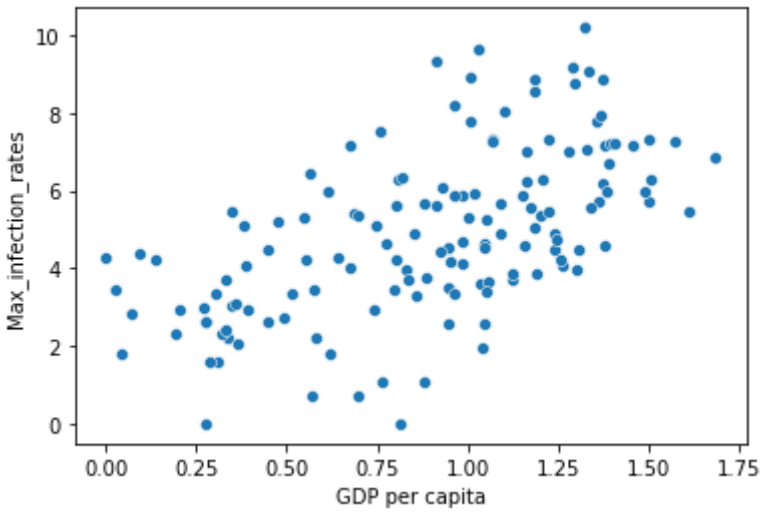
	Max_infection_rates	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
...	...	...	...	...	...
Venezuela	29.0	0.960	1.427	0.805	0.154
Vietnam	19.0	0.741	1.346	0.851	0.543
Yemen	5.0	0.287	1.163	0.463	0.143
Zambia	9.0	0.578	1.058	0.426	0.431
Zimbabwe	8.0	0.366	1.114	0.433	0.361

143 rows × 5 columns

In [66]:
```
#Visualization
x=data["GDP per capita"]
y=data["Max_infection_rates"]
sns.scatterplot(x,np.log(y))
```

```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```

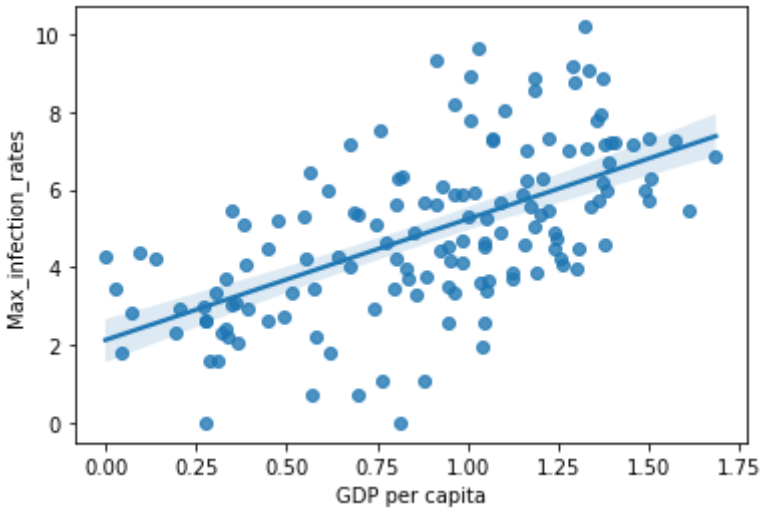
Out[66]:
```
<AxesSubplot:xlabel='GDP per capita', ylabel='Max_infection_rates'>
```



In [68]:
```
sns.regplot(x,np.log(y))
```

```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```

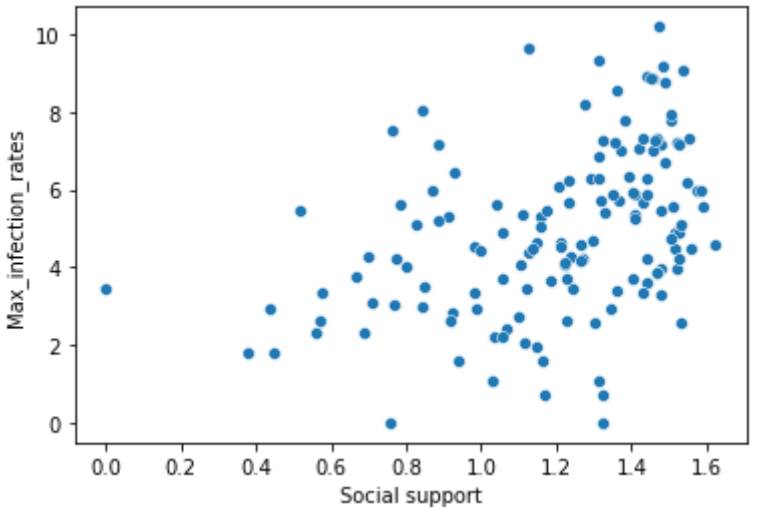
Out[68]:
```
<AxesSubplot:xlabel='GDP per capita', ylabel='Max_infection_rates'>
```



In [70]:
```
#Visualization
x=data["Social support"]
y=data["Max_infection_rates"]
sns.scatterplot(x,np.log(y))
```

```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```

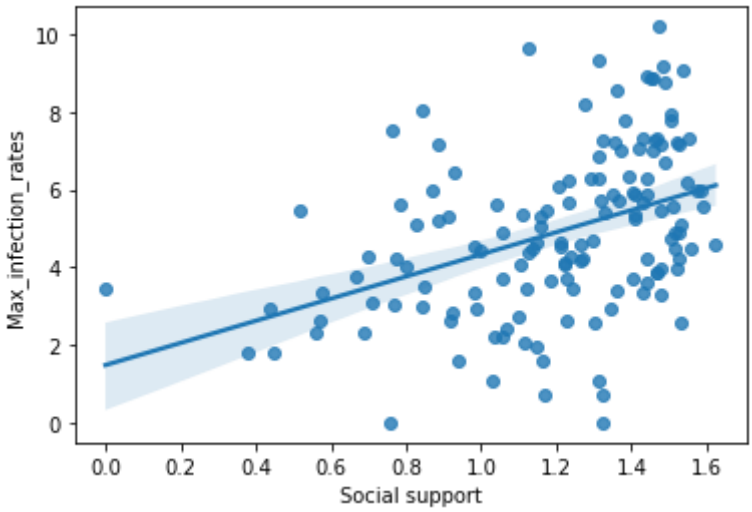
Out[70]:
```
<AxesSubplot:xlabel='Social support', ylabel='Max_infection_rates'>
```



In [71]:
```
sns.regplot(x,np.log(y))
```

```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```

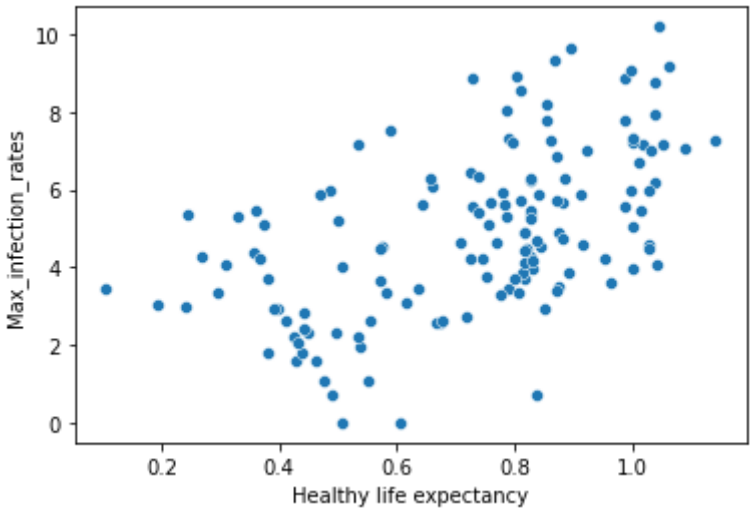
Out[71]:
```
<AxesSubplot:xlabel='Social support', ylabel='Max_infection_rates'>
```



In [72]:
```
#Visualization
x=data["Healthy life expectancy"]
y=data["Max_infection_rates"]
sns.scatterplot(x,np.log(y))
```

```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```

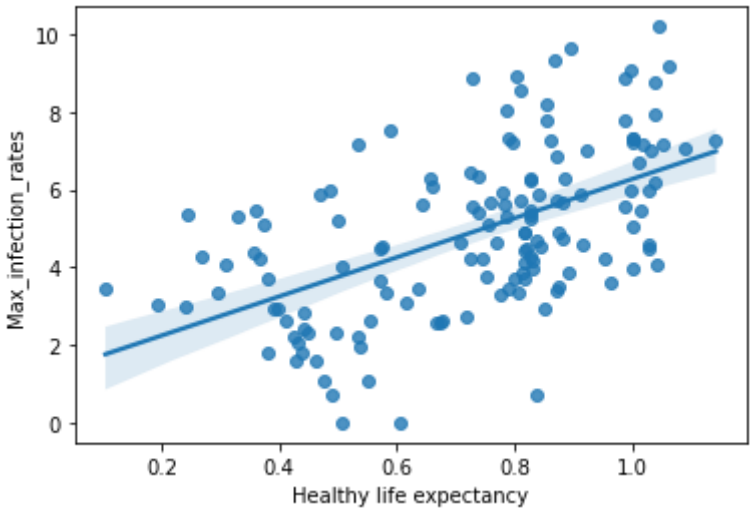
Out[72]:
```
<AxesSubplot:xlabel='Healthy life expectancy', ylabel='Max_infection_rates'>
```



In [73]:
```
sns.regplot(x,np.log(y))
```

```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```

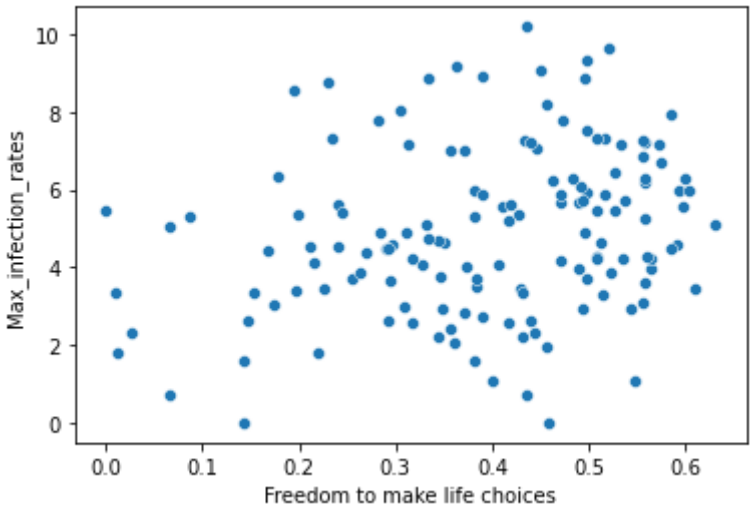
Out[73]:
```
<AxesSubplot:xlabel='Healthy life expectancy', ylabel='Max_infection_rates'>
```



In [74]:
```
#Visualization
x=data["Freedom to make life choices"]
y=data["Max_infection_rates"]
sns.scatterplot(x,np.log(y))
```

```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```

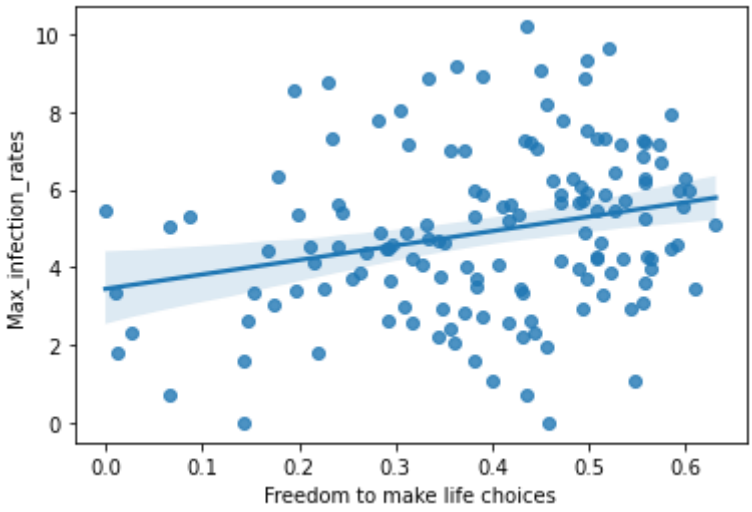
Out[74]:
```
<AxesSubplot:xlabel='Freedom to make life choices', ylabel='Max_infection_rates'>
```



In [75]:
```
sns.regplot(x,np.log(y))
```

```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[75]:
```
<AxesSubplot:xlabel='Freedom to make life choices', ylabel='Max_infection_rates'>
```


Max_infection_rates
Freedom to make life choices
0
2
4
6
8
10
0.0
0.1
0.2
0.3
0.4
0.5
0.6