

✓ Welcome to Covid19 Data Analysis Notebook

✓ Let's Import the modules

```
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.

```
corona_dataset_csv = pd.read_csv('covid19_Confirmed_dataset.csv')
corona_dataset_csv.head(187)
```

↗

	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	4/22/20	4/23/20
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0	0	...	1092	1176	12
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0	0	...	609	634	6
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0	0	...	2811	2910	30
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0	0	...	717	723	7
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0	0	...	24	25	
...	
182	NaN	Philippines	13.0000	122.0000	0	0	0	0	0	0	...	6599	6710	65
183	NaN	Poland	51.9194	19.1451	0	0	0	0	0	0	...	9856	10169	101
184	NaN	Portugal	39.3999	-8.2245	0	0	0	0	0	0	...	21379	21982	223
185	NaN	Qatar	25.3548	51.1839	0	0	0	0	0	0	...	6533	7141	71
186	NaN	Romania	45.9432	24.9668	0	0	0	0	0	0	...	9242	9710	100

187 rows × 104 columns

✓ Let's check the shape of the dataframe

```
corona_dataset_csv.shape
```

↗ (266, 104)

✓ Task 2.2: Delete the useless columns

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

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

	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	4/23/20
0	NaN	Afghanistan	0	0	0	0	0	0	0	0	...	1092	1176	1279
1	NaN	Albania	0	0	0	0	0	0	0	0	...	609	634	663
2	NaN	Algeria	0	0	0	0	0	0	0	0	...	2811	2910	3007
3	NaN	Andorra	0	0	0	0	0	0	0	0	...	717	723	723
4	NaN	Angola	0	0	0	0	0	0	0	0	...	24	25	25
5	NaN	Antigua and Barbuda	0	0	0	0	0	0	0	0	...	23	24	24
6	NaN	Argentina	0	0	0	0	0	0	0	0	...	3031	3144	3435
7	NaN	Armenia	0	0	0	0	0	0	0	0	...	1401	1473	1523
8	Australian Capital Territory	Australia	0	0	0	0	0	0	0	0	...	104	104	104
9	New South Wales	Australia	0	0	0	0	3	4	4	4	...	2969	2971	2976

10 rows × 102 columns

Task 2.3: Aggregating the rows by the country

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

```
corona_dataset_aggregated.head(187)
```

	Province/State	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	...	4/21/20	4/22/20	4/23/20
Country/Region														
Afghanistan		0	0	0	0	0	0	0	0	0	...	1092	1176	
Albania		0	0	0	0	0	0	0	0	0	...	609	634	
Algeria		0	0	0	0	0	0	0	0	0	...	2811	2910	
Andorra		0	0	0	0	0	0	0	0	0	...	717	723	
Angola		0	0	0	0	0	0	0	0	0	...	24	25	
...		
West Bank and Gaza		0	0	0	0	0	0	0	0	0	...	466	474	
Western Sahara		0	0	0	0	0	0	0	0	0	...	6	6	
Yemen		0	0	0	0	0	0	0	0	0	...	1	1	
Zambia		0	0	0	0	0	0	0	0	0	...	70	74	
Zimbabwe		0	0	0	0	0	0	0	0	0	...	28	28	

187 rows × 101 columns

```
corona_dataset_aggregated.shape
```

```
(187, 101)
```

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

visualization always helps for better understanding of our data.

```
print(corona_dataset_aggregated.head(186))
print(corona_dataset_aggregated.dtypes)
```

	Province/State	1/22/20	1/23/20	1/24/20	1/25/20	\
Country/Region						
Afghanistan		0	0	0	0	0
Albania		0	0	0	0	0

Algeria	0	0	0	0	0
Andorra	0	0	0	0	0
Angola	0	0	0	0	0
...
Vietnam	0	0	2	2	2
West Bank and Gaza	0	0	0	0	0
Western Sahara	0	0	0	0	0
Yemen	0	0	0	0	0
Zambia	0	0	0	0	0

	1/26/20	1/27/20	1/28/20	1/29/20	1/30/20	...	4/21/20	\
Country/Region						...		
Afghanistan	0	0	0	0	0	...	1092	
Albania	0	0	0	0	0	...	609	
Algeria	0	0	0	0	0	...	2811	
Andorra	0	0	0	0	0	...	717	
Angola	0	0	0	0	0	...	24	
...	
Vietnam	2	2	2	2	2	...	268	
West Bank and Gaza	0	0	0	0	0	...	466	
Western Sahara	0	0	0	0	0	...	6	
Yemen	0	0	0	0	0	...	1	
Zambia	0	0	0	0	0	...	70	

	4/22/20	4/23/20	4/24/20	4/25/20	4/26/20	4/27/20	\
Country/Region							
Afghanistan	1176	1279	1351	1463	1531	1703	
Albania	634	663	678	712	726	736	
Algeria	2910	3007	3127	3256	3382	3517	
Andorra	723	723	731	738	738	743	
Angola	25	25	25	25	26	27	
...	
Vietnam	268	268	270	270	270	270	
West Bank and Gaza	474	480	484	342	342	342	
Western Sahara	6	6	6	6	6	6	
Yemen	1	1	1	1	1	1	
Zambia	74	76	84	84	88	88	

	4/28/20	4/29/20	4/30/20
Country/Region			
Afghanistan	1828	1939	2171
Albania	750	766	773
Algeria	3649	3848	4006
Andorra	743	743	745
Angola	27	27	27
...
Vietnam	270	270	270
West Bank and Gaza	343	344	344
Western Sahara	6	6	6
Yemen	1	6	6
Zambia	95	97	106


[186 rows x 101 columns]
Province/State object

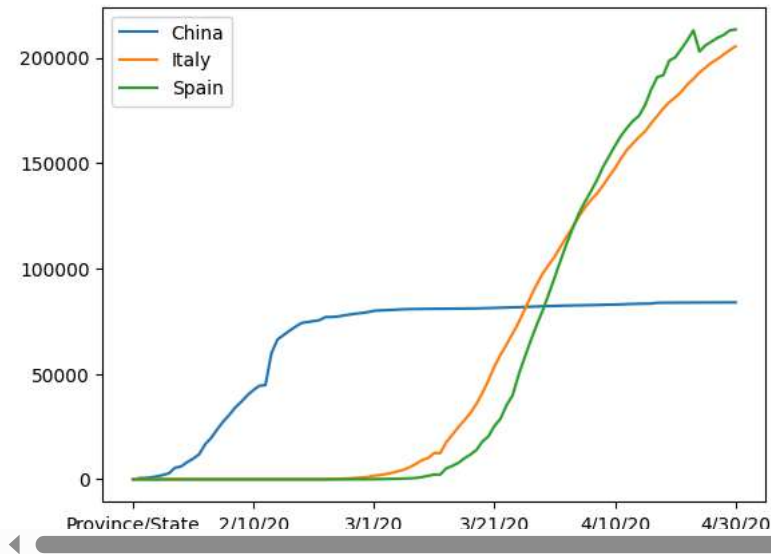
```
print(corona_dataset_aggregated.loc['China'])
```

```
Province/State  AnhuiBeijingChongqingFujianGansuGuangdongGuang...
1/22/20                548
1/23/20                643
1/24/20                920
1/25/20               1406
...
4/26/20                83912
4/27/20                83918
4/28/20                83940
4/29/20                83944
4/30/20                83956
Name: China, Length: 101, dtype: object
```

```
corona_dataset_aggregated = corona_dataset_aggregated.apply(pd.to_numeric, errors='coerce')
```

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


 <matplotlib.legend.Legend at 0x79450563fed0>

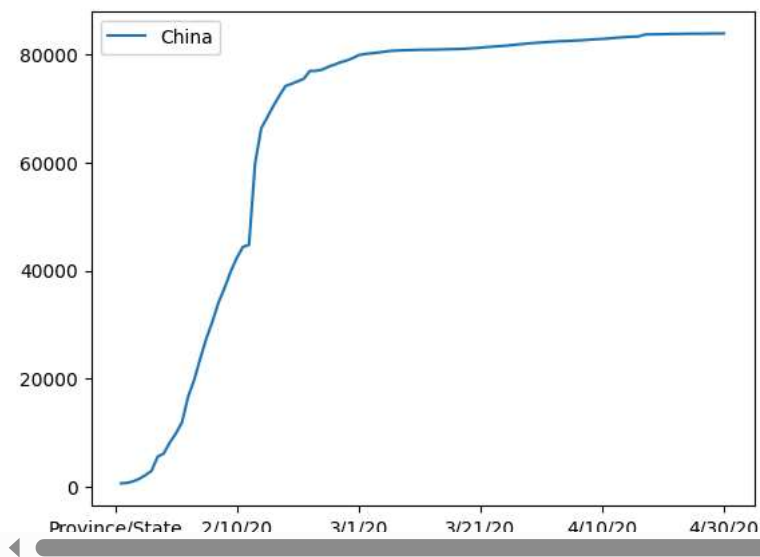


Task3: Calculating a good measure

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

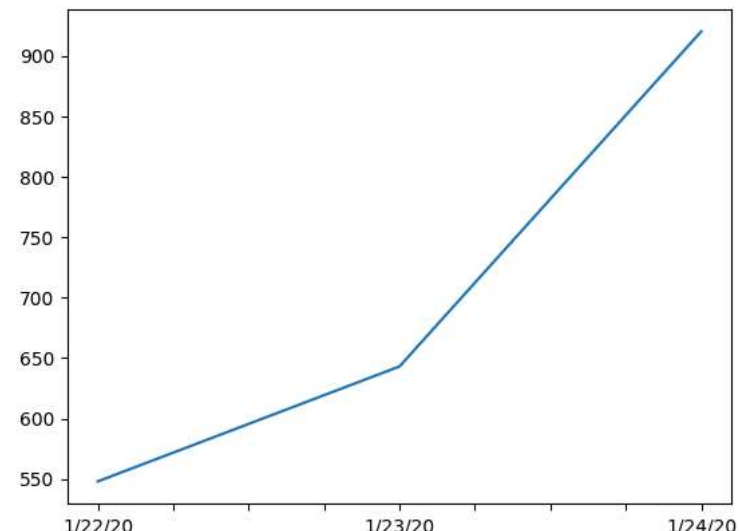
```
corona_dataset_aggregated.loc['China'].plot()
plt.legend()
```

 <matplotlib.legend.Legend at 0x794506bdb690>



```
corona_dataset_aggregated.loc["China"][:4].plot()
```

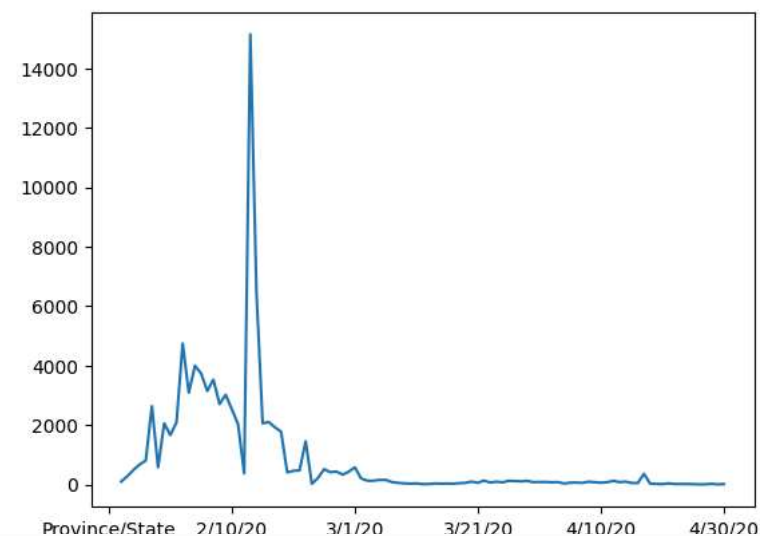
<Axes: >



task 3.1: caculating the first derivative of the curve

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

<Axes: >



task 3.2: find maximum infection rate for China

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

15136.0

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

6557.0

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


9630.0

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

```
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
```

```
corona_dataset_aggregated.head()
```



	Province/State	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	...	4/23/20	4/24/20	4/25/20
Country/Region														
0	Afghanistan	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	1279.0	1351.0	1423.0
1	Albania	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	663.0	678.0	693.0
2	Algeria	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	3007.0	3127.0	3247.0
3	Andorra	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	723.0	731.0	739.0
4	Angola	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	25.0	25.0	25.0

103 columns



Task 3.4: create a new dataframe with only needed column

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

```
corona_data.head()
```



	max_infection_rates
Country/Region	
0	Afghanistan 232.0
1	Albania 34.0
2	Algeria 199.0
3	Andorra 43.0
4	Angola 5.0



Next steps:

[Generate code with corona_data](#)[View recommended plots](#)[New interactive sheet](#)


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

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

```
happiness_report_csv.head()
```



	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



Next steps:

[Generate code with happiness_report_csv](#)[View recommended plots](#)[New interactive sheet](#)

Task 4.2: let's drop the useless columns

```
happiness_report_csv.drop(["Overall rank", "Score", "Generosity", "Perceptions of corruption"], axis=1, inplace=True)
happiness_report_csv.head()
```

	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

Next steps: [Generate code with happiness_report_csv](#) [View recommended plots](#) [New interactive sheet](#)

Task 4.3: changing the indices of the dataframe

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

```
happiness_report_csv.head()
```


	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

Next steps: [Generate code with happiness_report_csv](#) [View recommended plots](#) [New interactive sheet](#)

Task 4.4: now let's join two dataset we have prepared

Corona Dataset :

```
corona_data
```




max_infection_rates	
Country/Region	
Afghanistan	232.0
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

Next steps:


[Generate code with corona_data](#)[View recommended plots](#)[New interactive sheet](#)

corona_data.shape

 (187, 1)

▼ world happiness report Dataset :

happiness_report_csv




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

156 rows × 4 columns


Next steps:

[Generate code with happiness_report_csv](#)[View recommended plots](#)[New interactive sheet](#)


happiness_report_csv.shape

 (156, 4)

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

	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




Next steps:



[Generate code with data](#)[View recommended plots](#)[New interactive sheet](#)

Task 4.5: correlation matrix

```
data.corr()
```




	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	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

```
data.head()
```



	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

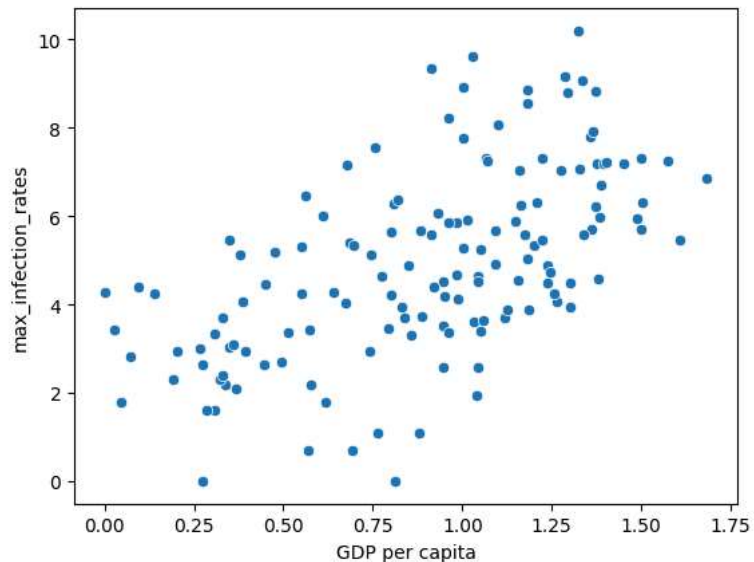
Next steps:

[Generate code with data](#)[View recommended plots](#)[New interactive sheet](#)

Task 5.1: Plotting GDP vs maximum Infection rate

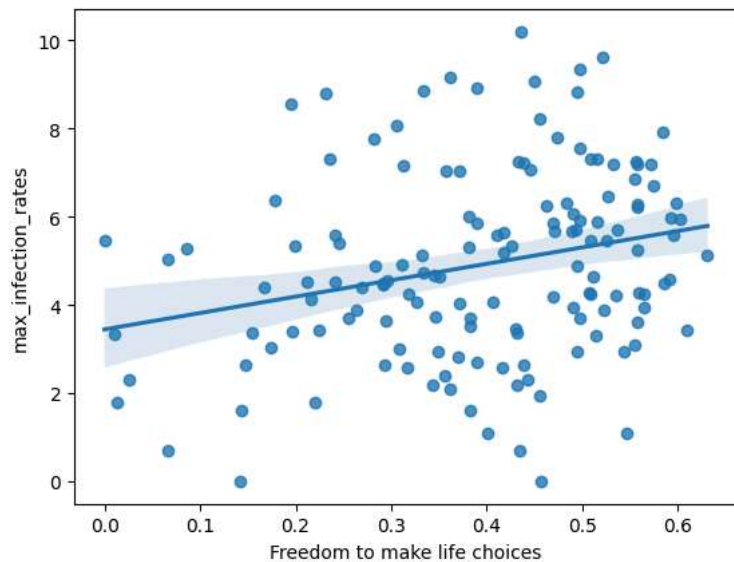
```
x = data["GDP per capita"]
y = data["max_infection_rates"]
sns.scatterplot(x=x,y=np.log(y))
```

```
<Axes: xlabel='GDP per capita', ylabel='max_infection_rates'>
```



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

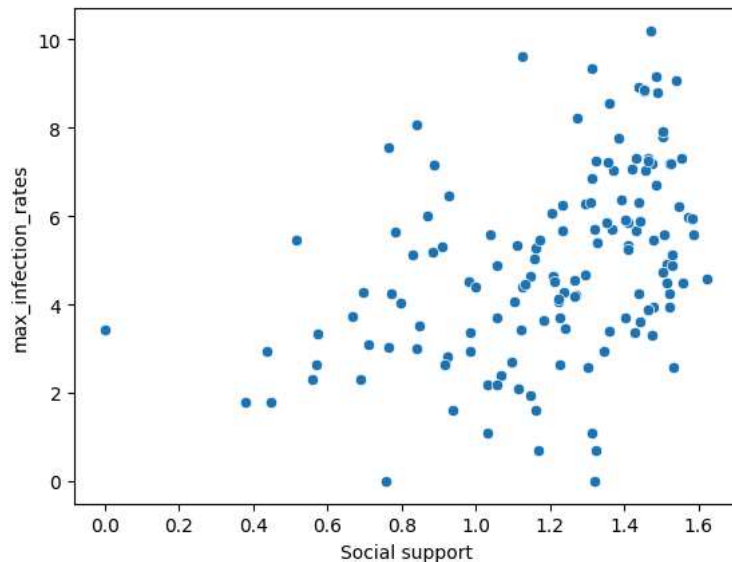
```
<Axes: xlabel='Freedom to make life choices', ylabel='max_infection_rates'>
```



Task 5.2: Plotting Social support vs maximum Infection rate

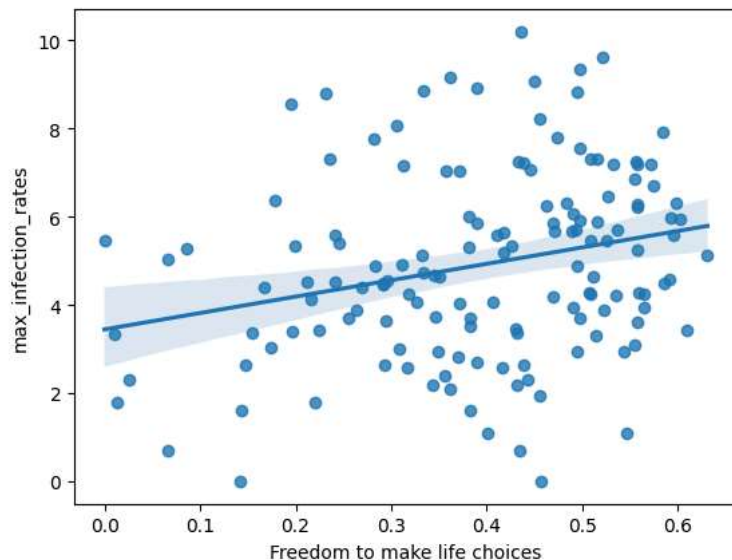
```
x = data["Social support"]
y = data["max_infection_rates"]
sns.scatterplot(x=x,y=np.log(y))
```

```
<Axes: xlabel='Social support', ylabel='max_infection_rates'>
```



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

```
<Axes: xlabel='Freedom to make life choices', ylabel='max_infection_rates'>
```



Task 5.3: Plotting Healthy life expectancy vs maximum Infection rate

```
x = data["Healthy life expectancy"]
y = data["max_infection_rates"]
sns.scatterplot(x=x,y=np.log(y))
```

```
<Axes: xlabel='Healthy life expectancy', ylabel='max_infection_rates'>
```



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

```
<Axes: xlabel='Freedom to make life choices', ylabel='max_infection_rates'>
```

