COVID 19 PROJECT – DATA SCIENCE CAPSTONE PROJECT

Introduction of COVID 19:

Covid 19 is a form of corona virus which has impacted the globe on large scale affecting millions of people and killing over a million people globally. The global economies are have faced downturn, which is amongst the worst in the history of mankind. This study of data science project is to understand the spread rate among humans and the causes of affecting the spread rate.

Data Science Methods:

During the course of our study, we will be doing the following steps.

1. Data Cleansing
2. Normalization and data Wrangling
3. Data Analysis
4. Data Visualization
5. Model Development.

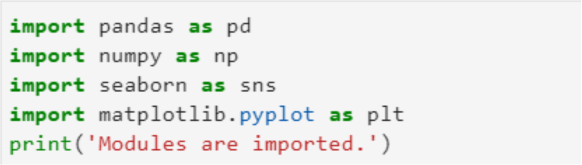
Source of Data:

Data is obtained from various media sources and education sites. We are going to understand the various data sources such as covid 19 infection rates, happiness index based on the countries and countries facing covid 19. We will study these data in detail and see how these data we interpret have a correlation. As we progress through the project we will dig deeper to understand every aspect of data. After the analysis, we will represent the information gathered in the form of visualizations.

We have two data sets provided for our analysis. One is the maximum infected cases data and the other is the world happiness report index. We will analyse the data and find the correlation between these two datasets.

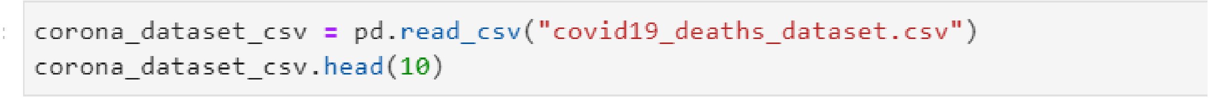
Import Modules:

Python has functions which run through modules. So we first run the modules, which helps us to do our analysis and visualizations. At first we read the data. The function we will be using pandas, matplot, seaborn for reading the data, analysing the data and do data visualization.

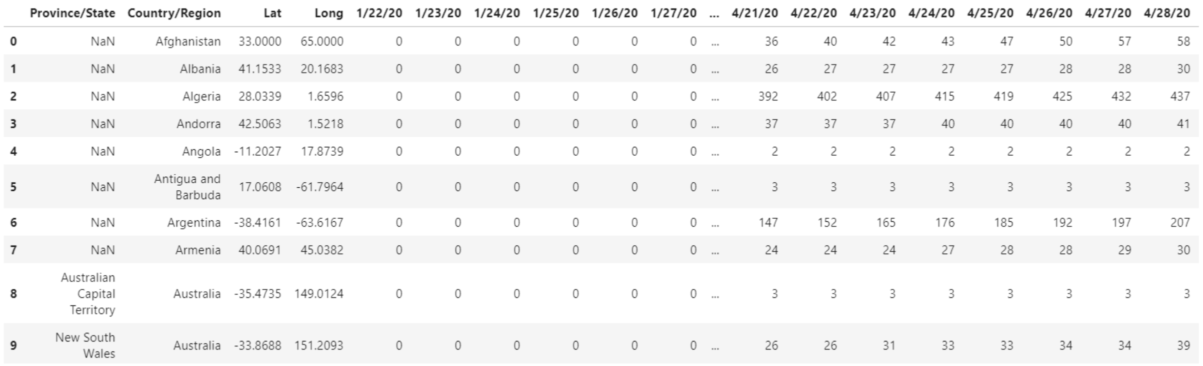


```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
print('Modules are imported.')
```

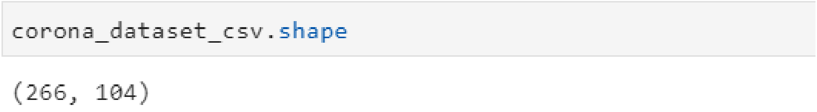
Using Pandas function we read the csv format file. After that use the head function to read the rows of the table. The head (10) represents the first 10 rows of the data we have read using pandas function. The figure shows the result of head functions. We can figure out the rows we need and based on our goal, we use only the columns necessary for our analysis and remove unnecessary data. Next step is to use the shape function to identify the number of rows and columns in table. Shape (x,y) where x is the number of rows and y is the number of columns. Refer to the figure for further reference.



```
corona_dataset_csv = pd.read_csv("covid19_deaths_dataset.csv")
corona_dataset_csv.head(10)
```



	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	4/24/20	4/25/20	4/26/20	4/27/20	4/28/20
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0	0	...	36	40	42	43	47	50	57	58
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0	0	...	26	27	27	27	27	28	28	30
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0	0	...	392	402	407	415	419	425	432	437
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0	0	...	37	37	37	40	40	40	40	41
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0	0	...	2	2	2	2	2	2	2	2
5	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	0	0	...	3	3	3	3	3	3	3	3
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	0	0	0	...	147	152	165	176	185	192	197	207
7	NaN	Armenia	40.0691	45.0382	0	0	0	0	0	0	...	24	24	24	27	28	28	29	30
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	0	0	0	...	3	3	3	3	3	3	3	3
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	0	0	0	...	26	26	31	33	33	34	34	39



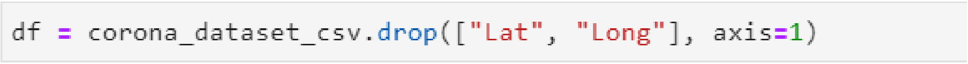
```
corona_dataset_csv.shape
```

```
(266, 104)
```

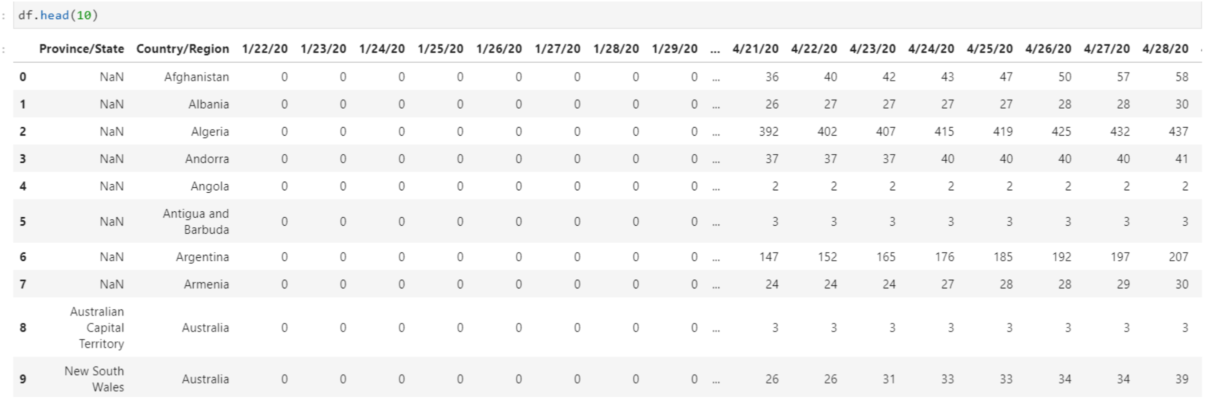
Data Cleansing:

As we obtain and gather data, we will see data in the raw form. This data is very difficult to gather information. At first we need to see the missing values and see how we can deal with these values. Either we can replace the missing values with the average value or replace with most frequently appearing values. Or if there are a few values missing, we can delete the rows depending on the importance.

We have to delete the unnecessary columns to make our analysis easy. In order to do this, we will use the drop function. We also mention the term axis, which refers to rows = 0 and columns =1. In this data set we remove the Lat and Long columns, as these data are not necessary for our analysis. We again use the head function to check if the columns were removed and we are happy to note that.



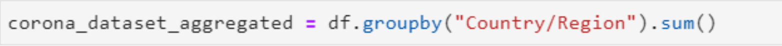
```
df = corona_dataset_csv.drop(["Lat", "Long"], axis=1)
```



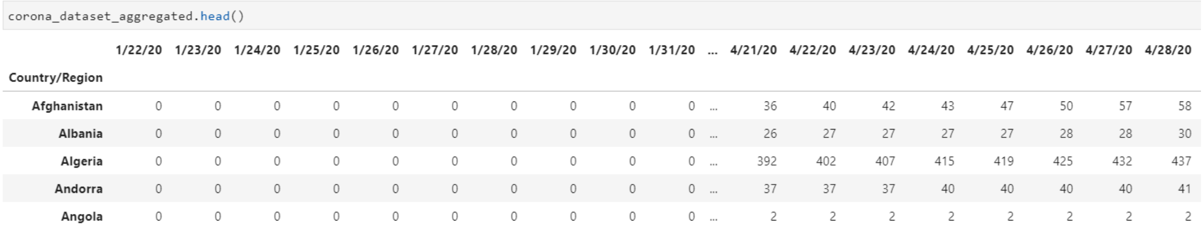
```
df.head(10)
```

	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	4/24/20	4/25/20	4/26/20	4/27/20	4/28/20
0	NaN	Afghanistan	0	0	0	0	0	0	0	0	...	36	40	42	43	47	50	57	58
1	NaN	Albania	0	0	0	0	0	0	0	0	...	26	27	27	27	27	28	28	30
2	NaN	Algeria	0	0	0	0	0	0	0	0	...	392	402	407	415	419	425	432	437
3	NaN	Andorra	0	0	0	0	0	0	0	0	...	37	37	37	40	40	40	40	41
4	NaN	Angola	0	0	0	0	0	0	0	0	...	2	2	2	2	2	2	2	2
5	NaN	Antigua and Barbuda	0	0	0	0	0	0	0	0	...	3	3	3	3	3	3	3	3
6	NaN	Argentina	0	0	0	0	0	0	0	0	...	147	152	165	176	185	192	197	207
7	NaN	Armenia	0	0	0	0	0	0	0	0	...	24	24	24	27	28	28	29	30
8	Australian Capital Territory	Australia	0	0	0	0	0	0	0	0	...	3	3	3	3	3	3	3	3
9	New South Wales	Australia	0	0	0	0	0	0	0	0	...	26	26	31	33	33	34	34	39

Now aggregate the data by the column Country/Region Now we create a dataset by the name corona_dataset_aggregated. We use the group function to aggregate the countries and regions with the dates to give the figure of the number of infected or affected cases of covid 19 date by date. Additionally, the sum function is used to get the total cases affected by the covid. We again use the shape function to check the number of rows and columns, which has 187 rows and 100 columns.

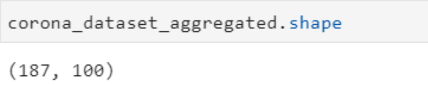


```
corona_dataset_aggregated = df.groupby("Country/Region").sum()
```



```
corona_dataset_aggregated.head()
```

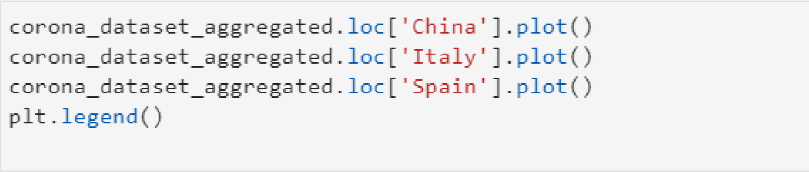
	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29/20	1/30/20	1/31/20	...	4/21/20	4/22/20	4/23/20	4/24/20	4/25/20	4/26/20	4/27/20	4/28/20
Country/Region																			
Afghanistan	0	0	0	0	0	0	0	0	0	0	...	36	40	42	43	47	50	57	58
Albania	0	0	0	0	0	0	0	0	0	0	...	26	27	27	27	27	28	28	30
Algeria	0	0	0	0	0	0	0	0	0	0	...	392	402	407	415	419	425	432	437
Andorra	0	0	0	0	0	0	0	0	0	0	...	37	37	37	40	40	40	40	41
Angola	0	0	0	0	0	0	0	0	0	0	...	2	2	2	2	2	2	2	2



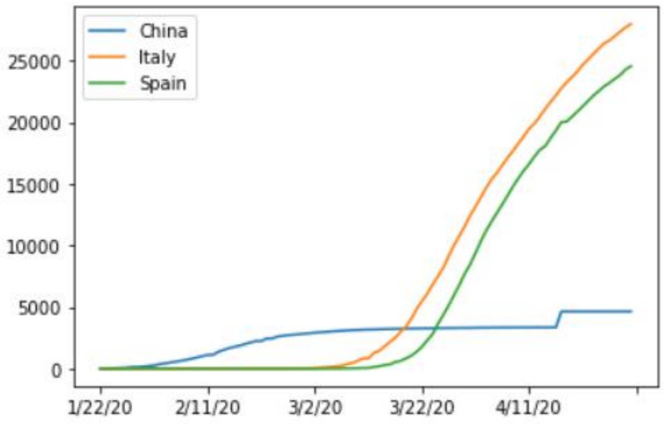
```
corona_dataset_aggregated.shape

(187, 100)
```

Data Visualizations:

This section involves plotting the aggregated data for various countries. Three countries China, Italy and Spain are plotted for data visualization. The below figures shows the code and the data visualization. The function .loc takes the corresponding values of the country and plotted using plot function. The plt.legend() function distinguishes the colours used for each country.

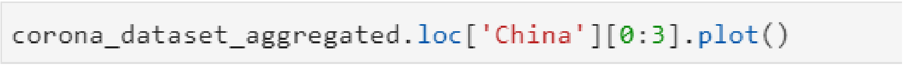


```
corona_dataset_aggregated.loc['China'].plot()
corona_dataset_aggregated.loc['Italy'].plot()
corona_dataset_aggregated.loc['Spain'].plot()
plt.legend()
```

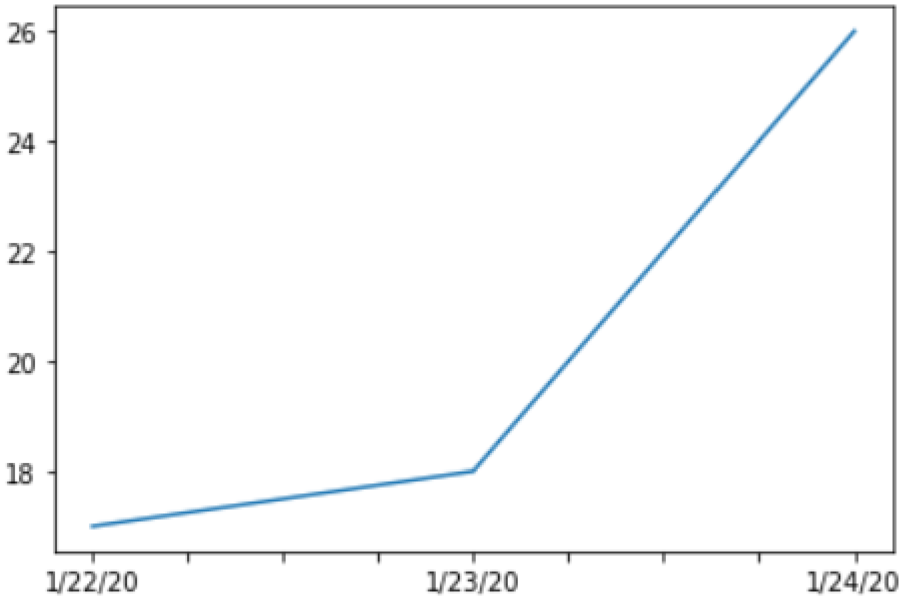


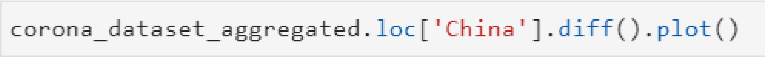
Now to get in deeper, between plot and loc function, [0:3} is used to refer the first three rows to show the first three dates for plotting data. To understand the peaks and the data better, we use the first derivative. This gives us the inner details of the visualizations. This is done using the function diff(). Then for finding the values using the max function, we will be able to find the day with maximum covid 19 cases. Refer to the following figures for reference.

To obtain data for all countries, we use the loop function. In this, we use the for loop function. We create the loop and obtain the data for all the countries. This loop creates a table for the max covid cases in a week. By creating a dataframe, we will be able to able to display the data in tabular form. Refer to the below snapshots for reference.

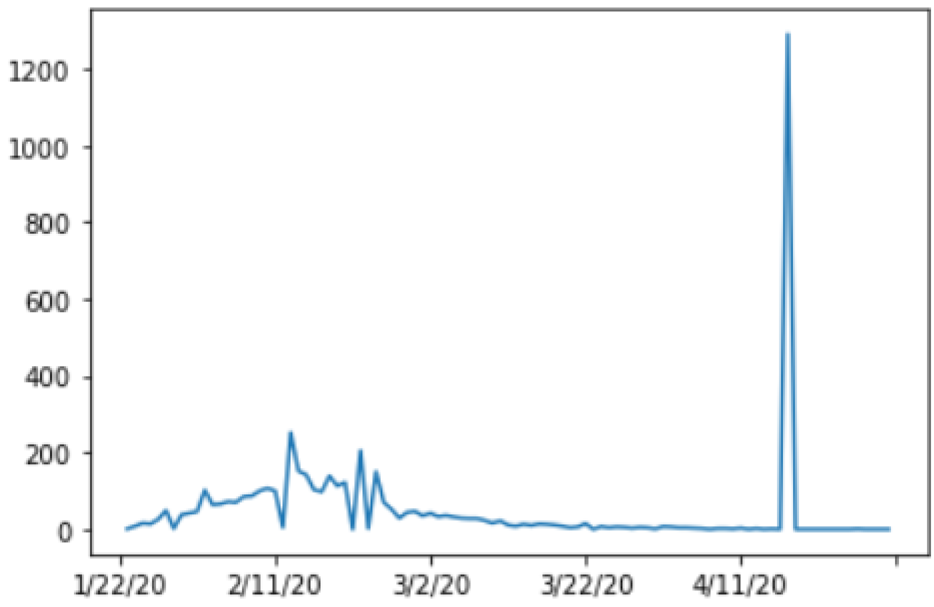


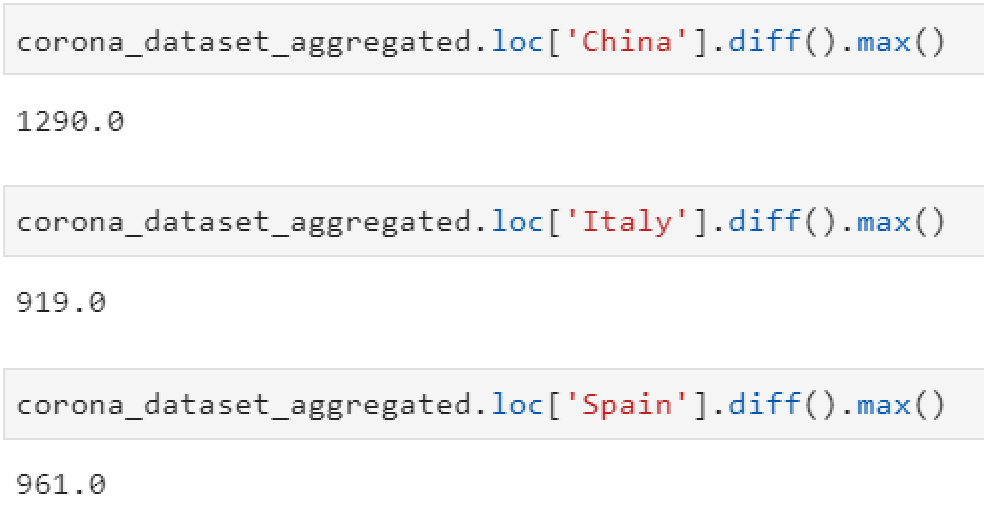
```
corona_dataset_aggregated.loc['China'][0:3].plot()
```





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





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

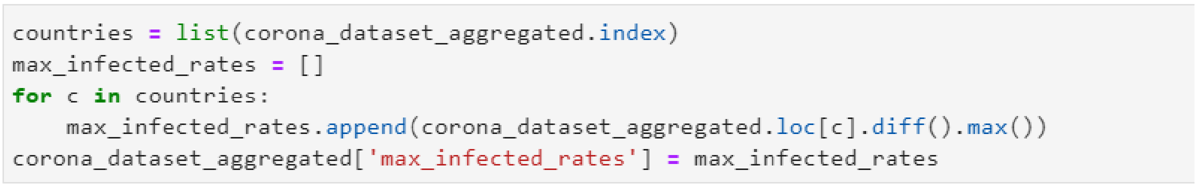
```
1290.0
```

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

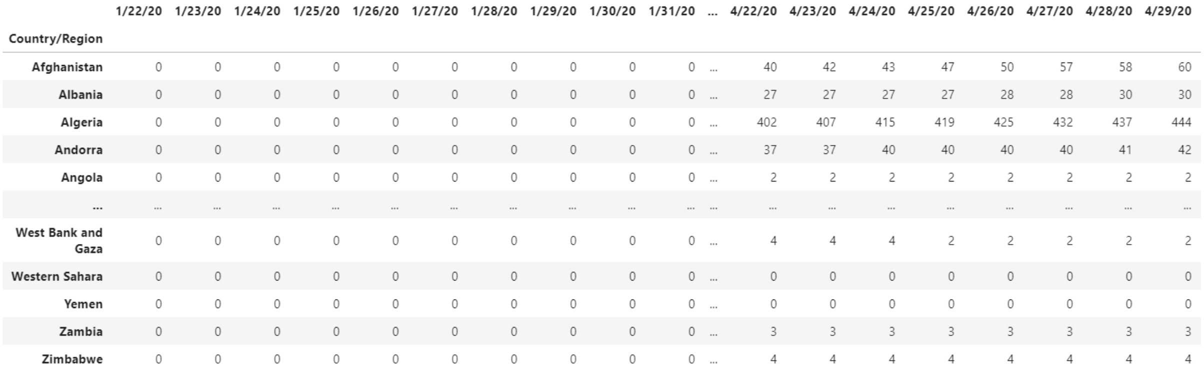
```
919.0
```

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

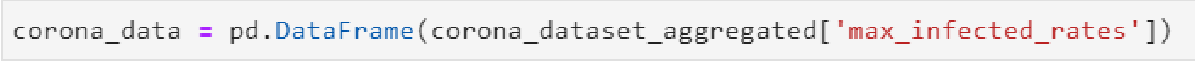
```
961.0
```



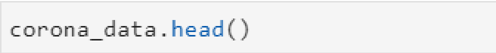
```
countries = list(corona_dataset_aggregated.index)
max_infected_rates = []
for c in countries:
    max_infected_rates.append(corona_dataset_aggregated.loc[c].diff().max())
corona_dataset_aggregated['max_infected_rates'] = max_infected_rates
```



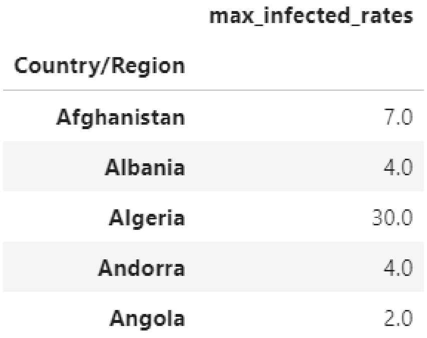
	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29/20	1/30/20	1/31/20	...	4/22/20	4/23/20	4/24/20	4/25/20	4/26/20	4/27/20	4/28/20	4/29/20
Country/Region																			
Afghanistan	0	0	0	0	0	0	0	0	0	0	...	40	42	43	47	50	57	58	60
Albania	0	0	0	0	0	0	0	0	0	0	...	27	27	27	27	28	28	30	30
Algeria	0	0	0	0	0	0	0	0	0	0	...	402	407	415	419	425	432	437	444
Andorra	0	0	0	0	0	0	0	0	0	0	...	37	37	40	40	40	40	41	42
Angola	0	0	0	0	0	0	0	0	0	0	...	2	2	2	2	2	2	2	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
West Bank and Gaza	0	0	0	0	0	0	0	0	0	0	...	4	4	4	2	2	2	2	2
Western Sahara	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
Yemen	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
Zambia	0	0	0	0	0	0	0	0	0	0	...	3	3	3	3	3	3	3	3
Zimbabwe	0	0	0	0	0	0	0	0	0	0	...	4	4	4	4	4	4	4	4



```
corona_data = pd.DataFrame(corona_dataset_aggregated['max_infected_rates'])
```



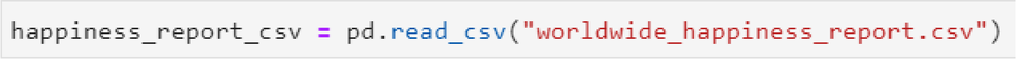
```
corona_data.head()
```



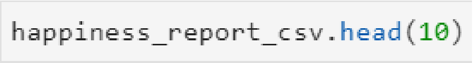
	max_infected_rates
Country/Region	
Afghanistan	7.0
Albania	4.0
Algeria	30.0
Andorra	4.0
Angola	2.0

World Happiness Report Index

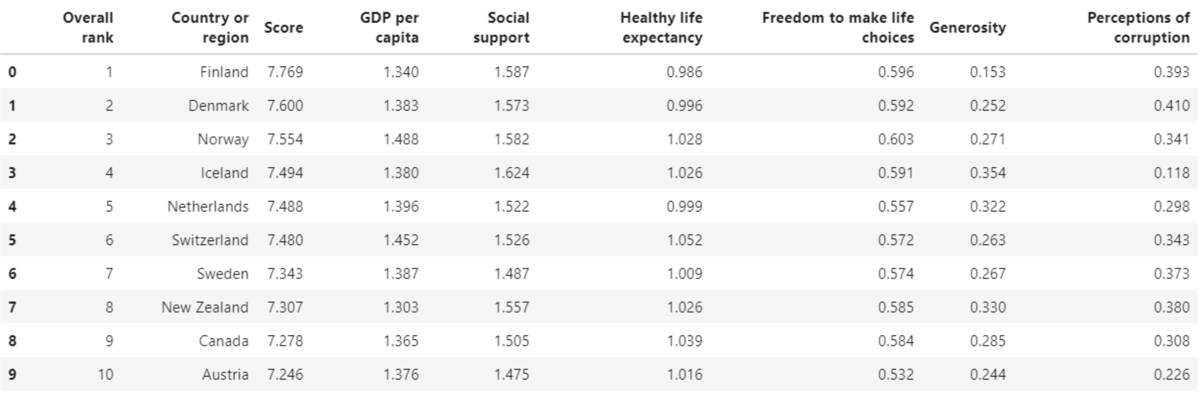
The world happiness report index is very similar to our previous data exploration. The below snapshots' can be referred for further explanation. After our data exploration, we will join the two tables using the join function, with a common index Country/Regions. Once joined, we perform correlations to check if the data between these two tables have any relation.



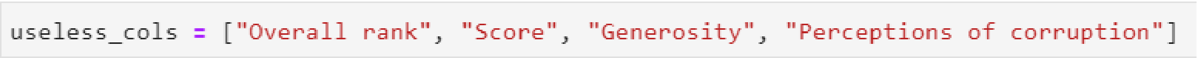
```
happiness_report_csv = pd.read_csv("worldwide_happiness_report.csv")
```



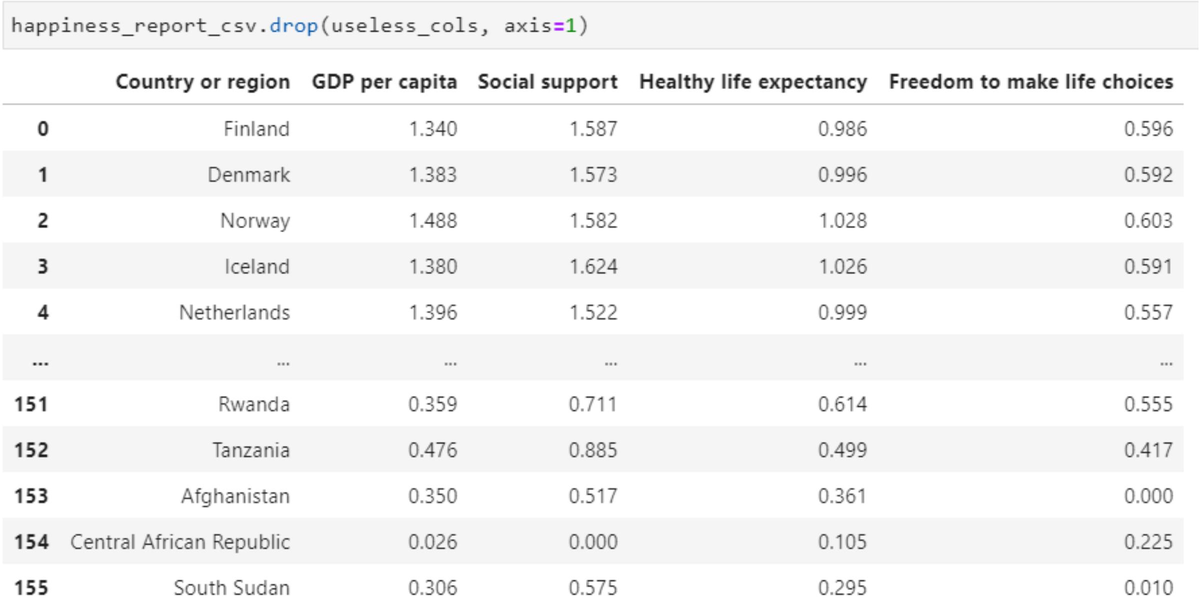
```
happiness_report_csv.head(10)
```



	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
5	6	Switzerland	7.480	1.452	1.526	1.052	0.572	0.263	0.343
6	7	Sweden	7.343	1.387	1.487	1.009	0.574	0.267	0.373
7	8	New Zealand	7.307	1.303	1.557	1.026	0.585	0.330	0.380
8	9	Canada	7.278	1.365	1.505	1.039	0.584	0.285	0.308
9	10	Austria	7.246	1.376	1.475	1.016	0.532	0.244	0.226

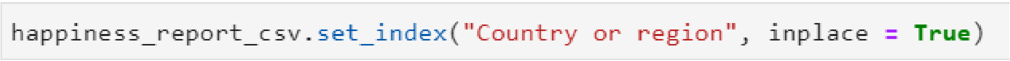


```
useless_cols = ["Overall rank", "Score", "Generosity", "Perceptions of corruption"]
```

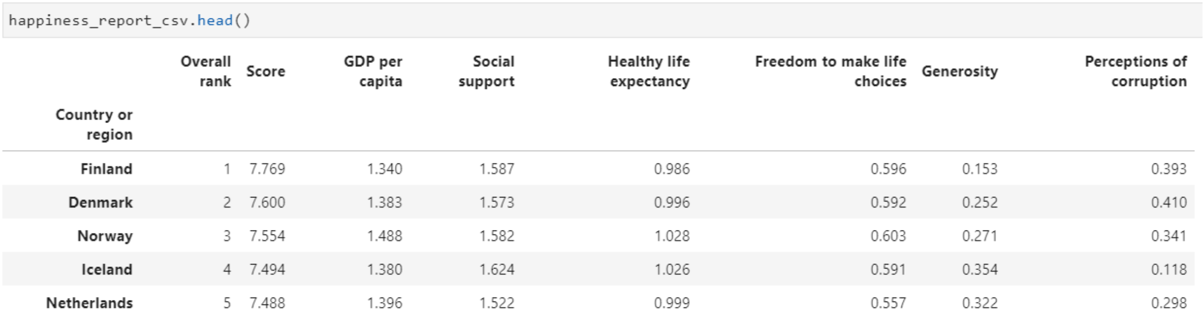


```
happiness_report_csv.drop(useless_cols, axis=1)
```

	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

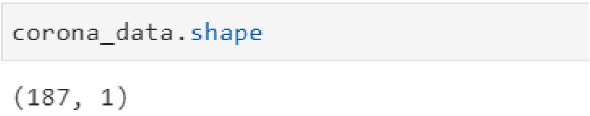


```
happiness_report_csv.set_index("Country or region", inplace = True)
```



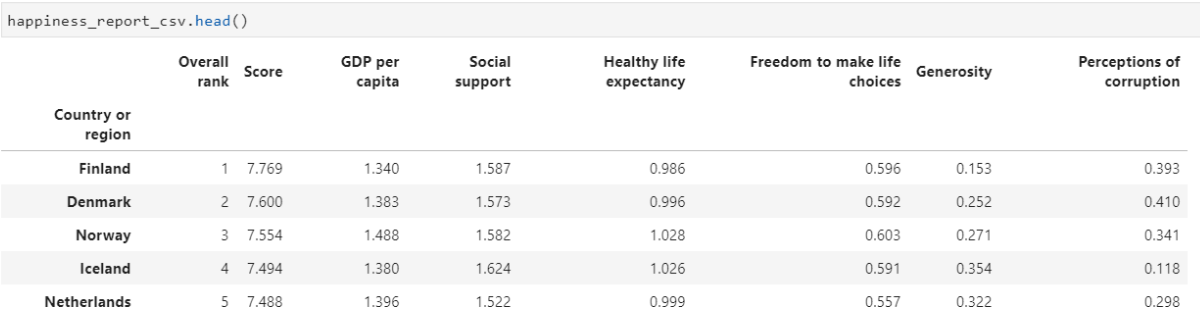
```
happiness_report_csv.head()
```

	Overall rank	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
Country or region								
Finland	1	7.769	1.340	1.587	0.986	0.596	0.153	0.393
Denmark	2	7.600	1.383	1.573	0.996	0.592	0.252	0.410
Norway	3	7.554	1.488	1.582	1.028	0.603	0.271	0.341
Iceland	4	7.494	1.380	1.624	1.026	0.591	0.354	0.118
Netherlands	5	7.488	1.396	1.522	0.999	0.557	0.322	0.298



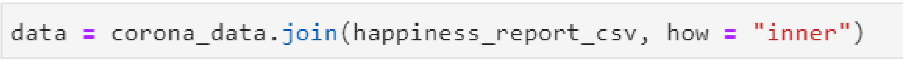
```
corona_data.shape
```

```
(187, 1)
```

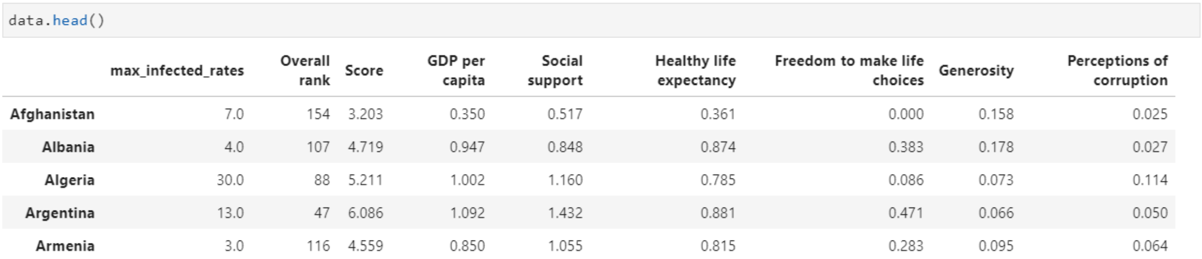


```
happiness_report_csv.head()
```

	Overall rank	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
Country or region								
Finland	1	7.769	1.340	1.587	0.986	0.596	0.153	0.393
Denmark	2	7.600	1.383	1.573	0.996	0.592	0.252	0.410
Norway	3	7.554	1.488	1.582	1.028	0.603	0.271	0.341
Iceland	4	7.494	1.380	1.624	1.026	0.591	0.354	0.118
Netherlands	5	7.488	1.396	1.522	0.999	0.557	0.322	0.298

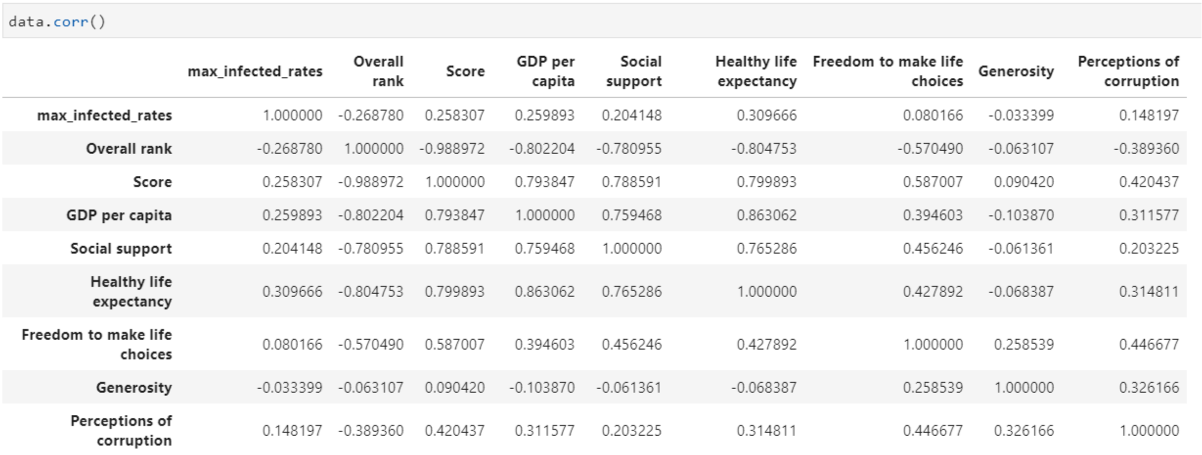


```
data = corona_data.join(happiness_report_csv, how = "inner")
```



```
data.head()
```

	max_infected_rates	Overall rank	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
Afghanistan	7.0	154	3.203	0.350	0.517	0.361	0.000	0.158	0.025
Albania	4.0	107	4.719	0.947	0.848	0.874	0.383	0.178	0.027
Algeria	30.0	88	5.211	1.002	1.160	0.785	0.086	0.073	0.114
Argentina	13.0	47	6.086	1.092	1.432	0.881	0.471	0.066	0.050
Armenia	3.0	116	4.559	0.850	1.055	0.815	0.283	0.095	0.064

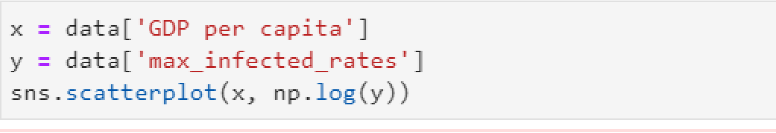


```
data.corr()
```

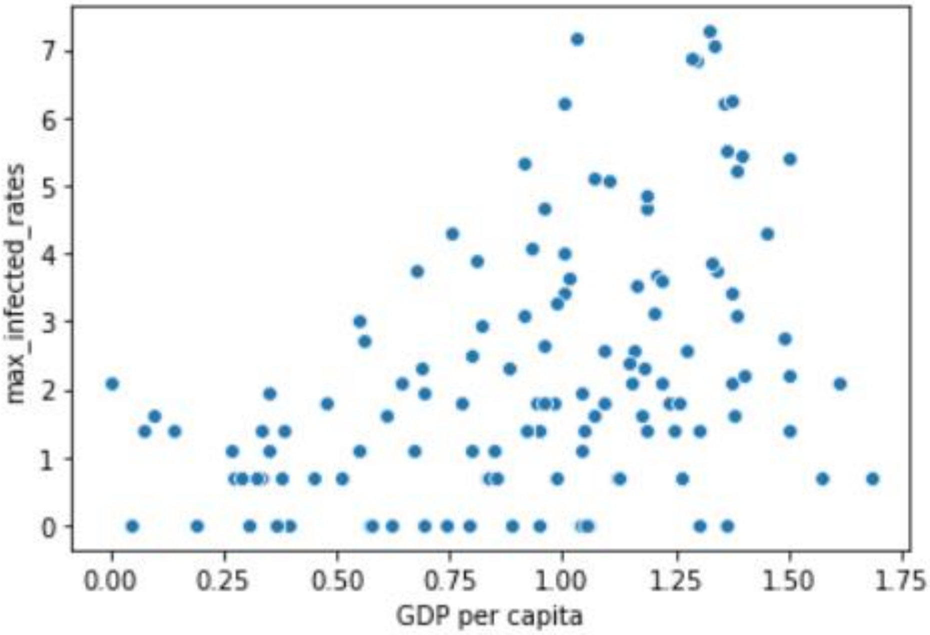
	max_infected_rates	Overall rank	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
max_infected_rates	1.000000	-0.268780	0.258307	0.259893	0.204148	0.309666	0.080166	-0.033399	0.148197
Overall rank	-0.268780	1.000000	-0.988972	-0.802204	-0.780955	-0.804753	-0.570490	-0.063107	-0.389360
Score	0.258307	-0.988972	1.000000	0.793847	0.788591	0.799893	0.587007	0.090420	0.420437
GDP per capita	0.259893	-0.802204	0.793847	1.000000	0.759468	0.863062	0.394603	-0.103870	0.311577
Social support	0.204148	-0.780955	0.788591	0.759468	1.000000	0.765286	0.456246	-0.061361	0.203225
Healthy life expectancy	0.309666	-0.804753	0.799893	0.863062	0.765286	1.000000	0.427892	-0.068387	0.314811
Freedom to make life choices	0.080166	-0.570490	0.587007	0.394603	0.456246	0.427892	1.000000	0.258539	0.446677
Generosity	-0.033399	-0.063107	0.090420	-0.103870	-0.061361	-0.068387	0.258539	1.000000	0.326166
Perceptions of corruption	0.148197	-0.389360	0.420437	0.311577	0.203225	0.314811	0.446677	0.326166	1.000000

Data Visualization of the joined data:

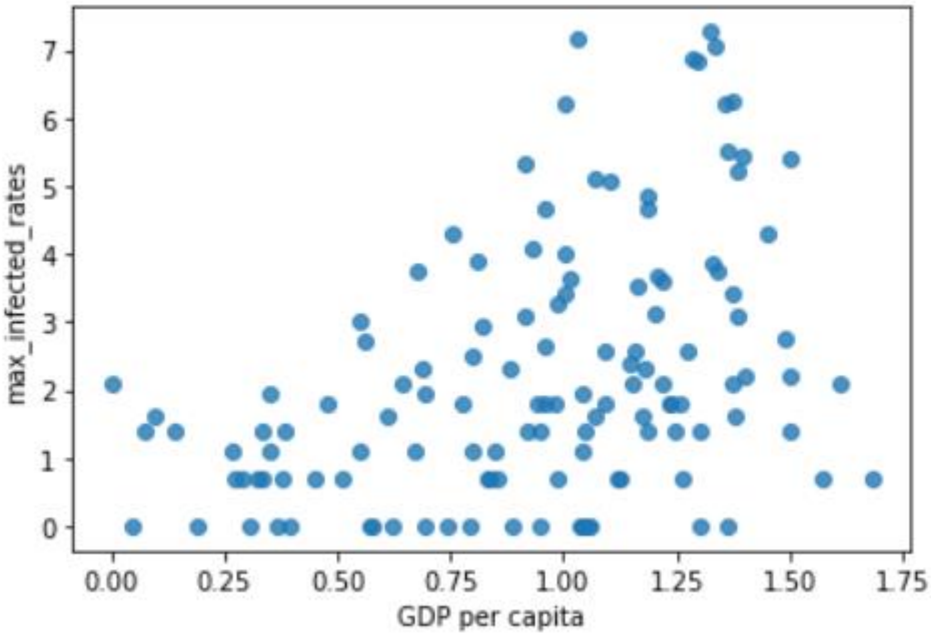
With these joined data, we do data visualizations such as comparing maximum infected cases with GDP per capita, social capita and health life support. Using np.log for y axis, we can get a detailed comparison based on the correlations' of the data.

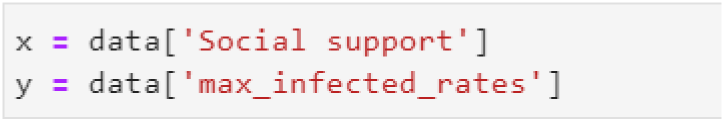


```
x = data['GDP per capita']
y = data['max_infected_rates']
sns.scatterplot(x, np.log(y))
```

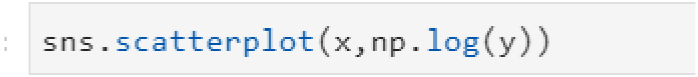


```
sns.regplot(x,np.log(y))
```

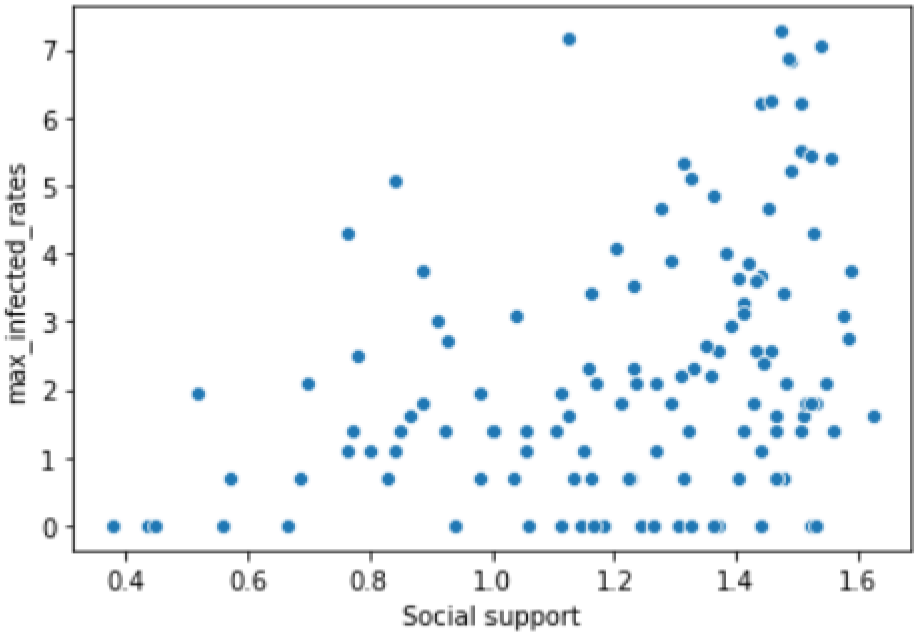


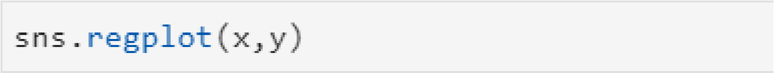


```
x = data['Social support']
y = data['max_infected_rates']
```

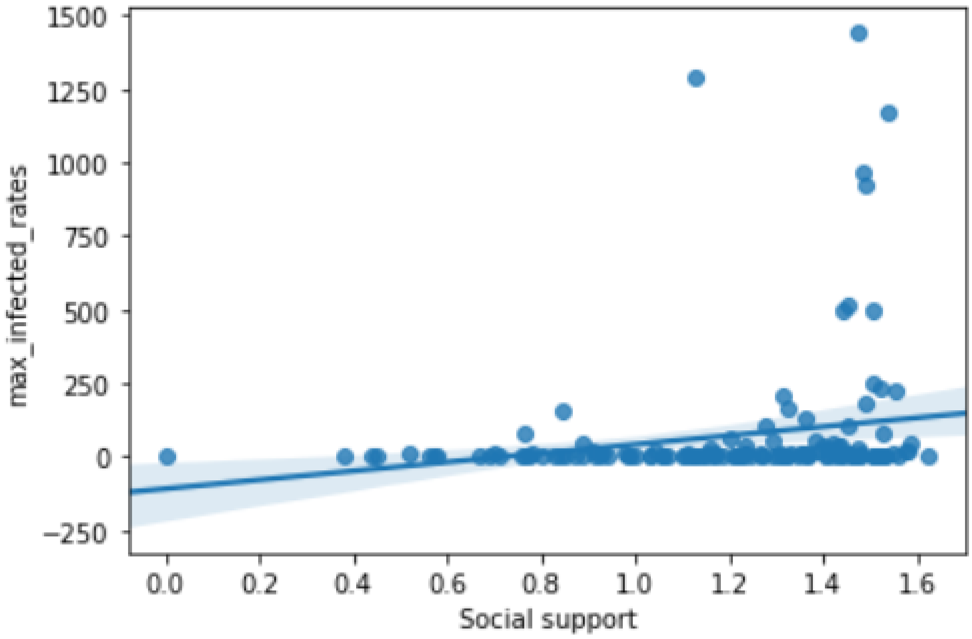


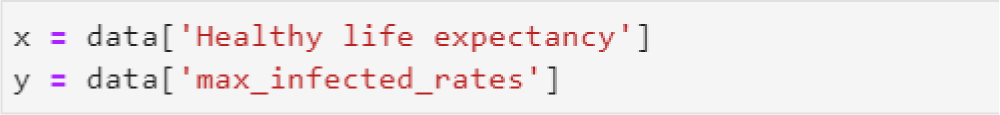
```
sns.scatterplot(x,np.log(y))
```



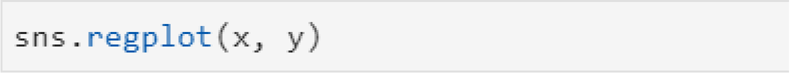


```
sns.regplot(x,y)
```

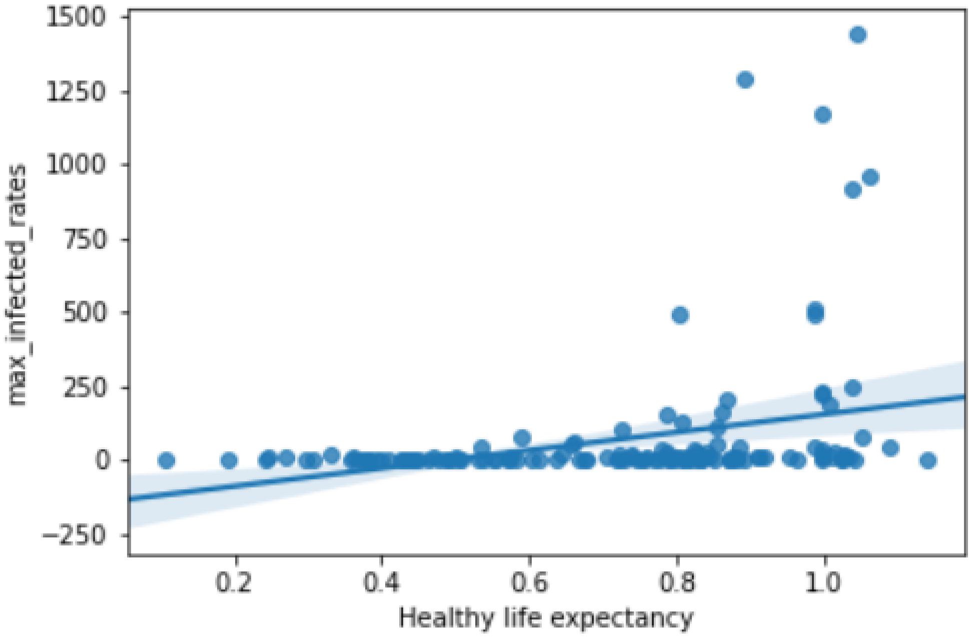




```
x = data['Healthy life expectancy']
y = data['max_infected_rates']
```



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



Result and Discussion:

From the information, we find that GDP per capita is more related to maximum infected rates. The more the GDP per capita, the more the maximum infected rates. The social support seems to have a positive correlation to maximum infected rates. From our analysis, we can conclude that there is a positive correlation these factors. On the other hand, the life expectancy does not change irrespective of the total number of cases.

Conclusion:

These data are the sample set of data we obtained and we performed the analysis. Based on the data we obtain as the time progresses, we can get a much clear picture on the factors affecting health life expectancy. Also we can check the social support and GDP per capita affecting factors.

Github Link of the Project:
https://github.com/pranavom95/Pranqv/blob/master/covid19%20data%20analysis%20notebook%20(1).ipynb