Final Project – Fall 2022
Data Analysis of Covid Data

Course: Introduction to Programming for Data Science DS5010

Instructor: Roi Yehoshua

Content:

Sai Vinay Teja Jakku

Abstract:

A novel virus is the source of COVID-19, a respiratory ailment. Fever, coughing, sore throat, and shortness of breath are symptoms. Although the virus can transmit from person to person, infection can be avoided with adequate cleanliness.

While COVID-19 might not be lethal, it spreads more quickly than other illnesses like the common cold. Every virus has a Basic Reproduction number (R0), which denotes the number of individuals that will contract the sickness from the infected person. According to preliminary research, R0 for COVID-19 is 2.7.

The current objective of all scientists worldwide is to "Flatten the Curve." The exponential growth rate that COVID-19 is currently experiencing on a global scale will be shown in the notebook that follows. Even though the number of confirmed cases is rising, flattening the curve normally suggests that the distribution of those cases should be spread out across a longer period. To put it simply, if COVID-19 were to infect 100,000 people, then those individuals should be infected in one year, not in a month. The reason to Flatten the Curve is to reduce the load on the medical facilities and to increase focus on research to find a cure/vaccine for the virus.

Introduction:

The objective of the project is to perform a variety of data analysis on the given data sets to draw meaningful insights from the data with the help of visualizations and summary. Accordingly, we answer questions which are critical to take mitigating actions to stop the spread of the virus. We also want to observe the trends of how the virus is spread in different countries and the different sub-factors effecting the spread such as population, climate, etc., that may help us in finding a cure/vaccine for the virus. The rest of the document is organized according to the table of contents mentioned in page 1.

Data Acquisition:

We have worked with three different datasets. Two of these datasets are stored in the format of json files and one data set is downloaded from the GitHub link. The original data is part of John Hopkins Institute of Medical research initiative to obtain and maintain clean data on COVID-19.

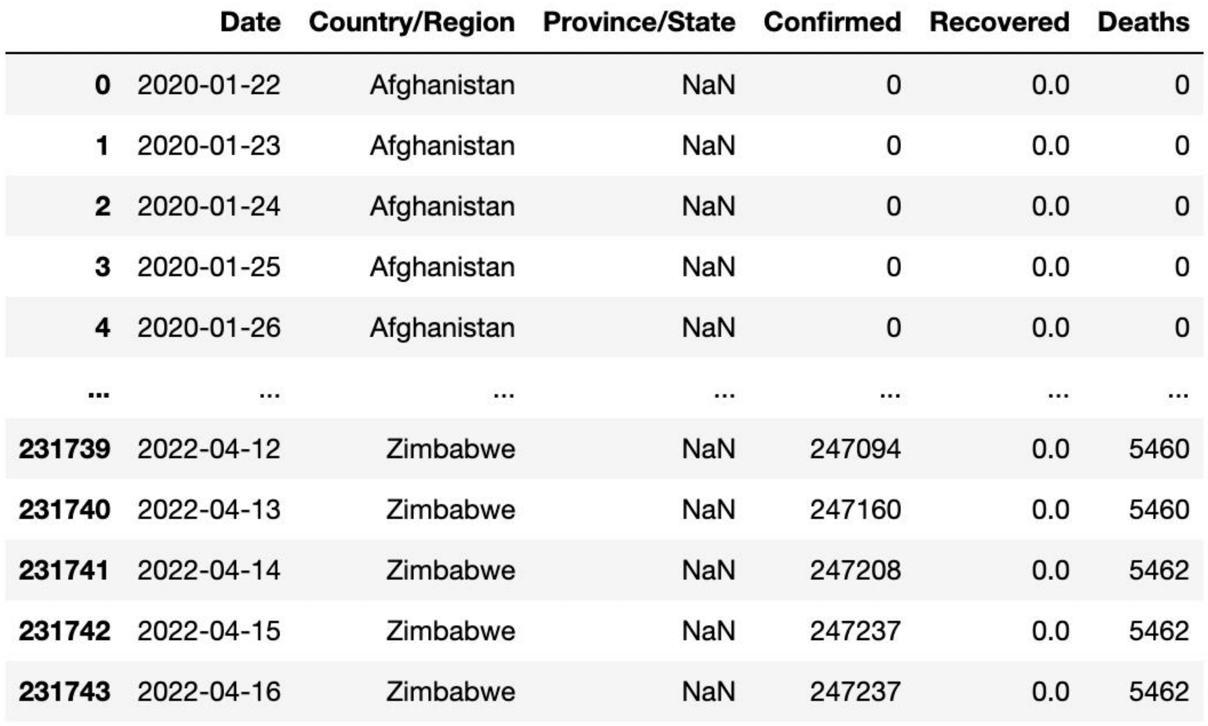
1. The first dataset downloaded from GitHub, say, Covid data has 231744 records and 6 columns. The six columns include Date, Country/Region, Province/State, number of Confirmed, Recovered, Death cases.

2. The second dataset we used for analysis is the climate dataset stored in json format. The dataset has 105 records with 4 columns which include id, City, Country, and monthly average which has 'high' temperature and 'low' temperature which will be used later on to calculate the monthly average temperatures

3. The third dataset we used is the world population dataset which is again stored in json format. This dataset has 195 records and 4 columns which are Rank, country, population, world.

Data Cleaning:

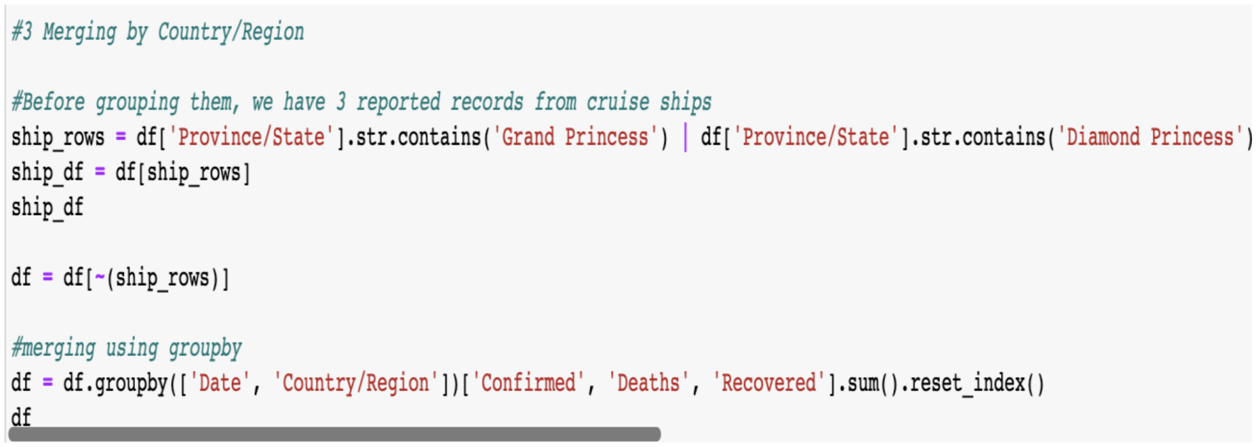
Initial look at the dataset:



	Date	Country/Region	Province/State	Confirmed	Recovered	Deaths
0	2020-01-22	Afghanistan	NaN	0	0.0	0
1	2020-01-23	Afghanistan	NaN	0	0.0	0
2	2020-01-24	Afghanistan	NaN	0	0.0	0
3	2020-01-25	Afghanistan	NaN	0	0.0	0
4	2020-01-26	Afghanistan	NaN	0	0.0	0
...	...	...	...	...	...	...
231739	2022-04-12	Zimbabwe	NaN	247094	0.0	5460
231740	2022-04-13	Zimbabwe	NaN	247160	0.0	5460
231741	2022-04-14	Zimbabwe	NaN	247208	0.0	5462
231742	2022-04-15	Zimbabwe	NaN	247237	0.0	5462
231743	2022-04-16	Zimbabwe	NaN	247237	0.0	5462

We can see a lot of NaN values. These values maybe removed using fillna() function, which will replace the NaN to whatever value we give.

We observed that some of the records has data which is from not a Country/Region, but from a ship. We identified those records and removed them to only analyze data from the countries.

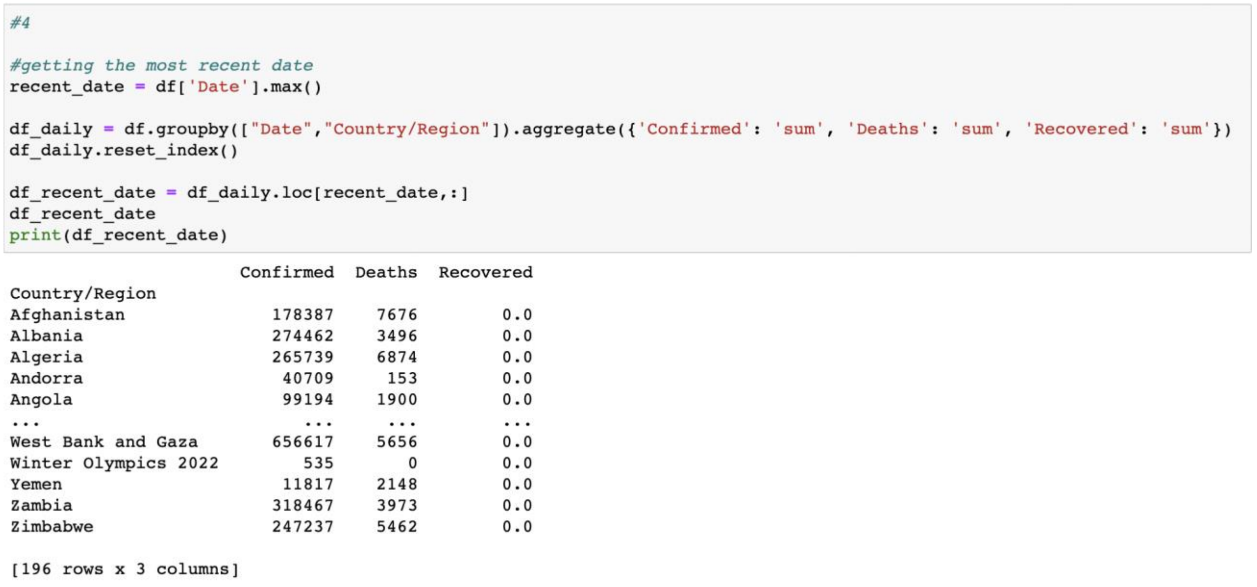


```
#3 Merging by Country/Region

#Before grouping them, we have 3 reported records from cruise ships
ship_rows = df['Province/State'].str.contains('Grand Princess') | df['Province/State'].str.contains('Diamond Princess')
ship_df = df[ship_rows]
ship_df

df = df[~(ship_rows)]

#merging using groupby
df = df.groupby(['Date', 'Country/Region'])['Confirmed', 'Deaths', 'Recovered'].sum().reset_index()
df
```

Data Cleaning:

We will then answer the questions which will help us derive insight into spread of the COVID-19 virus

1. Statistics of the data on the most recent date:

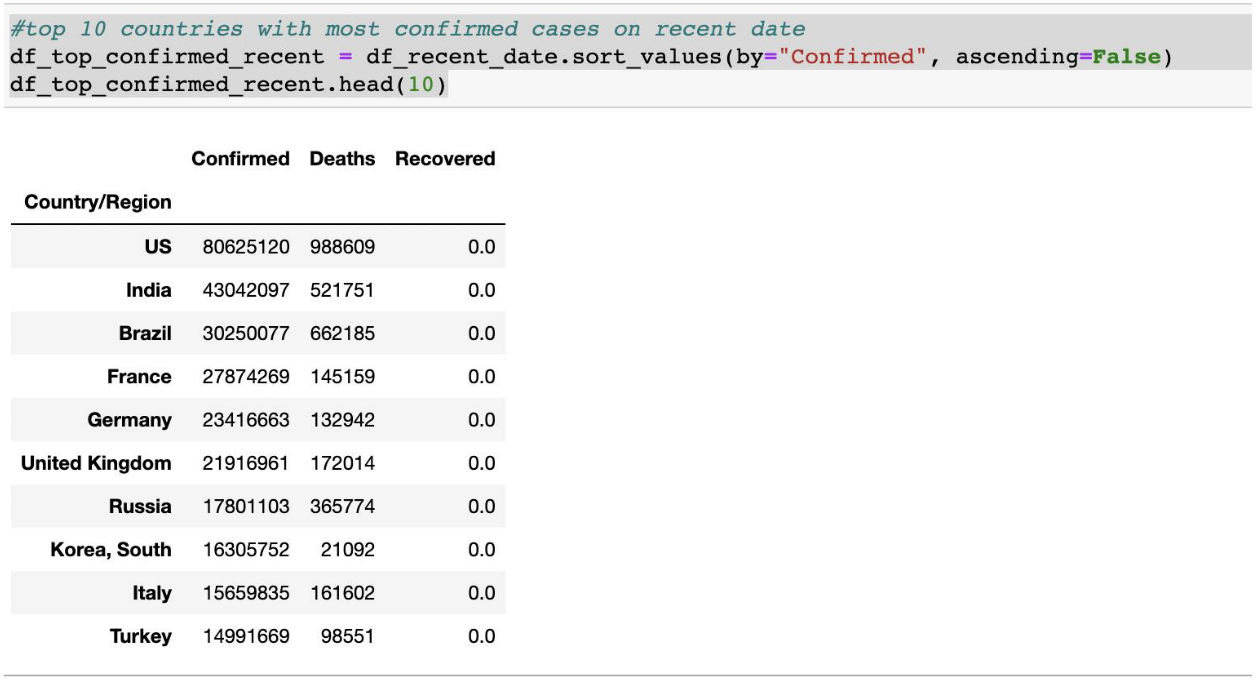


```
#4

#getting the most recent date
recent_date = df['Date'].max()

df_daily = df.groupby(["Date","Country/Region"]).aggregate({'Confirmed': 'sum', 'Deaths': 'sum', 'Recovered': 'sum'})
df_daily.reset_index()

df_recent_date = df_daily.loc[recent_date,:]
df_recent_date
print(df_recent_date)
```

```
                      Confirmed  Deaths  Recovered
Country/Region
Afghanistan              178387    7676        0.0
Albania                  274462    3496        0.0
Algeria                  265739    6874        0.0
Andorra                   40709     153        0.0
Angola                    99194    1900        0.0
...                         ...     ...        ...
West Bank and Gaza       656617    5656        0.0
Winter Olympics 2022        535       0        0.0
Yemen                     11817    2148        0.0
Zambia                   318467    3973        0.0
Zimbabwe                 247237    5462        0.0

[196 rows x 3 columns]
```

We can here see the sum of confirmed, deaths and recovered cases on that date which is the most recent date

2. Top 10 countries with most confirmed cases on recent date

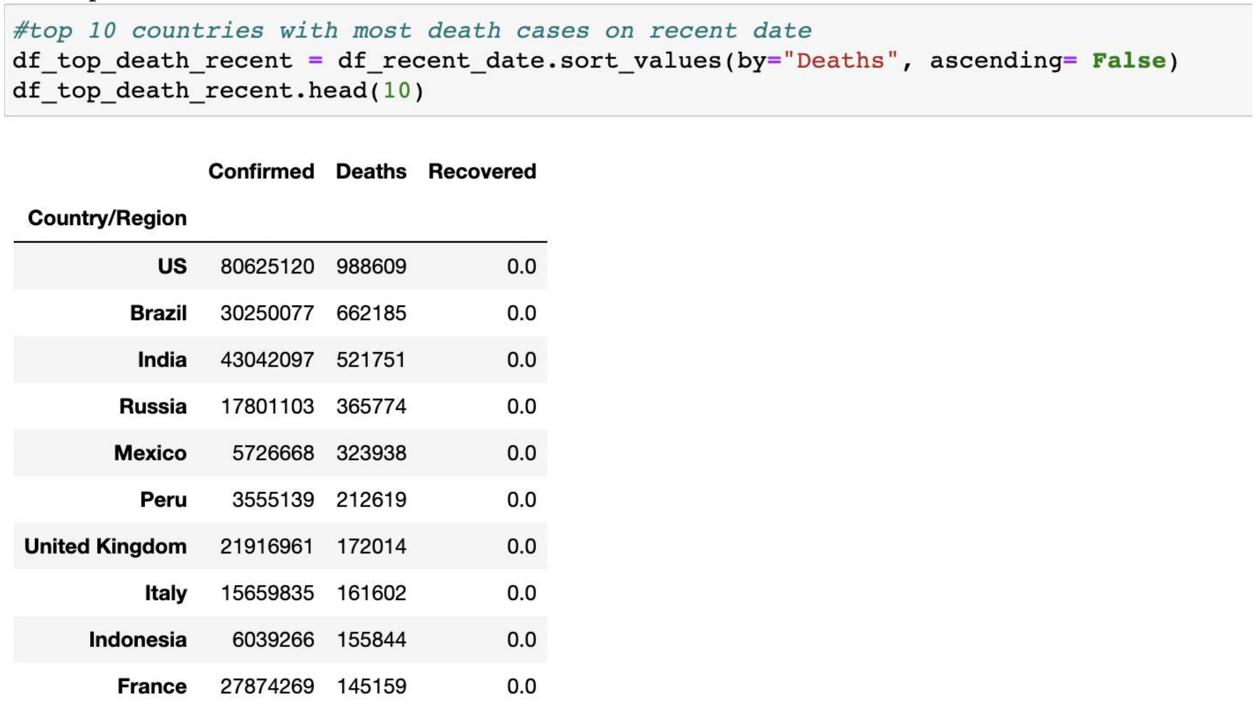


```
#top 10 countries with most confirmed cases on recent date
df_top_confirmed_recent = df_recent_date.sort_values(by="Confirmed", ascending=False)
df_top_confirmed_recent.head(10)
```

	Confirmed	Deaths	Recovered
Country/Region			
US	80625120	988609	0.0
India	43042097	521751	0.0
Brazil	30250077	662185	0.0
France	27874269	145159	0.0
Germany	23416663	132942	0.0
United Kingdom	21916961	172014	0.0
Russia	17801103	365774	0.0
Korea, South	16305752	21092	0.0
Italy	15659835	161602	0.0
Turkey	14991669	98551	0.0

We can see that USA, India, Brazil has the greatest number of confirmed cases with USA having more cases than India and Brazil combined. This suggests that USA was affected the most among all the countries

3. Top 10 countries with most death cases on the most recent date

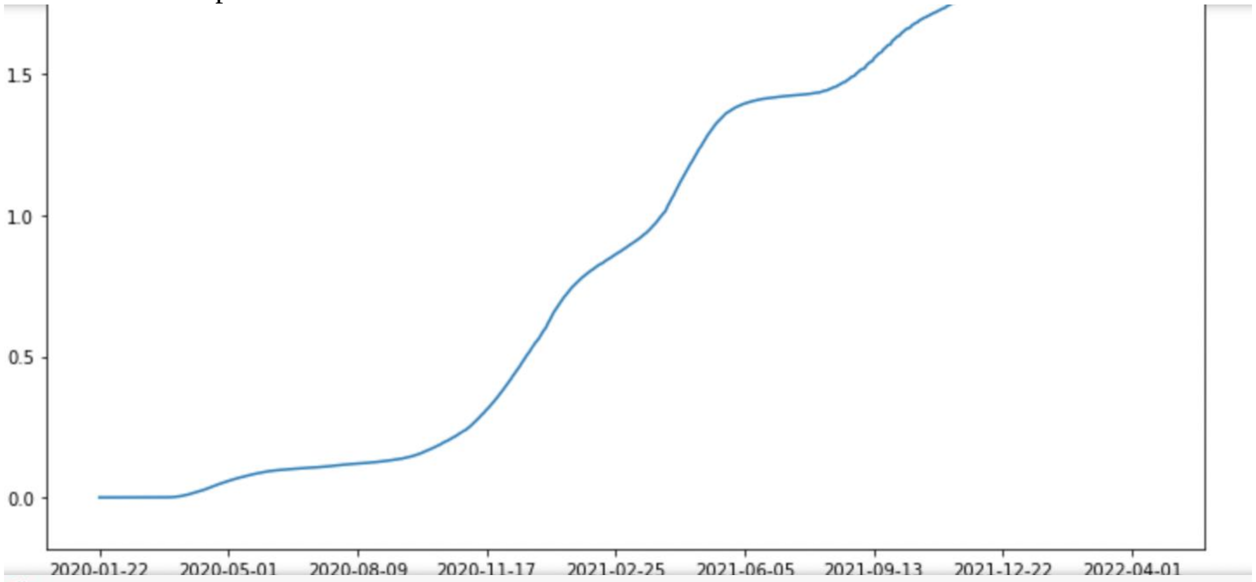


```
#top 10 countries with most death cases on recent date
df_top_death_recent = df_recent_date.sort_values(by="Deaths", ascending= False)
df_top_death_recent.head(10)
```

	Confirmed	Deaths	Recovered
Country/Region			
US	80625120	988609	0.0
Brazil	30250077	662185	0.0
India	43042097	521751	0.0
Russia	17801103	365774	0.0
Mexico	5726668	323938	0.0
Peru	3555139	212619	0.0
United Kingdom	21916961	172014	0.0
Italy	15659835	161602	0.0
Indonesia	6039266	155844	0.0
France	27874269	145159	0.0

Following the trend of Confirmed cases, USA, Brazil and India have high number of reported death cases. This suggests that high confirmed cases are leading to more death cases which implies that the fatality/mortility rate of the virus is very high.

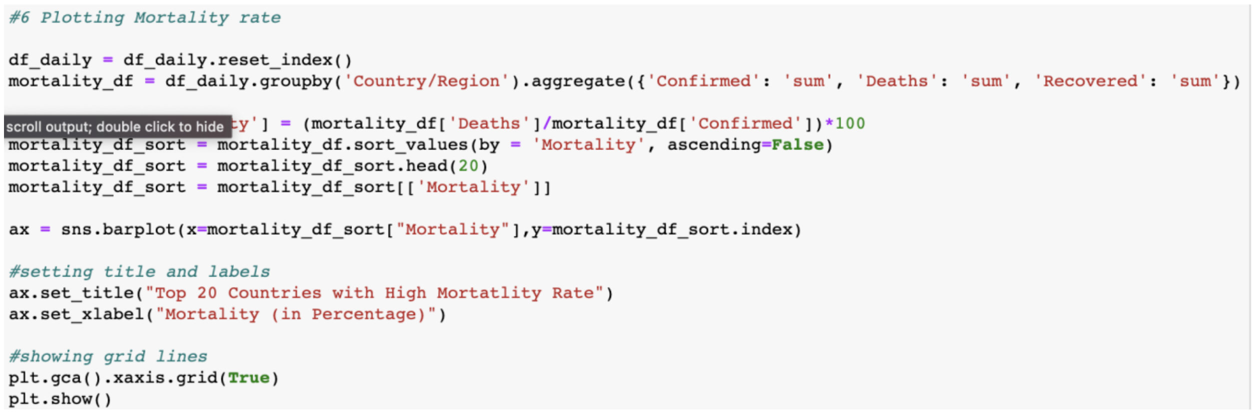
4. We then plot the rise of Covid-cases across all the countries in term of confirmed cases



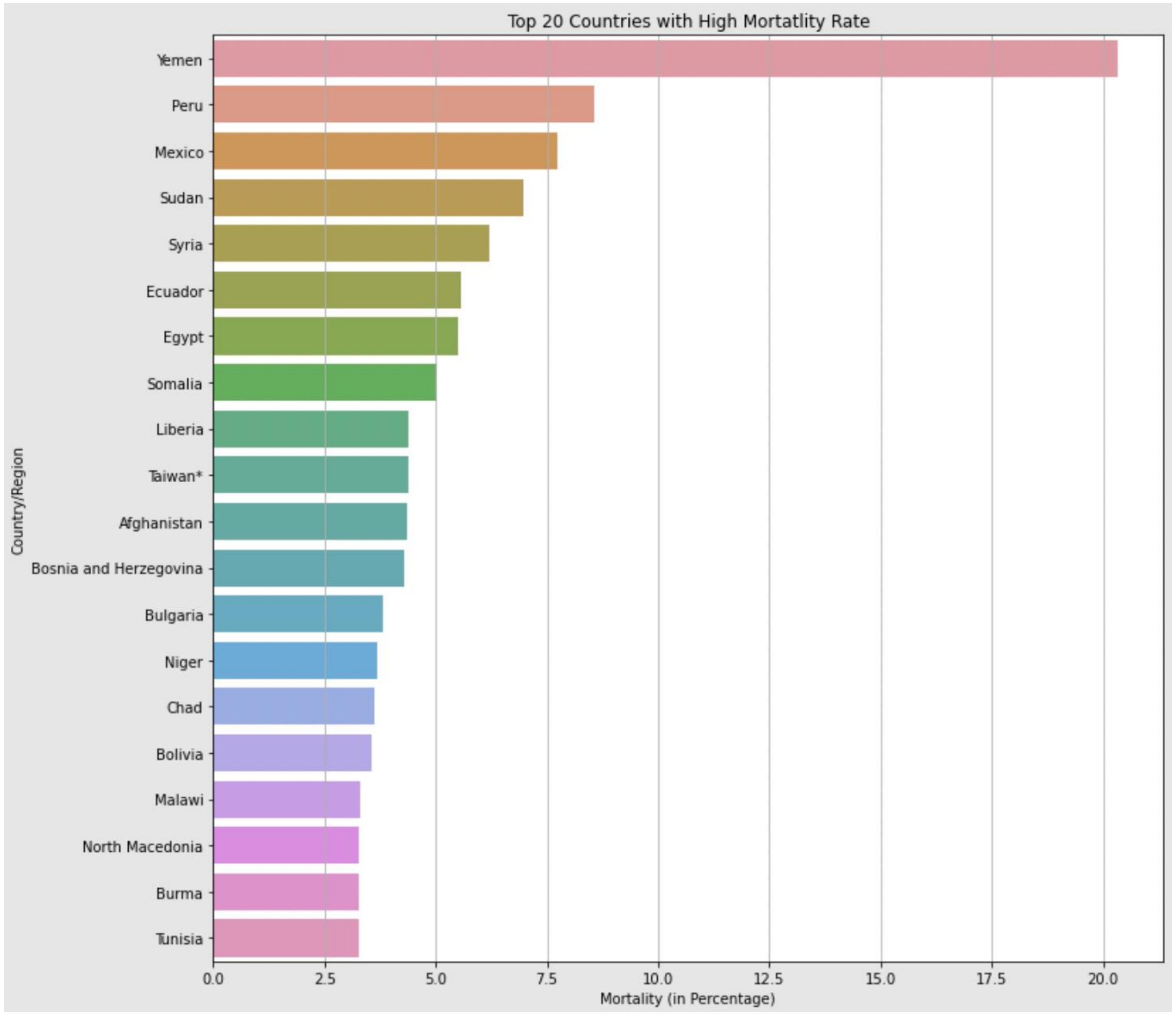
```
#5
#plotting graphs for all the countries with confirmed cases

temp_df = df.groupby(['Country/Region', 'Date'])['Confirmed'].sum()

for i in df['Country/Region'].unique():
    plt.xlabel("Date")
    plt.ylabel("Confirmed cases")
    plt.title(i)
    temp_df.loc[i].plot()
    plt.show()
```

The above graph is just a sample graph from ‘Canada’ and we can see clearly it belongs to one of the exponentially growing countries for the spread of COVID-19

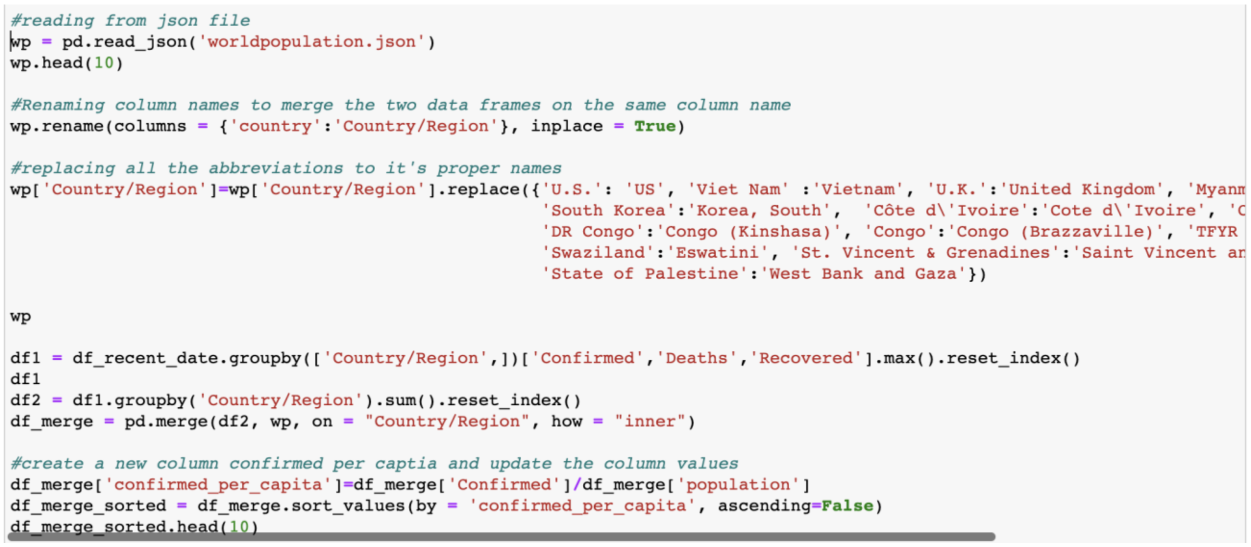
5. Plotting countries with high Mortality rate – We calculated this by using the formula total number of deaths/ total number of confirmed cases



```
#6 Plotting Mortality rate

df_daily = df_daily.reset_index()
mortality_df = df_daily.groupby('Country/Region').aggregate({'Confirmed': 'sum', 'Deaths': 'sum', 'Recovered': 'sum'})

scroll output; double click to hide ty'] = (mortality_df['Deaths']/mortality_df['Confirmed'])*100
mortality_df_sort = mortality_df.sort_values(by = 'Mortality', ascending=False)
mortality_df_sort = mortality_df_sort.head(20)
mortality_df_sort = mortality_df_sort[['Mortality']]

ax = sns.barplot(x=mortality_df_sort["Mortality"],y=mortality_df_sort.index)

#setting title and labels
ax.set_title("Top 20 Countries with High Mortatlity Rate")
ax.set_xlabel("Mortality (in Percentage)")

#showing grid lines
plt.gca().xaxis.grid(True)
plt.show()
```

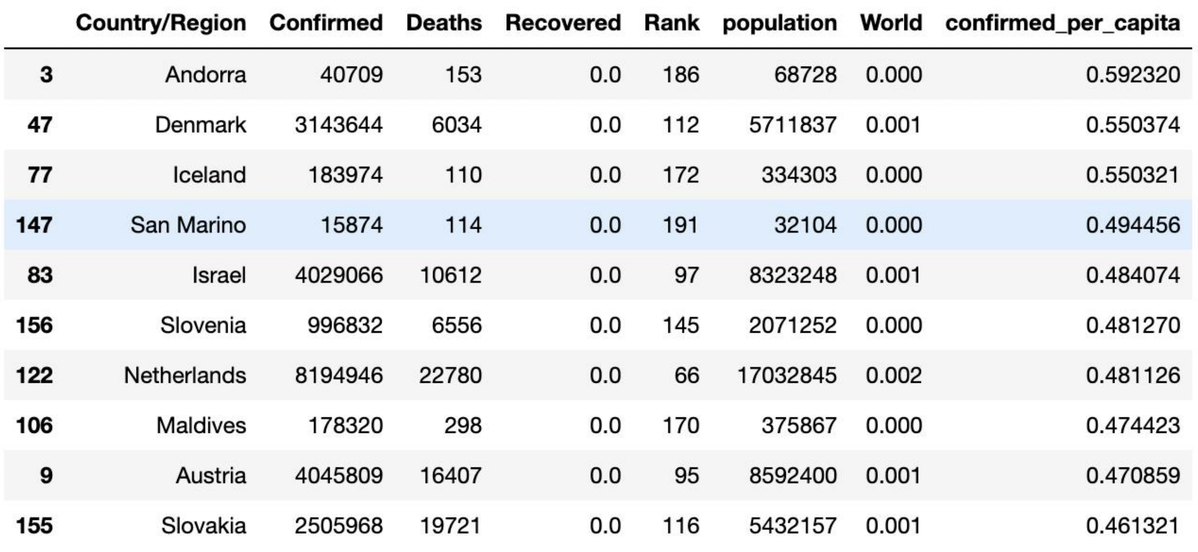


6. Merging the datasets from world population and Covid data on the common column name of ‘Country’



```
#reading from json file
wp = pd.read_json('worldpopulation.json')
wp.head(10)

#Renaming column names to merge the two data frames on the same column name
wp.rename(columns = {'country':'Country/Region'}, inplace = True)

#replacing all the abbreviations to it's proper names
wp['Country/Region']=wp['Country/Region'].replace({'U.S.': 'US', 'Viet Nam' :'Vietnam', 'U.K.':'United Kingdom', 'Myanm
                                                  'South Korea':'Korea, South',  'Côte d\'Ivoire':'Cote d\'Ivoire', 'C
                                                  'DR Congo':'Congo (Kinshasa)', 'Congo':'Congo (Brazzaville)', 'TFYR
                                                  'Swaziland':'Eswatini', 'St. Vincent & Grenadines':'Saint Vincent an
                                                  'State of Palestine':'West Bank and Gaza'})


wp

df1 = df_recent_date.groupby(['Country/Region',])['Confirmed','Deaths','Recovered'].max().reset_index()
df1
df2 = df1.groupby('Country/Region').sum().reset_index()
df_merge = pd.merge(df2, wp, on = "Country/Region", how = "inner")

#create a new column confirmed per captia and update the column values
df_merge['confirmed_per_capita']=df_merge['Confirmed']/df_merge['population']
df_merge_sorted = df_merge.sort_values(by = 'confirmed_per_capita', ascending=False)
df_merge_sorted.head(10)
```

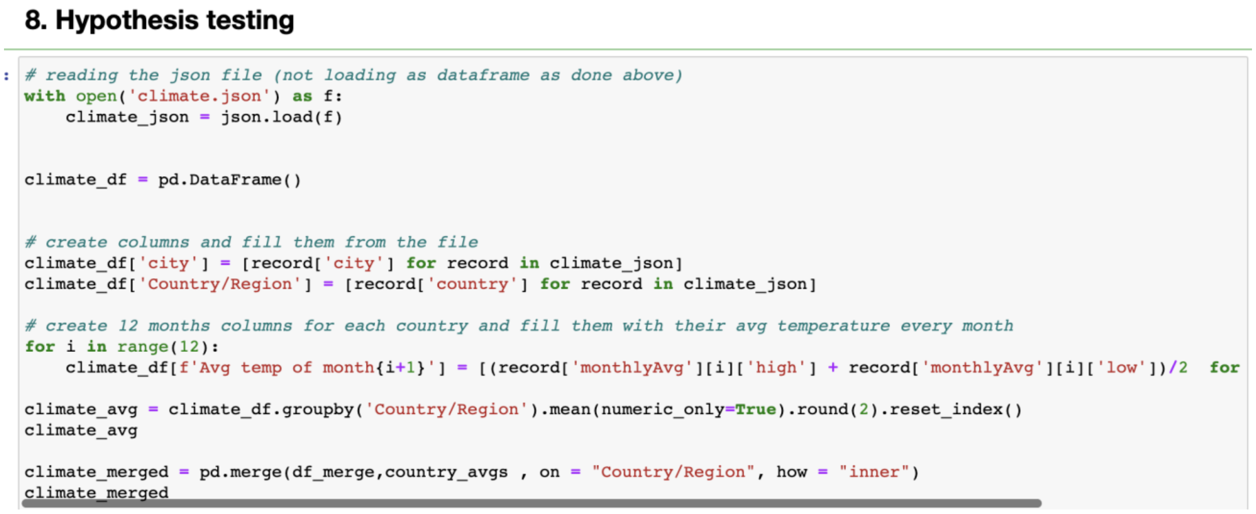
Data after merging the datasets



	Country/Region	Confirmed	Deaths	Recovered	Rank	population	World	confirmed_per_capita
3	Andorra	40709	153	0.0	186	68728	0.000	0.592320
47	Denmark	3143644	6034	0.0	112	5711837	0.001	0.550374
77	Iceland	183974	110	0.0	172	334303	0.000	0.550321
147	San Marino	15874	114	0.0	191	32104	0.000	0.494456
83	Israel	4029066	10612	0.0	97	8323248	0.001	0.484074
156	Slovenia	996832	6556	0.0	145	2071252	0.000	0.481270
122	Netherlands	8194946	22780	0.0	66	17032845	0.002	0.481126
106	Maldives	178320	298	0.0	170	375867	0.000	0.474423
9	Austria	4045809	16407	0.0	95	8592400	0.001	0.470859
155	Slovakia	2505968	19721	0.0	116	5432157	0.001	0.461321

We calculated the confirmed_per_captia_ using the formula total number of confirmed cases/world population. We then find the top countries which have high values of spread.

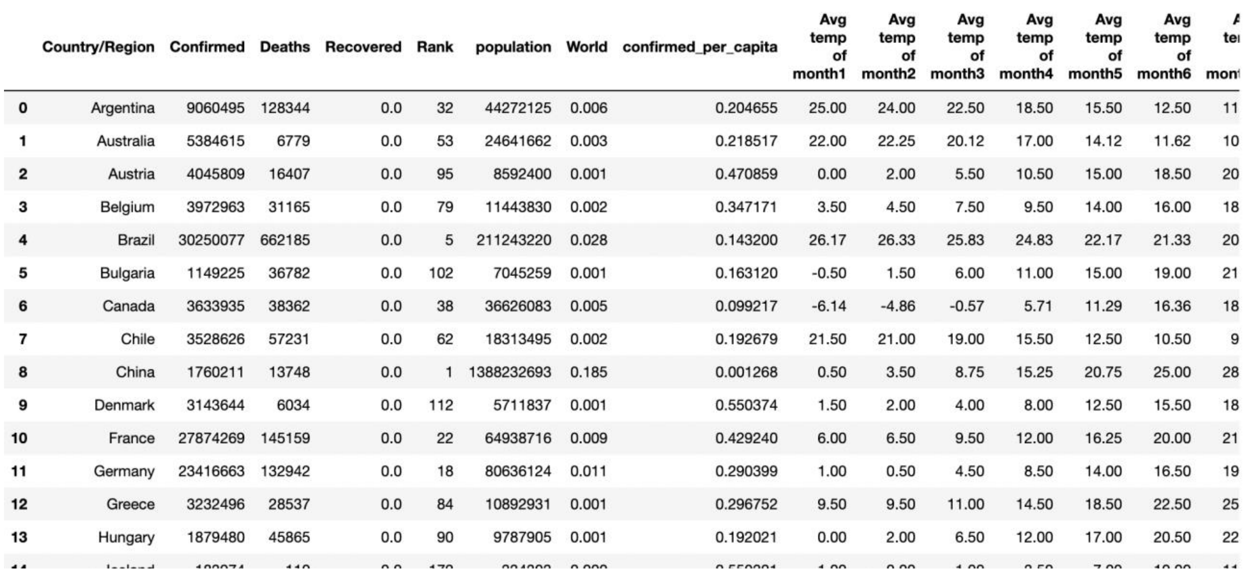
7. Testing the Hypothesis to check whether the temperature of a country effects the spread of the virus. We deduce this visually by comparing the graphs of the temperature and plot with confirmed cases over the time period.



8. Hypothesis testing

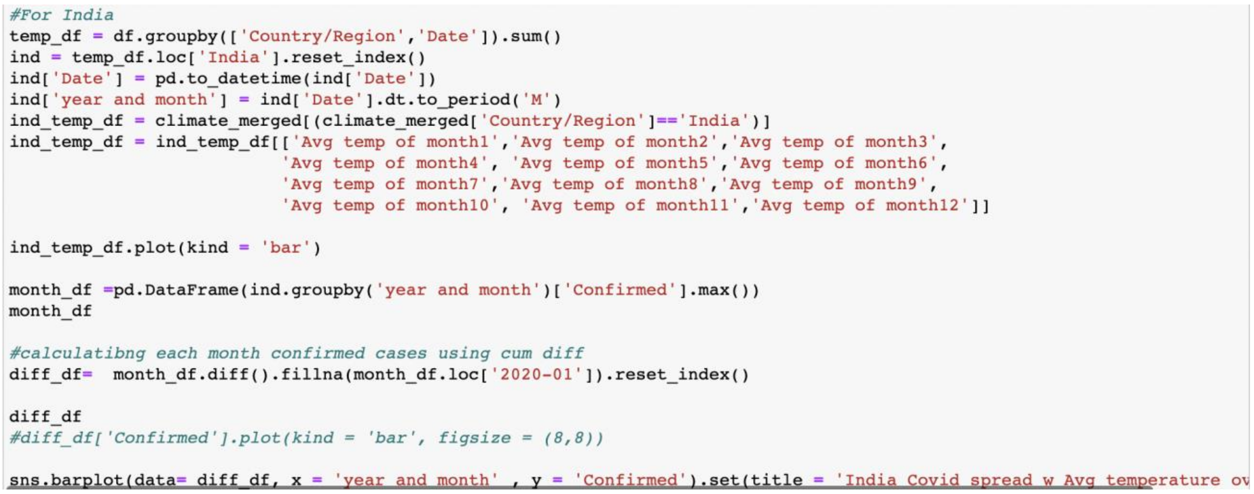
```
# reading the json file (not loading as dataframe as done above)
with open('climate.json') as f:
    climate_json = json.load(f)


climate_df = pd.DataFrame()


# create columns and fill them from the file
climate_df['city'] = [record['city'] for record in climate_json]
climate_df['Country/Region'] = [record['country'] for record in climate_json]

# create 12 months columns for each country and fill them with their avg temperature every month
for i in range(12):
    climate_df[f'Avg temp of month{i+1}'] = [(record['monthlyAvg'][i]['high'] + record['monthlyAvg'][i]['low'])/2  for

climate_avg = climate_df.groupby('Country/Region').mean(numeric_only=True).round(2).reset_index()
climate_avg

climate_merged = pd.merge(df_merge,country_avgs , on = "Country/Region", how = "inner")
climate_merged
```

We loaded the data from json file to data frame with suitable column names and calculated the avg temperature for the country using the ‘high’ and ‘low’ values and merged it with the original dataset using the common ‘Country/Region’ column name. The final dataset:



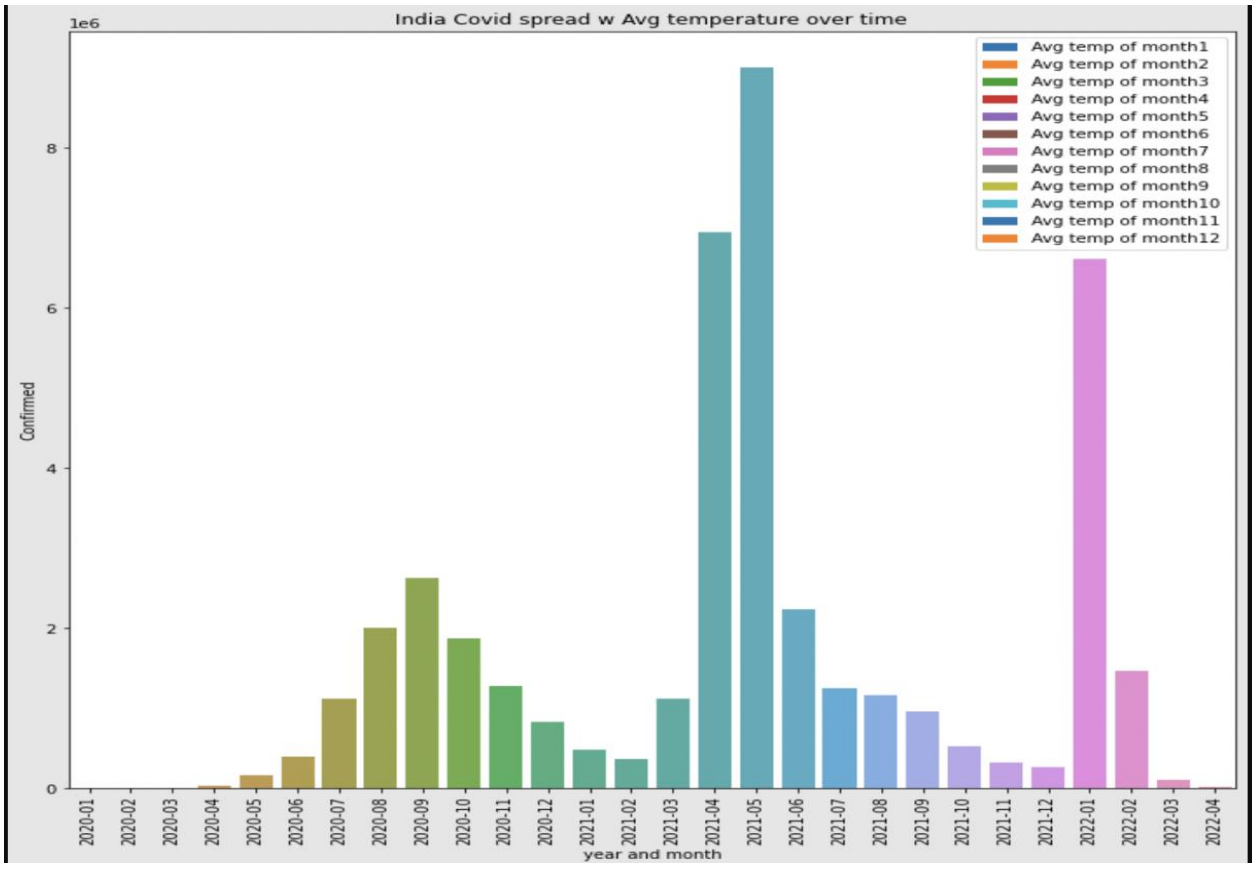
	Country/Region	Confirmed	Deaths	Recovered	Rank	population	World	confirmed_per_capita	Avg temp of month1	Avg temp of month2	Avg temp of month3	Avg temp of month4	Avg temp of month5	Avg temp of month6	A te mon
0	Argentina	9060495	128344	0.0	32	44272125	0.006	0.204655	25.00	24.00	22.50	18.50	15.50	12.50	11
1	Australia	5384615	6779	0.0	53	24641662	0.003	0.218517	22.00	22.25	20.12	17.00	14.12	11.62	10
2	Austria	4045809	16407	0.0	95	8592400	0.001	0.470859	0.00	2.00	5.50	10.50	15.00	18.50	20
3	Belgium	3972963	31165	0.0	79	11443830	0.002	0.347171	3.50	4.50	7.50	9.50	14.00	16.00	18
4	Brazil	30250077	662185	0.0	5	211243220	0.028	0.143200	26.17	26.33	25.83	24.83	22.17	21.33	20
5	Bulgaria	1149225	36782	0.0	102	7045259	0.001	0.163120	-0.50	1.50	6.00	11.00	15.00	19.00	21
6	Canada	3633935	38362	0.0	38	36626083	0.005	0.099217	-6.14	-4.86	-0.57	5.71	11.29	16.36	18
7	Chile	3528626	57231	0.0	62	18313495	0.002	0.192679	21.50	21.00	19.00	15.50	12.50	10.50	9
8	China	1760211	13748	0.0	1	1388232693	0.185	0.001268	0.50	3.50	8.75	15.25	20.75	25.00	28
9	Denmark	3143644	6034	0.0	112	5711837	0.001	0.550374	1.50	2.00	4.00	8.00	12.50	15.50	18
10	France	27874269	145159	0.0	22	64938716	0.009	0.429240	6.00	6.50	9.50	12.00	16.25	20.00	21
11	Germany	23416663	132942	0.0	18	80636124	0.011	0.290399	1.00	0.50	4.50	8.50	14.00	16.50	19
12	Greece	3232496	28537	0.0	84	10892931	0.001	0.296752	9.50	9.50	11.00	14.50	18.50	22.50	25
13	Hungary	1879480	45865	0.0	90	9787905	0.001	0.192021	0.00	2.00	6.50	12.00	17.00	20.50	22

We then pick 3 countries at random and check if the hypothesis is true

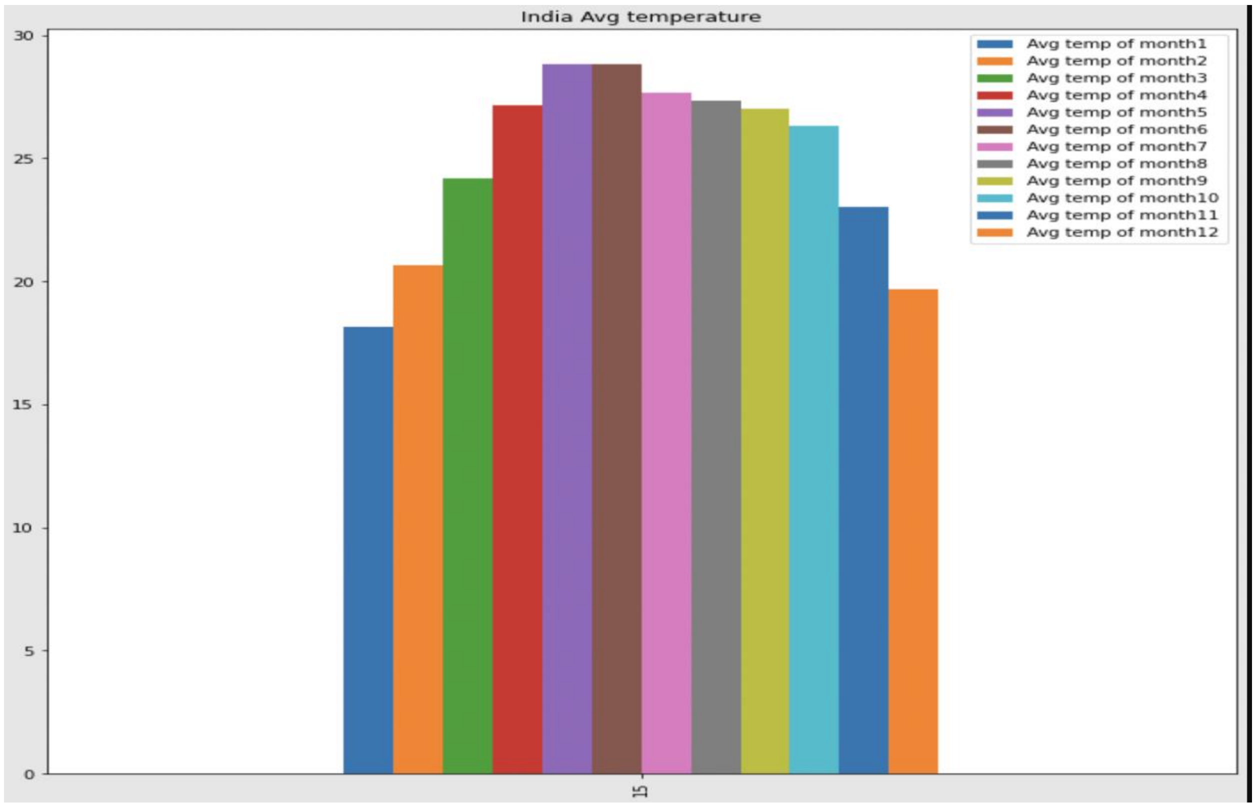


```
#For India
temp_df = df.groupby(['Country/Region','Date']).sum()
ind = temp_df.loc['India'].reset_index()
ind['Date'] = pd.to_datetime(ind['Date'])
ind['year and month'] = ind['Date'].dt.to_period('M')
ind_temp_df = climate_merged[(climate_merged['Country/Region']=='India')]
ind_temp_df = ind_temp_df[['Avg temp of month1','Avg temp of month2','Avg temp of month3',
                          'Avg temp of month4', 'Avg temp of month5','Avg temp of month6',
                          'Avg temp of month7','Avg temp of month8','Avg temp of month9',
                          'Avg temp of month10', 'Avg temp of month11','Avg temp of month12']]

ind_temp_df.plot(kind = 'bar')

month_df =pd.DataFrame(ind.groupby('year and month')['Confirmed'].max())
month_df

#calculatibng each month confirmed cases using cum diff
diff_df=  month_df.diff().fillna(month_df.loc['2020-01']).reset_index()

diff_df
#diff_df['Confirmed'].plot(kind = 'bar', figsize = (8,8))

sns.barplot(data= diff_df, x = 'year and month' , y = 'Confirmed').set(title = 'India Covid spread w Avg temperature ov
```

Checking it for India.

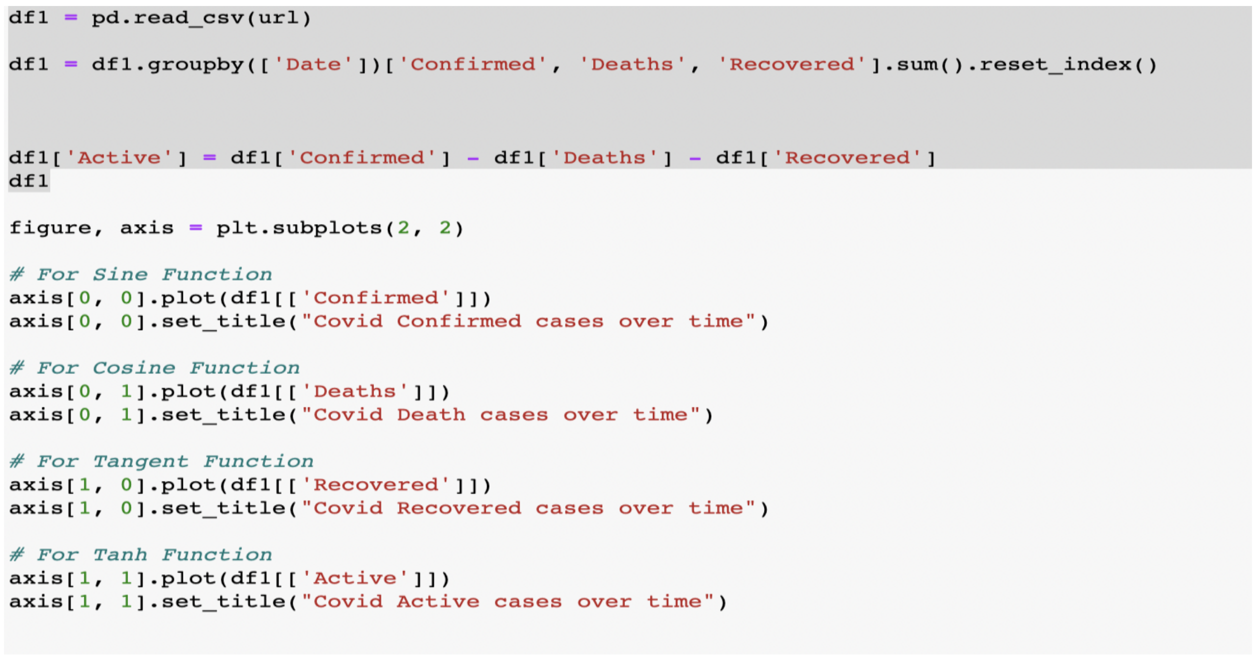


Avg temperature plot for India

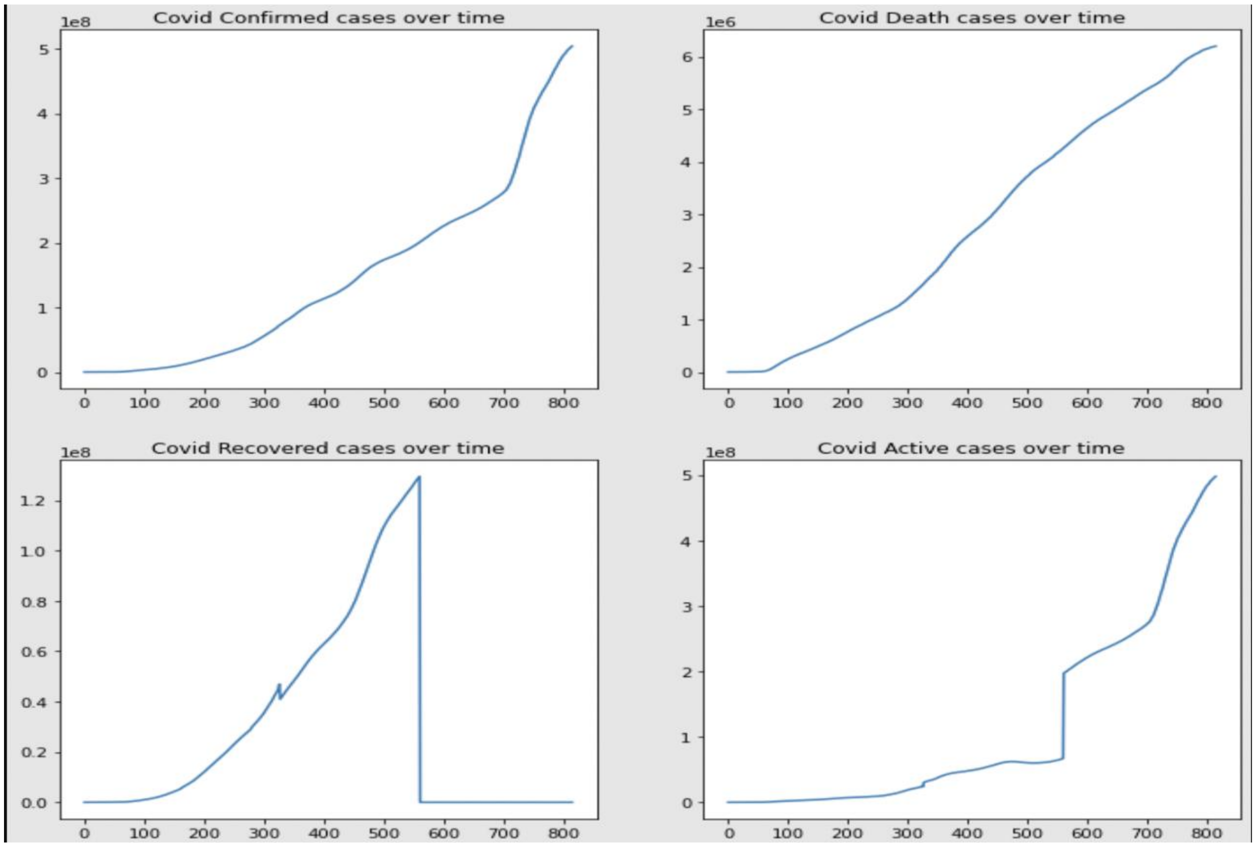


We do this again for Russia and China and we do see correlations of the confirmed cases being more when the average temperature is low. We can deduce that the hypothesis is true that a the average temperature of a country does affect the spread of the virus (in terms of confirmed cases).

We have also plotted active cases over a period for all countries to assess the spread on a global scale. We calculated the active cases by using the formula shown in the code.



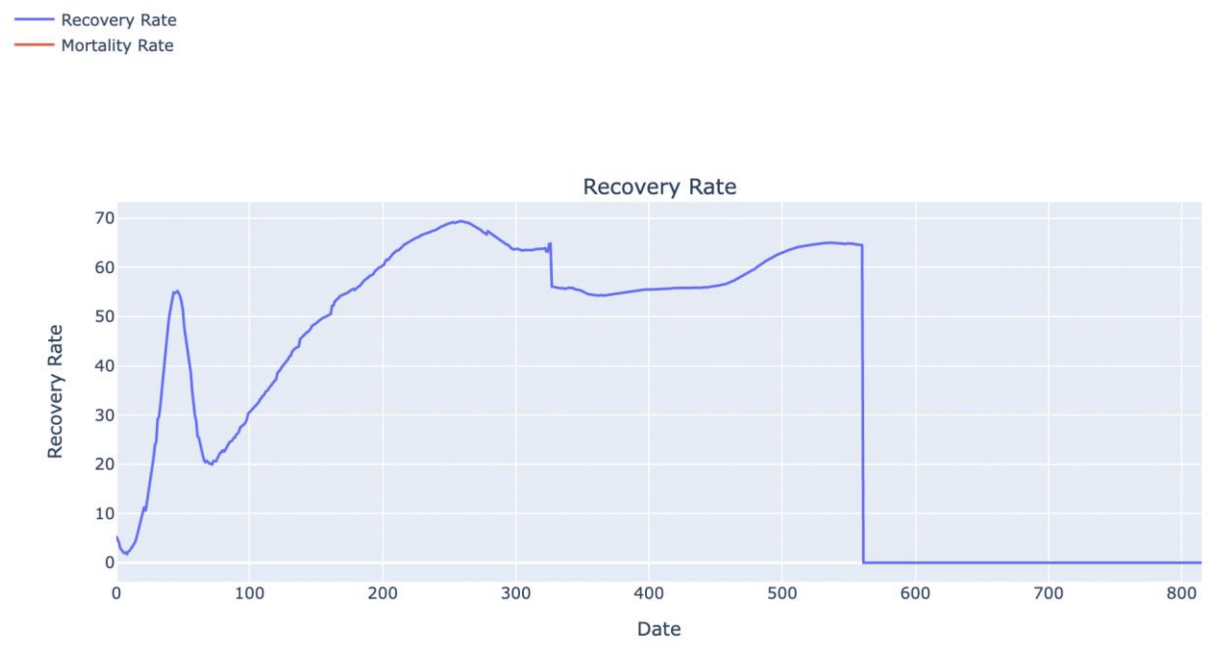
```
df1 = pd.read_csv(url)

df1 = df1.groupby(['Date'])['Confirmed', 'Deaths', 'Recovered'].sum().reset_index()


df1['Active'] = df1['Confirmed'] - df1['Deaths'] - df1['Recovered']
df1

figure, axis = plt.subplots(2, 2)

# For Sine Function
axis[0, 0].plot(df1[['Confirmed']])
axis[0, 0].set_title("Covid Confirmed cases over time")

# For Cosine Function
axis[0, 1].plot(df1[['Deaths']])
axis[0, 1].set_title("Covid Death cases over time")

# For Tangent Function
axis[1, 0].plot(df1[['Recovered']])
axis[1, 0].set_title("Covid Recovered cases over time")

# For Tanh Function
axis[1, 1].plot(df1[['Active']])
axis[1, 1].set_title("Covid Active cases over time")
```

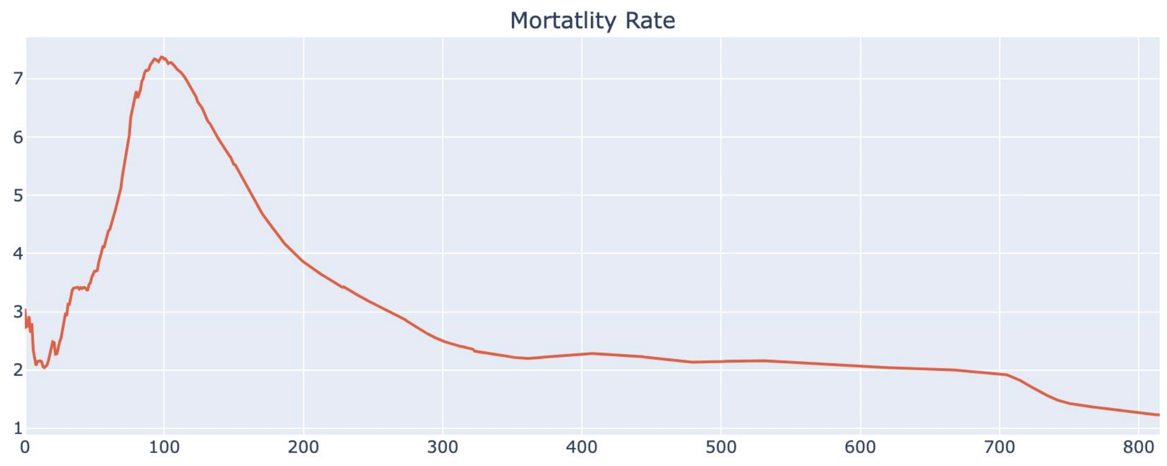


You can see the increasing trend line across all the countries in ‘Confirmed’, ‘Death’, ‘Active’ cases. The ‘Recovered cases’ have dropped to almost zero after certain time. This shows the severity of the fatality of the virus, or it could be attributed to absence of data.

We also plotted mortality and recovery rate for all the countries to assess which countries are being effected the most by the virus and the high recovering rates can tell us about our approach in mitigating the virus

```
df2 = pd.read_csv(url)

df2 = df2.groupby(['Date'])['Confirmed', 'Deaths', 'Recovered'].sum().reset_index()



df2['Active'] = df2['Confirmed'] - df2['Deaths'] - df2['Recovered']
df2

df2["Mortality Rate"]=(df2["Deaths"]/df2["Confirmed"])*100
df2["Recovery Rate"]=(df2["Recovered"]/df2["Confirmed"])*100

#Plotting Mortality and Recovery Rate
fig = make_subplots(rows=2, cols=1,
                   subplot_titles=("Recovery Rate", "Mortatlity Rate"))
fig.add_trace(
    go.Scatter(x=df2.index, y=(df2["Recovered"]/df2["Confirmed"])*100,name="Recovery Rate"),
    row=1, col=1
)
fig.add_trace(
    go.Scatter(x=df2.index, y=(df2["Deaths"]/df2["Confirmed"])*100,name="Mortality Rate"),
    row=2, col=1
)
fig.update_layout(height=1000,legend=dict(x=-0.1,y=1.2,traceorder="normal"))
fig.update_xaxes(title_text="Date", row=1, col=1)
fig.update_yaxes(title_text="Recovery Rate", row=1, col=1)
fig.update_xaxes(title_text="Date", row=1, col=2)
fig.update_yaxes(title_text="Mortality Rate", row=1, col=2)
fig.show()

#observations included in report
```





As observed in the above two plots, the recovey rate was high for certain duration and dropped which we attribute to absence of recovery cases data and the mortality rate peaked during the initial days and eventually dropped to almost zero, which can be considered as a positive outcome.

Conclusions:

We drew interesting insights about the spread of the virus in context of temperature and population. We observed high confirmed cases in densely populated countries such as India, China and high confirmed cases per capita in countries such as Denmark.

We tested the Hypothesis and concluded that the average temperature of a country does effect the spread of the virus (in term of confirmed cases). Our exploratory data analysis enables us to visualize how the COVID-19 effected all the countries in the Active cases graph. From the he 'Recovery' and 'Mortality' plot, we can conclude that we are in the recovery stage of the virus.

For further analysis, we can do more visualizations in terms of histograms and pie charts and world charts using python libraries. We can explore the time-series nature of the date to predict the spread of the for the next few days or months. This insight can especially be helpful to help prepare the countries have better medical infrastructure and facilities.

Citations and References:

1. https://raw.githubusercontent.com/datasets/covid-19/master/data/time-series-19-covid-combined.csv for the COVID Data

2. Canvas for climate.json and worldpopulation.json files

3. Google and Stack overflow