Time Series Analysis of Covid-19

February 8, 2024

0.0.1 About Dataset:

0.0.2 Context:

The data is related to Covid-19 disease. It contains the number of cases affected by covid-19, number of cases recovered, and number of covid-19 deaths worldwide.

0.0.3 Contents:

Death Data: Number of patients died in different countries.

Recovered Data: Contains Number of patients Recovered.

Reported Data: Contains the number of covid-19 cases reported

countries Data: Contains Total Number of cases in different countries.

0.0.4 Problem Statement:

Analyse and Visualize the how Covid-19 has affected countries over time.

[1]:
```
# Importing libraries

from __future__ import print_function
from ipywidgets import interact, interactive, fixed, interact_manual
from IPython.display import display, HTML

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
import folium
import plotly.graph_objects as go
import seaborn as sns
import ipywidgets as widgets
```

[2]:
```
# loading data(Source: John hopkins github repository):
death_df = pd.read_csv("C:/Users/amitm/Desktop/New folder/Task Impetus/Class/
 ↪Python/Case Study/Covid/time_series_covid19_deaths_global.csv")
```

```
confirmed_df = pd.read_csv("C:/Users/amitm/Desktop/New folder/Task Impetus/
 ↪Class/Python/Case Study/Covid/time_series_covid19_confirmed_global.csv")
recovered_df = pd.read_csv("C:/Users/amitm/Desktop/New folder/Task Impetus/
 ↪Class/Python/Case Study/Covid/time_series_covid19_recovered_global.csv")
country_df = pd.read_csv("C:/Users/amitm/Desktop/New folder/Task Impetus/Class/
 ↪Python/Case Study/Covid/cases_country.csv")
```

[3]:
```
confirmed_df.head().T
```

[3]:

	0	1	2	3	4
Province/State	NaN	NaN	NaN	NaN	NaN
Country/Region	Afghanistan	Albania	Algeria	Andorra	Angola
Lat	33.93911	41.1533	28.0339	42.5063	-11.2027
Long	67.709953	20.1683	1.6596	1.5218	17.8739
1/22/20	0	0	0	0	0
…	…	…	…	…	…
6/9/22	180864	276583	265920	43224	99761
6/10/22	180864	276638	265925	43224	99761
6/11/22	180864	276690	265925	43224	99761
6/12/22	180864	276731	265927	43224	99761
6/13/22	181120	276731	265937	43224	99761

[878 rows x 5 columns]

[4]:
```
recovered_df.head().T
```

[4]:

	0	1	2	3	4
Province/State	NaN	NaN	NaN	NaN	NaN
Country/Region	Afghanistan	Albania	Algeria	Andorra	Angola
Lat	33.93911	41.1533	28.0339	42.5063	-11.2027
Long	67.709953	20.1683	1.6596	1.5218	17.8739
1/22/20	0	0	0	0	0
…	…	…	…	…	…
6/9/22	0	0	0	0	0
6/10/22	0	0	0	0	0
6/11/22	0	0	0	0	0
6/12/22	0	0	0	0	0
6/13/22	0	0	0	0	0

[878 rows x 5 columns]

[5]:
```
death_df.head().T
```

[5]:

	0	1	2	3	4
Province/State	NaN	NaN	NaN	NaN	NaN
Country/Region	Afghanistan	Albania	Algeria	Andorra	Angola
Lat	33.93911	41.1533	28.0339	42.5063	-11.2027

```
Long              67.709953  20.1683   1.6596   1.5218  17.8739
1/22/20                   0        0        0        0        0
…                         …        …        …        …        …
6/9/22                 7709     3497     6875      153     1900
6/10/22                7709     3497     6875      153     1900
6/11/22                7709     3497     6875      153     1900
6/12/22                7709     3497     6875      153     1900
6/13/22                7710     3497     6875      153     1900

[878 rows x 5 columns]
```

[6]:
```
country_df.head()
```

[6]:
```
  Country_Region          Last_Update       Lat      Long_  Confirmed  Deaths  \
0    Afghanistan  2022-06-14 13:20:59  33.93911  67.709953     181120    7710
1        Albania  2022-06-14 13:20:59  41.15330  20.168300     276731    3497
2        Algeria  2022-06-14 13:20:59  28.03390   1.659600     265937    6875
3        Andorra  2022-06-14 13:20:59  42.50630   1.521800      43224     153
4         Angola  2022-06-14 13:20:59 -11.20270  17.873900      99761    1900

   Recovered  Active  Incident_Rate  People_Tested  People_Hospitalized  \
0        NaN     NaN     465.265139            NaN                  NaN
1        NaN     NaN    9616.060880            NaN                  NaN
2        NaN     NaN     606.455358            NaN                  NaN
3        NaN     NaN   55942.535430            NaN                  NaN
4        NaN     NaN     303.536136            NaN                  NaN

   Mortality_Rate  UID ISO3  Cases_28_Days  Deaths_28_Days
0        4.256846    4  AFG           1799              19
1        1.263682    8  ALB           1110               0
2        2.585199   12  DZA            114               0
3        0.353970   20  AND           1068               0
4        1.904552   24  AGO            474               0
```

[7]:
```
#### Data Cleaning and Transformation:
```

[8]:
```
# Renaming the column names:
country_df.columns = map(str.lower, country_df.columns)
confirmed_df.columns = map(str.lower, confirmed_df.columns)
death_df.columns = map(str.lower, death_df.columns)
recovered_df.columns = map(str.lower, recovered_df.columns)

# Changing province/state to state and country/region to country
confirmed_df = confirmed_df.rename(columns={'province/state': 'state', 'country/
 ↪region': 'country'})
recovered_df = confirmed_df.rename(columns={'province/state': 'state', 'country/
 ↪region': 'country'})
```

```
death_df = death_df.rename(columns={'province/state': 'state', 'country/region':
 ↪ 'country'})
country_df = country_df.rename(columns={'country_region': 'country'})
country_df.head()
```

[8]:

	country	last_update	lat	long_	confirmed	deaths	recovered	active	incident_rate	people_tested	people_hospitalized	mortality_rate	uid	iso3	cases_28_days	deaths_28_days
0	Afghanistan	2022-06-14 13:20:59	33.93911	67.709953	181120	7710	NaN	NaN	465.265139	NaN	NaN	4.256846	4	AFG	1799	19
1	Albania	2022-06-14 13:20:59	41.15330	20.168300	276731	3497	NaN	NaN	9616.060880	NaN	NaN	1.263682	8	ALB	1110	0
2	Algeria	2022-06-14 13:20:59	28.03390	1.659600	265937	6875	NaN	NaN	606.455358	NaN	NaN	2.585199	12	DZA	114	0
3	Andorra	2022-06-14 13:20:59	42.50630	1.521800	43224	153	NaN	NaN	55942.535430	NaN	NaN	0.353970	20	AND	1068	0
4	Angola	2022-06-14 13:20:59	-11.20270	17.873900	99761	1900	NaN	NaN	303.536136	NaN	NaN	1.904552	24	AGO	474	0

[9]:
```
country_df.shape
```

[9]:
```
(199, 16)
```

[10]:
```
country_df.isna().sum()
```

[10]:
```
country                 0
last_update             0
lat                     2
long_                   2
confirmed               0
deaths                  0
recovered             199
active                199
incident_rate           5
people_tested         199
people_hospitalized   199
mortality_rate          0
uid                     0
iso3                    4
cases_28_days           0
```

```
deaths_28_days            0
dtype: int64
```

[11]:
```
# total number of confirmed, death and recovered cases
confirmed_total = int(country_df['confirmed'].sum())
deaths_total = int(country_df['deaths'].sum())
recovered_total = int(country_df['recovered'].sum())
active_total = int(country_df['active'].sum())
```

[12]:
```
# displaying the total stats

display(HTML("<div style = 'background-color:'white'; padding: 30px '>" +
             "<span style='color: darkblue; font-size:30px;'> Confirmed: "  +␣
 ↪str(confirmed_total) +"</span>" +
             "<span style='color: red; font-size:30px;margin-left:20px;'>␣
 ↪Deaths: " + str(deaths_total) + "</span>"+
             "<span style='color: green; font-size:30px; margin-left:20px;'>␣
 ↪Recovered: " + str(recovered_total) + "</span>"+
             "</div>")
       )
```

```
<IPython.core.display.HTML object>
```

COVID-19 Confirmed/Death/Recovered cases by countries

[13]:
```
# Sorting the values by confirming descending order:
#country_df.sort_values('confirmed', ascending= False).head(10).style.
 ↪background_gradient(cmap='copper')
fig = go.FigureWidget( layout=go.Layout() )
def highlight_col(x):
    r = 'background-color: red'
    y = 'background-color: purple'
    g = 'background-color: grey'
    df1 = pd.DataFrame('', index=x.index, columns=x.columns)
    df1.iloc[:, 4] = y
    df1.iloc[:, 5] = r
    df1.iloc[:, 6] = g

    return df1

def show_latest_cases(n):
    n = int(n)
    return country_df.sort_values('confirmed', ascending= False).head(n).style.
 ↪apply(highlight_col, axis=None)

interact(show_latest_cases, n='10')

ipywLayout = widgets.Layout(border='solid 2px green')
```

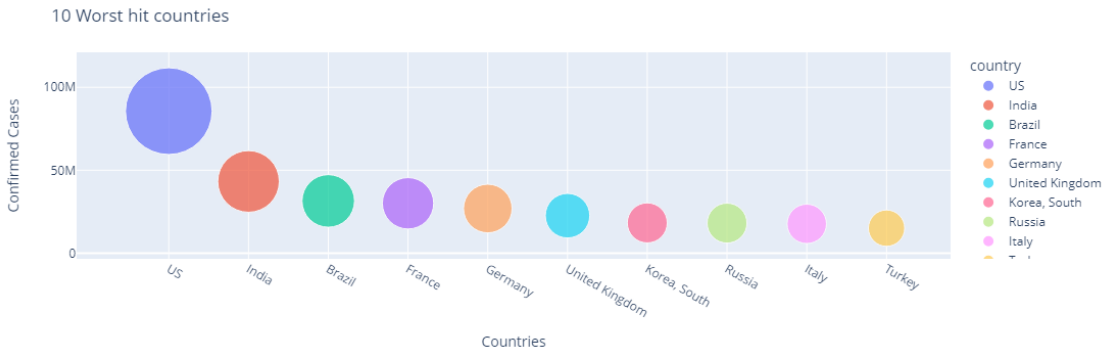
```
#ipywLayout.display='none'
widgets.VBox([fig], layout=ipywLayout)
```

```
interactive(children=(Text(value='10', description='n'), Output()),␣
 ↪_dom_classes=('widget-interact',))
```

[13]:
```
VBox(children=(FigureWidget({
    'data': [], 'layout': {'template': '…'}
}),), layout=Layout(border_bottom=…
```

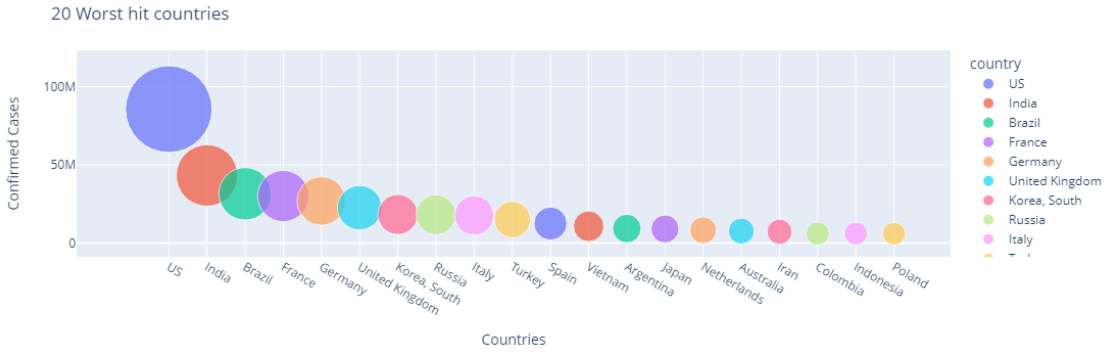
[14]:
```
sorted_country_df = country_df.sort_values('confirmed', ascending= False)
```

[15]:
```
# Plot Graph for Top 10 of worst hit countries
fig = px.scatter(sorted_country_df.head(10), x="country", y="confirmed",␣
 ↪size="confirmed", color="country",
                hover_name="country", size_max=60)
fig.update_layout(
    title=str(10) +" Worst hit countries",
    xaxis_title="Countries",
    yaxis_title="Confirmed Cases",
    width = 700,
    )
fig.show()
```



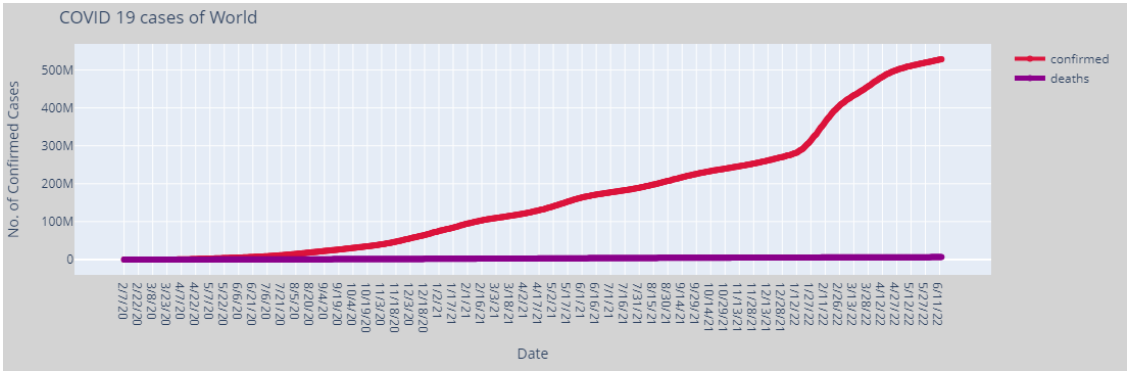
[16]:
```
# Plotting top 20 worst hit countries:
fig = px.scatter(sorted_country_df.head(20), x="country", y="confirmed",␣
 ↪size="confirmed", color="country",
                hover_name="country", size_max=60)
fig.update_layout(
    title=str(20) +" Worst hit countries",
    xaxis_title="Countries",
    yaxis_title="Confirmed Cases",
    width = 700,
    )
```

```
fig.show()
```

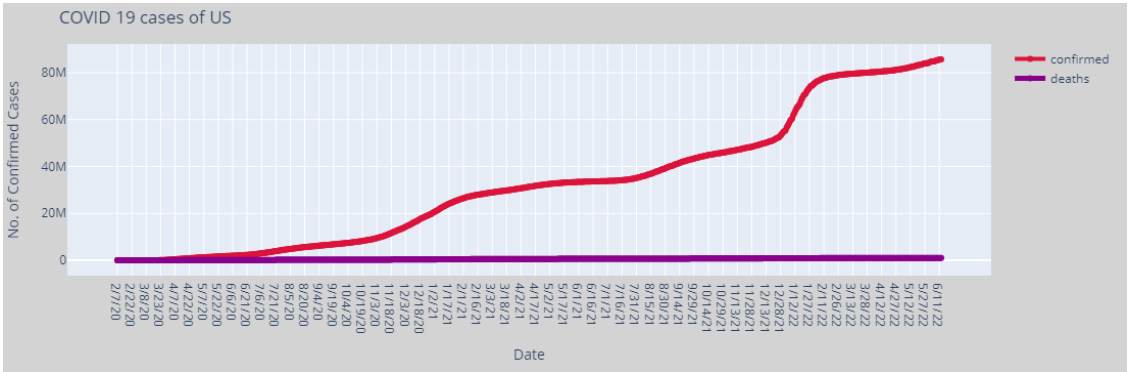


[17]:
```
# Plotting for total number of cases across globe:

def plot_cases_of_a_country(country):
    labels = ['confirmed', 'deaths']
    colors = ['crimson', 'darkmagenta']
    mode_size = [6, 8]
    line_size = [4, 5]

    df_list = [confirmed_df, death_df]

    fig = go.Figure();

    for i, df in enumerate(df_list):
        if country == 'World' or country == 'world':
            x_data = np.array(list(df.iloc[:, 20:].columns))
            y_data = np.sum(np.asarray(df.iloc[:,4:]),axis = 0)

        else:
            x_data = np.array(list(df.iloc[:, 20:].columns))
            y_data = np.sum(np.asarray(df[df['country'] == country].iloc[:,20:
 ↪]),axis = 0)

        fig.add_trace(go.Scatter(x=x_data, y=y_data, mode='lines+markers',
        name=labels[i],
        line=dict(color=colors[i], width=line_size[i]),
        connectgaps=True,
        text = "Total " + str(labels[i]) +": "+ str(y_data[-1])
        ));

    fig.update_layout(
        title="COVID 19 cases of " + country,
```

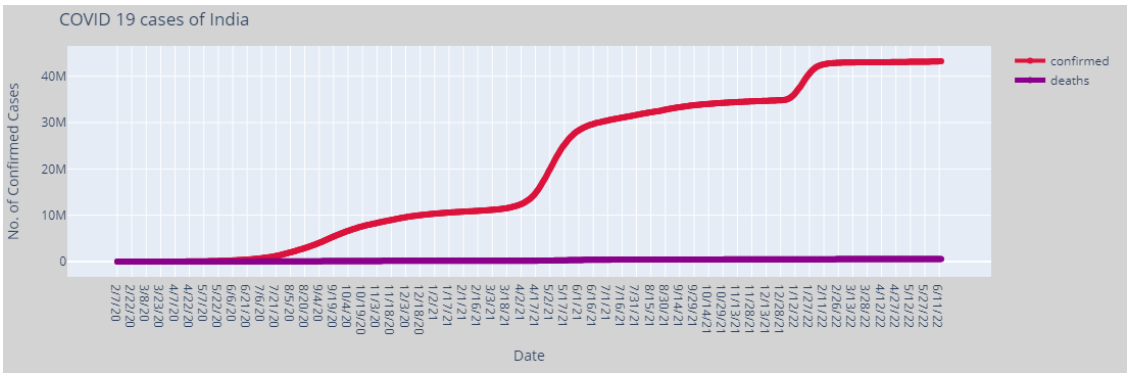
```
        xaxis_title='Date',
        yaxis_title='No. of Confirmed Cases',
        margin=dict(l=20, r=20, t=40, b=20),
        paper_bgcolor="lightgrey",
        width = 800,

    );

    fig.update_yaxes(type="linear")
    fig.show();
plot_cases_of_a_country('World')
```



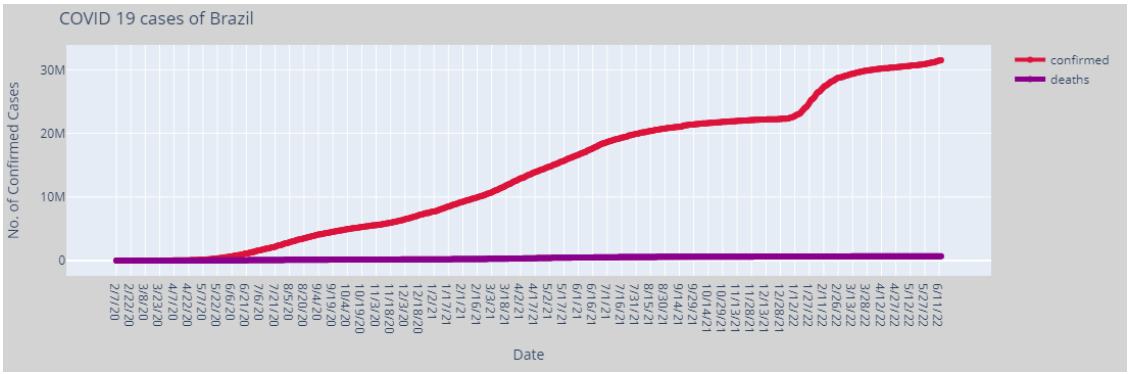
[18]:
```
#Plotting for Country US:
plot_cases_of_a_country('US')
```



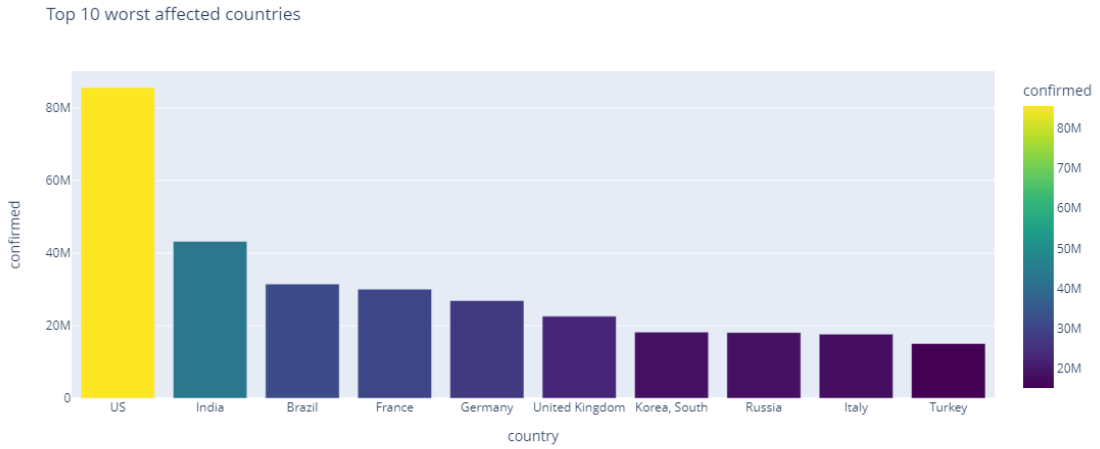
[19]:
```
# Plotting for cases in INDIA:
plot_cases_of_a_country('India')
```



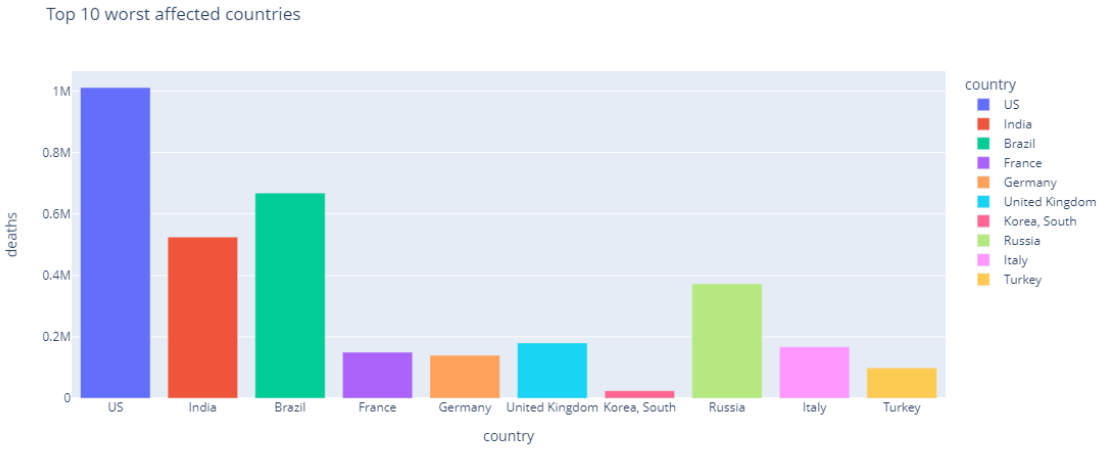
[20]:
```
# Plotting for Country Brazil:
plot_cases_of_a_country('Brazil')
```



[21]:
```
# 10 Worst hit countries - Confirmed cases:
fig = px.bar(
    sorted_country_df.head(10),
    x="country",
    y="confirmed",
    title="Top 10 worst affected countries",
    color="confirmed",
    color_continuous_scale='Viridis',
    height=500,
    width=800
)

fig.show()
```



[22]:
```
# 10 worst hit countries - Death cases
px.bar(
    sorted_country_df.head(10),
    x = "country",
    y = "deaths",
    title= "Top 10 worst affected countries",
    color="country",
    color_continuous_scale='Cividis',
    height=500,
    width=800
)
```



[23]:
```
len(sorted_country_df)
```

[23]:
```
199
```

[24]:
```
sorted_country_df.isnull().sum()
```

[24]:
```
country                 0
last_update             0
lat                     2
long_                   2
confirmed               0
deaths                  0
recovered             199
active                199
incident_rate           5
people_tested         199
people_hospitalized   199
mortality_rate          0
uid                     0
iso3                    4
cases_28_days           0
deaths_28_days          0
dtype: int64
```

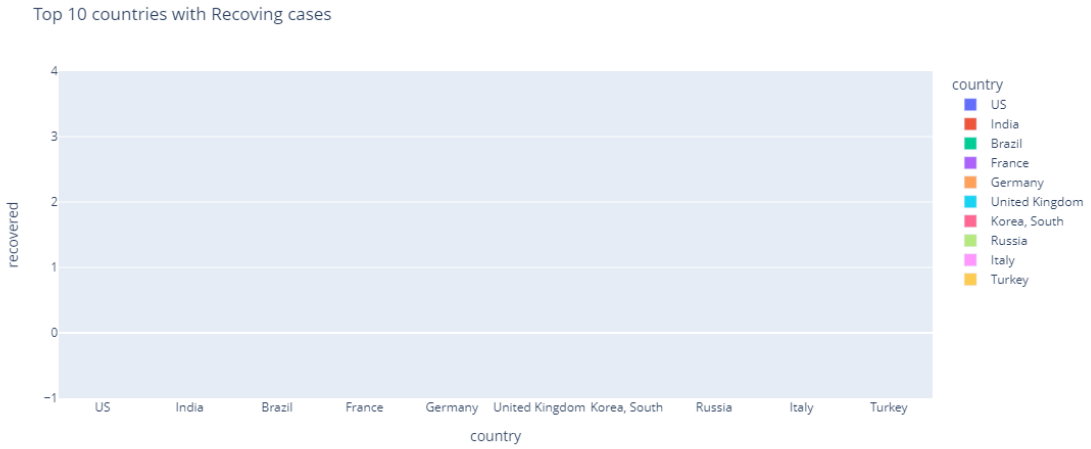
[25]:
```
sorted_country_df.head(10)
```

[25]:

	country	last_update	lat	long_	confirmed	deaths	recovered	active	incident_rate	people_tested
184	US	2022-06-14 13:20:59	40.000000	-100.000000	85633278	1011545	NaN	NaN	25991.514889	NaN
80	India	2022-06-14 13:20:59	20.593684	78.962880	43236695	524777	NaN	NaN	3133.083885	NaN
24	Brazil	2022-06-14 13:20:59	-14.235000	-51.925300	31497038	668180	NaN	NaN	14817.992837	NaN
63	France	2022-06-14 13:20:59	46.227600	2.213700	30057668	149880	NaN	NaN	46065.502410	NaN
67	Germany	2022-06-14 13:20:59	51.165691	10.451526	26915085	139914	NaN	NaN	32367.356101	NaN
188	United Kingdom	2022-06-14 13:20:59	55.000000	-3.000000	22600145	179941	NaN	NaN	33291.317309	NaN
94	Korea, South	2022-06-14 13:20:59	35.907757	127.766922	18239056	24390	NaN	NaN	35575.086110	NaN
146	Russia	2022-06-14 13:20:59	61.524000	105.318800	18113989	372452	NaN	NaN	12412.413764	NaN
86	Italy	2022-06-14 13:20:59	41.871900	12.567400	17664043					
183	Turkey	2022-06-14 13:20:59	38.963700	35.243300	15072747					

```
86    167432        NaN     NaN   29215.198389            NaN
183    98965        NaN     NaN   17871.607472            NaN

     people_hospitalized  mortality_rate  uid iso3  cases_28_days  \
184                  NaN        1.181252  840  USA        2955626
80                   NaN        1.213731  356  IND         111325
24                   NaN        2.121406   76  BRA         795138
63                   NaN        0.498641  250  FRA         565562
67                   NaN        0.519835  276  DEU        1096680
188                  NaN        0.796194  826  GBR         214942
94                   NaN        0.133724  410  KOR         408627
146                  NaN        2.056157  643  RUS         112802
86                   NaN        0.947869  380  ITA         592394
183                  NaN        0.656582  792  TUR          17171

     deaths_28_days
184            9411
80              517
24             2964
63             1272
67             2200
188            1878
94              619
146            2247
86             2086
183              61
```

[26]:
```
# Worst hit countries - Recovering cases
px.bar(
    sorted_country_df.head(10),
    x = "country",
    y = "recovered",
    title= "Top 10 countries with Recoving cases",
    color="country",
    color_continuous_scale='Cividis',
    height=500,
    width=800
)
```



- The data doesnot contain any number of recovery cases as per given timeline

Global spread of COVID-19

[27]:
```
# Remove rows with missing latitude or longitude values:
confirmed_df = confirmed_df.dropna(subset=['lat', 'long'])

world_map = folium.Map(location=[11, 0], tiles="cartodbpositron", zoom_start=2,␣
 ↪max_zoom=6, min_zoom=2)

for i in range(len(confirmed_df)):
    folium.Circle(
        location=[confirmed_df.iloc[i]['lat'], confirmed_df.iloc[i]['long']],
        fill=True,
        radius=(int((np.log(confirmed_df.iloc[i,-1] + 1.00001))) + 0.2) * 10000,
        color='teal',
        fill_color='olive',  # Set fill color to crimson
        tooltip="<div style='margin: 0; background-color: black; color: white;
 ↪'>" +
                "<h4 style='text-align:center;font-weight: bold'>" +␣
 ↪confirmed_df.iloc[i]['country'] + "</h4>"
                "<hr style='margin:10px;color: white;'>" +
                "<ul style='color: white;;list-style-type:circle;align-item:
 ↪left;padding-left:20px;padding-right:20px'>" +
                "<li>Confirmed: " + str(confirmed_df.iloc[i, -1]) + "</li>" +
                "<li>Deaths:   " + str(death_df.iloc[i, -1]) + "</li>" +
                "<li>Death Rate: " + str(np.round(death_df.iloc[i, -1] /␣
 ↪(confirmed_df.iloc[i, -1] + 1.00001) * 100,
                                                  2)) + "</li>" +
                "</ul></div>",
```

```
    ).add_to(world_map)

world_map
```

[27]:
```
<folium.folium.Map at 0x2ac98f83ed0>
```

COVID-19: Progression of spread

[28]:
```
import numpy as np;np.random.seed(sum(map(ord, 'calmap')))
from matplotlib import ticker
import pycountry as pc
import folium
import branca
from datetime import datetime, timedelta,date
from scipy.interpolate import make_interp_spline, BSpline
import plotly.express as px
import holoviews as hv
import calmap
import json, requests


import warnings
warnings.filterwarnings('ignore')

%matplotlib inline
```

[29]:
```
data = pd.read_csv("C:/Users/amitm/Desktop/New folder/Task Impetus/Class/Python/
  ↪Case Study/Covid/cases_country.csv")
print(data.columns)
```

```
Index(['Country_Region', 'Last_Update', 'Lat', 'Long_', 'Confirmed', 'Deaths',
       'Recovered', 'Active', 'Incident_Rate', 'People_Tested',
       'People_Hospitalized', 'Mortality_Rate', 'UID', 'ISO3', 'Cases_28_Days',
       'Deaths_28_Days'],
      dtype='object')
```

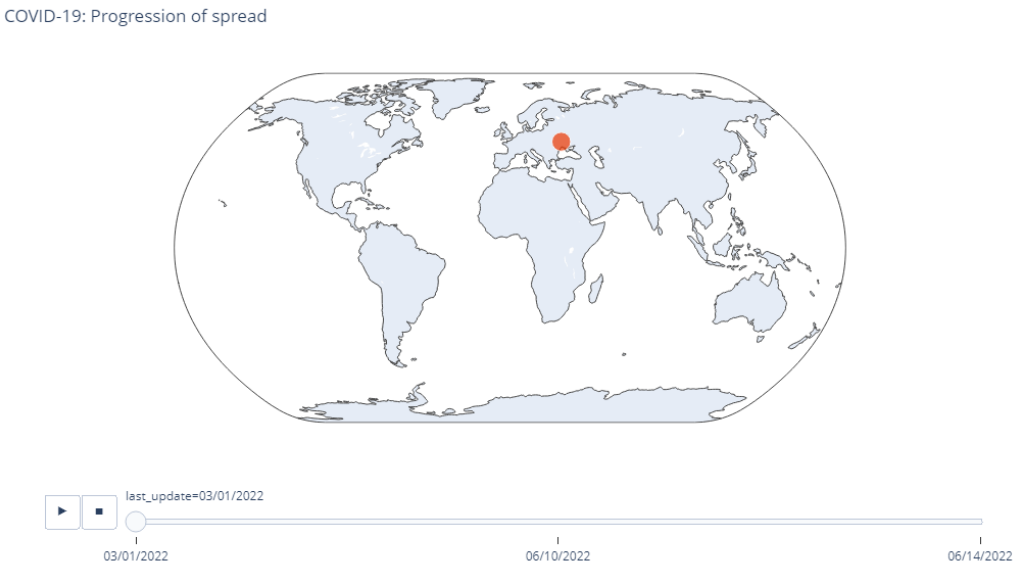
[30]:
```
data.rename(columns={'Last_Update':'last_update', 'Country_Region':'Country'},␣
  ↪inplace=True)

# Check column names after renaming
print(data.columns)
```

```
Index(['Country', 'last_update', 'Lat', 'Long_', 'Confirmed', 'Deaths',
       'Recovered', 'Active', 'Incident_Rate', 'People_Tested',
       'People_Hospitalized', 'Mortality_Rate', 'UID', 'ISO3', 'Cases_28_Days',
       'Deaths_28_Days'],
      dtype='object')
```

[31]:
```
d1 = data.groupby(['last_update', 'Country'])[['Confirmed', 'Deaths']].max().
  ↪reset_index().fillna(0)
d1["last_update"] = pd.to_datetime( d1["last_update"]).dt.strftime('%m/%d/%Y')
print(d1.head())
```

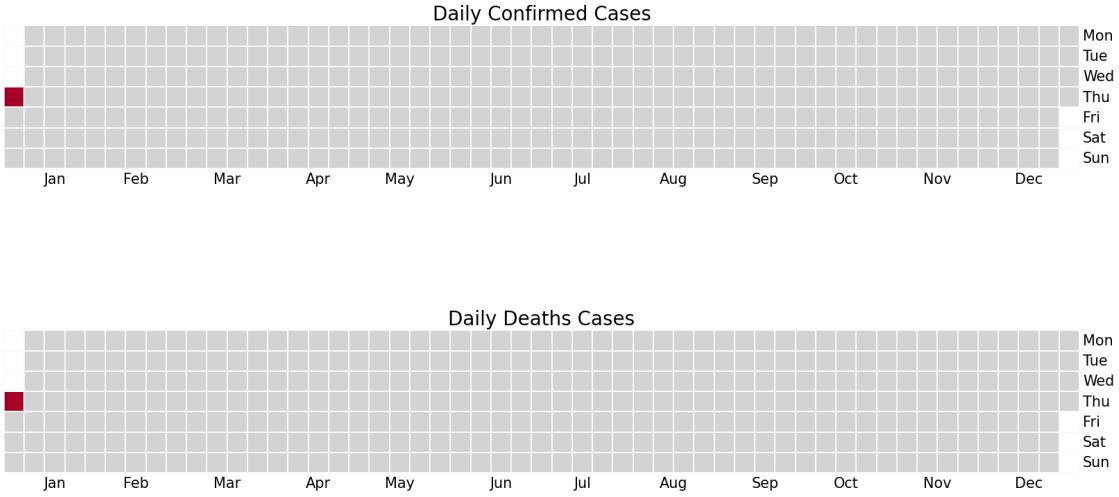
```
  last_update      Country  Confirmed  Deaths
0  03/01/2022      Ukraine    5040518  112459
1  06/10/2022    Indonesia    6056017  156604
2  06/14/2022  Afghanistan     181120    7710
3  06/14/2022      Albania     276731    3497
4  06/14/2022      Algeria     265937    6875
```

[32]:
```
fig = px.scatter_geo(d1, locations="Country", locationmode='country names',
                     color=np.power(d1["Confirmed"],0.3)-2 , size= np.
  ↪power(d1["Confirmed"]+1,0.3)-1, hover_name="Country",
                     hover_data=["Confirmed"],
                     range_color= [0, max(np.power(d1["Confirmed"],0.3))],
                     projection="natural earth", animation_frame="last_update",
                     color_continuous_scale=px.colors.sequential.Plasma,
                     title='COVID-19: Progression of spread'
                    )
fig.update_layout( height=600, width=1000)
fig.update_coloraxes(colorscale="hot")
fig.update(layout_coloraxis_showscale=False)
fig.show()
```



Plotting worldwide scenario on Calender maps

[33]:
```
# Plot first subplot:
data.index = pd.to_datetime(data.index)
f = plt.figure(figsize=(20,10))
f.add_subplot(2,1,1)

calmap.yearplot(data['Confirmed'], fillcolor='lightgrey', cmap='RdYlGn',␣
 ↪linewidth=0.3, linecolor="#fafafa",␣
 ↪monthlabels=["J","F","M","A","M","J","J","A","S","O","N","D"])
plt.title("Daily Confirmed Cases", fontsize=20)
plt.tick_params(labelsize=15)
plt.show()

# Plot second subplot:
f = plt.figure(figsize=(20,10))
calmap.yearplot(data['Deaths'], fillcolor='lightgrey', cmap='RdYlGn',␣
 ↪linewidth=0.3, linecolor="#fafafa",)
plt.title("Daily Deaths Cases", fontsize=20)
plt.tick_params(labelsize=15)

plt.show()
```



Conclusion: People may be sick with the virus for 1 to 14 days before developing symptoms. The most common symptoms of coronavirus disease (COVID-19) are: * fever * tiredness * cough * fever * tiredness * difficulty in breathing(severe cases).

Most people (about 80%) recover from the disease without needing special treatment and being under Quarantine for 14-15 days.