Analysing Covid-19 Dataset

October 4, 2020

1 Analysing Covid-19 Dataset

[59]:
```
# !pip install folium
# !pip install plotly
```

[60]:
```
conda install -c conda-forge folium
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done

# All requested packages already installed.


Note: you may need to restart the kernel to use updated packages.
```

[61]:
```
# imports
import plotly.express as px
import plotly.graph_objects as go
import plotly.figure_factory as ff
from plotly.subplots import make_subplots

import folium


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

%matplotlib inline

import math
import random
from datetime import timedelta

import warnings
warnings.filterwarnings('ignore')
```

```
#color pallette
cnf = '#393e46'
dth = '#ff2e63'
rec = '#21bf73'
act = '#fe9801'
```

1.1 Dataset Preparation

[62]:
```
import plotly as py
py.offline.init_notebook_mode(connected = True)
```

[63]:
```
import os
```

[64]:
```
try:
    os.system("rm -rf Covid-19-Preprocessed-Dataset")
except:
    print('File does not exist')
```

[65]:
```
!git clone https://github.com/laxmimerit/Covid-19-Preprocessed-Dataset.git
```

```
Cloning into 'Covid-19-Preprocessed-Dataset'…
remote: Enumerating objects: 374, done.
remote: Counting objects: 100% (374/374), done.
remote: Compressing objects: 100% (193/193), done.
remote: Total 771 (delta 197), reused 355 (delta 181), pack-reused 397
Receiving objects: 100% (771/771), 4.85 MiB | 3.11 MiB/s, done.
Resolving deltas: 100% (448/448), done.
```

[66]:
```
df = pd.read_csv('Covid-19-Preprocessed-Dataset/preprocessed/
 ↪covid_19_data_cleaned.csv', parse_dates=['Date'])

country_daywise = pd.read_csv('Covid-19-Preprocessed-Dataset/preprocessed/
 ↪country_daywise.csv', parse_dates=['Date'])
countywise = pd.read_csv('Covid-19-Preprocessed-Dataset/preprocessed/
 ↪countrywise.csv')
daywise = pd.read_csv('Covid-19-Preprocessed-Dataset/preprocessed/daywise.csv',␣
 ↪parse_dates=['Date'])
```

[67]:
```
df['Province/State'] = df['Province/State'].fillna("")
df.head()
```

[67]:

	Date	Province/State	Country	Lat	Long	Confirmed	\
0	2020-01-22		Afghanistan	33.93911	67.709953	0	
1	2020-01-23		Afghanistan	33.93911	67.709953	0	
2	2020-01-24		Afghanistan	33.93911	67.709953	0	
3	2020-01-25		Afghanistan	33.93911	67.709953	0	

```
4 2020-01-26                Afghanistan  33.93911  67.709953          0

   Recovered  Deaths  Active
0          0       0       0
1          0       0       0
2          0       0       0
3          0       0       0
4          0       0       0
```

[68]:
```
confirmed = df.groupby('Date').sum()['Confirmed'].reset_index()
recovered = df.groupby('Date').sum()['Recovered'].reset_index()
deaths = df.groupby('Date').sum()['Deaths'].reset_index()
deaths.head()
```

[68]:
```
        Date  Deaths
0 2020-01-22      17
1 2020-01-23      18
2 2020-01-24      26
3 2020-01-25      42
4 2020-01-26      56
```

[69]:
```
df.isnull().sum()
```

[69]:
```
Date              0
Province/State    0
Country           0
Lat               0
Long              0
Confirmed         0
Recovered         0
Deaths            0
Active            0
dtype: int64
```

[70]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 75735 entries, 0 to 75734
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            75735 non-null  datetime64[ns]
 1   Province/State  75735 non-null  object
 2   Country         75735 non-null  object
 3   Lat             75735 non-null  float64
 4   Long            75735 non-null  float64
 5   Confirmed       75735 non-null  int64
```

```
 6   Recovered       75735 non-null  int64
 7   Deaths          75735 non-null  int64
 8   Active          75735 non-null  int64
dtypes: datetime64[ns](1), float64(2), int64(4), object(2)
memory usage: 5.2+ MB
```

[71]:
```
df.query('Country == "US"')
```

[71]:
```
            Date Province/State Country   Lat   Long  Confirmed  Recovered  \
61710 2020-01-22                     US  40.0 -100.0          1          0
61711 2020-01-23                     US  40.0 -100.0          1          0
61712 2020-01-24                     US  40.0 -100.0          2          0
61713 2020-01-25                     US  40.0 -100.0          2          0
61714 2020-01-26                     US  40.0 -100.0          5          0
…           …              …      …    …    …     …          …          …
61960 2020-09-28                     US  40.0 -100.0    7149537    2794608
61961 2020-09-29                     US  40.0 -100.0    7191637    2813305
61962 2020-09-30                     US  40.0 -100.0    7233042    2840688
61963 2020-10-01                     US  40.0 -100.0    7277759    2860650
61964 2020-10-02                     US  40.0 -100.0    7332200    2873369

       Deaths   Active
61710       0        1
61711       0        1
61712       0        2
61713       0        2
61714       0        5
…         …       …
61960  205072  4149857
61961  205986  4172346
61962  206932  4185422
61963  207789  4209320
61964  208695  4250136

[255 rows x 9 columns]
```

1.2 Worldwide Total Confirmed, Recovered, and Deaths

[72]:
```
confirmed.tail()
```

[72]:
```
           Date  Confirmed
250 2020-09-28   33355107
251 2020-09-29   33642960
252 2020-09-30   33968093
253 2020-10-01   34287239
254 2020-10-02   34503125
```

[73]:
```
recovered.tail()
```

[73]:

	Date	Recovered
250	2020-09-28	23151081
251	2020-09-29	23387690
252	2020-09-30	23637164
253	2020-10-01	23857656
254	2020-10-02	23953871

[74]:
```
deaths.tail()
```

[74]:

	Date	Deaths
250	2020-09-28	1001646
251	2020-09-29	1007755
252	2020-09-30	1014161
253	2020-10-01	1022858
254	2020-10-02	1026756

[75]:
```
fig = go.Figure()
fig.add_trace(go.Scatter(x = confirmed['Date'], y = confirmed['Confirmed'],␣
 ↪mode = 'lines+markers', name = 'Confirmed', line = dict(color = "Orange",␣
 ↪width = 2)))
fig.add_trace(go.Scatter(x = recovered['Date'], y = recovered['Recovered'],␣
 ↪mode = 'lines+markers', name = 'Recovered', line = dict(color = "Green",␣
 ↪width = 2)))
fig.add_trace(go.Scatter(x = deaths['Date'], y = deaths['Deaths'], mode =␣
 ↪'lines+markers', name = 'Deaths', line = dict(color = "Red", width = 2)))
fig.update_layout(title = 'Worldwide Covid-19 Cases', xaxis_tickfont_size = 14,␣
 ↪yaxis = dict(title = 'Number of Cases'))

fig.show()
```

1.3 Cases Density Animation on World Map

[76]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 75735 entries, 0 to 75734
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            75735 non-null  datetime64[ns]
 1   Province/State  75735 non-null  object
 2   Country         75735 non-null  object
 3   Lat             75735 non-null  float64
 4   Long            75735 non-null  float64
 5   Confirmed       75735 non-null  int64
```

```
 6   Recovered       75735 non-null  int64
 7   Deaths          75735 non-null  int64
 8   Active          75735 non-null  int64
dtypes: datetime64[ns](1), float64(2), int64(4), object(2)
memory usage: 5.2+ MB
```

[77]:
```
df['Date'] = df['Date'].astype(str)
```

[78]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 75735 entries, 0 to 75734
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            75735 non-null  object
 1   Province/State  75735 non-null  object
 2   Country         75735 non-null  object
 3   Lat             75735 non-null  float64
 4   Long            75735 non-null  float64
 5   Confirmed       75735 non-null  int64
 6   Recovered       75735 non-null  int64
 7   Deaths          75735 non-null  int64
 8   Active          75735 non-null  int64
dtypes: float64(2), int64(4), object(3)
memory usage: 5.2+ MB
```

[79]:
```
df.head()
```

[79]:

	Date	Province/State	Country	Lat	Long	Confirmed	Recovered	Deaths	Active
0	2020-01-22		Afghanistan	33.93911	67.709953	0	0	0	0
1	2020-01-23		Afghanistan	33.93911	67.709953	0	0	0	0
2	2020-01-24		Afghanistan	33.93911	67.709953	0	0	0	0
3	2020-01-25		Afghanistan	33.93911	67.709953	0	0	0	0
4	2020-01-26		Afghanistan	33.93911	67.709953	0	0	0	0

[80]:
```
fig = px.density_mapbox(df, lat = 'Lat', lon = 'Long', hover_name = 'Country',␣
 ↪hover_data = ['Confirmed', 'Recovered', 'Deaths'], animation_frame='Date',␣
 ↪color_continuous_scale='Portland', radius = 7, zoom = 0, height= 700)
fig.update_layout(title = 'Worldwide Covid-19 Cases with Time Lapse')
```

```
fig.update_layout(mapbox_style = 'open-street-map', mapbox_center_lon = 0)


fig.show()
```

1.4 Total cases on ships

[81]:
```
df['Date'] = pd.to_datetime(df['Date'])
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 75735 entries, 0 to 75734
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            75735 non-null  datetime64[ns]
 1   Province/State  75735 non-null  object
 2   Country         75735 non-null  object
 3   Lat             75735 non-null  float64
 4   Long            75735 non-null  float64
 5   Confirmed       75735 non-null  int64
 6   Recovered       75735 non-null  int64
 7   Deaths          75735 non-null  int64
 8   Active          75735 non-null  int64
dtypes: datetime64[ns](1), float64(2), int64(4), object(2)
memory usage: 5.2+ MB
```

[82]:
```
# Ships
# ======================

ship_rows = df['Province/State'].str.contains('Grand Princess') | df['Province/
 ↪State'].str.contains('Diamond Princess') | df['Country'].str.contains('Grand␣
 ↪Princess') | df['Country'].str.contains('Diamond Princess') | df['Country'].
 ↪str.contains('MS Zaandam')
ship = df[ship_rows]

df = df[~ship_rows]
```

[83]:
```
ship_latest = ship[ship['Date'] == max(ship['Date'])]
ship_latest
```

[83]:

	Date	Province/State	Country	Lat	Long	Confirmed	\
10709	2020-10-02	Diamond Princess	Canada	0.0	0.0	0	
10964	2020-10-02	Grand Princess	Canada	0.0	0.0	13	
26264	2020-10-02		Diamond Princess	0.0	0.0	712	
43094	2020-10-02		MS Zaandam	0.0	0.0	9	

	Recovered	Deaths	Active
10709	0	1	-1
10964	0	0	13
26264	651	13	48
43094	0	2	7

[84]:
```
ship_latest.style.background_gradient(cmap = 'Pastel1_r')
```

[84]:
```
<pandas.io.formats.style.Styler at 0x12564ea30>
```

1.5 Cases Over the Time with Area Plot

[85]:
```
temp = df.groupby('Date')['Confirmed', 'Deaths', 'Recovered', 'Active'].sum().
 ↪reset_index()
temp = temp[temp['Date']==max(temp['Date'])].reset_index(drop = True)

tm = temp.melt(id_vars = 'Date', value_vars = ['Active', 'Deaths', 'Recovered'])
fig = px.treemap(tm, path = ['variable'], values = 'value', height = 250, width␣
 ↪= 800, color_discrete_sequence=[act, rec, dth])

fig.data[0].textinfo = 'label+text+value'
fig.show()
```

[86]:
```
temp = df.groupby('Date')['Recovered', 'Deaths', 'Active'].sum().reset_index()
temp = temp.melt(id_vars = 'Date', value_vars = ['Recovered', 'Deaths',␣
 ↪'Active'], var_name = 'Case', value_name = 'Count')

fig = px.area(temp, x = 'Date', y = 'Count', color= 'Case', height = 400, title␣
 ↪= 'Cases over time', color_discrete_sequence=[rec, dth,  act])
fig.update_layout(xaxis_rangeslider_visible=True)
fig.show()
```

1.6 Folium Maps

[87]:
```
temp = df[df['Date']==max(df['Date'])]

m = folium.Map(location=[0, 0], tiles='cartodbpositron', min_zoom = 1,␣
 ↪max_zoom=4, zoom_start=1)

for i in range(0, len(temp)):
    folium.Circle(location=[temp.iloc[i]['Lat'], temp.iloc[i]['Long']], color =␣
 ↪'crimson', fill = 'crimson',
                  tooltip = '<li><bold> Country: ' + str(temp.
 ↪iloc[i]['Country'])+
                            '<li><bold> Province: ' + str(temp.
 ↪iloc[i]['Province/State'])+
```

```
                              '<li><bold> Confirmed: ' + str(temp.
↪iloc[i]['Confirmed'])+
                              '<li><bold> Deaths: ' + str(temp.
↪iloc[i]['Deaths']),
                radius = int(temp.iloc[i]['Confirmed'])**0.5).add_to(m)


m
```

[87]:
```
<folium.folium.Map at 0x125c1d8e0>
```

1.7 Confirmed Cases with Choropleth Map

[88]:
```
country_daywise.head()
```

[88]:

	Date	Country	Confirmed	Deaths	Recovered	Active	New Cases	\
0	2020-01-22	Afghanistan	0	0	0	0	0	
1	2020-01-22	Albania	0	0	0	0	0	
2	2020-01-22	Algeria	0	0	0	0	0	
3	2020-01-22	Andorra	0	0	0	0	0	
4	2020-01-22	Angola	0	0	0	0	0	

	New Deaths	New Recovered
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0

[89]:
```
fig = px.choropleth(country_daywise, locations= 'Country',␣
↪locationmode='country names', color = np.log(country_daywise['Confirmed']),
                    hover_name = 'Country',␣
↪animation_frame=country_daywise['Date'].dt.strftime('%Y-%m-%d'),
                    title='Cases over time', color_continuous_scale=px.colors.
↪sequential.Inferno)

fig.update(layout_coloraxis_showscale = True)
fig.show()
```

[90]:
```
fig = px.choropleth(country_daywise, locations= 'Country',␣
↪locationmode='country names', color = country_daywise['Confirmed'],
                    hover_name = 'Country',␣
↪animation_frame=country_daywise['Date'].dt.strftime('%Y-%m-%d'),
                    title='Cases over time', color_continuous_scale=px.colors.
↪sequential.Inferno)
```

```
fig.update(layout_coloraxis_showscale = True)
fig.show()
```

1.8 New Cases and No. of Countries

[91]:
```
fig_c = px.bar(daywise, x = 'Date', y = 'Confirmed',␣
↪color_discrete_sequence=[act])
fig_d = px.bar(daywise, x = 'Date', y = 'No. of Countries',␣
↪color_discrete_sequence=[dth])

fig = make_subplots(rows = 1, cols = 2, shared_xaxes=False,␣
↪horizontal_spacing=0.1,
                    subplot_titles=('No. of New Cases per Day', 'No. of␣
↪Countries'))

fig.add_trace(fig_c['data'][0], row = 1, col = 1)
fig.add_trace(fig_d['data'][0], row = 1, col = 2)

fig.show()
```

1.9 Top 15 Countries Case Analysis

[92]:
```
countywise.columns
```

[92]:
```
Index(['Country', 'Confirmed', 'Deaths', 'Recovered', 'Active', 'New Cases',
       'Deaths / 100 Cases', 'Recovered / 100 Cases', 'Deaths / 100 Recovered',
       'Population', 'Cases / Million People', 'Confirmed last week',
       '1 week change', '1 week % increase'],
      dtype='object')
```

[93]:
```
top  = 15

fig_c = px.bar(countywise.sort_values('Confirmed').tail(top), x = 'Confirmed',␣
↪y = 'Country',
             text = 'Confirmed', orientation='h',␣
↪color_discrete_sequence=[act])
fig_d = px.bar(countywise.sort_values('Deaths').tail(top), x = 'Deaths', y =␣
↪'Country',
             text = 'Deaths', orientation='h', color_discrete_sequence=[dth])


fig_a = px.bar(countywise.sort_values('Active').tail(top), x = 'Active', y =␣
↪'Country',
             text = 'Active', orientation='h',␣
↪color_discrete_sequence=['#434343'])
```

```
fig_r = px.bar(countywise.sort_values('Recovered').tail(top), x = 'Recovered',␣
↪y = 'Country',
             text = 'Recovered', orientation='h',␣
↪color_discrete_sequence=[rec])


fig_dc = px.bar(countywise.sort_values('Deaths / 100 Cases').tail(top), x =␣
↪'Deaths / 100 Cases', y = 'Country',
             text = 'Deaths / 100 Cases', orientation='h',␣
↪color_discrete_sequence=['#f84351'])
fig_rc = px.bar(countywise.sort_values('Recovered / 100 Cases').tail(top), x =␣
↪'Recovered / 100 Cases', y = 'Country',
             text = 'Recovered / 100 Cases', orientation='h',␣
↪color_discrete_sequence=['#a45398'])


fig_nc = px.bar(countywise.sort_values('New Cases').tail(top), x = 'New Cases',␣
↪y = 'Country',
             text = 'New Cases', orientation='h',␣
↪color_discrete_sequence=['#f04341'])
temp = countywise[countywise['Population']>1000000]
fig_p = px.bar(temp.sort_values('Cases / Million People').tail(top), x = 'Cases␣
↪/ Million People', y = 'Country',
             text = 'Cases / Million People', orientation='h',␣
↪color_discrete_sequence=['#b40398'])



fig_wc = px.bar(countywise.sort_values('1 week change').tail(top), x = '1 week␣
↪change', y = 'Country',
             text = '1 week change', orientation='h',␣
↪color_discrete_sequence=['#c04041'])
temp = countywise[countywise['Confirmed']>100]
fig_wi = px.bar(temp.sort_values('1 week % increase').tail(top), x = '1 week %␣
↪increase', y = 'Country',
             text = '1 week % increase', orientation='h',␣
↪color_discrete_sequence=['#b00398'])


fig = make_subplots(rows = 5, cols = 2, shared_xaxes=False,␣
↪horizontal_spacing=0.2,
                    vertical_spacing=.05,
                   subplot_titles=('Confirmed Cases', 'Deaths Reported',␣
↪'Recovered Cases', 'Active Cases',
                                   'Deaths / 100 Cases', 'Recovered / 100 Cases',
                                   'New Cases', 'Cases / Million People',
```

```
                                    '1 week change', '1 week % increase'))

fig.add_trace(fig_c['data'][0], row = 1, col = 1)
fig.add_trace(fig_d['data'][0], row = 1, col = 2)

fig.add_trace(fig_r['data'][0], row = 2, col = 1)
fig.add_trace(fig_a['data'][0], row = 2, col = 2)

fig.add_trace(fig_dc['data'][0], row = 3, col = 1)
fig.add_trace(fig_rc['data'][0], row = 3, col = 2)

fig.add_trace(fig_nc['data'][0], row = 4, col = 1)
fig.add_trace(fig_p['data'][0], row = 4, col = 2)

fig.add_trace(fig_wc['data'][0], row = 5, col = 1)
fig.add_trace(fig_wi['data'][0], row = 5, col = 2)

fig.update_layout(height = 3000)
fig.show()
```

1.10 Scatter Plot for Deaths vs Confirmed Cases

[94]:
```
top = 15
fig = px.scatter(countywise.sort_values('Deaths', ascending = False).head(top),
                 x = 'Confirmed', y = 'Deaths', color = 'Country', size =␣
↪'Confirmed', height = 600,
                 text = 'Country', log_x = True, log_y = True, title='Deaths vs␣
↪Confirmed Cases (Cases are on log10 scale)')

fig.update_traces(textposition = 'top center')
fig.update_layout(showlegend = False)
fig.update_layout(xaxis_rangeslider_visible = True)
fig.show()
```

[95]:
```
countywise.sort_values('Deaths', ascending = False).head(15)
```

[95]:

	Country	Confirmed	Deaths	Recovered	Active	New Cases	\
174	US	6365325	190846	2387479	3787000	33552	
23	Brazil	4197889	128539	3611632	457718	35816	
79	India	4465863	75062	3471783	919018	95735	
113	Mexico	647321	69049	538514	39758	4461	
178	United Kingdom	357613	41683	1831	314099	2681	
85	Italy	281583	35577	211272	34734	1430	
62	France	383292	30805	89242	263245	9574	
134	Peru	696190	30123	536959	129108	4615	
158	Spain	543379	29628	150376	363375	8866	
81	Iran	393425	22669	339111	31645	2313	

37	Colombia	686851	22053	552885	111913	15318
140	Russia	1037526	18080	854069	165377	5172
156	South Africa	642431	15168	569935	57328	1990
35	Chile	427027	11702	399555	15770	1486
52	Ecuador	112166	10701	91242	10223	1409

	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	\
174	3.00	37.51	7.99	
23	3.06	86.03	3.56	
79	1.68	77.74	2.16	
113	10.67	83.19	12.82	
178	11.66	0.51	2276.52	
85	12.63	75.03	16.84	
62	8.04	23.28	34.52	
134	4.33	77.13	5.61	
158	5.45	27.67	19.70	
81	5.76	86.19	6.68	
37	3.21	80.50	3.99	
140	1.74	82.32	2.12	
156	2.36	88.72	2.66	
35	2.74	93.57	2.93	
52	9.54	81.35	11.73	

	Population	Cases / Million People	Confirmed last week	1 week change	\
174	330541757	19257.0	7034931	297269	
23	422706534	9931.0	4689613	157479	
79	-2147483648	-2080.0	5903932	490136	
113	255584572	2533.0	720858	32232	
178	134856358	2652.0	425767	44002	
85	120822834	2331.0	306235	13673	
62	68136441	5625.0	552421	77010	
134	65597846	10613.0	794584	23713	
158	93691843	5800.0	716481	73451	
81	83992953	4684.0	439882	24714	
37	99141018	6928.0	798317	43215	
140	292579178	3546.0	1131088	57840	
156	59308690	10832.0	668529	9304	
35	36692988	11638.0	453868	12722	
52	17643060	6358.0	132475	7059	

	1 week % increase
174	4.23
23	3.36
79	8.30
113	4.47
178	10.33
85	4.46

```
62              13.94
134              2.98
158             10.25
81               5.62
37               5.41
140              5.11
156              1.39
35               2.80
52               5.33
```

1.11 First and Last Case Report Time

[96]:
```
first_date = df[df['Confirmed']>0]
first_date = first_date.groupby('Country')['Date'].agg(['min']).reset_index()


last_date = df.groupby(['Country', 'Date'])['Confirmed', 'Deaths', 'Recovered']
last_date = last_date.sum().diff().reset_index()


mask = (last_date['Country'] != last_date['Country'].shift(1))

last_date.loc[mask, 'Confirmed'] = np.nan
last_date.loc[mask, 'Deaths'] = np.nan
last_date.loc[mask, 'Recovered'] = np.nan

last_date = last_date[last_date['Confirmed']>0]
last_date = last_date.groupby('Country')['Date'].agg(['max']).reset_index()


first_last = pd.concat([first_date, last_date['max']], axis = 1)
first_last['max'] = first_last['max'] + timedelta(days = 1)

first_last['Days'] = first_last['max'] - first_last['min']
first_last['Task'] = first_last['Country']

first_last.columns = ['Country', 'Start', 'Finish', 'Days', 'Task']

first_last = first_last.sort_values('Days')

colors = ['#' + ''.join([random.choice('0123456789ABCDEF') for j in range(6)])␣
 ↪for i in range(len(first_last))]

fig = ff.create_gantt(first_last, index_col = 'Country', colors = colors,␣
 ↪show_colorbar = False,
```

```
                    bar_width=0.2, showgrid_x = True, showgrid_y=True, height␣
↪= 2500)

fig.show()
```