INFO 5709 Data Visualization and Communications

Importance of data in monitoring COVID-19 spread

Amulya Akinapuram 11558530

```
# Import Python Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns

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

Loading the .csv file

```
country_wise_df=pd.read_csv("country_wise_latest.csv")
```

```
country_wise_df.head()
```

	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered
0	Afghanistan	36263	1269	25198	9796	106	10	18	3.50	69.49	5.04
1	Albania	4880	144	2745	1991	117	6	63	2.95	56.25	5.25
2	Algeria	27973	1163	18837	7973	616	8	749	4.16	67.34	6.17
3	Andorra	907	52	803	52	10	0	0	5.73	88.53	6.48
4	Angola	950	41	242	667	18	1	0	4.32	25.47	16.94

```
country_wise_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 187 entries, 0 to 186
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Country/Region          187 non-null    object
 1   Confirmed               187 non-null    int64
 2   Deaths                  187 non-null    int64
 3   Recovered               187 non-null    int64
 4   Active                  187 non-null    int64
 5   New cases               187 non-null    int64
 6   New deaths              187 non-null    int64
 7   New recovered           187 non-null    int64
 8   Deaths / 100 Cases      187 non-null    float64
 9   Recovered / 100 Cases   187 non-null    float64
 10  Deaths / 100 Recovered  187 non-null    float64
 11  Confirmed last week     187 non-null    int64
 12  1 week change           187 non-null    int64
 13  1 week % increase       187 non-null    float64
 14  WHO Region              187 non-null    object
dtypes: float64(4), int64(9), object(2)
memory usage: 22.0+ KB
```

```
country_wise_df.describe()
```

	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths 100 Case
count	1.870000e+02	187.000000	1.870000e+02	1.870000e+02	187.000000	187.000000	187.000000	187.00000
mean	8.813094e+04	3497.518717	5.063148e+04	3.400194e+04	1222.957219	28.957219	933.812834	3.01951
std	3.833187e+05	14100.002482	1.901882e+05	2.133262e+05	5710.374790	120.037173	4197.719635	3.45430
min	1.000000e+01	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.00000
25%	1.114000e+03	18.500000	6.265000e+02	1.415000e+02	4.000000	0.000000	0.000000	0.94500
50%	5.059000e+03	108.000000	2.815000e+03	1.600000e+03	49.000000	1.000000	22.000000	2.15000
75%	4.046050e+04	734.000000	2.260600e+04	9.149000e+03	419.500000	6.000000	221.000000	3.87500
max	4.290259e+06	148011.000000	1.846641e+06	2.816444e+06	56336.000000	1076.000000	33728.000000	28.56000

Checking for missing values

```
country_wise_df.isnull().sum()
```

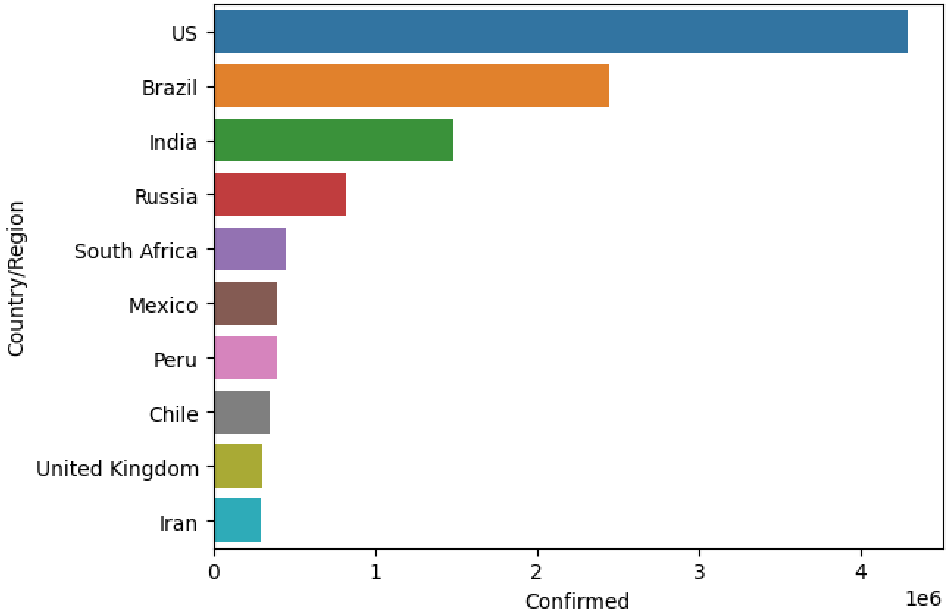
```
Country/Region            0
Confirmed                 0
Deaths                    0
Recovered                 0
Active                    0
New cases                 0
New deaths                0
New recovered             0
Deaths / 100 Cases        0
Recovered / 100 Cases     0
Deaths / 100 Recovered    0
Confirmed last week       0
1 week change             0
1 week % increase         0
WHO Region                0
dtype: int64
```

```
#sorting values
ConfirmedCases=country_wise_df[['Country/Region','Confirmed']].sort_values(by=['Confirmed'],ascending=False).head(10)
ConfirmedCases
```

Country/Region	Confirmed

```
#plotting bar plot
sns.barplot(ConfirmedCases,x='Confirmed',y='Country/Region')
```

```
<Axes: xlabel='Confirmed', ylabel='Country/Region'>
```



```
#correlation and heatmap
corr=country_wise_df.corr()
f, ax = plt.subplots(figsize=(12, 6))
mask = np.triu(np.ones_like(corr, dtype=bool))
sns.heatmap(corr, annot=True, mask = mask,cmap="Blues")
```

```
<Axes: >
```

```
# Import label encoder
from sklearn import preprocessing

# label_encoder object knows
# how to understand word labels.
label_encoder = preprocessing.LabelEncoder()

# Encode labels in column 'species'.
country_wise_df['Country/Region']= label_encoder.fit_transform(country_wise_df['Country/Region'])
country_wise_df['WHO Region']= label_encoder.fit_transform(country_wise_df['WHO Region'])


print(country_wise_df['Country/Region'].unique())
print(country_wise_df['WHO Region'].unique())
```

```
[  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35
  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53
  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71
  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89
  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107
 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125
 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143
 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161
 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179
 180 181 182 183 184 185 186]
[2 3 0 1 5 4]
```

predicting the death rate

```
# Putting feature variable to X
X = country_wise_df[['Country/Region','WHO Region','Confirmed','Active','Recovered']]
# Putting response variable to y
y = country_wise_df['Deaths']


# now lets split the data into train and test
from sklearn.model_selection import train_test_split
# Splitting the data into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
X_train.shape, X_test.shape
```

```
((149, 5), (38, 5))
```

```
from sklearn.ensemble import RandomForestClassifier
classifier_rf = RandomForestClassifier(random_state=42, n_jobs=-1, max_depth=5,
                                       n_estimators=100, oob_score=True)

classifier_rf.fit(X_train, y_train)

# checking the oob score
classifier_rf.oob_score_
```

```
0.06711409395973154
```

```
#prediction
y_pred=classifier_rf.predict(X_test)
y_pred
```

```
array([   78,    69,     1,   165,   543,   165,     0, 44022,    51,
         345,    15,   228,    45,     0,  4652,   474,    20,  8777,
         165,     7,   285,    11,     7,     0,     7,    80,   167,
       44022,  1676,  1676,    11,   165,   285,   285,     0, 44022,
          11,  5842])
```

```
from sklearn.metrics import accuracy_score
accuracy=accuracy_score(y_test,y_pred)
print('Accuracy:', round(accuracy, 2), '%.')
```

```
    Accuracy: 0.11 %.
```

This algorithm is not a good model to predict the death rates

Loading .csv file

```
covid_df=pd.read_csv("covid_19_clean_complete.csv")
```

```
covid_df.head()
```

	Province/State	Country/Region	Lat	Long	Date	Confirmed	Deaths	Recovered	Active	WHO Regio
0	NaN	Afghanistan	33.93911	67.709953	2020-01-22	0	0	0	0	Easter Mediterranea
1	NaN	Albania	41.15330	20.168300	2020-01-22	0	0	0	0	Europ
2	NaN	Algeria	28.03390	1.659600	2020-01-22	0	0	0	0	Afric
					2020-					

```
covid_df.describe()
```

	Lat	Long	Confirmed	Deaths	Recovered	Active
count	49068.000000	49068.000000	4.906800e+04	49068.000000	4.906800e+04	4.906800e+04
mean	21.433730	23.528236	1.688490e+04	884.179160	7.915713e+03	8.085012e+03
std	24.950320	70.442740	1.273002e+05	6313.584411	5.480092e+04	7.625890e+04
min	-51.796300	-135.000000	0.000000e+00	0.000000	0.000000e+00	-1.400000e+01
25%	7.873054	-15.310100	4.000000e+00	0.000000	0.000000e+00	0.000000e+00
50%	23.634500	21.745300	1.680000e+02	2.000000	2.900000e+01	2.600000e+01
75%	41.204380	80.771797	1.518250e+03	30.000000	6.660000e+02	6.060000e+02
max	71.706900	178.065000	4.290259e+06	148011.000000	1.846641e+06	2.816444e+06

```
covid_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49068 entries, 0 to 49067
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Province/State  14664 non-null  object 
 1   Country/Region  49068 non-null  object 
 2   Lat             49068 non-null  float64
 3   Long            49068 non-null  float64
 4   Date            49068 non-null  object 
 5   Confirmed       49068 non-null  int64  
 6   Deaths          49068 non-null  int64  
 7   Recovered       49068 non-null  int64  
 8   Active          49068 non-null  int64  
 9   WHO Region      49068 non-null  object 
dtypes: float64(2), int64(4), object(4)
memory usage: 3.7+ MB
```

Checking for missing values

```
covid_df.isnull().sum()
```

```
Province/State    34404
Country/Region        0
Lat                   0
Long                  0
Date                  0
Confirmed             0
Deaths                0
Recovered             0
Active                0
WHO Region            0
dtype: int64
```

Dropping missing values

```
covid_df=covid_df.dropna()
```

```
covid_df
```

	Province/State	Country/Region	Lat	Long	Date	Confirmed	Deaths	Recovered	Active	WHO Region
8	Australian Capital Territory	Australia	-35.4735	149.0124	2020-01-22	0	0	0	0	Western Pacific
9	New South Wales	Australia	-33.8688	151.2093	2020-01-22	0	0	0	0	Western Pacific
10	Northern Territory	Australia	-12.4634	130.8456	2020-01-22	0	0	0	0	Western Pacific
11	Queensland	Australia	-27.4698	153.0251	2020-01-22	0	0	0	0	Western Pacific
12	South Australia	Australia	-34.9285	138.6007	2020-01-22	0	0	0	0	Western Pacific
...	...	...	...	...	...	...	...	...	...	...
49052	Anguilla	United Kingdom	18.2206	-63.0686	2020-07-27	3	0	3	0	Europe
49053	British Virgin Islands	United Kingdom	18.4207	-64.6400	2020-07-27	8	1	7	0	Europe

Again checking of missing values

```
covid_df.isnull().sum()
```

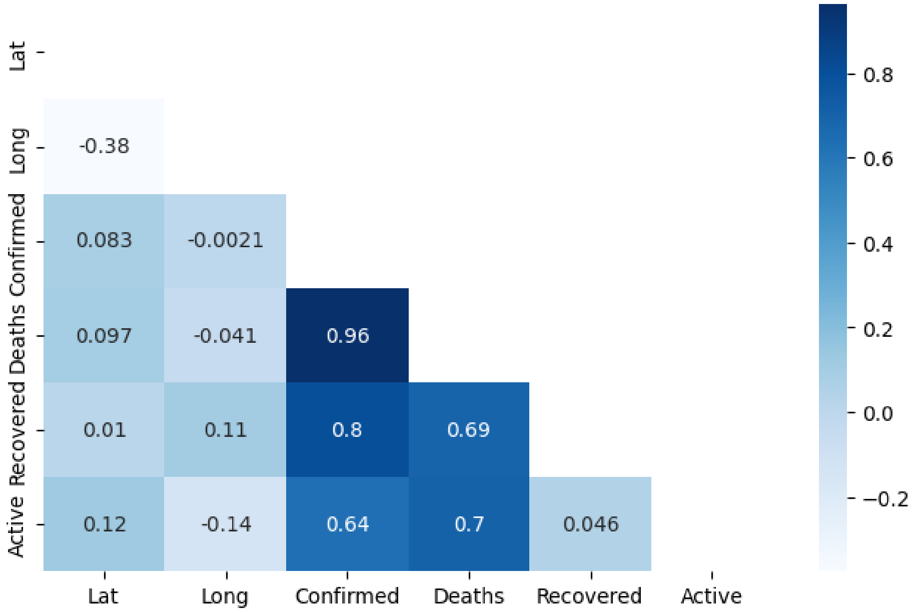
```
Province/State    0
Country/Region    0
Lat               0
Long              0
Date              0
Confirmed         0
Deaths            0
Recovered         0
Active            0
WHO Region        0
dtype: int64
```

```
#Renaming the column
covid_df=covid_df.rename(columns={'Country/Region':'Country'})
covid_df
```

	Province/State	Country	Lat	Long	Date	Confirmed	Deaths	Recovered	Active	WHO Region
8	Australian Capital Territory	Australia	-35.4735	149.0124	2020-01-22	0	0	0	0	Western Pacific
9	New South Wales	Australia	-33.8688	151.2093	2020-01-22	0	0	0	0	Western Pacific
10	Northern Territory	Australia	-12.4634	130.8456	2020-01-22	0	0	0	0	Western Pacific
11	Queensland	Australia	-27.4698	153.0251	2020-01-22	0	0	0	0	Western Pacific
12	South Australia	Australia	-34.9285	138.6007	2020-01-22	0	0	0	0	Western Pacific
...	...	...	...	...	...	...	...	...	...	...

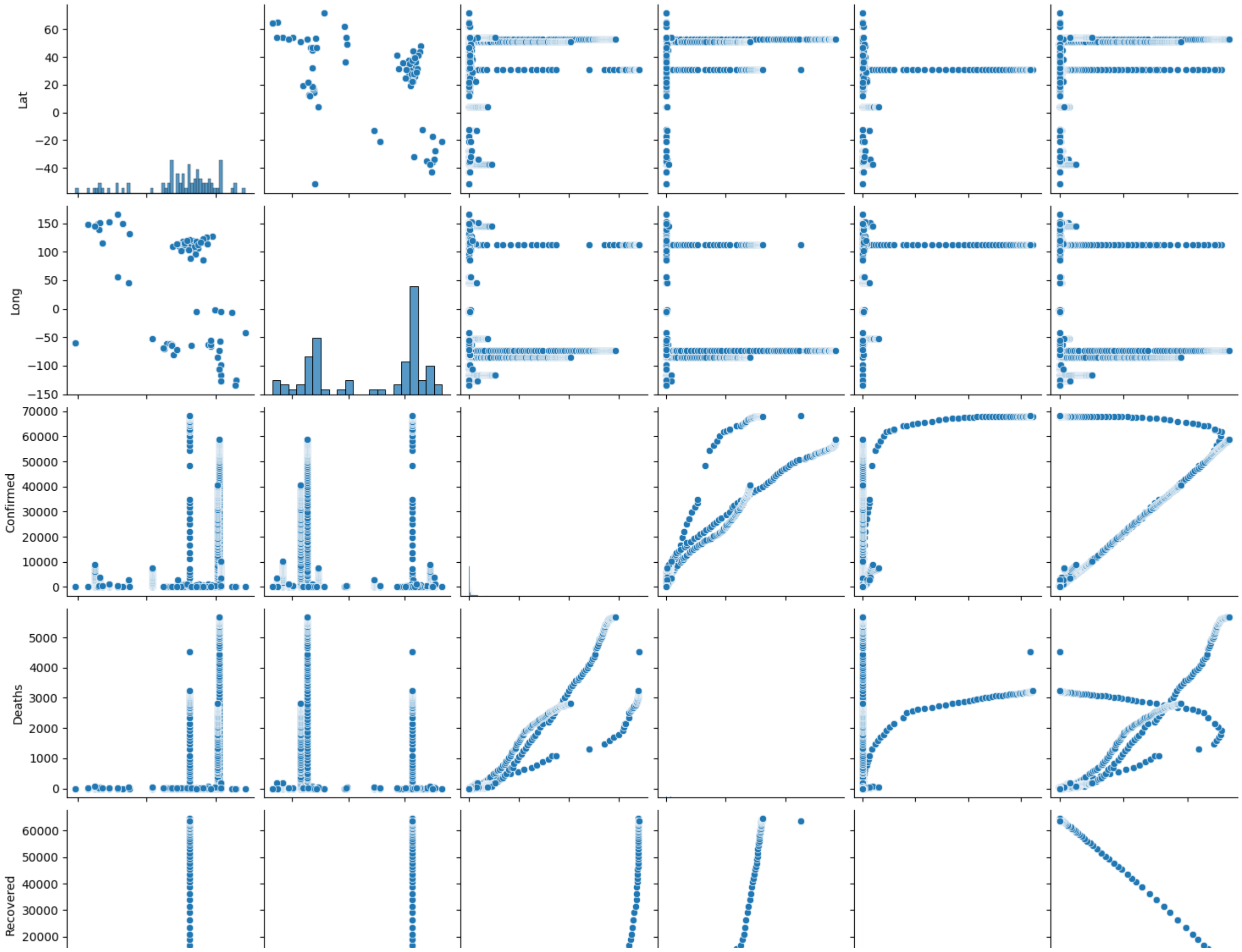
```
#correlation and heatmap
corr=covid_df.corr()
f, ax = plt.subplots(figsize=(8, 5))
mask = np.triu(np.ones_like(corr, dtype=bool))
sns.heatmap(corr, annot=True, mask = mask,cmap="Blues")
```

```
<Axes: >
```



```
#Plotting pairplot
sns.pairplot(covid_df)
```

```
<seaborn.axisgrid.PairGrid at 0x7fbb3f7f33d0>
```

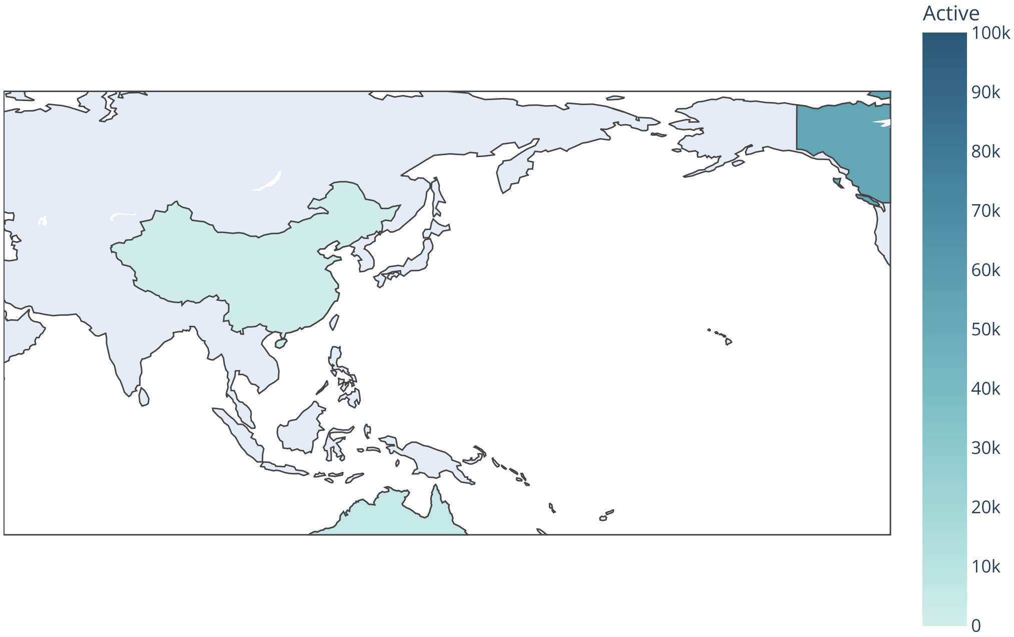


```
#choropleth map as GIS
import plotly.express as px
df1 = covid_df[covid_df[ 'Date' ] == max(covid_df[ 'Date'])]
df2 = df1.groupby('Country')['Confirmed'].max().reset_index()
fig = px.choropleth(df2, locations="Country",
locationmode='country names', color="Confirmed", range_color=[1,100000], color_continuous_scale="teal", title='Countries with C
fig.update(layout_coloraxis_showscale=True)
fig.update_layout(margin=dict(t=80,l=0, r=0,b=0))
fig
```

Countries with Confirmed Cases

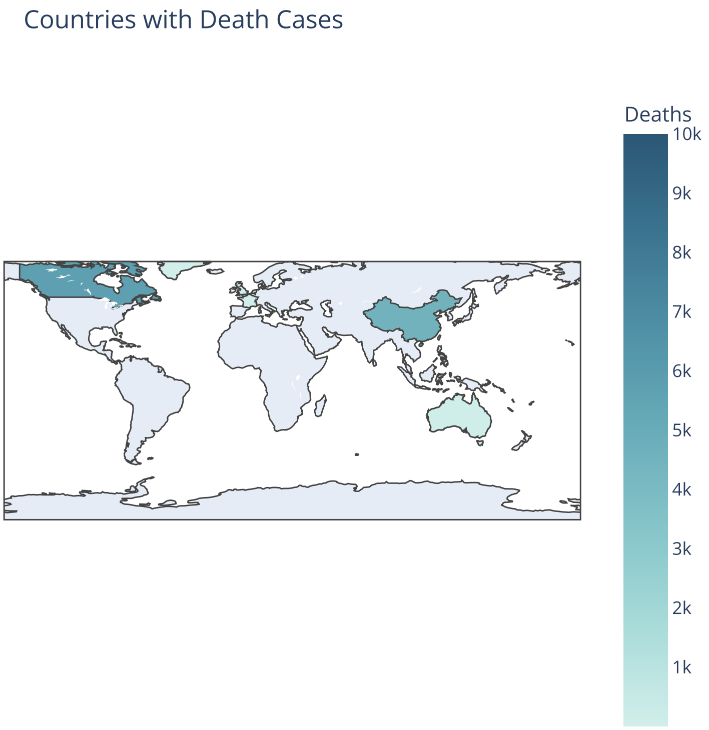
```
#choropleth map as GIS
import plotly.express as px
df1 = covid_df[covid_df[ 'Date' ] == max(covid_df[ 'Date'])]
df2 = df1.groupby('Country')['Active'].max().reset_index()
fig = px.choropleth(df2, locations="Country",
locationmode='country names', color="Active", range_color=[1,100000], color_continuous_scale="teal", title='Countries with Acti
fig.update(layout_coloraxis_showscale=True)
fig.update_layout(margin=dict(t=80,l=0, r=0,b=0))
fig
```

Countries with Active Cases

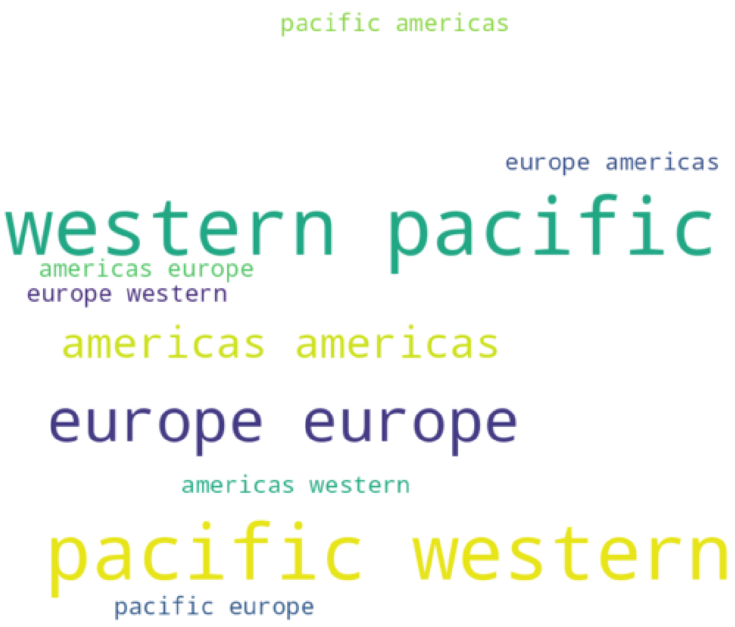


```
#choropleth map as GIS
import plotly.express as px
df1 = covid_df[covid_df[ 'Date' ] == max(covid_df[ 'Date'])]
df2 = df1.groupby('Country')['Recovered'].max().reset_index()
fig = px.choropleth(df2, locations="Country",
locationmode='country names', color="Recovered", range_color=[1,10000], color_continuous_scale="teal", title='Countries with Re
fig.update(layout_coloraxis_showscale=True)
fig.update_layout(margin=dict(t=80,l=0, r=0,b=0))
fig
```

```
#choropleth map as GIS
import plotly.express as px
df1 = covid_df[covid_df[ 'Date' ] == max(covid_df[ 'Date'])]
df2 = df1.groupby('Country')['Deaths'].max().reset_index()
fig = px.choropleth(df2, locations="Country",
locationmode='country names', color="Deaths", range_color=[1,10000], color_continuous_scale="teal", title='Countries with Death
fig.update(layout_coloraxis_showscale=True)
fig.update_layout(margin=dict(t=80,l=0, r=0,b=0))
fig
```



```
#Wordcloud
from wordcloud import WordCloud, STOPWORDS
Region = covid_df['WHO Region']
comment_words = ''
stopwords = set(STOPWORDS)

# iterate through the csv file
for val in Region:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()

    # Converts each token into lowercase
    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower()
        tokens[i] = tokens[i].lower()

    comment_words += " ".join(tokens)+" "

wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (5, 5), facecolor = None)
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.tight_layout(pad = 0)
plt.show()
```



Loading .csv file

```
world_df=pd.read_csv("worldometer_data.csv")
world_df.head()
```

	Country/Region	Continent	Population	TotalCases	NewCases	TotalDeaths	NewDeaths	TotalRecovered	NewRecovered	Act
0	USA	North America	3.311981e+08	5032179	NaN	162804.0	NaN	2576668.0	NaN	
1	Brazil	South America	2.127107e+08	2917562	NaN	98644.0	NaN	2047660.0	NaN	
2	India	Asia	1.381345e+09	2025409	NaN	41638.0	NaN	1377384.0	NaN	
3	Russia	Europe	1.459409e+08	871894	NaN	14606.0	NaN	676357.0	NaN	
4	South Africa	Africa	5.938157e+07	538184	NaN	9604.0	NaN	387316.0	NaN	

```
world_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209 entries, 0 to 208
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Country/Region    209 non-null    object 
 1   Continent         208 non-null    object 
 2   Population        208 non-null    float64
 3   TotalCases        209 non-null    int64  
 4   NewCases          4 non-null      float64
 5   TotalDeaths       188 non-null    float64
 6   NewDeaths         3 non-null      float64
 7   TotalRecovered    205 non-null    float64
 8   NewRecovered      3 non-null      float64
 9   ActiveCases       205 non-null    float64
 10  Serious,Critical  122 non-null    float64
 11  Tot Cases/1M pop  208 non-null    float64
 12  Deaths/1M pop     187 non-null    float64
 13  TotalTests        191 non-null    float64
 14  Tests/1M pop      191 non-null    float64
 15  WHO Region        184 non-null    object 
dtypes: float64(12), int64(1), object(3)
memory usage: 26.2+ KB
```

```
#dropping unnecessary columns
world_df=world_df.drop(columns=['NewCases','NewDeaths','NewRecovered'])
```

```
world_df=world_df.dropna()
```

```
world_df
```

	Country/Region	Continent	Population	TotalCases	TotalDeaths	TotalRecovered	ActiveCases	Serious,Critical	Cases
0	USA	North America	3.311981e+08	5032179	162804.0	2576668.0	2292707.0	18296.0	151
1	Brazil	South America	2.127107e+08	2917562	98644.0	2047660.0	771258.0	8318.0	137
2	India	Asia	1.381345e+09	2025409	41638.0	1377384.0	606387.0	8944.0	14
3	Russia	Europe	1.459409e+08	871894	14606.0	676357.0	180931.0	2300.0	59
4	South Africa	Africa	5.938157e+07	538184	9604.0	387316.0	141264.0	539.0	90
...	...	...	...	...	...	...	...	...	
153	Bahamas	North America	3.936160e+05	761	14.0	91.0	656.0	1.0	19
160	Guyana	South America	7.869360e+05	538	22.0	189.0	327.0	2.0	6
185	Monaco	Europe	3.927000e+04	125	4.0	105.0	16.0	2.0	31
187	Antigua and Barbuda	North America	9.801000e+04	92	3.0	76.0	13.0	1.0	9
189	Belize	North America	3.983120e+05	86	2.0	31.0	53.0	2.0	2

103 rows × 13 columns

```
world_df.isnull().sum()
```

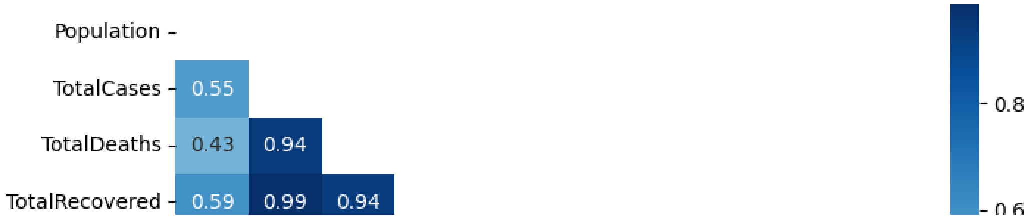
```
Country/Region      0
Continent           0
Population          0
TotalCases          0
TotalDeaths         0
TotalRecovered      0
ActiveCases         0
Serious,Critical    0
Tot Cases/1M pop    0
Deaths/1M pop       0
TotalTests          0
Tests/1M pop        0
WHO Region          0
dtype: int64
```

```
world_sum = world_df.groupby(by = 'Continent').sum()
print(world_sum)
```

```
                     Population  TotalCases  TotalDeaths  TotalRecovered  \
Continent
Africa             1.343515e+09     1011867      22114.0        693620.0
Asia               3.173656e+09     4689794     100627.0       3508170.0
Australia/Oceania  4.095791e+07       21735        281.0         12620.0
Europe             7.476775e+08     2982576     205232.0       1587302.0
North America      5.895035e+08     5919209     229855.0       3151678.0
South America      4.311105e+08     4543273     154885.0       3116150.0

                   ActiveCases  Serious,Critical  Tot Cases/1M pop  \
Continent
Africa                296133.0            1187.0           64456.0
Asia                 1080997.0           18749.0          192429.0
Australia/Oceania       8834.0              52.0            1446.0
Europe                475261.0            5200.0          209454.0
North America        2537676.0           25709.0           88547.0
South America        1272238.0           14295.0          108441.0

                   Deaths/1M pop  TotalTests  Tests/1M pop
Continent
Africa                   1003.28   8673853.0      806042.0
Asia                     1846.80  65353821.0     3433453.0
Australia/Oceania          15.30   5152811.0      347083.0
Europe                   9673.00  96125611.0     8286140.0
North America            3097.00  70173584.0     2069875.0
South America            2818.00  22379618.0     1093646.0
```

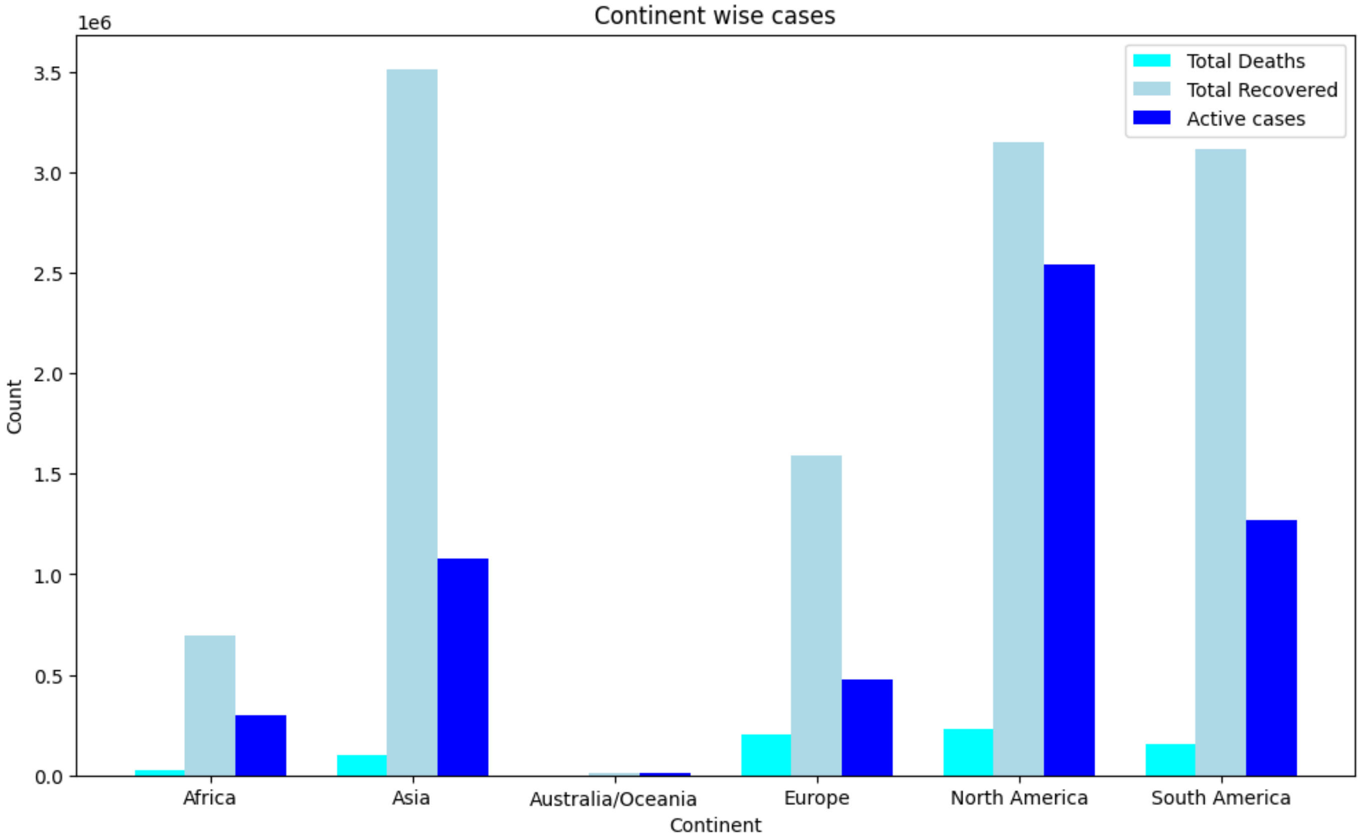
```
#correlation and heatmap
corr=world_df.corr()
f, ax = plt.subplots(figsize=(8, 5))
mask = np.triu(np.ones_like(corr, dtype=bool))
sns.heatmap(corr, annot=True, mask = mask,cmap="Blues")
```

```
<Axes: >
```



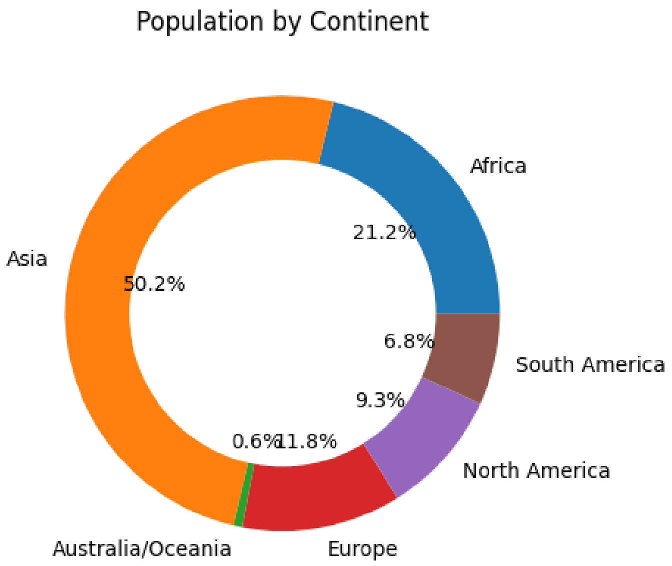
```
#wordCloud
from wordcloud import WordCloud, STOPWORDS

Region = world_df['Country/Region']
comment_words = ''
stopwords = set(STOPWORDS)

# iterate through the csv file
for val in Region:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()

    # Converts each token into lowercase
    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower()

    comment_words += " ".join(tokens)+" "


wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10,
                ).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (5, 5), facecolor = None)
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.tight_layout(pad = 0)
plt.show()
```

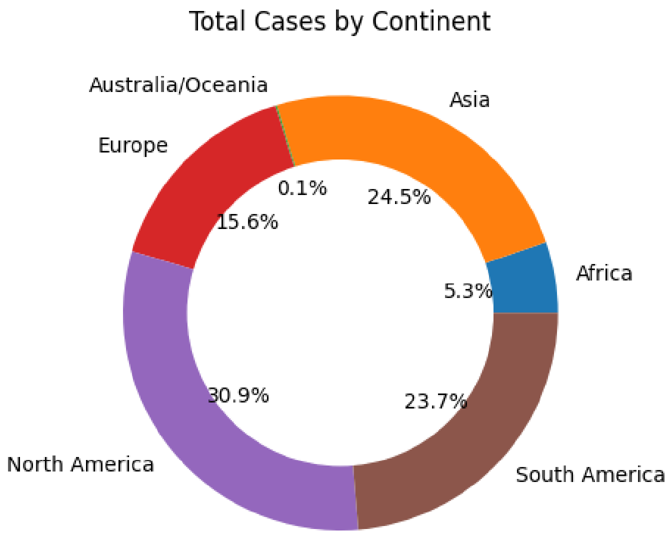
```
#side-by-side bar graph
import matplotlib.pyplot as plt
plt.figure(figsize = (12,7))
N = 6
ind = np.arange(N)
width = 0.25

bar1 = plt.bar(ind, world_sum['TotalDeaths'] , width, color = 'cyan')

bar2 = plt.bar(ind+width, world_sum['TotalRecovered'], width, color='lightblue')

bar3 = plt.bar(ind+width*2, world_sum['ActiveCases'], width, color = 'blue')

plt.xlabel("Continent")
plt.ylabel('Count')
plt.title("Continent wise cases")

plt.xticks(ind+width,['Africa', 'Asia', 'Australia/Oceania', 'Europe', 'North America', 'South America'])
plt.legend( (bar1, bar2, bar3), ('Total Deaths', 'Total Recovered', 'Active cases') )
plt.show()
```



```
# Group the data by continent and calculate the total population and cases
continent_data = world_df.groupby('Continent').agg({'Population': 'sum', 'TotalCases': 'sum'})

# Create a pie chart for population by continent
plt.pie(continent_data['Population'], labels=continent_data.index, autopct='%1.1f%%')
plt.title('Population by Continent')
my_circle=plt.Circle( (0,0), 0.7, color='white')
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.show()
```

```
# Create a pie chart for total cases by continent
plt.pie(continent_data['TotalCases'], labels=continent_data.index, autopct='%1.1f%%')
plt.title('Total Cases by Continent')
my_circle=plt.Circle( (0,0), 0.7, color='white')
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.show()
```





Loading .csv file

```
day_df=pd.read_csv('day_wise.csv')
day_df.head()
```

	Date	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	No. o countrie
0	2020-01-22	555	17	28	510	0	0	0	3.06	5.05	60.71	

```
day_df.info()
```

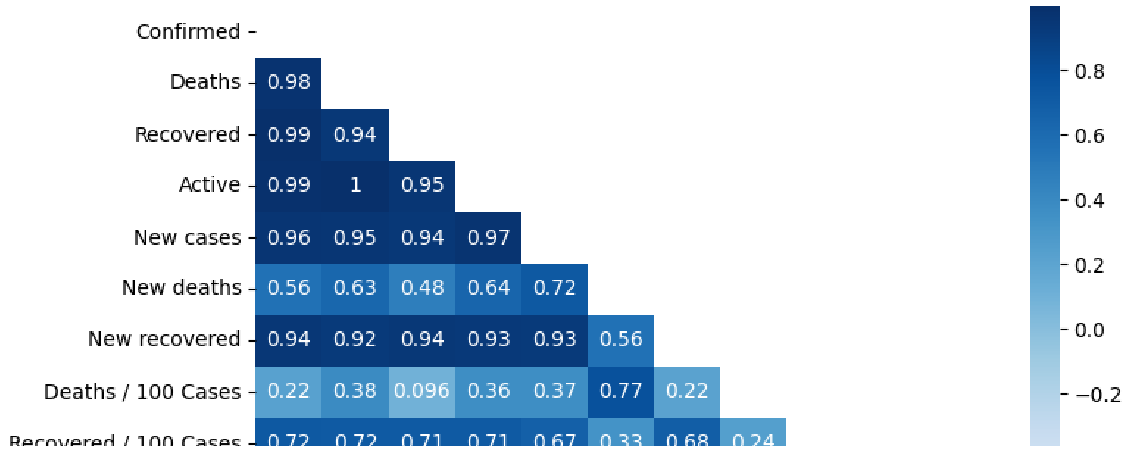
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 188 entries, 0 to 187
Data columns (total 12 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Date                   188 non-null    object 
 1   Confirmed              188 non-null    int64  
 2   Deaths                 188 non-null    int64  
 3   Recovered              188 non-null    int64  
 4   Active                 188 non-null    int64  
 5   New cases              188 non-null    int64  
 6   New deaths             188 non-null    int64  
 7   New recovered          188 non-null    int64  
 8   Deaths / 100 Cases     188 non-null    float64
 9   Recovered / 100 Cases  188 non-null    float64
 10  Deaths / 100 Recovered 188 non-null    float64
 11  No. of countries       188 non-null    int64  
dtypes: float64(3), int64(8), object(1)
memory usage: 17.8+ KB
```

```
day_df.describe()
```

	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Death 100 Ca
count	1.880000e+02	188.000000	1.880000e+02	1.880000e+02	188.000000	188.000000	188.000000	188.000
mean	4.406960e+06	230770.760638	2.066001e+06	2.110188e+06	87771.021277	3478.824468	50362.015957	4.860
std	4.757988e+06	217929.094183	2.627976e+06	1.969670e+06	75295.293255	2537.735652	56090.892479	1.579
min	5.550000e+02	17.000000	2.800000e+01	5.100000e+02	0.000000	0.000000	0.000000	2.040
25%	1.121910e+05	3935.000000	6.044125e+04	5.864175e+04	5568.500000	250.750000	2488.250000	3.510
50%	2.848733e+06	204190.000000	7.847840e+05	1.859759e+06	81114.000000	4116.000000	30991.500000	4.850
75%	7.422046e+06	418634.500000	3.416396e+06	3.587015e+06	131502.500000	5346.000000	79706.250000	6.297
max	1.648048e+07	654036.000000	9.468087e+06	6.358362e+06	282756.000000	9966.000000	284394.000000	7.180

```
#correlation and heatmap
corr=day_df.corr()
f, ax = plt.subplots(figsize=(8, 5))
mask = np.triu(np.ones_like(corr, dtype=bool))
sns.heatmap(corr, annot=True, mask = mask,cmap="Blues")
```

```
<Axes: >
```

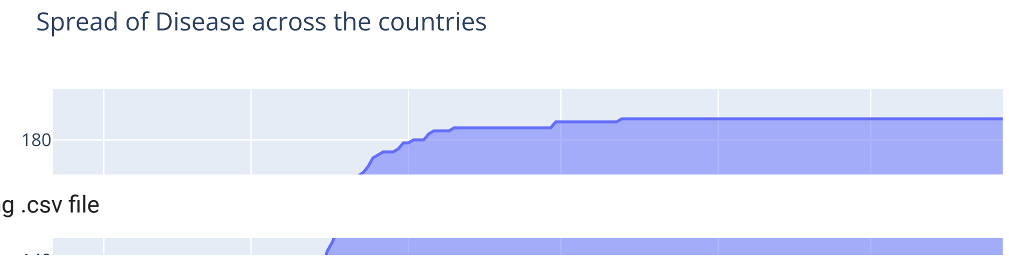


```
dates=day_df.groupby('Date')['Recovered','Deaths','Confirmed','Active','No. of countries'].max().reset_index()
dates
```

	Date	Recovered	Deaths	Confirmed	Active	No. of countries
0	2020-01-22	28	17	555	510	6
1	2020-01-23	30	18	654	606	8
2	2020-01-24	36	26	941	879	9
3	2020-01-25	39	42	1434	1353	11
4	2020-01-26	52	56	2118	2010	13
...	...	...	...	...	...	...
183	2020-07-23	8710969	633506	15510481	6166006	187
184	2020-07-24	8939705	639650	15791645	6212290	187
185	2020-07-25	9158743	644517	16047190	6243930	187
186	2020-07-26	9293464	648621	16251796	6309711	187
187	2020-07-27	9468087	654036	16480485	6358362	187

188 rows × 6 columns

```
#area plot
fig=px.area(dates, x='Date',y='No. of countries',title='Spread of Disease across the countries')
fig.update_layout(margin=dict(t=80,l=0,r=0,b=0))
fig
```



Spread of Disease across the countries

Loading .csv file

```
USA_df=pd.read_csv('usa_county_wise.csv')
USA_df
```

	UID	iso2	iso3	code3	FIPS	Admin2	Province_State	Country_Region	Lat	Long_
0	16	AS	ASM	16	60.0	NaN	American Samoa	US	-14.271000	-170.132000
1	316	GU	GUM	316	66.0	NaN	Guam	US	13.444300	144.793700
2	580	MP	MNP	580	69.0	NaN	Northern Mariana Islands	US	15.097900	145.673900
3	63072001	PR	PRI	630	72001.0	Adjuntas	Puerto Rico	US	18.180117	-66.754367
4	63072003	PR	PRI	630	72003.0	Aguada	Puerto Rico	US	18.360255	-67.175131
...	...	...	...	...	...	...	...	...	...	...
627915	84070016	US	USA	840	NaN	Central Utah	Utah	US	39.372319	-111.575868
627916	84070017	US	USA	840	NaN	Southeast Utah	Utah	US	38.996171	-110.701396
627917	84070018	US	USA	840	NaN	Southwest Utah	Utah	US	37.854472	-111.441876
627918	84070019	US	USA	840	NaN	TriCounty	Utah	US	40.124915	-109.517442
627919	84070020	US	USA	840	NaN	Weber-Morgan	Utah	US	41.271160	-111.914512

627920 rows × 14 columns

```
USA_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 627920 entries, 0 to 627919
Data columns (total 14 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   UID             627920 non-null  int64
 1   iso2            627920 non-null  object
 2   iso3            627920 non-null  object
 3   code3           627920 non-null  int64
 4   FIPS            626040 non-null  float64
 5   Admin2          626792 non-null  object
 6   Province_State  627920 non-null  object
 7   Country_Region  627920 non-null  object
 8   Lat             627920 non-null  float64
 9   Long_           627920 non-null  float64
 10  Combined_Key    627920 non-null  object
 11  Date            627920 non-null  object
 12  Confirmed       627920 non-null  int64
 13  Deaths          627920 non-null  int64
dtypes: float64(3), int64(4), object(7)
memory usage: 67.1+ MB
```

```
#Checking for missing values
USA_df.isnull().sum()
```

```
UID                  0
iso2                 0
iso3                 0
code3                0
FIPS              1880
Admin2            1128
Province_State       0
Country_Region       0
Lat                  0
Long_                0
Combined_Key         0
Date                 0
Confirmed            0
Deaths               0
dtype: int64
```

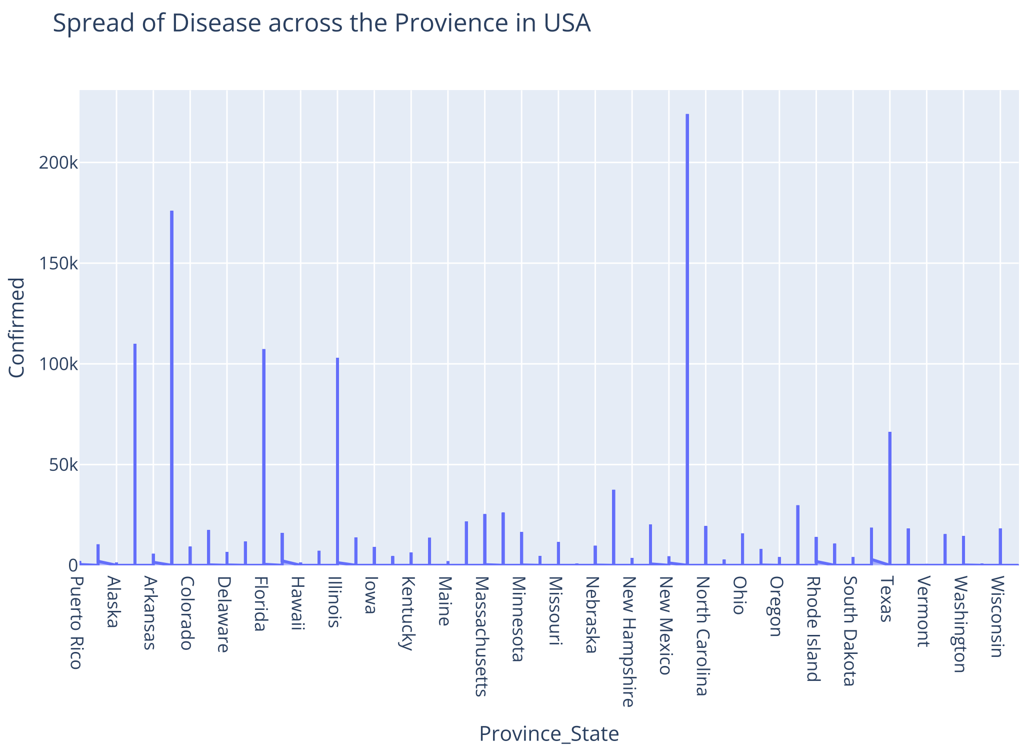
```
#dropping missing values
usa_df=USA_df.dropna()
usa_df
```

	UID	iso2	iso3	code3	FIPS	Admin2	Province_State	Country_Region	Lat	Long_
3	63072001	PR	PRI	630	72001.0	Adjuntas	Puerto Rico	US	18.180117	-66.754367
4	63072003	PR	PRI	630	72003.0	Aguada	Puerto Rico	US	18.360255	-67.175131
5	63072005	PR	PRI	630	72005.0	Aguadilla	Puerto Rico	US	18.459681	-67.120815
6	63072007	PR	PRI	630	72007.0	Aguas Buenas	Puerto Rico	US	18.251619	-66.126806
7	63072009	PR	PRI	630	72009.0	Aibonito	Puerto Rico	US	18.131361	-66.264131
...	...	...	...	...	...	...	...	...	...	...
627904	84090051	US	USA	840	90051.0	Unassigned	Virginia	US	0.000000	0.000000
627905	84090053	US	USA	840	90053.0	Unassigned	Washington	US	0.000000	0.000000
627906	84090054	US	USA	840	90054.0	Unassigned	West Virginia	US	0.000000	0.000000
627907	84090055	US	USA	840	90055.0	Unassigned	Wisconsin	US	0.000000	0.000000
627908	84090056	US	USA	840	90056.0	Unassigned	Wyoming	US	0.000000	0.000000

624912 rows × 14 columns

```
#area plot
fig=px.area(usa_df, y='Confirmed',x='Province_State',title='Spread of Disease across the Provience in USA')
```

```
    fig.update_layout(margin=dict(t=80,l=0,r=0,b=0))
    fig
```



✓ 1s completed at 4:07 PM