MAHATMA EDUCATION SOCIETY’S

PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE

(Autonomous)

NEW PANVEL

CA - II PROJECT ON

“covid -19 Dataset Analysis using Python’’

IN PARTIAL FULFILLMENT OF

BACHELOR OF SCIENCE IN INFORMATION TECHNOLOGY

SEMESTER III – 2023-24

PROJECT GUIDE

Prof. Pradnya Kumavat

SUBMITTED BY

Atharv sawant

Roll No. : 6208

INTRODUCTION

This project aims to harness the power of data visualization using Python to gain valuable insights from COVID-19 datasets. Through visual representations of the data, we can uncover trends, patterns, and anomalies that are not always apparent when examining raw numbers. By doing so, we can contribute to a better understanding of the pandemic's progression, the effectiveness of public health measures, and the impact of vaccination campaigns.

Project Goals:

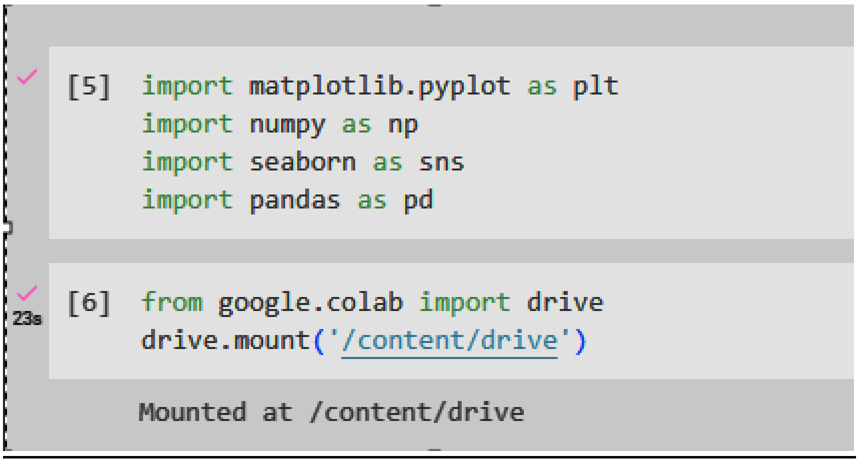
Data Exploration: We will start by exploring various COVID-19 datasets, which may include information on confirmed cases, deaths, recoveries, testing rates, vaccination coverage, and more. We will source these datasets from reputable sources such as the World Health Organization (WHO), the Centers for Disease Control and Prevention (CDC), or other reliable sources.

Data Preprocessing: Data preprocessing is a crucial step in any data analysis project. We will clean and format the data to ensure it is suitable for visualization. This may involve handling missing values, converting data types, and aggregating data by relevant categories (e.g., date, region).

Data Visualization: The heart of this project lies in creating meaningful and insightful visualizations. We will use Python libraries such as Matplotlib, Seaborn, and Plotly to generate various types of plots and charts, including line plots, bar charts, scatterplots, heatmaps, and more. These visualizations will provide clear representations of COVID-19 trends over time and across different variables.

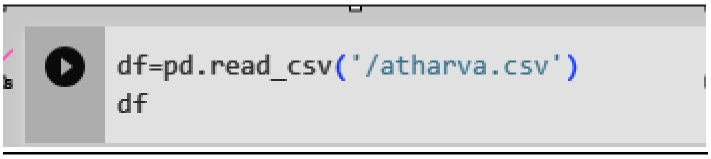
Interactive Dashboards: To enhance the project's interactivity, we may develop interactive dashboards using libraries like Dash or Bokeh. Dashboards enable users to explore the data dynamically, select specific regions or time periods, and gain a deeper understanding of the pandemic's impact.

ACTUAL ANALYSIS

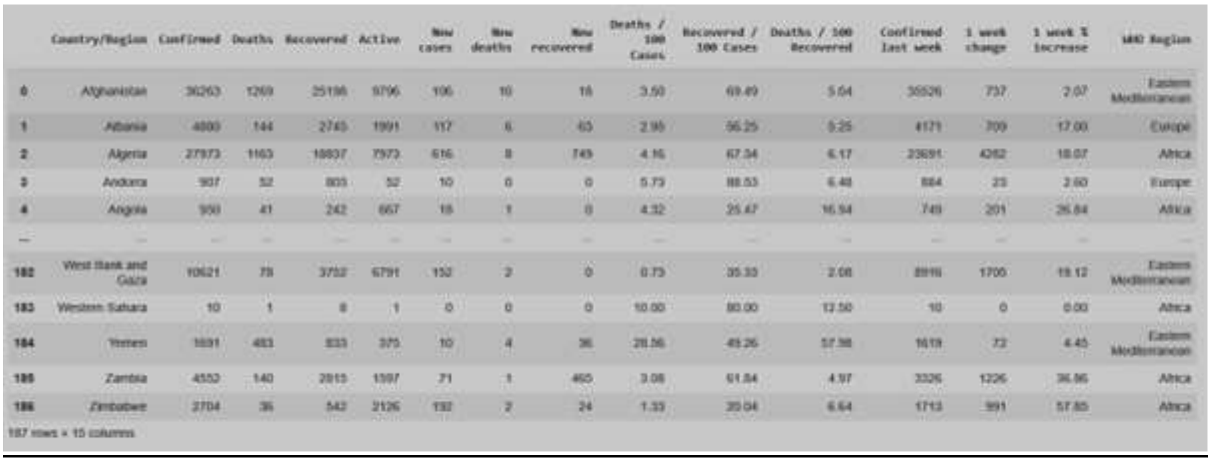


```
[5]  import matplotlib.pyplot as plt
     import numpy as np
     import seaborn as sns
     import pandas as pd
```

```
[6]  from google.colab import drive
     drive.mount('/content/drive')

     Mounted at /content/drive
```

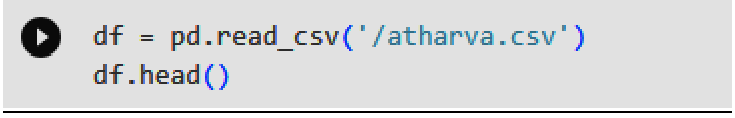


```
df=pd.read_csv('/atharva.csv')
df
```

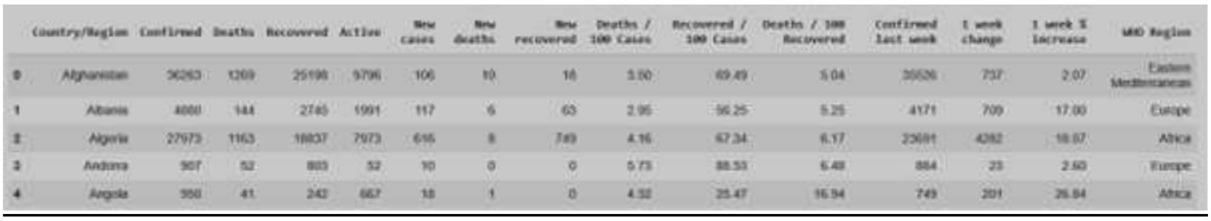


	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Confirmed last week	1 week change	1 week % increase	WHO Region
0	Afghanistan	36263	1269	25198	9796	106	10	18	3.50	69.49	5.04	35526	737	2.07	Eastern Mediterranean
1	Albania	4880	144	2745	1991	117	6	63	2.95	56.25	5.25	4171	709	17.00	Europe
2	Algeria	27973	1163	18837	7973	616	8	749	4.16	67.34	6.17	23691	4282	18.07	Africa
3	Andorra	907	52	803	52	10	0	0	5.73	88.53	6.48	884	23	2.60	Europe
4	Angola	950	41	242	667	18	1	0	4.32	25.47	16.94	749	201	26.84	Africa
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
182	West Bank and Gaza	10621	78	3752	6791	152	2	0	0.73	35.33	2.08	8916	1705	19.12	Eastern Mediterranean
183	Western Sahara	10	1	8	1	0	0	0	10.00	80.00	12.50	10	0	0.00	Africa
184	Yemen	1691	483	833	375	10	4	36	28.56	49.26	57.98	1619	72	4.45	Eastern Mediterranean
185	Zambia	4552	140	2815	1597	71	1	465	3.08	61.84	4.97	3326	1226	36.86	Africa
186	Zimbabwe	2704	36	542	2126	192	2	24	1.33	20.04	6.64	1713	991	57.85	Africa

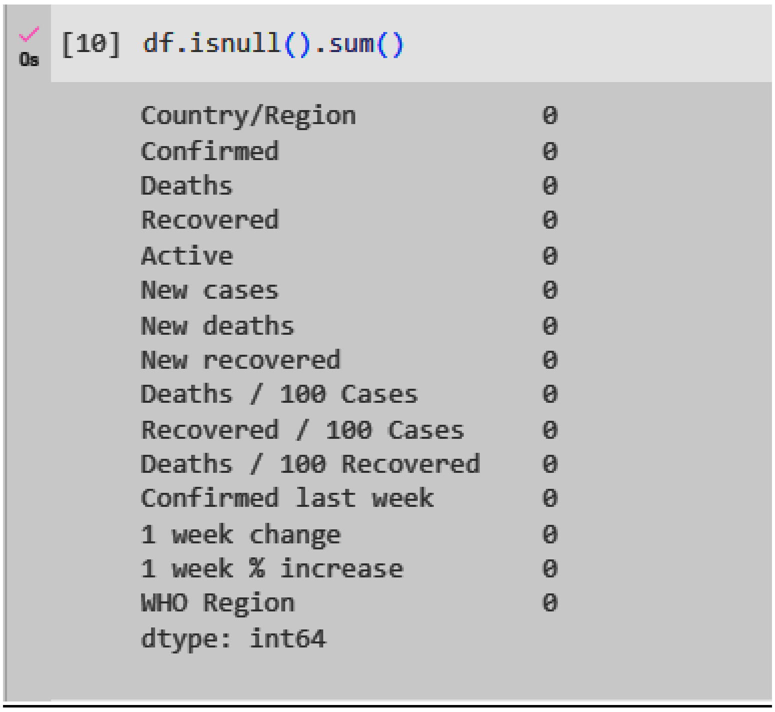
187 rows × 15 columns



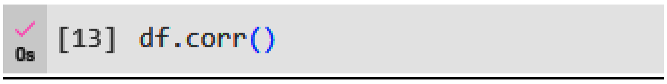
```
df = pd.read_csv('/atharva.csv')
df.head()
```



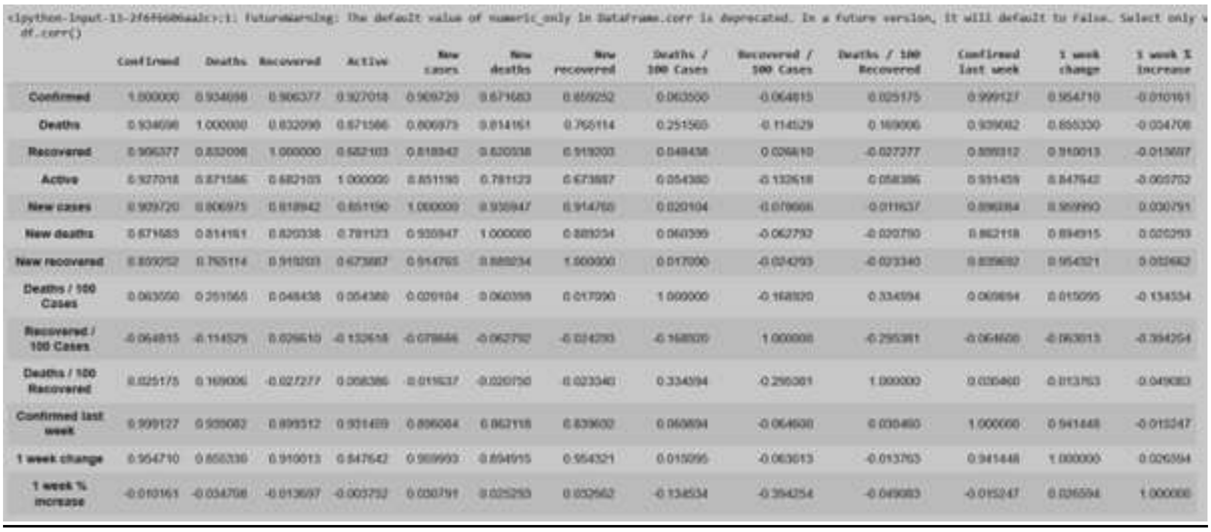
	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Confirmed last week	1 week change	1 week % increase	WHO Region
0	Afghanistan	36263	1269	25198	9796	106	10	18	3.50	69.49	5.04	35526	737	2.07	Eastern Mediterranean
1	Albania	4880	144	2745	1991	117	6	63	2.95	56.25	5.25	4171	709	17.00	Europe
2	Algeria	27973	1163	18837	7973	616	8	749	4.16	67.34	6.17	23691	4282	18.07	Africa
3	Andorra	907	52	803	52	10	0	0	5.73	88.53	6.48	884	23	2.60	Europe
4	Angola	950	41	242	667	18	1	0	4.32	25.47	16.94	749	201	26.84	Africa



```
0s
[10] 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
```

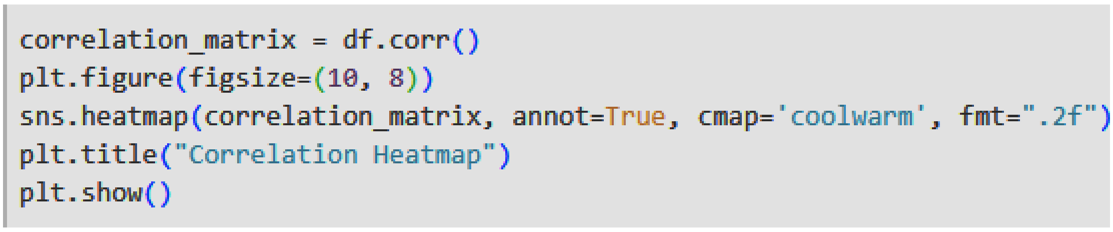


```
0s
[13] df.corr()
```

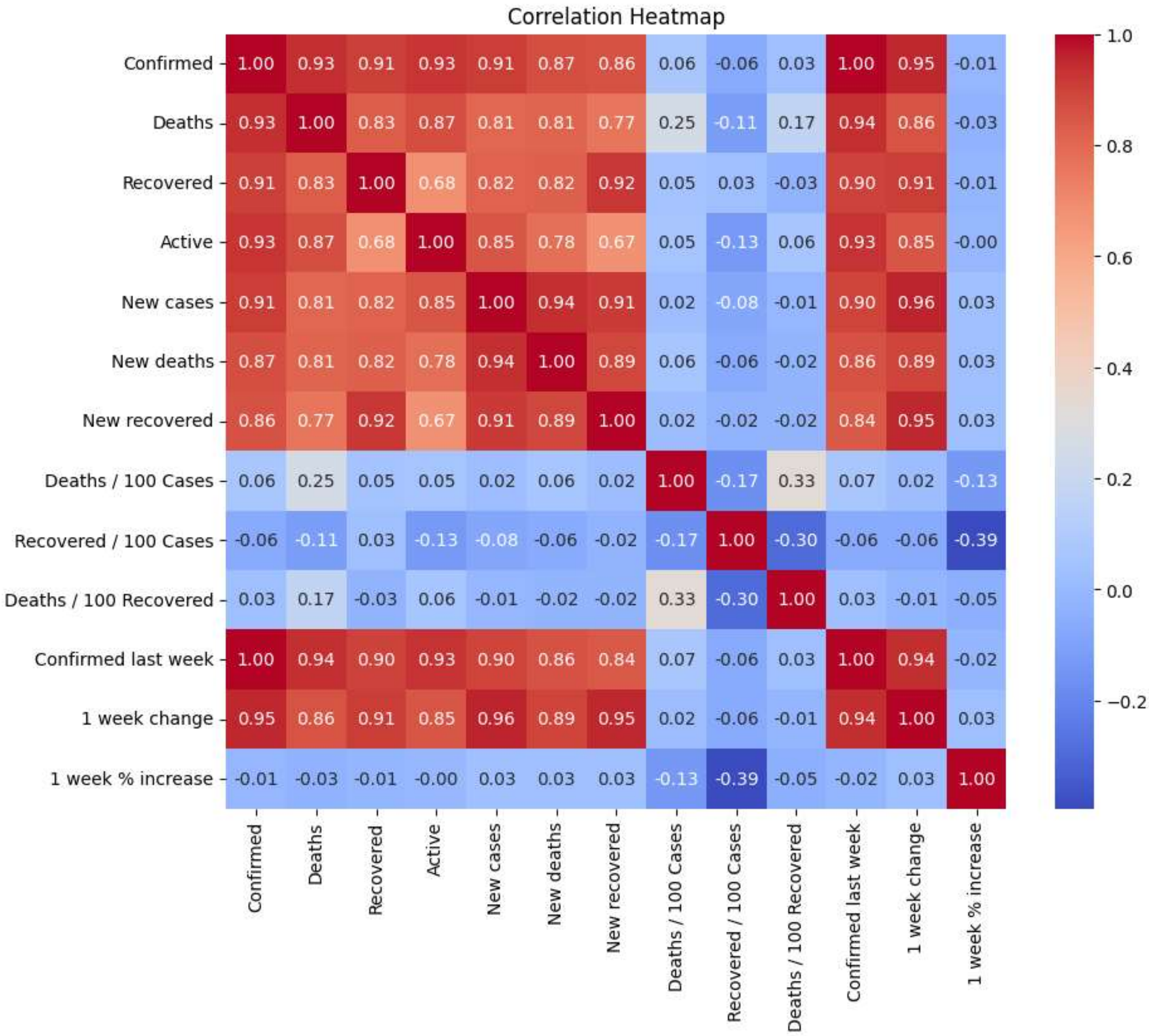


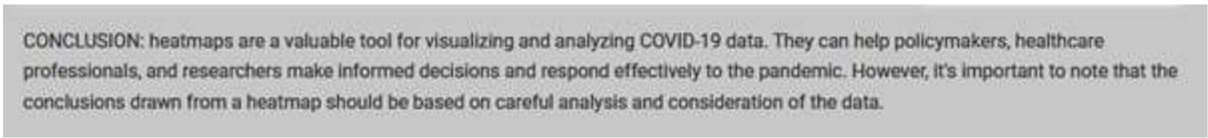
```
<ipython-input-13-2f6f6d0aa1c>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only v
  df.corr()
```

	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Confirmed last week	1 week change	1 week % increase
Confirmed	1.000000	0.934098	0.906377	0.927018	0.909720	0.871683	0.859252	0.063550	-0.064815	0.025175	0.999127	0.954710	-0.010161
Deaths	0.934098	1.000000	0.832098	0.871586	0.806975	0.814161	0.765114	0.251565	-0.114529	0.169006	0.939082	0.855330	-0.034708
Recovered	0.906377	0.832098	1.000000	0.682103	0.818942	0.820338	0.919203	0.048438	0.026610	-0.027277	0.899312	0.910013	-0.013697
Active	0.927018	0.871586	0.682103	1.000000	0.851190	0.781123	0.673887	0.054380	-0.132618	0.058386	0.931459	0.847642	-0.003752
New cases	0.909720	0.806975	0.818942	0.851190	1.000000	0.935947	0.914765	0.020104	-0.078666	-0.011637	0.896084	0.959990	0.030791
New deaths	0.871683	0.814161	0.820338	0.781123	0.935947	1.000000	0.889234	0.060399	-0.062792	-0.020750	0.862118	0.894915	0.025293
New recovered	0.859252	0.765114	0.919203	0.673887	0.914765	0.889234	1.000000	0.017090	-0.024293	-0.023340	0.839602	0.954321	0.032662
Deaths / 100 Cases	0.063550	0.251565	0.048438	0.054380	0.020104	0.060399	0.017090	1.000000	-0.168920	0.334594	0.060894	0.015095	-0.134534
Recovered / 100 Cases	-0.064815	-0.114529	0.026610	-0.132618	-0.078666	-0.062792	-0.024293	-0.168920	1.000000	-0.295381	-0.064600	-0.063013	-0.394254
Deaths / 100 Recovered	0.025175	0.169006	-0.027277	0.058386	-0.011637	-0.020750	-0.023340	0.334594	-0.295381	1.000000	0.030460	-0.013763	-0.049083
Confirmed last week	0.999127	0.939082	0.899312	0.931459	0.896084	0.862118	0.839602	0.060894	-0.064600	0.030460	1.000000	0.941448	-0.015247
1 week change	0.954710	0.855330	0.910013	0.847642	0.959990	0.894915	0.954321	0.015095	-0.063013	-0.013763	0.941448	1.000000	0.026594
1 week % increase	-0.010161	-0.034708	-0.013697	-0.003752	0.030791	0.025293	0.032662	-0.134534	-0.394254	-0.049083	-0.015247	0.026594	1.000000

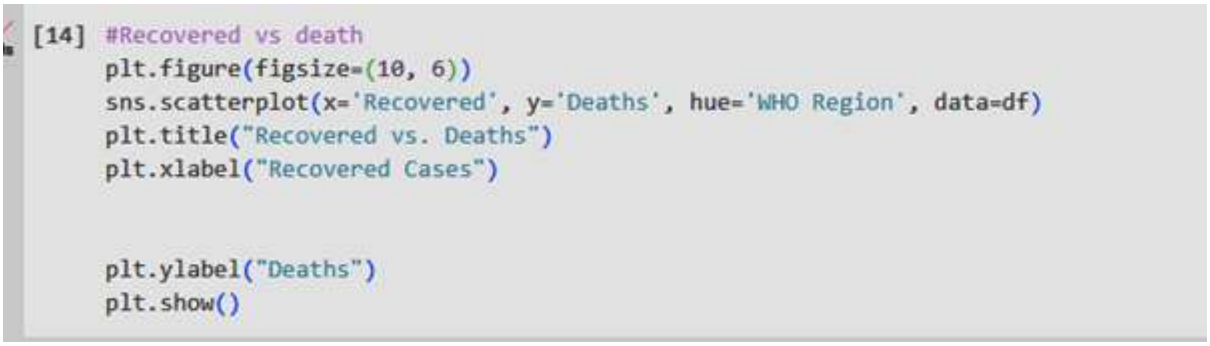


```
correlation_matrix = df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```

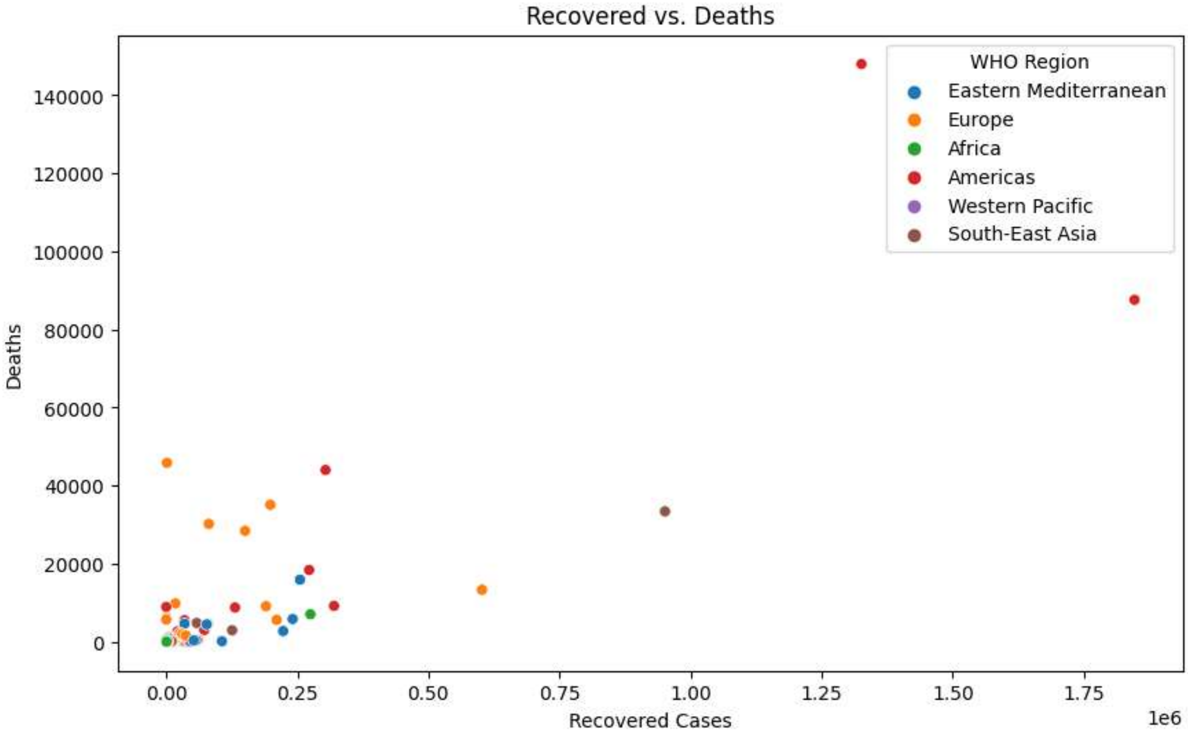


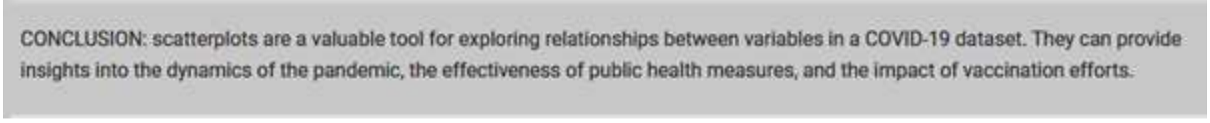


CONCLUSION: heatmaps are a valuable tool for visualizing and analyzing COVID-19 data. They can help policymakers, healthcare professionals, and researchers make informed decisions and respond effectively to the pandemic. However, it's important to note that the conclusions drawn from a heatmap should be based on careful analysis and consideration of the data.

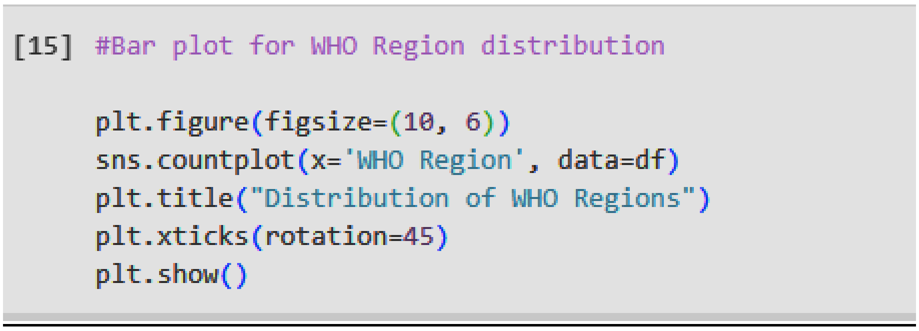


```
[14] #Recovered vs death
     plt.figure(figsize=(10, 6))
     sns.scatterplot(x='Recovered', y='Deaths', hue='WHO Region', data=df)
     plt.title("Recovered vs. Deaths")
     plt.xlabel("Recovered Cases")


     plt.ylabel("Deaths")
     plt.show()
```

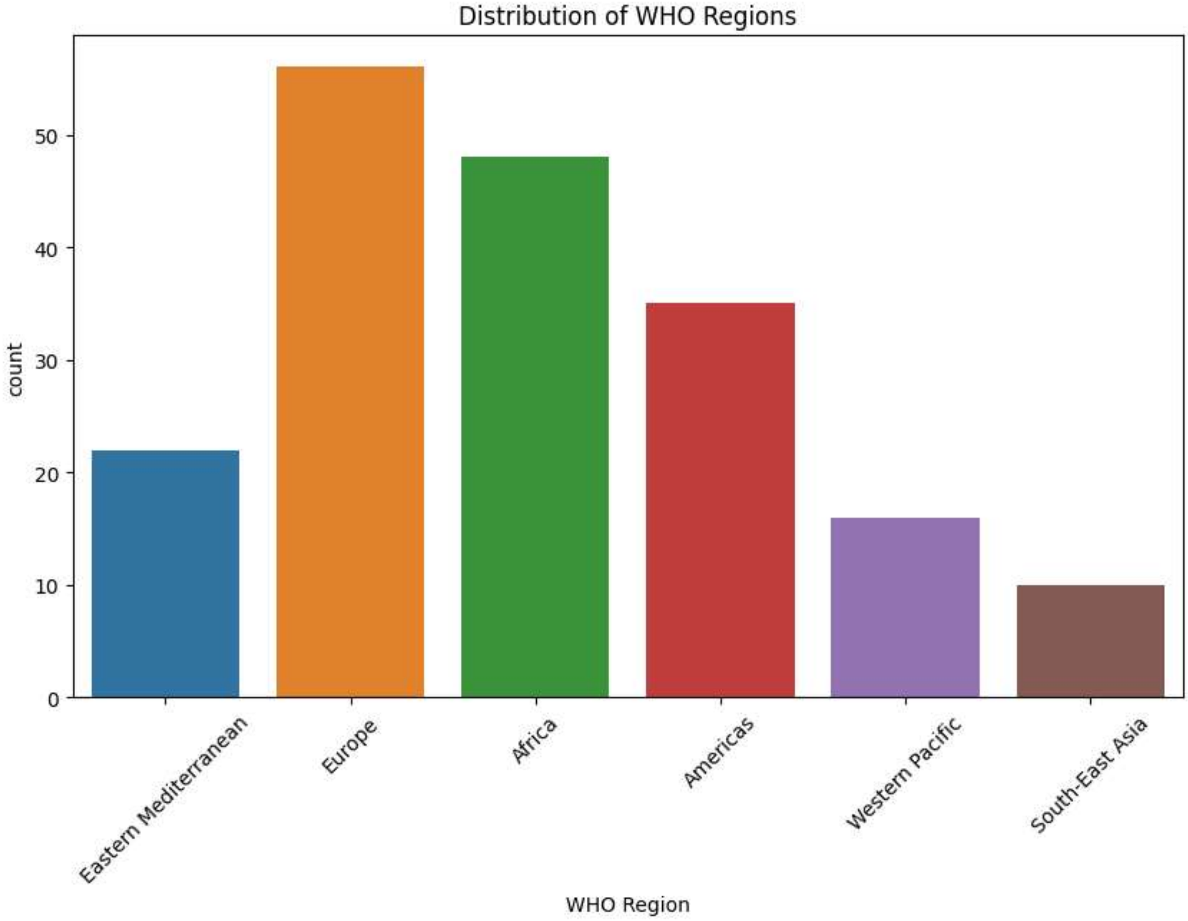


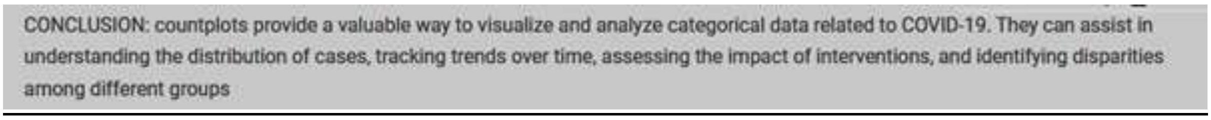


CONCLUSION: scatterplots are a valuable tool for exploring relationships between variables in a COVID-19 dataset. They can provide insights into the dynamics of the pandemic, the effectiveness of public health measures, and the impact of vaccination efforts.

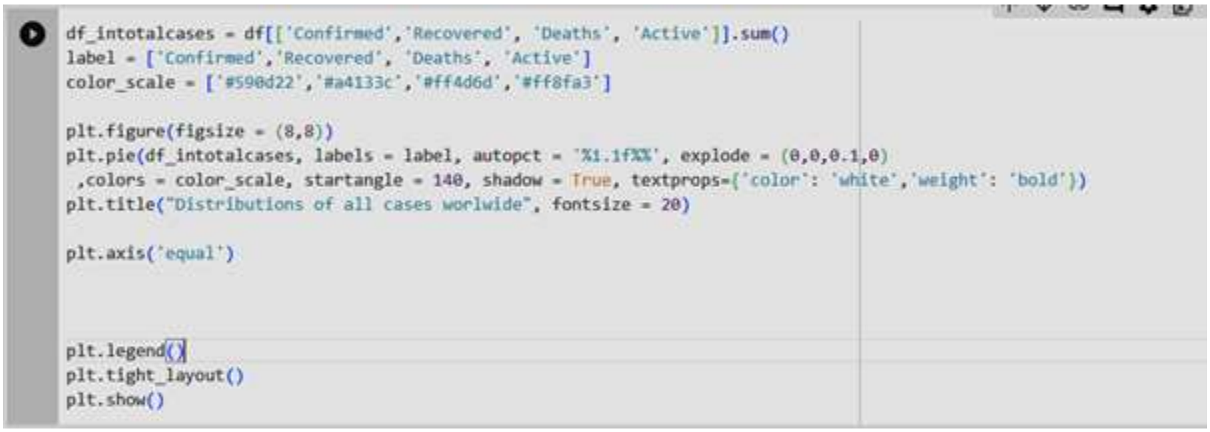


```
[15] #Bar plot for WHO Region distribution

     plt.figure(figsize=(10, 6))
     sns.countplot(x='WHO Region', data=df)
     plt.title("Distribution of WHO Regions")
     plt.xticks(rotation=45)
     plt.show()
```

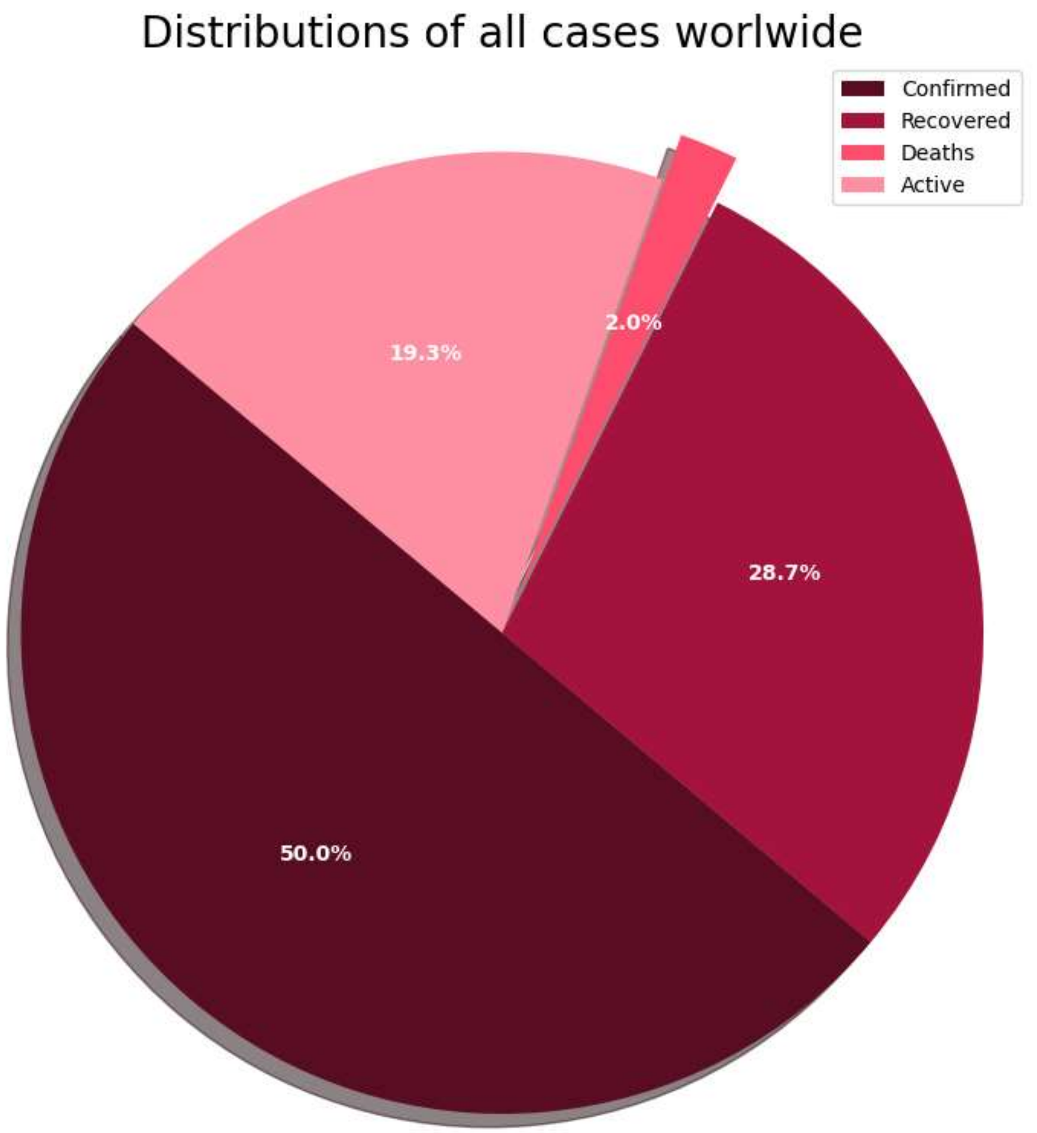




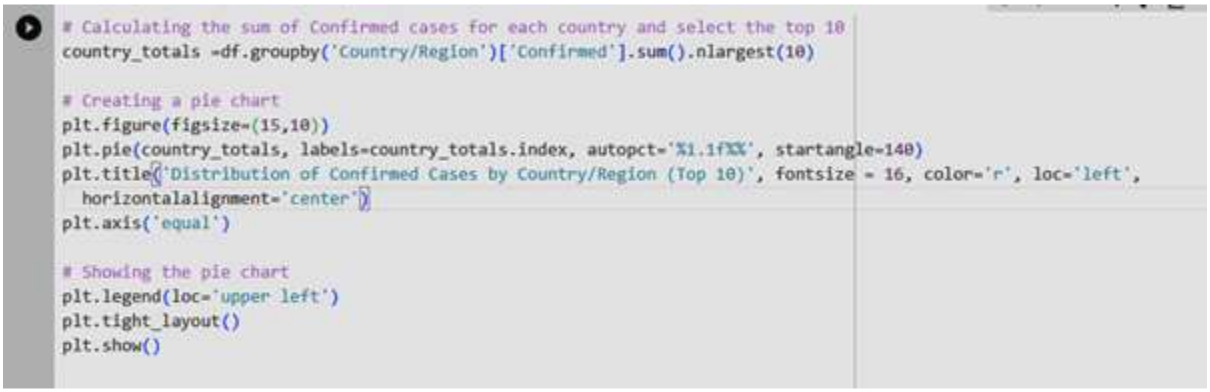
CONCLUSION: countplots provide a valuable way to visualize and analyze categorical data related to COVID-19. They can assist in understanding the distribution of cases, tracking trends over time, assessing the impact of interventions, and identifying disparities among different groups



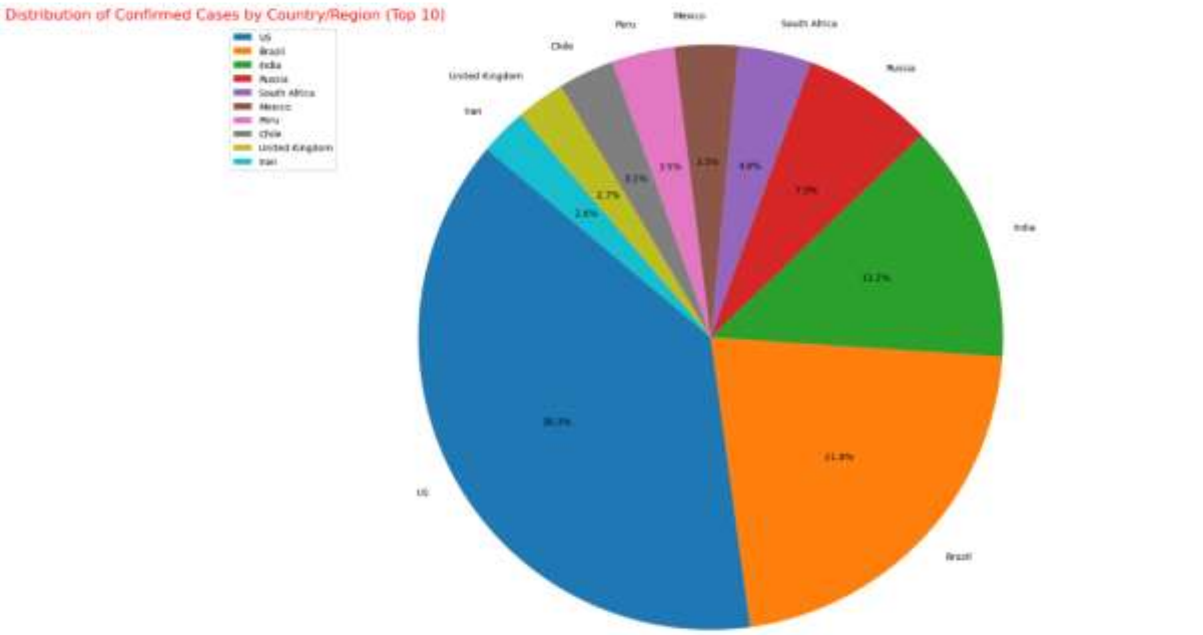
```
df_intotalcases = df[['Confirmed','Recovered', 'Deaths', 'Active']].sum()
label = ['Confirmed','Recovered', 'Deaths', 'Active']
color_scale = ['#590d22','#a4133c','#ff4d6d','#ff8fa3']

plt.figure(figsize = (8,8))
plt.pie(df_intotalcases, labels = label, autopct = '%1.1f%%', explode = (0,0,0.1,0)
 ,colors = color_scale, startangle = 140, shadow = True, textprops={'color': 'white','weight': 'bold'})
plt.title("Distributions of all cases worlwide", fontsize = 20)

plt.axis('equal')



plt.legend()
plt.tight_layout()
plt.show()
```

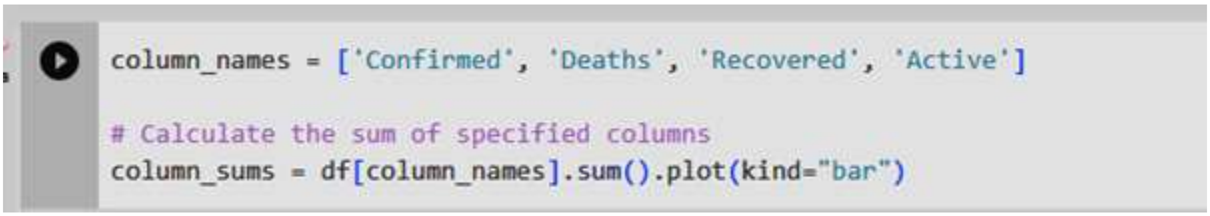


CONCLUSION: pie charts are a useful tool for visually representing the distribution of COVID-19 data across different categories or segments. They can provide quick insights into how cases or other relevant metrics are divided among various groups. It's important to use pie charts judiciously, as they are best suited for representing data with a limited number of categories, and the data should be well-suited for this type of visualization.

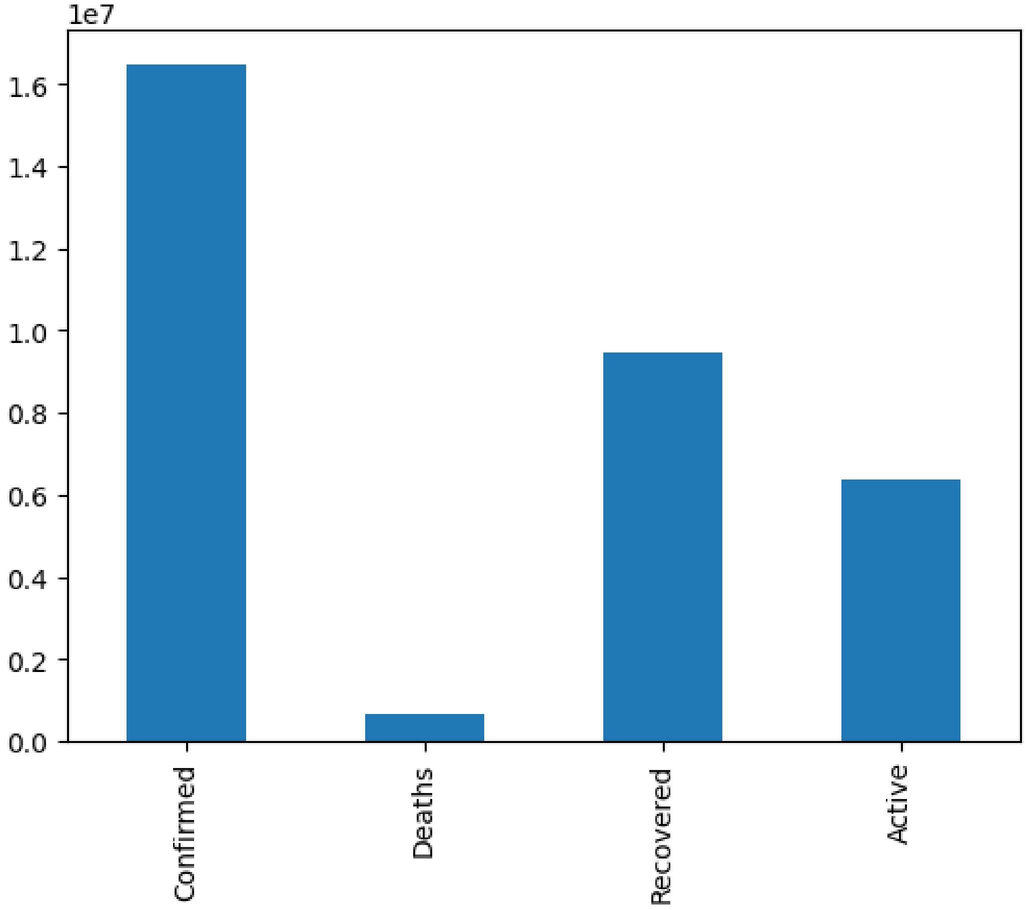


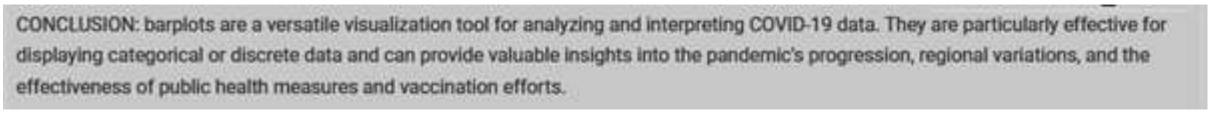
```
# Calculating the sum of Confirmed cases for each country and select the top 10
country_totals =df.groupby('Country/Region')['Confirmed'].sum().nlargest(10)

# Creating a pie chart
plt.figure(figsize=(15,10))
plt.pie(country_totals, labels=country_totals.index, autopct='%1.1f%%', startangle=140)
plt.title('Distribution of Confirmed Cases by Country/Region (Top 10)', fontsize = 16, color='r', loc='left',
  horizontalalignment='center')
plt.axis('equal')

# Showing the pie chart
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```



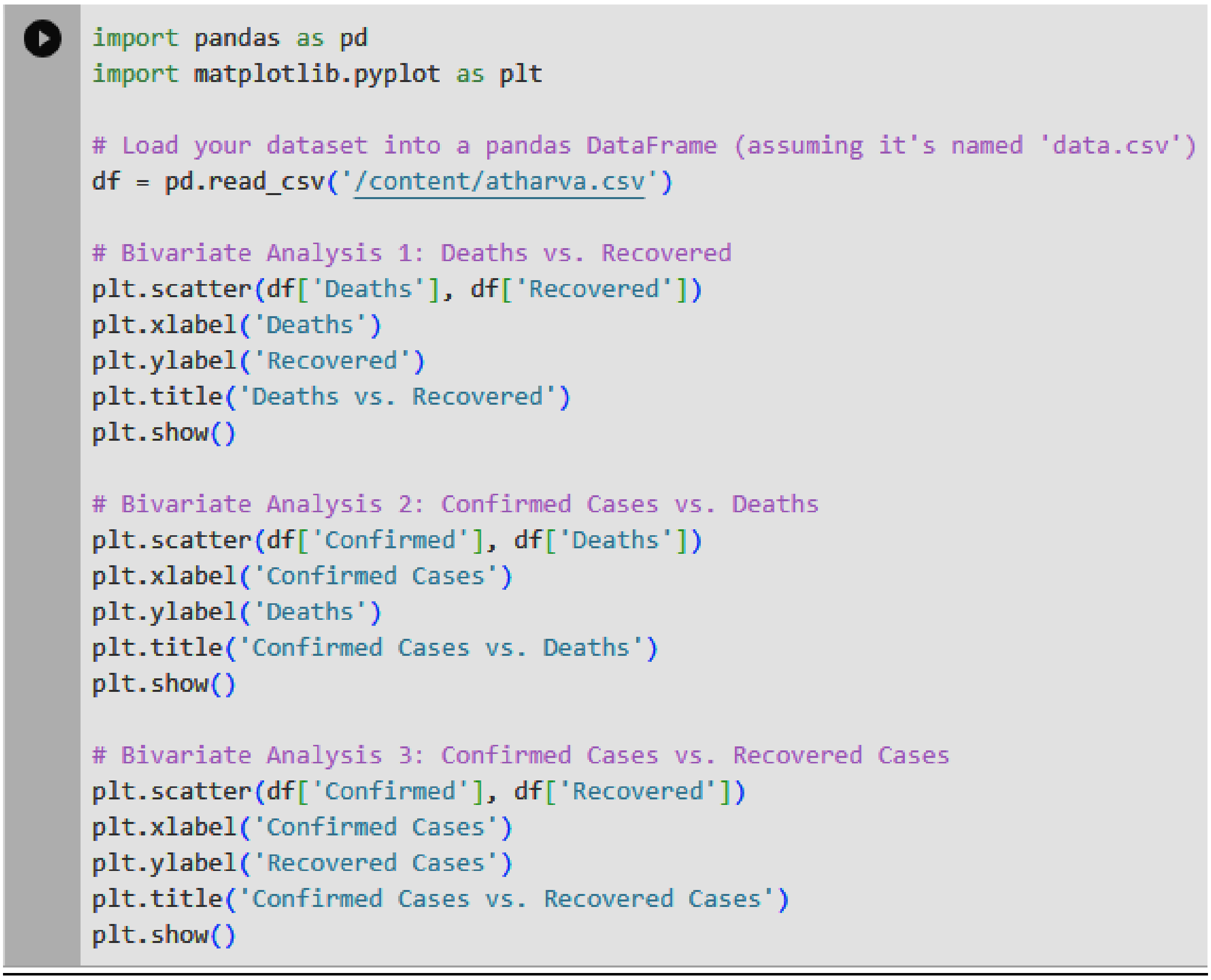


```
column_names = ['Confirmed', 'Deaths', 'Recovered', 'Active']

# Calculate the sum of specified columns
column_sums = df[column_names].sum().plot(kind="bar")
```

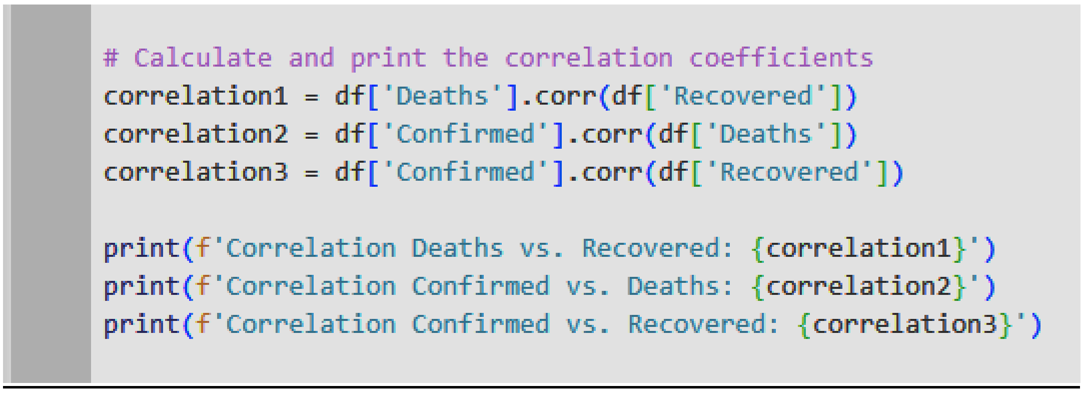




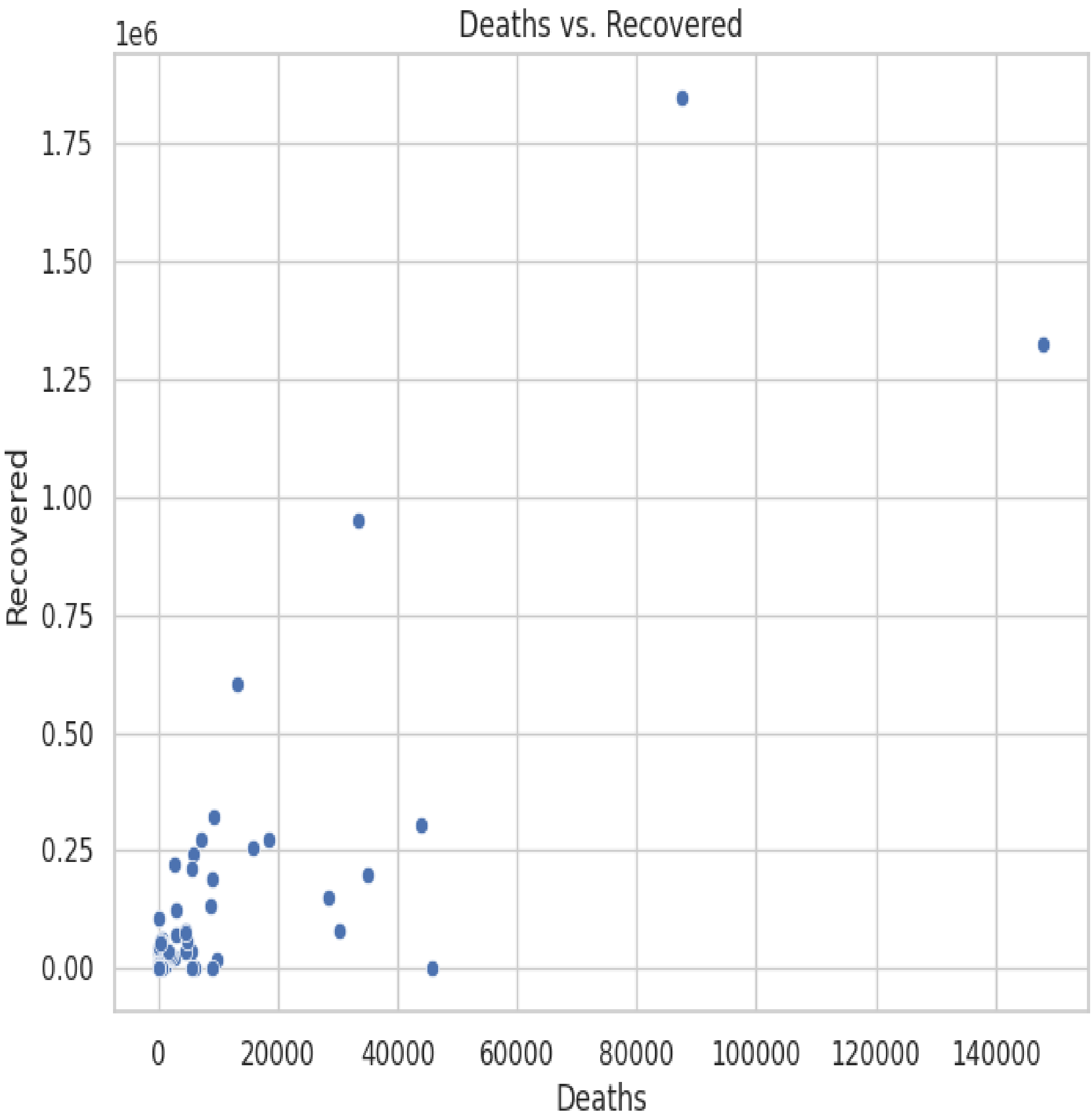
CONCLUSION: barplots are a versatile visualization tool for analyzing and interpreting COVID-19 data. They are particularly effective for displaying categorical or discrete data and can provide valuable insights into the pandemic's progression, regional variations, and the effectiveness of public health measures and vaccination efforts.

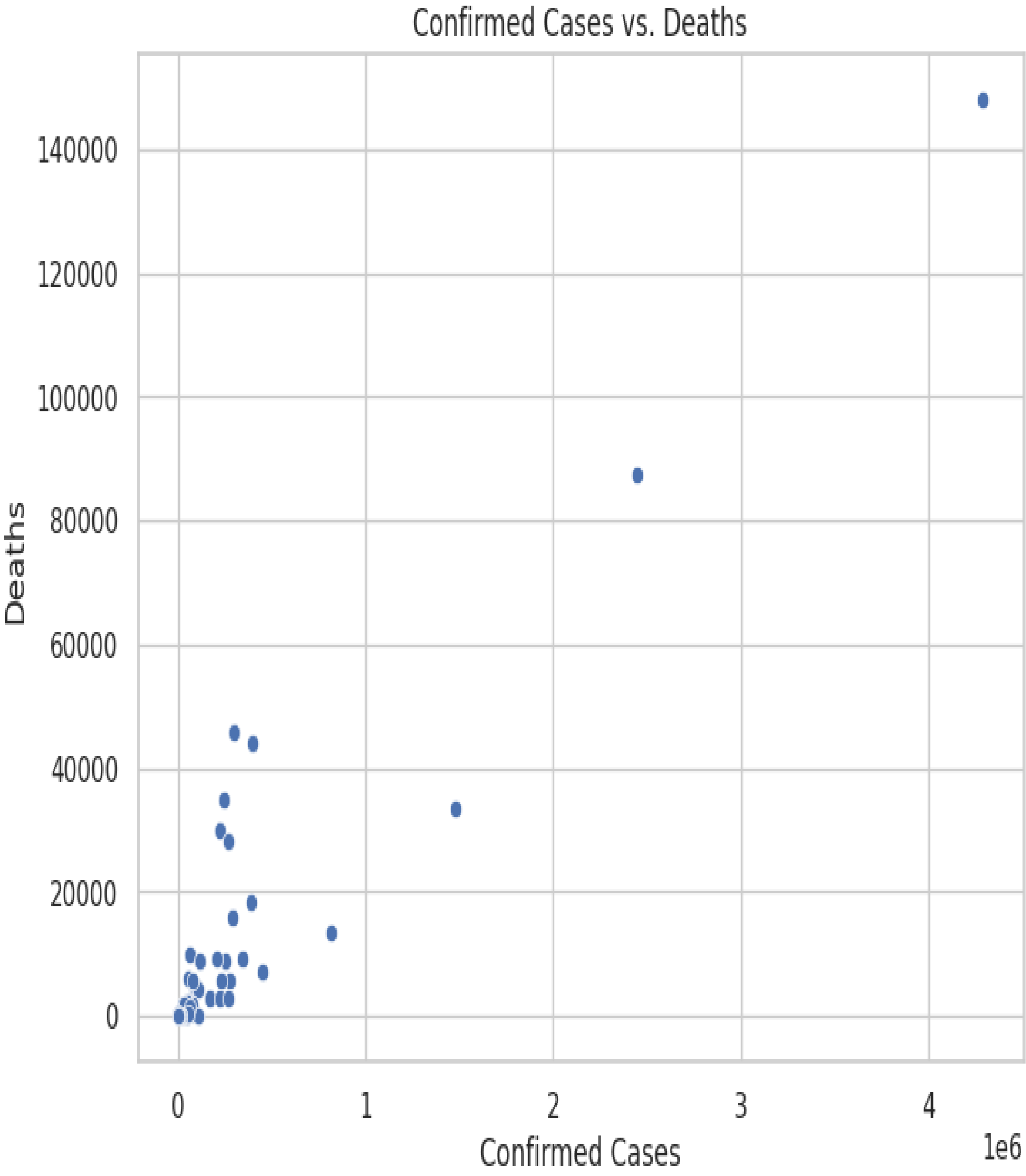


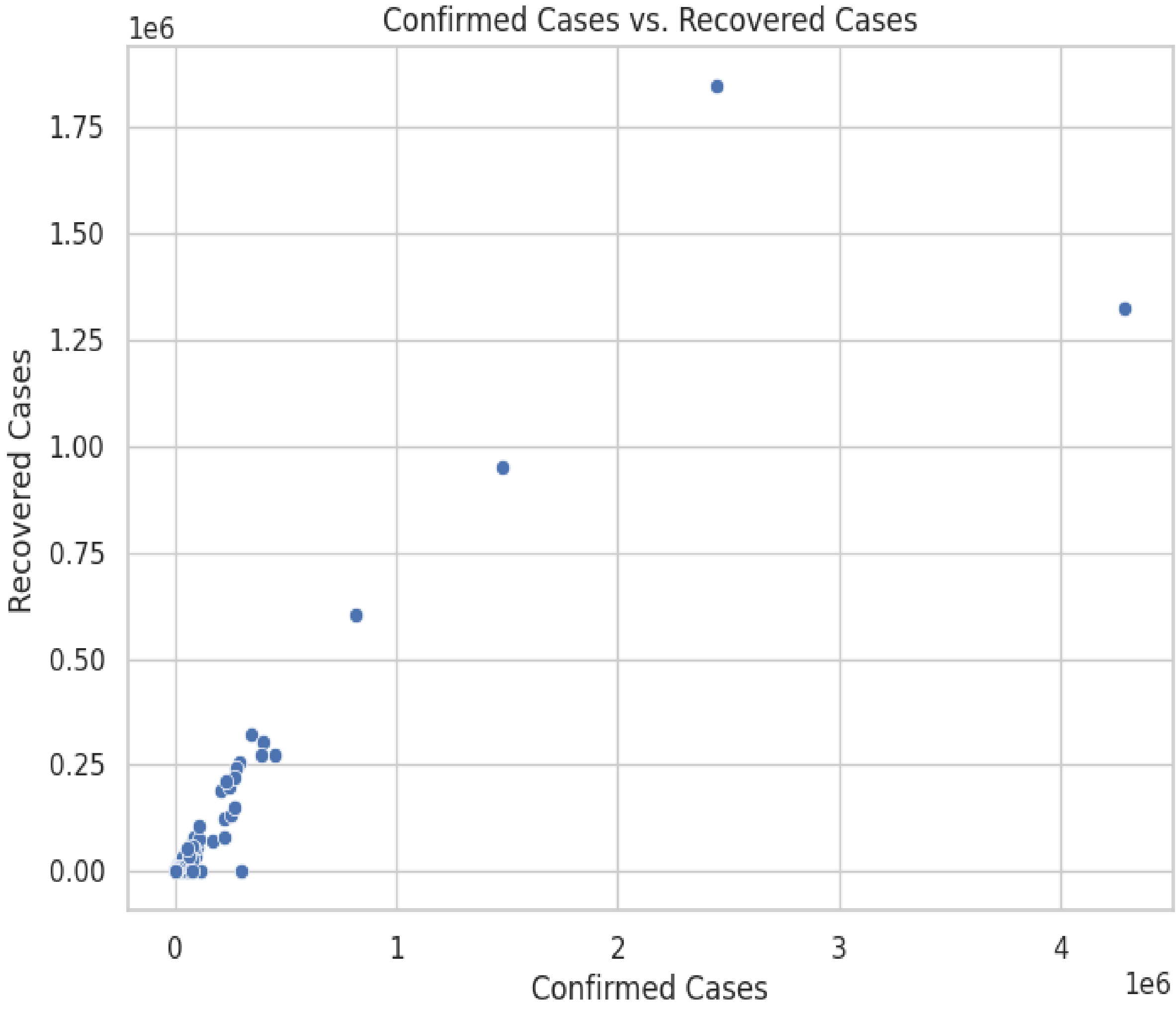
```python
import pandas as pd
import matplotlib.pyplot as plt

# Load your dataset into a pandas DataFrame (assuming it's named 'data.csv')
df = pd.read_csv('/content/atharva.csv')

# Bivariate Analysis 1: Deaths vs. Recovered
plt.scatter(df['Deaths'], df['Recovered'])
plt.xlabel('Deaths')
plt.ylabel('Recovered')
plt.title('Deaths vs. Recovered')
plt.show()

# Bivariate Analysis 2: Confirmed Cases vs. Deaths
plt.scatter(df['Confirmed'], df['Deaths'])
plt.xlabel('Confirmed Cases')
plt.ylabel('Deaths')
plt.title('Confirmed Cases vs. Deaths')
plt.show()

# Bivariate Analysis 3: Confirmed Cases vs. Recovered Cases
plt.scatter(df['Confirmed'], df['Recovered'])
plt.xlabel('Confirmed Cases')
plt.ylabel('Recovered Cases')
plt.title('Confirmed Cases vs. Recovered Cases')
plt.show()
```

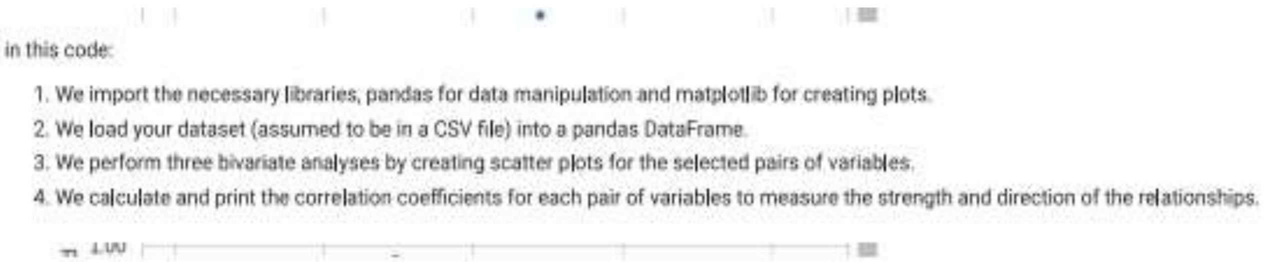


```python
# Calculate and print the correlation coefficients
correlation1 = df['Deaths'].corr(df['Recovered'])
correlation2 = df['Confirmed'].corr(df['Deaths'])
correlation3 = df['Confirmed'].corr(df['Recovered'])

print(f'Correlation Deaths vs. Recovered: {correlation1}')
print(f'Correlation Confirmed vs. Deaths: {correlation2}')
print(f'Correlation Confirmed vs. Recovered: {correlation3}')
```

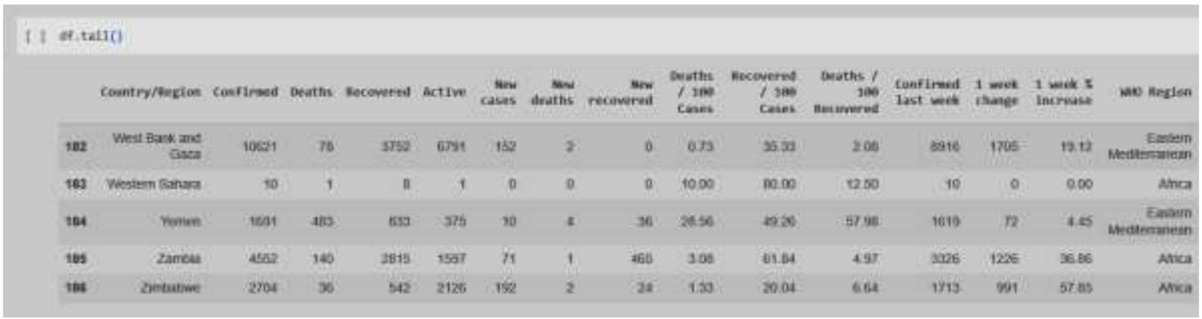

Deaths vs. Recovered
1e6
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Recovered
0
20000
40000
60000
80000
100000
120000
140000
Deaths


Confirmed Cases vs. Deaths
Deaths
140000
120000
100000
80000
60000
40000
20000
0
0
1
2
3
4
1e6
Confirmed Cases



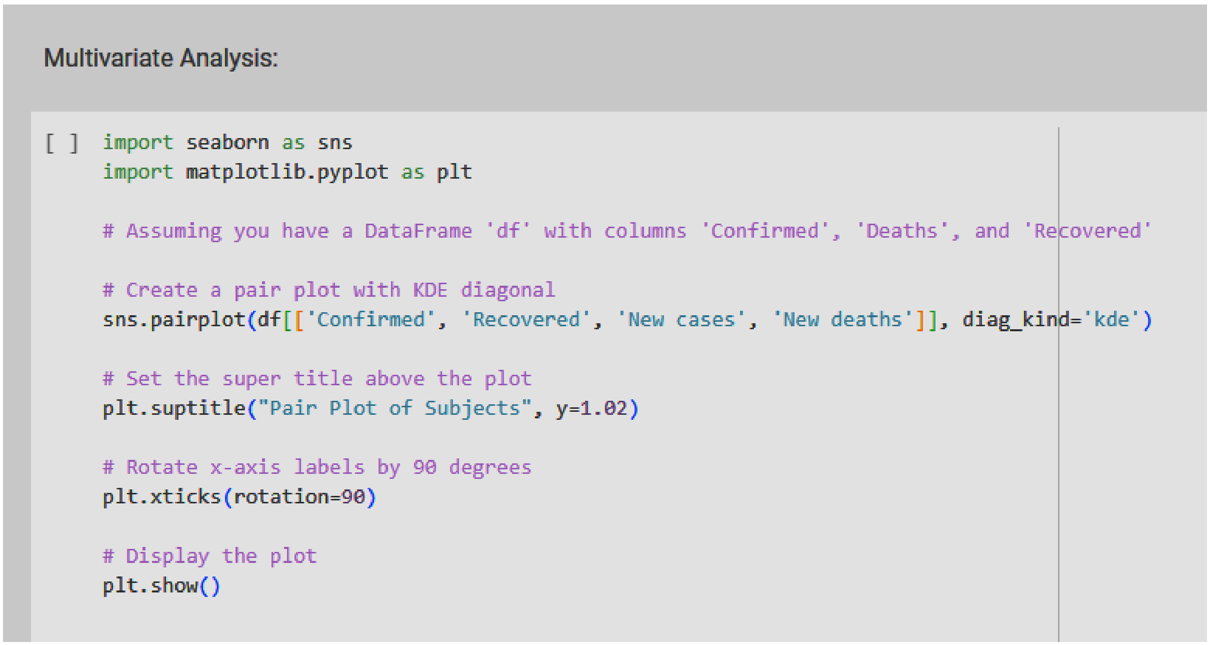

In this code:

1. We import the necessary libraries, pandas for data manipulation and matplotlib for creating plots.
2. We load your dataset (assumed to be in a CSV file) into a pandas DataFrame.
3. We perform three bivariate analyses by creating scatter plots for the selected pairs of variables.
4. We calculate and print the correlation coefficients for each pair of variables to measure the strength and direction of the relationships.



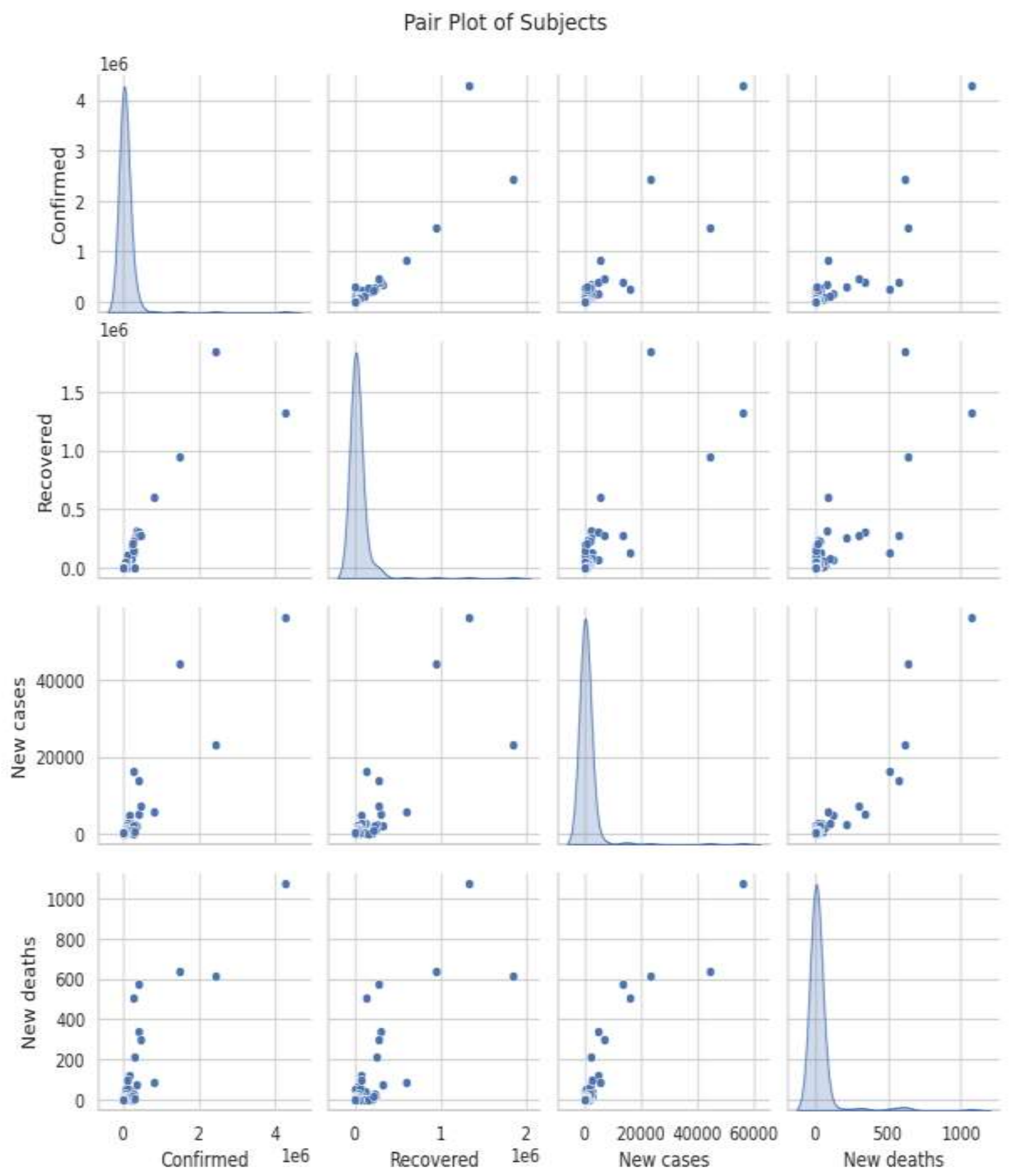
```
[ ] df.tail()
```

	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Confirmed last week	1 week change	1 week % increase	WHO Region
182	West Bank and Gaza	10621	78	3752	6791	152	2	0	0.73	35.33	2.08	8916	1705	19.12	Eastern Mediterranean
183	Western Sahara	10	1	8	1	0	0	0	10.00	80.00	12.50	10	0	0.00	Africa
184	Yemen	1691	483	833	375	10	4	36	28.56	49.26	57.98	1619	72	4.45	Eastern Mediterranean
185	Zambia	4552	140	2815	1597	71	1	465	3.08	61.84	4.97	3326	1226	36.86	Africa
186	Zimbabwe	2704	36	542	2126	192	2	24	1.33	20.04	6.64	1713	991	57.85	Africa



Multivariate Analysis:

```
[ ] import seaborn as sns
    import matplotlib.pyplot as plt

    # Assuming you have a DataFrame 'df' with columns 'Confirmed', 'Deaths', and 'Recovered'

    # Create a pair plot with KDE diagonal
    sns.pairplot(df[['Confirmed', 'Recovered', 'New cases', 'New deaths']], diag_kind='kde')

    # Set the super title above the plot
    plt.suptitle("Pair Plot of Subjects", y=1.02)

    # Rotate x-axis labels by 90 degrees
    plt.xticks(rotation=90)

    # Display the plot
    plt.show()
```



Pair Plot of Subjects

In this code:

We import the necessary libraries, pandas for data manipulation and matplotlib for creating plots. We load your dataset (assumed to be in a CSV file) into a pandas DataFrame. We perform three multivarient analyses by creating scatter plots for the selected pairs of variables.

CONCLUSION:-

This project set out to harness the power of Python and data visualization libraries to gain deeper insights into the COVID-19 pandemic through the analysis of relevant datasets. As we conclude this project, we can reflect on the key takeaways and contributions made in our journey of visualizing COVID-19 data.

Data Exploration and Understanding:

The initial phase of this project involved the exploration and selection of COVID-19 datasets from authoritative sources.. This foundational step allowed us to delve into the vast realm of COVID-19 data with confidence.

Data Preprocessing and Cleaning:

Data preprocessing was a critical aspect of our project. It involved addressing missing values, handling data types, and aggregating data by relevant attributes such as date, region, and demographic information. These efforts were essential to ensure that the data was in a suitable format for visualization.

Data Visualization:

The heart of our project revolved around data visualization. These visualizations were instrumental in revealing the pandemic's trends, disparities, and dynamics. They enabled us to present complex data in a comprehensible and engaging manner.

Insights and Discoveries:

Through our visualizations, we were able to draw several important conclusions and insights from the COVID-19 data. We tracked the progression of the pandemic over time, identifying waves of infections and assessing the impact of public health measures.

Documentation and Reporting:

Transparency and reproducibility are paramount in data analysis. We provided clear and comprehensive documentation of our code, methodologies, and data sources.

Future Directions:

As we conclude this project, it is essential to consider its potential future directions. COVID-19 remains a dynamic situation, and data collection continues. This project can serve as a foundation for ongoing analysis and monitoring of the pandemic. Future work may include the integration of machine learning models for predictive modeling, incorporation of additional variables and data sources, and ongoing updates to the interactive dashboards.

Github link:-

https://github.com/Atharv21sawant/data_visualization/blob/3252ac05c8da271fb83119387cef10d0411d3026/atharvaproject.ipynb