world-population-analysis

October 25, 2024

1 World Population Analysis

The global population reached 7.577 billion in 2019 and continues to grow, albeit at a slower rate. China and India are the two most populous nations, with India expected to surpass China by 2030 due to India's ongoing growth and China's projected decline. Eleven countries, including the U.S., Indonesia, and Brazil, have populations over 100 million, although Russia and Japan face population declines by 2050. Despite a declining growth rate, the world population could surpass 10 billion by mid-century. Countries like India, Nigeria, and several in Africa will significantly impact future population growth.

2 About Dataset

- In 2019, the global population reached 7.577 billion, continuing to grow but at a slower rate.
- China and India are the two most populous countries, each with populations exceeding 1 billion.
- By 2030, India is projected to surpass China as the world's most populous nation due to India's growth and China's projected decline.
- Eleven countries have populations over 100 million, including the U.S., Indonesia, Brazil, and Pakistan.
- Russia and Japan are expected to see population declines by 2030, with further reductions by 2050.
- Despite falling growth rates, the global population is expected to surpass 8 billion by 2030.
- The population may reach over 9 billion by 2040 and over 10 billion by 2055.
- Annual growth currently adds more than 80 million people worldwide.
- Nine countries, including India, Nigeria, and several African nations, will drive much of this growth.
- Several African nations are anticipated to double their populations before fertility rates decrease.

Year	World Population (Billions)	Country	Population (Millions/Billions)	Expected Growth (Yes/No)	Population Peak Year (If any)	Growth Rate (%)
2015	7.2	World	7200	Yes	N/A	1.12
2019	7.577	World	7577	Yes	N/A	1.12
2018	-	China	1400	No	2030	-
2018	-	India	1355	Yes	N/A	-
2030	8.0	World	8000	Yes	N/A	-
2040	9.0	World	9000	Yes	N/A	-

Year	World Population (Billions)	Country	Population (Millions/Billions)	Expected Growth (Yes/No)	Population Peak Year (If any)	Growth Rate (%)
2055	10.0	World	10000	Yes	N/A	-
2050	-	Russia	<100	No	-	-
2050	-	Japan	<100	No	-	-
2030	-	India	> China	Yes	-	-
-	-	Vatican City	0.000801	-	-	-

3 Importing Libraries

[39]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.subplots as sp
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import warnings
```

4 Suppress FutureWarning messages

[3]:

```
warnings.simplefilter(action='ignore', category=FutureWarning)
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
init_notebook_mode(connected=True)
```

5 Graph

[4]:

```
df = pd.read_csv(r"D:\Data Analysis\unfied mentor internship\world_population.
 ↪csv")
```

[ ]:

```
df.head()
```

[ ]:

```
   Rank CCA3 Country/Territory           Capital Continent  2022 Population  \
0    36  AFG       Afghanistan             Kabul      Asia         41128771
1   138  ALB           Albania            Tirana    Europe          2842321
2    34  DZA           Algeria           Algiers    Africa         44903225
3   213  ASM    American Samoa         Pago Pago   Oceania            44273
4   203  AND           Andorra  Andorra la Vella    Europe            79824

   2020 Population  2015 Population  2010 Population  2000 Population  \
0         38972230         33753499         28189672         19542982
1          2866849          2882481          2913399          3182021
2         43451666         39543154         35856344         30774621
3            46189            51368            54849            58230
4            77700            71746            71519            66097

   1990 Population  1980 Population  1970 Population  Area (km²)  \
0         10694796         12486631         10752971      652230
1          3295066          2941651          2324731       28748
2         25518074         18739378         13795915     2381741
3            47818            32886            27075         199
4            53569            35611            19860         468

   Density (per km²)  Growth Rate  World Population Percentage
0            63.0587       1.0257                         0.52
1            98.8702       0.9957                         0.04
2            18.8531       1.0164                         0.56
3           222.4774       0.9831                         0.00
4           170.5641       1.0100                         0.00
```

[7]:
```
df.shape
```

[7]:
```
(234, 17)
```

[5]:
```
df.isnull().sum()
```

[5]:
```
Rank                           0
CCA3                           0
Country/Territory              0
Capital                        0
Continent                      0
2022 Population                0
2020 Population                0
2015 Population                0
2010 Population                0
2000 Population                0
1990 Population                0
1980 Population                0
1970 Population                0
Area (km²)                     0
Density (per km²)              0
Growth Rate                    0
World Population Percentage    0
dtype: int64
```

[8]:
```
print(f"Amount of duplicates: {df.duplicated().sum()}")
```

```
Amount of duplicates: 0
```

[9]:

```python
df.columns
```

[9]:

```
Index(['Rank', 'CCA3', 'Country/Territory', 'Capital', 'Continent',
       '2022 Population', '2020 Population', '2015 Population',
       '2010 Population', '2000 Population', '1990 Population',
       '1980 Population', '1970 Population', 'Area (km²)', 'Density (per km²)',
       'Growth Rate', 'World Population Percentage'],
      dtype='object')
```

[10]:

```python
df.drop(['CCA3', 'Capital'], axis=1, inplace=True)
```

[11]:

```python
df.tail()
```

[11]:

```
     Rank Country/Territory Continent  2022 Population  2020 Population  \
229   226 Wallis and Futuna   Oceania            11572            11655
230   172    Western Sahara    Africa           575986           556048
231    46             Yemen      Asia         33696614         32284046
232    63            Zambia    Africa         20017675         18927715
233    74          Zimbabwe    Africa         16320537         15669666

     2015 Population  2010 Population  2000 Population  1990 Population  \
229            12182            13142            14723            13454
230           491824           413296           270375           178529
231         28516545         24743946         18628700         13375121
232         16248230         13792086          9891136          7686401
233         14154937         12839771         11834676         10113893

     1980 Population  1970 Population  Area (km²)  Density (per km²)  \
229            11315             9377         142            81.4930
230           116775            76371      266000             2.1654
231          9204938          6843607      527968            63.8232
232          5720438          4281671      752612            26.5976
233          7049926          5202918      390757            41.7665

     Growth Rate  World Population Percentage
229       0.9953                         0.00
230       1.0184                         0.01
231       1.0217                         0.42
232       1.0280                         0.25
233       1.0204                         0.20
```

[11]:

```python
countries_by_continent = df['Continent'].value_counts().reset_index()
```

6 Create the bar chart

[12]:

```
# Define the custom palette. For example:
custom_palette = ['red', 'green', 'blue', 'orange', 'purple']

fig = px.bar(
    countries_by_continent,
    x='Continent',
    y='count',
    color='Continent',
    text='count',
    title='Number of Countries by Continent',
    color_discrete_sequence=custom_palette # Now this variable is defined
)
```

7 Customize the layout

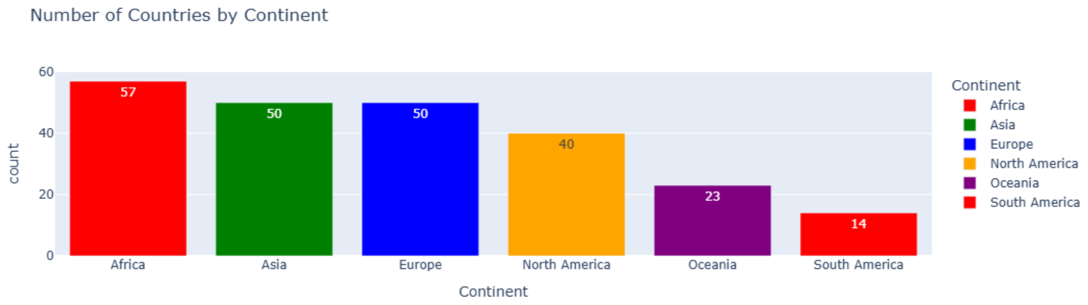
[17]:

```
fig.update_layout(
xaxis_title='Continents',
yaxis_title='Number of Countries',
plot_bgcolor='rgba(0,0,0,0)', # Set the background color to transparent
font_family='Arial', # Set font family
title_font_size=20 # Set title font size
)
```

8 Show the plot

[13]:

```
fig.show()
```



[14]:

```
continent_population_percentage = df.groupby('Continent')['World Population␣
  ↪Percentage'].sum().reset_index()
```

9 Create the pie chart

[35]:

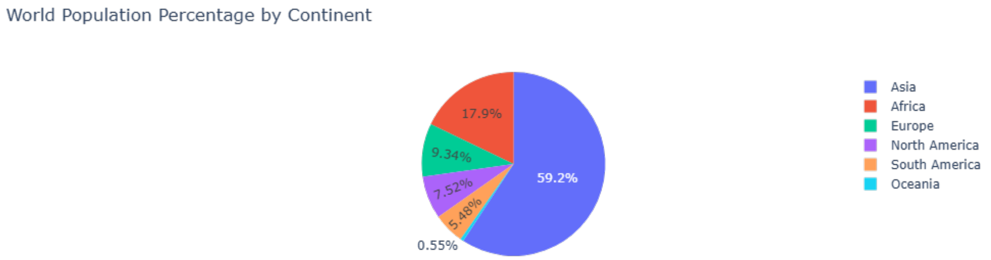
```
fig = go.Figure(data=[go.
 ↪Pie(labels=continent_population_percentage['Continent'],
values=continent_population_percentage['World Population Percentage'])])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[35], line 1
----> 1 fig = go.Figure(data=[go.
 ↪Pie(labels=continent_population_percentage['Continent'],
      2 values=continent_population_percentage['World Population Percentage'])]

NameError: name 'continent_population_percentage' is not defined
```

10 Update layout

[16]:

```
fig.update_layout(
title='World Population Percentage by Continent',
template='plotly',
paper_bgcolor='rgba(255,255,255,0)', # Set the paper background color to␣
 ↪transparent
plot_bgcolor='rgba(255,255,255,0)' # Set the plot background color to␣
 ↪transparent
)
```

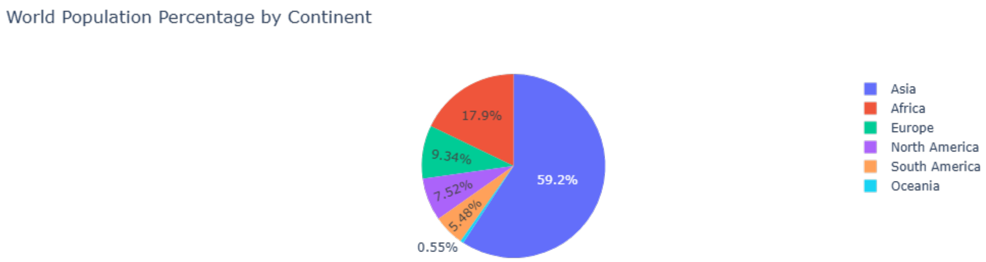


11 Update pie colors

[22]:
```
fig.update_traces(marker=dict(colors=custom_palette, line=dict(color='#FFFFFF',
width=1)))
```

12 Show the plot

[17]:
```
fig.show()
```



13 Melt the DataFrame to have a long format

[13]:
```
# Melt the DataFrame
df_melted = df.melt(
    id_vars=['Continent'],
    value_vars=[
        '2022 Population', '2020 Population', '2015 Population',
        '2010 Population', '2000 Population', '1990 Population',
        '1980 Population', '1970 Population'
    ],
    var_name='Year',
    value_name='Population'
)

# Check the data type of the 'Year' column
print(df_melted['Year'].dtype)

# Convert 'Year' to string if it is not already
df_melted['Year'] = df_melted['Year'].astype(str)

# Handle NaN values if necessary (e.g., drop NaN or fill with a placeholder)
df_melted = df_melted.dropna(subset=['Year'])  # Option to drop rows with NaN␣
  ↪in 'Year'
```

```
# Convert 'Year' to a more suitable format using regex
df_melted['Year'] = df_melted['Year'].str.extract(r'(\d+)').astype(int)

# Display the melted DataFrame
print(df_melted)
```

```
object
   Continent  Year  Population
0       Asia  2022  4600000000
1     Europe  2022   748000000
2       Asia  2020  4560000000
3     Europe  2020   743000000
4       Asia  2015  4400000000
5     Europe  2015   730000000
6       Asia  2010  4300000000
7     Europe  2010   724000000
8       Asia  2000  4000000000
9     Europe  2000   600000000
10      Asia  1990  3700000000
11    Europe  1990   500000000
12      Asia  1980  3200000000
13    Europe  1980   400000000
14      Asia  1970  2900000000
15    Europe  1970   300000000
```

14 Convert 'Year' to a more suitable format

[14]:

```
# Make sure 'Year' is treated as a string
df_melted['Year'] = df_melted['Year'].astype(str)

# Drop NaN values
df_melted = df_melted.dropna(subset=['Year'])

# Use split to get the first part
df_melted['Year'] = df_melted['Year'].str.split().str[0].astype(int)

# Display the melted DataFrame
print(df_melted)
```

```
   Continent  Year  Population
0       Asia  2022  4600000000
1     Europe  2022   748000000
2       Asia  2020  4560000000
3     Europe  2020   743000000
4       Asia  2015  4400000000
5     Europe  2015   730000000
```

```
6       Asia  2010  4300000000
7     Europe  2010   724000000
8       Asia  2000  4000000000
9     Europe  2000   600000000
10      Asia  1990  3700000000
11    Europe  1990   500000000
12      Asia  1980  3200000000
13    Europe  1980   400000000
14      Asia  1970  2900000000
15    Europe  1970   300000000
```

[15]:

```
# Convert 'Year' to a more suitable format
df_melted['Year'] = df_melted['Year']
# Changed split() to split(' ')
df_melted['Year'] = df_melted['Year'].astype(int)
```

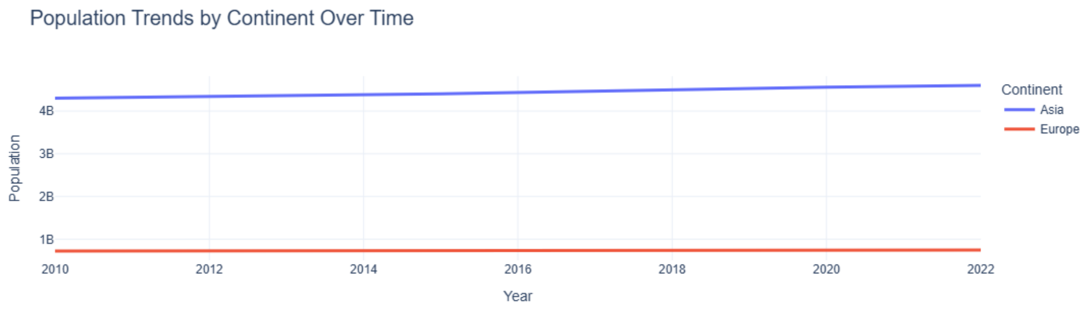
15 Aggregate population by continent and year

[16]:

```
population_by_continent = df_melted.groupby(['Continent',
'Year']).sum().reset_index()
```

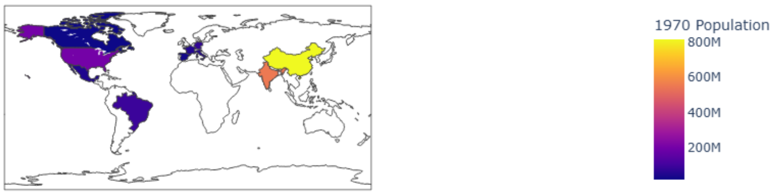
[20]:

```
'''fig = px.line(population_by_continent, x='Year', y='Population',␣
↪color='Continent',

title='Population Trends by Continent Over Time',
labels={'Population': 'Population', 'Year': 'Year'},
color_discrete_sequence=custom_palette)

fig.update_layout(

template='plotly_white',
xaxis_title='Year',
yaxis_title='Population',
font_family='Arial',
title_font_size=20,
)

fig.update_traces(line=dict(width=3))

fig.show()'''
# Sample DataFrame (replace with your actual data)
data = {
    'Continent': ['Asia', 'Asia', 'Europe', 'Europe', 'Asia', 'Europe', 'Asia',␣
↪'Europe'],
    'Year': [2022, 2020, 2022, 2020, 2015, 2015, 2010, 2010],
    'Population': [4600000000, 4560000000, 748000000, 743000000, 4400000000,␣
 ↪730000000, 4300000000, 724000000]
}

# Create the DataFrame
df_melted = pd.DataFrame(data)

# Group by Continent and Year, summing the Population
population_by_continent = df_melted.groupby(['Continent', 'Year']).sum().
 ↪reset_index()

# Define a custom color palette
custom_palette = ['#636EFA', '#EF553B', '#00CC96', '#AB63FA']  # Adjust colors␣
 ↪as needed

# Create the line plot
fig = px.line(
    population_by_continent,
    x='Year',
    y='Population',
    color='Continent',
    title='Population Trends by Continent Over Time',
    labels={'Population': 'Population', 'Year': 'Year'},
    color_discrete_sequence=custom_palette
)

# Update the layout of the figure
fig.update_layout(
    template='plotly_white',
    xaxis_title='Year',
    yaxis_title='Population',
    font_family='Arial',
    title_font_size=20,
)

# Update traces to adjust the line width
fig.update_traces(line=dict(width=3))

# Show the figure
fig.show()
```

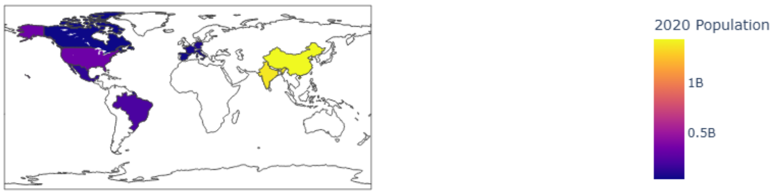


```
[24]: # Sample DataFrame (replace this with your actual data)
data = {
    'Country/Territory': ['USA', 'Canada', 'Mexico', 'Germany', 'France',␣
 ↪'Italy', 'Spain', 'Brazil', 'India', 'China'],
    '1970 Population': [203302031, 21049800, 48520000, 78025000, 55600000,␣
 ↪53600000, 30700000, 92400000, 553000000, 818000000],
    '2020 Population': [331002651, 37742154, 128932753, 83783942, 65273511,␣
 ↪60244639, 46754778, 212559417, 1380004385, 1439323776]
}

# Create the DataFrame
df = pd.DataFrame(data)

# Features to visualize
features = ['1970 Population', '2020 Population']

# Loop through each feature to create and display a choropleth map
for feature in features:
    fig = px.choropleth(
        df,
        locations='Country/Territory',
        locationmode='country names',
        color=feature,
        hover_name='Country/Territory',
        template='plotly_white',
        title=feature
    )

    fig.show()
```

1970 Population

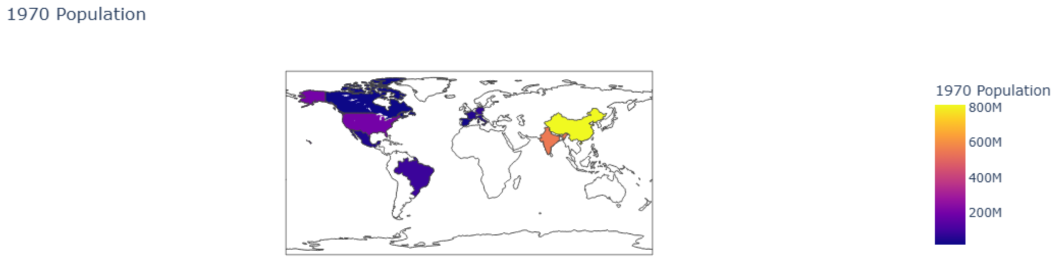


2020 Population



[25]:

```
features=['1970 Population' ,'2020 Population']
for feature in features:
    # indented block of code
    fig = px.choropleth(df,

    locations='Country/Territory',
    locationmode='country names',
    color=feature,
    hover_name='Country/Territory',
    template='plotly_white',
    title = feature)

    fig.show()
```



[29]:

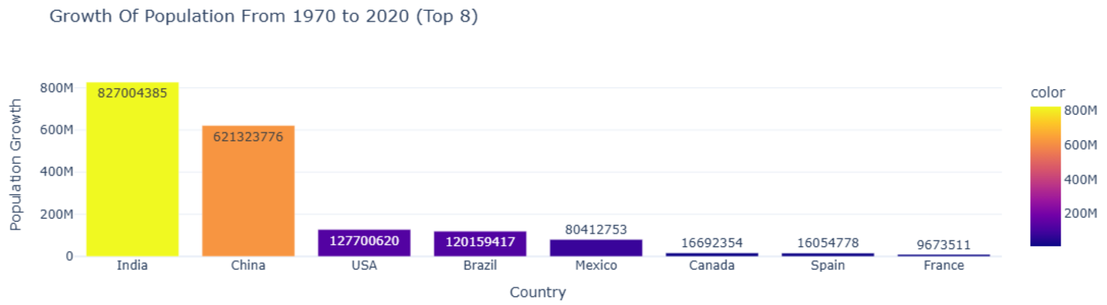
```
# Sample DataFrame (replace this with your actual data)
data = {
    'Country/Territory': ['USA', 'Canada', 'Mexico', 'Germany', 'France',␣
  ↪'Italy', 'Spain', 'Brazil', 'India', 'China'],
    '1970 Population': [203302031, 21049800, 48520000, 78025000, 55600000,␣
  ↪53600000, 30700000, 92400000, 553000000, 818000000],
    '2022 Population': [331002651, 37742154, 128932753, 83783942, 65273511,␣
  ↪60244639, 46754778, 212559417, 1380004385, 1439323776]
}

# Create the DataFrame
df = pd.DataFrame(data)

# Check column names
print("Column names:", df.columns)

# Strip whitespace from column names
df.columns = df.columns.str.strip()

# Calculate population growth
growth = (df.groupby(by='Country/Territory')['2022 Population'].sum() -
          df.groupby(by='Country/Territory')['1970 Population'].sum()).
  ↪sort_values(ascending=False).head(8)

# Display the result
print(growth)
```

```
Column names: Index(['Country/Territory', '1970 Population', '2022 Population'],
dtype='object')
Country/Territory
India      827004385
China      621323776
USA        127700620
Brazil     120159417
```

```
Mexico      80412753
Canada      16692354
Spain       16054778
France       9673511
dtype: int64
```

[30]:

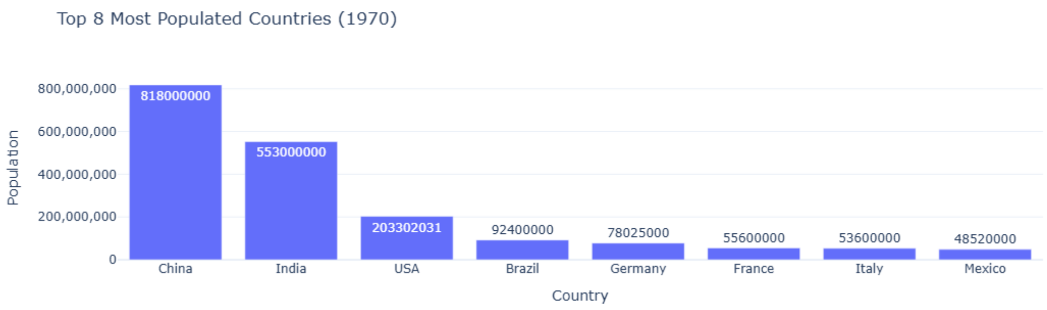
```
fig=px.bar(x=growth.index,
y=growth.values,
text=growth.values,
color=growth.values,
title='Growth Of Population From 1970 to 2020 (Top 8)',
template='plotly_white')
fig.update_layout(xaxis_title='Country',

yaxis_title='Population Growth')

fig.show()
```

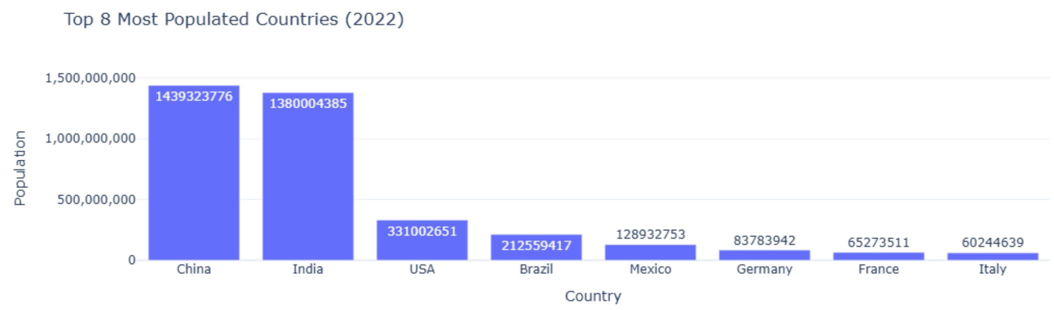


[32]:

```
# Sample DataFrame (replace this with your actual data)
data = {
    'Country/Territory': ['USA', 'Canada', 'Mexico', 'Germany', 'France',␣
 ↪'Italy', 'Spain', 'Brazil', 'India', 'China'],
    '1970 Population': [203302031, 21049800, 48520000, 78025000, 55600000,␣
 ↪53600000, 30700000, 92400000, 553000000, 818000000],
    '2022 Population': [331002651, 37742154, 128932753, 83783942, 65273511,␣
 ↪60244639, 46754778, 212559417, 1380004385, 1439323776]
}

# Create the DataFrame
df = pd.DataFrame(data)

# Get top 8 populated countries for 1970 and 2022
top_8_populated_countries_1970 = df.groupby('Country/Territory')['1970␣
 ↪Population'].sum().sort_values(ascending=False).head(8)
```

```
top_8_populated_countries_2022 = df.groupby('Country/Territory')['2022 Population'].sum().sort_values(ascending=False).head(8)

# Prepare features dictionary
features = {
    'Top 8 Populated Countries in 1970': top_8_populated_countries_1970,
    'Top 8 Populated Countries in 2022': top_8_populated_countries_2022
}

# Create bar plots
for feature_name, feature_data in features.items():
    year = feature_name.split()[-1]  # Extract the year from the feature name
    fig = px.bar(
        x=feature_data.index,
        y=feature_data.values,
        text=feature_data.values,
        title=f'Top 8 Most Populated Countries ({year})',
        template='plotly_white'
    )
    fig.update_layout(
        xaxis_title='Country',
        yaxis_title='Population',
        yaxis_tickformat=',',  # Add commas for better readability
    )

    fig.show()
```





[32]:

```python
top_8_populated_countries_1970 = df.groupby('Country/Territory')['1970␣
 ↪Population'].sum().sort_values(ascending=False).head(8)
top_8_populated_countries_2022 = df.groupby('Country/Territory')['2022␣
 ↪Population'].sum().sort_values(ascending=False).head(8)

features = {'top_8_populated_countries_1970': top_8_populated_countries_1970,␣
 ↪'top_8_populated_countries_2022': top_8_populated_countries_2022}

for feature_name, feature_data in features.items():
    # Indented block of code within the for loop
    year = feature_name.split('_')[-1] # Extract the year from the feature name
    fig = px.bar(x=feature_data.index,
    y=feature_data.values,
    text=feature_data.values,
    color=feature_data.values,
    title=f'Top 8 Most Populated Countries ({year})',
    template='plotly_white')
    fig.update_layout(xaxis_title='Country',
    yaxis_title='Population Growth')

    fig.show()
```

[33]:

```python
sorted_df_growth = df.sort_values(by='Growth Rate', ascending=False)

top_fastest = sorted_df_growth.head(6)
top_slowest = sorted_df_growth.tail(6)
```

[34]:

```python
def plot_population_trends(countries):
    # Calculate the number of rows needed
    n_cols = 2
    n_rows = (len(countries) + n_cols - 1) // n_cols

    # Add code here to define what the function should do
    # when called, for example:
    print(f'Number of rows: {n_rows}')
```

16 Create subplots

[35]:
```
def plot_population_trends(countries):
    # Calculate the number of rows needed
    n_cols = 2
    n_rows = (len(countries) + n_cols - 1) // n_cols

    # Add code here to define what the function should do
    # when called, for example:
    print(f'Number of rows: {n_rows}')
```

17 Filter data for the selected country

[36]:
```
def plot_population_trends(countries): # added countries as an argument
    # Calculate the number of rows needed
    n_cols = 2
    n_rows = (len(countries) + n_cols - 1) // n_cols

    # Add code here to define what the function should do
    # when called, for example:
    print(f'Number of rows: {n_rows}')

    for country in countries: # iterate over the countries argument
        country_df = df[df['Country/Territory'] == country] # this line will␣
 ↪now work as country is defined
        # add code here to use country_df
        print(country_df.head())

countries = ['United States', 'China'] # example list of countries
plot_population_trends(countries) # call the function with the list of countries
```

```
Number of rows: 1
     Rank Country/Territory      Continent  2022 Population  2020 Population  \
221     3     United States  North America        338289857        335942003

     2015 Population  2010 Population  2000 Population  1990 Population  \
221        324607776        311182845        282398554        248083732

     1980 Population  1970 Population  Area (km²)  Density (per km²)  \
221        223140018        200328340     9372610            36.0935

     Growth Rate  World Population Percentage
```

```
221       1.0038                         4.24
    Rank Country/Territory Continent  2022 Population  2020 Population  \
41     1              China      Asia       1425887337       1424929781

    2015 Population  2010 Population  2000 Population  1990 Population  \
41       1393715448       1348191368       1264099069       1153704252

    1980 Population  1970 Population  Area (km²)  Density (per km²)  \
41        982372466        822534450     9706961           146.8933

    Growth Rate  World Population Percentage
41          1.0                        17.88
```

18 Melt the DataFrame to have a long format

[38]:

```
# Load your data
country_df = pd.read_csv('rD:\Data Analysis\unfied mentor␣
 ↪internship\world_population.csv')
# Now you can melt the DataFrame
country_melted = country_df.melt(
    id_vars=['Country/Territory'],
    value_vars=[
        '2022 Population', '2020 Population', '2015 Population',
        '2010 Population', '2000 Population', '1990 Population',
        '1980 Population', '1970 Population'
    ],
    value_name='Population',
    var_name='Year'
)
```

```
  Cell In[38], line 2
    country_df = pd.read_csv('rD:\Data Analysis\unfied mentor␣
 ↪internship\world_population.csv')
                             ^
SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in␣
 ↪position 17-18: truncated \uXXXX escape
```

[38]:

```
def plot_population_trends(countries):
    # Calculate the number of rows needed
    n_cols = 2
    n_rows = (len(countries) + n_cols - 1) // n_cols

    # Add code here to define what the function should do
    # when called, for example:
    print(f'Number of rows: {n_rows}')
    for country in countries: # iterate over the countries argument
        country_df = df[df['Country/Territory'] == country] # this line will␣
 ↪now work as country is defined
        # add code here to use country_df
        print(country_df.head())

        # Move the following lines inside the function to access country_df
        country_melted = country_df.melt(id_vars=['Country/Territory'],
        value_vars=['2022 Population', '2020 Population', '2015 Population',
        '2010 Population', '2000 Population', '1990 Population',
        '1980 Population', '1970 Population'],
        value_name='Population', var_name='Year'
        )
        print(country_melted.head()) # Example: Print the head of the melted␣
 ↪DataFrame

countries = ['United States', 'China'] # example list of countries
plot_population_trends(countries) # call the function with the list of countries
```

```
Number of rows: 1
     Rank Country/Territory      Continent  2022 Population  2020 Population  \
221     3     United States  North America        338289857        335942003

     2015 Population  2010 Population  2000 Population  1990 Population  \
221        324607776        311182845        282398554        248083732

     1980 Population  1970 Population  Area (km²)  Density (per km²)  \
221        223140018        200328340     9372610            36.0935

     Growth Rate  World Population Percentage
221       1.0038                         4.24
  Country/Territory             Year  Population
0     United States  2022 Population   338289857
1     United States  2020 Population   335942003
2     United States  2015 Population   324607776
3     United States  2010 Population   311182845
4     United States  2000 Population   282398554
    Rank Country/Territory Continent  2022 Population  2020 Population  \
41      1             China      Asia       1425887337       1424929781

    2015 Population  2010 Population  2000 Population  1990 Population  \
41       1393715448       1348191368       1264099069       1153704252

    1980 Population  1970 Population  Area (km²)  Density (per km²)  \
41        982372466        822534450     9706961           146.8933
```

```
   Growth Rate  World Population Percentage
41          1.0                        17.88
  Country/Territory             Year  Population
0             China  2022 Population  1425887337
1             China  2020 Population  1424929781
2             China  2015 Population  1393715448
3             China  2010 Population  1348191368
4             China  2000 Population  1264099069
```

19 Convert ‘Year’ to a more suitable format

[39]:

```
country_melted['Year'] = country_melted['Year'].str.split().str[0].astype(int)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-39-17aa3ed57905> in <cell line: 1>()
----> 1 country_melted['Year'] = country_melted['Year'].str.split().str[0].
  ↪astype(int)

NameError: name 'country_melted' is not defined
```

[40]:

```
def plot_population_trends(countries):
    # Calculate the number of rows needed
    n_cols = 2
    n_rows = (len(countries) + n_cols - 1) // n_cols

    # Add code here to define what the function should do
    # when called, for example:
    print(f'Number of rows: {n_rows}')

    for country in countries: # iterate over the countries argument
        country_df = df[df['Country/Territory'] == country] # this line will␣
 ↪now work as country is defined
        # add code here to use country_df
        print(country_df.head())

        # Move the following lines inside the function to access country_df
        country_melted = country_df.melt(id_vars=['Country/Territory'],
        value_vars=['2022 Population', '2020 Population', '2015 Population',
        '2010 Population', '2000 Population', '1990 Population',
        '1980 Population', '1970 Population'],
        value_name='Population', var_name='Year'
        )
        print(country_melted.head()) # Example: Print the head of the melted␣
 ↪DataFrame
```

```
        # Process country_melted within the function
        country_melted['Year'] = country_melted['Year'].str.split().str[0].
↪astype(int)
        print(country_melted.head())

countries = ['United States', 'China'] # example list of countries
plot_population_trends(countries) # call the function with the list of countries
```

```
Number of rows: 1
     Rank Country/Territory      Continent  2022 Population  2020 Population  \
221     3     United States  North America        338289857        335942003

     2015 Population  2010 Population  2000 Population  1990 Population  \
221        324607776        311182845        282398554        248083732

     1980 Population  1970 Population  Area (km²)  Density (per km²)  \
221        223140018        200328340     9372610            36.0935

     Growth Rate  World Population Percentage
221       1.0038                         4.24
  Country/Territory             Year  Population
0     United States  2022 Population   338289857
1     United States  2020 Population   335942003
2     United States  2015 Population   324607776
3     United States  2010 Population   311182845
4     United States  2000 Population   282398554
  Country/Territory  Year  Population
0     United States  2022   338289857
1     United States  2020   335942003
2     United States  2015   324607776
3     United States  2010   311182845
4     United States  2000   282398554
    Rank Country/Territory Continent  2022 Population  2020 Population  \
41     1             China      Asia       1425887337       1424929781

    2015 Population  2010 Population  2000 Population  1990 Population  \
41       1393715448       1348191368       1264099069       1153704252

    1980 Population  1970 Population  Area (km²)  Density (per km²)  \
41        982372466        822534450     9706961           146.8933

    Growth Rate  World Population Percentage
41          1.0                        17.88
  Country/Territory             Year  Population
0             China  2022 Population  1425887337
1             China  2020 Population  1424929781
2             China  2015 Population  1393715448
```

```
3              China  2010 Population  1348191368
4              China  2000 Population  1264099069
  Country/Territory  Year  Population
0              China  2022  1425887337
1              China  2020  1424929781
2              China  2015  1393715448
3              China  2010  1348191368
4              China  2000  1264099069
```

20 Create a line plot for each country

[ ]:

```
line_fig = px.line(country_melted, x='Year', y='Population',

color='Country/Territory',

labels={'Population': 'Population', 'Year': 'Year'},
color_discrete_sequence=custom_palette)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-82-e5fedeae9967> in <cell line: 1>()
----> 1 line_fig = px.line(country_melted, x='Year', y='Population',

      2
      3 color='Country/Territory',
      4
      5 labels={'Population': 'Population', 'Year': 'Year'},

NameError: name 'country_melted' is not defined
```

[41]:

```
!pip install plotly
import plotly.express as px

def plot_population_trends(countries):
    # Calculate the number of rows needed
    n_cols = 2
    n_rows = (len(countries) + n_cols - 1) // n_cols

    # Add code here to define what the function should do
    # when called, for example:
    print(f'Number of rows: {n_rows}')

    for country in countries: # iterate over the countries argument
        country_df = df[df['Country/Territory'] == country] # this line will␣
 ↪now work as country is defined
        # add code here to use country_df
        print(country_df.head())

        # Move the following lines inside the function to access country_df
        country_melted = country_df.melt(id_vars=['Country/Territory'],
        value_vars=['2022 Population', '2020 Population', '2015 Population',
        '2010 Population', '2000 Population', '1990 Population',
        '1980 Population', '1970 Population'],
        value_name='Population', var_name='Year'
        )
        print(country_melted.head()) # Example: Print the head of the melted␣
 ↪DataFrame

        # Process country_melted within the function
        country_melted['Year'] = country_melted['Year'].str.split().str[0].
 ↪astype(int)
        print(country_melted.head())

        # Create and return the line figure within the function
        line_fig = px.line(country_melted, x='Year', y='Population',
                           color='Country/Territory',
                           labels={'Population': 'Population', 'Year': 'Year'})

        return line_fig # Return the figure from the function

countries = ['United States', 'China'] # example list of countries
line_fig = plot_population_trends(countries) # call the function and store the␣
 ↪returned figure
line_fig.show() # Display the figure
```

```
Requirement already satisfied: plotly in /usr/local/lib/python3.10/dist-packages
(5.24.1)
Requirement already satisfied: tenacity>=6.2.0 in
/usr/local/lib/python3.10/dist-packages (from plotly) (9.0.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from plotly) (24.1)
Number of rows: 1
     Rank Country/Territory      Continent  2022 Population  2020 Population  \
221     3     United States  North America        338289857        335942003

     2015 Population  2010 Population  2000 Population  1990 Population  \
221        324607776        311182845        282398554        248083732

     1980 Population  1970 Population  Area (km²)  Density (per km²)  \
221        223140018        200328340     9372610            36.0935

     Growth Rate  World Population Percentage
221       1.0038                         4.24
```

```
  Country/Territory             Year  Population
0     United States  2022 Population   338289857
1     United States  2020 Population   335942003
2     United States  2015 Population   324607776
3     United States  2010 Population   311182845
4     United States  2000 Population   282398554
  Country/Territory  Year  Population
0     United States  2022   338289857
1     United States  2020   335942003
2     United States  2015   324607776
3     United States  2010   311182845
4     United States  2000   282398554
```

21 Update the line plot to fit the subplot

[44]:

```
n_cols = 2  # Example value, adjust as needed
n_rows = 2  # Example value, adjust as needed
fig = make_subplots(rows=n_rows, cols=n_cols) # Create a subplot figure

for i in range(1, len(line_fig.data) + 1): # Define i and iterate over the␣
 ↪figures
    row = (i - 1) // n_cols + 1
    col = (i - 1) % n_cols + 1
    for trace in line_fig.data:
        fig.add_trace(trace, row=row, col=col) # Indent this line to include it␣
 ↪in the for loop
```

[45]:

```
# Assuming you want to add traces from line_fig to a new figure with subplots
import plotly.graph_objects as go

# Create a figure with subplots
fig = go.Figure()
# Assuming you have n_cols and n_rows defined somewhere
fig = make_subplots(rows=n_rows, cols=n_cols)

# Loop through traces in line_fig and add them to subplots
for i, trace in enumerate(line_fig.data):
    row = (i) // n_cols + 1  # Calculate row index starting from 0
    col = (i) % n_cols + 1   # Calculate column index
    fig.add_trace(trace, row=row, col=col)

fig.show()
```

[46]:

```
# Assuming you want to add traces from line_fig to a new figure with subplots
import plotly.graph_objects as go
from plotly.subplots import make_subplots # import the make_subplots function
```

```
# Define n_rows and n_cols here
n_cols = 2
n_rows = 1 # You'll need to calculate this based on the number of countries you want to plot

# Create a figure with subplots
fig = make_subplots(rows=n_rows, cols=n_cols)

# Loop through traces in line_fig and add them to subplots
for i, trace in enumerate(line_fig.data):
    row = (i) // n_cols + 1  # Calculate row index starting from 0
    col = (i) % n_cols + 1   # Calculate column index
    fig.add_trace(trace, row=row, col=col)

fig.show()
```

22 Update the layout of the subplots

[47]:

```
fig.update_layout(
title='Population Trends of Selected Countries Over Time',
template='plotly_white',
font_family='Arial',
title_font_size=20,
showlegend=False,
height=600*n_rows, # Adjust height for bigger plots
)

fig.update_traces(line=dict(width=3))
fig.update_xaxes(title_text='Year')
fig.update_yaxes(title_text='Population')

fig.show()
```

[48]:

```
fastest = top_fastest[['Country/Territory', 'Growth Rate']].sort_values(by='Growth Rate', ascending=False).reset_index(drop=True)
fastest
```

[48]:

```
  Country/Territory  Growth Rate
0           Moldova       1.0691
1            Poland       1.0404
2             Niger       1.0378
3             Syria       1.0376
4          Slovakia       1.0359
5          DR Congo       1.0325
```

[49]:
```
plot_population_trends(['Moldova', 'Poland', 'Niger', 'Syria', 'Slovakia', 'DR␣
↪Congo'])
```

```
Number of rows: 3
     Rank Country/Territory Continent  2022 Population  2020 Population  \
133   135           Moldova    Europe           3272996          3084847

     2015 Population  2010 Population  2000 Population  1990 Population  \
133          3277388          3678186          4251573          4480199

     1980 Population  1970 Population  Area (km²)  Density (per km²)  \
133          4103240          3711140       33846            96.7026

     Growth Rate  World Population Percentage
133       1.0691                         0.04
  Country/Territory             Year  Population
0           Moldova  2022 Population     3272996
1           Moldova  2020 Population     3084847
2           Moldova  2015 Population     3277388
3           Moldova  2010 Population     3678186
4           Moldova  2000 Population     4251573
  Country/Territory  Year  Population
0           Moldova  2022     3272996
1           Moldova  2020     3084847
2           Moldova  2015     3277388
3           Moldova  2010     3678186
4           Moldova  2000     4251573
```

[50]:
```
slowest = top_slowest[['Country/Territory', 'Growth Rate']].
↪sort_values(by='Growth Rate', ascending=False).reset_index(drop=True)
slowest
```

[50]:
```
  Country/Territory  Growth Rate
0            Latvia       0.9876
1         Lithuania       0.9869
2          Bulgaria       0.9849
3    American Samoa       0.9831
4           Lebanon       0.9816
5           Ukraine       0.9120
```

[51]:
```
plot_population_trends(['Latvia', 'Lithuania', 'Bulgaria', 'American Samoa',
'Lebanon', 'Ukraine'])
```

```
Number of rows: 3
     Rank Country/Territory Continent  2022 Population  2020 Population  \
111   151            Latvia    Europe           1850651          1897052

     2015 Population  2010 Population  2000 Population  1990 Population  \
111          1991955          2101530          2392530          2689391

     1980 Population  1970 Population  Area (km²)  Density (per km²)  \
111          2572037          2397414       64559             28.666

     Growth Rate  World Population Percentage
111       0.9876                         0.02
  Country/Territory             Year  Population
0            Latvia  2022 Population     1850651
1            Latvia  2020 Population     1897052
2            Latvia  2015 Population     1991955
3            Latvia  2010 Population     2101530
4            Latvia  2000 Population     2392530
  Country/Territory  Year  Population
0            Latvia  2022     1850651
1            Latvia  2020     1897052
2            Latvia  2015     1991955
3            Latvia  2010     2101530
4            Latvia  2000     2392530
```

[56]:

```
and_by_country = df.groupby('Country/Territory')['Area (km²)'].sum().
  ↪sort_values(ascending=False) # Changed 'Area (km2)' to 'Area (km²)'
most_land = and_by_country.head(5) # Changed 'land_by_country' to␣
  ↪'and_by_country'
least_land = and_by_country.tail(5) # Changed 'land_by_country' to␣
  ↪'and_by_country'
```

23 Create subplots

[57]:

```
fig = sp.make_subplots(rows=1, cols=2, subplot_titles=("Countries with Most␣
  ↪Land",
"Countries with Least Land"))
```

24 Plot countries with the most land

[58]:

```
fig.add_trace(go.Bar(x=most_land.index, y=most_land.values, name='Most Land',
marker_color=custom_palette[0]), row=1, col=1)
```

25 Plot countries with the least land

[59]:

```
fig.add_trace(go.Bar(x=least_land.index, y=least_land.values, name='Least Land',
marker_color=custom_palette[1]), row=1, col=2)
```

[60]:

```
fig.update_layout(
title_text="Geographical Distribution of Land Area by Country",
showlegend=False,
template='plotly_white'
)

fig.update_yaxes(title_text="Area (km2)", row=1, col=1)
fig.update_yaxes(title_text="Area (km2)", row=1, col=2)

fig.show()
```

[64]:

```
# Check for typos and correct the column name if necessary.
# For example if the column name is 'Area(km²)' use the following
# df['Area per Person'] = df['Area(km²)'] / df['2022 Population'] # Changed␣
 ↪'Area(km2)' to 'Area(km²)'
# To verify the column names present in your dataframe use:
print(df.columns)

# Assuming the column name is 'Area (km²)' based on the available data
df['Area per Person'] = df['Area (km²)'] / df['2022 Population'] # Corrected␣
 ↪column name to 'Area (km²)'
country_area_per_person = df.groupby('Country/Territory')['Area per Person'].
 ↪sum()
most_land_available = country_area_per_person.sort_values(ascending=False).
 ↪head(5)
least_land_available = country_area_per_person.sort_values(ascending=False).
 ↪tail(5)
```

```
Index(['Rank', 'Country/Territory', 'Continent', '2022 Population',
       '2020 Population', '2015 Population', '2010 Population',
       '2000 Population', '1990 Population', '1980 Population',
       '1970 Population', 'Area (km²)', 'Density (per km²)', 'Growth Rate',
       'World Population Percentage'],
      dtype='object')
```

[ ]:

```
# Check the DataFrame columns
df.columns
```

[ ]:

```
Index(['Rank', 'Country/Territory', 'Continent', '2022 Population',
       '2020 Population', '2015 Population', '2010 Population',
       '2000 Population', '1990 Population', '1980 Population',
       '1970 Population', 'Area (km²)', 'Density (per km²)', 'Growth Rate',
       'World Population Percentage'],
      dtype='object')
```

link code

26 Create subplots

[66]:

```
fig = sp.make_subplots(rows=1, cols=2, subplot_titles=("Countries with Most␣
↪Land Available Per Capita", "Countries with Least Land Available Per␣
↪Capita"))
```

27 Plot countries with the most land

[67]:

```
fig.add_trace(go.Bar(x=most_land_available.index, y=most_land_available.values,
name='Most Land', marker_color=custom_palette[2]), row=1, col=1)
```

28 Plot countries with the least land

[68]:

```
fig.add_trace(go.Bar(x=least_land_available.index, y=least_land_available.
↪values,
name='Least Land', marker_color=custom_palette[3]), row=1, col=2)
```

[9]:

```
fig.update_layout(
title_text="Distribution of Available Land Area by Country Per Capita",
showlegend=False,
template='plotly_white'
)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[9], line 1
----> 1 fig.update_layout(
      2 title_text="Distribution of Available Land Area by Country Per Capita",
      3 showlegend=False,
      4 template='plotly_white'
      5 )

NameError: name 'fig' is not defined
```

[70]:

```
fig.update_yaxes(title_text="Land Available Per Person", row=1, col=1)
fig.update_yaxes(title_text="Land Available Per Person", row=1, col=2)
```

[71]:

```
fig.show()
```

29 Presented by Aditya Prakash

30 Thankyou

[ ]:

```
```