Visualization with Python Plotly Library

Python Plotly Library is an open-source library that can be used for **data visualization and understanding data simply and easily**. Plotly s**upports various types of plots like line charts, scatter plots, histograms, cox plots**, etc. So you all must be wondering why Plotly over other visualization tools or libraries? Here's the answer –

- Plotly has **hover tool capabilities** that allow us to detect any outliers or anomalies in a large number of data points.
- It is **visually attractive** that can be accepted by a wide range of audiences.
- It **allows us for the endless customization of our graphs** that makes our plot **more meaningful and understandable** for others.

There are **three main modules** in Plotly. They are:

1. **plotly.plotly** acts as the **interface between the local machine and Plotly**. It contains functions that require **a response from Plotly's server**.
2. **plotly.graph_objects** module contains the **objects** (**Figure, layout, data, and the definition of the plots like scatter plot, line chart**) that are responsible for creating the plots. Consider the below example for better understanding.
3. **plotly.tools** module contains various tools in the **forms of the functions** that can **enhance** the Plotly experience.

Example 1

```
import plotly.express as px

# Creating the Figure instance
fig = px.line(x=[1,2, 3], y=[1, 2, 3])

# printing the figure instance
print(fig)
```

Output

***

let's create a simple plot using the pre-defined data sets defined by the plotly.

Example 2

```
import plotly.express as px
# Creating the Figure instance
fig = px.line(x=[1, 2, 3], y=[1, 2, 3])
# showing the plot
fig.show()
```

Output

***

Creating Different Types of Charts

With plotly we can create more than 40 charts and every plot can be created using the plotly.express and plotly.graph_objects class. Let's see some commonly used charts with the help of Plotly.

Line Chart

Example 3

```
import plotly.express as px

# using the iris dataset
df = px.data.iris()

# plotting the line chart
fig = px.line(df, x="species", y="petal_width")

# showing the plot
fig.show()
```

Output

***

Example 4

```
import plotly.express as px
df = px.data.tips()
plot = px.line(df, x = 'day', y = 'time')
plot.show()
```

Output

***

Example 5

```
import plotly.express as px
df = px.data.tips()
plot = px.line(df, x = 'time', y = 'total_bill', color = 'sex')
plot.show()
```

Output

***

Example 6: simple line plot with two different datasets.

```
import numpy as np
import plotly.graph_objects as go

x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
y = [1, 3, 4, 5, 6]
fig = go.Figure(data=go.Scatter(x = x, y = y))
fig.show()
```

Output

***

Example 7: Using the Iris dataset we will plot the line chart.

```
import plotly.express as px

# Loading the iris dataset
df = px.data.iris()

fig = px.line(df, x="sepal_width", y="sepal_length")
fig.show()
```

Output

***

Program (9): Write a Python program to draw *3D Plots* using *Plotty Libraries*.

Aim

To write a Python program to draw ***3D Plots using Plotly Libraries***.

Procedure

- This program generates a 3D surface plot of the function $z = sin(sqrt(x^2+y^2))$. You can modify the function or provide your own data to create different types of 3D plots. The visualization will be interactive, allowing you to rotate and explore the plot.

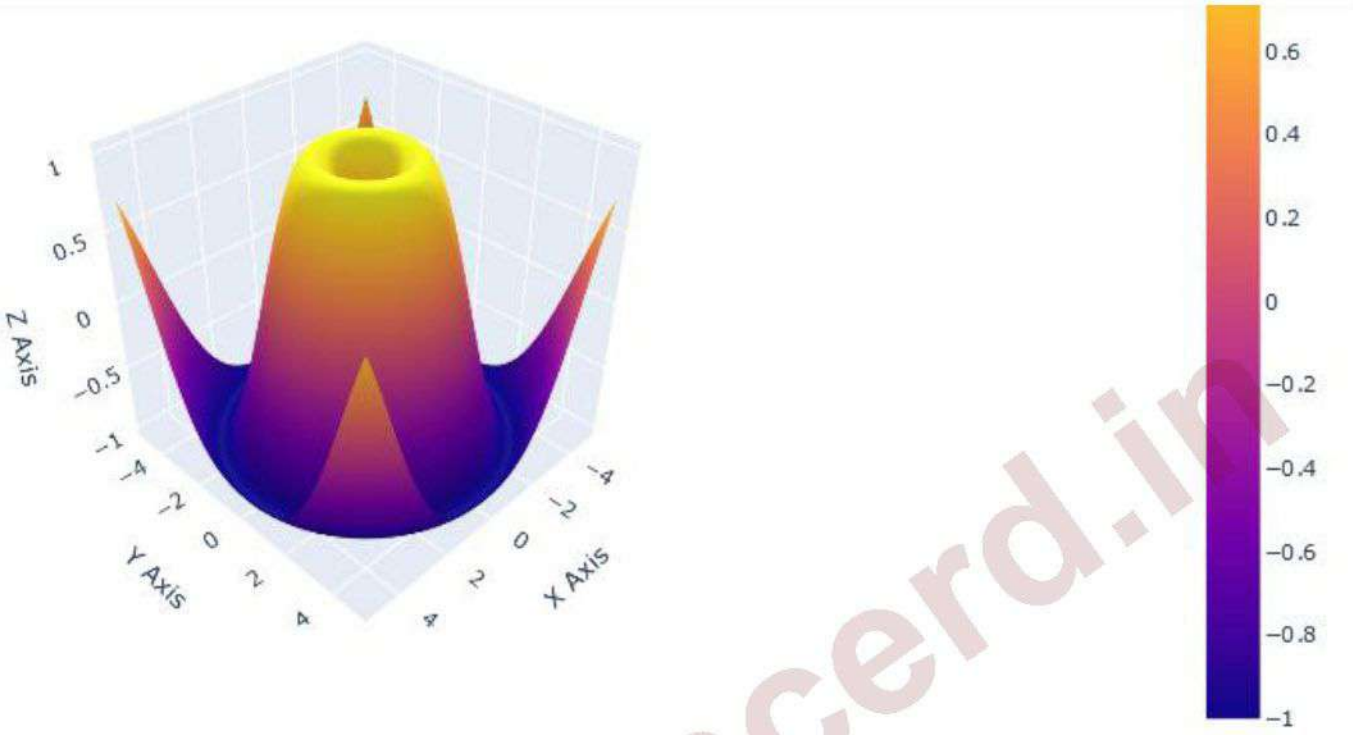
1. **Import Libraries:**
   - We start by importing the necessary libraries, including `plotly.express` for interactive visualizations.
2. **Data Loading:**
   - We load the Gapminder dataset and filter it to include only Asian countries.
3. **3D Line Plot:**
   - The key visualization is a 3D line plot created using `px.line_3d`.
   - The x-axis represents the GDP per capita (`gdpPercap`), the y-axis represents the population (`pop`), and the z-axis represents the year (`year`).
   - Each line corresponds to a different country, differentiated by color.
4. **Interactive Exploration:**
   - The resulting plot is interactive, allowing users to zoom, pan, and hover over data points to explore specific details.

- Users can observe how GDP per capita and population have changed over the years for various Asian countries. The color-coded lines help distinguish between different nations.

Program

```
import plotly.graph_objects as go
import numpy as np

# Generate sample 3D data
x = np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
x, y = np.meshgrid(x, y)
z = np.sin(np.sqrt(x**2 + y**2))

# Create a 3D surface plot
fig = go.Figure(data=[go.Surface(z=z, x=x, y=y)])

# Customize layout
fig.update_layout(scene=dict(
                    xaxis_title='X Axis',
                    yaxis_title='Y Axis',
                    zaxis_title='Z Axis'),
                  margin=dict(l=0, r=0, b=0, t=40),
                  title='3D Surface Plot of sin(sqrt(x^2 + y^2))')

# Display the 3D surface plot
fig.show()
```

Output



Result: Python program was successfully executed and drawn ***3D Plots using Plotly Libraries.***

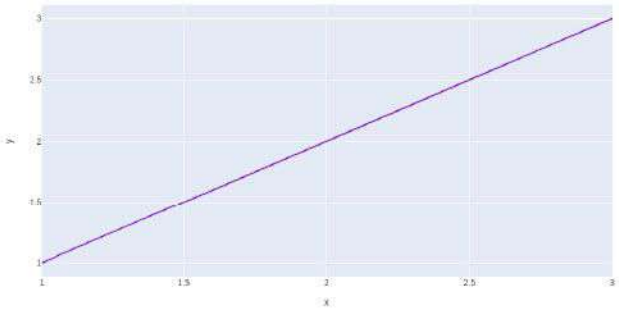
Do not write the following (Only for understanding purpose) - Output

Example 1

```
Figure({
    'data': [{'hovertemplate': 'x=%{x}<br>y=%{y}<extra></extra>',
              'legendgroup': '',
              'line': {'color': '#636efa', 'dash': 'solid'},
              'mode': 'lines',
              'name': '',
              'orientation': 'v',
              'showlegend': False,
              'type': 'scatter',
              'x': array([1, 2, 3]),
              'xaxis': 'x',
              'y': array([1, 2, 3]),
              'yaxis': 'y'}],
    'layout': {'legend': {'tracegroupgap': 0},
               'margin': {'t': 60},
               'template': '...',
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'x'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'y'}}}
})
```

Note: Figures are represented as trees where the root node has three top layer attributes – **data, layout, and frames** and the named nodes called 'attributes'. Consider the above example, **layout.legend** is a nested dictionary where the legend is the key inside the dictionary whose value is also a dictionary.

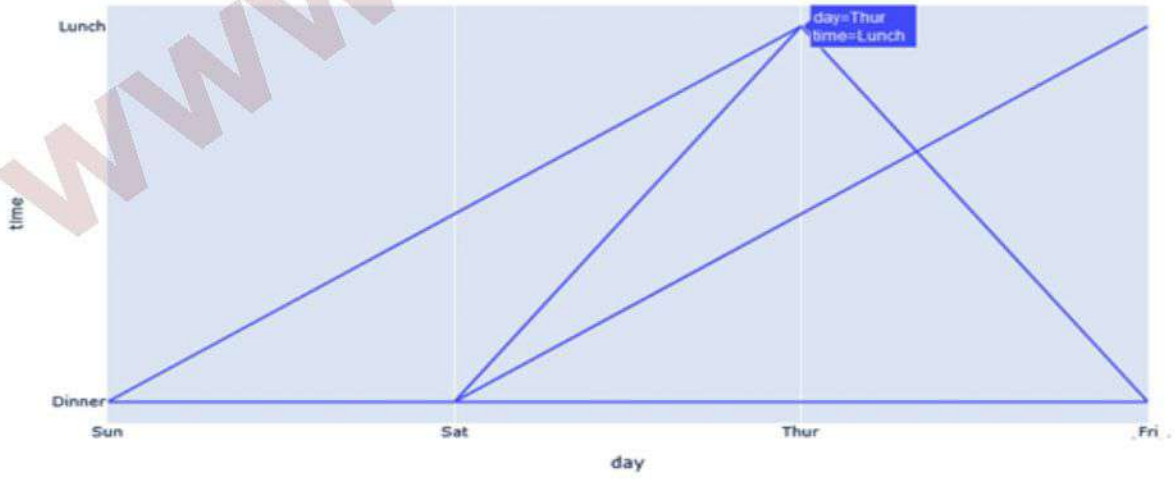
Example 2



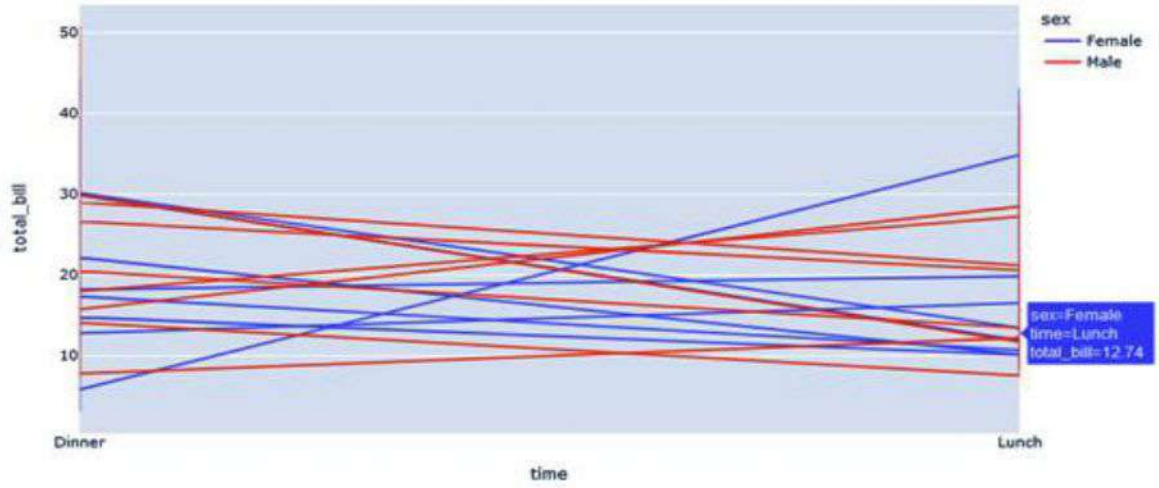
In the above example, the plotly.express module is imported which returns the Figure instance. We have created a simple line chart by passing the x, y coordinates of the points to be plotted. **Move mouse over the line and see the result.**

Example 3

Example 4

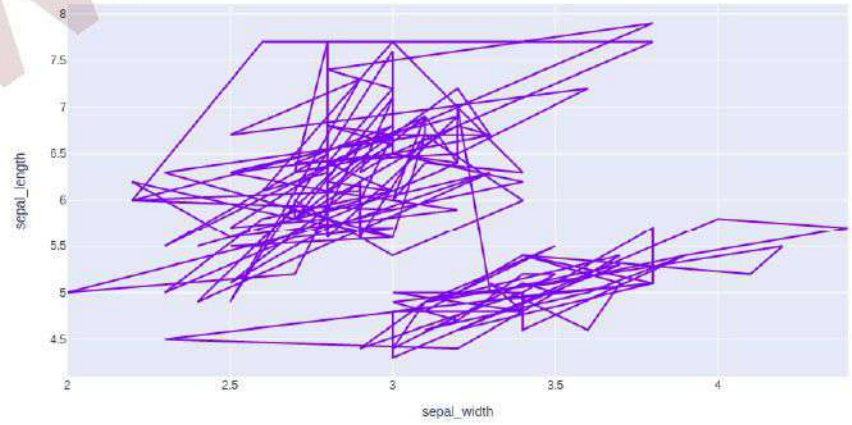


Example 5



Example 6

Example 7



***