Data Visualization Cheat Sheet

1. Matplotlib:

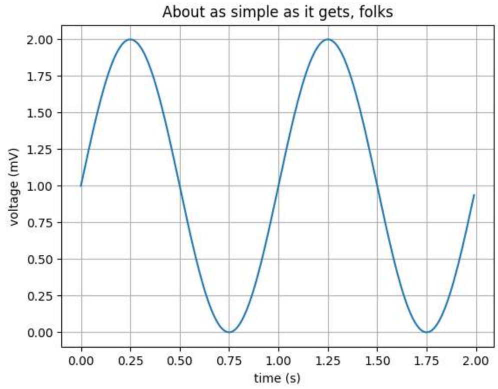
Line Plot:

- **Usage:** Display the relationship between two continuous variables over a continuous interval.

  import matplotlib.pyplot as plt

  import numpy as np

```
# Data for plotting
t = np.arange(0.0, 2.0, 0.01)
s = 1 + np.sin(2 * np.pi * t)

fig, ax = plt.subplots()
ax.plot(t, s)

ax.set(xlabel='time (s)', ylabel='voltage (mV)',
       title='About as simple as it gets, folks')
ax.grid()

fig.savefig("test.png")
plt.show()
```

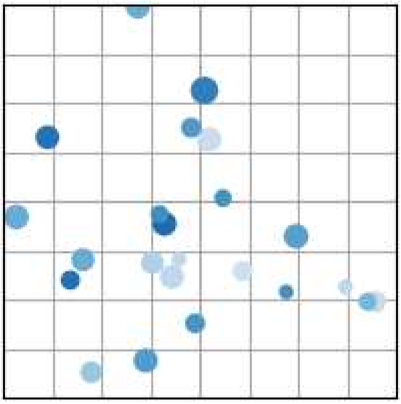


Scatter Plot:

- **Usage:** Show the distribution of individual data points and the relationship between two continuous variables.

```
plt.style.use('_mpl-gallery')

# make the data
```

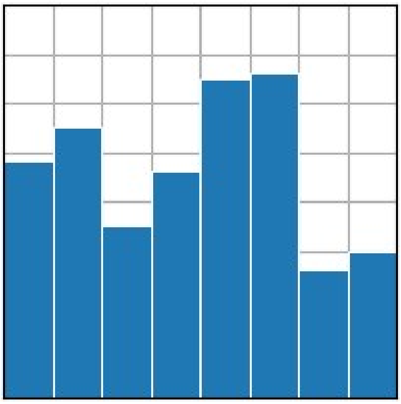
```
np.random.seed(3)
x = 4 + np.random.normal(0, 2, 24)
y = 4 + np.random.normal(0, 2, len(x))
# size and color:
sizes = np.random.uniform(15, 80, len(x))
colors = np.random.uniform(15, 80, len(x))

# plot
fig, ax = plt.subplots()

ax.scatter(x, y, s=sizes, c=colors, vmin=0, vmax=100)

ax.set(xlim=(0, 8), xticks=np.arange(1, 8),
       ylim=(0, 8), yticks=np.arange(1, 8))

plt.show()
```



Bar Plot:

- **Usage:** Compare values across different categories or display the frequency distribution of categorical data.

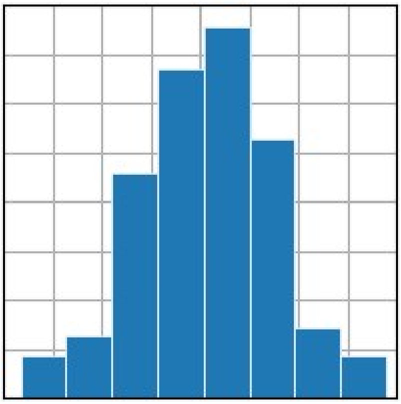
```
plt.style.use('_mpl-gallery')

# make data:
x = 0.5 + np.arange(8)
y = [4.8, 5.5, 3.5, 4.6, 6.5, 6.6, 2.6, 3.0]

# plot
fig, ax = plt.subplots()

ax.bar(x, y, width=1, edgecolor="white", linewidth=0.7)

ax.set(xlim=(0, 8), xticks=np.arange(1, 8),
       ylim=(0, 8), yticks=np.arange(1, 8))

plt.show()
```



Histogram:

- **Usage:** Illustrate the distribution of a single continuous variable and its frequency.

```
plt.style.use('_mpl-gallery')

# make data
np.random.seed(1)
x = 4 + np.random.normal(0, 1.5, 200)

# plot:
fig, ax = plt.subplots()

ax.hist(x, bins=8, linewidth=0.5, edgecolor="white")

ax.set(xlim=(0, 8), xticks=np.arange(1, 8),
       ylim=(0, 56), yticks=np.linspace(0, 56, 9))

plt.show()
```

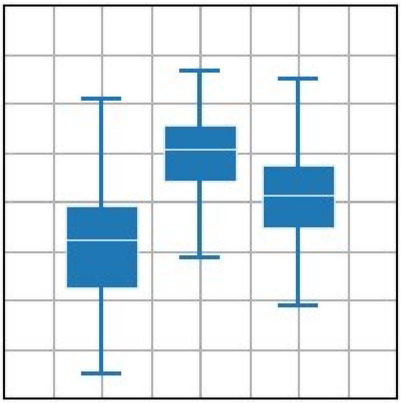


Box Plot:

- **Usage:** Display the summary statistics (median, quartiles) and identify outliers in a dataset.

```
plt.style.use('_mpl-gallery')

# make data:
np.random.seed(10)
D = np.random.normal((3, 5, 4), (1.25, 1.00, 1.25), (100, 3))
```

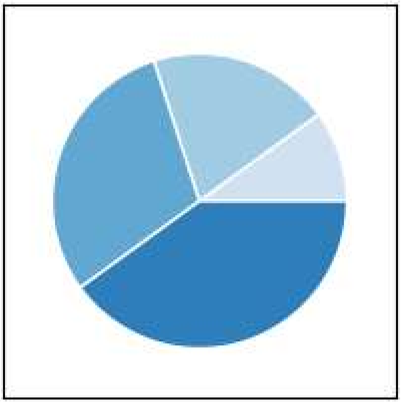
```
# plot
fig, ax = plt.subplots()
VP = ax.boxplot(D, positions=[2, 4, 6], widths=1.5, patch_artist=True,
                showmeans=False, showfliers=False,
                medianprops={"color": "white", "linewidth": 0.5},
                boxprops={"facecolor": "C0", "edgecolor": "white",
                          "linewidth": 0.5},
                whiskerprops={"color": "C0", "linewidth": 1.5},
                capprops={"color": "C0", "linewidth": 1.5})

ax.set(xlim=(0, 8), xticks=np.arange(1, 8),
       ylim=(0, 8), yticks=np.arange(1, 8))

plt.show()
```



Pie Chart:

- **Usage:** Show the composition of a whole in terms of percentages.

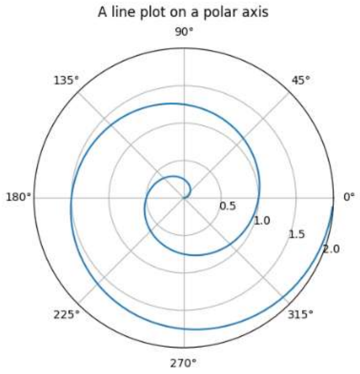
```
# make data
x = [1, 2, 3, 4]
colors = plt.get_cmap('Blues')(np.linspace(0.2, 0.7, len(x)))

# plot
fig, ax = plt.subplots()
ax.pie(x, colors=colors, radius=3, center=(4, 4),
       wedgeprops={"linewidth": 1, "edgecolor": "white"}, frame=True)

ax.set(xlim=(0, 8), xticks=np.arange(1, 8),
       ylim=(0, 8), yticks=np.arange(1, 8))

plt.show()
```



Polar Plot:

- **Usage:** Visualize data in a circular graph, often used for cyclic phenomena.

```
r = np.arange(0, 2, 0.01)
theta = 2 * np.pi * r

fig, ax = plt.subplots(subplot_kw={'projection': 'polar'})
ax.plot(theta, r)
ax.set_rmax(2)
ax.set_rticks([0.5, 1, 1.5, 2])  # Less radial ticks
ax.set_rlabel_position(-22.5)  # Move radial labels away from plotted
line
ax.grid(True)

ax.set_title("A line plot on a polar axis", va='bottom')
plt.show()
```

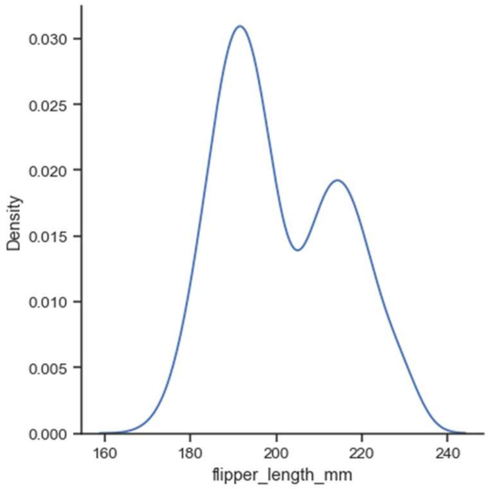


2. Seaborn:

Distribution Plot:

- **Usage:** Combine histogram and KDE to visualize the distribution of a single variable.

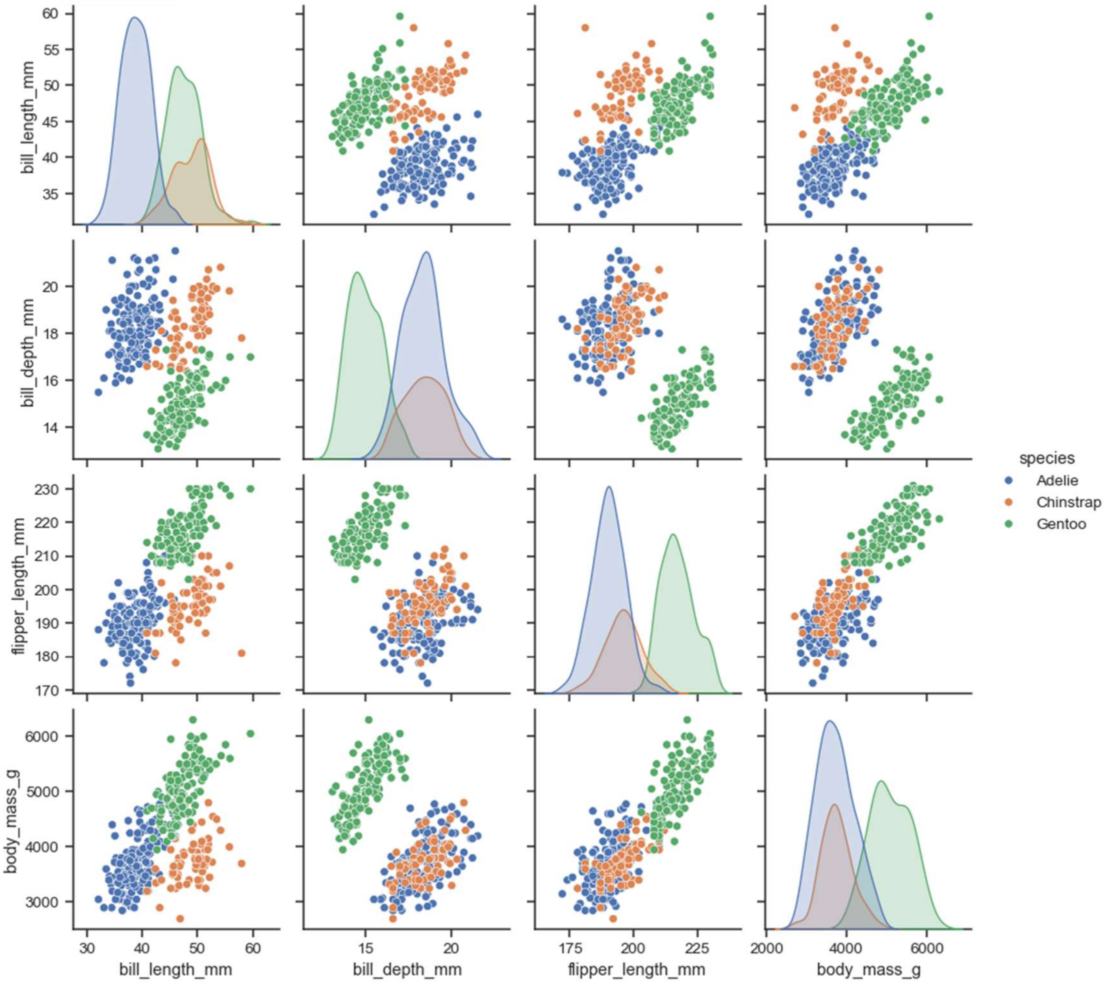
```
penguins = sns.load_dataset("penguins")
sns.displot(data=penguins, x="flipper_length_mm", kind="kde")
```



Pair Plot:

- **Usage:** Explore pairwise relationships in a dataset, useful for identifying patterns.

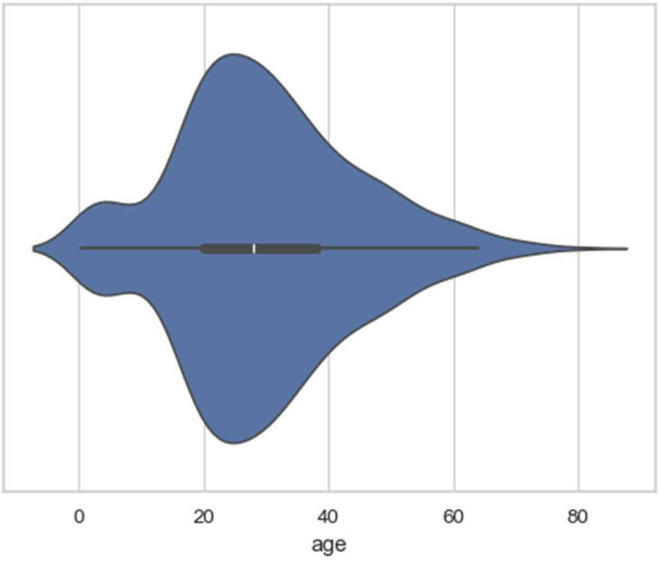
```
penguins = sns.load_dataset("penguins")
sns.pairplot(penguins, hue="species")
```



Violin Plot:

- **Usage:** Combine aspects of box plot and KDE to show the distribution of a variable for different categories.

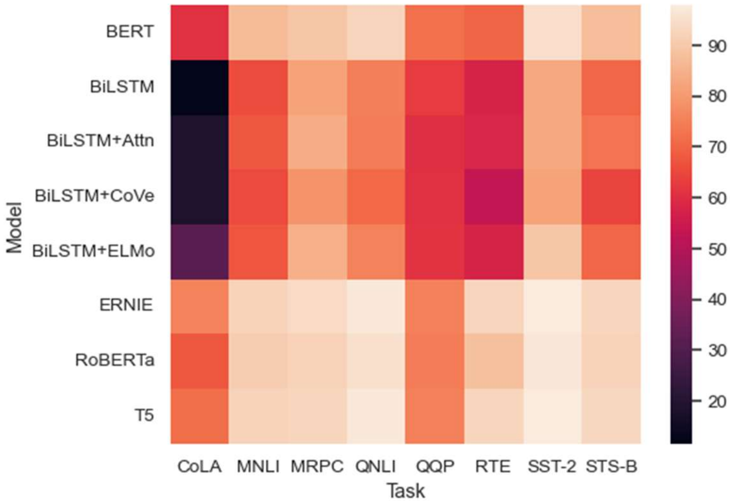
```
df = sns.load_dataset("titanic")
sns.violinplot(x=df["age"])
```



Heatmap:

- **Usage:** Display the correlation between variables in a matrix form.

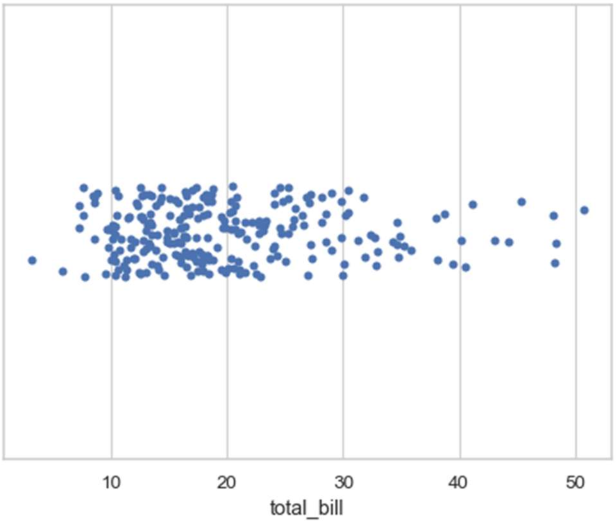
```
glue = sns.load_dataset("glue").pivot(index="Model", columns="Task",
values="Score")
sns.heatmap(glue)
```



Strip Plot:

- **Usage:** Show individual data points in relation to a categorical variable.

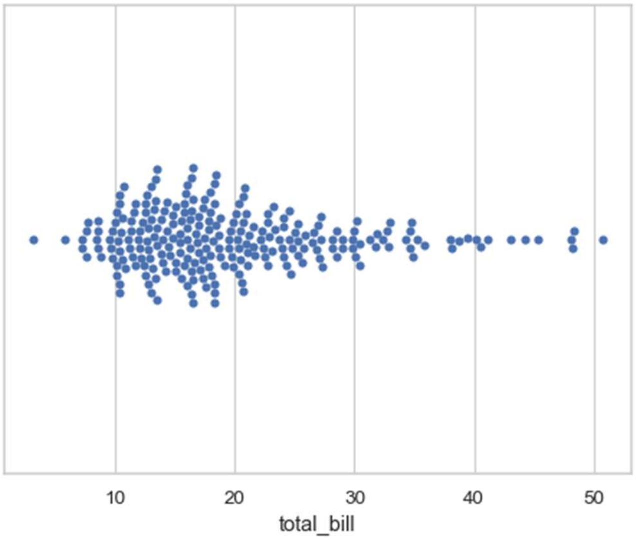
```
tips = sns.load_dataset("tips")
sns.stripplot(data=tips, x="total_bill")
```



Swarm Plot:

- **Usage:** Similar to strip plot, but points are adjusted to avoid overlap.

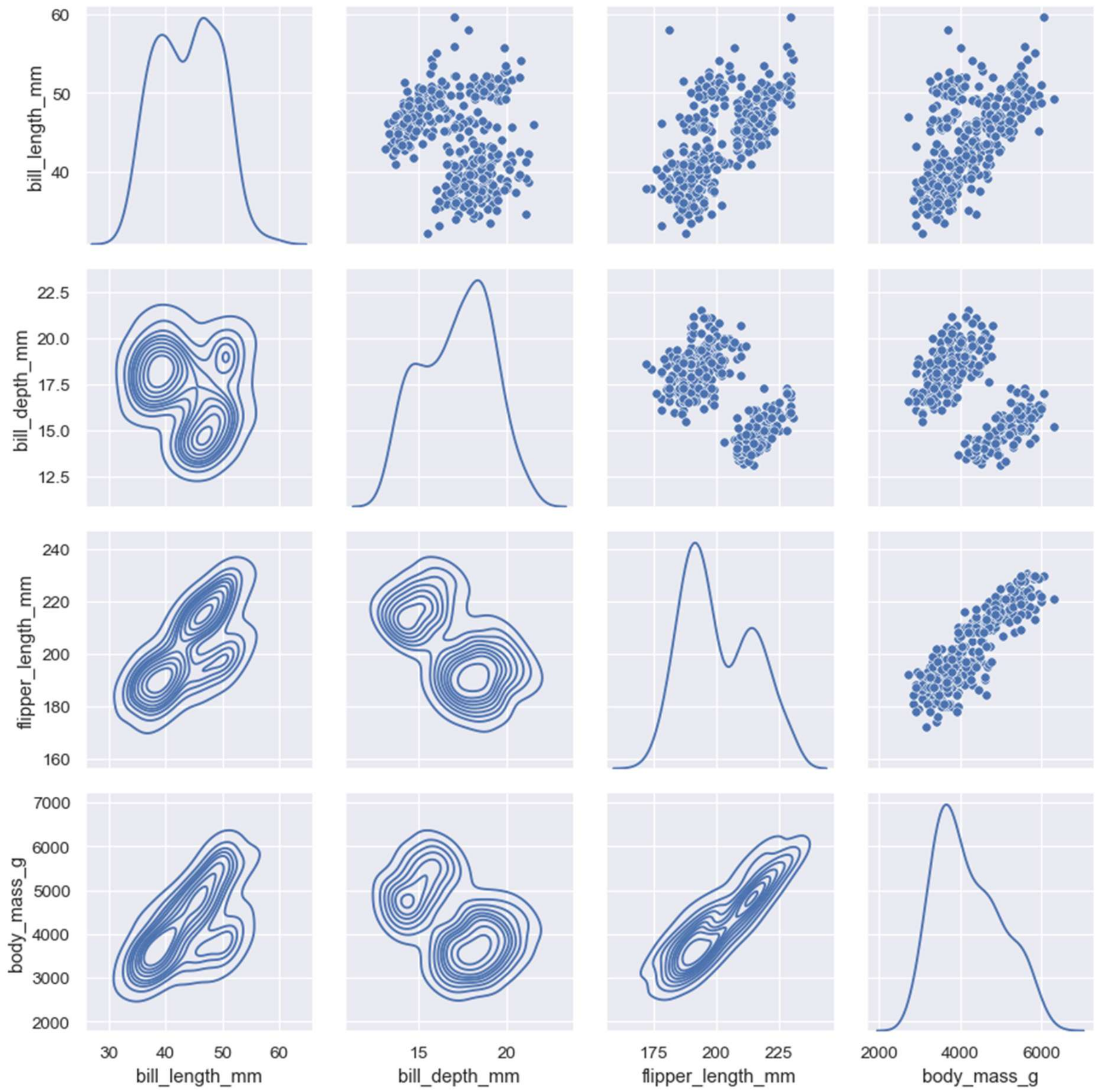
```
tips = sns.load_dataset("tips")
sns.swarmplot(data=tips, x="total_bill")
```



PairGrid:

- **Usage:** Create a grid of subplots to visualize pairwise relationships.

```
g = sns.PairGrid(penguins, diag_sharey=False)
g.map_upper(sns.scatterplot)
g.map_lower(sns.kdeplot)
g.map_diag(sns.kdeplot)
```



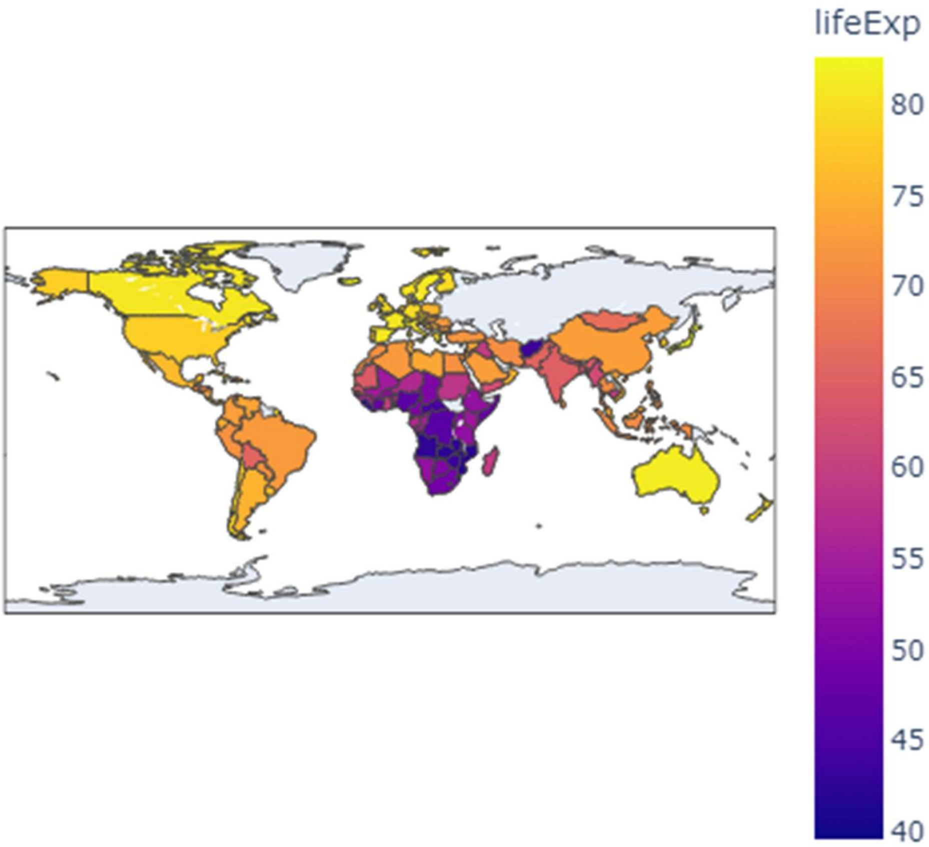
3. Plotly:

Choropleth Map:

- **Usage:** Display spatial variations and distributions over geographical regions.

```
import plotly.express as px

df = px.data.gapminder().query("year==2007")
fig = px.choropleth(df, locations="iso_alpha",
                    color="lifeExp", # lifeExp is a column of gapminder
                    hover_name="country", # column to add to hover information
                    color_continuous_scale=px.colors.sequential.Plasma)
```

```
fig.show()
```



Scatter Mapbox:

- **Usage:** Visualize geographical data on an interactive map.

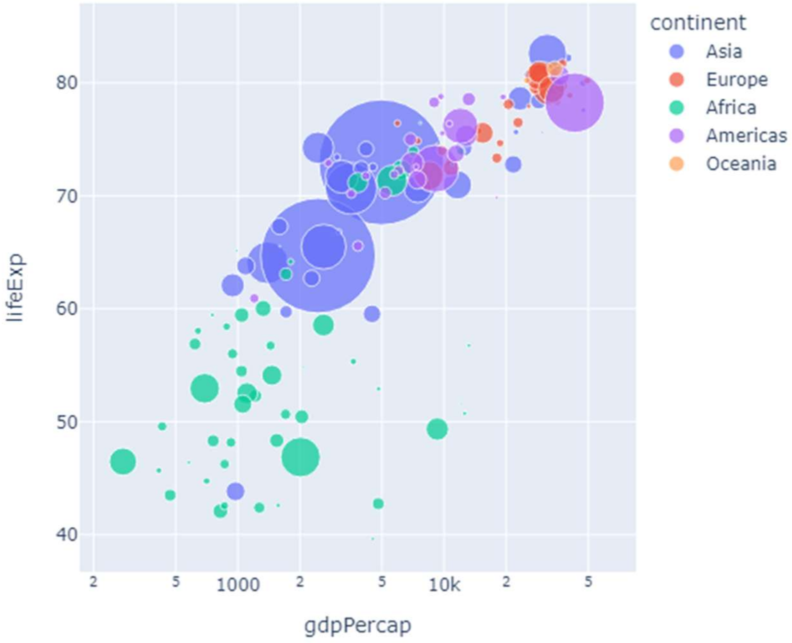
```
geo_df = gpd.read_file(gpd.datasets.get_path('naturalearth_cities'))


px.set_mapbox_access_token(open(".mapbox_token").read())
fig = px.scatter_mapbox(geo_df,
                        lat=geo_df.geometry.y,
                        lon=geo_df.geometry.x,
                        hover_name="name",
                        zoom=1)
fig.show()
```

Norway
United Kingdom
France
Spain
Italy
Ukraine
Kazakhstan
Uzbekistan
Turkey
Iraq
Iran
North Atlantic Ocean
Morocco
Egypt
Saudi Arabia
India
Mali
Senegal
Sudan
Sri Lanka
Cameroon
Kenya
Tanzania
Angola
Namibia
Madagascar
South Africa
South Atlantic Ocean
Indian Ocean
© Mapbox © OpenStreetMap

Bubble Chart:

- **Usage:** Show data points in a scatter plot with varying size.

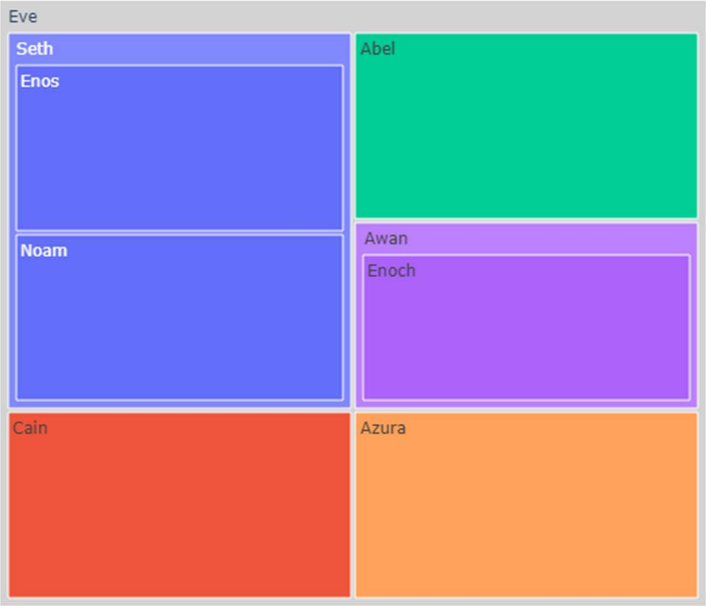
```
df = px.data.gapminder()


fig = px.scatter(df.query("year==2007"), x="gdpPercap", y="lifeExp",
                 size="pop", color="continent",
                 hover_name="country", log_x=True, size_max=60)
fig.show()
```



Treemap:

- **Usage:** Represent hierarchical data as nested rectangles.

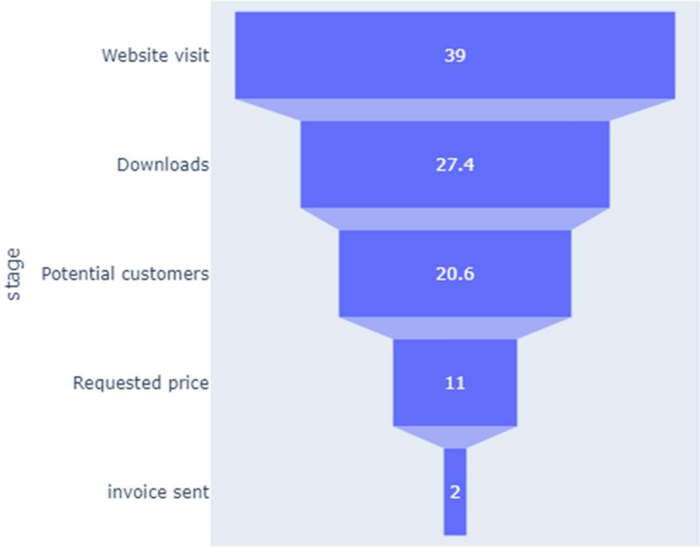
```
fig = px.treemap(
    names = ["Eve","Cain", "Seth", "Enos", "Noam", "Abel", "Awan", "Enoch", "Azura"],
    parents = ["", "Eve", "Eve", "Seth", "Seth", "Eve", "Eve", "Awan", "Eve"]
)
fig.update_traces(root_color="lightgrey")
fig.update_layout(margin = dict(t=50, l=25, r=25, b=25))
fig.show()
```



Funnel Chart:

- **Usage:** Illustrate stages in a process and visualize the flow of data.

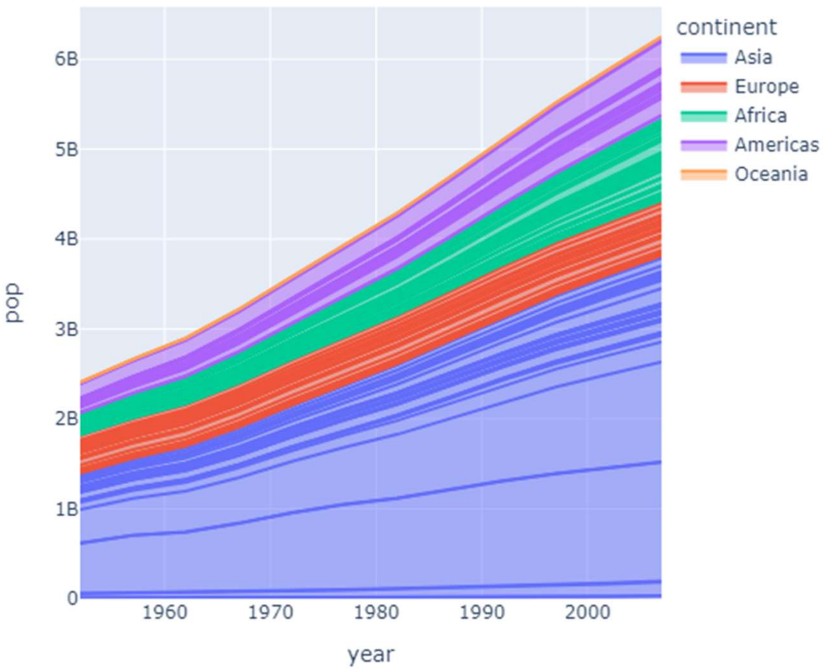
```
data = dict(
    number=[39, 27.4, 20.6, 11, 2],
    stage=["Website visit", "Downloads", "Potential customers", "Requested
price", "invoice sent"])
fig = px.funnel(data, x='number', y='stage')
fig.show()
```



Area Chart:

- **Usage:** Display the cumulative contribution of different variables over time.

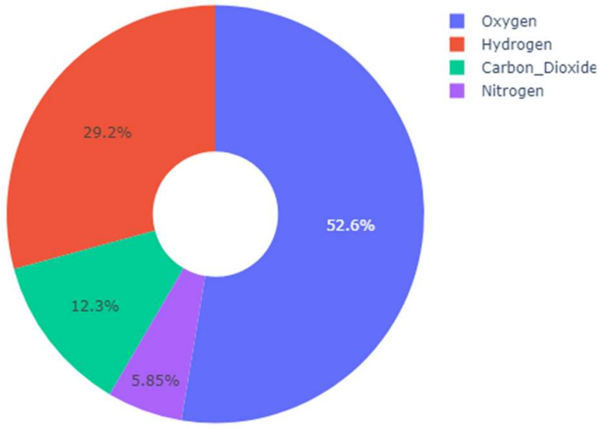
```
df = px.data.gapminder()

fig = px.area(df, x="year", y="pop", color="continent", line_group="country")

fig.show()
```



Donut Chart:

- **Usage:** A modified version of a pie chart with a hole in the center.

```
labels = ['Oxygen','Hydrogen','Carbon_Dioxide','Nitrogen']

values = [4500, 2500, 1053, 500]


# Use `hole` to create a donut-like pie chart

fig = go.Figure(data=[go.Pie(labels=labels, values=values, hole=.3)])

fig.show()
```


Oxygen
Hydrogen
Carbon_Dioxide
Nitrogen
29.2%
52.6%
12.3%
5.85%