DATA VISUALISATION

Data visualisation allows us to convert the raw data into a meaningful visual representation to understand the data in a better way . The visual representation can be a bargraph , line , heat map and so on.

MATPLOTLIB

Pyplot is used for visualisation of the data

CREATION OF PLOTS USING MATPLOTLIB

Import the matplotlib

```
import matplotlib.pyplot as plt
```

Create the grid for subplotting

fig, axs = plt.subplots(2, 2) # Create a 2x2 grid of subplots

We can plot the x and y arrays along with the title , x and y label , legend

```
x=[1,2,3,4]
y=[5,6,7,8]
plt.plot(x, y, color='blue', linestyle='--', marker='o', label='Data')
plt.xlabel('X-axis Label')
plt.ylabel('Y-axis Label')
plt.title('Customized Plot')
plt.legend()
```

Plt.show ()

->shows the plot

->can help to visualize the data and we can make changes if needed

plt.savefig()

It is particularly used for publications and creation of static images

```
plt.savefig('plot.png', dpi=300, bbox_inches='tight')
```

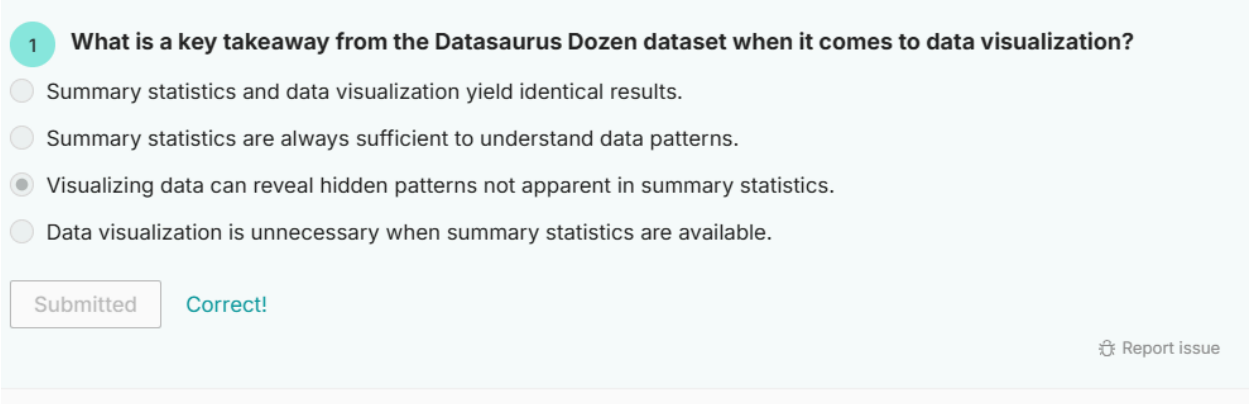
The `bbox_inches='tight'` argument ensures that the plot doesn't get cut off when saving.

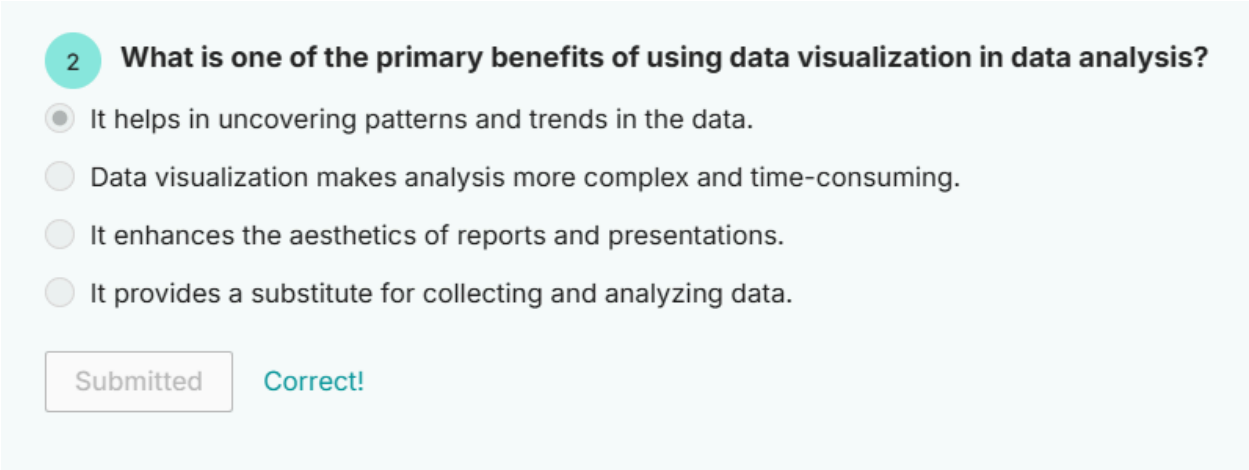
Dpi defines the pixels

TO DISPLAY THE DATA IN THE FORM OF SCATTERED DATA

```
plt.scatter(x, y, c=z, cmap='viridis')
```

DATA VISUALISATION:
IT REVEALS THE HIDDEN PATTERN IN THE DATA SETS



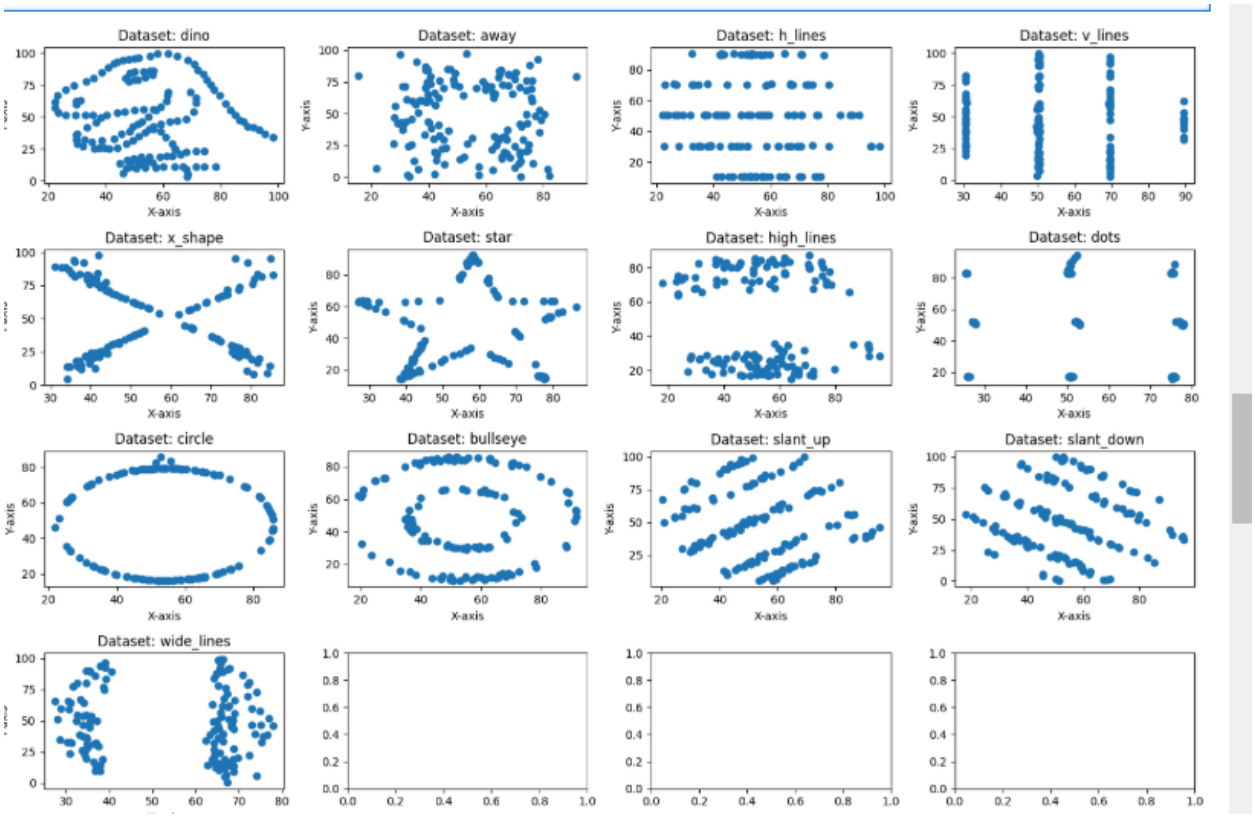


Grouping the dataset by the `dataset` column and them finding the `mean`, `median`, and `standard deviation` of the `x` and `y` columns for each dataset.

```
# Group the dataset by the 'group' column
grouped_data = df.groupby('dataset')
# Calculate summary statistics for each group
summary_stats = grouped_data.agg({
    'x': ['mean', 'median', 'std'],
    'y': ['mean', 'median', 'std']
})


# Print the summary statistics for each group
print(summary_stats)
```

Plotting the `x` and `y` columns for each dataset/category as `scatter plots`.

```
# List of unique dataset names
datasets = df['dataset'].unique()

# Create subplots with multiple columns (e.g., 4 columns per row)
num_cols = 4
num_rows = len(datasets) // num_cols + (len(datasets) % num_cols > 0)

# Create subplots
fig, axs = plt.subplots(num_rows, num_cols, figsize=(15, 10))

# Flatten the axs array to iterate over subplots
axs = axs.flatten()

# Iterate through datasets and create scatter plots
for i, dataset in enumerate(datasets):
    ax = axs[i]

    # Filter data for the current dataset
    subset_data = df[df['dataset'] == dataset]

    # Plot the scatter plot
    ax.scatter(subset_data['x'], subset_data['y'])

    # Set title for the subplot
    ax.set_title(f'Dataset: {dataset}')

    # Add labels to the axes
    ax.set_xlabel('X-axis')
    ax.set_ylabel('Y-axis')

# Remove any empty subplots
for i in range(len(datasets), num_rows * num_cols):
    fig.delaxes(axs[i])

# Adjust layout
plt.tight_layout()

# Show the plots
plt.show()
```



BASICS OF MATPLOTLIB

->LINE
->BAR
->SCATTER
->PIECHART

2 **Understanding Figures and Axes in Matplotlib**

In Matplotlib, what is the difference between a Figure and an Axes?

- A Figure and an Axes are the same thing in Matplotlib terminology.
- A Figure is a type of plot, and an Axes is the canvas where all plots are drawn.
- A Figure is the entire plotting area or canvas, while an Axes is an individual plot within the Figure.
- A Figure refers to the axes and labels of a plot, whereas an Axes refers to the graph itself.

Submitted Correct!

FOR BAR GRAPH

```
import matplotlib.pyplot as plt

# Create a Figure object
fig = plt.figure()

# Add an Axes object to the Figure
ax = fig.add_subplot(1, 1, 1)

# Plot data on the Axes
ax.plot([1, 2, 3, 4], [10, 20, 25, 30])

# Customize the plot
ax.set_title('Simple Plot')
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')

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

OBJECT ORIENTED NATURE OF MATPLOT

->FIGURE - a flask which holds everything like label , axes , title
->AXES - refers to each plot and graph

```
import matplotlib.pyplot as plt

# Create a Figure and Axes object
fig, ax = plt.subplots()

# Plot data on the Axes
ax.plot([1, 2, 3, 4], [10, 20, 25, 30])

# Customize the plot
ax.set_title('Simple Plot')
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')

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

PLOT GRAPH SHOULD BE USED WHEN 2 VARIABLES ARE INDEPENDENT
LINE GRAPH IS USED FOR CONTINUOUS DATA