Sprints

– Task 22 –

Basic and Advanced Data Visualizations with Matplotlib

Name: Basel Amr Barakat
Email: baselamr52@gmail.com

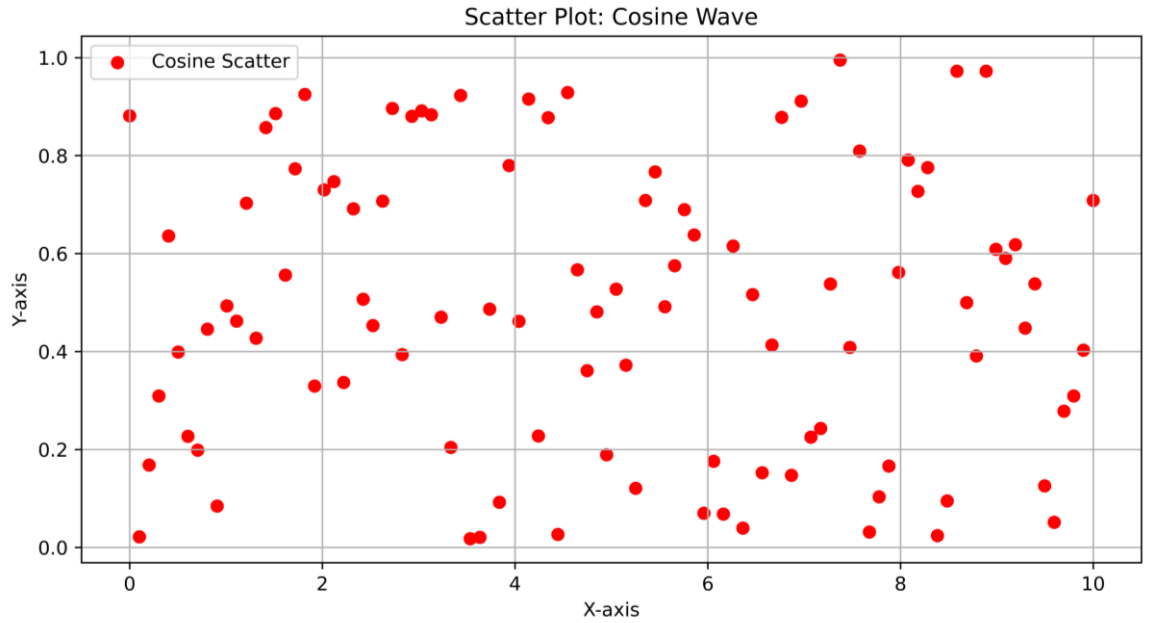
Requirement One

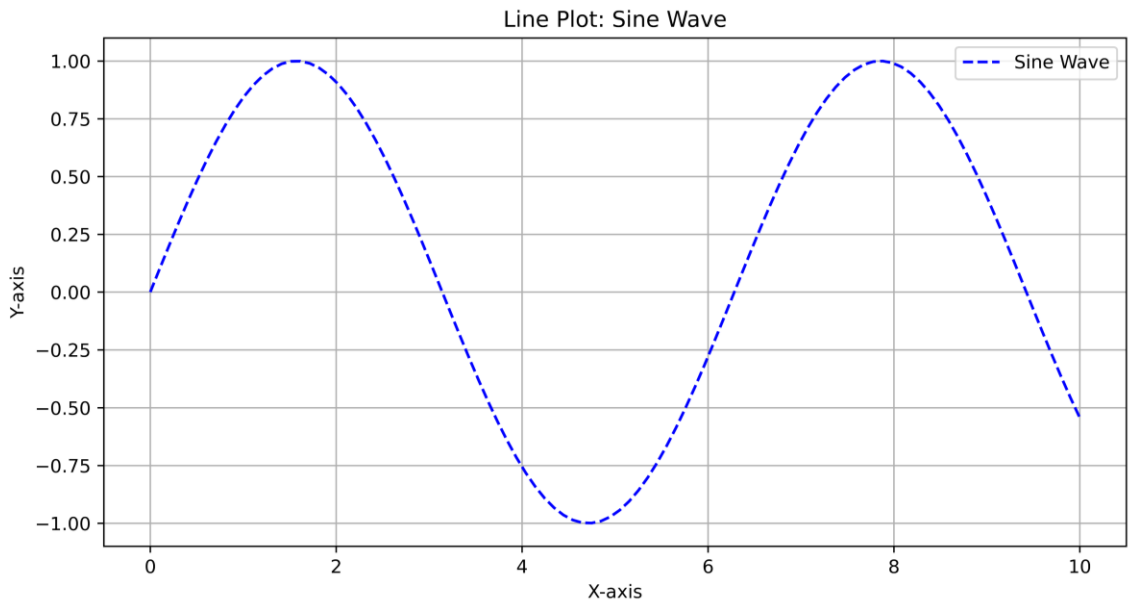
Line and Scatter Plots

1.1 Plotting a line-plot and Scatter plot Code

```
# Make The Data for Line and Scatter plot
x_line = np.linspace(0, 10, 100)
y_line = np.sin(x_line)
y_scatter = np.random.rand(100)

# Create Line Plot
print("Creating a line plot")
plt.figure(figsize=(10, 5))
plt.plot(x_line, y_line, label='Sine Wave', color='b', linestyle='--')
plt.title('Line Plot: Sine Wave')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid(True)
plt.savefig('01_line_plot.png',format="png",dpi=800)
plt.show()

# Create Scatter Plot
print("Creating a Scatter plot")
plt.figure(figsize=(10, 5))
plt.scatter(x_line, y_scatter, label='Cosine Scatter', color='r', marker='o')
plt.title('Scatter Plot: Cosine Wave')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid(True)
plt.savefig('01_scatter_plot.png',format="png",dpi=800)
plt.show()
```

1.1 Plotting a line-plot and Scatter plot Output





1.2 Custom Line plots code

```python
# Prepare Dates
dates = np.arange(1, 31)
# Prepare tempreture values
temperature = np.random.randint(20, 35, 30)

# Get all available styles in Matplotlib
styles = plt.style.available

# Create a grid for subplots
n_styles = len(styles)
cols = 4
rows = int(n_styles / cols)  # Calculate the number of rows required
print(n_styles,cols,rows)
# Create a figure for subplots
fig, axes = plt.subplots(rows, cols, figsize=(15, 5 * rows))
axes = axes.flatten()  # Flatten axes for easy iteration

# Loop through styles and plot on subplots
for i, style in enumerate(styles):
    plt.style.use(style)
    ax = axes[i]
    ax.plot(dates, temperature, label='Tempreture of January')
    ax.set_xlabel('Date')
    ax.set_ylabel('Temperature')
    ax.set_title(f'Style: {style}', fontsize=10)
    ax.legend()
    ax.grid(True)
    ax.tick_params(axis='both', which='major', labelsize=8)

# Hide any unused subplots
for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

plt.tight_layout()
plt.savefig('01_custome_line_plot.png',format="png",dpi=800)
plt.show()
```

1.2 Custom Line plots Output

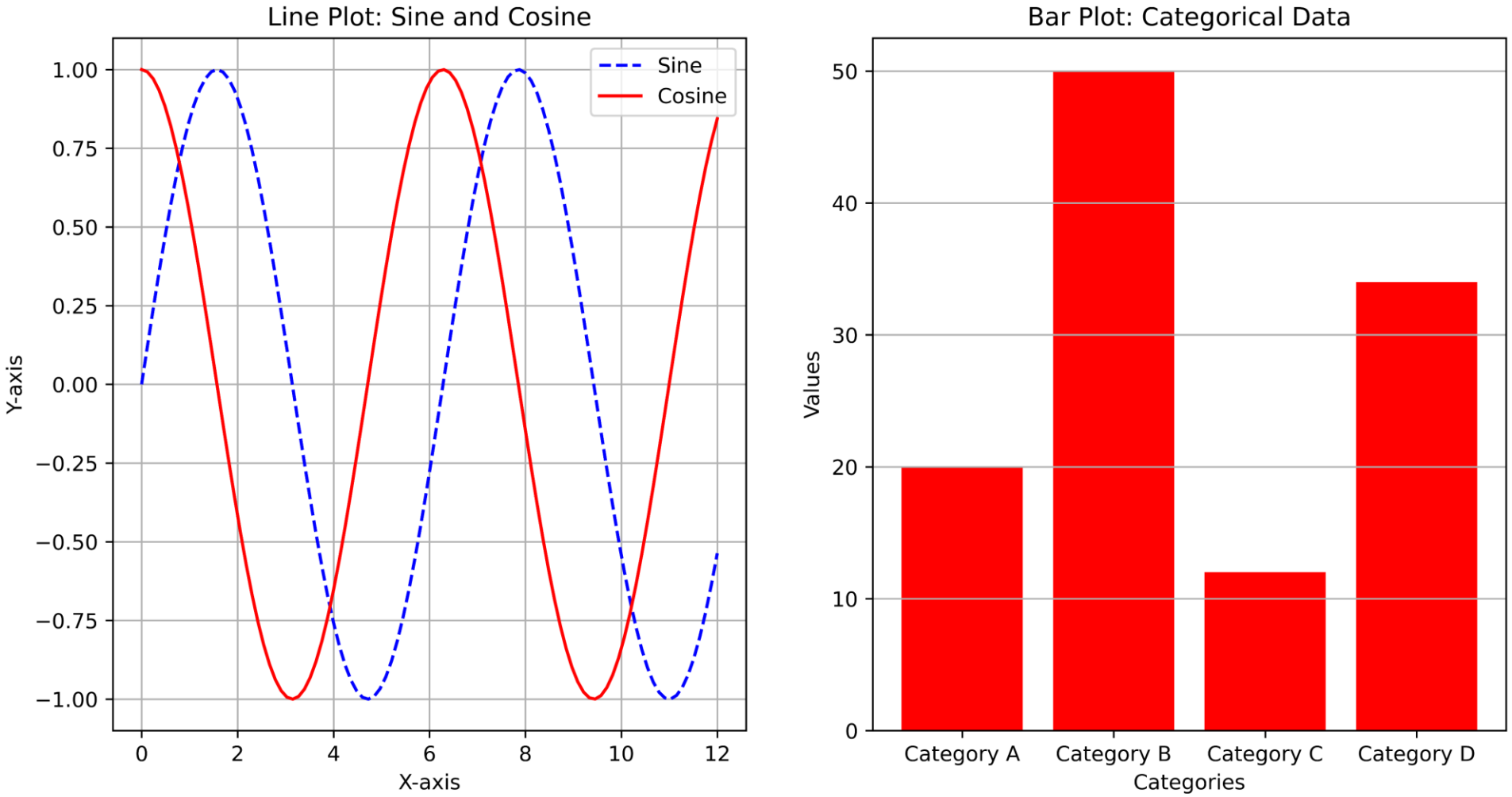
Requirement Two

Advanced Plotting with Subplots and Annotations

2.2 Creating two subplots with the same size code

```python
# Create the data
x = np.linspace(0, 12, 100)
y1 = np.sin(x)
y2 = np.cos(x)

# Create categorical data for bar plot
categories = ['Category A', 'Category B', 'Category C', 'Category D']
values = [20, 50, 12, 34]

# Create the figure with subplots
plt.figure(figsize=(12, 6))

# Subplot 1: Line plot
plt.subplot(1, 2, 1)
plt.plot(x, y1, label='Sine', color='b', linestyle='--')
plt.plot(x, y2, label='Cosine', color='r', linestyle='-')
plt.title('Line Plot: Sine and Cosine')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid(True)

# Subplot 2 : Bar plot
plt.subplot(1, 2, 2)
plt.bar(categories, values, color='r')
plt.title('Bar Plot: Categorical Data')
plt.xlabel('Categories')
plt.ylabel('Values')
plt.grid(axis='y')
plt.savefig('02_LineandBarPlot_SameSizes_.png',format="png",dpi=800)
# Adjust spacing between subplots
plt.tight_layout()
```

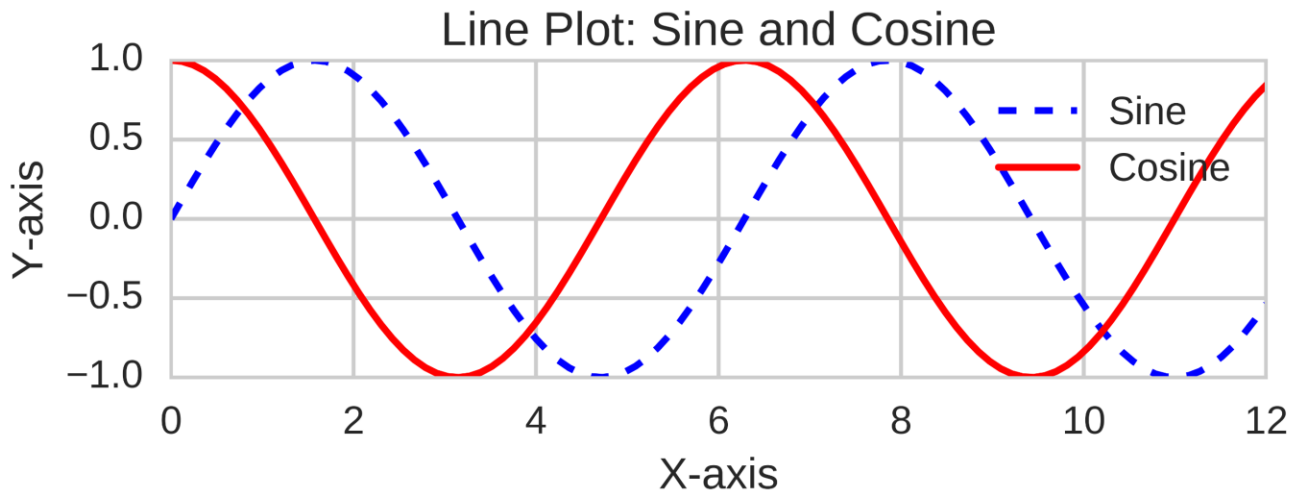
2.2 Creating two subplots with the same size Output

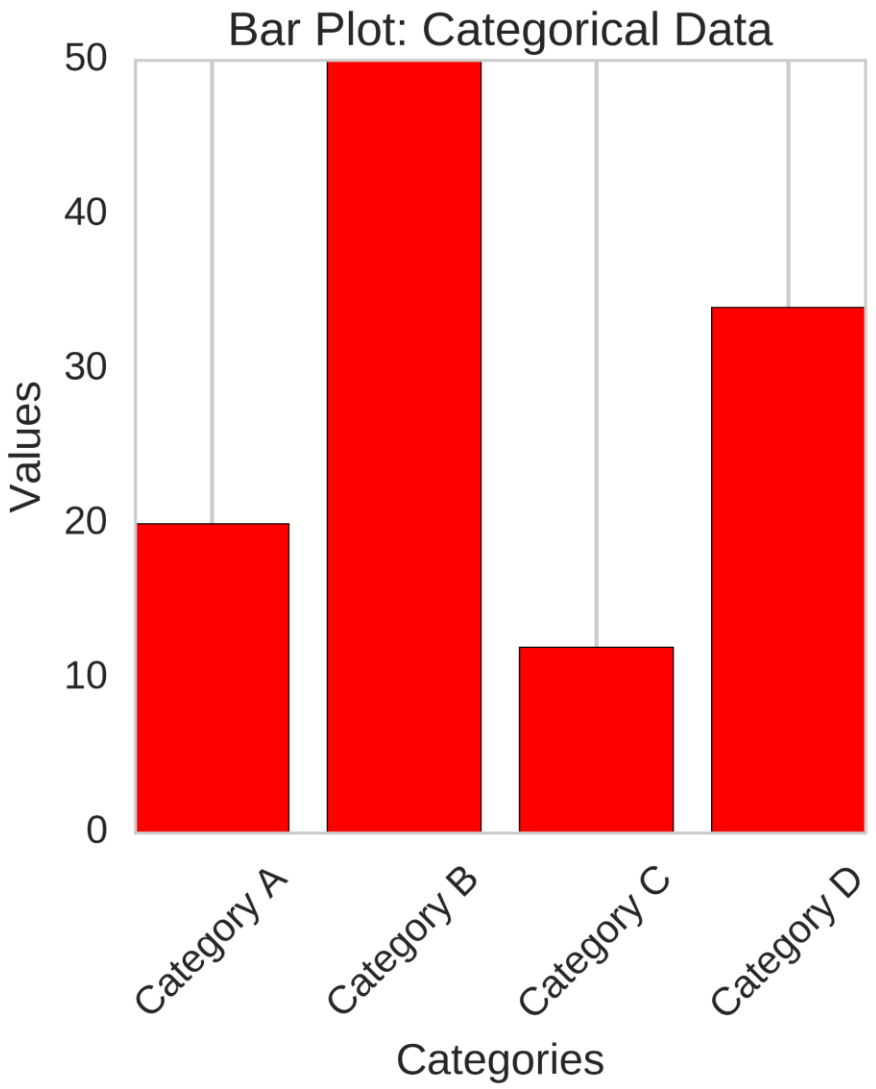


2.2 Creating two subplots with the diff size code

```
[7]
# Creating two suplots with different sizes
fig = plt.figure(constrained_layout=True, figsize=(10, 5))
gs = fig.add_gridspec(2, 5)

# Subplot 1: Line plot
ax1 = fig.add_subplot(gs[0, :3])
ax1.plot(x, y1, label='Sine', color='b', linestyle='--')
ax1.plot(x, y2, label='Cosine', color='r', linestyle='-')
ax1.set_xlabel('X-axis')
ax1.set_ylabel('Y-axis')
ax1.legend()
ax1.grid(True)
ax1.set_title('Line Plot: Sine and Cosine')

# Subplot 2 : Bar plot
ax2 = fig.add_subplot(gs[:, 3:])
ax2.bar(categories, values, color='r')
ax2.set_title('Bar Plot: Categorical Data')
ax2.set_xlabel('Categories')
ax2.set_ylabel('Values')
ax2.set_xticklabels(labels=categories,rotation=45)
ax2.grid(axis='y')
plt.savefig('02_LineandBarPlot_DifferentSizes_.png',format="png",dpi=800)
```

2.2 Creating two subplots with the diff size Output

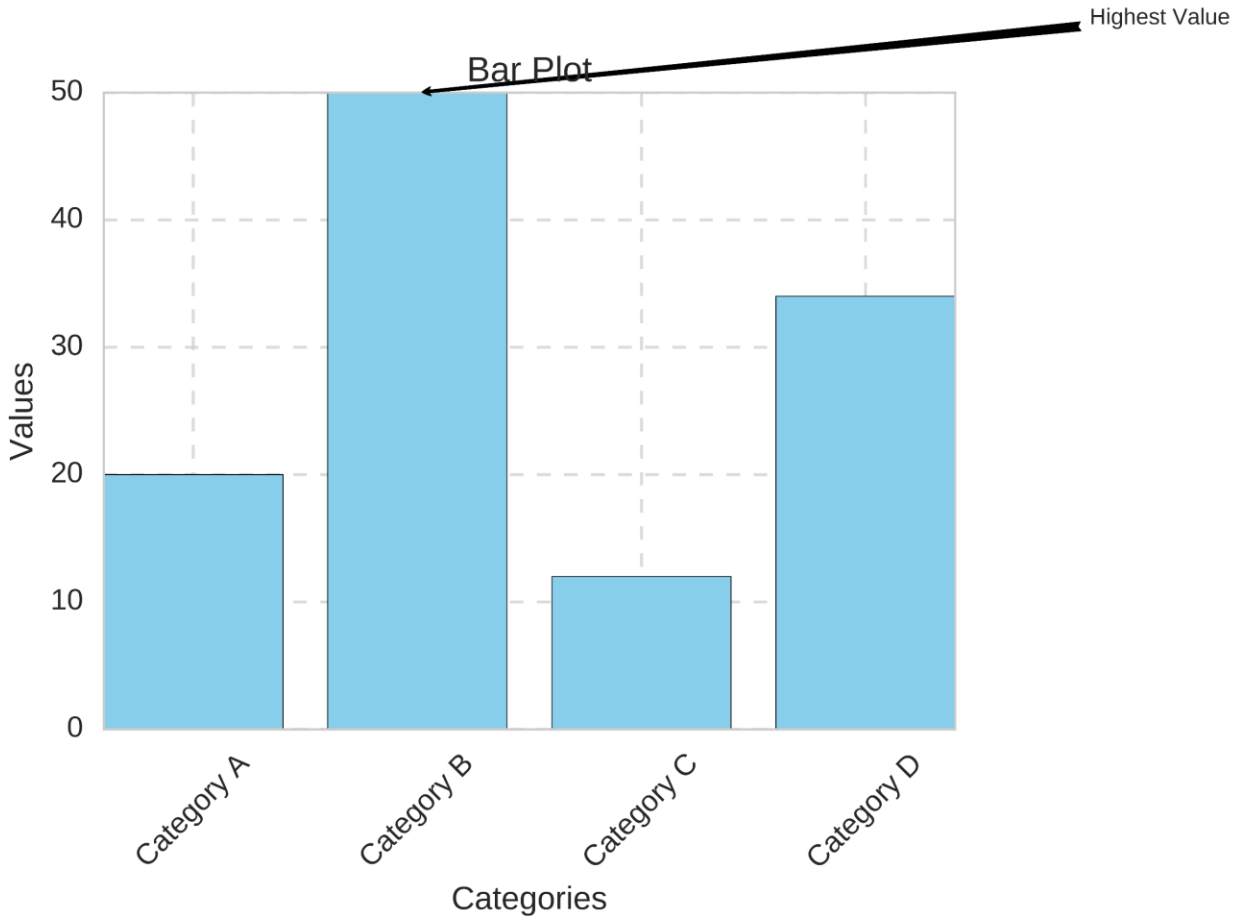


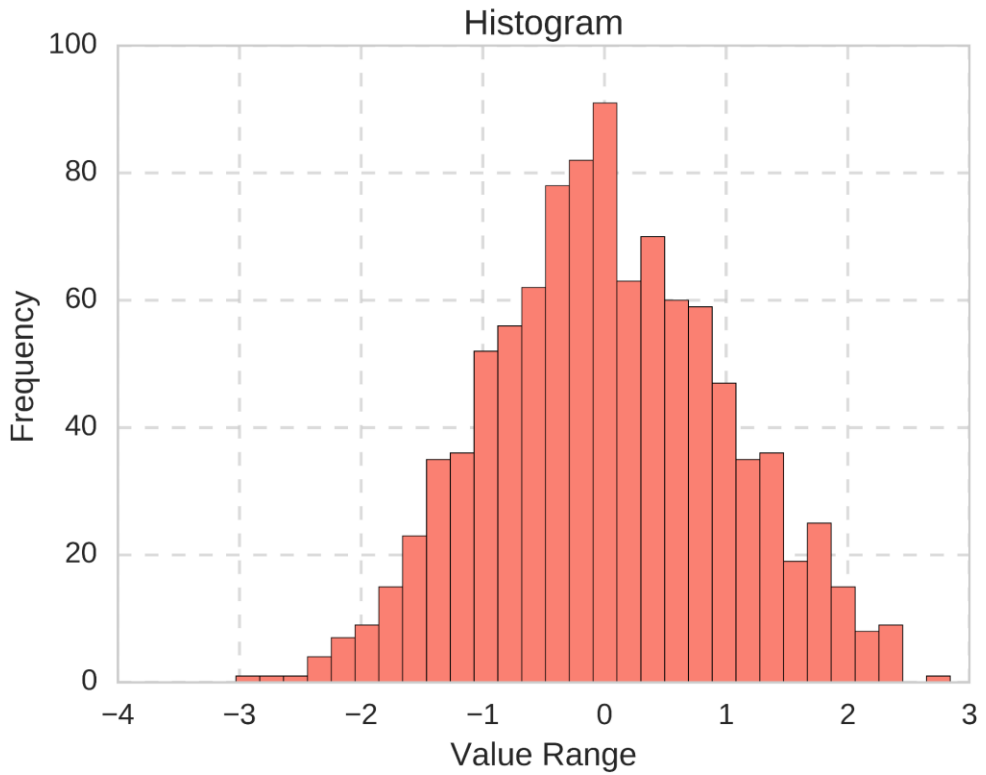


2.3 Creating Bar Plots with annotation Code

```
[8]  1 data = np.random.randn(1000)
     2
     3 fig, axs = plt.subplots(1, 2, figsize=(14, 6))
     4
     5 # Bar Plot
     6 axs[0].bar(categories, values, color='skyblue')
     7 axs[0].set_title('Bar Plot')
     8 axs[0].set_xlabel('Categories')
     9 axs[0].set_ylabel('Values')
    10 axs[0].grid(True, linestyle='--', alpha=0.7)
    11 axs[0].annotate('Highest Value', xy=('Category B', 50), xytext=('C', 55),
    12                 arrowprops=dict(facecolor='black', arrowstyle='fancy'),
    13                 horizontalalignment='left',
    14                 verticalalignment='bottom')
    15 axs[0].set_xticklabels(labels=categories,rotation=45)
    16
    17 # Histogram
    18 axs[1].hist(data, bins=30, color='salmon', edgecolor='black')
    19 axs[1].set_title('Histogram')
    20 axs[1].set_xlabel('Value Range')
    21 axs[1].set_ylabel('Frequency')
    22 axs[1].grid(True, linestyle='--', alpha=0.7)
    23
    24 plt.tight_layout()
    25 plt.savefig('02_LineandBarPlot_Annotation_.png',format="png",dpi=800)
    26 plt.show()
```

2.3 Creating Bar Plots with annotation Output



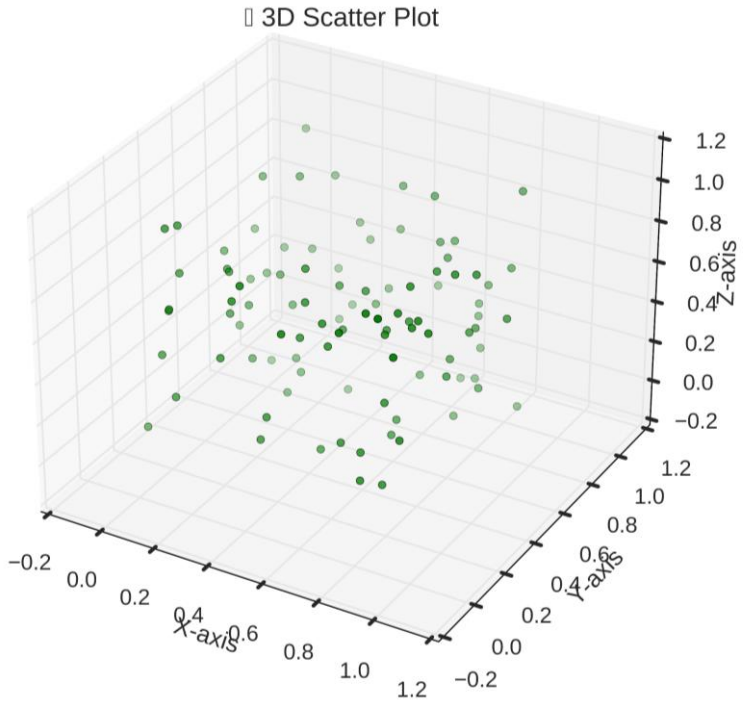


Requirement Three

3D Plotting

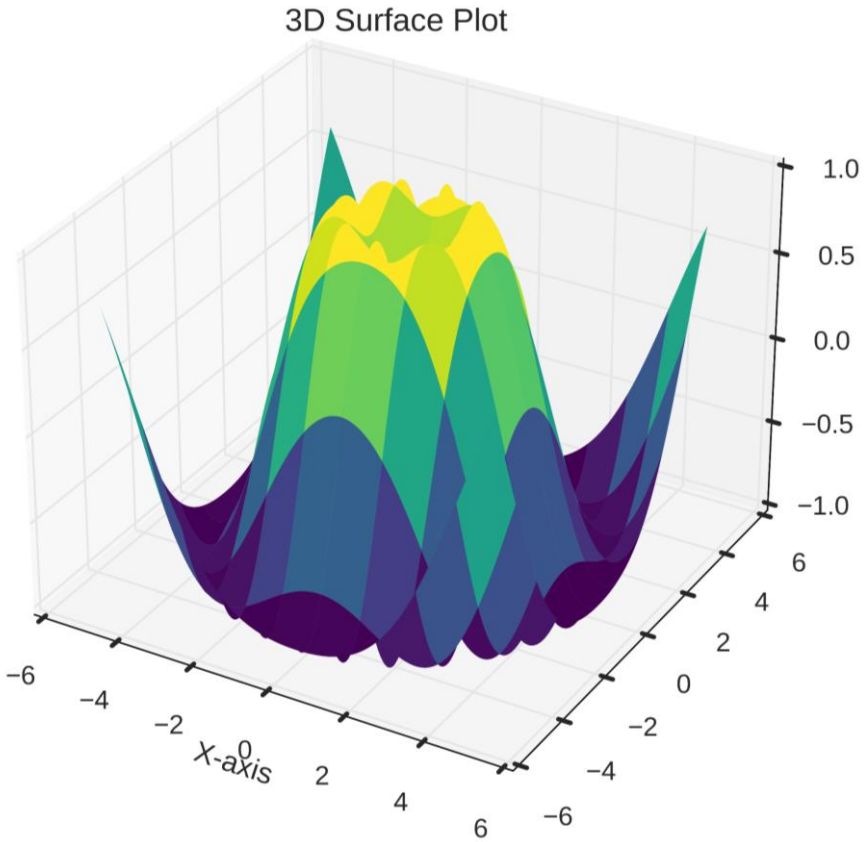
3.1 Scatter Plot

```
[10]  1 # Sample data
      2 x = np.linspace(-5, 5, 100)
      3 y = np.linspace(-5, 5, 100)
      4 X, Y = np.meshgrid(x, y)
      5 z = np.sin(np.sqrt(X**2 + Y**2))
      6
      7 # Create the 3D surface plot
      8 fig = plt.figure(figsize=(10, 7))
      9 ax = fig.add_subplot(111, projection='3d')
     10 ax.plot_surface(X, Y, z, cmap='viridis', edgecolor='none')
     11
     12 ax.set_title('3D Surface Plot')
     13 ax.set_xlabel('X-axis')
     14 plt.savefig('03_3d_SurfacePlot.png',format="png",dpi=800)
     15 plt.show()
```



3.2 3D Surface Plot

```
[10]  1 # Sample data
      2 x = np.linspace(-5, 5, 100)
      3 y = np.linspace(-5, 5, 100)
      4 X, Y = np.meshgrid(x, y)
      5 z = np.sin(np.sqrt(X**2 + Y**2))
      6
      7 # Create the 3D surface plot
      8 fig = plt.figure(figsize=(10, 7))
      9 ax = fig.add_subplot(111, projection='3d')
     10 ax.plot_surface(X, Y, z, cmap='viridis', edgecolor='none')
     11
     12 ax.set_title('3D Surface Plot')
     13 ax.set_xlabel('X-axis')
     14 plt.savefig('03_3d_SurfacePlot.png',format="png",dpi=800)
     15 plt.show()
```



3.3 Customizing 3D Plot

```python
# Customize 3D plots
fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, z, cmap='plasma', edgecolor='k',alpha = 0.7)
ax.set_title('Custom 3D Surface Plot')
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
ax.set_zlabel('Z-axis')
ax.set_zlim(-1, 1)
plt.savefig('03_3d_CustomSurfacePlot.png',format="png",dpi=800)
plt.show()
```