People Dataset :- https://drive.google.com/file/d/13x8f8HNKieSRAzxTIzAojaYp8Up8cefk/view?usp=sharing

1.Demonstrate three different methods for creating identical 2D arrays in NumPy. Provide the code for each method and the final output after each method

```
# Method-1
import numpy as np
array_1 = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print("Method 1: Using np.array()")
print(array_1)

# Method 2
array_2 = np.zeros((3, 3), dtype=int)
array_2[0] = [1, 2, 3]
array_2[1] = [4, 5, 6]
array_2[2] = [7, 8, 9]
print("\nMethod 2: Using np.zeros() and filling it")
print(array_2)

# Method 3
array_3 = np.full((3, 3), fill_value=0)
array_3[0] = [1, 2, 3]
array_3[1] = [4, 5, 6]
array_3[2] = [7, 8, 9]
print("\nMethod 3: Using np.full()")
print(array_3)
```

```
Method 1: Using np.array()
[[1 2 3]
 [4 5 6]
 [7 8 9]]

Method 2: Using np.zeros() and filling it
[[1 2 3]
 [4 5 6]
 [7 8 9]]

Method 3: Using np.full()
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

2.Using the Numpy function, generate an array of 100 evenly spaced numbers between 1 and 10 and Reshape that 1D array into a 2D array

```
array = np.linspace(1, 10, 100).reshape(10, 10)
print(array)
```

```
[[ 1.          1.09090909  1.18181818  1.27272727  1.36363636
1.45454545
   1.54545455  1.63636364  1.72727273  1.81818182]
 [ 1.90909091  2.          2.09090909  2.18181818  2.27272727
2.36363636
   2.45454545  2.54545455  2.63636364  2.72727273]
 [ 2.81818182  2.90909091  3.          3.09090909  3.18181818
3.27272727
   3.36363636  3.45454545  3.54545455  3.63636364]
 [ 3.72727273  3.81818182  3.90909091  4.          4.09090909
4.18181818
   4.27272727  4.36363636  4.45454545  4.54545455]
 [ 4.63636364  4.72727273  4.81818182  4.90909091  5.
5.09090909
   5.18181818  5.27272727  5.36363636  5.45454545]
 [ 5.54545455  5.63636364  5.72727273  5.81818182  5.90909091  6.
   6.09090909  6.18181818  6.27272727  6.36363636]
 [ 6.45454545  6.54545455  6.63636364  6.72727273  6.81818182
6.90909091
   7.          7.09090909  7.18181818  7.27272727]
 [ 7.36363636  7.45454545  7.54545455  7.63636364  7.72727273
7.81818182
   7.90909091  8.          8.09090909  8.18181818]
 [ 8.27272727  8.36363636  8.45454545  8.54545455  8.63636364
8.72727273
   8.81818182  8.90909091  9.          9.09090909]
 [ 9.18181818  9.27272727  9.36363636  9.45454545  9.54545455
9.63636364
   9.72727273  9.81818182  9.90909091 10.        ]]
```

3. Explain the following terms:a.)The difference in np.array, np.asarray and np.asanyarray b.)The difference between Deep copy and shallow copy

```
import pandas as pd
df_sample = pd.read_csv('./People Data.csv', nrows=5)
print(df_sample.columns)
```

```
Index(['Index', 'User Id', 'First Name', 'Last Name', 'Gender',
'Email',
       'Phone', 'Date of birth', 'Job Title', 'Salary'],
      dtype='object')
```

4.Generate a 3x3 array with random floating-point numbers between 5 and 20. Then, round each number in the array to 2 decimal places.

```
array_random = np.random.uniform(5, 20, (3, 3))
array_rounded = np.round(array_random, 2)
print("Original Array:\n", array_random)
print("Rounded Array:\n", array_rounded)

Original Array:
 [[15.54711008  7.53148098  5.12692451]
 [12.06510083 18.0682046   8.74240213]
 [12.87074081  9.64034747  5.95632442]]
Rounded Array:
 [[15.55  7.53  5.13]
 [12.07 18.07  8.74]
 [12.87  9.64  5.96]]
```

5. Create a NumPy array with random integers between 1 and 10 of shape (5, 6). After creating the array perform the following operations: a)Extract all even integers from array. b)Extract all odd integers from array

```
array_int = np.random.randint(1, 11, (5, 6))
print("Original Array:\n", array_int)
even_integers = array_int[array_int % 2 == 0]
print("Even Integers:\n", even_integers)
odd_integers = array_int[array_int % 2 != 0]
print("Odd Integers:\n", odd_integers)

Original Array:
 [[ 8 10  7  6  7  2]
 [ 9  8  4  3 10  9]
 [ 6 10  8  2  4  1]
 [ 3  5  3  2  9 10]
 [ 5  3  5  6  7  5]]
Even Integers:
 [ 8 10  6  2  8  4 10  6 10  8  2  4  2 10  6]
Odd Integers:
 [7 7 9 3 9 1 3 5 3 9 5 3 5 7 5]
```

6.Create a 3D NumPy array of shape (3, 3, 3) containing random integers between 1 and 10. Perform the following operations: a) Find the indices of the maximum values along each depth level (third axis). b) Perform element-wise multiplication of between both array

```
array_3d = np.random.randint(1, 11, (3, 3, 3))
print("3D Array:\n", array_3d)
max_indices = np.argmax(array_3d, axis=2)
print("Indices of Max Values:\n", max_indices)
element_wise_multiplication = array_3d * array_3d
print("Element-wise Multiplication:\n", element_wise_multiplication)

3D Array:
 [[[ 6  8  7]
```

```
  [ 8  1  4]
  [ 2  8  8]]

 [[ 1  3  8]
  [ 6  4  3]
  [ 5  7 10]]

 [[ 5  8  1]
  [ 4  3  7]
  [ 7  9  7]]]
Indices of Max Values:
 [[1 0 1]
 [2 0 2]
 [1 2 1]]
Element-wise Multiplication:
 [[[ 36  64  49]
  [ 64   1  16]
  [  4  64  64]]

 [[  1   9  64]
  [ 36  16   9]
  [ 25  49 100]]

 [[ 25  64   1]
  [ 16   9  49]
  [ 49  81  49]]]
```

7. Clean and transform the 'Phone' column in the sample dataset to remove non-numeric characters and convert it to a numeric data type. Also display the table attributes and data types of each column

```
df = pd.read_csv('./People Data.csv')
filtered_df = df[(df['Last Name'].str.contains("Duke")) &
                 (df['Gender'] == "Female") &
                 (df['Salary'] < 85000)]
print(filtered_df)

     Index          User Id First Name Last Name  Gender  \
45      46  99A502C175C4EBd     Olivia      Duke  Female
210    211  DF17975CC0a0373    Katrina      Duke  Female
457    458  dcE1B7DE83c1076      Traci      Duke  Female
729    730  c9b482D7aa3e682     Lonnie      Duke  Female

                        Email                  Phone Date of birth  \
45        diana26@example.net  001-366-475-8607x04350    13-10-1934
210       robin78@example.com            740.434.0212    21-09-1935
457  perryhoffman@example.org     +1-903-596-0995x489    11-02-1997
729   kevinkramer@example.net            982.692.6257    12-05-2015

           Job Title  Salary
```

```
45            Dentist   60000
210   Producer, radio   50000
457         Herbalist   50000
729      Nurse, adult   70000
```

8.Perform the following tasks using people dataset: a) Read the 'data.csv' file using pandas, skipping the first 50 rows. b) Only read the columns: 'Last Name', ‘Gender’,’Email’,‘Phone’ and ‘Salary’ from the file. c) Display the first 10 rows of the filtered dataset. d) Extract the ‘Salary’' column as a Series and display its last 5 values

```
random_series = pd.Series(np.random.randint(1, 7, 35))
df_7x5 = random_series.values.reshape(7, 5)
df_final = pd.DataFrame(df_7x5)
print(df_final)

   0  1  2  3  4
0  2  1  6  5  2
1  2  2  5  6  6
2  3  3  3  1  1
3  3  4  1  1  3
4  2  4  5  6  3
5  3  2  6  3  3
6  5  1  2  3  3
```

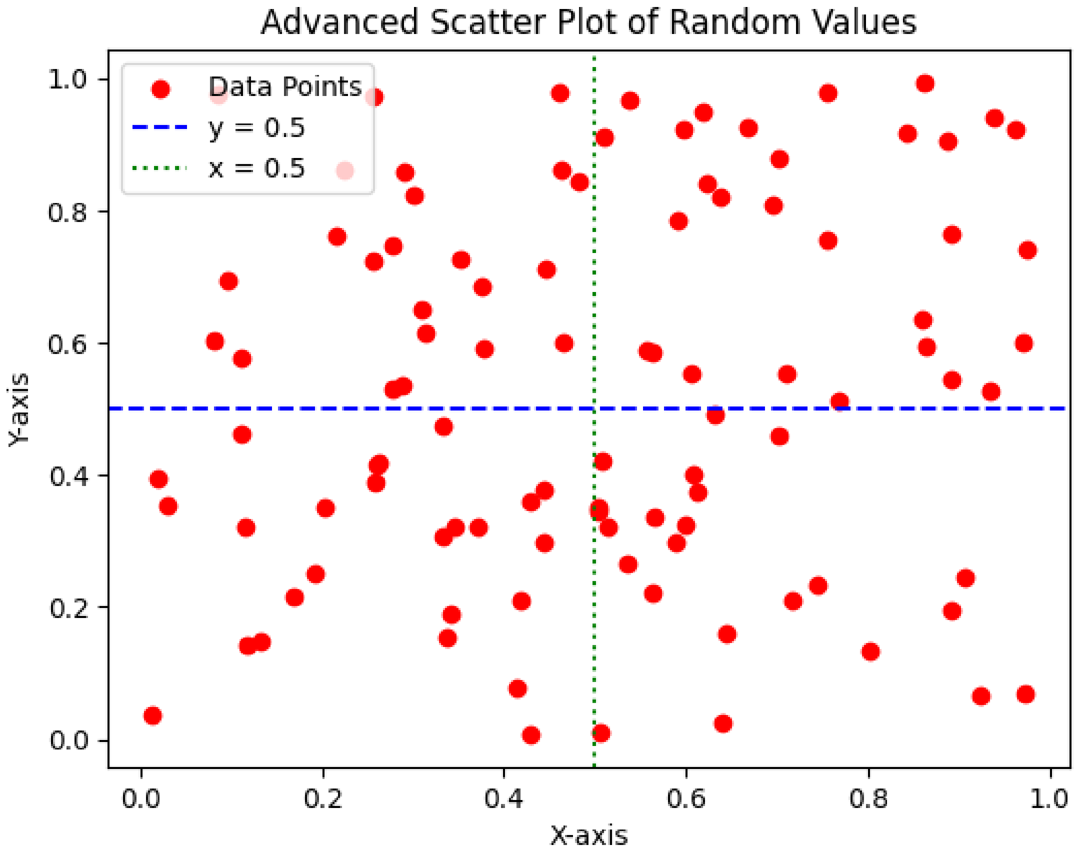
9. Filter and select rows from the People_Dataset, where the “Last Name' column contains the name 'Duke','Gender' column contains the word Female and ‘Salary’ should be less than 85000

```
series1 = pd.Series(np.random.randint(10, 51, 50))
series2 = pd.Series(np.random.randint(100, 1001, 50))
df_combined = pd.DataFrame({'col1': series1, 'col2': series2})
print(df_combined.head())

   col1  col2
0    15   502
1    13   810
2    20   403
3    34   241
4    17   865
```

11.Create two different Series, each of length 50, with the following criteria: a)The first Series should contain random numbers ranging from 10 to 50.b)The second Series should contain random numbers ranging from 100 to 1000. c)Create a DataFrame by joining these Series by column, and, change the names of the columns to 'col1', 'col2',etc

```
import matplotlib.pyplot as plt
x = np.random.rand(100)
y = np.random.rand(100)
```

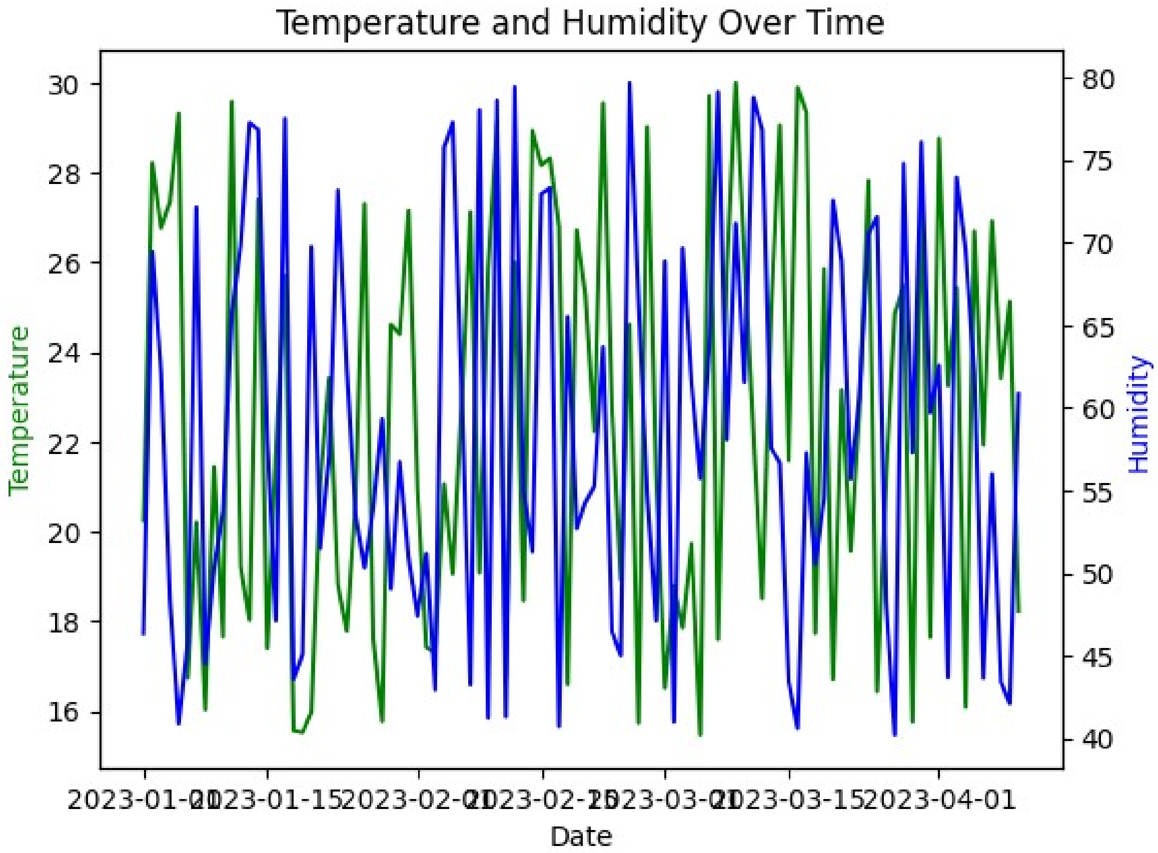
```
plt.scatter(x, y, color='red', marker='o', label='Data Points')
plt.axhline(0.5, color='blue', linestyle='--', label='y = 0.5')
plt.axvline(0.5, color='green', linestyle=':', label='x = 0.5')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Advanced Scatter Plot of Random Values')
plt.legend()
plt.show()
```



12. Perform the following operations using people data set:a) Delete the 'Email', 'Phone', and 'Date of birth' columns from the dataset.b) Delete the rows containing any missing values.d) Print the final output also

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
date_range = pd.date_range(start='2023-01-01', periods=100, freq='D')
temperature = np.random.uniform(15, 30, size=100)
humidity = np.random.uniform(40, 80, size=100)
df_time_series = pd.DataFrame({'Date': date_range, 'Temperature':
temperature, 'Humidity': humidity})
fig, ax1 = plt.subplots()
```

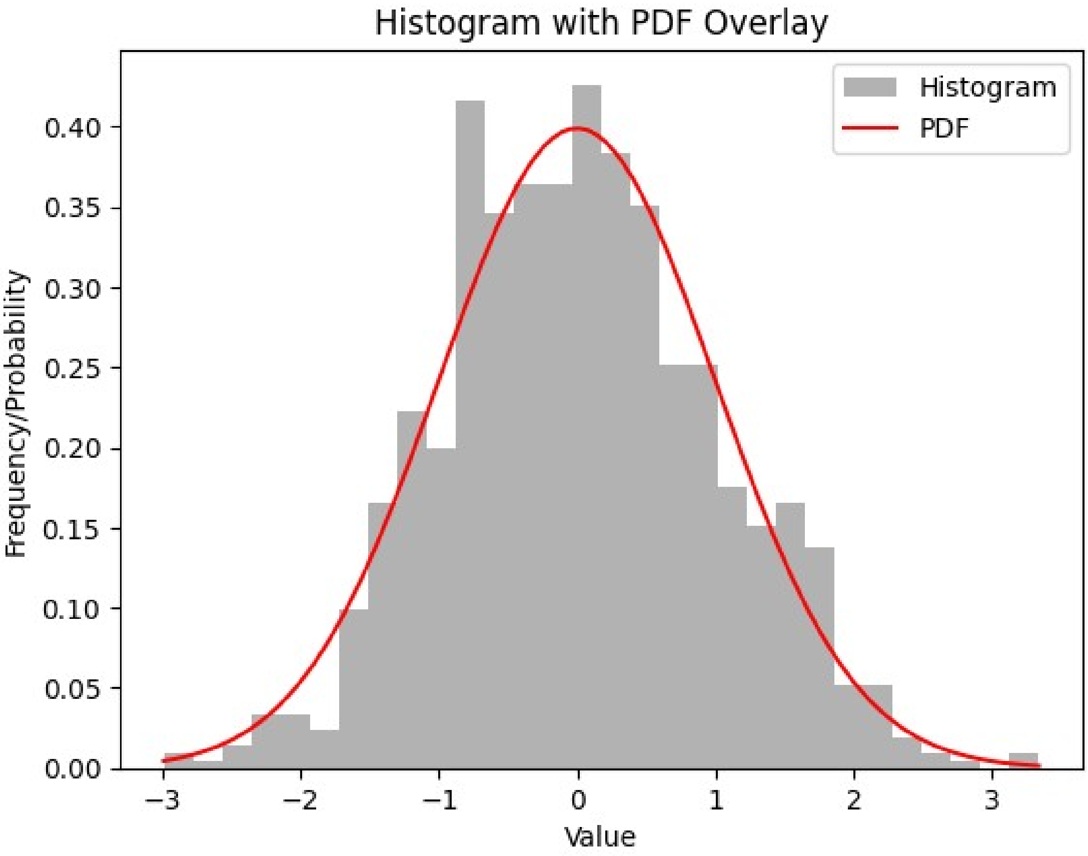
```
ax2 = ax1.twinx()
ax1.plot(df_time_series['Date'], df_time_series['Temperature'], 'g-',
label='Temperature')
ax2.plot(df_time_series['Date'], df_time_series['Humidity'], 'b-',
label='Humidity')
ax1.set_xlabel('Date')
ax1.set_ylabel('Temperature', color='g')
ax2.set_ylabel('Humidity', color='b')
plt.title('Temperature and Humidity Over Time')
plt.show()
```



13. Create two NumPy arrays, x and y, each containing 100 random float values between 0 and 1. Perform the following tasks using Matplotlib and NumPy: a) Create a scatter plot using x and y, setting the color of the points to red and the marker style to 'o'. b) Add a horizontal line at y = 0.5 using a dashed line style and label it as 'y = 0.5'. c) Add a vertical line at x = 0.5 using a dotted line style and label it as 'x = 0.5'. d) Label the x-axis as 'X-axis' and the y-axis as 'Y-axis'. e) Set the title of the plot as 'Advanced Scatter Plot of Random Values'. f) Display a legend for the scatter plot, the horizontal line, and the vertical line.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
```

```
data = np.random.normal(loc=0, scale=1, size=1000)
pdf_x = np.linspace(min(data), max(data), 100)
pdf_y = norm.pdf(pdf_x, loc=0, scale=1)
plt.hist(data, bins=30, density=True, alpha=0.6, color='gray',
label='Histogram')
plt.plot(pdf_x, pdf_y, 'r-', label='PDF')
plt.xlabel('Value')
plt.ylabel('Frequency/Probability')
plt.title('Histogram with PDF Overlay')
plt.legend()
plt.show()
```



14.Create a time-series dataset in a Pandas DataFrame with columns: 'Date', 'Temperature', 'Humidity' and Perform the following tasks using Matplotlib: a) Plot the 'Temperature' and 'Humidity' on the same plot with different y-axes (left y-axis for 'Temperature' and right y-axis for 'Humidity'). b) Label the x-axis as 'Date'. c) Set the title of the plot as 'Temperature and Humidity Over Time'

```
from bokeh.plotting import figure, show, output_notebook
import numpy as np
output_notebook()
x = np.linspace(0, 4 * np.pi, 100)
```

```
y = np.sin(x)
p = figure(title="Sine Wave Function", x_axis_label="X",
y_axis_label="Y")
p.line(x, y, legend_label="Sine Wave", line_width=2)
p.grid.visible = True
show(p)

""

""
```

15. Create a NumPy array data containing 1000 samples from a normal distribution. Perform the following tasks using Matplotlib: a) Plot a histogram of the data with 30 bins. b) Overlay a line plot representing the normal distribution's probability density function (PDF). c) Label the x-axis as 'Value' and the y-axis as 'Frequency/Probability'. d) Set the title of the plot as 'Histogram with PDF Overlay'.

```
from bokeh.plotting import figure, show, output_notebook
import numpy as np
output_notebook()
x = np.linspace(0, 4 * np.pi, 100)
y = np.sin(x)
p = figure(title="Sine Wave Function", x_axis_label="X",
y_axis_label="Y")
p.line(x, y, legend_label="Sine Wave", line_width=2)
p.grid.visible = True
show(p)

""

""
```

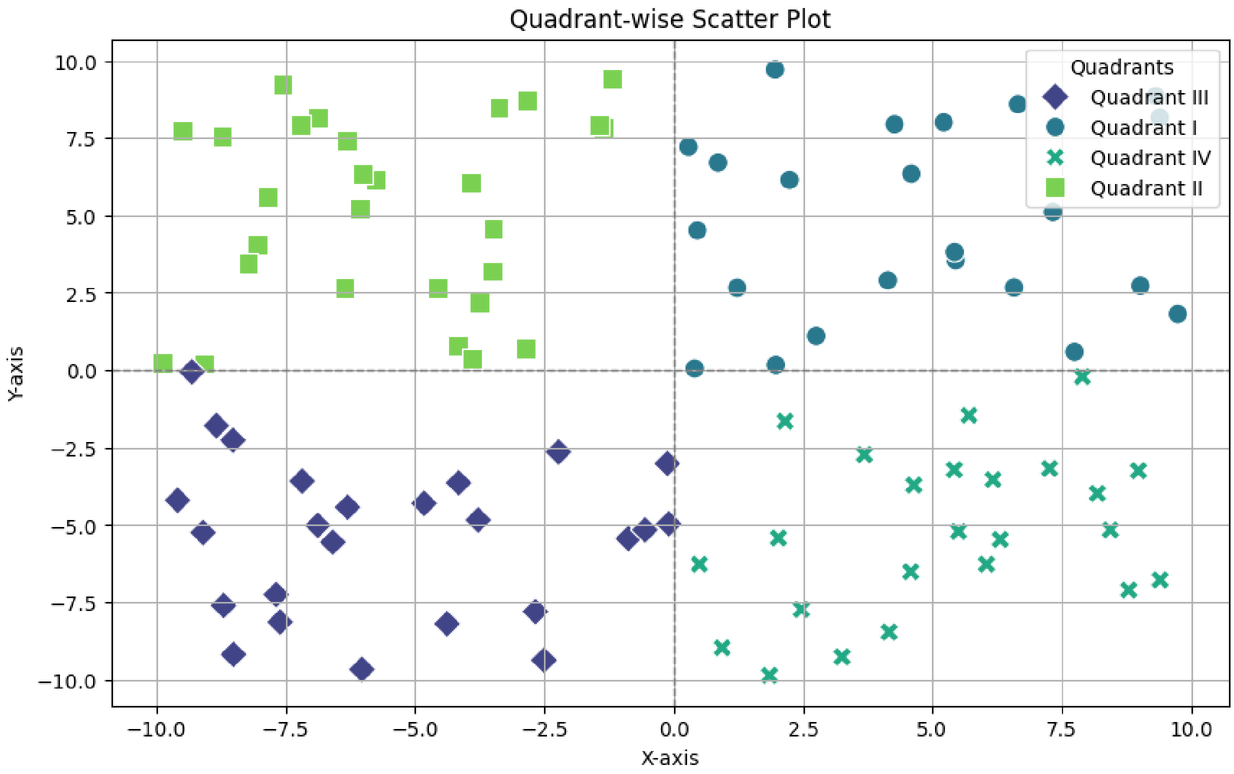
16. Set the title of the plot as 'Histogram with PDF Overlay'.

```
from bokeh.io import output_notebook
from bokeh.plotting import figure, show
from bokeh.models import ColumnDataSource, HoverTool
import pandas as pd
import numpy as np
output_notebook()
categories = ["Category A", "Category B", "Category C", "Category D",
"Category E"]
values = np.random.randint(10, 50, size=5)
df = pd.DataFrame({'category': categories, 'value': values})
source = ColumnDataSource(df)
p = figure(x_range=categories, title="Random Categorical Bar Chart",
y_axis_label="Value", toolbar_location=None)
p.vbar(x='category', top='value', width=0.5, color="blue",
source=source)
```

```
p.add_tools(HoverTool(tooltips=[("Category", "@category"), ("Value",
"@value")]))
show(p)

""

""
```

17.Create a Seaborn scatter plot of two random arrays, color points based on their position relative to the origin (quadrants), add a legend, label the axes, and set the title as 'Quadrant-wise Scatter Plot

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
np.random.seed(42)
x = np.random.rand(100) * 20 - 10
y = np.random.rand(100) * 20 - 10
data = pd.DataFrame({'X': x, 'Y': y})
def determine_quadrant(row):
    if row['X'] > 0 and row['Y'] > 0:
        return 'Quadrant I'
    elif row['X'] < 0 and row['Y'] > 0:
        return 'Quadrant II'
    elif row['X'] < 0 and row['Y'] < 0:
        return 'Quadrant III'
    elif row['X'] > 0 and row['Y'] < 0:
        return 'Quadrant IV'
    else:
        return 'Origin'
data['Quadrant'] = data.apply(determine_quadrant, axis=1)
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data, x='X', y='Y', hue='Quadrant',
palette='viridis', style='Quadrant', markers={'Quadrant I': 'o',
'Quadrant II': 's', 'Quadrant III': 'D', 'Quadrant IV': 'X'}, s=100)
plt.legend(title='Quadrants')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Quadrant-wise Scatter Plot')
plt.grid()
plt.axhline(0, color='gray', lw=1, ls='--')
plt.axvline(0, color='gray', lw=1, ls='--')
plt.show()
```



18. With Bokeh, plot a line chart of a sine wave function, add grid lines, label the axes, and set the title as 'SineWave Function'.

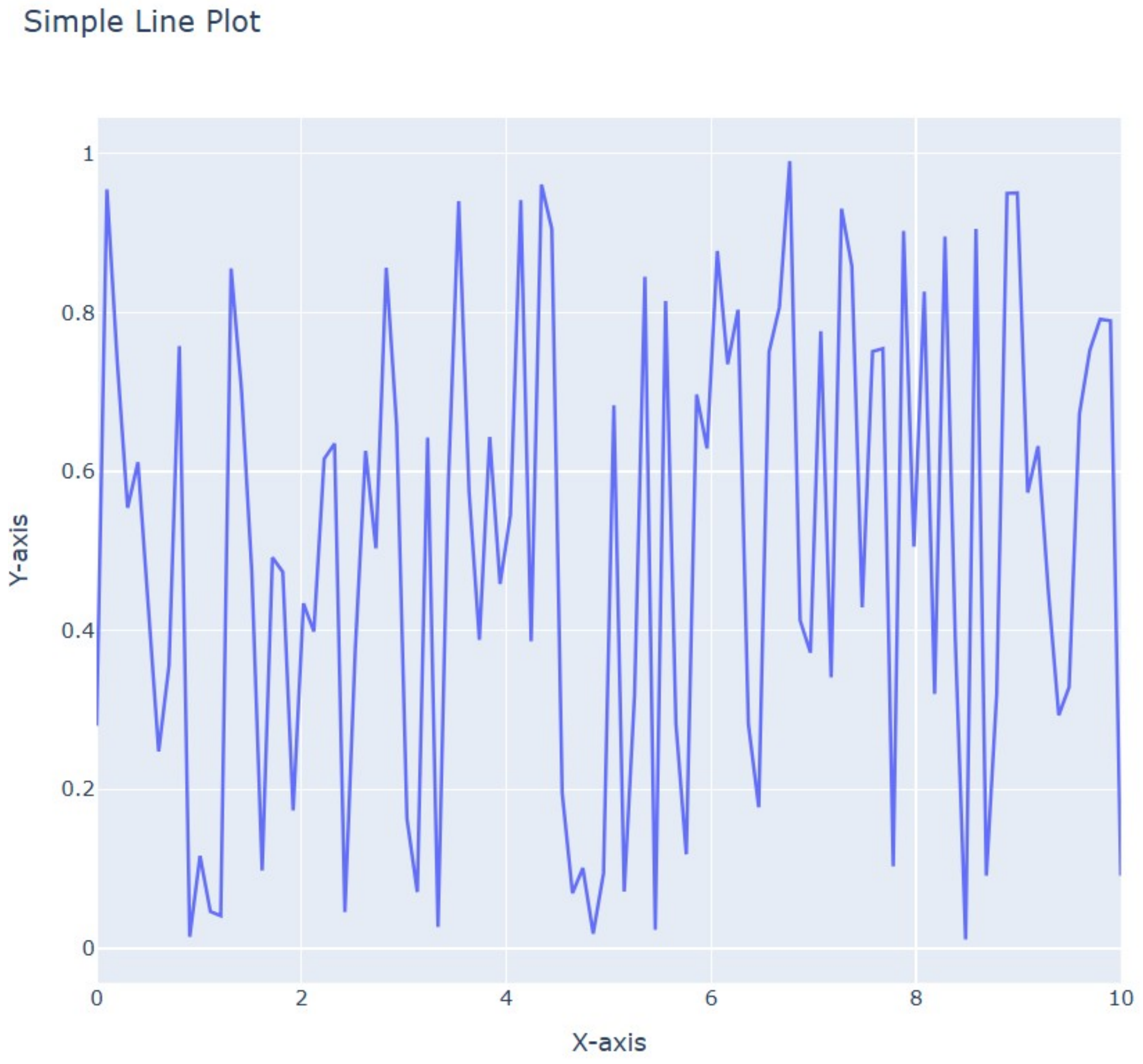
```
from math import pi
import numpy as np
from bokeh.plotting import figure, show
from bokeh.io import output_notebook
output_notebook()
x = np.linspace(0, 2 * pi, 100)
y = np.sin(x)
p = figure(title='Sine Wave Function', x_axis_label='X',
y_axis_label='Sin(X)', width=800, height=400)
p.line(x, y, legend_label='sin(x)', line_width=2, color='blue')
p.grid.grid_line_alpha = 0.5
show(p)

""

""
```

20.Using Plotly, create a basic line plot of a randomly generated dataset, label the axes, and set the title as 'Simple Line Plot'.

```
import numpy as np
import plotly.graph_objects as go
from plotly.offline import init_notebook_mode, plot
```

```python
init_notebook_mode(connected=True)
x = np.linspace(0, 10, 100)
y = np.random.rand(100)
fig = go.Figure()
fig.add_trace(go.Scatter(x=x, y=y, mode='lines', name='Random Data'))
fig.update_layout(
    title='Simple Line Plot',
    xaxis_title='X-axis',
    yaxis_title='Y-axis',
    template='plotly'
)
plot(fig)
```

```
'temp-plot.html'
```


Simple Line Plot
Y-axis
X-axis