ML (CSE(AIML) - 5th Semester)

SUPRATIM NAG/AIML/22/057

Assignemnt 1(b)-- PART-2

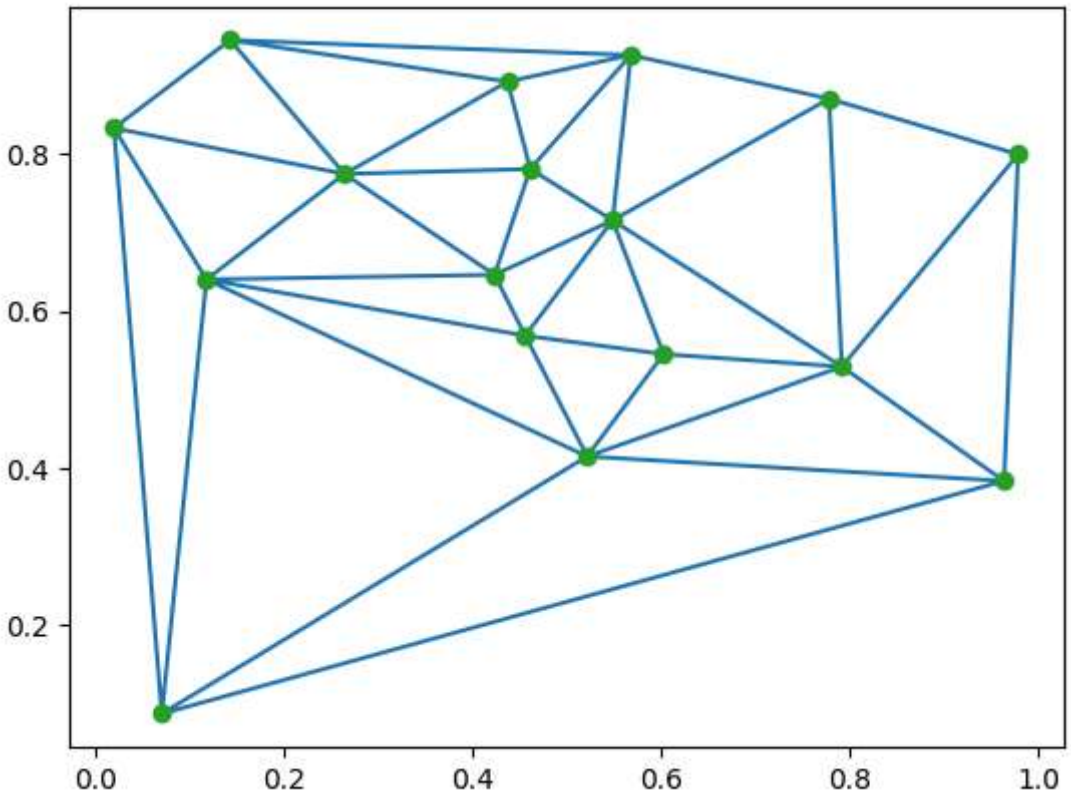
Q-1: What is Triangulation? Write a python code to perform a Delaunay triangulation on a set of 2D points and visualize the resulting triangulation.

Ans: Triangulation is the division of a surface or plane polygon into a set of triangles. Usually, two adjacent triangles entirely share each triangle vertex.The concept of triangulation makes the plotting of a continuous surface possible in python. Either triangle coordinates can be defined by the user or by using Delaunay triangulation which covers the set of points in a triangle that makes the surface plot persistent.

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.spatial import Delaunay

points = np.random.rand(17, 2)
# points
tri = Delaunay(points)

plt.triplot(points[:,0], points[:,1], tri.simplices.copy())
plt.plot(points[:,0], points[:,1], 'o')
plt.show()
```

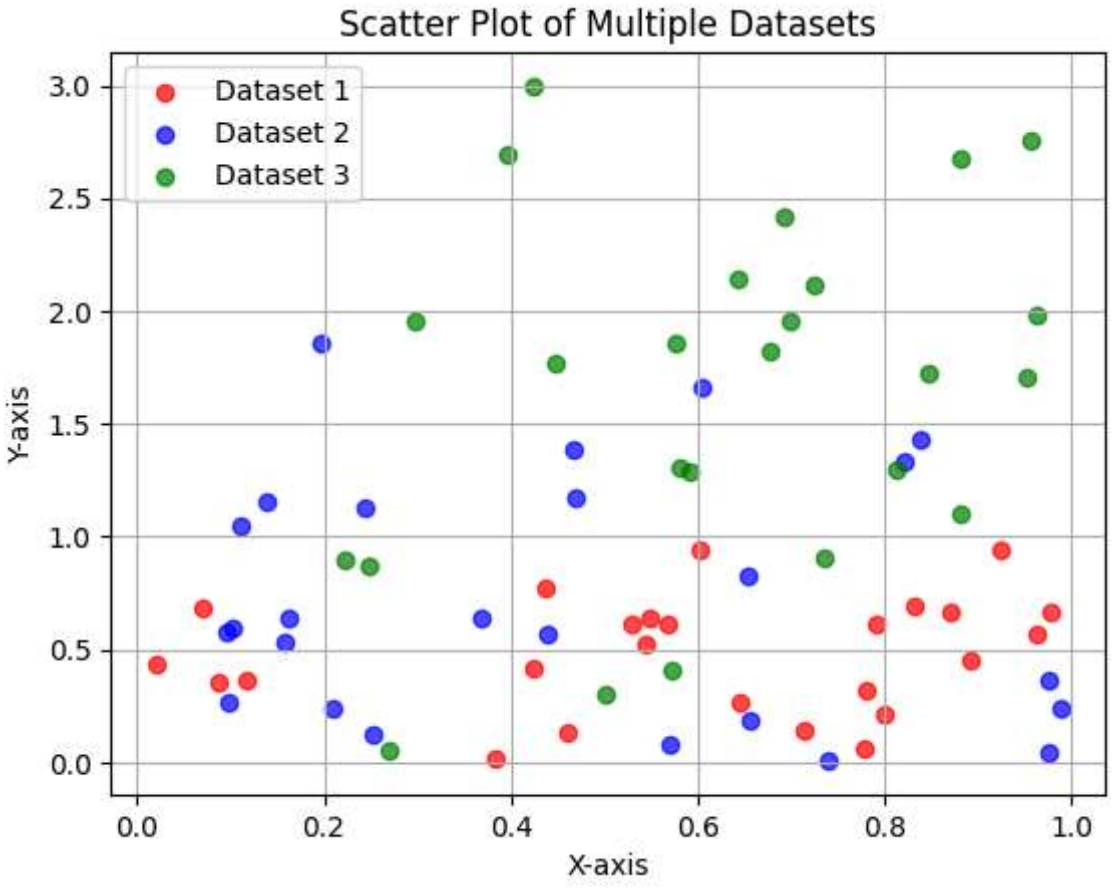


Q-2: Plot the Scatter Plot on multiple datasets.

In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np

np.random.seed(0)
x1, y1 = np.random.rand(25), np.random.rand(25)
x2, y2 = np.random.rand(25), np.random.rand(25) * 2
x3, y3 = np.random.rand(25), np.random.rand(25) * 3

plt.scatter(x1, y1, color='red', label='Dataset 1', alpha=0.7)
plt.scatter(x2, y2, color='blue', label='Dataset 2', alpha=0.7)
plt.scatter(x3, y3, color='green', label='Dataset 3', alpha=0.7)
plt.title('Scatter Plot of Multiple Datasets')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid(True)
plt.show()
```

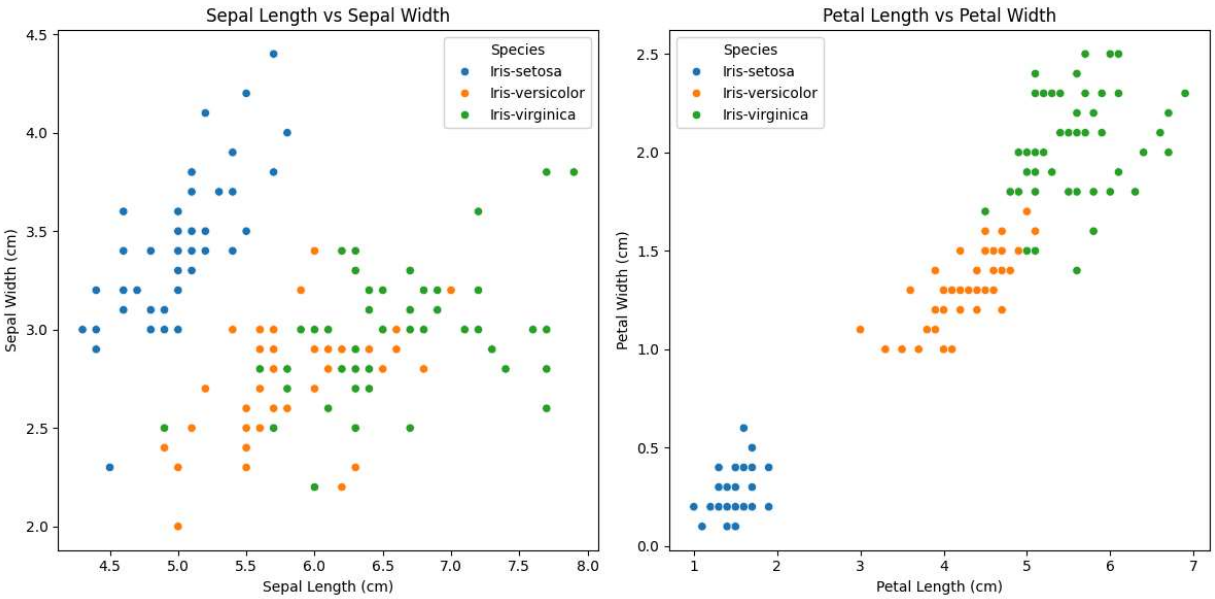


Q-3: Perform Scatter Plot on iris dataset.

Q-4: Save the figure generated in Q-3 in a pdf file.

In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# Load the dataset
file_path = 'DSD Datasets/Iris.csv'
df = pd.read_csv(file_path)
plt.figure(figsize=(12, 6))
# Sepal Length vs Sepal Width
plt.subplot(1, 2, 1)
sns.scatterplot(data=df, x='SepalLengthCm', y='SepalWidthCm', hue='Species')
plt.title('Sepal Length vs Sepal Width')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
# Petal Length vs Petal Width
plt.subplot(1, 2, 2)
sns.scatterplot(data=df, x='PetalLengthCm', y='PetalWidthCm', hue='Species')
plt.title('Petal Length vs Petal Width')
plt.xlabel('Petal Length (cm)')
plt.ylabel('Petal Width (cm)')
plt.tight_layout()
# Save the figure as a PDF file
pdf_file_path = 'scatter_plots.pdf'
```

```
plt.savefig(pdf_file_path, format='pdf')
# to sjow the plot
plt.show()
print(f"Figure saved as {pdf_file_path}")
```



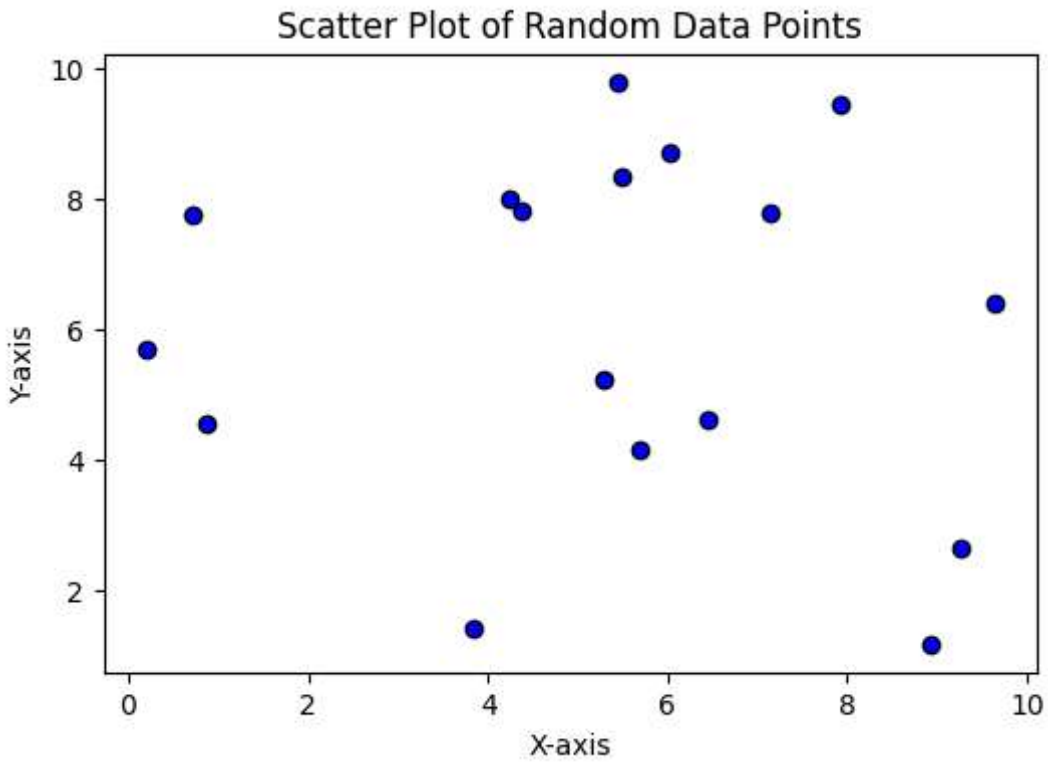

```
Figure saved as scatter_plots.pdf
```

Q-5: Generate 10 data points. Plot scatter plot on the points and save the generated figure in a pdf file.

In [ ]:

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(0)
x = np.random.rand(17) * 10
y = np.random.rand(17) * 10
# Create a scatter plot
plt.figure(figsize=(6, 4))
plt.scatter(x, y, color='blue', marker='o', edgecolor='k')
# Customize the plot
plt.title('Scatter Plot of Random Data Points')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
# Save the figure as a PDF file
pdf_file_path = 'scatter_plot_random_data.pdf'
plt.savefig(pdf_file_path, format='pdf')
# to show the plot
plt.show()
print(f"Figure saved as {pdf_file_path}")
```




```
Figure saved as scatter_plot_random_data.pdf
```

Q-6: Write a python code to show the use of Marker and styles in charts.

In [ ]:

```
import numpy as np
import matplotlib.pyplot as plt
# Generate sample data
np.random.seed(0)
x = np.linspace(0, 10, 10)
y1 = np.sin(x)
y2 = np.cos(x)
# Create a figure and a set of subplots
plt.figure(figsize=(12, 6))
# Scatter plot with different markers and colors
plt.subplot(1, 2, 1)
plt.scatter(x, y1, color='blue', marker='o', label='Circle')
plt.scatter(x + 1, y2, color='red', marker='^', label='Triangle')
plt.scatter(x + 2, y1 + 1, color='green', marker='s', label='Square')
plt.scatter(x + 3, y2 + 1, color='purple', marker='D', label='Diamond')
plt.title('Scatter Plot with Different Markers')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
# Line plot with different line styles
plt.subplot(1, 2, 2)
plt.plot(x, y1, color='blue', linestyle='-', marker='o', label='Solid Line with Cir
plt.plot(x, y2, color='red', linestyle='--', marker='^', label='Dashed Line with Tr
plt.plot(x, y1 + 1, color='green', linestyle='-.', marker='s', label='Dash-dot Line
plt.plot(x, y2 + 1, color='purple', linestyle=':', marker='D', label='Dotted Line w
plt.title('Line Plot with Different Line Styles and Markers')
plt.xlabel('X-axis')
```

```
plt.ylabel('Y-axis')
plt.legend()
# Adjust layout
plt.tight_layout()
# Save the figure as a PDF file
# pdf_file_path = 'marker_styles_and_line_styles.pdf'
# plt.savefig(pdf_file_path, format='pdf')
# Show the plot
plt.show()
# print(f"Figure saved as {pdf_file_path}")
```