**Assignment No-4**

**Set A**

**1.Generate a random array of 50 integers and display them using a line chart, scatter plot, histogram and box plot. Apply appropriate color, labels and styling options.**

import numpy as np

import matplotlib.pyplot as plt

# 1. Generate random array of 50 integers between 1 and 100

data = np.random.randint(1, 101, 50)

# Create a figure with 4 subplots

plt.figure(figsize=(12, 10))

# Line chart

plt.subplot(2, 2, 1)

plt.plot(data, marker='o', color='blue', linestyle='-', linewidth=1.5)

plt.title('Line Chart', fontsize=14)

plt.xlabel('Index')

plt.ylabel('Value')

plt.grid(True, linestyle='--', alpha=0.7)

# Scatter plot

plt.subplot(2, 2, 2)

plt.scatter(range(len(data)), data, color='red', edgecolor='black')

plt.title('Scatter Plot', fontsize=14)

plt.xlabel('Index')

plt.ylabel('Value')

plt.grid(True, linestyle=':', alpha=0.7)

# Histogram

plt.subplot(2, 2, 3)

plt.hist(data, bins=8, color='green', edgecolor='black', alpha=0.7)

plt.title('Histogram', fontsize=14)

plt.xlabel('Value Range')

plt.ylabel('Frequency')

# Box plot

plt.subplot(2, 2, 4)

plt.boxplot(data, patch\_artist=True, boxprops=dict(facecolor='orange'))

plt.title('Box Plot', fontsize=14)

plt.ylabel('Value')

# Adjust layout and show

plt.tight\_layout()

plt.show()

Generates: A random list of integers between 1–100.

Plots:

* Line chart → Shows data trends.
* Scatter plot → Shows data distribution.
* Histogram → Groups data into bins to see frequency.
* Box plot → Shows median, quartiles, and outliers.

**2.Add two outliers to the above data and display the box plot.**

import numpy as np

import matplotlib.pyplot as plt

# Original random array of 50 integers

data = np.random.randint(1, 101, 50)

# Add two outliers

data\_with\_outliers = np.append(data, [200, 250]) # extreme high values

# Display box plot

plt.figure(figsize=(6, 5))

plt.boxplot(data\_with\_outliers, patch\_artist=True,

boxprops=dict(facecolor='orange', color='black'),

flierprops=dict(marker='o', color='red', markersize=8))

plt.title('Box Plot with Outliers', fontsize=14)

plt.ylabel('Value')

plt.grid(True, linestyle='--', alpha=0.6)

plt.show()

**3.Create two lists, one representing subject names and the other representing marks obtained in those subjects. Display the data in a pie chart and bar chart.**

import matplotlib.pyplot as plt

# Lists for subjects and marks

subjects = ["Math", "Science", "English", "History", "Computer"]

marks = [85, 90, 78, 88, 95]

# --- Pie Chart ---

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)

plt.pie(marks, labels=subjects, autopct='%1.1f%%', startangle=140, colors=['skyblue', 'lightgreen', 'orange', 'pink', 'yellow'])

plt.title("Marks Distribution - Pie Chart", fontsize=14)

# --- Bar Chart ---

plt.subplot(1, 2, 2)

plt.bar(subjects, marks, color='purple', edgecolor='black')

plt.title("Marks by Subject - Bar Chart", fontsize=14)

plt.xlabel("Subjects")

plt.ylabel("Marks")

plt.ylim(0, 100)

plt.grid(axis='y', linestyle='--', alpha=0.7)

# Show plots

plt.tight\_layout()

plt.show()

**4.Write a Python program to create a Bar plot to get the frequency of the three species of the Iris data.**

import seaborn as sns

import matplotlib.pyplot as plt

# Load the Iris dataset

iris = sns.load\_dataset('iris')

# Count the frequency of each species

species\_count = iris['species'].value\_counts()

# Create the bar plot

plt.figure(figsize=(6, 5))

sns.barplot(x=species\_count.index, y=species\_count.values, palette='viridis')

# Labels and title

plt.title("Frequency of Iris Species", fontsize=14)

plt.xlabel("Species")

plt.ylabel("Frequency")

plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.show()

**5.Write a Python program to create a Pie plot to get the frequency of the three species of the Iris data.**

import seaborn as sns

import matplotlib.pyplot as plt

# Load Iris dataset

iris = sns.load\_dataset('iris')

# Count frequency of each species

species\_count = iris['species'].value\_counts()

# Create pie chart

plt.figure(figsize=(6, 6))

plt.pie(species\_count, labels=species\_count.index, autopct='%1.1f%%',

startangle=140, colors=['skyblue', 'lightgreen', 'orange'])

plt.title("Iris Species Frequency - Pie Chart", fontsize=14)

plt.show()

### 

### **Explanation**

* **value\_counts()** → Counts how many flowers belong to each species.
* **plt.pie()** → Creates a pie chart with percentages.
* **colors** → Custom color scheme for each slice.
* **startangle=140** → Rotates the chart for better visual arrangement.

**6.Write a Python program to create a histogram of the three species of the Iris data.**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

**# Load Iris dataset**

**iris = sns.load\_dataset('iris')**

**# Create histogram for sepal length by species**

**plt.figure(figsize=(8, 6))**

**sns.histplot(data=iris, x='sepal\_length', hue='species', kde=False,**

**multiple='stack', palette='Set2', edgecolor='black')**

**# Labels and title**

**plt.title("Histogram of Sepal Length by Iris Species", fontsize=14)**

**plt.xlabel("Sepal Length (cm)")**

**plt.ylabel("Frequency")**

**plt.grid(axis='y', linestyle='--', alpha=0.7)**

**plt.show()**

### **How it works**

* **hue='species' → Separates the histogram by species.**
* **multiple='stack' → Stacks bars instead of overlapping them.**
* **palette='Set2' → Uses soft distinct colors for each species.**
* **edgecolor='black' → Improves bar visibility.**

**Set B**

**1.Write a Python program to create a graph to find relationship between the petal length**

**and petal width.**

import seaborn as sns

import matplotlib.pyplot as plt

# Load Iris dataset

iris = sns.load\_dataset('iris')

# Create scatter plot

plt.figure(figsize=(8, 6))

sns.scatterplot(data=iris, x='petal\_length', y='petal\_width',

hue='species', palette='Set1', s=80, edgecolor='black')

# Labels and title

plt.title("Relationship between Petal Length and Petal Width", fontsize=14)

plt.xlabel("Petal Length (cm)")

plt.ylabel("Petal Width (cm)")

plt.grid(True, linestyle='--', alpha=0.7)

plt.show()

### **Explanation**

* **hue='species'** → Colors points by species for better distinction.
* **s=80** → Sets point size for better visibility.
* **edgecolor='black'** → Gives points an outline to stand out.
* This scatter plot helps identify **correlation patterns**—for example, *setosa* tends to have smaller petals.

**2.Write a Python program to draw scatter plots to compare two features of the iris dataset.**

import seaborn as sns

import matplotlib.pyplot as plt

# Load Iris dataset

iris = sns.load\_dataset('iris')

# Create scatter plot comparing Sepal Length vs Sepal Width

plt.figure(figsize=(8, 6))

sns.scatterplot(data=iris, x='sepal\_length', y='sepal\_width',

hue='species', palette='Set2', s=80, edgecolor='black')

# Labels and title

plt.title("Sepal Length vs Sepal Width - Iris Dataset", fontsize=14)

plt.xlabel("Sepal Length (cm)")

plt.ylabel("Sepal Width (cm)")

plt.grid(True, linestyle='--', alpha=0.7)

plt.show()

### Explanation

* hue='species' → Colors points by flower species.
* s=80 → Makes points larger for better visibility.
* Set2 palette → Uses soft distinct colors.
* Grid lines help in measuring the relationship visually.

**3.Write a Python program to create box plots to see how each feature i.e. Sepal Length, Sepal Width, Petal Length, Petal Width are distributed across the three species.**

import seaborn as sns

import matplotlib.pyplot as plt

# Load Iris dataset

iris = sns.load\_dataset('iris')

# Create box plots for each feature grouped by species

plt.figure(figsize=(12, 8))

for i, feature in enumerate(['sepal\_length', 'sepal\_width', 'petal\_length', 'petal\_width'], start=1):

plt.subplot(2, 2, i)

sns.boxplot(data=iris, x='species', y=feature, palette='Set3')

plt.title(f"{feature.replace('\_', ' ').title()} by Species")

plt.xlabel("Species")

plt.ylabel(feature.replace('\_', ' ').title())

plt.tight\_layout()

plt.show()

### **How it works**

* **subplot(2, 2, i) → Creates a 2×2 grid of box plots (one for each feature).**
* **replace('\_', ' ').title() → Makes titles look cleaner.**
* **palette='Set3' → Uses soft pastel colors for clarity.**
* **Box plot shows:**
  + **Median (center line inside box)**
  + **Interquartile range (box height)**
  + **Potential outliers (dots outside whiskers)**

**Set C**

**1.Write a Python program to create a pairplot of the iris data set and check which flower species seems to be the most separable.**

import seaborn as sns

import matplotlib.pyplot as plt

# Load Iris dataset

iris = sns.load\_dataset('iris')

# Create pairplot

sns.pairplot(iris, hue='species', palette='Set1', diag\_kind='hist', markers=["o", "s", "D"])

plt.suptitle("Iris Dataset - Pairplot", y=1.02, fontsize=14)

plt.show()

### Explanation

* pairplot → Creates scatter plots for all combinations of features, plus histograms on the diagonal.
* hue='species' → Colors data points by species.
* markers → Assigns different shapes for each species for better visual distinction.
* diag\_kind='hist' → Shows histograms for each individual feature instead of KDE plots.

**2.Write a Python program to generate a box plot to show the Interquartile range and outliers for the three species for each feature.**

import seaborn as sns

import matplotlib.pyplot as plt

# Load Iris dataset

iris = sns.load\_dataset('iris')

# Create subplots for each feature

plt.figure(figsize=(12, 8))

features = ['sepal\_length', 'sepal\_width', 'petal\_length', 'petal\_width']

for i, feature in enumerate(features, start=1):

plt.subplot(2, 2, i)

sns.boxplot(data=iris, x='species', y=feature, palette='Set2')

plt.title(f"{feature.replace('\_', ' ').title()} by Species")

plt.xlabel("Species")

plt.ylabel(feature.replace('\_', ' ').title())

plt.tight\_layout()

plt.show()

### **What you’ll see**

* **Boxes** → Represent the Interquartile Range (Q1–Q3)
* **Line inside box** → Median value
* **Whiskers** → Extend to 1.5×IQR from Q1 and Q3
* **Dots** → Outliers beyond whiskers

**3 Write a Python program to create a join plot using "kde" to describe individual distributions on the same plot between Sepal length and Sepal width. Note: The kernel density estimation (kde) procedure visualizes a bivariate distribution. In seaborn, this kind of plot is shown with a contour plot and is available as a style in joint plot().**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

**# Load the iris dataset**

**iris = sns.load\_dataset('iris')**

**# Create KDE join plot**

**sns.jointplot(**

**data=iris,**

**x="sepal\_length",**

**y="sepal\_width",**

**kind="kde", # KDE contour plot**

**fill=True, # Fill the contours**

**cmap="Blues" # Color map**

**)**

**plt.suptitle("KDE Joint Plot - Sepal Length vs Sepal Width", y=1.02, fontsize=14)**

**plt.show()**

### **Explanation**

* **kind="kde" → Creates a Kernel Density Estimation plot.**
* **fill=True → Fills the contour areas.**
* **cmap="Blues" → Changes the color scheme.**
* **Contours → Represent density levels (thicker lines = higher density).**
* **Marginal KDE plots → Appear on the top and right, showing the 1D distribution of each variable.**