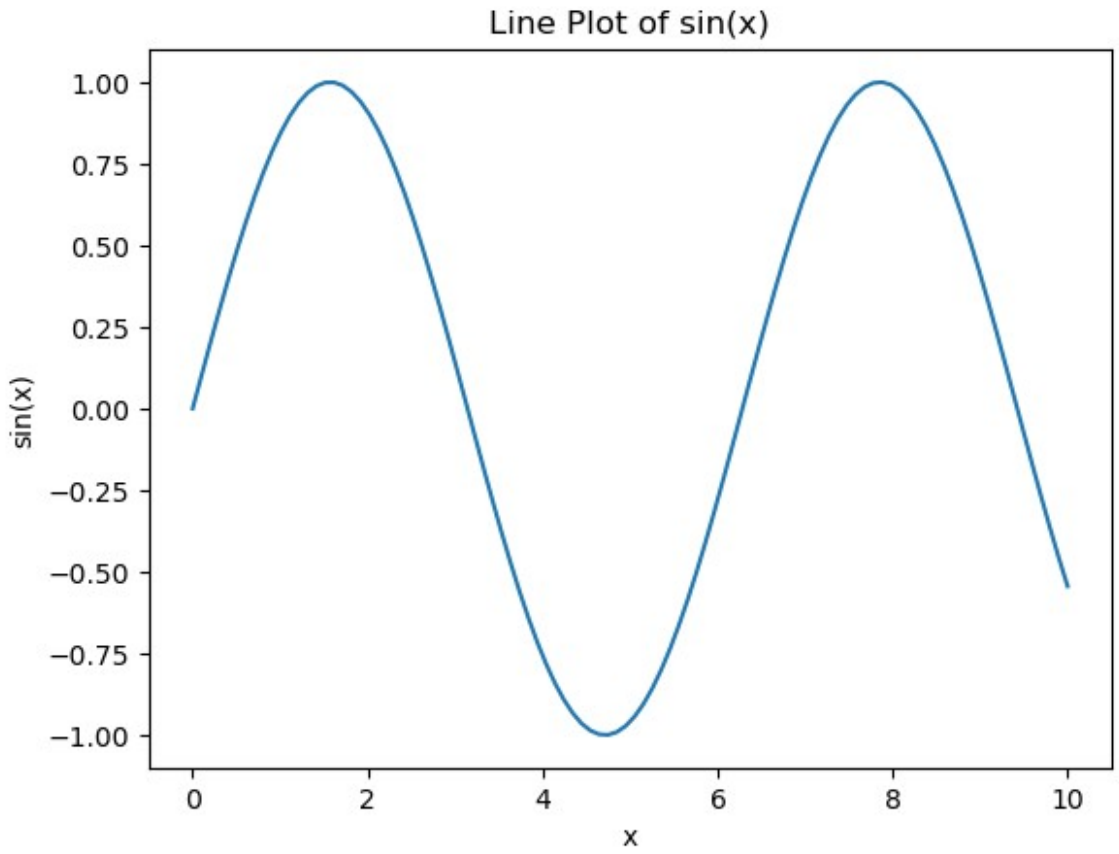
# Introduction to Matplotlib

Matplotlib is a powerful plotting library for Python. It provides a wide array of functionalities to create static, animated, and interactive visualizations. This notebook serves as an introduction to some of the basic and commonly used plotting features of Matplotlib.

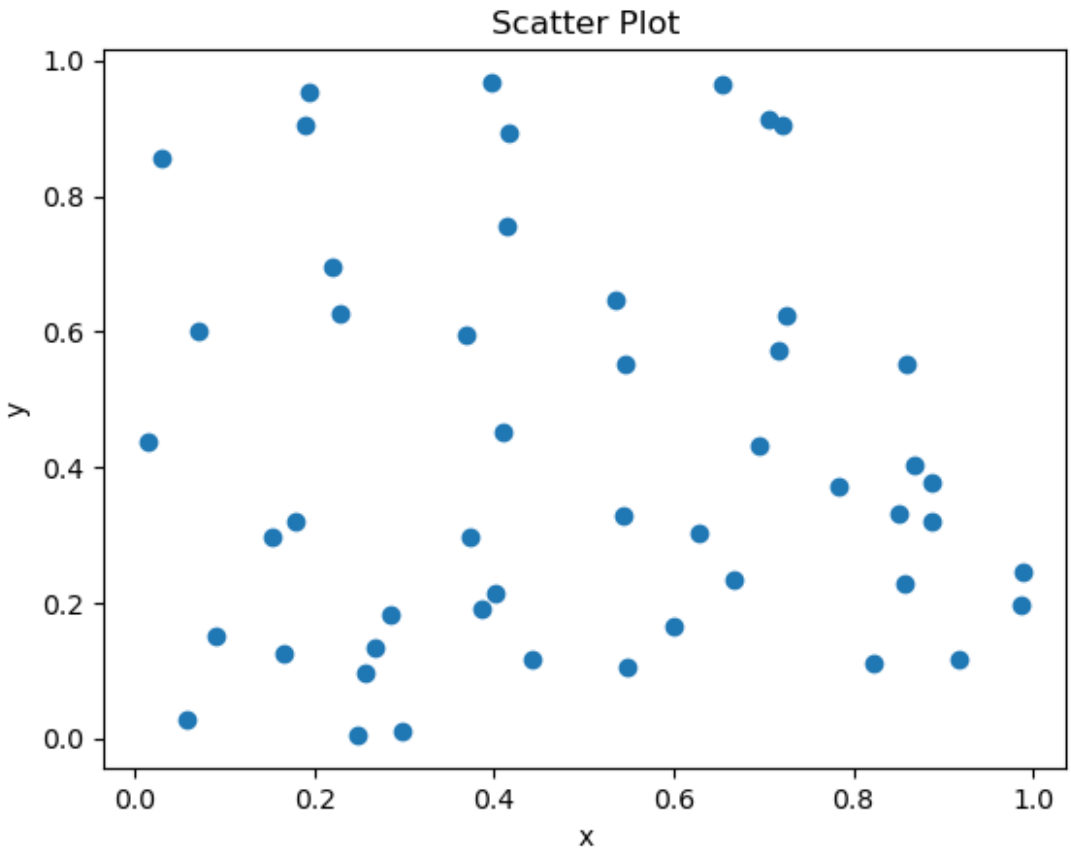
## Basic Plotting with Matplotlib

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

# Generate sample data
x = np.linspace(0, 10, 100)
y = np.sin(x)

# Create a line plot
plt.plot(x, y)
plt.title("Line Plot of sin(x)")
plt.xlabel("x")
plt.ylabel("sin(x)")
plt.show()
```



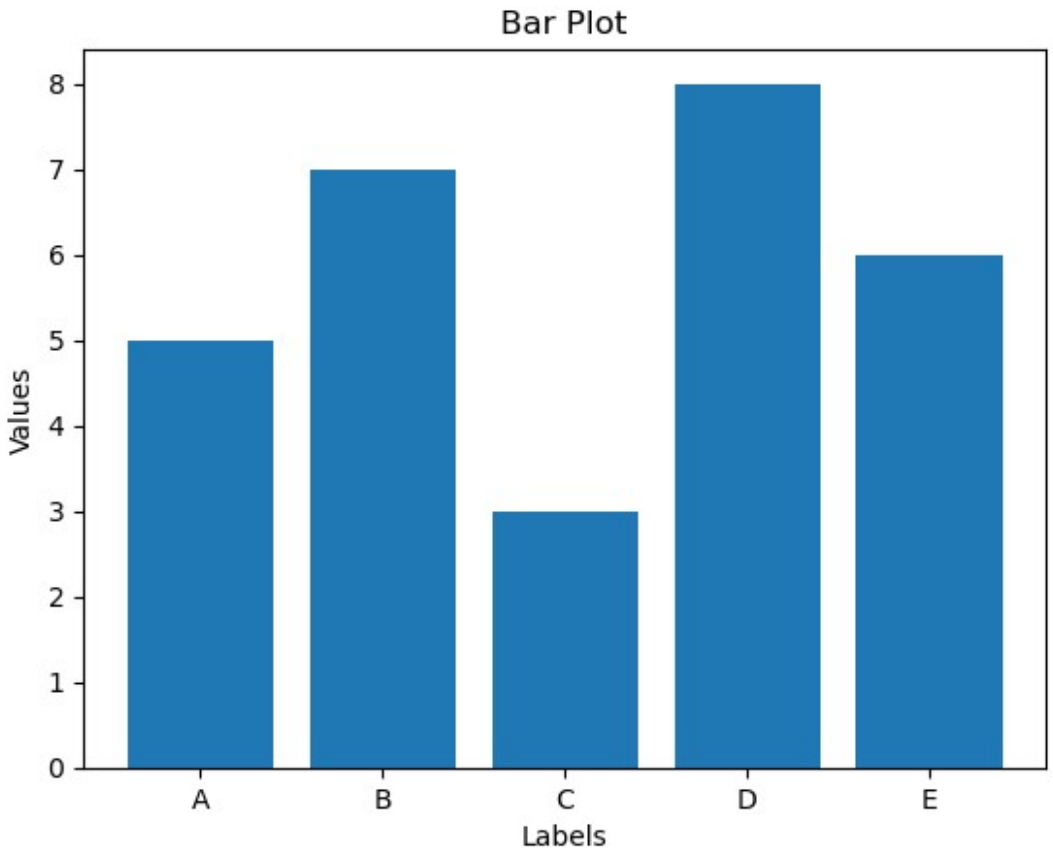

## Scatter Plots

```
# Generate random data for scatter plot
x = np.random.rand(50)
y = np.random.rand(50)

plt.scatter(x, y)
plt.title("Scatter Plot")
plt.xlabel("x")
plt.ylabel("y")
plt.show()
```



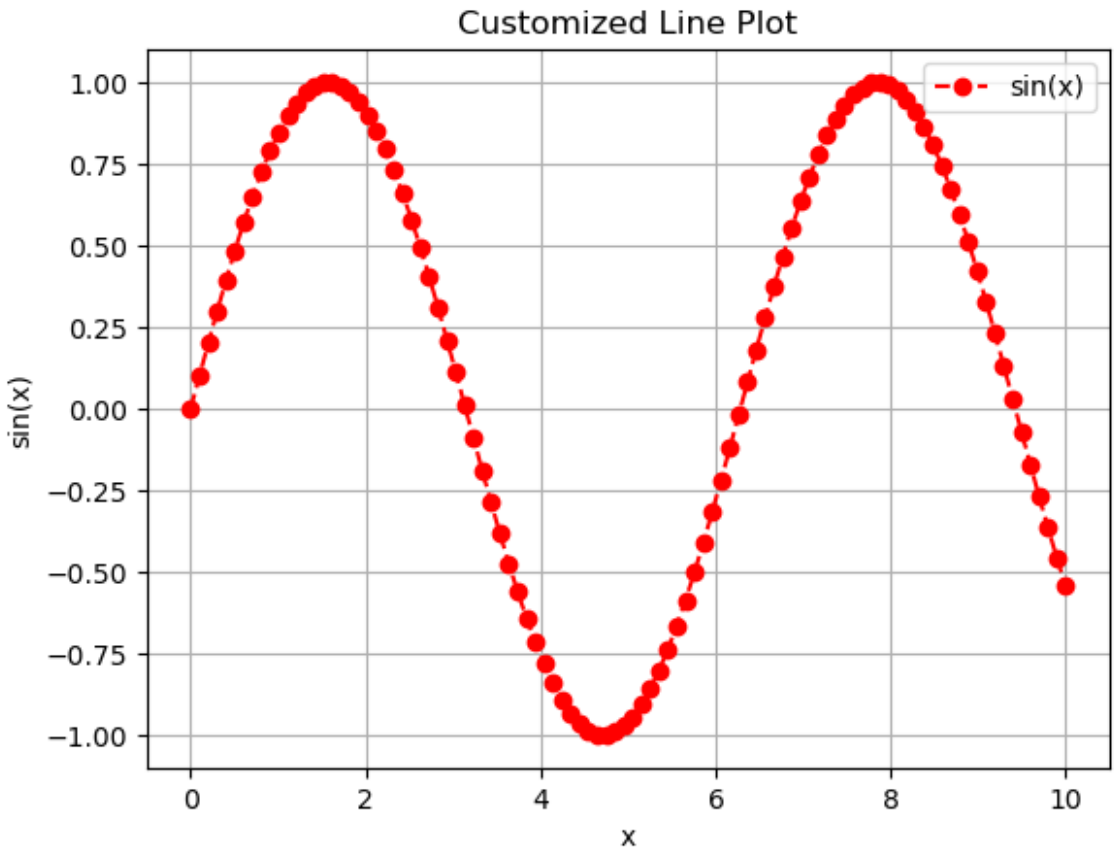

## Bar Plots

```
labels = ['A', 'B', 'C', 'D', 'E']
values = [5, 7, 3, 8, 6]

plt.bar(labels, values)
plt.title("Bar Plot")
plt.xlabel("Labels")
plt.ylabel("Values")
plt.show()
```




## Customizing Your Plots

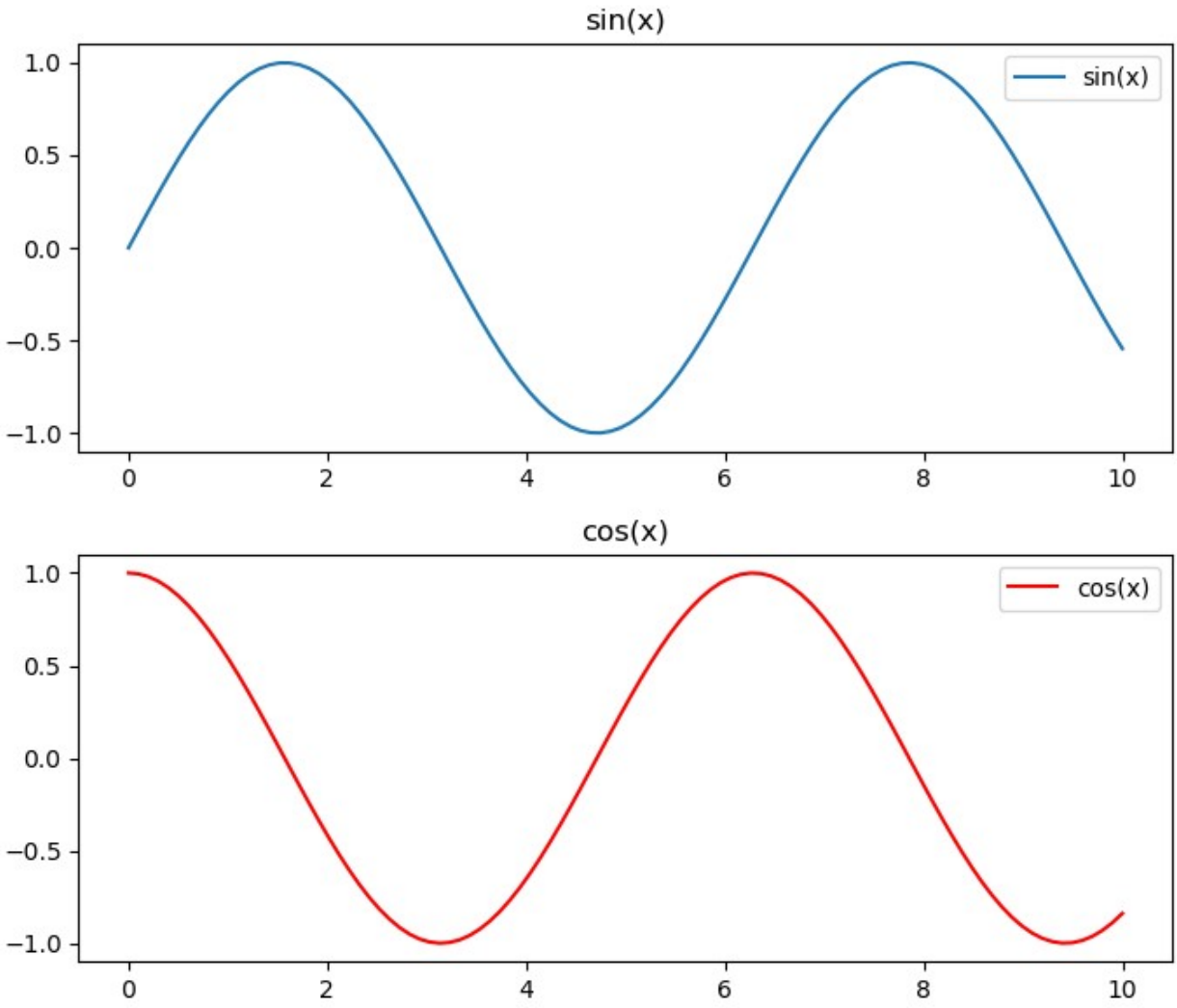
```
x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y, label="sin(x)", color="red", linestyle="--",
marker="o")
plt.title("Customized Line Plot")
plt.xlabel("x")
plt.ylabel("sin(x)")
plt.legend()
plt.grid(True)
plt.show()
```




## Subplots and Multiple Figures

You can create multiple plots in a single figure using subplots.

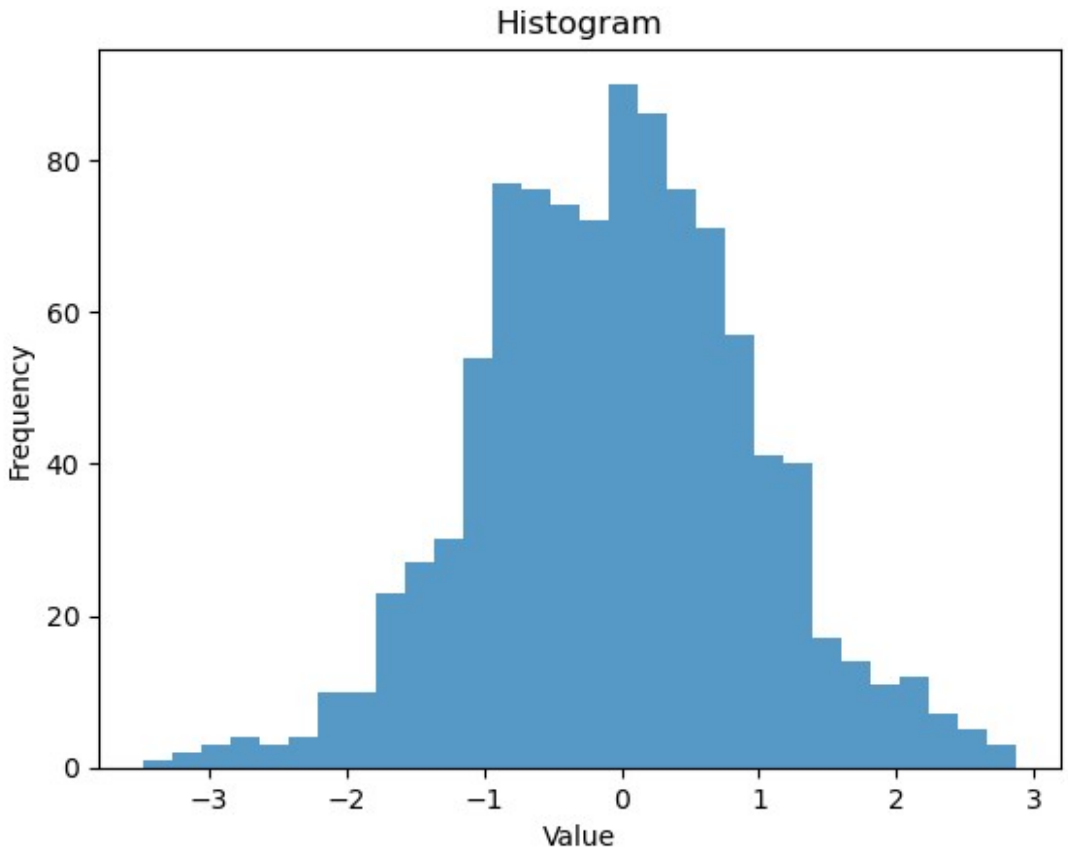
```
x = np.linspace(0, 10, 100)

# Create a figure and a set of subplots
fig, ax = plt.subplots(nrows=2, ncols=1, figsize=(7, 6))

# First subplot
ax[0].plot(x, np.sin(x), label="sin(x)")
ax[0].set_title("sin(x)")
ax[0].legend()

# Second subplot
ax[1].plot(x, np.cos(x), label="cos(x)", color="red")
ax[1].set_title("cos(x)")
ax[1].legend()

# Adjust layout
plt.tight_layout()
plt.show()
```




# Advanced Plotting

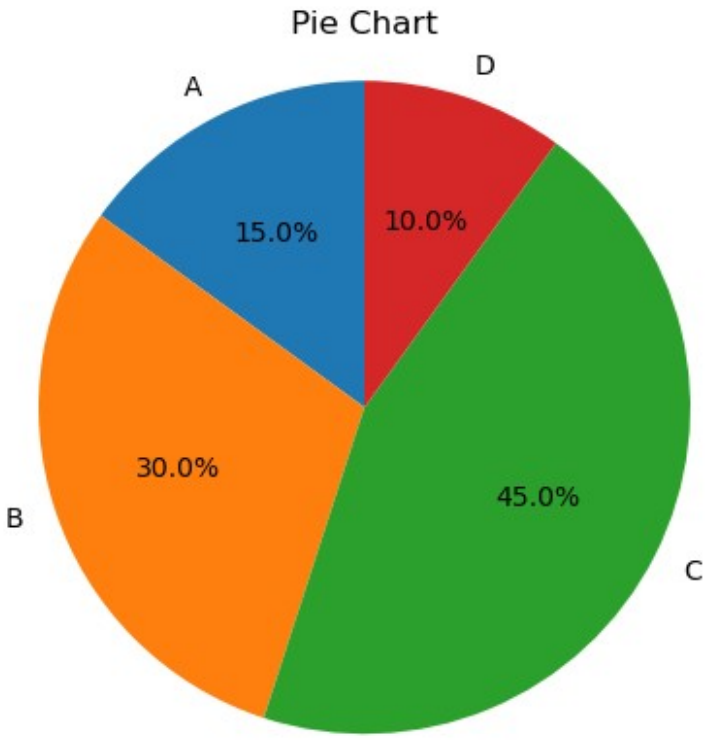
## Histograms

```
# Generate random data
data = np.random.randn(1000)

plt.hist(data, bins=30, alpha=0.75)
plt.title("Histogram")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.show()
```



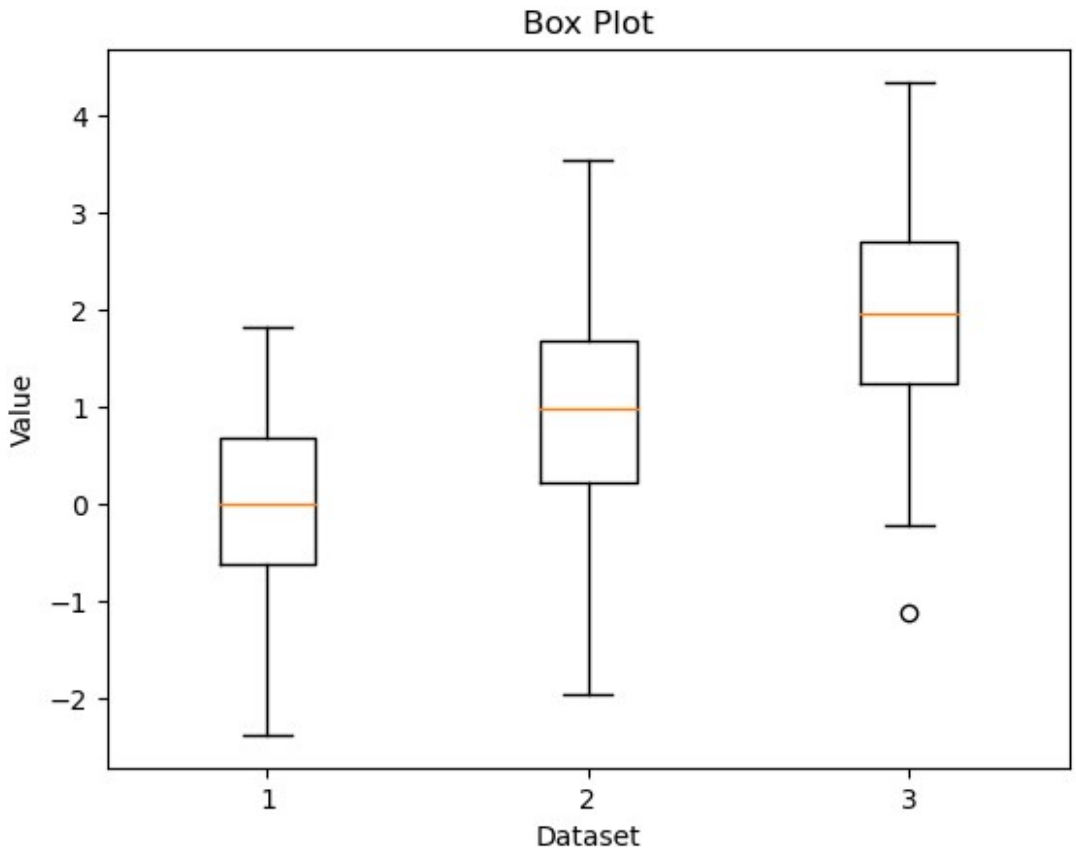

## Pie Charts

```
labels = ["A", "B", "C", "D"]
sizes = [15, 30, 45, 10]

plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
plt.title("Pie Chart")
plt.axis("equal")  # Equal aspect ratio ensures that pie is drawn as a
circle.
plt.show()
```

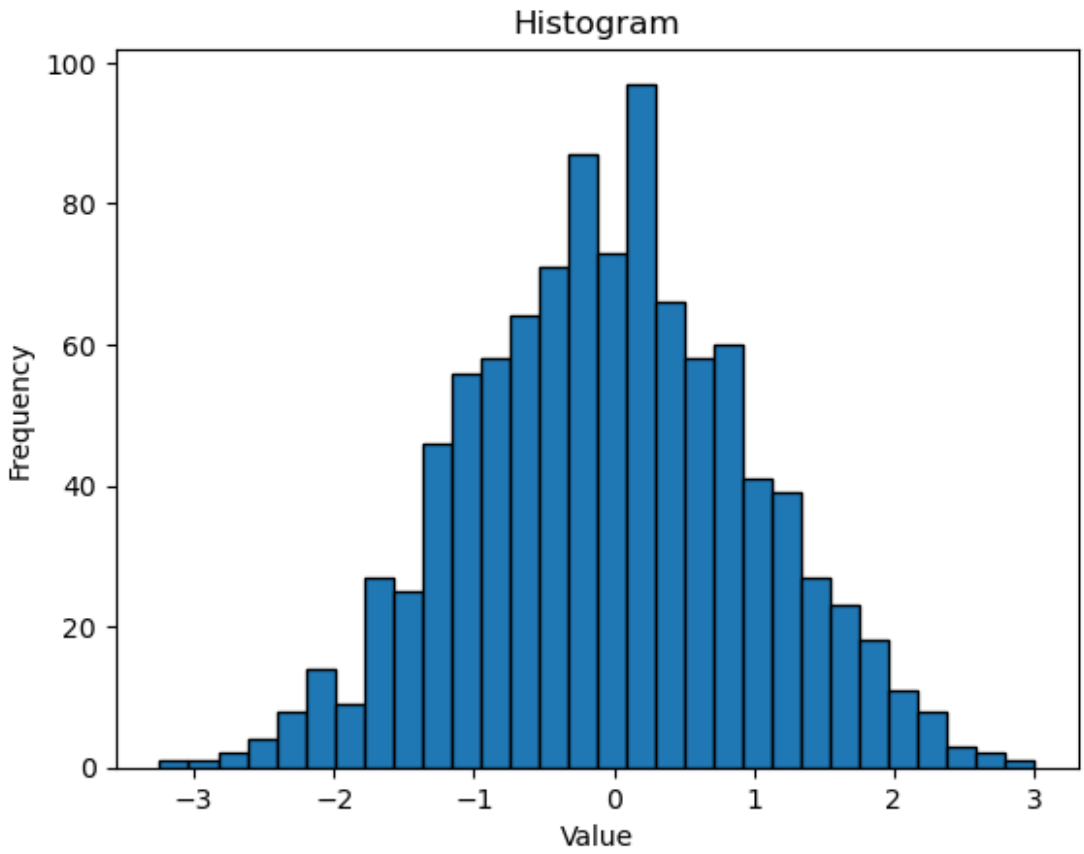



## Box Plots

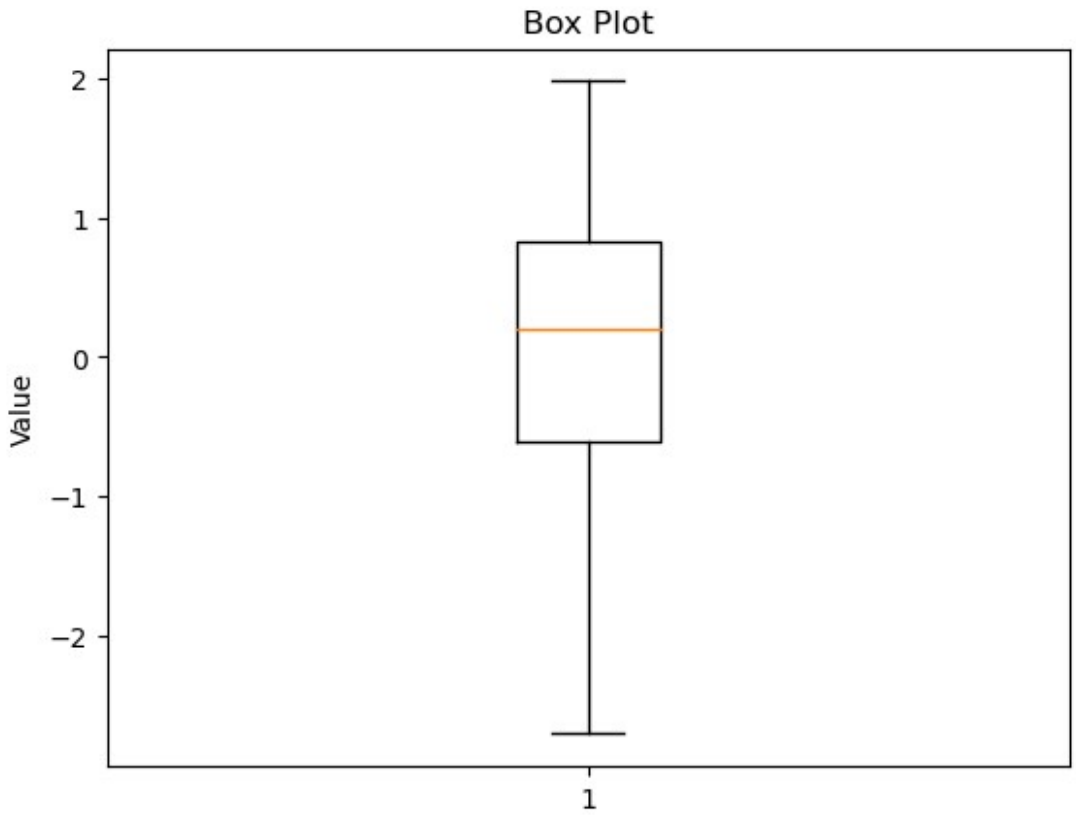
```
# Generate three sets of random data
data = [np.random.randn(100) + i for i in range(3)]

plt.boxplot(data)
plt.title("Box Plot")
plt.xlabel("Dataset")
plt.ylabel("Value")
plt.show()
```



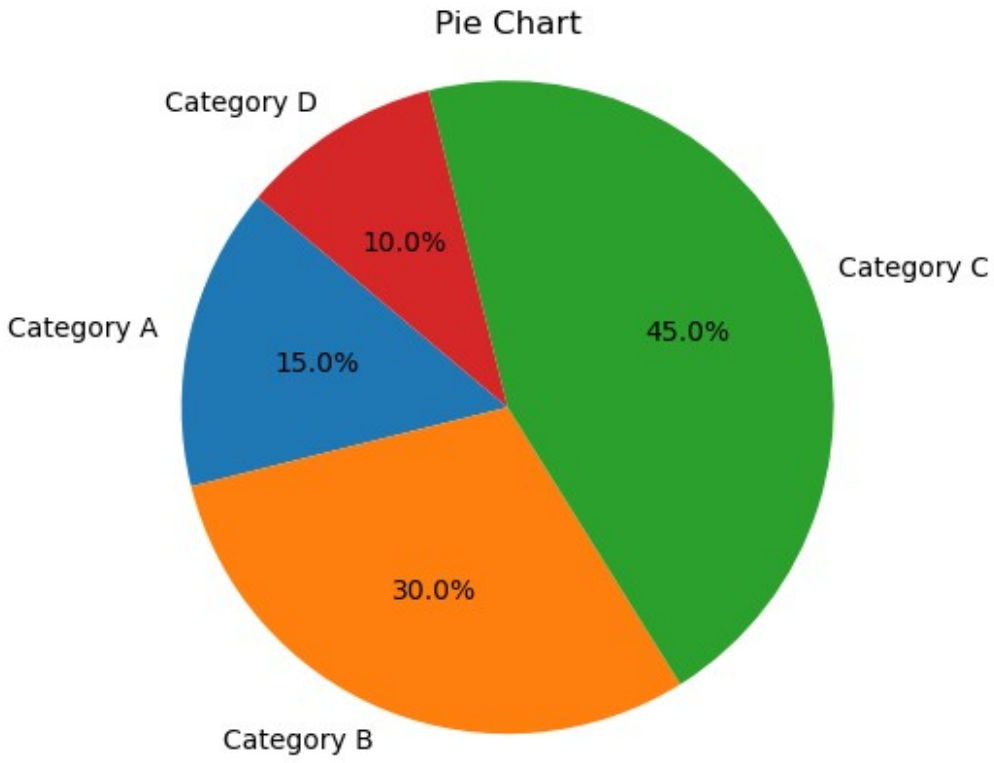

```
# Generate random data for histogram
data = np.random.randn(1000)

# Create a histogram
plt.hist(data, bins=30, edgecolor='black')
plt.title("Histogram")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.show()
```



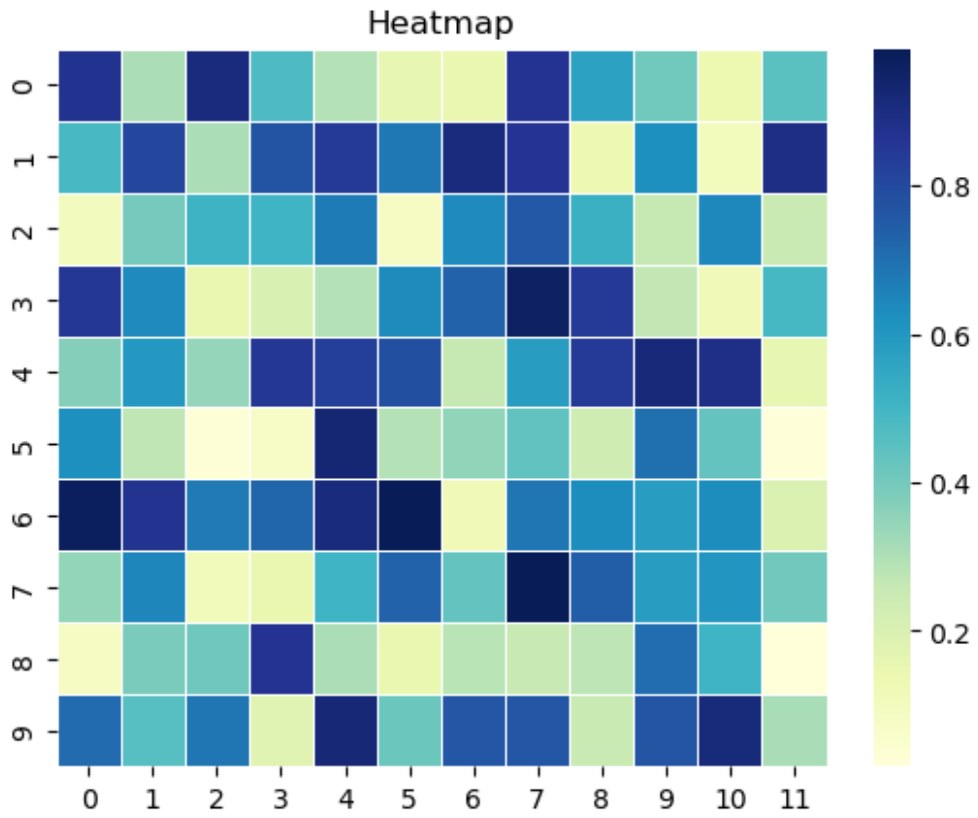

```
# Generate random data for box plot
data = np.random.randn(100)

# Create a box plot
plt.boxplot(data)
plt.title("Box Plot")
plt.ylabel("Value")
plt.show()
```



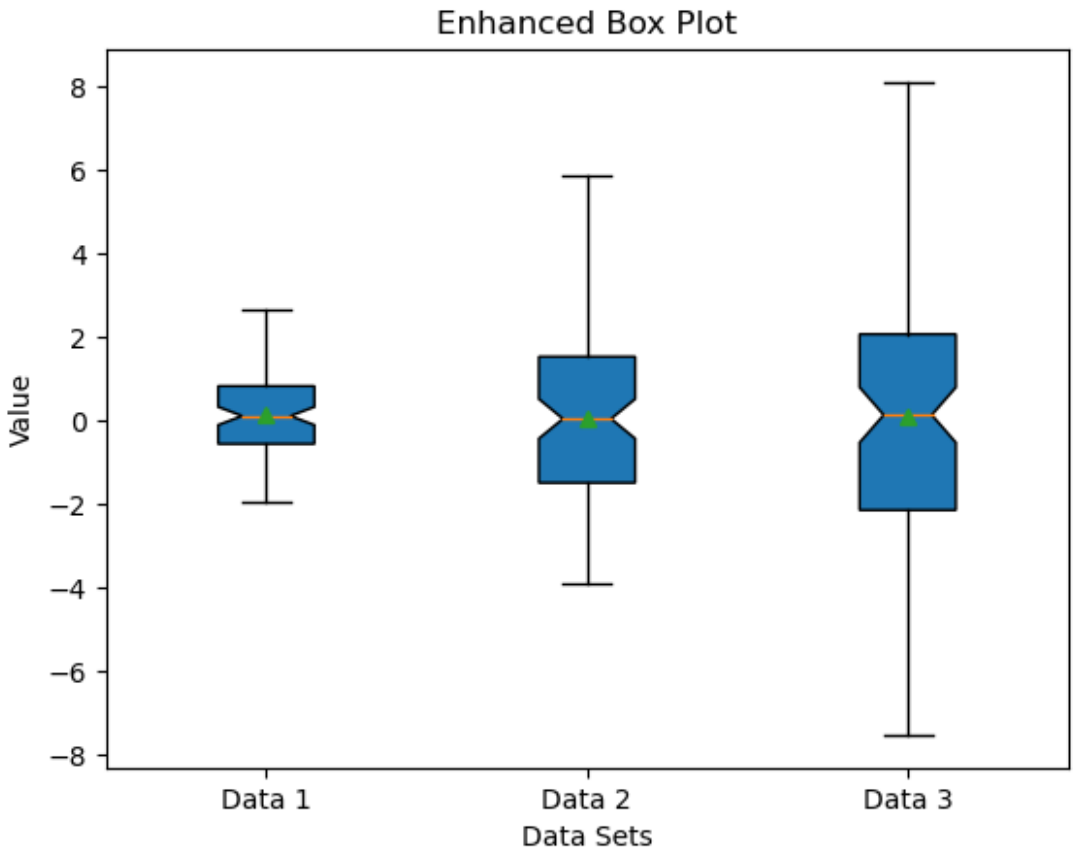

```
# Data for pie chart
labels = ['Category A', 'Category B', 'Category C', 'Category D']
sizes = [15, 30, 45, 10]

# Create a pie chart
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)
plt.title("Pie Chart")
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a
circle.
plt.show()
```




```
import seaborn as sns  # Importing seaborn for better visual aesthetics

# Generate random data for heatmap
data = np.random.rand(10, 12)

# Create a heatmap
sns.heatmap(data, cmap='YlGnBu', linewidths=.5)
plt.title("Heatmap")
plt.show()
```




```
# Generate multiple random data sets for box plot
data1 = np.random.randn(100)
data2 = np.random.randn(100) * 2
data3 = np.random.randn(100) * 3

data = [data1, data2, data3]

# Create a box plot
plt.boxplot(data, patch_artist=True, notch=True, showmeans=True)
plt.title("Enhanced Box Plot")
plt.xlabel("Data Sets")
plt.ylabel("Value")
plt.xticks([1, 2, 3], ["Data 1", "Data 2", "Data 3"])
plt.show()
```


Enhanced Box Plot
Value
8
6
4
2
0
−2
−4
−6
−8
Data 1
Data 2
Data 3
Data Sets