Plotting

In this tutorial, we will learn how to plot using the `matplotlib` library.

Installing matplotlib

To install matplotlib, run the following command in your terminal:

```
pip install matplotlib
```

Once you have installed `matplotlib`, you can import it into your Python code:

```
import matplotlib.pyplot as plt
```

Line Plot

Let's start by plotting a simple line graph. We will use the plot method provided by Matplotlib.

```
import matplotlib.pyplot as plt

# x-axis values
x = [1, 2, 3, 4, 5]
# y-axis values
y = [2, 4, 6, 8, 10]

# plotting the line graph
plt.plot(x, y)

# adding labels to the axes
plt.xlabel("x-axis")
plt.ylabel("Y-axis")

# adding a title to the graph
plt.title("Line Graph")

# displaying the graph
plt.show()
```

The `plot` method takes two arrays, one for the x-axis values and one for the y-axis values. The `xlabel` and `ylabel` methods are used to add labels to the x and y axes, respectively. The

`title` method is used to add a title to the graph. Finally, the `show` method is used to display the graph.

Changing Line Style and Color

To make the plot more informative, you can add labels to the axes and a title to the plot:

```
# Create a dashed line with blue color and circle markers
plt.plot(x, y, linestyle="--", color="blue", marker="o")
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.title("A Customized Line Plot")

plt.show()
```

Scatter Plot

Next, let's plot a scatter plot. We will use the `scatter` method provided by Matplotlib.

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

# x-axis values
x = np.random.rand(50)
# y-axis values
y = np.random.rand(50)

# plotting the scatter plot
plt.scatter(x, y)

# adding labels to the axes
plt.xlabel("x-axis")
plt.ylabel("Y-axis")

# adding a title to the graph
plt.title("Scatter Plot")

# displaying the graph
plt.show()
```

The `scatter` method takes two arrays, one for the x-axis values and one for the y-axis values. The `xlabel` and `ylabel` methods are used to add labels to the x and y axes, respectively. The title method is used to add a title to the graph. Finally, the `show` method is used to display the graph.

Bar Chart

Next, let's plot a bar chart. We will use the `bar` method provided by Matplotlib.

```
import matplotlib.pyplot as plt

# x-axis values
x = ["A", "B", "C", "D", "E"]
# y-axis values
y = [10, 24, 36, 40, 15]

# plotting the bar chart
plt.bar(x, y)

# adding labels to the axes
plt.xlabel("x-axis")
plt.ylabel("Y-axis")

# adding a title to the graph
plt.title("Bar Chart")

# displaying the graph
plt.show()
```

The `bar` method takes two arrays, one for the x-axis values and one for the y-axis values. The `xlabel` and `ylabel` methods are used to add labels to the x and y axes, respectively. The `title` method is used to add a title to the graph. Finally, the `show` method is used to display the graph.

Pie Charts

Finally, let's plot a pie chart. We will use the `pie` method provided by Matplotlib.

```
import matplotlib.pyplot as plt

# labels for the sections of the pie chart
```

```
labels = ["A", "B", "C", "D", "E"]
# sizes of each section
sizes = [10, 24, 36, 40, 15]

# plotting the pie chart
plt.pie(sizes, labels=labels)

# adding a title to the graph
plt.title("Pie Chart")

# displaying the graph
plt.show()
```

The `pie` method takes an array for the sizes of each section of the pie chart and a list of labels for each section. The `title` method is used to add a title to the graph. Finally, the `show` method is used to display the graph.

Saving a Plot

To save a plot as an image file, you can use the `savefig` method:

```
plt.plot(x, y)
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.title("A Line Plot to be Saved")

# Save the plot as a PNG file
plt.savefig("line_plot.png")

# Close the plot
plt.close()
```

Exercises

1. Plot a line graph showing the sales trend of a company over the past year. Label the x-axis as "Month" and the y-axis as "Sales" with a title "Sales Trend".
2. Create a scatter plot showing the relationship between the weight and height of a group of individuals. Label the x-axis as "Weight" and the y-axis as "Height" with a title "Weight vs. Height".

3. Plot a bar chart showing the number of medals won by a group of countries in the 2020 Olympics. Label the x-axis as "Country" and the y-axis as "Medals" with a title "2020 Olympics Medal Count".
4. Create a pie chart showing the percentage of different types of fruits in a fruit basket. Label each section with the name of the fruit and add a title "Fruit Basket".
5. Plot a histogram showing the distribution of grades in a class. Label the x-axis as "Grade" and the y-axis as "Frequency" with a title "Grade Distribution".
6. Create a boxplot showing the distribution of the heights of basketball players in a team. Label the y-axis as "Height" with a title "Basketball Player Height Distribution".
7. Plot a line graph showing the temperature trend of a city over the past week. Label the x-axis as "Day" and the y-axis as "Temperature" with a title "Temperature Trend".
8. Create a scatter plot showing the relationship between the price and quality of a group of products. Label the x-axis as "Price" and the y-axis as "Quality" with a title "Price vs. Quality".
9. Plot a stacked bar chart showing the percentage of different types of expenses in a budget. Label the x-axis as "Category" and the y-axis as "Percentage" with a title "Budget Expenses".
10. Create a radar chart showing the performance of a team in different areas such as communication, teamwork, and leadership. Label each axis with the name of the area and add a title "Team Performance".

Project: Visualizing COVID-19 Data

Project Description

The objective of this project is to visualize the COVID-19 data using the Matplotlib library. The data will be retrieved from an API and will be displayed in various types of graphs such as line charts, bar charts, and pie charts.

Project Requirements

The objective of this project is to visualize the COVID-19 data using the Matplotlib library. The data will be retrieved from an API and will be displayed in various types of graphs such as line charts, bar charts, and pie charts.

To complete this project, you'll need the following:

- Python 3.x

- Matplotlib library
- Requests library

Project Steps

1. Installed the required libraries.
2. Retrieve the COVID-19 data from an API using the following code:

```
import requests

url = 'https://api.covid19api.com/dayone/country/canada'
response = requests.get(url)
data = response.json()
```

3. Create a line chart showing the total number of confirmed cases in Canada over time.
4. Create a bar chart showing the total number of confirmed cases and deaths in Canada.
5. Create a pie chart showing the percentage of confirmed cases and deaths in Canada.
6. Explore the data and see what other plots you can come up with!
7. Save the graphs as images.