Task 1 (Easy)

**Visualization Libraries Documentation**

### **Matplotlib:**

## **Library Overview:** Matplotlib is a widely used Python library for creating static, interactive, and animated visualizations. It provides a MATLAB-like interface and is highly customizable, making it suitable for various use cases from simple plotting to complex visualizations. Matplotlib is well-suited for generating basic plots quickly and efficiently.

## **Graph Type:**

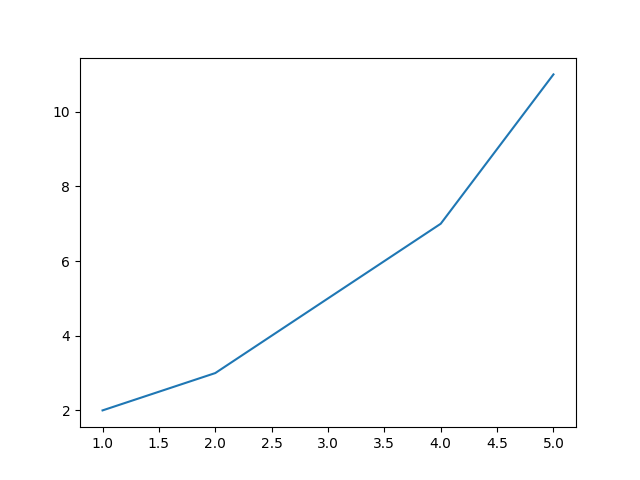
import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]

y = [2, 3, 5, 7, 11]

plt.plot(x, y)

plt.show()

****

1. **Plotly:**

## **Library Overview:** Plotly is a powerful Python library for creating interactive visualizations. It supports a wide range of chart types and allows for easy creation of interactive web-based plots. Plotly is commonly used for building dashboards, presenting data online, and creating visually appealing plots with hover tooltips and zoom functionalities.

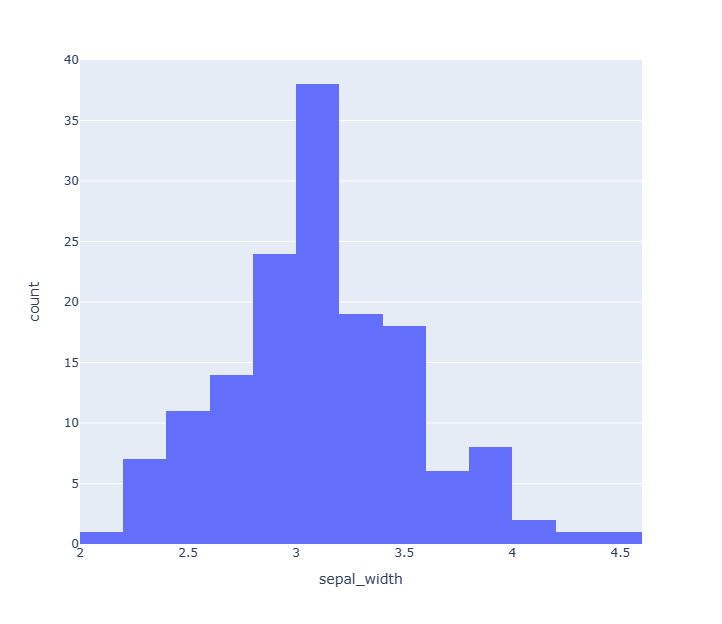
## **Graph Type:**

import plotly.express as px

data = px.data.iris()

fig = px.histogram(data, x="sepal\_width")

fig.show()



**Comparison:**

|  |  |  |
| --- | --- | --- |
| **Library** | **Strengths** | **Weaknesses** |
| Matplotlib | Extensive customization options, mature and stable. | Steeper learning curve, less intuitive syntax. |
| Seaborn | Simplified syntax, attractive default styles. | Limited customization compared to Matplotlib. |
| Plotly | Highly interactive, suitable for web-based deployment. | Requires internet connection for full functionality. |
| Bokeh | Powerful interactivity, easy integration with web apps. | Steeper learning curve compared to Matplotlib. |
| Pandas | Seamless integration with DataFrame objects. | Limited customization compared to standalone plots. |

**Resources:**

* Matplotlib:https://matplotlib.org/stable/users/explain/quick\_start.html#quick-start
* Plotly:https://plotly.com/python/distplot/

Task 2 (Intermediate)

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load AQI data into a DataFrame

aqi\_data = pd.read\_csv('All Task 1,23\Task 2\delhiaqi.csv', parse\_dates=['date'])

# Display summary statistics and first few rows of the dataset

print(aqi\_data.describe())

print(aqi\_data.head())

# Exploratory Data Analysis (EDA)

# Distribution plot of AQI

plt.figure(figsize=(10, 6))

sns.histplot(aqi\_data['pm2\_5'], bins=30, kde=True)  # Assuming pm2\_5 is a measure of air quality

plt.title('Distribution of PM2.5 Levels in Delhi')

plt.xlabel('PM2.5')

plt.ylabel('Frequency')

plt.show()

# Time series analysis of AQI

plt.figure(figsize=(12, 6))

sns.lineplot(x='date', y='pm2\_5', data=aqi\_data)  # Assuming pm2\_5 is a measure of air quality

plt.title('PM2.5 Trend Over Time')

plt.xlabel('Date')

plt.ylabel('PM2.5')

plt.show()

# Seasonal Variation Analysis

aqi\_data['month'] = aqi\_data['date'].dt.month

plt.figure(figsize=(10, 6))

sns.boxplot(x='month', y='pm2\_5', data=aqi\_data)  # Assuming pm2\_5 is a measure of air quality

plt.title('Seasonal Variation of PM2.5 Levels in Delhi')

plt.xlabel('Month')

plt.ylabel('PM2.5')

plt.show()

# Correlation Analysis

corr\_matrix = aqi\_data.corr()

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

sns.heatmap(corr\_matrix, annot=True, cmap='coolwarm', fmt=".2f")

plt.title('Correlation Matrix of AQI Data')

plt.show()

# Long-Term Trend Analysis

plt.figure(figsize=(12, 6))

sns.lineplot(x='date', y='pm2\_5', data=aqi\_data, estimator='mean', ci=None)  # Assuming pm2\_5 is a measure of air quality

plt.title('Long-Term Trend of PM2.5 Levels in Delhi')

plt.xlabel('Year')

plt.ylabel('Mean PM2.5')

plt.show()

