**Experiment 1: Data Loading, Cleaning, and Preprocessing**

**Aim:**

The goal of this experiment is to understand how to load a dataset, clean it, and perform basic preprocessing steps to prepare it for analysis.

**1. Importing Required Libraries**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

**Explanation:**

* **pandas**: For handling data in tabular format.
* **numpy**: For numerical computations.

**matplotlib.pyplot**: For data visualization.

* **MinMaxScaler**: For normalizing data.

**2. Loading the Dataset**

data = pd.read\_csv('covid19\_data.csv')

**Explanation:** We use pd.read\_csv() to load the COVID-19 dataset into a DataFrame.

**3. Understanding the Data**

print(data.describe())

print(data.info())

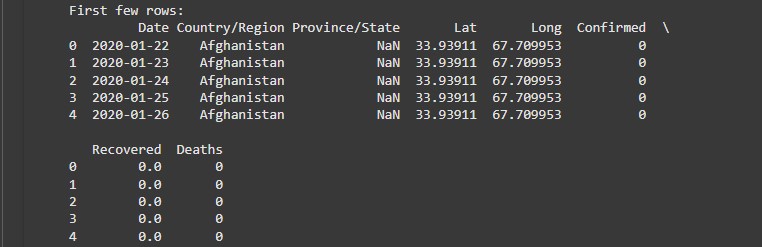
**Explanation:**

* describe(): Provides a statistical summary of the dataset, including mean, median, min, and max values.
* info(): Displays metadata such as column names, data types, and non-null counts.

**4. Displaying the First Few Rows**

print(data.head())

**Explanation:** head(): Displays the first five rows of the dataset to provide an overview of its structure and content.



**5. Visualizing Outliers with Box Plot**

plt.figure(figsize=(10, 8))

plt.boxplot(data['Confirmed'], vert=False)

plt.title('Box Plot of Confirmed COVID-19 Cases')

plt.xlabel('Confirmed Cases')

plt.show()

**Explanation:** A box plot helps visualize the distribution of confirmed cases, detecting outliers and understanding the spread of the data.

**6. Checking for Missing Values**

print(data.isnull().sum())

data.fillna(method='ffill', inplace=True)

**Explanation:**

* isnull().sum(): Displays the number of missing values in each column.
* fillna(method='ffill'): Fills missing values using forward fill to maintain time series consistency.

**7. Visualizing the Distribution of Confirmed Cases with Histogram**

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

plt.hist(data['Confirmed'], bins=20, color='blue', edgecolor='black')

plt.title('Distribution of Confirmed COVID-19 Cases')

plt.xlabel('Confirmed Cases')

plt.ylabel('Frequency')

plt.grid(True)

plt.show()

**Explanation:** A histogram shows the frequency distribution of confirmed cases, providing insight into common ranges of case counts.

**8. Handling Outliers Using the IQR Method**

def handle\_outliers(df, column):

Q1 = df[column].quantile(0.25)

Q3 = df[column].quantile(0.75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

return np.clip(df[column], lower\_bound, upper\_bound)

for col in ['Confirmed', 'Deaths', 'Recovered']:

data[col] = handle\_outliers(data, col)

**Explanation:** The interquartile range (IQR) method removes extreme outliers to improve data quality.

**9. Normalizing Data**

scaler = MinMaxScaler()

data[['Confirmed', 'Deaths', 'Recovered']] = scaler.fit\_transform(data[['Confirmed', 'Deaths', 'Recovered']])

**Explanation:** Normalization scales numerical data to a range between 0 and 1, improving consistency for analysis.

**10. Visualizing Trends Over Time**

data['Date'] = pd.to\_datetime(data['Date'])

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

plt.plot(data['Date'], data['Confirmed'], label='Confirmed Cases', color='blue')

plt.plot(data['Date'], data['Deaths'], label='Deaths', color='red')

plt.title('COVID-19 Trends Over Time')

plt.xlabel('Date')

plt.ylabel('Normalized Values')

plt.legend()

plt.show()

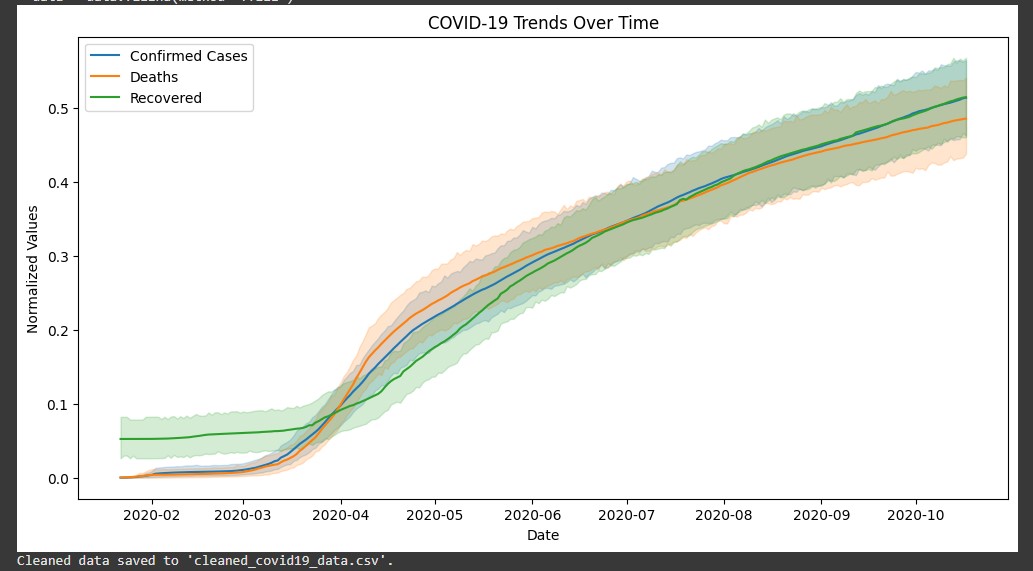
**Explanation:** A line plot visualizes normalized trends of confirmed cases and deaths over time.

**11. Saving the Cleaned Data**

data.to\_csv('cleaned\_covid19\_data.csv', index=False)

print("Cleaned data saved to 'cleaned\_covid19\_data.csv'.")

**Explanation:** The cleaned and preprocessed data is saved as a CSV file for further use.



**Result:**

Thus, the COVID-19 dataset has been loading , cleaning it, and perform basic preprocessing has been completed successfully.