Lab 2
Basics of Data Handling & Plotting

Objectives

At the end, of this practical, you should be able to

- Familiarise with Python modules such as Pandas, Numpy, and Matplotlib.
- Generate basic plots (bar charts, histograms, line plots etc.)
- Fit a model to data.

Dataset

Consider the Corona virus dataset that contains information about number of confirmed cases, number of deaths, and number of recovered patients for various countries of the world. The dataset, maintained by Johns Hopkins University, can be found through this Github link (https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series).
Using the given dataset, carry out the following tasks:

Preparing and describing the data

1. Read the data in the files `time_series_covid19_confirmed_global.csv`, `time_series_covid19_deaths_global.csv`, and `time_series_covid19_recovered_global.csv`. You should read the files using their URLs so as to take account of the daily updates of the data. Consider the dataframe storing the number of deaths for the various countries.
    a. Show the first 10 rows of this dataframe
    b. check for missing values for this dataframe
    c. show the summary statistics for it.

Basic data handling and visualisation

2. Use the pandas function `groupby` to group each of the three obtained dataframes by 'country/Region' and then use `sort_values` to display the 10 countries with the highest total numbers of deaths, confirmed, and recovered cases.
    a. Then use `plt.barh` to plot the bar chart representing the top 10 countries with the highest numbers of confirmed cases.
    b. Complete this question by doing the same thing for the dataframes containing the numbers of recovery and deaths.

3. Consider the following function `get_total_confirmed_world()`, which returns a dataframe containing the total (cumulative) number of confirmed cases over time in the world.

```
def get_total_confirmed_world():
    total = confi.iloc[:, 4: ].apply(sum, axis=0)
    total.index = pd.to_datetime(total.index)
    return total
```

   a. Similarly, write a function get_total_confirmed_ofcountry(country), which returns the total (cumulative) number of confirmed cases over time for the given country 'country'.
   b. Consider the following function line_plot_of_confirmed_country(country, col):

```
def line_plot_ofcountry(name, col):
    data = get_total_confirmed_ofcountry(name)
    plt.figure(figsize=(12, 8))
    plt.title(name.upper()+': Total cases reported', fontsize=14)
    plt.plot(data.index, data, color=col, lw=5)
    plt.ylabel('Total cases')
    plt.grid()
    plt.show()
```

      This plots a curve of the cumulative total number of confirmed cases for the country 'country' in the colour 'col'.
      Similarly, write a function hist_total_confirmed_ofcountry(country), which plots the histogram of the cumulative total number of confirmed cases for the country 'country'. Test both functions with the input 'US' and observe that the histogram does not reveal too much information in this case. Produce a bar plot this time and see if this kind of plot is more appropriate.

   c. Plot the cumulative total number of confirmed cases for the UK, US, and India in the same graph as a line graph. Your graph should use a different colour for each plot.

4. Consider the following function get_daily_confirmed_country(country), which takes as input 'country' and returns a dataframe that contains the daily number of cases for that country.

```
def get_daily_confirmed_country(name):
    df_country = confi['Country/Region']==name
    cases = confi[df_country].iloc[:,4: ].apply(lambda x: x.sum())
    dates = pd.to_datetime(cases.index)
    frame = {'Dates':dates, 'Cases':cases}
    df = pd.DataFrame(frame)
    df['Lag'] = df.Cases.shift(1).fillna(0)
    df['Daily Cases'] = df.Cases - df.Lag
    return df[['Dates', 'Daily Cases']]
```

   a. Write a function moving_averages (country, n=7), which takes an input 'country' and returns a dataframe containing the n days simple moving averages of confirmed cases for that country. If need be, check out Wikipedia for the formal definition of SMA (simple moving average).
   b. Write a function plot_daily_and_avg_country(country), which takes as input 'country' and produces a bar plot of the number of confirmed daily cases along with a line plot of their 7 days averages in the same graph.
   c. Test your functions with the inputs country 'US', 'India' and 'UK'.

Model fitting

5. Consider the total number of confirmed cases for the UK.
   a. Using the Python function '`curve_fit`' from `scipy.optimize`, fit that data with a polynomial model of your choice.
   b. Show the standard deviations of the calculated coefficients of your polynomial.
   c. Plot the predicted and observed numbers of cases in the same graph. Your graph should contain a legend showing the labels for each plot.