ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

# TRỰC QUAN HÓA DỮ LIỆU

LAB 01

## MỐI QUAN HỆ TRONG DỮ LIỆU

**GVHD: Thầy Bùi Tiến Lên**

**Thầy Lê Ngọc Thành**

Mã nhóm : 05

1. Võ Thế Minh – 18120211
2. Lê Đức Thành – 18120238
3. Nguyễn Thị Ngọc Trâm – 18120246
4. Nguyễn Huy Tú – 18120254

Thành phố Hồ Chí Minh, tháng 04/2021

## MỤC LỤC

# 1. Danh mục hình

# 2. Danh mục bảng

## 3. Danh sách thành viên

| STT | Thành viên | MSSV |
|---|---|---|
| SV1 | Võ Thế Minh | 18120211 |
| SV2 | Lê Đức Thành | 18120238 |
| SV3 | Nguyễn Thị Ngọc Trâm | 18120246 |
| SV4 | Nguyễn Huy Tú | 18120254 |

*Bảng 1 Danh sách thành viên*

## 4. Đánh giá

| STT | Yêu cầu | Đánh giá (%) |
|---|---|---|
| 1 | Thu thập số liệu thống kê từng ngày từ trang Worldmeter | 100% |
| 2 | Nhận xét code/thuật toán để thể hiện trực quan các mối quan hệ giữa các trường dữ liệu | 90% |
| 3 | NSV giữ lại các dữ liệu để có thể thực hiện tiếp cho các bài sau. | 100% |

*Bảng 2 Danh sách yêu cầu*

| STT | Yêu cầu | Thực hiện | Đánh giá (%) |
|---|---|---|---|
| 1 | Thu thập dữ liệu trên trang Wordometer | SV2 | 100% |
| 2 | Vẽ và phân tích biểu đồ Parento, Area và biểu đồ Lolipop | SV4 | 100% |
| 3 | Vẽ và phân tích Scatter Chart và Line Chart | SV3,SV2 | 100% |
| 3 | Vẽ và phân tích Bar Chart, Scatter chart và pie chart | SV1 | 100% |
| 4 | Tổng hợp và chỉnh sửa báo cáo | Cả nhóm | 100% |

*Bảng 3 Kết quả tự đánh giá*

# 5. Báo cáo

## 3.1. Kỹ thuật thu thập dữ liệu

**Ngôn ngữ sử dụng:** Python

**Các thư viện sử dụng:**

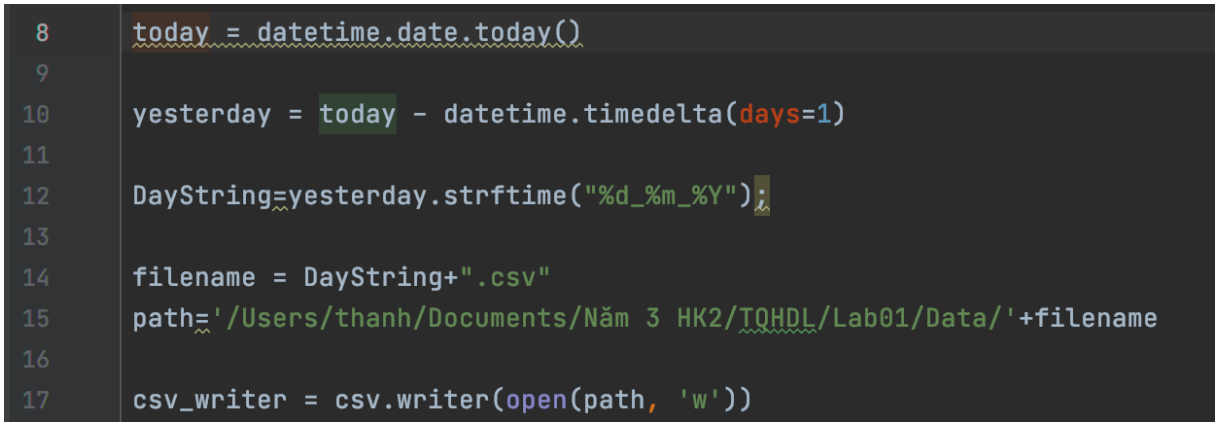
- BeatifulSoup
- Requests
- csv
- Datetime

**Các bước thực hiện và chi tiết:**

1. Tiến hành import các thư viện cần thiết. Đối với thư viên BeatifulSoup cần mở command và thực hiện dòng lệnh pip install beatifulsoup để tải về và sử dụng ( matplotlib.org, không ngày tháng)


*Hình 1 Import thư viện cho thu thập dữ liệu*

2. Cài đặt đoạn code để tạo file csv và đặt tên file theo ngày thu thập dữ liệu




*Hình 2 Đoạn code tạo file csv*

Đoạn code dòng số 08 sử dụng thư viện datetime để lấy ngày hôm nay. Sau đó tiến hành để lấy ngày hôm qua thông qua phép trừ ngày. Sau đó đoạn code số 12 nhằm đưa dạng file về với dưới dạng string theo format là dd_mm_yyyy. Sau đó dòng số 14 nhằm tạo file name từ chuỗi ngày vừa tạo + với đuôi file là csv thành tên file hoàn chỉnh. Để tiện lưu trữ, dòng thứ 15 nhằm xác định địa chỉ đường dẫn lưu file csv vừa tạo. Dòng 17 để tiến hành tạo file và mở file lên ở chế độ ghi ( nếu file chưa tồn tại thì sẽ tự động tạo file) thông qua sử dụng thư viện csv

3. Tiến hành get trang wordomter thông qua thư viện requests

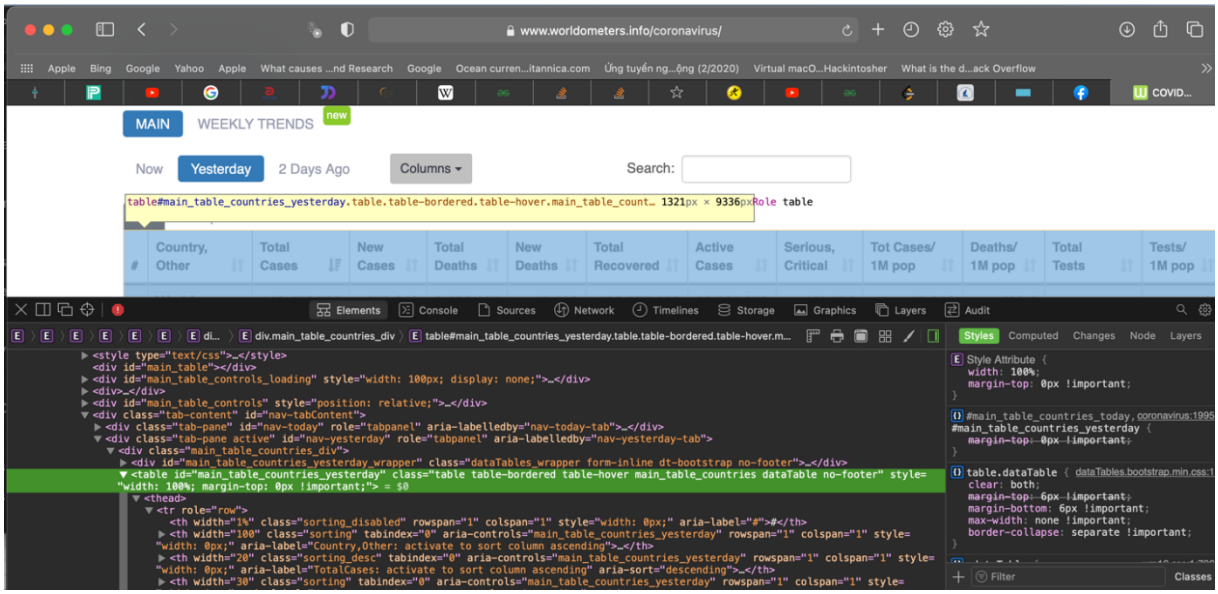
```
page=requests.get('https://www.worldometers.info/coronavirus/#countries')
```

4. Chuẩn bị cho bước thu thập dữ liệu, tiến hành sử dụng thư viện BeatifulSoup để lấy dữ liệu thô html ra

```
soup=BeautifulSoup(page.content,"html.parser")
```

5. Bắt đầu quá trình thu thập dữ liệu:

Đầu tiên, vào trang web Wordometer và chọn bảng dữ liệu về yesterday sau đó chọn inspect element để xem đoạn code html. Chú ý vào id của tag table




*Hình 3 Tìm kiếm id của tag table*

Ở đây id của table dữ liệu yesterday là main_table_countries_yesterday. Sau khi xác định được id của table dữ liệu tiến hành các bước sau:

```
30  data = []
31  table = soup.find('table', attrs={'id':'main_table_countries_yesterday
32  table_body = table.find('tbody')
33  talbe_header=table.find('thead')
34
35  rows = talbe_header.find_all('tr')
36  for row in rows:
37      cols = row.find_all('th')
38      cols = [ele.text.strip() for ele in cols]
39      data.append([ele for ele in cols if ele])
```

*Hình 4 Tiến hành thu thập*

Dòng code 30: khởi tạo mảng data để chứa dữ liệu các quốc gia
Dòng code 31: gán nguyên khối code html của tag table có id là main_table_countries_yesterday vào biến table
Dòng code 32: dùng để tìm khối html con chứa dữ liệu của bảng, gán vào biến table_body
Dòng code 33: dùng để tìm khối html con chứa header ( tên cột) của bảng, gán vào biến table_header

Dòng code 35: tìm kiếm tất cả các dòng là tr gán vào biến rows
Dòng code 36->39: xử lý tìm kiếm lọc loại bỏ các dòng html không cần thiết thông qua phương thức strip. Dữ liệu tên cột sau khi được làm sạch sẽ được append vào mảng data. Data sẽ được là 1 mảng có phần tử là 1 mảng

6. Thực hiện tương tự với phần còn lại của data của bảng html sử dụng vòng for và phương thức strip để lấy được dữ liệu sạch

```
41      rows = table_body.find_all('tr')
42      for row in rows:
43          cols = row.find_all('td')
44          cols = [ele.text.strip() for ele in cols]
45          data.append(cols)
```

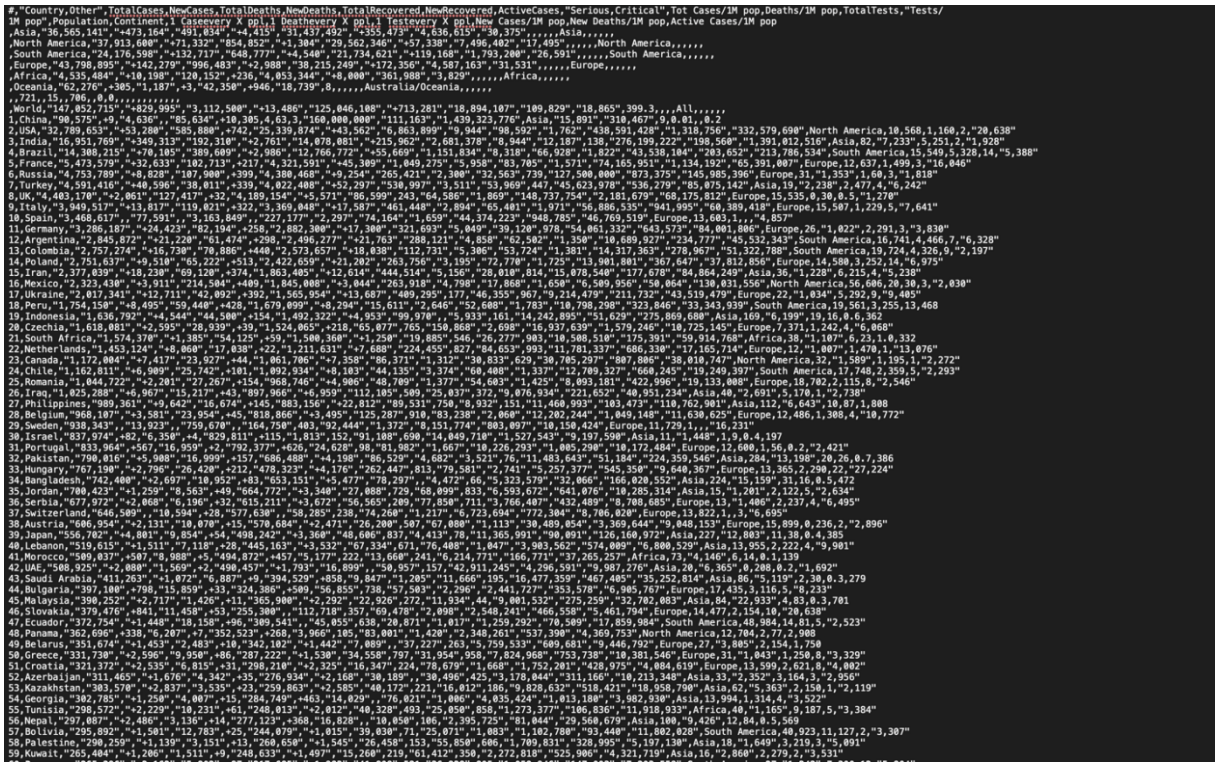
*Hình 5 Tiến hành thu thập (2)*

7. Sau khi đã append toàn bộ dữ liệu từ file html vào mảng data. Tiến hành sử dụng vòng for quét hết mảng data và với từng phần tử gọi phương thức write row vào file csv đã tạo ở bước trên

```
51      for dt in data:
52
53          print(dt)
54          csv_writer.writerow(dt)
```

*Hình 6 Điền vào file csv đã tạo*

8. Kết quả sẽ được ghi vào file csv tương ứng để xử lý như hình dưới đây



*Hình 7 Kết quả sau khi thu thập*

9. Các bước tiếp theo có thể sử dụng công cụ như Excel hoặc Pandas dataframe để xử lý dữ liệu

*Đối với bộ dữ liệu về Ấn Độ, nhóm tiến hành thu thập bằng tay đặt tên file là india_newcase_newdeath.csv

## 3.2. Các loại biểu đồ sử dụng và nhận xét

### 3.2.1. Pareto Chart

**Ngôn ngữ sử dụng**: Python, Jupyter Notebook

**Lý do sử dụng:** biểu đồ xác định đâu là nơi có nhiều ca mắc mới bộc phát nhất. Mục đích là để tìm ra trong một nhóm nguyên nhân đâu là nguyên nhân quan trọng nhất (lục địa/quốc gia).

**Quan hệ biểu diễn:** số ca mắc COVID-19 mới trong 1 ngày (23/04/2021) theo lục địa & quốc gia

**Thư viện sử dụng**:

- Matplotlib
- Seaborn
- Numpy
- Pandas

**Kỹ thuật vẽ**:

10. Tiến hành import các thư viện cần thiết

```
"import matplotlib.pyplot as plt\n",
"import seaborn as sns\n",
"from matplotlib.ticker import PercentFormatter\n",
"# set seaborn style\n",
"sns.set_theme()\n",
"# make the plot bigger \n",
"plt.rcParams['figure.figsize'] = [12, 8]\n",
"plt.rcParams['figure.dpi'] = 70 \n",
"#pd.set_option('display.max_rows', 250)"
```

*Hình 8 Import thư viện để vẽ Parento Chart*

11. Tiến hành import dữ liệu vào dataframe và tiền xử lý

```
    "raw_data = pd.read_csv('23_04_2021.csv')"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "metadata": {},
   "outputs": [],
   "source": [
    "def dataframeCleaner(df):\n",
    "    for columnname in df: #looping through titles of the table \n",
    "        temp = []     \n",
    "        for column in df[columnname]:   #geting column elements for the each title\n",
    "            column = str(column)\n",
    "            column = column.replace(',','')# Removing unwanted data clutter\n",
    "            column = column.replace('+','')#Removing unwanted '+'sign  \n",
    "            try:   #using try except block to convert datatype string to integer while avoiding error\n",
    "                column = int(column)\n",
    "            except:\n",
    "                pass\n",
    "            \n",
    "            temp.append(column)\n",
    "        df[columnname] = temp\n",
    "        \n",
    "    df.replace('nan', 0, regex=True,inplace=True) # delete N/A\n",
    "    df.replace(['\\n'], '', regex=True, inplace=True) #delete unwanted newline\n",
    "    df.replace([','], '', regex=True, inplace=True) #delete comma\n",
    "    df.replace(r'^\\s*$', 0, regex=True,inplace=True)# converting empty string to 0\n",
    "\n",
    "    return df"
```

*Hình 9 Xử lý dữ liệu*

Dòng 59: tiến hành đọc dữ liệu thô vào dataframe

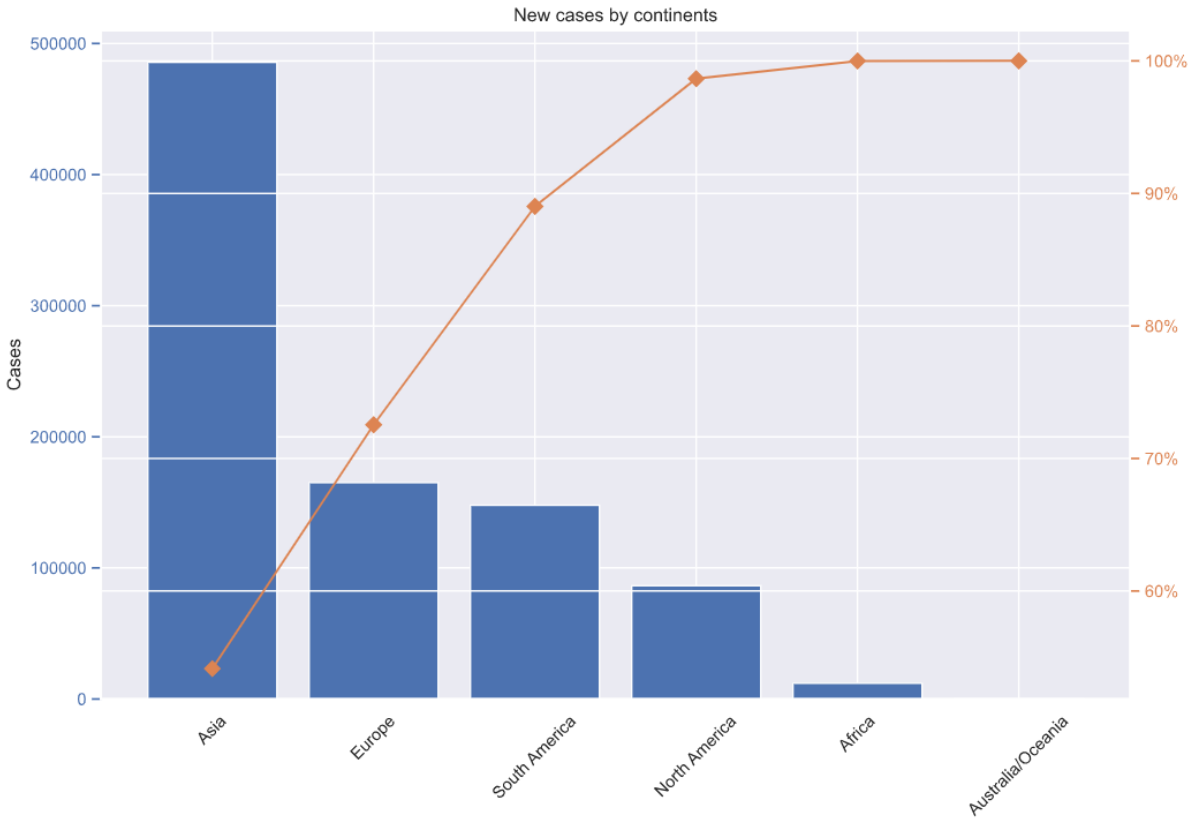
Dòng 69-88: tiến hành xử lý dữ liệu dataframe

12. Tiến hành vẽ

```
    "def paretoChart(df,title):\n",
    "    df = df.sort_values(['NewCases'],ascending=False) \n",
    "    # add cumulative frequency\n",
    "    df[\"cumpercentage\"] = df[\"NewCases\"].cumsum()/df[\"NewCases\"].sum()*100\n",
    "\n",
    "    fig, ax = plt.subplots()\n",
    "    ax.bar(df.index, df[\"NewCases\"], color=\"C0\")\n",
    "    ax.set_ylabel('Cases')\n",
    "    ax.set_title(title)\n",
    "\n",
    "    ax2 = ax.twinx()\n",
    "    ax2.plot(df.index, df[\"cumpercentage\"], color=\"C1\", marker=\"D\", ms=7)\n",
    "    ax2.yaxis.set_major_formatter(PercentFormatter())\n",
    "\n",
    "    ax.tick_params(axis=\"y\", colors=\"C0\")\n",
    "    ax2.tick_params(axis=\"y\", colors=\"C1\")\n",
    "\n",
    "    for tick in ax.get_xticklabels():\n",
    "        tick.set_rotation(45)\n",
    "\n",
    "    plt.show()"
   ]
```

*Hình 10 Xử lý vẽ Parento Chart*

13. Kết quả sau khi chạy:




*Hình 11 Kết quả Parento Chart (1)*

**Nhận xét**

Đây là chart được vẽ dựa trên nguyên tắc Pareto: đại đa số mọi thứ trong cuộc sống không được phân phối đều nhau. *Khoảng 80% kết quả là do 20% nguyên nhân gây ra.* Biểu đồ này bao gồm *bar chart* và *line chart*. Các cột trong chart được sắp xếp theo thứ tự từ cao đến thấp theo tần số, còn các giá trị tần suất tích lũy được biểu diễn bằng đường thẳng.

Quan sát biểu đồ, cột trái có đơn vị là tần số, cột phải có đơn vị là tần số tích lũy, đường màu cam thể hiện giá trị tần số tích lũy

Chiều cao của cột thể hiện *tần số (frequency)* – đó là số lượng ca mắc mới trong mỗi nhóm châu lục. Ví dụ, có tổng cộng gần 200,000 ca mắc mới ở Châu Âu.

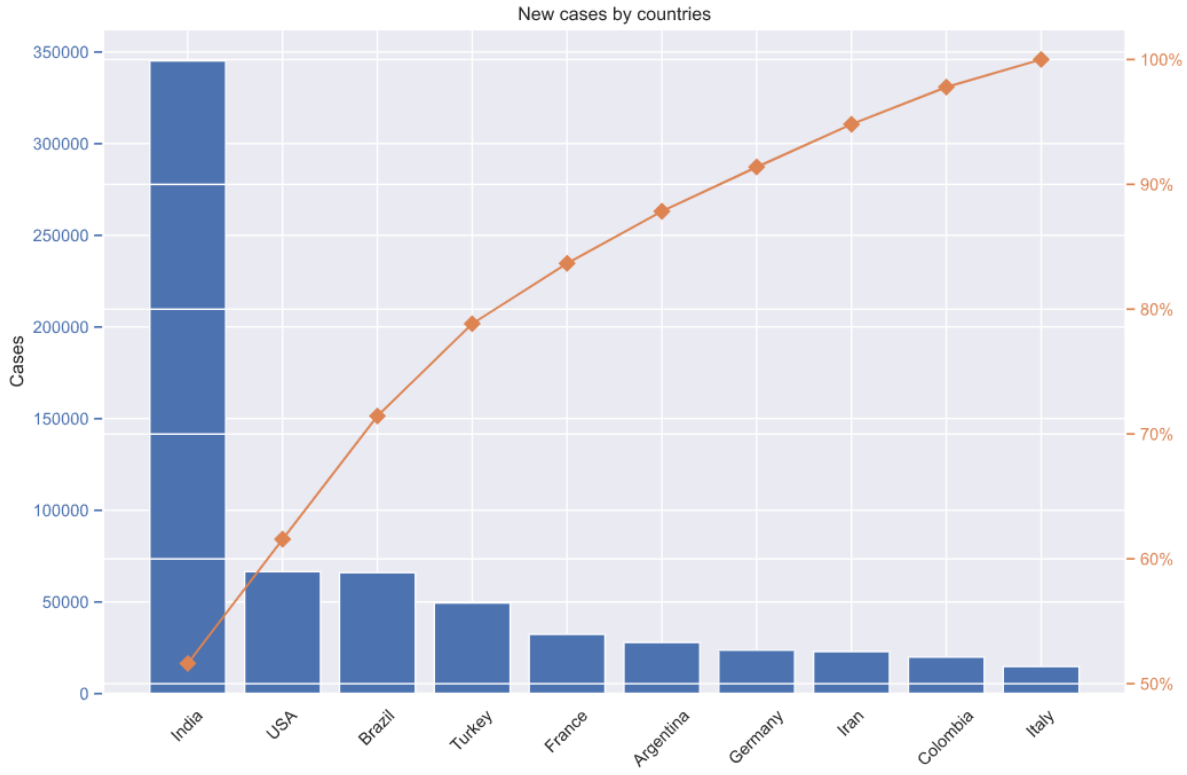
Đường màu cam thể hiện *tần số tích lũy (cumulative frequency)* – số lượng ca mắc ở các quốc gia có số ca mắc nhỏ hơn và bao gồm số mốc đó. Kết thúc ở 100%.

Dựa vào đường tần số tích lũy này, chiếu xuống trục hoành, ta thấy 80% ca mắc mới thuộc về hầu hết ở 3 châu lục chính: Châu Á, Châu Âu và Nam Mỹ. Điều này có thể chứng tỏ rằng dịch bệnh đang lan rộng và phát triển ở các châu lục này.

Đặc biệt là ở Châu Á, khi số ca mắc mới đạt gần 500,000 trong 1 ngày, cao gấp 2,5 lần so với Châu Âu và cách biệt hoàn toàn so với các châu lục khác.

Ngoài ra, lục địa Châu Úc/Châu Đại Dương không ghi nhận ca mắc mới nào. Đây là mặt tích cực của vấn đề.

➔ Ta sẽ xem kĩ vào quan hệ với top 10 nước mắc ca nhiễm mới để biết nguyên nhân gây ra số ca nhiễm vượt trội tại Châu Á.




*Hình 12 Kết quả Parento Chart (2)*

**Nhận xét:** Tập trung vào 80% ca mắc mới thì còn có 4 nước là Ấn Độ, USA, Brazil và Thổ Nhĩ Kỳ.

Đáng chú ý là Ấn Độ, chiếm hơn 50% ca mắc mới trong tổng cộng 10 nước có số ca mắc mới nhiều nhất. Tình hình lây nhiễm tại đây đã bỏ xa cả thế giới. Ngày 23-4 là ngày đầu tiên Ấn Độ ghi nhận số ca nhiễm mới vượt 300.000 ca/ngày, sau chuỗi *8 ngày liên tiếp* nước này có số ca nhiễm mới trên *200.000 ca/ngày*.

Đây là hậu quả của sự lơ là của người dân trong thực hiện các biện pháp phòng dịch sau khi làn sóng dịch thứ nhất lắng xuống. Người dân lại tụ tập đông trong các sự kiện lễ hội tôn giáo, vận động tranh cử.

➔ Để xem dữ liệu dưới nhiều góc nhìn, ta tiếp tục sử dụng ***Area chart*** để xem sự thay đổi của số lượng ca mắc mới của các quốc gia trên.

### 3.2.2. Area Chart

**Ngôn ngữ sử dụng:** Python, Jupyter Notebook

**Lý do sử dụng:** quan sát được các *tổng số (totals)* theo *thời gian* và tiện lợi để so sánh các quốc gia. Dễ để kiểm chứng với thông tin của ***Pareto Chart*** đã vẽ

**Quan hệ biểu diễn:** sự thay đổi số ca mắc COVID-19 mới trong 3 ngày (từ 22/4 đến 22/4) của top 16 quốc gia có số lượng ca mắc mới lớn nhất trong ngày

**Thư viện sử dụng:**

- Seaborn
- Matplotlib
- Numpy
- Pandas

**Kỹ thuật vẽ:**

1. Tiến hành import thư viện (đã import ở chart trước)
2. Đọc dữ liệu từ file csv vào dataframe

```
In [28]: raw_data_18 = pd.read_csv('_18_04_2021.csv')
         raw_data_19 = pd.read_csv('_19_04_2021.csv')
         raw_data_20 = pd.read_csv('_20_04_2021.csv')
         raw_data_21 = pd.read_csv('_21_04_2021.csv')
         raw_data_22 = pd.read_csv('22_04_2021.csv')
         raw_data_23 = pd.read_csv('23_04_2021.csv')
         raw_data_24 = pd.read_csv('24_04_2021.csv')
         country = ["India","USA","Brazil","Turkey","France","Argentina","Germany","Iran","Colombia","Italy","Ukraine","Sp
```

*Hình 13 Xử lý dữ liệu vào dataframe (Area Chart)*

3. Tiến hành tiền xử lý dataframe để cho các bước vẽ kế tiếp

```
In [29]: def createDFArea(dataset,date):
             df = cleanerForNewCases(dataset) #df_22.index = df_22['Country,Other']
             is_country = df['Country,Other'].isin(country)
             df = df[is_country]
             df = df[['Country,Other','NewCases']]
             df = df.sort_values(['NewCases'],ascending=False)
             df['Date'] = date
             return df
```

*Hình 14 Tiền xử lý dữ liệu (Area Chart)*

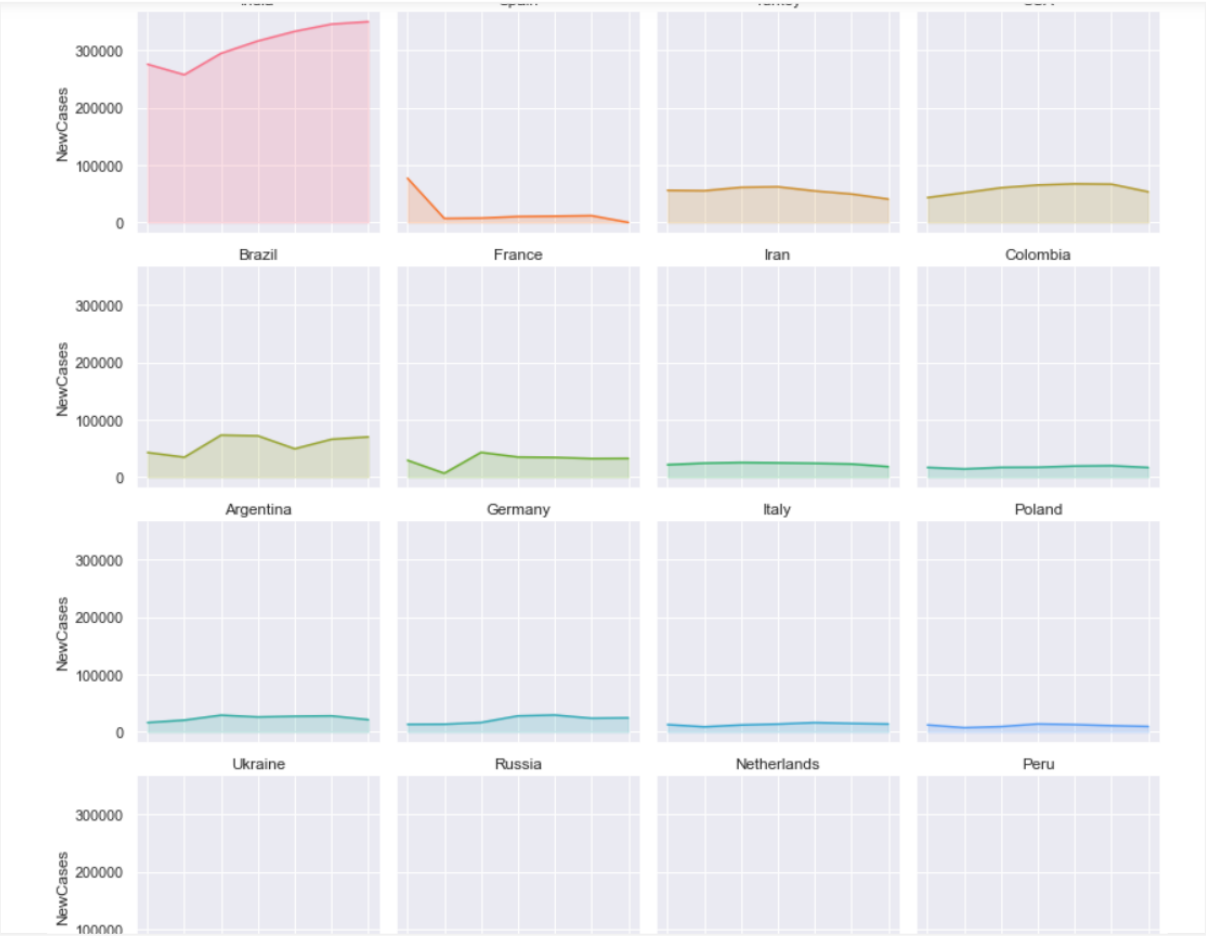
4. Tiến hành lập trình hàm để vẽ Area chart

```
In [30]: def areaChartFace(df,title):
             g = sns.FacetGrid(df, col='Country,Other', hue='Country,Other', col_wrap=4, )  # Create a grid : initialize i
             g = g.map(plt.plot, 'Date', 'NewCases')     # Add the line over the area with the plot function
             g = g.map(plt.fill_between, 'Date', 'NewCases', alpha=0.2).set_titles("{col_name} Country,Other")     # Fill
             g = g.set_titles("{col_name}")     # Control the title of each facet
             plt.subplots_adjust(top=0.92)     # Add a title for the whole plot
             g = g.fig.suptitle(title);
             plt.show()
```

*Hình 15 Tiến hành vẽ Area Chart*

5. Gọi hàm và hiển thị kết quả

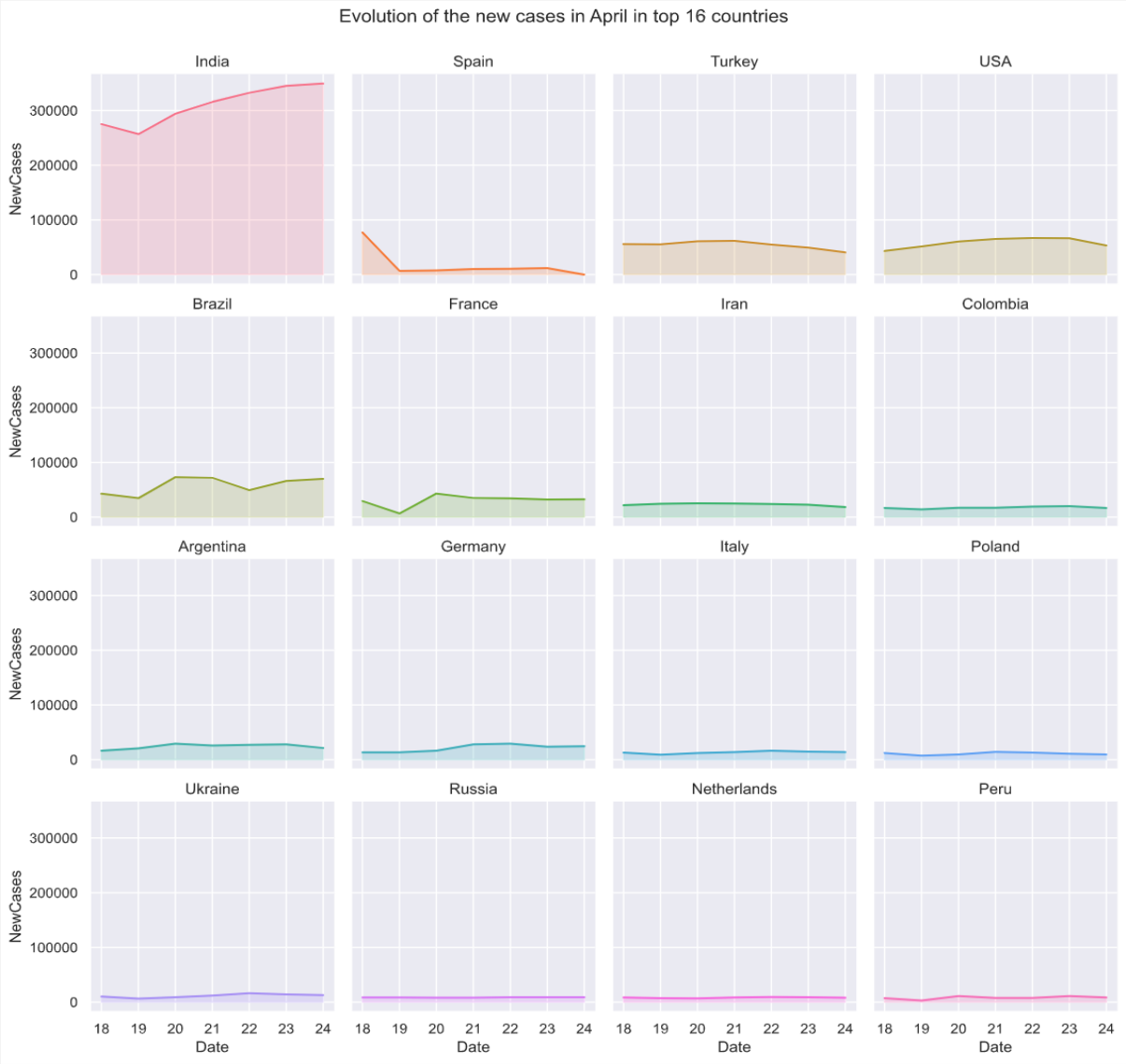
```
In [32]: areaChartFace(df,'Evolution of the new cases in April in top 16 countries');
```




*Hình 16 Kết quả Area Chart*

**Nhận xét:**

Đây là chart bao gồm một line chart và khu vực giữa line và trục được tô màu để đánh dấu nên có tên gọi là area chart. Thông thường, area chart sẽ được vẽ để xếp chồng lên nhau. Tuy nhiên, vì số lượng chart lớn (16 chart) nên sẽ sử dụng kĩ thuật *faceting*: chia nhỏ các biến dữ liệu trên nhiều ô con và kết hợp các ô con đó thành một hình duy nhất. Vì vậy, thay vì sử dụng thêm một bar chart, chúng ta có thể có quan sát được nhiều biểu đồ hơn, được sắp xếp cùng nhau trong một lưới.

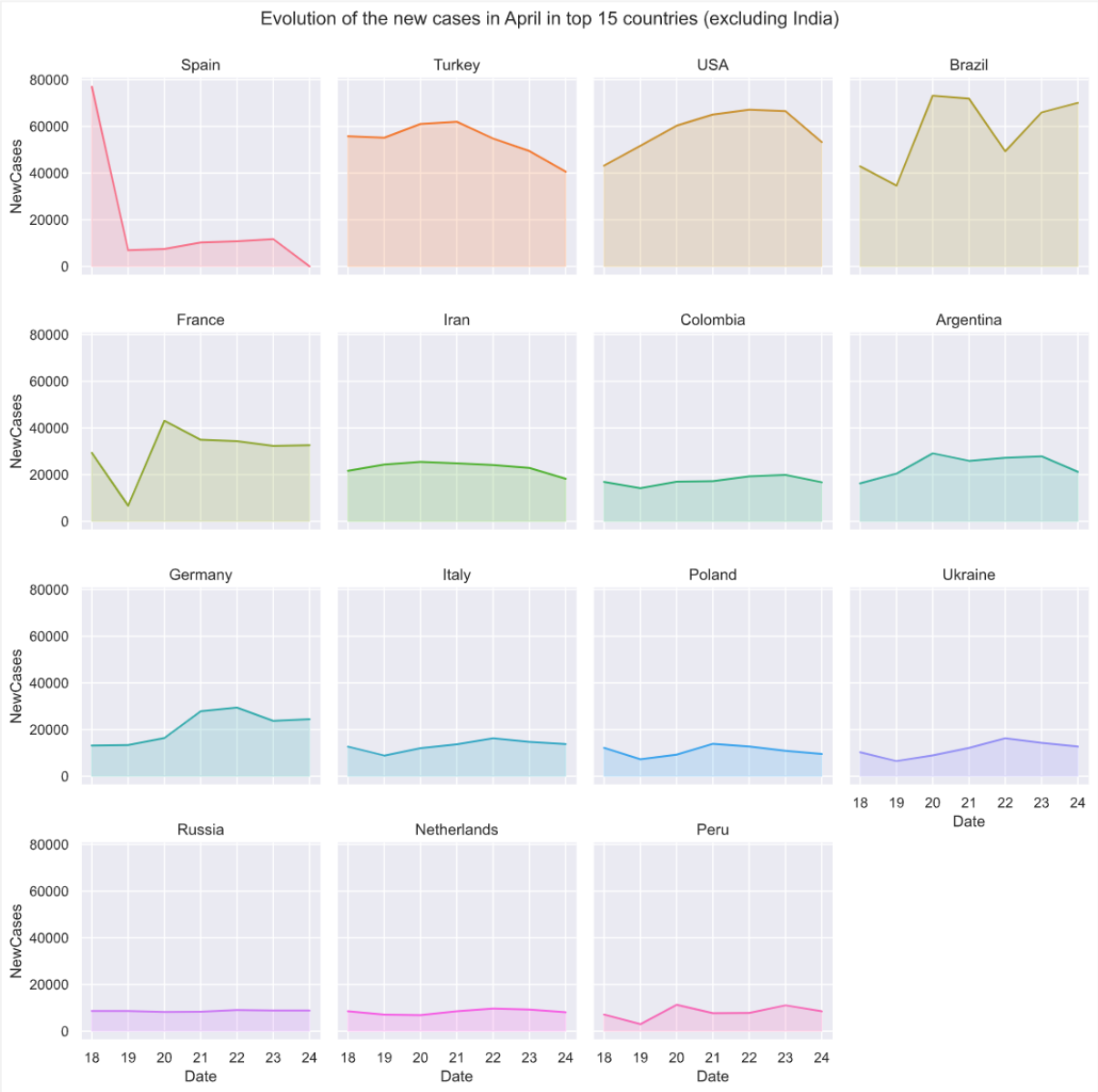



*Hình 17 Sự biến chuyển của ca mới ở 16 nước đứng đầu*

**Nhận xét:** Ấn Độ độc chiếm tier 1, dẫn đầu về số ca mắc mới, gấp nhiều lần so với các nước trong tier 2 như Mỹ, Brazil, Thổ Nhĩ Kỳ, Tây Ban Nha. Đặc biệt từ ngày 19, số ca mắc có chiều hướng tăng mạnh theo từng ngày. Trong các ngày tới, Ấn Độ có thể sẽ cán mốc 400,000 ca/ngày.

Các nước tier 3 như Ukraine, Ba Lan, Peru, Nga, ... thì có số ca mắc mới chỉ bằng 1/2 các nước tier 2.

Tạm loại bỏ Ấn Độ để xem sự thay đổi của các nước này tốt hơn.




*Hình 18 Sự thay đổi của 15 nước ( bỏ đi Ấn Độ)*

**Nhận xét:** Số ca nhiễm mới ở Mỹ, Brazil và Thổ Nhĩ Kỳ có giá trị cao gấp 2 đến 3 lần so với các nước khác trong top. Song các nước đều có xu hướng giảm đều, chỉ có số ca mắc mới trong ngày ở Brazil đang có chiều hướng tăng trở lại.

Pháp cũng ghi nhận số ca mắc mới tăng đột biến trong ngày 19 nhưng cũng giảm dần ổn định ngay sau đó.

Đặc biệt, ở Tây Ban Nha có một sự giảm ca mới đáng kể trong ngày 19, từ gần 80,000 xuống chỉ còn dưới 20,000 ca. Các nước còn lại cũng ghi nhận số ca mắc mới có biên độ chênh lệch giữa các ngày không biến động nhiều.

### 3.2.3. Lolipop Chart

**Ngôn ngữ sử dụng**: Python, Jupyter Notebook

**Lý do sử dụng:** giúp quan sát giá trị quan sát cạnh nhau trên cùng một dòng tốt hơn, bằng cách chỉ hiển thị sự khác biệt của chúng thay vì hiển thị các giá trị trên các dòng khác nhau

**Quan hệ biểu diễn:** Số ca nhiễm mới xuất hiện và số ca đã hồi phục trong ngày (23/04/2021) của từng quốc gia.

**Thư viện sử dụng**:

- Matplotlib
- Seaborn
- Numpy
- Pandas

**Kỹ thuật vẽ**:

1. Tiến hành import các thư viện cần thiết (đã thực hiện ở các bước trên)
2. Tiến hành import data từ file csv vào dataframe và tiền xử lý dữ liệu

```
In [42]: def createDFLolipop(dataset):
             df = cleanerForNewCases(dataset)
             is_country = df['Country,Other'].isin(country)
             df = df[is_country]
             df = df[['Country,Other','NewCases','NewRecovered']]
             df = df.sort_values(['NewCases'],ascending=False)
             return df
```

*Hình 19 Import dữ liệu vào dataframe (Lolipop Chart)*

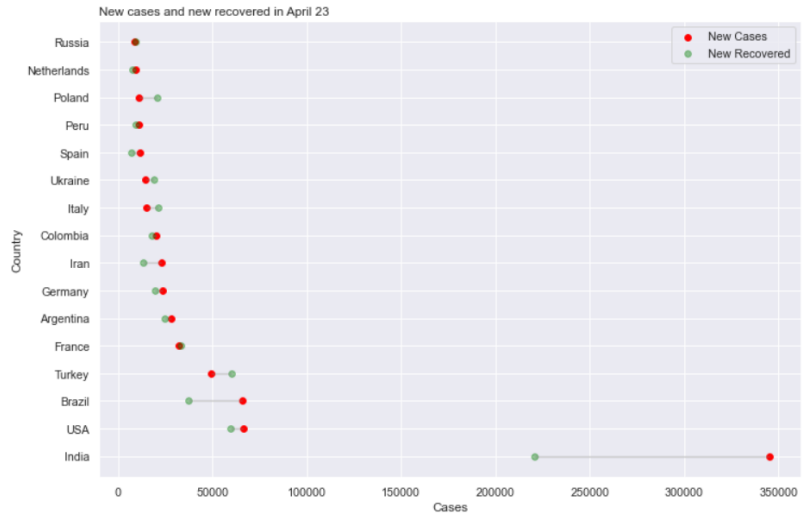
3. Tiến hành gọi hàm để vẽ

```
In [43]: def lolipopChart(df,title):
             my_range=range(1,len(df.index)+1)
             # The horizontal plot is made using the hline function
             plt.hlines(y=my_range, xmin=df['NewCases'], xmax=df['NewRecovered'], color='grey', alpha=0.4)
             plt.scatter(df['NewCases'], my_range, color='red', alpha=1, label='New Cases')
             plt.scatter(df['NewRecovered'], my_range, color='green', alpha=0.4 , label='New Recovered')
             plt.legend()
             # Add title and axis names
             plt.yticks(my_range, df['Country,Other'])
             plt.title(title, loc='left')
             plt.xlabel('Cases')
             plt.ylabel('Country')
             # Show the graph
             plt.show()
```

*Hình 20 Tiến hành vẽ Lolipop Chart*

4. Kết quả sau khi chạy

```
In [44]: country = ["India","USA","Brazil","Turkey","France","Argentina","Germany","Iran","Colombia","Italy","Ukraine","Sp
df = createDFLolipop(raw_data_23)
lolipopChart(df,'New cases and new recovered in April 23')
```

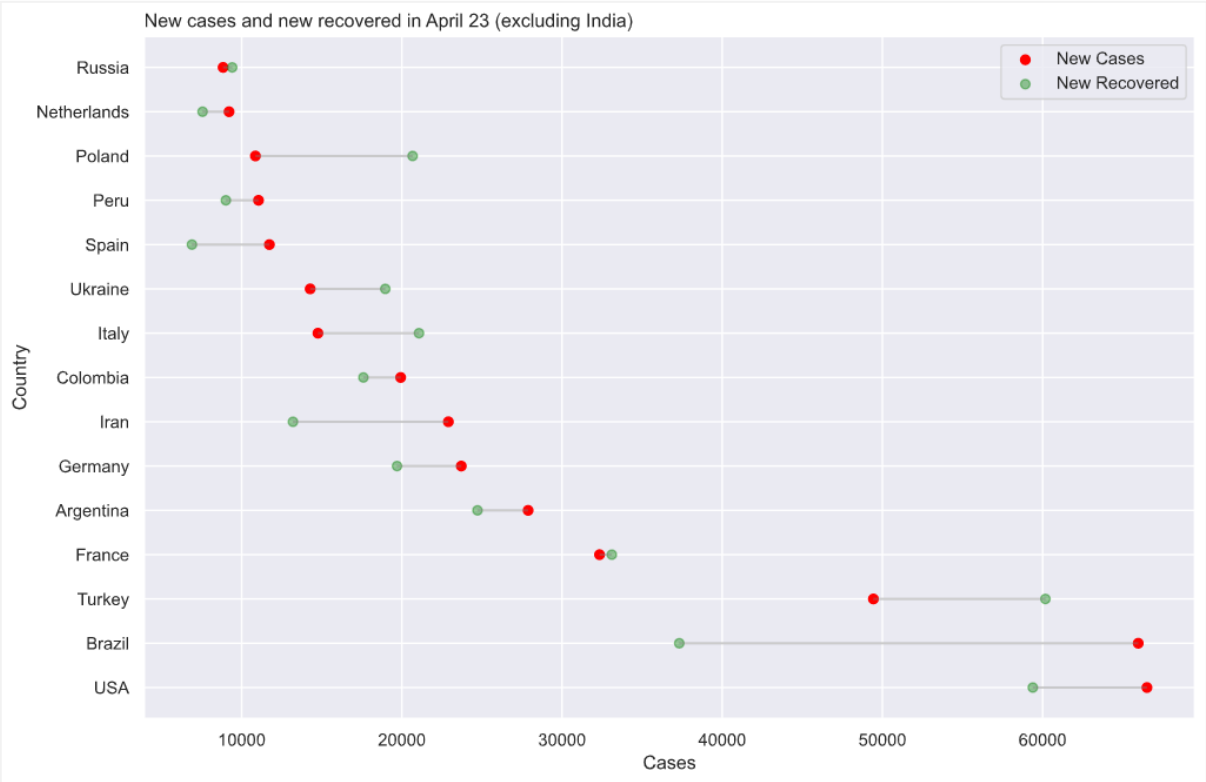



*Hình 21 Kết quả Lolipop Chart sau khi chạy*

**Nhận xét:**

Có thể thấy, không chỉ đứng đầu về số ca mới bùng phát trong 1 ngày, Ấn Độ cũng dẫn đầu các nước khác về số ca hồi phục, điều này là hợp lí với tỉ lệ dân số. Tuy nhiên, nếu tập trung về khoảng cách, thì sự chênh lệch giữa ca nhiễm mới và ca hồi phục vẫn còn khá lớn, cho thấy tốc độ hồi phục vẫn còn rất chậm so với tốc độ lây nhiễm. Nếu không có biện pháp khắc phục, Ấn Độ có thể vỡ trận phòng chống COVID-19.

⇨ Do chênh lệch khá lớn. Ta tạm loại bỏ Ấn Độ để xem sự thay đổi của các nước còn lại tốt hơn.




**Nhận xét:** Như vậy, tuy đứng thứ 2 về lượt nhiễm mới, nhưng số ca hồi phục của Mỹ cũng có chiều hướng gần hơn với số ca lây nhiễm.

Đặc biệt là Thổ Nhĩ Kỳ, tuy đứng thứ ba về số ca mắc mới, nhưng số ca hồi lại phục vượt đến hơn 10,000 ca so với ca mắc mới, đây là tín hiệu tốt. Tương tự với quốc gia khác như Ba Lan, Ukraine, Ý.

Các quốc gia còn lại theo xu hướng chung là số ca mới sẽ nhiều hơn một phần so với số ca hồi phục. Đáng lưu ý là Brazil, khi chênh lệch giữa số ca mới và số ca hồi phục là khá lớn. Nếu không cẩn thận phòng chống, rất có thể sẽ kết cục như Ấn Độ.

### 3.2.4. Line Chart with Moving Average Technique

**Ngôn ngữ sử dụng:** Python thuần

**Lý do sử dụng:** Mục đích của việc sử dụng biểu đồ đường chính là thể hiện rõ nét xu hướng thay đổi của dữ liệu - ở đây là xu hướng thay đổi của số ca nhiễm mới và số ca tử vong - qua đó có thể nhận xét về xu hướng cũng như là sự tương quan giữa 2 trường dữ liệu.

**Quan hệ biểu diễn:** Số ca nhiễm mới và Số ca tử vong của Ấn Độ từ 15/02/2020 đến ngày 28/04/2021

**Thư viện sử dụng:**

- Pandas: sử dụng để xử lý dataframe
- Matplotlib: dùng để vẽ đồ thị
- Datetime: xử lý ngày tháng cần vẽ

**Kỹ thuật vẽ:**

1. Tiến hành import các thư viện cần thiết vào chương trình

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3  import datetime as dt
```

*Hình 22 Import thư viện (Line Chart)*

2. Tiến hành đọc file csv dữ liệu vào dataframe, khởi tạo các biến để chứa các cột dữ liệu

```
7  # import and normalize data - remove null with 0
8  df=pd.read_csv('india_newcase_newdeath.csv')
9  df['Daily new cases'] = df['Daily new cases'].fillna(0)
10 df['Daily new deaths'] = df['Daily new deaths'].fillna(0)
11
12 newCase = df['Daily new cases']
13 newDeath = df['Daily new deaths']
```

*Hình 23 Đọc dữ liệu vào dataframe*

Dòng 8: tiến hành import dữ liệu từ file csv vào dataframe
Dòng 9,10: tiến hành điền các cột không dữ liệu ( hàng trống) thành 0 ở 2 cột Daily new cases và Daily new deaths
Dòng 12,13: gán dữ liệu 2 cột Daily new cases và Daily new deaths vào 2 biến newCase và biến newDeath

3. Tiến hành tính tổng dữ liệu của 2 cột newCase và newDeath và tính % của từng phần tử trong cột so với tổng dữ liệu đã tính

```
15 sumNewCase = df['Daily new cases'].sum()
16 sumNewDeath = df['Daily new deaths'].sum()
17
18 percentageNewCase = df['Daily new cases']*100/sumNewCase
19 percentageNewDeath = df['Daily new deaths']*100/sumNewDeath
```

*Hình 24 Tiền xử lý dữ liệu trước khi vẽ*

Dòng 15,16: tính tổng của 2 cột dữ liệu
Dòng 18,19: tính % của từng phần tử và gán vào 2 biến percentageNewCase và percentageNewDeath

4. Tiến hành sử dụng kỹ thuật moving average (trung bình trượt) để thực hiện vẽ line chart. Ở bước này, chuẩn bị về mặt dữ liệu

```
33  newCaseSeries = pd.Series(percentageNewCase)
34
35  windowsNewCase = newCaseSeries.rolling(windowSize)
36
37  movingAvrNewCase = windowsNewCase.mean()
38
39  movingAvrNewCaseList = movingAvrNewCase.tolist()
40
41  avrNewCaseWithNoNaN = movingAvrNewCaseList[windowSize - 1:]
```

*Hình 25 Xử lý Moving Average với cột NewCase*

Dòng 33: tiến hành biến percentageNewCase thành dạng chuỗi
Dòng 35,37: tiến hành slide window ( ở đây là 7) và tính trung bình của 7 ngày và gán nó vào ngày thứ 7
Dòng 41: bỏ đi 6 giá trị đầu tiên không được tính

5. Tiến hành làm tương tự với giá trị newDeath

```
43  # newDeath
44  newDeathSeries = pd.Series(percentageNewDeath)
45
46  windowsNewDeath = newDeathSeries.rolling(windowSize)
47
48  movingAvrNewDeath = windowsNewDeath.mean()
49
50  movingAvrNewDeathList = movingAvrNewDeath.tolist()
51
52  avrNewDeathWithNoNaN = movingAvrNewDeathList[windowSize - 1:]
```

*Hình 26 Xử lý Moving Average với NewDeath*

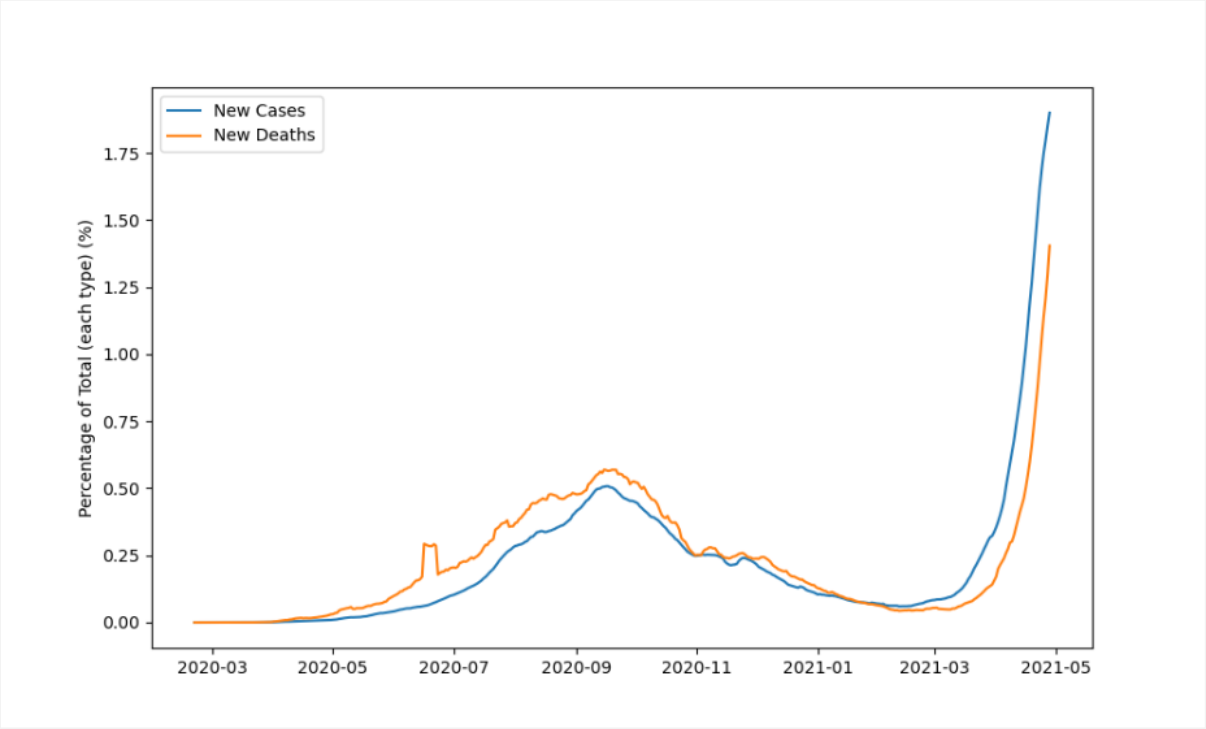
6. Sau khi đã xử lý dữ liệu, tiến hành vẽ biểu đồ

```
58  fig=plt.figure(figsize=(10,6))
59  x_date = [dt.datetime.strptime(d, '%d/%m/%Y').date() for d in dateData]
60  plt.plot(x_date, avrNewCaseWithNoNaN, label="New Cases")
61  plt.plot(x_date, avrNewDeathWithNoNaN, label="New Deaths")
```

*Hình 27 Tiến hành vẽ Line Chart*

Dòng 58: tạo figure có kích thước (10,6)
Dòng 59: tạo mảng là trục x theo thời gian từ biến dateData đã tạo ở bước trên
Dòng 60: vẽ line chart theo biến NewCase đã xử lý moving average và đặt tên được là New Cases
Dòng 61: vẽ line chart theo biến NewDeath đã xử lý moving average và đặt tên là New Deaths

7. Kết quả sau khi chạy




*Hình 28 Kết quả sau khi vẽ Line Chart*

**Nhận xét:**

Biểu đồ trên là hình ảnh trực quan của Số ca nhiễm mới và số ca tử vong hàng ngày tính theo tỉ lệ phần trăm trên tổng số lượng mỗi loại của Ấn Độ kể từ đầu dịch Covid-19.

Từ biểu đồ ta có thể rút ra các nhận xét sau:

Ta có thể thấy Số ca mắc mới và số ca tử vong mỗi ngày có sự tương đồng trong xu hướng thay đổi
+ Số ca mắc mới và số ca tử vong trong 1 ngày đã tăng dần đều từ đầu dịch cho đến giữa tháng 9/ 2020
+ Số ca mắc mới và số ca tử vong trong 1 ngày đã cùng đạt cực đại (địa phương) vào khoảng giữa tháng 9/2020
+ Số ca mắc mới và số ca tử vong đã giảm dần từ tháng 9/2020 và chạm đáy vào khoảng nửa cuối tháng 2/2021
+ Từ tháng 3/2021 đến nay, số lượng ca mắc mới và ca tử vong bắt đầu tăng mạnh và chưa có dấu hiệu đạt đỉnh.

Kết luận: Dữ liệu của Số ca mắc mới và Số ca tử vong mỗi ngày ở Ấn Độ có xu hướng giống nhau nhau. Ta sẽ xem xét mối quan hệ này bằng cách thiết lập mô hình hồi quy tuyến tính giữa Số ca mắc mới và Số ca tử vong để có góc nhìn cụ thể hơn về mối quan hệ giữa 2 trường dữ liệu

### 3.2.5. Scatter Chart

**Ngôn ngữ sử dụng:** Python thuần

**Lý do sử dụng:** Scatter Chart thích hợp cho việc quan sát 2 biến và thể hiện mối quan hệ giữa chúng, ngoài ra còn có thể quan sát được pattern(mẫu) của dữ liệu

**Quan hệ biểu diễn:** Số ca nhiễm mới và Số ca tử vong của Ấn Độ từ 15/02/2020 đến ngày 28/04/2021

**Thư viện sử dụng:**

- Pandas: dùng để import dataframe và xử lý số liệu
- Numpy
- Scipy: dùng để tính giá trị a và b của đường hồi quy
- Csv: dùng để đọc file csv
- Matplotlib: dùng để vẽ biểu đồ

**Kỹ thuật vẽ:**

1. Tiến hành import các thư viện cần thiết

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3  import numpy as np
4  from scipy import stats
```

*Hình 29 Import thư viện (Scatter Chart)*

2. Tiến hành đọc file csv dữ liệu load vào dataframe và xử lý số liệu ở dataframe

```
6  df=pd.read_csv('india_newcase_newdeath.csv')
7  df['Daily new cases'] = df['Daily new cases'].fillna(0)
8  df['Daily new deaths'] = df['Daily new deaths'].fillna(0)
9
10 newCase = df['Daily new cases']
11 newDeath = df['Daily new deaths']
```

*Hình 30 Import dữ liệu vào dataframe*

Dòng 6: tiến hành đọc file csv load vào dataframe
Dòng 7,8: tiến hành điền số 0 vào các cột dữ liệu trống ở 2 cột Daily new cases và Daily new deaths
Dòng 8,9: gán dữ liệu 2 cột Daily new cases và cột Daily new deaths vào 2 biến newCase và newDeath

3. Tiến hành tính toán đường hồi quy và vẽ biểu đồ scatter

```
14    slope, intercept, r_value, p_value, std_err = stats.linregress(newCase, newDeath)
15    fig=plt.figure(figsize=(10,6))
16    plt.scatter(newCase, newDeath)
```

*Hình 31 Tiến hành vẽ Scatter Chart*

Dòng 14: tính toán các giá trị cho đường hồi quy
Dòng 16: tiến hành vẽ biểu đồ scatter theo 2 trường dữ liệu newCase và newDeath

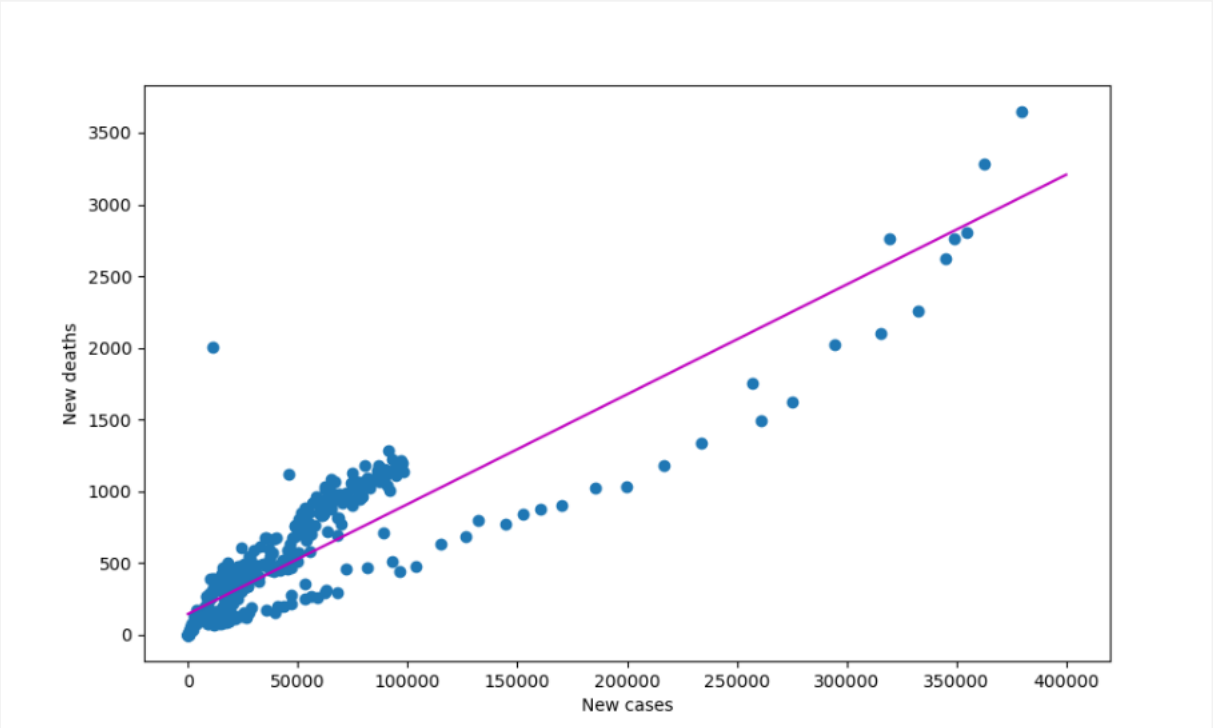
4. Tiến hành vẽ đường hồi quy dựa trên giá trị slope và intercept đã tính

```
24    x=np.arange(0, 400000, 1)
25    line = slope*x+intercept
26    plt.plot(x,line,c='m')
27    plt.show()
```

*Hình 32 Tiến hành vẽ đường hồi quy*

Dòng 24: tạo mảng x có giá trị từ 0 tới 400000 (step là 1)
Dòng 26: tiến hành vẽ đường hồi quy
Dòng 27: hiển thị biểu đồ đã vẽ

5. Kết quả sau khi vẽ:




*Hình 33 Kết quả sau khi vẽ Scatter và đường hồi quy*

**Nhận xét:**

Từ biểu đồ trên ta có thể thấy được Số ca tử vong đồng biến với Số ca mắc mới (Positive Correlation)

Gọi Số ca mắc mới là x, Số ca tử vong là y. Ta có thể biểu diễn y qua x bằng công thức sau

*y = 0.0076508867281668185*x + 146.42323700356758*

Đồng thời ta tính được một số giá trị về mặt thống kê:

*p – value = 4.534349991282551e-157*

*r – squared = 0.8046800860120003*

Ta có thể đưa ra các kết luận như sau:

- *Biến Số ca mắc mới có ý nghĩa đối với mô hình (hồi quy tuyến tính) về mặt thống kê; Mô hình phù hợp tốt với dữ liệu quan sát về mặt thống kê*
  *(p–value =4.534349991282551e-157)*
- *Biến Số ca mắc mới có thể giải thích được 80,47% sự thay đổi của biến Số ca tử vong*
- *Phương trình hồi quy:*

*Số ca tử vong= 0.0076508867281668185 * Số ca mắc mới + 146.42323700356758*

### 3.2.6. Bar Chart

**Ngôn ngữ sử dụng:** Python thuần

**Lý do sử dụng:** Bar Chart cho phép người dùng dễ hình dung về cao độ và hiển thị được bộ dữ liệu lớn và dễ xếp hạng

**Quan hệ biểu diễn:** Tỉ lệ về tổng số case trên 1 triệu dân, tổng số ca trên 1 triệu dân, số lượng. ca tử vong mới trên 1 triệu dân, số lượng ca hiện tại trên 1 triệu ở 30 quốc giá có giá trị cao nhất

**Thư viện sử dụng:**

- Pandas: dùng để import dataframe và xử lý số liệu
- Numpy
- Csv: dùng để đọc file csv
- Matplotlib: dùng để vẽ biểu đồ

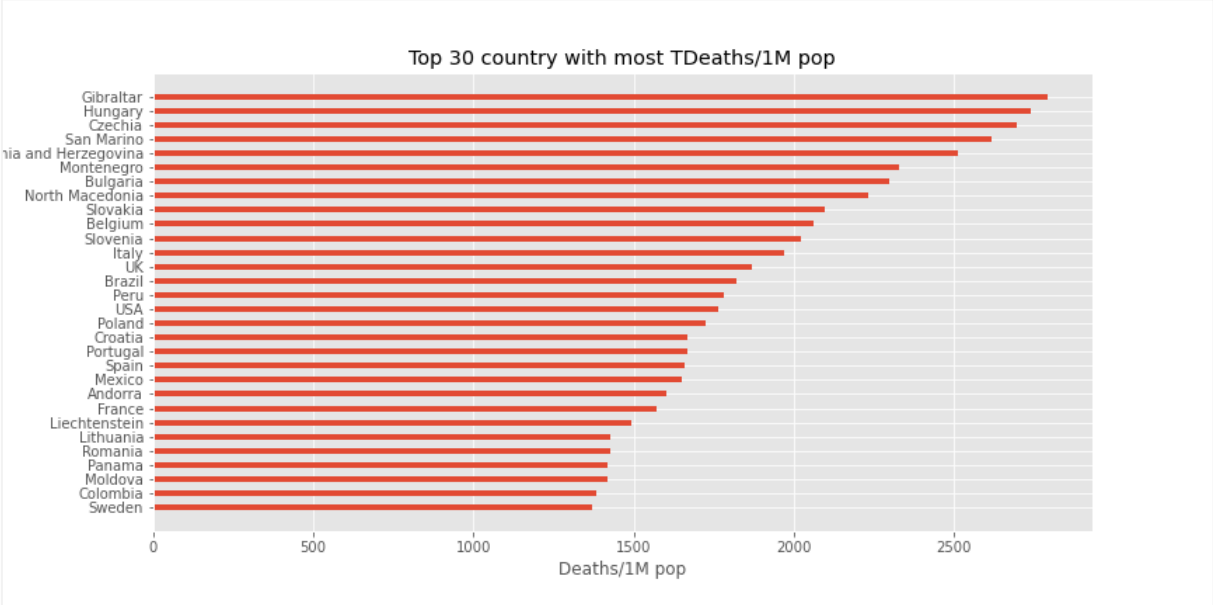
**Kỹ thuật vẽ:**

1. import các thư viện cần thiết
2. Đọc dữ liệu từ datashet đã qua tiền xử lí
3. Xây dựng hàm vẽ đồ thị thang theo dữ liệu chọn lọc

```
def drawBarChart(ax, X, Y, xlabel, ylabel, title):
    ind = np.arange(len(top30)) * 2
    ax.set_yticks(ind)
    ax.set_yticklabels(Y)
    ax.set_xlabel(xlabel)
    ax.set_ylabel(ylabel)
    ax.set_title(title)
    ax.barh(ind, X)
```

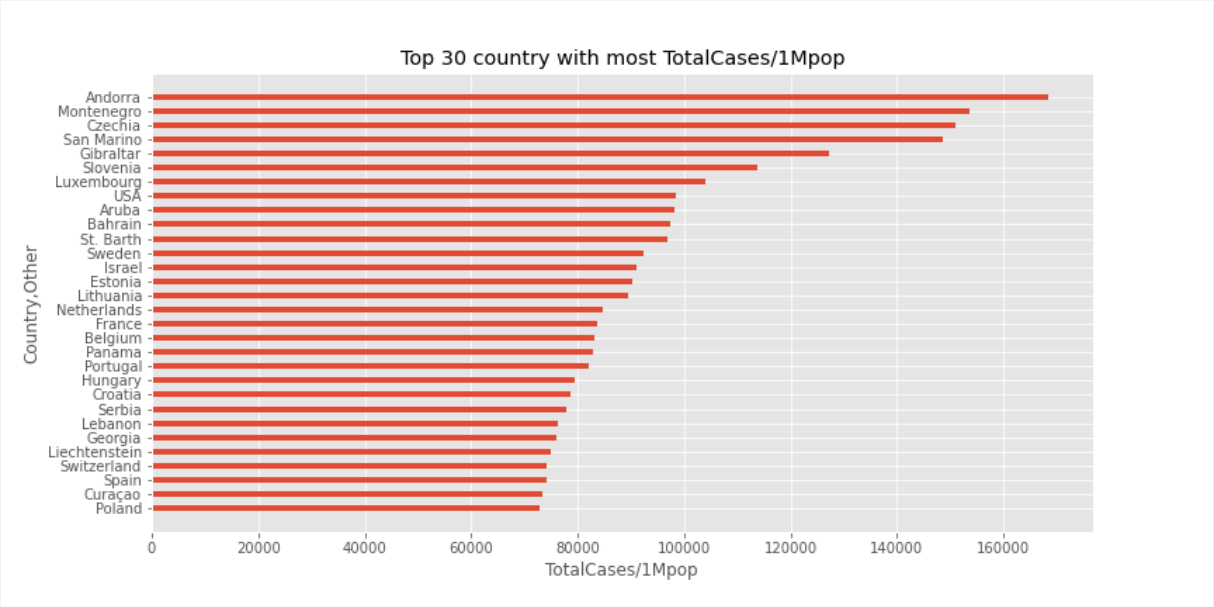
*Hình 34 Tiến hành vẽ Bar Chart*

**Nhận xét:**



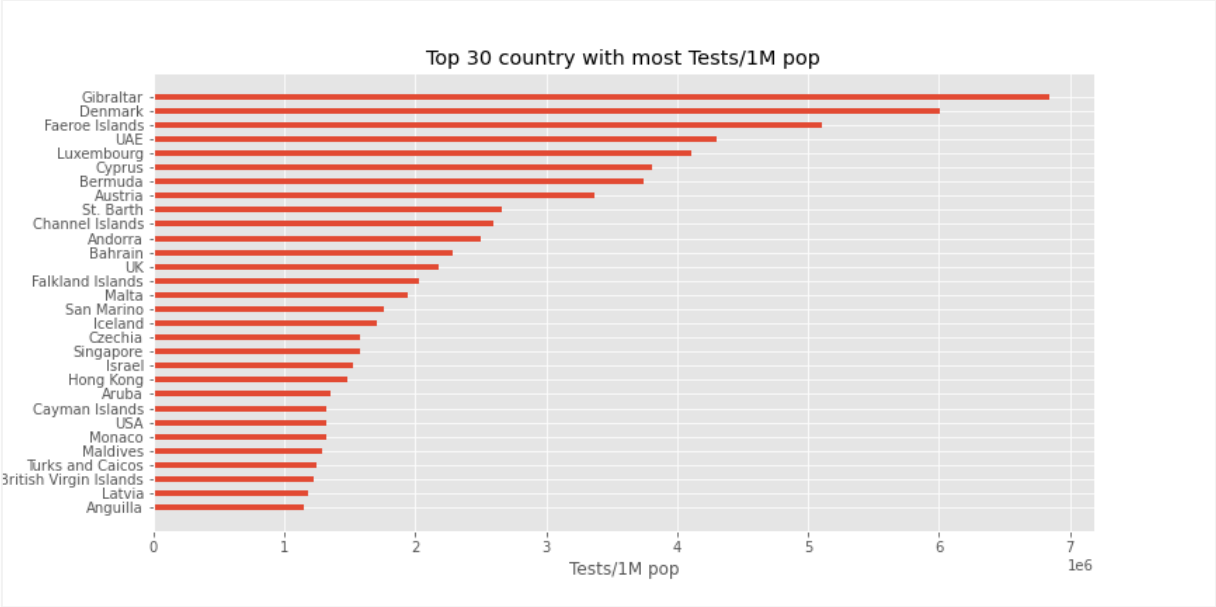

*Hình 35 Bar Chart 30 nước có tỉ lệ tử vong/1 triệu người cao nhất*

- Đối với Bar Chart tỷ lệ người nhiễm
  - Các quốc gia có tỷ lệ người nhiễm cao nhất thế giới phân bố phần lớn ở Châu Âu, còn lại là châu Á và châu Mĩ
  - Quốc gia có tỷ lệ người nhiễm bệnh lớn nhất là Andorra với hơn 160000 ca nhiễm/1 triệu dân, tức là cứ 6 người sẽ có 1 người bị nhiễm
  - Tiếp sau đó là các nước như Montenegro, Czechia, San Marino, Solvenia, Luxembourg, USA cũng có tỷ lệ người nhiễm bệnh rất cao




*Hình 36 Bar Chart 30 nước có tổng số ca/1 triệu dân cao nhất*

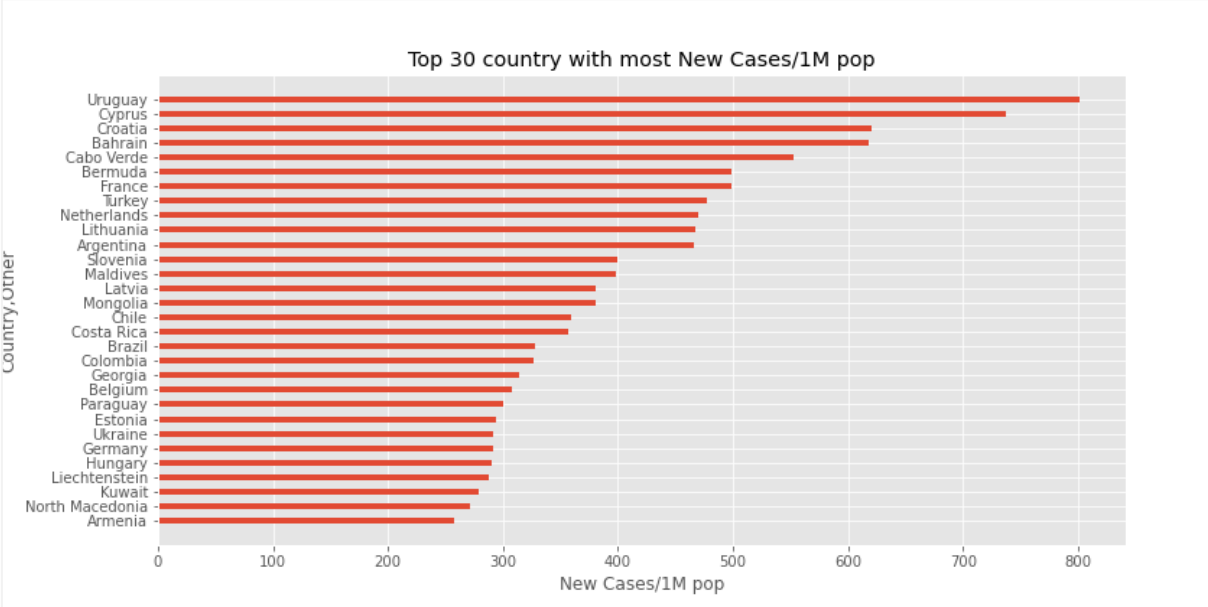
- Đối với Bar Chart tỷ lệ ca tử vong
  - Các quốc gia có tỷ lệ người chết vì dịch bệnh cao nhất thế giới phân bố phần lớn ở Châu Âu, còn lại là châu Á và châu Mĩ
  - Quốc gia có tỷ lệ người nhiễm bệnh lớn nhất là Gibralatar với gần 2700 ca tử vong/1 triệu dân, tức là cứ 3700 người sẽ có 1 người chết vì dịch bệnh
  - Tiếp sau đó là các nước như Hungary, Czechia, San Marino, herzagovina, Bulgaria, USA cũng có tỷ lệ người chết vì bệnh dịch rất cao




*Hình 37 Bar Chart 30 nước có số lượng lượt test/1 triệu dân cao nhất*

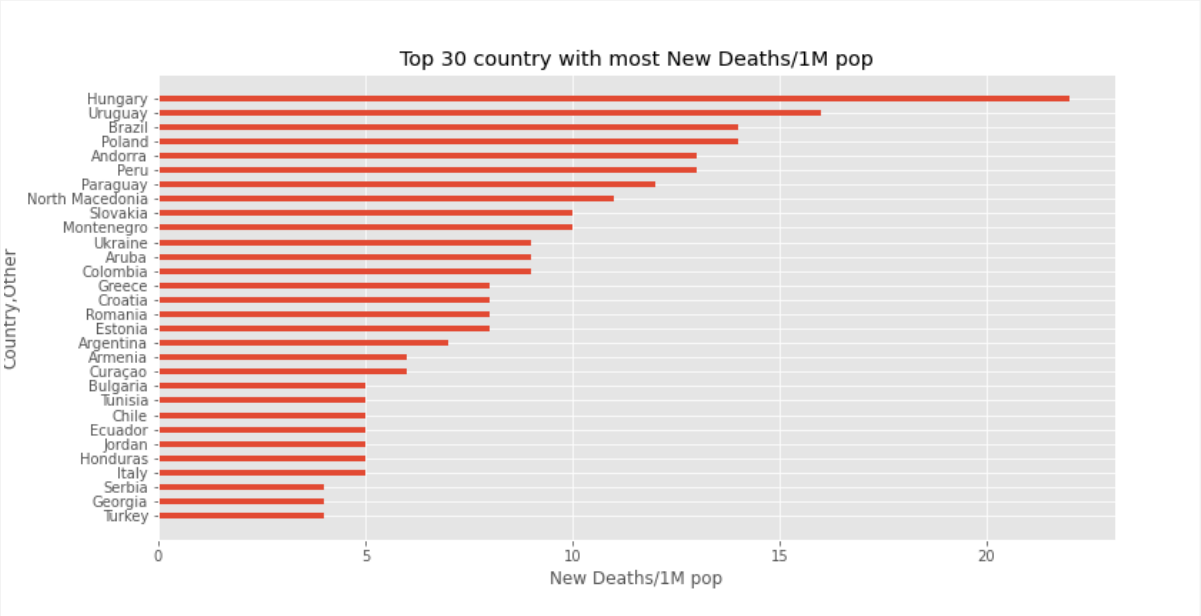
- Đối với Bar Chart có số lượng lượt test

- Các quốc gia có tỷ lệ thực hiện kiểm tra bệnh dịch cao nhất thế giới phân bố phần lớn ở Châu Âu, còn lại là châu Á và châu Mĩ
- Quốc gia có tỷ lệ thực hiện kiểm tra bệnh dịch lớn nhất là Gibralatar với hơn 7x10^6 cuộc kiểm tra/1 triệu dân, tức là cứ 1 người sẽ thực hiện trung bình 7 cuộc kiểm tra
- Tiếp sau đó là các nước như Denmark, UAE, San Marino, herzagovina, Bulgaria, USA cũng có tỷ lệ thực hiện kiểm tra bệnh dịch rất cao



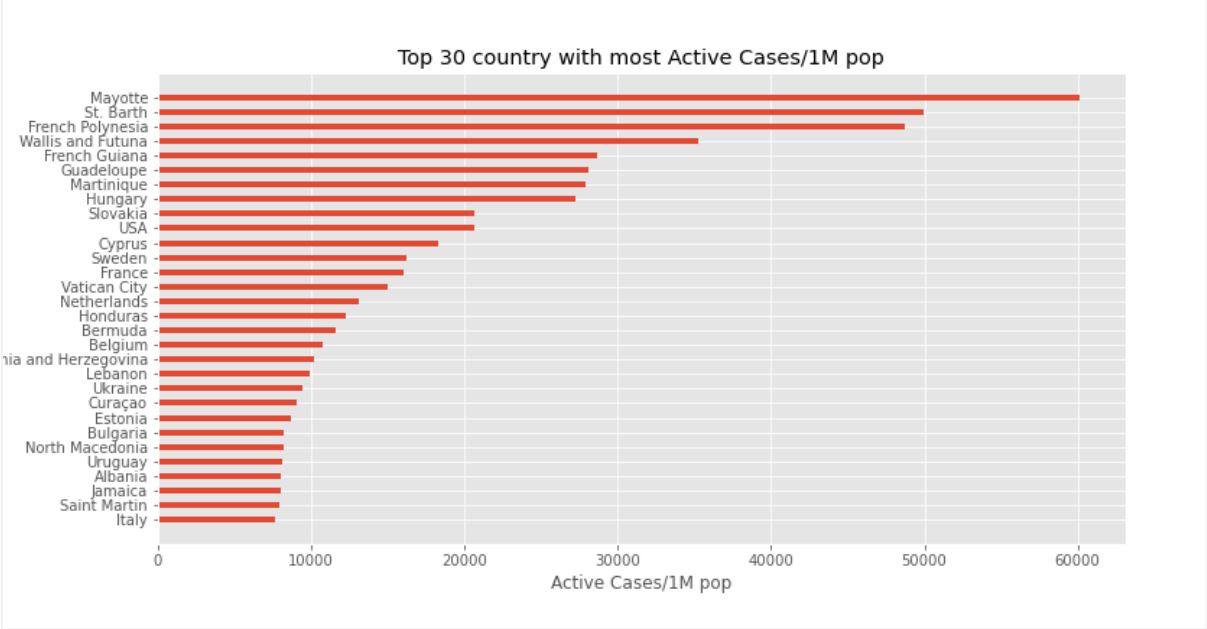

*Hình 38 Bar Chart có 30 nước có số lượng ca mới/1 triệu dân lớn nhất*

- Đối với Bar Chart tỉ lệ ca nhiễm
  - Các quốc gia có tỷ lệ ca nhiễm mới cao nhất thế giới phân bố phần lớn ở Châu Âu, còn lại là châu Á và châu Mĩ
  - Quốc gia có tỷ lệ ca nhiễm mới lớn nhất là Uruquay với hơn 800 ca nhiễm mới/1 triệu dân, tức là cứ 12500 người sẽ có 1 ca nhiễm mới
  - Tiếp sau đó là các nước như Cyprus, Croatia, San Marino, Bermuda, France, USA cũng có tỷ lệ ca nhiễm mới rất cao




*Hình 39 Bar Chart 30 nước có số ca tử vong mới/1 triệu dân cao nhất*

- Đối với Bar Chart tỷ lệ ca tử vong mới
  - Các quốc gia có tỷ lệ ca tử vong mới cao nhất thế giới phân bố phần lớn ở Châu Âu, còn lại là châu Á và châu Mĩ
  - Quốc gia có tỷ lệ ca tử vong mới lớn nhất là Hungary với gần 25 ca tử vong mới/1 triệu dân, tức là cứ 400000 người sẽ có 1 ca tử vong mới
  - Tiếp sau đó là các nước như Uruquay, Brazil, Poland, Peru, Paraquay, Ukraine cũng có tỷ lệ ca tử vong mới rất cao




*Hình 40 Bar Chart 30 nước có số ca hiện tại/1 triệu dân cao nhất*

- Đối với Bar Chart tỷ lệ ca dương tính mới
  - Các quốc gia có tỷ lệ ca dương tính mới cao nhất thế giới phân bố phần lớn ở Châu Âu, còn lại là châu Á và châu Mĩ

- Quốc gia có tỷ lệ ca dương tính mới lớn nhất là Mayotte với hơn 600000 ca dương tính mới/1 triệu dân, tức là cứ 17 người sẽ có 1 ca dương tính mới
- Tiếp sau đó là các nước như St.Barth, France Ponelesia, Poland, Peru, Hungary, Ukraine cũng có tỷ lệ ca dương tính mới rất cao

### 3.2.7. Pie Chart

**Ngôn ngữ sử dụng:** Python thuần

**Lý do sử dụng:** Pie chart giúp hiển thị tỉ lệ giữa các loại với nhau dễ dàng và dễ hình dung

**Quan hệ biểu diễn:** Trực quan tình trạng người nhiễm bệnh của mỗi quốc gia kể trên, có 3 tình trạng là Recovered, Active, Death. TotalRecovereds+TotalActives+TotalDeaths = TotalCases

**Thư viện sử dụng:**

- Pandas: dùng để import dataframe và xử lý số liệu
- Numpy
- Csv: dùng để đọc file csv
- Matplotlib: dùng để vẽ biểu đồ

**Kỹ thuật vẽ:**

1. Import các thư viện cần thiết
2. Nhập dữ liệu từ datashet đã qua tiền xử lí
3. Xây dựng hàm vẽ đồ thị tròn biểu diễn sự phân bố tình trạng người dân các nước

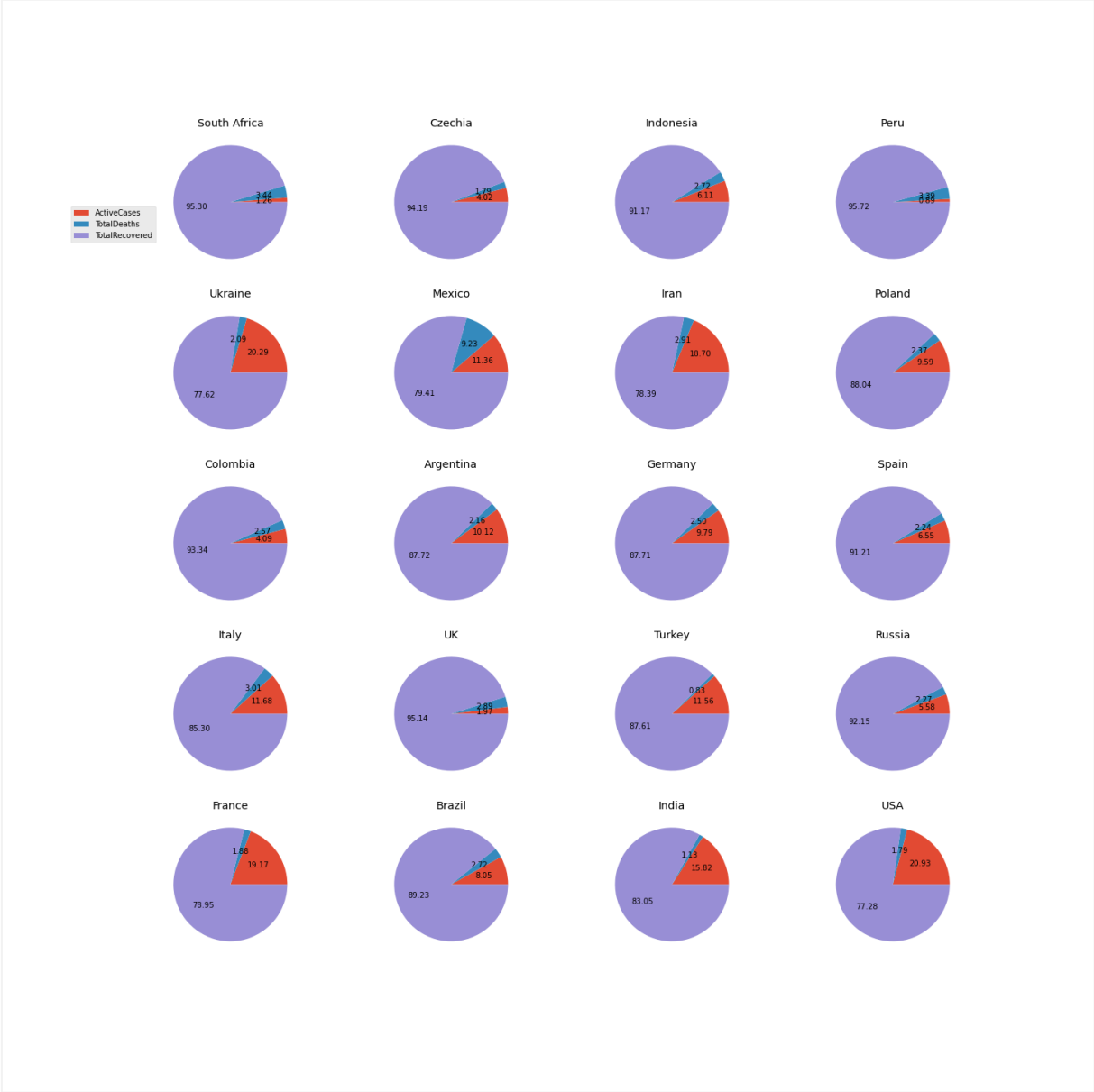
```
top20 = df.sort_values(by='TotalCases', ascending = False).head(20).sort_values(by='TotalCases', ascending = True)

X = top20['Country,Other']
activeCases = top20['ActiveCases']
totalDeaths = top20['TotalDeaths']
totalRecovered = top20['TotalRecovered']

fig, axes = plt.subplots(nrows=5, ncols=4, figsize=(20, 20))
for ax, country, a, b, c in zip(axes.flat, X, activeCases, totalDeaths, totalRecovered):
    ax.pie([a, b, c], autopct='%.2f')
    ax.set(ylabel='', title=country, aspect='equal')
axes[0, 0].legend( labels = ['ActiveCases','TotalDeaths','TotalRecovered'], bbox_to_anchor=(0, 0.5))
plt.show()
```

*Hình 41 Tiến hành vẽ Pie Chart*

**Nhận xét:**




*Hình 42 Kết quả sau khi vẽ Pie Chart*

- Các nước được sắp xếp tăng dần theo số ca nhiễm
- Quốc gia có số ca nhiễm lớn nhất thế giới là USA
- Trong các biểu đồ thì chiếm phần lớn chính là số người đã hồi phục, nhỏ hơn là đang dương tính cuối cùng là những người đã tử vong
- Chỉ riêng Peru và South Africa có tỷ lệ người tử vong cao hơn tỷ lệ người dương tính
- Tỷ lệ người đã phục hồi chiếm hơn 75%

## 6. Tham khảo

[1] "https://www.kite.com/python/answers/how-to-find-the-moving-average-of-a-list-in-python," [Trực tuyến].

[2] "https://stackoverflow.com/questions/53577630/how-to-make-pareto-chart-in-python," [Trực tuyến].

[3] "https://www.datafied.world/web-scraping-live-covid-19-data-and-its-analysis-190," [Trực tuyến].

[4] "https://www.python-graph-gallery.com/242-area-chart-and-faceting," [Trực tuyến].