Chapter 8 - exercise 2: Canada

Dữ liệu Canada.xlsx chứa thông tin nhập cư vào Canada từ năm 1980 đến năm 2013. Bộ dữ liệu chứa dữ liệu hàng năm về dòng người di cư đến Canada được ghi nhận, trình bày thông tin inflows and outflows theo nơi sinh, quốc tịch hoặc nơi cư trú trước đó / tiếp theo cho cả người nước ngoài và quốc tịch. chúng tôi sẽ tập trung vào dữ liệu nhập cư Canada

Map

1. Hiển thị bản đồ thế giới
2. Tạo bản đồ với center là Canada (location=[56.130, -106.35]) và zoom level (zoom_start=4)
3. Tạo Stamen Toner map với center là Canada với zoom level là 4.
4. Tạo Stamen Terrain với center là Canada với zoom level là 4.
5. Tạo Mapbox Bright Map với center là Canada with zoom level 6.

Choropleth Map

1. Đọc dữ liệu Canada.xlsx và gán vào df_can, tìm hiểu về dữ liệu với: describe, head, shape, columns
2. Làm sạch dữ liệu:
   - bỏ đi những cột không cần thiết như 'AREA','REG','DEV','Type','Coverage'
   - đổi tên cho một số cột 'OdName' => 'Country', 'AreaName' => 'Continent', 'RegName' => 'Region'
   - Để nhất quán tạo tên tất cả các cột trong dữ liệu là kiểu string
   - Thêm cột total chứa tổng lượng nhập cư các năm
3. Xem thông tin dữ liệu lúc này: head, shape
4. Tạo danh sách các năm từ 1980 đến 2013.
5. Lấy file GeoJSON có tên là world_countries.json
6. tạo world map, với center [0, 0] là *latitude* và *longitude*, zoom level là 2, sử dụng tiles là *Mapbox Bright*
7. Tạo choropleth map sử dụng total nhập cư của từng quốc gia vào Canada từ năm 1980 đến năm 2013
8. Chỉnh threshold = 0 thay cho threshold = -6,918, tạo lại choropleth map sử dụng total nhập cư của từng quốc gia vào Canada từ năm 1980 đến năm 2013

In [1]:
```
import folium
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Map

In [2]:
```
# define the world map
world_map = folium.Map()

# display world map
world_map
```

Out[2]:

In [3]:
```
# define the world map centered around Canada with a low zoom level
world_map = folium.Map(location=[56.130, -106.35], zoom_start=4)

# display world map
world_map
```

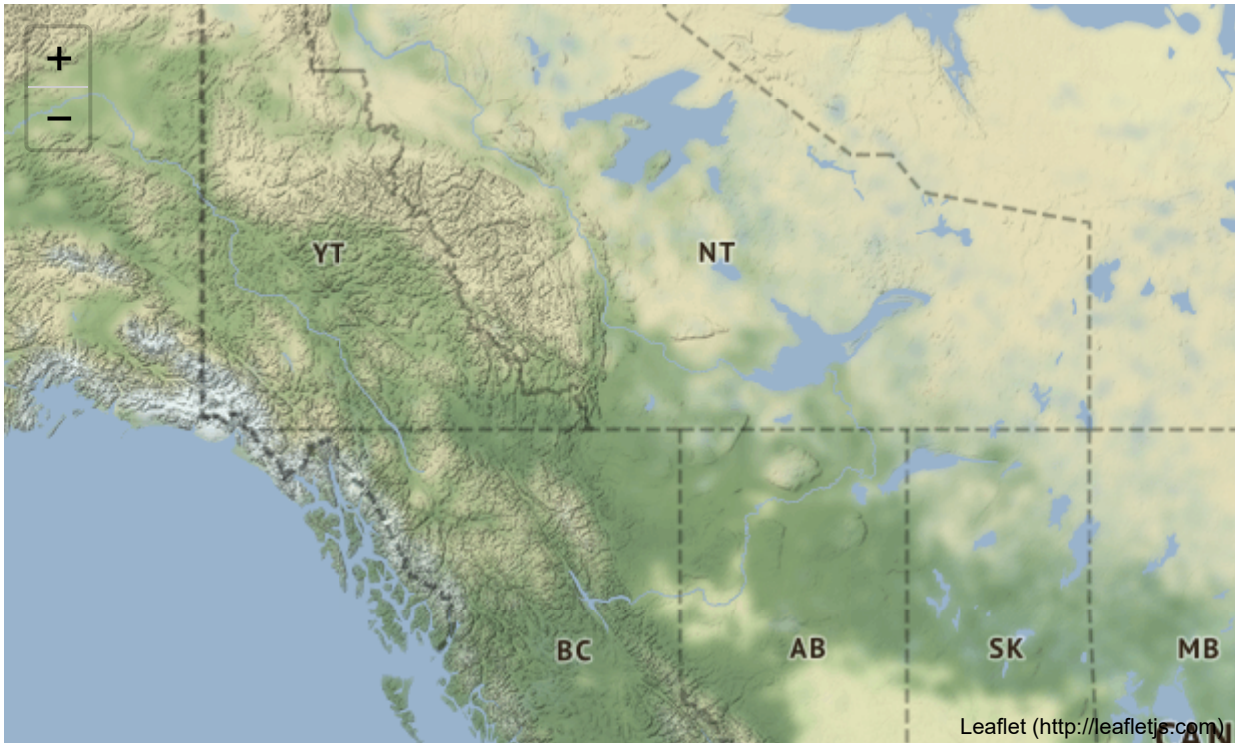
Out[3]:

In [4]:
```
# create a Stamen Toner map of the world centered around Canada with a zoom level
world_map = folium.Map(location=[56.130, -106.35], zoom_start=4, tiles='Stamen Ton

# display map
world_map
```

Out[4]:

In [5]:
```
# create a Stamen Terrain map of the world centered around Canada  with zoom level
world_map = folium.Map(location=[56.130, -106.35], zoom_start=4, tiles='Stamen Ter

# display map
world_map
```

Out[5]:



In [6]:
```
# create a Mapbox Bright Map of the world centered around Canada  with zoom level
world_map = folium.Map(location=[56.130, -106.35], zoom_start=6, tiles='Mapbox Bri

# display map
world_map
```

Out[6]:

Choropleth Map

1. Đọc dữ liệu Canada.xlsx và gán vào df_can, tìm hiểu về dữ liệu với: describe, head, shape, columns
2. Làm sạch dữ liệu:
   - bỏ đi những cột không cần thiết như 'AREA','REG','DEV','Type','Coverage'
   - đổi tên cho một số cột 'OdName' => 'Country', 'AreaName' => 'Continent', 'RegName' => 'Region'
   - Để nhất quán tạo tên tất cả các cột trong dữ liệu là kiểu string
   - Thêm cột total chứa tổng lượng nhập cư các năm
3. Xem thông tin dữ liệu lúc này: head, shape
4. Tạo danh sách các năm từ 1980 đến 2013.
5. Lấy file GeoJSON có tên là world_countries.json
6. tạo world map, với center [0, 0] là *latitude* và *longitude*, zoom level là 2, sử dụng tiles là *Mapbox Bright*
7. Tạo choropleth map sử dụng total nhập cư của từng quốc gia vào Canada từ năm 1980 đến năm 2013
8. Chỉnh threshold = 0 thay cho threshold = -6,918, tạo lại choropleth map sử dụng total nhập cư của từng quốc gia vào Canada từ năm 1980 đến năm 2013

In [7]:
```
df_can = pd.read_excel('data/Canada.xlsx',
                       sheet_name='Canada by Citizenship',
                       skiprows=range(20),
                       skipfooter=2)
```

In [8]:
```
df_can.describe()
```

Out[8]:

	AREA	REG	DEV	1980	1981	1982	198
count	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.00000
mean	912.764103	1249.015385	901.753846	508.394872	566.989744	534.723077	387.43589
std	13.082835	1185.526885	0.431878	1949.588546	2152.643752	1866.997511	1204.33359
min	903.000000	905.000000	901.000000	0.000000	0.000000	0.000000	0.00000
25%	903.000000	914.000000	902.000000	0.000000	0.000000	0.000000	0.00000
50%	908.000000	922.000000	902.000000	13.000000	10.000000	11.000000	12.00000
75%	922.000000	925.500000	902.000000	251.500000	295.500000	275.000000	173.00000
max	935.000000	5501.000000	902.000000	22045.000000	24796.000000	20620.000000	10015.00000

8 rows × 37 columns

In [9]:
```
df_can.head()
```

Out[9]:

	Type	Coverage	OdName	AREA	AreaName	REG	RegName	DEV	DevName	1980	.
0	Immigrants	Foreigners	Afghanistan	935	Asia	5501	Southern Asia	902	Developing regions	16	.
1	Immigrants	Foreigners	Albania	908	Europe	925	Southern Europe	901	Developed regions	1	.
2	Immigrants	Foreigners	Algeria	903	Africa	912	Northern Africa	902	Developing regions	80	.
3	Immigrants	Foreigners	American Samoa	909	Oceania	957	Polynesia	902	Developing regions	0	.
4	Immigrants	Foreigners	Andorra	908	Europe	925	Southern Europe	901	Developed regions	0	.

5 rows × 43 columns

In [10]:
```
# print the dimensions of the dataframe
df_can.shape
```

Out[10]:
```
(195, 43)
```

In [11]:
```
df_can.columns
```

Out[11]:
```
Index([    'Type', 'Coverage',   'OdName',     'AREA', 'AreaName',      'REG',
       'RegName',      'DEV',  'DevName',       1980,       1981,       1982,
            1983,       1984,       1985,       1986,       1987,       1988,
            1989,       1990,       1991,       1992,       1993,       1994,
            1995,       1996,       1997,       1998,       1999,       2000,
            2001,       2002,       2003,       2004,       2005,       2006,
            2007,       2008,       2009,       2010,       2011,       2012,
            2013],
      dtype='object')
```

In [12]:
```
# clean up the dataset to remove unnecessary columns (eg. REG)
df_can.drop(['AREA','REG','DEV','Type','Coverage'], axis=1, inplace=True)

# let's rename the columns so that they make sense
df_can.rename(columns={'OdName':'Country', 'AreaName':'Continent','RegName':'Regio

# for sake of consistency, let's also make all column labels of type string
df_can.columns = list(map(str, df_can.columns))

# add total column
df_can['Total'] = df_can.sum(axis=1)
```

In [13]:
```
df_can.shape
```

Out[13]:
```
(195, 39)
```

In [14]:
```
# the first five items of the cleaned dataframe.
df_can.head()
```

Out[14]:

	Country	Continent	Region	DevName	1980	1981	1982	1983	1984	1985	...	2005	200
0	Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	...	3436	300
1	Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	...	1223	85
2	Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	...	3626	480
3	American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0	...	0	
4	Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0	...	0	

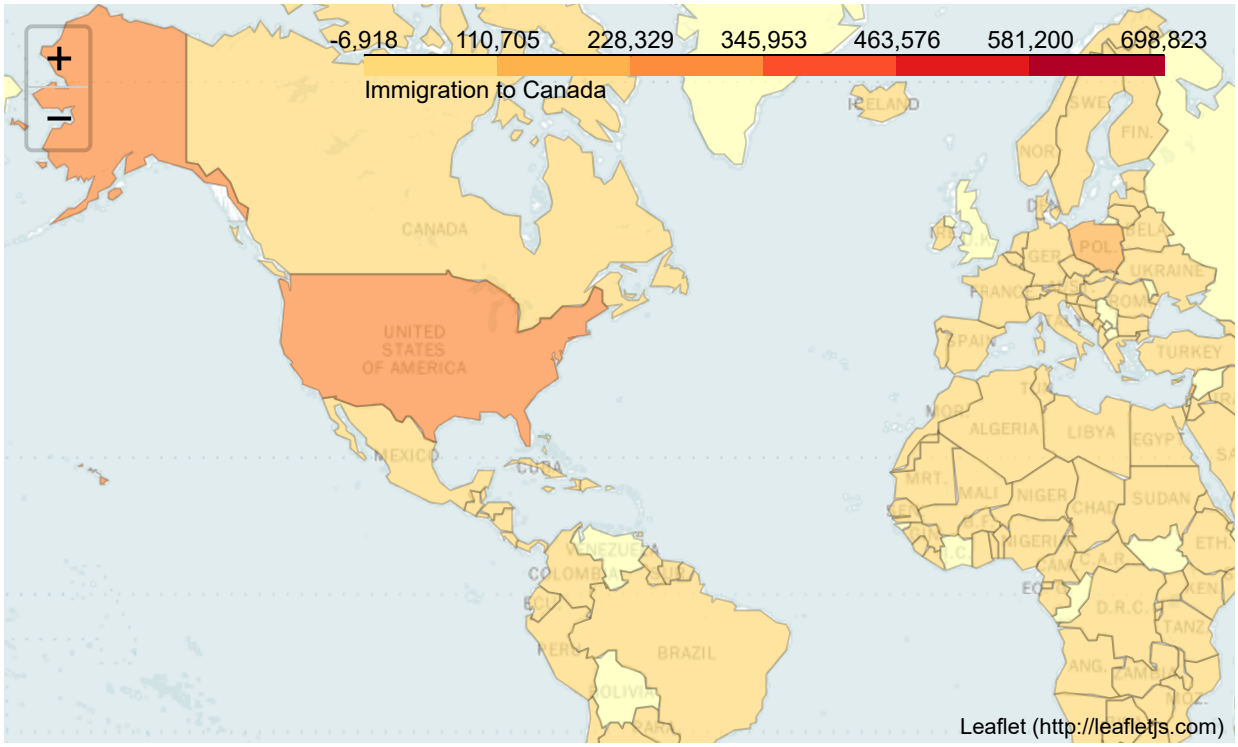
5 rows × 39 columns

In [15]:
```
# create list of years from 1980 to 2013
years = list(map(str, range(1980, 2014)))
```

In [16]:
```
# create a world map, centered around **[0, 0]** *latitude* and *longitude* values
world_map = folium.Map(location=[0, 0], zoom_start=2, tiles='Mapbox Bright')
```

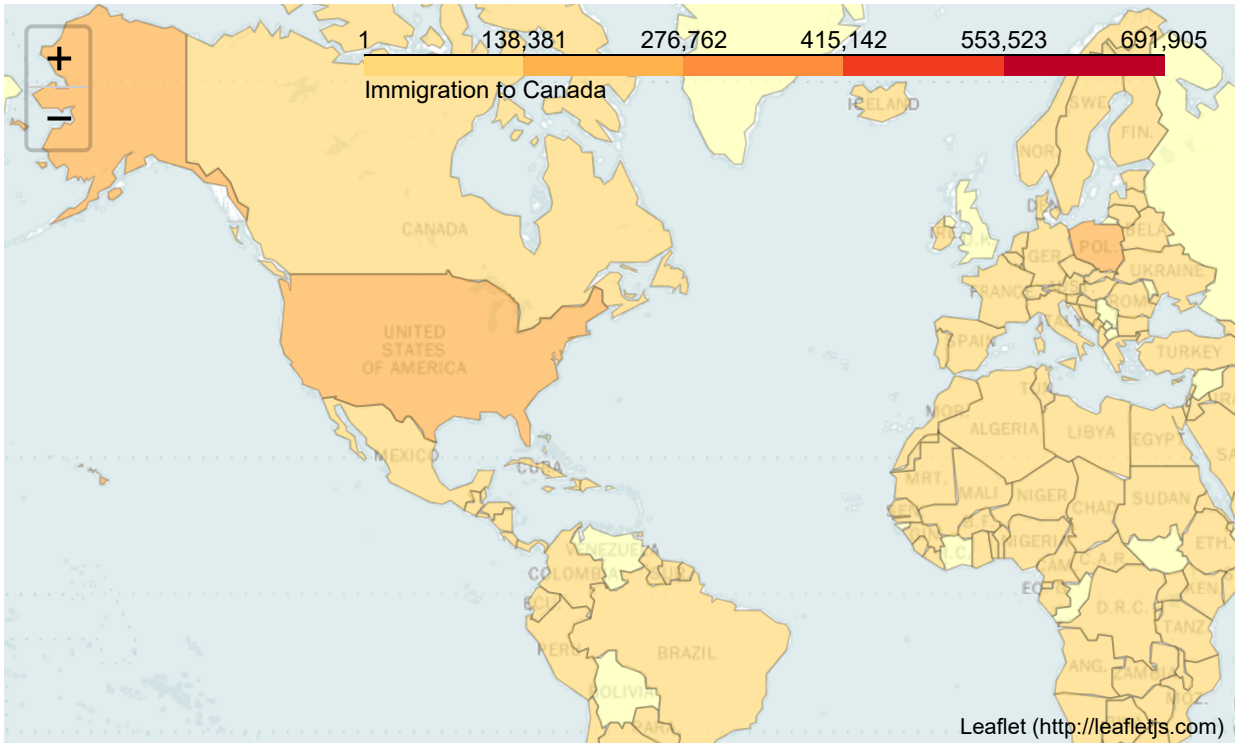
In [17]:
```
# get GeoJSON file
world_geo = r'world-countries.json' # geojson file

# generate choropleth map using the total immigration of each country to Canada fr
world_map.choropleth(
    geo_data=world_geo,
    data=df_can,
    columns=['Country', 'Total'],
    key_on='feature.properties.name',
    fill_color='YlOrRd',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Immigration to Canada'
)

# display map
world_map
```

Out[17]:



In [18]:
```
# Defining the thresholds and starting with 0 instead of -6,918!
world_geo = r'data/world-countries.json'

# create a numpy array of length 6 and has linear spacing from the minium total im
threshold_scale = np.linspace(df_can['Total'].min(),
                              df_can['Total'].max(),
                              6, dtype=int)
threshold_scale = threshold_scale.tolist() # change the numpy array to a list
threshold_scale[-1] = threshold_scale[-1] + 1 # make sure that the last value of t

# let Folium determine the scale.
world_map = folium.Map(location=[0, 0], zoom_start=2, tiles='Mapbox Bright')
world_map.choropleth(
    geo_data=world_geo,
    data=df_can,
    columns=['Country', 'Total'],
    key_on='feature.properties.name',
    threshold_scale=threshold_scale,
    fill_color='YlOrRd',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Immigration to Canada',
    reset=True
)
world_map
```

Out[18]:



Copyright © 2018 Cognitive Class (https://cognitiveclass.ai/?utm_source=bducopyrightlink&utm_medium=dswb&utm_campaign=bdu). This notebook and its source code are released under the terms of the MIT License (https://bigdatauniversity.com/mit-license/).

In [ ]: