

# DATA VISUALIZATION

## User-Defined Index

|  |    |
|--|----|
| PARTE 1. INTRODUCCIÓN A LA VISUALIZACIÓN DE DATOS..... | 3  |
| 1.1. INTRODUCCIÓN.....                                 | 3  |
| 1.2. INTRODUCCIÓN A MATPLOTLIB.....                    | 6  |
| 1.3. GRÁFICAS SENCILLAS CON MATPLOTLIB.....            | 11 |
| 1.4. DATASET DE LAS INMIGRACIONES A CANADA.....        | 16 |
| 1.5. GRÁFICOS DE LÍNEA.....                            | 19 |
| PARTE 2. HERRAMIENTAS BÁSICAS DE VISUALIZACIÓN.....    | 22 |
| 2.1. GRÁFICOS DE ÁREA.....                             | 22 |
| 2.2. HISTOGRAMAS.....                                  | 29 |
| 2.3. GRÁFICOS DE BARRAS.....                           | 32 |
| 2.4. PIE CHARTS.....                                   | 34 |
| 2.5. BOX PLOTS.....                                    | 37 |
| 2.6. GRÁFICOS DE DISPERSIÓN.....                       | 40 |
| PARTE 3. HERRAMIENTAS DE VISUALIZACIÓN AVANZADAS.....  | 43 |
| 3.1. WAFFLE CHARTS.....                                | 43 |
| 3.2. NUBES DE PALABRAS (word clouds).....              | 44 |
| 3.3. SEABORN Y GRÁFICOS DE REGRESIÓN.....              | 45 |
| 3.4. FOLIOS.....                                       | 47 |
| 3.5. MAPAS Y MARCADORES.....                           | 51 |
| 3.6. MAPA DE CLOROPLETAS.....                          | 53 |

## PARTE 1.INTRODUCCIÓN A LA VISUALIZACIÓN DE DATOS

### 1.1. INTRODUCCIÓN

Introduciremos la visualización de datos y veremos ejemplos de cómo transformar una imagen dada en una que sea más efectiva, atractiva e impactante.

Por qué aprender visualización de datos?

- Es una forma de mostrar datos complejos de una forma gráfica y fácil de entender.
  - Esto puede ser muy útil cuando se trata de explorar los datos y obtener conocimientos a partir de ellos.
- Una imagen vale lo que 1000 palabras, así los gráficos pueden ser muy efectivos para dar una clara descripción de los datos, por ejemplo frente a una audiencia.
- Pueden ser muy útiles a la hora de apoyar recomendaciones realizadas a clientes, managers o tomadores de decisiones.

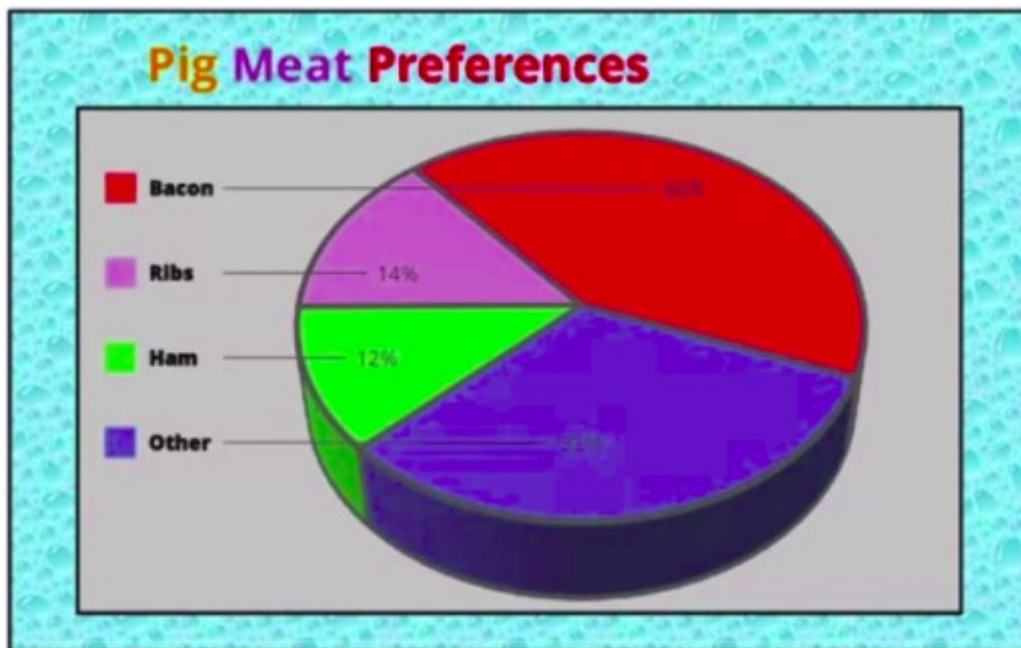
Darkhorse Analytics es una compañía que ha hecho un gran trabajo en el campo de visualización de datos. Su enfoque al crear visualizaciones se basa en 3 puntos:

- Menos es más efectivo.
- Menos es más atractivo.
- Menos es más impactante.

En otras palabras característica o diseño que incorpore a su gráfico para hacerlo más atractivo o agradable debe soportar el mensaje que se quiere dar y no distraernos de él.

Miremos un ejemplo.

En la figura siguiente se muestra un pie chart que muestra las preferencias de la gente en lo que se refiere a las diferentes carnes de cerdo.



**Figura.** Pie chart.

El mensaje del gráfico es que casi la mitad de la gente prefiere bacon respecto a los otros tipos de carne.

Ahora, hay muchas características en este gráfico que no sabemos si son relevantes, como el fondo azul o la orientación 3D. De hecho, esto nos distrae del mensaje principal y puede confundir a la audiencia.

Apliquemos ahora el enfoque de Dark Horse.

- El mensaje principal es que casi la mitad de la gente prefiere bacon, así que deshagamonos de todo lo que nos distraiga de esto.
- Lo primero es deshacernos de los colores de fondo.
- También deshagamonos de las fronteras, ya que no brindan información extra y de las leyendas redundantes puesto que ya se tiene un código de colores.
- El 3D no aporta nada, así que adiós a él.
- El texto en negrita es innecesario.
- Deshagamonos de los diferentes colores y las porciones.
- Eso si, espesemos las líneas para hacerlas más significativas.

Esto luce familiar. Es un diagrama de barras horizontales.

Finalmente, enfatizemos bacon para que resalte respecto a los otros tipos.

Ahora, juntemos los diagramas y veamos cuál es más fácil de entender. El diagrama de barras es más simple, claro, distrae menos y es más fácil de leer.



**Figura.** Comparación de los diagramas.

## 1.2. INTRODUCCIÓN A MATPLOTLIB

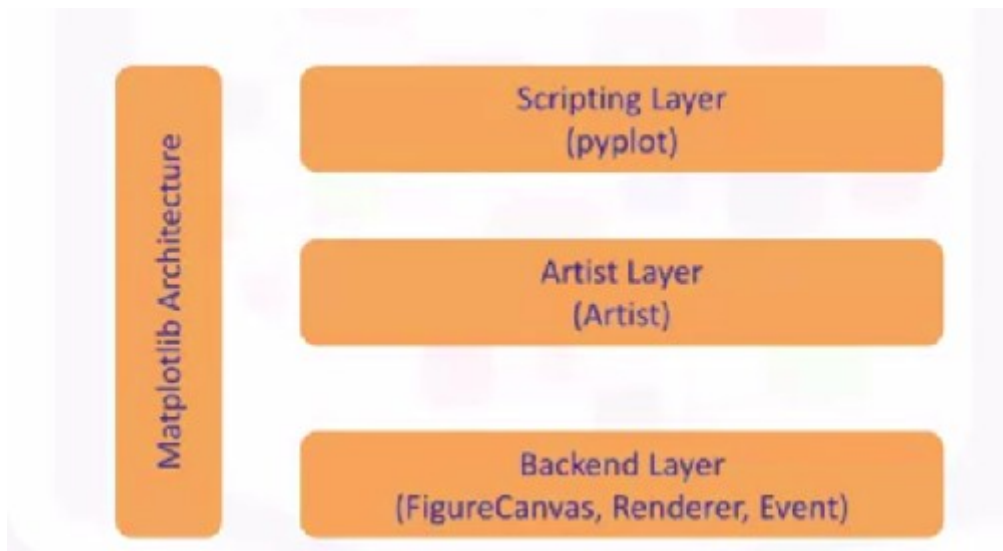
Matplotlib fue creada por John Hunter, que era neurobiólogo y formó parte de un equipo de investigación que estaba trabajando en el análisis de señales de electrocorticografía (eCoG). El equipo estaba utilizando un software propietario para el análisis y estaban tomando turnos para usarla. Para superar esta limitación John propuso reemplazar el SW propietario con una versión basada en MATLAB que podrían utilizar él y sus compañeros de equipo y que podría ser extendida por múltiples desarrolladores. Así nació Matplotlib.



**Figura.** Historia de matplotlib.

La arquitectura de matplotlib tiene 3 capas principales:

- La capa de backend.
- La capa de artista (artist)
  - Donde ocurre gran parte del trabajo pesado y suele ser el paradigma de programación apropiado al escribir un servidor de aplicaciones web, o una aplicación de interfaz de usuario o un script compartido con otros.
- La capa de scripting.
  - Es la adecuada para los fines cotidianos y se considera una interfaz de scripting más ligera para simplificar las tareas comunes y para una generación rápida y sencilla de gráficos.



**Figura.** Arquitectura de Matplotlib.

La **capa de backend** tiene 3 clases de interfaz abstractas incorporadas:

- FigureCanvas
  - Define y abarca el área en la que se dibuja la figura.
- Renderer
  - Una instancia de la clase renderizador que sabe cómo dibujar en el lienzo la figura.
- Event
  - Maneja las entradas del usuario, cómo los trazos del teclado y los clics del mouse.

La **capa de artista** se compone de un objeto principal: el artista. El artista es el objeto que sabe cómo tomar el Renderer y usarlo para poner tinta en el lienzo. Todo lo que se ve en una figura de Matplotlib es una instancia de artist. El título, las líneas, las etiquetas de marca, las imágenes, etc. corresponden a un artista individual.

Hay 2 tipos de objetos artist:

- Tipo primitivo
  - Por ejemplo una línea, un rectángulo, un círculo, un texto.
- Tipo compuesto
  - Por ejemplo la figura o los ejes.

El objeto Matplotlib de nivel superior que contiene y administra todos los elementos de un gráfico dado es el artista de la figura, y el artista compuesto más importante son los ejes

porque es dónde se definen la mayoría de los métodos de trazado de la API, incluidos los métodos para crear y manipular las líneas del eje, la cuadrícula o el fondo del trazado.

Cada artista compuesto puede contener otros artistas compuestos, así como artistas primitivos. Un artista de figuras, por ejemplo, contendría un artista de eje, así como un rectángulo o artistas de texto.

Veamos un ejemplo de generación de un gráfico.

Intentemos generar un histograma de 10.000 números aleatorios usando la capa de artista.

```
"""
```

1. Generación de un histograma de 10000 números aleatorios

```
"""
```

```
# import FigureCanvas
from matplotlib.backends.backend_agg import FigureCanvasAgg as FigureCanvas
# import Figura artist
from matplotlib.figure import Figure
fig = Figure()
canvas = FigureCanvas(fig)
# crear 10000 números aleatorios usando numpy
import numpy as np
x = np.random.randn(10000)
ax = fig.add_subplot(111) # crea un eje artista (axes artist)
ax.hist(x, 100) # genera un histograma de los 10000 números
# le agrega un título a la figura y la guarda
ax.set_title("Normal distribution with  $\mu=0$ ,  $\sigma=1$ ")
fig.save("matplotlib_histogram.png")
```

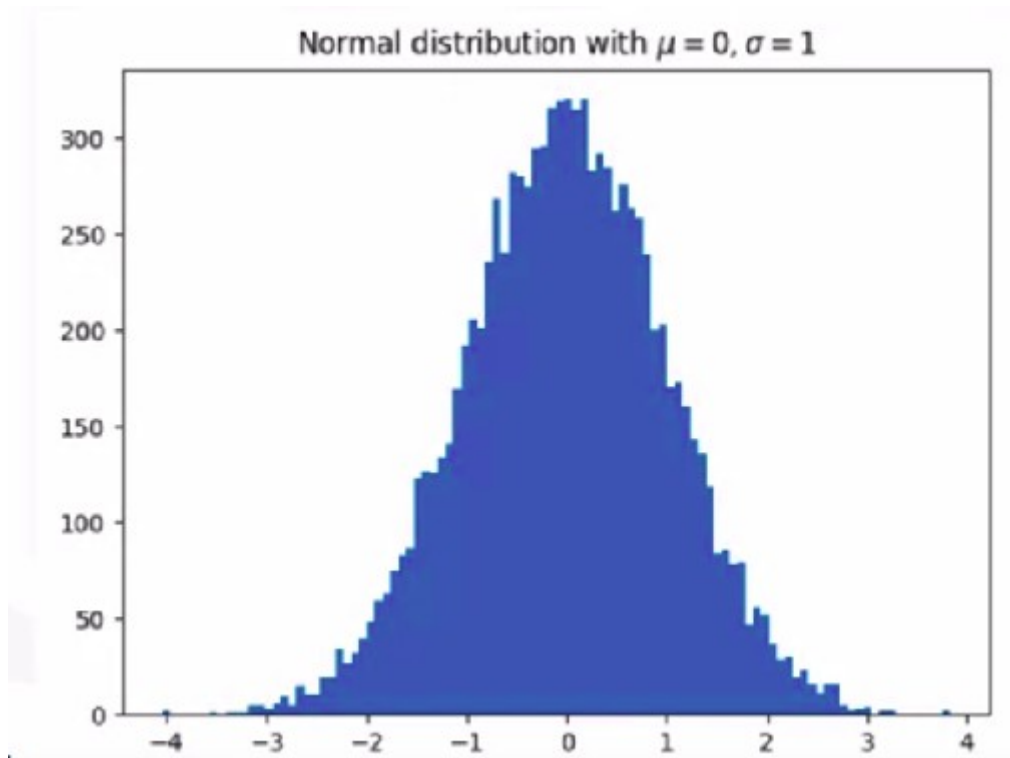
Algunas notas respecto al código anterior:

- El agg en backends.backend\_agg significa “geometría anti grano”, que es una biblioteca de alto rendimiento para producir imágenes atractivas.
- El artista de ejes se añade automáticamente al contenedor de ejes de figura, fig.axes.
- (111) es de la convención de MATLAB, por lo que crea una cuadrícula con una fila y una columna y utiliza la primera celda de esa cuadrícula para la ubicación de los nuevos ejes.



- Hist crea una secuencia de artistas rectángulo para cada barra del histograma y los agrega al contenedor de ejes. Aquí 100 significa 100 compartimientos (bins).

El histograma generado se muestra en la figura siguiente:



**Figura.** Histograma con Matplotlib.

La **capa de scripting** fue desarrollada para científicos que no son desarrolladores profesionales, y es esencialmente la interfaz Matplotlib.pyplot que automatiza el proceso de definir un lienzo, una instancia de artista de figura y conectarlos.

Veamos ahora cómo se vería el mismo código del ejemplo anterior.

```
import matplotlib.pyplot as plt
import numpy as np

x = np.random.randn(10000)
plt.hist(x,100)
plt.title(r"Normal distribution with $\mu=0$, $\sigma=1$")
```

```
plt.savefig("matplotlib_histogram.png")  
plt.show()
```

### 1.3. GRÁFICAS SENCILLAS CON MATPLOTLIB

Jupyter tiene un soporte especializado para Matplotlib, si tienes una notebook, todo lo que debes hacer es importar Matplotlib y ya estás listo para comenzar.

```
In [1]: import matplotlib as mpl

In [2]: print ("matplotlib version: ", mpl.__version__ )
('matplotlib version: ', '2.0.0')
```

**Figura.** Importando Matplotlib desde una Jupyter Notebook.

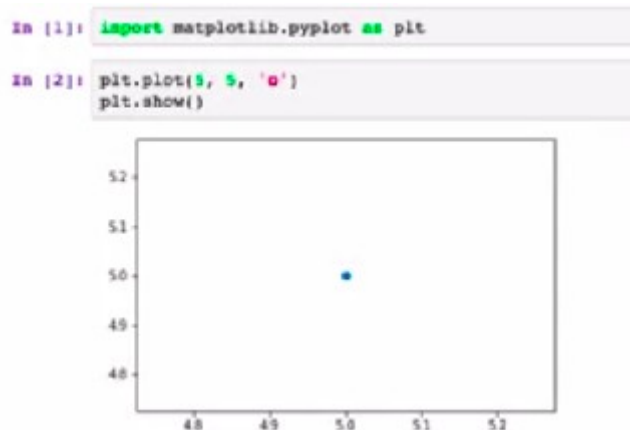
En este curso trabajaremos principalmente con la interfaz de scripting. Con esta interfaz puede crear casi todas las herramientas de visualización convencionales como:

- Histogramas
- Gráficos de barras.
- Gráficos de cajas.

con una sola función: plot.

Comencemos con un ejemplo.

Tracemos una marca circular en la posición (5,5).



**Figura.** Marca circular en la posición (5, 5)

Observe que se generó la gráfica dentro del navegador y no en una ventana separada.

Si la gráfica se genera en una nueva ventana, puede imponer la generación de parcelas dentro del navegador usando lo que se llama una función mágica. Una función mágica

comienza con % y para hacer que las gráficas se representen dentro del navegador se pasa en línea (inline) como el backend.

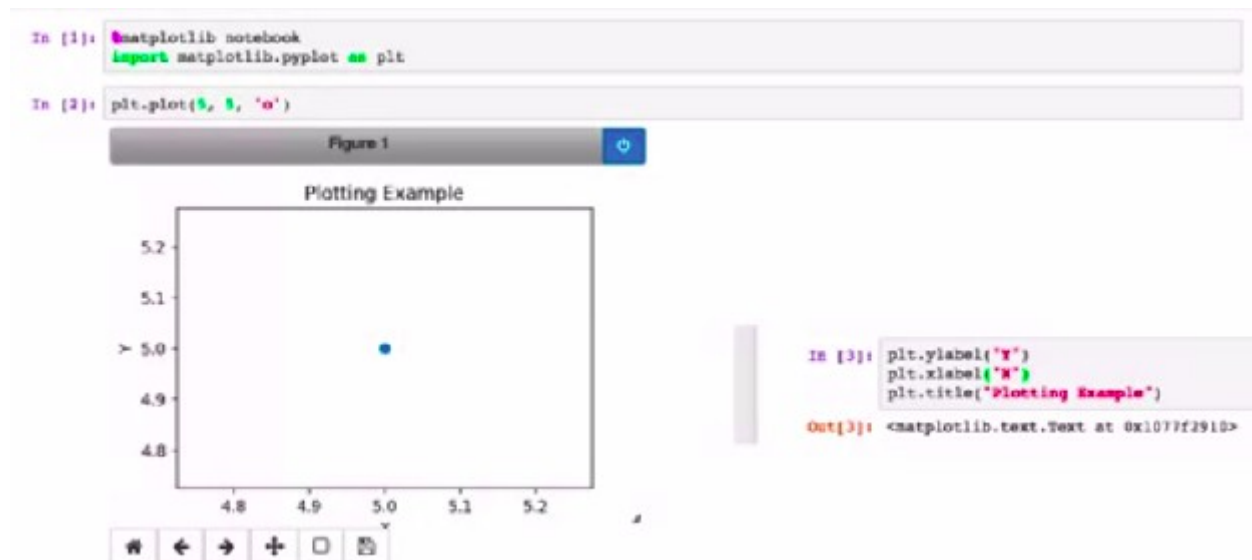


**Figura.** Utilización de una función mágica.

Matplotlib tiene una serie de backends disponibles. Una limitación de este backend es que no se puede modificar una figura una vez que se representa. Entonces, después de representar la figura anterior, no hay forma de que agreguemos, por ejemplo, un título de figura o etiquetemos sus ejes. Deberá generar un nuevo trazado y añadir un título y las etiquetas antes de llamar a la función show.

Un backend que supera esta limitación es el backend portátil; con él, si se llama a una función plt, se comprueba si existe una figura activa y cualquier función que llame se aplicará a esta figura activa. Si una figura no existe, representa una nueva.

Así que cuando llamamos a la función plt.plot para trazar una marca circular en la posición (5, 5) el backend comprueba si existe una figura activa. Como no la hay, genera una y añade una marca circular a la posición (5, 5). Y ahora podemos añadir fácilmente un título o etiquetas a los ejes después de renderizar la trama, sin la necesidad de regenerar la figura.



**Figura.** Backend portátil.

Pandas tiene una implementación incorporada de Matplotlib, por lo que es muy fácil trabajar con gráficas sobre un dataframe o una serie Pandas.

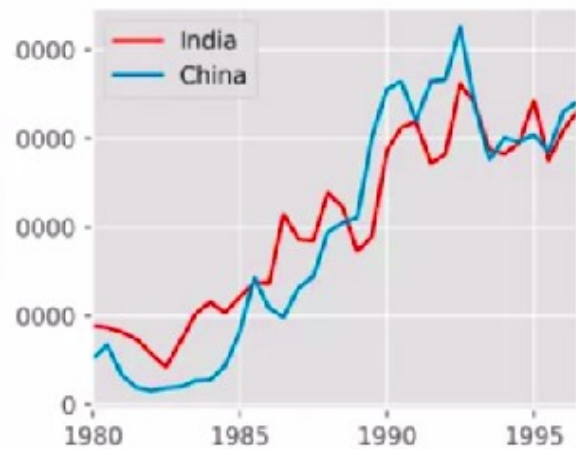
Digamos que tenemos un dataframe del número de inmigrantes de India y China a Canadá de 1980 a 1996 y queremos generar una gráfica de tipo “línea” de esos datos. Todo lo que debe hacerse es llamar a la función plot en ese dataframe que llamaremos “India\_China\_DF” y establecer el parámetro de tipo en “line”.

india\_china\_df

|      | India | China |
|------|-------|-------|
| 1980 | 8880  | 5123  |
| 1981 | 8670  | 6682  |
| 1982 | 8147  | 3308  |
| 1983 | 7338  | 1863  |
| 1984 | 5704  | 1527  |

```
india_china_df.plot(kind="line")
```

Figure 1



**Figura.** Gráfico de línea sobre un dataframe.

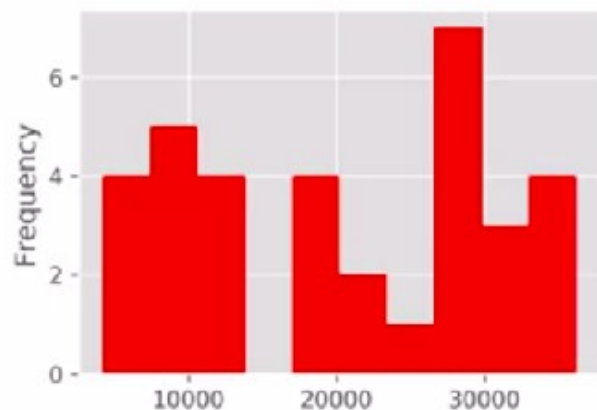
Traza un histograma es similar. Digamos que nos gustaría trazar un histograma de la columna India. Todo lo que tenemos que hacer es llamar a la función plot en esa columna y establecer el tipo de parámetro en "hist" y ahí lo tienes: el número de inmigrantes indios a Canadá entre 1980 y 1996.

india\_china\_df

|      | India | China |
|------|-------|-------|
| 1980 | 8880  | 5123  |
| 1981 | 8670  | 6682  |
| 1982 | 8147  | 3308  |
| 1983 | 7338  | 1863  |
| 1984 | 5704  | 1527  |

```
india_china_df["India"].plot(kind="hist")
```

Figure 1



**Figura.** Generación de un histograma a partir de un dataframe.

## 1.4. DATASET DE LAS INMIGRACIONES A CANADA

Vamos a aprender más sobre el conjunto de datos que utilizaremos en el curso.

La división de población de las Naciones Unidas recopiló datos sobre inmigración relativos a 45 países. Los datos consisten en el número total de inmigrantes procedentes de todo el mundo a cada uno de los 45 países, así como otros metadatos relativos a los países inmigrantes de origen. En este curso, nos centraremos en la inmigración a Canadá y trabajaremos principalmente con el conjunto de datos que involucra la inmigración al gran norte blanco.

La figura siguiente muestra una instantánea de los datos de la ONU sobre la inmigración a Canadá en forma de un archivo de Excel.

United Nations  
Population Division  
Department of Economic and Social Affairs

*International Migration Flows to and from Selected Countries: The 2015 Revision*

POP/DB/MIG/Flow/Rev.2015  
December 2015 - Copyright © 2015 by United Nations. All rights reserved  
Suggested citation: United Nations, Department of Economic and Social Affairs, Population Division (2015).  
International Migration Flows to and from Selected Countries: The 2015 Revision. (United Nations database).

Reporting country: Canada  
Criterion: Citizenship

| Classification |            | Origin/Destination | Major area | Region   | Development region | 1980            | 1981 | 1982               | 1983 | 1984 | 1985 |
|----------------|------------|--------------------|------------|----------|--------------------|-----------------|------|--------------------|------|------|------|
| Type           | Coverage   | OdName             | AREA       | AreaName | REG                | RegName         | DEV  | DevName            |      |      |      |
| Immigrants     | Foreigners | Afghanistan        | 935        | Asia     | 5501               | Southern Asia   | 902  | Developing regions | 16   | 39   | 39   |
| Immigrants     | Foreigners | Albania            | 908        | Europe   | 925                | Southern Europe | 901  | Developed regions  | 1    | 0    | 0    |
| Immigrants     | Foreigners | Algeria            | 903        | Africa   | 912                | Northern Africa | 902  | Developing regions | 80   | 67   | 71   |
| Immigrants     | Foreigners | American Samoa     | 909        | Oceania  | 957                | Polynesia       | 902  | Developing regions | 0    | 1    | 0    |
| Immigrants     | Foreigners | Andorra            | 008        | Europe   | 926                | Southern Europe | 901  | Developed regions  | 0    | 0    | 0    |

**Figura.** Datos de la ONU acerca de la inmigración a Canadá.

- Las primeras 20 filas contienen datos textuales sobre el departamento de las Naciones Unidas y otra información irrelevante.
- La fila 21 contiene las etiquetas de las columnas.
- Luego, cada fila representa un país y contiene metadatos sobre el país, como en qué continente reside, a qué región pertenece, y si la región está en desarrollo o ya es desarrollada. Cada fila contiene también el número total de inmigrantes precedentes de ese país para los años 1980 hasta 2013.

En este curso usaremos Pandas para el análisis de datos previo a la visualización.

El siguiente código importa las librerías necesarias (en particular xlrd para extraer datos desde Excel) e importa los datos.



```

import numpy as np
import pandas as pd
from __future__ import print_function # agrega compatibilidad con python 2

# instala xlrd en Jupyter
!pip install xlrd
print("xlrd instalado")

df_can =
pd.read_excel("https://ibm.box.com/shared/static/lw190pt9zpy5bd1ptyg2aw15awomz9pu.xlsx",
              sheetname="Canada by Citizenship",
              skiprows=range(20),
              skip_footer=2)

```

Observe que nos saltamos las primeras 20 filas para leer solamente los datos correspondientes de cada país.

Si desea confirmar que ha importado correctamente sus datos, puede usar la función head.

|   | Type       | Coverage   | OdName         | AREA | AreaName | REG  | RegName         | DEV | DevName            | 1980 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |
|---|------------|------------|----------------|------|----------|------|-----------------|-----|--------------------|------|-----|------|------|------|------|------|------|------|------|------|------|
| 0 | Immigrants | Foreigners | Afghanistan    | 935  | Asia     | 5501 | Southern Asia   | 902 | Developing regions | 16   | ... | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1 | Immigrants | Foreigners | Albania        | 908  | Europe   | 925  | Southern Europe | 901 | Developed regions  | 1    | ... | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2 | Immigrants | Foreigners | Algeria        | 903  | Africa   | 912  | Northern Africa | 902 | Developing regions | 80   | ... | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3 | Immigrants | Foreigners | American Samoa | 909  | Oceania  | 957  | Polynesia       | 902 | Developing regions | 0    | ... | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |
| 4 | Immigrants | Foreigners | Andorra        | 908  | Europe   | 925  | Southern Europe | 901 | Developed regions  | 0    | ... | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    | 1    |

**Figura.** df\_can.head()

Como puede ver, la salida de la función principal parece correcta, con las columnas que tienen las etiquetas correctas y cada fila representa un país y contiene el número total de inmigrantes de ese país.

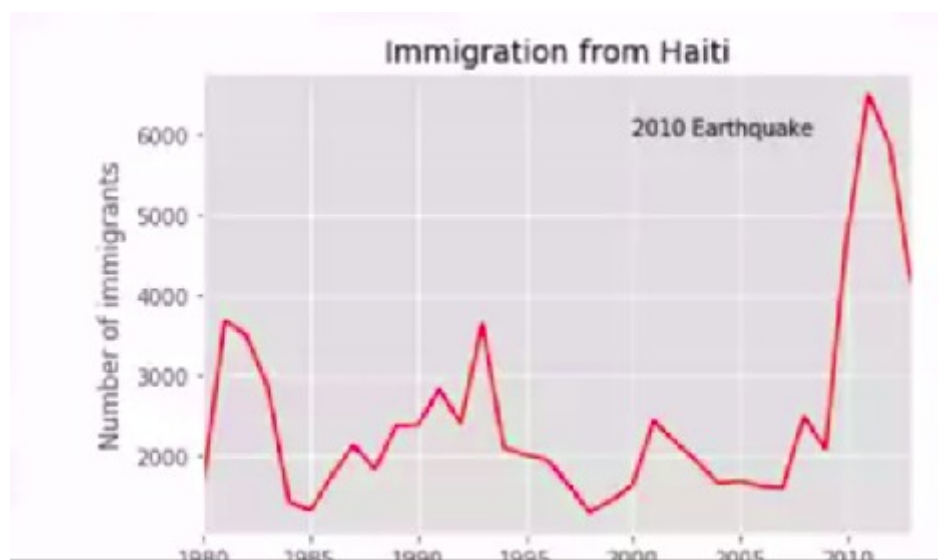
## 1.5. GRÁFICOS DE LÍNEA

Generaremos nuestra primera herramienta de visualización: la gráfica de líneas.

Una **gráfica de líneas** es una gráfica en forma de una serie de puntos de datos conectados por segmentos de línea recta.

La pregunta más importante es **cuándo usar gráficas de línea**. El mejor caso de uso es cuando se tiene un conjunto de datos continuo y está interesado en visualizar los datos durante un período de tiempo.

Como ejemplo digamos que estamos interesados en la tendencia de los inmigrantes de Haití a Canadá. La figura siguiente muestra una gráfica de líneas donde se observa la tendencia de las inmigraciones de Haití a Canadá entre 1980 y 2013.



**Figura.** Gráfica de línea inmigraciones desde Haití a Canadá.

Basándonos en esta gráfica, podemos investigar justificaciones de anomalías o cambios obvios. Vemos en el ejemplo que hay un pico de inmigración de Haití a Canadá en 2010. Una búsqueda en Google de los principales acontecimientos ocurridos en Haití en 2010 devolvería el trágico terremoto que tuvo lugar en 2010, y por tanto esta afluencia de inmigración a Canadá se debió principalmente a ello.

Antes de ver cómo generar gráficas de línea hagamos un repaso de nuestro conjunto de datos.

Cada fila representa un país y contiene metadatos sobre el país, cómo dónde se encuentra geográficamente y si está en desarrollo o ya es desarrollado. Cada fila contiene también cifras numéricas de la inmigración anual de ese país a Canadá entre 1980 y 2013.

Ahora vamos a procesar el dataframe:

- Hacemos que el nombre del país se convierta en el índice de cada fila.
  - Esto facilita la consulta de países específicos.
- Vamos a añadir una columna adicional que representa la suma acumulada de la inmigración anual de cada país de 1980 a 2013.
  - Por ejemplo, para Afganistán es de 58.639 en total y para Albania 15.699.
- Pongámosle el nombre df\_canada a nuestro dataframe.

|   | Type       | Coverage   | OdName         | AREA | AreaName | REG  | RegName         | DEV | DevName            | 1980 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |
|---|------------|------------|----------------|------|----------|------|-----------------|-----|--------------------|------|-----|------|------|------|------|------|------|------|------|------|------|
| 0 | Immigrants | Foreigners | Afghanistan    | 935  | Asia     | 5501 | Southern Asia   | 902 | Developing regions | 16   | ... | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1 | Immigrants | Foreigners | Albania        | 908  | Europe   | 925  | Southern Europe | 901 | Developed regions  | 1    | ... | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2 | Immigrants | Foreigners | Algeria        | 903  | Africa   | 912  | Northern Africa | 902 | Developing regions | 80   | ... | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3 | Immigrants | Foreigners | American Samoa | 909  | Oceania  | 957  | Polynesia       | 902 | Developing regions | 0    | ... | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |
| 4 | Immigrants | Foreigners | Andorra        | 908  | Europe   | 925  | Southern Europe | 901 | Developed regions  | 0    | ... | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    | 1    |

**Figura.** Dataset antes del procesamiento.

|                | Continent | Region          | DevName            | 1980 | 1981 | 1982 | 1983 | 1984 | 1985 | 1986 | ... | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | Total |
|----------------|-----------|-----------------|--------------------|------|------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|-------|
| Country        |           |                 |                    |      |      |      |      |      |      |      |     |      |      |      |      |      |      |      |      |      |       |
| Afghanistan    | Asia      | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340  | 496  | ... | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 | 58639 |
| Albania        | Europe    | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0    | 1    | ... | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  | 15699 |
| Algeria        | Africa    | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44   | 69   | ... | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 | 69439 |
| American Samoa | Oceania   | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0    | 0    | ... | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 6     |
| Andorra        | Europe    | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0    | 2    | ... | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    | 1    | 15    |

**Figura.** Dataset luego del procesamiento.

Ahora generemos el gráfico de línea correspondiente a la inmigración de Haití.

```
import matplotlib as mpl
```

```
import matplotlib.pyplot as plt
```

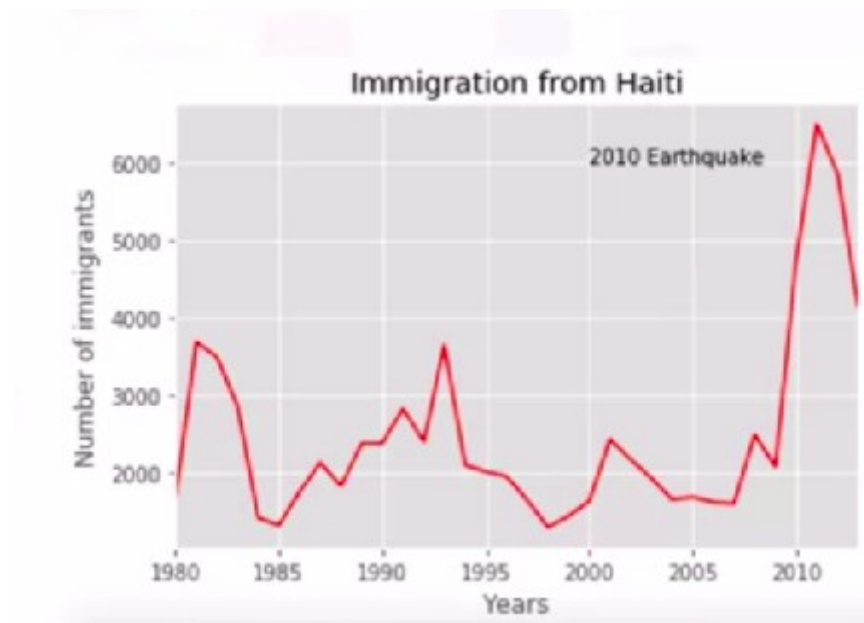
```
years = list(map(str, range(1980, 2014)))
```

```
df_canada.loc["Haití", years].plot(kind = "line")
plt.title("Immigration from Haiti")
plt.ylabel("Number of immigrants")
plt.xlabel("Years")

plt.show()
```

Notas respecto al código:

- Utilizamos years, que es una lista que contiene el formato de cadena de años de 1980 a 2013 para excluir la columna de inmigración total que hemos añadido.
- Se usó la función mágica %matplotlib con el backend inline.



**Figura.** Gráfica de línea que muestra la inmigración de Haití a Canadá desde 1980 a 2013.

## PARTE 2. HERRAMIENTAS BÁSICAS DE VISUALIZACIÓN

### 2.1. GRÁFICOS DE ÁREA

Estudiaremos la gráfica de área, que es una extensión de la gráfica de líneas.

Un **gráfico de área** es un tipo de gráfica que representa los totales acumulados utilizando números o porcentajes a lo largo del tiempo. Se basa en la gráfica de líneas y se utiliza comúnmente cuando se intenta comparar 2 o más cantidades.

Repasemos nuestro conjunto de datos.

Cada fila representa un país y contiene metadatos sobre el mismo. A su vez, cada fila contiene cifras numéricas de la inmigración a Canadá desde ese país entre 1980 y 2013.

|   | Type       | Coverage   | OdName         | AREA | AreaName | REG  | RegName         | DEV | DevName            | 1980 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |
|---|------------|------------|----------------|------|----------|------|-----------------|-----|--------------------|------|-----|------|------|------|------|------|------|------|------|------|------|
| 0 | Immigrants | Foreigners | Afghanistan    | 935  | Asia     | 5501 | Southern Asia   | 902 | Developing regions | 16   | ... | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1 | Immigrants | Foreigners | Albania        | 908  | Europe   | 925  | Southern Europe | 901 | Developed regions  | 1    | ... | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2 | Immigrants | Foreigners | Algeria        | 903  | Africa   | 912  | Northern Africa | 902 | Developing regions | 80   | ... | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3 | Immigrants | Foreigners | American Samoa | 909  | Oceania  | 957  | Polynesia       | 902 | Developing regions | 0    | ... | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |
| 4 | Immigrants | Foreigners | Andorra        | 908  | Europe   | 925  | Southern Europe | 901 | Developed regions  | 0    | ... | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    | 1    |

**Figura.** Nuestro conjunto de datos.

Ahora procesamos el dataframe para que el nombre del país se convierta en el índice de cada fila, lo que facilita la recuperación de filas correspondientes a países específicos.







|                | Continent | Region          | DevName            | 1980 | 1981 | 1982 | 1983 | 1984 | 1985 | 1986 | ... | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | Total |
|----------------|-----------|-----------------|--------------------|------|------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|-------|
| Country        |           |                 |                    |      |      |      |      |      |      |      |     |      |      |      |      |      |      |      |      |      |       |
| Afghanistan    | Asia      | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340  | 496  | ... | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 | 58639 |
| Albania        | Europe    | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0    | 1    | ... | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  | 15699 |
| Algeria        | Africa    | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44   | 69   | ... | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 | 69439 |
| American Samoa | Oceania   | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0    | 0    | ... | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 6     |
| Andorra        | Europe    | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0    | 2    | ... | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    | 1    | 15    |

**Figura.** Hacemos que el nombre del país se convierta en el índice de cada fila.

También añadiremos una columna adicional que representa la suma acumulada de la inmigración anual de cada país de 1980 a 2013.

|                | Continent | Region          | DevName            | 1980 | 1981 | 1982 | 1983 | 1984 | 1985 | 1986 | ... | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | Total |
|----------------|-----------|-----------------|--------------------|------|------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|-------|
| Country        |           |                 |                    |      |      |      |      |      |      |      |     |      |      |      |      |      |      |      |      |      |       |
| Afghanistan    | Asia      | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340  | 496  | ... | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 | 58639 |
| Albania        | Europe    | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0    | 1    | ... | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  | 15699 |
| Algeria        | Africa    | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44   | 69   | ... | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 | 69439 |
| American Samoa | Oceania   | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0    | 0    | ... | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 6     |
| Andorra        | Europe    | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0    | 2    | ... | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    | 1    | 15    |

**Figura.** Agregamos una columna que representa la suma total de inmigraciones.

También renombramos el dataframe como “df\_canada”.

Generemos parcelas de área para los países con mayor número de inmigración a Canadá.

Podemos encontrar estos países ordenando de forma descendente nuestro dataframe respecto a la suma acumulada. Para ellos utilizamos la función `sort_values`.

```
df_canada.sort_values(['Total'], ascending = False, axis = 0, inplace = True)
```

|  | Continent | Region             | DevName            | 1980  | 1981  | 1982  | 1983  | 1984  | 1985 | 1986 | ... | 2005  | 2006  | 2007  | 2008  | 2009  | 2010  | 2011  | 2012  | 2013  | Total  |
|--|-----------|--------------------|--------------------|-------|-------|-------|-------|-------|------|------|-----|-------|-------|-------|-------|-------|-------|-------|-------|-------|--------|
| Country  |           |                    |                    |       |       |       |       |       |      |      |     |       |       |       |       |       |       |       |       |       |        |
| India  | Asia      | Southern Asia      | Developing regions | 8880  | 8670  | 8147  | 7338  | 5704  | 4211 | 7150 | ... | 36210 | 33848 | 28742 | 28261 | 29456 | 34235 | 27509 | 30933 | 33067 | 691904 |
| China  | Asia      | Eastern Asia       | Developing regions | 5123  | 6682  | 3308  | 1863  | 1527  | 1816 | 1980 | ... | 42584 | 33518 | 27642 | 30037 | 29622 | 30391 | 28502 | 33024 | 34129 | 659962 |
| United Kingdom of Great Britain and Northern Ireland | Europe    | Northern Europe    | Developed regions  | 22045 | 24796 | 20620 | 10015 | 10170 | 9564 | 9479 | ... | 7258  | 7140  | 8216  | 8979  | 8876  | 8724  | 6204  | 6195  | 5827  | 551500 |
| Philippines  | Asia      | South-Eastern Asia | Developing regions | 6051  | 5921  | 5249  | 4562  | 3801  | 3150 | 4166 | ... | 18139 | 16400 | 19637 | 24887 | 28573 | 38617 | 36755 | 34315 | 29544 | 511391 |
| Pakistan   | Asia      | Southern Asia      | Developing regions | 978   | 972   | 1201  | 900   | 668   | 514  | 691  | ... | 14314 | 13127 | 10124 | 8994  | 7217  | 6811  | 7468  | 11227 | 12603 | 241600 |

**Figura.** Ordenando el dataframe.

Así, India, China, Reino Unido, Filipinas y Pakistán son los países con el mayor número de inmigraciones a Canadá.

Entonces, ¿ Podemos seguir adelante y generar las gráficas de área usando las primeras 5 filas de este dataframe?

Todavía no.

Primero tenemos que crear un nuevo dataframe con sólo estos 5 países y excluir la columna total, y lo que es más importante, necesitamos que los años se traen en el eje horizontal y la inmigración anual se organice en el eje vertical.

Matplotlib traza los índices de un dataframe en el eje horizontal, y, con el dataframe como se muestra, los países se trazarán allí. Para solucionar esto, transponemos el dataframe.

```
years = list(map(str, range(1980, 2014)))
```

```
df_canada.sort_values(["Total"], ascending=False, axis=0, inplace=True)
```

```
df_top5 = df_canada.head()
```

```
df_top5 = df_top5[years].transpose()
```

| Country | India | China | United Kingdom of Great Britain and Northern Ireland | Philippines | Pakistan |
|---------|-------|-------|--|-------------|----------|
| 1980    | 8880  | 5123  | 22045  | 6051        | 978      |
| 1981    | 8670  | 6682  | 24796  | 5921        | 972      |
| 1982    | 8147  | 3308  | 20620  | 5249        | 1201     |
| 1983    | 7338  | 1863  | 10015  | 4562        | 900      |
| 1984    | 5704  | 1527  | 10170  | 3801        | 668      |

**Figura.** Arreglando el dataframe para la gráfica de área.

El dataframe ahora es tal cual lo queremos, con 5 columnas donde cada una representa los países principales, y los años son los índices.

Ahora podemos realizar el gráfico.

```
import matplotlib as mpl
import matplotlib.pyplot as plt

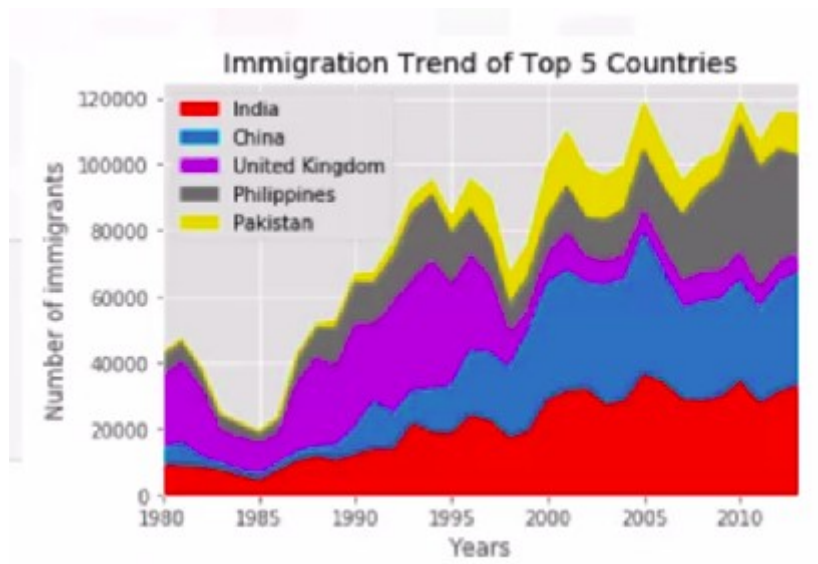
df_top5.plot(kind="area")

plt.title("Immigration trend of top 5 countries")
plt.ylabel("Number of immigrants")
plt.xlabel("Years")

plt.show()
```

Tenga en cuenta que aquí estamos generando la gráfica de área usando el backend en línea.

Y ahí lo tienes: una trama de área que muestra la tendencia migratoria de los cinco países con mayor inmigración a Canadá entre 1980 y 2013.



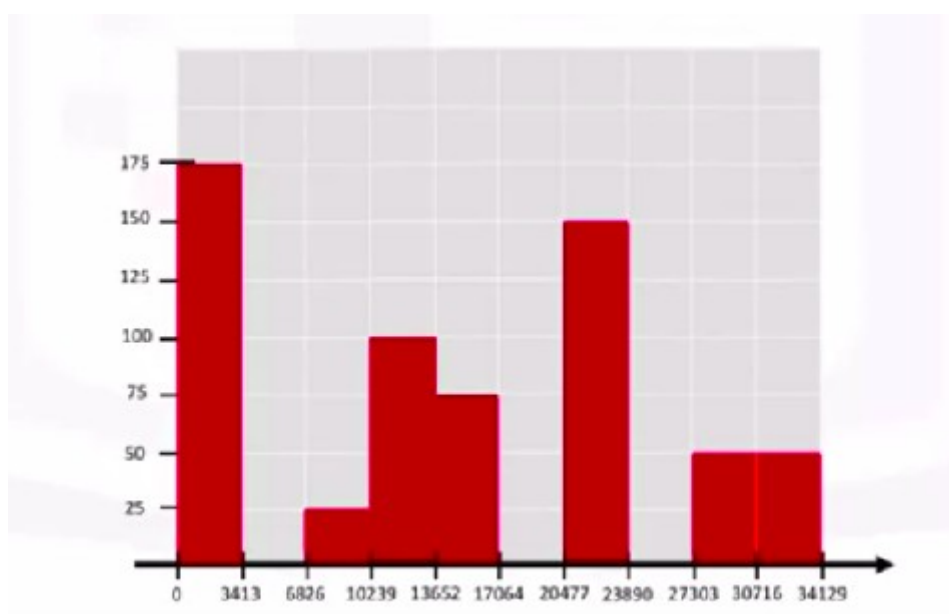
**Figura.** Gráfico de área.

## 2.2. HISTOGRAMAS

Un **histograma** es una forma de representar la distribución de frecuencia de un dataset numérico.

La forma en que funciona es particionar la propagación de los datos numéricos en bins (contenedores), asignar cada punto de los datos del dataset a un bin y a continuación contar el número de puntos de datos asignado a cada bin. Así, el número vertical es en realidad la frecuencia o el número de puntos de datos en cada bin.

Por ejemplo, supongamos que el rango de los valores numéricos en el conjunto de datos es 34.129. El primer paso para crear el histograma es particionar el eje horizontal en, digamos, 10 bins de igual ancho y luego contamos cuántos puntos de datos tienen un valor que está entre los límites de la primera bin, segunda bin, etc. Supongamos que el número de puntos de datos que tienen un valor entre 0 y 3413 (en el ejemplo) es 175; dibujamos entonces una barra de esa altura para ese contenedor. Repetimos lo mismo para los demás contenedores. Si no hay puntos de datos en un contenedor, el mismo tendría altura 0.



**Figura.** Ejemplo de histograma.

Trabajaremos con los datos ya preprocesados (países como índices, agregado de columna con la suma total y renombrado a `df_canada`).

Digamos que estamos interesados en visualizar la distribución de inmigrantes a Canadá en el año 2013.

La forma más sencilla de hacerlo es generar un histograma de los datos en la columna 2013.

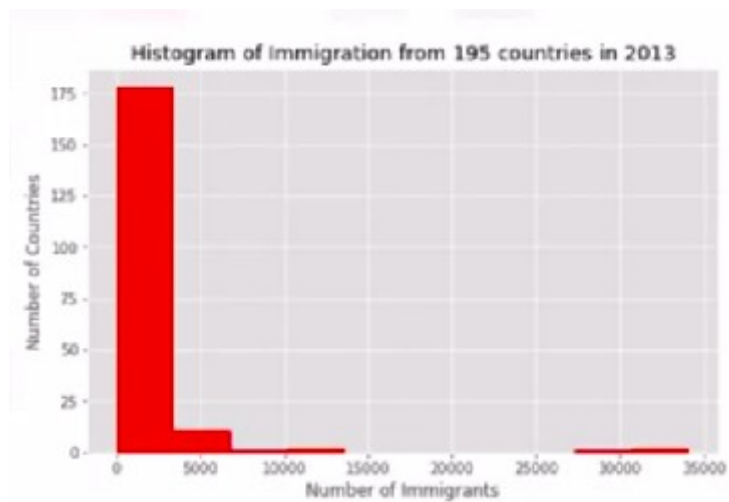
Hagámoslo con Matplotlib.

```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

```
df_canada["2013"].plot(kind="hist")
```

```
plt.title("Histogram of Immigration from 195 countries in 2013")
plt.ylabel("Number of countries")
plt.xlabel("Number of immigrants")
```

```
plt.show()
```



**Figura.** Histograma generado con Matplotlib.

Puede verse que los contenedores no están alineados con las marcas de graduación en el eje horizontal. Esto puede dificultar la lectura del histograma. Una forma de resolver esto es utilizar NumPy:

```
Import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
```

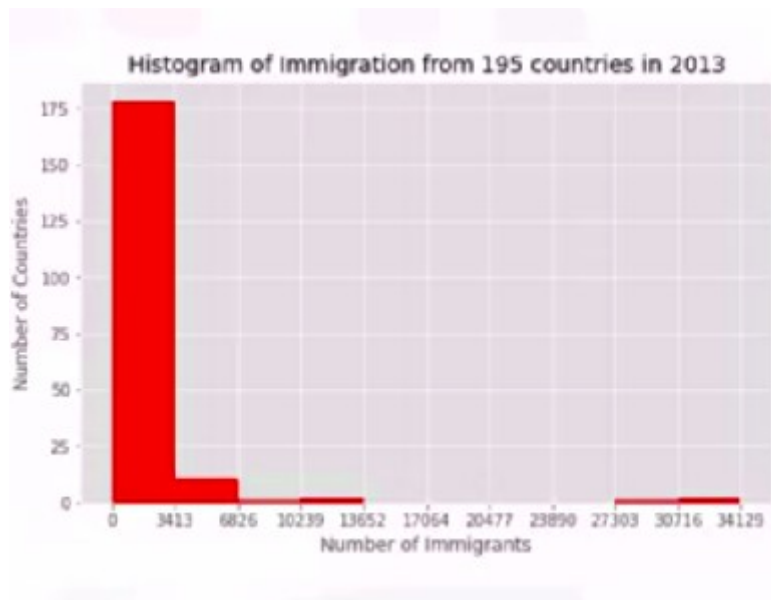
```
count, bin_edges = np.histogram(df_canada["2013"])
```

```
df_canada["2013"].plot(kind="hist", xticks=bin_edges)

plt.title("Histogram of Immigration from 195 countries in 2013")
plt.ylabel("Number of countries")
plt.xlabel("Number of immigrants")

plt.show()
```

En el ejemplo, la función histogram de numpy particiona el spread de los datos en la columna 2013 en 10 bins de igual ancho, calcula el número de puntos de datos que caen en cada bin y devuelve la frecuencia de cada bin (se la asignamos a "count") y los bordes de bin que asignamos a la variable "bin\_edges". Luego, pasamos los bordes de bin como un parámetro adicional para la función plot.



**Figura.** Histograma más prolijo utilizando la ayuda de NumPy.

## 2.3. GRÁFICOS DE BARRAS

Un **gráfico de barras** es un tipo de gráfico donde la longitud de cada barra es proporcional al valor del ítem que representa.

Es comúnmente utilizado para comparar valores de una variable en un punto dado del tiempo.

Por ejemplo, suponga que estamos interesados en visualizar de una forma discreta cómo se vio la inmigración de Islandia a Canadá desde 1980 a 2013. Una forma de hacerlo es construyendo un gráfico de barras donde la altura representa la inmigración total de Islandia a Canadá en un año en particular.

Trabajaremos con el dataframe `df_canada` procesado como en los casos anteriores.

Creamos el gráfico de barras:

```
import matplotlib as mpl
import matplotlib.pyplot as plt

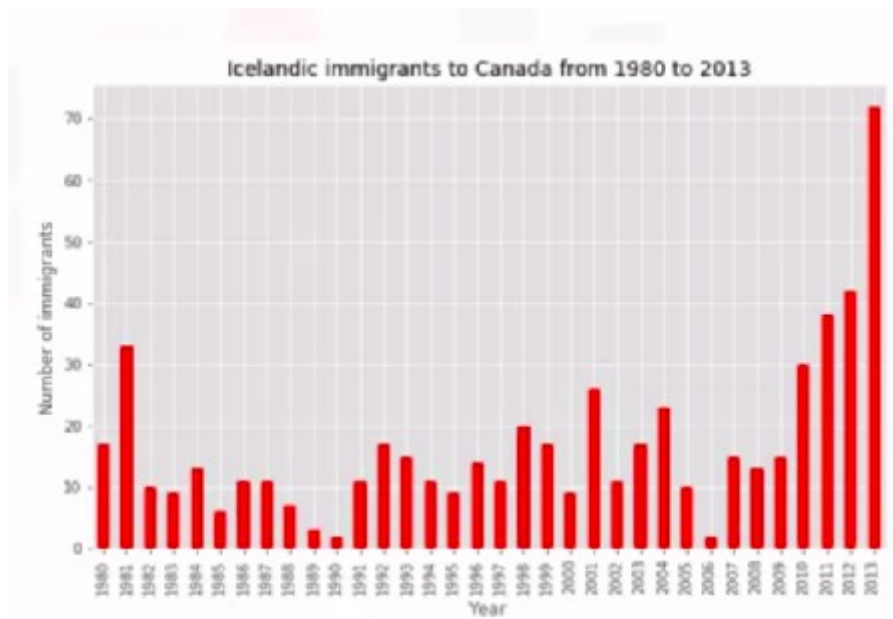
years = list(map(str, range(1980, 2014)))
df_iceland = df_canada.loc["Iceland", years] # datos pertenecientes a la inmigración de
Islandia a Canadá excluyendo la columna total (por eso se utiliza years)

df_iceland.plot(kind="bar") # kind=bar para gráfico de barras

plt.title("Iceland immigrants to Canada from 1980 to 2013")
plt.xlabel("Year")
plt.ylabel("Number of inmigrants")

plt.show()
```





**Figura.** Gráfico de barras.

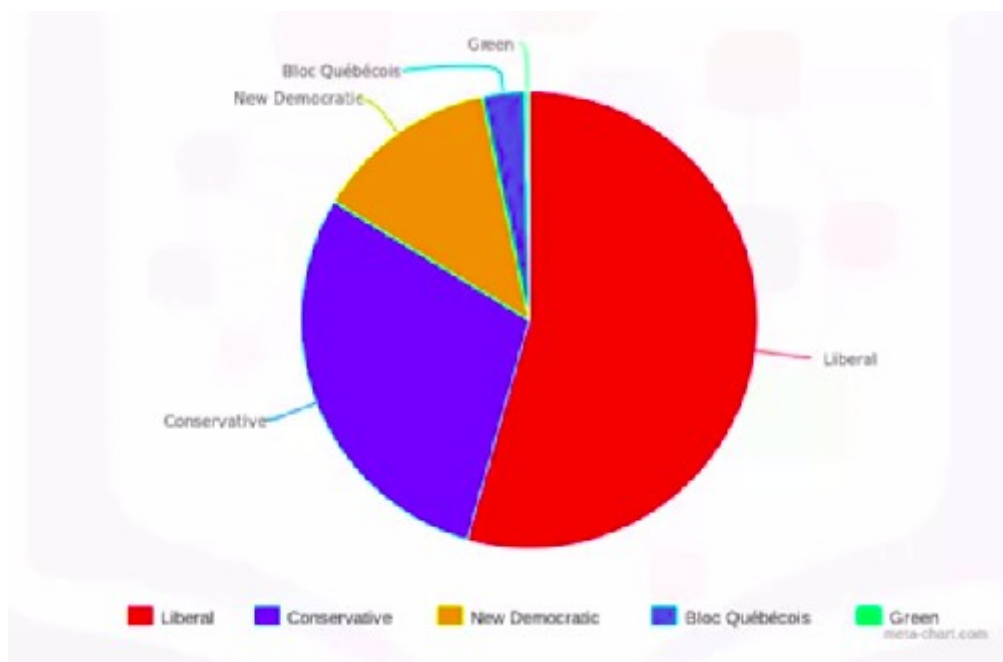
Examinando el gráfico, puede verse que la inmigración tiene una tendencia a aumentar desde el 2010.

## 2.4. PIE CHARTS

Estudiaremos los pie charts o gráficos circulares.

Un gráfico circular es un gráfico estadístico circular dividido en sectores para ilustrar la proporción numérica.

El siguiente es un gráfico circular de las elecciones federales canadienses en 2015 donde los liberales en rojo ganaron.



**Figura.** Ejemplo de pie chart.

Trabajaremos con dataframe procesado como antes `df_canada`.

Digamos que estamos interesados en visualizar un desglose de la inmigración al continente canadiense.

El primer paso es agrupar nuestros datos por continente utilizando la columna “continental”. Llamamos a la función `groupby` en `df_canada` y sumamos el número de inmigrantes de los países que pertenecen al mismo continente. El dataframe resultante (al que llamamos `df_continents`) es el siguiente:

```
df_continents = df_canada.groupby('Continent', axis = 0).sum()
```

|                                 | 1980  | 1981  | 1982  | 1983  | 1984  | 1985  | 1986  | 1987  | 1988  | 1989  | ... | 2005   | 2006   | 2007   | 2008   | 2009   | 2010   | 2011   | 2012   | 2013   | Total   |
|---------------------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-----|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------|
| Continent                       |       |       |       |       |       |       |       |       |       |       |     |        |        |        |        |        |        |        |        |        |         |
| Africa                          | 3951  | 4363  | 3819  | 2671  | 2639  | 2650  | 3782  | 7494  | 7552  | 9894  | ... | 27523  | 29188  | 28284  | 29890  | 34534  | 40892  | 35441  | 38083  | 38543  | 618948  |
| Asia                            | 31025 | 34314 | 30214 | 24896 | 27274 | 23850 | 28739 | 43203 | 47454 | 60256 | ... | 159253 | 149054 | 133459 | 139894 | 141434 | 163845 | 146894 | 152218 | 155075 | 3317794 |
| Europe                          | 39760 | 44802 | 42720 | 24838 | 22287 | 20844 | 24370 | 46698 | 54726 | 60893 | ... | 35955  | 33053  | 33495  | 34892  | 35078  | 33425  | 26778  | 29177  | 28691  | 1410947 |
| Latin America and the Caribbean | 13081 | 15215 | 16769 | 15427 | 13678 | 15171 | 21179 | 28471 | 21924 | 25060 | ... | 24747  | 24676  | 26011  | 26547  | 26867  | 28818  | 27856  | 27173  | 24950  | 765148  |
| Northern America                | 9378  | 10030 | 9074  | 7100  | 6661  | 6543  | 7074  | 7705  | 6469  | 6790  | ... | 8394   | 9613   | 9463   | 10190  | 8995   | 8142   | 7677   | 7892   | 8503   | 241142  |
| Oceania                         | 1942  | 1839  | 1675  | 1018  | 878   | 920   | 904   | 1200  | 1181  | 1539  | ... | 1585   | 1473   | 1693   | 1834   | 1860   | 1834   | 1548   | 1679   | 1775   | 55174   |

**Figura.** Agrupamos los datos por continente.

El dataframe resultante tiene 6 filas, cada una representando un continente y 35 columnas que representan los años de 1980 a 2013 más la suma acumulada de inmigración para cada continente.

Ahora podemos crear nuestro gráfico circular.

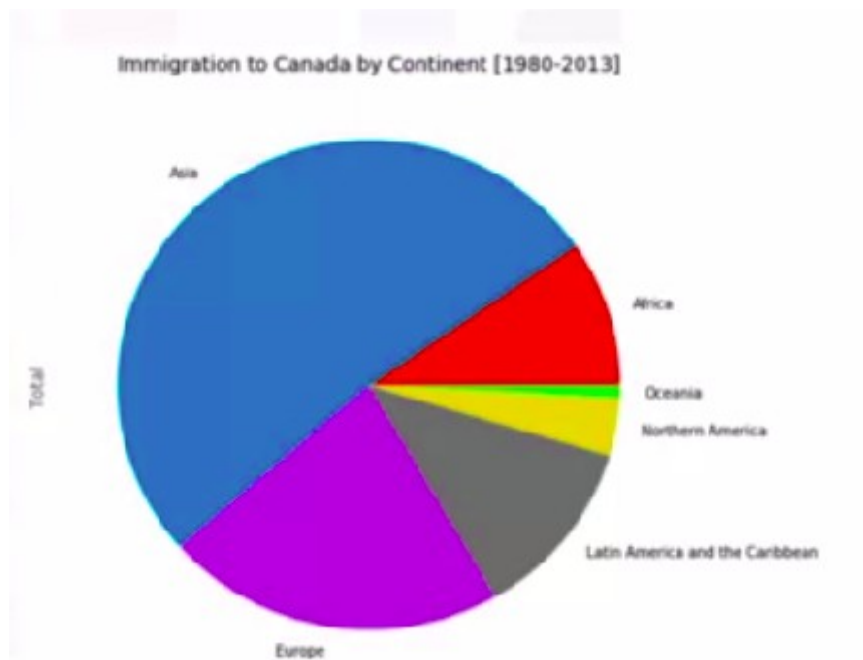
```
import matplotlib as mpl
```

```
import matplotlib.pyplot as plt
```

```
df_continents["Total"].plot(kind="pie")
```

```
plt.title("Immigration to Canada by Continent [1980-2013]")
```

```
plt.show()
```



**Figura.** Gráfico circular que muestra la proporción de inmigración de cada continente a Canadá de 1980 a 2013.

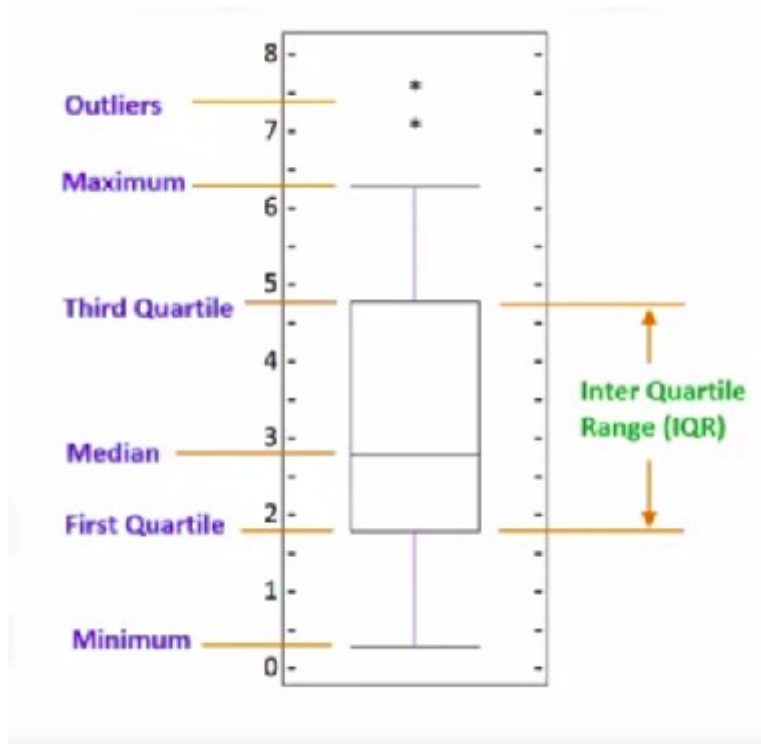
Un comentario sobre los gráficos circulares. Hay quienes se oponen a utilizarlos bajo cualquier circunstancia.

## 2.5. BOX PLOTS

Un **box plot (diagrama de caja)** es una forma de representar estadísticamente la distribución de datos a través de 5 dimensiones principales:

- Mínimo (Min)
  - Número más pequeño de los datos ordenados.
  - Su valor se puede obtener restando 1.5 veces el IQR donde IQR es el rango intercuartílico del primer cuartil.
- Primer cuartil
  - Es el 25% del camino a través de los datos ordenados.
  - En otras palabras,  $\frac{1}{4}$  de los puntos de datos son menores que este valor.
- Mediana
  - Es la mediana de los datos ordenados.
- Tercer cuartil
  - Es el 75% del camino a través de los datos ordenados.
  - En otras palabras,  $\frac{3}{4}$  de los puntos de datos son menores que este valor.
- Máximo (Max)
  - Número más alto en los datos ordenados.

Las gráficas de caja también muestran valores atípicos como puntos individuales que se producen fuera de los extremos superior e inferior.



**Figura.** Box plot

Trabajaremos con el dataframe `df_canada` procesado como en los ejemplos anteriores. Digamos que estamos interesados en crear una gráfica de caja para visualizar la inmigración de Japón a Canadá.

```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

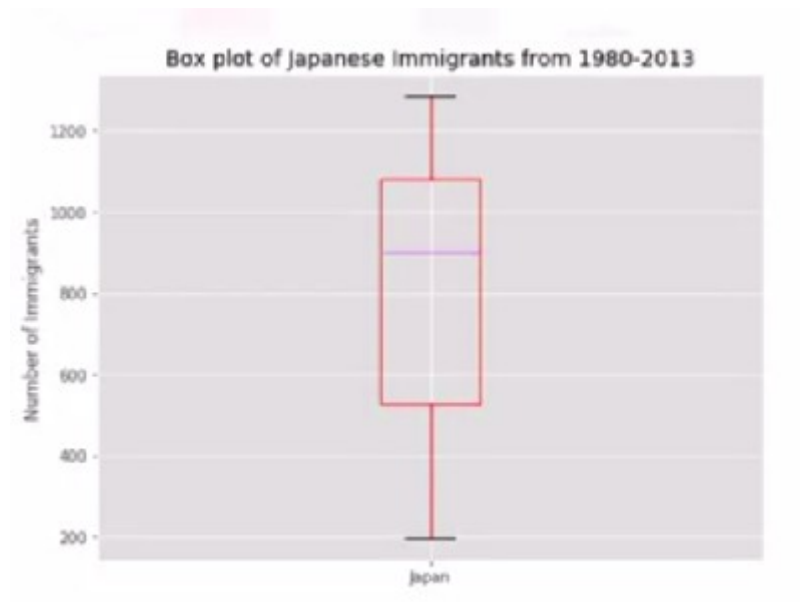
```
df_japon = df_canada.loc[["Japan", years].transpose()]
```

```
df_japon.plot(kind="box")
```

```
plt.title("Box plots of Japanese Immigrants from 1980-2013")
```

```
plt.ylabel("Number of inmigrants")
```

```
plt.show()
```

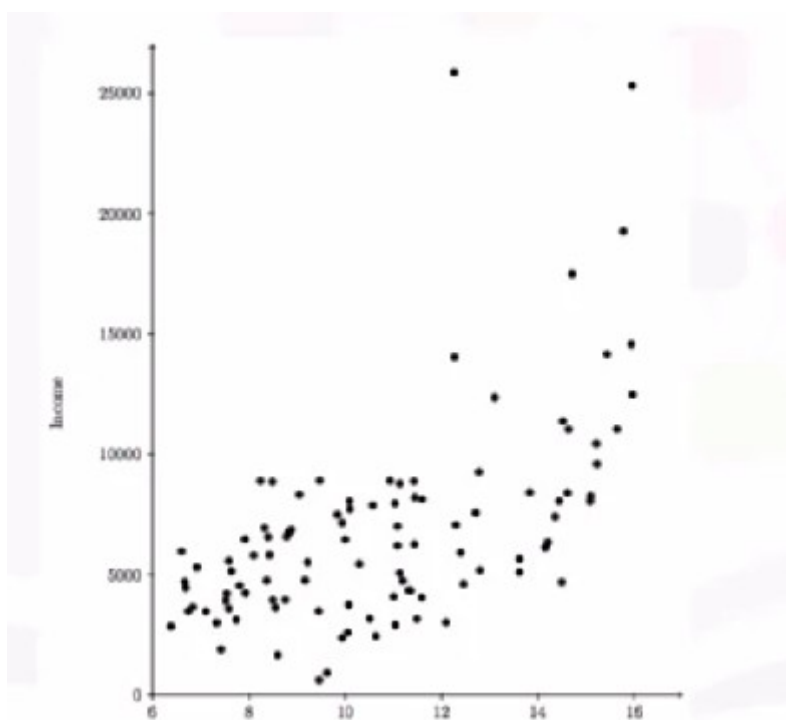


**Figura.** Diagrama de caja.

## 2.6. GRÁFICOS DE DISPERSIÓN

Un **gráfico de dispersión** es un tipo de gráfico que muestra valores pertenecientes típicamente a 2 variables una contra la otra. Por lo general, es una variable dependiente para ser trazada contra una variable independiente con el fin de determinar si existe alguna correlación entre las mismas.

En la figura siguiente se muestra una gráfica de dispersión de ingresos frente a educación. Mirando los datos se puede concluir que un individuo con más años de educación es probable gane un ingreso más alto que uno con menos.



**Figura.** Ejemplo de gráfico de dispersión.

Trabajaremos con el dataframe `df_canada` procesado como en los ejemplos anteriores.

Digamos que estamos interesados en trazar una gráfica de dispersión de la inmigración total anual a Canadá de 1980 a 2013.

Primero debemos crear un nuevo dataframe que muestre cada año y el número total de inmigrantes procedentes de todos los países del mundo como se muestra en la figura siguiente.



| df_total |        |
|----------|--------|
| year     | total  |
| 1980     | 99137  |
| 1981     | 110563 |
| 1982     | 104271 |
| 1983     | 75550  |
| 1984     | 73417  |
| .        | .      |
| .        | .      |

**Figura.** Nuevo dataframe para el gráfico de dispersión.

Al nuevo dataframe lo llamaremos df\_total.

```
Import matplotlib as mpl
import matplotlib.pyplot as plt
```

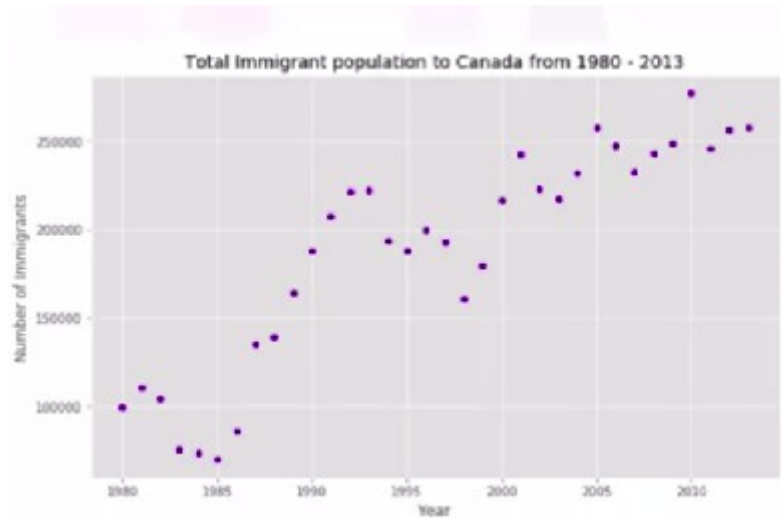
```
df_total.plot(kind="scatter",
              x="year",
              y="total")
```

```
plt.title("Total Immigration population to Canada from 1980 – 2013")
plt.xlabel("Year")
plt.ylabel("Number of Immigrants")
```

```
plt.show()
```

Ahora, a diferencia de con las otras herramientas de visualización de datos donde solo pasar el parámetro kind era suficiente para generar la gráfica, con gráficas de dispersión también necesitamos pasar la variable a ser trazada en el eje horizontal como el parámetro x, y la

variable a ser trazada en el eje vertical como el parámetro y. En este caso, pasamos el año de la columna como x y el total como y.



**Figura.** Gráfico de dispersión que muestra la inmigración desde países de todo el mundo desde 1980 a 2013.

El gráfico de dispersión muestra claramente una tendencia general al alza de la inmigración con el tiempo.

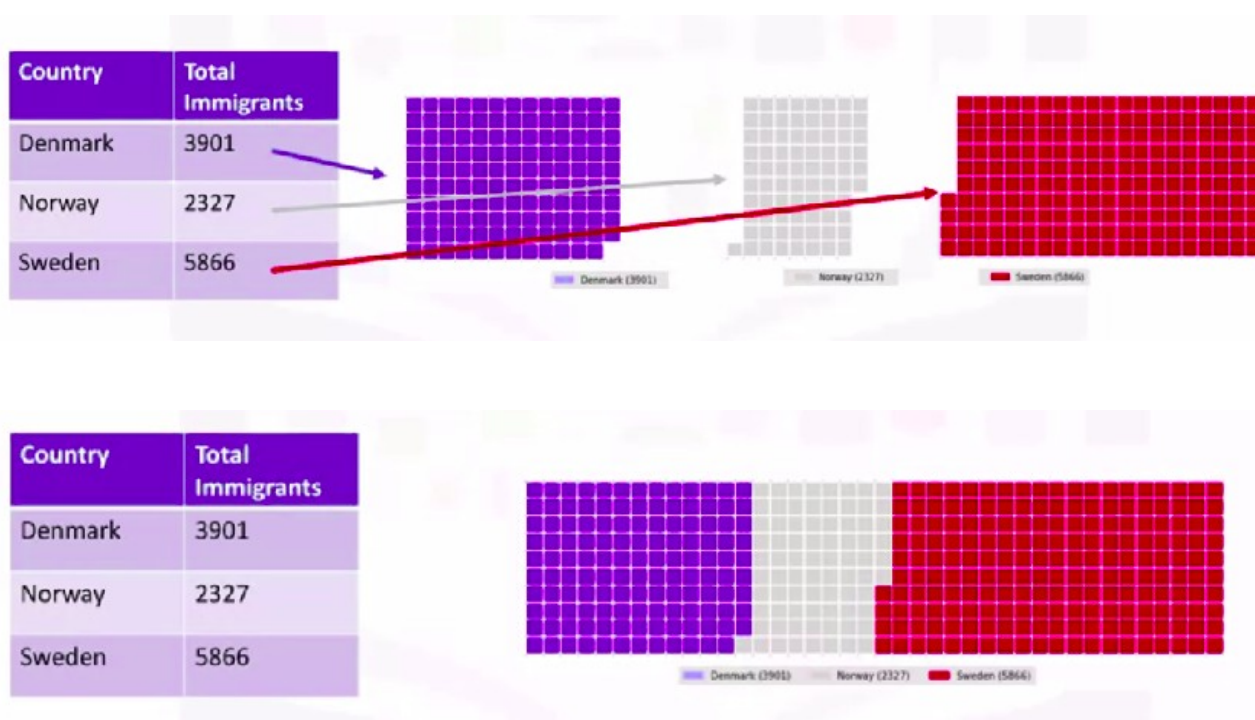
## PARTE 3. HERRAMIENTAS DE VISUALIZACIÓN AVANZADAS

### 3.1. WAFFLE CHARTS

Un **waffle chart** es una excelente manera de visualizar datos en relación con un todo o de resaltar el progreso frente a un umbral determinado.

Digamos que la inmigración de Escandinavia a Canadá se compone sólo de la inmigración de Dinamarca, Noruega y Suecia y queremos visualizar la contribución de cada uno de estos países a la inmigración escandinava a Canadá.

La idea es un waffle chart, donde se definen alto y ancho y la contribución de cada país se transforma en un número de fichas que es proporcional a la contribución del país al total, de modo que más la contribución más las baldosas.



**Figura.** Waffle Chart.

Matplotlib no tiene una función incorporada para crear waffle charts, pero en el laboratorio veremos cómo hacerlo directamente con Python.

### 3.2. NUBES DE PALABRAS (word clouds)

Una **nube de palabras** es una representación de la importancia de las diferentes palabras en el cuerpo del texto.

Una nube de palabras funciona de forma simple: cuanto más aparece una palabra específica en una fuente de datos textuales, más grande y más audaz aparece en la nube mundial.

Así que dados algunos datos de texto sobre reclutamiento generamos como ejemplo la siguiente nube de palabras:



**Figura.** Nube de palabras.

Esta nube nos dice que palabras como reclutamiento (recruitment), talento (talent), candidatos, etc. son las que destacan en estos documentos de texto.

Suponiendo que no sabíamos nada sobre el contenido de estos documentos, una nube de palabras puede ser muy útil para asignar un tema a algunos datos textuales desconocidos.

Matplotlib no tiene una función incorporada para generar nubes de palabras, pero una biblioteca de Python creada por Andreas Mueller está disponible públicamente.

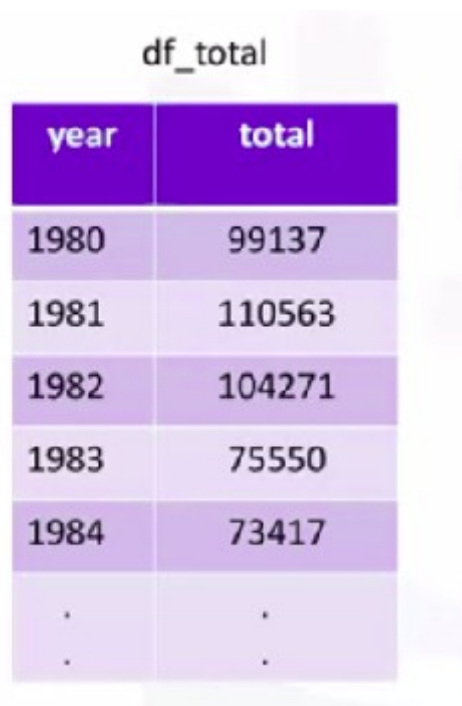
### 3.3. SEABORN Y GRÁFICOS DE REGRESIÓN

Aunque Seaborn es otra biblioteca de visualización de datos, se basa en Matplotlib. Fue construido para proporcionar una interfaz de alto nivel para dibujar gráficos estadísticos atractivos.

Seaborn hace que la creación de parcelas sea muy eficiente, con Seaborn puede generar parcelas con alrededor de 5 veces menos código que con Matplotlib.

Veamos cómo usar Seaborn para crear un gráfico de regresión.

Digamos que tenemos el dataframe `df_total` de inmigración total a Canadá con el año en una columna y la inmigración total correspondiente en otra, y que queremos crear una gráfica de dispersión junto con una línea de regresión para resaltar cualquier tendencia en los datos.



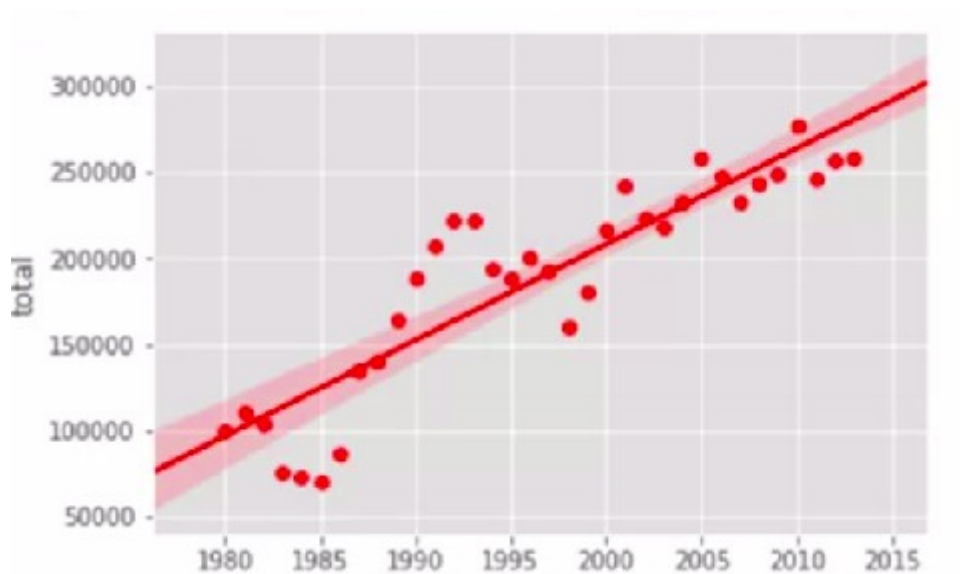
| df_total |        |
|----------|--------|
| year     | total  |
| 1980     | 99137  |
| 1981     | 110563 |
| 1982     | 104271 |
| 1983     | 75550  |
| 1984     | 73417  |
| ⋮        | ⋮      |
| ⋮        | ⋮      |

Figura. Dataframe para los ejemplos.

Con Seaborn puedes hacerlo con una línea de código (luego de importar seaborn):

```
import seaborn as sns
ax = sns.regplot(x="year", y="total", data=df_tot)
```

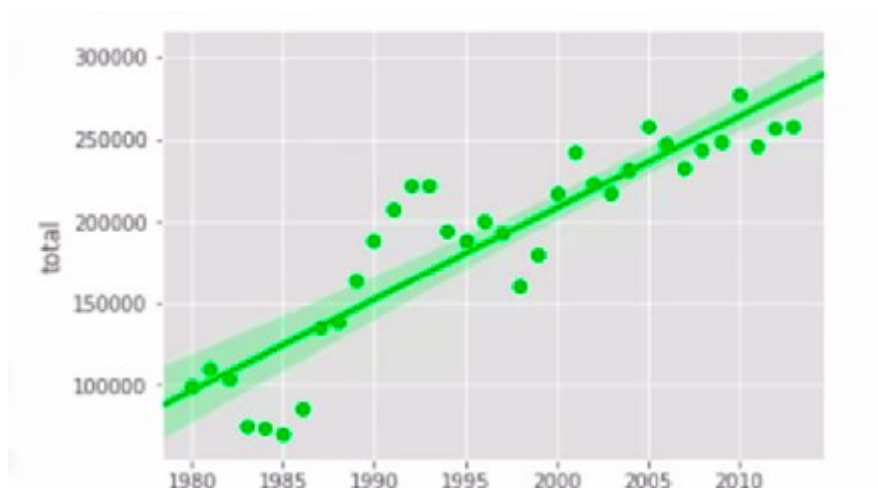
La salida de este código es una gráfica de dispersión con una recta de regresión y un intervalo de confianza del 95%.



**Figura.** Gráfico de dispersión y recta de regresión con Seaborn.

regplot acepta parámetros adicionales para realizar personalizaciones. Por ejemplo cambiar el color:

```
import seaborn as sns
ax = sns.regplot(x="year", y="total", data=df_tot, color="green")
```



**Figura.** Ejemplo de personalización con Seaborn.

### 3.4. FOLIUM

Folium es una biblioteca de visualización de datos en Python que se creó para ayudar a las personas a visualizar datos geoespaciales.

Con Folium puede:

- Crear un mapa de cualquier ubicación del mundo siempre y cuando conozca sus valores de latitud y longitud.
- Crear un mapa y superponer marcadores así como grupos de marcadores en la parte superior del mapa.
- Crear mapas de diferentes estilos como mapa de nivel de calle, mapa de estambre y otros.

Crear un mapa es sencillo:

```
# definimos el mapa mundial  
world_map = folium.Map()
```

```
# desplegamos el mapa  
world_map
```



**Figura.** Mapa con Folium.

Lo interesante de los mapas con Folium es que son interactivos, por lo que puede acercarse y alejarse después de renderizar el mapa, lo que es muy útil.

El estilo de mapa predeterminado es el mapa de calles abierto, que muestra una vista de calle de un área cuando se hace zoom y muestra los bordes de los países del mundo cuando se está alejando.

Ahora vamos a crear un mapa del mundo centrado en Canadá. Para ello pasamos los valores de latitud y longitud de Canadá utilizando el parámetro de ubicación y establecemos el zoom inicial utilizando el parámetro de inicio del zoom; es inicial porque puede cambiar fácilmente el nivel de zoom después de que el mapa se renderiza acercando o alejando. En este caso fijemos el zoom inicial en 4.

```
# definimos el mapa del mundo centrado en Canadá con un zoom bajo  
world_map = folium.Map(location=[56.130, -106.35]), zoom_start=4)
```

```
# desplegamos el mapa  
world_map
```

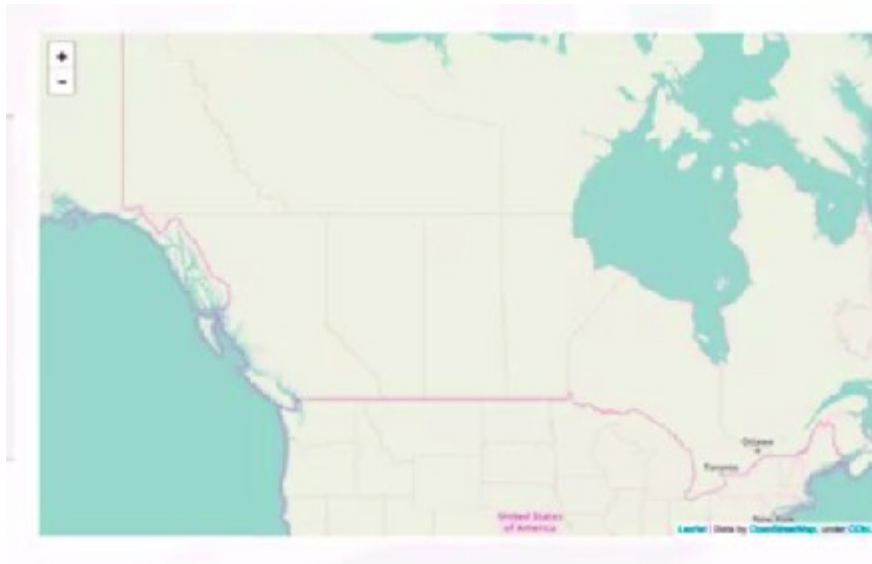


Figura. Mapa del mundo centrado en Canadá.

Folium puede crear diferentes estilos de mapas utilizando el parámetro tiles.

A continuación creamos un mapa de tónos de estambre para Canadá, que es ideal para visualizar y explorar meandros de ríos y zonas costeras.

```
world_map = folium.Map(location=[56.130, -106.35], zoom_start=4, tiles="Stamen Tomer")
```

```
world_map
```





**Figura.** Personalizando el mapa.

Ahora vamos a crear un mapa de Canadá en terreno de estambre. Este estilo es ideal para visualizar el sombreado de las colinas y los colores de la vegetación natural.

```
world_map = folium.Map(location=[56.130, -106.35], zoom_start=4, tiles="Stamen Terrain")
world_map
```



### 3.5. MAPAS Y MARCADORES

Veremos cómo imponer marcadores en un mapa.

Previamente aprendimos cómo crear un mapa del mundo centrado en Canadá; vamos a crear este mapa de nuevo y llamarlo `canada_map`.

Ontario es una provincia canadiense que contiene alrededor del 40% de la población; se considera la provincia más poblada de Canadá. Veamos cómo podemos añadir una marca circular al centro de Ontario.

Para hacer eso, necesitamos crear lo que se llama grupo de entidades. Vamos a crear un grupo de características llamado Ontario. Cuando se crea un grupo de entidades está vacío, entonces lo que sigue es crear lo que se denomina hijos y agregarlos. Vamos a crear un niño en forma de una marca circular roja ubicada en el centro de Ontario. Especificamos la ubicación del niño pasando los valores de latitud y longitud. Una vez que hayamos terminado de agregar hijos al grupo de entidades, añadimos el grupo destacado al mapa.

Sería bueno etiquetar este marcador para que otras personas sepan lo que representa. Para ello usamos la función de marcador y el parámetro `popup` para pasar el texto que queremos añadir.

```
# generamos el mapa de Canadá
```

```
canada_map = folium.Map(location=[56.130, -106.35], zoom_start=4)
```

```
## agregar un marcador rojo para Ontario
```

```
# crear un grupo de características
```

```
ontario = folium.map.FeatureGroup()
```

```
# darle estilo al grupo de características
```

```
ontario.add_child(folium.features.CircleMarker)[51.25, -85.32], radius=5, color="red",  
fill_color="Red")  
)
```

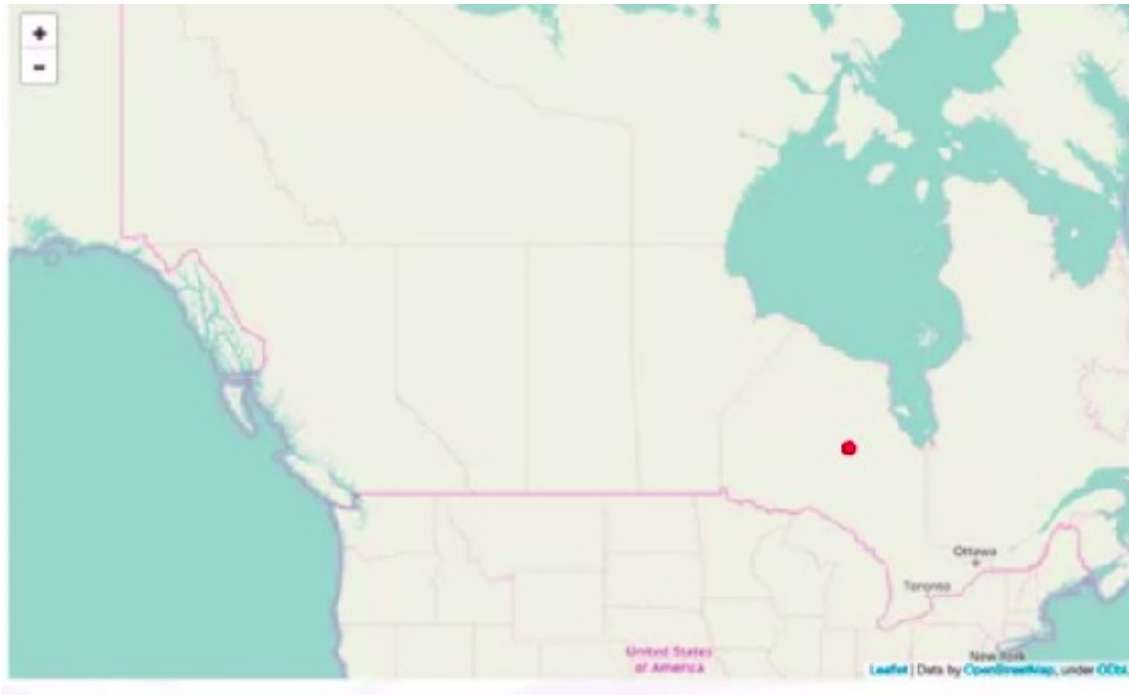
```
# agregamos el grupo de características al mapa
```

```
canada_map.add_child(ontario)
```

```
# etiquetamos el marcador
```

```
folium.Marker([51.25, -85.32], popup="Ontario").add_to(canada_map)
```

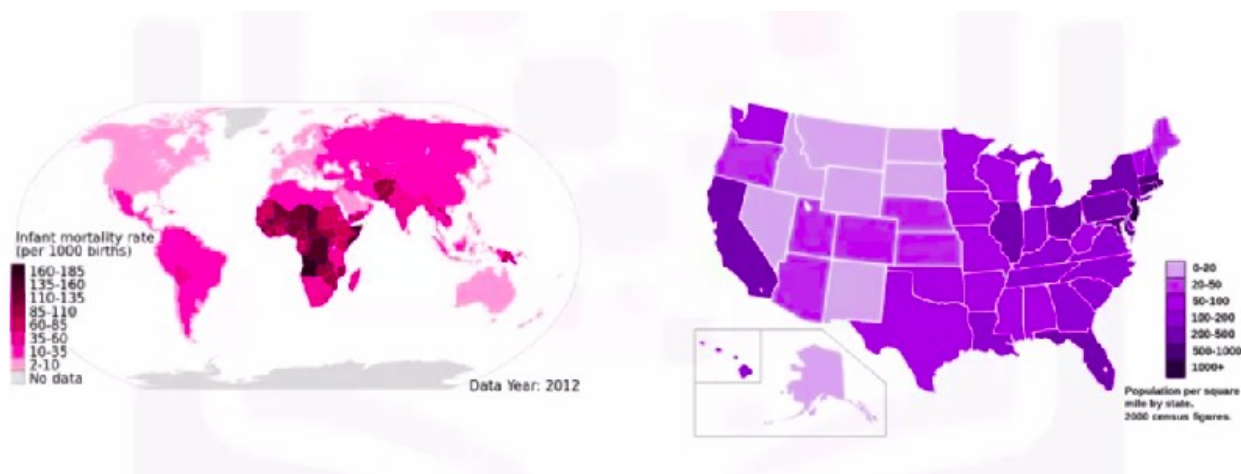
```
# desplegamos el mapa  
canada_map
```



**Figura.** Agregando marcadores.

### 3.6. MAPA DE CLOROPLETAS

La mayoría ha visto mapas similares a los de la figura siguiente, éstos se llaman mapas de cloropletas.



**Figura.** Mapas de cloropletas.

Un mapa de cloropletas es un mapa temático en el que las áreas están sombreadas o modeladas en proporción a la medida de la variable estadística que se muestra en el mapa, como la densidad de población o el ingreso per cápita. Cuánto mayor sea la medición, más oscuro será el color.

El mapa a la izquierda de la figura anterior es un mapa de cloropletas del mundo que muestra la tasa de mortalidad infantil por cada 1000 nacimientos. Cuánto más oscuro sea el color, mayor la tasa de mortalidad infantil. Según el mapa, los países africanos tienen tasas de mortalidad muy altas y algunos de ellos comunican una tasa superior a 160 por cada 1000 nacimientos.

El mapa de la derecha de la figura anterior es un mapa de cloropletas de Estados Unidos que muestra la población por milla cuadrada por estado. Cuánto más oscuro el color, mayor es la población. Según el mapa, los estados de la parte oriental de EE.UU tienden a ser más poblados que los de parte occidental, siendo California una excepción.

Para crear un mapa de coropletas de una región de interés, Folium requiere un archivo Geo JSON que incluya datos geoespaciales de la región. Para un mapa de coropletas del mundo, necesitaríamos un archivo Geo JSON que enumera cada país junto con cualquier dato geoespacial para definir sus fronteras y límites. En la figura siguiente se muestra un ejemplo de lo que el archivo Geo JSON incluiría sobre cada país. El ejemplo se refiere a Brunei.

```
{
  "type": "FeatureCollection",
  "features": [{
    "type": "Feature",
    "properties": {
      "name": "Brunei"
    },
    "geometry": {
      "type": "Polygon",
      "coordinates": [
        [
          [114.204017, 4.525874], [114.599961, 4.900011], [115.45071, 5.44773],
          [115.4057, 4.955228], [115.347461, 4.316636], [114.869557, 4.348314],
          [114.659596, 4.007637], [114.204017, 4.525874]
        ]
      ]
    }
  ]
},
  "id": "BRN"
},
```

**Figura.** Ejemplo de Geo JSON.

El archivo incluye el nombre del país, su ID, la forma de la geometría y las coordenadas que definen los límites del país.

Veamos cómo crear un mapa de coropletes que muestre la inmigración a Canadá.

Trabajaremos con el dataframe `df_canada` procesado como en ejemplos anteriores.

Creamos el mapa del mundo utilizando el conjunto de mosaicos brillantes del mapbox. El resultado es un mapa del mundo que muestra el nombre de cada país. Ahora, para convertir este mapa en un mapa de coropletas, primero definimos una variable que apunta a nuestro archivo Geo JSON. Luego aplicamos la función `coropleth` a nuestro mapa del mundo y le decimos que use las columnas «País» y «Total» en nuestro marco de datos `DF_Canada`, y que use los nombres de los países para buscar la información geoespacial sobre cada país en el archivo Geo JSON.

```
# creamos un mapa plano del mundo
```

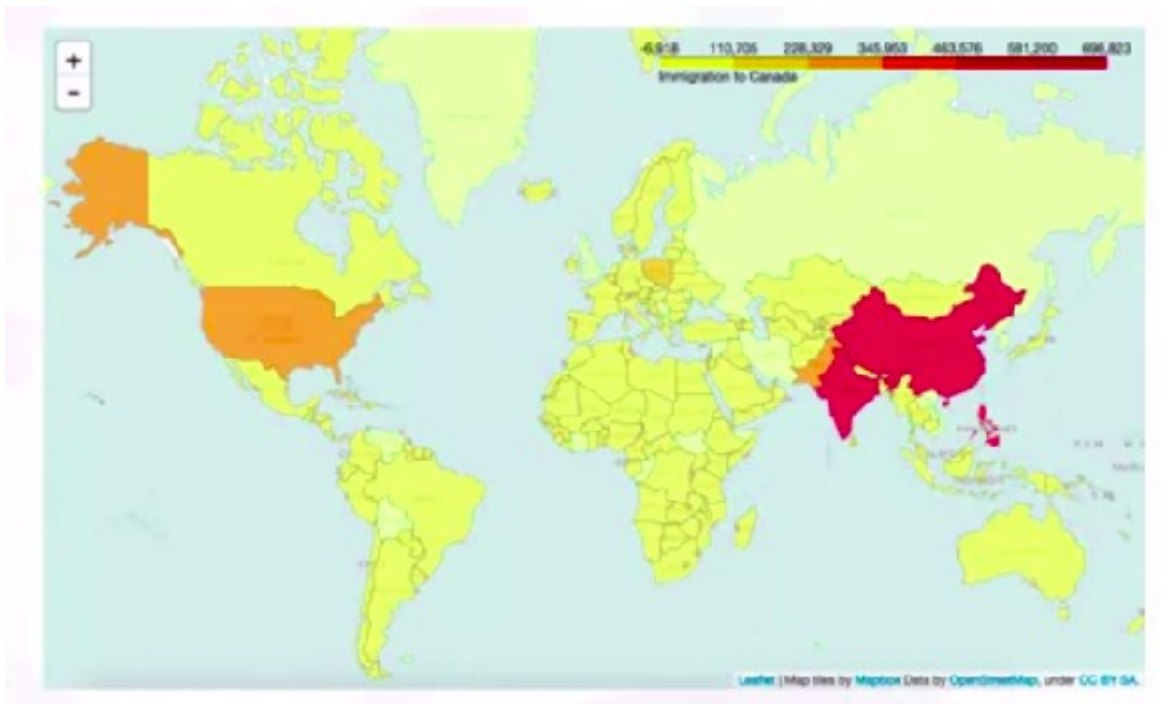
```
world_map = folium.Map(zoom_start=2, tiles="Mapbox Bright")
```

```
# archivo geojson
```

```
world_geo = r"world_countries.json"
```

```
# generamos el mapa de cloropletas usando la población total de cada país
# que emigró a Canadá
world_map.choropleth(
    geo_path = world_geo,
    data = df_canada,
    columns = ["Country", "Total"],
    key_on = "features.properties.name",
    fill_color = "YlOrRd",
    legend_name = "Immigration to Canada"
)

# desplegamos el mapa
world_map
```



**Figura.** Mapa de cloropletas.