Seguimiento de datos COVID-19 en Colombia

Santiago Gutiérrez Orjuela
Escuela de Ciencias exactas e Ingeniería
Universidad Sergio Arboleda - Bogotá, Colombia
santiago.gutierrez02@correo.usa.edu.co

Valeria Bermúdez Galván
Escuela de Ciencias exactas e Ingeniería
Universidad Sergio Arboleda - Bogotá, Colombia
valeria.bermudez01@correo.usa.edu.co

Juan Sebastián Bueno Ramírez
Escuela de Ciencias exactas e Ingeniería
Universidad Sergio Arboleda - Bogotá, Colombia
juan.bueno01@correo.usa.edu.co

Resumen

En el presente documento se aplica la extracción de datos en internet sobre el coronavirus en Colombia para de forma selectiva presentarlos mediante diferentes gráficos. Se realiza código en Python, haciendo uso de web scraping para extraer los datos del Covid-19, se manipularán y visualizarán empleando librerias tales como pandas y matplotlib.

Palabras cláve:

Python, web scraping, datos, Covid-19.

1. Marco teórico

El **coronavirus** es una familia de virus que causa infecciones respiratorias que pueden ir desde el resfriado común hasta el síndrome repiratorio agudo severo. El coronavirus que se ha descubierto más recientemente causa la enfermedad por coronavirus **COVID-19**, del cuál los síntomas más habituales son fiebre, tos seca y cansancio. Para el estudio del impacto que ha causado esta enfermedad causada por el virus, se representará en gráficos la información que se tiene de los casos confirmados en el país de Colombia.

Los *datos abiertos* del gobierno de Colombia nos permitió acceder a la información asociada a los casos positivos de COVID-19, haciendo uso de web scraping en Python con sodapy que es una biblioteca que funciona como cliente en la *Socrata Open Data API*, estableciendo así conexión con la página del gobierno y obteniendo los datos necesarios, guardándolos como formato csv, estos se guardan en un dataframe con el uso de **Pandas**, el cual es una biblioteca que facilita la manipulación de estructuras de datos y nos brinda herramientas para el análisis de datos [1]. Para la visualización de los datos, se hace uso de **Matplotlib**, esta biblioteca nos permite hacer gráficas en 2D in Python, originalmente fue un emulador de los comandos de gráficos de MATLAB.

2. Resultados

Para el uso de *sodapy* para acceder a `Socrata`, escribimos las siguientes líneas

```
from sodapy import Socrata
import pandas as pd
from datetime import datetime
import matplotlib.pyplot as plt
from collections import Counter


client = Socrata("www.datos.gov.co", None) #Se determina la direcci n de la cual se extraer n ...
    los datos
results = client.get("gt2j-8ykr", limit=700000) #Se obtienen los datos y se establece un l mite

# Convertir a dataframe de pandas
results_df = pd.DataFrame.from_records(results)
```

Con el fin de usar buenas prácticas, se llenan los espacios en blanco con el campo ”No definido”.

```
df=results_df
df=df.fillna('No Definido') #Se llenan los datos vac os
tupla = (("T00:00:00.000", ""),("-", "/"))
for c in df.index:
    for a, b in tupla:
        df['fecha_de_notificaci_n'][c] = df['fecha_de_notificaci_n'][c].replace(a, b)
        df['fis'][c] = df['fis'][c].replace(a, b)
        df['fecha_diagnostico'][c] = df['fecha_diagnostico'][c].replace(a, b)
        df['fecha_recuperado'][c] = df['fecha_recuperado'][c].replace(a, b)
        df['fecha_reporte_web'][c] = df['fecha_reporte_web'][c].replace(a, b)
        df['fecha_de_muerte'][c] = df['fecha_de_muerte'][c].replace(a, b)
df.head()
```

A continuación , se muestra el código usado para preparar la data y crear las gráficas necesarias.

```
#Se convierten los datos de string a datatime object
for c in df.index:
    if(df['fecha_de_notificaci_n'][c]!='No Definido'):
        df['fecha_de_notificaci_n'][c]=datetime.strptime(df['fecha_de_notificaci_n'][c], '%Y/%m/%d')
    if(df['fis'][c]!='No Definido'):
        df['fis'][c]=datetime.strptime(df['fis'][c], '%Y/%m/%d')
    if(df['fecha_diagnostico'][c]!='No Definido'):
        df['fecha_diagnostico'][c]=datetime.strptime(df['fecha_diagnostico'][c], '%Y/%m/%d')
    if(df['fecha_recuperado'][c]!='No Definido'):
        df['fecha_recuperado'][c]=datetime.strptime(df['fecha_recuperado'][c], '%Y/%m/%d')
    if(df['fecha_reporte_web'][c]!='No Definido'):
        df['fecha_reporte_web'][c]=datetime.strptime(df['fecha_reporte_web'][c], '%Y/%m/%d')
    if(df['fecha_de_muerte'][c]!='No Definido'):
        df['fecha_de_muerte'][c]=datetime.strptime(df['fecha_de_muerte'][c], '%Y/%m/%d')

x=[]
x=df['fecha_de_notificaci_n'] #Se llena con la fecha de notificaci n
c=Counter(x) #Se cuentan los datos seg n la fecha de notificaci n
df1 = pd.DataFrame() #Se crea una nueva referencia del DataFrame
df1['Fecha']=c.keys() #Se extraen los elementos  nicos
df1['No. Casos']=c.values() #Se extrae la frecuencia de cada elemento  nico
df1=df1.sort_values(by='Fecha') #Se ordenan seg n la fecha

x = df1['Fecha'] #Se asignan los valores al eje x
y = df1['No. Casos'] #Se asignan los valores al eje y
plt.plot(x,y) #Se grafica
plt.title('N mero de Casos')
plt.xlabel('Fecha')
plt.ylabel('No. Casos')
plt.savefig('N mero de Casos.png') #Se guarda la imagen

import matplotlib
import numpy as np

y=[]
y=df['ciudad_de_ubicaci_n'] #Se llena con la ciudad
d=Counter(y) #Se cuentan los datos seg n la ciudad
df2 = pd.DataFrame()
df2['Ciudad']=d.keys() #Se extraen los elementos  nicos
df2['No. Casos']=d.values() #Se extrae la frecuencia de cada elemento  nico
df2=df2.sort_values(by='No. Casos',ascending=False) #Se ordena seg n el n mero de casos

plt.figure(figsize=(10,7)) #Se establece el tama o de la figura
#Colocamos etiquetas a los ejes
plt.xlabel("Ciudades")
plt.ylabel("N mero de Casos")

#Creamos la grafica de barras utilizando 'Ciudad' como eje X y 'N mero de casos' como eje y.
x = df2['Ciudad'].iloc[0:6]
y = df2['No. Casos'].iloc[0:6]
plt.title("Ciudades con mayor n mero de Contagiados")
plt.bar(x, y)
#Finalmente mostramos la grafica con el metodo show()
plt.savefig('Ciudades.png')
plt.show()
```

```
56
57 tipo=[]
58 tipo=df['tipo']
59 #Se cuentan los casos seg n el tipo de contagio
60 e=Counter(tipo)
61 df3 = pd.DataFrame()
62 #Se establecen los elementos y la frecuencia de cada uno
63 df3['Tipo']=e.keys()
64 df3['No. Casos']=e.values()
65 #Se ordenan
66 df3=df3.sort_values(by='No. Casos',ascending=False)
67
68 #Se grafica el diagrama de torta
69 plt.pie(df3['No. Casos'], labels=df3['Tipo'], autopct="%0.1f %%")
70 plt.axis("equal")
71 plt.title("Casos Totales por tipo de contagio")
72 plt.savefig('TipoContagio.png')
73 #Se muestra la grafica
74 plt.show()
75
76 estado=[]
77 estado=df['estado']
78 #Se cuentan los casos seg n el estado de los pacientes
79 f=Counter(estado)
80 df4 = pd.DataFrame()
81 #Se establecen los elementos y la frecuencia de cada uno
82 df4['Estado']=f.keys()
83 df4['No. Casos']=f.values()
84 df4=df4.sort_values(by='No. Casos',ascending=False)
85
86 #Se grafica el diagrama de torta
87 plt.pie(df4['No. Casos'], labels=df4['Estado'], autopct="%0.1f %%")
88 plt.axis("equal")
89 plt.title("Estado de los pacientes")
90 plt.legend()
91 plt.savefig('EstadoPacientes.png')
92 plt.show()
93
94 pais=[]
95 pais=df['pa_s_de_procedencia']
96 #Se cuentan los casos seg n el pa s de procedencia
97 h=Counter(pais)
98 df6 = pd.DataFrame()
99 #Se establecen los elementos y la frecuencia de cada uno
100 df6['Pais de Procedencia']=h.keys()
101 df6['No. Casos']=h.values()
102 df6=df6.sort_values(by='No. Casos',ascending=False)
103 #Se hace una copia del data frame
104 df6copy = df6.iloc[1:8].copy()
105 #Se organizan seg n el n mero de casos
106 df6copy=df6copy.sort_values(by='No. Casos')
107
108 plt.figure(figsize=(9,7))
109 #Creamos la grafica pasando los valores en el eje X, Y, donde X = No de Casos y Y = Pa s de ...
      Procedencia
110 x = df6copy['Pais de Procedencia']
111 y = df6copy['No. Casos']
112 plt.barh(x, y, align='center', alpha=0.5)
113 #a adimos una etiqueta en el eje X
114 plt.xlabel('N mero de Casos')
115 plt.title('Pa s de Procedencia')
116 plt.savefig('Pais.png')
117 plt.show()
118
119 muertes=[]
120 muertes=df['fecha_de_muerte']
121 #Se cuentan los casos seg n la fecha de muerte
122 g=Counter(muertes)
123 df5 = pd.DataFrame()
124 #Se establecen los elementos y la frecuencia de cada uno
125 df5['Muertes']=g.keys()
```

```
df5['No. Casos']=g.values()
df5=df5.sort_values(by='No. Casos',ascending=False)
#Se hace una copia del data frame
copiamuertes=df5.iloc[1::].copy()

#Se organiza seg n el n mero de muertes
copiamuertes=copiamuertes.sort_values(by='Muertes')
x = copiamuertes['Muertes']
y = copiamuertes['No. Casos']
#Se grafica
plt.plot(x,y)
plt.title('Fallecimientos  ')
plt.xlabel('Fecha')
plt.ylabel('Muertes')
#Se guarda la gr fica como .png
plt.savefig('Fallecimientos.png')
```

Mediante los datos extraídos de la página del Ministerio de Colombia, se realizó la gráfica de crecimiento de los casos desde el mes de marzo hasta el mes de julio , correspondiente al número de datos extraídos. Evidenciando el aumento de los casos a lo largo de los meses.

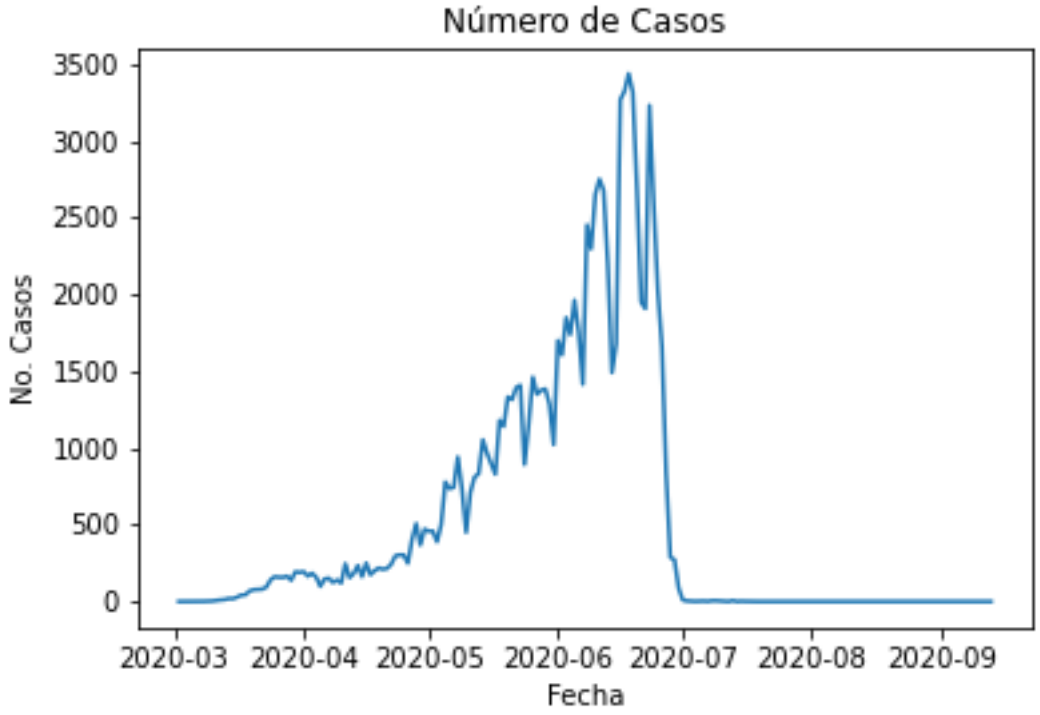

Figura 1: Número de casos positivos por meses en Colombia

En la siguiente gráfica, se muestran las ciudades con mayor número de contagios dando cuenta que Bogotá DC es la que posee mayor número de contagios.

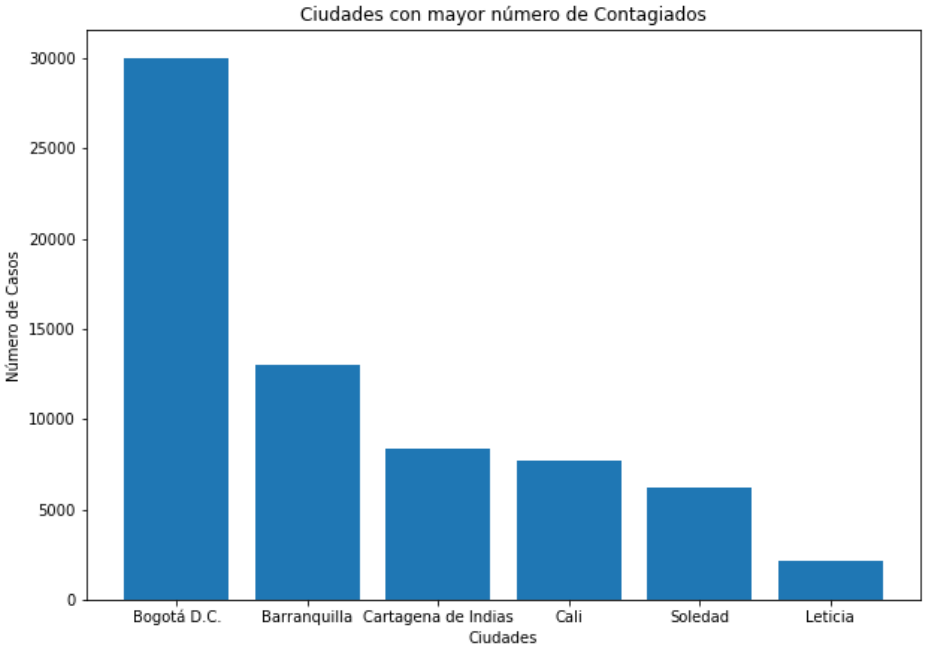

Figura 2: Número de contagios por ciudad en Colombia

Luego, se realizó una gráfica de tortas con el fin de observar el estado de los pacientes según sus sintomas, ya sea leve, asintomatico, moderado, fallecido, grave, N/A permitiendo observar que el 83 % de los pacientes poseen sintomatología leve.

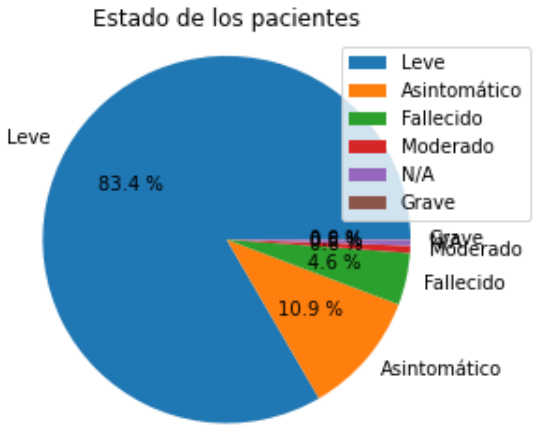

Figura 3: Estado de los pacientes de COVID-19 en Colombia

También, se realizó un gráfico de tortas para saber qué tipos de contagios poseen las personas aunque con el aumento exponencial de casos la mayoría de casos han quedado en estudio.

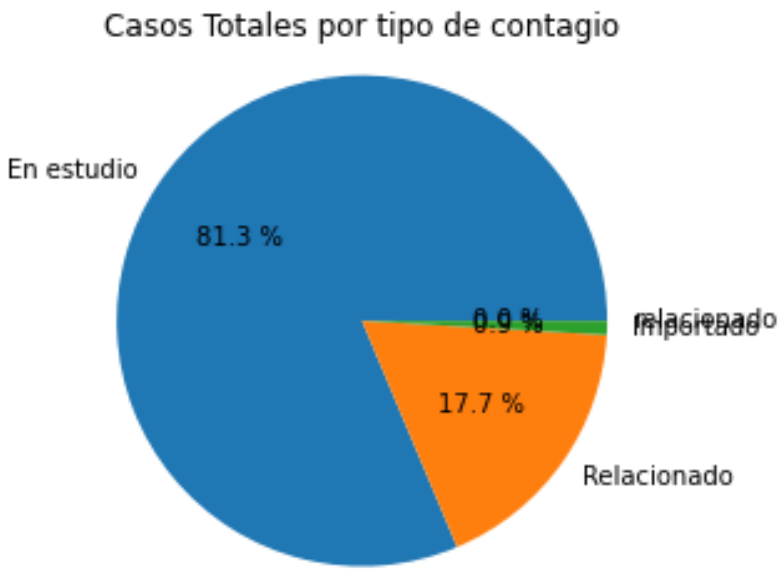

Figura 4: Tipo de contagio en pacientes colombianos

Posteriormente, se graficó el número de fallecidos por mes donde se puede evidenciar que el mes con mayor cantidad de muertos hasta el día de hoy es Julio.

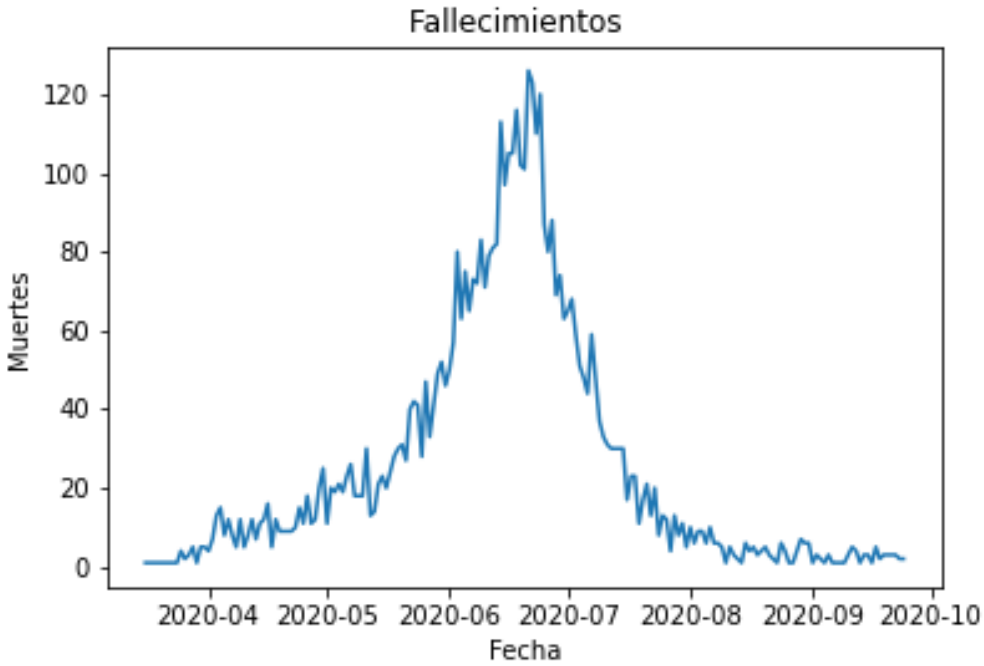

Figura 5: Número de fallecidos por mes en Colombia

Por último, mediante un gráfico de barras se mostró que la mayoría de casos importados provienen de los países España y Estados Unidos, quienes ya tuvieron su pico de la pandemia.

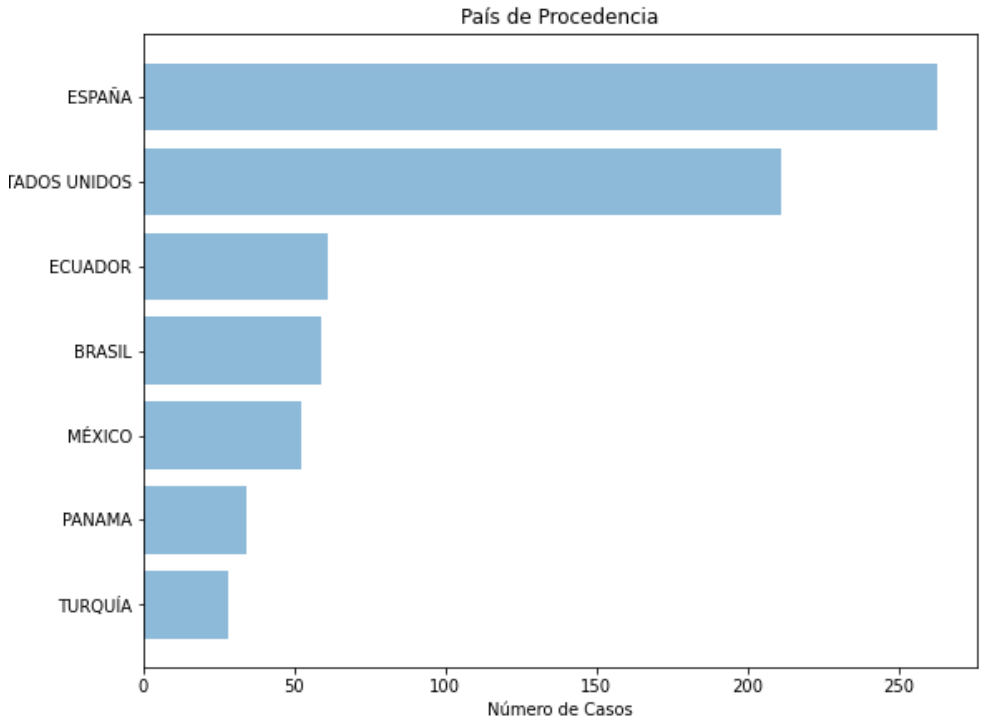

Figura 6: País de procedencia de los pacientes importados

3. Conclusiones

Se identifica que Pandas y Matplotlib nos facilita el manejo de los datos y su visualización.

Se evidencia que el número de contagios crece exponencialmente, tal como se ve en la figura 1.

Más del 80 % de los contagios no se sabe de dónde vienen, siguen en estudio, esto debido al crecimiento exponencial de casos no permitiendo determinar la proveniencia de la mayoría de contagios.

Bogotá es la ciudad con más contagios debido a ser la que tiene más habitantes y además donde llegan los vuelos internacionales.

Referencias

[1] T. pandas development team, "User guide." The pandas development team, 2008-2020.

[2] O. M. de la Salud, "Preguntas y respuestas sobre la enfermedad por coronavirus (covid-19)." OMS, 2020.

[3] P. S. Foundation., "Python 3.8.6 documentation." Python Software Foundation, 2001-2020.

[4] F. D. Hunter J, Dale and the Matplotlib development team, "Tutorials." The Matplotlib development team, 2012 - 2020.

[5] S. J, "Data visualization with plotly and pandas." Socrata, 2016.

[6] M. de Tecnologías de la Información y las Comunicaciones, "Casos positivos de covid-19 en colombia." Datos Abiertos, 2020.