Proyecto final seguimiento covid

Johan Villalba
Escuela de Ciencias Exactas e Ingeniería
Universidad Sergio Arboleda-Bogotá, Colombia
johan.villalba01@correo.usa.edu.co

Sebastián Merchán
Escuela de Ciencias Exactas e Ingeniería
Universidad Sergio Arboleda-Bogotá, Colombia
sebastian.merchan01@correo.usa.edu.co

Miguel Angel Rippe
Escuela de Ciencias Exactas e Ingeniería
Universidad Sergio Arboleda-Bogotá, Colombia
miguel.rippe01@correo.usa.edu.co

Resumen

Para este proyecto se muestra la solución a una aplicación la cual muestra en tiempo real la cantidad de contagios de Covid-19 tanto en Bogotá como en Colombia, la cantidad de recuperados, los respectivos decesos en el país y con la ayuda del machine learning se realiza una predicíon a 2 meses del comportamiento de este. Dicha aplicación se realizo en el lenguaje interpretado de python.

1. Marco teórico

1.1. Python

Python es un lenguaje de programación de propósito general, este lenguaje se define comúnmente como un lenguaje de secuencias de comandos las cuales están orientado a objetos, pero no solo se utiliza para este paradigma de programación, ya que es un lenguaje multiparadigmas, python trabaja con los paradigmas procedimentales, funcionales y oriendo a objetos, por eso es un lenguaje muy conocido y ulitizado. [1]

1.2. Covid-19

Los corona virus son una familia de virus que pueden generar enfermedades tanto en humanos, como en animales. Esta enfermedad se manifiesta en los humanos en su mayoria como infecciones, respiratorias las cuales pueden ir desde resfriados leves hasta enfermedades graves. El covid-19 es el tipo de corona virus más reciente que se a encontrado en la ciudad de Wuhan en China y actualmente se a convertido en una pandemia que afecta a muchos países en el mundo. [2]

1.3. Web scraping

El web scraping es el termino comúnmente usado para la extracción de datos con base a una pagina web. Existen diferentes métodos de extracción de datos con base al software que sera usado, estos datos pueden ser usados en varios softwares, tales como: lenguajes de programación, bases de datos, archivos de tipo excel, entre otros.

2. Recursos y software

Para realizar este proyecto y llegar a lo requerido por el docente se utilizaron diferentes recursos los cuales ayudaron con el buen funcionamiento del proyecto como lo son:

- **Pycharm:** Se utilizo este interprete de python para la construcción del código sobre el cual se desarrollara el presente proyecto.
- **BeautifulSoup:** Es una librería propia del lenguaje de programación python, la cual permite analizar documentos de tipo html, es decir paginas web que hayan diseñadas en html, mediante esta librería se es posible buscar información y exportar datos sobre una pagina web dada.
- **requests:** Es una librería propia de python la cual permite generar conexiones, avisos y permisos sobre la url de una pagina web, esto para permitir el acceso sobre esta.

- **matplotlib:** Es una libreria para generar gráficos a partir de datos conglomerados en arrays en lenguaje de programación Python y con una extensión matemática NumPy.

- **Folium:** Esta es la libreria que se utiliza para extraer los mapas, y adicionale diferentes figuras y guardarlos como html.

- **Mysql:** Es un sistema de gestión de bases de datos, esta se basa en el funcionamiento de los datos de manera relacional.

3. Metodologia

3.1. Extracción de datos

Para la extracción de los datos hicimos uso de la pagina web Wikipedia, esto debido a que esta documentando la situación con base al día a día, y además de esto, la extracción de datos sobre esta resulto ser muy eficiente, permite extraer todo el contenido e información de la pagina, y no esta protegida contra el web scraping. Esta fuente es usada por Google Noticias para dar las actualizaciones acerca del comportamiento del Covid-19 en Colombia. Nos puede proporcionar datos muy específicos, como la población de hombres, mujeres, menores de edad y mayores de edad que han contraído la enfermedad,también divide los casos por municipios, y por ciudades, nos brinda información de los mapas de calor del covid, así podrán ser usados en posteriores entregas y a lo largo del desarrollo de este proyecto.

		Grupo Etario (Años)[4]										Total [4 8 9]	
		0 a 9	10 a 19	20 a 29	30 a 39	40 a 49	50 a 59	60 a 69	70 a 79	80 a 89	≥90	Hoy	Acumulado
Estado[4]	Recuperado[1]	26747	48217	164434	177229	121713	95743	51773	24088	10581	2011	*11064*	**722536**
	En Casa	1825	3713	12204	12367	8609	6859	3814	1824	819	162	*-6004*	**52196**
	En Hospital	702	351	1110	1572	1922	2646	2694	1950	1070	188	*-25*	**14205**
	En UCI	41	15	62	154	247	431	495	322	129	11	*-63*	**1907**
	Fallecido[1]	40	33	289	745	1660	3514	5944	6588	5386	1442	*153*	**25641**
Total [4 8 9]	Hoy	*182*	*369*	*1099*	*1165*	*861*	*643*	*456*	*223*	*124*	*25*	***5147***	
	Acumulado	**29392**	**52357**	**178171**	**192127**	**134270**	**109387**	**65086**	**35170**	**18325**	**3918**		**818203**

Figura 1: Tabla de datos de los contagios proporcionada por Wikipedia.

Una vez ubicados los datos a ser usados, mediante el lenguaje de programación python y a un interpretador de nombre ”pycharm”se iba a realizar la extracción de datos. Para la extracción de datos era necesario hacer uso de varias librerías provistas por python, las cuales serian: BeautifulSoup y requests.

3.2. Creación de la base de datos

La creación de la base de datos se realizo con el programa MySql Workbench, el nombre que se le dio fue covid, y se usara después para que sea conectada con Python, la base de datos consta de una tabla llamada casos, la tabla tendrá atributos que serán la mayoría de tipo entero, y solo el idcasos sera de tipo varchar, el id sera la llave primaria, y en nuestro caso se utiliza la fecha para que no se repitan los datos en un mismo día. Las sentencias utilizadas fueron las que se muestran a continuación:

```
drop database if exists covid;
create database covid;
use covid;

create table casos(
id_casos varchar(15) primary key,
numCasos int,
numCasosHoy int,
numHospi int,
numHospiHoy int,
numFalle int,
numFalleHoy int,
numCasa int,
```

Figura 2: Sentencias utilizadas para la creación de la base de datos

3.3. Conexión a la base de datos

Para la conexión sobre la base de datos se hizo uso de una librería de nombre: pymysql, la cual permite generar una conexión entre un programa de python y una base de datos de tipo mysql.

Con base a la base de datos creada anteriormente se generara la conexión entre el script de python y esta, para ello es necesario declarar una variable con los atributos de la librería previamente mencionada y indicar los datos sobre la base de datos, tales como: el nombre del host, el usuario, el nombre asignado a la base de datos, la contraseña de la misma, el formato de los datos que se exportaran a la base de datos y el cursor el cual nos permitirá ejercer operaciones sobre la base de datos desde el script de python.

```
connection = pymysql.connect(host='localhost',
                             user='root',
                             password='1234',
                             db='covid',
                             charset='utf8mb4',
                             cursorclass=pymysql.cursors.DictCursor)
```

Figura 3: Conexión y parámetros usados para la conexión

4. Resultados

En la parte de resultados tendremos dividido en dos partes significativas, primero los datos que fueron obtenidos con el procedimiento del código, van a ser almacenados en una base de dato. En este caso usamos MySQL para que todos los datos de contagios, recuperados y personas fallecidas queden guardados en la base de datos.
Los datos van a ser almacenados cada día que se haga la ejecución y compilación del programa, es decir si se compila dos veces el programa en el día solo va almacenar en la base de datos la información correspondiente a ese mismo día. Asi mismo los datos son almacenados en una tabla que se llama casos, y que contiene los siguientes datos

- Numero de contagiados totales en Colombia
- Numero de nuevos contagios en el día de hoy
- Numero de recuperados
- Numero de personas que se están recuperando en casa tanto en el día de hoy como el acumulado
- Numero de personas en el hospital tanto en el día de hoy como el acumulado
- Numero de personas que están en la UCI tanto en el día de hoy como el acumulado

La tabla queda representada en la base de datos de la siguiente forma:

id_casos	numCasos	numCasosHoy	numHospi	numHospiHoy	numFalle	numFalleHoy	numCasa	numCasaHoy	numRecupe	numRecupeHoy	numUci	numUciHoy
1982	818203	5147	14205	-25	25641	153	52196	-6004	722536	11064	1907	-63
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figura 4: Tabla de la base de datos

A medida que el programa sea ejecutado y se requieran los datos de un día en especifico se irán guardando en esta tabla, y en la base de datos. Estos datos pueden ser utilizados después para hacer un estudio de los casos a lo largo del tiempo.

Luego de especificar los datos, se quiere mostrar un cambio en el comportamiento que han tenido desde el primer día de contagio mostrando un acumulado y por otra parte un cambio más reciente como lo es mostrar una cantidad del día. Es importante destacar que en las gráficas se muestran valores negativos, los cuales muestran un comportamiento inverso al de contagio, en otras palabras las personas que se han recuperado de Covid-19, estos valores negativos se comparan con un conjunto de personas las que poseen la enfermedad para lograr ver la velocidad a la que se desocupan las UCI y bajan los contagios.

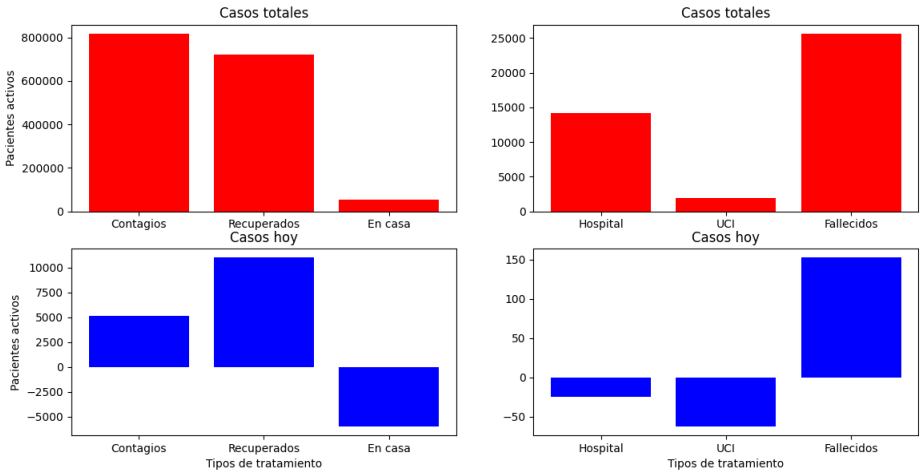


Figura 5: Gráficas de barra

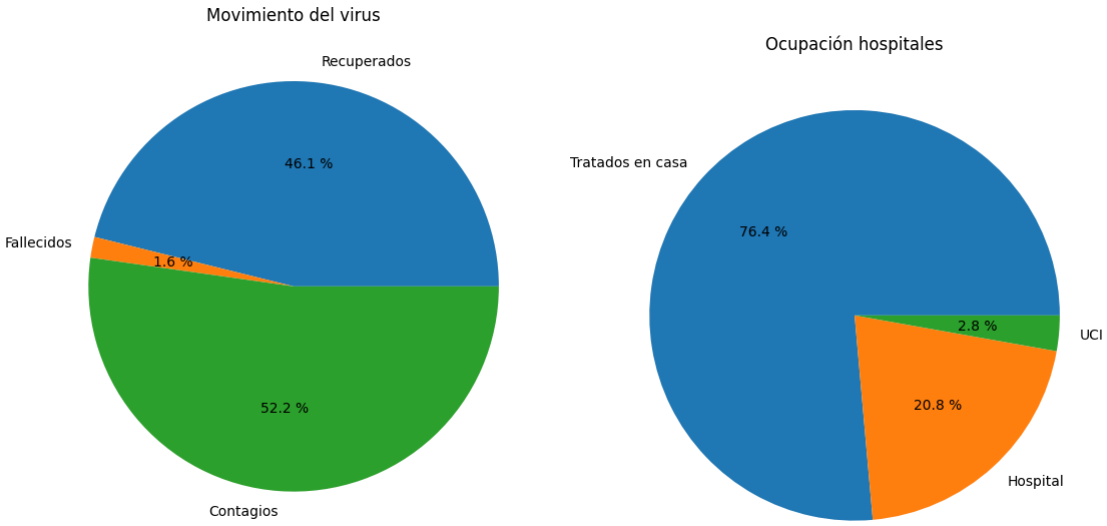


Figura 6: Porcentaje contagio y ocupación hospitales

En la segunda parte el objetivo era extraer los datos por localidades del nivel de contagio de covid-19 en Bogota. Para esto usamos otras dos paginas adicionales a las que ya teniamos con el fin de realizar el web scrapping de Bogota. Esto se realizo de la siguiente manera:

```
contagios_bogota=soup_1.find_all('li')
contagios_localidad=soup_1.find_all('strong')
contagios_colombia=soup_2.find_all('span')
datos=soup_3.find_all('th')

Total_cbta=list()
Total_cbta_1=list()
Total_ccol=list()
total=list()

#Web scraping por localidades Bogotá
for i in contagios_bogota:
   Total_cbta.append(i.text)


for i in Total_cbta:
    if(recorrer<19):
        for e in i:
            if(e==':'):
                espacio=1
            if((espacio==1)and(e!=(".")and(e!=':')and(e!=" ")and(e!=('\xa0')))):
                longitud=longitud+1
                #print(e)
                localidades.insert(0,e)


        #print('\n'+str(longitud)+'\n')
        aux.insert(0,longitud)
        espacio = 0
        longitud=0
        recorrer+=1
```

Figura 7: Web Scraping sobre los datos de Bogotá

Teniendo los datos ya extraidos podemos realizar las graficas correspondientes a cada localidad y a el numero de casos existentes en la ciudad de Bogota.

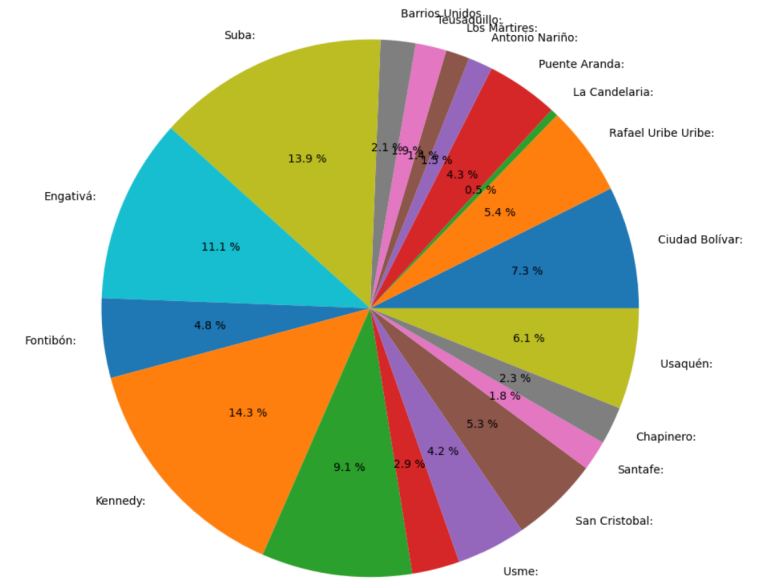


Figura 8: Grafico de torta por localidades de Bogotá

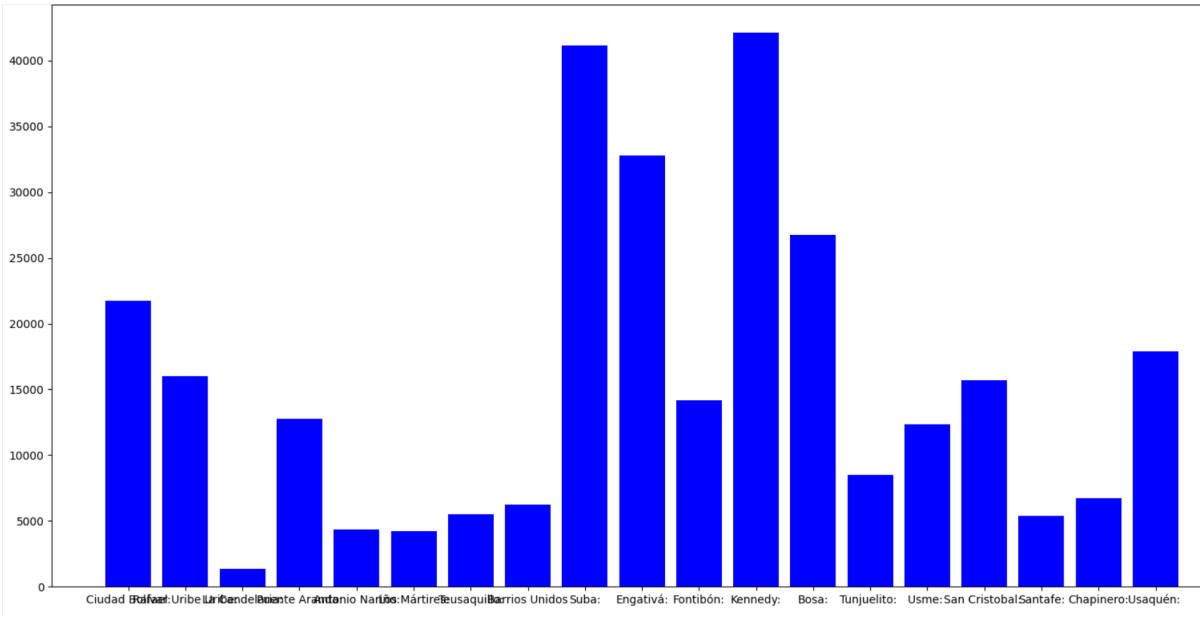


Figura 9: Grafico de barras por localidades de Bogotá

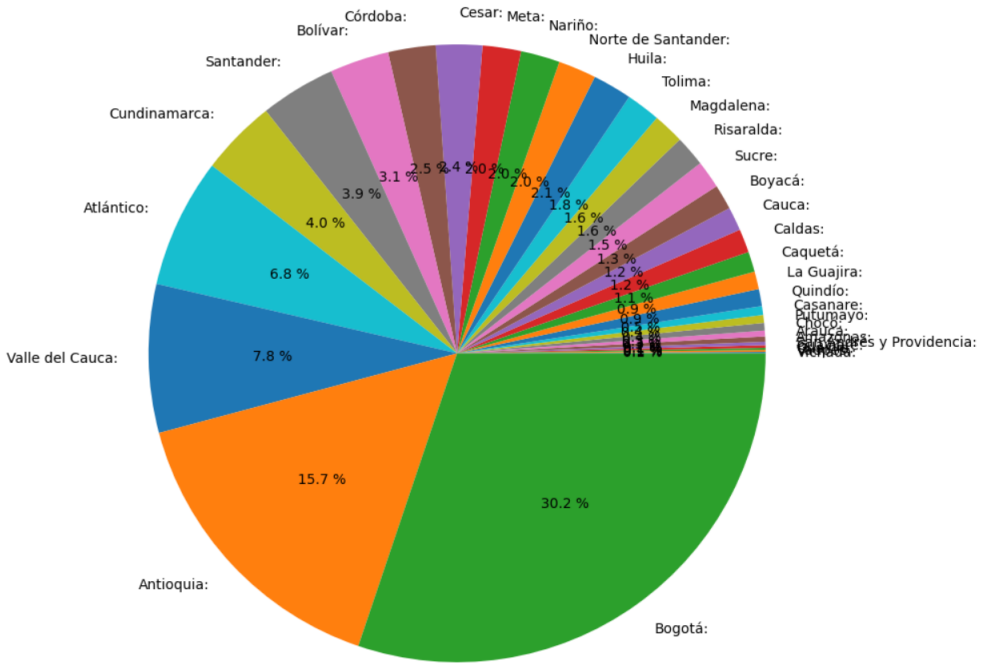
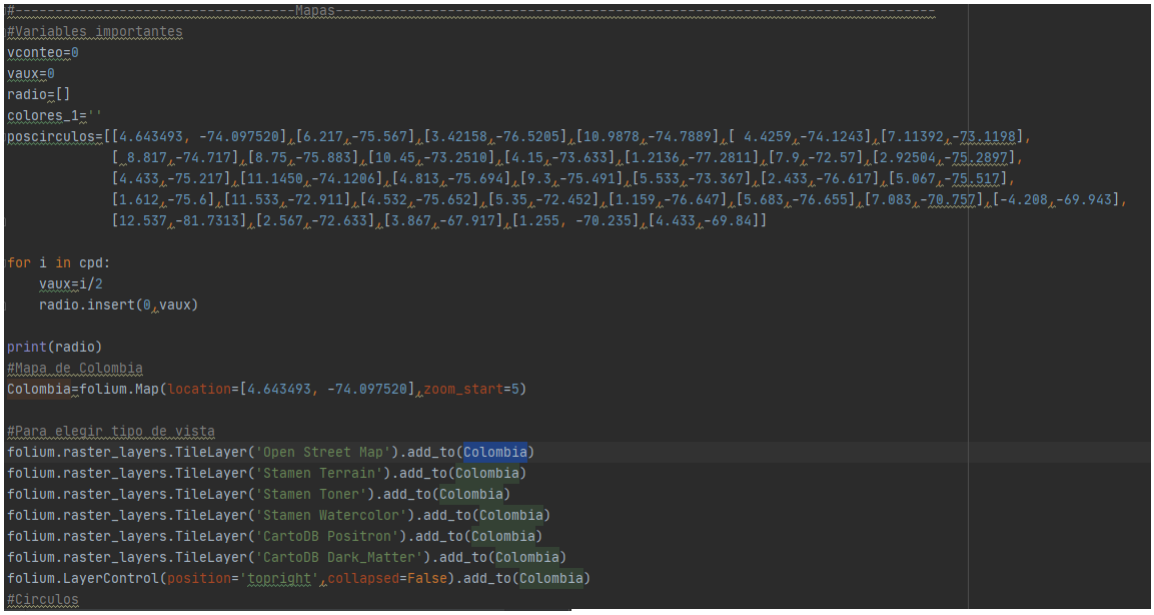


Figura 10: Grafico de torta por departamentos de Colombia

Ahora se realizara la cracion de los mapas, para esto se dejan inicializadas algunas variables que permitiran determinar el radio, el color del mapa de calor, ademas de las latitudes y ubicacion geografica del pais, y de sus respectivas ciudades

El mapa se desarrollara sobre el aplicativo de google maps con el objetivo de poder desarrollarlo sobre en tiempo real y mediante la ubicacion geografica(latitud y longitud) de manera que dados estos datos el mapa de google permita ubicar el pais

Se implementaron distintos tipos de vista sobre ambos el mapa de Colombia esto de manera que permita visualizar el mapa de calor de distintas formas



```
#-------------------------------------Mapas-------------------------------------------------------------------------------------
#Variables importantes
vconteo=0
vaux=0
radio=[]
colores_1=''
poscirculos=[[4.643493, -74.097520],[6.217,-75.567],[3.42158,-76.5205],[10.9878,-74.7889],[ 4.4259,-74.1243],[7.11392,-73.1198],
             [ 8.817,-74.717],[8.75,-75.883],[10.45,-73.2510],[4.15,-73.633],[1.2136,-77.2811],[7.9,-72.57],[2.92504,-75.2897],
             [4.433,-75.217],[11.1450,-74.1206],[4.813,-75.694],[9.3,-75.491],[5.533,-73.367],[2.433,-76.617],[5.067,-75.517],
             [1.612,-75.6],[11.533,-72.911],[4.532,-75.652],[5.35,-72.452],[1.159,-76.647],[5.683,-76.655],[7.083,-70.757],[-4.208,-69.943],
             [12.537,-81.7313],[2.567,-72.633],[3.867,-67.917],[1.255, -70.235],[4.433,-69.84]]

for i in cpd:
    vaux=i/2
    radio.insert(0,vaux)

print(radio)
#Mapa de Colombia
Colombia=folium.Map(location=[4.643493, -74.097520],zoom_start=5)

#Para elegir tipo de vista
folium.raster_layers.TileLayer('Open Street Map').add_to(Colombia)
folium.raster_layers.TileLayer('Stamen Terrain').add_to(Colombia)
folium.raster_layers.TileLayer('Stamen Toner').add_to(Colombia)
folium.raster_layers.TileLayer('Stamen Watercolor').add_to(Colombia)
folium.raster_layers.TileLayer('CartoDB Positron').add_to(Colombia)
folium.raster_layers.TileLayer('CartoDB Dark_Matter').add_to(Colombia)
folium.LayerControl(position='topright',collapsed=False).add_to(Colombia)
#Circulos
```

Figura 11: Ubicacion geografica de Colombia y sus departamentos

De misma forma se determinaran los valores sobre la ubicacion geografica de la ciudad de Bogotá y sus respectivas localidades.

Ahora para el desarrollo de los mapas se tomaran en cuenta la cantidad de poblacion afectada por el virus, para este caso se trabajara sobre el mapa de Colombia, en el cual dada la cnatidad de poblacion afectada se determinara un color para el respectivo departamento. De esta forma con base al departamento ingresado se le asignara su respectivo color ademas del radio del circulo correspondiente al mapa de calor, el cual estara relacionado sobre la cantidad de poblacion contagiada.

```
while(vconteo<33):

    if((radio[vconteo]<150000)and(radio[vconteo]>30000)):
        colores_1='orange'
    elif(radio[vconteo]<30000):
        colores_1='yellow'
    else:
        colores_1='red'

    folium.Circle(radius=radio[vconteo], location=poscirculos[vconteo], color=colores_1, fill=True, fill_color=colores_1).add_to(Colombia)
    vconteo+=1

#Ponomos en el mapa a la izquierda las opciones y guardamos el mapa

Colombia.save("C:\\Users\\sfmer\\Proyectos_SA\\Corte_2\\Colombia.html")
```

Figura 12: Generacion del mapa de calor sobre Colombia

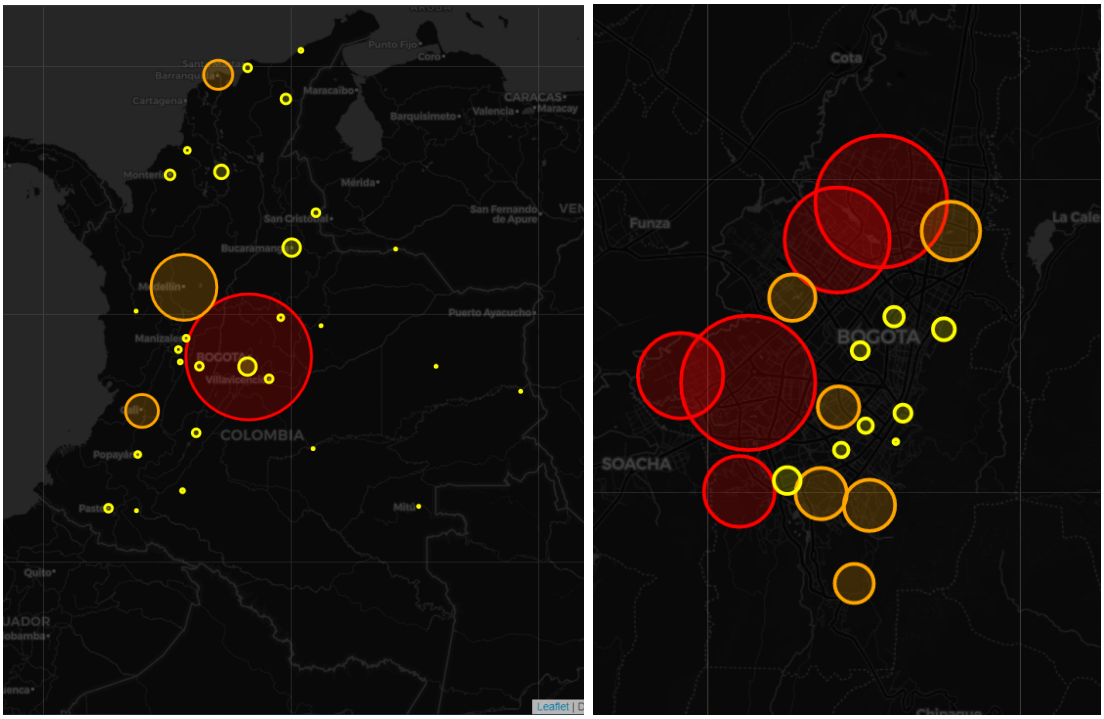


Figura 13: Mapas de calor de Colombia y Bogotá

Para la ultima entrega se realizo la prediccion con redes neurolanes, el objetivo era que a partir de unos datos que fueron obtenidos con respecto a la cantidad de casos en el mes pasado, el algoritmo tenga la capacidad de predecir los casos de Covid en dos meses a futuro. Logrando realizar este procedmiento con machine learnig.

Para ello se debe precargar información relevante al programa para entrenar a la red neuronal y con ello el programa tenga una base real con la que pueda realizar las respectivas predicciones, dicha base de entrenamiento para la red neuronal se adjunta a el proyecto en una archivo con extención .csv, el cual posee un conglomerado de datos que inicial en día 4 de octubre y finalizan el 2 de diciembre.

Dias	Contagios	Fecha
1	867074	4-oct
2	873979	5-oct
3	881085	6-oct
4	888735	7-oct
5	896610	8-oct
6	905106	9-oct
7	913227	10-oct
8	921674	11-oct
9	930243	12-oct
10	938010	13-oct
11	943025	14-oct
12	949086	15-oct
13	955909	16-oct
14	964281	17-oct

Figura 14: Datos de entrenamiento

Con la ayuda de pandas, se lee la base de datos ya antes subida y se procede a entrenar la red neuronal sklear que nos brinda python. Para evitar algun tipo de dato que se pueda llegar a conciderar atipico, solo se guardan las datos a la tupla que cumplan con un rango mayor del 0.95 de entrenamiento de la red neuronal.

```
time=[]
datos = pd.read_csv('bateriaco.csv')
x = datos['fecha']
y = datos['datos']
X = x[:, np.newaxis]

X_train, X_test, y_train, y_test = train_test_split(X, y)

datos = pd.read_csv('bateriaco.csv')
x = datos['fecha']
y = datos['datos']
X = x[:, np.newaxis]

X_train, X_test, y_train, y_test = train_test_split(X, y)

while True:
    from sklearn.model_selection import train_test_split
    X_train, X_test, y_train, y_test = train_test_split(X, y)

    mlr = MLPRegressor(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(3, 3), random_state=1)
    mlr.fit(X_train, y_train)
    print(mlr.score(X_train, y_train))
    if mlr.score(X_train,y_train) > 0.95:
        break

predicciones=[]
aux = 0
for i in range(60, 120):
  print(i)
```

Figura 15: Codigo de redes neuronales

Con base a los datos suministrados lo siguiente sera la creacion de la grafica con base a los datos de la prediccion, donde el tiempo estara ubicado en el eje x y los datos de la prediccion sobre el eje y, esto se trabajara en un tiempo de 2 meses, por lo cual se obtendran un total de 60 nuevos datos, hecho esto los valores los cuales estan almacenados en arreglos seran modelados en una grafica mediante la libreria de matplotlib.

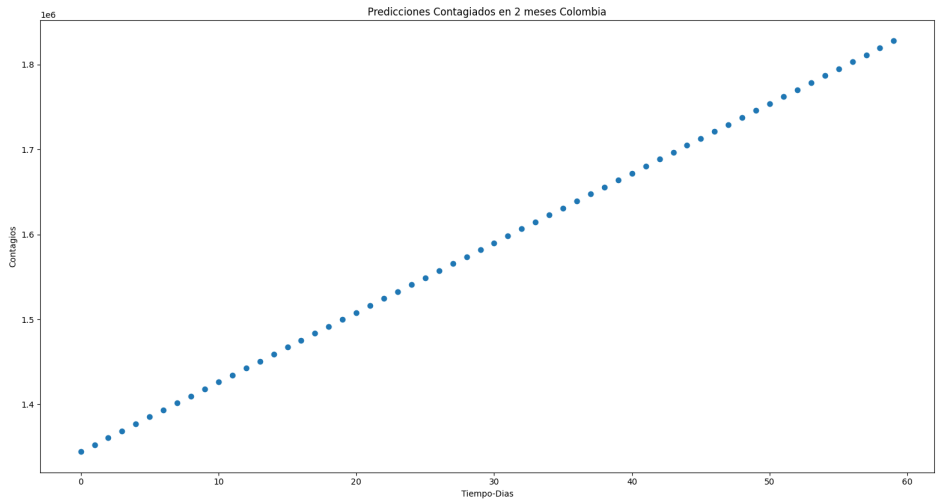


Figura 16: Grafica de predicciones

5. Conclusiones

- Python es un lenguaje sumamente flexible y dinámico, provee una amplia cantidad de herramientas que buscan ofrecer efectividad sobre los procesos, y debido a esto es un lenguaje sumamente fácil de usar y su interacción con

distintos tipos de datos suele ser muy flexible y simple.

- Almacenar los datos en una Base de Datos permite que sean mas accesibles y fáciles de adquirir posteriormente. MySql permite crear estas bases de datos, y facilita la conectividad entre los programas.

- Este proyecto nos dará una visión mas limpia y concreta acerca de los casos de Covid-19 en Colombia, no solo porque trataremos la información sensible, si no que permitirá que otras personas puedan informarse con nuestro programa.

- Mediante las latitudes y longitudes se es posible determinar una ubicacion, con los datos obtenidos estos se pueden ingresar en google maps para asi poder visualizar una zona en concreto.

- Existen distintas formas de poder visualizar un mapa, cada visualizacion corresponde a una distinta paleta de colores, estas herramientas son proporcionadas por las distintas funciones de google maps.
- Concluimos que la red neuronal sklear es bastante util para hacer predicciónes en python.
- Para obtener una gráfica menos lineal y más exponencial, se debe contar con una frecuecia de muestreo menor.
- Se concluye que se necesita primero enseñarle a la maquina para que pueda replicar acciones o predecir datos a futuro. En nuestro caso fue con los datos pasados de casos de Covid-19.
- A pesar de que se le pueda entregar un gran número de datos para el entrenamiento de una red neurona, esta puede predecir datos atipicos, si no se le hace una restriccíon al nivel de entrenamiento de esta.
- Con base a una serie de datos del pasado se es posible mediante un algoritmo de machine learning generar distintas tipos de aproximaciones para cada valor de tiempo determinado.

Referencias

[1] M. Lutz, *Learning Python*, 2013. [Online]. Available: https://cfm.ehu.es/ricardo/docs/python/Learning_Python.pdf

[2] OMS, *Preguntas y respuestas sobre la enfermedad por coronavirus (COVID-19)*, 2020. [Online]. Available: https://www.who.int/es/emergencies/diseases/novel-coronavirus-2019/advice-for-public/q-a-coronaviruses?gclid=CjwKCAjw8MD7BRArEiwAGZsrBSQWuE0uPnTjyhm1ghBMWQywIOcoj5NtEs1lrriQ-ZPT_TbVxjWfDxoCh-sQAvD_BwE

[3] Varios, *Relacion señal-ruido*. [Online]. Available: https://la.mathworks.com/help/signal/ref/snr.html

[4] G. Signal, *Extracting Data from HTML with BeautifulSoup*, 2019. [Online]. Available: https://www.pluralsight.com/guides/extracting-data-html-beautifulsoup

[5] I. Naokil, *PyMySQL*. [Online]. Available: https://github.com/PyMySQL/PyMySQL

[6] Varios, *Pandemia de enfermedad por coronavirus de 2020 en ColombiaL*. [Online]. Available: https://es.wikipedia.org/wiki/Pandemia_de_enfermedad_por_coronavirus_de_2020_en_Colombia

[7] Cctmexico, *Redes Neuronales— Predicción — SciKitLearn — Machine Learning—*. [Online]. Available: https://www.youtube.com/watch?v=9J0Ab4pgvYc&feature=youtu.be