Bitacora de Buzz Lightyear

vf.hurtadodemendozad

December 2019

1. Extraccion de nodos de Open Street Maps

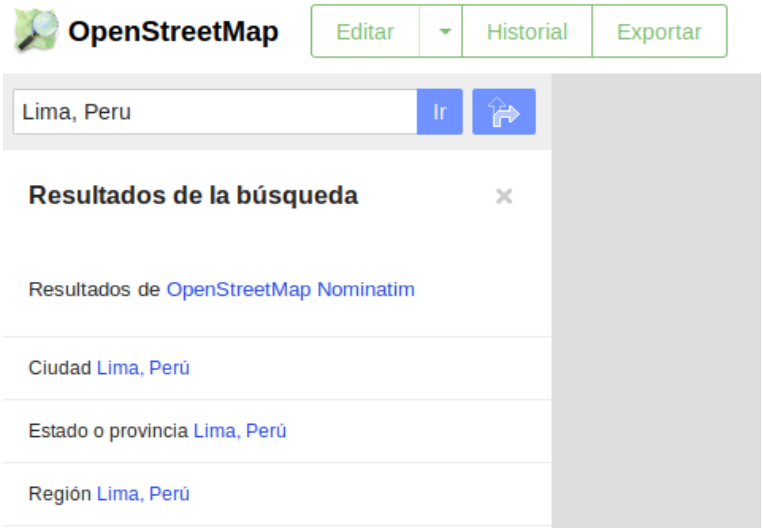
Para poder extraer los datos sobre las redes de las calles usaremos un modulo de python llamado **osmnx**. Para ello, en el terminal ejecutaremos el comando pip install osmnx o pip3 install osmnx (dependiendo de la version de python)
Una vez instalado, procedemos a desarrollar nuestro codigo.

```
import osmnx as ox
place_name = "Lima, Perú"
graph = ox.graph_from_place(place_name,which_result=1,network_type='drive')
```

En la primera linea de codigo, importamos la libreria que utilizaremos, en este caso osmnx. Luego declaramos una variales en la que guardaremos el nombre del lugar que deseamos extraer. Finalmente, llamamos a la funcion **ox.graph_from_place** definiendo los parametros **place_name** (lugar que escogimos), **network_type** como "driveza que lo que buscamos son caminos por los puedan circular vehiculos y finalmente la variable **which_result** que explicaremos posteriormente.
Para poder visualizar el area extraida, utilizaremos la siguiente linea de codigo:

```
fig, ax = ox.plot_graph(graph)
```

Podemos notar que el area capturada es solo Lima distrito, lo cual no es lo que buscamos. Para arreglar este problema, cambiaremos el valor de nuestra **which_result**. Si buscamos Lima, Peru en Open Street Maps veremos que nos muestra distintas opciones que podrian referirse a la palabra clave buscada. Podemos notar que Lima región es la tercera opcion y la que se desea obtener.



Por lo tanto, nuestra variable **which_result** sera finalmente 3.
Ahora, si visualizamos el mapa obtendremos lo siguiente:

```
fig, ax = ox.plot_graph(graph)
```

Despues de extraer el area requerida nos toca conseguir la data de cada nodo y arista de las calles. Para ello usaremos la funcion **ox.graph_to_gdfs(graph)** la cual enviará todos los datos espaciales obtenidos a un dataframe.

```
nodes, edges = ox.graph_to_gdfs(graph)
print(nodes)
```

```
                    y           x       osmid           highway  ref  \
1738539013 -12.035662 -76.988852  1738539013               NaN  NaN
197656584  -12.122907 -77.035870   197656584               NaN  NaN
4335599626 -12.119177 -77.027573  4335599626               NaN  NaN
197656586  -12.123298 -77.036945   197656586               NaN  NaN
5499175423 -11.864833 -77.089699  5499175423               NaN  NaN
...                ...        ...         ...             ...  ...
442761203  -12.163186 -76.990775   442761203  traffic_signals  NaN
6242172919 -11.886359 -77.015505  6242172919               NaN  NaN
1738539004 -12.033600 -76.986739  1738539004               NaN  NaN
1253310462 -11.942150 -77.053062  1253310462               NaN  NaN
386138111  -12.134593 -77.009598   386138111  traffic_signals  NaN

                            geometry
1738539013  POINT (-76.98885 -12.03566)
197656584   POINT (-77.03587 -12.12291)
4335599626  POINT (-77.02757 -12.11918)
197656586   POINT (-77.03695 -12.12330)
5499175423  POINT (-77.08970 -11.86483)
...                               ...
442761203   POINT (-76.99078 -12.16319)
6242172919  POINT (-77.01550 -11.88636)
1738539004  POINT (-76.98674 -12.03360)
1253310462  POINT (-77.05306 -11.94215)
386138111   POINT (-77.00960 -12.13459)

[113422 rows x 6 columns]
```

Dividimos el dataframe en 2 que representaran los nodos y aristas.
Finalmente, guardaremos ambos dataframes en archivos csv para uso posterior.

```
nodes.to_csv("limanodes.csv")
edges.to_csv("limaedges.csv")
```