Clase9_1_GeoPandas

May 20, 2022

1 Seminario de Lenguajes - Python

1.1 Cursada 2022

1.1.1 GeoPandas

2 GeoPandas

- Esta librería agrega soporte para manipular datos geoespaciales.
- Las dos estructuras principales extienden dos clases de pandas:
  - geopandas.GeoDataFrame, subclase de pandas.DataFrame,
  - geopandas.GeoSeries, subclase de pandas.Series,

Estas clases agregan los datos específicos a este tipo de datos, referidos a las geometrías: puntos, polígonos, etc.

- Más info en la documentación oficial

3 Vamos a ver algunas de las funcionalidades a través de un ejemplo: ubicación de rampas en CABA

- Dataset: rampas de accesibilidad - Relevamiento 2016 (CABA) **descarga csv**
- Archivo con la información geográfica de las comunas de CABA **descarga shapefile**

4 Comencemos ...

[76]:
```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
import os.path
import os
```

[77]:
```
dir_datasets = "rampas"
comunas_caba_shp = "comunas.shp"
rampas_relevamiento_2016 = "rampas-de-accesibilidad-relevamiento-2016.csv"

path_datasets = os.path.join(os.getcwd(), dir_datasets)
```

5 ¿Qué información contiene el archivo de información geográfica (comunas.shp)?

[78]:
```
comunas = gpd.read_file(os.path.join(path_datasets, comunas_caba_shp))
```

[79]:
```
type(comunas)
```

[79]:
```
geopandas.geodataframe.GeoDataFrame
```

[82]:
```
comunas.head(3)
```

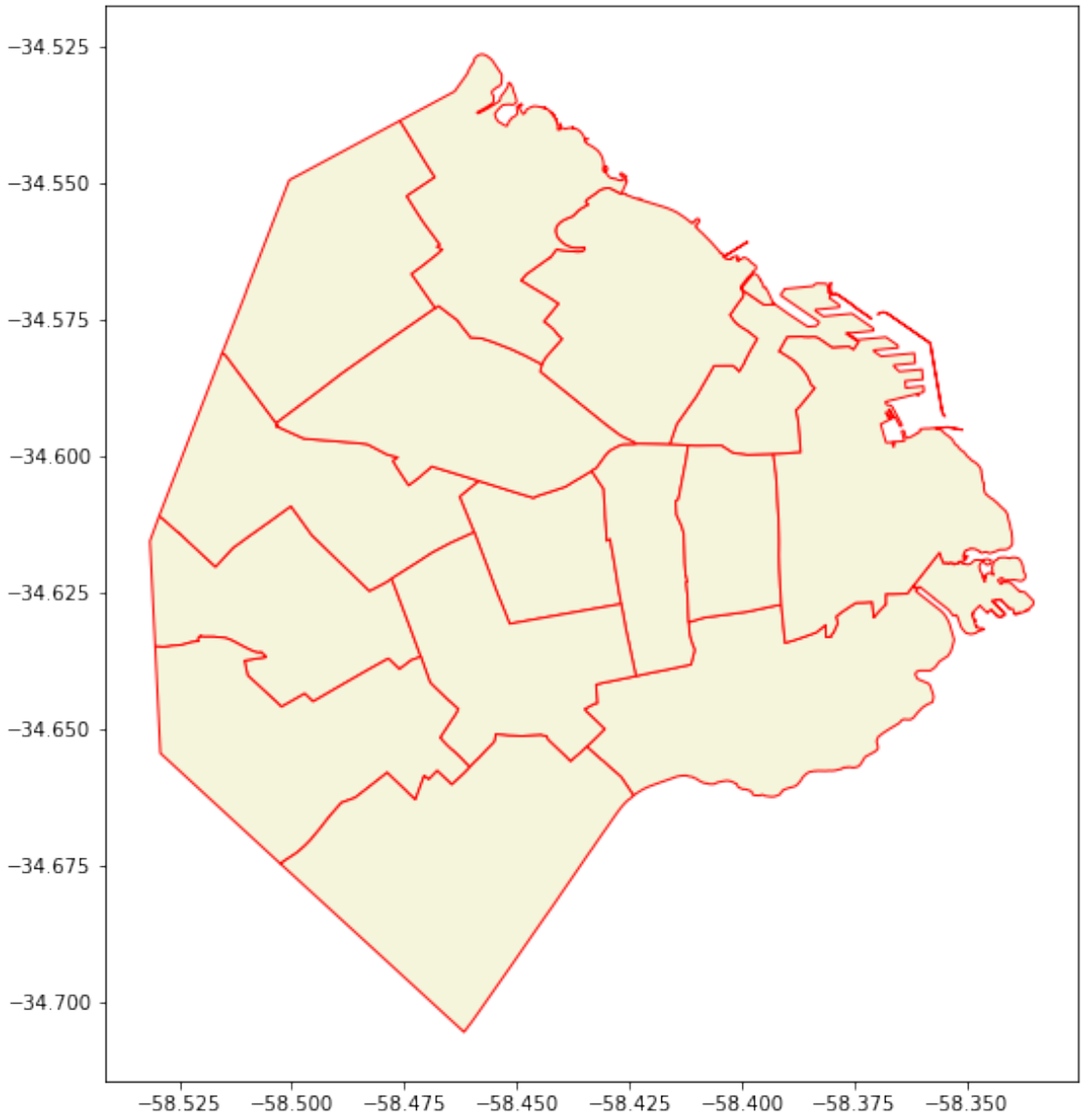
[82]:
```
   ID          OBJETO  COMUNAS  \
0   1  LIMITE COMUNAL      2.0
1   3  LIMITE COMUNAL      6.0
2   6  LIMITE COMUNAL     10.0

                                          BARRIOS     PERIMETRO  \
0                                        RECOLETA  21452.838648
1                                       CABALLITO  10990.964471
2  FLORESTA - MONTE CASTRO - VELEZ SARSFIELD - VE…  18332.037457

           AREA                                           geometry
0  6.317265e+06  POLYGON ((-58.38000 -34.57002, -58.38002 -34.5…
1  6.851029e+06  POLYGON ((-58.43061 -34.60705, -58.43056 -34.6…
2  1.265656e+07  POLYGON ((-58.48834 -34.62016, -58.48820 -34.6…
```

La columna **geometry** contiene la información geográfica de cada una de las comunas como tipo de geometría polígono.

6 Graficamos

[83]:
```
comunas.plot(figsize=(10,10), color='beige',edgecolor='red')
```

[83]:
```
<AxesSubplot:>
```



7 Podemos agregar algunos datos al gráfico

Como podemos observar, el mapa anterior no nos brinda el nombre de cada una de las comunas, para poder subsanar el problema debemos trabajar un poco el GeoDataFrame realizando los siguientes pasos: Crear una nueva columna de tipo **POINT**. Este dato es importante ya que es en este punto, calculado con el método **representative_point()**, donde se va a escribir el nombre de la comuna sobre el mapa.

[84]:
```
comunas['coords'] = comunas['geometry'].apply(lambda x: x.
↪representative_point().coords[:])
```

[85]:
```
comunas['coords']
```

[85]:
```
0       [(-58.39335394881411, -34.58402012780999)]
1       [(-58.44345867055251, -34.61687384741095)]
2      [(-58.503129090909304, -34.62748921603462)]
3       [(-58.50082020876248, -34.60296451410308)]
4       [(-58.49057305772952, -34.56625739628468)]
5       [(-58.42270975961914, -34.57427408391895)]
6       [(-58.4668147137596, -34.590124613219686)]
7      [(-58.45423819580623, -34.554787950559515)]
8      [(-58.391714309443955, -34.63994109590304)]
9     [(-58.420549629232745, -34.618844705165436)]
10      [(-58.44834809730983, -34.63526070464371)]
11      [(-58.40227550909182, -34.61438802928427)]
12     [(-58.496753683144135, -34.65374549776169)]
13    [(-58.468149405683874, -34.678105831485006)]
14      [(-58.36914454898066, -34.60589526999567)]
Name: coords, dtype: object
```

- Dado que el paso anterior nos devuelve una **GeoSerie** debemos quedarnos con el punto generado para cada registro del GeoDataFrame.

[86]:
```
comunas['coords'] = [coords[0] for coords in comunas['coords']]
```

[87]:
```
comunas['coords']
```

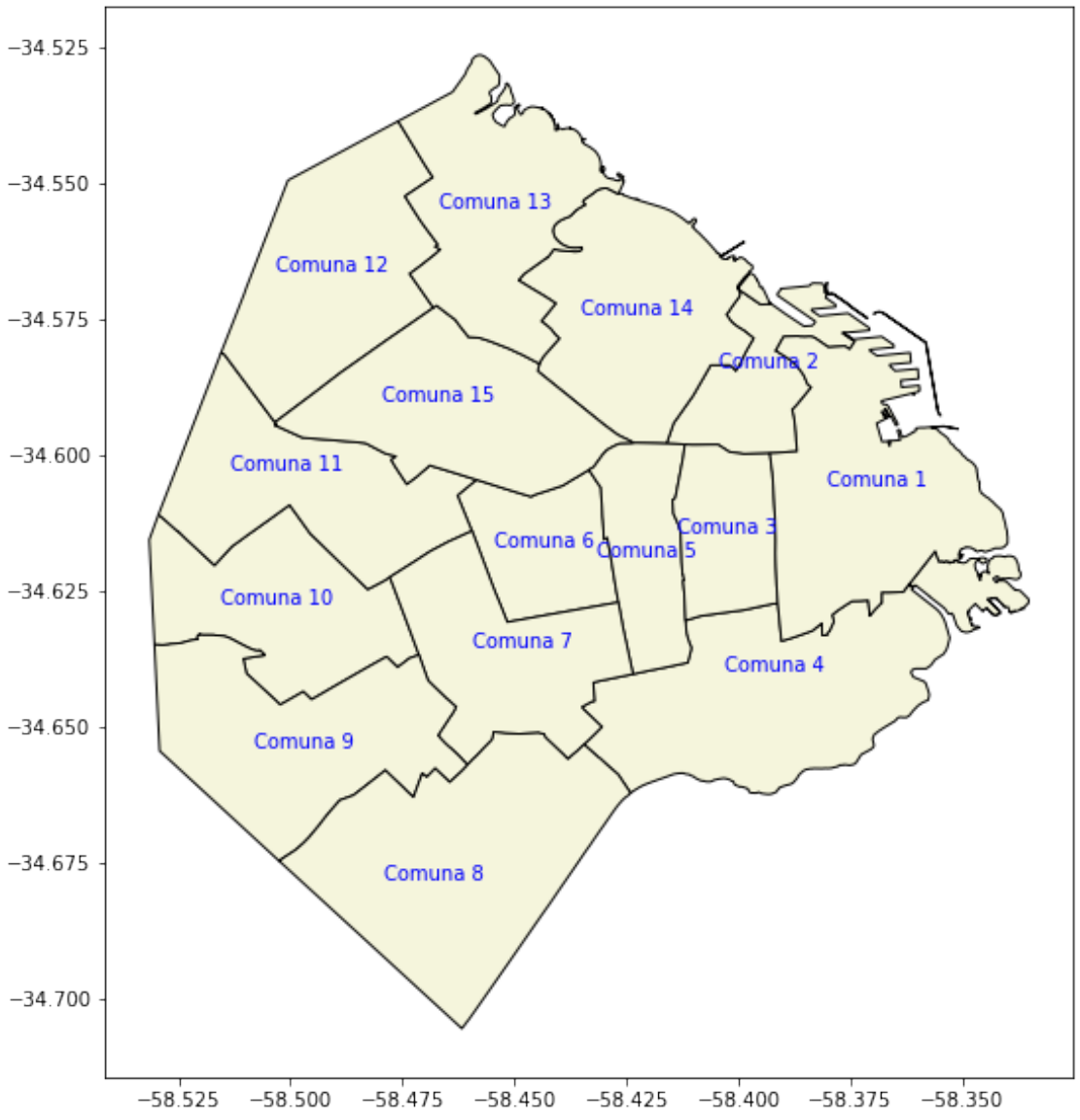
[87]:
```
0       (-58.39335394881411, -34.58402012780999)
1       (-58.44345867055251, -34.61687384741095)
2      (-58.503129090909304, -34.62748921603462)
3       (-58.50082020876248, -34.60296451410308)
4       (-58.49057305772952, -34.56625739628468)
5       (-58.42270975961914, -34.57427408391895)
6       (-58.4668147137596, -34.590124613219686)
7      (-58.45423819580623, -34.554787950559515)
8      (-58.391714309443955, -34.63994109590304)
9     (-58.420549629232745, -34.618844705165436)
10      (-58.44834809730983, -34.63526070464371)
11      (-58.40227550909182, -34.61438802928427)
12     (-58.496753683144135, -34.65374549776169)
13    (-58.468149405683874, -34.678105831485006)
14      (-58.36914454898066, -34.60589526999567)
Name: coords, dtype: object
```

Por último, creamos un campo llamado **label** el cual completamos, para cada registro del GeoDataFrame, con la concatenación del string 'Comuna' más el dato contenido en la columna COMUNAS (es en esta columna donde se encuentra el número de comuna).

[88]:
```
comunas['label'] = comunas.apply(lambda x: 'Comuna ' + str(int(x['COMUNAS'])),␣
↪axis=1)
```

8 Y ahora graficamos

[89]:
```
fig, ax = plt.subplots(figsize = (10,10))
comunas.plot(ax=ax, color='beige', edgecolor='black')
for idx, row in comunas.iterrows():
    plt.annotate(text=row['label'], xy=row['coords'],␣
↪horizontalalignment='center', color='blue')
```



9 Exploremos el dataset de las rampas

[90]:
```
rampas_2016 = pd.read_csv(os.path.join(path_datasets,␣
 ↪rampas_relevamiento_2016), delimiter=';')
```

[91]:
```
rampas_2016
```

[91]:
```
              X          Y    ID        MES    SEMANA   ZONA  \
0    -58.401517 -34.613463     1     AGOSTO    1 AL 6  2 Y 3
1    -58.403174 -34.598833     2     AGOSTO    1 AL 6  2 Y 3
2    -58.394427 -34.618371     3     AGOSTO    1 AL 6  2 Y 3
3    -58.512667 -34.609169     4     AGOSTO    1 AL 6      4
4    -58.510953 -34.609638     5     AGOSTO    1 AL 6      4
…           …          …   …         …        …    …
2950 -58.482653 -34.581977  2997  NOVIEMBRE  21 AL 26     15
2951 -58.481805 -34.581305  2998  NOVIEMBRE  21 AL 26     15
2952 -58.482766 -34.582043  2999  NOVIEMBRE  21 AL 26     15
2953 -58.482867 -34.581925  3000  NOVIEMBRE  21 AL 26     15
2954 -58.482791 -34.581881  3001  NOVIEMBRE  21 AL 26     15

                     CALLE  ALTURA        ESTADO              DOM_NORMA  \
0                  ALBERTI   315.0    FINALIZADO            ALBERTI 315
1               CORDOBA AV  2554.0    FINALIZADO       CORDOBA AV. 2554
2         INDEPENDENCIA AV  1998.0    FINALIZADO  INDEPENDENCIA AV. 1998
3     SIMBRON Y MARCOS PAZ     NaN    FINALIZADO   SIMBRON Y PAZ, MARCOS
4       TINOGASTA Y ZURICH     NaN    FINALIZADO      TINOGASTA Y ZURICH
…                      …     …           …                       …
2950             BALLIVIAN  2896.0  EN EJECUCIÓN          BALLIVIAN 2896
2951             BALLIVIAN  2799.0  EN EJECUCIÓN          BALLIVIAN 2799
2952             BALLIVIAN  2902.0  EN EJECUCIÓN          BALLIVIAN 2902
2953             BALLIVIAN  2901.0  EN EJECUCIÓN          BALLIVIAN 2901
2954             BALLIVIAN  2899.0  EN EJECUCIÓN          BALLIVIAN 2899

                    DOM_GEO
0                315 ALBERTI
1           2554 CORDOBA AV.
2     1998 INDEPENDENCIA AV.
3      SIMBRON & PAZ, MARCOS
4         TINOGASTA & ZURICH
…                        …
2950          2896 BALLIVIAN
2951          2799 BALLIVIAN
2952          2902 BALLIVIAN
2953          2901 BALLIVIAN
2954          2899 BALLIVIAN

[2955 rows x 11 columns]
```

Este archivo contiene la latitud y la longitud de la ubicación en el mapa para cada una de las rampas. Su **sistemas de referencia de coordenadas** es el **EPSG: 4326**.

10 ¿Cómo agregamos los puntos donde se encuentran las rampas?

- Debemos crear las geometrías para cada una de las rampas contenidas en el dataset de rampas_2016.
- Para eso importamos de la librería shapely.geometry las geometría POINT

[92]:
```
from shapely.geometry import Point
geometry = [Point(xy) for xy in zip(rampas_2016["X"], rampas_2016["Y"])]
```

[93]:
```
geometry[:3]
```

[93]:
```
[<shapely.geometry.point.Point at 0x7f9aada17400>,
 <shapely.geometry.point.Point at 0x7f9aada07820>,
 <shapely.geometry.point.Point at 0x7f9aada05f30>]
```

11 Ahora creamos el GeoDataFrame ...

[94]:
```
rampas_2016_gdf = gpd.GeoDataFrame(rampas_2016,
                                   crs='epsg:4326',
                                   geometry=geometry)
```

[95]:
```
rampas_2016_gdf.head()
```

[95]:

	X	Y	ID	MES	SEMANA	ZONA	CALLE	ALTURA	ESTADO	DOM_NORMA	DOM_GEO	geometry
0	-58.401517	-34.613463	1	AGOSTO	1 AL 6	2 Y 3	ALBERTI	315.0	FINALIZADO	ALBERTI 315	315 ALBERTI	POINT (-58.40152 -34.61346)
1	-58.403174	-34.598833	2	AGOSTO	1 AL 6	2 Y 3	CORDOBA AV	2554.0	FINALIZADO	CORDOBA AV. 2554	2554 CORDOBA AV.	POINT (-58.40317 -34.59883)
2	-58.394427	-34.618371	3	AGOSTO	1 AL 6	2 Y 3	INDEPENDENCIA AV	1998.0	FINALIZADO	INDEPENDENCIA AV. 1998	1998 INDEPENDENCIA AV.	POINT (-58.39443 -34.61837)
3	-58.512667	-34.609169	4	AGOSTO	1 AL 6	4	SIMBRON Y MARCOS PAZ	NaN	FINALIZADO	SIMBRON Y PAZ, MARCOS	SIMBRON & PAZ, MARCOS	POINT (-58.51267 -34.60917)
4	-58.510953	-34.609638	5	AGOSTO	1 AL 6	4	TINOGASTA Y ZURICH	NaN	FINALIZADO	TINOGASTA Y ZURICH	TINOGASTA & ZURICH	POINT (-58.51095 -34.60964)

¿Qué diferencias tiene con el dataframe original?

12 El estado de las rampas

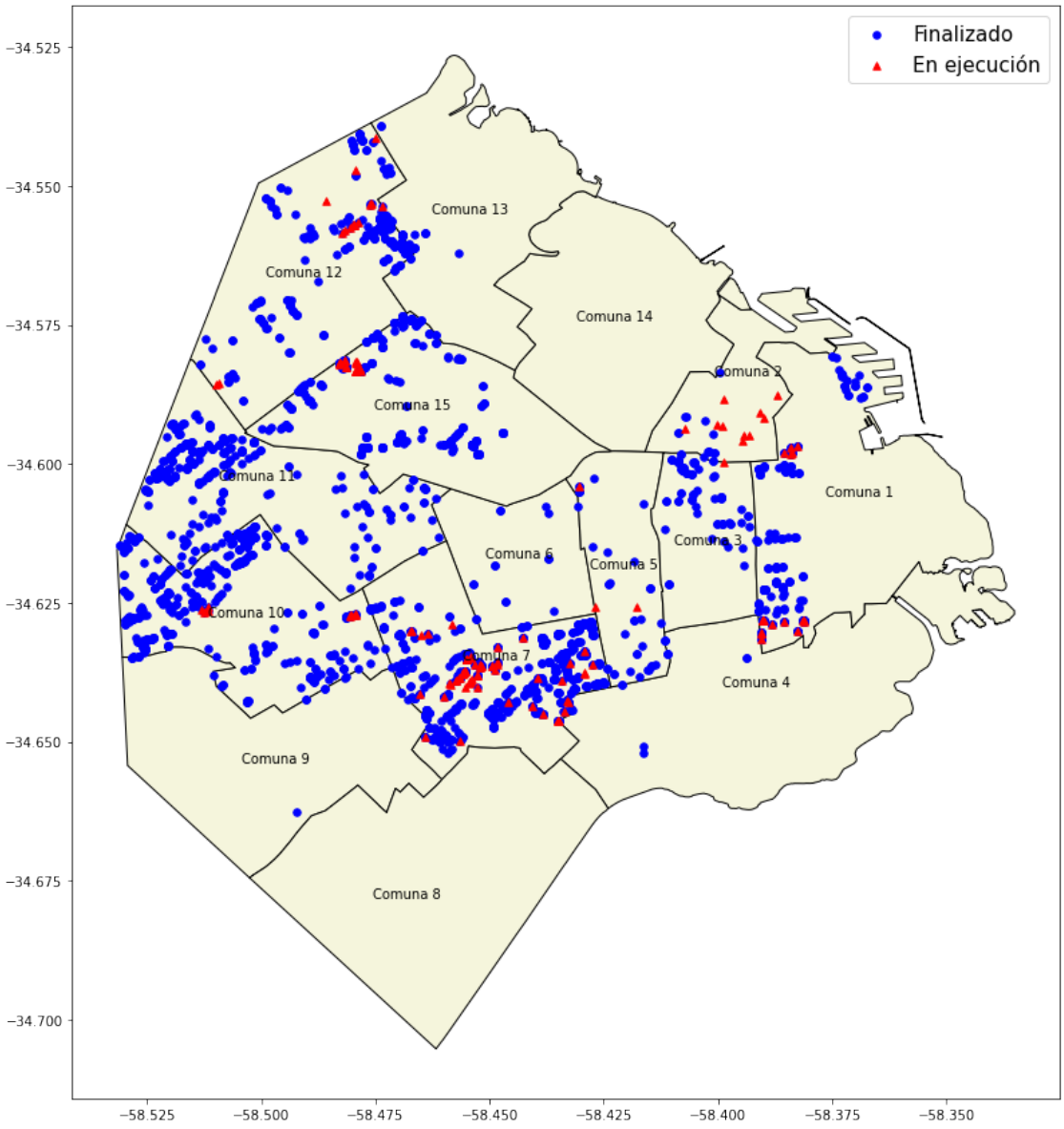
[96]:
```
rampas_2016_gdf['ESTADO'].unique()
```

[96]:
```
array(['FINALIZADO', 'EN EJECUCIÓN'], dtype=object)
```

Generamos el mapa con las marcas correspondientes según el estado. - Tabla de colores: https://matplotlib.org/stable/gallery/color/named_colors.html

[97]:
```
fig, ax = plt.subplots(figsize = (15,15))
comunas.plot(ax = ax, color='beige', edgecolor='black')
for idx, row in comunas.iterrows():
    plt.annotate(text=row['label'], xy=row['coords'],␣
 ↪horizontalalignment='center', color='black')

rampas_2016_gdf[rampas_2016_gdf['ESTADO']=='FINALIZADO'].plot(ax=ax,␣
 ↪markersize=30, color='blue', marker='o', label='Finalizado')
rampas_2016_gdf[rampas_2016_gdf['ESTADO']=='EN EJECUCIÓN'].plot(ax=ax,␣
 ↪markersize=30, color='red', marker='^', label='En ejecución')
plt.legend(prop={'size': 15})
```

[97]:
```
<matplotlib.legend.Legend at 0x7f9ab49d9f60>
```



13 Seguimos la próxima ...