1_TodoMx

June 4, 2020

1 import libraries

[35]:

```
import pandas as pd
import requests
import json
import datetime
import folium
from folium import plugins
import time
import matplotlib
```

2 Read the geo json with the geometric shape of the regions

[2]:

```
# file name, for all the municipalities in Mexico
file_path_geo_json_mx = 'todos_mun_mx.json'

# read
with open(file_path_geo_json_mx) as f:
    geo_json_mx = json.load(f)
```

[3]:

```
# define a new geo json that will contain the clean data
# make an explicit copy of the dictionary
geo_json_mx_clean = dict(geo_json_mx)

# remove the style from the geo json
del geo_json_mx_clean['style']

# more cleaning
for i in range(0,len( geo_json_mx_clean['features'] )):

    # remove style feature in each municipality
    del geo_json_mx_clean['features'][i]['properties']['style']

    # make mun_code a string
    geo_json_mx_clean['features'][i]['properties']['mun_code'] = str(␣
↪geo_json_mx_clean['features'][i]['properties']['mun_code'][0] )
    # add zeroes to the left of the mun_code string if necessary
    geo_json_mx_clean['features'][i]['properties']['mun_code'] = ( 3 - len(␣
↪geo_json_mx_clean['features'][i]['properties']['mun_code'] )␣
↪)*'0'+geo_json_mx_clean['features'][i]['properties']['mun_code']

    # make state_code a string
    geo_json_mx_clean['features'][i]['properties']['state_code'] = str(␣
↪geo_json_mx_clean['features'][i]['properties']['state_code'][0] )
    # add zeroes to the left of the state_code string if necessary
    geo_json_mx_clean['features'][i]['properties']['state_code'] = ( 2 - len(␣
↪geo_json_mx_clean['features'][i]['properties']['state_code'] )␣
↪)*'0'+geo_json_mx_clean['features'][i]['properties']['state_code']

    # add state code to string code
    geo_json_mx_clean['features'][i]['properties']['mun_code'] =␣
↪geo_json_mx_clean['features'][i]['properties']['state_code'] +␣
↪geo_json_mx_clean['features'][i]['properties']['mun_code']
```

[4]:

```
# store cleaned geo json with municipalities
with open('todos_mun_mx_Clean.json', 'w') as f:
    json.dump(geo_json_mx_clean, f)
```

[5]:

```
# list of codes for municipalities
mun_code_list = []
mun_name_list = []

for municipio in geo_json_mx_clean['features']:
    mun_code_list.append( municipio['properties']['mun_code'] )
    mun_name_list.append( municipio['properties']['mun_name'] )

n_mun = len(mun_code_list)
```

[6]:

```
print( 'Number of municipality codes:', len(mun_code_list), 'and are all of the␣
↪codes different?', len(mun_code_list)==len(set(mun_code_list)) )
```

```
Number of municipality codes: 2436 and are all of the codes different? True
```

[8]:

```
# after this section I no longer need file_path_geo_json_mx, geo_json_mx,␣
↪geo_json_mx_clean
# I still need mun_code_list and mun_name_list. Maybe I could use n_mun
```

3 DataFrame to store data from INEGI databank

[9]:
```
df_mx = pd.DataFrame(mun_name_list, columns=['Municipality'],␣
 ↪index=mun_code_list)
df_mx.index.name = 'Code'
df_mx['State Code'] = mun_code_list
df_mx['State Code'] = df_mx['State Code'].apply(lambda x: x[0:2])
#df_mx['State'] = ''
```

[10]:
```
print( 'shape', df_mx.shape )

print( 'dtypes', df_mx.dtypes )

df_mx.head(4)
```

```
shape (2436, 2)
dtypes Municipality    object
State Code      object
dtype: object
```

[10]:
```
                   Municipality State Code
Code
09012                   Tlalpan         09
09013                Xochimilco         09
09008  La Magdalena Contreras         09
09002              Azcapotzalco         09
```

4 Read from INEGI data bank

4.1 my tokens and test urls

[12]:
```
# look at
# 'https://www.inegi.org.mx/app/api/indicadores/desarrolladores/jsonxml/
 ↪INDICATOR/1002000002,1002000003/en/19026/true/BISE/2.0/'+key+'?type=json'
# there, the 19 is for Nuevo León State and the 026 is for Guadalupe␣
 ↪Municipality. I will look through all the municipalities in the text

my_token_banco = 'XXXX'

url_Ini = 'https://www.inegi.org.mx/app/api/indicadores/desarrolladores/jsonxml/
 ↪INDICATOR/'
indicadores = '1002000002,1002000003'
```

4.2 read

[13]:
```
mun_male_list = [-1]*len(df_mx)
mun_fema_list = [-1]*len(df_mx)

num_restantes = len(df_mx)

i_whiles = 0

print('begin---',datetime.datetime.now())

while 0<num_restantes and i_whiles<30:

    for i, mun_code in zip( range( 0, len(df_mx) ), mun_code_list ):

        if mun_male_list[i]==-1 and mun_fema_list[i]==-1:

            resp_tmp = requests.get( url_Ini + indicadores + '/en/' + mun_code␣
 ↪+ '/true/BISE/2.0/'+my_token_banco+'?type=json' )

            if resp_tmp.status_code==200:

                json_tmp = json.loads(resp_tmp.content)

                if (
                    (
                        json_tmp['Series'][0]['INDICADOR']=='1002000002'
                    ) and (
                       ␣
 ↪json_tmp['Series'][0]['OBSERVATIONS'][-1]['TIME_PERIOD']=='2010'
                    ) and (
                        json_tmp['Series'][1]['INDICADOR']=='1002000003'
                    ) and (
                       ␣
 ↪json_tmp['Series'][1]['OBSERVATIONS'][-1]['TIME_PERIOD']=='2010'
                    )
                ):

                    mun_male_list[i] = int(float(␣
 ↪json_tmp['Series'][0]['OBSERVATIONS'][-1]['OBS_VALUE'] ))
                    mun_fema_list[i] = int(float(␣
 ↪json_tmp['Series'][1]['OBSERVATIONS'][-1]['OBS_VALUE'] ))

                    num_restantes = num_restantes-1

                    time.sleep(0.2) # wait time
                else:

                    print('json loads but data does not satisfy INDICATOR and␣
 ↪TIME_PERIOD conditions', mun_code)

    i_whiles = i_whiles+1
    print(
        str(i_whiles) + ' while loops so far and ' + str(num_restantes) + '␣
 ↪municipalities remaining',
        datetime.datetime.now()
    )

print('end-----',datetime.datetime.now())
print( str(i_whiles) + ' while loops and ' + str(num_restantes) + '␣
 ↪municipalities not yet read')
```

```
begin--- 2020-05-28 09:51:14.189718
1 while loops so far and 1440 municipalities remaining 2020-05-28
10:18:36.041099
2 while loops so far and 891 municipalities remaining 2020-05-28 10:34:22.854279
3 while loops so far and 212 municipalities remaining 2020-05-28 10:48:08.194807
4 while loops so far and 105 municipalities remaining 2020-05-28 10:50:50.193033
5 while loops so far and 50 municipalities remaining 2020-05-28 10:52:14.265752
6 while loops so far and 26 municipalities remaining 2020-05-28 10:52:53.588387
7 while loops so far and 13 municipalities remaining 2020-05-28 10:53:12.311437
8 while loops so far and 6 municipalities remaining 2020-05-28 10:53:21.441428
9 while loops so far and 3 municipalities remaining 2020-05-28 10:53:25.578891
10 while loops so far and 2 municipalities remaining 2020-05-28 10:53:27.824086
11 while loops so far and 1 municipalities remaining 2020-05-28 10:53:29.565909
12 while loops so far and 1 municipalities remaining 2020-05-28 10:53:29.851412
13 while loops so far and 1 municipalities remaining 2020-05-28 10:53:30.147946
14 while loops so far and 0 municipalities remaining 2020-05-28 10:53:31.171437
end----- 2020-05-28 10:53:31.171972
14 while loops and 0 municipalities not yet read
```

[14]:

```
# after this I no longer need resp_tmp, json_tmp
```

[15]:

```
df_mx['Males'] = mun_male_list
df_mx['Females'] = mun_fema_list
df_mx['People'] = df_mx['Males'] + df_mx['Females']
df_mx['Asymmetry'] = (df_mx['Males'] - df_mx['Females'])/(df_mx['Males'] +␣
 ↪df_mx['Females'])
```

[16]:

```
df_mx = df_mx[['Municipality','People','Males','Females','Asymmetry','State␣
 ↪Code']]
```

[17]:

```
print(df_mx.shape)
print(df_mx.dtypes)
```

```
(2436, 6)
Municipality     object
People            int64
Males             int64
Females           int64
Asymmetry       float64
State Code       object
dtype: object
```

[22]:

```
#df_mx.iloc[0]
df_mx.head(20)
```

[22]:

```
                   Municipality   People   Males  Females  Asymmetry State Code
Code
09012                   Tlalpan   650567  312139   338428  -0.040409         09
09013                Xochimilco   415007  205305   209702  -0.010595         09
09008  La Magdalena Contreras     239086  114492   124594  -0.042253         09
09002              Azcapotzalco   414711  196053   218658  -0.054508         09
09014             Benito Juárez   385439  176410   209029  -0.084628         09
09015                Cuauhtémoc   531831  251725   280106  -0.053365         09
09010            Álvaro Obregón   727034  346041   380993  -0.048075         09
09005        Gustavo A. Madero   1185772  571233   614539  -0.036521         09
09004     Cuajimalpa de Morelos   186391   88642    97749  -0.048860         09
09016            Miguel Hidalgo   372889  172667   200222  -0.073896         09
09006                Iztacalco   384326  182534   201792  -0.050109         09
09003                  Coyoacán   620416  292491   327925  -0.057113         09
09017      Venustiano Carranza    430978  203651   227327  -0.054936         09
09011                   Tláhuac   360265  175210   185055  -0.027327         09
09007                Iztapalapa  1815786  880998   934788  -0.029624         09
09009                Milpa Alta   130582   64192    66390  -0.016832         09
12006                   Apaxtla    12389    5949     6440  -0.039632         12
12017                    Cocula    14707    7060     7647  -0.039913         12
12026     Cuetzala del Progreso     9166    4443     4723  -0.030548         12
12028        Chilapa de Álvarez   120790   57940    62850  -0.040649         12
```

4.3 store

[23]:

```
# store csv with municipalities
df_mx.to_csv('datos_municipios.csv')
```

[24]:

```
# after this I no longer need df_mx, just the path of the csv with the dataframe
```

4.4 read and clean stored csv

[25]:

```
# read new csv with municipalities
df_mx2 = pd.read_csv('datos_municipios.csv')

# convert municipality code to string
df_mx2[['Code']] = df_mx2[['Code']].astype(str)

# add zeros to the left if necessary
# it is important to distinguish between apply and apply map
df_mx2[['Code']] = df_mx2[['Code']].applymap(
    lambda x: ((5-len(x))*'0')+x
)

# convert state code code to string
df_mx2[['State Code']] = df_mx2[['State Code']].astype(str)

# add zeros to the left if necessary
# it is important to distinguish between apply and apply map
df_mx2[['State Code']] = df_mx2[['State Code']].applymap(
    lambda x: ((2-len(x))*'0')+x
)

# set 'Code' as the index of the dataframe
df_mx2.set_index('Code',inplace=True)
```

[26]:

```
df_mx2.head(20)
```

[26]:

```
                 Municipality   People   Males  Females  Asymmetry State Code
Code
09012                 Tlalpan   650567  312139   338428  -0.040409         09
09013              Xochimilco   415007  205305   209702  -0.010595         09
09008  La Magdalena Contreras   239086  114492   124594  -0.042253         09
09002            Azcapotzalco   414711  196053   218658  -0.054508         09
09014           Benito Juárez   385439  176410   209029  -0.084628         09
09015              Cuauhtémoc   531831  251725   280106  -0.053365         09
09010          Álvaro Obregón   727034  346041   380993  -0.048075         09
09005       Gustavo A. Madero  1185772  571233   614539  -0.036521         09
09004    Cuajimalpa de Morelos   186391   88642    97749  -0.048860         09
09016          Miguel Hidalgo   372889  172667   200222  -0.073896         09
09006              Iztacalco   384326  182534   201792  -0.050109         09
09003               Coyoacán   620416  292491   327925  -0.057113         09
09017     Venustiano Carranza   430978  203651   227327  -0.054936         09
09011                 Tláhuac   360265  175210   185055  -0.027327         09
09007              Iztapalapa  1815786  880998   934788  -0.029624         09
09009             Milpa Alta   130582   64192    66390  -0.016832         09
12006                 Apaxtla    12389    5949     6440  -0.039632         12
```

```
12017                 Cocula    14707    7060     7647  -0.039913         12
12026   Cuetzala del Progreso     9166    4443     4723  -0.030548         12
12028      Chilapa de Álvarez   120790   57940    62850  -0.040649         12
```

[27]:
```python
print(df_mx2.shape)
print(df_mx2.dtypes)
#list(df_mx2.index.values)
```

```
(2436, 6)
Municipality     object
People            int64
Males             int64
Females           int64
Asymmetry       float64
State Code       object
dtype: object
```

[28]:
```python
# after storing the original dataframe, df_mx, in a cvs file, I no longer need.␣
 ↪I just read the csv again and clean it
```

5 Colors

[29]:
```python
# this uses
# see https://www.codespeedy.com/convert-rgb-to-hex-color-code-in-python/
# import matplotlib
```

[30]:
```python
# color function red for -max of the abs of the values, blue for +max of the␣
 ↪abs of the values
def color_rwb(x,xMax,xMin):
    xAbsMax = max( abs(xMax), abs(xMin) )

    if 0<=x:
        xCol = matplotlib.colors.to_hex(
            [
                1-x/xAbsMax,
                1-x/xAbsMax,
                1
            ]
        )

    if x<0:
        xCol = matplotlib.colors.to_hex(
            [
                1,
                1+x/xAbsMax,
                1+x/xAbsMax
            ]
        )

    return xCol
```

[31]:

```
#import branca # works
```

[32]:

```
#import geopandas # works
```

6 Make map

[36]:

```
latCen_mx = +023.00
lonCen_mx = -100.00

# define a map
map_mx = folium.Map(location=[latCen_mx, lonCen_mx], width=970, height=600,␣
 ↪zoom_start=5, min_zoom=5, max_zoom=10)

max_Asymm = df_mx2['Asymmetry'].max()
min_Asymm = df_mx2['Asymmetry'].min()
```

[37]:

```
# see
# https://github.com/python-visualization/folium/blob/master/examples/
 ↪GeoJSON_and_choropleth.ipynb

# nuevo intento con la asimetría
folium.GeoJson(
    r'todos_mun_mx_Clean.json',
    style_function=lambda feature: {
        'fillColor': color_rwb( float(df_mx2['Asymmetry'].
 ↪loc[feature['properties']['mun_code']]), max_Asymm, min_Asymm ) if␣
 ↪feature['properties']['mun_code'] in list(df_mx2.index.values) else 'black',
        'fillOpacity': 0.6,
        'color': 'black',
        'weight': 1
    }
).add_to(map_mx)
```

[37]:

```
<folium.features.GeoJson at 0x7f09a0b20c90>
```

[38]:

```
map_mx
```

[38]:

```
<folium.folium.Map at 0x7f09a0b12a90>
```

[39]:

```
# see https://github.com/python-visualization/folium/issues/35

import os
import subprocess
outdir = 'screenshots' # this directory has to exist..
map_mx.save('1_TodoMx_map.html')
url = 'file://{}/tmp.html'.format(os.getcwd())
outfn = os.path.join(outdir,'outfig.png')
subprocess.check_call(['cutycapt','--url={}'.format(url), '--out={}'.
 ↪format(outfn)])

#map_mx.create_map('1_TodoMx_Map.html')
#map.create_map('1_TodoMx_Map.html')
```

```
      ␣
↪---------------------------------------------------------------------------

       FileNotFoundError                         Traceback (most recent call␣
↪last)

       <ipython-input-39-32c84ac784aa> in <module>
         7 url = 'file://{}/tmp.html'.format(os.getcwd())
         8 outfn = os.path.join(outdir,'outfig.png')
   ----> 9 subprocess.check_call(['cutycapt','--url={}'.format(url), '--out={}'.
↪format(outfn)])
        10
        11 #map_mx.create_map('1_TodoMx_Map.html')


       ~/anaconda3/lib/python3.7/subprocess.py in check_call(*popenargs,␣
↪**kwargs)
       356     check_call(["ls", "-l"])
       357     """
   --> 358     retcode = call(*popenargs, **kwargs)
       359     if retcode:
       360         cmd = kwargs.get("args")


       ~/anaconda3/lib/python3.7/subprocess.py in call(timeout, *popenargs,␣
↪**kwargs)
       337     retcode = call(["ls", "-l"])
       338     """
   --> 339     with Popen(*popenargs, **kwargs) as p:
       340         try:
       341             return p.wait(timeout=timeout)
       ~/anaconda3/lib/python3.7/subprocess.py in __init__(self, args, bufsize,␣
↪executable, stdin, stdout, stderr, preexec_fn, close_fds, shell, cwd, env,␣
↪universal_newlines, startupinfo, creationflags, restore_signals,␣
↪start_new_session, pass_fds, encoding, errors, text)
       798                                 c2pread, c2pwrite,
       799                                 errread, errwrite,
   --> 800                                 restore_signals, start_new_session)
       801         except:
       802             # Cleanup if the child failed starting.


       ~/anaconda3/lib/python3.7/subprocess.py in _execute_child(self, args,␣
↪executable, preexec_fn, close_fds, pass_fds, cwd, env, startupinfo,␣
↪creationflags, shell, p2cread, p2cwrite, c2pread, c2pwrite, errread, errwrite,␣
↪restore_signals, start_new_session)
      1549                             if errno_num == errno.ENOENT:
      1550                                 err_msg += ': ' + repr(err_filename)
   -> 1551                         raise child_exception_type(errno_num, err_msg,␣
↪err_filename)
      1552                     raise child_exception_type(err_msg)
      1553


       FileNotFoundError: [Errno 2] No such file or directory: 'cutycapt':␣
↪'cutycapt'
```