03_Calculate_contextual_characters

March 27, 2020

1 Measure contextual morphometric characters

Computational notebook 03 for *NAME* Dal Cin, Fleischmann.. - ADD Reference.

Contact: martin@martinfleischmann.net

Date: 27/03/2020

This notebook measure contextual (code uses older term summative) characters. It requires data from `02_Measure_morphometric_characters.ipynb` and additional manually assigned attributes:

- Attribute `part` in `name_blg` for cases which were divided into parts. Each part should be marked by unique `int`.
- Attribute `case` in `name_str` capturing which LineStrings form the seashore street itself. (1 - True)

Structure of GeoPackages:

```
./data/
    atlantic.gpkg
        name_blg    - Polygon layers
        name_str    - LineString layers
        name_case   - Polygon layers
        name_tess   - Polygon layers
        name_blocks - Polygon layers
        ...
     preatl.gpkg
        name_blg
        name_str
        name_case
        ...
     premed.gpkg
        name_blg
        name_str
        name_case
        ...
     med.gpkg
        name_blg
        name_str
        name_case
        ...
```

CRS of the original data is EPSG:3763.

```
<Projected CRS: EPSG:3763>
Name: ETRS89 / Portugal TM06
Axis Info [cartesian]:
- X[east]: Easting (metre)
- Y[north]: Northing (metre)
Area of Use:
- name: Portugal - mainland - onshore
- bounds: (-9.56, 36.95, -6.19, 42.16)
Coordinate Operation:
- name: Portugual TM06
- method: Transverse Mercator
Datum: European Terrestrial Reference System 1989
- Ellipsoid: GRS 1980
- Prime Meridian: Greenwich
```

[3]:
```
import geopandas as gpd
import numpy as np
import scipy as sp
import momepy as mm
import pandas as pd
import fiona
import inequality
from inequality.theil import Theil
```

[4]:
```
fiona.__version__, gpd.__version__, mm.__version__, sp.__version__, np.
 ↪__version__, pd.__version__, inequality.__version__
```

[4]:
```
('1.8.13', '0.7.0', '0.1.1', '1.4.1', '1.18.1', '1.0.3', '1.0.0')
```

[ ]:
```
folder = 'data/'
```

[ ]:
```
summative = pd.DataFrame()
```

[ ]:
```
parts = ['atlantic', 'preatl', 'premed', 'med']
for part in parts:
    path = folder + part + '.gpkg'
    layers = [x[:-4] for x in fiona.listlayers(path) if 'blg' in x]
    for l in layers:
        buildings = gpd.read_file(path, layer=l + '_blg')
        edges = gpd.read_file(path, layer=l + '_str')
        tessellation = gpd.read_file(path, layer=l + '_tess')
        blocks = gpd.read_file(path, layer=l + '_blocks')
        buildings = buildings.merge(edges.drop(columns='geometry'), on='nID',␣
↪how='left')
        buildings = buildings.merge(tessellation.drop(columns=['bID',␣
↪'geometry', 'nID']), on='uID', how='left')
        data = buildings.merge(blocks.drop(columns='geometry'), on='bID',␣
↪how='left')

        to_summ = ['sdbAre', 'sdbPer', 'ssbCCo', 'ssbCor', 'ssbSqu', 'ssbERI',
                   'ssbElo', 'ssbCCD', 'stbCeA', 'mtbSWR', 'mtbAli', 'mtbNDi',␣
↪'ldbPWL',
                   'stbSAl', 'ltcBuA', 'sssLin', 'sdsSPW', 'stsOpe', 'svsSDe',␣
↪'sdsAre', 'sdsBAr', 'sisBpM',
                   'sdcLAL', 'sdcAre', 'sscERI', 'sicCAR', 'stcSAl', 'ldkAre',␣
↪'lskElo', 'likGra', 'meshedness',
                   ]
        spec = ['sdsLen']

        if 'part' in data.columns:
            for part in set(data.part):
                subset = data.loc[data.part == part]
                for col in to_summ:
                    values = subset[col]
                    values_IQ = mm.limit_range(values, rng=(25, 75))
                    values_ID = mm.limit_range(values, rng=(10, 90))

                    summative.loc[l + str(part), col + '_meanIQ'] = np.
↪mean(values_IQ)
                    summative.loc[l + str(part), col + '_rangeIQ'] = sp.stats.
↪iqr(values)
                    summative.loc[l + str(part), col + '_TheilID'] =␣
↪Theil(values_ID).T
                for col in spec:
                    values = subset.loc[subset.case == 1][col]
                    values_IQ = mm.limit_range(values, rng=(25, 75))
                    values_ID = mm.limit_range(values, rng=(10, 90))

                    summative.loc[l + str(part), col + '_meanIQ'] = np.
↪mean(values_IQ)
                    summative.loc[l + str(part), col + '_rangeIQ'] = sp.stats.
↪iqr(values)
                    summative.loc[l + str(part), col + '_TheilID'] =␣
↪Theil(values_ID).T

        else:
            for col in to_summ:
                values = data[col]
                values_IQ = mm.limit_range(values, rng=(25, 75))
                values_ID = mm.limit_range(values, rng=(10, 90))

                summative.loc[l, col + '_meanIQ'] = np.mean(values_IQ)
                summative.loc[l, col + '_rangeIQ'] = sp.stats.iqr(values)
                summative.loc[l, col + '_TheilID'] = Theil(values_ID).T

            for col in spec:
                values = data.loc[data.case == 1][col]
                values_IQ = mm.limit_range(values, rng=(25, 75))
                values_ID = mm.limit_range(values, rng=(10, 90))

                summative.loc[l, col + '_meanIQ'] = np.mean(values_IQ)
                summative.loc[l, col + '_rangeIQ'] = sp.stats.iqr(values)
                summative.loc[l, col + '_TheilID'] = Theil(values_ID).T
```

```
[ ]: summative.to_csv('data/summative_data.csv')
```