Bandwidth_visualization

July 19, 2020

In [ ]:
```
#@title Python imports
# A bit of imports
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings('ignore')

import numpy as np
import pandas as pd
import ipywidgets as widgets
from ipywidgets import interact, interact_manual
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
from scipy import stats
import geopandas as gpd
from shapely.geometry import Point, Polygon
%matplotlib inline
sns.set(color_codes=True)
from sklearn import linear_model
from matplotlib.colors import LinearSegmentedColormap
from matplotlib_scalebar.scalebar import ScaleBar
import statsmodels.api as statm

import libpysal as ps
from mgwr.gwr import GWR
from mgwr.gwr import MGWR
from mgwr.sel_bw import Sel_BW
from spglm.family import Gaussian, Binomial, Poisson
import multiprocessing as mp
pool = mp.Pool()
import io
```

In [23]:
```
df = pd.read_csv("example_dataset.csv")
```

In [24]:
```
df.columns
```

```
Out[24]: Index(['the_geom', 'longitude', 'the_geom_webmercator', 'avg_tech', 'latitude',
                'houses', 'avg_unemp', 'cartodb_id', 'avg_index', 'avg_price',
                'avg_basement', 'avg_sqft', 'avg_water_dist', 'avg_age', 'ind'],
               dtype='object')

In [26]: df['ln_price']=np.log(df['avg_price'])

In [29]: df['ln_price']

Out[29]: 0      12.611636
         1      12.312879
         2      12.563657
         3      12.715962
         4      12.619820
                  ...
         354    12.936781
         355    12.733797
         356    12.839368
         357    13.007454
         358    13.426880
         Name: ln_price, Length: 359, dtype: float64

In [40]: df=df.dropna()

In [42]: import statsmodels.api as statm
         X=df[['avg_tech','avg_unemp','avg_index','avg_sqft','avg_basement','avg_water_dist','a
         X_std = (X-X.mean(axis=0))/X.std(axis=0)
         X_std=statm.add_constant(X_std)
         y=df['ln_price']
         y_std = (y-y.mean(axis=0))/y.std(axis=0)
         model = statm.OLS(y_std,X_std).fit()
         predictions=model.predict(X_std)

         model.summary()

Out[42]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                     OLS Regression Results
         ==============================================================================
         Dep. Variable:               ln_price   R-squared:                       0.898
         Model:                            OLS   Adj. R-squared:                  0.896
         Method:                 Least Squares   F-statistic:                     438.4
         Date:                Wed, 20 May 2020   Prob (F-statistic):          1.15e-168
         Time:                        12:47:31   Log-Likelihood:                -98.782
         No. Observations:                 357   AIC:                             213.6
         Df Residuals:                     349   BIC:                             244.6
         Df Model:                           7
         Covariance Type:            nonrobust
         ==============================================================================
```

```
                    coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          6.721e-15      0.017   3.93e-13      1.000      -0.034       0.034
avg_tech          0.2186      0.029      7.433      0.000       0.161       0.276
avg_unemp        -0.2990      0.028    -10.802      0.000      -0.353      -0.245
avg_index        -0.1231      0.030     -4.115      0.000      -0.182      -0.064
avg_sqft          0.5865      0.023     25.043      0.000       0.540       0.633
avg_basement      0.0984      0.023      4.212      0.000       0.052       0.144
avg_water_dist   -0.2106      0.034     -6.280      0.000      -0.277      -0.145
avg_age           0.1563      0.025      6.371      0.000       0.108       0.205
==============================================================================
Omnibus:                       17.936   Durbin-Watson:                   1.793
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               44.569
Skew:                           0.139   Prob(JB):                     2.10e-10
Kurtosis:                       4.708   Cond. No.                         4.47
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spec
"""
```

```
In [43]: import libpysal as ps
         from mgwr.gwr import GWR
         from mgwr.gwr import MGWR
         from mgwr.sel_bw import Sel_BW
         from spglm.family import Gaussian, Binomial, Poisson
         import multiprocessing as mp
         pool = mp.Pool()
```

```
In [44]: df.columns
```

```
Out[44]: Index(['the_geom', 'longitude', 'the_geom_webmercator', 'avg_tech', 'latitude',
                'houses', 'avg_unemp', 'cartodb_id', 'avg_index', 'avg_price',
                'avg_basement', 'avg_sqft', 'avg_water_dist', 'avg_age', 'ind',
                'ln_price'],
               dtype='object')
```

```
In [45]: coords = np.array(list(zip(df['longitude'],df['latitude'])))
         y = np.array(df['ln_price']).reshape((-1,1))
         y_std = (y-y.mean(axis=0))/y.std(axis=0)
         X=df[['avg_tech','avg_unemp','avg_index','avg_sqft','avg_basement','avg_water_dist','a
         X_std=(X-X.mean(axis=0))/X.std(axis=0)
         selector_gwr = Sel_BW(coords, y_std, X_std)
```

```
In [48]: selector_mgwr = Sel_BW(coords, y_std, X_std, multi=True)
```

```
In [49]: selector_mgwr.search(pool=pool)
```

```
Out[49]: array([ 43., 164., 226., 191.,  45.,  81.,  45., 354.])
```

```
In [50]: %%time
         model_mgwr = MGWR(coords,y_std,X_std,selector_mgwr,fixed=False,kernel='bisquare',sigma
         results_mgwr=model_mgwr.fit()

HBox(children=(IntProgress(value=0, description='Inference', max=1), HTML(value='')))



Wall time: 7.64 s


In [51]: results_mgwr.R2

Out[51]: 0.9775393617728864

In [147]: df2 = pd.read_csv("MGWR_session_results.csv")

In [148]: df2['w_43']=results_mgwr.W[0][171]
          df2['w_164']=results_mgwr.W[1][171]
          df2['w_226']=results_mgwr.W[2][171]
          df2['w_81']=results_mgwr.W[5][171]
          df2['w_354']=results_mgwr.W[7][171]

In [149]: df2.to_csv("MGWR_session_results.csv")
```