Chapter 8 - Comparison of taxa and land use

November 10, 2020

[1]:
```
import pandas as pd
import geopandas as gpd
import seaborn as sns
import matplotlib.pyplot as plt
import husl
from random import shuffle
from legendgram import legendgram
import mapclassify
from matplotlib_scalebar.scalebar import ScaleBar
from matplotlib.colors import ListedColormap
from tqdm import tqdm
```

[2]:
```
clusters = pd.read_csv('/Users/martin/Dropbox/Academia/Data/Geo/Prague/
↪Clustering/complete data/200218_clusters_complete_n20.csv', index_col=0)
```

[3]:
```
landuse = pd.read_csv('/Users/martin/Dropbox/Academia/Data/Geo/Prague/
↪Validation/land_use.csv', index_col=0)
```

[5]:
```
landuse.KOD.unique().shape
```

[5]: (123,)

[4]:
```
joined = clusters.merge(landuse[['uID', 'KOD']], on='uID', how='left')
```

[5]:
```
joined.head(4)
```

[5]:

	uID	cluster	KOD
0	0	1	PRR
1	1	11	BD
2	2	13	BRR
3	3	18	BRR

[5]:
```
counts = joined.KOD.value_counts()
```

[49]:
```
counts[counts>1000].index
```

```
[49]: Index(['BRR', 'BD', 'ZHB', 'BRV', 'PND', 'SLK', 'SAM', 'PRR', 'BQ', 'SLU',
        'RAZ', 'XO', 'PRS', 'SQ', 'ZHV'],
       dtype='object')
```

Landuse is not a great resource, but there is a clear link between taxa and use.

```
[20]: brr = joined[joined.KOD == 'BRR']
```

```
[ ]: counts = brr.cluster.value_counts(normalize=True)

sns.set(context="paper", style="ticks", rc={'patch.force_edgecolor': False})
fig, ax = plt.subplots(figsize=(10, 5))
sns.barplot(ax=ax, x=counts.index, y=counts, order=counts.index,␣
 ↪palette=symbology)
sns.despine(offset=10)
plt.ylabel('count')
plt.xlabel('cluster')
```

```
[29]: bd = joined[joined.KOD == 'BD']
```

```
[ ]: counts = bd.cluster.value_counts(normalize=True)

sns.set(context="paper", style="ticks", rc={'patch.force_edgecolor': False})
fig, ax = plt.subplots(figsize=(10, 5))
sns.barplot(ax=ax, x=counts.index, y=counts, order=counts.index,␣
 ↪palette=symbology)
sns.despine(offset=10)
plt.ylabel('count')
plt.xlabel('cluster')
```

```
[ ]: s = joined[joined.KOD == 'PND']

counts = s.cluster.value_counts(normalize=True)

sns.set(context="paper", style="ticks", rc={'patch.force_edgecolor': False})
fig, ax = plt.subplots(figsize=(10, 5))
sns.barplot(ax=ax, x=counts.index, y=counts, order=counts.index,␣
 ↪palette=symbology)
sns.despine(offset=10)
plt.ylabel('count')
plt.xlabel('cluster')
```

```
[11]: buildings = gpd.read_file('/Users/martin/Dropbox/Academia/Data/Geo/Prague/
 ↪Clustering/geometry.gpkg', layer='buildings')
```

```
[12]: buildings = buildings.merge(joined, on='uID', how='left')
```

1 get the most frequent use within 3 steps

[64]:
```
import libpysal

from tqdm import tqdm
```

[56]:
```
spatial_weights = libpysal.io.open('/Users/martin/Dropbox/Academia/Data/Geo/
 ↪Prague/Clustering/queen3_uID.gal', 'r').read()
spatial_weights.neighbors = {int(k): [int(i) for i in v] for k, v in␣
 ↪spatial_weights.neighbors.items()}
```

```
/Users/martin/anaconda3/envs/geo_dev/lib/python3.8/site-
packages/libpysal/weights/weights.py:165: UserWarning: The weights matrix is not
fully connected:
 There are 128 disconnected components.
  warnings.warn(message)
```

[13]:
```
buildings = buildings.set_index('uID')
```

[75]:
```
results = []
for index, row in tqdm(buildings.iterrows(), total=buildings.shape[0]):
    try:
        neighbours = spatial_weights.neighbors[index].copy()
        neighbours.append(index)

        uses = buildings.loc[neighbours]['KOD']
        results.append(uses.value_counts().index[0])
    except:
        results.append(None)
buildings['land_use_3'] = results
```

```
100%|      | 140408/140408 [04:36<00:00, 507.72it/s]
```

[77]:
```
buildings['land_use_3'].to_csv('/Users/martin/Dropbox/Academia/Data/Geo/Prague/
 ↪Validation/land_use_interpolated.csv')
```

[ ]:
```
interp = pd.read_csv('/Users/martin/Dropbox/Academia/Data/Geo/Prague/Validation/
 ↪land_use_interpolated.csv', index_col=0)
interp
```

[ ]:
```
buildings['land_use_3'] = interp
```

[16]:
```
counts = buildings.land_use_3.value_counts()
```

[19]:
```
counts[counts>1400]
```

[19]:
```
BRR    84085
BD     39963
BRV     2965
PRR     2035
PND     1557
Name: land_use_3, dtype: int64
```

[17]:
```python
generalised = []
major = counts[counts>1400].index
for _, use in buildings.land_use_3.iteritems():
    if use in major:
        generalised.append(use)
    else:
        generalised.append('other')
```

[18]:
```python
buildings['lu_gen'] = generalised
```

[28]:
```python
import numpy as np

def show_values_on_bars(axs):
    def _show_on_single_plot(ax):
        for p in ax.patches:
            _x = p.get_x() + p.get_width() / 2
            _y = p.get_y() + p.get_height() + 0.02
            value = '{:.2f}'.format(p.get_height())
            ax.text(_x, _y, value, ha="center")

    if isinstance(axs, np.ndarray):
        for idx, ax in np.ndenumerate(axs):
            _show_on_single_plot(ax)
    else:
        _show_on_single_plot(axs)

colors = [(257, 71, 27), (98, 93, 78), (14, 79, 58),
  (75, 90, 85), (347, 72, 60), (246, 79, 60)]
pal = [husl.husl_to_hex(*color) for color in colors]
```

[ ]:
```python
# save all clusters
for cl in range(20):
    data = buildings.loc[buildings['cluster'].isin([cl])]['lu_gen'].
 ↪value_counts(sort=False, normalize=True)

    sns.set(context="paper", style="ticks", rc={'patch.force_edgecolor': False})
    fig, ax = plt.subplots(figsize=(10, 5))
    sns.barplot(ax=ax, x=data.index, y=data, order=['BD', 'BRR', 'BRV', 'PND',␣
 ↪'PRR', 'other'],
                palette=pal)
```

```
    sns.despine(offset=10)
    ax.set_xticklabels(['Multi-family housing','Single-family␣
↪housing','Villas','Industry small','Industry large','Other'])
    plt.ylabel('frequency')
    plt.xlabel('land use category')
    plt.ylim(0, 1)
    show_values_on_bars(ax)
    for ext in ['pdf', 'png']:
        plt.savefig('figures/PRG_cluster_' + str(cl) + '_landuse.' + ext,␣
↪bbox_inches='tight')
    plt.close()
```

[ ]:
```
fig, ax = plt.subplots(2, 2, figsize=(14, 10))
labels = ['Multi-family','Single-family','Villas','Industry small','Industry␣
↪large','Other']

data = buildings.loc[buildings['cluster'].isin([11, 15, 5])]['lu_gen'].
↪value_counts(sort=False, normalize=True)
sns.barplot(ax=ax[0, 0], x=data.index, y=data, order=['BD', 'BRR', 'BRV',␣
↪'PND', 'PRR', 'other'],
                palette=pal)
sns.despine(offset=10)
ax[0,0].set_ylabel('frequency')
ax[0,0].set_title('compact city')
ax[0,0].set_ylim(0, 1)
show_values_on_bars(ax[0, 0])
ax[0,0].set_xticklabels(labels, rotation=45)

data = buildings.loc[buildings['cluster'].isin([3, 0, 8, 9, 13, 17])]['lu_gen'].
↪value_counts(sort=False, normalize=True)
sns.barplot(ax=ax[0, 1], x=data.index, y=data, order=['BD', 'BRR', 'BRV',␣
↪'PND', 'PRR', 'other'], palette=pal)
sns.despine(offset=10)
ax[0,1].set_ylabel('frequency')
ax[0,1].set_title('low-rise city')
ax[0,1].set_xticklabels(labels)
ax[0,1].set_ylim(0, 1)
show_values_on_bars(ax[0, 1])

data = buildings.loc[buildings['cluster'].isin([1, 19])]['lu_gen'].
↪value_counts(sort=False, normalize=True)
sns.barplot(ax=ax[1, 0], x=data.index, y=data, order=['BD', 'BRR', 'BRV',␣
↪'PND', 'PRR', 'other'], palette=pal)
sns.despine(offset=10)
ax[1,0].set_ylabel('frequency')
ax[1,0].set_xticklabels(labels)
```

```
ax[1,0].set_title('industrial city')
ax[1,0].set_ylim(0, 1)
show_values_on_bars(ax[1, 0])

data = buildings.loc[buildings['cluster'].isin([12, 14, 2, 10])]['lu_gen'].
 ↪value_counts(sort=False, normalize=True)
sns.barplot(ax=ax[1, 1], x=data.index, y=data, order=['BD', 'BRR', 'BRV',␣
 ↪'PND', 'PRR', 'other'], palette=pal)
sns.despine(offset=10)
ax[1,1].set_ylabel('frequency')
ax[1,1].set_xticklabels(labels)
ax[1,1].set_title('heterogenous dense city branch')
ax[1,1].set_ylim(0, 1)
show_values_on_bars(ax[1, 1])

plt.tight_layout()
plt.savefig('figures/PRG_branch_landuse_subplot.pdf')
```

```
[ ]: fig, ax = plt.subplots(2, 2, figsize=(14, 10))
labels = ['Multi-family','Single-family','Villas','Industry small','Industry␣
 ↪large','Other']

data = buildings.loc[buildings['cluster'].isin([11])]['lu_gen'].
 ↪value_counts(sort=False, normalize=True)
sns.barplot(ax=ax[0, 0], x=data.index, y=data, order=['BD', 'BRR', 'BRV',␣
 ↪'PND', 'PRR', 'other'],
                palette=pal)
sns.despine(offset=10)
ax[0,0].set_ylabel('frequency')
ax[0,0].set_title('cluster 11')
ax[0,0].set_ylim(0, 1)
show_values_on_bars(ax[0, 0])
ax[0,0].set_xticklabels(labels, rotation=45)

data = buildings.loc[buildings['cluster'].isin([5])]['lu_gen'].
 ↪value_counts(sort=False, normalize=True)
sns.barplot(ax=ax[0, 1], x=data.index, y=data, order=['BD', 'BRR', 'BRV',␣
 ↪'PND', 'PRR', 'other'], palette=pal)
sns.despine(offset=10)
ax[0,1].set_ylabel('frequency')
ax[0,1].set_title('cluster 5')
ax[0,1].set_xticklabels(labels)
ax[0,1].set_ylim(0, 1)
show_values_on_bars(ax[0, 1])
```

```
data = buildings.loc[buildings['cluster'].isin([12])]['lu_gen'].
 ↪value_counts(sort=False, normalize=True)
sns.barplot(ax=ax[1, 0], x=data.index, y=data, order=['BD', 'BRR', 'BRV',␣
 ↪'PND', 'PRR', 'other'], palette=pal)
sns.despine(offset=10)
ax[1,0].set_ylabel('frequency')
ax[1,0].set_xticklabels(labels)

ax[1,0].set_title('cluster 12')
ax[1,0].set_ylim(0, 1)
show_values_on_bars(ax[1, 0])

data = buildings.loc[buildings['cluster'].isin([13])]['lu_gen'].
 ↪value_counts(sort=False, normalize=True)
sns.barplot(ax=ax[1, 1], x=data.index, y=data, order=['BD', 'BRR', 'BRV',␣
 ↪'PND', 'PRR', 'other'], palette=pal)
sns.despine(offset=10)
ax[1,1].set_ylabel('frequency')
ax[1,1].set_xticklabels(labels)
ax[1,1].set_title('cluster 13')
ax[1,1].set_ylim(0, 1)
show_values_on_bars(ax[1, 1])

plt.tight_layout()
plt.savefig('figures/PRG_cluster_landuse_subplot.pdf')
```

```
[ ]: colors = [(257, 71, 27), (98, 93, 78), (14, 79, 58), (26, 0, 50),
               (75, 90, 85), (347, 72, 60), (246, 79, 60)]
     color = (257, 71, 27)  # here for arrow, title, scalebar

     # plotting
     c = husl.husl_to_hex(*color)

     cmap = ListedColormap([husl.husl_to_hex(*color) for color in colors])
     for cl in tqdm(range(20), total=20):

         # requires geopandas PR 1159
         sub = buildings[buildings.cluster == cl]
         bounds = sub.total_bounds
         ax = sub.plot('lu_gen', categorical=True, figsize=(30, 30), cmap=cmap,␣
      ↪zorder=2,
                       categories=['BD', 'BRR', 'BRV', 'PND', 'PRR', 'other'],␣
      ↪legend=True,
                         legend_kwds=dict(loc='center right', frameon=False))
         buildings.cx[bounds[0]:bounds[2], bounds[1]:bounds[3]].plot('lu_gen',ax=ax,␣
      ↪categorical=True, cmap=cmap,
```

```
                     alpha=.2, zorder=1,
                     categories=['BD', 'BRR', 'BRV', 'PND', 'PRR', 'other'])
    ax.set_axis_off()


    # add scalebar
    scalebar = ScaleBar(dx=1,
                        color=c,
                        location=1,
                        height_fraction=0.001,
                        #fixed_value=1000,
                        label='cluster {} and historical period'.format(cl),
                        label_loc='bottom'
                        )
    ax.add_artist(scalebar)

    # add arrow
    north_arrow(plt.gcf(), ax, -7.5, legend_size=(.04,.04), outline=1,␣
↪edgecolor=c, facecolor=c)
    for ext in ['pdf', 'png']:
        plt.savefig('figures/PRG_cluster_{}_landuse_map.'.format(cl) + ext,␣
↪bbox_inches='tight')
    plt.close()
```

1.1 statistics

[125]:
```
buildings[['cluster', 'lu_gen']]
```

[125]:
```
        cluster lu_gen
uID
0           1.0    PRR
1          11.0     BD
2          13.0    BRR
3          18.0    BRR
4          14.0     BD
…           …     …
140457     11.0     BD
140458     13.0    BRR
140459      5.0     BD
140460      5.0     BD
140461      4.0    BRR

[140408 rows x 2 columns]
```

[129]:
```
import scipy.stats as ss

def cramers_v(x, y):
```

```python
    confusion_matrix = pd.crosstab(x,y)
    chi2 = ss.chi2_contingency(confusion_matrix)[0]
    n = confusion_matrix.sum().sum()
    phi2 = chi2/n
    r,k = confusion_matrix.shape
    phi2corr = max(0, phi2-((k-1)*(r-1))/(n-1))
    rcorr = r-((r-1)**2)/(n-1)
    kcorr = k-((k-1)**2)/(n-1)
    return np.sqrt(phi2corr/min((kcorr-1),(rcorr-1)))
```

[130]:
```python
cramers_v(buildings.cluster, buildings.lu_gen)
```

[130]: 0.5009730826143093

The resulting value of 0.5 indicates moderate to high relationship between clustering and generalised land use.

Chi-square test of independence of variables in a contingency table

[137]:
```python
confusion_matrix = pd.crosstab(buildings.cluster, buildings.lu_gen)
chi, p, dof, exp = ss.chi2_contingency(confusion_matrix)
```

[138]:
```python
chi
```

[138]: 176165.83103092626

[139]:
```python
p
```

[139]: 0.0

[140]:
```python
dof
```

[140]: 95

p-value is < 0.001

there is a significant dependency between variables