Self Organising Maps

June 5, 2020

[1]:
```
# Run the below code in the terminal to install the SOMPY package
# git clone https://github.com/hhl60492/SOMPY_robust_clustering.git
# cd path/to/the/cloned/folder/
# python setup.py install
```

[2]:
```
import sys
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline
import joblib
sys.path.append('/Users/vishwajit/Desktop/SOMPY_robust_clustering-master/')␣
 ↪#path/to/the/cloned/folder/
import sompy
from sompy.sompy import SOMFactory
from sompy.visualization import mapview
from sompy.visualization.umatrix import UMatrixView
from sompy.visualization.hitmap import HitMapView
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-
packages/sklearn/externals/joblib/__init__.py:15: FutureWarning:
sklearn.externals.joblib is deprecated in 0.21 and will be removed in 0.23.
Please import this functionality directly from joblib, which can be installed
with: pip install joblib. If this warning is raised when loading pickled models,
you may need to re-serialize those models with scikit-learn 0.21+.
  warnings.warn(msg, category=FutureWarning)
```

[3]:
```
df = pd.read_csv('Cleaned_percent_data.csv',index_col='zip code tabulation␣
 ↪area')
```

[4]:
```
df.shape
```

[4]:
```
(33120, 238)
```

[5]:
```
sm = SOMFactory().build(df.values, normalization = 'var', initialization='pca')
sm.train(n_job=1, verbose=False, train_rough_len=2, train_finetune_len=5)
# Takes around 8-10 minutes of time to run
# No need to initialize number of nodes as it is calculated based on the␣
 ↪eigenvalues of the matrix
```

[6]:

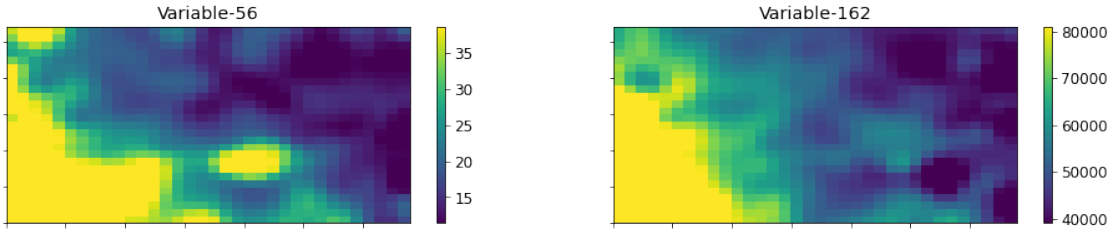
```
sm.codebook.mapsize
# There are total of 918 nodes (34 horizontal and 27 vertical), each node has a␣
 ↪weight of length 238.
```

[6]:

```
[27, 34]
```

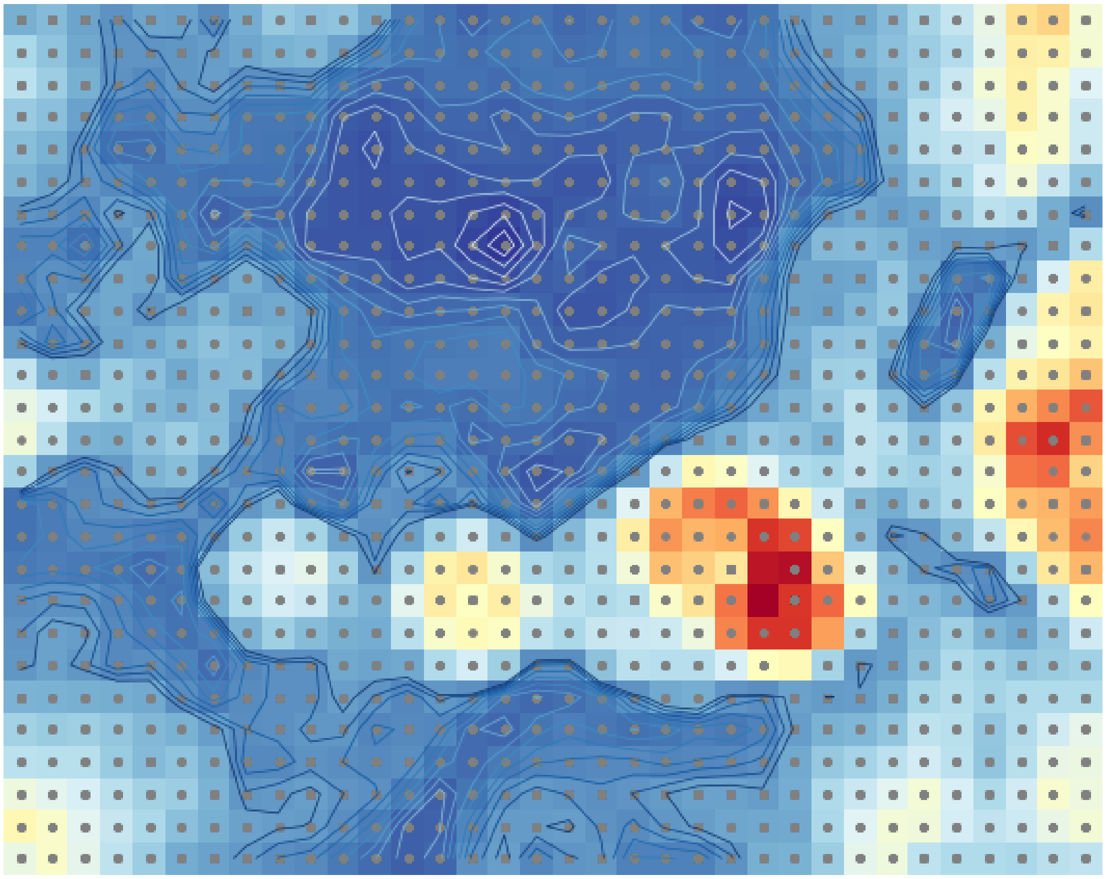
[7]:

```
view2D  = mapview.View2D(10,10,"rand data",text_size=12)
view2D.show(sm, col_sz=2, which_dim=[55,161], desnormalize=True)
#which_dim takes a list of attributes/variables to be visualized using colormap.
#Variable-56 corresponds to percentage of population with Bachelor's degree or␣
 ↪higher
#and Variable-162 to the median household income. As expected both the␣
 ↪colormaps look similar.
```



[8]:

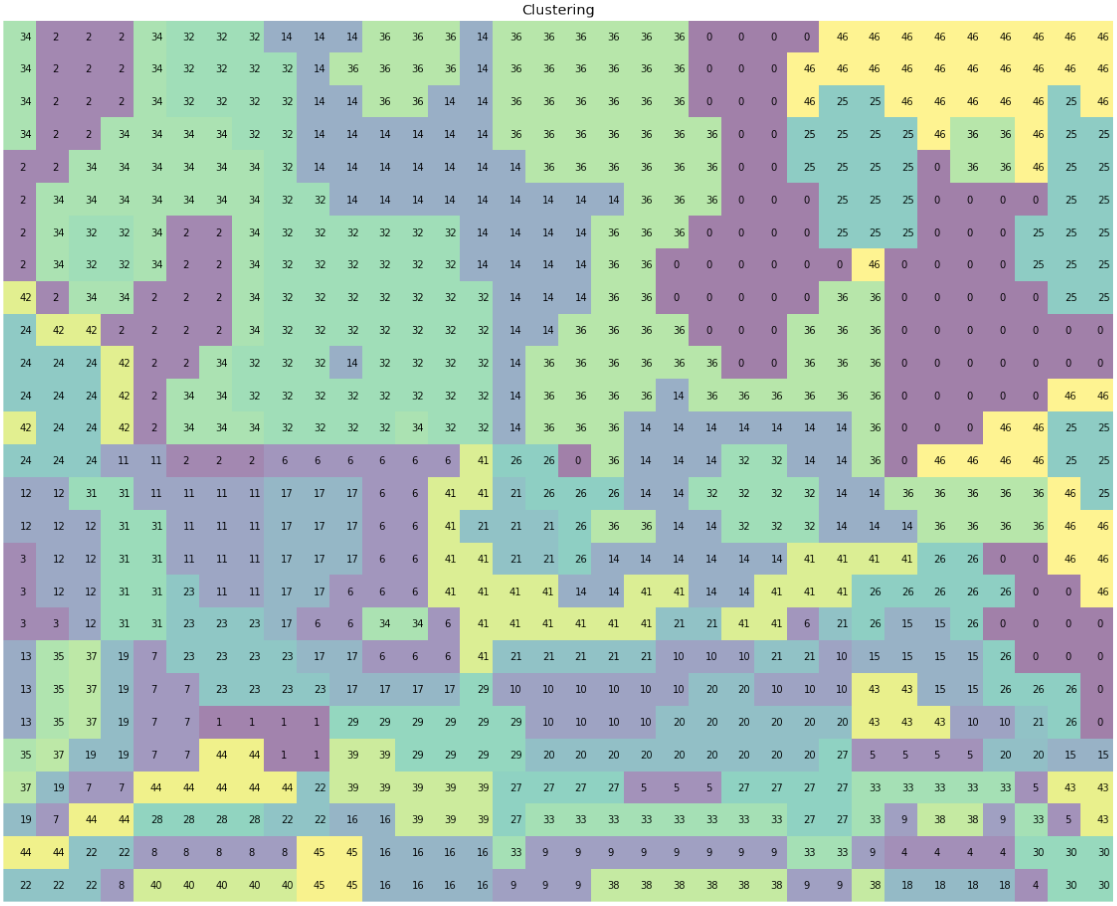
```
umat  = UMatrixView(width=10,height=10,title='U-matrix')
UMatrix = umat.show(sm)
#UMatrix contains distance between neighboring nodes. Red regions are the ones␣
 ↪with large distances
#whereas blue with smaller distances between neighbors.
#This helps in identifying regions with dense clusters(smaller distances)
```



```
[9]: #sm.cluster uses 'elbow' method to find the optimal number of clusters. It␣
      ↪calculates SSE for K ranging from 2 to
     #k_end (specified). One can also give directly the optimal number of clusters
     K = 50 # stop at this k for SSE sweep
     K_opt = 47
     [labels, km, norm_data] = sm.cluster(opt=K_opt)
     hits  = HitMapView(20,20,"Clustering",text_size=12)
     a = hits.show(sm)
```

```
Performing K-means SSE elbow sweep…
```

```
/Users/vishwajit/Desktop/SOMPY_robust_clustering-
master/sompy/visualization/hitmap.py:23: MatplotlibDeprecationWarning: Adding an
axes using the same arguments as a previous axes currently reuses the earlier
instance.  In a future version, a new instance will always be created and
returned.  Meanwhile, this warning can be suppressed, and the future behavior
ensured, by passing a unique label to each axes instance.
  ax = self._fig.add_subplot(111)
```



Clustering

34	2	2	2	34	32	32	32	14	14	14	36	36	36	14	36	36	36	36	36	36	0	0	0	0	46	46	46	46	46	46	46	46	46
34	2	2	2	34	32	32	32	32	14	36	36	36	36	14	36	36	36	36	36	36	0	0	0	46	46	46	46	46	46	46	46	46	46
34	2	2	2	34	32	32	32	32	14	14	36	36	14	14	36	36	36	36	36	36	0	0	0	46	25	25	46	46	46	46	46	25	46
34	2	2	34	34	34	34	32	32	14	14	14	14	14	14	36	36	36	36	36	36	36	0	0	25	25	25	25	46	36	36	46	25	25
2	2	34	34	34	34	34	34	32	14	14	14	14	14	14	14	36	36	36	36	36	36	0	0	25	25	25	25	0	36	36	46	25	25
2	34	34	34	34	34	34	34	32	32	14	14	14	14	14	14	14	14	14	36	36	36	0	0	0	25	25	25	0	0	0	0	25	25
2	34	32	32	34	2	2	34	32	32	32	32	32	32	14	14	14	14	36	36	36	0	0	0	0	25	25	25	0	0	0	25	25	25
2	34	32	32	34	2	2	34	32	32	32	32	32	32	14	14	14	14	36	36	0	0	0	0	0	0	46	0	0	0	0	25	25	25
42	2	34	34	2	2	2	34	32	32	32	32	32	32	32	14	14	14	36	36	0	0	0	0	0	36	36	0	0	0	0	0	25	25
24	42	42	2	2	2	2	34	32	32	32	32	32	32	32	14	14	36	36	36	36	0	0	0	36	36	36	0	0	0	0	0	0	0
24	24	24	42	2	2	34	32	32	32	14	32	32	32	32	14	36	36	36	36	36	36	0	0	36	36	36	0	0	0	0	0	0	0
24	24	24	42	2	34	34	32	32	32	32	32	32	32	32	14	36	36	36	36	14	36	36	36	36	36	36	0	0	0	0	0	46	46
42	24	24	42	2	34	34	34	32	32	32	32	34	32	32	14	36	36	36	14	14	14	14	14	14	14	36	0	0	0	46	46	25	25
24	24	24	11	11	2	2	2	6	6	6	6	6	6	41	26	26	0	36	14	14	14	32	32	14	14	36	0	46	46	46	46	25	25
12	12	31	31	11	11	11	11	17	17	17	6	6	41	41	21	26	26	26	14	14	32	32	32	32	14	14	36	36	36	36	36	46	25
12	12	12	31	31	11	11	11	17	17	17	6	6	41	21	21	21	26	36	36	14	14	32	32	32	14	14	14	36	36	36	36	46	46
3	12	12	31	31	11	11	11	17	17	17	6	6	41	41	21	21	26	14	14	14	14	14	14	41	41	41	41	26	26	0	0	46	46
3	12	12	31	31	23	11	11	17	17	6	6	6	41	41	41	41	14	14	41	41	14	14	41	41	41	26	26	26	26	26	0	0	46
3	3	12	31	31	23	23	23	17	6	6	34	34	6	41	41	41	41	41	41	21	21	41	41	6	21	26	15	15	26	0	0	0	0
13	35	37	19	7	23	23	23	23	17	17	6	6	6	41	21	21	21	21	21	10	10	10	21	21	10	15	15	15	15	26	0	0	0
13	35	37	19	7	7	23	23	23	23	17	17	17	17	29	10	10	10	10	10	10	20	20	10	10	10	43	43	15	15	26	26	26	0
13	35	37	19	7	7	1	1	1	1	29	29	29	29	29	29	10	10	10	10	20	20	20	20	20	20	43	43	43	10	10	21	26	0
35	37	19	19	7	7	44	44	1	1	39	39	29	29	29	29	20	20	20	20	20	20	20	20	20	27	5	5	5	5	20	20	15	15
37	19	7	7	44	44	44	44	44	22	39	39	39	39	39	27	27	27	27	5	5	5	27	27	27	27	33	33	33	33	33	5	43	43
19	7	44	44	28	28	28	28	22	22	16	16	39	39	39	27	33	33	33	33	33	33	33	33	27	27	33	9	38	38	9	33	5	43
44	44	22	22	8	8	8	8	8	45	45	16	16	16	16	33	9	9	9	9	9	9	9	9	33	33	9	4	4	4	4	30	30	30
22	22	22	8	40	40	40	40	40	45	45	16	16	16	16	9	9	9	38	38	38	38	38	38	9	9	38	18	18	18	18	4	30	30

[ ]:

```
# still to be done: Find the zip codes belonging to each clusters,
# justify the above results using some method/reasons, derive insights from the results
```