COMP47750/COMP47990 Machine Learning with Python

Spectral Clustering

Part I
Fundamentals

Pádraig Cunningham

School of Computer Science

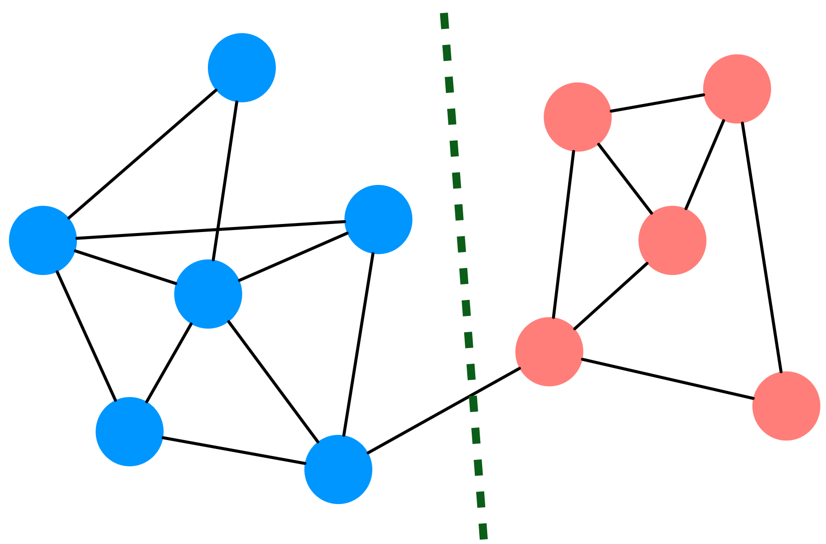
Overview

- Equivalence of Clustering & Graph Partitioning
- Graph Partitioning using Eigenvectors
- Spectral Clustering
- Spectral Clustering on Feature-Vector data
- Spectral Clustering in scikit learn

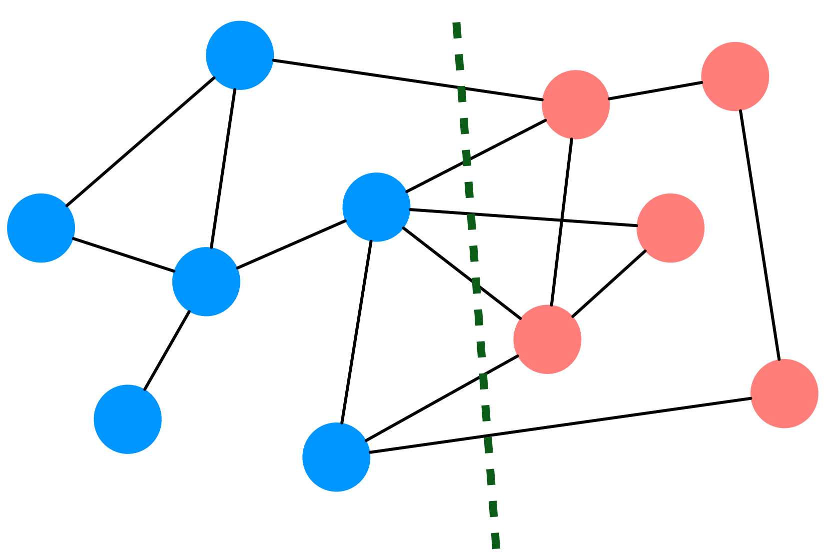
2 Notebook:
18 Spectral Clustering
18 Spectral Clustering sklearn

Graph Partitioning

- **Goal:** Divide a graph into two or more "good" parts - e.g. split a social network into two or more meaningful communities.
- Many approaches, generally with broadly similar motivations to clustering algorithms.
  1. High level of internal connections (i.e. within parts).
  2. Low level of external connections (i.e. between parts).



Good Partition

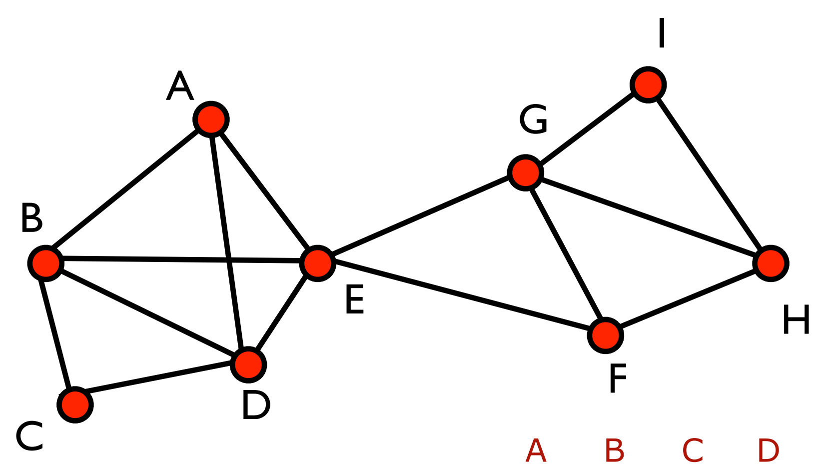


Bad Partition

- Often try to minimise the cut size - i.e. number of edges in the graph that are cut by the partition.

Spectral Partitioning

- Turn graph partitioning into a spectral problem.
- Compute the eigendecomposition of the Laplacian matrix **L** of the adjacency matrix **S** of a graph.
- To partition graph in 2: Split using eigenvector associated with the second smallest eigenvalue of **L** - the Fiedler Vector.



Adjacency Matrix **S**

	A	B	C	D	E	F	G	H	I
A	0	1	0	1	1	0	0	0	0
B	1	0	1	1	1	0	0	0	0
C	0	1	0	1	0	0	0	0	0
D	1	1	1	0	1	0	0	0	0
E	1	1	0	1	0	1	1	0	0
F	0	0	0	0	1	0	1	1	0
G	0	0	0	0	1	1	0	1	1
H	0	0	0	0	0	1	1	0	1
I	0	0	0	0	0	0	1	1	0

Let’s work through an example

Laplacian

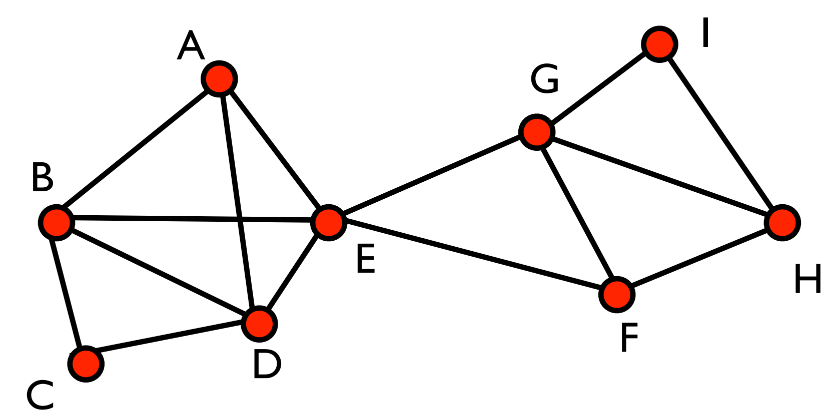
Notebook:
18 Spectral Clustering

$$L = D - S$$

$$\overset{\Large L}{\begin{matrix} 3 & -1 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 4 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & -1 & 5 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 4 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 2 \end{matrix}} = \overset{\Large D}{\begin{matrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{matrix}} - \overset{\Large S}{\begin{matrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{matrix}}$$

Degree Matrix
(row/column sums)

Original Adjacency Matrix

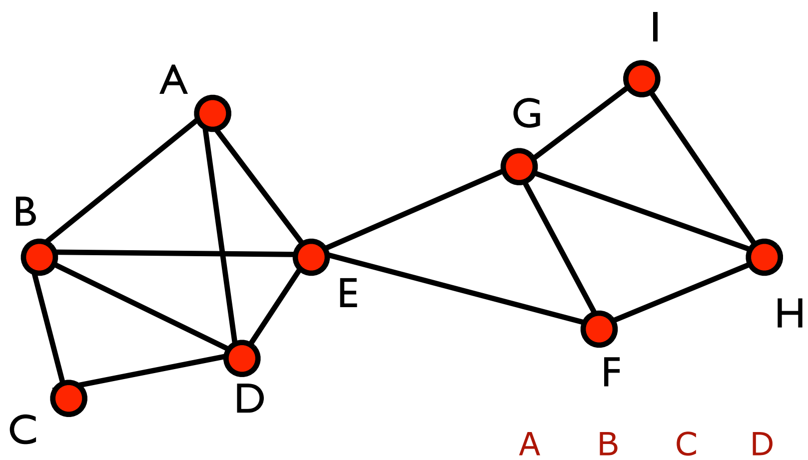



```python
from scipy.sparse.csgraph import laplacian
...
Lkc = laplacian(kc)
```

Spectral Partitioning

```python
from scipy.sparse.csgraph import laplacian
...
Lkc = laplacian(kc)
e_val, e_vec = np.linalg.eig(Lkc)
```

- Turn graph partitioning into a spectral problem.
- Compute the eigendecomposition of the Laplacian matrix **L** of the adjacency matrix **S** of a graph.
- To partition graph in 2: Split using eigenvector associated with the second smallest eigenvalue of **L** - the Fiedler Vector.

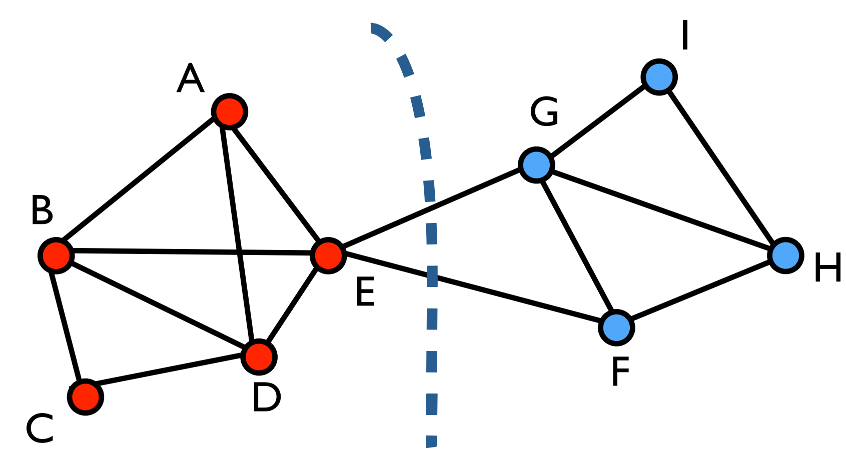


Eigenvectors of Matrix **L**

0.33	-0.29	-0.37	0.45	-0.27	0.54	-0.3	0	0.1
0.33	-0.32	-0.03	0.1	0	-0.35	0.3	0.71	0.25
0.33	-0.42	0.64	-0.41	0	0.26	-0.21	0	-0.11
0.33	-0.32	-0.03	0.1	0	-0.35	0.3	-0.71	0.25
0.33	-0.08	-0.29	0.01	0.27	-0.21	-0.06	0	-0.82
0.33	0.25	-0.36	-0.53	0.27	0.42	0.38	0	0.18
0.33	0.3	-0.08	-0.1	0.27	-0.33	-0.7	0	0.35
0.33	0.41	0.04	-0.17	-0.8	-0.16	0.05	0	-0.14
0.33	0.47	0.47	0.54	0.27	0.18	0.23	0	-0.05

Adjacency Matrix **S**

	A	B	C	D	E	F	G	H	I
A	0	1	0	1	1	0	0	0	0
B	1	0	1	1	1	0	0	0	0
C	0	1	0	1	0	0	0	0	0
D	1	1	1	0	1	0	0	0	0
E	1	1	0	1	0	1	1	0	0
F	0	0	0	0	1	0	1	1	0
G	0	0	0	0	1	1	0	1	1
H	0	0	0	0	0	1	1	0	1
I	0	0	0	0	0	0	1	1	0



Minimum Cut

Some detail on why this works in M3 Matrices lecture

Example: Spectral Analysis

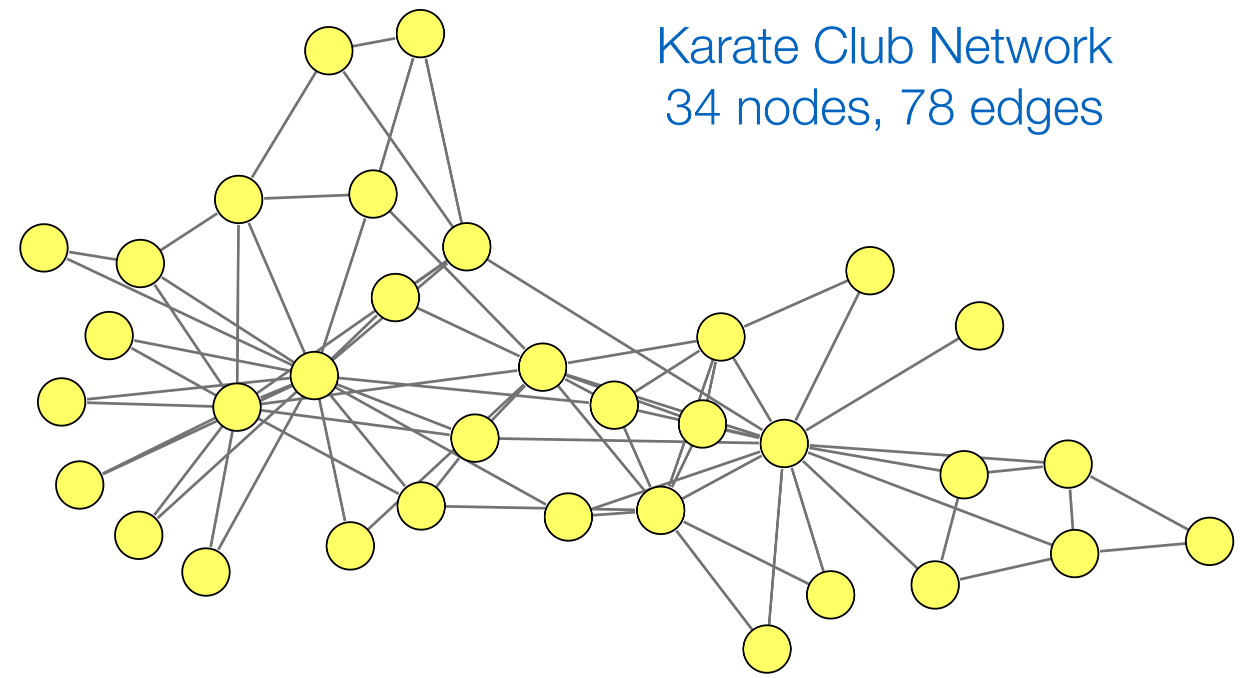
Example: Apply spectral bi-partitioning to Zachary karate club network.

1. Construct 34x34 adjacency matrix **S**.
2. Construct 34x34 diagonal matrix **D**.

$$\mathbf{D}_{ii} = \sum_{j=1}^{n} S_{ij}$$

3. Construct 34x34 Laplacian matrix **L**.

$$\mathbf{L} = \mathbf{D} - \mathbf{S}$$



34x34 Laplacian matrix

16	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	0	0	0	-1	0	-1	0	-1	0	0	0	0	0	0	0	0	0	-1	0	0
-1	9	-1	-1	0	0	0	-1	0	0	0	0	0	-1	0	0	0	-1	0	-1	0	-1	0	0	0	0	0	0	0	0	-1	0	0	0
-1	-1	10	-1	0	0	0	-1	-1	-1	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1	0	0	0	-1	0
-1	-1	-1	6	0	0	0	-1	0	0	0	0	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	0	0	0	3	0	-1	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	0	0	0	0	4	-1	0	0	0	-1	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	0	0	0	-1	-1	4	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	-1	-1	-1	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	0	-1	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	-1	-1
0	0	-1	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1
-1	0	0	0	-1	-1	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	0	0	-1	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1
0	0	0	0	0	-1	-1	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1
-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	-1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	-1	-1
-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	-1	-1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	-1	0	-1	0	-1	0	0	-1	-1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	-1	0	-1	0	0	0	-1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1	3	0	0	0	0	0	-1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	-1	0	0	0	-1
0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1	0	0	4	0	0	0	0	0	-1
0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	-1	0	-1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	-1	0	0	4	0	0	-1	-1
0	-1	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	-1	-1
-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1	0	0	-1	0	0	6	-1	-1
0	0	-1	0	0	0	0	0	-1	0	0	0	0	0	-1	-1	0	0	-1	0	-1	0	-1	-1	0	0	0	0	0	-1	-1	-1	12	-1
0	0	0	0	0	0	0	0	-1	-1	0	0	0	-1	-1	-1	0	0	-1	-1	-1	0	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	17

Example: Spectral Analysis

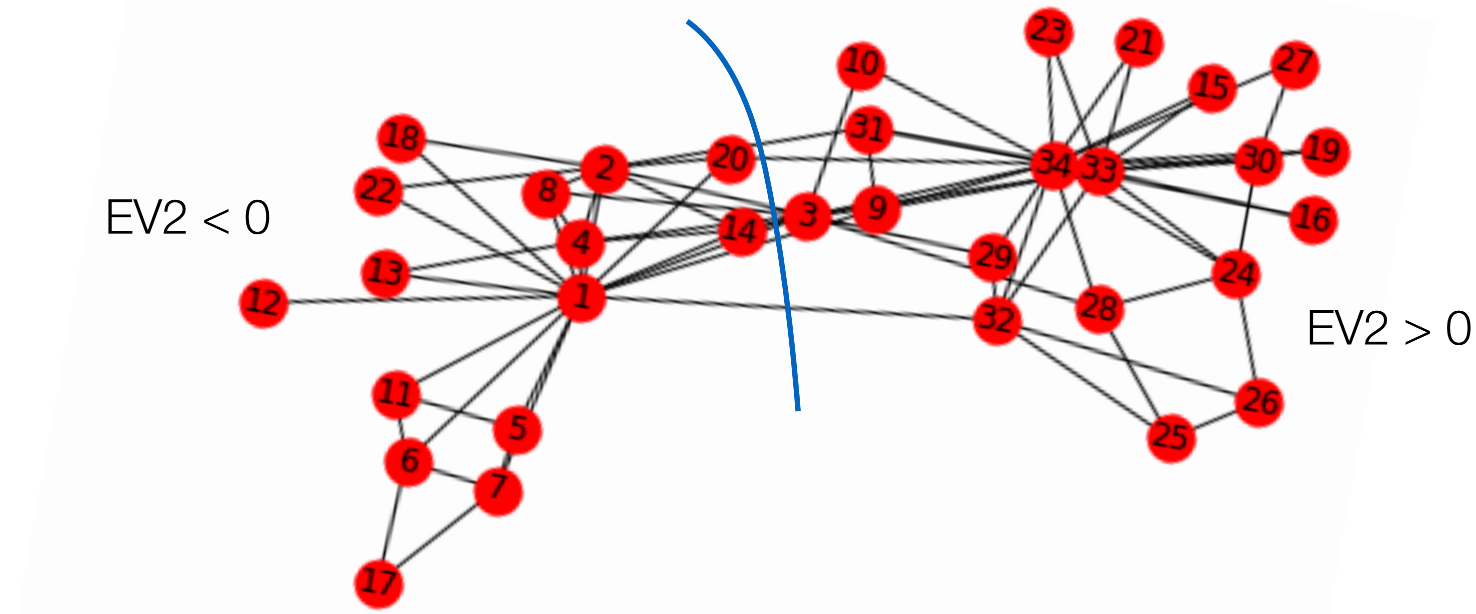
Analyse second smallest eigenvector of **L** (i.e. the “Fiedler vector”).
Note: ignore the redundant EV1, which contains constant values.

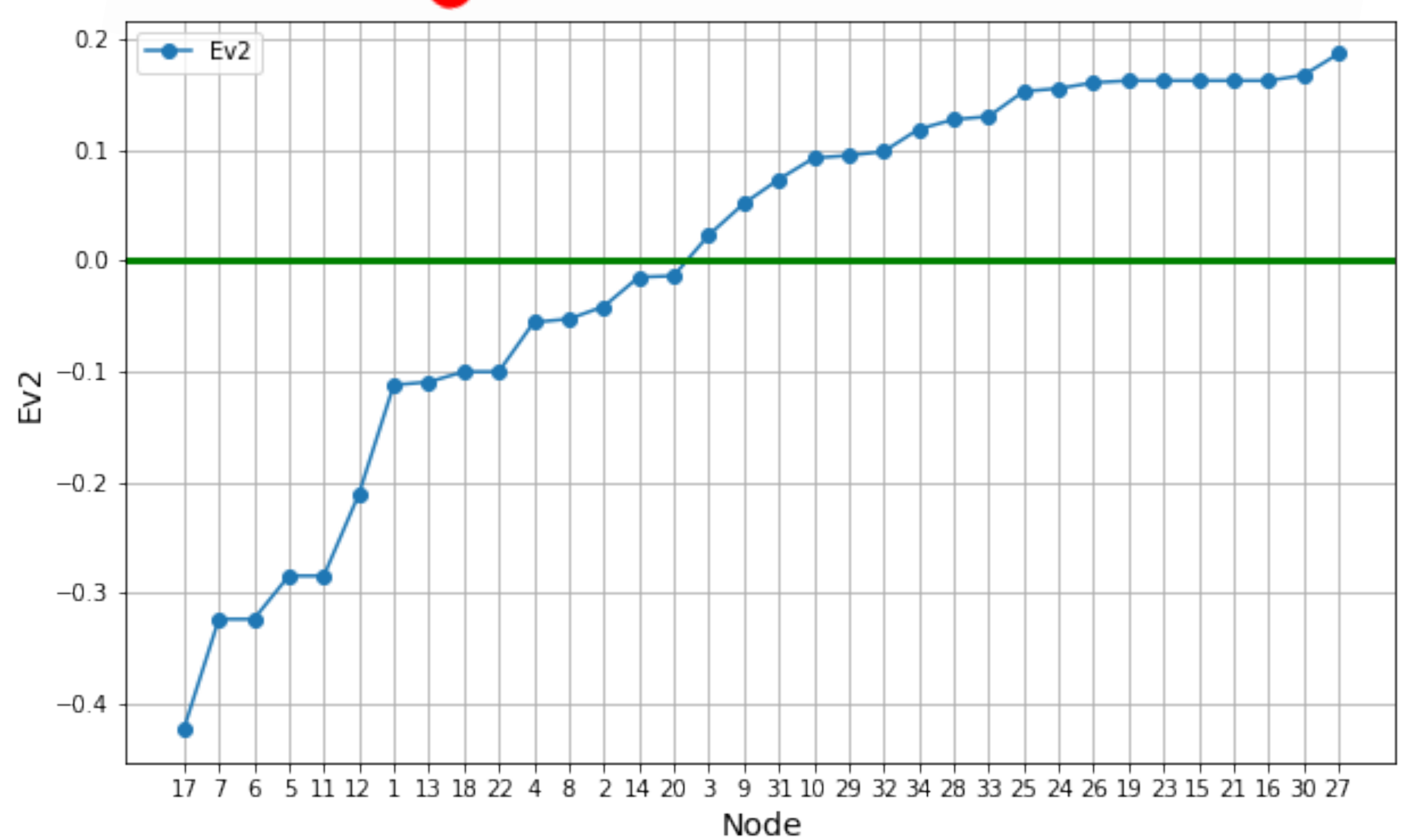
Node	EV1	EV2
1	0.1715	-0.1121
2	0.1715	-0.0413
3	0.1715	0.0232
4	0.1715	-0.0555
5	0.1715	-0.2846
6	0.1715	-0.3237
7	0.1715	-0.3237
8	0.1715	-0.0526
9	0.1715	0.0516
10	0.1715	0.0928
…	…	…
…	…	…
30	0.1715	0.1677
31	0.1715	0.0735
32	0.1715	0.0988
33	0.1715	0.1303
34	0.1715	0.1189

Smallest eigenvectors of Laplacian **L**

Node	EV2
17	-0.4228
6	-0.3237
7	-0.3237
5	-0.2846
11	-0.2846
12	-0.211
1	-0.1121
13	-0.1095
18	-0.1002
22	-0.1002
4	-0.0555
8	-0.0526
2	-0.0413
14	-0.0147
20	-0.0136
3	0.0232
…	…

Sorted Fiedler vector EV2

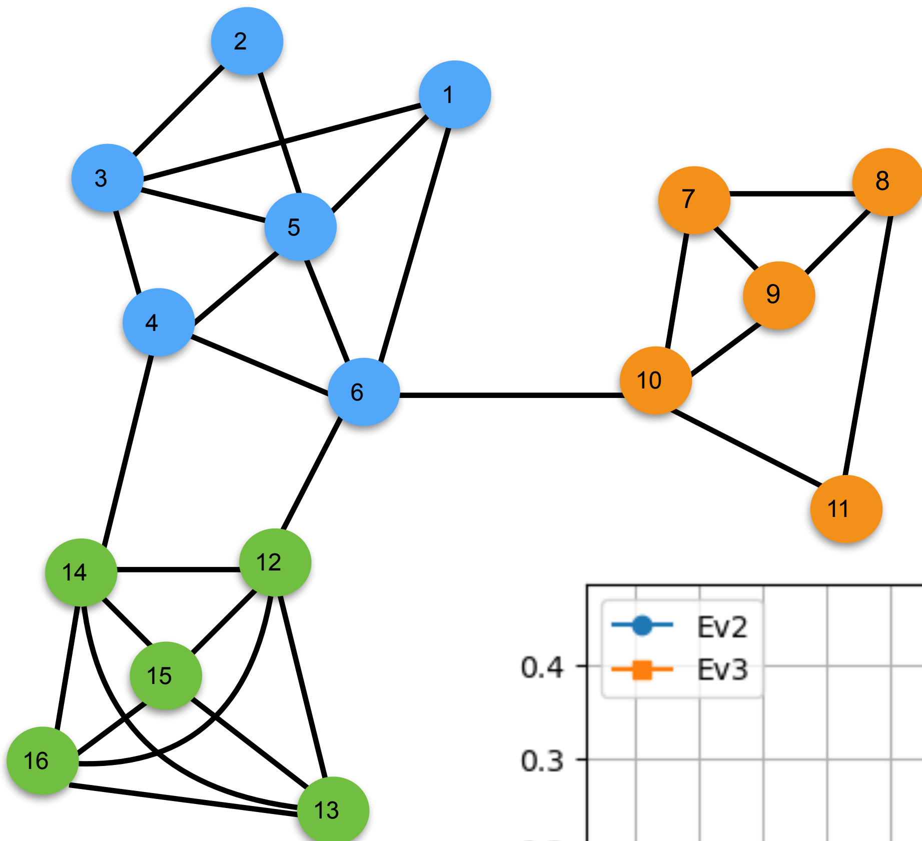


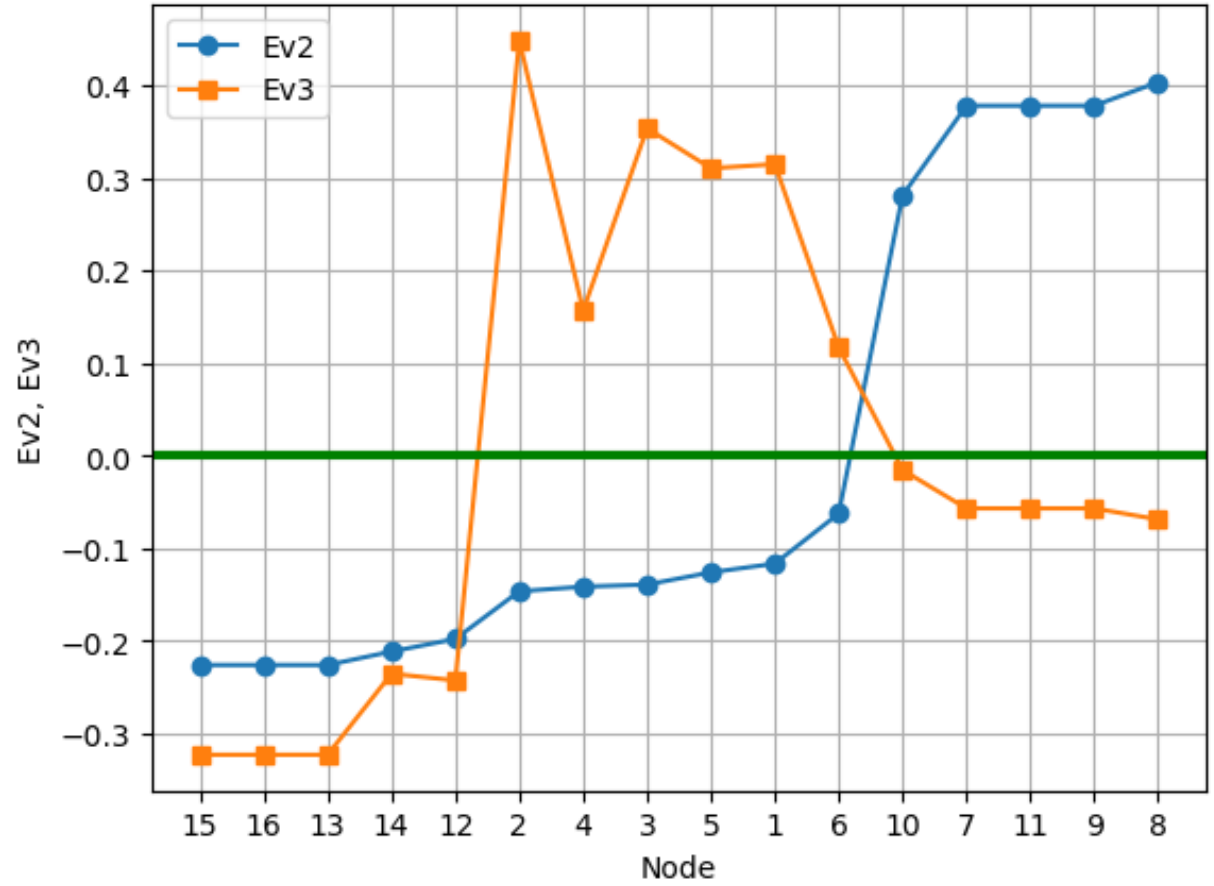


3 Clusters

Notebook:
18 Spectral Clustering

- Network with 3 clear clusters



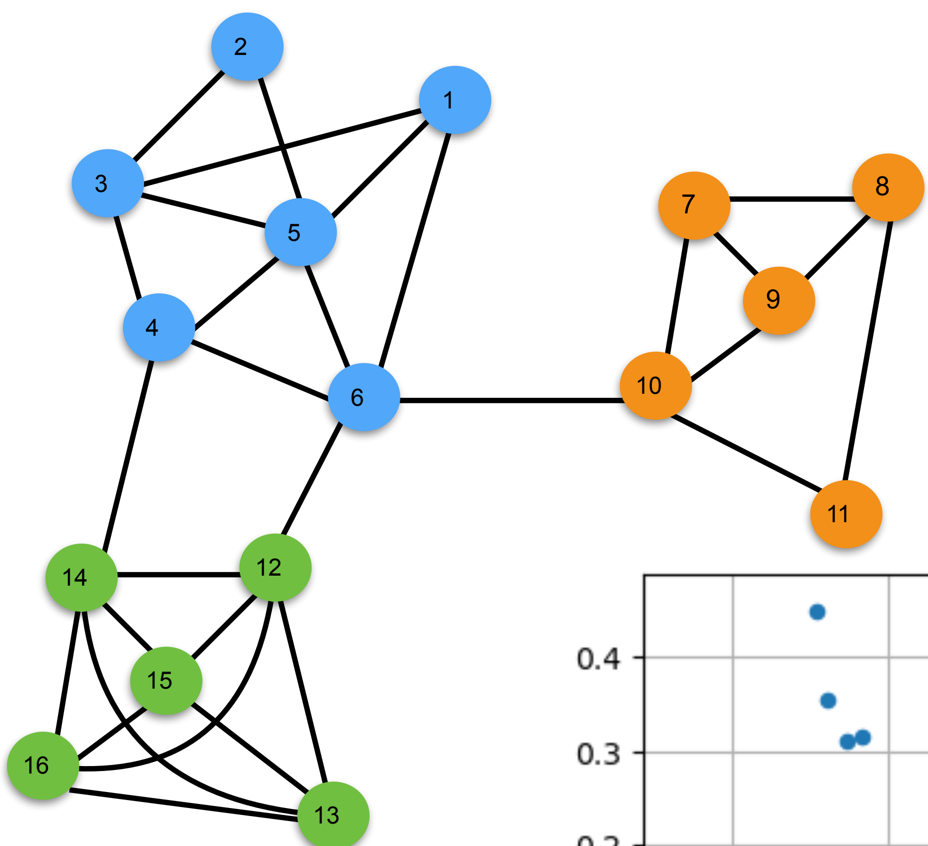


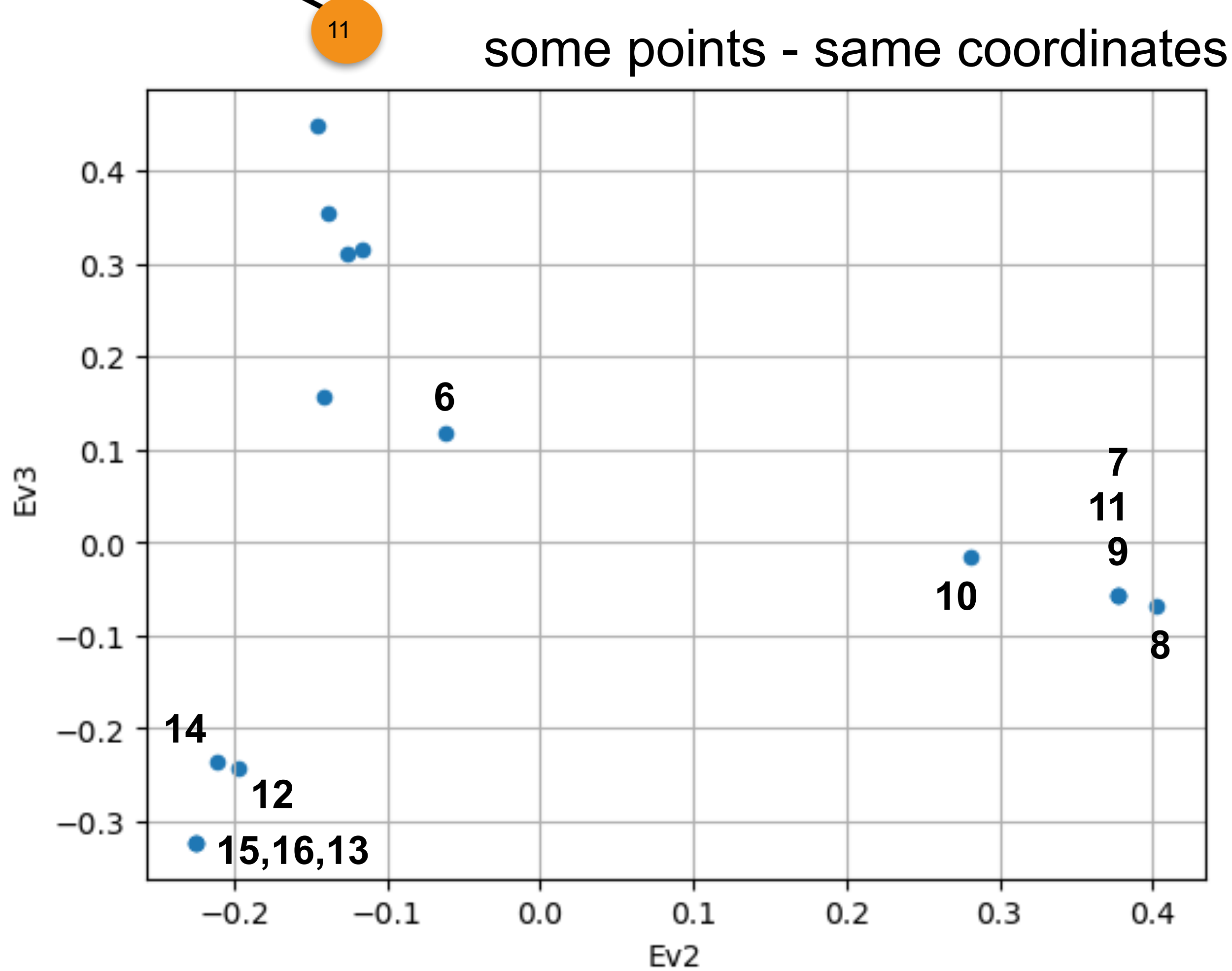
Eigenvectors of Laplacian

	Ev1	Ev2	Ev3
15	0.25	-0.226	-0.323
16	0.25	-0.226	-0.323
13	0.25	-0.226	-0.323
14	0.25	-0.211	-0.235
12	0.25	-0.198	-0.242
2	0.25	-0.146	0.448
4	0.25	-0.141	0.157
3	0.25	-0.139	0.353
5	0.25	-0.126	0.310
1	0.25	-0.116	0.315
6	0.25	-0.062	0.117
10	0.25	0.281	-0.015
7	0.25	0.378	-0.057
11	0.25	0.378	-0.057
9	0.25	0.378	-0.057
8	0.25	0.403	-0.069

3 Clusters

- Assign clusters using *k*-means in embedded (EV) space





Eigenvectors of Laplacian

	Ev1	Ev2	Ev3
15	0.25	-0.226	-0.323
16	0.25	-0.226	-0.323
13	0.25	-0.226	-0.323
14	0.25	-0.211	-0.235
12	0.25	-0.198	-0.242
2	0.25	-0.146	0.448
4	0.25	-0.141	0.157
3	0.25	-0.139	0.353
5	0.25	-0.126	0.310
1	0.25	-0.116	0.315
6	0.25	-0.062	0.117
10	0.25	0.281	-0.015
7	0.25	0.378	-0.057
11	0.25	0.378	-0.057
9	0.25	0.378	-0.057
8	0.25	0.403	-0.069

COMP47750/COMP47990
Machine Learning with Python

Spectral Clustering

Part II
Spectral Clustering
In scikit-learn

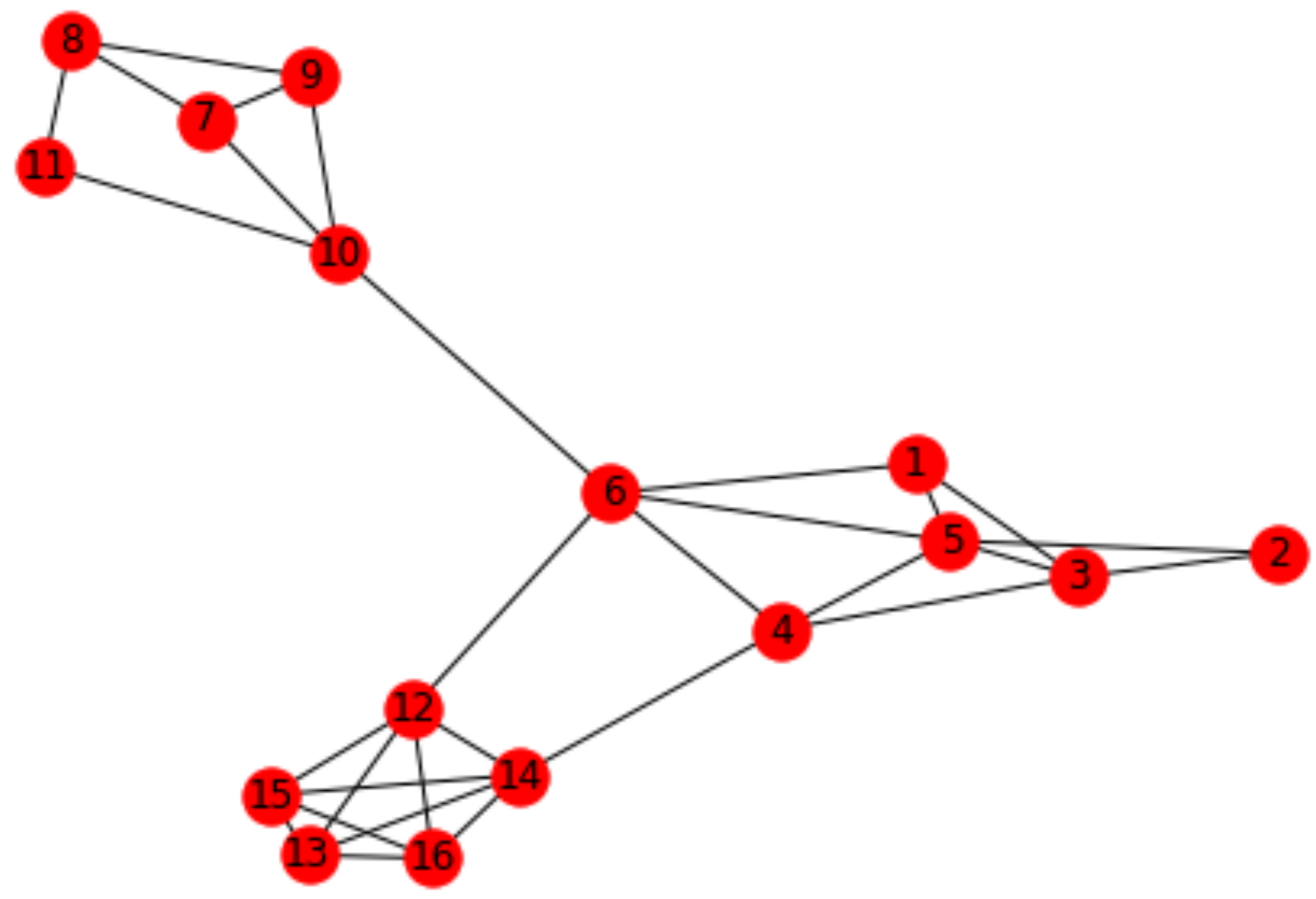
Pádraig Cunningham

School of Computer Science

Spectral Clustering in scikit-learn

```
from sklearn.cluster import SpectralClustering
sclust = SpectralClustering(
    n_clusters=3,
    affinity= 'precomputed')# data will be passed as an affinity matrix
sclust.fit(n3);
In [28]:
sclust.labels_
Out[28]:
array([2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0], dtype=int32)
```

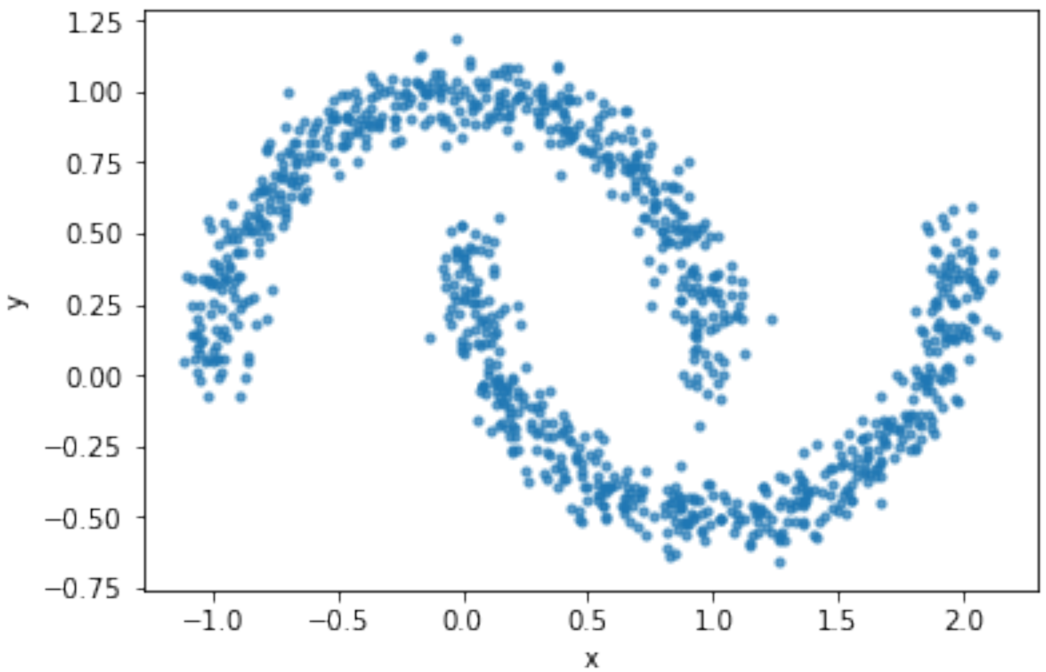
```
n3
Out[22]:
array([[0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
       [1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1],
       [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0]])
```



Based on an affinity matrix

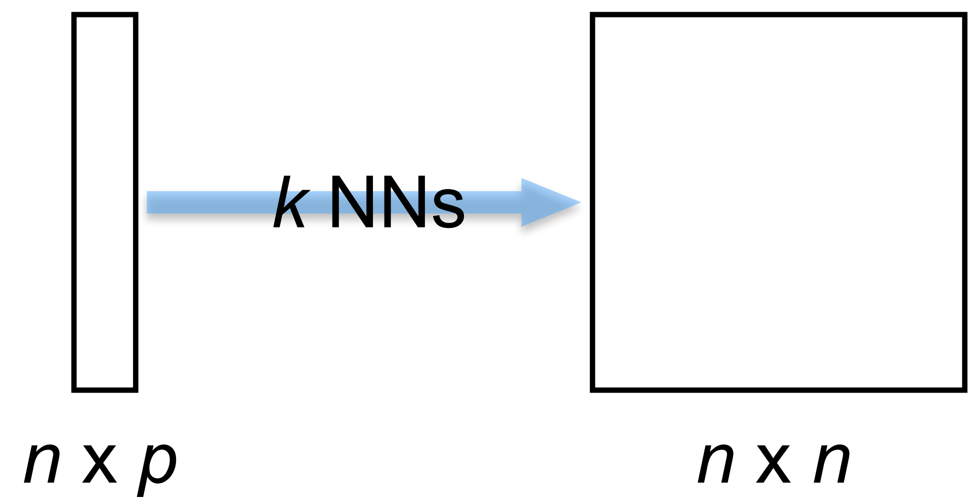
Spectral Clustering - feature vector data

- 2-D Synthetic data



How do we convert feature vector data to an affinity matrix?

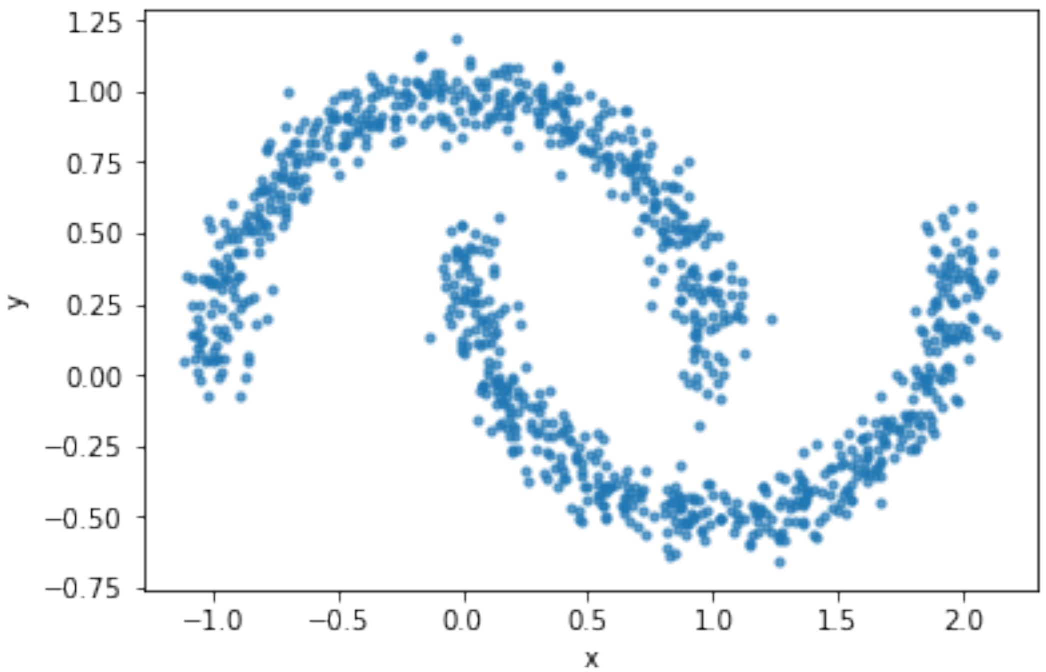
Create an affinity matrix with each item connected to it's *k* NNs



Few links between separate clusters

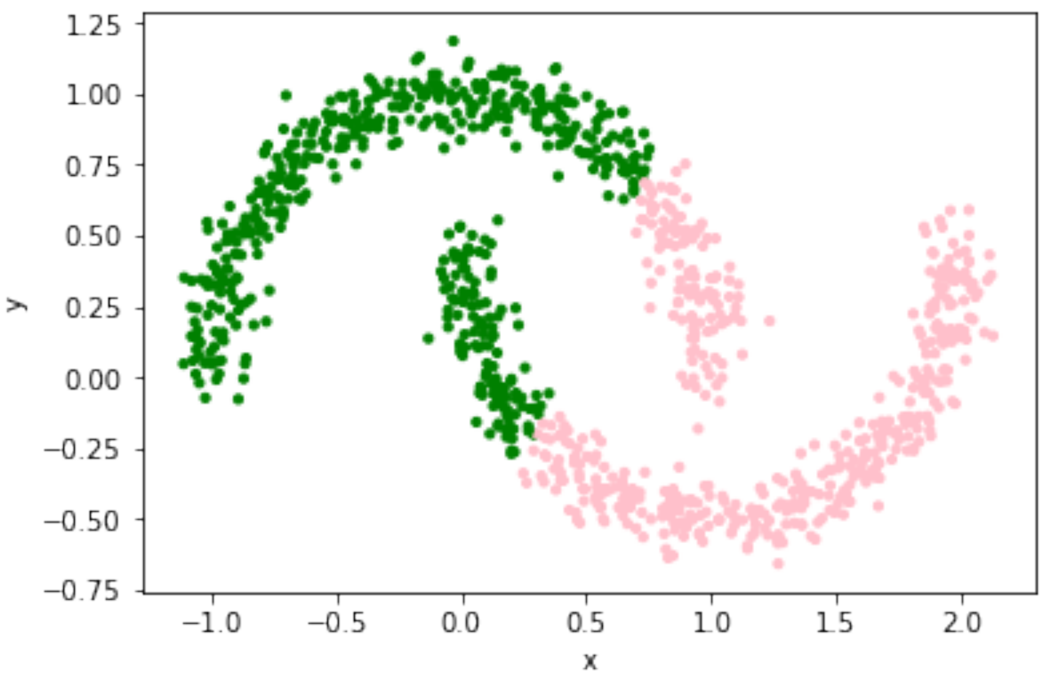
Spectral Clustering - feature vector data

- 2-D Synthetic data

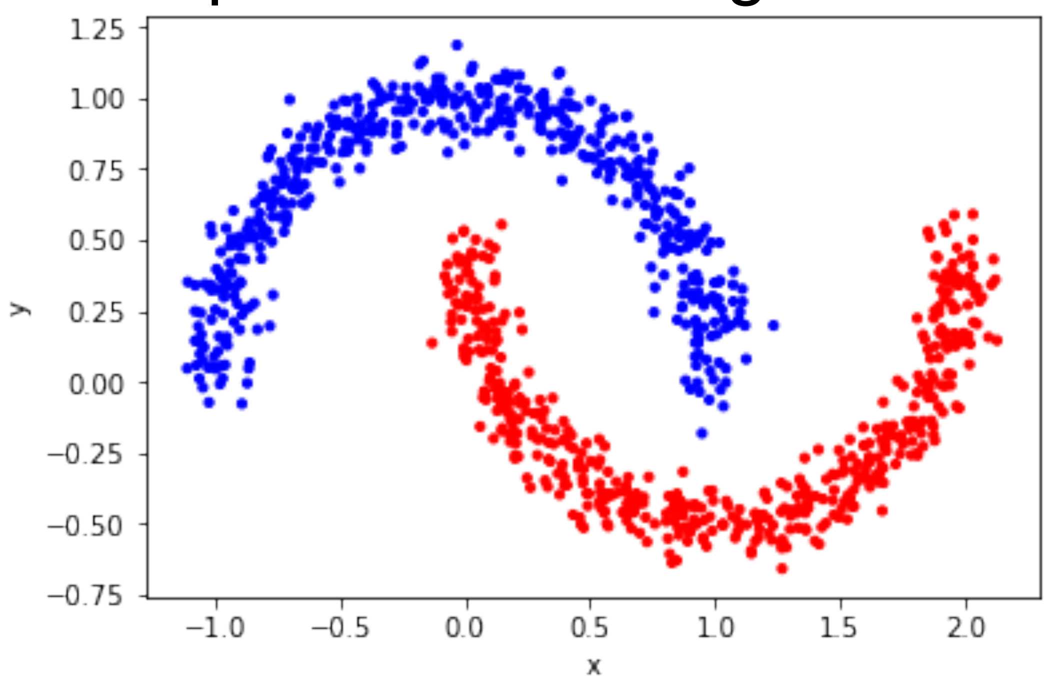


Notebook:
18 Spectral Clustering sklearn

k-Means



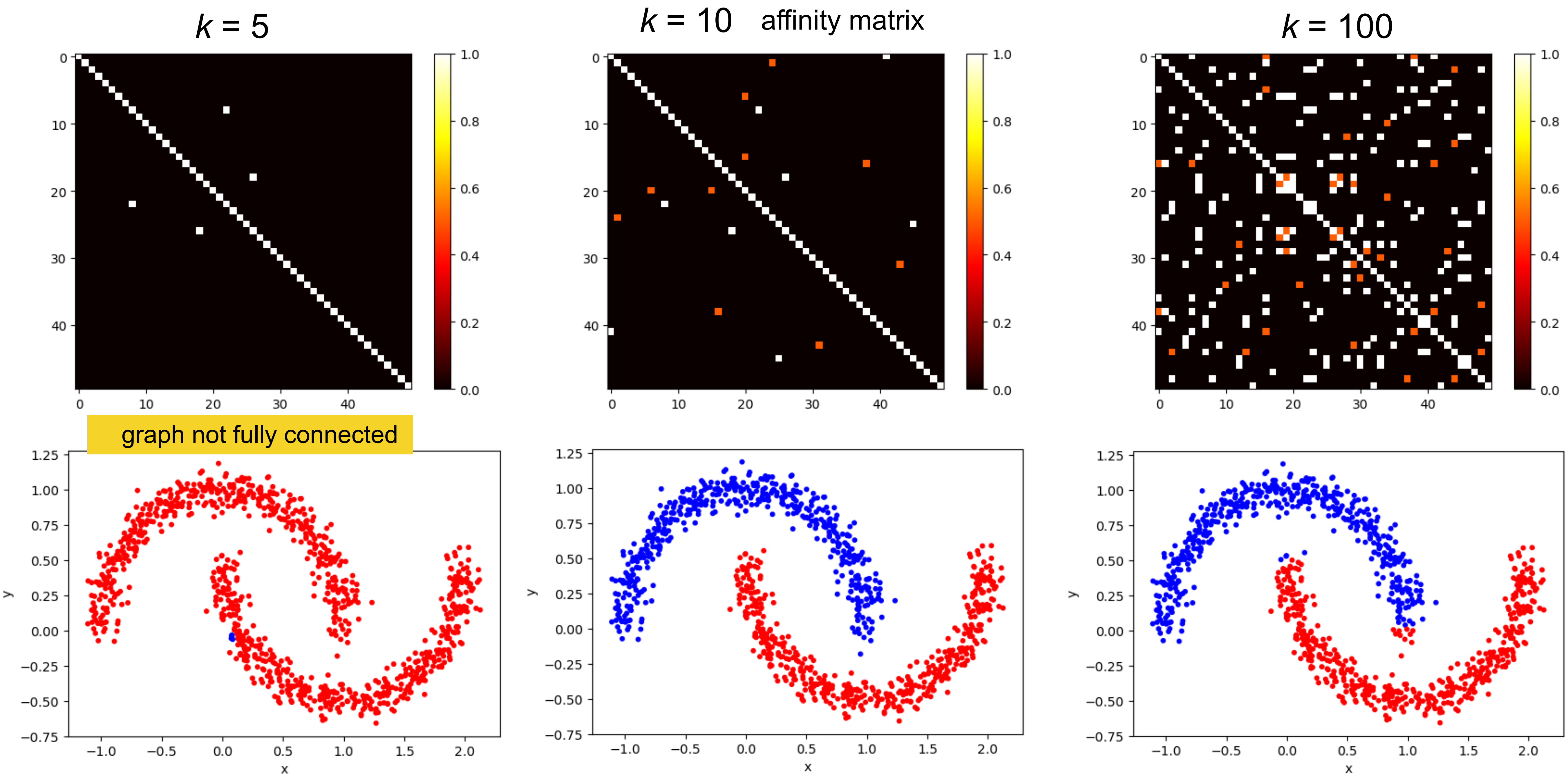
Spectral Clustering




```
from sklearn.cluster import SpectralClustering
sclust = SpectralClustering(
    n_clusters=2,
    affinity='nearest_neighbors',
    n_neighbors=10,
)
sclust.fit(moons[['x', 'y']]);
```

Spectral Clustering - feature vector data

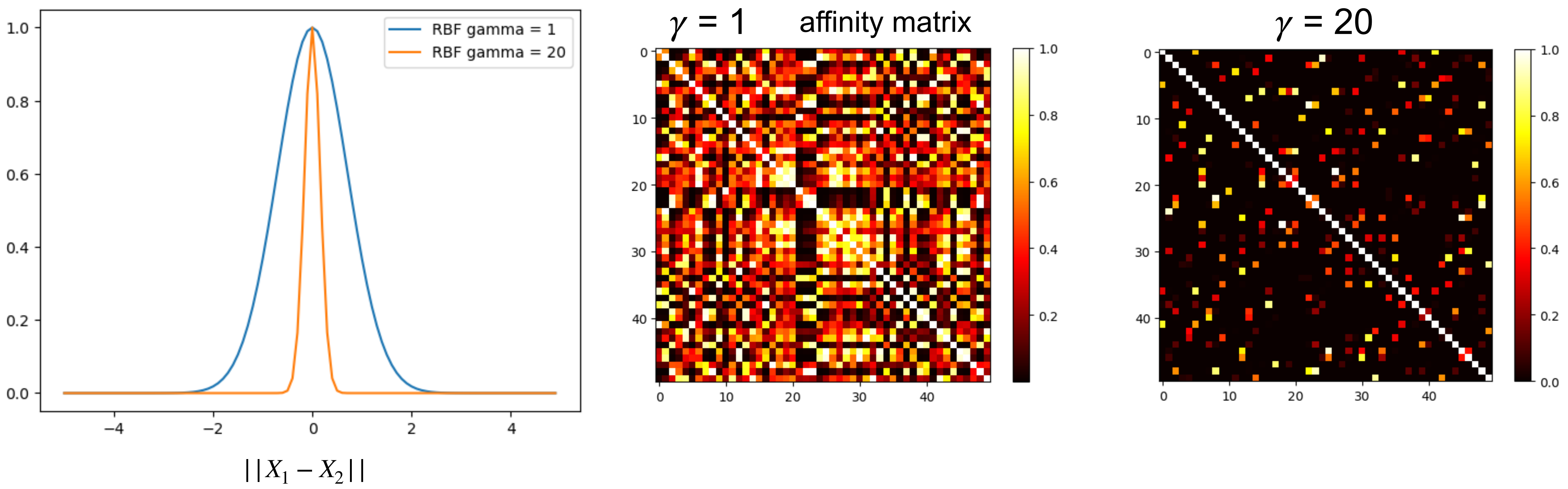
- Constructing affinity matrix using *k-NN*
  - Impact of *k*



Spectral Clustering - RBF

- scikit-learn offers a second option for producing an affinity matrix from feature vector data
  - Radial Basis Function (RBF) kernel: $K(X_1, X_2) = e^{-\frac{||X_1 - X_2||^2}{2\sigma^2}}$

    $= e^{-\gamma||X_1 - X_2||^2}$ where $\gamma = \frac{1}{2\sigma^2}$
  - γ controls the width



Spectral Clustering - Harry Potter data

	Name	Magic	Cunning	Courage	Wisdom	Temper	Group
21	'Lucius Malfoy'	88	24	10	60	9	0
13	'Gregory Goyle'	10	14	7	2	8	0
12	'Draco Malfoy'	42	22	10	12	9	0
11	'Vincent Crabbe'	10	13	8	4	7	0
17	'Cho Chang'	40	8	25	31	3	1
15	'Parvati Patil'	24	11	23	15	2	1
14	'Padma Patil'	24	9	23	13	1	1
20	'Neville Longbottom	24	9	28	15	2	1
10	'Arthur Weasley'	62	5	29	60	2	1
7	'Rubeus Hagrid'	12	11	30	8	7	1
6	'Prof. Moody'	82	20	35	69	5	2
5	'Prof. McGonagall'	95	19	29	76	5	2
4	'Prof. Snape'	85	24	19	71	7	2
3	'Prof. Dumbledore'	105	24	39	82	0	2
16	'Fleur Delacour'	59	19	36	54	6	2
1	Hermione Granger'	60	16	40	73	2	2
18	'Cedric Diggory'	58	23	40	55	2	2
9	'George Weasley'	87	13	30	22	4	3
2	'Ron Weasley'	45	14	40	22	4	3
19	'Viktor Krum'	56	22	38	30	7	3
8	'Fred Weasley'	87	13	30	22	4	3
0	'Harry Potter'	62	21	42	26	7	3

```
sclust = SpectralClustering(
    n_clusters=4,
    random_state=42,
    affinity= 'nearest_neighbors'
    n_neighbors=7,
)
sclust.fit(X_scal);
In [10]:
TT_df['Group'] = sclust.labels_
```

```
sclust.affinity_matrix_
```

```
1.0 0.0 1.0 0.0 0.5 0.0 0.5 0.0 1.0 0.5 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.5 1.0 0.0 0.0
0.0 1.0 0.5 0.5 0.0 1.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.5 0.0 1.0 0.0 0.0 0.0
1.0 0.5 1.0 0.0 0.0 0.0 0.0 0.5 1.0 1.0 0.0 0.0 0.0 0.0 0.5 0.5 1.0 1.0 0.5 0.5 1.0 0.0
0.0 0.5 0.0 1.0 0.5 1.0 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.0 1.0 0.0 0.0 0.0
0.5 0.0 0.0 0.5 1.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.5 0.0 0.0 0.0 0.5 0.0 0.0 0.5 0.0 1.0
0.0 1.0 0.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.5 0.0 0.0 0.5
0.5 1.0 0.0 0.5 1.0 1.0 1.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 1.0 0.0 0.5
0.0 0.0 0.5 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.5 1.0 0.5 1.0 0.0 0.5 0.0 0.0 1.0 0.0
1.0 0.0 1.0 0.0 0.0 0.0 0.5 0.0 1.0 1.0 0.5 0.0 0.0 0.0 0.0 0.5 0.5 1.0 0.0 0.0 0.0 0.0
0.5 0.0 1.0 0.0 0.0 0.0 0.5 0.0 1.0 1.0 0.5 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.5 0.0 0.0
0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 1.0 0.0 0.5 0.0 0.5 0.0 0.0 0.5 0.0
0.0 0.0 0.0 0.0 0.5 0.0 0.0 0.5 0.0 0.0 0.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.5 0.0 1.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 1.0 0.0 0.5 0.0 0.5 0.0 0.0 0.5 0.0
0.0 0.0 0.5 0.0 0.0 0.0 0.0 0.5 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 0.0 0.0 1.0 0.0
0.0 0.0 0.5 0.0 0.0 0.0 0.0 1.0 0.5 0.0 0.0 0.5 0.0 0.5 1.0 1.0 0.0 1.0 0.0 0.0 1.0 0.0
1.0 0.5 1.0 0.5 0.5 1.0 1.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 1.0 0.0 0.5
0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.5 1.0 0.5 1.0 0.5 0.0 0.5 1.0 1.0 0.0 1.0 0.0 0.0 1.0 0.0
0.5 1.0 0.5 1.0 0.0 0.5 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 1.0 0.0 0.0
1.0 0.0 0.5 0.0 0.5 0.0 1.0 0.0 0.0 0.5 0.0 0.0 0.5 0.0 0.0 0.0 1.0 0.0 1.0 1.0 0.0 0.5
0.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.5 0.0 0.5 1.0 1.0 0.0 1.0 0.0 0.0 1.0 0.0
0.0 0.0 0.0 0.0 1.0 [illegible] 0.5 0.0 0.0 0.5 0.0 1.0
```

How come there are entries with value 0.5?

Spectral Clustering steps

- **Inputs:**
  - Data in feature vector format or affinity matrix
  - *k:* number of clusters

- **Process:**
  1. If feature vector format generate affinity matrix (e.g. *k*-NN)
  2. Generate Laplacian matrix
  3. Get eigenvectors of Laplacian - select *k* components
     – e.g *k* eigenvectors for *k* clusters
     – *k* dimension representation of data in embedded space
  4. Cluster this *k* dimension representation using *k*-Means

Overview

- Equivalence of Clustering & Graph Partitioning
- Graph Partitioning using Eigenvectors
- Spectral Clustering
- Spectral Clustering on Feature-Vector data
- Spectral Clustering in scikit learn