Clustering validation

Is there any real clustering? How good is it?

- Book: Chapter 6.9
- External material: Halkidi et al. (2002): Cluster Validity Methods: Part I. ACM SIGMOD Record 31(2): 40–45. https://doi.org/10.1145/565117.565124

Three similar problems

1. Clustering tendency: is there any clustering in data presented with certain features?
2. Determining number of clusters (or other parameters)
3. Evaluating goodness of clustering
   - compare different methods
   - compare against classification

All three depend on the **clustering objective**!

- assumptions on clusters (e.g., compactness, shape)
- separation between clusters

Evaluating goodness of clustering

1. **Internal criteria**
   - validity indices, similar to objective functions
   - do not work, if clustering had a different objective!
   - can be used to i) evaluate a single clustering or ii) compare clusterings (as **relative indices**)
2. **External criteria**
   - compare clustering to a predefined classification
   - classes may not reflect natural clusters
3. **Statistical hypothesis testing**
   - maybe the most sound approach, but computationally demanding

Internal validity indices

- indices assume some clustering objective $\rightarrow$ reward methods with the same objective
  - even a good clustering can get a bad score if a different objective!
  - many indices assume/favor spherical or convex clusters
- best for comparing similar algorithms and tuning parameters
- Some popular indices:
  - **Average silhouette**
  - **Calinski-Harabasz index**
  - **Davies-Bouldin index**

Silhouette index

Silhouette of a point $\mathbf{x}$ is

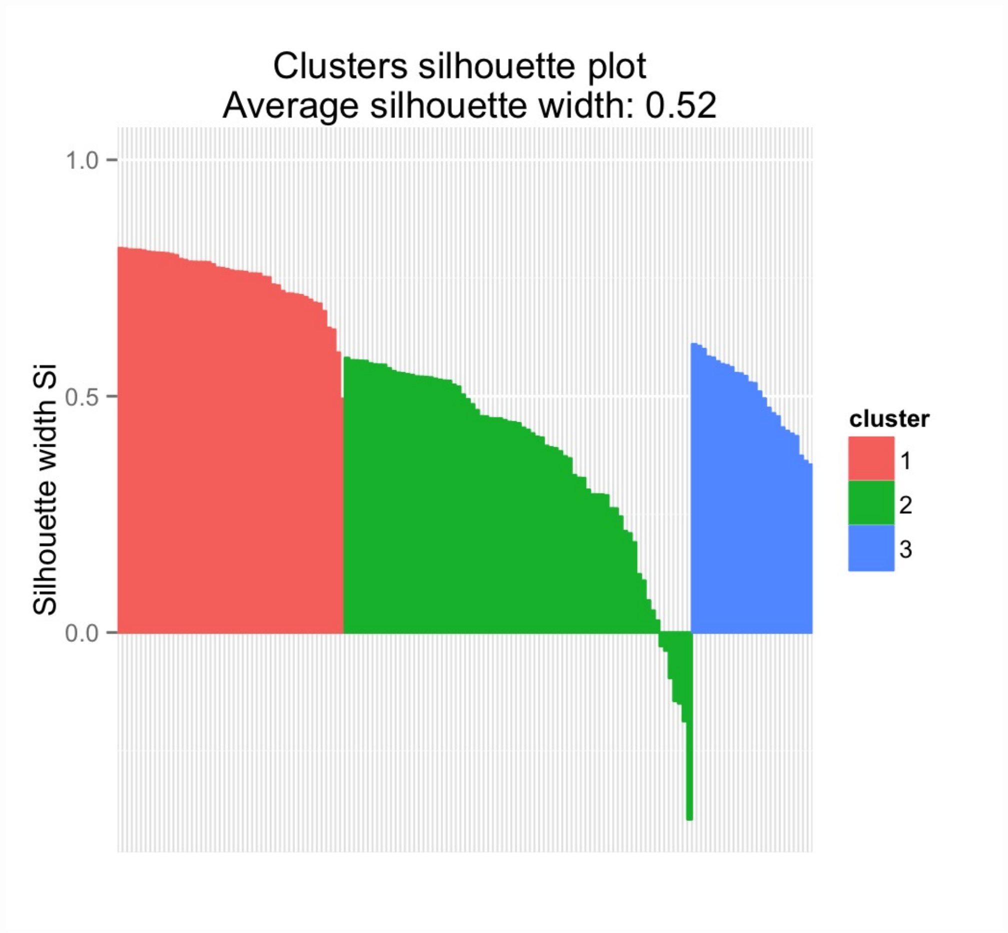
$$S(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \text{ a cluster of its own} \\ \frac{b-a}{\max\{a,b\}} & \text{otherwise} \end{cases}$$

$a = avg\{d(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in C, \mathbf{y} \in C\}$
$b = \min_q avg\{d(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in C, \mathbf{y} \in C_q, C \neq C_q\}$

$\approx$ how closely $\mathbf{x}$ matches its own cluster and how loosely the neighbouring cluster

- $S(\mathbf{x}) \in [-1, 1]$, **high values good**
- **Average silhouette** describes goodness of entire clustering
- flexible: any distance function d

Example: Silhouette of points



What negative values mean?

$$S(\mathbf{x}) = \begin{cases} 0 & \text{if singleton} \\ \frac{b-a}{\max\{a,b\}} & \text{otherwise} \end{cases}$$

$a = avg\{d(\mathbf{x},\mathbf{y}) \mid \mathbf{x} \in C, \mathbf{y} \in C\}$
$b = \min_q avg\{d(\mathbf{x},\mathbf{y}) \mid \mathbf{x} \in C, \mathbf{y} \in C_q, C \neq C_q\}$

image source `http://www.sthda.com/english/wiki/wiki.php?id_contents=7952`

Calinski-Harabasz index

$$S_{CH} = \frac{(n-K)B}{(K-1)W}$$

- **between-cluster variance** $B = \sum_{i=1}^{K} |C_i| L_2^2(\mathbf{c}_i, \mathbf{m})$, where $\mathbf{m}$ is the mean of the whole data
- **within-cluster variance** $W = \sum_{i=1}^{K} \sum_{\mathbf{x} \in C_i} L_2^2(\mathbf{x}, \mathbf{c}_i)$
- requires $K \geq 2$
- range $[0, \infty[$, **high values good**
- When could you get value 0?

Calinski-Harabasz index (cont'd)

$$S_{CH} = \frac{(n-K)B}{(K-1)W} = \frac{(n-K)\sum_{i=1}^{K}|C_i|L_2^2(\mathbf{c}_i,\mathbf{m})}{(K-1)\sum_{i=1}^{K}\sum_{\mathbf{x}\in C_i}L_2^2(\mathbf{x},\mathbf{c}_i)}$$

Note: $W = SSE(\mathbf{C})$. K-means criterion minimizes $W \Rightarrow$ maximizes B, because

$$\sum_{\mathbf{x}\in\mathcal{D}} L_2^2(\mathbf{x},\mathbf{m}) = \sum_{i=1}^{K}\sum_{\mathbf{x}\in C_i} L_2^2(\mathbf{x},\mathbf{c}_i)^2 + \sum_{i=1}^{K}|C_i|L_2^2(\mathbf{c}_i,\mathbf{m})$$

$\Rightarrow S_{CH}$ favours especially K-means!

Important: need to use L_2 in clustering!

Davies-Bouldin index

$$S_{DB} = \frac{1}{K}\sum_{i=1}^{K} \max_{j \neq i} \frac{S_i + S_j}{D_{ij}} \quad \text{, where}$$

- $S_i = \left(\frac{1}{|C_i|}\sum_{\mathbf{x}\in C_i} L_p^q(\mathbf{x}, \mathbf{c}_i)\right)^{\frac{1}{q}}$ measures dispersion of C_i
  - usually $q = 2$ (stdev of distances)
  - if $q = 1$, average distances
- $D_{ij} = L_p(\mathbf{c}_i, \mathbf{c}_j)$ measures separation between C_i and C_j
- max: for each C_i, evaluate relation to most problematic C_j
- possible to take avg instead of max

Important: use the same L_p as the clustering algorithm!

Davies-Bouldin index (cont'd)

$$S_{DB} = \frac{1}{K} \sum_{i=1}^{K} \max_{j \neq i} \frac{S_i + S_j}{D_{ij}} \quad \text{, where}$$

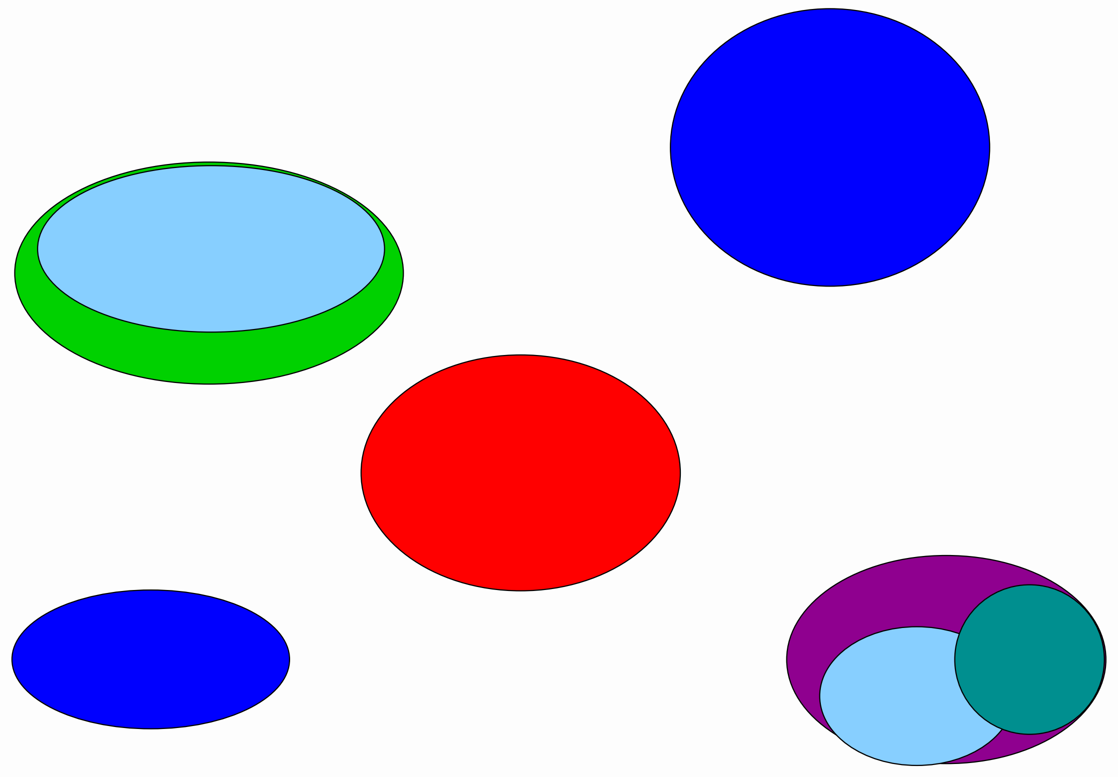
$S_i = \left(\frac{1}{|C_i|} \sum_{\mathbf{x} \in C_i} L_p^q(\mathbf{x}, \mathbf{c}_i) \right)^{\frac{1}{q}}$ and $D_{ij} = L_p(\mathbf{c}_i, \mathbf{c}_j)$

- range $[0, \infty[$, **small values good**
- When could you get value 0?

Possible strategies when S_{DB} used to determine K:

- restrict number of singletons (e.g., 0 or a few)
- define $S_i = a$ for some large a, when $|C_i| = 1$

External validation: Compare clustering against predefined classification



🟩 class 1

🟦 class 2

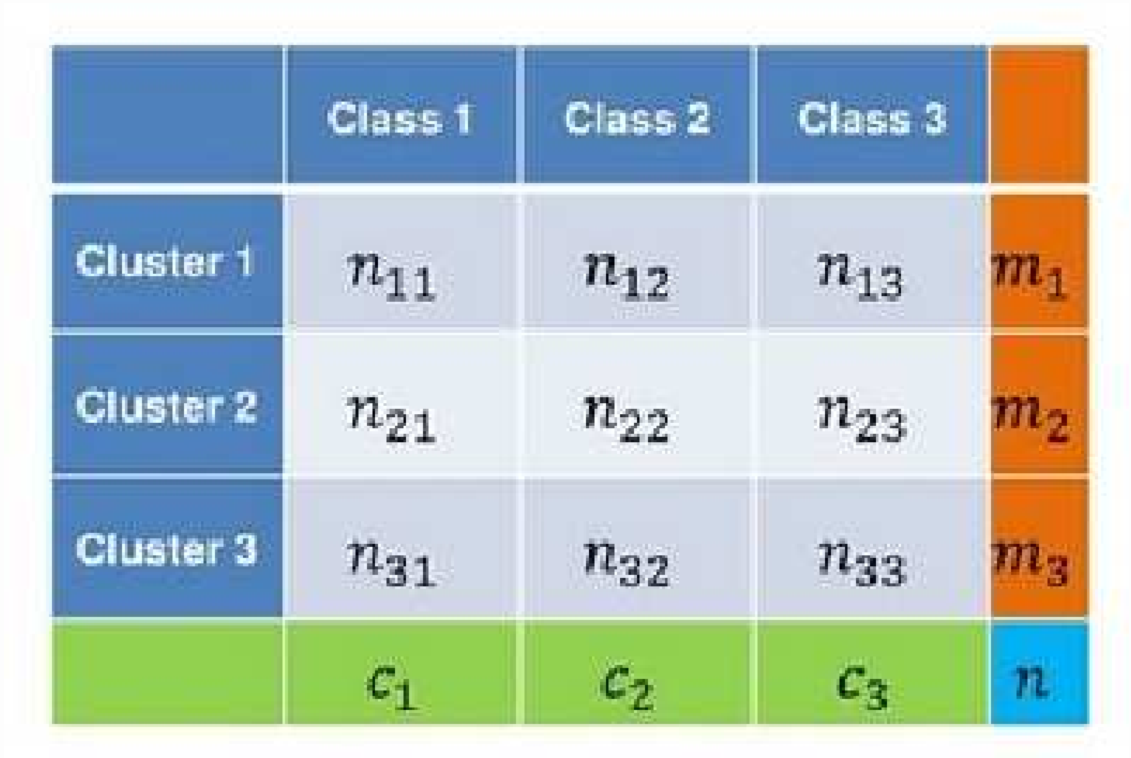
🟥 class 3

🟦 class 4

🟦 class 5

🟪 class 6

A confusion matrix: clustering vs. classification



	Class 1	Class 2	Class 3	
Cluster 1	n_{11}	n_{12}	n_{13}	m_1
Cluster 2	n_{21}	n_{22}	n_{23}	m_2
Cluster 3	n_{31}	n_{32}	n_{33}	m_3
	c_1	c_2	c_3	n

image source Cunnigham `https://slideplayer.com/slide/14318989/`

External validation

Given clustering $C_1, \ldots, C_K$ and classification $D_1, \ldots, D_q$. Many validation indices! E.g.,

- **purity**

$$Pur(C) = \frac{1}{n} \sum_{i=1}^{K} \max_j |C_i \cap D_j|$$

  - be careful! (increases with K)
- **normalized mutual information** NMI (robust, independent of K)
- **Rand index**

Normalized mutual information

Normalized mutual information by Strehl and Ghosh (2003):

$$NMI = \frac{I(C,D)}{\sqrt{H(C)H(D)}}$$

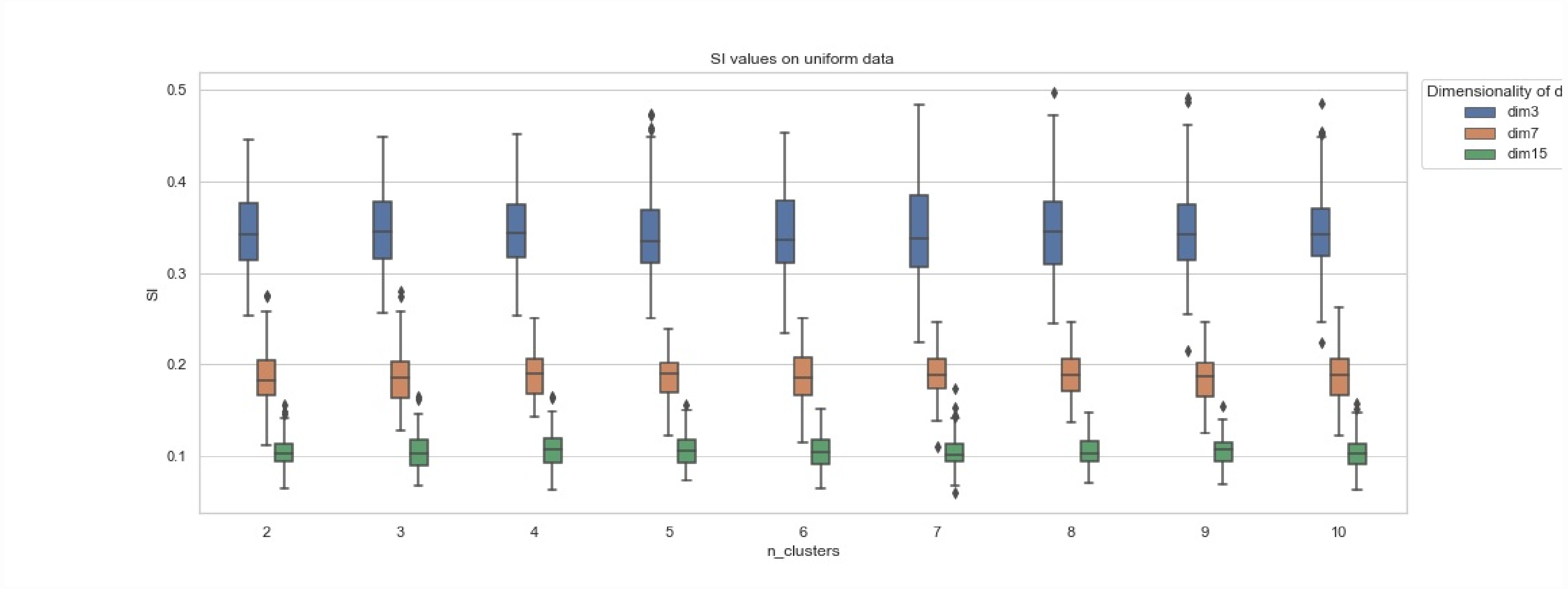
mutual information $I = \sum_{C_i \in C} \sum_{D_j \in D} P(C_i, D_j) \log \frac{P(C_i, D_j)}{P(C_i)P(D_j)}$
entropy $H(C) = -\sum_{C_i \in C} P(C_i) \log P(C_i)$

+ does not depend on the number of clusters
– many singleton clusters can cause problems

Note: Also other variants of normalized mutual information, give always equation and/or reference what you use!

Statistical hypothesis testing: motivation

SI can be pretty good even for random data!



- each feature generated independently from uniform distribution
- 100 randomizations
- K-means repeated 100 times $\rightarrow$ best result for each K

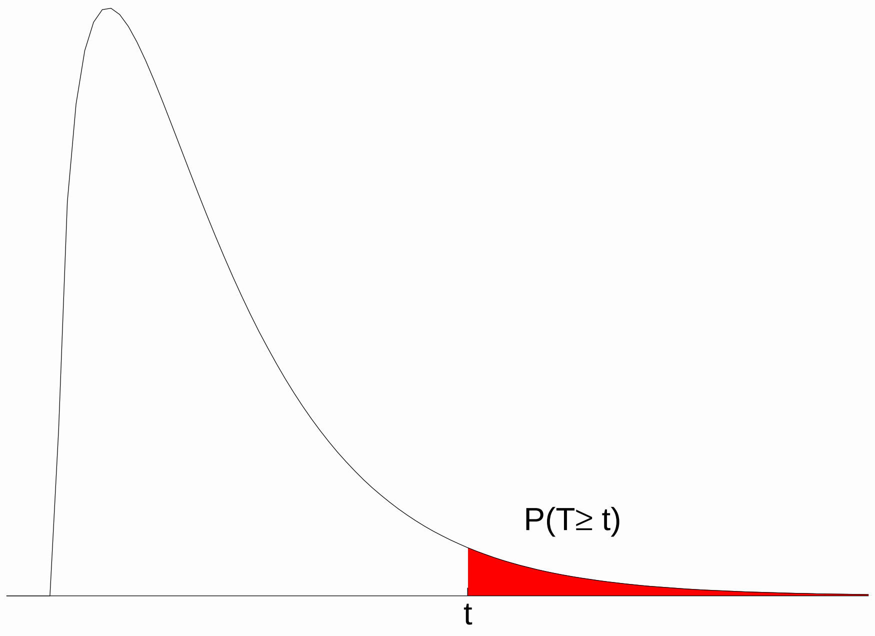
Experiment by Georgy Ananov for MDM 2023

Statistical hypothesis testing

Procedure:

1. decide a **null hypothesis** H_0 to test
   - describes the state where there isn't any clustering
   - e.g., H_0: All sets of n locations in certain region are equally likely.
2. decide a **test statistic** T
   - may be a validity index
3. What is the probability to obtain at least as good test statistic values as in data (where $T = t$) if H_0 was true?

Statistical hypothesis testing



Assume that large T value good

Idea: If $P(T \geq t)$ very small $\Rightarrow$ unlikely that the observed clustering had occurred by chance

- $P(T \geq t)$ is the **p-value** that can be used as a significance measure

Statistical hypothesis testing

Problem: How to evaluate p-value? (T's distribution seldom known!)

- often by Monte Carlo experiments (randomization tests):
  - generate random data sets fulfilling H_0, cluster them and evaluate T
  - p-value $\approx$ proportion of random sets that obtained $T \geq t$ (if large T good)
- – computationally demanding (a lot of simulations!)
- – many alternatives for H_0s and Ts

Other evaluation: What the clustering reveals?

- Look at cluster sizes (e.g., C_1: $n - 2$ data points and C_2: 2 points – likely outliers!)
- How do the clusters differ? (selected and external features)
  - e.g., rats clustered by body measurements (weight, tail and body length, organ weights)
  - 2 clusters: big and small rats
  - vs. 3 clusters: C_1: young or sick rats, C_2: pregnant or nursing females, C_3: other adults
- Are all clusters clear? (e.g., C_1 and C_3 intermingled, C_2 separate)

Summary

- Remember validation, but be cautious!
  - even random data can produce clusterings, but they seldom pass validation
  - problem: indices biased or do not reflect the underlying clustering
  - try always more than one validation technique
- Objective, distance measure, clustering method and validation should match!

Sources and further reading

- Halkidi et al. (2001): On clustering validation techniques, Journal of Intelligent Information Systems 17: 107–145. https://www.researchgate.net/publication/2500099_On_Clustering_Validation_Techniques
- Jain and Dubes (1988): Algorithms for clustering data, Ch 4.
- Gan, Ma, Wu (2007): Data clustering - theory, algorithms, and applications, Ch 17, https://www.researchgate.net/publication/220694937_Data_Clustering_Theory_Algorithms_and_Applications

Sources and further reading

- Vargha, Bergman, Takacs: Performing Cluster Analysis Within a Person-Oriented Context: Some Methods for Evaluating the Quality of Cluster Solutions. Journal of Person-Oriented Research, 2: 78-86, 2016.

Spectral clustering

Contents:

- Matrices from the similarity graph
- 1D spectral embedding & clustering
- Unnormalized and normalized spectral clustering
- Important choices

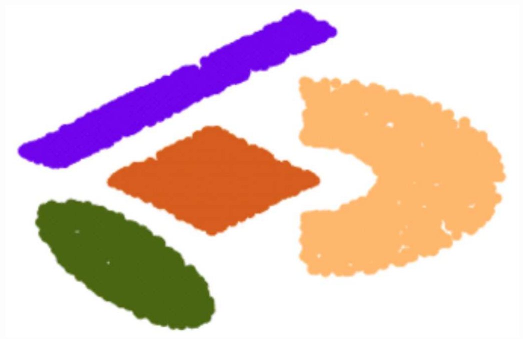
Book: Sections 2.4.4.3, 6.7, 19.3.4

Recommended external material:
von Luxburg (2007): A Tutorial on Spectral Clustering.

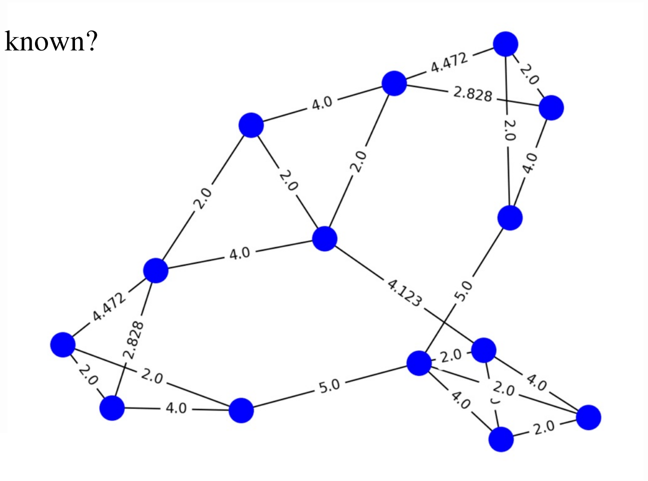
Presemo: `https://presemo.aalto.fi/mdm2023`

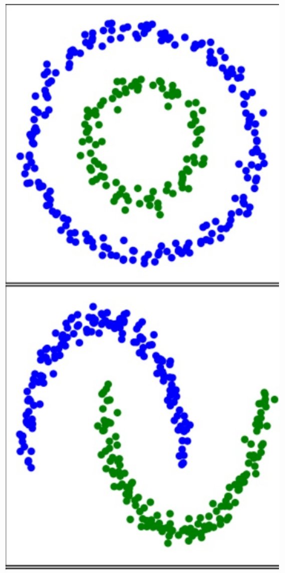
Recap: How could you cluster these?

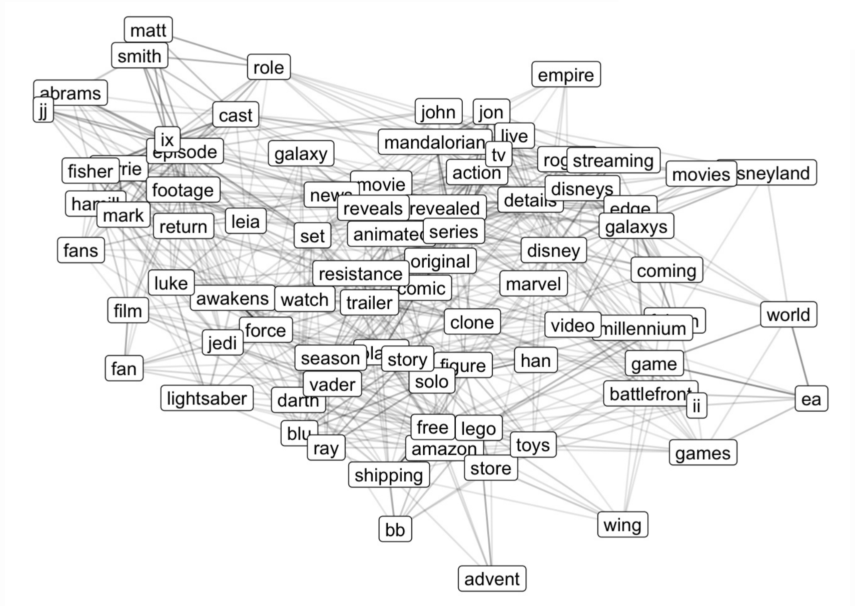
Arbitrary shapes?



Only similarity graphs known?







Images: White (2019) https://www.markhw.com/blog/word−similarity−graphs, Cooper (2021) https://spin.atomicobject.com/2021/09/07/spectral−clustering/, Park & Kim (2020) https://doi.org/10.1115/DETC2020−22642, Scikit−learn documentation https://ogrisel.github.io/scikit−learn.org/sklearn−tutorial/auto_examples/cluster/plot_cluster_comparison.html

General idea of graph-based clustering

1. Present data as a similarity (neighbourhood) graph $\mathbf{G}$
2. Cluster nodes of G with a network clustering or community detection algorithm

+ can detect arbitrary-shaped clusters

+ even varying cluster densities (given k nearest neighbour similarity graph)

+ for any data type (if pairwise similarity/distance defined)

– computationally costly

– many parameter choices

Spectral clustering: Idea

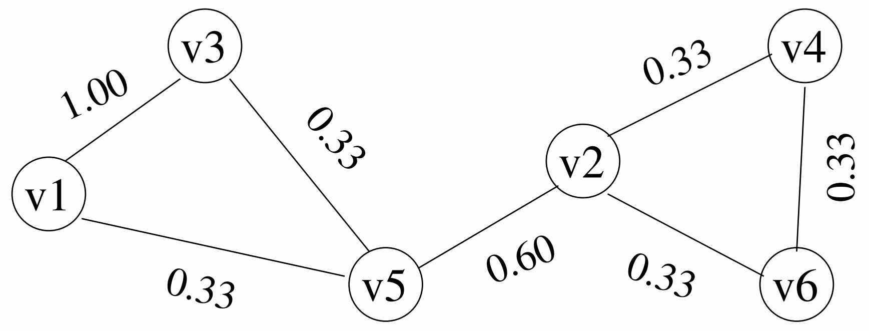
1. Create **similarity graph G**
   - node v_i for the ith data point ($i = 1, \ldots, n$)
   - edge weight w_{ij} = similarity between nodes v_i and v_j
2. **Present data in** (low-dimensional) **vector space** (i.e., find vectors $\mathbf{y}_1, \ldots, \mathbf{y}_n$) such that **local similarity/clustering structure** is preserved
   - idea: choose $\mathbf{Y}$ to minimize $cost(\mathbf{G}, \mathbf{Y}) = \sum \sum w_{ij} L_2^2(\mathbf{y}_i, \mathbf{y}_j)$
   - intuition: large w_{ij} tends to produce small $d(\mathbf{y}_i, \mathbf{y}_j)$
   - $\rightarrow$ easy after reformulation with a Laplacian matrix
3. **Cluster $\mathbf{y}_i$**s with K-means (etc.)

What is needed?

From $\mathbf{G}$ derive:

1. weight matrix $\mathbf{W}$
2. diagonal degree matrix $\mathbf{\Lambda}$
3. Laplacian matrix $\mathbf{L} = \mathbf{\Lambda} - \mathbf{W}$
4. normalized Laplacian matrices $\mathbf{L}_{rw}$, $\mathbf{L}_{sym}$ if desired)

Similarity graph and weight matrix $\mathbf{W}$



$$\begin{bmatrix} 0.00 & 0.00 & 1.00 & 0.00 & 0.33 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.33 & 0.60 & 0.33 \\ 1.00 & 0.00 & 0.00 & 0.00 & 0.33 & 0.00 \\ 0.00 & 0.33 & 0.00 & 0.00 & 0.00 & 0.33 \\ 0.33 & 0.60 & 0.33 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.33 & 0.00 & 0.33 & 0.00 & 0.00 \end{bmatrix}$$

- $\mathbf{W}$ adjacency matrix of a weighted graph
- $W_{ij} = w_{ij}$ (similarity between nodes v_i and v_j)
- if unweighted graph, use weights 1 (edge) or 0

Diagonal degree matrix $\mathbf{\Lambda}$ *(*$\Lambda_{ii} = \sum_{j=1}^{n} W_{ij}$*)*

$$\mathbf{\Lambda} = \begin{bmatrix} 1.33 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.26 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.33 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.66 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.26 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.66 \end{bmatrix}$$

$$\mathbf{W} = \begin{bmatrix} 0.00 & 0.00 & 1.00 & 0.00 & 0.33 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.33 & 0.60 & 0.33 \\ 1.00 & 0.00 & 0.00 & 0.00 & 0.33 & 0.00 \\ 0.00 & 0.33 & 0.00 & 0.00 & 0.00 & 0.33 \\ 0.33 & 0.60 & 0.33 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.33 & 0.00 & 0.33 & 0.00 & 0.00 \end{bmatrix}$$

(Unnormalized) Laplacian matrix $\mathbf{L} = \mathbf{\Lambda} - \mathbf{W}$

$$\mathbf{L} = \begin{bmatrix} 1.33 & 0.00 & -1.00 & 0.00 & -0.33 & 0.00 \\ 0.00 & 1.26 & 0.00 & -0.33 & -0.60 & -0.33 \\ -1.00 & 0.00 & 1.33 & 0.00 & -0.33 & 0.00 \\ 0.00 & -0.33 & 0.00 & 0.66 & 0.00 & -0.33 \\ -0.33 & -0.60 & -0.33 & 0.00 & 1.26 & 0.00 \\ 0.00 & -0.33 & 0.00 & -0.33 & 0.00 & 0.66 \end{bmatrix}$$

$\Rightarrow$ normalized Laplacian matrices:

- Random-walk Laplacian $\mathbf{L}_{rw} = \mathbf{\Lambda}^{-1}\mathbf{L}$
- Symmetric Laplacian $\mathbf{L}_{sym} = \mathbf{\Lambda}^{-0.5}\mathbf{L}\mathbf{\Lambda}^{-0.5}$

Idea of 1D spectral embedding & clustering

Goal: find embedding $\mathbf{y} = (y_1, \ldots, y_n)^T$, where each y_i corresponds v_i and $cost(G, \mathbf{y})$ minimal.

$$\boxed{cost(G, \mathbf{y}) = \sum \sum w_{ij}(y_i - y_j)^2 = 2\mathbf{y}^T \mathbf{L} \mathbf{y}}$$

- we want to avoid trivial solution $\forall i : y_i = 0 \rightarrow$
- scaling constraint (e.g.) $\mathbf{y}^T \mathbf{y} = 1$ (i.e., $\sum_i y_i^2 = 1$)
- $\mathbf{L}$ is positive semidefinite (eigenvalues λ_i real, $\lambda_i \geq 0$)
- solution smallest non-trivial eigenvector of $\mathbf{L}$

Extra: Why eigenvectors $\mathbf{y}$ of $\mathbf{L}$ would be the solution?

Task: Find $\mathbf{y}$ such that $2\mathbf{y}^T\mathbf{L}\mathbf{y}$ minimal given constraint $\mathbf{y}^T\mathbf{y} = 1$

Method of Lagrange multipliers:

1. Reformulate as a Lagrangian function $\mathcal{L}(\mathbf{y}, \lambda) = \mathbf{y}^T\mathbf{L}\mathbf{y} - \lambda(\mathbf{y}^T\mathbf{y} - 1)$
2. Set the partial derivatives (with respect to $\mathbf{y}$ and λ) as 0
3. Reduces to $\mathbf{L}\mathbf{y} = \lambda\mathbf{y}$ **Eigenvalue & -vector definition!**

Idea of 1D spectral embedding & clustering

- solution smallest non-trivial eigenvector $\mathbf{y}$ of $\mathbf{L}$
- $cost = 2\mathbf{y}^T\mathbf{L}\mathbf{y} = 2\mathbf{y}^T\lambda\mathbf{y} = 2\lambda(y_1^2 + \ldots + y_n^2) = 2\lambda$ (λ eigenvalue)
- $cost$ minimal, when λ minimal (recall $\mathbf{y}^T\mathbf{y} = 1$)
- but **skip trivial solution** $\lambda = 0$ with $\mathbf{y}$ (proportional to) $\mathbf{1} = (1, \ldots, 1)^T$
  - exists always when $\mathbf{G}$ connected
- $\rightarrow$ optimal solution **eigenvector corresponding to the 2nd smallest** λ
- cluster elements of $\mathbf{y}$ with K-means

Example

Unnormalized Laplacian L

$$\begin{bmatrix} 1.33 & 0.00 & -1.00 & 0.00 & -0.33 & 0.00 \\ 0.00 & 1.26 & 0.00 & -0.33 & -0.60 & -0.33 \\ -1.00 & 0.00 & 1.33 & 0.00 & -0.33 & 0.00 \\ 0.00 & -0.33 & 0.00 & 0.66 & 0.00 & -0.33 \\ -0.33 & -0.60 & -0.33 & 0.00 & 1.26 & 0.00 \\ 0.00 & -0.33 & 0.00 & -0.33 & 0.00 & 0.66 \end{bmatrix}$$

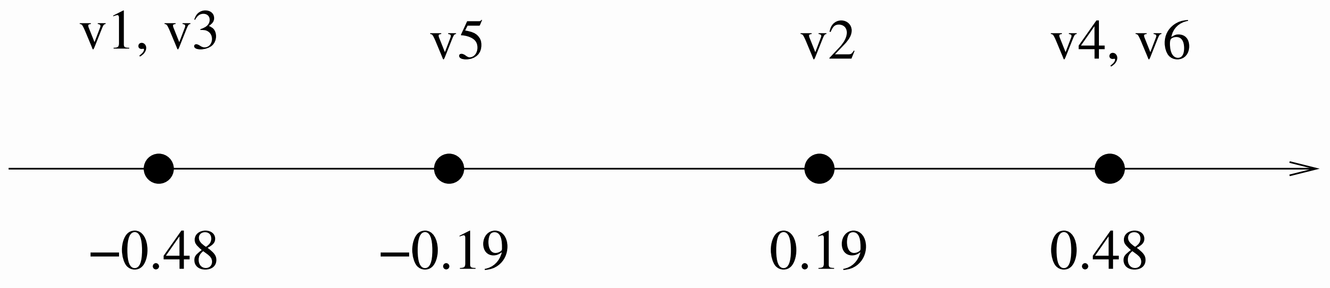
Eigenvalues:

$\approx$0, 0.20, 0.99, 0.99, 1.99, 2.33 *

Second smallest eigenvector:

$(0.48, -0.19, 0.48, -0.48, 0.19, -0.48)^T$

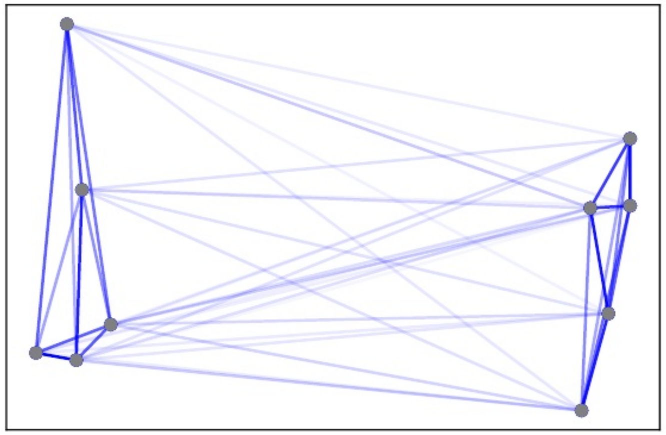
The new representation can be clustered by K-means:



* 1st eigenvalue 1.9e-16 due to imprecision (should be 0)

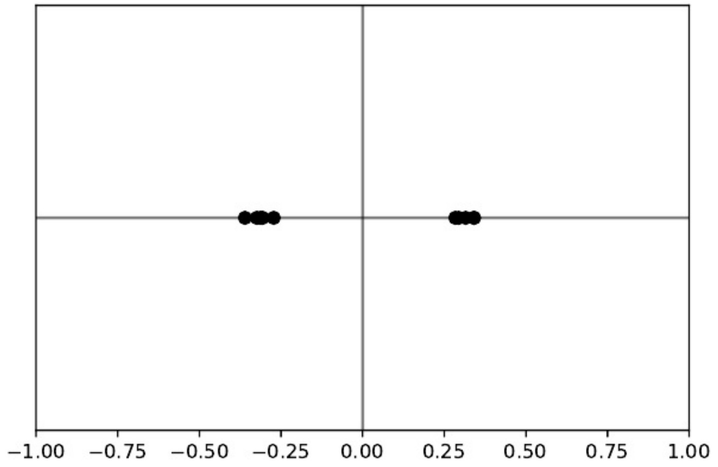
Another example with 1D embedding

Fully connected weighted graph.



Eigenvector:

$(0.32, 0.34, 0.28, 0.34, 0.29, -0.32, -0.27, -0.31, -0.36, -0.31)^T$



$n = 10$

Example by Bruno Ordozgoiti, MDM 2020

Generalization with multidimensional embedding

Unnormalized spectral clustering
Input: Graph $\mathbf{G}$ with adjacency matrix $\mathbf{W}$, number of clusters K.

1. Compute the Laplacian $\mathbf{L} = \mathbf{\Lambda} - \mathbf{W}$
2. Compute the **eigenvectors** $\mathbf{y}_1, \ldots, \mathbf{y}_k$ of $\mathbf{L}$ corresponding to the k smallest eigenvalues (excluding $\lambda = 0$)
3. Present the data as matrix $\mathbf{Y}$ whose columns are $\mathbf{y}_1, \ldots, \mathbf{y}_k$.
4. Cluster $\mathbf{Y}$ with K-means.

Note: Usually $k = K$ or $k < K$. Eigengap $|\lambda_{k+1} - \lambda_k|$ can be used to choose k.

Eigengap heuristic for choosing k

Choose k such that $\lambda_1, \ldots, \lambda_k$ small but λ_{k+1} relatively large.

clear gap no gap

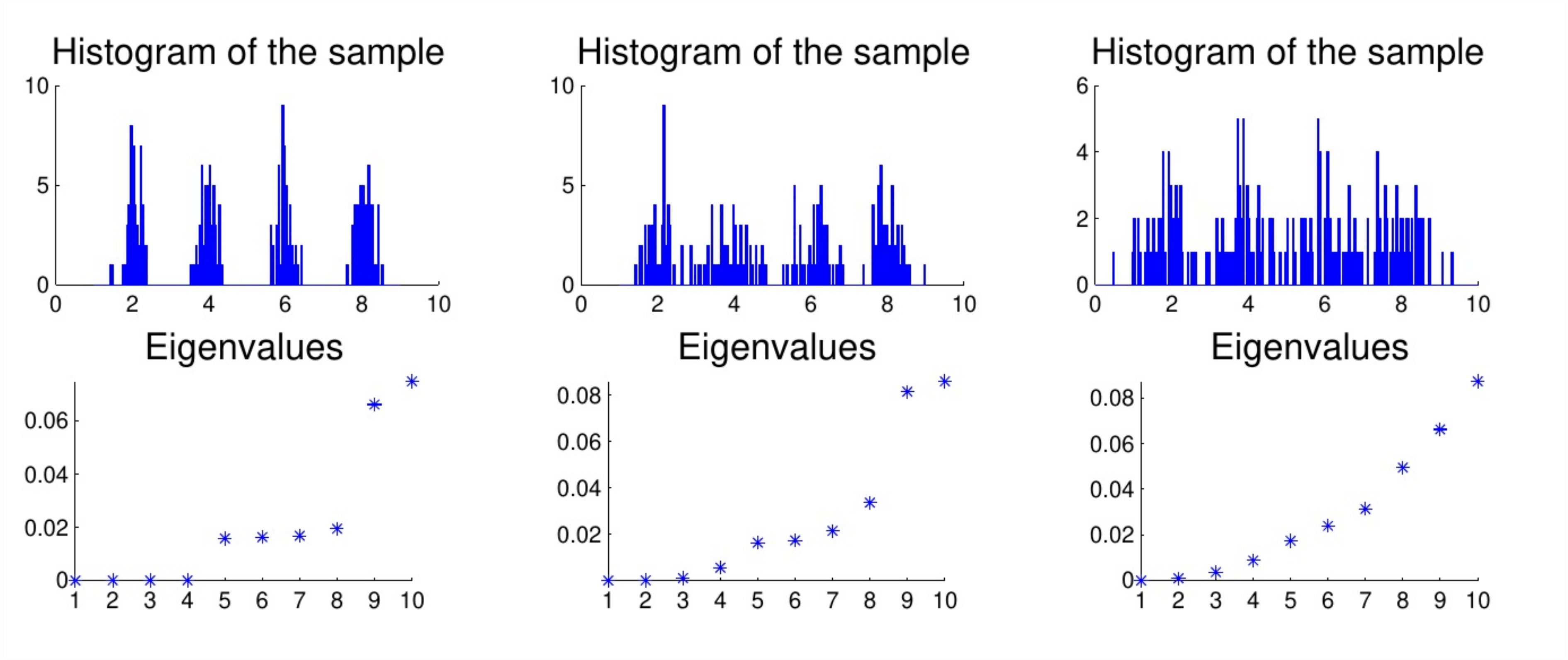


Image source: Fig 4 by von Luxburg (2006)

Normalized spectral clustering using random walk Laplacian L_{rw}

Input: Graph $\mathbf{G}$ with adjacency matrix $\mathbf{W}$, number of clusters K.

1. Compute the **random walk** Laplacian $\mathbf{L}_{rw} = \mathbf{\Lambda}^{-1}L$
2. Compute the right eigenvectors $\mathbf{y}_1, \ldots, \mathbf{y}_k$ of $\mathbf{L}_{rw}$ corresponding to the k smallest eigenvalues (excluding $\lambda = 0$)
3. Present the data as matrix $\mathbf{Y}$ whose columns are $\mathbf{y}_1, \ldots, \mathbf{y}_k$.
4. Normalize the columns of $\mathbf{Y}$ to unit norm.
5. Cluster $\mathbf{Y}$ with K-means.

Normalized spectral clustering using symmetric normalized Laplacian $\mathbf{L}_{sym}$

Input: Graph $\mathbf{G}$ with adjacency matrix $\mathbf{W}$, number of clusters K.

1. Compute the **symmetric normalized** Laplacian $\mathbf{L}_{sym} = \mathbf{\Lambda}^{-1/2} L \mathbf{\Lambda}^{-1/2}$
2. Compute the eigenvectors $\mathbf{y}_1, \ldots, \mathbf{y}_k$ of $\mathbf{L}_{sym}$ corresponding to the k smallest eigenvalues (excluding $\lambda = 0$)
3. Present the data as matrix $\mathbf{Y}$ whose columns are $\mathbf{y}_1, \ldots, \mathbf{y}_k$.
4. Normalize the rows of $\mathbf{Y}$ to unit norm.
5. Cluster $\mathbf{Y}$ with K-means.

Important choices

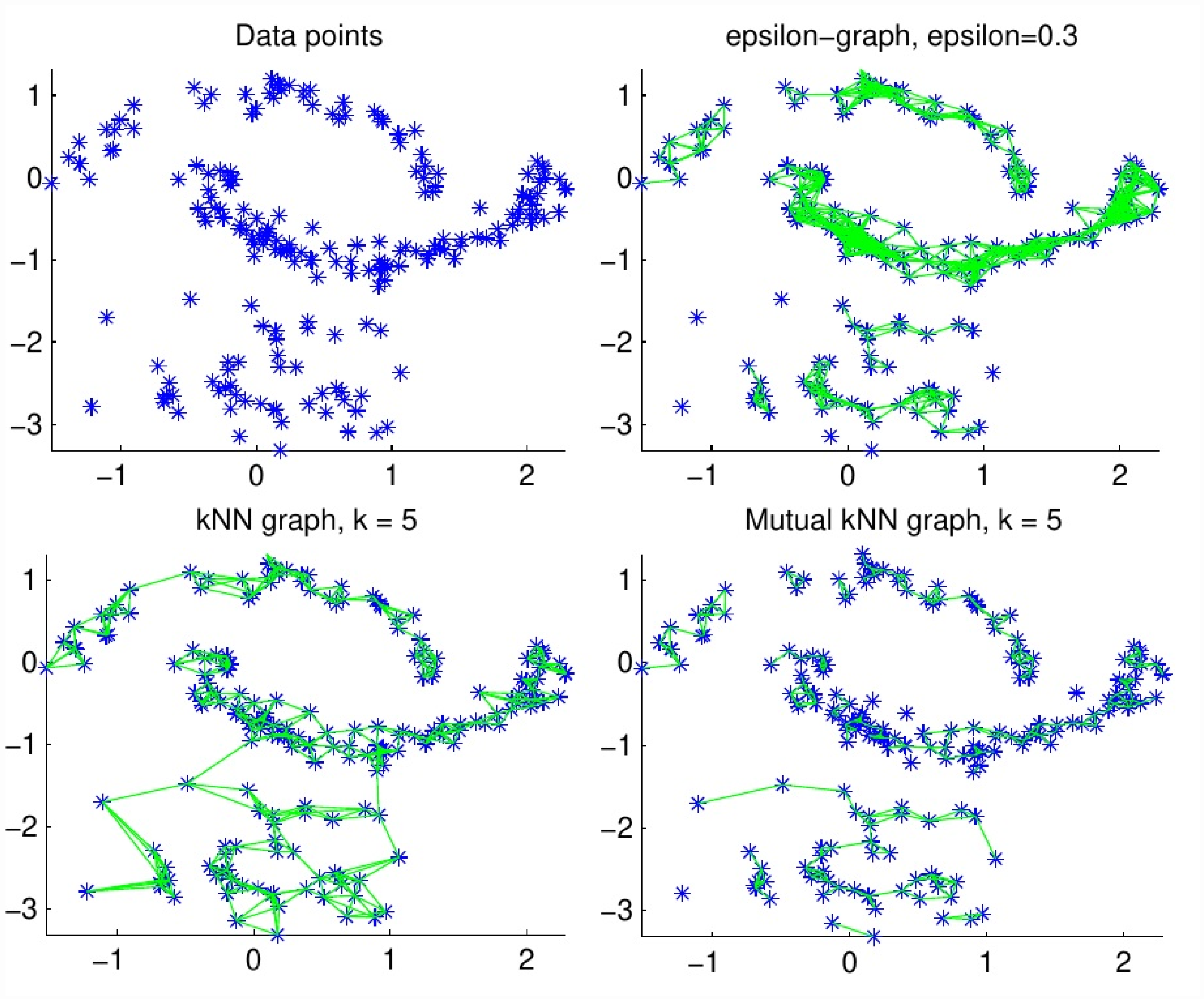
- **Method**: Unnormalized, random walk or symmetric normalized?
  - Usually normalization helps. Suggestion: try random walk first.
- **Similarity measure**
  - should measure local similarity reliably (close neighbours)
  - for numeric data, Gaussian similarity $exp\left(\frac{-||\mathbf{x}_i-\mathbf{x}_j||^2}{2\sigma^2}\right)$ often used
- **Similarity graph** and its parameters
  - this has a strong effect on results!

Common choices for the similarity graph

General goal: sparse but connected graph (or number of connected components $<< K$)

1. **ϵ-neighbourhood** graph: keep only $w_{ij} \geq \epsilon$
   - problems if clusters of different densities
2. **k-nearest neighbour** graph: v_i among k nearest neighbours of v_j **or** vice versa
   - often a good first choice
   - can break the graph into disconnected components
3. **mutual k-nearest neighbour** graph: v_i among k nearest neighbours of v_j **and** vice versa

Similarity graph examples (von Luxburg, Fig 3)



Similarity graph (cont)

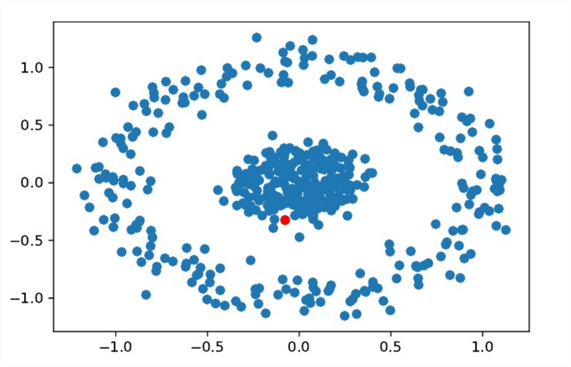
4. **fully connected graph**

- often with Gaussian similarity $\kappa(\mathbf{x}_i, \mathbf{x}_j) = exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$ (radial basis function, RBF)
- how to choose σ?
- Note: in scikitlearn parameter $\gamma = \frac{1}{2\sigma^2}$
- graph not sparse $\rightarrow$ heavy computation

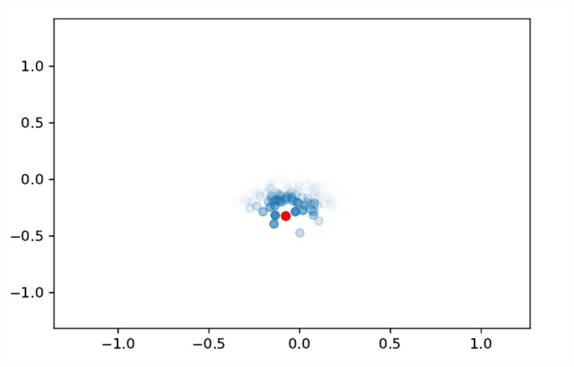
Choice of parameters (ϵ, k, σ) affects a lot, too!

Example: neighbourhood with $\kappa(\mathbf{x}_i, \mathbf{x}_j) = exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$

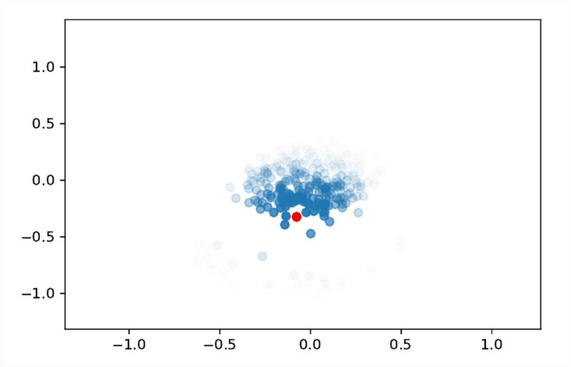
Data, query point red



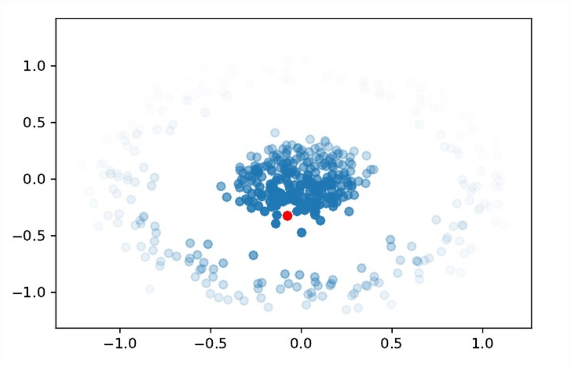
$\sigma = 0.1$



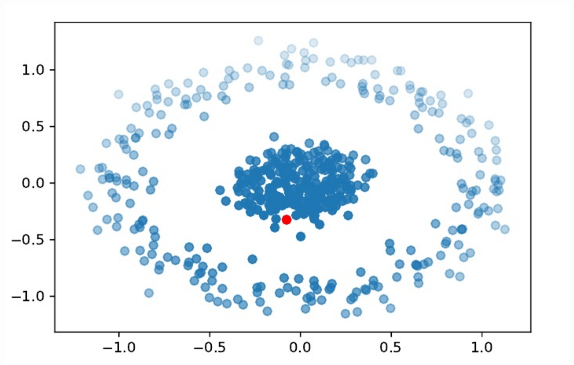
$\sigma = 0.2$



$\sigma = 0.4$



$\sigma = 0.8$



$\sigma = 1.6$

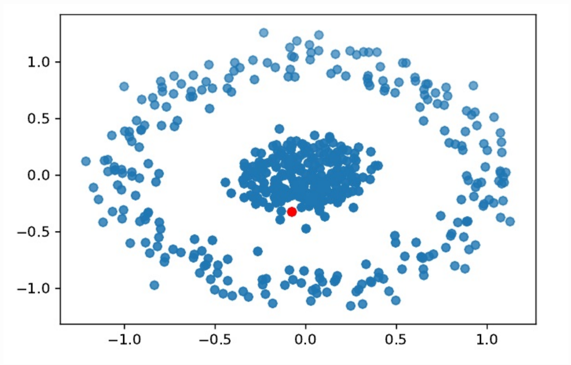
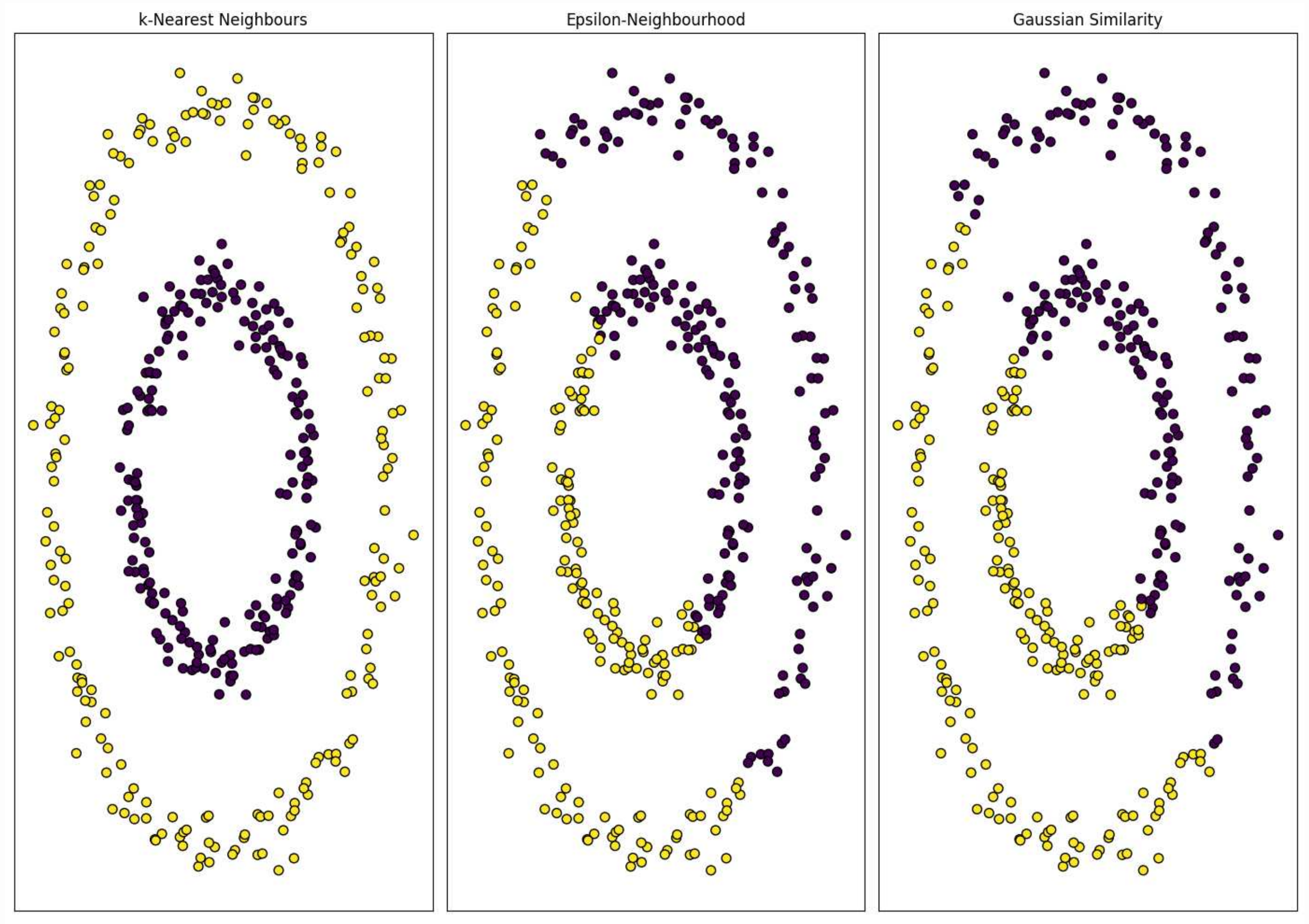


Image source: Bruno Ordozgoiti, MDM 2020 slides

Example: Different clustering results with different similarity graphs



Parameters: $k = 2$, $\epsilon = 0.3$, RBF with $\gamma = 10$ (i.e., $\sigma = \sqrt{5}$)

Experiment by Lai Khoa for MDM 2023

Summary

Idea: similarity graph $\rightarrow$ low-dimensional VS presentation (eigenvectors) $\rightarrow$ clustering (K-means etc.)

+ very powerful (virtually any datatype, arbitrary shapes)
– computationally expensive
  - creating similarity graph $\mathcal{O}(n^2)$, spectral decomposition $\mathcal{O}(n^3)$
– many important parameter choices

Further reading

von Luxburg: A Tutorial on Spectral Clustering. Statistics and Computing, vol. 17, pp. 395–416, 2007.

Reading guide: Sec 2 overview, 2.2, Sec 3 overview + definitions of Laplacian matrices from 3.1-3.2, Sec 4, Sec 5 overview, (possibly Sec 6 overview), Sec 8. [a]

[a] section overview = text before subsections