Lesson 13

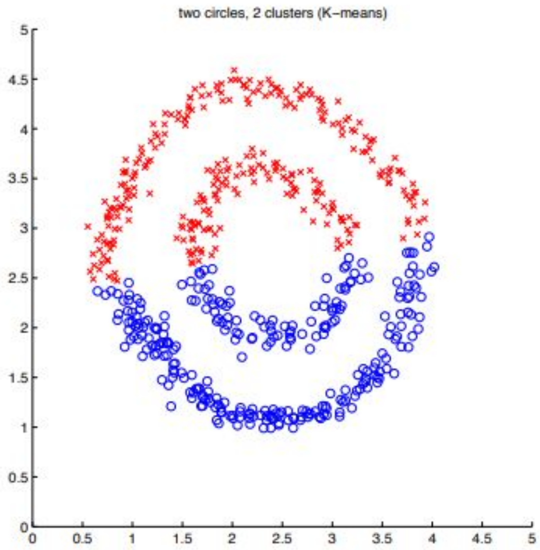
Final Project, Python

GUIDELINES

- Submission Date: 12/09/2021
- NO EXTENSION!!!

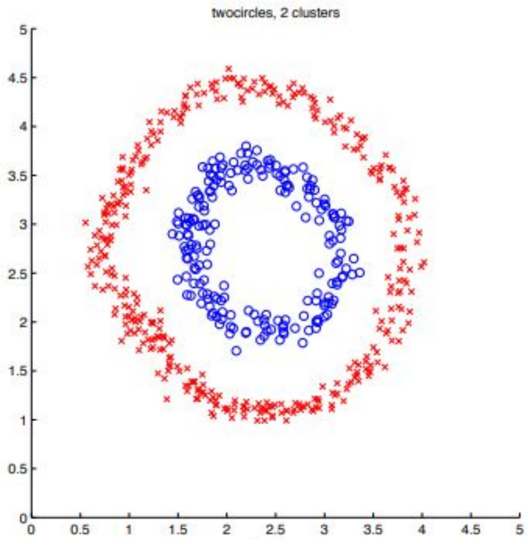
Motivation - Spectral clustering

K-means



(i)

Spectral clustering

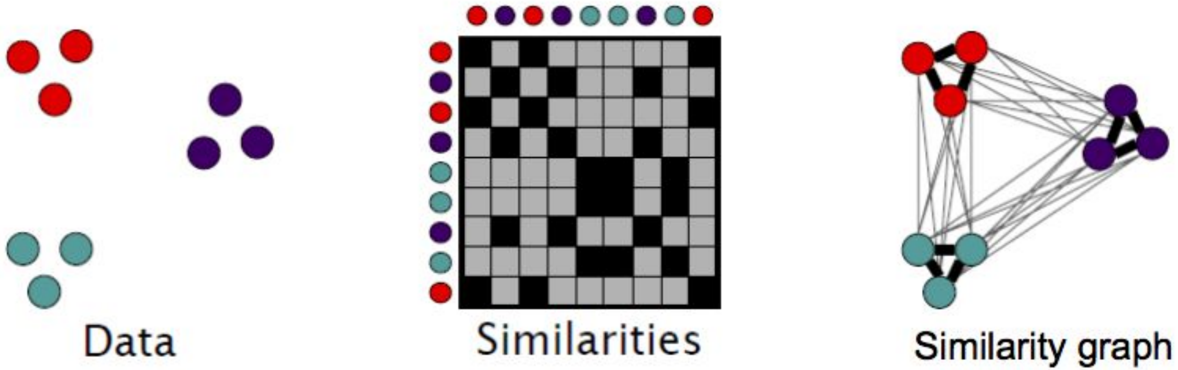


(e)

[Shi & Malik ‘00; Ng, Jordan, Weiss NIPS ‘01]

From Data points to graph

- Given: d-dimensional points: x_1, x_2, …, x_n.
- Transform them into a graph (Similarity Graph).
  - G = (V, E; W) - undirected with no self loops.



Weighted Adjacency Matrix

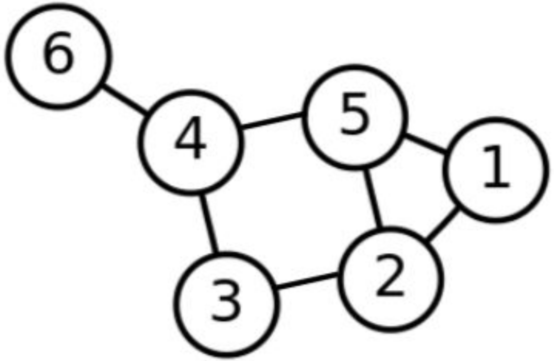
- Gaussian RBF

$$w_{ij} = \exp\left[-\frac{(x_i - x_j)^2}{2}\right]$$

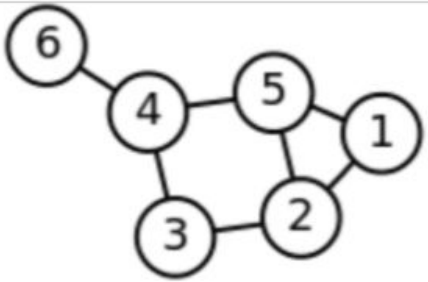
- W is symmetric, non-negative and no self loops(W_ii=0)

Example

- For simplicity, in the next example we will show a non fully connected graph, with all weights set to 1
- We are given d-dimensional data points: x_1 … x_n
- Choose random points and connect them, and we get:



Graph notations



D (diagonal) degree matrix

$$D_{ii} = \sum_{j=1}^{n} w_{ij} \quad \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

W weight matrix

$$W = (w_{ij}) \quad \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

L graph Laplacian

$$L = D - W$$

$$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$$

$\mathbf{L}_{norm}$ normalized graph Laplacian

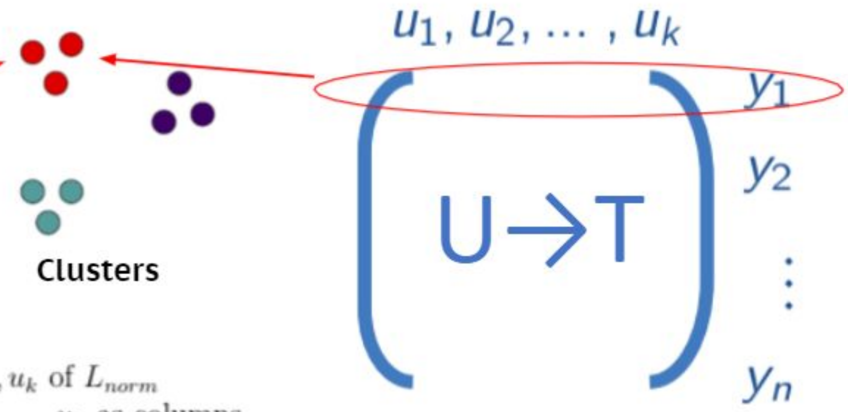
$$L_{norm} = \boxed{D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}}$$

$$\begin{pmatrix} 1 & -\frac{1}{\sqrt{6}} & 0 & 0 & -\frac{1}{\sqrt{6}} & 0 \\ -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{\sqrt{6}} & 0 & -\frac{1}{3} & 0 \\ 0 & -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{\sqrt{6}} & 0 & 0 \\ 0 & 0 & -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{3} & -\frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{6}} & -\frac{1}{3} & 0 & -\frac{1}{3} & 1 & 0 \\ 0 & 0 & 0 & -\frac{1}{\sqrt{3}} & 0 & 1 \end{pmatrix}$$

Source Wikipedia

Normalized spectral clustering (Ng, Jordan, and Weiss)

Given n points $X = \{x_1, x_2, \ldots, x_n\}$



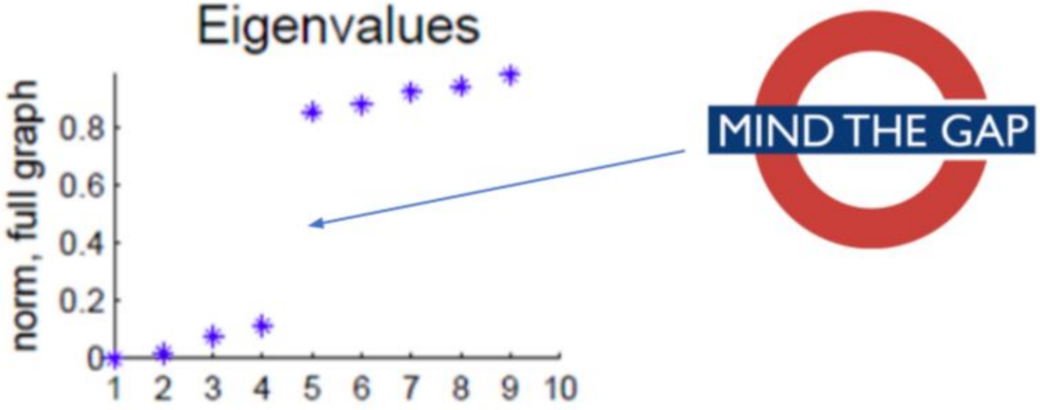
1: Form the weighted adjacency matrix W from X
2: Compute the normalized graph Laplacian L_{norm}
3: Determine k and obtain the first k eigenvectors $u_1, \ldots, u_k$ of L_{norm}
4: Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns
5: Form the matrix $T \in \mathbb{R}^{n \times k}$ from U by renormalizing each of U's rows to have unit length, that is set $t_{ij} = u_{ij} / (\sum_j u_{ij}^2)^{1/2}$
6: Treating each row of T as a point in $\mathbb{R}^k$, cluster them into k clusters via the K-means algorithm
7: Assign the original point x_i to cluster j if and only if row i of the matrix T was assigned to cluster j

DETERMINE K ?

- K-Means and Spectral Clustering require k (the number of clusters) as an input
- Possible solutions:
    - Prior knowledge about the data (e.g. as user input)
    - Eigengap Heuristic

Eigengap Heuristic

- Sort $0 \leq \lambda_1 \leq \ldots \leq \lambda_n$
- Delta: $\delta_i = |\lambda_i - \lambda_{i+1}|$
- Gap: $k = argmax_i(\delta_i), \quad i = 1, \ldots, \frac{n}{2}$



Finding Eigenvalues and Eigenvectors - Jacobi algorithm

(a) Build a rotation matrix P (as explained below).

(b) Transform the matrix A to:

$$A' = P^T AP$$

(c) Repeat a,b until A' is diagonal matrix.

(d) The diagonal of final A' is the eigenvalues of A.

(e) Calculate eigenvectors of A by multiplying all the rotation matrices:

$$V = P_1 P_2 P_3 \dots$$

Rotation Matrix P

Let S be a symmetric matrix, and P is Jacobi rotation matrix of the form:

$$P = \begin{pmatrix} 1 & & & & & & \\ & \cdots & & & & & \\ & & c & \cdots & s & & \\ & & \vdots & 1 & \vdots & & \\ & & -s & \cdots & c & & \\ & & & & & \cdots & \\ & & & & & & 1 \end{pmatrix}$$

Jacobi Terminology

Pivot

The A_{ij} is the off-diagonal element with **the largest absolute value.**

Obtain c, t

$$\theta = \cot 2\phi = \frac{A_{jj} - A_{ii}}{2A_{ij}}$$

$$t = \frac{sign(\theta)}{|\theta| + \sqrt{\theta^2 + 1}}$$

$$c = \frac{1}{\sqrt{t^2 + 1}}, \quad s = tc$$

Note: We define $sign(0) = 1$

Update A-matrix efficiently

After each transformation in step 2, the changed elements of A are only the i and j rows and columns. Therefore, using the symmetry of A we can obtain the following formula to calculate A':

$$
\begin{aligned}
a'_{ri} &= ca_{ri} - sa_{rj} \qquad r \neq i, j \\
a'_{rj} &= ca_{rj} + sa_{ri} \qquad r \neq i, j \\
a'_{ii} &= c^2 a_{ii} + s^2 a_{jj} - 2sca_{ij} \\
a'_{jj} &= s^2 a_{ii} + c^2 a_{jj} + 2sca_{ij} \\
a'_{ij} &= (c^2 - s^2) a_{ij} + sc(a_{ii} - a_{jj}) \Rightarrow a'_{ij} = 0
\end{aligned}
$$

Note: A' **is always symmetric.**

Jacobi example

(a) Build a rotation matrix P (as explained below).

(b) Transform the matrix A to:

$$A' = P^T A P$$

(c) Repeat a,b until A' is diagonal matrix.

(d) The diagonal of final A' is the eigenvalues of A.

(e) Calculate eigenvectors of A by multiplying all the rotation matrices:

$$V = P_1 P_2 P_3 \dots$$

Let S be a symmetric matrix, and P is Jacobi rotation matrix of the form:

$$P = \begin{pmatrix} 1 & & & & & & \\ & \cdots & & & & & \\ & & c & \cdots & s & & \\ & & \vdots & 1 & \vdots & & \\ & & -s & \cdots & c & & \\ & & & & & \cdots & \\ & & & & & & 1 \end{pmatrix}$$

Pivot

The A_{ij} is the off-diagonal element with **the largest absolute value.**

Obtain c, t

$$\theta = \cot 2\phi = \frac{A_{jj} - A_{ii}}{2A_{ij}}$$

$$t = \frac{sign(\theta)}{|\theta| + \sqrt{\theta^2 + 1}}$$

$$c = \frac{1}{\sqrt{t^2 + 1}}, \quad s = tc$$

Note: We define $sign(0) = 1$

$$a'_{ri} = ca_{ri} - sa_{rj} \qquad r \neq i, j$$

$$a'_{rj} = ca_{rj} + sa_{ri} \qquad r \neq i, j$$

$$a'_{ii} = c^2 a_{ii} + s^2 a_{jj} - 2sca_{ij}$$

$$a'_{jj} = s^2 a_{ii} + c^2 a_{jj} + 2sca_{ij}$$

$$a'_{ij} = (c^2 - s^2)a_{ij} + sc(a_{ii} - a_{jj}) \Rightarrow a'_{ij} = 0$$

EXAM!

STRUCTURE

- 2 hours
- 1 paper formula sheet
- 4 questions:
  - 1 x open question in C
  - 2 x short answer (fill missing code) Python
  - 1 x short Eli’s material
- Python Material:
  - Everything we learned in lectures
  - Data science oriented Python questions
  - https://www.practicaldatascience.org/html/class_schedule.html

שאלה 1

נתון כמות הייצור השנתית של 5 עובדים לאורך שנים (בסדר עולה) בטבלת PRODUCTION . מעוניינים לאמוד את התפוקה של עובדים חדשים NEW_EMPS (מכילה עמודת ID,VETEK), לפי הקריטריון הבא: תפוקה צפויה = ותק*אלפא, כך שאלפא זה ממוצע התפוקה של עובד 3 ו4 בחמשת השני הראשונות שלהם. השלם את השורה כדי לתת תחזית לתפוקת העובדים החדשים.

```
>>> production.shape
(20,5)
# ---------------------------
>>> print(new_emps['production'])
42105
265455
333008
...
```

שאלה 1

נתון כמות הייצור השנתית של 5 עובדים לאורך שנים (בסדר עולה) בטבלת PRODUCTION . מעוניינים לאמוד את התפוקה של עובדים חדשים NEW_EMPS (מכילה עמודת ID,VETEK), לפי הקריטריון הבא: תפוקה צפויה = ותק*אלפא, כך שאלפא זה ממוצע התפוקה של עובד 3 ו4 בחמשת השני הראשונות שלהם. השלם את השורה כדי לתת תחזית לתפוקת העובדים החדשים.

```
>>> production.shape
(20,5)
>>> new_emps['production'] = new_emps['vetek']*production[:5,[2,3]].mean()
>>> print(new_emps['production'])
42105
265455
333008
...
```

שאלה 2

For each **continent** show the **continent** and number of countries with populations of at least 10 million.

```
import numpy as np
import pandas as pd

biggest_countries = world[world['population']>=10000000]
------
print(big_per_cont)
```

name	continent	area	population	gdp
Afghanistan	Asia	652230	25500100	20343000000
Albania	Europe	28748	2831741	12960000000
Algeria	Africa	2381741	37100000	188681000000
Andorra	Europe	468	78115	3712000000
Angola	Africa	1246700	20609294	100990000000
...				

שאלה 2

For each **continent** show the **continent** and number of countries with populations of at least 10 million.

```
import numpy as np
import pandas as pd

big_countries = world[world['population']>=10000000]
big_per_cont = big_countries.groupy(['name']).name.count()
print(big_per_cont)
```

name	continent	area	population	gdp
Afghanistan	Asia	652230	25500100	20343000000
Albania	Europe	28748	2831741	12960000000
Algeria	Africa	2381741	37100000	188681000000
Andorra	Europe	468	78115	3712000000
Angola	Africa	1246700	20609294	100990000000
...				

שאלה 3

נתון מודל מאומן של סיווג על ידי רגרסיה לוגיסטית model, ברצוננו לבצע סיווג לנקודה (-0.79415228, 2.10495117), השלם את החלק החסר:

```
# example of making a single class prediction
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import make_blobs
# generate 2d classification dataset
X, y = make_blobs(n_samples=100, centers=2, n_features=2, random_state=1)

-----------------------------------------------
```

שאלה 3

נתון מודל מאומן של סיווג על ידי רגרסיה לוגיסטית model, ברצוננו לבצע סיווג לנקודה:(-0.794, 2.104), השלם את החלק החסר:

```
# example of making a single class prediction
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import make_blobs
# generate 2d classification dataset
X, y = make_blobs(n_samples=100, centers=2, n_features=2, random_state=1)

Xnew = [[-0.794, 2.104]]
model.predict(Xnew)
```