# Structure Discovery

Machine Learning for Behavioral Data

April 25, 2025

# Today's Topic

| Week | Lecture/Lab |
|---|---|
| 9 | Unsupervised Learning |
| 10 | Spring Break |
| 11 | Unsupervised Learning |
| 12 | Ethical Machine Learning |
| 13 | Ethical Machine Learning |
| 14 | Reserve |
| 15 | Poster Presentations |

- **K-Means, Spectral Clustering**
- **Choosing the optimal K***
- **Clustering time-series data**

# Getting ready for today's lecture…

- **If not done yet**: clone the repository containing the Jupyter notebook and data for today's lecture into your Noto workspace
- SpeakUp room for today's lecture:

**https://go.epfl.ch/speakup-mlbd2025**

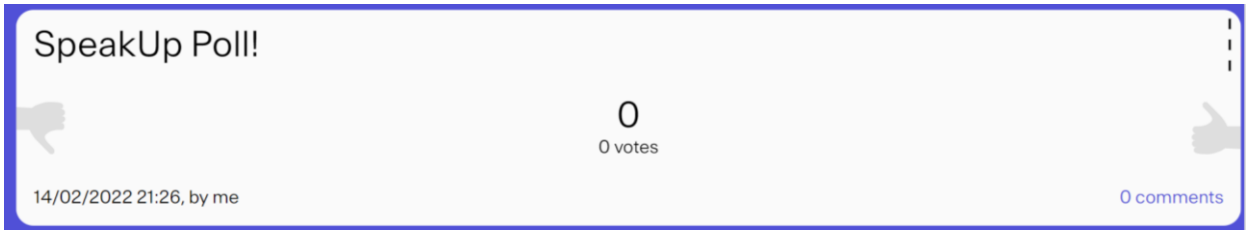
# Short quiz about the past…

Which of the following NN architecture has the largest number of parameters?

a) Vanilla RNN

b) GRU

c) LSTM



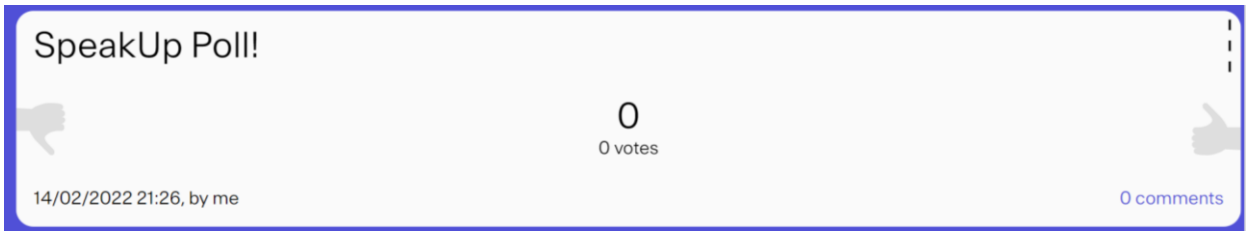

https://go.epfl.ch/speakup-mlbd2025

# Short quiz about the past…

In contrast to Vanilla RNNs, GRU units include an additional memory cell.

a) True

b) False



# Short quiz about the past…

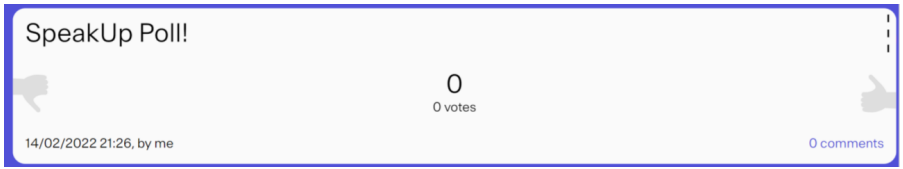
Which of the following should be changed to adapt a NN used for classification to a regression task?

a) The dimension of the hidden layer

b) The activation function of the output layer

c) The batch size

d) The drop-out rate




https://go.epfl.ch/speakup-mlbd2025

# Why doing structure discovery?

- We are interested in finding different groups of users
  - for analytical purposes (e.g., to analyze how different types of users use our services)
  - to adapt the environment to different user types
- Examples:
  - Finding groups of students with similar strategies
  - Identifying different types of new users on Snapchat (personalized retention)
  - Grouping tourists by their mobility patterns (recommendation)

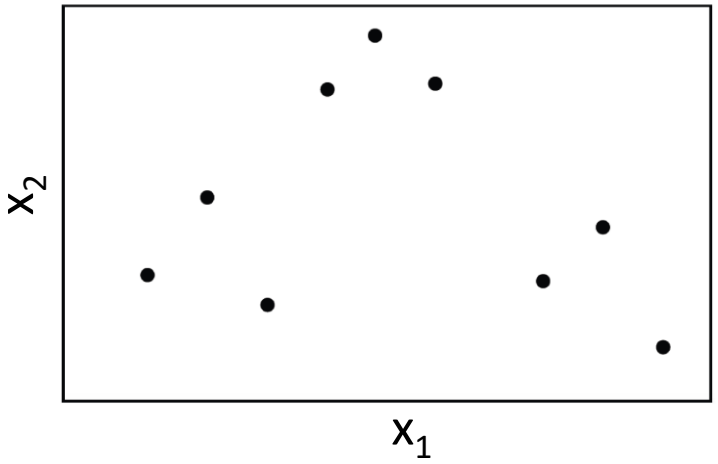
# Agenda

- Clustering Algorithms
  - K-Means Clustering
  - Spectral Clustering
- Choosing the optimal number of clusters

# Agenda

- Clustering Algorithms
  - **K-Means Clustering**
  - Spectral Clustering
- Choosing the optimal number of clusters

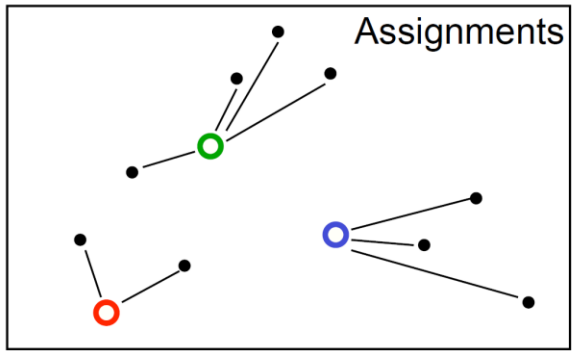
# K-Means Clustering

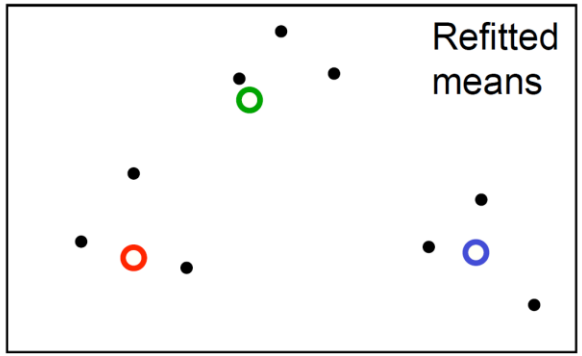



- Assume the data $\{\boldsymbol{x_1}, \ldots, \boldsymbol{x_N}\}$ lives in a **Euclidean** space, $\boldsymbol{x_n} \in \mathbb{R}^D$
- Assume the data belongs to $K$ different classes (groups)
- How can we identify those classes (data points that belong to each class)?

# K-Means Algorithm

- Initialization: randomly assign cluster centers
- Algorithm iteratively alternates between two steps:
  - Assignment step: assign each data point to the closest cluster
  - Update step: move each cluster center to the center of gravity of the data assigned to it




[Image credit: Urtasun, CSC-411]

# K-Means Algorithm

- Initialization: set $K$ cluster means $\boldsymbol{m_1}, \ldots, \boldsymbol{m_K}$ to random values
- Repeat until convergence (until assignments do not change):
  - Assignment: each data point $\boldsymbol{x_n}$ assigned to nearest mean

$$\hat{k}^n = arg\min_k d(\boldsymbol{m_k}, \boldsymbol{x_n})$$

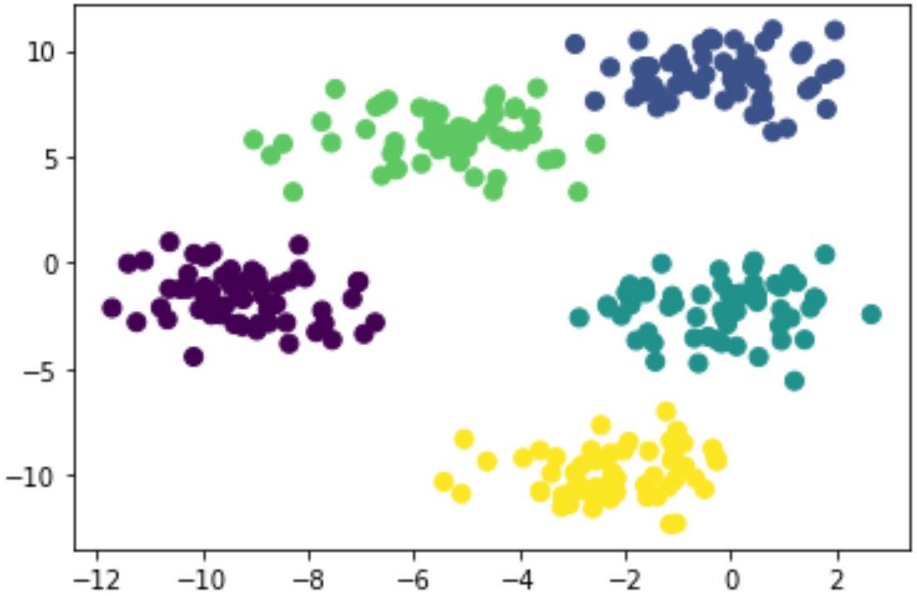
(e.g, with Euclidean distance: $d(\boldsymbol{m_k}, \boldsymbol{x_n}) = \|\boldsymbol{m_k} - \boldsymbol{x_n}\|_2$)

  - Update: adjust means to match sample means of data points they are responsible for:

$$\boldsymbol{m_k} = \frac{\sum_n r_k^{(n)} \boldsymbol{x_n}}{\sum_n r_k^{(n)}}$$ , where $r_k^{(n)} = 1$, if $\hat{k}^n = \mathrm{k}$

# K-Means Example

Synthetic data with k=5 clusters



# Observations

- Solution (goodness of solution) depends on the initial positioning of the cluster centers
- Solution (goodness of solution) depends on the choice of $k$ (the number of clusters)

→ How should we initialize the cluster centers?

→ How to choose the optimal number of clusters?

# Initialization of cluster centers

- Random restarts:
  - Run to convergence using different random initializations
  - Choose the one the minimizes distortion (squared distance of data to cluster means)
- Distortion $D$ (measure of in-cluster variance):

$$D = \sum_{n} \left( d\left( \boldsymbol{m}_{\widehat{\boldsymbol{k}}^{n}}, \boldsymbol{x}_{\boldsymbol{n}} \right) \right)^{2}$$

(e.g, with Euclidean distance: $d(\boldsymbol{m}_{\boldsymbol{k}}, \boldsymbol{x}_{\boldsymbol{n}}) = \|\boldsymbol{m}_{\boldsymbol{k}} - \boldsymbol{x}_{\boldsymbol{n}}\|_2$)

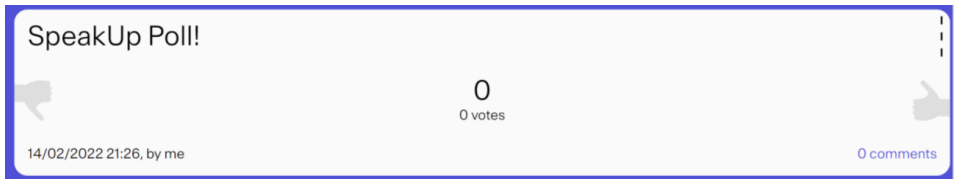
# Minimizing Distortion

- Distortion $D$ (measure of in-cluster variance):

$$D = \sum_{n} \left( d\left( \boldsymbol{m}_{\hat{k}^n}, \boldsymbol{x}_n \right) \right)^2$$

(e.g, with Euclidean distance: $d(\boldsymbol{m}_k, \boldsymbol{x}_n) = \|\boldsymbol{m}_k - \boldsymbol{x}_n\|_2$)

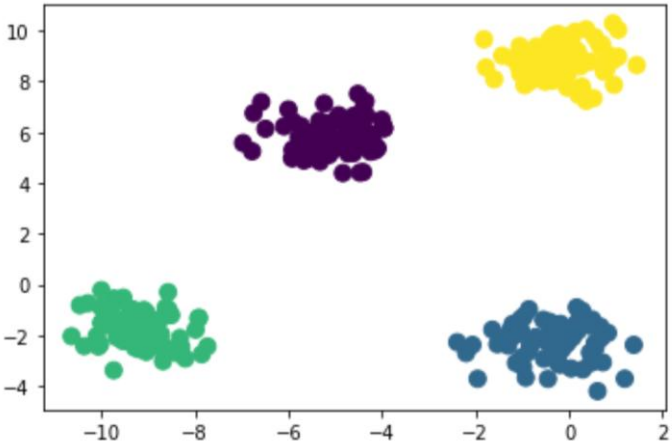
- Could we determine the optimal number of clusters by minimizing D?

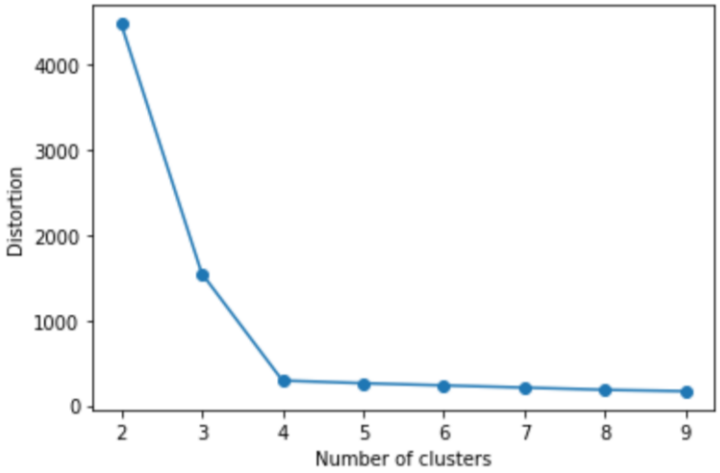



a) Yes
b) No

# Selecting $k^*$ - Elbow Method

- Elbow Method (Heuristic): choose $k^*$ such that adding another cluster does not lead to a much better model of the data



# Selecting $k^*$ - Silhouette Score

- Silhouette width ($-1 \leq s \leq 1$): Silhouette width measures how similar a data point is to its own cluster (cohesion) compared to other clusters (separation)
- Can be computed for $k = 2, \dots, N$
- For a data point $\boldsymbol{x_i}$ in cluster $C_k$:

$$a(i) = \frac{1}{|C_k| - 1} \sum_{j \in C_k, i \neq j} d(\boldsymbol{x_i}, \boldsymbol{x_j}) \qquad b(i) = \min_{l \neq k} \frac{1}{|C_l|} \sum_{j \in C_l} d(\boldsymbol{x_i}, \boldsymbol{x_j})$$

$$s(i) = \frac{b(i) - a(i)}{\max\{b(i), a(i)\}} \text{ if } |C_k| > 1 \qquad s(i) = 0 \text{ if } |C_k| = 1$$

# Selecting $k^*$ - Silhouette Score

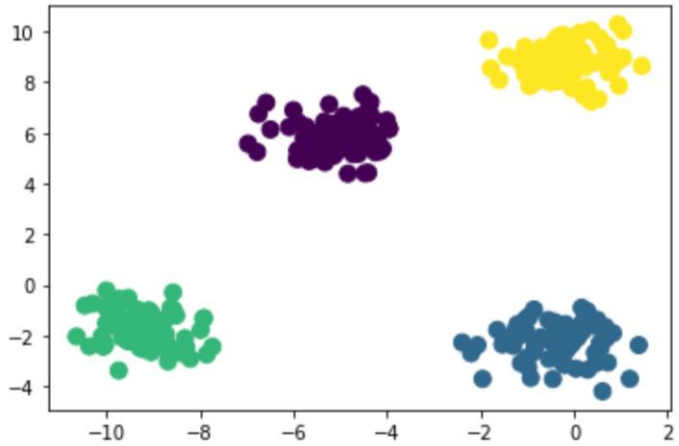
- Overall average silhouette width: average over all data points $\boldsymbol{x_i}$, given a cluster number $k$:

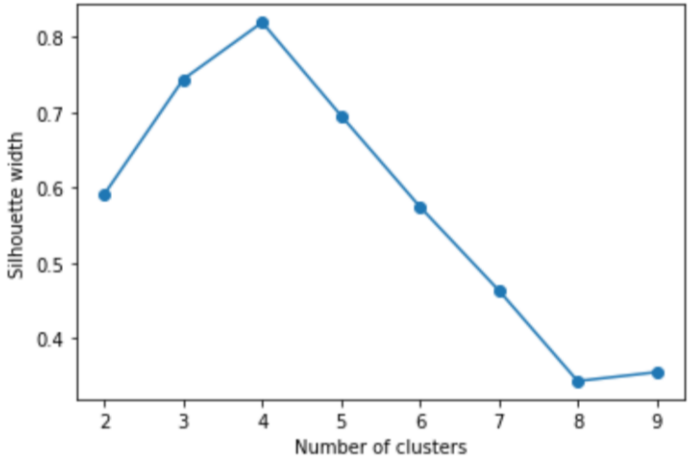
$$\bar{s}_k = \frac{1}{N}\sum_{i=1}^{N} s(i)$$

- Find $k^*$ that maximizes the overall average silhouette width:

$$k^* = arg\max_k \bar{s}_k$$

# Example: Silhouette Width



# Selecting $k^*$ - BIC Score

- Assumptions of K-Means:
  - Data points live in an Euclidean space
  - Data points are spherically distributed around centroid of clusters (spherical Gaussians)

➡ We can compute the likelihood of our cluster solution for a given $k$

➡ We can use the $BIC$ to determine $k^*$

# Bayesian Information Criterion (BIC)

$$BIC = -2 \cdot LL + \log(N) \cdot d$$

- $d$ is the number of parameters of our model $f$
- $LL$ is the log-likelihood (logarithm of the likelihood) of the sample data $T$ given a specific $k$
- $N$ is the number of samples in the data set, i.e. $|T| = N$

# Schwarz Criterion (BIC)

$$BIC = -2 \cdot LL + \log(N) \cdot d$$

$$BIC = LL - \frac{d}{2} \cdot \log(N)$$

# Likelihood for one data point

- Probability for a data point $x_i$, where $\hat{r}^i$ denotes the cluster assignments for $x_i$ (spherical gaussian assumption):

$$p(x_i) = \frac{|C_{\hat{r}^i}|}{N} \cdot \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{1}{2\sigma^2} \cdot \left\| \boldsymbol{x_i} - \boldsymbol{m}_{\hat{r}^i} \right\|_2^2}$$

- Computing the variance $\sigma^2$ of the data:

$$\sigma^2 = \frac{1}{N-k} \cdot \sum_{i=1}^{N} \left( \boldsymbol{x_i} - \boldsymbol{m}_{\hat{r}^i} \right)^2$$

# Computing the log-likelihood

- Compute the log-likelihood over all data points:

$$LL = \log \prod_{i=1}^{N} p(x_i)$$

$$= \sum_{i=1}^{N} \log \frac{\left|C_{\hat{r}^i}\right|}{N} + \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} \cdot \left\| \boldsymbol{x_i} - \boldsymbol{m}_{\hat{r}^i} \right\|_2^2$$

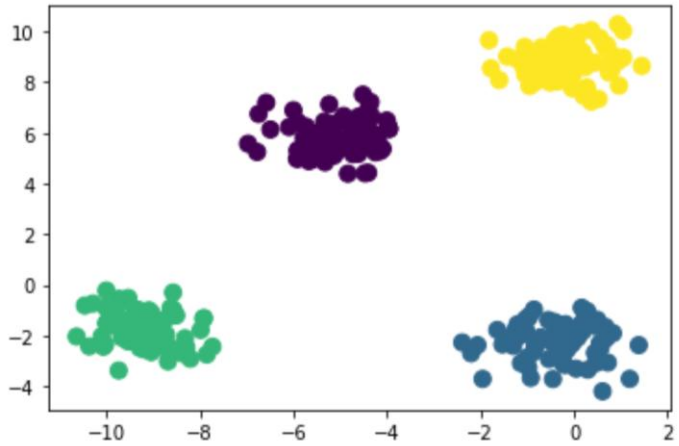
# Computing the number of parameters $d$

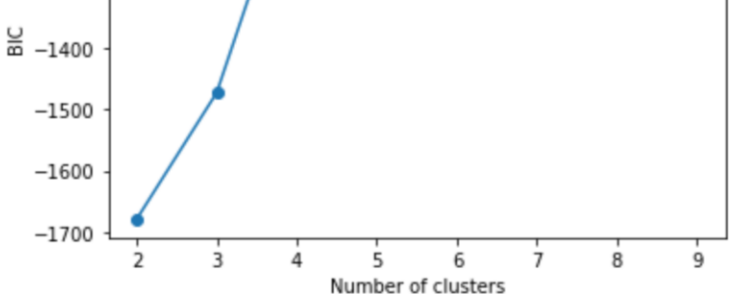
- Here, number of parameters is equivalent to degrees of freedom:

$$d = (k - 1) + 1 + k \cdot D$$

- $k$ is the number of clusters, $D$ is the number of dimensions of the data points $\boldsymbol{x_i}$
- We estimate the following:
  - $k - 1$ prior probabilities (for the $k$ clusters)
  - $1$ variance estimate ($\sigma^2$)
  - $k \cdot D$ centroid coordinates

# BIC Example


10
8
6
4
2
0
−2
−4
−10
−8
−6
−4
−2
0
2


BIC
−1100
−1200
−1300
−1400
−1500
−1600
−1700
2
3
4
5
6
7
8
9
Number of clusters

# Agenda

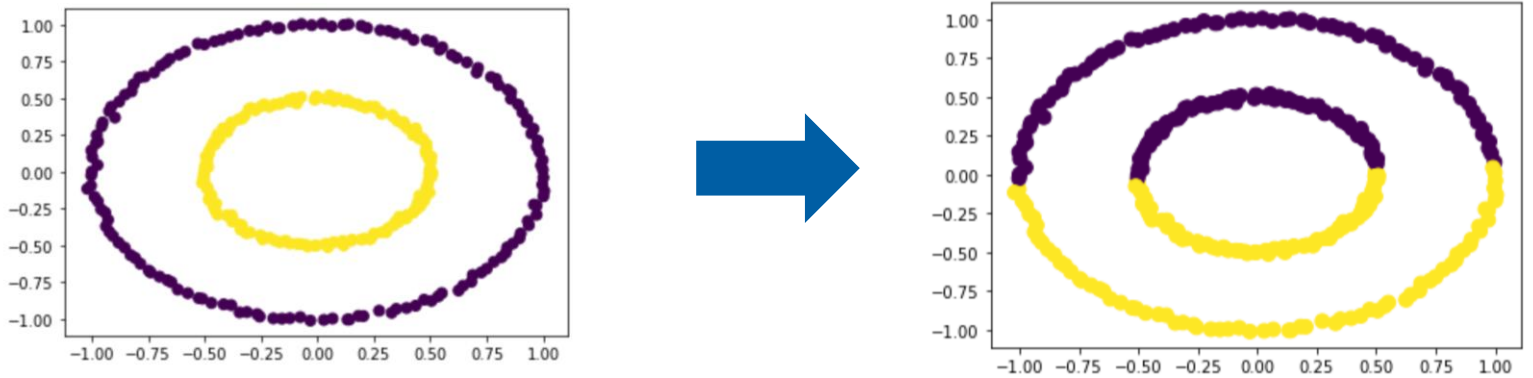
- Clustering Algorithms
  - K-Means Clustering
  - **Spectral Clustering**
- Choosing the optimal number of clusters

# Two broad assumptions for clustering

- Compactness: Points that lie **close** to each other fall in the same cluster and are compact around the cluster center. The closeness can be measured by the distance between the observations.
- Connectivity: Points that are **connected** or immediately next to each other are put in the same cluster. Even if 2 points are close together, if they are not connected, they are not clustered together.

# Assumptions of K-Means

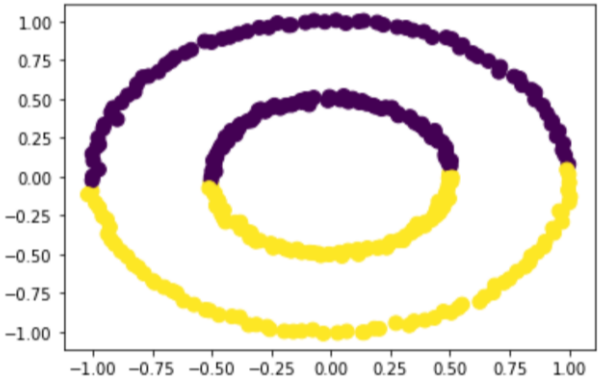
- K-Means assumes a Euclidean space
- K-Means assumes that the variance of the distribution of each cluster is spherical



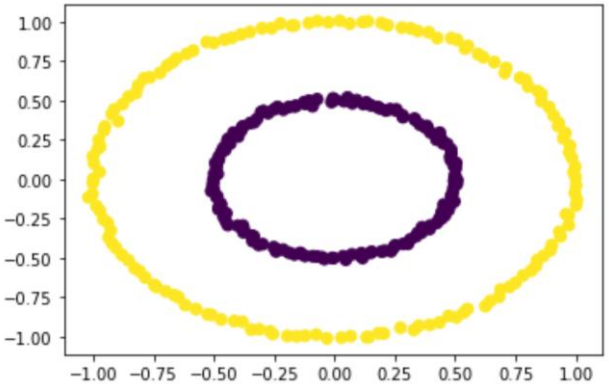
# Assumptions of K-Means

- K-Means assumes a Euclidean space
- K-Means assumes that the variance of the distribution of each cluster is spherical

**K-Means**




**Spectral Clustering**



# Spectral Clustering

- No assumption is made about the form/shape of the clusters
- Data points are treated as nodes of graphs
- Algorithm consists of three steps:
  1. Compute the pairwise similarities $s(\boldsymbol{x_i}, \boldsymbol{x_j})$ between all pairs of data points $i$ and $j$
  2. Construct a similarity graph
  3. Compute first $k$ eigenvectors ($k$ is the number of clusters) of graph Laplacian
  4. Perform clustering on transformed data

# Spectral Clustering

- No assumption is made about the form/shape of the clusters
- Data points are treated as nodes of graphs
- Algorithm consists of three steps:
  1. **Compute the pairwise similarities $s(x_i, x_j)$ between all pairs of data points $i$ and $j$**
  2. Construct a similarity graph
  3. Compute first $k$ eigenvectors ($k$ is the number of clusters) of graph Laplacian
  4. Perform clustering on transformed data

# Similarity Measures

- Quantify similarity between two samples
- No single definition exists, can usually be seen as the inverse of distance metrics

# Similarity Measures

- Cosine Similarity: for vectors, often used for document comparison

$$S_{cos}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\|_2 \cdot \|\boldsymbol{y}\|_2} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}$$

- Jaccard Similarity: for set data

$$s(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

# Similarity Measures

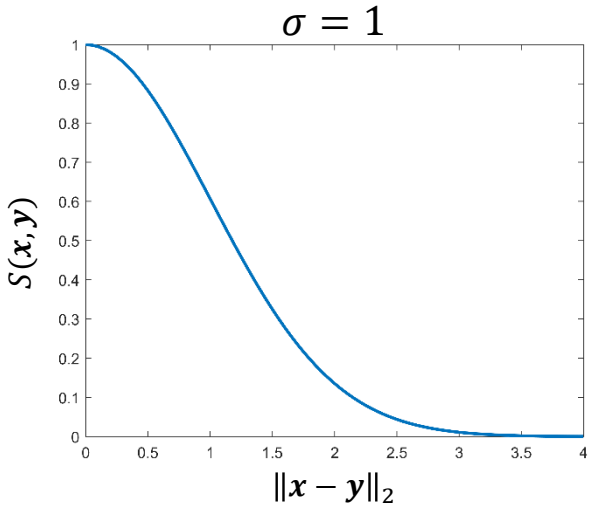
- Gaussian Kernel with Euclidean distance (takes into account local neighborhood)

$$S(\boldsymbol{x}, \boldsymbol{y}) = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|_2^2}{2\sigma^2}}$$

# Similarity Measures

- Gaussian Kernel with Euclidean distance (takes into account local neighborhood)

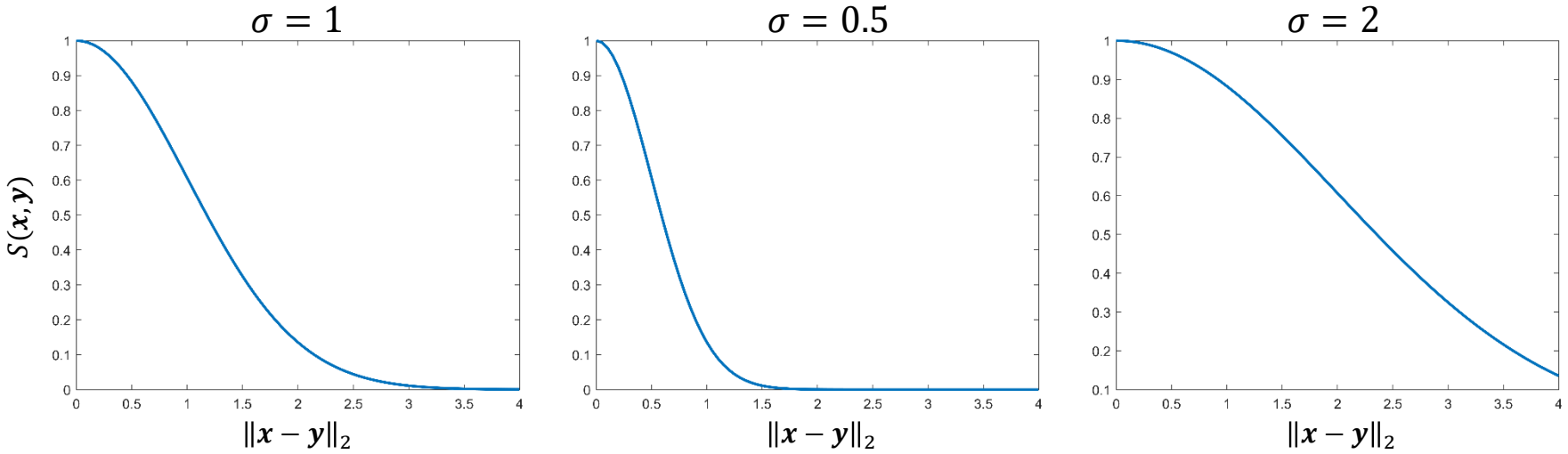
$$S(\boldsymbol{x}, \boldsymbol{y}) = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|_2^2}{2\sigma^2}}$$



# Similarity Measures

- Gaussian Kernel with Euclidean distance (takes into account local neighborhood)

$$S(\boldsymbol{x}, \boldsymbol{y}) = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|_2^2}{2\sigma^2}}$$



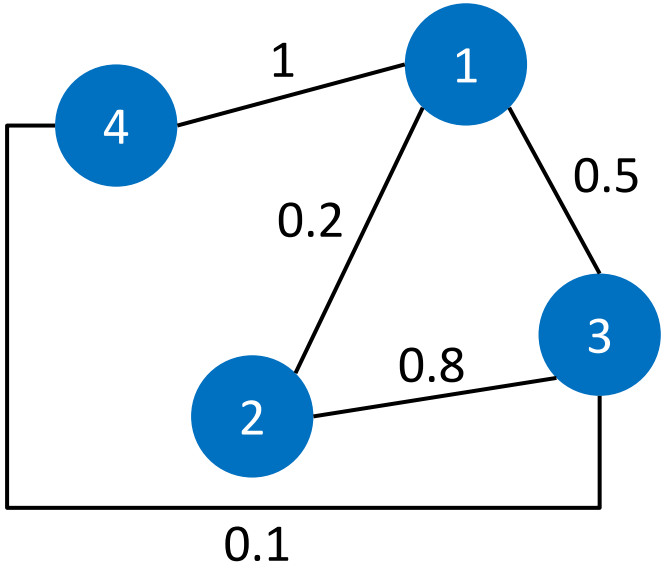
# Spectral Clustering

- No assumption is made about the form/shape of the clusters
- Data points are treated as nodes of graphs
- Algorithm consists of three steps:
  1. Compute the pairwise similarities $s(x_i, x_j)$ between all pairs of data points $i$ and $j$
  2. **Construct a similarity graph**
  3. Compute first $k$ eigenvectors ($k$ is the number of clusters) of graph Laplacian
  4. Perform clustering on transformed data

# Undirected Graphs - Notation

- Weighted adjacency matrix $W$

$$W = \begin{pmatrix} 0 & 0.2 & 0.5 & 1 \\ 0.2 & 0 & 0.8 & 0 \\ 0.5 & 0.8 & 0 & 0.1 \\ 1 & 0 & 0.1 & 0 \end{pmatrix}$$




- Degree $d_i$ of a node $i$: $d_i = \sum_{j=1}^{n} w_{ij}$

- Degree matrix $D$: $D = \begin{pmatrix} d_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & d_n \end{pmatrix}$

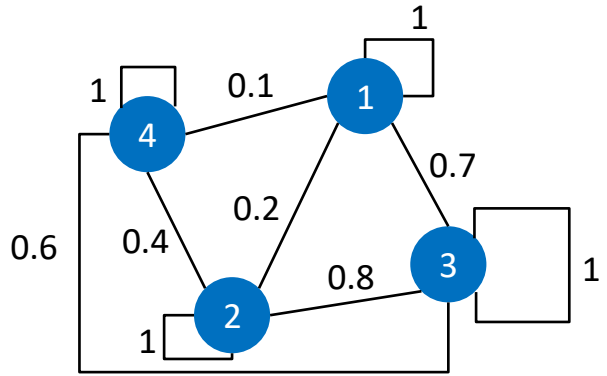
# Constructing Similarity Graphs

- Given: a set of data points $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}$ and some notion of similarity $s_{ij} \geq 0$ between all pairs of data points $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$
- We assume that
  - Each data point $\boldsymbol{x_i}$ represents a vertex of a graph
  - Two "vertices" $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ are connected, if $s_{ij}$ is larger than a threshold (or zero), and the edge is weighted with $s_{ij}$

# Constructing Similarity Graphs

- *Fully connected graph*: simply connect all data points $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ with positive similarity $s_{ij}$ and weight the edges with $s_{ij}$
- Example ($S$ denotes the similarity matrix):

$$S = \begin{pmatrix} 1 & 0.2 & 0.7 & 0.1 \\ 0.2 & 1 & 0.8 & 0.4 \\ 0.7 & 0.8 & 1 & 0.6 \\ 0.1 & 0.4 & 0.6 & 1 \end{pmatrix}$$

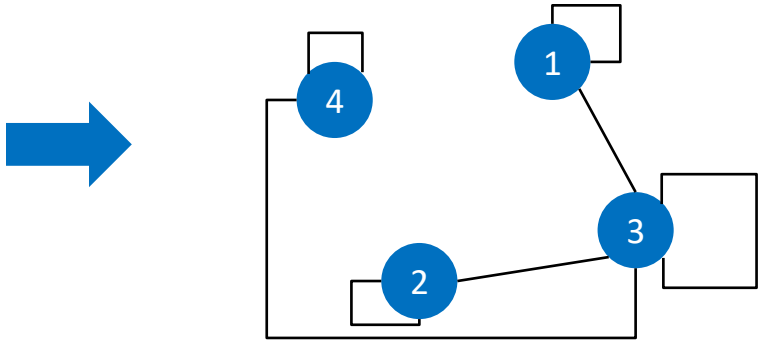



$W = S$

# Constructing Similarity Graphs

- $\varepsilon$ *−neighborhood graph*: we connect all data points $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ with similarity $s_{ij} > \varepsilon$ and treat the graph as unweighted
- Example with $\varepsilon = 0.5$ ($S$ denotes the similarity matrix):

$$S = \begin{pmatrix} 1 & 0.2 & 0.7 & 0.1 \\ 0.2 & 1 & 0.8 & 0.4 \\ 0.7 & 0.8 & 1 & 0.6 \\ 0.1 & 0.4 & 0.6 & 1 \end{pmatrix}$$

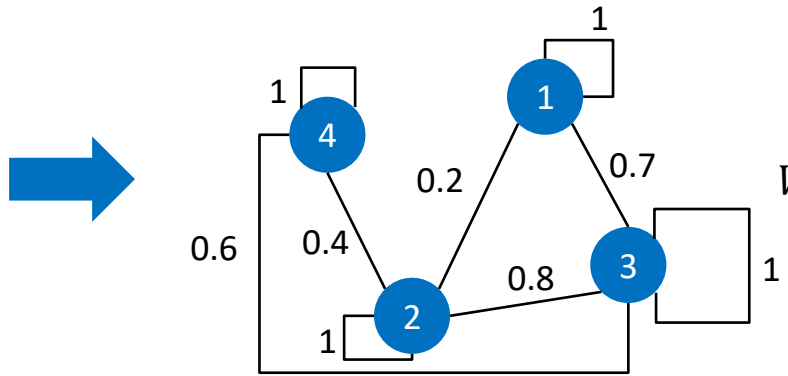



$$W = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

# Constructing Similarity Graphs

- $k$ *−nearest neighbor graph*: we connect all data points $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ if $\boldsymbol{x_i}$ is among the $k$ nearest neighbors of $\boldsymbol{x_j}$ **or** $\boldsymbol{x_j}$ is among the $k$ nearest neighbors of $\boldsymbol{x_i}$
- *mutual* $k$ *−nearest neighbor graph*: we connect all data points $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ if $\boldsymbol{x_i}$ is among the $k$ nearest neighbors of $\boldsymbol{x_j}$ **and** $\boldsymbol{x_j}$ is among the $k$ nearest neighbors of $\boldsymbol{x_i}$
- Example with $k = 2$ ($S$ denotes the similarity matrix):

$$S = \begin{pmatrix} 1 & 0.2 & 0.7 & 0.1 \\ 0.2 & 1 & 0.8 & 0.4 \\ 0.7 & 0.8 & 1 & 0.6 \\ 0.1 & 0.4 & 0.6 & 1 \end{pmatrix}$$




$$W = \begin{pmatrix} 1 & 0.2 & 0.7 & 0 \\ 0.2 & 1 & 0.8 & 0.4 \\ 0.7 & 0.8 & 1 & 0.6 \\ 0 & 0.4 & 0.6 & 1 \end{pmatrix}$$

# Spectral Clustering - Algorithm

## Unnormalized spectral clustering

Input: similarity matrix $S \in \mathbb{R}^{n \times n}$, number $k$ of clusters to construct

- Construct a similarity graph and compute $W$ (weighted adjacency matrix) and $D$ (degree matrix)
- Compute the unnormalized graph Laplacian $L = D - W$
- Compute the first $k$ eigenvectors $u_1, \ldots, u_k$ of $L$
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the $i$-th row of $U$
- Cluster the points $y_i$ into clusters $C_1, \ldots, C_k$ using k-means clustering (e.g., using Euclidean distance)

Output: Clusters $C_1, \ldots, C_k$

https://people.csail.mit.edu/dsontag/courses/ml14/notes/Luxburg07_tutorial_spectral_clustering.pdf

# Spectral Clustering - Algorithm

## Unnormalized spectral clustering

Input: similarity matrix $S \in \mathbb{R}^{n \times n}$, nur

- Construct a similarity graph and co… …d adjacency matrix, and $D$ (degree matrix)
- Compute the unnormalized graph Laplacian $L = D - W$
- Compute the first $k$ eigenvectors $u_1, \ldots, u_k$ of $L$
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the $i$-th row of $U$
- Cluster the points $y_i$ into clusters $C_1, \ldots, C_k$ using k-means clustering (e.g. using Euclidean distance)

Output: Clusters $C_1, \ldots, C_k$

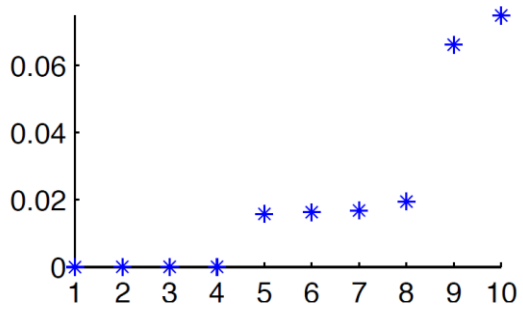
Normalized Laplacian: $L = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$

Random Walk Laplacian: $L = I - D^{-1} W$

https://people.csail.mit.edu/dsontag/courses/ml14/notes/Luxburg07_tutorial_spectral_clustering.pdf

# Selecting the optimal number of clusters $k^*$

- *Eigengap Heuristic*: choose $k^*$ such that the first $k$ eigenvalues $\lambda_1, \ldots, \lambda_k$ are very small, but $\lambda_{k+1}$ is relatively large




- We can also use the *Silhouette* score to select the optimal number of clusters (on the embedding space)

https://www.cs.cornell.edu/courses/cs6241/2020sp/readings/vonLuxburg-2007-spectral.pdf

# Agenda

- Clustering Algorithms
  - K-Means Clustering
  - Spectral Clustering
- **Choosing the optimal number of clusters**

# Choosing $k^*$ - Flipped Classroom Data

- Participants: 288 EPFL students of a course taught in *flipped classroom* mode with a duration of 10 weeks
- Structure:
  - Preparation: watch videos (and solve simple quizzes) on **new content** at home as a preparation for the lecture
  - Lecture: discuss open questions and solve more complex tasks
  - Lab session: solve paper-an-pen assignments
- Data: clickstream data (all interactions of the student with the system)

# Choosing $k^*$ - Your Turn

- In practice, clusters are not always as well separable…
- Your Task:
  1. Choose one of the suggested feature groups (Effort or Proactivity)
  2. Cluster the students based on these feature groups
  3. Compute (and visualize) the eigengap heuristic as well as the Silhouette score
  4. Discuss your findings: what number of clusters would you choose? Why?
- If you have time: repeat for the second feature group

# Summary

- K-Means Clustering
  - Popular, easy to implement
  - Assumptions: data points live in Euclidean space, spherical distribution around cluster centroids
  - Need to choose: initialization, distance measure (e.g., Euclidean distance), number of clusters $k$
- Spectral Clustering
  - Flexible, no assumptions about shape/form of clusters
  - Need to choose: similarity measure, computation of similarity graph, computation of graph Laplacian, number of clusters $k$