University Of Tamanrasset
Department of Mathematics and Computer Science
Ai engineering & Data science

Machine Learning

Spectral Clustering

Presented by : Hafidi Moussa & Zaki Alaadine

Table of contents

01

Abstract

Abstract

In recent years, spectral clustering has become one of the most popular modern clustering algorithms. It is simple to implement, can be solved efficiently by standard linear algebra software, and very often outperforms traditional clustering algorithms such as the k-means algorithm. On the first glance spectral clustering appears slightly mysterious, and it is not obvious to see why it works at all and what it really does. The goal of this tutorial is to give some intuition on those questions.

We describe different graph Laplacians and their basic properties, present the most common spectral clustering algorithms, and derive those algorithms from scratch by several different approaches. Advantages and disadvantages of the different spectral clustering algorithms are discussed

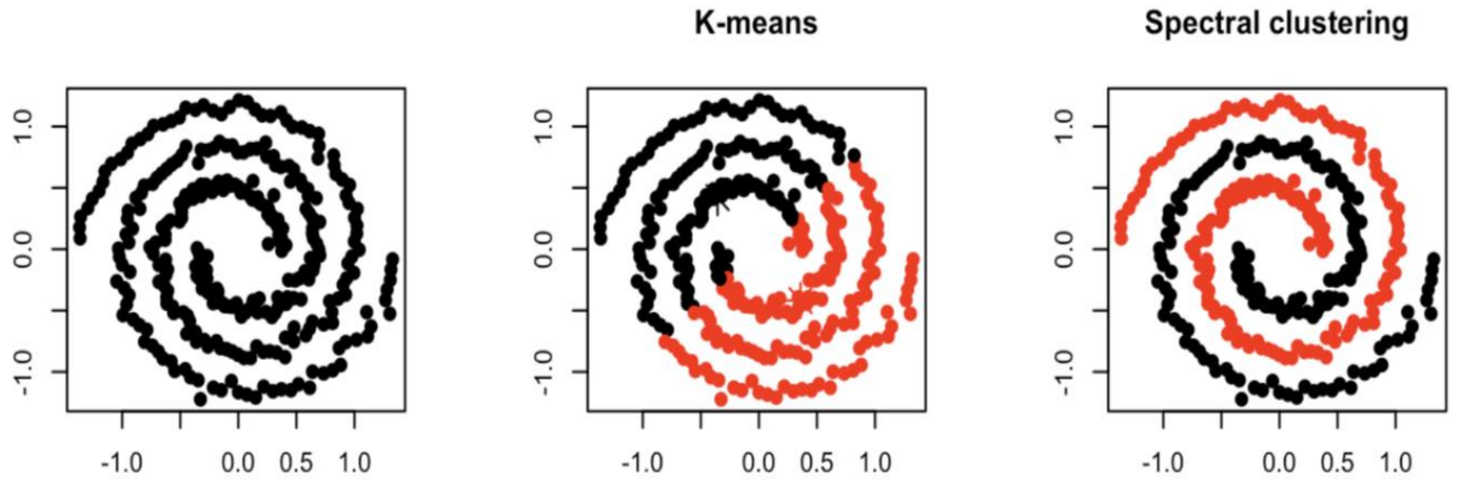
02

Introduction

Introduction

Clustering is one of the most widely used techniques for exploratory data analysis, with applications ranging from statistics, computer science, biology to social sciences or psychology. In virtually every scientific field dealing with empirical data, people attempt to get a first impression on their data by trying to identify groups of “similar behavior” in their data. In this article we would like to introduce the reader to the family of spectral clustering algorithms. Compared to the “traditional algorithms” such as k-means or single linkage, spectral clustering has many fundamental advantages. Results obtained by spectral clustering often outperform the traditional approaches, spectral clustering is very simple to implement and can be solved efficiently by standard linear algebra methods.

This tutorial is set up as a self-contained introduction to spectral clustering. We derive spectral clustering from scratch and present different points of view to why spectral clustering works. Apart from basic linear algebra. However, we do not attempt to give a concise review of the whole subject on spectral clustering, which is impossible due to the overwhelming amount of infromations on this subject.

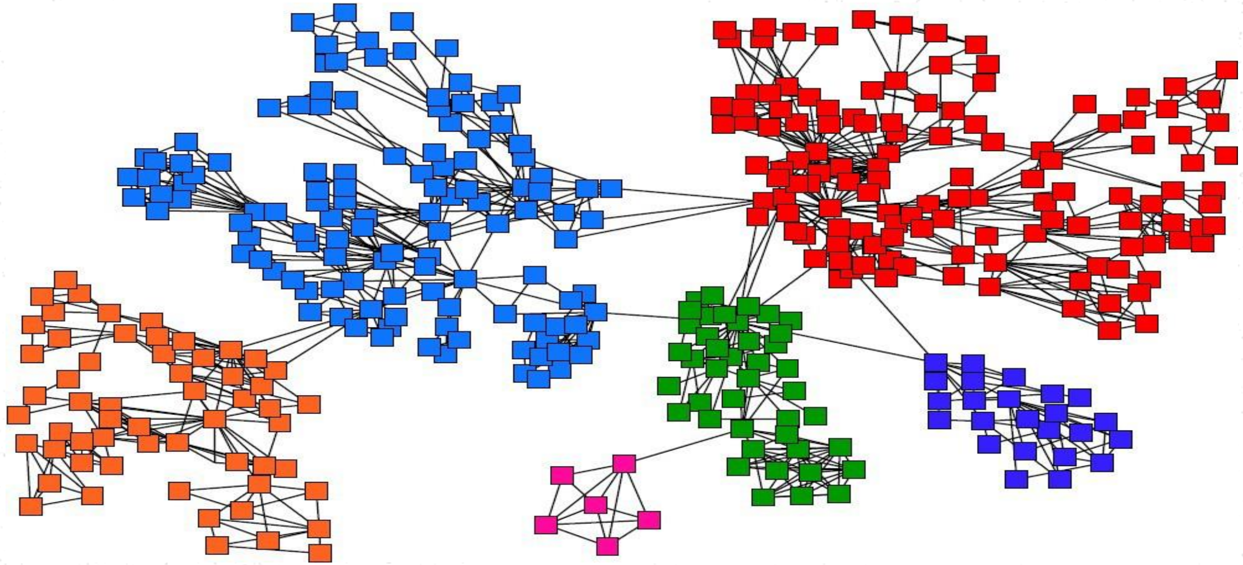

K-means
Spectral clustering
1.0
0.0
-1.0
-1.0
0.0
0.5
1.0

03

Similarity Graphs

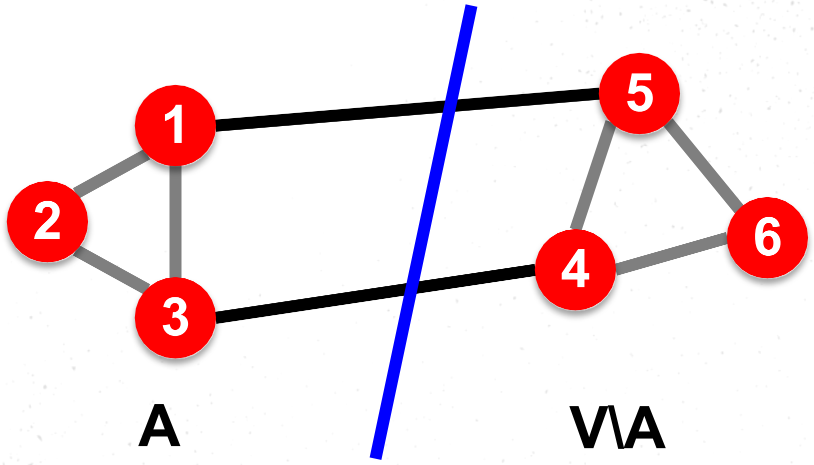
Similarity Graphs

The intuitive goal of clustering is to divide the data points into several groups such that points in the same group are similar and points in different groups are dissimilar to each other. If we do not have more information than similarities between data points, a nice way of representing the data is in form of the similarity graph G = (V,E).



What Makes A Good Cluster

- What makes a good cluster?
  - Maximize the number of within-cluster connections
  - Minimize the number of between-cluster connections

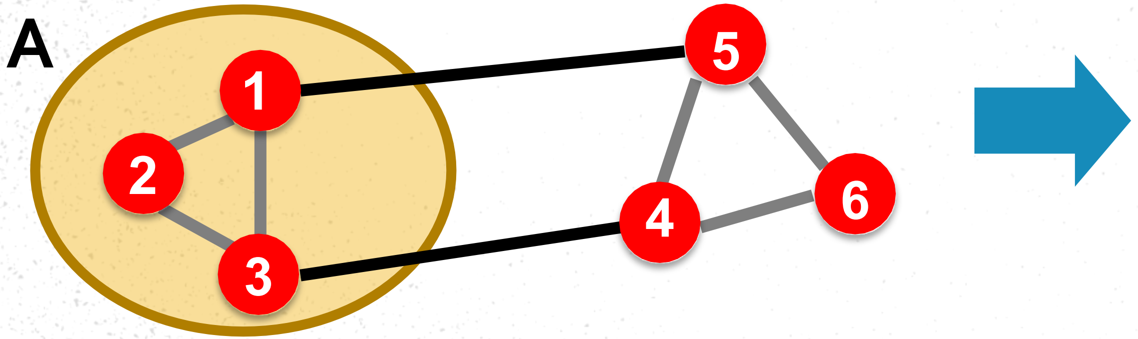


Graph Cuts

- **Express cluster quality as a function of the "edge cut" of the cluster**
- **Cut:** Set of edges (edge weights) with only one node in the cluster:

$$cut(A) = \sum_{i \in A, j \notin A} w_{ij}$$

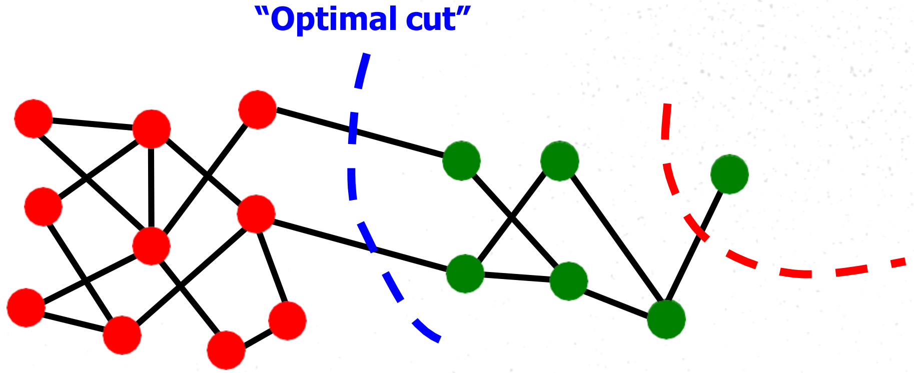
Note: This works for weighted and unweighted (set all $\mathbf{w_{ij}=1}$) graphs



$$cut(A) = 2$$

Cut Score

- **Partition quality : Cut score**
  - Quality of a cluster is the weight of connections pointing outside the cluster
- **Degenerate case:**



- **Problem:**
  - **Only considers external cluster connections**
  - **Does not consider internal cluster connectivity**

Graph Partitioning Criteria

Criterion: Conductance

Connectivity of the group to the rest of the network relative to the density of the group

The two most common objective functions to encode this are RatioCut (Hagen and Kahng, 1992) and the normalized cut Ncut (Shi and Malik, 2000). In RatioCut, the size of a subset A of a graph is measured by its number of vertices |A|, while in Ncut the size is measured by the weights of its edges vol(A). The definitions are:

$$\operatorname{RatioCut}(A_1, \ldots, A_k) := \frac{1}{2}\sum_{i=1}^{k}\frac{W(A_i, \overline{A}_i)}{|A_i|} = \sum_{i=1}^{k}\frac{\operatorname{cut}(A_i, \overline{A}_i)}{|A_i|}$$

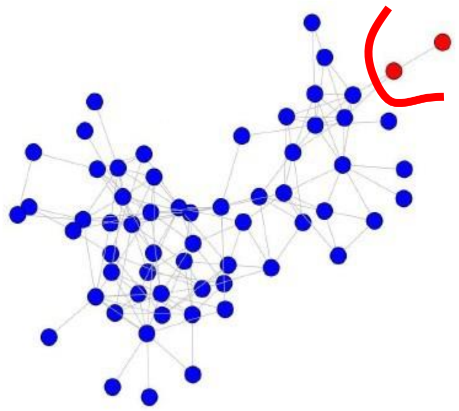
$$\operatorname{Ncut}(A_1, \ldots, A_k) := \frac{1}{2}\sum_{i=1}^{k}\frac{W(A_i, \overline{A}_i)}{\operatorname{vol}(A_i)} = \sum_{i=1}^{k}\frac{\operatorname{cut}(A_i, \overline{A}_i)}{\operatorname{vol}(A_i)}.$$

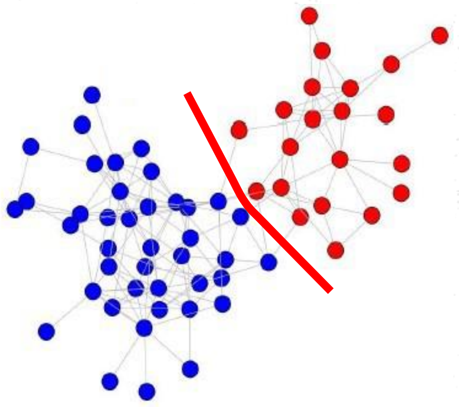
Note : The commonly used symbol for conductance is : ϕ

Why Use This criterion ?

Produces more balanced partitions

Example: Conductance Score





$\phi = 2/4 = 0.5$

$\phi = 6/92 = 0.065$

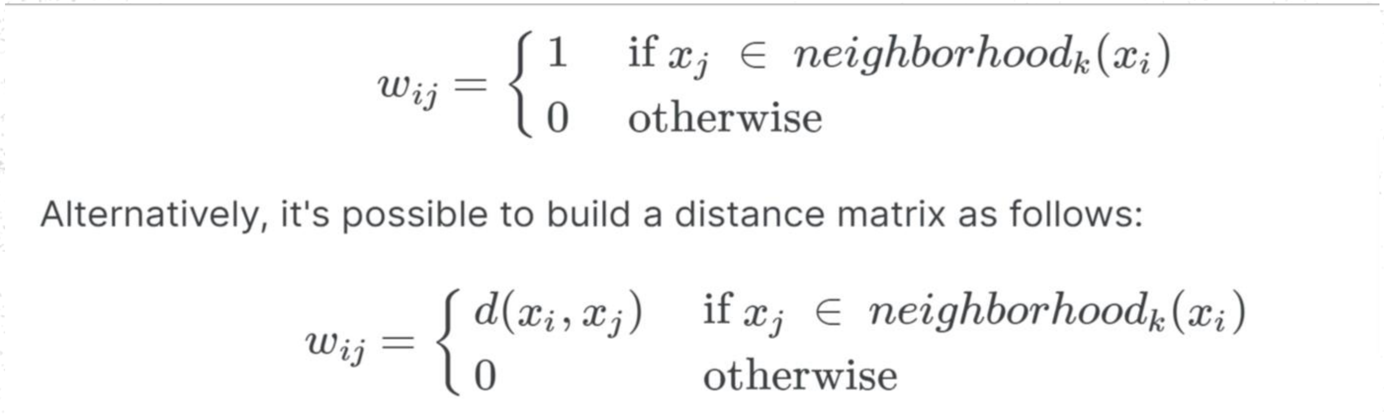
04

Similarity Matirx

Similarity Matrix

We can build a graph **G={V,E},** where the vertices are the points and the edges are determined using an affinity matrix W. Each element **wij** must express the affinity (similarity) between the points **xi** and **xj**. W is normally built using two different approaches:

k-nearest neighbors: In this case, we can build the number of neighbors to take into account for each point xi. W can be built as a connectivity matrix (expressing only the existence of a connection between two samples) if we adopt the criterion:



$$w_{ij} = \begin{cases} 1 & \text{if } x_j \in \textit{neighborhood}_k(x_i) \\ 0 & \text{otherwise} \end{cases}$$

Alternatively, it's possible to build a distance matrix as follows:

$$w_{ij} = \begin{cases} d(x_i, x_j) & \text{if } x_j \in \textit{neighborhood}_k(x_i) \\ 0 & \text{otherwise} \end{cases}$$

Similarity Matrix

The fully connected graph: The previous method can lead to graphs that are not fully connected because samples can exist that have no neighbors. In order to obtain a fully connected graph, An example for such a similarity function is the Gaussian similarity function :

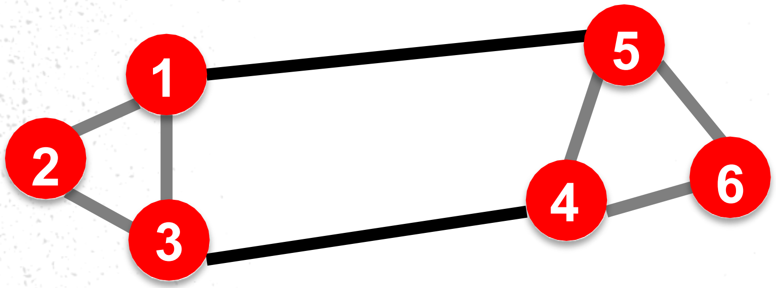
$$A_{ij} = \exp(-||s_i - s_j||^2/2\sigma^2)$$

if $i \neq j$, and $A_{ii} = 0$.

The parameter σ controls the width of the Gaussian function and determines how quickly the similarity decreases with distance. Smaller values of σ result in a sharper decrease and a more localized similarity measure, while larger values of σ lead to a smoother decrease and a more spread-out similarity measure

Example 01 :

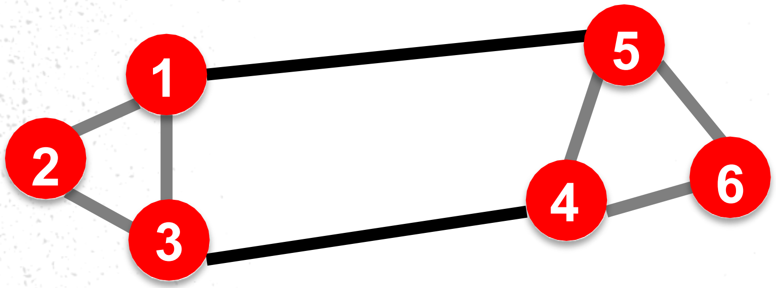
Connectivity matrix :



	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

Example 02 :

Similarity matrix :



$$\begin{array}{c|cccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & 0 & 0.8 & 0.2 & 0 & 0.6 & 0 \\
2 & 0.2 & 0 & 0.9 & 0 & 0 & 0 \\
3 & 0.3 & 0.4 & 0 & 1 & 0 & 0 \\
4 & 0 & 0 & 1 & 0 & 0.3 & 0.6 \\
5 & 0.7 & 0 & 0 & 0.8 & 0 & 0.1 \\
6 & 0 & 0 & 0 & 0.7 & 0.5 & 0
\end{array}$$

05

Graph Laplacian

Graph Laplacian Matrix

❖ The Laplacian graph, also known as the Laplacian matrix, is a matrix representation of a graph that encodes the pairwise relationships between data points. It plays a fundamental role in spectral clustering algorithms.

❖ The role of the Laplacian matrix in spectral clustering is multifaceted:

1) **Capturing Graph Structure:** The Laplacian matrix captures the underlying graph structure represented by the similarity or affinity matrix. It encodes the pairwise relationships between data points and reflects the connectivity of the graph.

2) **Spectral Information**: The Laplacian matrix's eigenvalues and eigenvectors contain valuable spectral information about the graph. By analyzing these eigenvalues and eigenvectors, we can uncover the latent structure, clusters, or patterns in the data.

3) **Dimensionality Reduction:** The eigenvectors corresponding to the smallest eigenvalues of the Laplacian matrix can be used for dimensionality reduction. These eigenvectors form a reduced-dimensional representation of the data that retains the essential clustering information.

Graph Laplacian Matrix

4) **Graph Partitioning:** Spectral clustering utilizes the Laplacian matrix to partition the data points into clusters. By applying clustering algorithms, such as k-means, on the reduced eigenvector space, the data points can be grouped into distinct clusters.

5) **Normalization:** Different forms of the Laplacian matrix, such as the normalized or symmetric normalized Laplacians, offer normalization techniques that can improve the performance and robustness of spectral clustering. Normalization accounts for variations in graph connectivity and scales the Laplacian matrix accordingly.

❖ In summary, the Laplacian graph or Laplacian matrix is a crucial component in spectral clustering. It captures the graph structure, provides spectral information, enables dimensionality reduction, facilitates graph partitioning, and offers normalization options. These properties make spectral clustering an effective algorithm for clustering and analyzing data with complex structures or non-linear separability.

Graph Laplacian Matrix

- The Laplacian matrix is derived from the affinity or similarity matrix of the data points. It is computed by subtracting the adjacency matrix (or a transformation of it) from a diagonal degree matrix. There are different formulations of the Laplacian matrix, including unnormalized, normalized, and symmetric normalized Laplacians .

1) **The unnormalized graph Laplacian :** The unnormalized graph Laplacian matrix is defined as :

$$L = D - W.$$

D is the diagonal (degree) matrix
W is the affinity (similarity) matrix

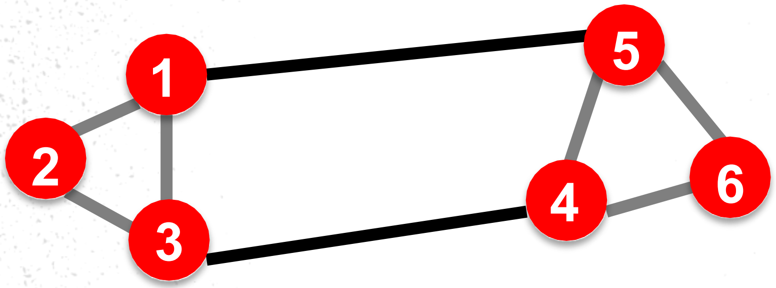
2) **The normalized graph Laplacians :** There are two matrices which are called normalized graph Laplacians in the literature. Both matrices are closely related to each other and are defined as :

$$L_{\text{sym}} := D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$$
$$L_{\text{rw}} := D^{-1} L = I - D^{-1} W.$$

I is the identity matrix

Adjacency Matrix

Connectivity matrix "W" :



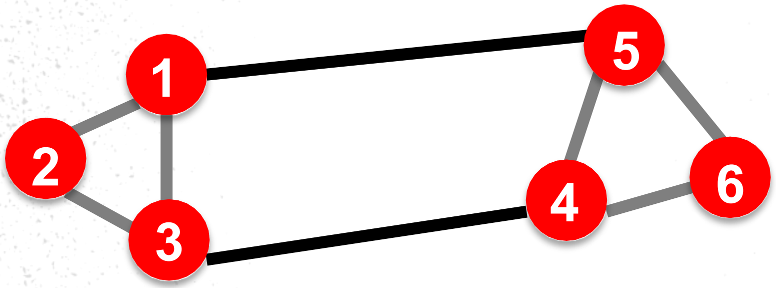
$$
\begin{array}{c|cccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
2 & 1 & 0 & 1 & 0 & 0 & 0 \\
3 & 1 & 1 & 0 & 1 & 0 & 0 \\
4 & 0 & 0 & 1 & 0 & 1 & 1 \\
5 & 1 & 0 & 0 & 1 & 0 & 1 \\
6 & 0 & 0 & 0 & 1 & 1 & 0
\end{array}
$$

❖ **Important properties :**

1. Symmetric matrix
2. Eigenvectors are real and orthogonal

The Degree Matrix

The Diagonal Matrix “D” :



$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 3 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 3 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 3 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 3 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 2 \end{array}$$

Example :

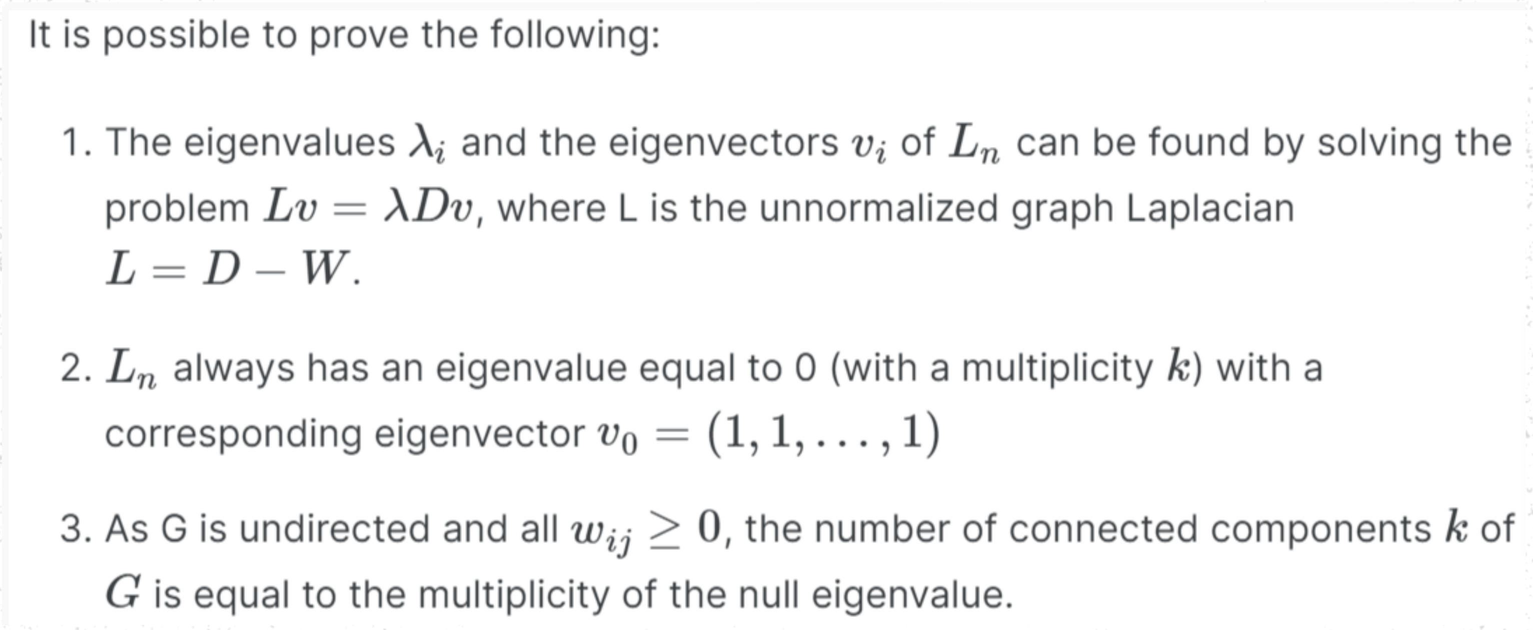
Connectivity matrix :

$$L = D - W. \longrightarrow \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 3 & -1 & -1 & 0 & -1 & 0 \\ 2 & -1 & 2 & -1 & 0 & 0 & 0 \\ 3 & -1 & -1 & 3 & -1 & 0 & 0 \\ 4 & 0 & 0 & -1 & 3 & -1 & -1 \\ 5 & -1 & 0 & 0 & -1 & 3 & -1 \\ 6 & 0 & 0 & 0 & -1 & -1 & 2 \end{array}$$

06

Eigendecomposition

Graph Laplacian Matrix



It is possible to prove the following:

1. The eigenvalues λ_i and the eigenvectors v_i of L_n can be found by solving the problem $Lv = \lambda Dv$, where L is the unnormalized graph Laplacian $L = D - W$.
2. L_n always has an eigenvalue equal to 0 (with a multiplicity k) with a corresponding eigenvector $v_0 = (1, 1, \ldots, 1)$
3. As G is undirected and all $w_{ij} \geq 0$, the number of connected components k of G is equal to the multiplicity of the null eigenvalue.

Solving for Eigenvalues and Eigenvectors

$$A - \lambda I = \begin{bmatrix} 1 & 1 \\ 4 & 1 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} = \begin{bmatrix} 1-\lambda & 1 \\ 4 & 1-\lambda \end{bmatrix}$$

$$\begin{vmatrix} 1-\lambda & 1 \\ 4 & 1-\lambda \end{vmatrix} = 0$$

set determinant equal to zero

solve equation to find eigenvalues

$$(1-\lambda)(1-\lambda) - (4)(1) = 0$$

$$1 - 2\lambda + \lambda^2 - 4 = 0$$

$$\lambda^2 - 2\lambda - 3 = 0$$

$$(\lambda - 3)(\lambda + 1) = 0$$

$$\lambda = 3, \lambda = -1$$

Solving for Eigenvalues and Eigenvectors

$$A = \begin{bmatrix} 1 & 1 \\ 4 & 1 \end{bmatrix} \qquad \lambda = 3 \qquad \lambda = -1 \qquad (A - \lambda I)\vec{x} = \vec{0}$$

$$A - 3I = \begin{bmatrix} 1 & 1 \\ 4 & 1 \end{bmatrix} - \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 4 & -2 \end{bmatrix} \longrightarrow \begin{bmatrix} -2 & 1 \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -2 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \begin{matrix} -2x_1 + x_2 = 0 \\ x_2 = 2 \end{matrix} \qquad \vec{x} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

any multiple of this vector also works

Solving for Eigenvalues and Eigenvectors

$$A = \begin{bmatrix} 1 & 1 \\ 4 & 1 \end{bmatrix} \qquad \lambda = 3 \qquad \lambda = -1 \qquad (A - \lambda I)\vec{x} = \vec{0}$$

$$A - (-1)I = \begin{bmatrix} 1 & 1 \\ 4 & 1 \end{bmatrix} - \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 4 & 2 \end{bmatrix} \longrightarrow \begin{bmatrix} 2 & 1 \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \begin{gathered} 2x_1 + x_2 = 0 \\ x_2 = -2 \end{gathered} \qquad \vec{x} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

any multiple of this vector also works

Fact :

$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 3 & -1 & -1 & 0 & -1 & 0 \\ 2 & -1 & 2 & -1 & 0 & 0 & 0 \\ 3 & -1 & -1 & 3 & -1 & 0 & 0 \\ 4 & 0 & 0 & -1 & 3 & -1 & -1 \\ 5 & -1 & 0 & 0 & -1 & 3 & -1 \\ 6 & 0 & 0 & 0 & -1 & -1 & 2 \end{array} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \vec{0}$$

07

Spectral Algorithms

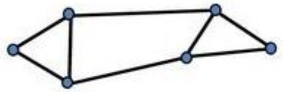
Spectral Clustering Algorithms

- **Three basic stages:**
  - 1) **Pre-processing**
    - **Construct a matrix representation of the graph**
  - 2) **Decomposition**
    - **Compute eigenvalues and eigenvectors of the matrix**
    - **Map each point to a lower-dimensional representation based on one or more eigenvectors**
  - 3) **Grouping**
    - **Assign points to two or more clusters, based on the new representation**

Spectral Partitioning Algorithm

Pre-processing:

Build Laplacian matrix L of the graph



	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

Decomposition:

– Find eigenvalues λ and eigenvectors x of the matrix L

λ =

0.0
1.0
3.0
3.0
4.0
5.0

X =

0.4	0.3	-0.5	-0.2	-0.4	-0.5
0.4	0.6	0.4	-0.4	0.4	0.0
0.4	0.3	0.1	0.6	-0.4	0.5
0.4	-0.3	0.1	0.6	0.4	-0.5
0.4	-0.3	-0.5	-0.2	0.4	0.5
0.4	-0.6	0.4	-0.4	-0.4	0.0

– Map vertices to corresponding components of λ_2

1	0.3
2	0.6
3	0.3
4	-0.3
5	-0.3
6	-0.6

How do we now find the clusters?

Spectral Partitioning Algorithm

Grouping:

- Sort components of reduced 1-dimensional vector
- Identify clusters by splitting the sorted vector in two

- How to choose a splitting point?
  - Naïve approaches:
    - Split at 0 or median value

1	0.3
2	0.6
3	0.3
4	-0.3
5	-0.3
6	-0.6

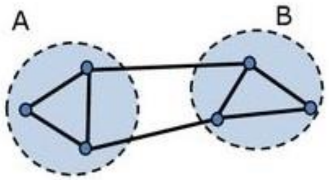
Split at 0:

Cluster A: Positive points

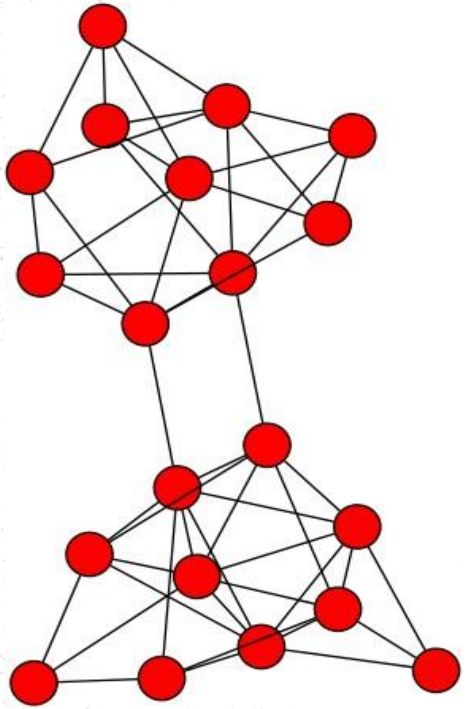
Cluster B: Negative points

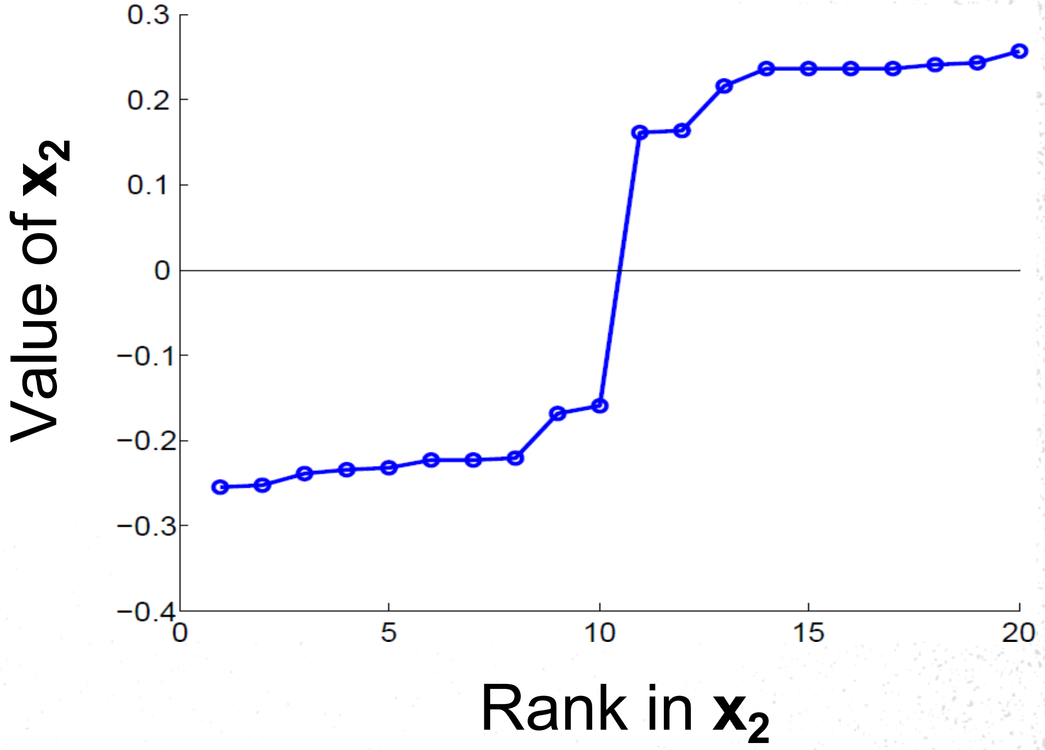
1	0.3
2	0.6
3	0.3

4	-0.3
5	-0.3
6	-0.6

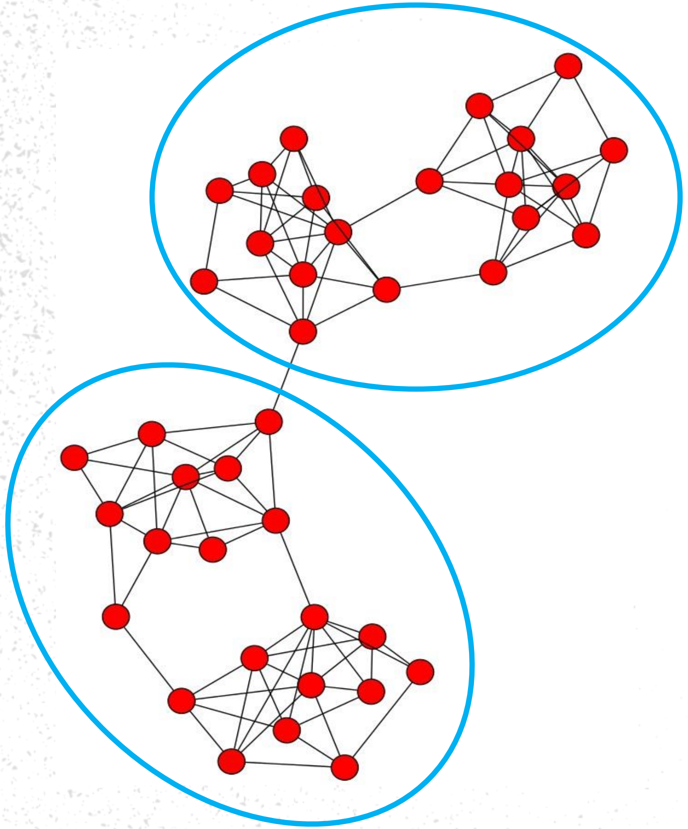


Example: Spectral Partitioning

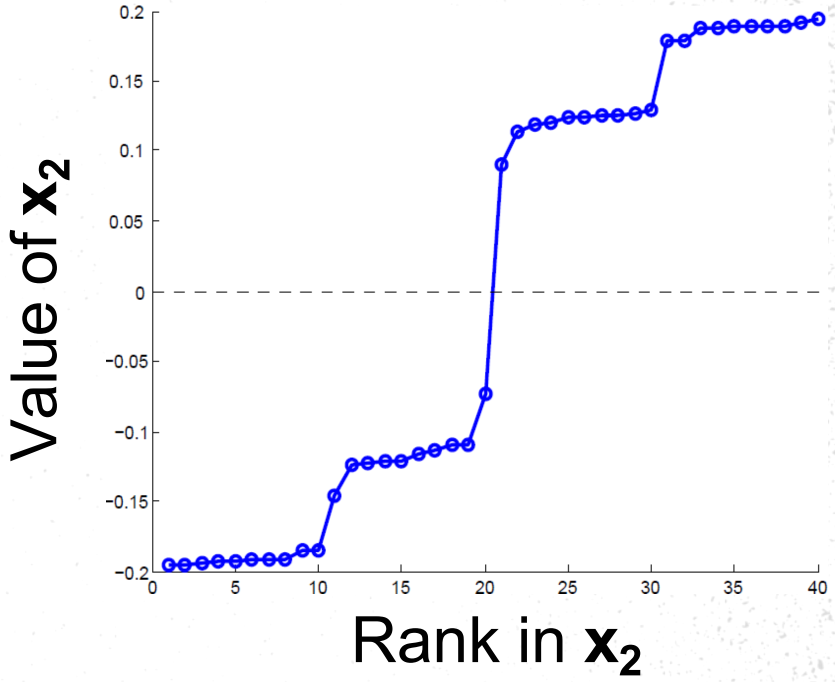




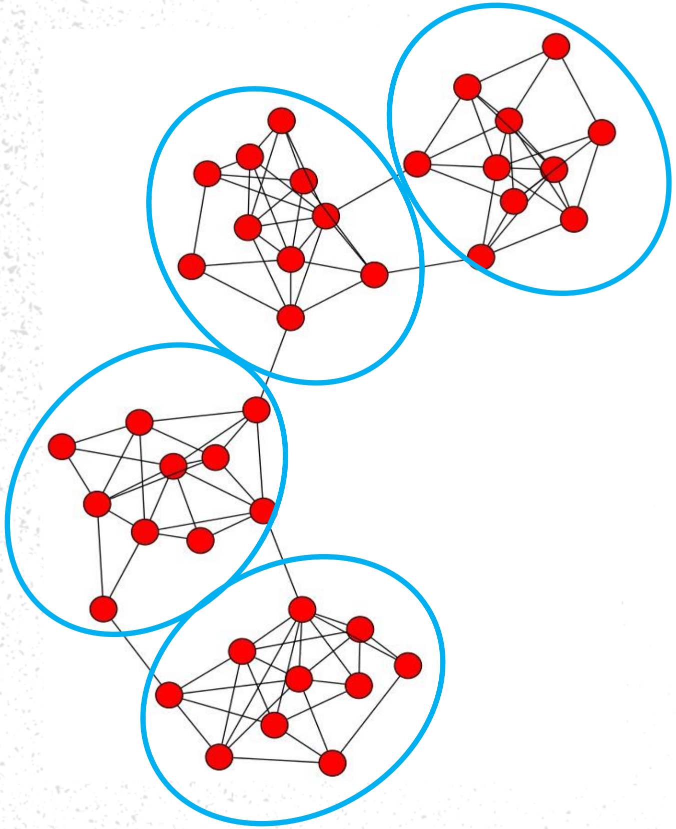
Example: Spectral Partitioning

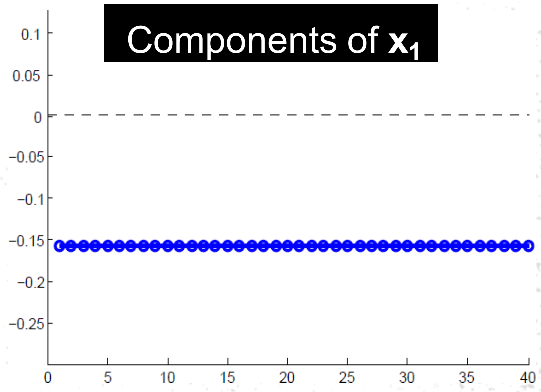


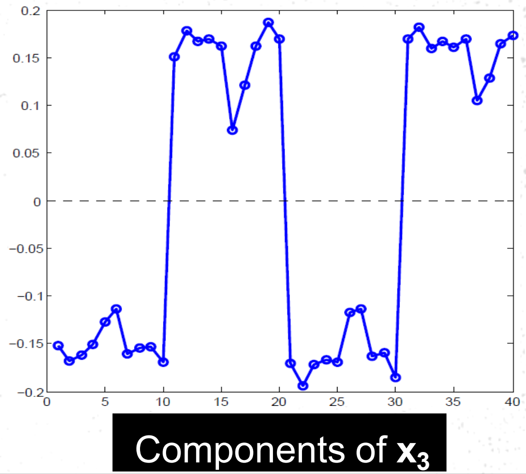
Components of x_2



Example: Spectral Partitioning







k-Way Spectral Clustering

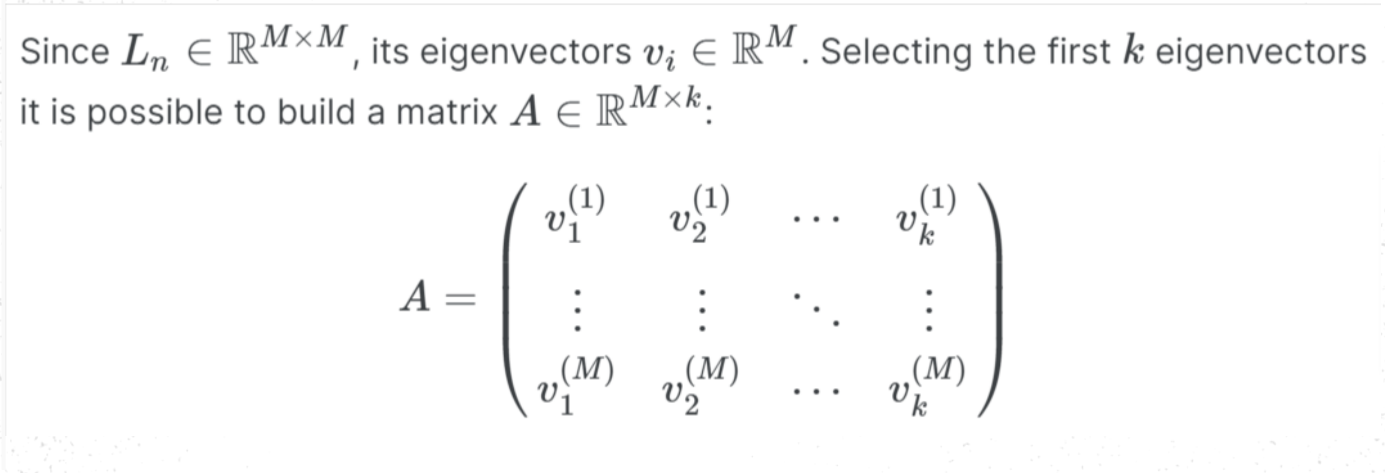
- How do we partition a graph into *k* clusters?

- Two basic approaches:
  - Recursive bi-partitioning [Hagen et al., '92]
    - Recursively apply bi- partitioning algorithm in a hierarchical divisive manner
    - Disadvantages: IneflLicient, unstable
  - Cluster multiple eigenvectors [Shi-Malik, '00]
    - Build a reduced space from multiple eigenvectors
    - Commonly used in recent papers
    - A preferable approach…

Shi-Malik spectral clustering algorithm

1) Select a graph construction method **k-NN** or **Gaussian kernel:**
2) Select the parameter **k** or the parameter **σ**
3) Select the expected number of clusters **k**
4) Compute the matrices **W** and **D**
5) Compute the normalized graph Laplacian **Ln**
6) Compute the first k eigenvectors of **Ln**
7) Build the matrix **A**
8) Cluster the rows of **A** using **k-means** or **k-means++**

Transformed Matrix « A »



Since $L_n \in \mathbb{R}^{M \times M}$, its eigenvectors $v_i \in \mathbb{R}^M$. Selecting the first k eigenvectors it is possible to build a matrix $A \in \mathbb{R}^{M \times k}$:

$$A = \begin{pmatrix} v_1^{(1)} & v_2^{(1)} & \cdots & v_k^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ v_1^{(M)} & v_2^{(M)} & \cdots & v_k^{(M)} \end{pmatrix}$$

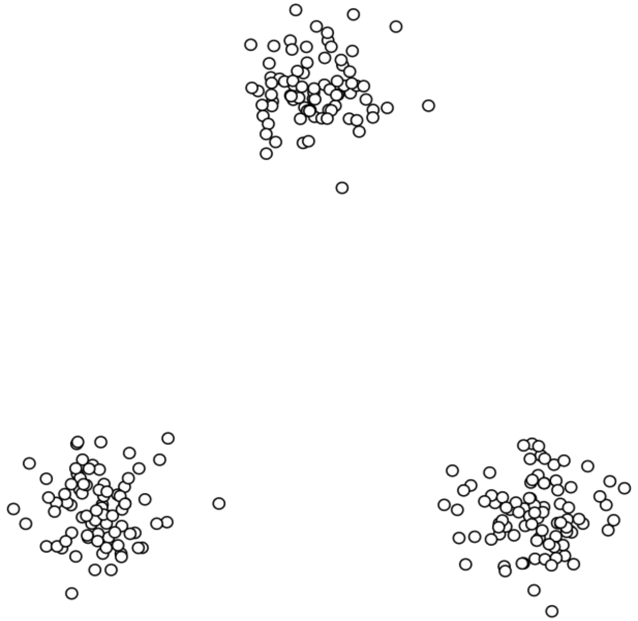
❖ **Feature Scaling :** subtracts the mean and divides by the standard deviation for each feature, resulting in a dataset where each feature has a mean of 0 and a standard deviation of 1.

K-mean Algorithm

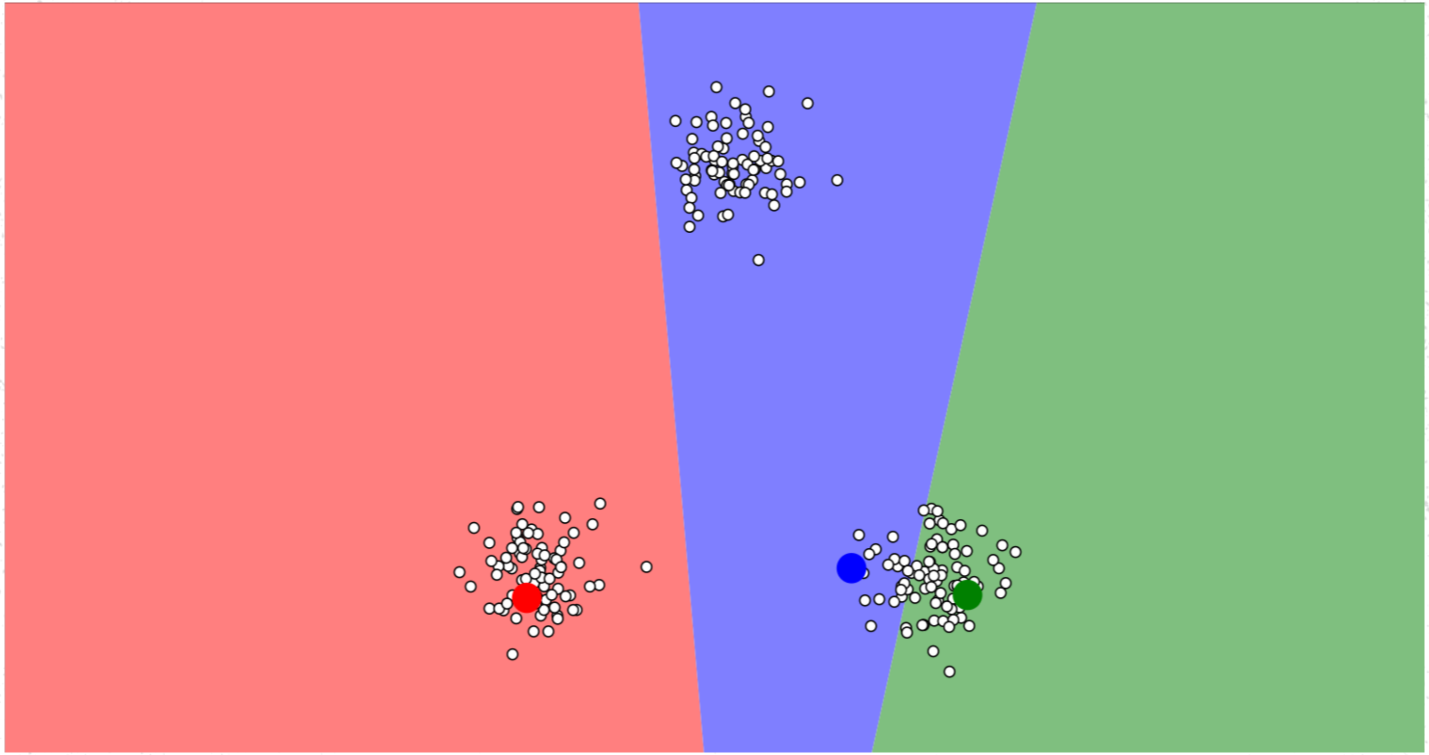
The k-means algorithm captures the insight that each point in a cluster should be near to the center of that cluster. It works like this: first we choose k, the number of clusters we want to find in the data. Then, the centers of those k clusters, called *centroids*, are initialized in some fashion.

The algorithm then proceeds in two alternating parts: In the *Reassign Points* step, we assign every point in the data to the cluster whose centroid is nearest to it. In the *Update Centroids* step, we recalculate each centroid's location as the mean (center) of all the points assigned to its cluster. We then iterate these steps until the centroids stop moving, or equivalently until the points stop switching clusters

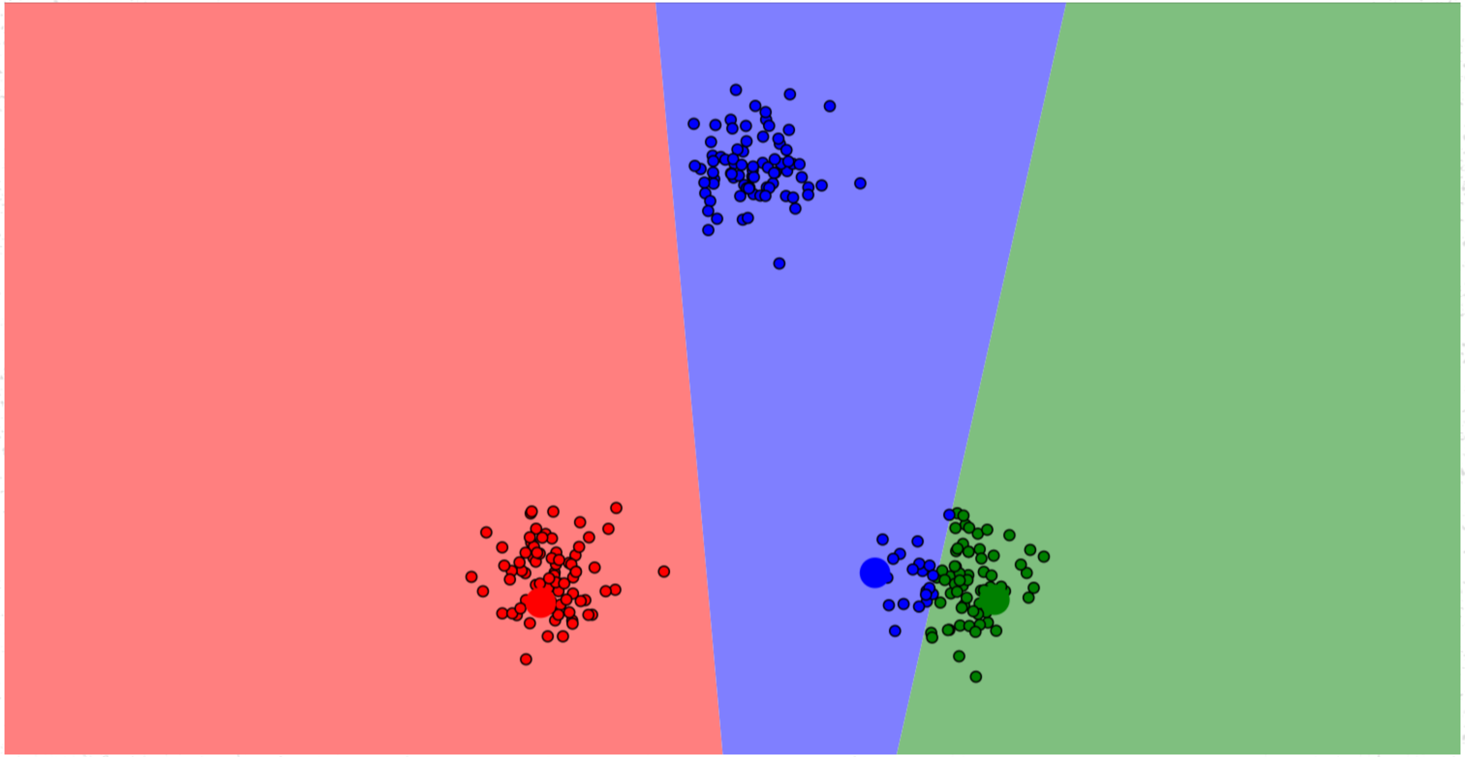
K-mean clustering



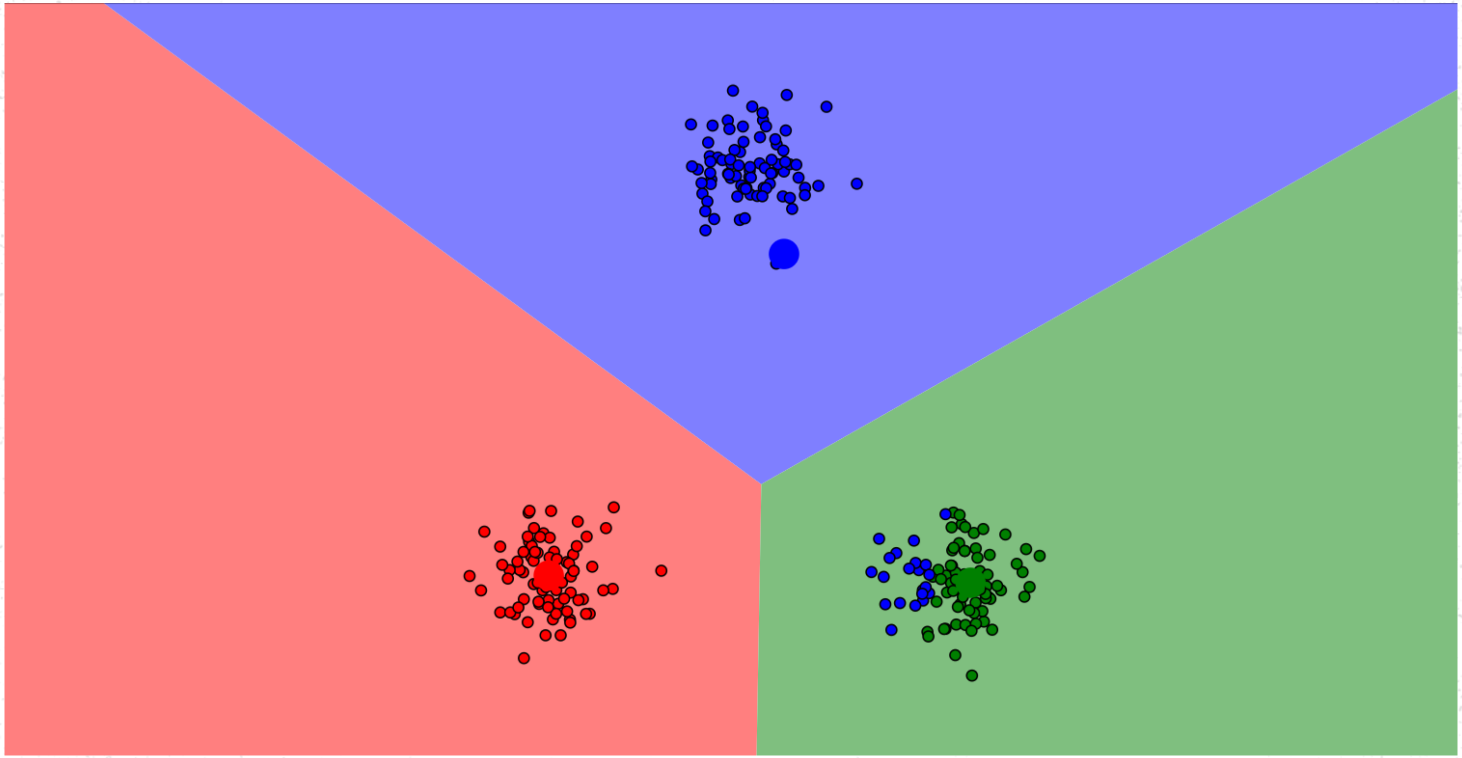
K-mean clustering



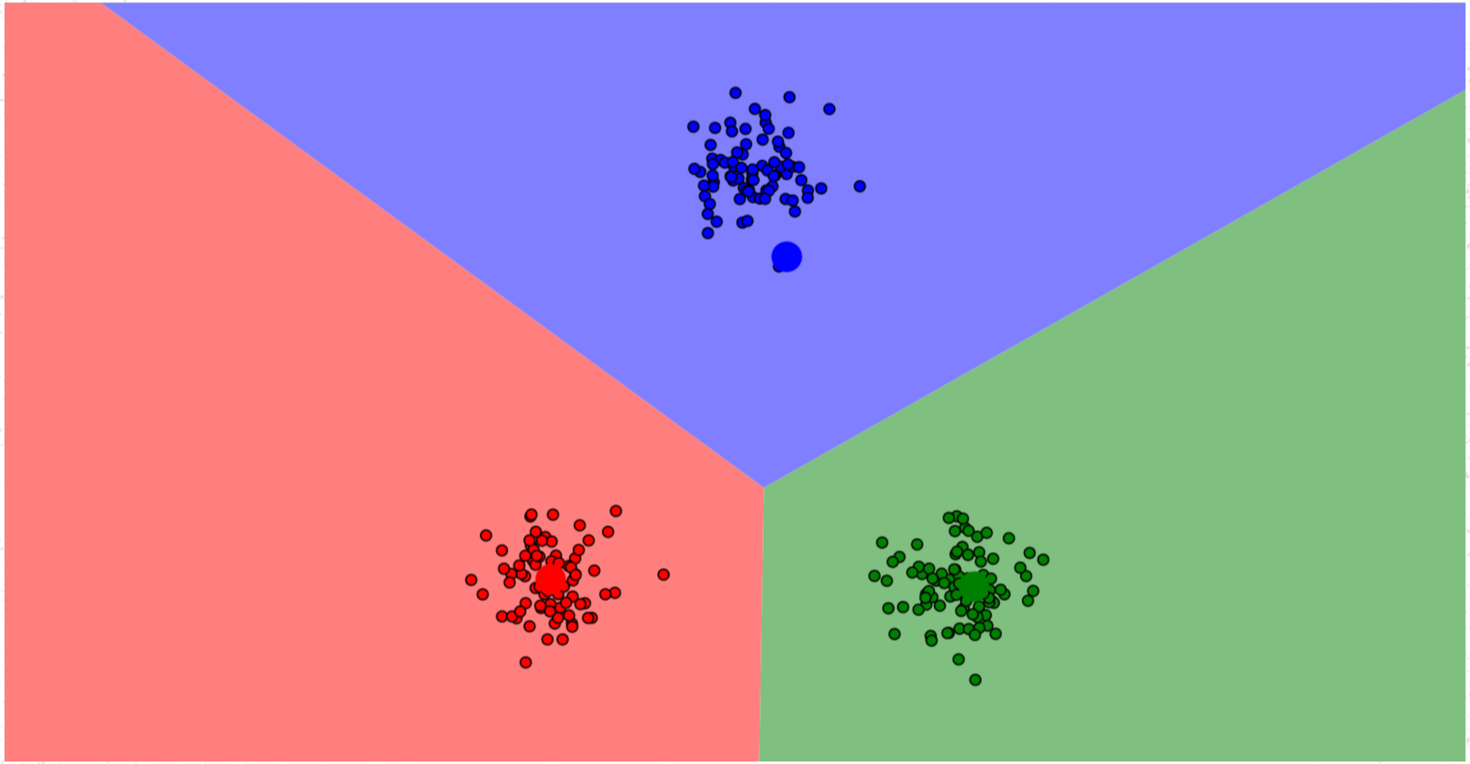
K-mean clustering



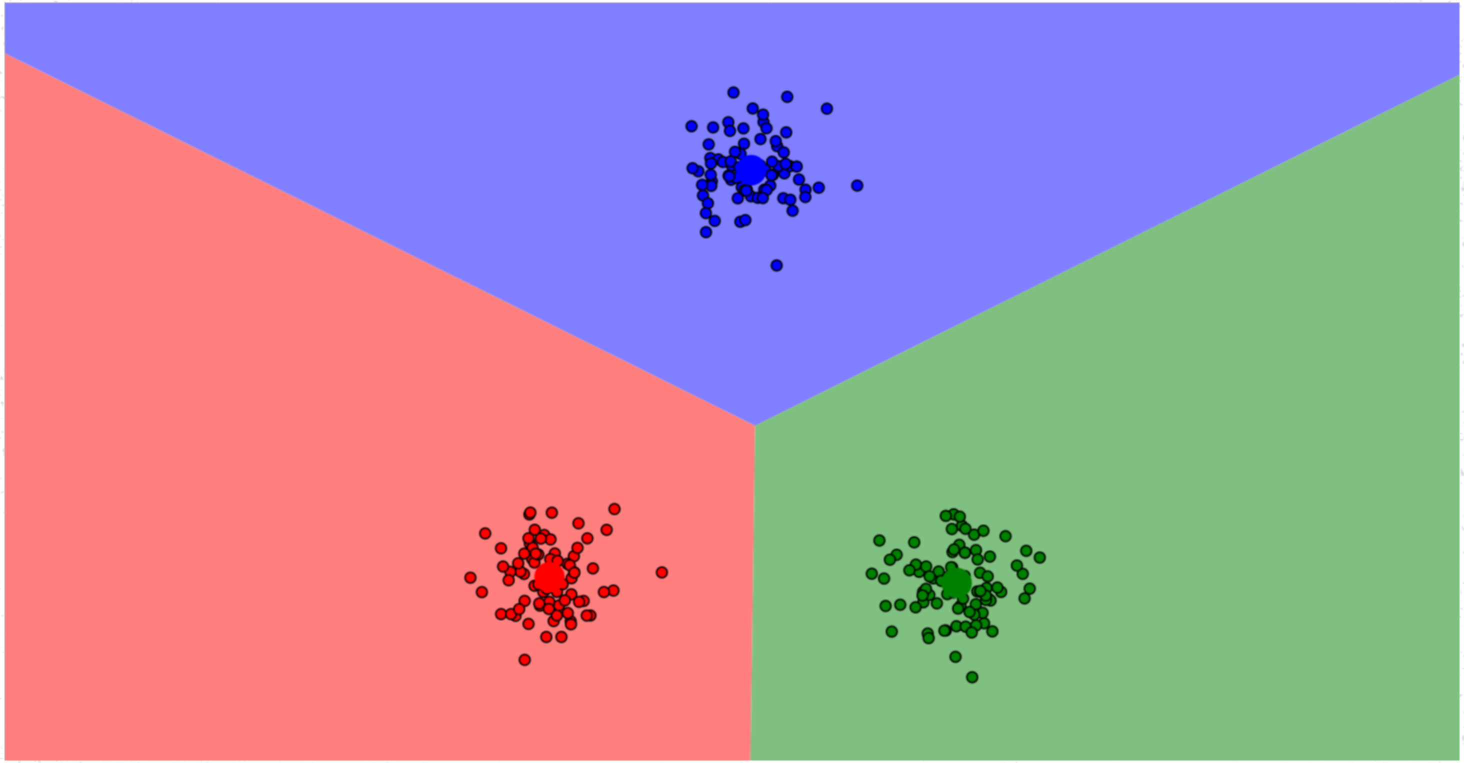
K-mean clustering



K-mean clustering



K-mean clustering



08

Applications Pros & Cons

Applications:

Image segmentation: Grouping pixels in an image based on their color or intensity.

Document clustering: Grouping documents based on their content similarity.

Social network analysis: Identifying communities of users in a social network.

Bioinformatics: Clustering genes based on their expression patterns

Advantages:

Handles non-convex shapes: Unlike k-means clustering, spectral clustering can effectively handle data with non-convex shapes and complex structures.

Global perspective: It considers the global structure of the data, leading to more robust and accurate clustering results.

Flexibility: It can be adapted to various data types and similarity measures.

Limitations:

Computational complexity: Spectral clustering can be computationally expensive, especially for large datasets.

Choice of parameters: The performance of spectral clustering depends on the choice of parameters, such as the number of clusters and the similarity measure.

Interpretability: Understanding the clusters formed by spectral clustering can be challenging due to the underlying mathematical concepts.

Resources

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

https://ai.stanford.edu/~ang/papers/nips01-spectral.pdf

https://scikitlearn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html

https://people.eecs.berkeley.edu/~malik/papers/SM-ncut.pdf

https://elizavetalebedeva.com/ml-with-graphs-notes-part-1/

Thanks!

Do you have any questions?

Created By :
Hafidi Moussa & Zaki Alaadine