Spectral Clustering

Theory, Hyperparameters and Application

Léo Nicollier, Noah Scheider, Agnès Haldi

Statistical Machine Learning with Guillaume Obozinski

December 2022

Assumption: the data has a certain form, ie K roundish cluster of similar sizes with centroids

$$\mu_1, \dots, \mu_K$$

the algorithm is a minimization problem :

$$\min_{\mu,\dots,\mu_K} (\frac{1}{n} \sum_{i=1}^{n} \min_k (\|x_i - \mu_k\|^2))$$

the algorithm assigns each point to its closest class, from solving the minimization problem above.
then we recompute the

$$\mu_i$$

and solve the minimization problem again.

- **Similarities** between our data points $\rightarrow$ clusters

- **Similarity Matrix** : *intuitive goal* find a **disjoint partition** of this graph $\rightarrow$ this "cutting" problem gives conditions too complicated to solve

Then :

Construction of two matrices W and D s.t. :

$(W)_{ij} = w_{ij}$ the **weight** between our two data points at a coordinate $(i; j)$

D is **diagonal** formed by the vector $d_i = \sum_{i=2}^{n} w_{ij}$

(for j fixed and $n = \#$ of lines of our graph)

Intuition

Use Spectral graph theory. We will so construct a **Laplacian Graph**.

$$L = \begin{pmatrix} L_1 & & & \\ & L_2 & & \\ & & \dots & \\ & & & L_k \end{pmatrix}$$

and take the k **smallest eigenvalues** and their corresponding **eigenvectors** to then form the centroids

$$\mu_i \quad (i = 1, \dots, k)$$

for the k-means algorithm.

Differences with k-means ?

For k-means : **roundish** formed data
No similar need for spectral clustering

Spectral clustering uses k-means in its last step but the **algebraic transformation** we compute on the graph beforehand allows it to work in a more efficient way.

The use of **graphs**

Algorithm as in von Luxburg, 2007

a) **Input:** Similarity matrix number of cluster

b) Construct a similarity graph

c) Compute Laplacian Graph L

d) Get first k eigenvectors of L

e) For $U \in \mathbb{R}^{nxk}$ the matrix containing these vectors as columns, its lines y_i in $\mathbb{R}^k$ will be our centroids for the k-means algorithm $\rightarrow$ we will obtain the clusters $C_1, \ldots, C_k$.

f) **Output:** Clusters $A_1, \ldots, A_k$ with $A_i = \{j \ s.t. \ y_j \in C_i\}$

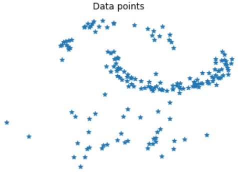
The Laplacian Graph

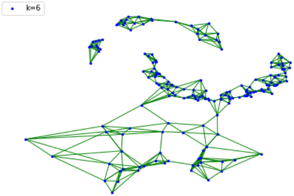
unnormalized Laplacian: $L = D - W$
normalized Laplacian: $L_{sym} := I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$
normalized Laplacian: ($\sim$ random walks) $L_{rw} := I - D^{-1} W$

Graph $G = (V, E)$, with vertices $V := \{v_1, \ldots, v_n\} \equiv \{x_1, \ldots, x_n\}$ and edges $S := E$ being a connection measure we denote as $e_{ij} = v_i \sim v_j, \forall i, j = 1, \ldots, n$

k-**nearest neighbor:** Transform E into S with Hyperparameter k
$K_i = \arg\min_{I \subset V, |I| = k} \sum_{v_j \in I} \|v_i - v_j\|_2$

$$(S)_{ij} := e_{ij} = \begin{cases} \|v_i - v_j\|_2, & \text{if } v_i \in K_j \vee v_j \in K_i \\ 0, & \text{else} \end{cases}$$



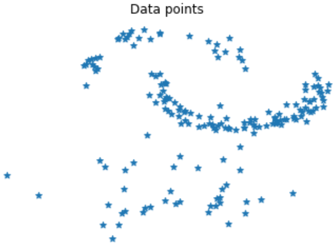


Similarity Matrix

Mutual k**-neighbor:** Transform E into S with Hyperparameter k

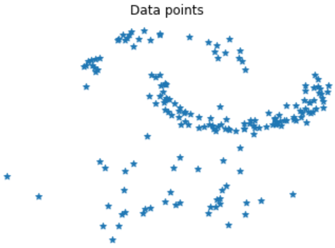
$K_i = \arg\min_{I \subset V, |I|=k} \sum_{v_j \in I} \|v_i - v_j\|_2$

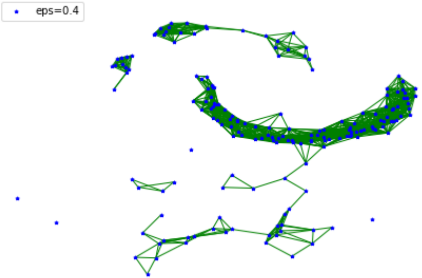
$$(S)_{ij} := e_{ij} = \begin{cases} \|v_i - v_j\|_2, & \text{if } v_i \in K_j \wedge v_j \in K_i \\ 0, & \text{else} \end{cases}$$



ϵ-**neighborhood:** Transform E into S with Hyperparameter $\epsilon > 0$

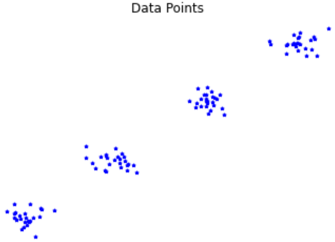
$$(S)_{ij} := e_{ij} = \begin{cases} \|v_j - v_i\|_2, & \text{if } v_j \in \mathcal{B}_\epsilon(v_i) \\ 0, & \text{else} \end{cases}$$

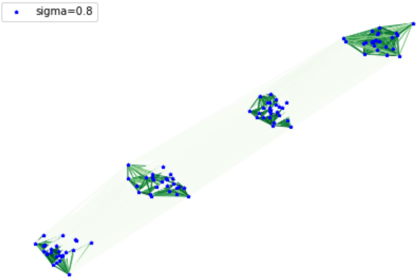


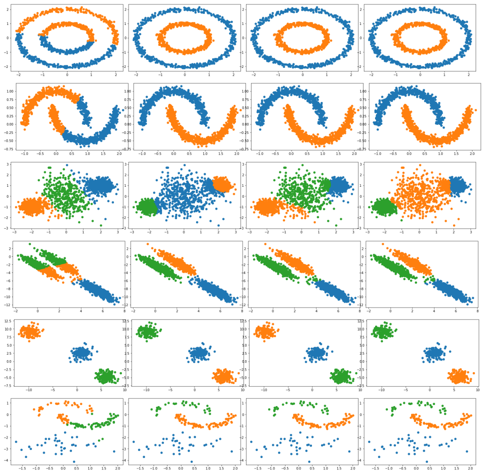


Fully gaussian connected: Transform E into S with Hyperparameter $\sigma > 0$, but sparsity issue

$$(S)_{ij} := e_{ij} = \mathrm{e}^{\frac{-\|v_i - v_j\|^2}{2\sigma^2}}$$







Algos = [Kmeans, UnormalizedSC, NormalisedSymSC, NormalisedRWSC]
Sim = SimKNearestNeighborGraphs(12)

Choose k

Split the data into two part A and B (randomnly)
for k in $K = [k_1, \ldots, k_r]$:
Cluster A and B with a method of spectral clustering with hyperparameter k. ($C_1^A, \ldots, C_k^A, C_1^B, \ldots, C_k^B$ the different clusters)
for C_i^A in $[C_1^A, \ldots, C_k^A]$:
Look in which cluster C_i^A is sent in the dataset B and stock in a matrix
for C_i^B in $[C_1^B, \ldots, C_k^B]$:
Look in which cluster C_i^B is sent in the dataset A and stock in a matrix
end

We obtain two matrices C_A^k and C_B^k such that:

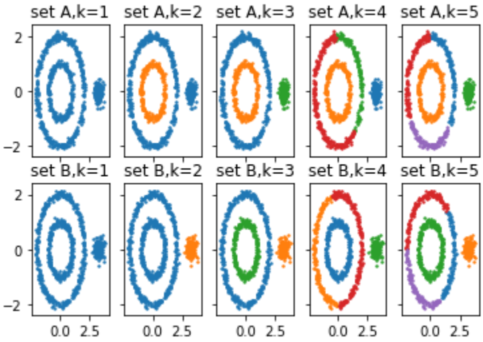
$$C_A^k = \begin{bmatrix} C_{A_1 \to B_1} & \cdots & C_{A_1 \to B_k} \\ \vdots & \ddots & \vdots \\ C_{A_k \to B_1} & \cdots & C_{A_k \to B_k} \end{bmatrix} \text{ with } C_{A_i \to B_j} = \mathbb{P}(A_i \text{ sent in } B_j)$$

$$C_B^k = \begin{bmatrix} C_{B_1 \to A_1} & \cdots & C_{B_1 \to A_k} \\ \vdots & \ddots & \vdots \\ C_{B_k \to A_1} & \cdots & C_{B_k \to A_k} \end{bmatrix} \text{ with } C_{B_i \to A_j} = \mathbb{P}(B_i \text{ sent in } A_j)$$

$\implies$ biggest k s.t. C_A^k and C_B^k are closest to a shuffled identity matrix.

$$C_B^3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$C_A^4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0.27 & 0 & 0.73 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$



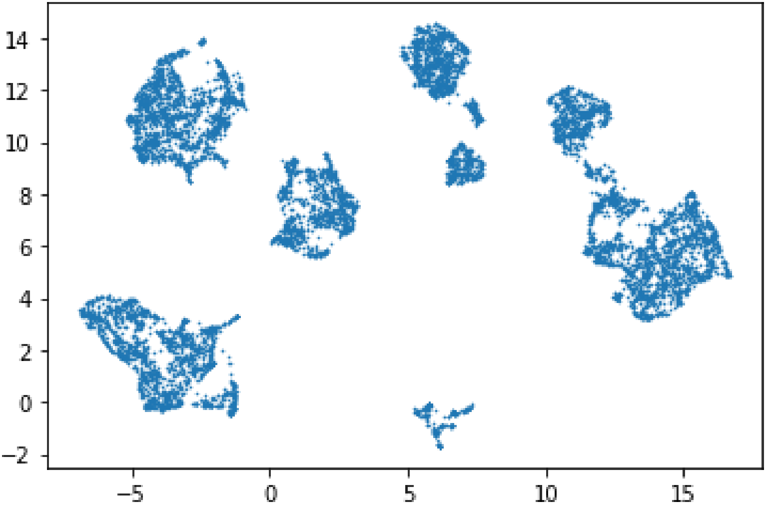
k	1	2	3	4	5
Score	1	0.61	1	0.68	0.61

Application to real data

Dataframe Credit card Dataset (8950 observations, 17 variables)
- **Variables** Balance, balance frequency, purchase, etc.
- **Goal** Clustering people's behavior

Application to real data

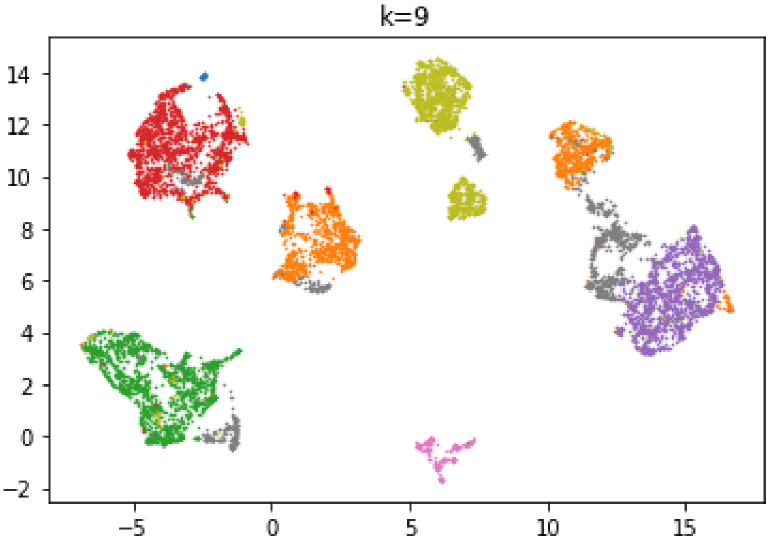
A reprensation of our data set in two dimensions (with UMAP)



Application to real data

k	7	8	9	10	11	12	13
Score	0.48	0.5	0.55	0.46	0.53	0.52	0.32

Application to real data



References

Von Luxburg, Ulrike (2007). "A Tutorial on Spectral Clustering". In: DOI: 10.48550/ARXIV.0711.0189. URL: https://arxiv.org/abs/0711.0189.