k-means Clustering

k-means aims to partition n observations $x_1, \dots, x_n \in \mathbb{R}^d$ into clusters $C_1, \dots, C_k$ so as to minimize

$$\underset{C_1,\dots,C_k}{\operatorname{argmin}} \sum_{i=1}^{k} \frac{1}{|C_i|} \sum_{x,y \in C_i} \|x-y\|^2 \qquad (*)$$

Note: This problem is NP-hard (even when $k=2$)!

- Algorithm (Lloyd's Algorithm): Specify k means and map each observation to the closest one
  - Initialize k means $m_1^{(0)}, \dots, m_k^{(0)}$
  - Assignment: Assign each point to cluster of nearest mean

$$C_i^{(t)} = \left\{ x_p : \|x_p - m_i^{(t)}\|^2 \le \|x_p - m_j^{(t)}\|^2 \quad \forall j,\ 1 \le i \le k \right\}$$

  - Update: $m_i^{(t+1)} = \dfrac{1}{|C_i^{(t)}|} \displaystyle\sum_{x_j \in C_i^{(t)}} x_j$
  - Stop when decrease in $(*)$ becomes small or cluster assignments stop changing

Assign any new point $x \in \mathbb{R}^d$ to its nearest cluster centre m_j.

Note: k-means is not guaranteed to find the optimal solution to $(*)$!

→ It only converges to a local minimum (in the sense that re-assignment and update changes nothing)

→ However, each step of the algorithm is guaranteed to decrease $(*)$

Ostrovsky Effective of Lloyd type methods

Property: The decision regions

$$R_i = \left\{ x \in \mathbb{R}^d : \|x - m_i\|^2 \le \|x - m_j\|^2 \quad \forall i,j \in [k] \right\}$$

resulting from the k-means clustering algorithm are convex.

Pf: $\|x - m_i\|^2 \le \|x - m_j\|^2 \iff -2\langle x, m_i \rangle + \|m_i\|^2 \le -2\langle x, m_j \rangle + \|m_j\|^2$

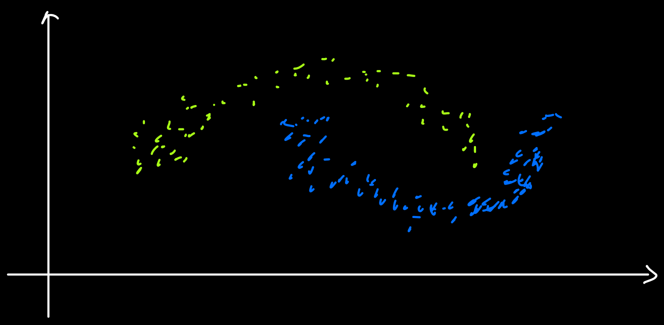
$$\iff \langle m_j - m_i, x \rangle \le \frac{1}{2}\left(\|m_i\|^2 - \|m_j\|^2 \right)$$

This region has the form $\langle a, x \rangle \leq b$ (i.e., a halfspace).

But then $R_i = \bigcap_{i \neq j} \{x \in \mathbb{R}^d : \|x - m_i\|^2 \leq \|x - m_j\|^2\}$, which is an intersection of halfspaces.

In particular, intersections of halfspaces are convex. $\square$

Problem: What if linear decision boundaries aren't enough?



We cannot "globally" separate the clusters with a linear decision boundary. However, the clusters exhibit interesting local structure. There are points within a cluster which are far from the centroid but close to each other.

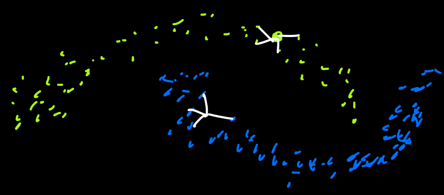
Spectral Clustering

Idea: Encode local neighbourhood structure with a graph $G = (V, E)$, where $V = \{x_1, \ldots, x_n\}$ and the edges in E have weights based on how "similar" two vertices are.

For every pair (v_i, v_j), define a weight/similarity $w_{ij} \geq 0$. Encode weights in a symmetric matrix $W \in \mathbb{R}^{d \times d}$.

Common choices of W:

- ε-neighbourhood (i.e., $W_{ij} = 1$ iff $\|x_i - x_j\| \leq \varepsilon$)
- k-nearest neighbours
- Fully connected with a similarity kernel (e.g. $W_{ij} = \exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)$ for some $\sigma^2 > 0$)



– Key Graph Quantities

- Degree of vertex i: $d_i = \sum_{j=1}^{n} w_{ij}$
- Degree matrix $D = \mathrm{diag}(d_1, \ldots, d_n)$
- $W(A,B) := \sum_{i \in A, j \in B} w_{ij}$
- $\mathrm{vol}(A) = \sum_{i \in A} d_i$

– The Graph Laplacian

<u>Def</u>: The (unnormalized) graph Laplacian of G is $L = D - W \in \mathbb{R}^{n \times n}$.

$$L_{ij} = D_{ij} - W_{ij} = \begin{cases} d_i, & i = j \\ -w_{ij}, & i \neq j \end{cases}$$ (assuming no self-edges)

Prop (Properties of L):

(1) $\forall u \in \mathbb{R}^n$: $u^T L u = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} (u_i - u_j)^2$

Pf: $u^T L u = u^T (D-W) u = \sum_{i=1}^{n} d_i u_i^2 - \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} u_i u_j$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} u_i^2 - \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} u_i u_j$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (u_i^2 - u_i u_j)$$

$$= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (u_i^2 - u_i u_j) + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (u_j^2 - u_i u_j)$$

$$= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (u_i - u_j)^2$$

$\square$

(2) L is symmetric and PSD.

Pf: Duh. $\square$

(3) L has a zero eigenvalue with corresponding eigenvector $\mathbb{1} = (1, 1, \ldots, 1)^T \in \mathbb{R}^n$.

Pf: $[L\mathbb{1}]_i = \sum_{j=1}^{n} L_{ij} = d_i - \sum_{j=1}^{n} w_{ij} = 0$. $\square$

(4) If L has k connected components $A_1, \ldots, A_k$, the eigenvalue 0 has multiplicity k with associated eigenvectors $\mathbb{1}_{A_i}$, whose entries are of the form $[\mathbb{1}_{A_i}]_j = \begin{cases} 1, & v_j \in A_i \\ 0, & v_j \notin A_i. \end{cases}$

Note: (4) implies that if the graph is already perfectly partitioned, the eigenvectors exactly give us the clustering.

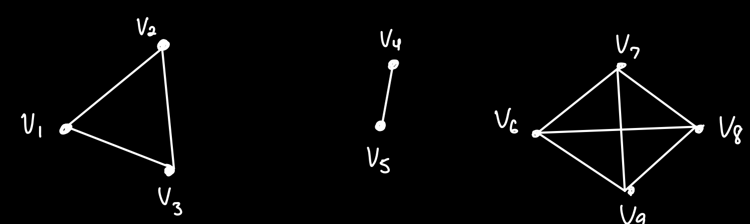
Write the eigenvalues of L as $0 = \lambda_1 \leq \ldots \leq \lambda_n$

and the associated eigenvectors as $u_1, \ldots, u_n \in \mathbb{R}^n$

- Spectral Clustering Algorithm

  - Given W, construct L
  - Obtain the eigenvectors $u_1, \ldots, u_k \in \mathbb{R}^n$ and encode them in $U \in \mathbb{R}^{n \times k}$
  - For $i \in [n]$, let y_i be the i^{th} row of U
  - Use the k-means algorithm to cluster $(y_i)_{i=1}^n$ into $C_1, \ldots, C_k$
  - Map back to clusters in the original space

Ex (Ideal Case): $k = 3$ connected components



Wlog we have $u_1 = \mathbb{1}_{C_1}$, $u_2 = \mathbb{1}_{C_2}$, $u_3 = \mathbb{1}_{C_3}$

Then, $y_1 = y_2 = y_3 = (1,0,0)^T$ $\quad y_4 = y_5 = (0,1,0)^T$ $\quad y_6 = y_7 = y_8 = y_9 = (0,0,1)^T$

- Normalized Laplacians

  - It's natural to consider the weights associated to a vertex v_i relative to the degree d_i

$$L_{sym} := D^{-1/2} L D^{-1/2} \qquad [L_{sym}]_{ij} = \begin{cases} 1, & i = j \\ -\dfrac{w_{ij}}{\sqrt{d_i}\sqrt{d_j}}, & i \neq j \end{cases}$$

$$L_{rw} := D^{-1} L \qquad [L_{rw}]_{ij} = \begin{cases} 1, & i = j \\ -\dfrac{w_{ij}}{d_i}, & i \neq j \end{cases}$$

Explaining Spectral Clustering

I will cover the connection to graph cuts, which is illustrative but flawed (as I will explain).

Recall $W(A,B) = \sum_{i\in A, j\in B} w_{ij}$, $\quad \mathrm{vol}(A) = \sum_{i\in A} d_i$

Define $\bar{A} := V \setminus A$ and

$$\mathrm{cut}(A_1,\dots,A_k) := \frac{1}{2}\sum_{i=1}^{k} W(A_i, \bar{A}_i).$$

Consider the objective of partitioning ("cutting") V into subsets $A_1,\dots,A_k$ with as little weight as possible between distinct subsets.

$$\mathrm{RatioCut}(A_1,\dots,A_k) := \frac{1}{2}\sum_{i=1}^{k} \frac{W(A_i,\bar{A}_i)}{|A_i|} \qquad (1)$$

$$\mathrm{Ncut}(A_1,\dots,A_k) = \frac{1}{2}\sum_{k=1}^{k} \frac{W(A_i,\bar{A}_i)}{\mathrm{vol}(A_i)}.$$

Ncut is a very natural objective for clustering: minimize similarity between clusters while maximizing similarity between clusters.

For simplicity, we will focus on the connection between RatioCut and spectral clustering with L. The connection between Ncut and L_{sym} arises similarly.

- Approximating RatioCut for $k=2$

Objective: $\min_{A\subseteq V} \mathrm{RatioCut}(A,\bar{A}) \qquad (2)$

Define a bijection between the chosen partition given by A and $u \in \mathbb{R}^n$:

Given $A \subseteq V$, write $u = (u_1, \dots, u_n)^T \in \mathbb{R}^n$,
$$u_i = \begin{cases} \sqrt{|\bar{A}|/|A|}, & v_i \in A \\ -\sqrt{|A|/|\bar{A}|}, & v_i \in \bar{A} \end{cases} \qquad (3)$$

We will show that we can relate RatioCut to a collection of quadratic forms involving L.

$$\begin{aligned}
u^T L u &= \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} (u_i - u_j)^2 \\
&= \frac{1}{2} \sum_{i \in A, j \in \bar{A}} w_{ij} \left(\sqrt{\frac{|\bar{A}|}{|A|}} + \sqrt{\frac{|A|}{|\bar{A}|}} \right)^2 + \frac{1}{2} \sum_{i \in \bar{A}, j \in A} w_{ij} \left(-\sqrt{\frac{|\bar{A}|}{|A|}} - \sqrt{\frac{|A|}{|\bar{A}|}} \right)^2 \\
&= \mathrm{cut}(A, \bar{A}) \left(\frac{|\bar{A}|}{|A|} + \frac{|A|}{|\bar{A}|} + 2 \right) \\
&= \mathrm{cut}(A, \bar{A}) \left(\frac{\overbrace{|\bar{A}| + |A|}^{n}}{|A|} + \frac{\overbrace{|\bar{A}| + |A|}^{n}}{|\bar{A}|} \right) \\
&= n \, \mathrm{RatioCut}(A, \bar{A}) \qquad \text{by (1)}
\end{aligned}$$

Moreover, $\displaystyle\sum_{i=1}^{n} u_i = \sum_{i \in A} \sqrt{\frac{|\bar{A}|}{|A|}} - \sum_{i \notin A} \sqrt{\frac{|A|}{|\bar{A}|}} = \sqrt{|\bar{A}||A|} - \sqrt{|\bar{A}||A|} = 0$

$\Rightarrow u_i \perp \mathbb{1}$

And, $\displaystyle\|u\|^2 = \sum_{i \in A} \frac{|\bar{A}|}{|A|} + \sum_{i \notin A} \frac{|A|}{|\bar{A}|} = |\bar{A}| + |A| = n.$

$\Rightarrow$ (2) is equivalent to $\displaystyle\min_{A \subseteq V} u^T L u$, u_i as in (3)

This is (still) an NP-hard discrete optimization problem. Let's take a continuous relaxation and impose $u^T \mathbb{1} = 0$ and $\|u\|^2 = n$ as constraints:

$$\min_{u \in \mathbb{R}^n} u^T L u \quad \text{s.t.} \quad u^T \mathbb{1} = 0, \ \|u\|^2 = n.$$

Since $u_1 = \mathbb{1}$, if G is connected the solution must be u_2, the eigenvector associated with λ_2, with appropriate rescaling so that $\|u_2\|^2 = n$.

$$u_2 = (u_{2,1}, u_{2,2}, \ldots, u_{2,n})^T \in \mathbb{R}^n$$

Scalar representations of each data point

Cluster the entries of u_2 into two clusters.

Note: The graph cut obtained from spectral clustering is not guaranteed to be close to the optimal ratio cut. In fact, there exists an example where spectral clustering yields a value of the objective in (2) which is $\Theta(n)$ times worse than optimal.

This reasoning can be extended to $k>2$ by defining a bijection between $A_1, \ldots, A_k$ and k indicator vectors (instead of one).

Laplacian Eigenmaps

We now discuss a very related unsupervised learning algorithm.

Data: $x_1, \ldots, x_n \in \mathbb{R}^d$

Motivation: Like PCA, want to embed the data into $\mathbb{R}^k$, where $k \ll d$. Unlike PCA, we want to be able to do this nonlinearly.

Use the same idea of a graph $G = (V, E, W)$ and a notion of similarity.

– Algorithm

- Identify connected components of G (e.g., by DFS/BFS), then do the following on each component
- Compute first $k+1$ eigenpairs of $L_{rw} = D^{-1}L$: $(\lambda_0, u_0), \dots, (\lambda_k, u_k)$
- Embed $x_i \mapsto (u_{1,i}, \dots, u_{k,i})$

Similarly to spectral clustering, we can motivate this with an optimization problem.

Assume $k=1$ and consider

$$\min_{y \in \mathbb{R}^n} \underbrace{\frac{1}{2} \sum_{i=1}^{n} \frac{1}{d_i} \sum_{j=1}^{n} w_{ij} (y_i - y_j)^2}_{= y^T D^{-1} L y} \quad \text{s.t. } \|y\|^2 = n \qquad \underbrace{y^T \mathbb{1} = 0}_{\text{Avoids trivial solution}}$$

Similar points should be "close".

As before, the solution is $y = u_1$.

This can be generalized to $k > 1$:

$$\min_{Y \in \mathbb{R}^{n \times k}} \underbrace{\frac{1}{2} \sum_{i=1}^{n} \frac{1}{d_i} \sum_{j=1}^{n} w_{ij} \|y_i - y_j\|^2}_{tr(Y^T D^{-1} L Y)} \quad \text{s.t. } V^T V = I_k, \quad V^T \mathbb{1} = 0.$$

Optimal solution is $Y = \begin{bmatrix} | & & | \\ u_1 & \cdots & u_k \\ | & & | \end{bmatrix}$.

Theoretical Guarantees

Belkin and Niyogi (2005): Towards a Theoretical Foundation for Laplacian-Based Manifold Methods

→ Assume the data points $x_1, \dots, x_n$ are drawn from a manifold $\mathcal{M}$ in $\mathbb{R}^d$

→ Then, in the limit (in probability) as $n \to \infty$, the graph Laplacian converges to the <u>Laplace-Beltrami operator</u> $\Delta_{\mathcal{M}}$ defined by $\Delta_{\mathcal{M}} f = \operatorname{div}(\nabla_{\mathcal{M}} f)$ for twice-differentiable $f: \mathcal{M} \to \mathbb{R}$.

Von Luxburg, Belkin, Bousquet (2008): Consistency of Spectral Clustering

→ The eigenvectors of the normalized Laplacian converge to the eigenfunctions of a limit operator