

COMP5211: Machine Learning

Lecture 17

Minhao Cheng

Clustering

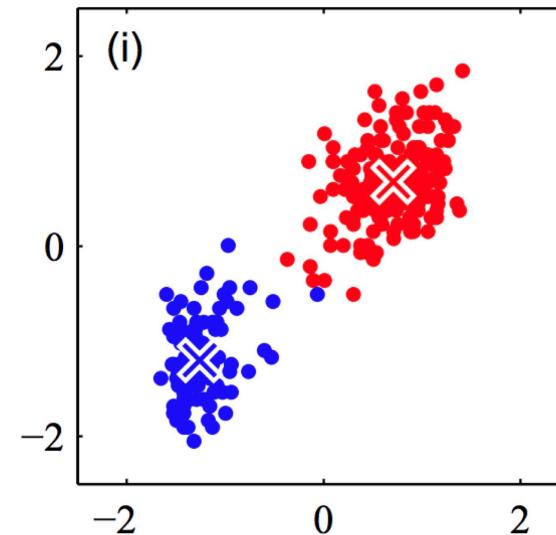
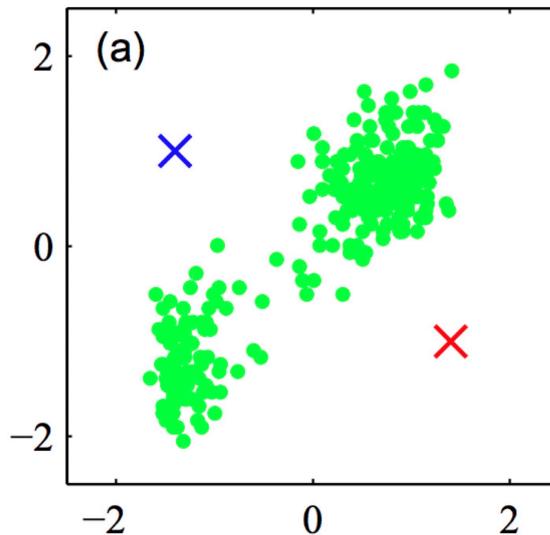
Supervised versus unsupervised learning

- Supervised learning:
 - Learning from **labeled** observations
 - Classification, regression
- Unsupervised learning:
 - Learning from **unlabeled** observations
 - Discover hidden patterns
 - Clustering (today)

Clustering

Definition

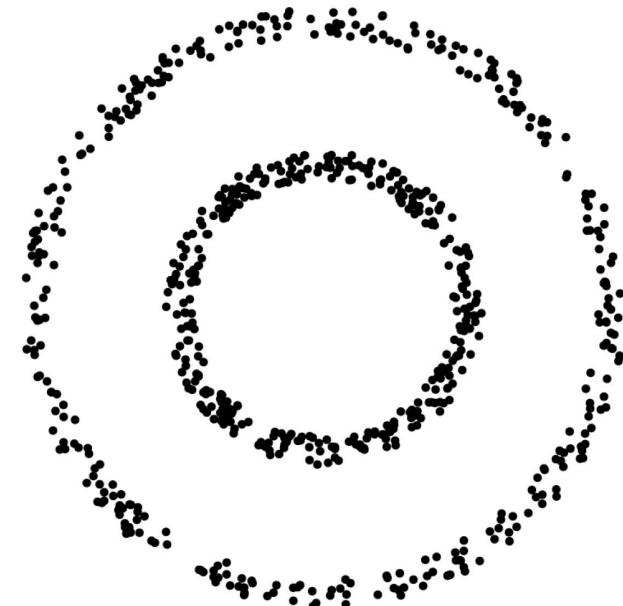
- Given $\{x_1, x_2, \dots, x_n\}$ and K (number of clusters)
- Output $A(x_i) \in \{1, 2, \dots, K\}$ (cluster membership)



Clustering

Two circles

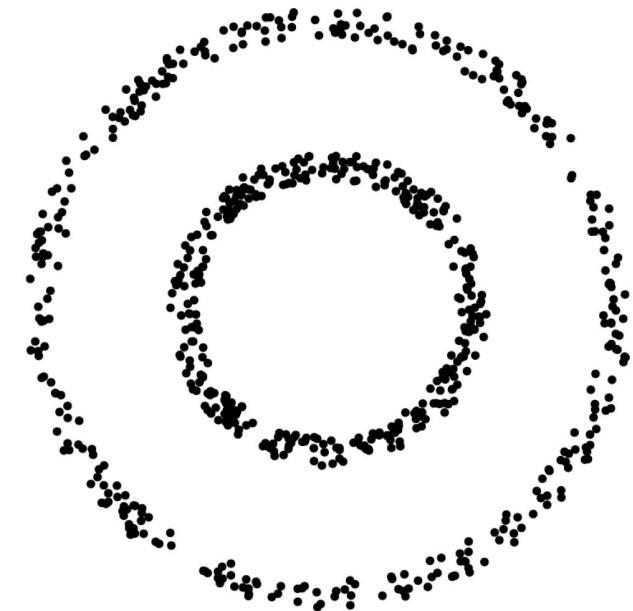
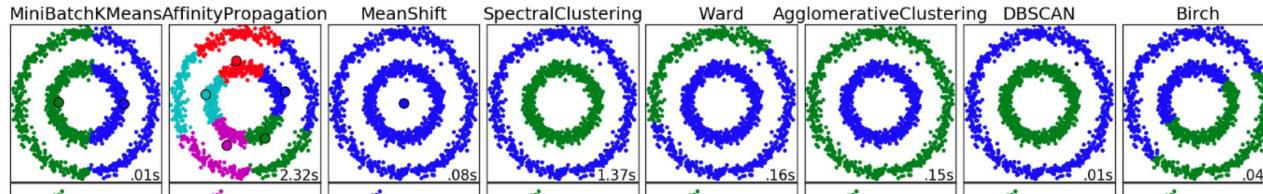
- Can we split the data into two clusters?



Clustering

Two circles

- Can we split the data into two clusters?

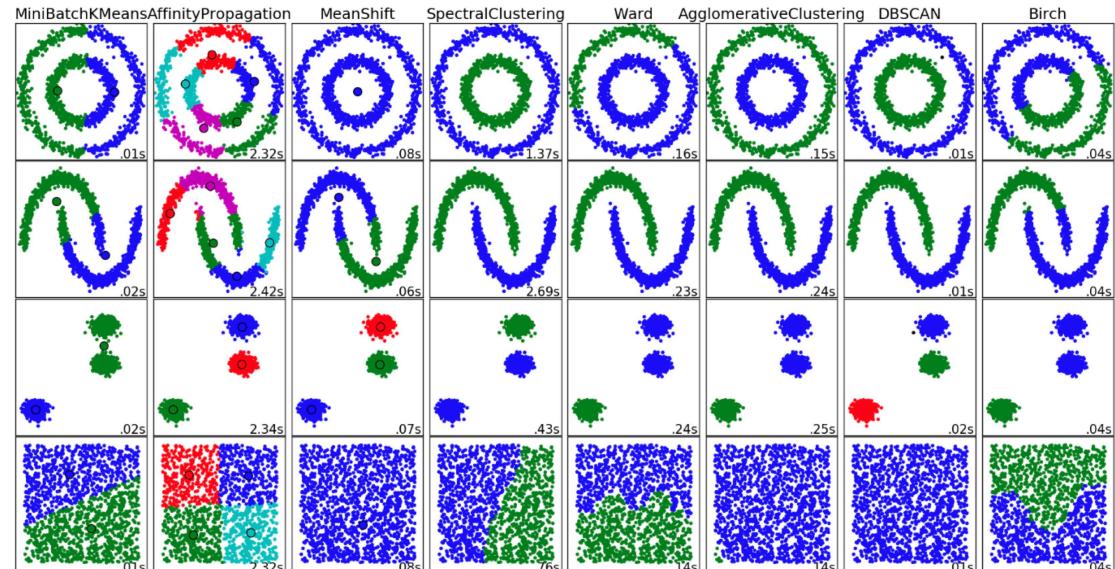


Clustering

Clustering is subjective

How to say one is better than another

- Non-trivial to say one partition is better than others
 - Each algorithm has two parts:
 - Define the **objective function**
 - Design an algorithm to **minimize this objective function**



Clustering

K-means

partition into K cluster

- Partition datasets into C_1, C_2, \dots, C_k to minimize the following objective:

$$J = \sum_{k=1}^K \sum_{x \in C_k} \|x - m_k\|_2^2$$

- Where m_k is the mean of C_k

Clustering

K-means

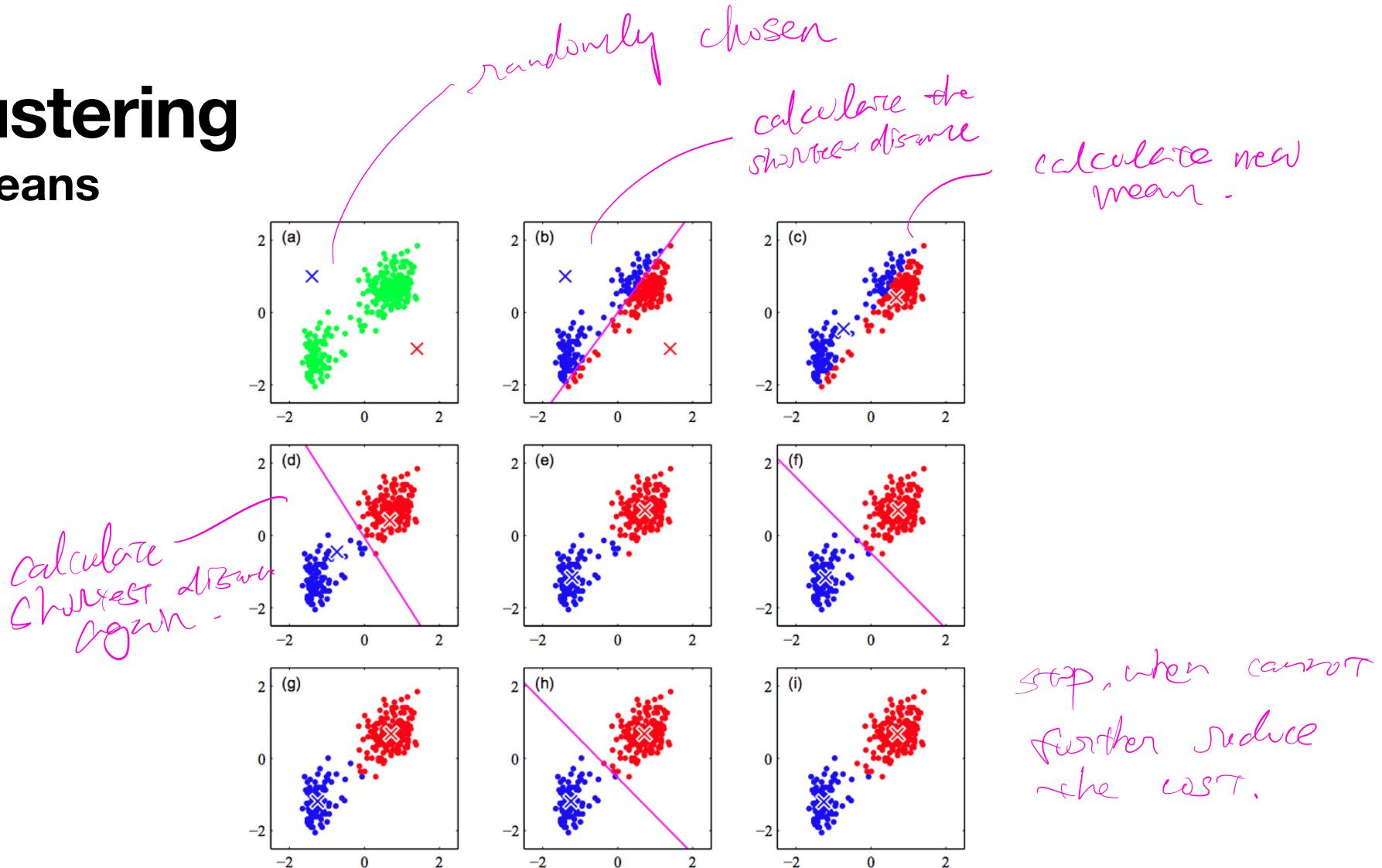
- Partition datasets into C_1, C_2, \dots, C_k to minimize the following objective:

$$\bullet J = \sum_{k=1}^K \sum_{x \in C_k} \|x - m_k\|_2^2$$

- Where m_k is the mean of C_k
- Multiple ways to minimize this objective
 - Hierarchical Agglomerative Clustering
 - Kmeans Algorithm (Today)
 - ...

Clustering

K-means



Clustering

K-means Algorithm

- Re-write objective:

- $$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - m_k\|_2^2$$

- Where $r_{nk} \in \{0,1\}$ is an indicator variable
 - $r_{nk} = 1$ if and only if $x_n \in C_k$

- Alternative optimization between $\{r_{nk}\}$ and $\{m_k\}$

- Fix $\{m_k\}$ and update $\{r_{nk}\}$
- Fix $\{r_{nk}\}$ and update $\{m_k\}$

Fix one and optimize another
do it alternatively.

Clustering

K-means Algorithm

- Step 0: initialize $\{m_k\}$ to some values

Clustering

K-means Algorithm

- Step 0: initialize $\{m_k\}$ to some values
- Step 1: Fix $\{m_k\}$ and minimize over $\{r_{nk}\}$:
 - $r_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_j \|x_n - m_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$

Clustering

K-means Algorithm

- Step 0: initialize $\{m_k\}$ to some values
- Step 1: Fix $\{m_k\}$ and minimize over $\{r_{nk}\}$:

$$\bullet \quad r_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_j \|x_n - m_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

- Step 2: Fix $\{r_{nk}\}$ and minimize over $\{m_k\}$:

$$\bullet \quad m_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

Clustering

K-means Algorithm

- Step 0: initialize $\{m_k\}$ to some values
- Step 1: Fix $\{m_k\}$ and minimize over $\{r_{nk}\}$:
 - $r_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_j \|x_n - m_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$
- Step 2: Fix $\{r_{nk}\}$ and minimize over $\{m_k\}$:
 - $m_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$
- Step 3: Return to step 1 unless stopping criterion is met

Clustering

K-means Algorithm

- Equivalent to the following procedure:
 - Step 0: initialize centers $\{m_k\}$ to some values
 - Step 1: Assign each x_n to the nearest center:
 - $A(x_n) = \arg \min_j \|x_n - m_j\|_2^2$
 - Update cluster:
 - $C_k = \{x_n : A(x_n) = k\} \quad \forall k = 1, \dots, K$
 - Step 2: Calculate mean of each cluster C_k :
 - $m_k = \frac{1}{|C_k|} \sum_{x_n \in C_k} x_n$
 - Step 3: Return to step 1 unless stopping criterion is met

Clustering

More on K-means Algorithm

- Always decrease the objective function for each update
- Objective function will remain unchanged when step 1 doesn't change cluster assignment \Rightarrow Converged

Clustering

More on K-means Algorithm

- Always decrease the objective function for each update
- Objective function will remain unchanged when step 1 doesn't change cluster assignment \Rightarrow Converged
- May not convene to global minimum
 - Sensitive to initial values

Clustering

More on K-means Algorithm

- Always decrease the objective function for each update
- Objective function will remain unchanged when step 1 doesn't change cluster assignment \Rightarrow Converged
- May not convene to global minimum
 - Sensitive to initial values
- Kmeans++: A better way to initialize the clusters

Clustering

Graph Clustering

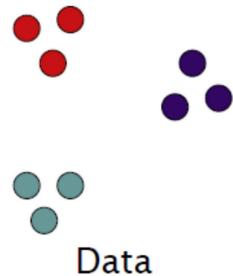
- Given a graph $G = (V, E, W)$
 - V : nodes $\{v_1, \dots, v_n\}$
 - E : edges $\{e_1, \dots, e_m\}$
 - W : weight matrix
 - $W_{ij} = \begin{cases} w_{ij}, & \text{if } (i, j) \in E \\ 0, & \text{otherwise} \end{cases}$
- Goal: Partition V into k clusters of nodes
 - $V = V_1 \cup V_2 \cup \dots \cup V_k, \quad V_i \cap V_j = \varnothing, \forall i, j$

Clustering Similarity Graph

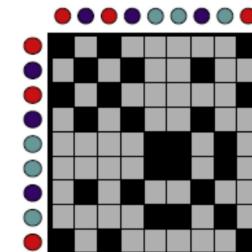
- Example: similarity graph
- Given samples x_1, \dots, x_n
- Weight (similarities) indicates “closeness of samples”

Similarity Graph: $G(V,E,W)$

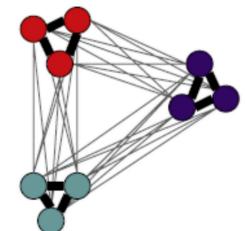
V – Vertices (Data points)
E – Edge if similarity > 0
W - Edge weights (similarities)



Data



Similarities



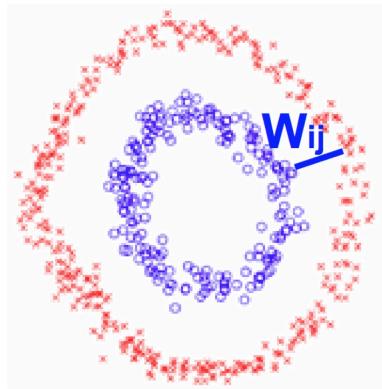
Similarity graph

Partition the graph so that edges within a group have large weights and edges across groups have small weights.

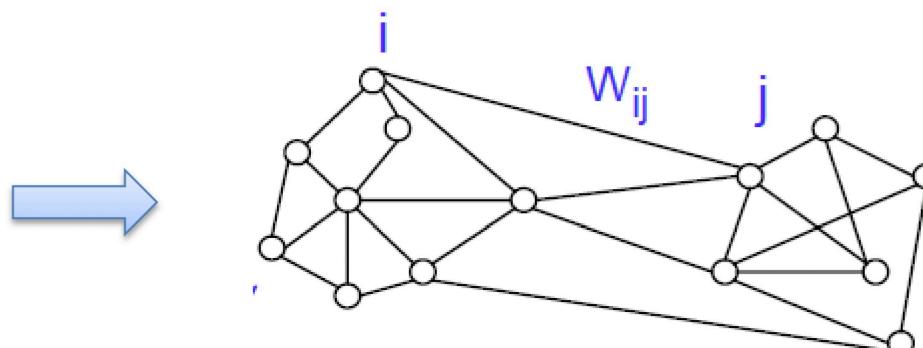
Clustering

Similarity graph

- E.g., Gaussian kernel $W_{ij} = e^{-\|x_i - x_j\|^2/\sigma^2}$



Data clustering



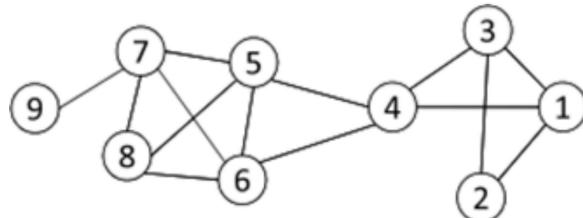
$$G = \{V, E\}$$

Clustering

Social graph

- Nodes: users in social network
- Edges: $W_{ij} = 1$ if user i and j are friends, otherwise $W_{ij} = 0$

| Graph Representation



find an edge

| Matrix Representation

Node	1	2	3	4	5	6	7	8	9
1	-	1	1	1	0	0	0	0	0
2	1	-	1	0	0	0	0	0	0
3	1	1	-	1	0	0	0	0	0
4	1	0	1	-	1	1	0	0	0
5	0	0	0	1	-	1	1	1	0
6	0	0	0	1	1	-	1	1	0
7	0	0	0	0	1	1	-	1	1
8	0	0	0	0	1	1	1	-	0
9	0	0	0	0	0	0	1	0	-

TO CUT !

Clustering

Partitioning into two clusters

- Partition graph into two sets V_1, V_2 to minimize **the cut value**:

$$\bullet \text{cut}(V_1, V_2) = \sum_{v_i \in V_1, v_j \in V_2} W_{ij}$$

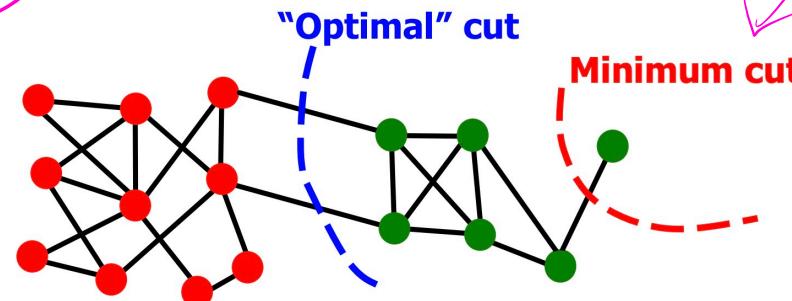
Clustering

Partitioning into two clusters

- Partition graph into two sets V_1, V_2 to minimize **the cut value**:

$$\text{cut}(V_1, V_2) = \sum_{v_i \in V_1, v_j \in V_2} W_{ij}$$

- Also, the size of V_1, V_2 needs to be similar (**balance**)



Clustering

Partitioning into two clusters

- Partition graph into two sets V_1, V_2 to minimize **the cut value**:

- $\text{cut}(V_1, V_2) = \sum_{v_i \in V_1, v_j \in V_2} W_{ij}$

- Also, the size of V_1, V_2 needs to be similar (**balance**)
- One classical way of enforcing balance:

$$\min_{V_1, V_2} \text{cut}(V_1, V_2)$$

- s.t. $|V_1| = |V_2|, V_1 \cup V_2 = \{1, \dots, n\}, V_1 \cap V_2 = \varnothing$
- \Rightarrow This is NP-hard (cannot be solved in polynomial time)

Clustering

Kernaghan-Lin Algorithm

- Start with some partitioning V_1, V_2
- Calculate change in cut if 2 vertices are swapped
- Swap the vertices (1 in V_1 & 1 in V_2) that decease the cut the most
- Iterate until convergence

Clustering

Kernaghan-Lin Algorithm

- Start with some partitioning V_1, V_2
- Calculate change in cut if 2 vertices are swapped
- Swap the vertices (1 in V_1 & 1 in V_2) that decease the cut the most
- Iterate until convergence
- Used when we need **exact balanced clusters** (e.g. circuit design)

Clustering

Objective function that consider **balance**

- Ratio-Cut:

$$\bullet \min_{V_1, V_2} \left\{ \frac{\text{cut}(V_1, V_2)}{|V_1|} + \frac{\text{cut}(V_1, V_2)}{|V_2|} \right\} := RC(V_1, V_2)$$

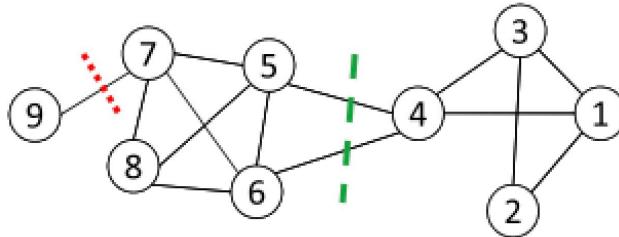
- Normalized-Cut:

$$\bullet \min_{V_1, V_2} \left\{ \frac{\text{cut}(V_1, V_2)}{\deg(V_1)} + \frac{\text{cut}(V_1, V_2)}{\deg(V_2)} \right\} := RC(V_1, V_2)$$

$$\bullet \text{Where } \deg(V_c) := \sum_{v_i \in V_c, (i,j) \in E} W_{i,j} = \text{links}(V_c, V)$$

Clustering

Cut example



$$\text{Cut(Red)} = 1$$

$$\text{Cut(Green)} = 2$$

$$\text{Ratio-Cut(Red)} = \frac{1}{1} + \frac{1}{8} = \frac{9}{8}$$

$$\text{Ratio-Cut(Green)} = \frac{2}{5} + \frac{2}{4} = \frac{18}{20}$$

$$\text{Normalized-Cut(Red)} = \frac{1}{1} + \frac{1}{27} = \frac{28}{27}$$

$$\text{Normalized-Cut(Green)} = \frac{2}{12} + \frac{2}{16} = \frac{14}{48}$$

Minimizing **Normalized-cut** is even better for Green due to density constraint (volume)

Clustering

Generalize to k clusters

- Ratio-Cut:

$$\min_{V_1, \dots, V_k} \sum_{c=1}^k \frac{\text{cut}(V_c, V - V_c)}{|V_c|}$$

- Normalized-Cut:

$$\min_{V_1, \dots, V_k} \sum_{c=1}^k \frac{\text{cut}(V_c, V - V_c)}{\deg(V_c)}$$

Clustering

Reformulation

- Recall $\deg(V_c) = \text{links}(V_c, V)$
- Define a diagonal matrix

$$D = \begin{bmatrix} \deg(v_1) & 0 & 0 & \dots \\ 0 & \deg(v_2) & 0 & \dots \\ 0 & 0 & \deg(v_3) & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- $y_c = \{0,1\}^n$: indicator vector for the c-th cluster

Clustering

Reformulation

- Recall $\deg(V_c) = \text{links}(V_c, V)$
- Define a diagonal matrix

$$D = \begin{bmatrix} \deg(v_1) & 0 & 0 & \dots \\ 0 & \deg(v_2) & 0 & \dots \\ 0 & 0 & \deg(v_3) & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- $y_c = \{0,1\}^n$: indicator vector for the c-th cluster
- We have

$$y_c^T y_c = |V_c|$$

$$y_c^T D y_c = \deg(V_c)$$

$$y_c^T W y_c = \text{links}(V_c, V_c)$$

Clustering

Ratio Cut

$$\begin{aligned} RC(V_1, \dots, V_k) &= \sum_{c=1}^k \frac{\text{cut}(V_c, V - V_c)}{|V_C|} \\ &= \sum_{c=1}^k \frac{\deg(V_c) - \text{links}(V_c, V_c)}{|V_C|} \\ &= \sum_{c=1}^k \frac{y_c^T D y_c - y_c^T W y_c}{y_c^T y_c} \\ &= \sum_{c=1}^k \frac{y_c^T (\mathbf{D} - \mathbf{W}) y_c}{y_c^T y_c} \\ &= \sum_{c=1}^k \frac{y_c^T \mathbf{L} y_c}{y_c^T y_c} \quad (\mathbf{L} = \mathbf{D} - \mathbf{W} \text{ is "Graph Laplacian"}) \end{aligned}$$

Clustering

More on graph laplacian

- L is symmetric positive semi-definite

Clustering

Solving Ratio-Cut

- We have shown Ratio-Cut is equivalent to

- $\text{RCut} = \sum_{c=1}^k \frac{y_c^T L y_c}{y_c^T y_c} = \sum_{c=1}^k \left(\frac{y_c}{\|y_c\|} \right)^T L \left(\frac{y_c}{\|y_c\|} \right)$

- Define $\bar{y}_c = y_c / \|y_c\|$ (normalized indicator),

- $Y = [\bar{y}_1, \bar{y}_2, \dots, \bar{y}_k] \Rightarrow Y^T Y = I$

Clustering

Solving Ratio-Cut

- We have shown Ratio-Cut is equivalent to

- $\text{RCut} = \sum_{c=1}^k \frac{y_c^T L y_c}{y_c^T y_c} = \sum_{c=1}^k \left(\frac{y_c}{\|y_c\|} \right)^T L \left(\frac{y_c}{\|y_c\|} \right)$

- Define $\bar{y}_c = y_c / \|y_c\|$ (normalized indicator),

- $Y = [\bar{y}_1, \bar{y}_2, \dots, \bar{y}_k] \Rightarrow Y^T Y = I$

- Relaxed to real valued problem

- $\min_{Y^T Y = I} \text{Trace}(Y^T L Y)$

Clustering

Solving Ratio-Cut

- We have shown Ratio-Cut is equivalent to

- $\text{RCut} = \sum_{c=1}^k \frac{y_c^T L y_c}{y_c^T y_c} = \sum_{c=1}^k \left(\frac{y_c}{\|y_c\|} \right)^T L \left(\frac{y_c}{\|y_c\|} \right)$

- Define $\bar{y}_c = y_c / \|y_c\|$ (normalized indicator),

- $Y = [\bar{y}_1, \bar{y}_2, \dots, \bar{y}_k] \Rightarrow Y^T Y = I$

- Relaxed to real valued problem

- $\min_{Y^T Y = I} \text{Trace}(Y^T L Y)$

- Solution: Eigenvectors corresponding to the **smallest k eigenvalues of L**

Clustering

Solving Ratio-Cut

- Let $Y^* \in \mathbb{R}^{n \times k}$ be these eigenvectors. Are we done?

Clustering

Solving Ratio-Cut

- Let $Y^* \in \mathbb{R}^{n \times k}$ be these eigenvectors. Are we done?
- No, Y^* does not have 0/1 values (not indicators)
 - (Since we are solving a **relaxed** problem)
- Solution: Run k-means on the rows of Y^*

Clustering

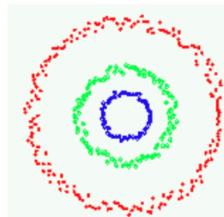
Solving Ratio-Cut

- Let $Y^* \in \mathbb{R}^{n \times k}$ be these eigenvectors. Are we done?
- No, Y^* does not have 0/1 values (not indicators)
 - (Since we are solving a **relaxed** problem)
- Solution: Run k-means on the rows of Y^*
- Summary of Spectral clustering algorithms:
 - Compute $Y^* \in \mathbb{R}^{n \times k}$: eigenvectors corresponds to k smallest eigenvalues of (normalized) Laplacian matrix
 - Run k-means to cluster rows of Y^*

Clustering

Eigenvectors of Laplacian

- If graph is disconnected (k connected components), Laplacian is block diagonal and first k Eigen-vectors are:



OR



$$L = \begin{pmatrix} L_1 & & & \\ & \ddots & & 0 \\ & & L_2 & \\ & \ddots & & \\ 0 & & & L_3 \end{pmatrix}$$

$$\begin{array}{c|c|c} \begin{matrix} 1 \\ 0 \\ \vdots \\ 0 \end{matrix} & \begin{matrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{matrix} & \begin{matrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{matrix} \end{array}$$

First three eigenvectors

Clustering

Eigenvectors of Laplacian

- What if the graph is connected?

Clustering

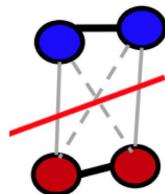
Eigenvectors of Laplacian

- What if the graph is connected?
- There will be only one smallest eigenvalue/eigenvector:
 - $L\mathbf{1} = (D - A)\mathbf{1} = 0$
 - ($\mathbf{1} = [1, 1, \dots, 1]^T$ is the eigenvector with eigenvalue 0)

Clustering

Eigenvectors of Laplacian

- What if the graph is connected?
- There will be only one smallest eigenvalue/eigenvector:
 - $L\mathbf{1} = (D - A)\mathbf{1} = 0$ ($\mathbf{1} = [1, 1, \dots, 1]^T$ is the eigenvector with eigenvalue 0)
- However, the 2nd to k-th smallest eigenvectors are still useful for clustering



1	1	.2	0
1	1	0	.1
.2	0	1	1
0	.1	1	1



1st evec is constant
since graph is connected

.50
.50
.50
.50
.50

.47
.52
.50
-.47
-.52

Sign of 2nd evec
indicates blocks

Clustering

Normalized Cut

- Rewrite Normalized Cut:

$$NCut = \sum_{c=1}^k \frac{\text{cut}(V_c, V - V_c)}{\deg(V_c)}$$

- $= \sum_{c=1}^k \frac{y_c^T (\mathbf{D} - \mathbf{A}) y_c}{y_c^T D y_c}$

- Let $\tilde{y}_c = \frac{D^{1/2} y_c}{\|D^{1/2} y_c\|}$, then

- $NCut = \sum_{c=1}^k \frac{\tilde{y}_c^T D^{-1/2} (\mathbf{D} - \mathbf{A}) D^{-1/2} \tilde{y}_c}{\tilde{y}_c^T \tilde{y}_c}$

- Normalized Laplacian:

- $\tilde{L} = D^{-1/2} (\mathbf{D} - \mathbf{A}) D^{-1/2} = I - D^{-1/2} A D^{-1/2}$

- Normalized Cut \rightarrow eigenvectors correspond to the smallest eigenvalues

Clustering

Kmeans vs Spectral Clustering

- Kmeans: decision boundary is linear
 - Spectral clustering: boundary can be non-convex curves
- σ in $W_{ij} = e^{\frac{-\|x_i - x_j\|^2}{\sigma^2}}$ controls the clustering results (focus on local or global structure)

Clustering

Kmeans vs Spectral Clustering

