CS260: Machine Learning Algorithms

Lecture 5: Clustering

Cho-Jui Hsieh
UCLA

Jan 23, 2019

Supervised versus Unsupervised Learning

Supervised Learning:

- Learning from labeled observations
- Classification, regression, ...

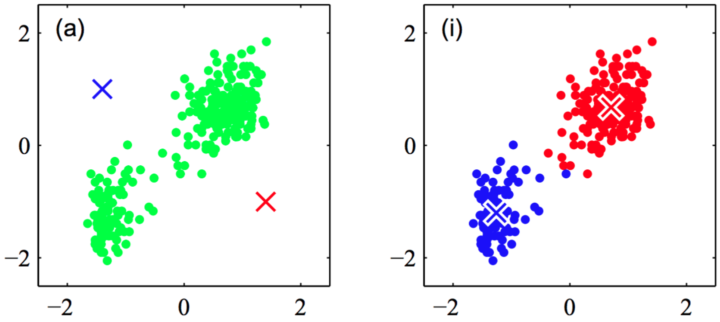
Unsupervised Learning:

- Learning from unlabeled observations
- Discover hidden patterns
- Clustering (today)

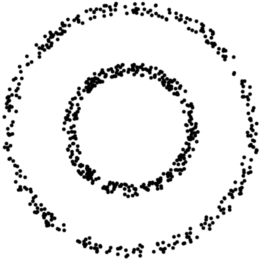
Kmeans Clustering

Clustering

- Given $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ and K (number of clusters)
- Output $A(\boldsymbol{x}_i) \in \{1, 2, \ldots, K\}$ (cluster membership)

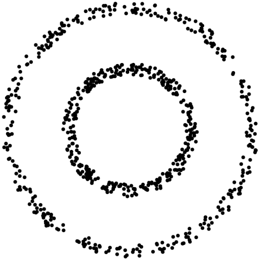


Two circles

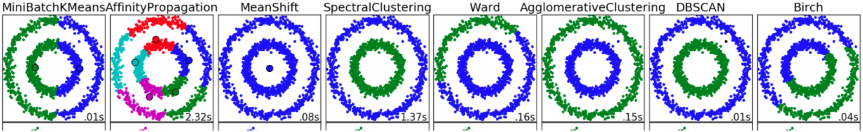


Can we split the data into two clusters?

Two circles

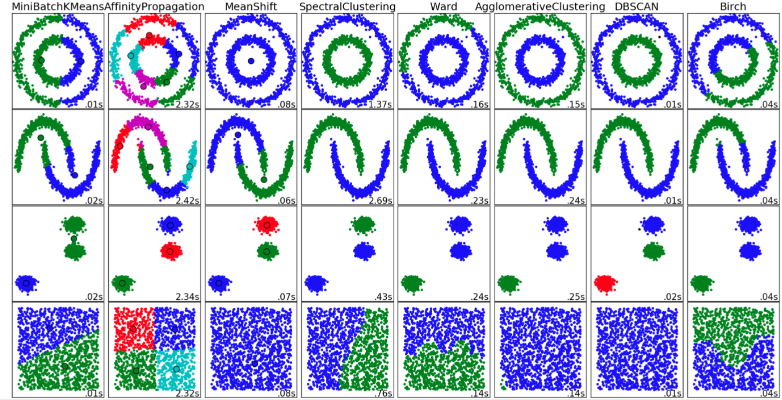


Can we split the data into two clusters?



Clustering is Subjective

- 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



K-means Objective Function

- Partition dataset into $C_1, C_2, \ldots, C_K$ to minimize the following objective:

$$J = \sum_{k=1}^{K} \sum_{\boldsymbol{x} \in C_k} \|\boldsymbol{x} - \boldsymbol{m}_k\|_2^2,$$

where $\boldsymbol{m}_k$ is the mean of C_k.

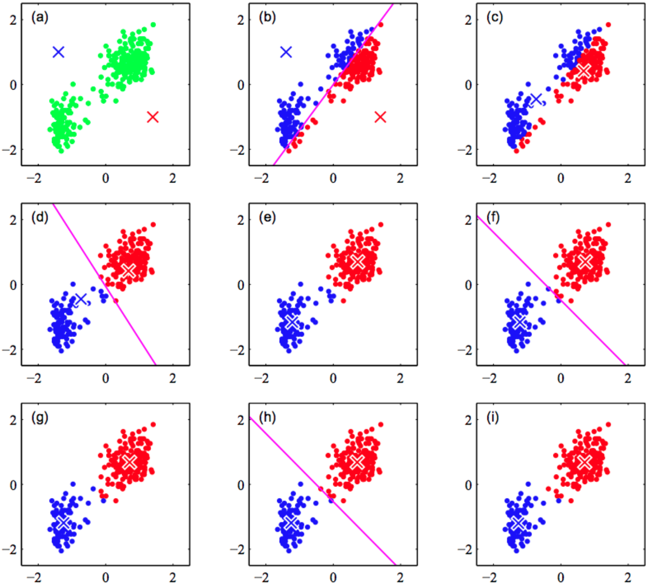
K-means Objective Function

- Partition dataset into $C_1, C_2, \ldots, C_K$ to minimize the following objective:

$$J = \sum_{k=1}^{K} \sum_{\boldsymbol{x} \in C_k} \|\boldsymbol{x} - \boldsymbol{m}_k\|_2^2,$$

  where $\boldsymbol{m}_k$ is the mean of C_k.
- Multiple ways to minimize this objective
  - Hierarchical Agglomerative Clustering
  - Kmeans Algorithm (Today)
  - ...

K-means Algorithm



K-means Algorithm

- Re-write objective:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{m}_k\|_2^2,$$

where $r_{nk} \in \{0, 1\}$ is an indicator variable

$$r_{nk} = 1 \;\text{ if and only if }\; \boldsymbol{x}_n \in C_k$$

- Alternative optimization between $\{r_{nk}\}$ and $\{\boldsymbol{m}_k\}$
  - Fix $\{\boldsymbol{m}_k\}$ and update $\{r_{nk}\}$
  - Fix $\{r_{nk}\}$ and update $\{\boldsymbol{m}_k\}$

K-means Algorithm

- Step 0: Initialize $\{\boldsymbol{m}_k\}$ to some values

K-means Algorithm

- Step 0: Initialize $\{\boldsymbol{m}_k\}$ to some values
- Step 1: Fix $\{\boldsymbol{m}_k\}$ and minimize over $\{r_{nk}\}$:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\boldsymbol{x}_n - \boldsymbol{m}_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

K-means Algorithm

- Step 0: Initialize $\{\boldsymbol{m}_k\}$ to some values
- Step 1: Fix $\{\boldsymbol{m}_k\}$ and minimize over $\{r_{nk}\}$:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\boldsymbol{x}_n - \boldsymbol{m}_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

- Step 2: Fix $\{r_{nk}\}$ and minimize over $\{\boldsymbol{m}_k\}$:

$$\boldsymbol{m}_k = \frac{\sum_n r_{nk} \boldsymbol{x}_n}{\sum_n r_{nk}}$$

K-means Algorithm

- Step 0: Initialize $\{\boldsymbol{m}_k\}$ to some values
- Step 1: Fix $\{\boldsymbol{m}_k\}$ and minimize over $\{r_{nk}\}$:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\boldsymbol{x}_n - \boldsymbol{m}_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

- Step 2: Fix $\{r_{nk}\}$ and minimize over $\{\boldsymbol{m}_k\}$:

$$\boldsymbol{m}_k = \frac{\sum_n r_{nk}\boldsymbol{x}_n}{\sum_n r_{nk}}$$

- Step 3: Return to step 1 unless stopping criterion is met

K-means Algorithm

Equivalent to the following procedure:

- Step 0: Initialize centers $\{\boldsymbol{m}_k\}$ to some values
- Step 1: Assign each $\boldsymbol{x}_n$ to the nearest center:

$$A(\boldsymbol{x}_n) = \arg\min_j \|\boldsymbol{x}_n - \boldsymbol{m}_j\|_2^2$$

  Update clusters:

$$C_k = \{\boldsymbol{x}_n : A(\boldsymbol{x}_n) = k\} \quad \forall k = 1, \ldots, K$$

- Step 2: Calculate mean of each cluster C_k:

$$\boldsymbol{m}_k = \frac{1}{|C_k|} \sum_{\boldsymbol{x}_n \in C_k} \boldsymbol{x}_n$$

- Step 3: Return to step 1 unless stopping criterion is met

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

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 converge to global minimum

  Sensitive to initial values

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 converge to global minimum

  Sensitive to initial values
- Kmeans++: A better way to initialize the clusters

Graph Clustering

Graph Clustering

- Given a graph $G = (V, E, W)$
  - V: nodes $\{v_1, \cdots, v_n\}$
  - E: edges $\{e_1, \cdots, e_m\}$
  - 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

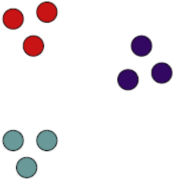
$$V = V_1 \cup V_2 \cup \cdots \cup V_k, \quad V_i \cap V_j = \varphi, \ \forall i, j$$

Similarly Graph

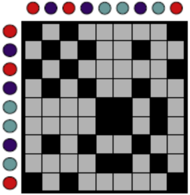
- Example: similarity graph
- Given samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$
- Weight (similarities) indicates "closeness of samples"

Similarity Graph: G(V,E,W)

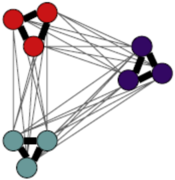
V – Vertices (Data points)
E – Edge if similarity > 0
W - Edge weights (similarities)



Data



Similarities

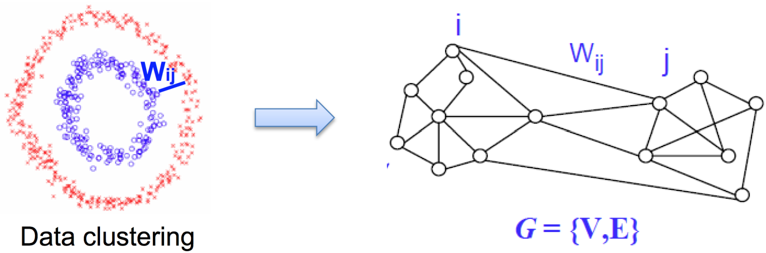


Similarity graph

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

Similarity Graph

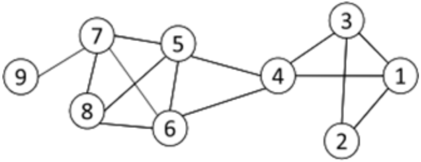
E.g., Gaussian kernel $W_{ij} = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2}$



Social graph

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

Graph Representation



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	-

Partitioning into Two Clusters

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

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

Partitioning into Two Clusters

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

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

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

Partitioning into Two Clusters

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

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

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

$$\min_{V_1, V_2} \quad \text{cut}(V_1, V_2)$$
$$\text{s.t.} \quad |V_1| = |V_2|, \quad V_1 \cup V_2 = \{1, \cdots, n\}, V_1 \cap V_2 = \varphi$$

$\Rightarrow$ this is NP-hard (cannot be solved in polynomial time)

Kernighan-Lin Algorithm

- Starts 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
- Iterative until convergence

Kernighan-Lin Algorithm

- Starts 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
- Iterative until convergence
- Used when we need exact balanced clusters
  (e.g., circuit design)

Objective function that considers balance

- Ratio-Cut:

$$\min_{V_1, V_2} \left\{ \frac{\mathrm{Cut}(V_1, V_2)}{|V_1|} + \frac{\mathrm{Cut}(V_1, V_2)}{|V_2|} \right\} := \mathrm{RC}(V_1, V_2)$$

- Normalized-Cut:

$$\min_{V_1, V_2} \left\{ \frac{\mathrm{Cut}(V_1, V_2)}{\deg(V_1)} + \frac{\mathrm{Cut}(V_1, V_2)}{\deg(V_2)} \right\} := \mathrm{NC}(V_1, V_2),$$

where

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

Generalize to k clusters

- Ratio-Cut:

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

- Normalized-Cut:

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

Reformulation

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

$$D = \begin{bmatrix} \deg(V_1) & 0 & 0 & \cdots \\ 0 & \deg(V_2) & 0 & \cdots \\ 0 & 0 & \deg(V_3) & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

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

Reformulation

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

$$D = \begin{bmatrix} \deg(V_1) & 0 & 0 & \cdots \\ 0 & \deg(V_2) & 0 & \cdots \\ 0 & 0 & \deg(V_3) & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

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

$$\mathbf{y}_c^T \mathbf{y}_c = |V_c|$$
$$\mathbf{y}_c^T D \mathbf{y}_c = \deg(V_c)$$
$$\mathbf{y}_c^T W \mathbf{y}_c = \text{links}(V_c, V_c)$$

Ratio Cut

- Rewrite the ratio-cut objective:

$$
\begin{aligned}
RC(V_1, \cdots, V_k) &= \sum_{c=1}^{k} \frac{\text{Cut}(V_c, V - V_c)}{|V_c|} \\
&= \sum_{c=1}^{k} \frac{\deg(V_c) - \text{links}(V_c, V_c)}{|V_c|} \\
&= \sum_{c=1}^{k} \frac{\mathbf{y}_c^T D \mathbf{y}_c - \mathbf{y}_c^T W \mathbf{y}_c}{\mathbf{y}_c^T \mathbf{y}_c} \\
&= \sum_{c=1}^{k} \frac{\mathbf{y}_c^T (D - W) \mathbf{y}_c}{\mathbf{y}_c^T \mathbf{y}_c} \\
&= \sum_{c=1}^{k} \frac{\mathbf{y}_c^T L \mathbf{y}_c}{\mathbf{y}_c^T \mathbf{y}_c} \quad (L = D - W \text{ is ``Graph Laplacian''})
\end{aligned}
$$

More on Graph Laplacian

- L is symmetric positive semi-definite

More on Graph Laplacian

- L is symmetric positive semi-definite
- For any $\boldsymbol{x}$,

$$\boldsymbol{x}^T L \boldsymbol{x} = \frac{1}{2} \sum_{(i,j)} W_{ij} (x_i - x_j)^2$$

Solving Ratio-Cut

- We have shown Ratio-Cut is equivalent to

$$\text{RCut} = \sum_{c=1}^{k} \frac{\mathbf{y}_c^T L \mathbf{y}_c}{\mathbf{y}_c^T \mathbf{y}_c} = \sum_{c=1}^{k} (\frac{\mathbf{y}_c}{\|\mathbf{y}_c\|})^T L \frac{\mathbf{y}_c}{\|\mathbf{y}_c\|}$$

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

$$Y = [\bar{\mathbf{y}}_1, \bar{\mathbf{y}}_2, \cdots, \bar{\mathbf{y}}_k] \;\Rightarrow\; Y^T Y = I$$

Solving Ratio-Cut

- We have shown Ratio-Cut is equivalent to

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

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

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

- Relaxed to real valued problem:

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

Solving Ratio-Cut

- We have shown Ratio-Cut is equivalent to

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

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

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

- 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

Solving Ratio-Cut

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

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)

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^*

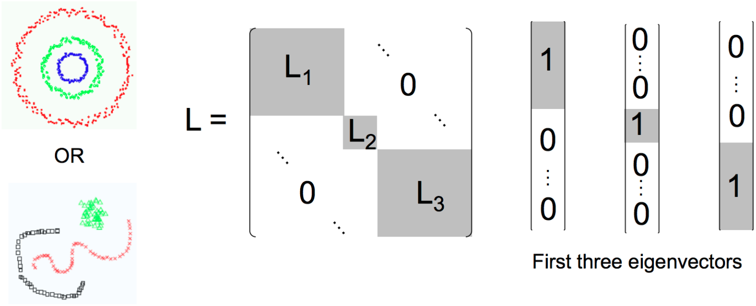
Solving Ratio-Cut

- 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^*

Eigenvectors of Laplacian

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



Eigenvectors of Laplacian

- What if the graph is connected?

Eigenvectors of Laplacian

- What if the graph is connected?
- There will be only one smallest eigenvalue/eigenvector:

$$L\mathbf{1} = (D - A)\mathbf{1} = 0$$

$(\mathbf{1} = [1, 1, \cdots, 1]^T$ is the eigenvector with eigenvalue 0)

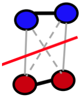
Eigenvectors of Laplacian

- What if the graph is connected?
- There will be only one smallest eigenvalue/eigenvector:

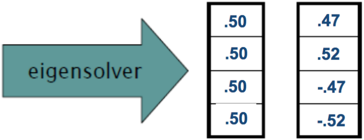
$$L\mathbf{1} = (D - A)\mathbf{1} = 0$$

($\mathbf{1} = [1, 1, \cdots, 1]^T$ 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

Sign of 2nd evec indicates blocks

Normalized Cut

- Rewrite Normalized Cut:

$$\text{NCut} = \sum_{c=1}^{k} \frac{\text{Cut}(V_c, V - V_c)}{\text{deg}(V_c)}$$
$$= \sum_{c=1}^{k} \frac{\mathbf{y}_c^T (D - A)\mathbf{y}_c}{\mathbf{y}_c^T D \mathbf{y}_c}$$

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

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

- Normalized Laplacian:

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

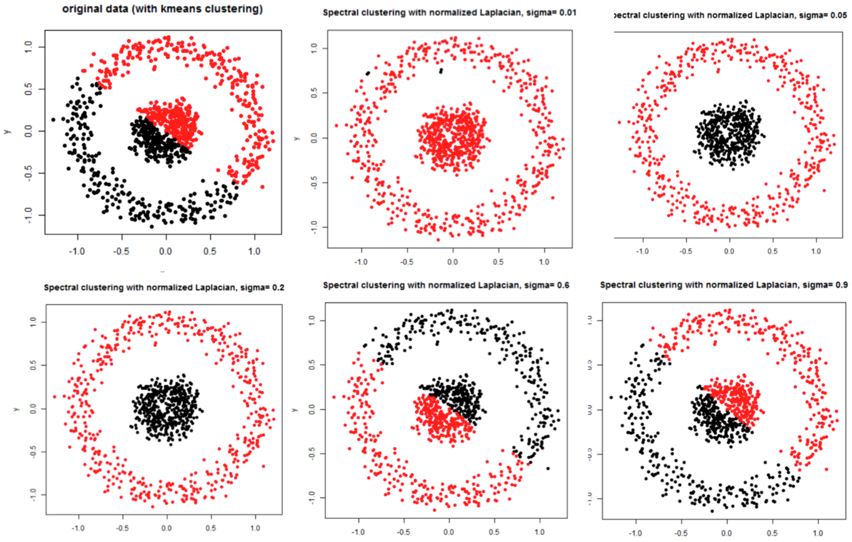
- Normalized Cut $\to$ eigenvectors correspond to the smallest eigenvalues of $\tilde{L}$

Kmeans vs Spectral Clustering

- Kmeans: decision boundary is linear
- Spectral clustering: boundary can be non-convex curves

  σ in $W_{ij} = e^{\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}}$ controls the clustering results (focus on local or global structure)

Kmeans vs Spectral Clustering



Conclusions

- Kmeans objective: minimize the distance to centers
- Kmeans algorithm: an optimization algorithm to minimize this objective
- Graph clustering $\Rightarrow$ related to eigenvectors of graphs

Questions?