Spectral Clustering

Andira Putri

1. Graph Theory and Linear Algebra

Similarity Graphs

Given a dataset $x_1, ..., x_n$ and a measure of similarity $s_{ij} \geq 0$ between all pairs of points x_i and x_j in the set, the goal of clustering is to divide observations into groups such that points in the same group are similar, and points in different groups are dissimilar. A good way of representing similarity between observations is through a **similarity graph**, notated as G=(V,E). Each vertex, $v_i \in V$, represents an observationin the dataset. Two vertices are connected by an edge when their similarity s_{ij} exceeds a threshold value. An edge is weighted by s_{ij}; the higher the similarity, the higher the weight and vice-versa. Our clustering problem now is to find a separation of the similarity graph such that edges between different groups have low weights, and edges within the same group have high weights.

An **adjacency matrix** is a numerical represenation of a similarity graph. In our case, we have a weighted adjacency matrix notated as $W = (w_{ij})_{i,j=1,..,n}$. If $w_{ij} = 0$, vertices v_i and v_j are not connected by an edge. An example of a similarity graph and its adjacency matrix is shown in Figure 1.

There are several different types of similarity graphs. The types described below are commonly used for spectral clustering.

ϵ-Neighborhood Graph

We connect the observations whose pairwise distances are smaller than ϵ. Distances between connected points cannot exceed ϵ, so weighting the edges would not give more information about the data to the graph. For this reason, the ϵ-neighborhood graph is considered an unweighted graph.

K-Nearest Neighbor (KNN) Graph

In this graph, the goal is to connect vertex v_i with vertex v_j if v_j is among is among the k-th smallest distances from v_i to other objects from the dataset. After connecting the two vertices, the edge is weighted by the similarity of their endpoints. KNN graphs are directed graphs since the relationship between vertices is not symmetric- however, in most cases, we just ignore the direction aspect of the edges and view KNN graphs as undirected.

The Fully Connected Graph

All vertices are connected with positive similarity to each other. All edges are weighted by s_{ij}. This type of graph is only useful if it represents local neigborhoods. A good example of a useful fully connected graph is the Gaussian similarity function, where a parameter σ controls the width of the neighborhoods; the parameter has a similar role to ϵ in the first type of graph.

Graph Laplacians

Graph Laplacian matrices are the main tools for spectral clustering. As we define the different graph Laplacians, we always assume that G is an undirected graph $G = (V, E)$ with a weighted matrix W, where $w_{ij} = w_{ji} \geq 0$. We also assume that the eigenvectors of a matrix are not normalized; a constant eigenvector

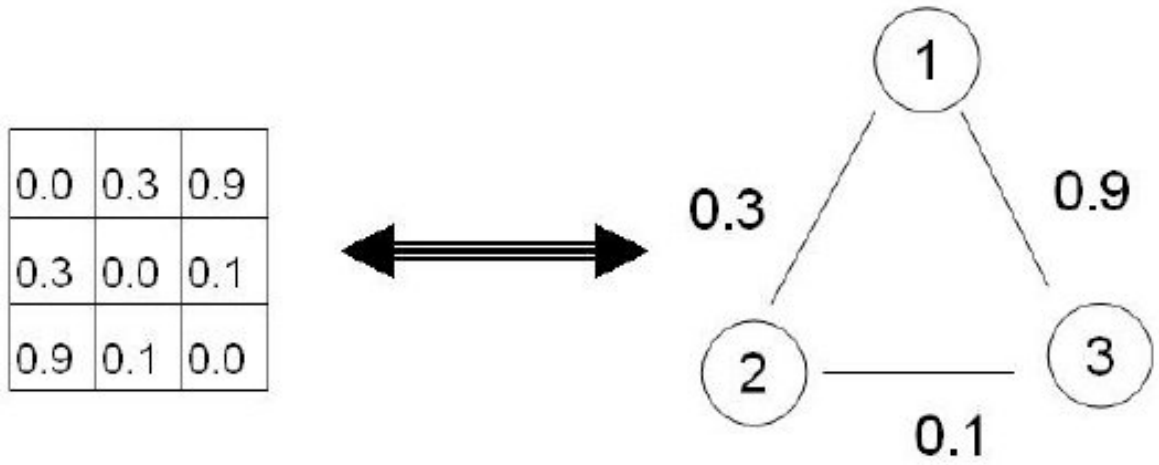

Figure 1: Similarity graph on the right and its corresponding weighted adjacancy matrix on the left. The adjacancy matrix is symmetric about the main diagonal.

and a scalar multiple of it will be considered the same. Finally, we define the degree matrix D. The degree of a vertex is given by $d_i = \sum_{j=1}^{n} w_{ij}$, and D is a diagonal matrix with the degrees $d_1, ..., d_n$ on its diagonal.

The Unnormalized Graph Laplacian

The Unnormalized Graph Laplacian matrix L is defined as $L = D - W$. The matrix L has the following properties:

1. For every vector $f \in R^n$, where R is the set of real numbers, we have: $f'Lf = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$.
2. L is symmetric and positive semidefinite. Positive semidefinite means that for a square matrix z, $z^T Lz$ is strictly positive for every non-zero column vector in z.
3. The smallest eigenvalue of L is 0, with a corresponding eigenvector of the constant one vector.
4. L has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$.
5. The multiplicity k of the eigenvalue 0 of L equals the number of connected components $A_1, ..., A_k$ in the graph. The eigenspace of eigenvalue 0 is spanned by the indicator vectors $1_{A_1}, ..., 1_{A_k}$.

Normalized Graph Laplacian

The normalized graph Laplacian we are interested in is notated as L_{sym} because it is a symmeytic matrix, and it is given by:

$L_{sym} = D^{-1/2}LD^{-1/2} = I - D^{-1/2}WD^{-1/2}$.

Another normalized graph Laplacian of interest is L_{rw}, which is closely resembles to a random walk:

$L_{rw} = D^{-1}L = I - D^{-1}W$

The most important properties of the normalized graph Laplacians for spectral clustering are as follows:

1. For every $f \in R^n$, we have $f'L_{sym}f = \frac{1}{2}\sum_{i,j=1}^{n}(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}})^2$.
2. λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ is an eigenvalue of L_{sym} with eigenvector $w = D^{1/2}u$.

3. λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ and u solve the generalized eigenproblem $Lu = \lambda Du$.

4. 0 is an eigenvalue of L_{rw} with the constant one vector 1 as an eigenvector. 0 is an eigenvalue of L_{sym} with eigenvector $D^{1/2}1$.

5. L_{sym} and L_{rw} are positive semi-definite and have n non-negative real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$.

2. Spectral Clustering

Before introducing three algorithms for spectral clustering, we first discuss k-means clustering.

k-Means Clustering

Background on K-means clustering

In unsupervised learning, clustering methods aim to find subgroups (or clusters) in the data in hopes of discovering some structure. The partitions in the data set are created by assigning similar observations to one cluster and different observations in other subgroups. What it means for observations to be "similar" or "different" is domain specific– you need to have a good understanding of the data's context. There are two main clustering methods; K-means and hierarchical clustering. K-means partitions a data set into K distinct, non-overlapping clusters.

Algorithm

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as the initial cluster assignments for the observations.

2. Iterate until the cluster assignments stop changing.

- For each of the K clusters, compute the cluster centroid which is the vector of `p` feature means for the observations in the k-th cluster.
- Assign each observation to the cluster whose centroid is the closest (via Euclidean distance)

K-means finds a local optimum rather than global. With that being said, results do depend on initial cluster assignments. The algorithm should be performed multiple times from different initial configurations. Choose the result that minimizes the Euclidean distance between observations in a cluster (i.e. within-cluster variation).

Now, we present two commonly-used spectral clustering algorithms using both unnmormalized and normalized graph Laplacians.

Unnormalized Spectral Clustering - Algorithm

Input: Similarity matrix and k, number of clusters to construct

- Construct a similarity graph. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L.
- **Compute the first k eigenvectors $u_1, ..., u_k$ of L.**
- Let U be the matrix containing the vectors $u_1, ..., u_k$ as columns.
- For $i = 1, ..., n$, let y_i be the vector corresponding to the i-th row of U.

- Cluster the points $(y_i)_{i=1,...,n}$ with the k-means algorithm into clusters $C_1, ..., C_k$.

Output: Clusters $A_1, ...A_k$ with $A_i = (j|y_j \in C_i)$

Normalized Spectral Clustering - Algorithm 1

Input: Similarity matrix and k, number of clusters to construct

- Construct a similarity graph. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L.
- **Compute the first k generalized eigenvectors $u_1, ..., u_k$ of the generalized eigenproblem $Lu = \lambda Du$.**
- Let U be the matrix containing the vectors $u_1, ..., u_k$ as columns.
- For $i = 1, ..., n$, let y_i be the vector corresponding to the i-th row of U.
- Cluster the points $(y_i)_{i=1,...,n}$ with the k-means algorithm into clusters $C_1, ..., C_k$.

Output: Clusters $A_1, ...A_k$ with $A_i = (j|y_j \in C_i)$

Normalized Spectral Clustering - Algorithm 2

Input: Similarity matrix and k, number of clusters to construct

- Construct a similarity graph. Let W be its weighted adjacency matrix.
- **Compute the normalized Laplacian L_{sym}.**
- **Compute the first k eigenvectors $u_1, ..., u_k$ of L_{sym}.**
- Let U be the matrix containing the vectors $u_1, ..., u_k$ as columns.
- **Form the matrix T from U by normalizing the rows to norm 1, that is set $t_{ij} = u_{ij}/\sqrt{\sum_k u_{ik}^2}$.**
- For $i = 1, ..., n$, let y_i be the vector corresponding to the i-th row of T.
- Cluster the points $(y_i)_{i=1,...,n}$ with the k-means algorithm into clusters $C_1, ..., C_k$.

Output: Clusters $A_1, ...A_k$ with $A_i = (j|y_j \in C_i)$

3. Exercises

http://lasa.epfl.ch/teaching/lectures/ML_MSc_Advanced/Exercises/Ex_ML_SpectralClustering.pdf