Homework 7: Graphs and clusters

Bi 183

Winter 2020

Problem 1: Minimum spanning trees

(A) Implement the following algorithm for finding the Minimum Spanning Tree of a graph from a root node v_r.

Consider a graph, $G = (V, E)$, with weight function $w(E_{i,j}) \to \mathbb{R}$. We use a series of variables for tracking the progress of the algorithm. U will contain the nodes in your MST. T will contain the edges. L will track the length or cost.

(1) Set $U = \{v_r\}$, $T = \emptyset$ and $L = 0$.

(2) while $V \setminus U \neq \emptyset$ (while $U \neq V$)

(3) Find all the edges emanating from nodes within U that connect to nodes that are not yet in U. Call this set F. $F = \{E_{ij}\}$ such that $i \in U$ and $j \in V \setminus U$.

(4) Select the edge from F that has minimum weight call this F_{min}. (*key step)

(5) Add F_{min} to T, your minimum spanning tree.

(6) Now increment $L = L + w(F_{\text{min}})$

(7) end while when $V \setminus U = \emptyset$

(8) return the tree, T, and the length, L

(B) Construct a procedure for generating a random graph, G, with 10 nodes and apply your algorithm to it. Plot your graph and plot your minimum spanning tree for two example graphs.

(C) Do graphs exist that have more than one minimum spanning tree? If so, construct such a graph.

(D) Is your algorithm in part (A) deterministic. Will it return a single tree? How could it be modified to return the complete set of minimum spanning trees?

Problem 2: Louvain Clustering implementation

(A) Write an algorithm that implements Louvain clustering on a graph by partitioning it into a set of communities. Lets call our graph G with nodes, v_i.

(1) First, write a function that calculates the modularity of a graph, $Q(A, c)$, given a partition

of the graph into a set of communities. Lets consider graphs that are unweighted with adjacency matrix, A, and $A_{i,j} = 1$ if G contains an edged, (i, j):

The modularity of is defined as:

$$Q = \frac{1}{2\ m} \sum_{ij} (A_{ij} - \frac{k_i\ k_j}{2m}) \delta(c_i, c_j),$$

where $k_i = \text{degree}(v_i)$, $m = \frac{1}{2} \sum_{ij} A_{ij}$, and $\delta(c_i, c_j) = 1$ if $i = j$ and $\delta(c_i, c_j) = 0$ if $c_i \neq c_j$.

(2) Initialize $c_i = i$ putting each node into its own community.

(3) Now, iterate through nodes in the graph. For each node, i, place the node into the community of each neighboring node while removing it from its own community. When you move a node into a community, c_j, remove the node from community c_i. Calculate $Q_{i \to j}$.

(4) For each round, select the move with maximum $Q_{i \to j}$, and make the corresponding change in community structure. If no move increase Q, leave the community partition unchanged.

(5) Iterate the algorithm until Q reaches a max score.

(B) Test your Louvain clustering by generating, n, points from two multivariate normal distributions with μ_1 and μ_2. Sample points from two multivarite normal distributions in a 2D space. Select μ_i and covariance so that the clusters slightly overlap.

Generate data and construct a k-NN graph using the sampled points, and run Louvain clustering on the resulting graph. Plot the points and color the points by community.

(C) Perform Louvain clustering while changing the space between the centroids (μ_1 and μ_2) as well as the covariance. Select two more examples of interest and show community architure and K-nn graph. Select your examples to show how Louvain performs when communities become close together.

(D) Now perform Louvain on synthetic examples of size $10, 50, 100$, and plot Q vs *iteration* on the same plot.

Problem 3: Louvain on the PBMC data

Down load PBMC data from last week.

(A) Generate a k-NN graph for the data for $k = 4, 10, 20$, and plot the resulting k-NN graphs in the 2D space defined by the first two principle components of the data.

When you generate the graph, you will need to enforce symmetry.
(B) Perform Louvain clustering using your code for $k = 4$. If the code is too slow, please seek help

on optimization.

(C) Plot the cells from the data set in 2D and 3D pca space defined by the first 2 or 3 PCs and color points by community. What do you notice?
(D) generate a heatmap of the data matrix where you include a new row that colors cell by community.