Clustering Methods and Correlated Data: Simulations

Parker Knight

9/29/2021

Overview

Methods of Interest

1. K-means
2. Hierarchical clustering
3. Spectral clustering

K-means

Let $X \in \mathbb{R}^{n \times p}$ be a given data matrix. Our goal is to partition the n data points into k non-overlapping clusters, denoted by the set $S = \{S_1, S_2, ..., S_k\}$.

The k-means algorithms does so by minimizing the Euclidean distance between each point and the centroid of it's assigned cluster:

$$\min_S \sum_{i=1}^k \sum_{X_j \in S_i} ||X_j - \mu_i||_2^2$$

where μ_i is centroid of the points in S_i.

K-means (cont.)

Key points:

- k must be specified up front
- Computation is NP-hard (objective function is not convex)
- By default, uses Euclidean distances
    - alternate versions exist (K-medoids)
- Assumes separation between clusters is convex

Hierarchical clustering

Let $D = \mathrm{dist}(X)$ be the pairwise distance matrix of the n samples.

In agglomerative HC, we start with every observation in a singleton cluster, join clusters G and H together based on the following criteria:

   i. $d_{SL} = \min_{i \in G, j \in H} d_{i,j}$ (single linkage)
  ii. $d_{CL} = \max_{i \in G, j \in H} d_{i,j}$ (complete linkage)
 iii. $d_{GA} = \frac{1}{N_G N_H} \sum_{i \in G} \sum_{j \in H} d_{i,j}$ (group average)

In divisive HC, we begin with data all in one cluster and split into sub-clusters with K-means.

HC (cont.)

Key points:

- don't need to specify k up front, can 'cut' the tree at any depth
- relies on pairwise distances, sensitive to outliers
- greedy algorithm
- 'big data' issues

Spectral clustering

From our data matrix X, compute a similarity matrix $S \in \mathbb{R}^{n \times n}$.
Then:

1. Compute a similarity graph from S, and define W to be the 'weighted' adjacency matrix
2. Construct the Laplacian L from W
3. Compute $L = UDU^T$
4. Cluster the first k columns of U using k-means

Spectral (cont.)

Key points:

- ▶ How do we construct S? How do we construct a graph from S?
- ▶ Normalized or unnormalized Laplacian?
- ▶ Need to specify k
- ▶ Can handle non-convexity between clusters

Simulation Design

Question: how robust are these methods to correlation structure between the samples in X?

We vary the following parameters:

- n = 100
- p = 50,100
- # of clusters = 2, 3, 4
- # of groups = 2, 10, 50
  - the 'groups' determine the block-diagonal structure of the covariance matrix
- ρ = 0.2,0.5,0.8
  - within-group correlation
- difference in mean between groups: 0.25,1
  - 'signal strength'

Simulation Design (cont.)

Given a set of parameters, repeat R times:

1. Generate block-diagonal Σ with `ngroups` blocks, set
   within-group correlation ρ

```
Sigma <- Matrix::bdiag(lapply(1:(n/ngroups), function(i){
    I <- (1-rho) * diag(ngroups)
    ones <- rho * rep(1 ,ngroups) %*% t(rep(1, ngroups))
    I + ones
  }))
```

Simulation Design (cont.)

2. Generate vector of cluster assignments $\mathbf{C} = (C_1, C_2, ..., C_n)^T$ where C_i may equal $1, 2, \ldots, k$. Set the mean vector $\mu = (\mu_{C_1}, \mu_{C_2}, ..., \mu_{C_n})^T$ where μ_{C_i} is the mean of cluster C_i.

```r
cluster_names <- LETTERS[1:nclust]
cluster_assignment <- sample(cluster_names,
                             size = n,
                             replace = TRUE)
mu <- (0:(nclust - 1)) * mu_step_size
mu_vec <- rep(0, n)
for (cl in 1:length(cluster_names))
    mu_vec <- mu_vec + ifelse(cluster_assignment ==
                              cluster_names[cl],
                        mu[cl],
                        0)
```

Simulation Design (cont.)

3. Draw $X \sim MVN(\mu, \Sigma)$.

```
X <- t(MASS::mvrnorm(n = p, mu = mu_vec, n/nclust,
        Sigma = Sigma))
```

4. Cluster the rows of X with each method to get an estimated cluster assignment $\hat{\mathbf{C}} = (\hat{C}_1, ..., \hat{C}_n)^T$ and compare results...

Simulation Design (cont.)

How do we evaluate the performance of each clustering method?
Rand index:

$$RI = \frac{1}{\binom{n}{2}} \sum_{i=1}^{n} \sum_{j \neq i} I \left[(C_i = C_j \wedge \hat{C}_i = \hat{C}_j) \vee (C_i \neq C_j \wedge \hat{C}_i \neq \hat{C}_j) \right]$$
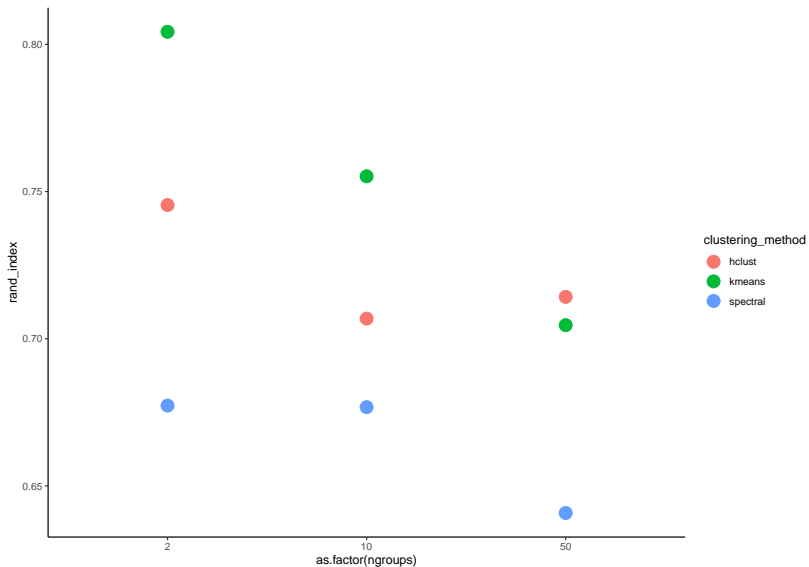
Simulation Design (cont.)

Some immediate concerns:

1. We set k to be the true number of clusters
2. We assume that cluster means are equally spaced apart (signals are equally strong).
3. Many options for spectral clustering... our method (after some trial-and-error) is:

- ▶ Construct S with a Gaussian similarity kernel with $\sigma^2 = 25$
- ▶ Build an unweighted K-nearest neighbors graph with $K = 25$ (relatively dense)
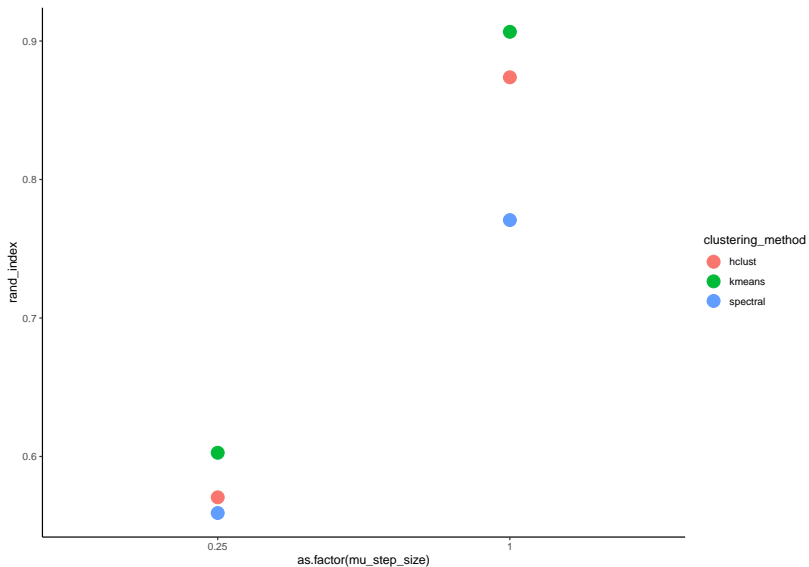- ▶ Use a normalized Laplacian $L = I - D^{-1/2}AD^{-1/2}$

Results

clustering_method	average_ri
hclust	0.7221509
kmeans	0.7546772
spectral	0.6649400

Results (cont.)

Results (cont.)

Results (cont.)