STAT 3690 Lecture Note

Part X: Clustering

Zhiyang Zhou (zhiyang.zhou@umanitoba.ca, zhiyanggeezhou.github.io)

2023/Apr/09 20:41:23

Clustering

- Problem: given observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^p$ group the observations into K populations
  - Unknown K
  - Unsupervised: no label/training data
- Why
  - Summarize a representation of the full data set
  - Exploration for structure of the data
  - Checking the validity of pre-existing group assignments
  - Assistance for prediction: sometimes clustering prior to prediction
- Find a function C such that
  - $C(i) = k$: assign the ith subject to group k

K-means

- Within-cluster scatter

$$W(K) = \frac{1}{2}\sum_{k=1}^{K}\frac{1}{n_k}\sum_{i:C(i)=k}\sum_{j:C(j)=k}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 = \sum_{k=1}^{K}\sum_{i:C(i)=k}\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|_2^2$$

  - $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$: the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$
  - $\bar{\boldsymbol{x}}_k = n_k^{-1}\sum_{i:C(i)=k}\boldsymbol{x}_i$
  - Smaller $W(K)$ is better

- (Approximately) minimizing the within-cluster scatter

$$\min_C W(K) = \min_{C,\boldsymbol{c}_1,\ldots,\boldsymbol{c}_K}\sum_{k=1}^{K}\sum_{i:C(i)=k}\|\boldsymbol{x}_i - \boldsymbol{c}_k\|_2^2$$

- Implementation:

  1. Specify K and start with an initial guess for $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K$, then repeat
     a. Labeling each point based the closest center: for each i, put $\boldsymbol{x}_i$ to the kth cluster such that $\boldsymbol{c}_k$ is closest to $\boldsymbol{x}_i$
     b. Replacing each center by the average of points in its cluster: for each k, take $\boldsymbol{c}_k = \bar{\boldsymbol{x}}_k$
  2. Terminate when $W(K)$ doesn't change

- Comments
  - Always converge via the expectation maximization (EM) algorithm
  - No guarantee to lead to the smallest W
  - Depend on K and initial cluster centers
    * Typically run K-means multiple times and pick up the result with the smallest W
- Determining K
  - Between-cluster variation

$$B(K) = \sum_{k=1}^{K} n_k \|\bar{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}\|_2^2$$

    * $\bar{\boldsymbol{x}} = n^{-1} \sum_{i=1}^{n} \boldsymbol{x}_i$
  - CH index (Caliński & Harabasz (1974), *Communications in Statistics*, 3:1–27)

$$\mathrm{CH}(K) = \frac{B(K)/(K-1)}{W(K)/(n-K)}$$

  - To choose K as the maximizer of CH(K), i.e.,

$$\widehat{K} = \arg \max_{K \in \{2,\ldots,K_{\max}\}} \mathrm{CH}(K)$$

- Example 10.1 (K-means for `iris`)

```
set.seed(1)
options(digits = 4)
x = iris[, !(names(iris) %in% c('Species'))]
y = (iris$Species == unique(iris$Species)[1]) +
  2*(iris$Species == unique(iris$Species)[2]) +
  3*(iris$Species == unique(iris$Species)[3])

## K-means via the first two PCs of x (to facilitate the visualization)
decomp = prcomp(x)
s = 2
PCscores = decomp$x[,1:s]
K = 3; cols = c("red","darkgreen","blue")
km = kmeans(PCscores, centers=K, nstart=1, algorithm="Lloyd", iter.max = 100)
# clustering plot with centers
plot(PCscores,col=cols[km$cluster])
points(km$centers,pch=19,cex=2,col=cols)
# comparison with true groups
par(mfrow=c(1,2))
plot(PCscores,col=cols[km$cluster], main="K-means")
plot(PCscores,col=cols[y], main="True")

## determine K for K-means via original x
set.seed(3690)
Ks = 2:20
Ws = numeric(length(Ks))
Bs = numeric(length(Ks))
CHs = numeric(length(Ks))
for(l in 1:length(Ks)){
  km = kmeans(x, centers=Ks[l], nstart=1, algorithm="Lloyd", iter.max = 100)
  Ws[l] = km$tot.withinss
  Bs[l] = sum(km$size * rowSums(sweep(km$centers, 2, colMeans(PCscores))^2))
  CHs[l] = (Bs[l]/(Ks[l]-1))/(Ws[l]/(nrow(PCscores)-Ks[l]))
```

```
}
plot(Ks, CHs,
  type="b", pch = 19,
  xlab="Number of clusters K",
  ylab="CH index")
```

- Color quantization/vector quantization (an application of K-means to image compression)
  - Basic idea: compress images by reducing the color palette of an image to K colors
  - Implementation:
    1. Let x_i be the quantifed color of the ith pixel to be
    2. Initiate with K colors, say $c_1, \ldots, c_K$, and then repeat the following steps until convergence
       a. Classifying the ith pixel into the kth cluster if x_i is closest to c_k AND replacing x_i with c_k
       b. Updating c_k by $\sum_{i \in \text{ the } k\text{th cluster}} x_i/n_k$ with n_k as the size of the kth cluster

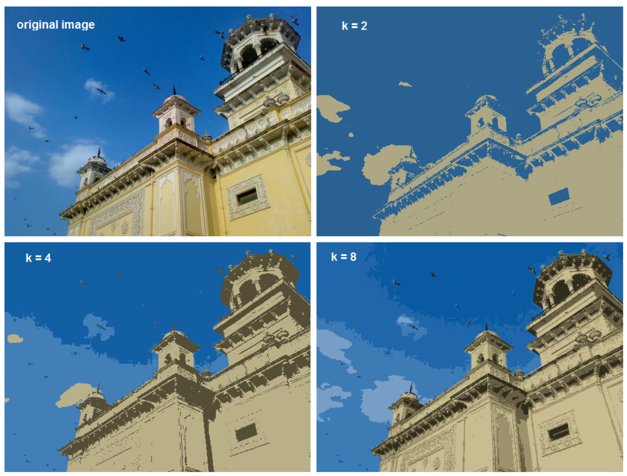

Figure 1: Image compression with K-means clustering (http://opencvpython.blogspot.com/2012/12/k-means-clustering-2-working-with-scipy.html)

- Example 10.2 (Color quantization for hand-written digits)

```
options(digits = 4)
mnist = dslabs::read_mnist()

# The 3690th image in the training set
i0 = 3690
image(z = matrix(mnist$train$images[i0,], ncol = 28),
  col = gray.colors(256, start = 0, rev = TRUE), axes = FALSE, main = "True")

# Shrink the number of colors to be 2
set.seed(1)
K = 2
km = kmeans(mnist$train$images[i0,], centers=K, nstart=1, algorithm="Lloyd", iter.max = 100)
image(z = matrix(km$cluster, ncol = 28),
  col = gray.colors(K, start = range(km$centers)[1]/255, end = range(km$centers)[2]/255, rev = T),
  axes = FALSE, main = "Compressed")
```

Hierarchical clustering

- A simple example
  - Step 1: {1}, {2}, {3}, {4}, {5}, {6}, {7};
  - Step 2: {1}, {2, 3}, {4}, {5}, {6}, {7};
  - Step 3: {1, 7}, {2, 3}, {4}, {5}, {6};
  - Step 4: {1, 7}, {2, 3}, {4, 5}, {6};
  - Step 5: {1, 7}, {2, 3, 6}, {4, 5};
  - Step 6: {1, 7}, {2, 3, 4, 5, 6};
  - Step 7: {1, 2, 3, 4, 5, 6, 7}.
- Dendrogram: a tree displaying a hierarchical sequence of clustering assignments
  - Node representing a group
    - Leaf node representing a singleton (i.e., a group containing a single data point)
    - Root node representing the group containing all the data points
    - Internal node: has two children nodes, representing the the groups that were merged to form it
  - Height: draw each internal node at a height proportional to the dissimilarity between its two children nodes (if fix the leaf nodes at height zero)
- Distances
  - Dissimilarity d_{ij}: (Euclidean) distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$
  - Linkage: distance between groups G and H
    - Options
      - Single linkage
$$d_{\text{single}}(G, H) = \min_{i \in G, j \in H} d_{ij}$$
      - Complete linkage
$$d_{\text{complete}}(G, H) = \max_{i \in G, j \in H} d_{ij}$$
      - Average linkage
$$d_{\text{average}}(G, H) = \frac{1}{n_G n_H} \sum_{i \in G, j \in H} d_{ij}$$
      - Centroid linkage
$$d_{\text{centroid}}(G, H) = \|\bar{\boldsymbol{x}}_G - \bar{\boldsymbol{x}}_H\|_2$$
      - Minimax linkage
$$d_{\text{minimax}}(G, H) = \min_{i \in G \cup H} \max_{j \in G \cup H} d_{ij}$$
    - Situation-dependent

- Example 10.3 (hierarchical clustering for `iris`)

Modern alternatives

- Density-based spatial clustering of applications with noise (DBSCAN, M. Ester, H. Kriegel, J. Sander, X. Xu (1996), *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD).*)
- Uniform manifold approximation and projection for dimension reduction (UMAP, L. McInnes & J. Healy (2018), arXiv:1802.03426)