STAT 154 Lab 4: Clustering methods

Yuansi Chen and Raaz Dwivedi

Feb 25, 2019

The goal of this lab is to go over three methods of clustering: Gaussian Mixture Model, K-means, hierarchical clustering.

Remark: You need to set the eval=TRUE on line 8 of the Rnw file to Run the code when you compile the pdf. No change is needed if you only run code chunks.

1 K-means clustering

First, we discuss one of the powerful methods in machine learning namely, K-means.

1.1 K-means math

Recall that given data samples $x_1, \ldots, x_n$ each $\in \mathbb{R}^d$, in K-means clustering we attempt to minimize the following objective,

$$\min_{C_1, C_2, \ldots, C_K} \sum_{i=1}^{K} \sum_{x_j \in C_i} \|x_j - \mu_i\|_2^2 , \tag{1}$$

where

$$\mu_i = \arg\min_{\mu_i \in \mathbb{R}^d} \sum_{x_j \in C_i} \|x_j - \mu_i\|_2^2 = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j, \ i = 1, 2, \ldots, k.$$

C_i is the set of samples assigned to cluster i. Each sample is assigned to exactly one cluster, and clusters are non-empty.

(a) What is the minimum value of the objective (1) when $k = n$ (the number of clusters equals the number of samples)?

(b) Formulate the PCA computation as an approximate matrix factorization problem.

(c) Formulate the K-means computation as another approximate matrix factorization problem. What is the difference and similarity to the previous one?

(d) (Regularized K-means) Suppose we add a regularization term to the objective (1). That is, the new objective becomes

$$\min_{C_1,C_2,\ldots,C_K} \sum_{i=1}^{K} \left(\lambda \left\|\mu_i\right\|_2^2 + \sum_{x_j \in C_i} \left\|x_j - \mu_i\right\|_2^2 \right). \tag{2}$$

Show that the optimum of the minimization problem

$$\min_{\mu_i \in \mathbb{R}^d} \lambda \left\|\mu_i\right\|_2^2 + \sum_{x_j \in C_i} \left\|x_j - \mu_i\right\|_2^2.$$

is obtained at

$$\mu_i = \frac{1}{|C_i| + \lambda} \sum_{x_j \in C_i} x_j.$$

(e) When do we need to use regularized K-means? Suppose there are n students who wish to share rides efficiently to the lecture tomorrow. University provides K vehicles for taking students to the campus. We need to figure out K good locations to meet up. The students will then walk to the closest meet up point and then the vehicles will deliver them to the campus. Define x_j as the home location of student j, and $(0, 0)$ be the campus location. Assume that we can drive as the crow flies, i.e. by taking the shortest paths between two points. Write an objective that minimizes the total distance that the students and vehicles need to travel. [1]

(f) What if we minimize the total travel time?

(g) (K-medoids) The k-medoidsis a clustering algorithm reminiscent to the k-means algorithm. In contrast to the k-means algorithm, k-medoids chooses datapoints as centers (medoids) and can be used with arbitrary distances. How would you change the objective (1)?

1.2 K-means implementation

One practical implementation of $K - means$ works as follows

1. Randomly select K rows of the dataset and treat them as the initial cluster centroids $g_1, \ldots, g_K$.
2. For each observation x_i,
   (a) Compute $d(x_i, g_k) = \left\|x_i - g_k\right\|_2^2$ for each k.

[1] Example taken from Cathy Wu, Ece Kamar, Eric Horvitz. Clustering for Set Partitioning with a Case Study in Ridesharing. IEEE Intelligent Transportation Systems Conference (ITSC), 2016.

(b) Find $k^* = \arg\min_{k=1,\ldots,K} d(x_i, g_k)$.

(c) Assign x_i to cluster k^*.

3. For each cluster C_k, compute the cluster centroid $g_k = \frac{1}{|C_K|} \sum_{i \in C_k} x_k$.

4. Repeat Step 2 and Step 3 until convergence (That is, the cluster assignment does not change anymore).

Implement a function **my_kmeans()** in R, which takes X a $n \times d$ matrix and K the number of clusters as input, and outputs a list of the following,

- **cluster_sizes**: a length K vector of cluster sizes
- **cluster_means**: a $K \times d$ matrix in which each row correspond to a cluster center
- **clustering_vector**: a vector of length n containing the cluster assignment for each observation.
- **wss_cluster**: a vector of length K containing the within-cluster sum-of-squares.
- **tss**: the total sum-of-squares.

On the **iris** dataset, with $K = 3$. Compare the following methods

1. **my_kmeans()**
2. **kmeans()** in R
3. **fastkmed()** the K-medoid method in the **kmed** package of R.
4. **my_kmeans_regularized()**, your regularized version to be implemented.

2 From K-means to EM

We now discuss the connection between K-means and EM. The parts are briefly sketched here (and will be discussed in detail in in the lab).

1. Suppose that we modify the K-means objective to a weighted one as follows:

$$\min_{\mu_1,\mu_2,\ldots,\mu_K} \sum_{i=1}^{K} \sum_{j=1}^{n} w_{ij} \|x_j - \mu_i\|_2^2 \quad \text{where} \quad w_{ij} = \frac{\exp(-\alpha \|x_j - \mu_i\|_2^2)}{\sum_{j'=1}^{n} \exp(-\alpha \|x_{j'} - \mu_i\|_2^2)},$$

where $\alpha > 0$ is a tuning parameter. What will be a sensible heuristic to solve this problem? Do you see it as a "soft" version of K-means?

2. What happens when we take $\alpha \to \infty$?

3. We shall discuss the derivation of EM on board.

4. Generate $N = 400$ data points from different three mixture of Gaussians in two dimensions. Scatter plot them and color them based on their labels. Use K-means and EM to obtain their centers. Repeat the experiments with different mixture weights, covariances and compare the performances of K-means and EM. Plot the true centers and estimated centers on the scatter plot. Comment on the differences. The following function might be useful for generating the data. Read about different functions and then you may be able to use the code well.

```
# change the eval to TRUE if you want to
# run this code while generating the pdf.

gen_mixture_data <- function(probs, mus, sigs, N){

  num_points <- table(sample(1:K,prob=probs,size=N,replace=TRUE))

  x <- matrix(, nrow = 0, ncol=2)
  z <- c()
  for (k in 1:K){
    x <- rbind(x, mvrnorm(num_points[k], mus[k, ], sigs[,,k]))
    z <- c(z, rep(k, num_points[k]))
  }
  return(list("x"=x, "z"=z))
}


# N <-
# K <-
# probs <-
# mus <-
# vars <-
# sigs <-
#
# mixture_data <- gen_mixture_data(probs, mus, sigs, N)
#
# x <- mixture_data£x
# z <- mixture_data£z

# now plot the data using plot OR ggplot() + geom_point
#  ggplot() + geom_point(aes(x=x[, 1], y=x[, 2], col=colors[z])) +
  # labs(x = "X1", y = "X2")
```

5. When is K-means expected to perform poorly compared to EM? What might be some issues with EM?

3 Hierarchical clustering

Hierarchical clustering is a clustering method that builds the cluster assignment bottom-up: we first begin with each of the observation being a cluster (and hence there are n clusters initially), and gradually merge the clusters until there is only one cluster left. It results in a tree-based representation of the observations, called a *dendrogram*.

To decide which clusters to merge at each step, we have to define a suitable distance between two clusters (that is, two groups of observations). Once such a distance is defined, we can simply compute the pairwise distances among all of the clusters and merge the two clusters with the shortest distance from each other. There are four common types of cluster dissimilarity measures, also known as *linkages*: complete linkage, average linkage, single linkage, and centroid linkage. See TABLE 10.2. in the textbook.

(a) Use **hclust()** to perform hierarchical clustering on the iris dataset. Compare linkage, average linkage and single linkage. Save the resulting fits to hc.complete, hc.average and hc.single respectively.

(b) For each linkage,

- plot the dendrogram using **plot()**.
- use **cutree()** with $K = 3$.

(c) Based on the results from **cutree()**, how does each linkage perform? Do they all separate (roughly) the observations into correct group?