MATH 690: Topics in Probablity Theory October 3, 2017

Clustering

Lecturer: Xiuyuan Cheng *Scribes: Dev Dabke and Andrew Cho*

1 Introduction

The lecture covered

- A discussion of the consistency of Lloyd's k-means algorithm [KK10]
- An introduction to Spectral Clustering

2 k-means clustering

2.1 Lloyd's algorithm

Recall Lloyd's algorithm from last lecture that given some data points as input $\{x_i\}_{i=1}^n$, we want to search for k clusters $C = \{C_1, \dots, C_k\}$ with "centers" $\mu = \{\mu_1, \dots, \mu_k\}$.

Definition 2.1: Objective Functions

Depending on our constraints, also recall we can choose different objective functions and have different interpretations of what these clusters represent.

$$\min_{C,\mu} \sum_{l=1}^{k} \sum_{i \in C_l} \|x_i - \mu_l\|_2^2 \qquad \text{k means}$$

$$\min_{C,\mu} \sum_{l=1}^{k} \sum_{i \in C_l} \|x_i - \mu_l\|_1 \qquad \text{k medians}$$

$$\min_{C,\mu} \sum_{l=1}^{k} \sum_{i \in C_l} \|x_i - \mu_l\|_2 \qquad L_2 \text{ - } L_1 \text{ norm}$$

We can either choose initial seeds by randomly selecting k points or by singular value decomposition (SVD). Ultimately, we are interested in knowing if these results are *consistent*. Can these seeding methods lead to the true centroids?

2.2 Strong Law of Large Numbers

Definition 2.2: Loss

$$\mathcal{L}_n(\mu) = \int \|x - \mu\|^2 dP_n(x)$$

where

$$dP_n(x) = \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}(x)dx$$

which is the empirical measure of the data.

Definition 2.3: Element-Set Norm

Where $A = \{a_1, \ldots, a_n\}$ for all $x \in \mathbb{R}$, we define $\|x - A\|$ to be

$$\|x - A\| = \min_{1 \le i \le n} \|x - a_i\|$$

Theorem 2.1: Strong Law of Large Numbers

As $n \to \infty$

$$\mathcal{L}(\mu) = \int \|x - \mu\|^2 dP(x)$$
$$\min_{\mu} \mathcal{L}_n(\mu) \to \min_{\mu} \mathcal{L}(\mu)$$

2.3 Consistency of Lloyd

If we satisfy some requirements of "separation," then we can provide some bound for misclassification, assuming some true centroid partition. [KK10] $\|\mu_k^* - \mu_l^*\|$ if $k \neq l$ needs to be bigger than a gap Δ_{kl}.

$$\Delta_{kl} > c\frac{\sigma_l}{\sqrt{w_{min}}} \qquad w_{min} = \min_l \frac{|C_l^*|}{n}$$

w_{min} denotes proportions of points in cluster l. Letting σ_k^2 be the variance of the data in the k^{th} cluster and if we can assume that $\sigma_k \approx \sigma_l$, then we know that misclassification error from SVD initialization occurs less than some $\epsilon * n$ with high probability.

2.4 In Practice

There are some practical considerations for k-means.

1. What is k?
2. How to choose an initial seed
3. Some cases of k-means fails
   (a) σ_k might be too large compared to separation
   (b) Clusters might be too small (i.e. cluster sizes may not be balanced)
   (c) Cluster might be convex and piecewise can't do concave (Voronoi)

3 Spectral clustering

3.1 Graph Laplacian

Given some data $\{x_i\}_{i=1}^n$ and k, we want to

1. Build positive-definite, symmetric affinity matrix $W_{n\times n}$ by k nearest neighbors, ϵ - neighbor, or the Gaussian kernel $W_{ij} = e^{-\frac{\|x_i - x_j\|^2}{\epsilon}}$.
2. Consider the eigenvalue decomposition of the graph Laplacian $\mathcal{L}$. Note that

$$\begin{aligned} L_{un} &= D - W && \text{Unnormalized} \\ L_{rw} &= D^{-1}(D - W) && \text{Shi-Malik '00} \\ L_{sym} &= I - D^{-1/2} W D^{-1/2} && \text{Ng-Jordan-Weiss '02} \end{aligned}$$

3. Apply k means to $\Psi = [\varphi_i, \ldots, \varphi_k]_{n\times k}$ and denote y_i as the i^{th} row of Ψ.

Remark 3.1: Thresholding

If $k = 2$, you can use thresholding because the first eigenvalue is constant. Thus, you can use $\text{sign}(\Psi_2)$ to indicate clustering.

Definition 3.1: Connected Components

Let $\mathcal{G} = (V, E)$ such that on each edge $(i, j) \in \mathcal{G}$ that $w_{ij} > 0$. If node i is connected to node j, there is a path from i to j. So then, set A is a connected component if every pair of i and j is connected and A is the maximum set that satisfies this condition to preserve connectivity.

Theorem 3.1: Eigenspace of $\lambda = 0$ of $\mathcal{L}$

Suppose the graph has k connected components $A_1, \ldots, A_k$. Then the eigenspace of $\lambda = 0$ of $\dim k$ is spanned by

$$\{\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_k}\}$$

where

$$\mathbf{1}_{A_i}(i) = \begin{cases} 1 & i \in A \\ 0 & \text{otherwise} \end{cases}$$

Proof 3.1: Eigenspace of $\lambda = 0$ of $\mathcal{L}$, Theorem 3.1

Suppose $\mathbf{f}$ is an eigenvector with $\lambda = 0$. So $\boldsymbol{\mathcal{L}f} = \mathbf{0}$ and $\mathbf{f}^T\boldsymbol{\mathcal{L}f} = \mathbf{0}$. We also know

$$\boldsymbol{f}^T\boldsymbol{\mathcal{L}f} = \mathbf{0} = \frac{1}{2}\sum_{(i,j)} w_{i,j}(f_i - f_j)^2$$

and

$$\boldsymbol{f}^T\boldsymbol{\mathcal{L}f} = \mathbf{0} \iff \forall i, j \; f_i = f_j$$

whenever $w_{i,j} > 0$. Thus, $\boldsymbol{f}$ is a piecewise constant in each of the connected components.

Meanwhile, for each $\boldsymbol{v} \in \text{span}\{\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_k}\}$, we see that $\boldsymbol{v}^T \mathcal{L} \boldsymbol{v} = \mathbf{0}$.

Exercises

- Does this generalize to $\mathcal{L}_{rw}$ and $\mathcal{L}_{sym}$?
- How consistent is spectral clustering?

4 Applications

4.1 Spectral vs. k-means Clustering

When should we use spectral clustering and when should we use k-means clustering? One prototypical example is with spirals. k-means excels when the clusters are assumed to be dense point clouds separated from other dense point clouds. Spectral clustering works well where clusters may have many points with small local distances (with some metric), i.e. concentric circles. The connective relation between clusters are more important for spectral clustering.

Spectral clustering can perform as well as k-means in most clustering tasks. But one disadvantage of spectral clustering is its *computational expense*. It requires an additional step in computing the affinity matrix. Furthermore, for a data set of N points and P features, the dimension for k-means is N $\times$ P, whereas it is N $\times$ N for spectral clustering.

To analyze this situation, we can use `R` to study this behavior.

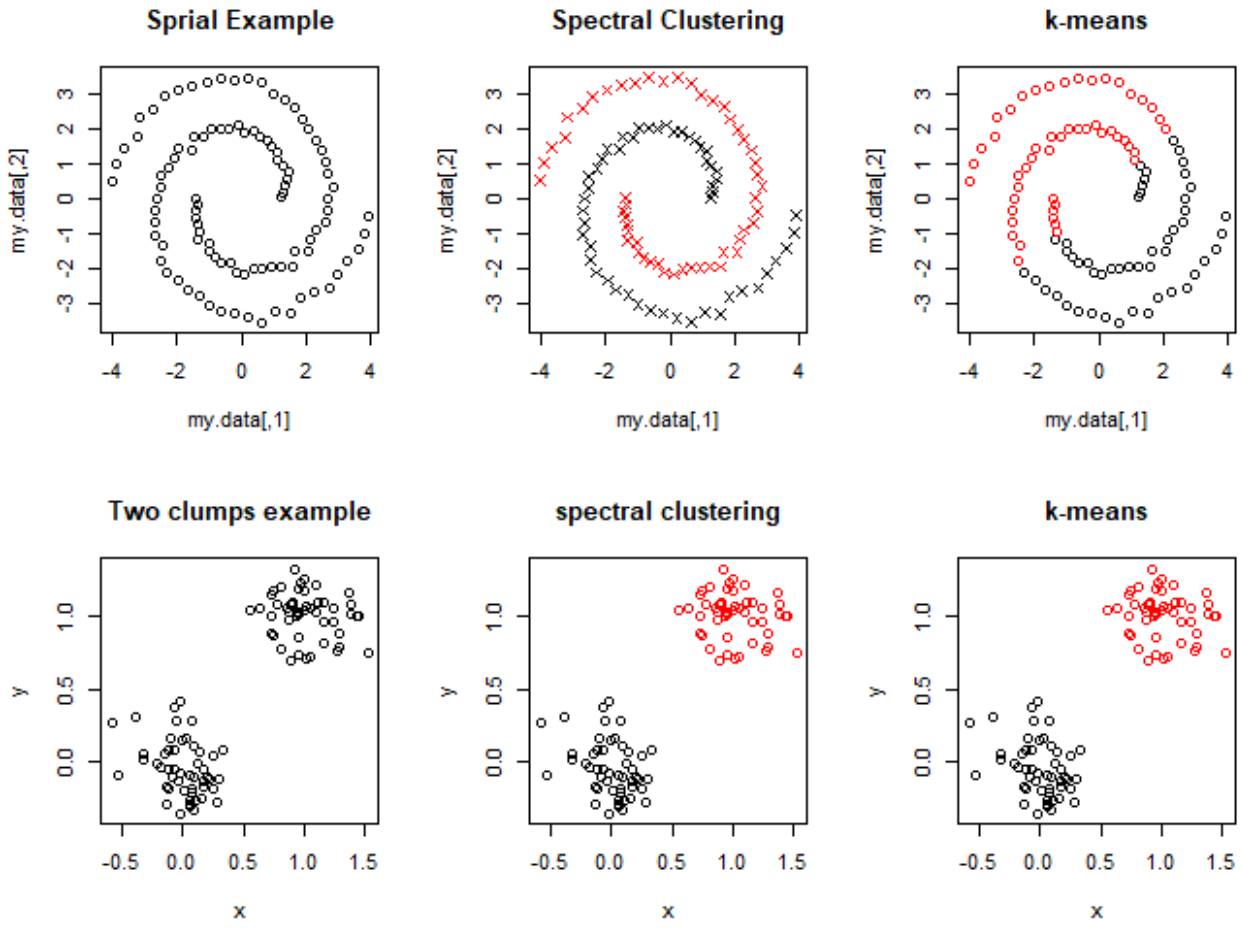

Figure 1: Example of spectral clustering vs. k-means.

```
library(kernlab)
library(mlbench)
par(mfrow = c(2,3))

#################################### Spiral example ################################
obj = mlbench.spirals(100,1,0.025)
my.data = 4 * obj$x

# spectral clustering
plot(my.data, main = "<<Sprial Example>>")

sc = specc(my.data, centers=2)
plot(my.data, col=sc, pch=4, main  = "spectral clustering")

# kmeans
clusters = kmeans(my.data, 2, iter.max = 10, nstart = 1, algorithm = "Lloyd", trace=FALSE)
plot(my.data, col = clusters$cluster, main  = "k-means")

################################3#### Two clumps example ############################
x = rbind(matrix(rnorm(100, sd = 0.2), ncol = 2),
          matrix(rnorm(100, mean = 1, sd = 0.2), ncol = 2))
colnames(x) = c("x", "y")
plot(x, main = "<<Two clumps example>>")

# spectral clustering
sc = specc(x, centers = 2)
plot(x, col = sc, main = "spectral clustering")

# kmeans
clusters = kmeans(x, 2)
plot(x, col = clusters$cluster, main = "k-means")
```

sc_kmeans_example.R

4.2 Using k-means on Images

We can also use k-means to compress an image. Though not an immediately obvious application of clustering, image compression can rely on k-means to cluster pixel values represented by 0 to 255. Thus, if we have uni-color photo that is defined by pixel values 0 to 255, we can find k groups to cluster pixels and compress the original image using k pixel values.

We can, as an example, compress the Raccoon image in Figure 2. The original image and compression from $k = 2$, 4, and 6 are shown as examples.

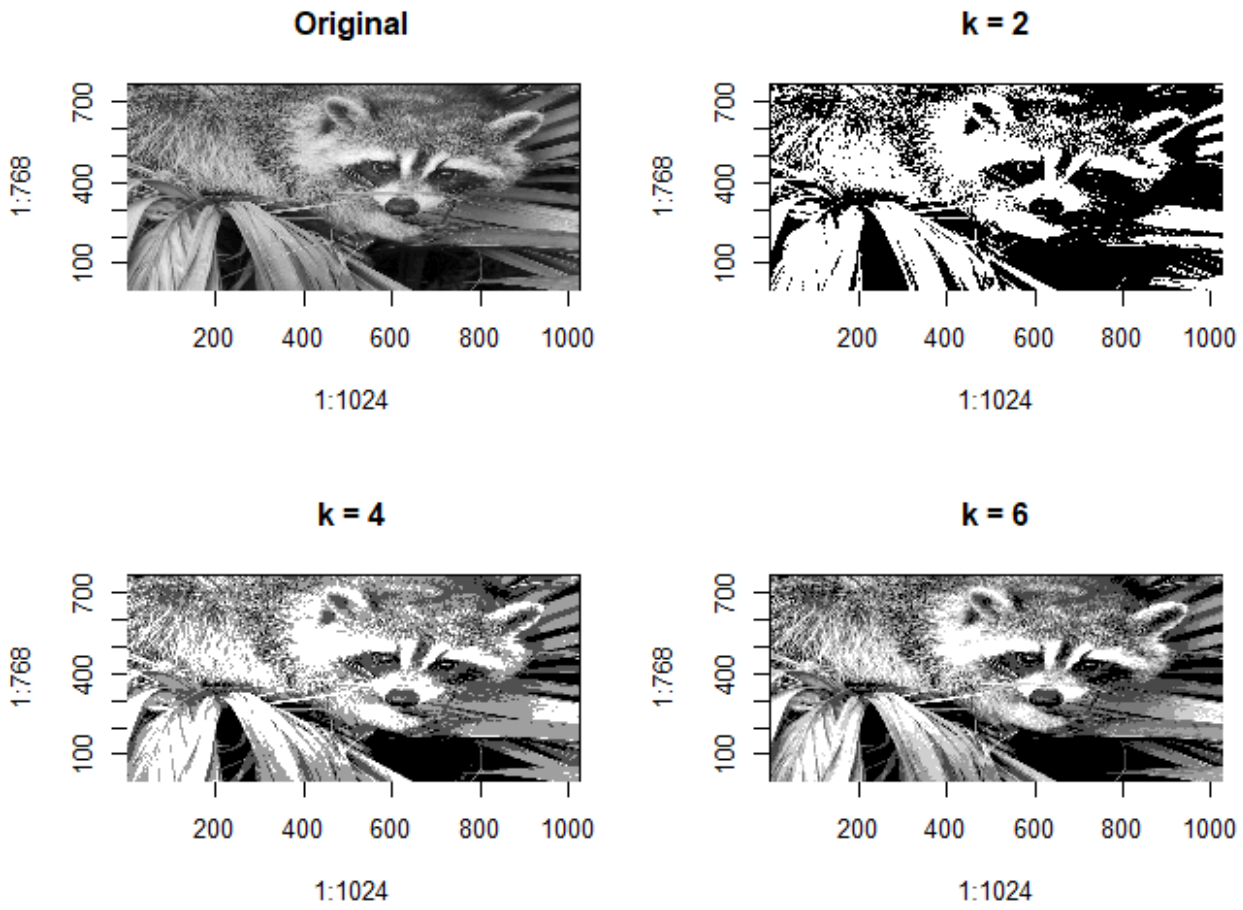

Figure 2: A black and white image of a raccoon.

```
library(png)
library(dplyr)

photo = readPNG("raccoon.png")

get_kmeans_centers = function(locs, k){
  # choose two random centers to start
  centers = sample(locs, k)

  counter = 1 # just curious how many iterations it takes to converge

  avgdiff = 1
  while(avgdiff > 0.01){
    # find distance
    dist = list()

    for (i in 1:k){
      center = centers[i]
      dist[paste0("group",i)] = list(sapply(locs, function(pixel) abs(pixel-center)))
```

```
    }

    dist_m = bind_rows(dist) # distance matrix
    clusters = apply(dist_m, 1, which.min) # cluster assignments

    new_locs = as.data.frame(cbind(locs, clusters))
    new_centers = tapply(new_locs$locs, new_locs$clusters, mean)

    avgdiff = sum(abs(centers-new_centers))/k

    locs = new_locs$locs
    centers = new_centers

    counter = counter + 1
  }

  res = list("cluster" = clusters, "centers" = centers)

  return (res) # returns a list of group assignments and mean for each group
}

compress = function(k, data){
  v = as.vector(data)
  km = get_kmeans_centers(v, k)

  # quantization
  compressed_m = matrix(km$cluster, ncol = dim(data)[2])
  book = km$centers

  print (book*256) # print the centers in terms of 256 scale for reference

  # plotting
  compressed_v = as.vector(compressed_m)
  new_v = rep(0,length(v))
  for (i in 1:length(v)){
    new_v[i] = book[compressed_v[i]]
  }
  new_m = matrix(new_v, ncol = dim(photo)[2])

  # rotating clockwise (R plots this weird)
  new_m = apply(new_m, 2, rev)
  image(1:1024, 1:768, t(new_m), col = gray((0:32)/32), useRaster = TRUE)
}

par(mfrow = c(2,2))
original = readPNG("raccoon.png")
original = apply(original, 2, rev)
image(1:1024, 1:768, t(original), col = gray((0:32)/32), useRaster = TRUE)
title(main = "Original")

# testing k = 2
compress(2, photo); title(main = "k = 2")

# testing k = 4
compress(4, photo); title(main = "k = 4")

# testing k = 6
compress(6, photo); title(main = "k = 6")
```

kmeans_compression_example.R

4.3 Basketball Position Data Analysis

Spectral clustering can also be used with basketball position data. In fact, using position information from basketball players, it is possible to use spectral clustering to develop a rough classification scheme for different plays.

References

[KK10] Amit Kumar and Ravindran Kannan. Clustering with spectral norm and the k-means algorithm. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 299–308, Washington, DC, USA, 2010. IEEE Computer Society.