Classifying Handwritten Digits Using EM and PCA

Josh Hancock

2017-06-03

In this post, we'll take the well-know Semeion Handwritten Digits dataset (http://archive.ics.uci.edu/ml/datasets/semeion+handwritten+digit) and cluster the handwritten digits data using the EM algorithm with a principle components step within each maximization.

First, we'll read in the data, load the additonal libraries, and create our initial data table.

```
library("mvtnorm")
library("data.table")
#Reading data and convert to data table
setwd("C:/Users/Josh/Documents/GitHub/joshuahancock.github.io/data_sets/")
data <- read.csv("semeion.csv", header = FALSE)
x <- data.table(data[,1:256])
```

Each row of the data represents one handwritten digit, which were digitally scanned and stretched into a 16x16 pixel box. Each of these 256 pixels, orignally in greyscale, was thresholded into a binary value that indicates 'black' or 'white' for that pixel. There are 10 addtional columns (also binary), which indicate group membership. We'll need to separate those labels into their own data table.

```
labels <-apply(data[,257:266],1,function(xx){ return(which(xx=="1")-1)})
```

Before we start clustering, we need to take care of a few global variables and run our initial clustering algorithm.

```
# k is the number of clusters
k <- 10
# n is the number of observations
n <- nrow(x)
# d is the number of diwmensions
d <- ncol(x)
# q represents the number of principal components and will need to be manually changed
q <- 0
# x.clusters are the clusters using k means and 100 random starts
x.clusters <- kmeans(x, k, nstart = 100)
```

Now that we have preliminary clusters, we'll initialize our ?? matrix, which will hold the cluster membership probabilities for each observation. We then use ?? to calculate ??k, the proportion of observations assigned to cluster k, and µk, the mean of the observations within each cluster k.

```
# n by k matrix, initialized with zeros
gamma <- matrix(0,n,k)
# indicate the inital custer membership with a binary label
for(i in 1:n) {gamma[i,x.clusters$cluster[i]]=1}
# the number of members in each cluster
N <- colSums(gamma)
# the percentage of the data set in cluster k
piHat <- N/n
# the mean for each pixel in each cluster
# note: a matrix is required for the t() function
muHat <- (t(gamma) %*% data.matrix(x))/ N
```