Homework7

Jun Rao

9/21/2020

Dirichlet Process Mixture Models The goal of this homework is to use the R package dirichletprocess to implement an infinite dimensional Gaussian mixture model, then compare it to finite Gaussian mixtures with model selection via cross-validation (mclust).

1.First use rnorm to generate random data from 5 cluster centers, as we did in class last week (use 20 data points per cluster center). Scale the data set.

library(ggplot2)  
library(data.table)  
library(mclust)  
library(dirichletprocess)  
N.true.clusters <- 5  
true.means <- 1:N.true.clusters  
N.simulated.data <- 20\*N.true.clusters  
means.simulated.data <- rep(true.means, each=N.simulated.data/N.true.clusters)  
  
# set.seed(1)  
true.sd <- 0.2  
sim.data.vec <- rnorm(N.simulated.data, means.simulated.data, sd=true.sd)  
  
## linear transformation on each column so that mean=0 and sd=1.  
df <- scale(sim.data.vec)

Then use dirichletprocess::DirichletProcessGaussian to fit the mixture model, Use system.time on the model fitting process – how long does it take?

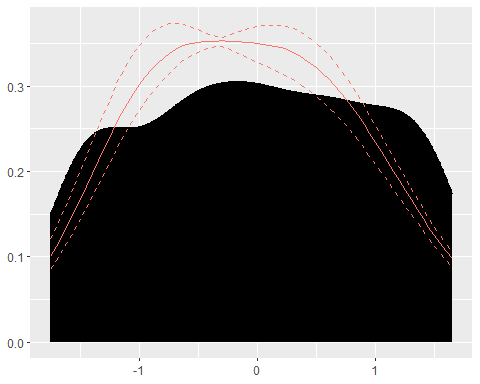
dp <- DirichletProcessGaussian(df)  
  
system.time(dp<- Fit(dp, its=500))#iterations of sampling algorithm.

## user system elapsed   
## 3.3 0.0 3.3

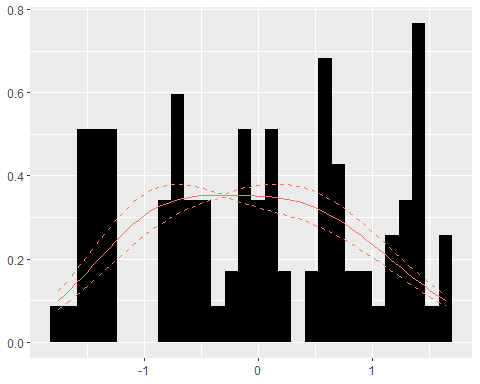
***It takes 3.3seconds***

then use the plot function to make a figure similar to Figure 1 (right) in vignette (“dirichletprocess”, package=”dirichletprocess”) (histogram for data, solid/dashed lines for the model).

plot(dp)



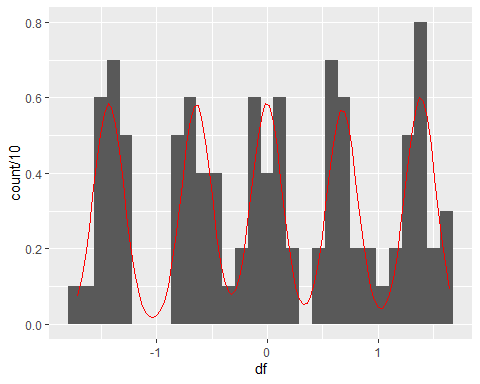
plot(dp, data\_method="hist")



2.Now use mclust to fit a model with 5 clusters to the scaled data. Make a plot similar to the previous one. Use geom\_bar or geom\_histogram to plot the data. Use the mclust::dens function to compute the mixture density to plot with geom\_line. Do the two models look similar?

#Now use mclust to fit a model with 5 clusters to the scaled data.  
fit <- Mclust(df,5)  
conc.vec <- seq(min(df),max(df),l=100)  
  
#Use the mclust::dens function to compute the mixture density  
Dens <- dens(modelName = fit$modelName,data = conc.vec,parameters = fit$parameters)  
  
density <- data.table(concentration = conc.vec,den = Dens)  
  
ggplot() + geom\_histogram(aes(x=df, y =stat(count) /10)) + geom\_line(aes(concentration,den),data=density,color='red')

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



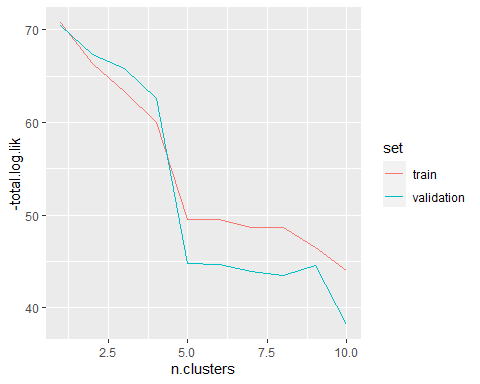
***These two models don’t simlier to each other***

3.Divide the data into 50% train, 50% validation. Use mclust to fit a mixture model with K=1 to 10 components on the train set, then compute and plot y=validation negative log likelihood as a function of x=number of components. Use system.time on the whole process (excluding the plot) – how much time does it take to do model selection yourself? Is that faster or slower than the Dirichlet Process mixture model from question?

set.prop.vec <- c(validation=0.5, train=0.5)  
N <- length(df)  
rounded.counts <- floor(set.prop.vec\*(N))  
not.shuffled.sets <- rep(names(set.prop.vec), rounded.counts)  
  
set.seed(1)  
shuffled.sets <- sample(not.shuffled.sets)  
clusters <- 10  
  
lik.dt.list <- list()  
  
  
system.time(  
 for(n.clusters in 1:clusters){  
 for (set in names(set.prop.vec)) {  
 data.set <- df[shuffled.sets == set,]  
 mclust <- Mclust(data.set, n.clusters, modelNames="E")  
 log.lik.vec <- dens(  
 modelName = mclust[["modelName"]],  
 data = data.set,  
 parameters = mclust[["parameters"]],  
 logarithm=TRUE)  
 total.log.lik <- sum(log.lik.vec)  
 rbind(my=total.log.lik, mclust=mclust[["loglik"]])  
 lik.dt.list[[paste(n.clusters, set)]] <- data.table(n.clusters, set, total.log.lik)  
 }  
  
}  
)

## user system elapsed   
## 0.11 0.00 0.11

lik.dt <- do.call(rbind, lik.dt.list)  
  
ggplot()+  
 geom\_line(aes(  
 n.clusters, -total.log.lik, color=set),  
 data=lik.dt)



It takes 0.11seconds. It is much faster than the Dirichlet Process mixture model