Assignment 4 code

```
library(mvtnorm)
library(plotly)
library(dplyr)

main <- function(){


  sample <- read.csv("/Users/utkarshjha/Downloads/kmeans.csv")  #/Users/utkarshjha/Downloads
  sample <- as.matrix(sample)

  # will ignore inf and NA while summation
  finitesum <- function(data) {
    sum(data[is.finite(data)])
  }
  mu<-kmeans(sample,3)
  mu<-mu$centers
  cat("initial means are\n")
  print(mu)
  # p1,p2,p3 are probabilities in which sample is divided
  p1 = 1/3
  p2 = 1/3
  p3 = 1/3
  mu1<-mu[1,] # initial means with kmeans
  mu2<-mu[2,]
  mu3<-mu[3,]

  # 1st,2nd and 3rd covariance is initialized
  cov1 = matrix(diag(5), byrow=T, ncol=5)
  cov1
  cov2 = matrix(diag(5), byrow=T, ncol=5)
  cov3 = matrix(diag(5), byrow=T, ncol=5)

  k = 2 #k will be used for iterations
  ta  = rep(0,3)
  q<- rep(NA, 1000) # this is log likelihood
  q[1]<-0
  q[2]<-0.1


  n  = nrow(sample)# total size of sample


  min = 0.00001# used to check convergence of log likelihood
  while(abs(q[k]-q[k-1]) >= min) {
    cat("\niteration no is  ",k-1)

    #Expectation step
    den = p1*dmvnorm(sample,mu1,cov1) + p2*dmvnorm(sample,mu2,cov2) +
p3*dmvnorm(sample,mu3,cov3)#calculating denominator for bayesian
    ta1<-(p1*dmvnorm(sample,mu1,cov1))/den # calculating using bayesian
    ta2<-(p2*dmvnorm(sample,mu2,cov2))/den
    ta3<-(p3*dmvnorm(sample,mu3,cov3))/den
    ta1[is.na(ta1)] <- 1/3
    ta2[is.na(ta2)] <- 1/3
    ta3[is.na(ta3)] <- 1/3
```

```
    #Maximiztion step

    cat("\n\ncluster1\n\n")
    p1 <-finitesum(ta1)/n

    mu1 = colSums(ta1*sample)/sum(ta1) # as per the mean formula
    cat("\nmean1\n")
    print(mu1)

    row_len = nrow(sample)
    mat_mu = matrix(rep(mu1,row_len),ncol  = row_len) # used to calculate difference between
sample and mean.Creating a matrix where every row is same as mean ,useful for subtracting

    mat_mu = t(mat_mu)
    temp_sample = (sample - mat_mu) # calculating x - mu

    sqr1 = sqrt(ta1)# we are taking square root of tau here because it will be equally distributed
between itself and transpose and when we multiply with transpose it will become tau
    temp_sample = sqr1*temp_sample
    cov1 = (t(temp_sample)%*%(temp_sample)) /finitesum(ta1) #cov = sum(ta*x-mu*t(x-mu))/
sum(ta)
    cat("\ncov1\n")
    print(cov1)


    cat("\n\ncluster2\n\n")
    p2 <-finitesum(ta2)/n
    mu2 = colSums(ta2*sample)/sum(ta2)
    cat("\nmean2\n")
    print(mu2)
    row_len = nrow(sample)
    mat_mu = matrix(rep(mu2,row_len),ncol  = row_len) # difference between sample and mean
    mat_mu = t(mat_mu) #
    temp_sample = (sample - mat_mu) # calculating x - mu

    sqr2 = sqrt(ta2)
    temp_sample = sqr2*temp_sample
    temp =  (t(temp_sample)%*%(temp_sample))

    cov2 = (t(temp_sample)%*%(temp_sample)) /finitesum(ta2)

    cat("cov2")
    print(cov2)


    cat("\n\ncluster3\n\n")
    cat("cov3")
    p3 <-finitesum(ta3)/n
    mu3 = colSums(ta3*sample)/sum(ta3)
    cat("\nmean3\n")

    print(mu3)
    row_len = nrow(sample)
    mat_mu = matrix(rep(mu3,row_len),ncol  = row_len) # difference between sample and mean
    mat_mu = t(mat_mu) #
    temp_sample = (sample - mat_mu) # calculating x - mu

    sqr3 = sqrt(ta3)
    temp_sample = sqr3*temp_sample
```

```
    temp =  (t(temp_sample)%*%(temp_sample))
    cov3 = (temp ) /finitesum(ta3)
    cat("cov3")
    print(cov3)



    cat("\nprobs are\n",p1,p2,p3)
    q[k+1] = finitesum(ta1*log(p1)+dmvnorm(sample,mu1,cov1,log = TRUE))
+finitesum(ta2*log(p2)+dmvnorm(sample,mu2,cov2,log = TRUE))
+finitesum(ta3*log(p3)+dmvnorm(sample,mu3,cov3,log = TRUE))
    k<-k+1
    cat("\n\ncurrent log likelihood is\n\n")
    print(q[k])#current log likelihood
    cat("\n\nprevious log likelihood is\n\n")
    print(q[k-1])#previous log likelihood
  }
  sample <- as.data.frame(sample)
  #dividing into clusters
  x = 0
  y =0
  z = 0

  for (i in 1:nrow(sample)){

    m = max(ta1[i],ta2[i],ta3[i])
    if(m == ta1[i]){
      sample$cluster[i]<- "Cluster1"
      x = x+1
    }
    else if(m == ta2[i]){
      sample$cluster[i]<- "Cluster2"
      y = y+1
    }
    else if(m == ta3[i] ){
      sample$cluster[i]<- "Cluster3"
      z = z+1
    }
  }

  cat("\ntotal elements in respective clusters are\n")
  print(x)
  print(y)
  print(z)

  z = sample[,1:3] #plotting 3 dimensions(a,b,c) for the random variable

  z$cluster = factor(sample$cluster)


  pl <- plot_ly(z, x=~a, y=~b,
                z=~c, color=~cluster) %>%
    add_markers(size=1.5)
  print(p)

  cat("\nFinal mean for cluster 1\n")
  print(mu1)
  cat("\nFinal mean for cluster 2\n")
  print(mu2)
  cat("\nFinal mean for cluster 3\n")
```

```
  print(mu3)

  cat("\nFinal covariance for cluster 1")
  print(cov1)
  cat("\nFinal covariance for cluster 2")
  print(cov2)
  cat("\nFinal covariance for cluster 3")
  print(cov3)

}
```