Hello, since my Rmarkdown could not knit the code in HTML I had to share my homework without knitting in (copying everthing in Rmds file and converting it to pdf using word). The error is called “pandoc document conversion failed with error 11” and according to web it is a random error –a gap in R that the company is trying to solve-. Please forgive me but the code works perfectly on console. I hope it would not create any problem for you :/

---

title: "Bonus Homework"

author: "Goksin Aydogan"

date: "28 AralÄ±k 2018"

output:

  html_document: default

```{r setup, include=TRUE}

knitr::opts_chunk$set(echo = TRUE)

rm()#This removes all objects in your working space

rm(list=ls()) #It removes everything in working space

require (rockchalk)

require (randomForest)

require (cluster)

require (arules)

require (Rfast)

myCov1 <- lazyCov(Rho = 0.4, Sd = 1.0, d = 8)

myCov2 <- lazyCov(Rho = 0.5, Sd = 1.0, d = 8)

myCov3 <- lazyCov(Rho = 0.6, Sd = 1.0, d = 8)

myCov4 <- lazyCov(Rho = 0.7, Sd = 1.0, d = 8)
```

```
set.seed(1)
D1 <- mvrnorm(n=500, mu = rep(0, 8), Sigma = myCov1)
D1 <- data.frame(D1)

set.seed(1)
D2 <- mvrnorm(n=500, mu = rep(5, 8), Sigma = myCov2)
D2 <- data.frame(D2)


set.seed(1)
D3 <- mvrnorm(n=500, mu = rep(10, 8), Sigma = myCov3)
D3 <- data.frame(D3)


set.seed(1)
D4 <- mvrnorm(n=500, mu = rep(20, 8), Sigma = myCov4)
D4 <- data.frame(D4)

data <- rbind(D1, D2, D3, D4)
data_syn <- data


data_syn$X1 = sample(data_syn$X1, replace=FALSE)
data_syn$X2 = sample(data_syn$X2, replace=FALSE)
data_syn$X3 = sample(data_syn$X3, replace=FALSE)
data_syn$X4 = sample(data_syn$X4, replace=FALSE)
```

```
data_syn$X5 = sample(data_syn$X5, replace=FALSE)

data_syn$X6 = sample(data_syn$X6, replace=FALSE)

data_syn$X7 = sample(data_syn$X7, replace=FALSE)

data_syn$X8 = sample(data_syn$X8, replace=FALSE)


data$class <- 1

data_syn$class <- 2


data <- rbind(data, data_syn)


data.proximity <- randomForest(data[,-9], data[,9], ntree=500, proximity=TRUE)$proximity


data.dissimilarity <- dist(sqrt(1-data.proximity[1:2000,1:2000]), method="euclidean")



method0 <- kmeans(data[1:2000,-9], 4, nstart=25)


method1 <- pam(data.dissimilarity, 4)


method2 <- hclust(data.dissimilarity, method="ward.D")

method2 <- cutree(method2, k=4)

method0$centers


for(i in 1:4){

  print(i)

  print(colMeans(data[which(method0$cluster==i),1:8]))

  print(cov(data[which(method0$cluster==i),1:8]))
```

```
  print(colMeans(data[which(method1$clustering==i),1:8]))
  print(cov(data[which(method1$clustering==i),1:8]))
  print(colMeans(data[which(method2==i),1:8]))
  print(cov(data[which(method2==i),1:8]))
}

data <- rbind(D1, D2, D3, D4)

data <- data+rbinom(2000,1,0.5)

data.proximity <- randomForest(data, ntree=500, proximity=TRUE)$proximity

data.dissimilarity <- dist(sqrt(1-data.proximity), method="euclidean")


method0 <- kmeans(data, 4, nstart=25)

method1 <- pam(data.dissimilarity, 4)

method2 <- hclust(data.dissimilarity, method="ward.D")
method2 <- cutree(method2, k=4)
method0$centers

for(i in 1:4){
  print(i)
  print(colMeans(data[which(method0$cluster==i),1:8]))
  print(cov(data[which(method0$cluster==i),1:8]))
```

```
  print(colMeans(data[which(method1$clustering==i),1:8]))

  print(cov(data[which(method1$clustering==i),1:8]))

  print(colMeans(data[which(method2==i),1:8]))

  print(cov(data[which(method2==i),1:8]))
}
```

As one can observe k-means works better than random forest methods, since it gives closer mean and covarience values; however when the noices are added, k-means falters whereas random forest results are not affected that much. This shows random forest method works better with noisy data than k-means, since k-means clustering is a naive method -no training only based on observation-. To conclude, when clustering gets complicated in terms of data complication, random forest models will work better. Hence, we can expect that as the dimension increases the model will work better than k-means.