```
####################################################################################
#####              This is for STAT3019 ICA 1 part1 - Clustering              #####
#####                                                                          #####
####################################################################################
library(MASS)
library(flexclust)
library(fpc)
library(pdfCluster)
library(mclust)
library(cluster)

############################################
#####  Step 1: Reading data            #####
############################################
dortmund <- read.table("Dortmund_G3019ica.dat",header=TRUE)
View(dortmund)
str(dortmund) # 170 obs. of 30 variables
summary(dortmund)
attach(dortmund)

buildings.years <- data.frame(buildings_until_1900,
buildings_1900.1918,buildings_1919.1948,buildings_1949.1957,
                            buildings_1958.1962,
buildings_1963.1972,buildings_1973.1982,buildings_1983.1992,
                            buildings_1993.2001)
age <- data.frame(age_under_26, age_26.35, age_36.45, age_46.55, age_56.65,
age_above_65)

############################################
#####  Step 2: Exploratory Analysis    #####
############################################
##### 2.1 Boxplots
quartz(width=10, height = 6)

## Births & death
boxplot(births, deaths, ylab="Number of inhabitants",
        main="Number of births and deaths in each district", xaxt="n")
axis(1, at=c(1,2), labels= c("births", "deaths"))
points(c(mean(births), mean(deaths)), col="red", pch=15)
legend("topright", col="red", pch=15, legend = c("mean"))
dev.copy(pdf,"Boxplot_birthdeath.pdf")
dev.off()
## Buildings
boxplot(buildings.years, xaxt="n",xlab="Years", ylab= "Numbers of buildings",
        main="Numbers of buildings by year of construction")
axis(1,at=c(1:9), labels=c(".-1900","1900-1918", "1919-1948", "1949-1957",
                           "1958-1962", "1963-1972","1973-1982", "1983-1992", "1993-
2001"))
points(apply(buildings.years,2,mean),col="red",pch=15)
legend("topright", col="red", pch=15, legend = c("mean"))
dev.copy(pdf,"Boxplot_buildingnum.pdf")
dev.off()
## cars; trucks; motorbikes
boxplot(cars, trucks,motorbikes, ylab="Number",
        main="Boxplots for numbers of transportations means in each district", xaxt="n")
axis(1, at=c(1:3), labels= c("cars", "trucks","motorbikes"))
points(c(mean(cars), mean(trucks), mean(motorbikes)), col="red", pch=15)
legend("topright", col="red", pch=15, legend = c("mean"))
dev.copy(pdf,"Boxplot_transportation.pdf")
dev.off()
## children; male; female;
boxplot(children, female, male, ylab="Number of inhabitants",
        main="Boxplots for number of inhabitants by gender in each district", xaxt="n")
axis(1, at=c(1:3), labels= c("children","female", "male"))
points(c(mean(children),mean(female), mean(male)), col="red", pch=15)
legend("topright", col="red", pch=15, legend = c("mean"))
dev.copy(pdf,"Boxplot_gender.pdf")
dev.off()
```

```
## moves_in; moves_out; households; benefits; social_insurance; unemployed
boxplot(moves_in, moves_out, households, benefits, social_insurance, unemployed,
        ylab="Number", main="Boxplots for number of the following factors in each
district", xaxt="n")
axis(1, at=c(1:6), labels= c("moves_in", "moves_out", "households", "benefits",
"social_insurance",
                             "unemployed"))
points(c(mean(moves_in),mean(moves_out), mean(households), mean(benefits),
         mean(social_insurance), mean(unemployed)), col="red", pch=15)
legend("topright", col="red", pch=15, legend = c("mean"))
dev.copy(pdf,"Boxplot_multiple_moves_social.pdf")
dev.off()
## age
boxplot(age, xaxt="n", xlab="Age group", ylab="Number of inhabitants",
        main="Boxplot for Number of inhabitants by age group in each district")
axis(1,at=c(1:6), labels=c("under26", "26-35", "36-45", "46-55", "56-65", "above65"))
points(apply(age,2,mean),col="red",pch=15)
legend("topright", col="red", pch=15, legend = c("mean"))
dev.copy(pdf,"Boxplot_agenum.pdf")
dev.off()
## area_buildings
boxplot(area_buildings, xlab="Area buildings", ylab="square metres",
        main="Boxplot of the area of buildings in each district")
points(mean(area_buildings), col="red",pch=15)
legend("topright", col="red", pch=15, legend = c("mean"))
dev.copy(pdf,"Boxplot_area.pdf")
dev.off()
par(mfrow=c(4,2))
## There are some outliers
dortmund[which.max(households),] # Scharnhorst-Ost
dortmund[which.max(social_insurance),] # Scharnhorst-Ost
dortmund[which.max(benefits),]# Scharnhorst-Ost
dortmund[which.max(births),] # Obereving
dortmund[which.max(buildings_1949.1957),] # MSA-Siedlung
dortmund[which.max(age_46.55),] # Scharnhorst-Ost
dortmund[which.max(moves_in),] # Marsbruchstrasse
dortmund[which.max(moves_out),]  # Marsbruchstrasse
dortmund[which.max(area_buildings),] # Scharnhorst-Ost
dortmund[which.max(cars),] # Scharnhorst-Ost
dortmund[which.max(trucks),] # City-West
dortmund[which.max(motorbikes),] #Oberdorstfeld
## Use order() to see the whole patterns
o <- list()
d <- list()
for(i in 1:30){
  o[[i]] <- order(dortmund[,i])
  d[[i]] <- dortmund[o[[i]],]
}
##### 2.2 Checking for collinearity and correlations/dissimilarities between variables
dortmund.cor <- data.frame(cor(dortmund, method = "spearman"))  # correlations
write.csv(dortmund.cor, file="dortmund_cor.csv")
0.5-dortmund.cor/2  #dissimilarities
abs(dortmund.cor)>=0.9  # Find out correlation that has absolute value greater than 0.9
0.5-dortmund.cor/2 <= 0.1

cor(age) # positively correlated in a quiet substential way, so we might want to reduce
the dimensions
pairs(age) # Prove the above results
dev.copy(pdf,"age_pairs.pdf")
dev.off()
cor(buildings.years) # Not significantly correlated
pairs(buildings.years) # looks like random scatters
dev.copy(pdf, "buildings_pairs.pdf")
dev.off()
pairs(data.frame(households, unemployed, benefits, births, children, deaths,
area_buildings, age_under_26, social_insurance))
pairs(data.frame(area_buildings, cars, male, trucks, female, social_insurance,
motorbikes))
```

```
par(mfrow=c(4,2))
plot(area_buildings, cars)
lines(lowess(area_buildings,cars), col='red', lwd=2)
plot(male,area_buildings)
lines(lowess(male,area_buildings), col='red', lwd=2)
plot(male, cars)
lines(lowess(male, cars), col='red', lwd=2)
plot(male,female)
lines(lowess(male,female), col='red', lwd=2)
plot(cars, motorbikes)
lines(lowess(cars,motorbikes), col='red', lwd=2)
plot(households, children)
lines(lowess(households,children), col='red', lwd=2)
plot(benefits~unemployed)
lines(lowess(benefits~unemployed),  col="red", lwd=2)
plot(moves_in, moves_out)
lines(lowess(moves_in, moves_out), col='red', lwd=2)
dev.copy(pdf, "variables_corplot.pdf")
dev.off()
par(mfrow=c(1,1))

##### 2.3 Treating the collinearities of variables
## As seen in 2.2 that there are lots of variables having high correlation,
## and some clearly linear patterns. To treat with these variables for reducing
dimensions and
## for deliminating replicable information for clustering, we here used several methods.
## (1) PCA: as age group has strong positive correlations between each other, we
introduced
## PCA and extracted the first two PCs of this group to indicate the data.
## (2) Aggregation:
## (3) Leave out some variables based on the high linear dependency.

## Firstly, creating a function for printing and visualizing pca results
Ref.pca <- function(x){
  pca.scale <- scale(x)
  test.pr <- prcomp(pca.scale)
  options(digits = 4)
  print(summary(test.pr))
  print(test.pr)
  screeplot(test.pr, type="lines")
}

Ref.pca(age) # The first two PCs has occupied 95.4%. So we take two PCs.
age.pca <- prcomp(age, scale. = TRUE)$x[,1:2]
Ref.pca(buildings.years) # Not working very well.
Ref.pca(data.frame(moves_in, moves_out)) # one for 0.975, good!
Ref.pca(data.frame(male,female)) # PC1 0.964
Ref.pca(data.frame(children, households, births)) # One for 0.955!
Ref.pca(data.frame(cars, motorbikes, trucks)) # two for 0.969... not as good as expected
Ref.pca(data.frame(cars, area_buildings)) # 0.974
Ref.pca(data.frame(unemployed, benefits)) # 0.974
Ref.pca(data.frame(social_insurance, area_buildings)) #0.986

## Then let's think about other ways to treat the buildings data - there are 9 of them!!
## i. try kmeans
buildings.cg<-clusGap(scale(t(buildings.years)), kmeans,K.max =
8,B=100,d.power=2,spaceH0="original",nstart=100)  # 8... not working
print(buildings.cg, method = "Tibs2001SEmax") #8
print(buildings.cg, method = "firstSEmax") #8
plot(buildings.cg)# Not working!

## ii. try hierarical clustering - still not working quiet well
eu.buil <- dist(scale(t(buildings.years)), method = "euclidean")
plot(hclust(eu.buil, method="average"))
plot(hclust(eu.buil, method="single"))
plot(hclust(eu.buil, method="complete"))

## iii. Dissimilarities
```

```
dis.buil <-0.5-cor(buildings.years)/2
dis.buil
mc <- cmdscale(dis.buil,k=2)
plot(mc,type="n")
text(mc,labels=dimnames(cor(buildings.years))[[2]])
dev.copy(pdf, "buil_mds.pdf")
dev.off()
# From the plot we can manually put them in 3 groups - looks very promising...
# k1: until 1900, 1900.1918
# k2: 1919.1948, 1949.1957, 1958.1962
# k3: 1963.1972, 1973.1982,  1983.1992, 1993.2001

## Aggregate them by suming up - do it in later construction of the new dataset
## Now we should think about whether to use pca or drop directly for some variables.
## First of all, as can be seen from the boxplots and the summary statistics that, the 
range
## of the different variables varies a lot! Except the fact that area_building is 
measured
## in square metres that distincts with others. So we need to scale the data.
## To protect some information from directly deleting some relavant but replicated 
variables,
## PCA seems like a better choice for us. Also if we use some variables for PCA value 
and some
## for original data, we need to scale it before doing clustering analysis.



dortmund.new <-  data.frame(age1=age.pca[,1], #age pc1
                           age2= age.pca[,2], # age pc2
                           moves= prcomp(data.frame(moves_in,moves_out), scale. = 
TRUE)$x[,1],  # move in and move out

buildings1=rowSums(data.frame(buildings_until_1900,buildings_1900.1918)),
                           buildings2=rowSums(data.frame(buildings_1919.1948, 
buildings_1949.1957, buildings_1958.1962)),
                           buildings3=rowSums(data.frame(buildings_1963.1972, 
buildings_1973.1982, buildings_1983.1992, buildings_1993.2001)),
                           chihousbirths= prcomp(data.frame(children, households, 
births), scale. = TRUE)$x[,1], # pc1
                           areasocial = prcomp(data.frame(area_buildings, 
social_insurance), scale. = TRUE)$x[,1], # cars deleted
                           benefits= prcomp(data.frame(unemployed, benefits), scale. = 
TRUE)$x[,1], # unemployed deleted
                           deaths=dortmund$deaths,
                           trucks=dortmund$trucks)


cor(dortmund.new, method="spearman")

Ref.pca(data.frame(children, households, births,area_buildings, social_insurance))

dortmund.new <-  data.frame(age1=age.pca[,1], #age pc1
                           age2= age.pca[,2], # age pc2
                           moves= prcomp(data.frame(moves_in,moves_out), scale. = 
TRUE)$x[,1],  # move in and move out

buildings1=rowSums(data.frame(buildings_until_1900,buildings_1900.1918)),
                           buildings2=rowSums(data.frame(buildings_1919.1948, 
buildings_1949.1957, buildings_1958.1962)),
                           buildings3=rowSums(data.frame(buildings_1963.1972, 
buildings_1973.1982, buildings_1983.1992, buildings_1993.2001)),
                           demon1= prcomp(data.frame(children, households, 
births,area_buildings, social_insurance), scale. = TRUE)$x[,1],
                           demon2=prcomp(data.frame(children, households, 
births,area_buildings, social_insurance), scale. = TRUE)$x[,2],
                           benefits= prcomp(data.frame(unemployed, benefits), scale. = 
TRUE)$x[,1],
                           deaths=dortmund$deaths,
                           trucks=dortmund$trucks) # cars, female and male are deleted
```

```
cor(dortmund.new)
write.table(dortmund.new, "dortmund_new.txt", sep="\t")
pairs(dortmund.new)
sdort.new<- scale(dortmund.new)
pairs(sdort.new)
cor(sdort.new)

cor(dortmund.new$age1, cars) #0.9774
cor(dortmund.new$age1, male) #0.9953
cor(dortmund.new$age1, female) # 0.9566
############################################
#####  Step 3: Clustering Analysis     #####
############################################
##### 3.1 Kmeans
set.seed(123456)
cg.new <- clusGap(sdort.new,
kmeans,30,iter.max=20,B=100,d.power=2,spaceH0="original",nstart=100)
print(cg.new,method="Tibs2001SEmax")
plot(cg.new)
dev.copy(pdf, "clusGap_final.pdf")
dev.off()

plot(1:30,cg.new$Tab[,1],xlab="k",ylab="log_Sk",type="l",ylim=c(4.5,9))
points(1:30,cg.new$Tab[,2],xlab="k",ylab="log_Sk",type="l",lty=2)
legend(20,9,c("log_Sk in data","E(log_Sk)"),lty=1:2,cex=0.5,bty="n")
dev.copy(pdf, "logSk_Exp.pdf")
dev.off()

set.seed(123456)
sdort.k11 <- kmeans(sdort.new, centers=11, nstart=100) #K=11 is from the result of
clusGap
sdort.k11$size
pairs(sdort.new, col=sdort.k11$cluster, pch=clusym[sdort.k11$cluster])
#For K-means with 11 centers,
#find the districts that are in the clusters which size is 1 and size is 2 respectively.
size1 <- which(sdort.k11$size==1)
k11cluster <- sdort.k11$cluster
dortmund[which(k11cluster==size1),]

size2 <- which(sdort.k11$size==2)
k11cluster <- sdort.k11$cluster
dortmund[which(k11cluster==size2),]
#Find the largest K which clusters' sizes are all not equal to 1.
size <- list()
for(i in 2:20){
  size[[i]] <- kmeans(sdort.new,centers = i,nstart = 100)$size
}

sdort.k4 <- kmeans(sdort.new, centers = 4, nstart = 100) #the largest K
sdort.k4$size
pairs(sdort.new, col=sdort.k4$cluster, pch=clusym[sdort.k4$cluster])
##### Average Silhouette Width function
eusdort <- dist(sdort.new, method = "euclidean")
asw <- function(model){
summary(silhouette(model, dist = eusdort))$avg.width}
asw(sdort.k11$cluster)#0.2539
asw(sdort.k4$cluster)#0.2496

##### 3.2 Hierarchical
sdortsing <- hclust(eusdort, method = "single")
plot(sdortsing)
sdortsing11 <- cutree(sdortsing, 11)
sdortsing4 <- cutree(sdortsing, 4)
asw(sdortsing11)
asw(sdortsing4)
sdortcomp <- hclust(eusdort, method = "complete")
plot(sdortcomp)
sdortcomp11 <- cutree(sdortcomp, 11)
asw(sdortcomp11)
```

```r
sdortcomp4 <- cutree(sdortcomp, 4)
asw(sdortcomp4)
sdortavg<- hclust(eusdort, method="average")
plot(sdortavg)
sdortavg11 <- cutree(sdortavg, 11)
asw(sdortavg11)

asw.max <- function(model){
sdorth <- list()
aswh <- NA
maxh <- NA

for (k in 2:30){
  aswh[1]<- -1
  sdorth[[k]]<- cutree(model, k)
  aswh[k]<- asw(sdorth[[k]])
  maxh<-which.max(aswh)
}
return(maxh)
}

asw.max(sdortcomp) #2..
asw.max(sdortavg) #2...
asw.max(sdortsing) #2....  Hierarchy is hard

##### 3.3 PAM
mansdort <- dist(sdort.new,method="manhattan")  #choose Manhattan distance for PAM
#find the number of clusters K using the criteria of ASW
pclust <- list()
psil <- list()
pasw <- NA
for(k in 2:20){
  pclust[[k]] <- pam(mansdort,k)
  psil[[k]] <- silhouette(pclust[[k]],dist=mansdort)
  pasw[[k]] <- summary(psil[[k]])$avg.width
}  #k=7 with asw=0.2764
plot(1:20,pasw,type="l",xlab="Number of clusters",ylab="ASW")

sdort.p7 <- pam(mansdort,7)
dortmund$pam7 <- sdort.p7$clustering #the clustering result for 170 observations

plot(sdort.p7)
pairs(sdort.new, col=sdort.p7$clustering, pch=clusym[sdort.p7$clustering])
sdort.p7$clusinfo

##### 3.4 mixture models
dortmc <- Mclust(dortmund.new, G=2:20)
summary(dortmc)  #k=4
pairs(dortmund.new, col=dortmc$classification, pch=clusym[dortmc$classification])
dev.copy(pdf, "mclust_4.pdf")
dev.off()

#### Comparing Average Silhouette Width for some methods uses Manhattan distance
summary(silhouette(sdort.k11$cluster, dist = mansdort))$avg.width #0.2688
summary(silhouette(sdort.k4$cluster, dist = mansdort))$avg.width  #0.2664
summary(silhouette(sdort.p7$clustering, dist = mansdort))$avg.width #0.2764: the best
summary(silhouette(dortmc$classification, dist = mansdort))$avg.width #0.0461: the worst

##### 3.5 MDS plots
mdsdortmund <- cmdscale(mansdort,k=2)
par(mfrow=c(2,2))
#K=4,11(from kmeans)
kclust4 <- kmeans(sdort.new,centers = 4,nstart = 100)
kclust11 <- kmeans(sdort.new,centers = 11,nstart = 100)
plot(mdsdortmund,col=kclust4$cluster,pch=clusym[kclust4$cluster],main="kmeans_k=4")
plot(mdsdortmund,col=kclust11$cluster,pch=clusym[kclust11$cluster],main="kmeans_k=11")
#K=4(from mixture)
plot(mdsdortmund,col=dortmc$classification,pch=clusym[dortmc$classification],main="mixture
 model")
```

```
#K=7 (from pam)
pclust7 <- pam(dman,7)
plot(mdsdortmund,col=pclust7$clustering,pch=clusym[pclust7$clustering],main="pam")
par(mfrow=c(1,1))


##### 3.6 Use original data to compare PAM7 and Kmeans11
pairs(age, col=sdort.k11$cluster, pch=clusym[sdort.k11$cluster])
pairs(age, col=sdort.p7$cluster, pch=clusym[sdort.p7$cluster])

demo<- data.frame(unemployed, area_buildings, births, deaths, moves_in, moves_out,
social_insurance, benefits)
pairs(demo, col=sdort.k11$cluster, pch=clusym[sdort.k11$cluster])
pairs(demo, col=sdort.p7$cluster, pch=clusym[sdort.p7$cluster])

pairs(buildings.years, col=sdort.k11$cluster, pch=clusym[sdort.k11$cluster])
pairs(buildings.years, col=sdort.p7$cluster, pch=clusym[sdort.p7$cluster])

popu<- data.frame(households, children, male, female, cars, trucks, motorbikes)
pairs(popu, col=sdort.k11$cluster, pch=clusym[sdort.k11$cluster])
pairs(popu, col=sdort.p7$cluster, pch=clusym[sdort.p7$cluster])
```