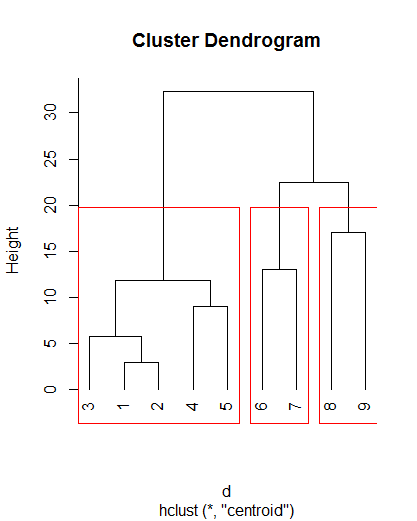
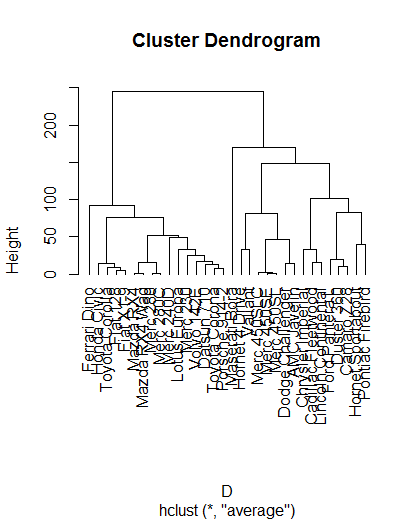
Sean Asaro

April 26 2016

Homework 7 stat 460

For this I would use 3 clusters. The coefficient is about 69%. The cluster structure is good.

1. I would use 2, maybe 3 clusters.



> model.cascade$results #CH indexes

2 groups 3 groups 4 groups

SSE 152473.37644 91261.22091 63247.62895

calinski 92.63271 84.52864 82.64205

5 groups 6 groups 7 groups 8 groups

SSE 42796.06522 34328.56341 26696.53652 19365.5639

calinski 91.55577 89.21192 93.11095 106.9188

9 groups 10 groups

SSE 12270.355 9245.882

calinski 143.161 162.338

Silhouette of 32 units in 3 clusters from silhouette.default(x = model.kmeans$cluster, dist = dist(mtcars[, from 1:7], method = "euclidean")) :

Cluster sizes and average silhouette widths:

16 7 9

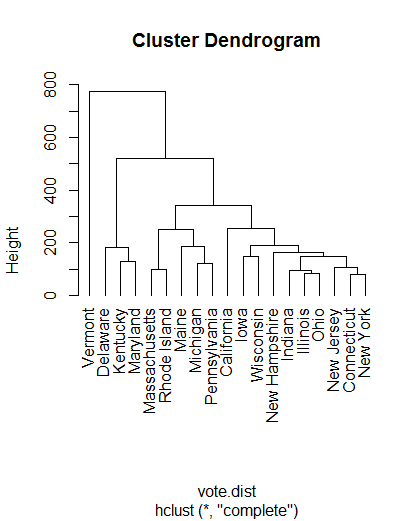
0.6358525 0.5702539 0.3496561

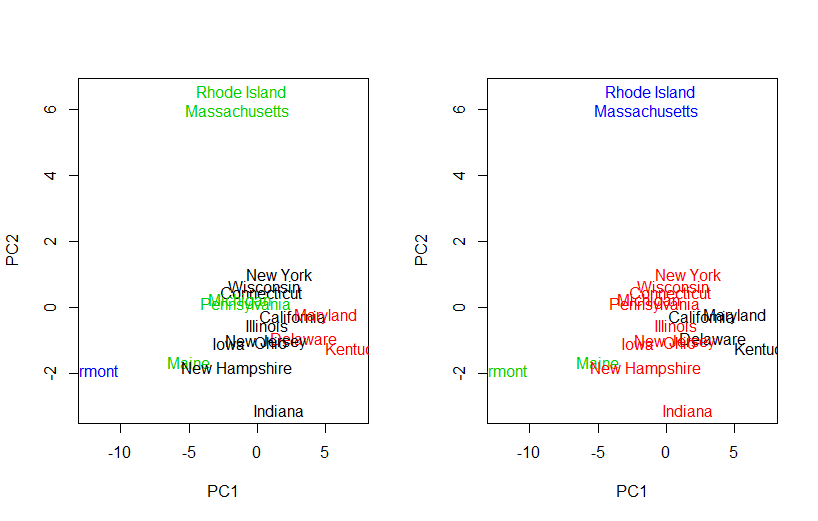
Individual silhouette widths:

Min. 1st Qu. Median Mean 3rd Qu. Max.

-0.04593 0.44960 0.57230 0.54100 0.69210 0.75180

1. Based on the dendrogram I would propose 4 clusters.





Based on the cascade CH index results I would pick 5 clusters.

CODE:

points <- c(1,4,9,16,25,36,49,64,81)

d=dist(points)

clust = hclust(d, method = 'centroid')

par(mfrow = c(1,1))

plot(clust, hang = -1)

rect.hclust(clust, k = 3, border = 2)

clust$height

clust$merge

par(mfrow = c(1, 1))

res = numeric(8)

for (K in 1:8)

{

res[K] = kmeans(d, centers = K, nstart = 1)$betweenss #W(K)

}

plot(res, type = 'b', pch = 20, ylab = 'B(K)', xlab = 'K')

require(cluster)

coef.hclust(clust) #69%

################ 2

head(mtcars[,1:7])

D = dist(mtcars[,1:7])

car.hclust = hclust(D, method = 'average')

par(mfrow = c(1,1))

plot(car.hclust, hang = -1)

model.kmeans = kmeans(mtcars[,1:7], centers = 3, nstart = 100) #basic algorith

round(model.kmeans$betweenss / model.kmeans$totss \* 100, 2) #percent of between clusters variavility in total vaiability

model.kmeans$cluster

nrow(mtcars)

par(mfrow = c(1, 1))

res = numeric(20)

for (K in 1:20)

{

res[K] = kmeans(mtcars[,1:7], centers = K, nstart = 1)$betweenss #W(K)

}

plot(res, type = 'b', pch = 20, ylab = 'B(K)', xlab = 'K')

res = numeric(20)

N = nrow(mtcars)

for (K in 2:20)

{

model.kmeans = kmeans(mtcars[,1:7], centers = K, nstart = 20) #W(K)

res[K] = (model.kmeans$betweenss / (K - 1)) / (model.kmeans$tot.withinss / (N - K))

}

plot(res, type = 'b', pch = 20, ylab = 'CH(K)', xlab = 'K')

which.max(res) #best K

library(vegan) #for cascadeKM command

model.cascade = cascadeKM(mtcars[,1:7], 2, 10)

model.cascade$results #CH indexes

plot(model.cascade)

library(cluster) #for silhouette command

sil.index = silhouette(model.kmeans$cluster, dist = dist(mtcars[,1:7], method = 'euclidean'))

summary(sil.index) #mean s(i) for each cluster

plot(sil.index) #no narrow silhouettes - good clustering

sil.index[,3] #s(i)

mean(sil.index[,3]) #mean s(i)

res = numeric(20)

dist.ED = dist(mtcars[,1:7], method = 'euclidean')

for (K in 2:20)

{

model.kmeans = kmeans(mtcars[,1:7], centers = K, nstart = 20)

res[K] = mean(silhouette(model.kmeans$cluster, dist = dist.ED)[,3])

}

plot(res, type = 'b', pch = 20, ylab = expression(bar(s)), xlab = 'K')

which.max(res) #best K (2)

######### 3

votes.repub

vote.c <- votes.repub[complete.cases(votes.repub),]

head(vote.c)

vote.dist <- dist(vote.c, method = "manhattan")

vote.clust <- hclust(vote.dist, method = "complete")

plot(vote.clust, hang = -1)

vote.pca <- prcomp(vote.c, scale = T)

cluster.vote.pca = hclust(dist(vote.pca$x), method = 'complete') #clustering on PCA's

par(mfrow = c(1, 2))

plot(vote.pca$x, pch = 20, col = cutree(vote.clust, k = 4), cex = 1.5, type = 'n') #groups from clustering on orginal data in PCA space

text(vote.pca$x, labels = rownames(vote.c), col = cutree(vote.clust, k = 4))

plot(vote.pca$x, pch = 20, col = cutree(cluster.vote.pca, k = 4), cex = 1.5, type = 'n') #groups from clustering on PC data in PCA space

text(vote.pca$x, labels = rownames(vote.c), col = cutree(cluster.vote.pca, k = 4))

?cascadeKM

vote.cascade = cascadeKM(vote.c, 2, 10)

vote.cascade$results

plot(vote.cascade)

res = numeric(20)

N = nrow(vote.c)

for (K in 2:20)

{

model.kmeans = kmeans(vote.c, centers = K, nstart = 20) #W(K)

res[K] = (model.kmeans$betweenss / (K - 1)) / (model.kmeans$tot.withinss / (N - K))

}

plot(res, type = 'b', pch = 20, ylab = 'CH(K)', xlab = 'K')

which.max(res) # 2