hw2

Benjamin Panny

2023-11-29

Theory 1. This question is modified from Question 1 in Chapter 10.7 in ISLR. This problem involves the K-means clustering algorithm.

Let (C_1, ..., C_K) be a clustering of n data points, the objective function of K-means is

$$\min_{(C_1,\ldots,C_K)} \sum_{k=1}^{K} WCSS(C_k) = \min (C_1,\ldots C_K)\left[\frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}\left(x_{ij}-x_{i'j}\right)^2\right]$$

C_k is the number of data points in C_k

$$= \min (C_1,\ldots C_K)\left[\frac{1}{|C_k|}\sum_{i,i'\in C_k}\|\mathbf{x}_i-\mathbf{x}_{i'}\|^2\right]$$

$$\mu_{kj} = \frac{1}{|C_k|}\sum_{i\in C_k} x_{ij},\ j=1,\ldots,p$$

$$\mu_k = \frac{1}{|C_k|}\sum_{i\in C_k}\mathbf{x}_i$$

Prove the within-cluster sum of squares WCSS(C_k) is equivalent to

$$\min_{(C_1,\ldots,C_K)} \sum_{k=1}^{K}\left[2\cdot\sum_{i\in C_k}\sum_{j=1}^{p}\left(x_{ij}-\mu_{kj}\right)^2\right],\ \min_{(C_1,\ldots,C_K)} \sum_{k=1}^{K}\left[2\cdot\sum_{i\in C_k}\|\mathbf{x}_i-\mu_k\|^2\right].$$

$$\sum_{k=1}^{K} 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} \left(x_{ij} - \mu_{kj}\right)^2 = 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} \left(x_{ij}^2 - 2x_{ij}\mu_{kj} + \mu_{kj}^2\right)$$

$$= 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 - 4 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}\mu_{kj} + 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} \mu_{kj}^2$$

$$= 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 - 4 \cdot \sum_{j=1}^{p} |C_k| \cdot \mu_{kj}^2 + 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} \mu_{kj}^2$$

$$= 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 - 4 \cdot \sum_{j=1}^{p} \mu_{kj}^2 \cdot |C_k| + 2 \cdot \sum_{j=1}^{p} \mu_{kj}^2 \cdot |C_k|$$

$$= 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 - 2 \cdot \sum_{j=1}^{p} \mu_{kj}^2 \cdot |C_k|$$

$$= 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 - 2 \sum_{j=1}^{p} \sum_{i \in C_k} x_{ij}\mu_{kj}$$

$$= 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 - 2 \sum_{j=1}^{p} \mu_{kj} \sum_{i \in C_k} x_{ij}$$

$$= 2 \cdot \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 - 2 \cdot \sum_{j=1}^{p} \frac{1}{|C_k|} \sum_{i' \in C_k} x_{i'j} \sum_{i \in C_k} x_{ij}$$

$$= \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 + \sum_{i \in C_k} \sum_{j=1}^{p} x_{ij}^2 - 2 \sum_{j=1}^{p} \frac{1}{|C_k|} \sum_{i' \in C_k} x_{i'j} \sum_{i \in C_k} x_{ij}$$

$$= \frac{1}{|C_k|} \sum_{i, i' \in C_k} \sum_{j=1}^{p} (x_{ij}^2 - 2x_{ij}x_{i'j} + x_{i'j}^2)$$

$$= \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i, i' \in C_k} \sum_{j=1}^{p} \left(x_{ij} - x_{i'j}\right)^2.$$

On the basis of the identity proved in (a), argue that the K-means (Lloyd's) clustering algorithm decreases the above objective function at each iteration.

1. Randomly assign 1 to K to each of the observations
2. Iterate until convergence.

a. For each of the K clusters, compute the cluster centroid. The kth cluster centroid is the vector of p feature means fro the observations in the kth cluster
b. Assign each observation to the cluster whose centroid is closest (by euclidean distance)

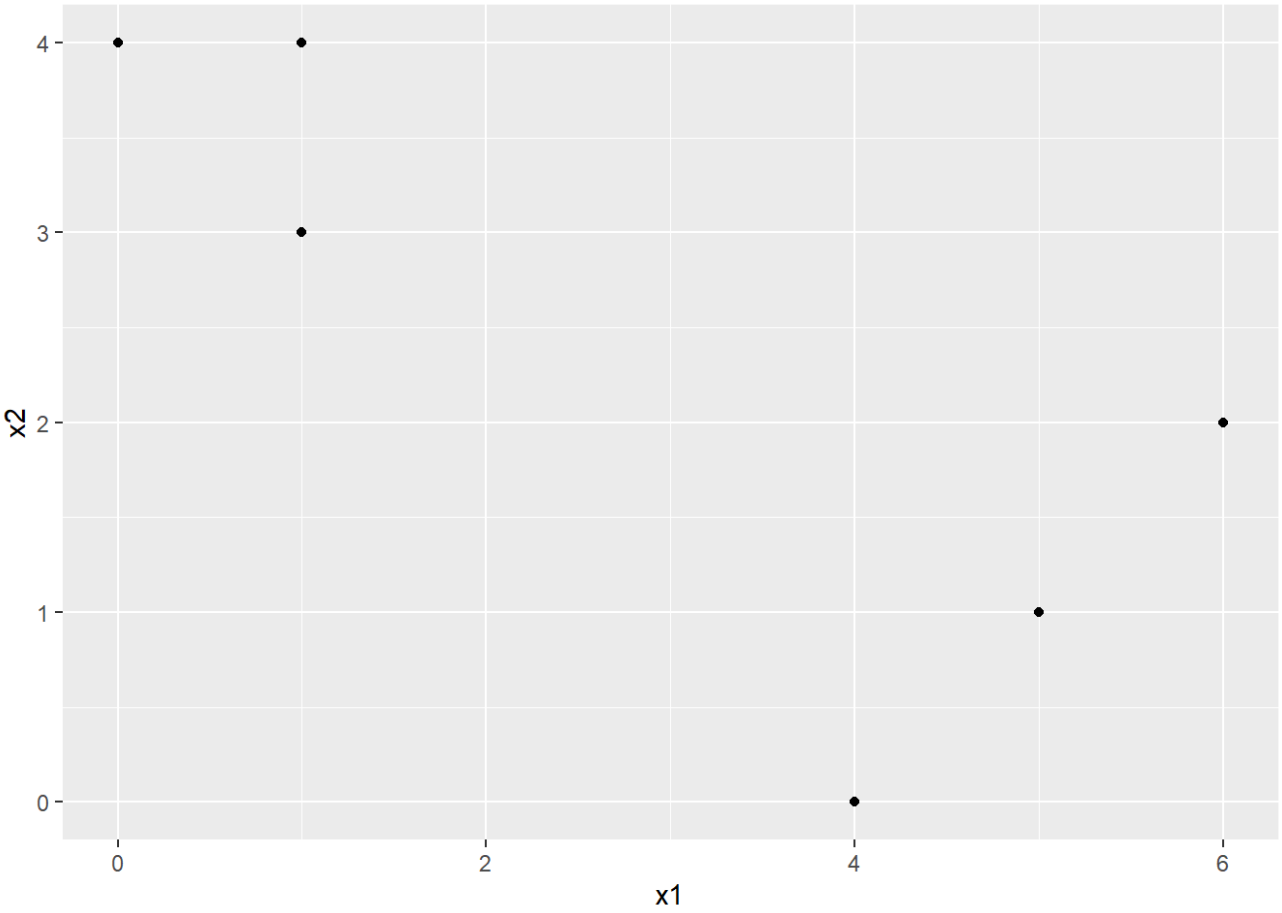
K-means clustering algorithm decreases the above objective function at each iteration because the algorithm minimizes the distance between the observations and the centroids at each step. The alternating calculation ensures that we calculate the centroids, then move such that distances are minimized, then calculate the centroids again. If we move no observations, then the centroids stay the same and we are done. If we move observations, we only move them in the direction that minimizes the objective (by 'moving' to a cluster whose centroid the observation is closer to). By the identity, this means we move points such that it is not only closer to the centroids but closer to each of the other observations in that centroid for that feature.

Question 3 in Chapter 10.7 in ISLR. In this problem, you will perform K-means clustering manually, with K = 2, on a small example with n = 6 observations and p = 2 features. The observations are as follows. (6 points)

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.2     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.2     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
q2 <- tibble(x1 = c(1,1,0,5,6,4), x2 = c(4,3,4,1,2,0))
q2 %>% ggplot(aes(x = x1, y = x2)) + geom_point()
```

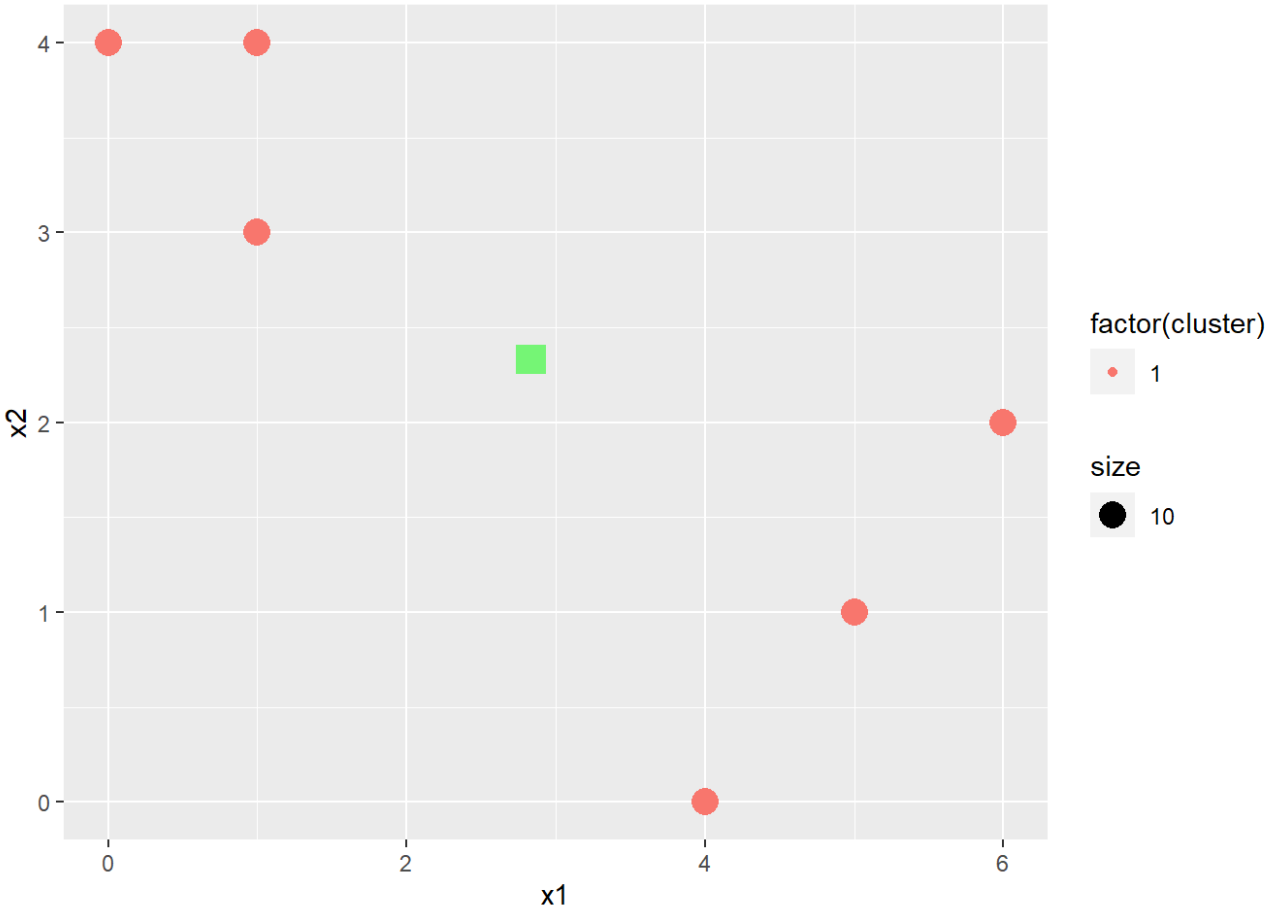



```
q2$initial_cluster <- sample(1:2, size = nrow(q2), replace = T, prob = c(.5,.5))
q2$cluster <- q2$initial_cluster
```

```
while(TRUE) {
  means <- q2 %>% group_by(cluster) %>% summarise(mean1 = mean(x1), mean2 = mean(x2))

  for (i in 1:nrow(q2)) {
    row <- q2[i, c('x1', 'x2')]
    distances <- (means$mean1 - row$x1)^2 + (means$mean2 - row$x2)^2
    q2$cluster[i] <- which.min(distances)
  }

  new_means <- q2 %>% group_by(cluster) %>% summarise(mean1 = mean(x1), mean2 = mean(x2))

  if (all(means$mean1 == new_means$mean1) && all(means$mean2 == new_means$mean2)) {
    break
  }

}
q2
```

```
## # A tibble: 6 × 4
##      x1    x2 initial_cluster cluster
##   <dbl> <dbl>           <int>   <int>
## 1     1     4               1       1
## 2     1     3               1       1
## 3     0     4               1       1
## 4     5     1               1       1
## 5     6     2               1       1
## 6     4     0               1       1
```

```
centroids <- q2 %>% group_by(cluster) %>% summarise(x1centroids = mean(x1), x2centroids = mean(x2))
q2 %>% ggplot(aes(x = x1, y = x2, shape = factor(cluster), color = factor(cluster), size = 10)) + geom_point()
+ geom_point(data = centroids, aes(x=x1centroids,y=x2centroids),shape = 'square', size = 5,color='green',inher
it.aes=F, alpha = .5)
```



with centroids as green squares.

This question is modified from Question 9 in Chapter 10.7 in ISLR. Consider the USArrests data set from the R package ISLR. Write a data analysis report to address the following problems. (9 points)

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 4.3.2
```

```
arrest <- USArrests
```

a. Apply PCA to the dataset. Plot the data in the first two PCs labeled by state names. Do the states appear to have any clustering patterns in the plot?

Principal component analysis (PCA)

The function `prcomp()` can be used to perform PCA.

```
pca <- prcomp(arrest, center = TRUE, scale. = TRUE)

# matrix of variable loadings
# head(pca$rotation)
dim(pca$rotation)
```

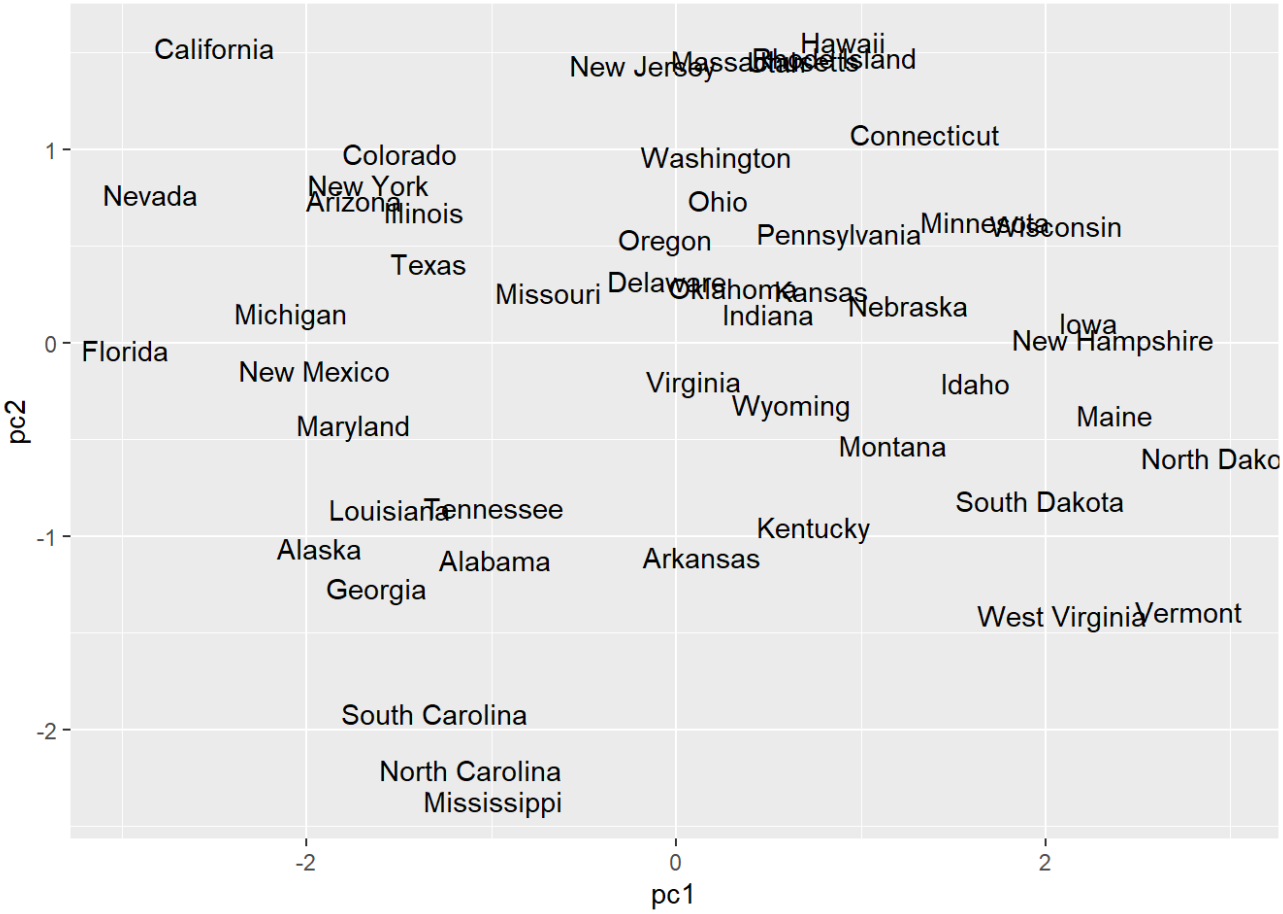
```
## [1] 4 4
```

```
# PCA demension 1
pca1 <- pca$x[,1]
# PCA demension 2
pca2 <- pca$x[,2]
```

Visualize PCA

Although class lables were not used in PCA analysis, the three classes are well-separated based on just the first two PCs.

```
pctbl <- tibble(pc1 = pca1, pc2 = pca2, state = rownames(arrest))
pctbl %>% ggplot(aes(x = pc1, y = pc2, label = state)) + geom_text()
```



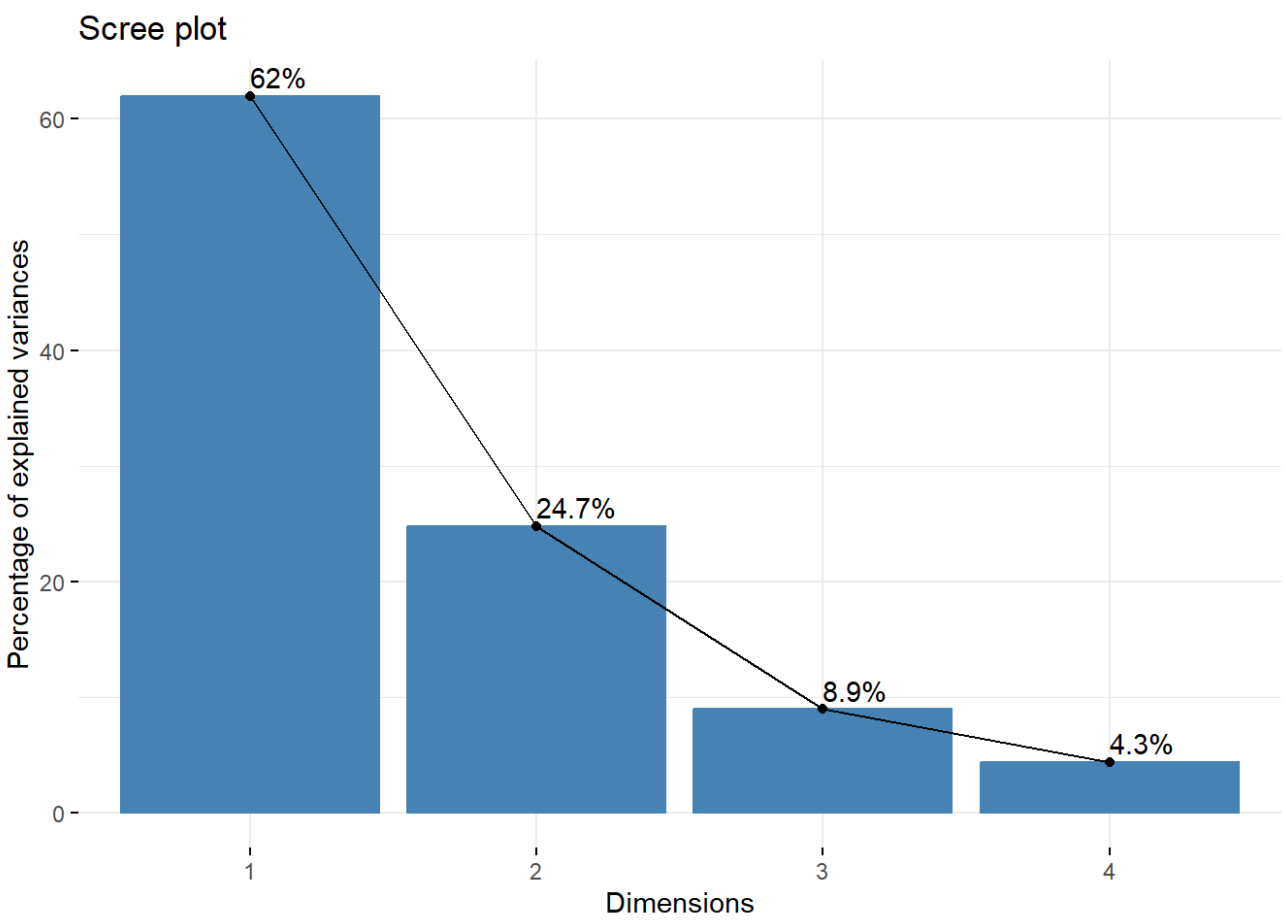
Yes it appears there is some segregation in the plot and also areas of the plot where there are lots of state names on top of one another.

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.3.2
```

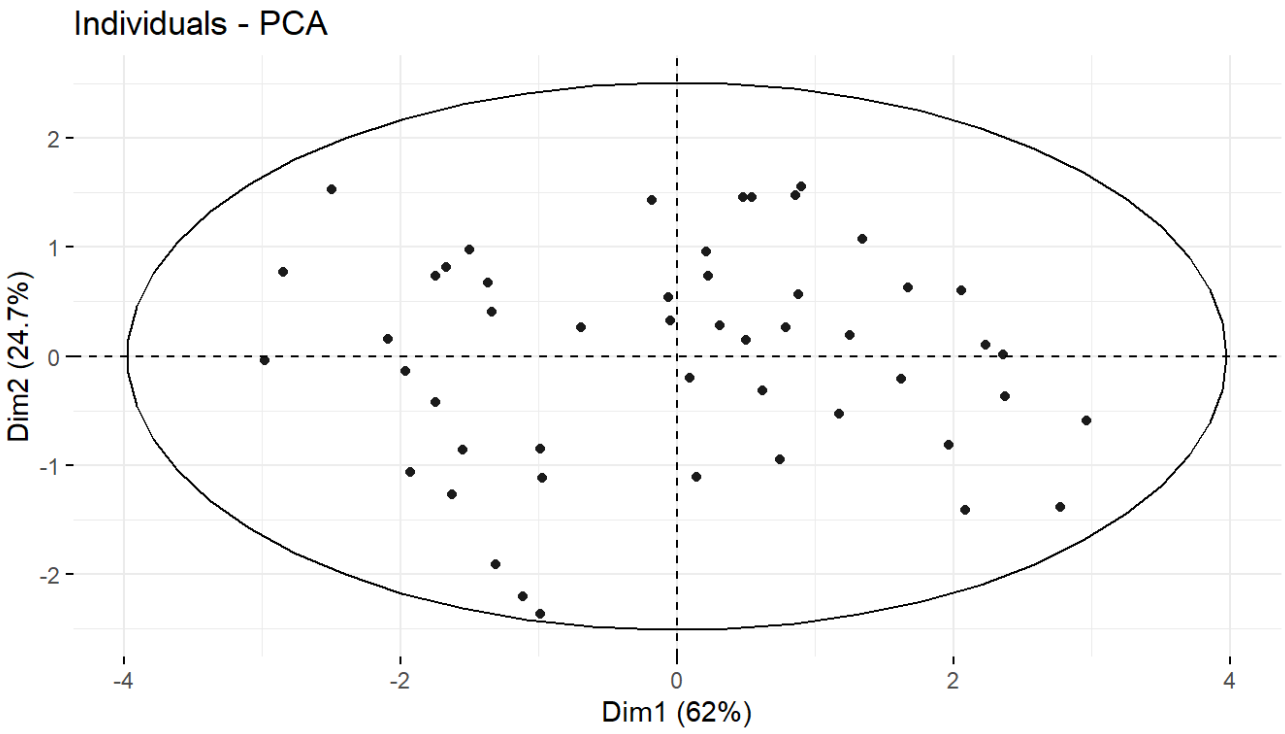
```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
fviz_eig(pca, addlabels = TRUE)
```



The function `fviz_pca_ind()` can be used to obtain the graph of individuals.

```
fviz_pca_ind(pca,
             label = "none",
             addEllipses = TRUE)
```



b. Run K-means with K from 1 to 6 and plot the associated within cluster sum of squares (WCSS). Note that when K = 1, WCSS is actually the total sum of squares (TSS) since there is no between cluster sum of square (i.e. BCSS = 0).

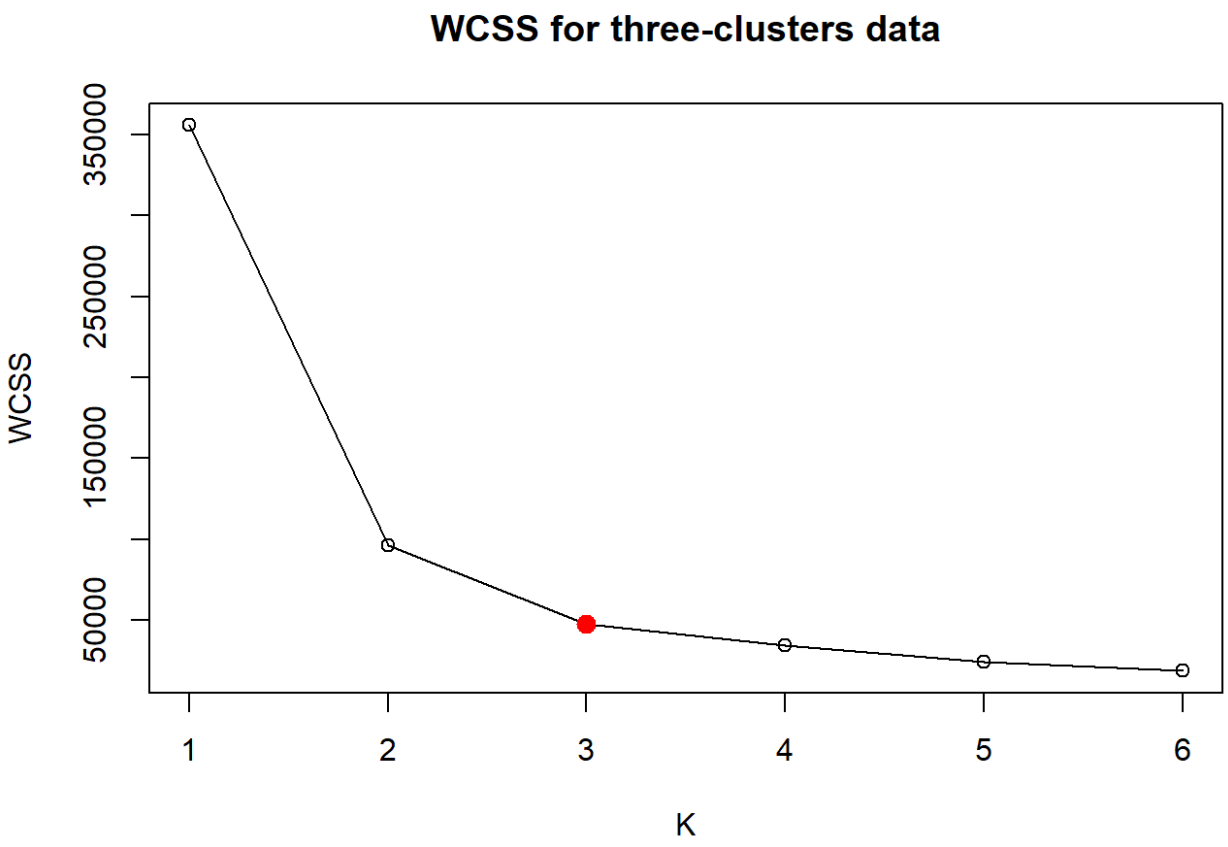
K means

Using the elbow method

You can use `kmeans()` function to perform K-means clustering.

```
# within-cluster sum of squares
WCSS<-rep(0,5)
# choose k:
for(k in 1:6){
  # Perform k-means clustering on a data matrix.
  res<-kmeans(x = arrest, centers = k, nstart = 100)
  # Total within-cluster sum of squares
  WCSS[k]<-res$tot.withinss
}
```

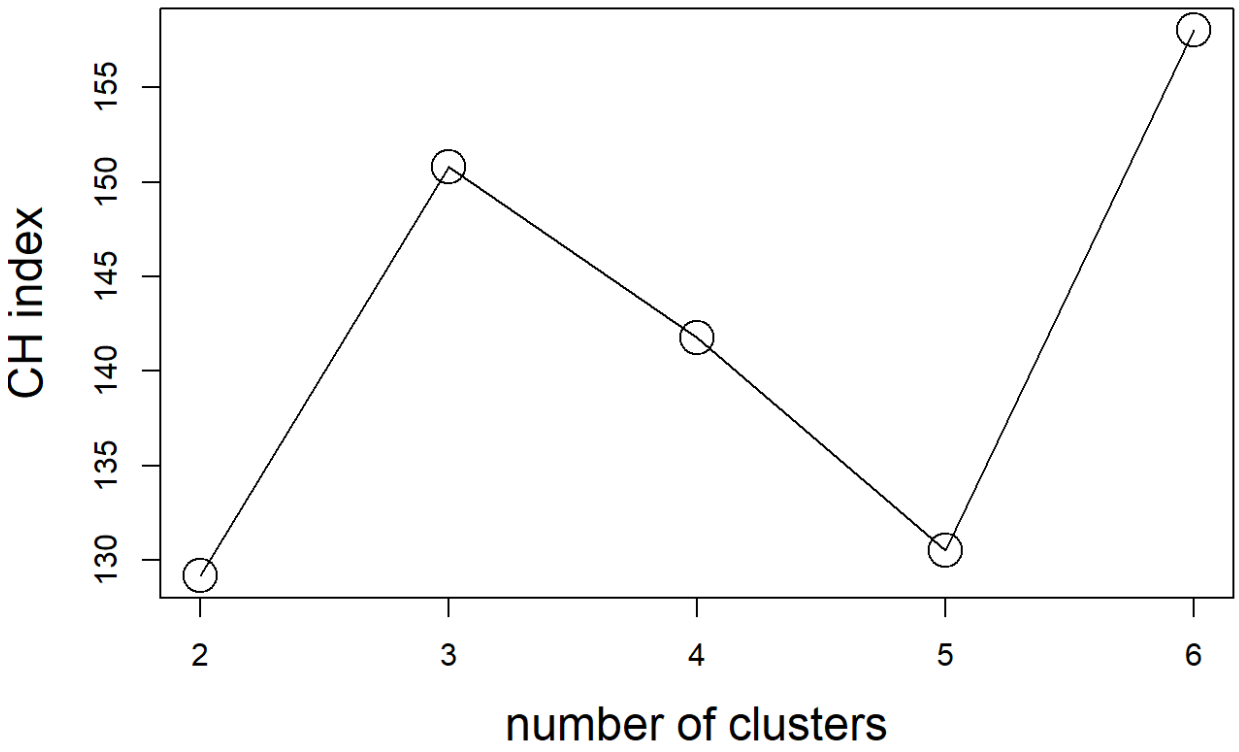
```
#plot WCSS vs different K values
plot(1:6, WCSS,
     type="o", col="black",
     xlab="K", ylab="WCSS")
title("WCSS for three-clusters data")
points(3, WCSS[3], col="red",cex=2,pch=20)
```



Looks like 3 clusters is optimal

We can use CH index, Hartigan index, KL index and Silhouette index to estimate K.

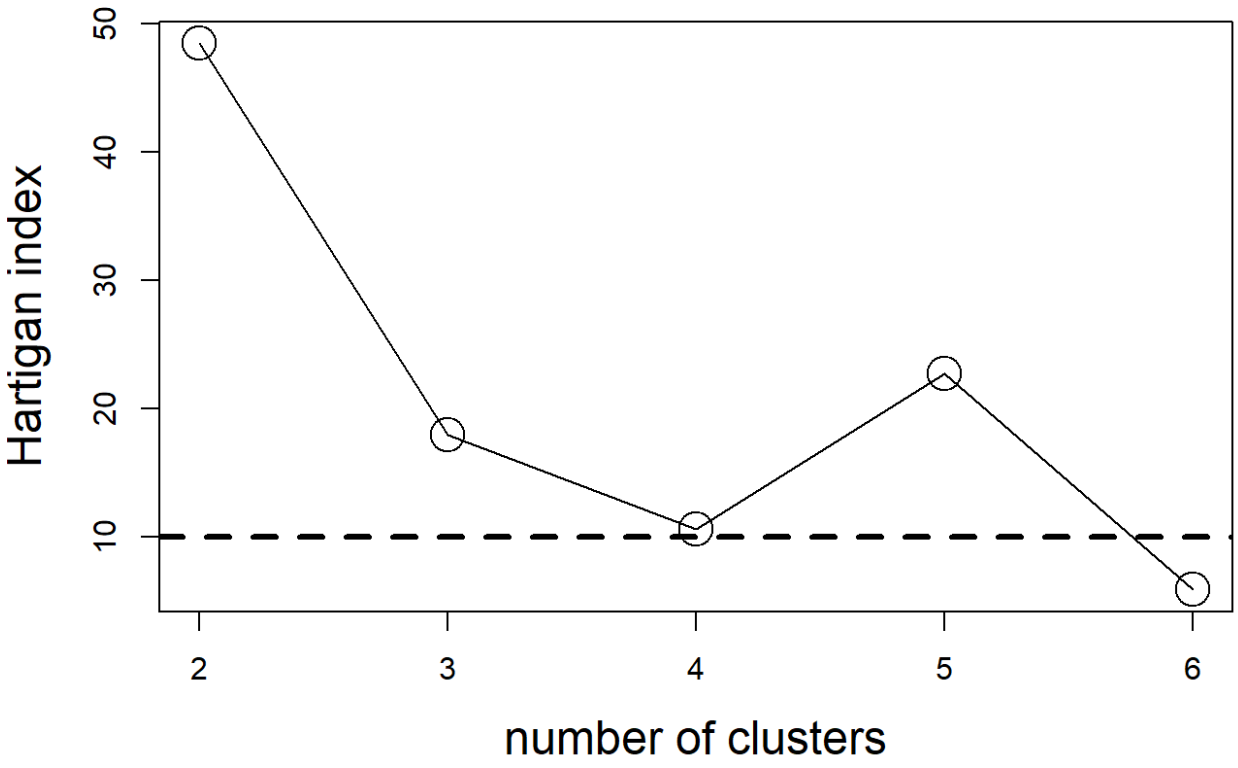
```
library(NbClust)
# We can use NbClust package to calculate the four indices
# CH index
CH <- NbClust(data=arrest, min.nc = 2, max.nc = 6,
              method="kmeans", index="ch")

plot(2:6, CH$All.index, type="o",
     xlab="number of clusters", ylab="CH index",
     "cex.lab"=1.5, cex=2.5)
```




```
CH$Best.nc[1]
```

```
## Number_clusters
##               6
```

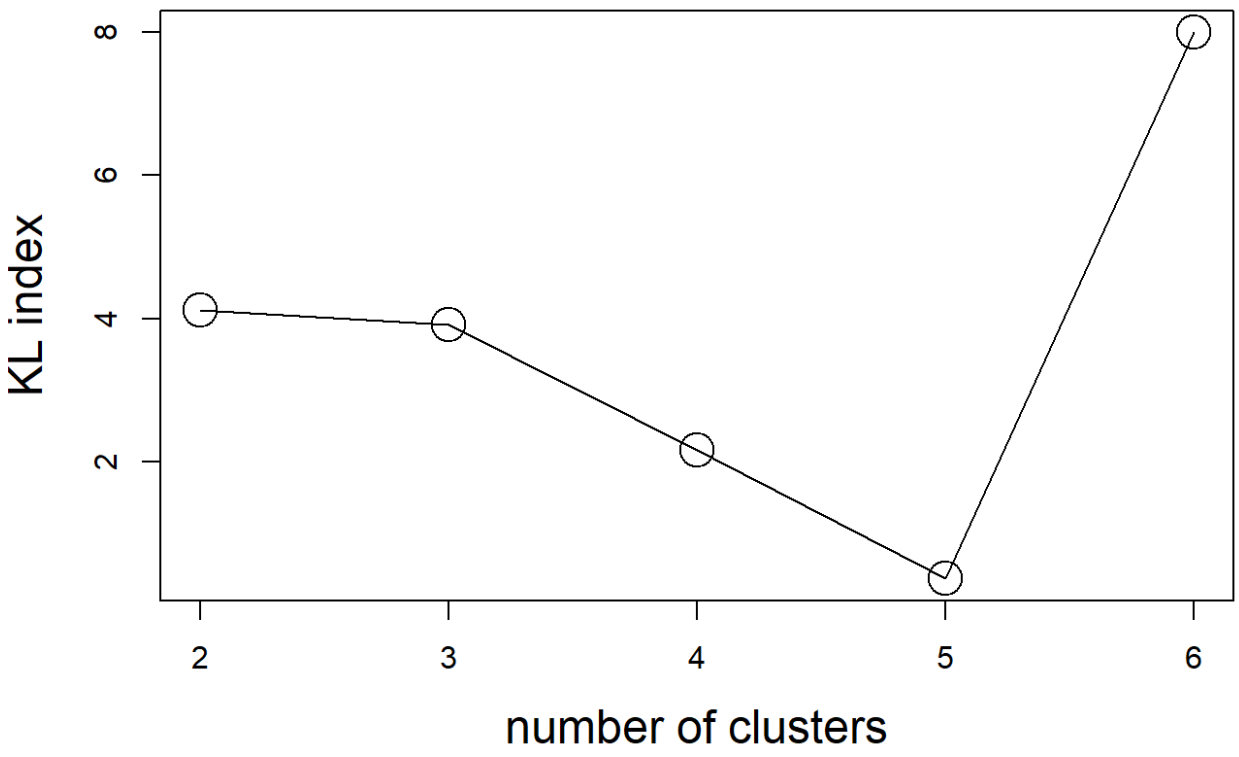
```
# CH$Best.partition  #we can also get Partition that
#### corresponds to the best number of clusters

# Hartigan index
H <- NbClust(data=arrest, min.nc = 2, max.nc = 6,
             method="kmeans", index="hartigan")

plot(2:6, H$All.index, type="o",
     xlab="number of clusters",ylab="Hartigan index",
     "cex.lab"=1.5, cex=2.5)
abline(h=10,lty=2,lwd=3) # stop when H(k) < 10
```




```
H$Best.nc[1]
```

```
## Number_clusters
##               3
```

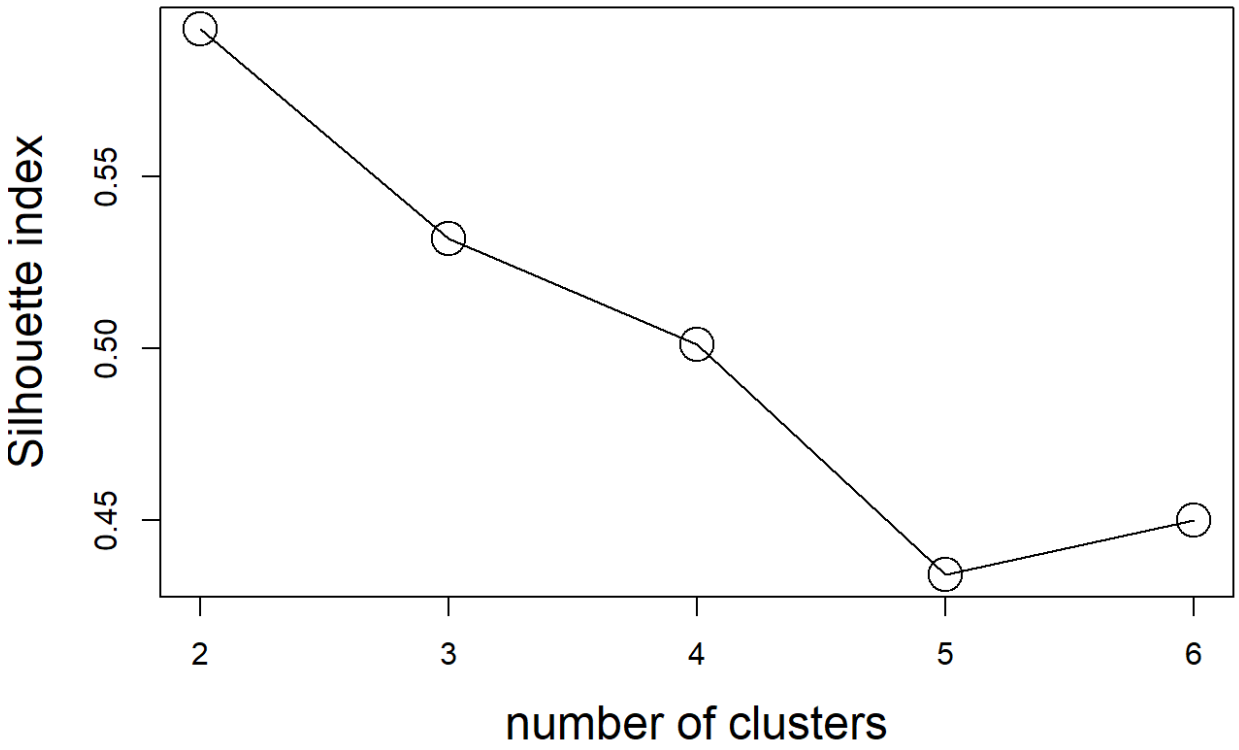
```
# KL index
KL <- NbClust(data=arrest, min.nc = 2, max.nc = 6,
              method="kmeans",index="kl")
plot(2:6, KL$All.index, type="o",
     xlab="number of clusters",ylab="KL index",
     "cex.lab"=1.5, cex=2.5)
```




```
KL$Best.nc[1]
```

```
## Number_clusters
##               6
```

```
# Silhouette index
Sil <- NbClust(data=arrest, min.nc = 2, max.nc = 6,
               method="kmeans", index="silhouette")
plot(2:6, Sil$All.index, type="o",
     xlab="number of clusters",ylab="Silhouette index",
     "cex.lab"=1.5, cex=2.5)
```



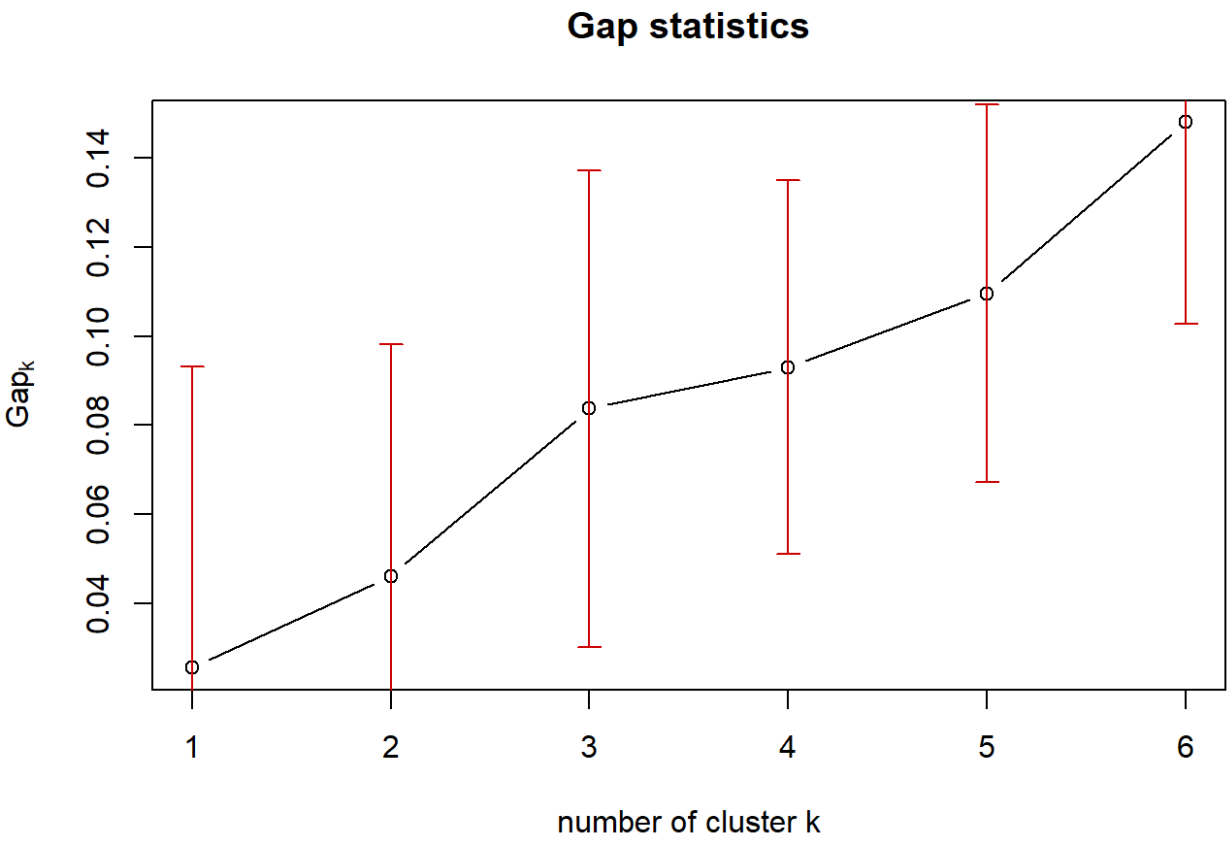

```
Sil$Best.nc[1]
```

```
## Number_clusters
##               2
```

Different indices indicate differing number of clusters as being best.

Gap statistics

```
library(cluster)
gapst <- clusGap(arrest, FUN = kmeans, nstart = 100,
                     K.max = 6, B = 50,
                     spaceH0 ="scaledPCA")
plot(gapst, xlab = "number of cluster k", main = "Gap statistics")
```




```
gaprs <- with(gapst,
                  maxSE(Tab[,"gap"],Tab[,"SE.sim"]))
gaprs
```

```
## [1] 5
```

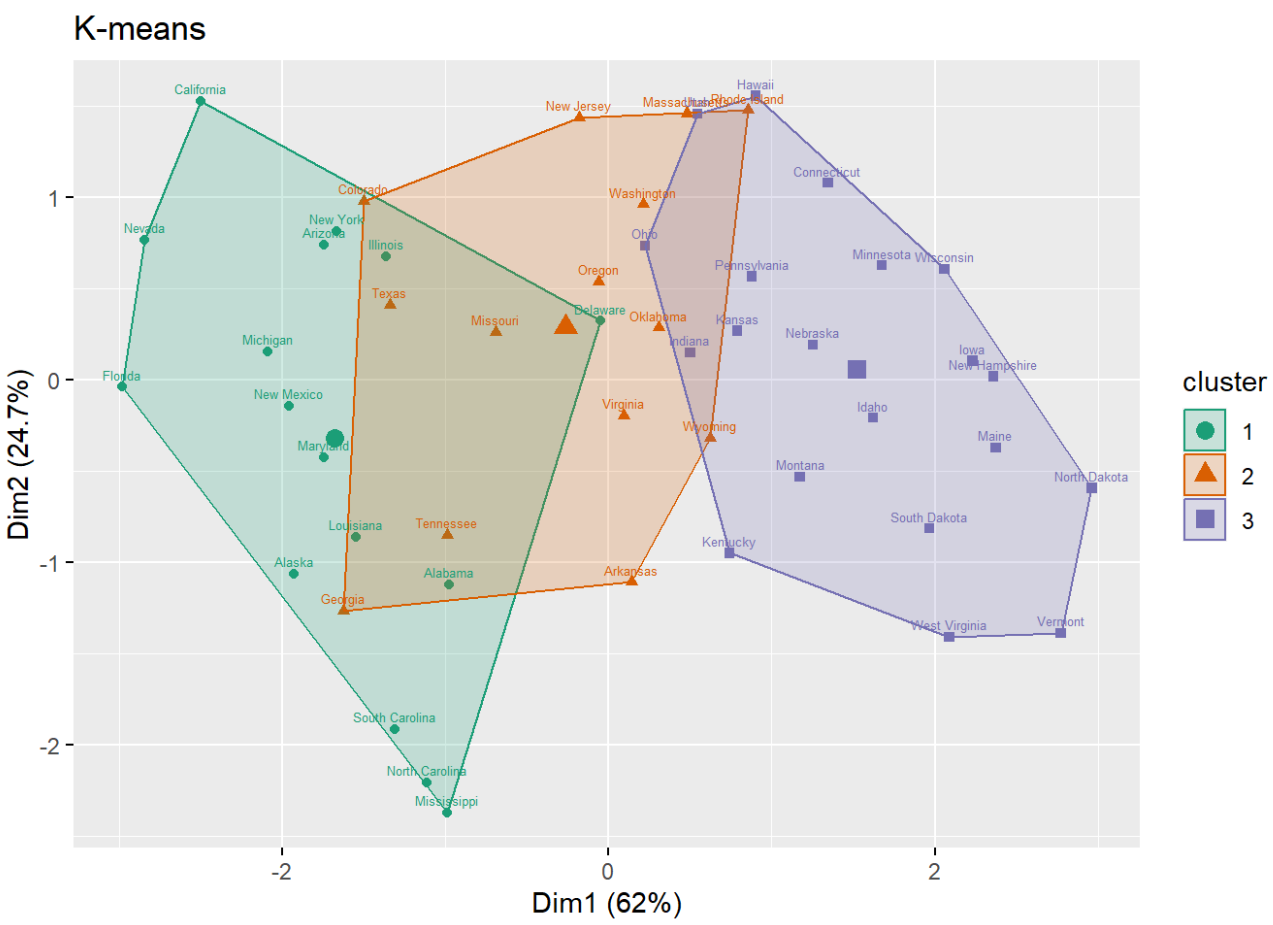
```
# gaprs <- maxSE(gapst$Tab[,"gap"],gapst$Tab[,"SE.sim"])
# print(gaprs, method = "firstmax")
```

The elbow, indices, and gap statistics yield differing K estimates.

c. Visualize your clustering for when K = 3 in the plot generated in (a).

K = 3 Visualization

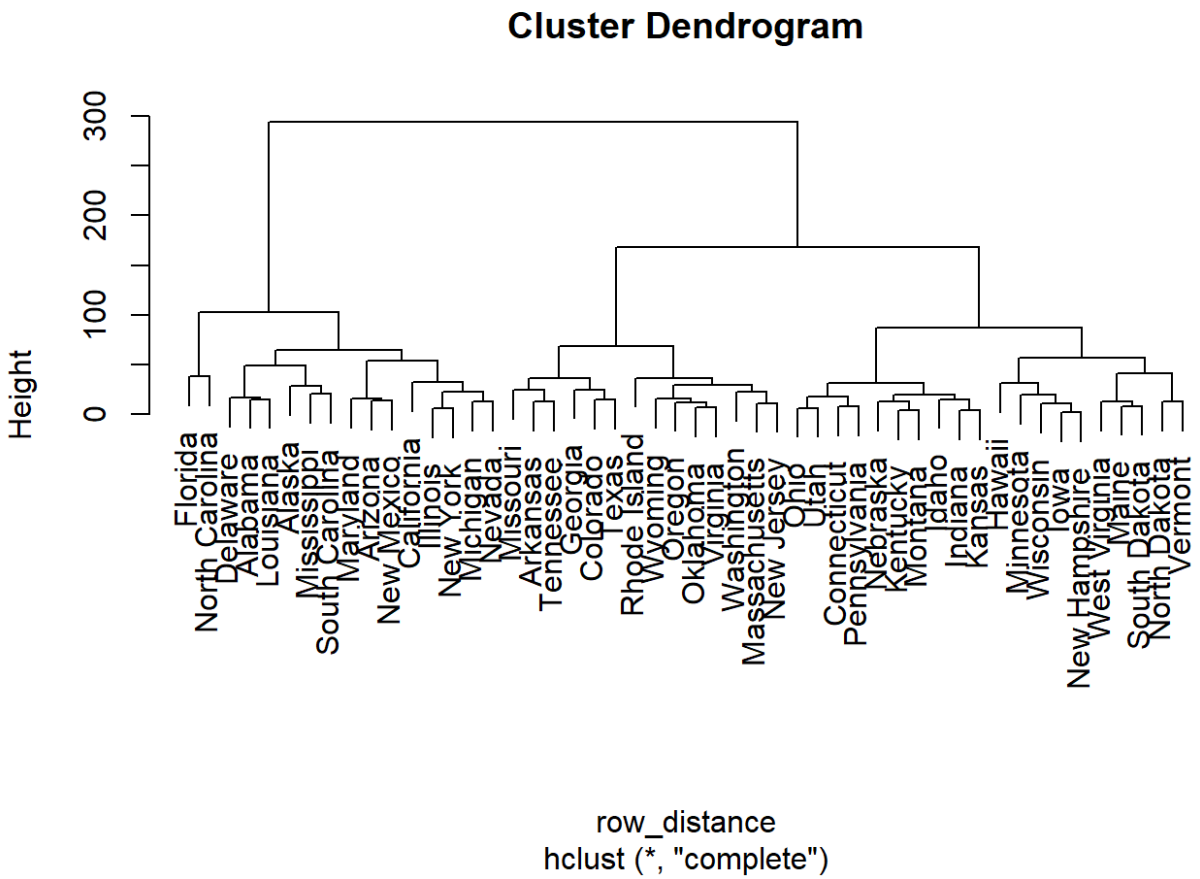
```
# Observations are represented by points in the plot, using principal components
km <- kmeans(arrest, centers = 3, nstart=10)
km_label <- km$cluster
km_vis <- fviz_cluster(list(data = arrest, cluster = km_label),
ellipse.type = "convex",
geom = c("point","text"),
labelsize = 5,
palette = "Dark2") + labs(title = "K-means")
km_vis
```



d. Now, using hierarchical clustering with complete linkage and Euclidean distance, cluster the states.

Hierarchical Clustering

```
row_distance = dist(arrest, method = "euclidean")
# try different linkage
hc.complete <- hclust(d = row_distance, method = "complete")
plot(hc.complete)
```



e. Cut the dendrogram at a height that results in three distinct clusters. Which states belong to which clusters?

```
hc_label<-cutree(hc.complete, k = 3)
# hc_label<-cutree(hc.complete, k = 3)
# hc_label<-cutree(hc.average, k = 3)
# hc_label<-cutree(hc.single, k = 3)
# hc_label<-cutree(hc.centroid, k = 3)
hc_label
```

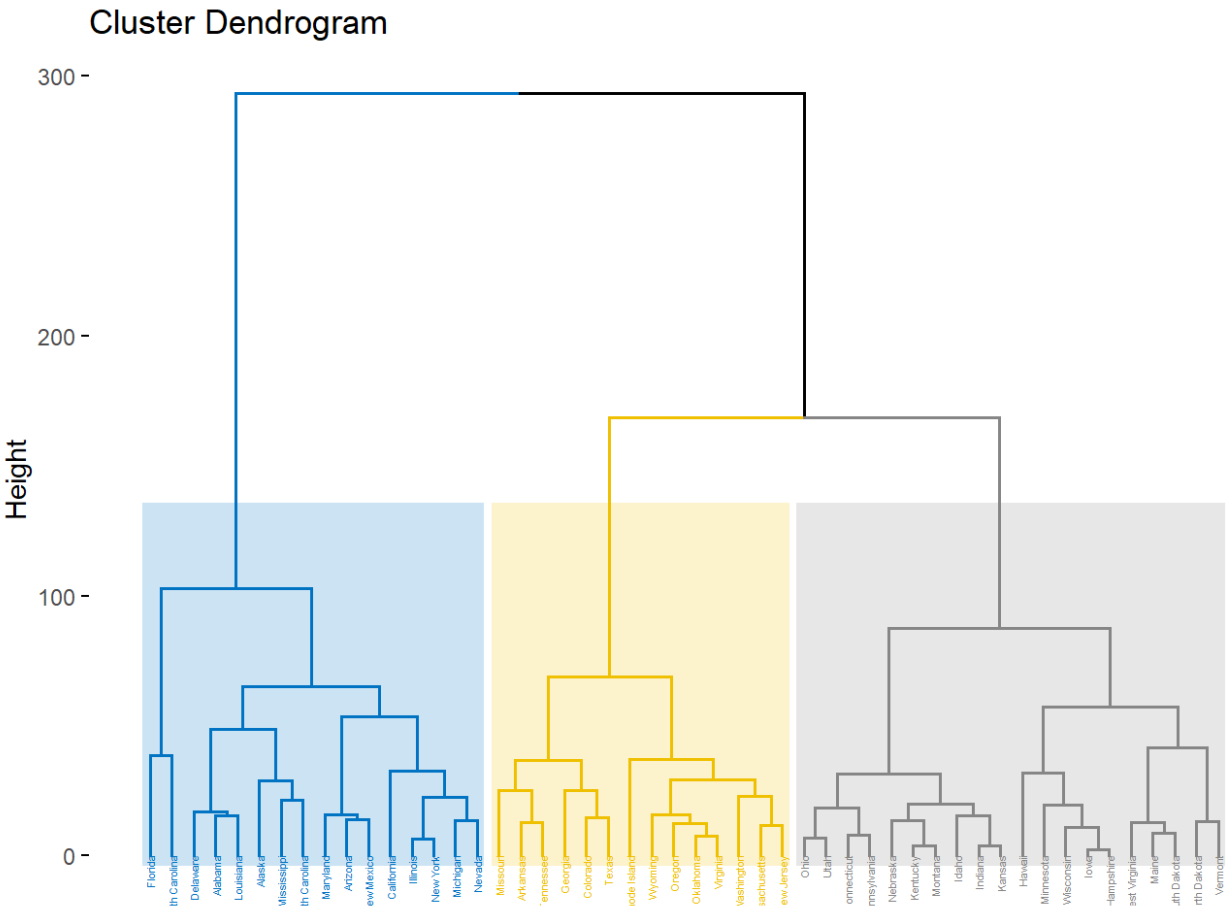
```
##        Alabama         Alaska        Arizona       Arkansas     California
##              1              1              1              2              1
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              2              3              1              1              2
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              3              3              1              3              3
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              3              3              1              3              1
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              2              1              3              1              2
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              3              3              1              3              2
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              1              1              1              3              3
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              2              2              3              2              1
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              3              2              2              3              3
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              2              2              3              3              2
```

Illustrates the cluster assignments for the top-3 clusters

f.

```
fviz_dend(hc.complete,
k = 3,
cex = 0.3,
palette = "jco",
color_labels_by_k = TRUE,
rect = TRUE,
rect_fill = TRUE,
rect_border = "jco",
labels_track_height = 3.5)
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## ℹ The deprecated feature was likely used in the factoextra package.
##   Please report the issue at <https://github.com/kassambara/factoextra/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



f. Compare the clustering from K-means (K = 3) and the hierarchical clustering in a confusion table. Do the clustering results from two methods agree? Compute the Rand index between the two clustering results.

K-means vs. hierarchical clustering

```
table(hc_label,km_label)
```

```
##         km_label
## hc_label  1  2  3
##        1 16  0  0
##        2  0 14  0
##        3  0  0 20
```

the cluster assignments are the same between hc top-3 and km

```
library(mclust) # Adjusted Rand Index
```

```
## Warning: package 'mclust' was built under R version 4.3.2
```

```
## Package 'mclust' version 6.0.1
## Type 'citation("mclust")' for citing this R package in publications.
```

```
##
## Attaching package: 'mclust'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

```
adjustedRandIndex(hc_label, km_label)
```

```
## [1] 1
```

the ARI is 1

```
state_pairs <- expand.grid(names(hc_label),names(hc_label))
state_pairs <- tibble(state_pairs[-which(state_pairs[,1]==state_pairs[,2]),]) %>%
  rename(state1 = Var1, state2 = Var2)
clusts <- tibble(state1 = names(hc_label), hc = hc_label, km = km$cluster)
full <- clusts %>%
  full_join(state_pairs, by = 'state1') %>%
  left_join(clusts %>% rename(state2 = state1), by = c('state2'))
full_eval <- full %>%
  mutate(together_idx = (hc.x==km.x & hc.y==km.y),
         separate_idx = (hc.x!=km.x & hc.y!=km.y),
         discord_idx = (!(together_idx | separate_idx)))

rand_idx <- (sum(full_eval$together_idx) + sum(full_eval$separate_idx)) / nrow(full_eval)
rand_idx
```

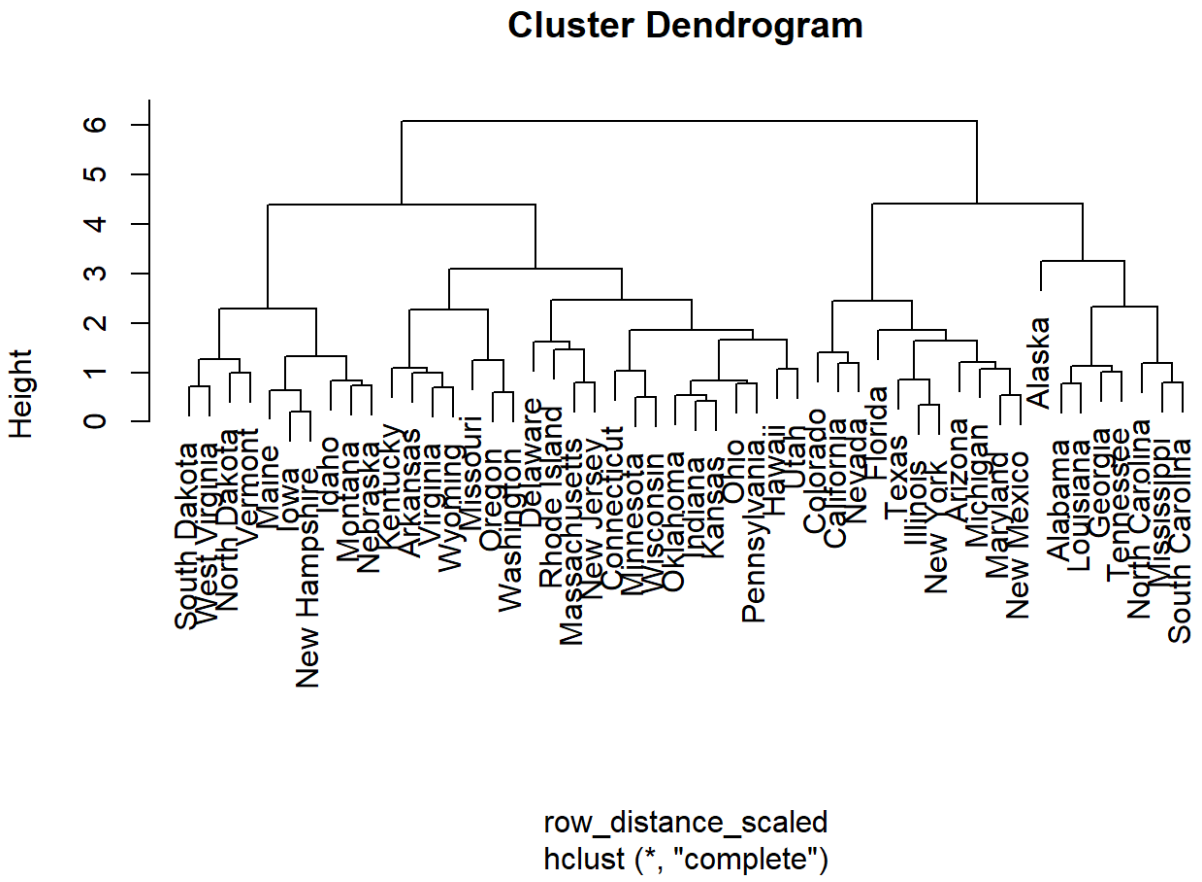
```
## [1] 1
```

the RI is 1, as expected.

g. Hierarchically cluster the states using complete linkage and Euclidean distance, after scaling the variables to have standard deviation one.

Complete Linkage HC after scaling

```
arrest_scaled <- arrest %>%
  mutate(across(everything(), scale))

row_distance_scaled = dist(arrest_scaled, method = "euclidean")
hc.complete_scaled <- hclust(d = row_distance_scaled, method = "complete")
plot(hc.complete_scaled)
```



h. What effect does scaling the variables have on the hierarchical clustering obtained? In your opinion, should the variables be scaled before the inter-observation dissimilarities are computed? Provide a justification for your answer.

It changes the hierarchical clusters obtained, as can be glimpsed by the table for membership in the top-3 clusters.

```
hc_label<-cutree(hc.complete, k = 3)
hc_label_scaled<-cutree(hc.complete_scaled, k = 3)
table(hc_label, hc_label_scaled)
```

```
##         hc_label_scaled
## hc_label  1  2  3
##        1  6  9  1
##        2  2  2 10
##        3  0  0 20
```

Pearson Correlation vs. Euclidean Distance

If each feature vector is standardized to mean zero and variance one, show that Pearson correlation (r) and Euclidean distance (d) are related:

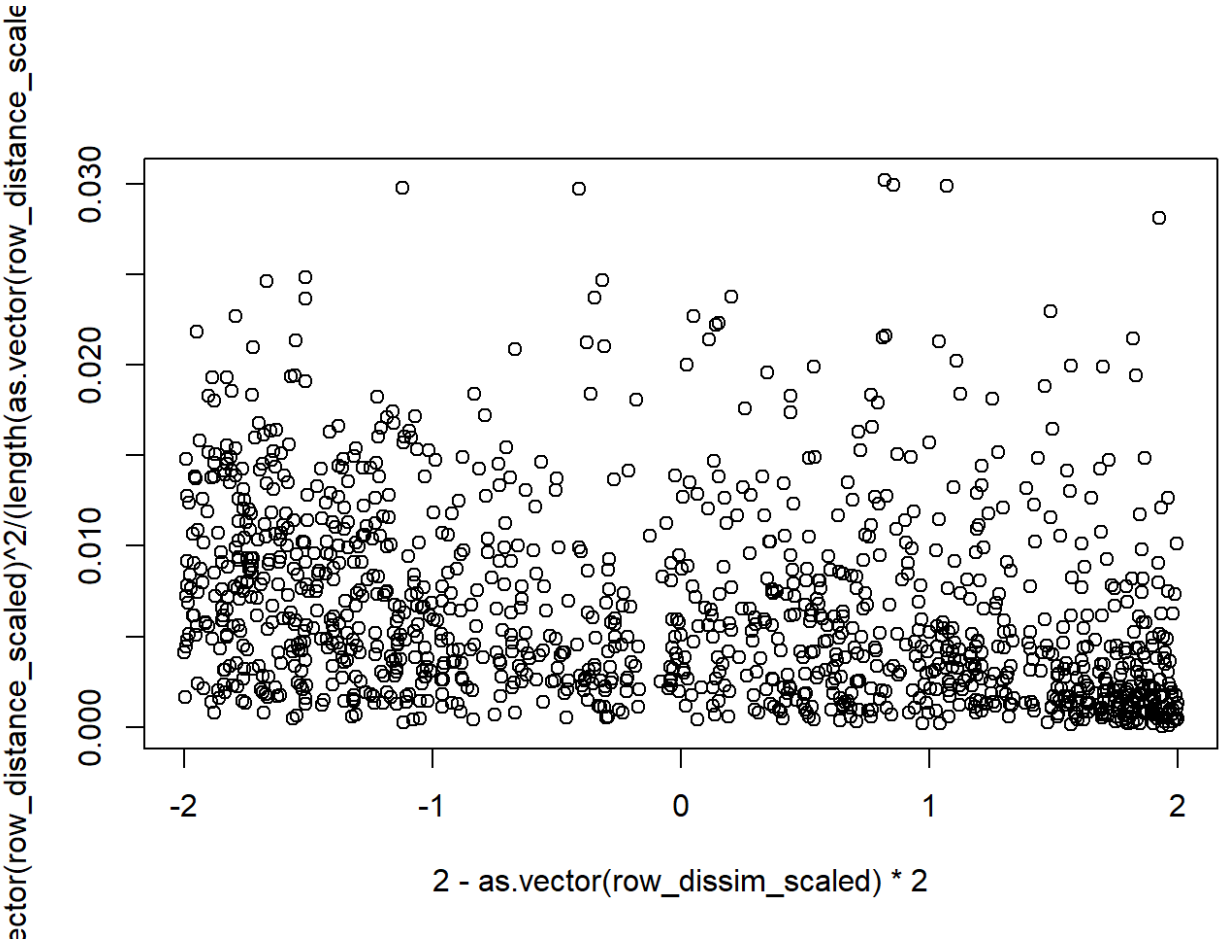
$$\frac{d^2}{n-1} = 2 - 2 \cdot r$$

In other words, large Pearson correlation results in small Euclidean distance. One may use 1 − r as the dissimilarity measure for clustering. (2 points)

```
row_dissim_scaled = as.dist(1 - cor(t(arrest_scaled)))
hc.dissim_scaled <- hclust(d = row_dissim_scaled, method = "complete")
hc_label_dissim<-cutree(hc.dissim_scaled, k = 3)
table(hc_label_scaled, hc_label_dissim)
```

```
##                hc_label_dissim
## hc_label_scaled  1  2  3
##               1  8  0  0
##               2  4  4  3
##               3  7  5 19
```

```
plot(2 - as.vector(row_dissim_scaled)*2, as.vector(row_distance_scaled)^2 / (length(as.vector(row_distance_sca
led)) - 1))
```




```
# plot(hc.complete_scaled)
```

The cluster assignments seem quite different still. However, the plot comparing the dissimilarity/similarity metrics appears to show a higher density of points at the conjunction of low dissimilarity (for positive correlations) and low euclidean distance.