HW5_Yuefei_Chen_Cluster_Analysis

Yuefei Chen

2024-03-06

Question 1, for each model, decide the optimal number of clusters and explain why.

ANS: According to the result of the plot about hierarchical clustering models and non-hierarchical models, the optimal number of clusters is 2 for both models. That is because in the hierachical clustering models and dendrogram, we can find one cluster has most rooms memberships which have close distances and the other one contains outlier points. In the non-hierachical clustering models, there is a significantly decrease in variance from 2 clusters to 3 clusters. Thus, 2 clusters is a optimal choice for the non-hierarchical model.

Question 2, show the membership for each cluster.

ANS:

Hierarchical clustering models: cluster 1: Room 10, Room 12, Room 17 cluster 2: Room 1, Room 2, Room 3, Room 4, Room 5, Room 6, Room 7, Room 8, Room 9, Room 11, Room 13, Room 14, Room 15, Room 16, Room 18, Room 19, Room 20

Non-hierarchical clustering models: cluster 1: Room1, Room5, Room6, Room9, Room10, Room12, Room19 cluster 2: Room2, Room3, Room4, Room7, Room8, Room11, Room13, Room14, Room15, Room16, Room17, Room18, Room20

Question 3

ANS: Click to Visualization

```r
library(cluster)
library(readr)
library(factoextra)
```

Hierarchical Clustering Models

```
## Loading required package: ggplot2
```

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

```r
library(magrittr)
library(NbClust)

Rent <- read.csv("Dataset/Rent_House_random_20_cluster.csv",row.names=1)

attach(Rent)
dim(Rent)
```

```
## [1] 20  8
```

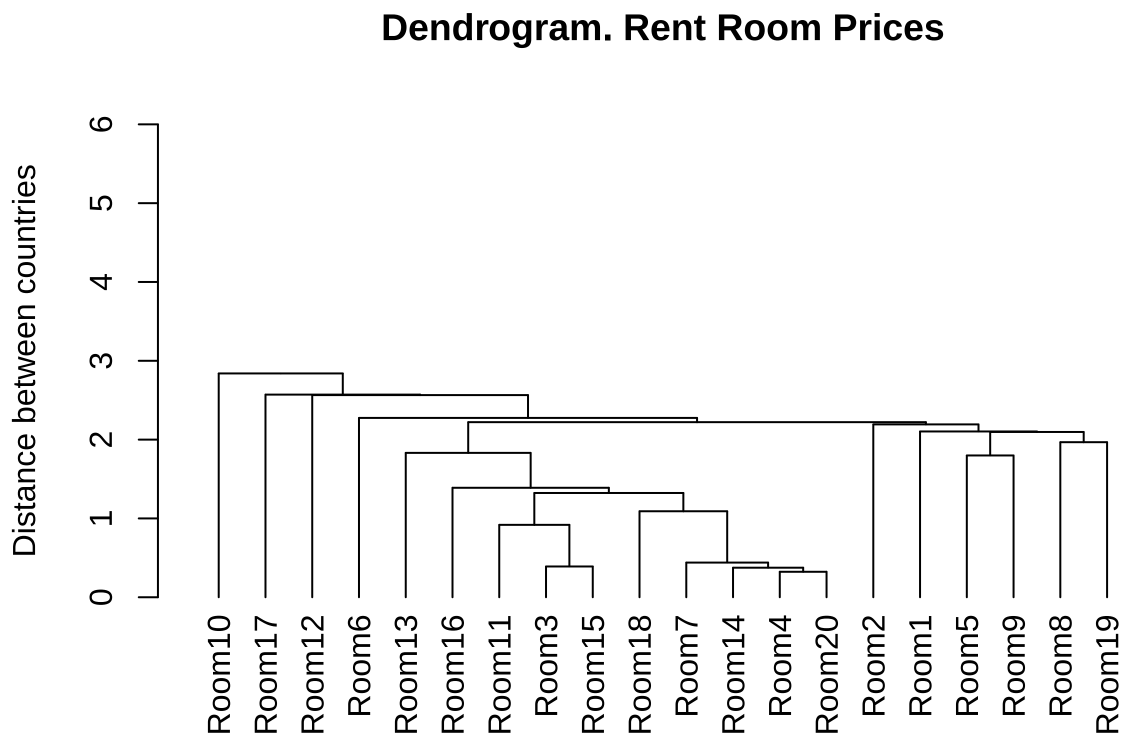
```r
str(Rent)
```

```
## 'data.frame':    20 obs. of  8 variables:
##  $ area          : int  120 45 50 35 204 177 15 70 180 180 ...
##  $ rooms         : int  3 1 2 1 4 3 1 2 3 4 ...
##  $ bathroom      : int  4 1 1 1 4 3 1 2 3 4 ...
##  $ parking.spaces: int  3 1 1 0 2 4 0 1 2 2 ...
##  $ hoa           : int  1350 3000 226 260 0 2700 0 1800 700 2600 ...
##  $ rent.amount   : int  5600 5520 750 1400 3440 6900 1200 4200 2700 2000 ...
##  $ property.tax  : int  560 0 0 0 100 509 0 250 175 584 ...
##  $ fire.insurance: int  71 70 10 18 62 89 16 55 40 26 ...
```

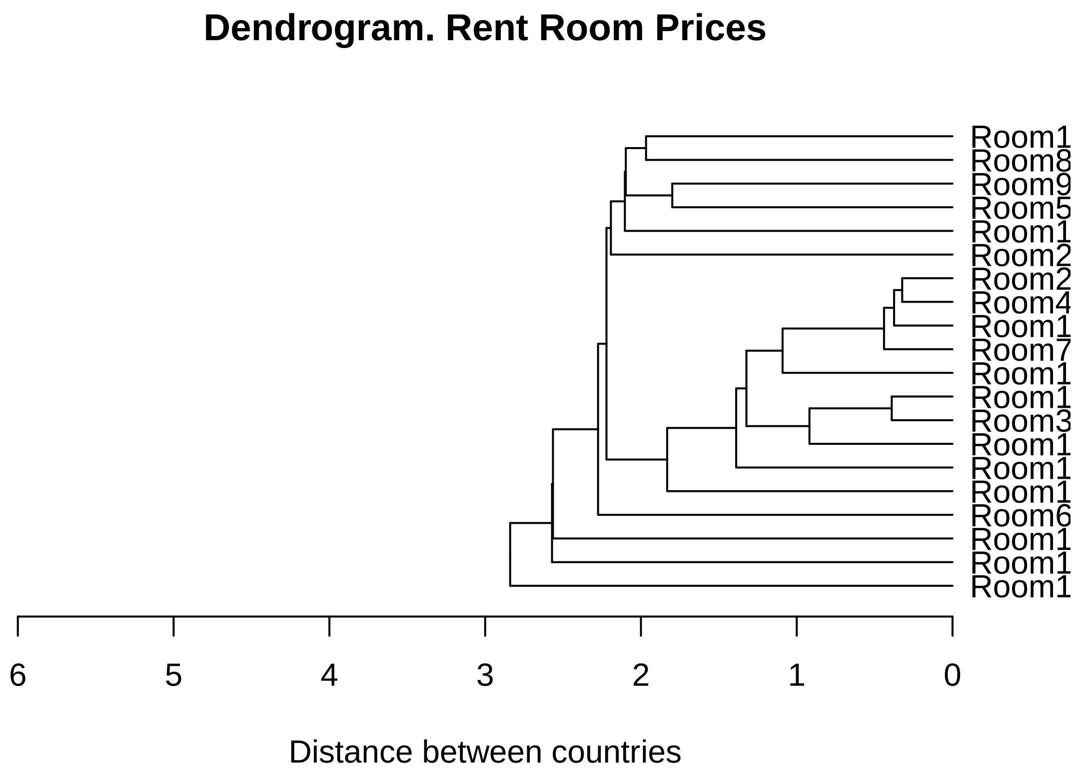
```r
str(Rent)
```

```
## 'data.frame':    20 obs. of  8 variables:
##  $ area          : int  120 45 50 35 204 177 15 70 180 180 ...
##  $ rooms         : int  3 1 2 1 4 3 1 2 3 4 ...
##  $ bathroom      : int  4 1 1 1 4 3 1 2 3 4 ...
##  $ parking.spaces: int  3 1 1 0 2 4 0 1 2 2 ...
##  $ hoa           : int  1350 3000 226 260 0 2700 0 1800 700 2600 ...
##  $ rent.amount   : int  5600 5520 750 1400 3440 6900 1200 4200 2700 2000 ...
##  $ property.tax  : int  560 0 0 0 100 509 0 250 175 584 ...
##  $ fire.insurance: int  71 70 10 18 62 89 16 55 40 26 ...
```

```r
# Hirerarchic cluster analysis, Nearest-neighbor

# Standardizing the data with scale()
matstd.Rent <- scale(Rent)
# Creating a (Euclidean) distance matrix of the standardized data
dist.Rent <- dist(matstd.Rent, method="euclidean")
# Invoking hclust command (cluster analysis by single linkage method)
clusRent.nn <- hclust(dist.Rent, method = "single")

plot(as.dendrogram(clusRent.nn),ylab="Distance between countries",ylim=c(0,6),
     main="Dendrogram. Rent Room Prices")
```




```r
plot(as.dendrogram(clusRent.nn), xlab= "Distance between countries", xlim=c(6,0),
     horiz = TRUE,main="Dendrogram. Rent Room Prices")
```




```
# We will use agnes function as it allows us to select option for data standardization, the distance me
(agn.Rent <- agnes(Rent, metric="euclidean", stand=TRUE, method = "single"))
```

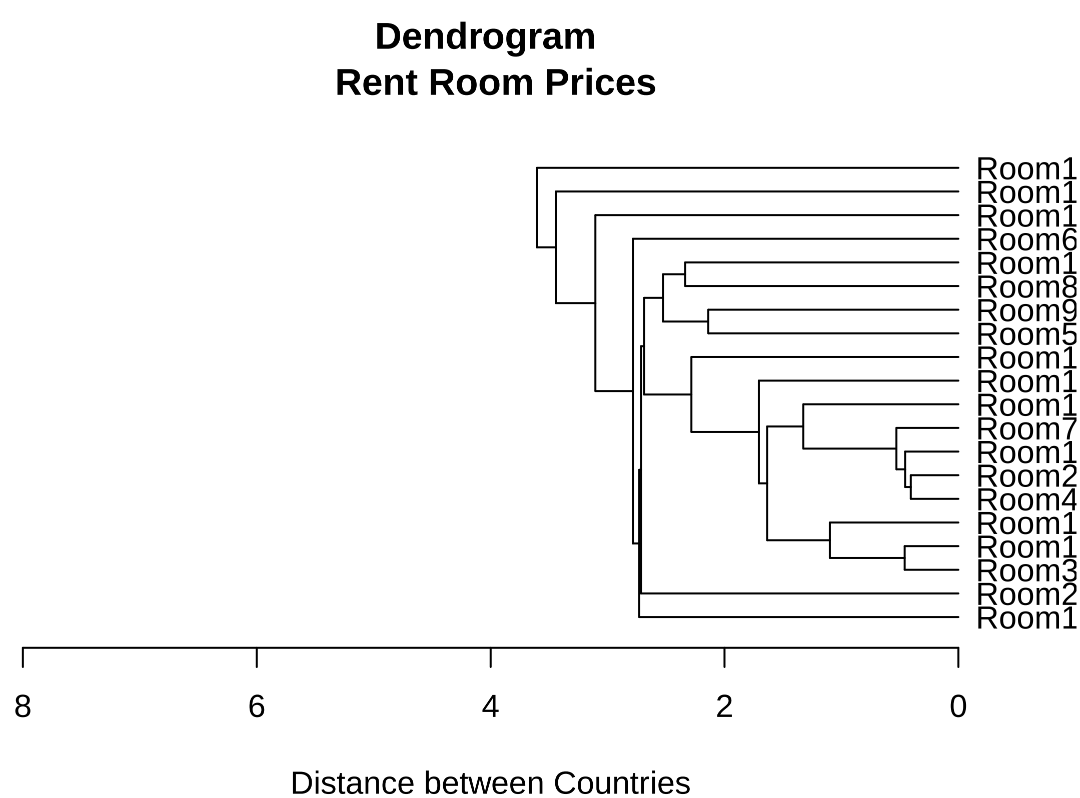
```
## Call:     agnes(x = Rent, metric = "euclidean", stand = TRUE, method = "single")
## Agglomerative coefficient:  0.4941766
## Order of objects:
##  [1] Room1  Room2  Room3  Room15 Room11 Room4  Room20 Room14 Room7  Room18
## [11] Room16 Room13 Room5  Room9  Room8  Room19 Room6  Room17 Room12 Room10
## Height (summary):
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4071  1.2126  2.2831  1.9984  2.7218  3.6042
##
## Available components:
## [1] "order"     "height"    "ac"        "merge"     "diss"      "call"
## [7] "method"    "order.lab" "data"
```

```
#View(agn.employ)

#  Description of cluster merging
agn.Rent$merge
```

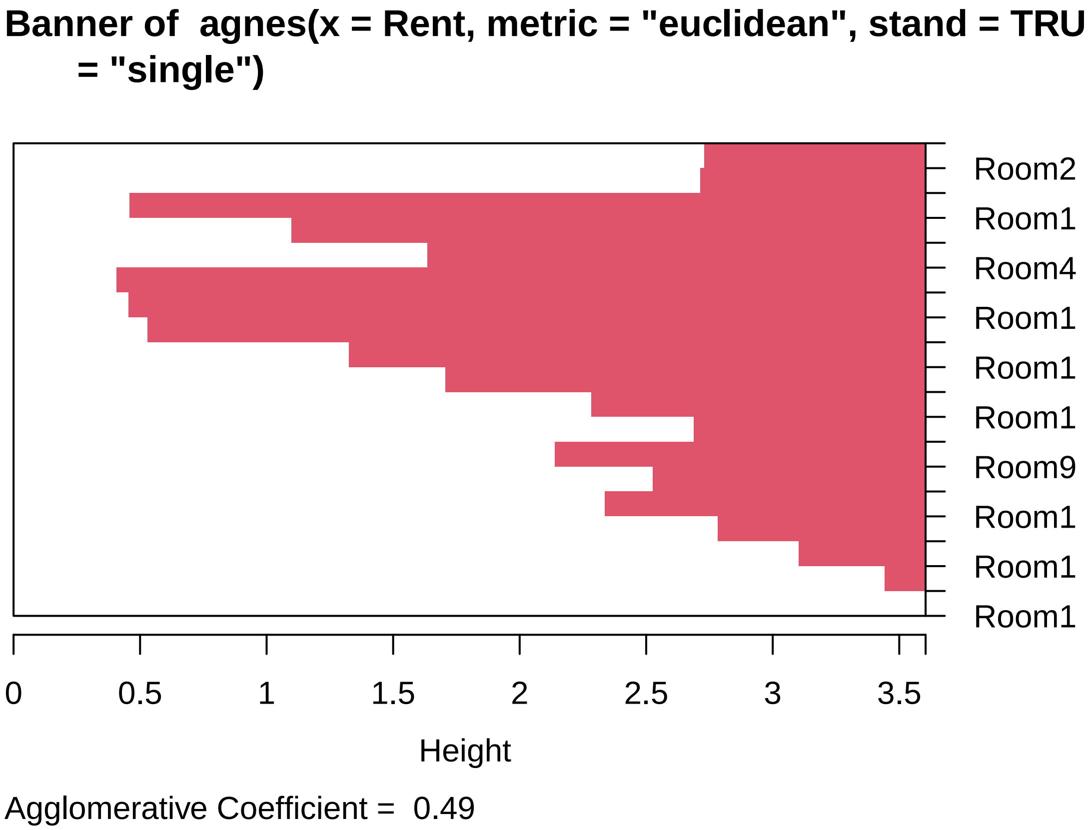
```
##       [,1] [,2]
##  [1,]   -4  -20
##  [2,]    1  -14
##  [3,]   -3  -15
```

```
## [4,]    2  -7
## [5,]    3 -11
## [6,]    4 -18
## [7,]    5   6
## [8,]    7 -16
## [9,]   -5  -9
## [10,]   8 -13
## [11,]  -8 -19
## [12,]   9  11
## [13,]  10  12
## [14,]  -2  13
## [15,]  -1  14
## [16,]  15  -6
## [17,]  16 -17
## [18,]  17 -12
## [19,]  18 -10
```

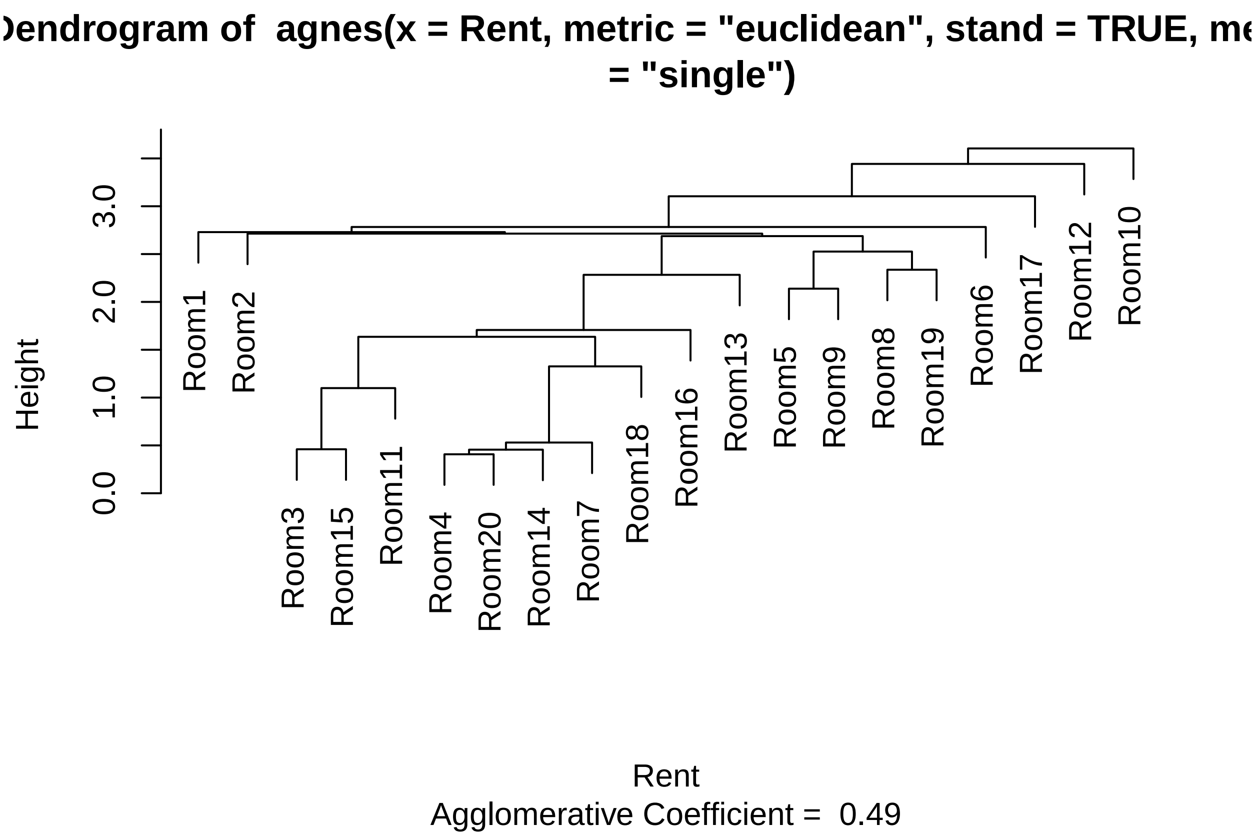
```
#Dendogram
plot(as.dendrogram(agn.Rent), xlab= "Distance between Countries",xlim=c(8,0),
     horiz = TRUE,main="Dendrogram \n Rent Room Prices")
```




```
#Interactive Plots
#plot(agn.employ,ask=TRUE)
plot(agn.Rent, which.plots=1)
```




```
plot(agn.Rent, which.plots=2)
```




```r
plot(agn.Rent, which.plots=3)
```

```r
# K-Means Clustering
matstd.Rent <- scale(Rent)
# K-means, k=2, 3, 4, 5, 6
# Centers (k's) are numbers thus, 10 random sets are chosen
(kmeans2.Rent <- kmeans(matstd.Rent,2,nstart = 10))
```

Non-hierarchy Clustering Models

```
## K-means clustering with 2 clusters of sizes 13, 7
##
## Cluster means:
##         area      rooms   bathroom parking.spaces        hoa rent.amount
## 1 -0.6443338 -0.5880235 -0.6474769     -0.5824038 -0.2777729  -0.2951752
## 2  1.1966198  1.0920436  1.2024571      1.0816071  0.5158640   0.5481825
##   property.tax fire.insurance
## 1   -0.4835749     -0.4312383
## 2    0.8980677      0.8008712
##
## Clustering vector:
##   Room1   Room2   Room3   Room4   Room5   Room6   Room7   Room8   Room9 Room10 Room11
```

```
##      2      1      1      1      2      2      1      1      2      2      1
## Room12 Room13 Room14 Room15 Room16 Room17 Room18 Room19 Room20
##      2      1      1      1      1      1      1      2      1
##
## Within cluster sum of squares by cluster:
## [1] 42.76427 31.10739
##  (between_SS / total_SS =  51.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# Computing the percentage of variation accounted for. Two clusters
perc.var.2 <- round(100*(1 - kmeans2.Rent$betweenss/kmeans2.Rent$totss),1)
names(perc.var.2) <- "Perc. 2 clus"
perc.var.2
```

```
## Perc. 2 clus
##         48.6
```

```
# Computing the percentage of variation accounted for. Three clusters
(kmeans3.Rent <- kmeans(matstd.Rent,3,nstart = 10))
```

```
## K-means clustering with 3 clusters of sizes 7, 3, 10
##
## Cluster means:
##         area      rooms   bathroom parking.spaces        hoa rent.amount
## 1  1.1966198  1.0920436  1.2024571      1.0816071  0.5158640   0.5481825
## 2 -0.5857095 -0.7852641 -0.4088757     -0.1268925  0.8789775   1.2858376
## 3 -0.6619210 -0.5288513 -0.7190572     -0.7190572 -0.6247981  -0.7694791
##   property.tax fire.insurance
## 1    0.8980677      0.8008712
## 2   -0.2936090      0.9002212
## 3   -0.5405647     -0.8306762
##
## Clustering vector:
##  Room1  Room2  Room3  Room4  Room5  Room6  Room7  Room8  Room9 Room10 Room11
##      1      2      3      3      1      1      3      2      1      1      3
## Room12 Room13 Room14 Room15 Room16 Room17 Room18 Room19 Room20
##      1      3      3      3      3      2      3      1      3
##
## Within cluster sum of squares by cluster:
## [1] 31.107386  6.982283 12.564089
##  (between_SS / total_SS =  66.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
perc.var.3 <- round(100*(1 - kmeans3.Rent$betweenss/kmeans3.Rent$totss),1)
names(perc.var.3) <- "Perc. 3 clus"
perc.var.3
```

```
## Perc. 3 clus
##         33.3
```

```r
# Computing the percentage of variation accounted for. Four clusters
(kmeans4.Rent <- kmeans(matstd.Rent,4,nstart = 10))
```

```
## K-means clustering with 4 clusters of sizes 4, 10, 3, 3
##
## Cluster means:
##         area      rooms   bathroom parking.spaces        hoa rent.amount
## 1  1.2622866  1.2980896  1.1420321      0.5075698  0.1361292   0.1068662
## 2 -0.6619210 -0.5288513 -0.7190572     -0.7190572 -0.6247981  -0.7694791
## 3 -0.5857095 -0.7852641 -0.4088757     -0.1268925  0.8789775   1.2858376
## 4  1.1090641  0.8173156  1.2830237      1.8469902  1.0221772   1.1366043
##   property.tax fire.insurance
## 1    0.2287971      0.3786338
## 2   -0.5405647     -0.8306762
## 3   -0.2936090      0.9002212
## 4    1.7904284      1.3638544
##
## Clustering vector:
##  Room1  Room2  Room3  Room4  Room5  Room6  Room7  Room8  Room9 Room10 Room11
##      4      3      2      2      1      4      2      3      1      1      2
## Room12 Room13 Room14 Room15 Room16 Room17 Room18 Room19 Room20
##      4      2      2      2      2      3      2      1      2
##
## Within cluster sum of squares by cluster:
## [1] 11.524571 12.564089  6.982283  7.028519
##  (between_SS / total_SS =  74.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
perc.var.4 <- round(100*(1 - kmeans4.Rent$betweenss/kmeans4.Rent$totss),1)
names(perc.var.4) <- "Perc. 4 clus"
perc.var.4
```

```
## Perc. 4 clus
##         25.1
```

```r
# Computing the percentage of variation accounted for. Five clusters
(kmeans5.Rent <- kmeans(matstd.Rent,5,nstart = 10))
```

```
## K-means clustering with 5 clusters of sizes 5, 3, 3, 5, 4
##
```

```
## Cluster means:
##         area       rooms   bathroom parking.spaces        hoa rent.amount
## 1 -0.3085767  0.04807739 -0.4652723     -0.4652723 -0.6315643  -0.8143334
## 2 -0.5857095 -0.78526405 -0.4088757     -0.1268925  0.8789775   1.2858376
## 3  1.1090641  0.81731565  1.2830237      1.8469902  1.0221772   1.1366043
## 4 -1.0152653 -1.10577999 -0.9728421     -0.9728421 -0.6180319  -0.7246247
## 5  1.2622866  1.29808956  1.1420321      0.5075698  0.1361292   0.1068662
##   property.tax fire.insurance
## 1   -0.4668132     -0.8538579
## 2   -0.2936090      0.9002212
## 3    1.7904284      1.3638544
## 4   -0.6143161     -0.8074945
## 5    0.2287971      0.3786338
##
## Clustering vector:
##  Room1  Room2  Room3  Room4  Room5  Room6  Room7  Room8  Room9 Room10 Room11
##      3      2      1      4      5      3      4      2      5      5      1
## Room12 Room13 Room14 Room15 Room16 Room17 Room18 Room19 Room20
##      3      1      4      1      1      2      4      5      4
##
## Within cluster sum of squares by cluster:
## [1]  5.191694  6.982283  7.028519  1.426926 11.524571
##  (between_SS / total_SS =  78.8 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
perc.var.5 <- round(100*(1 - kmeans5.Rent$betweenss/kmeans5.Rent$totss),1)
names(perc.var.5) <- "Perc. 5 clus"
perc.var.5
```

```
## Perc. 5 clus
##         21.2
```

```r
(kmeans6.Rent <- kmeans(matstd.Rent,6,nstart = 10))
```

```
## K-means clustering with 6 clusters of sizes 3, 5, 3, 3, 1, 5
##
## Cluster means:
##         area       rooms   bathroom parking.spaces        hoa rent.amount
## 1  1.2369716  1.13783159  1.0010405      0.4370740 -0.3919196   0.3050009
## 2 -0.3085767  0.04807739 -0.4652723     -0.4652723 -0.6315643  -0.8143334
## 3  1.1090641  0.81731565  1.2830237      1.8469902  1.0221772   1.1366043
## 4 -0.5857095 -0.78526405 -0.4088757     -0.1268925  0.8789775   1.2858376
## 5  1.3382316  1.77886347  1.5650069      0.7190572  1.7202756  -0.4875377
## 6 -1.0152653 -1.10577999 -0.9728421     -0.9728421 -0.6180319  -0.7246247
##   property.tax fire.insurance
## 1  -0.09917336      0.6812833
## 2  -0.46681320     -0.8538579
## 3   1.79042836      1.3638544
## 4  -0.29360902      0.9002212
```

```
## 5   1.21270862     -0.5293146
## 6  -0.61431611     -0.8074945
##
## Clustering vector:
##  Room1  Room2  Room3  Room4  Room5  Room6  Room7  Room8  Room9 Room10 Room11
##      3      4      2      6      1      3      6      4      1      5      2
## Room12 Room13 Room14 Room15 Room16 Room17 Room18 Room19 Room20
##      3      2      6      2      2      4      6      1      6
##
## Within cluster sum of squares by cluster:
## [1] 4.703459 5.191694 7.028519 6.982283 0.000000 1.426926
##  (between_SS / total_SS =  83.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
# Computing the percentage of variation accounted for. Six clusters
perc.var.6 <- round(100*(1 - kmeans6.Rent$betweenss/kmeans6.Rent$totss),1)
names(perc.var.6) <- "Perc. 6 clus"
perc.var.6
```

```
## Perc. 6 clus
##         16.7
```

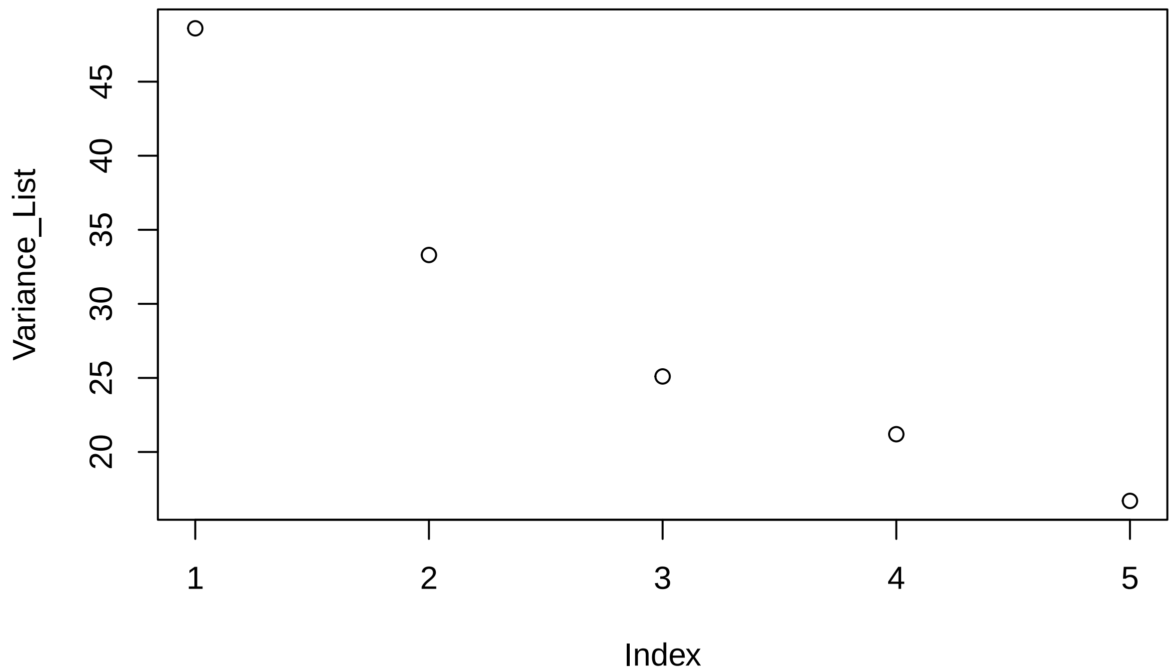
```r
attributes(perc.var.6)
```

```
## $names
## [1] "Perc. 6 clus"
```

```r
Variance_List <- c(perc.var.2,perc.var.3,perc.var.4,perc.var.5,perc.var.6)
Variance_List
```

```
## Perc. 2 clus Perc. 3 clus Perc. 4 clus Perc. 5 clus Perc. 6 clus
##         48.6         33.3         25.1         21.2         16.7
```

```r
plot(Variance_List)
```

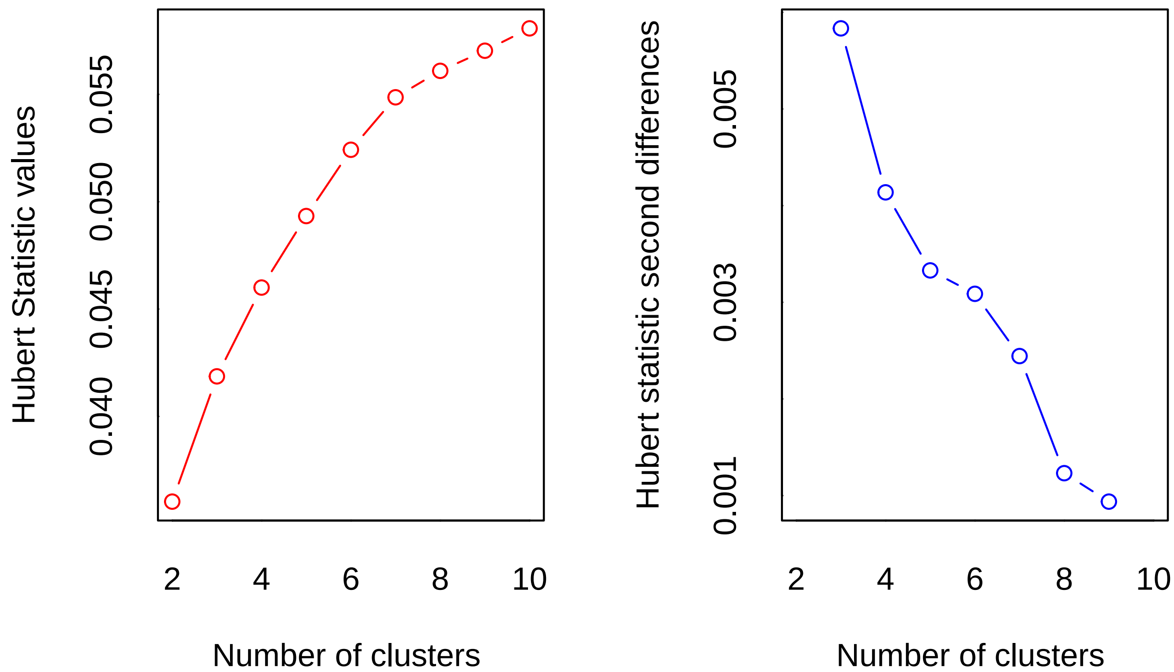



```
kmeans2.Rent$cluster
```

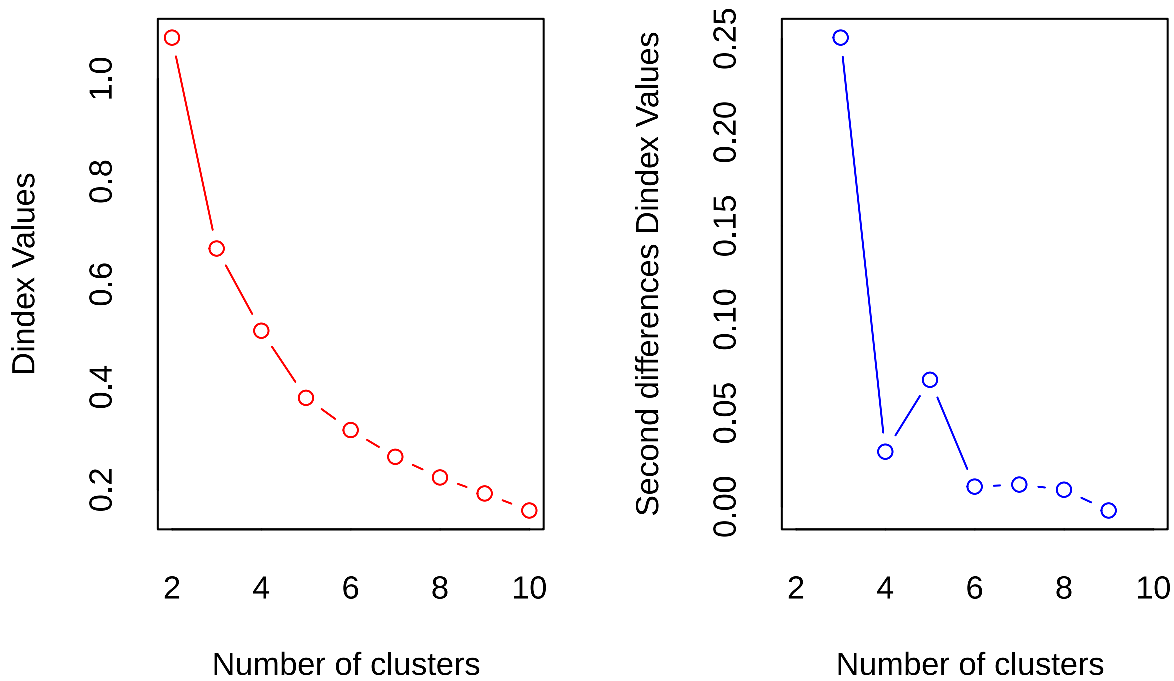
```
##  Room1  Room2  Room3  Room4  Room5  Room6  Room7  Room8  Room9 Room10 Room11
##      2      1      1      1      2      2      1      1      2      2      1
## Room12 Room13 Room14 Room15 Room16 Room17 Room18 Room19 Room20
##      2      1      1      1      1      1      1      2      1
```

visualization

```
# use PC1 and PC2
rent_pca <- prcomp(Rent, scale = TRUE)
rent.nbclust <- rent_pca$x[,c(1,2)] %>% scale() %>% NbClust(distance = "euclidean", min.nc = 2, max.nc =
```



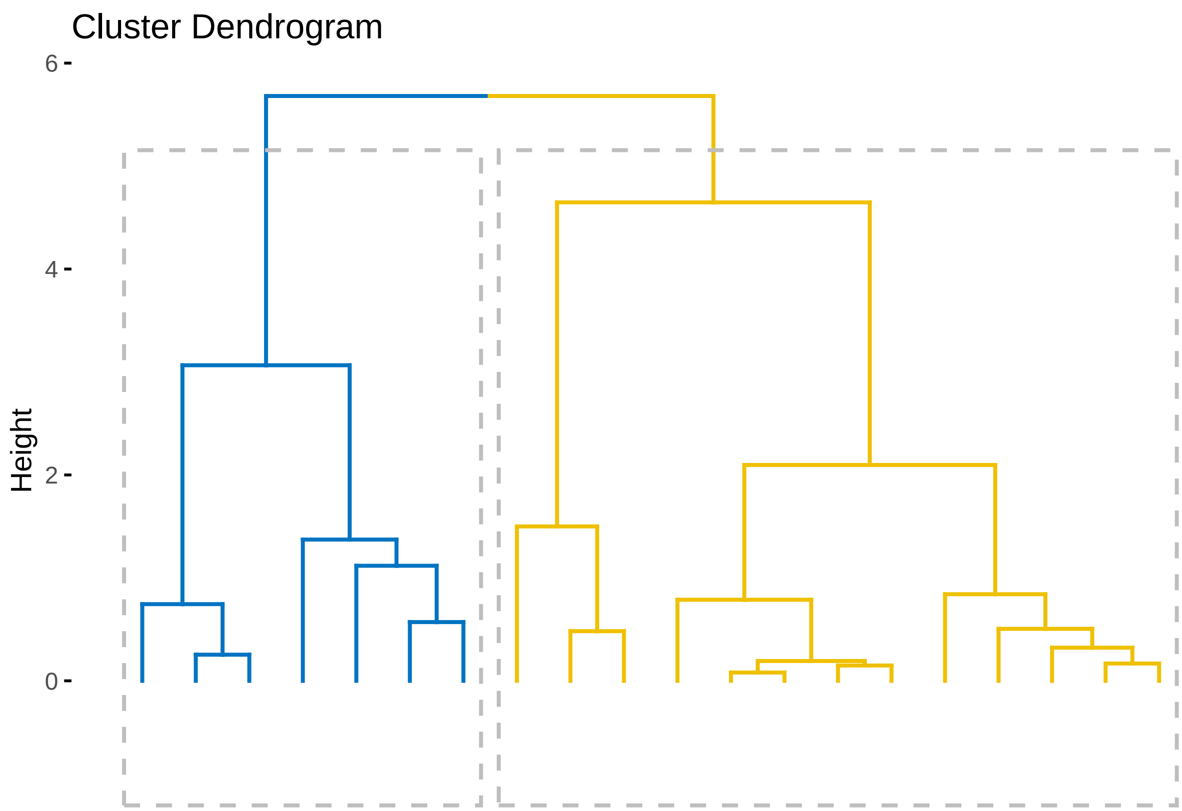

```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```




```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 4 proposed 2 as the best number of clusters
## * 8 proposed 3 as the best number of clusters
## * 2 proposed 4 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 7 proposed 10 as the best number of clusters
##
##                    ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  3
##
##
## *******************************************************************
```

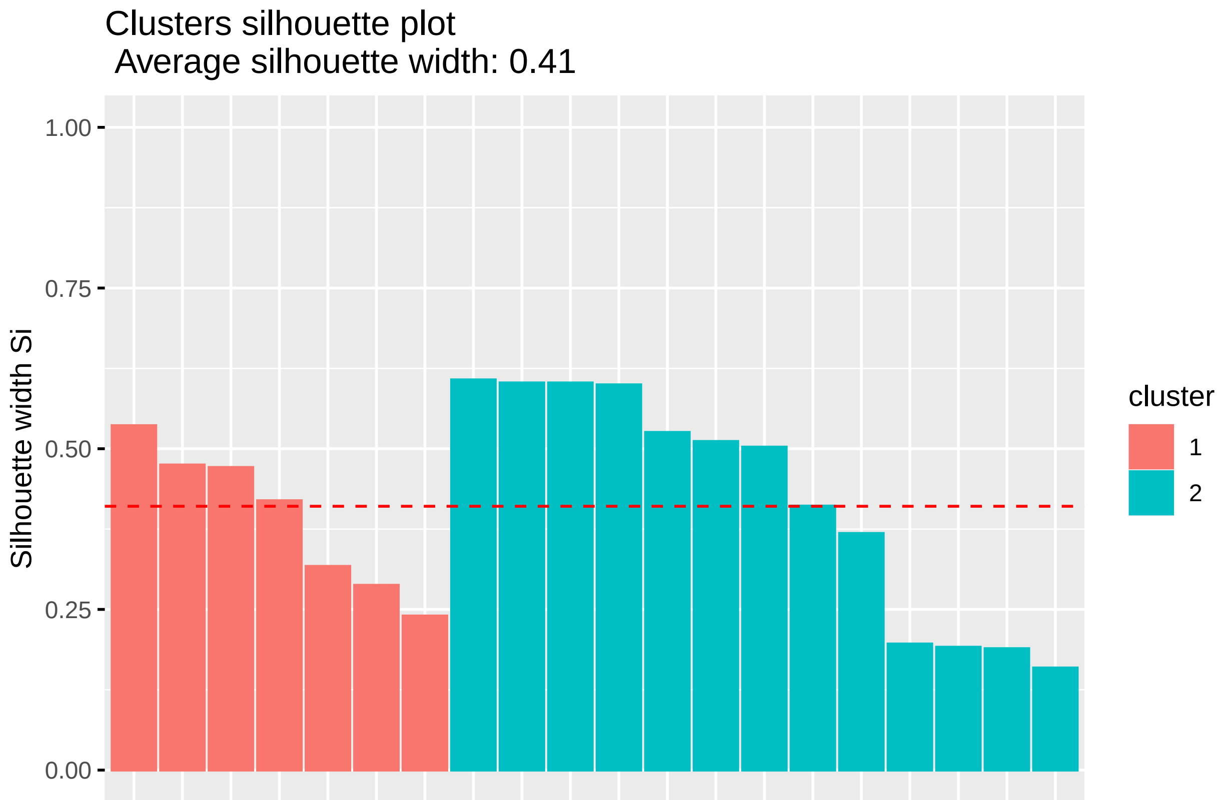
```r
rent.hc <- rent_pca$x[,c(1,2)] %>% scale() %>%
  eclust("hclust", k = 2, graph = FALSE)

fviz_dend(rent.hc, palette = "jco",
          rect = TRUE, show_labels = FALSE)
```

```
## 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.
```




```r
#Inspect the silhouette plot:
fviz_silhouette(rent.hc)
```

```
##   cluster size ave.sil.width
## 1       1    7          0.39
## 2       2   13          0.42
```


Clusters silhouette plot
Average silhouette width: 0.41
Silhouette width Si
1.00
0.75
0.50
0.25
0.00
cluster
1
2