Hierarchical Clustering

Arnab Panja

2022-04-17

Introduction

Today we will discuss about one of the popular methods of unsupervised learning called the Hierarchical Clustering. Hierarchical Clustering evaluates the observations and clusters them into groups based on the distance between the observations. The observations which are closer to each other are grouped similarly.

We will the US Arrests data within the Base R to demonstrate the concepets of Hierarchical clustering.

```
df.arrests <- USArrests

dim(df.arrests)
```

```
## [1] 50  4
```

```
names(df.arrests)
```

```
## [1] "Murder"   "Assault"  "UrbanPop" "Rape"
```

As we can see the US Arrests data has 50 observations and 4 variables. The observations correspond to each of the 50 states of the US. The data set has 5 variables including the US State. We can get a glimpse of the data by using the head function.

```
head(df.arrests)
```

```
##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7
```

As we can see the row names are the US States. We would better like to convert the row names to a proper data frame column. We perform the below steps to achieve that.

```
df.arrests$State <- rownames(df.arrests)

rownames(df.arrests) <- 1:nrow(df.arrests)

head(df.arrests)
```

```
##   Murder Assault UrbanPop Rape      State
## 1   13.2     236       58 21.2    Alabama
## 2   10.0     263       48 44.5     Alaska
## 3    8.1     294       80 31.0    Arizona
## 4    8.8     190       50 19.5   Arkansas
## 5    9.0     276       91 40.6 California
## 6    7.9      204        78 38.7    Colorado
```

Now the data frame has been converted as desired.

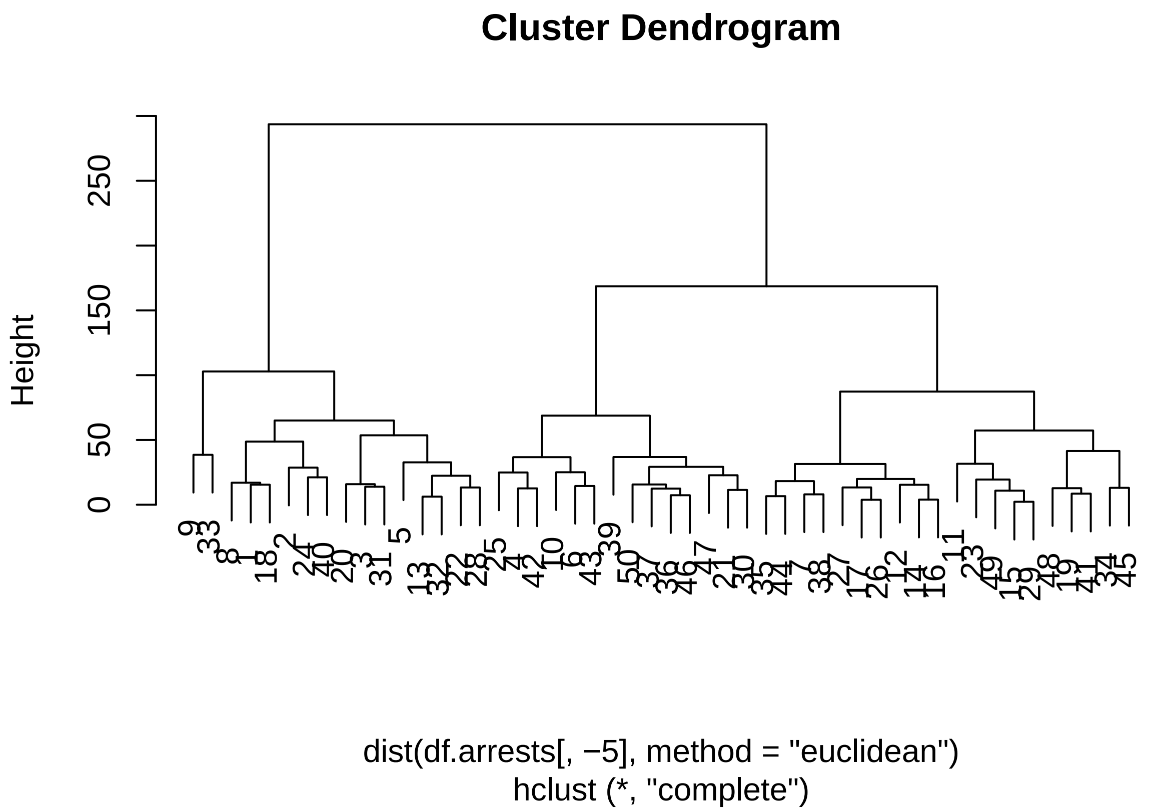
Hierarchical Clustering - Unscaled

We will perform hierarchical clustering using the **hclust** function of **Base R** using un-scaled values first and then with scaled values next. Hierarchical clustering algorithm uses different linkage methods to find the dissimilarity between groups of observations. Here we will use the **Complete** method of linkage. This can be specified as an argument to the hclust function as shown below.

```
set.seed(1234)

hc.complete <- hclust(d = dist(df.arrests[, -5],
                               method = "euclidean"),
                      method = "complete")
```

The hierarchical clustering thus obtained can be visualized using a **Dendrogram** plot. The below figure shows the Dendrogram plot obtained from above.

```
plot(hc.complete)
```



As we can see the leaf nodes are each of the 50 observations of the data. They are clustered and joined according to their proximity calculated using the euclidean distance between any two observations.

Let us now cut the cluster at a level 3 so that all the 50 observations can be divided into one of the 3 clusters.

```
vclust <- cutree(tree = hc.complete, k = 3)
```

```
matrix(vclust, ncol = 5)
```

```
##       [,1] [,2] [,3] [,4] [,5]
##  [1,]    1    3    2    1    3
##  [2,]    1    3    1    1    2
##  [3,]    1    1    3    1    2
##  [4,]    2    3    1    3    3
##  [5,]    1    3    2    3    3
##  [6,]    2    3    3    2    2
##  [7,]    3    3    3    2    2
##  [8,]    1    1    1    3    3
##  [9,]    1    3    3    2    3
## [10,]    2    1    2    1    2
```

For clarity of the data, the clusters created are shown in the matrix form. The first column of the matrix shows the clusters of the first 10 observations, the 2nd column for the observations from 11 to 20 and so on. As we can see each of the 50 observations have been clustered into one of 3 clusters numbered as 1 or a 2 or a 3.

Let us combine the cluster numbers to the original data frame.

```
unscaled.clust <- cbind(df.arrests, clust = vclust)

head(unscaled.clust)
```

```
##   Murder Assault UrbanPop Rape      State clust
## 1   13.2     236       58 21.2    Alabama     1
## 2   10.0     263       48 44.5     Alaska     1
## 3    8.1     294       80 31.0    Arizona     1
## 4    8.8     190       50 19.5   Arkansas     2
## 5    9.0     276       91 40.6 California     1
## 6    7.9     204       78 38.7   Colorado     2
```

We can observe the distribution of clusters using a very simple method of aggregation.

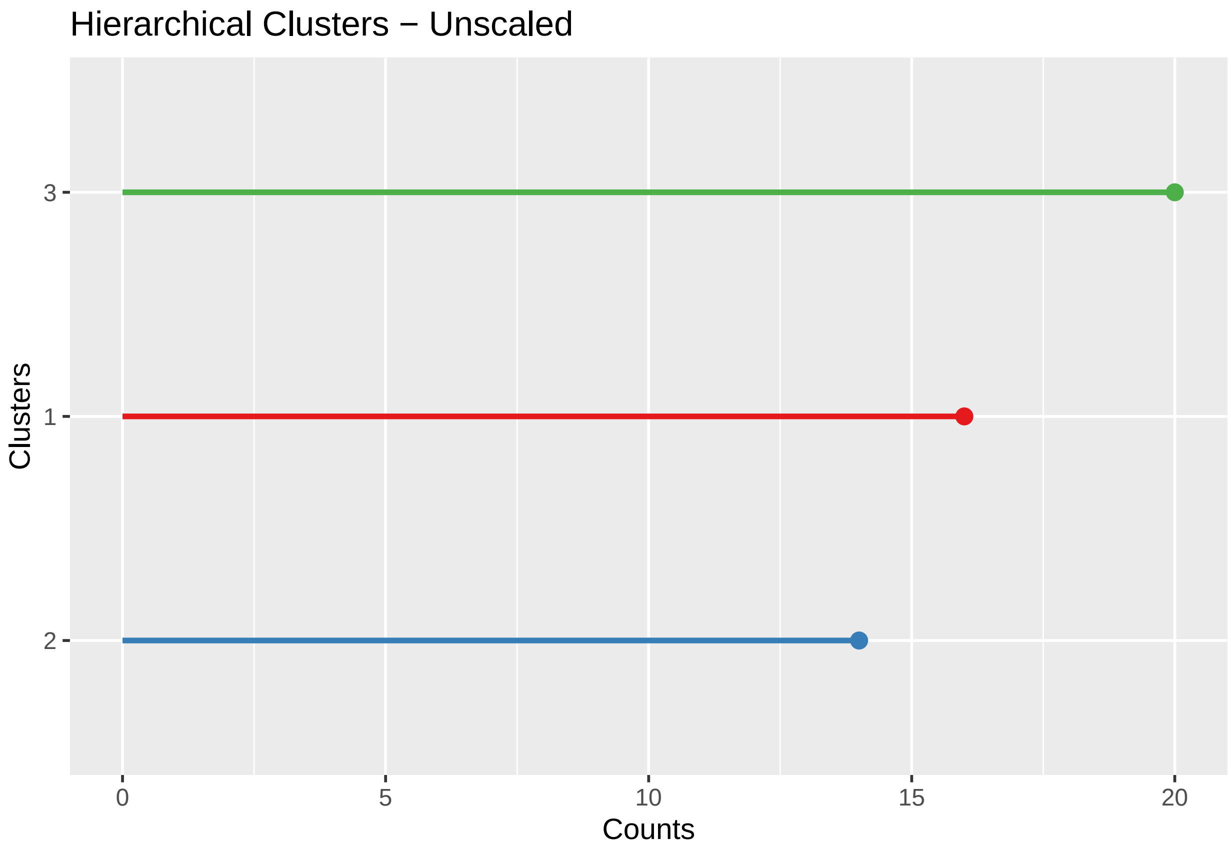
```
unscaled.clust.dist <-  data.frame(clust = as.factor(unique(unscaled.clust$clust)),
           cnt = apply(X = as.matrix(unique(unscaled.clust$clust), drop = FALSE),
                       MARGIN = 1,
                       FUN = function(x) nrow(unscaled.clust[unscaled.clust$clust == x[1], ])))



unscaled.clust.dist
```

```
##   clust cnt
## 1     1  16
## 2     2  14
## 3     3  20
```

16 observations are in cluster 1, 14 are in cluster 2 and 20 observations are in cluster 3.

We can also plot the distribution of 3 clusters into a bar plot as below.

```
p_unscaled <- ggplot(data = unscaled.clust.dist) +
  geom_segment(mapping = aes(x = 0,
                             xend = cnt,
                             y =  reorder(clust, cnt),
                             yend = reorder(clust, cnt),
                             color = clust), size = 1.0,
               show.legend = FALSE) +
  geom_point(mapping = aes(x = cnt,
                           y = reorder(as.factor(clust), cnt),
                           color = clust), size = 2.5,
             show.legend = FALSE) +
  scale_fill_brewer(palette = "Set1") +
  scale_color_brewer(palette = "Set1") +
  labs(x = "Counts",
       y = "Clusters",
       title = "Hierarchical Clusters - Unscaled")


p_unscaled
```



Hierarchical Clustering - Scaled

The variables of the data frame are in a different scale. It is always advisable in such situations to first scale the variables and then perform the hierarchical clustering. So let us scale the variables to a mean of 1.0 and standard deviation of 0. This can be achieved by using the scale function in Base R.

```
df.arrests.scaled <-  as.data.frame(cbind(apply(X = df.arrests[, 1:4],
                                    MARGIN = 2,
                                    FUN = scale), State = df.arrests$State))

df.arrests.scaled[, 1:4] <- apply(X = df.arrests.scaled[, 1:4],
                                  MARGIN = 2,
                                  FUN = as.numeric)

head(df.arrests.scaled)
```

```
##       Murder   Assault   UrbanPop         Rape      State
## 1 1.24256408 0.7828393 -0.5209066 -0.003416473    Alabama
## 2 0.50786248 1.1068225 -1.2117642  2.484202941     Alaska
## 3 0.07163341 1.4788032  0.9989801  1.042878388    Arizona
## 4 0.23234938 0.2308680 -1.0735927 -0.184916602   Arkansas
## 5 0.27826823 1.2628144  1.7589234  2.067820292 California
## 6 0.02571456 0.3988593  0.8608085  1.864967207   Colorado
```

The data frame variables are now scaled. Are the values scaled to a mean of 0 and standard deviation of 1? Let us verify this

```
print("== Mean after scaling ==")
```

```
## [1] "== Mean after scaling =="
```

```
apply(X = df.arrests.scaled[, 1:4],
      MARGIN = 2,
      FUN = function(x)
        round(mean(x),digits = 2))
```

```
##   Murder  Assault UrbanPop     Rape
##        0        0        0        0
```

```
print("== Standard deviation after scaling ==")
```

```
## [1] "== Standard deviation after scaling =="
```

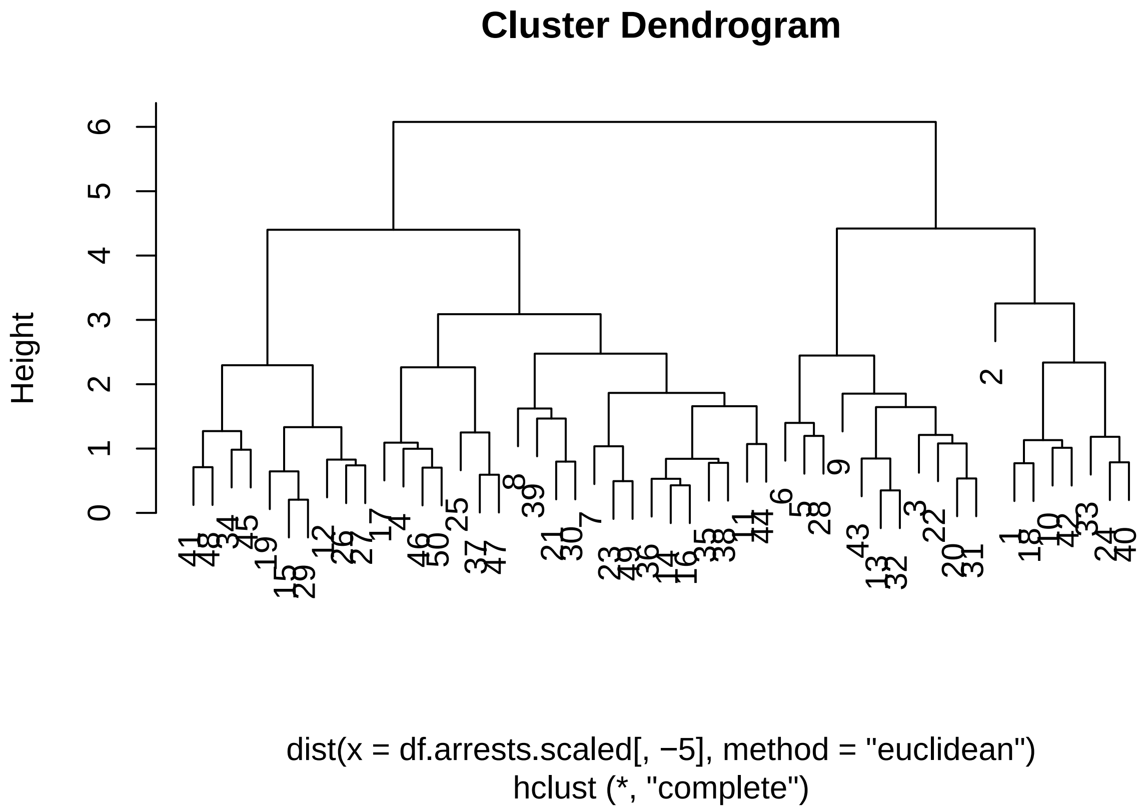
```
apply(X = df.arrests.scaled[, 1:4],
      MARGIN = 2,
      FUN = sd)
```

```
##   Murder  Assault UrbanPop     Rape
##        1        1        1        1
```

The variables have been converted to a mean of 0 and standard deviation of 1.

We will now apply hierarchical clustering on this scaled variables and plot the dendrogram.

```
set.seed(1234)

hc.scaled <- hclust(d = dist(x = df.arrests.scaled[, -5],
                             method = "euclidean"),
                    method = "complete")
plot(hc.scaled)
```



Let us now cut the dendrogram to to classify the scaled observations into 3 clusters.

```
vscaled <- cutree(tree = hc.scaled, k = 3)

matrix(vscaled, ncol = 5)
```

```
##       [,1] [,2] [,3] [,4] [,5]
##  [1,]    1    3    3    2    3
##  [2,]    1    3    2    2    1
##  [3,]    2    2    3    1    2
##  [4,]    3    3    1    3    3
##  [5,]    2    3    3    3    3
##  [6,]    2    3    3    3    3
##  [7,]    3    3    3    3    3
##  [8,]    3    1    2    3    3
##  [9,]    2    3    3    3    3
## [10,]    1    2    3    1    3
```

we now combine the clusters to the original scaled data frame and observe the results. Also in the process we round the variables to 2 digits for a better and consistent view of the final clustered and scaled data set.

```
df.arrests.scaled <- cbind(df.arrests.scaled,
                           clust = vscaled)

df.arrests.scaled[, 1:4] <- apply(X = df.arrests.scaled[, 1:4], MARGIN = 2, FUN = function(x) round(x, 

head(df.arrests.scaled)
```

```
##   Murder Assault UrbanPop  Rape      State clust
## 1   1.24    0.78    -0.52  0.00    Alabama     1
## 2   0.51    1.11    -1.21  2.48     Alaska     1
## 3   0.07    1.48     1.00  1.04    Arizona     2
## 4   0.23    0.23    -1.07 -0.18   Arkansas     3
## 5   0.28    1.26     1.76  2.07 California     2
## 6   0.03    0.40     0.86  1.86   Colorado     2
```

The distribution of the clusters after scaling can be seen as below.

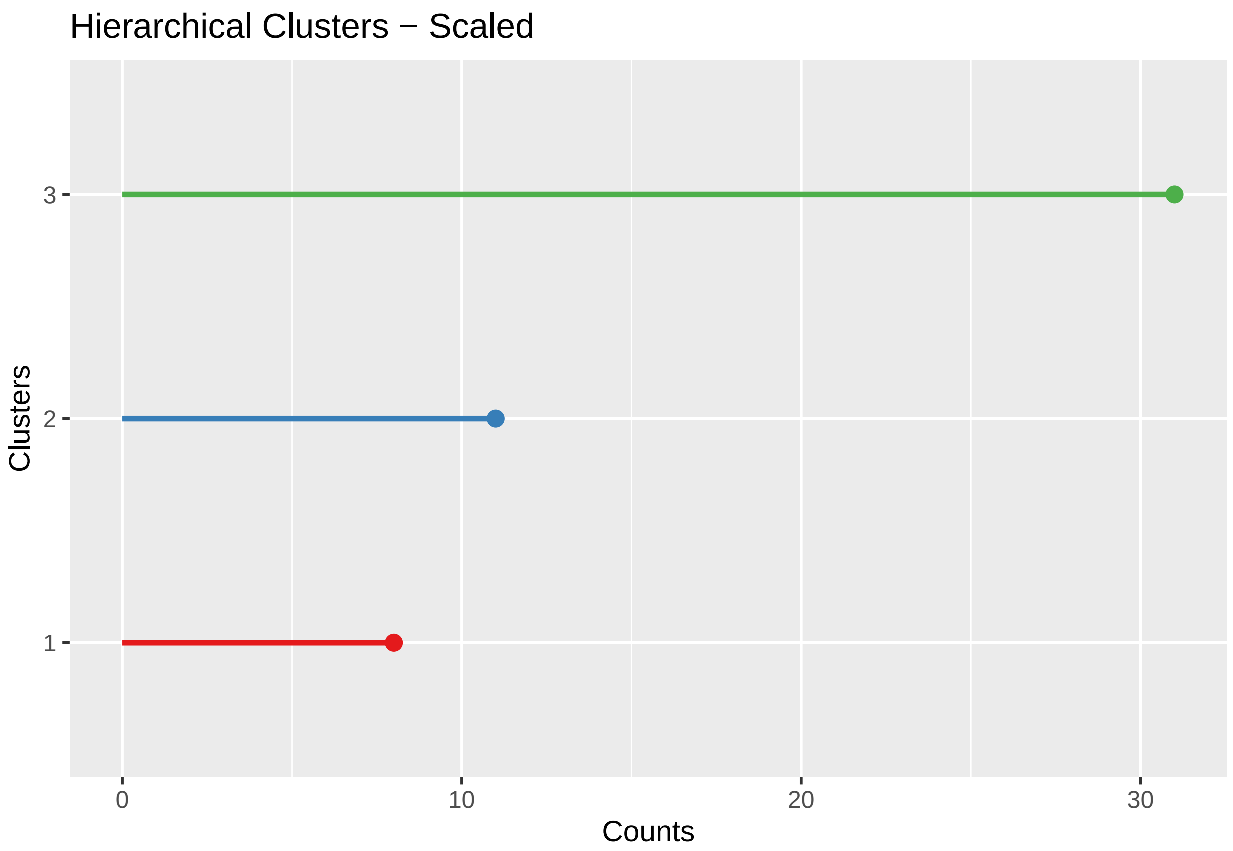
```
scaled.clust.dist <-  data.frame(clust = as.factor(unique(df.arrests.scaled$clust)),
           cnt = apply(X = as.matrix(unique(df.arrests.scaled$clust), drop = FALSE),
                       MARGIN = 1,
                       FUN = function(x) nrow(df.arrests.scaled[df.arrests.scaled$clust == x[1], ])))


scaled.clust.dist
```

```
##   clust cnt
## 1     1   8
## 2     2  11
## 3     3  31
```

The distributions of the clusters have changed and rightly so. Now 8 observations are in cluster 1, 11 are in cluster 2 and 31 observations are in cluster 3. This distribution is more correct representation of the data as all the variables have been scaled to the same dimension before applying hierarchical clustering algorithm.

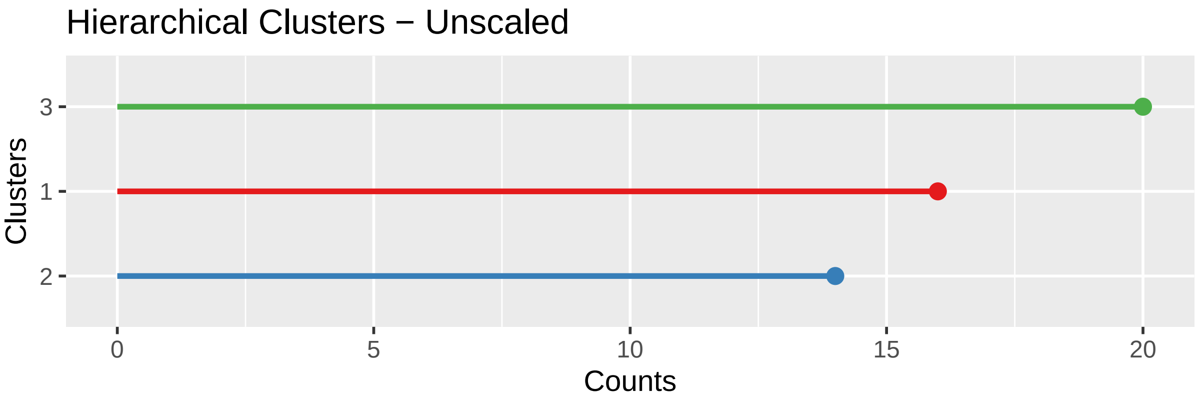
We can plot this distribution and check the results.

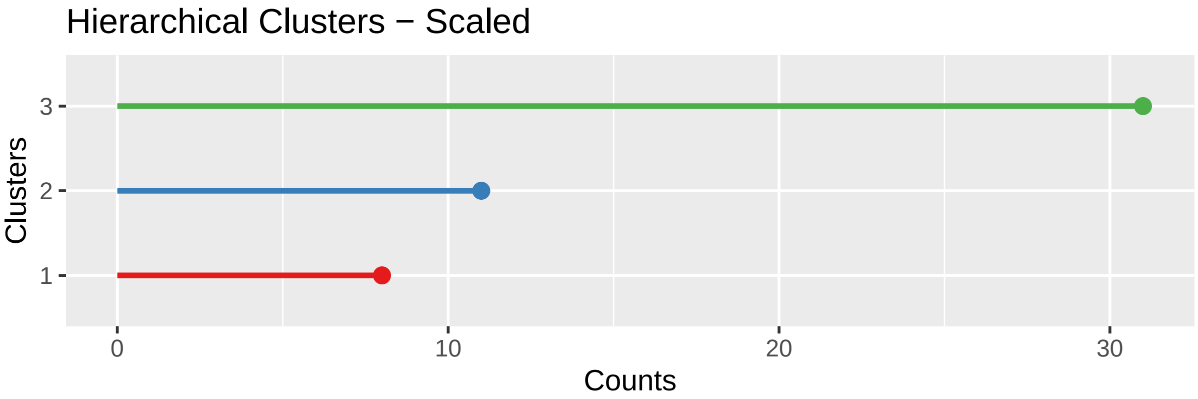
```
p_scaled <- ggplot(data = scaled.clust.dist) +
  geom_segment(mapping = aes(x = 0,
                             xend = cnt,
                             y =  reorder(clust, cnt),
                             yend = reorder(clust, cnt),
                             color = clust), size = 1.0,
               show.legend = FALSE) +
  geom_point(mapping = aes(x = cnt,
                           y = reorder(as.factor(clust), cnt),
                           color = clust), size = 2.5,
             show.legend = FALSE) +
  scale_fill_brewer(palette = "Set1") +
  scale_color_brewer(palette = "Set1") +
  labs(x = "Counts",
       y = "Clusters",
       title = "Hierarchical Clusters - Scaled")


p_scaled
```



A plot says a thousand words. We can even use the patchwork package to combine the above plots.

```
p_unscaled / p_scaled
```





This paper has demonstrated the implementation of Hierarchical Clustering on the US Arrests data set contained in Base R. The methods used are inspired from the book **An Introduction to Statistical Learning with Applications in R**. This is one of the exercises in the Applied section of the Chaper on "Unsupervised Learning".

Hope it is a good read and good tutorial for practicing Data Scientists !!!!