Clustering

Daniel Shang

```
# Load the packages necessary for the project
library(psych)
```

```
## Warning: package 'psych' was built under R version 4.0.3
```

```
library(ggfortify)
```

```
## Warning: package 'ggfortify' was built under R version 4.0.3
```

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

```
## Warning: package 'ggplot2' was built under R version 4.0.3
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.0.3
```

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

```
# In this case, we will use the 'iris' data set for illustration. A quick summary
## and the structure of the data set is shown.
data = iris
summary(data)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
```

```
##  versicolor:50
##  virginica :50
##
##
##
```

```
str(data)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

K-Means Clustering

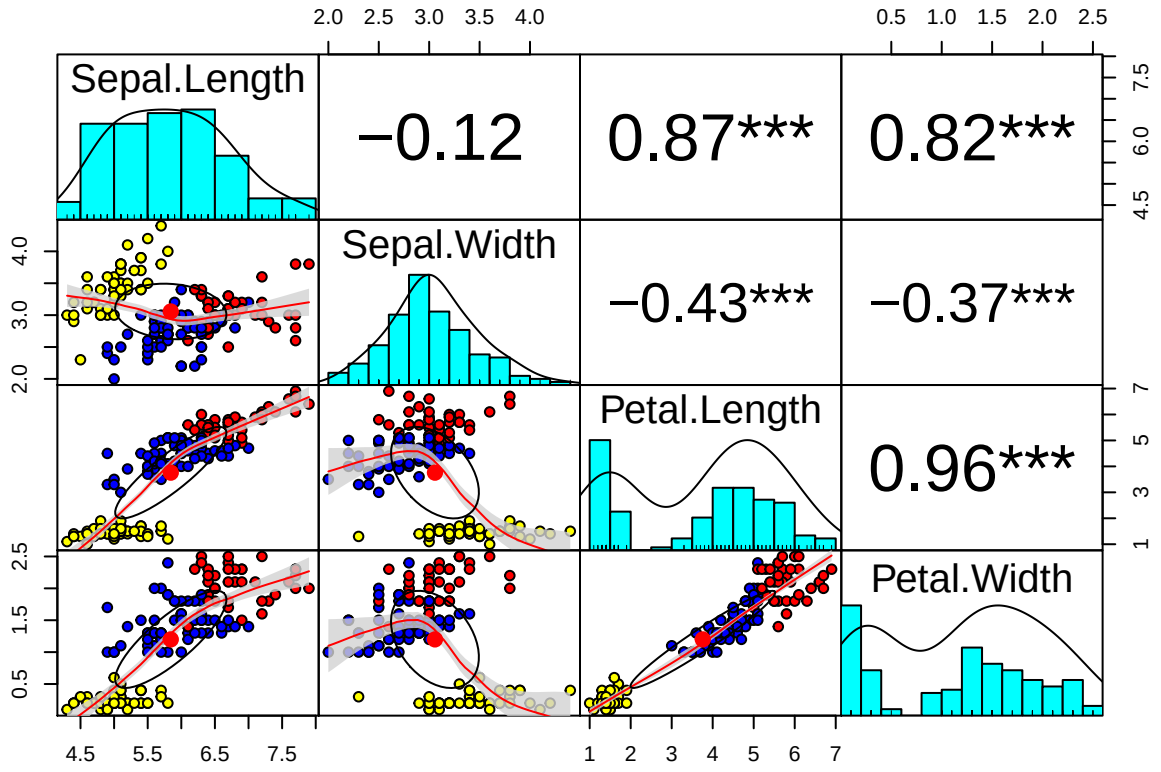
```
# We set a random seed and build a K-Means model. Since the target variable (Species)
## of the iris data set has three categories, we will use three as the value K
## for now. We will come back to check if it is a good choice.
set.seed(123)
kmeans1 = kmeans(iris[, -5], 3)
kmeans1
```

```
## K-means clustering with 3 clusters of sizes 50, 62, 38
##
## Cluster means:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     5.006000    3.428000     1.462000    0.246000
## 2     5.901613    2.748387     4.393548    1.433871
## 3     6.850000    3.073684     5.742105    2.071053
##
## Clustering vector:
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3 3 2 3 3 3 3
## [112] 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2 2 3 3 3 3 2 3 3 3 3 2 3 3 3 2 3 3 3 2 3
## [149] 3 2
##
## Within cluster sum of squares by cluster:
## [1] 15.15100 39.82097 23.87947
##  (between_SS / total_SS =  88.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# Here, we build a table showing the three species categories and the three clusters
## determined by K-Means model. According to the table, we can see that the model
## correctly clustered 'setosa' and did a fairly good job clustering 'versicolor.'
## However, the model seems to be not sure about which cluster 'virginica' should
## be put in.
table(iris$Species, kmeans1$cluster)
```

```
##
##                1  2  3
##   setosa      50  0  0
##   versicolor   0 48  2
##   virginica    0 14 36
```

```
# Here, we visualize the data and color each point based on the cluster it is
## assigned to. We can see that for those pairs of variables that are not highly
## correlated, some points mingle and drawing a clear line is difficult. To
## interpret this plot, we can see that, whenever 'Sepal.Width' comes in, it is
## very close to another cluster.For example, in the graph at the second column
## and third row, the red and blue clusters are very close to each other in terms
## of the distance on the y-axis. Since 'Sepal.Width' is on the y-axis, we know that
## this variable probably contributes to the bad clustering. This is also shown in
## the model result from previous chunk, in which we see that the cluster means of
## the three cluster for 'Sepal.Width' are close to each other.
pairs.panels(x = iris[, -5], gap = 0, bg = c('yellow', 'blue', 'red')[kmeans1$cluster],
             pch = 21, stars = TRUE, ci = TRUE, alpha = 0.1)
```

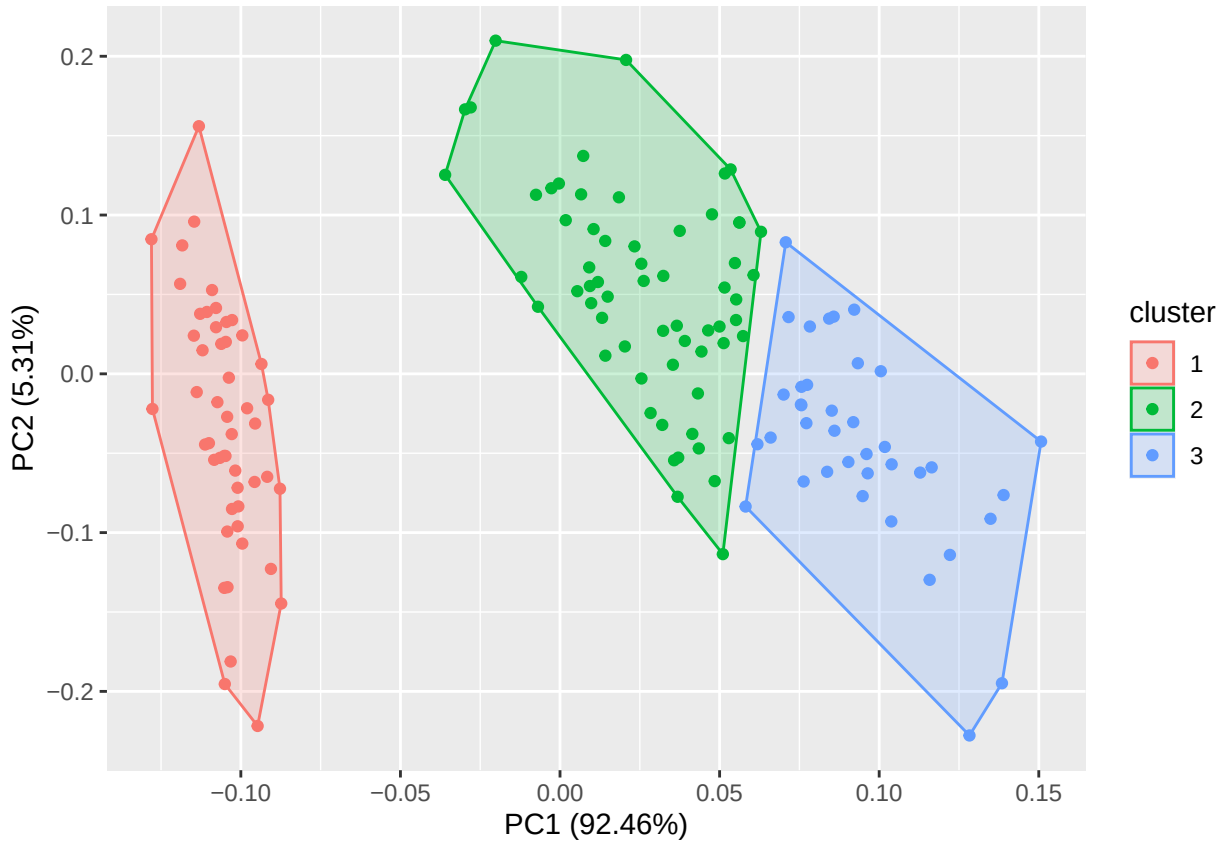



```
# If we plot the three clusters on a graph, we see that the green and blue clusters
## are very close to each other, and they could potentially be clustered in the same
## cluster.
autoplot(kmeans1, iris[, -5], frame = TRUE)
```

```
## Warning: `select_()` is deprecated as of dplyr 0.7.0.
```

```
## Please use ‘select()‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_warnings()‘ to see where this warning was generated.

## Warning: ‘group_by_()‘ is deprecated as of dplyr 0.7.0.
## Please use ‘group_by()‘ instead.
## See vignette('programming') for more help
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_warnings()‘ to see where this warning was generated.
```




```
# As we can see, the center of 'Sepal.Width' is very close to each other.
kmeans1$centers
```

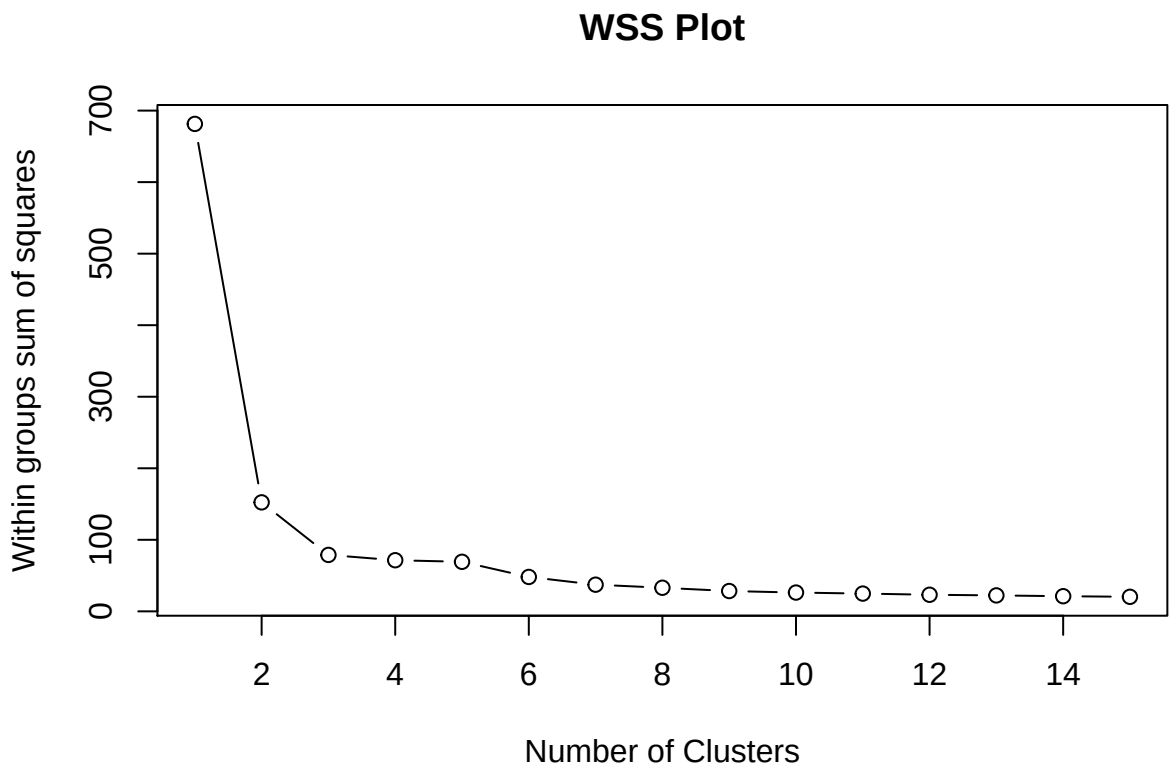
```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     5.006000    3.428000     1.462000    0.246000
## 2     5.901613    2.748387     4.393548    1.433871
## 3     6.850000    3.073684     5.742105    2.071053
```

```
# To figure out if other K could have been the better choice, we use the WSS Plot,
## also known as an elbow plot.
## This source of this function can be found at: https://www.r-statistics.com/2013/08/k-means-clusterin

wssplot <- function(data, nc=15, seed=123){
               wss <- (nrow(data)-1)*sum(apply(data,2,var))
```

```
                for (i in 2:nc){
                     set.seed(seed)
                     wss[i] <- sum(kmeans(data, centers=i)$withinss)}
                 plot(1:nc, wss, type="b", xlab="Number of Clusters",
                      ylab="Within groups sum of squares", main = 'WSS Plot')}
```

```
# The plot shows the variance of each value of K. After K equals 2, the slope of
## line is not as high as before. This means that a K equals 2 could be a better
## choice. However, this judgment is subjective, as the slope when K equals 3 is
## also distinguishable from the slopes when K is greater than 3. We will use K
## equals 2 and see how the model performs.
wssplot(iris[, -5])
```




```
# We repeat the same process as before except that we use an K equals 2 this time.
set.seed(123)
kmeans2 = kmeans(iris[, -5], 2)
kmeans2
```

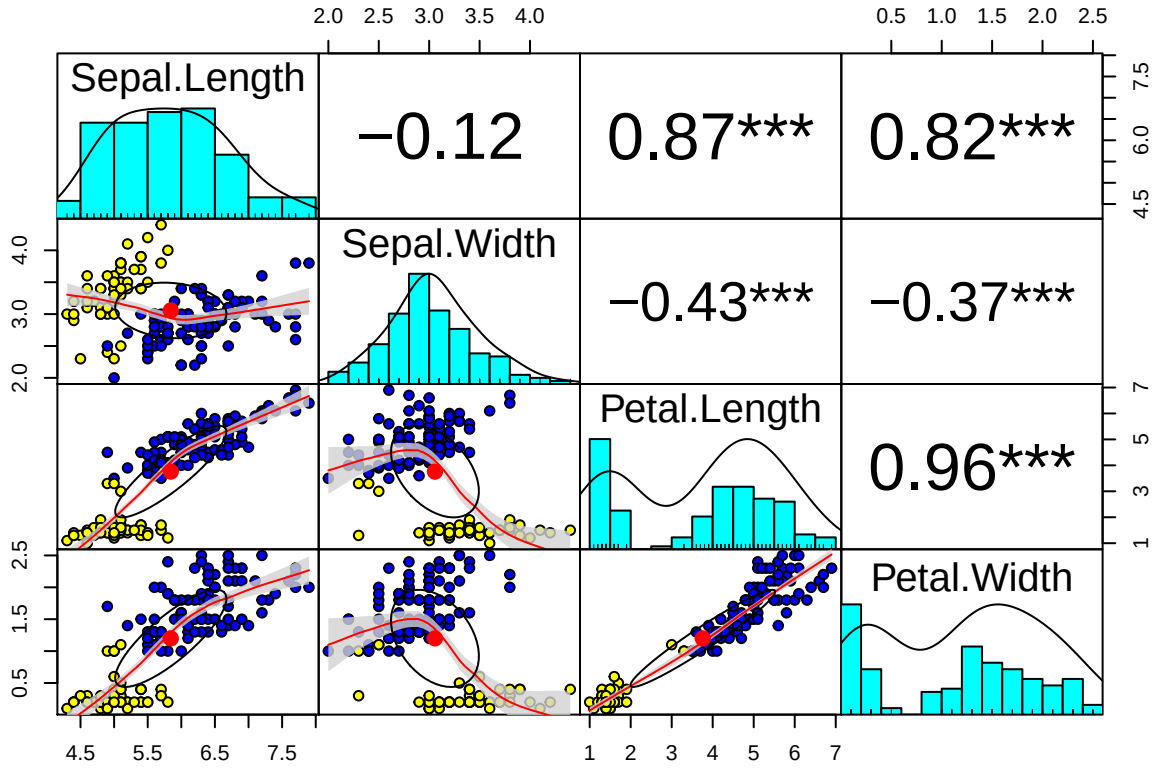
```
## K-means clustering with 2 clusters of sizes 53, 97
##
## Cluster means:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     5.005660    3.369811     1.560377    0.290566
## 2     6.301031    2.886598     4.958763    1.695876
##
```

```
## Clustering vector:
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2
## [112] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [149] 2 2
##
## Within cluster sum of squares by cluster:
## [1]  28.55208 123.79588
##  (between_SS / total_SS =  77.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# Putting the clustering result in a table, we do see that each cluster is 'purer.'
## The model put 'versicolor' and 'virginica' in the same cluster.
table(iris$Species, kmeans2$cluster)
```

```
##
##                1  2
##   setosa      50  0
##   versicolor   3 47
##   virginica    0 50
```

```
pairs.panels(x = iris[, -5], gap = 0, bg = c('yellow', 'blue')[kmeans2$cluster],
             pch = 21, stars = TRUE, ci = TRUE, alpha = 0.1)
```




```
autoplot(kmeans2, iris[, -5], frame = TRUE)
```


```
# As shown by all the plots and statistics, this way of clustering is also acceptable.
## When putting the K-Means Clustering into practice, we should use domain knowledge
## to make this kind of choice. Does clustering 'versicolor' and 'virginica' make
## sense? Are they biologically similar to each other? Knowing these can help
## analysts to make the proper decision of choosing K.
kmeans2$centers
```

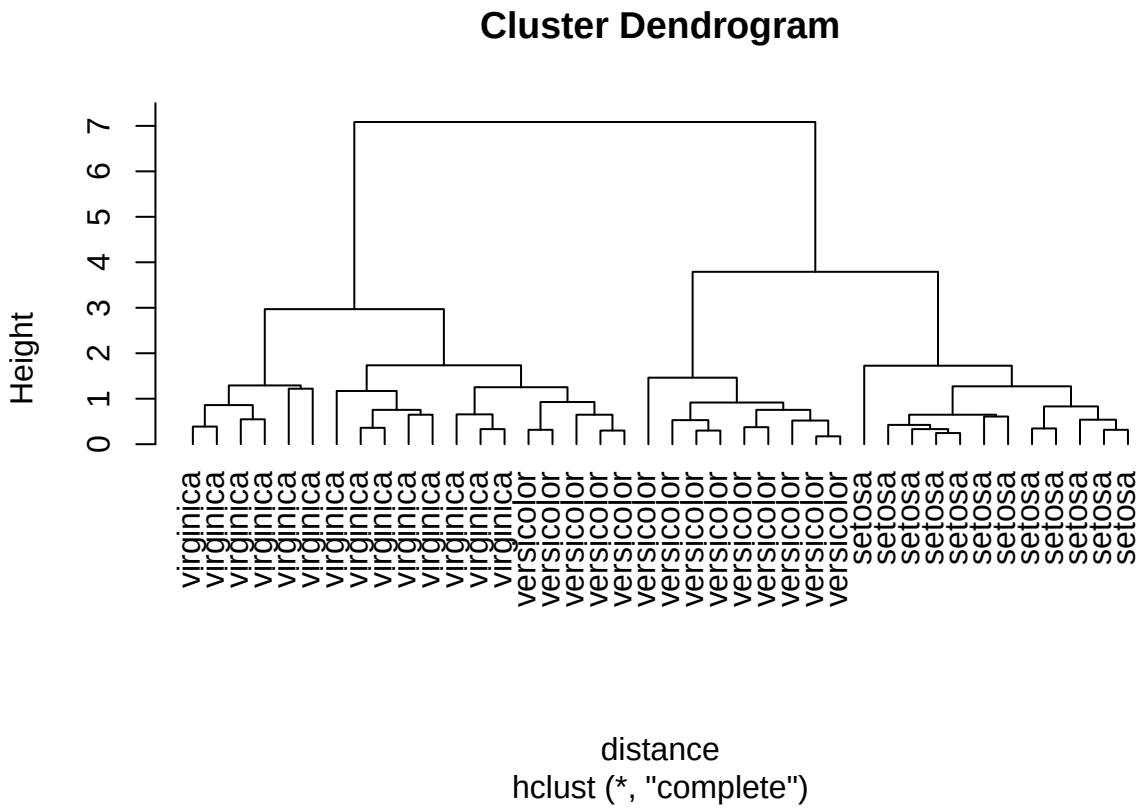
```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     5.005660    3.369811     1.560377    0.290566
## 2     6.301031    2.886598     4.958763    1.695876
```

Hierarchical Clustering

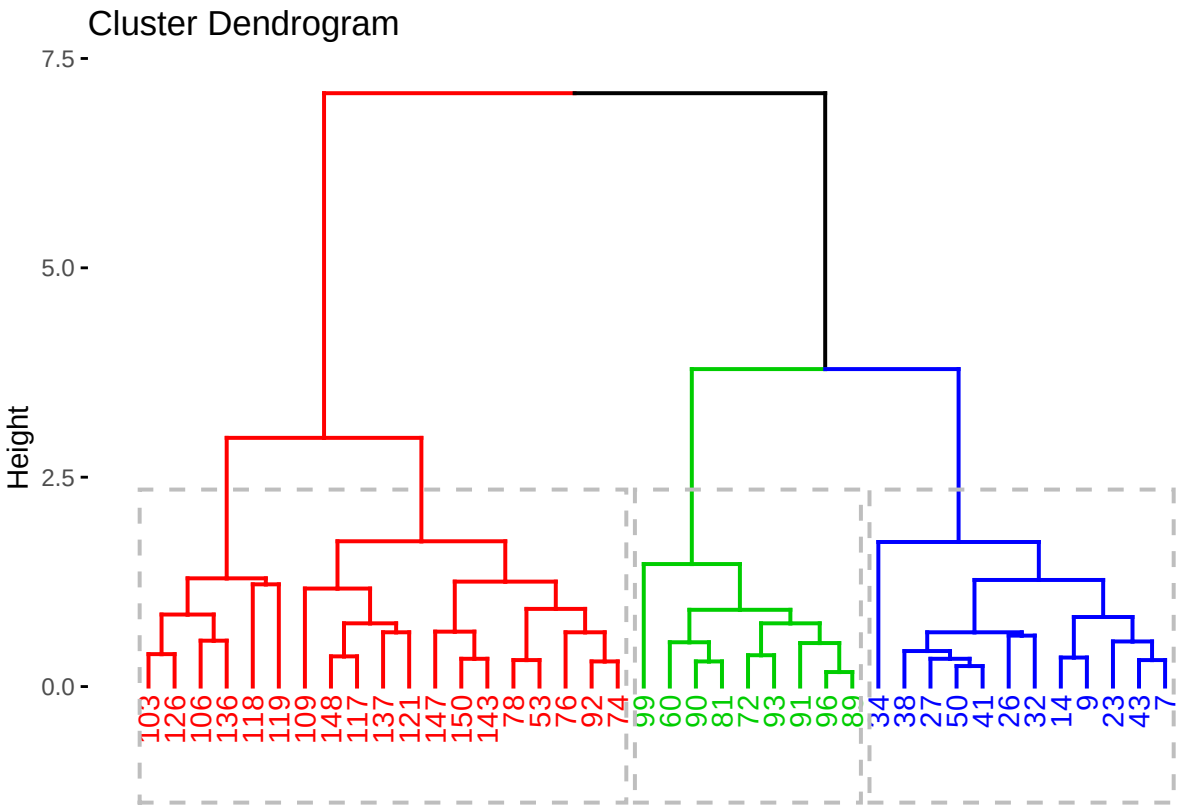
```
# Set a random seed and randomly select 40 samples as the data set to be used in
## this example
set.seed(123)
index_1 = sample(150, nrow(iris), replace = FALSE)
iris_sample = iris[index_1[1:40], ]
```

```
# Hierarchical clustering model clusters data based on the euclidian distance between
## points. Thus, we first compute the distance between points and feed the model
## with the distance calculated. Then, we make a plot to show how the model cluster
## the data. We also add the species name onto the plot to take a glance at the
## model's performance.
distance = dist(iris_sample[, -5])
```

```
hc1 = hclust(distance)
plot(hc1, labels = iris_sample$Species, hang = -1)
```




```
# Unlike the situation with K-Means, determining K for hierarchical clustering
## relies more on domain knowledge. Since there are three species in the data set,
## we will go with a K equals to three here. We reflect the size of K on the graph.
fviz_dend(x = hc1, k = 3, k_colors = c('red', 'green3', 'blue'),
          rect = TRUE, rect_border = 'gray')
```




```
# Here, we cut the model into three pieces because we choose three as our size of K.
## Then, we can look at the detail of each attribute within each cluster. For example,
## the first and second row indicate that, in the first cluster, the average
## 'Petal.Length' is 1.366667. With this data, we get a better understanding about
## how different attribute looks like within a cluster.
clusterGroups = cutree(hc1, k = 3)
tapply(iris_sample$Petal.Length, clusterGroups, mean)
```

```
##        1        2        3
## 1.366667 5.515789 3.933333
```

```
tapply(iris_sample$Petal.Width, clusterGroups, mean)
```

```
##         1         2         3
## 0.2333333 1.8894737 1.2333333
```

```
tapply(iris_sample$Sepal.Length, clusterGroups, mean)
```

```
##        1        2        3
## 4.841667 6.752632 5.555556
```

```
tapply(iris_sample$Sepal.Width, clusterGroups, mean)
```

```
##        1        2        3
## 3.375000 2.989474 2.677778
```

```r
# Here, we check out what is cluster by the model as cluster number three. We can
## see that this cluster is solely consisted of 'versicolor' species. In application,
## the clustering analysis can be used to, for example, segment customers based on
## their characteristics, so that we have a better idea about how to design different
## marketing campaign to target different segment of customers.
subset(iris_sample, clusterGroups == 3)
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 90          5.5         2.5          4.0         1.3 versicolor
## 91          5.5         2.6          4.4         1.2 versicolor
## 99          5.1         2.5          3.0         1.1 versicolor
## 72          6.1         2.8          4.0         1.3 versicolor
## 81          5.5         2.4          3.8         1.1 versicolor
## 60          5.2         2.7          3.9         1.4 versicolor
## 96          5.7         3.0          4.2         1.2 versicolor
## 89          5.6         3.0          4.1         1.3 versicolor
## 93          5.8         2.6          4.0         1.2 versicolor
```