INFS 692 Data Science Final Project: Model 3

Chanpreet Kaur

2022-12-16

Model 3- Unsupervised Learning

In this model we attempt Unsupervised Learning as we will not include target variable. So without considering the *binary output* and *categorical variables* in the dataset, we would compare three clustering technique results: from K-Means, Hierarchial and Model Based.

All code in this file is referenced from week 10 lecture class and week10 assignment.

Importing required libraries

```
# import libraries
##################
# Helper packages
library(dplyr)       # for data manipulation
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)     # for data visualization
library(stringr)     # for string functionality
library(gridExtra)   # for manipulaiting the grid
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(tidyverse)  # data manipulation
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v tibble  3.1.8     v purrr   0.3.4
## v tidyr   1.2.1     v forcats 0.5.2
## v readr   2.1.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x gridExtra::combine() masks dplyr::combine()
## x dplyr::filter()      masks stats::filter()
## x dplyr::lag()         masks stats::lag()
```

```
library(cluster)     # for general clustering algorithms
library(factoextra)  # for visualizing cluster results
```

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

```
library(mclust)   # for fitting clustering algorithms
```

```
## Package 'mclust' version 6.0.0
## Type 'citation("mclust")' for citing this R package in publications.
##
## Attaching package: 'mclust'
##
## The following object is masked from 'package:purrr':
##
##     map
```

```
# read data file
data <-  read.csv('./radiomics_completedata.csv')
```

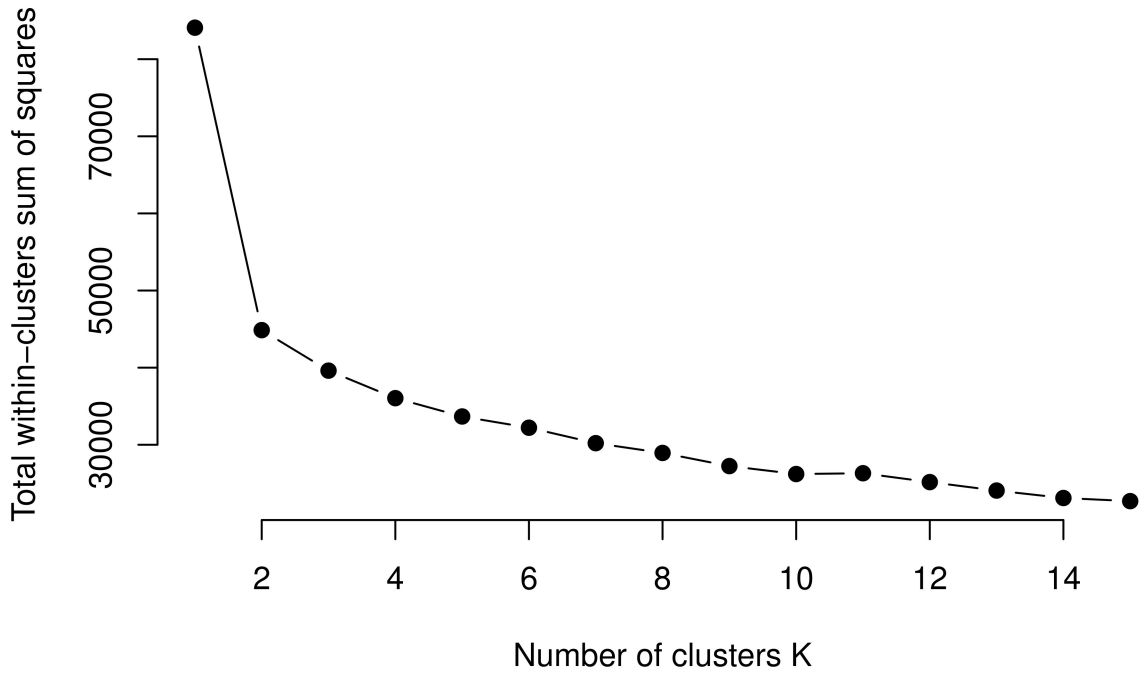
********** K-Means clustering **********

Reference code from week10 assignment

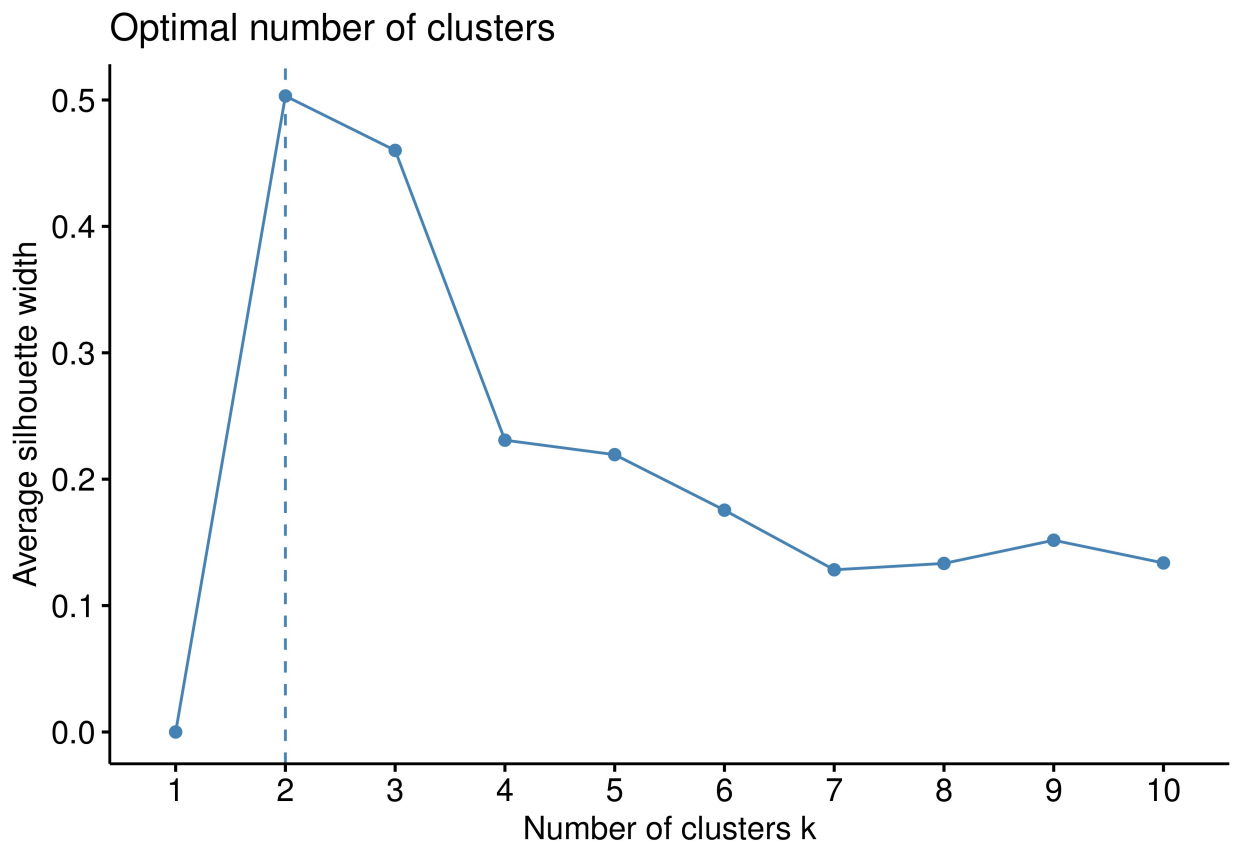
```
#Data Pre-processing
#focus on numeric data
num <- sapply(data, is.numeric) #getting rid of categorical

data <- data[num]

data <- Filter(function(x) !all(x %in% c(0, 1)), data)

# Checking for null values
data <- na.omit(data)
final_data <- scale(data)

final_data <- as.data.frame(final_data)
```

Once data is processed, we can initiate the number of clusters for the K-Means Clustering and compute the statistics with optimal clusters

```r
# Determining Optimal Number of Clusters
set.seed(123)

#function to compute total within-cluster sum of square
wss <- function(k) {
  kmeans(final_data, k, nstart = 10)$tot.withinss
}

# Compute and plot wss for k = 1 to k = 15
k.values <- 1:15

# extract wss for 2-15 clusters
wss_values <- map_dbl(k.values, wss)

plot(k.values, wss_values,
     type="b", pch = 19, frame = FALSE,
     xlab="Number of clusters K",
     ylab="Total within-clusters sum of squares")
```



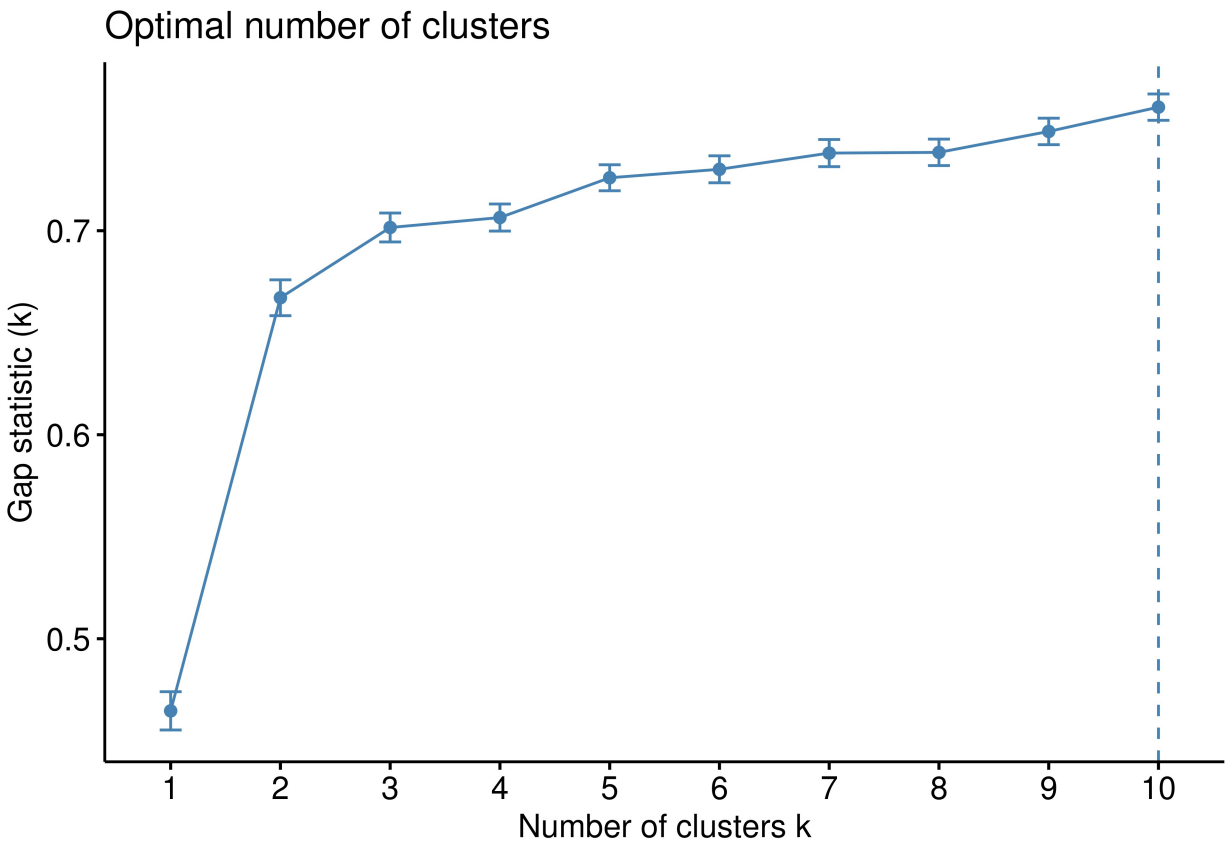

```r
#We can also use below silhouette method to evaluate
fviz_nbclust(final_data, kmeans, method = "silhouette")
```



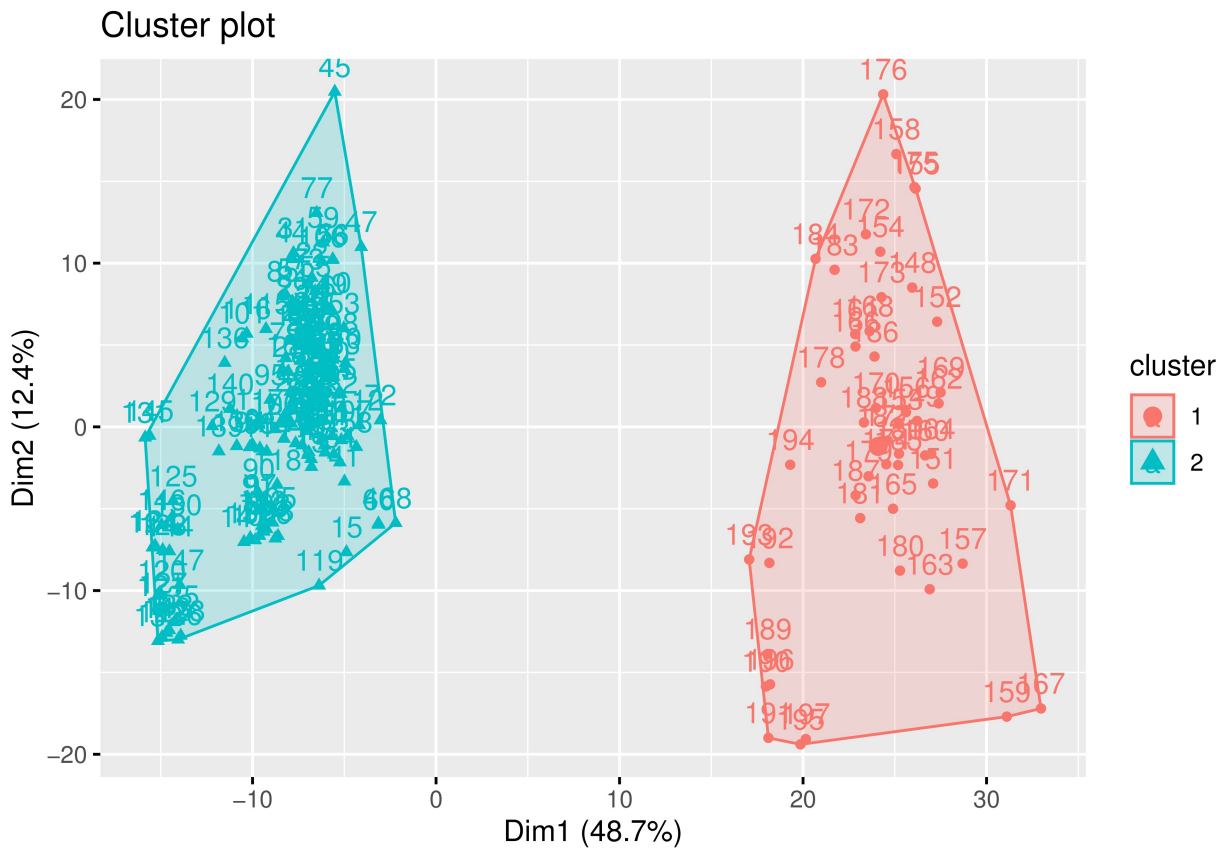

```
# compute gap statistic
set.seed(123)
gap_stat <- clusGap(final_data, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
# Print the result
print(gap_stat, method = "firstmax")
```

```
## Clustering Gap statistic ["clusGap"] from call:
## clusGap(x = final_data, FUNcluster = kmeans, K.max = 10, B = 50, nstart = 25)
## B=50 simulated reference sets, k = 1..10; spaceH0="scaledPCA"
##  --> Number of clusters (method 'firstmax'): 10
##           logW   E.logW       gap      SE.sim
##  [1,] 7.171204 7.635853 0.4646496 0.009379996
##  [2,] 6.879524 7.546674 0.6671493 0.008786338
##  [3,] 6.798848 7.500436 0.7015873 0.007082545
##  [4,] 6.760004 7.466467 0.7064633 0.006632270
##  [5,] 6.715614 7.441579 0.7259645 0.006374244
##  [6,] 6.689522 7.419633 0.7301115 0.006603869
##  [7,] 6.661683 7.399745 0.7380616 0.006654018
##  [8,] 6.643211 7.381624 0.7384134 0.006480643
##  [9,] 6.616471 7.365139 0.7486677 0.006484664
## [10,] 6.588968 7.349544 0.7605765 0.006453097
```

```
fviz_gap_stat(gap_stat)
```




```
# Compute k-means clustering with k = 2
set.seed(123)
KMeans <- kmeans(final_data, 2, nstart = 25)
#print(KMeans)

#final cluster data plot
fviz_cluster(KMeans, data = final_data)
```

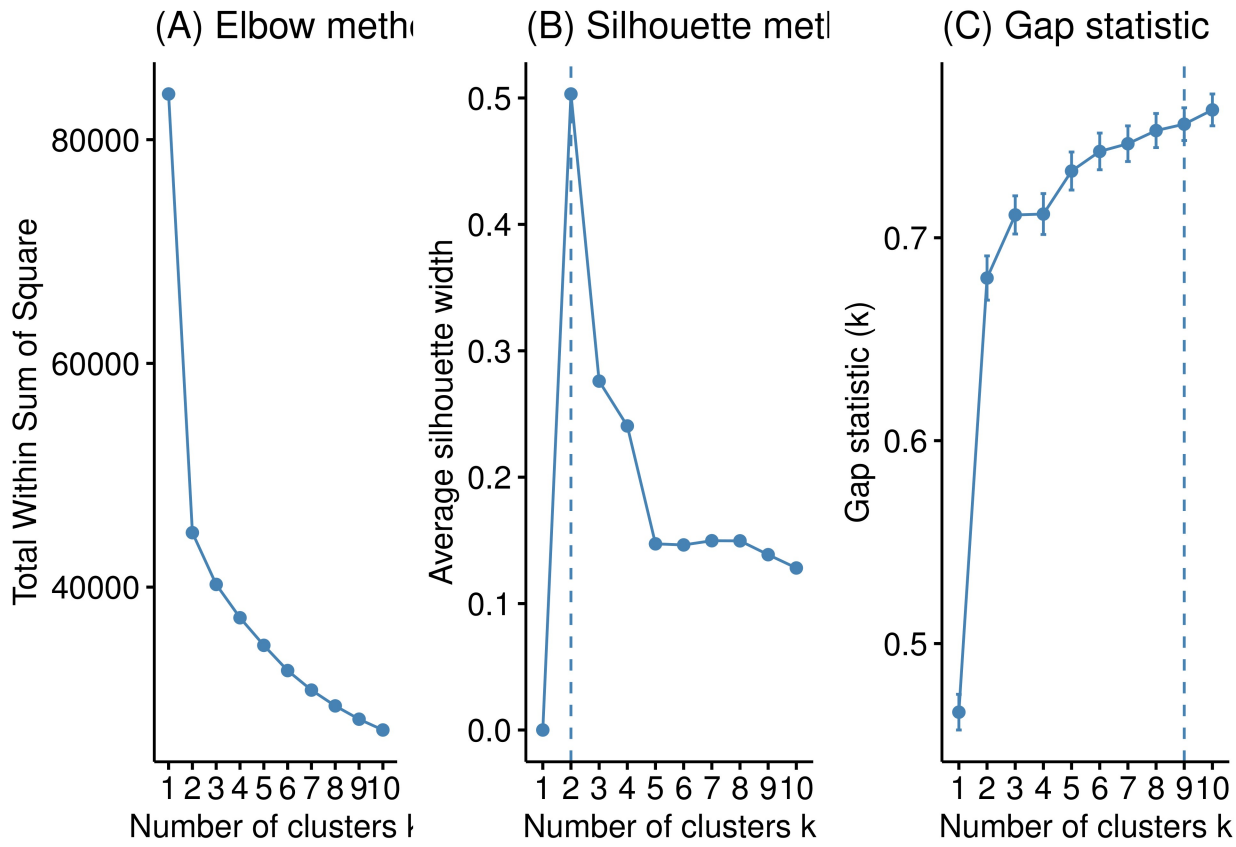


********** Hierarchical Clustering **********

Reference code from week10 assignment

In heirarchial clustering, we do not pre-specify the number of clusters. An advantage of this type of clustering is that we can visualize results using dendrograms.

```
# Dissimilarity matrix
d <- dist(final_data, method = "euclidean")

# Plot cluster results
p1 <- fviz_nbclust(final_data, FUN = hcut, method = "wss",
                   k.max = 10) +
  ggtitle("(A) Elbow method")
p2 <- fviz_nbclust(final_data, FUN = hcut, method = "silhouette",
                   k.max = 10) +
  ggtitle("(B) Silhouette method")
p3 <- fviz_nbclust(final_data, FUN = hcut, method = "gap_stat",
                   k.max = 10) +
  ggtitle("(C) Gap statistic")

# Display plots side by side for Elbow, silhouette and gap statistic
gridExtra::grid.arrange(p1, p2, p3, nrow = 1)
```




```
# Construct dendrogram for the given data
hc5 <- hclust(d, method = "ward.D2" )
dend_plot <- fviz_dend(hc5)
```

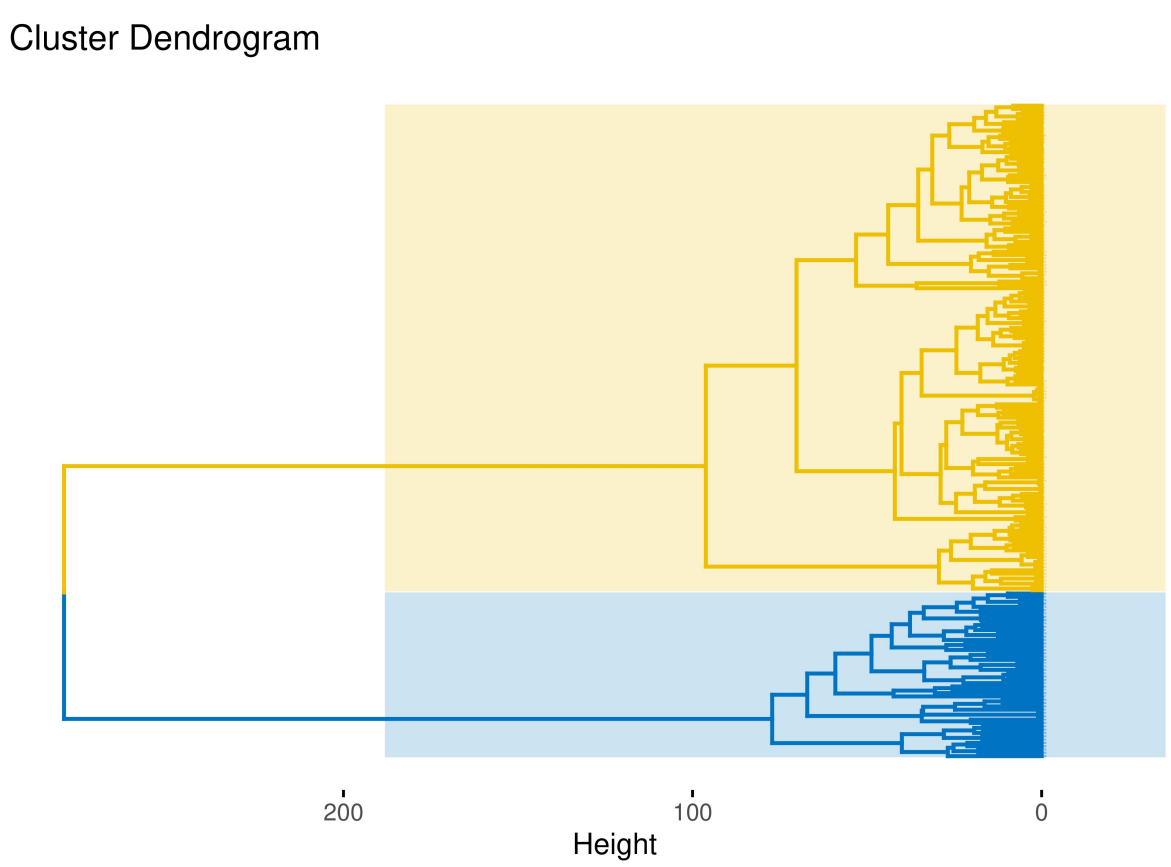
```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
dend_data <- attr(dend_plot, "dendrogram")
dend_cuts <- cut(dend_data, h = 2)

# Ward's method
hc5 <- hclust(d, method = "ward.D2" )

# Identify clusters by Cut tree into 2 groups
sub_grp <- cutree(hc5, k = 2)

# Number of members in the 2 clusters
table(sub_grp)
```

```
## sub_grp
##   1   2
## 147  50
```

```
# Plot full dendrogram
fviz_dend(
  hc5,
  k = 2,
  horiz = TRUE,
  rect = TRUE,
  rect_fill = TRUE,
  rect_border = "jco",
  k_colors = "jco",
  cex = 0.1
)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
#Since dendrogram is not legible, zooming in 1 cluster (reference code from Slide 33 week 10 lecture)
dend_plot <- fviz_dend(hc5)                # create full dendogram
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
dend_data <- attr(dend_plot, "dendrogram") # extract plot info
dend_cuts <- cut(dend_data, h = 70.5)      # cut the dendogram at
# designated height
```

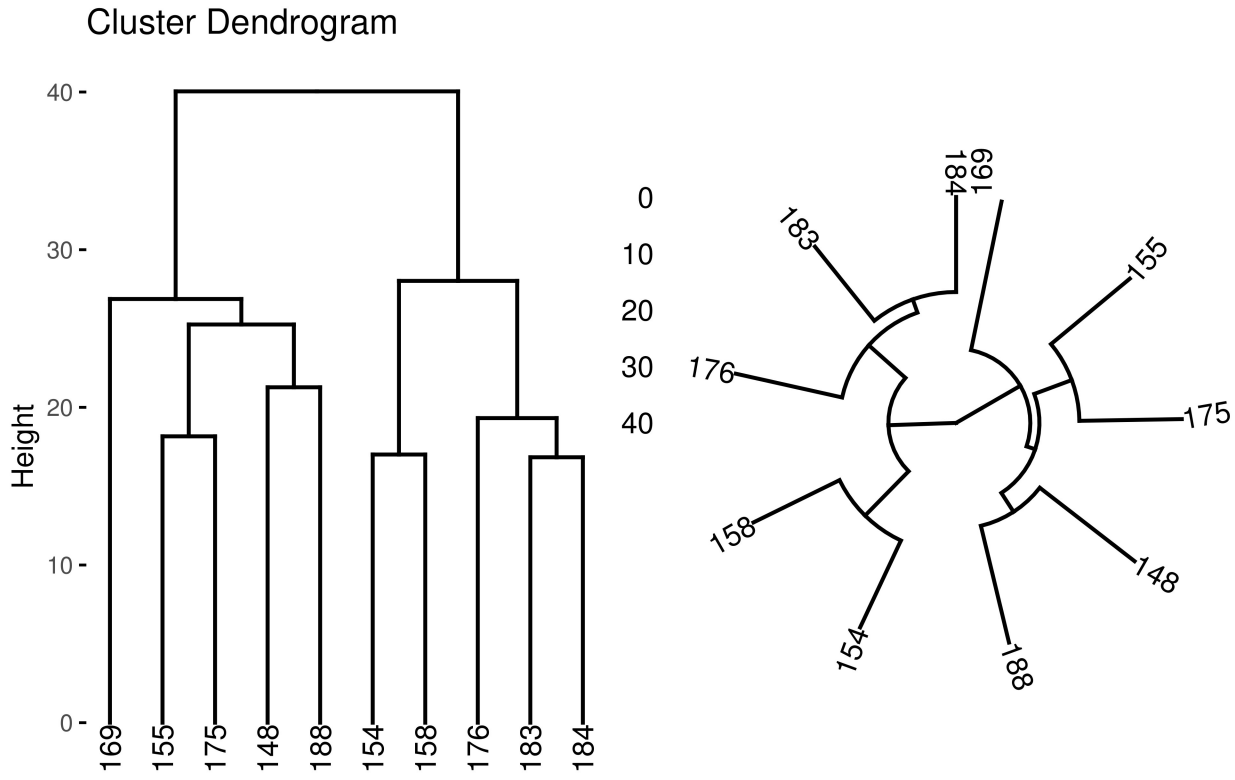
```
# Create sub dendrogram plots
p1 <- fviz_dend(dend_cuts$lower[[1]])
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
p2 <- fviz_dend(dend_cuts$lower[[1]], type = 'circular')
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

```
# Side by side plots
gridExtra::grid.arrange(p1, p2, nrow = 1)
```

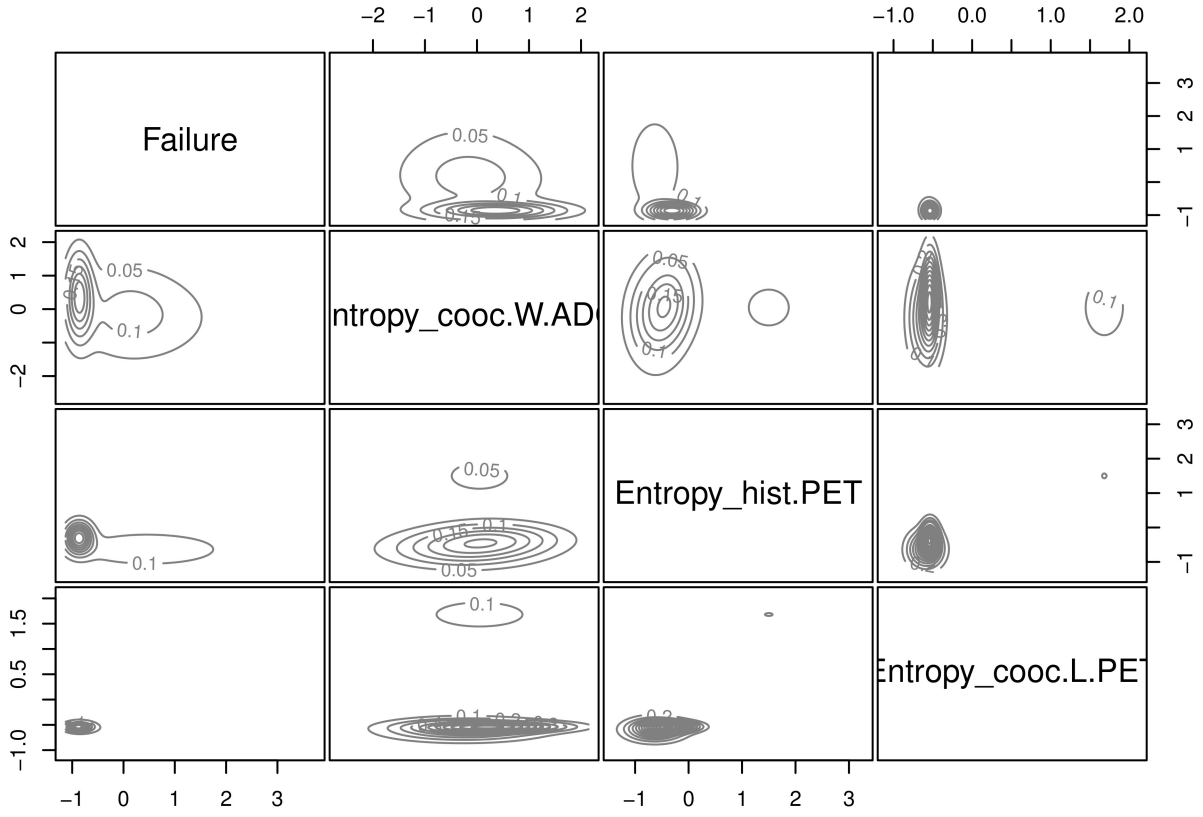


********** Model-based Clustering **********

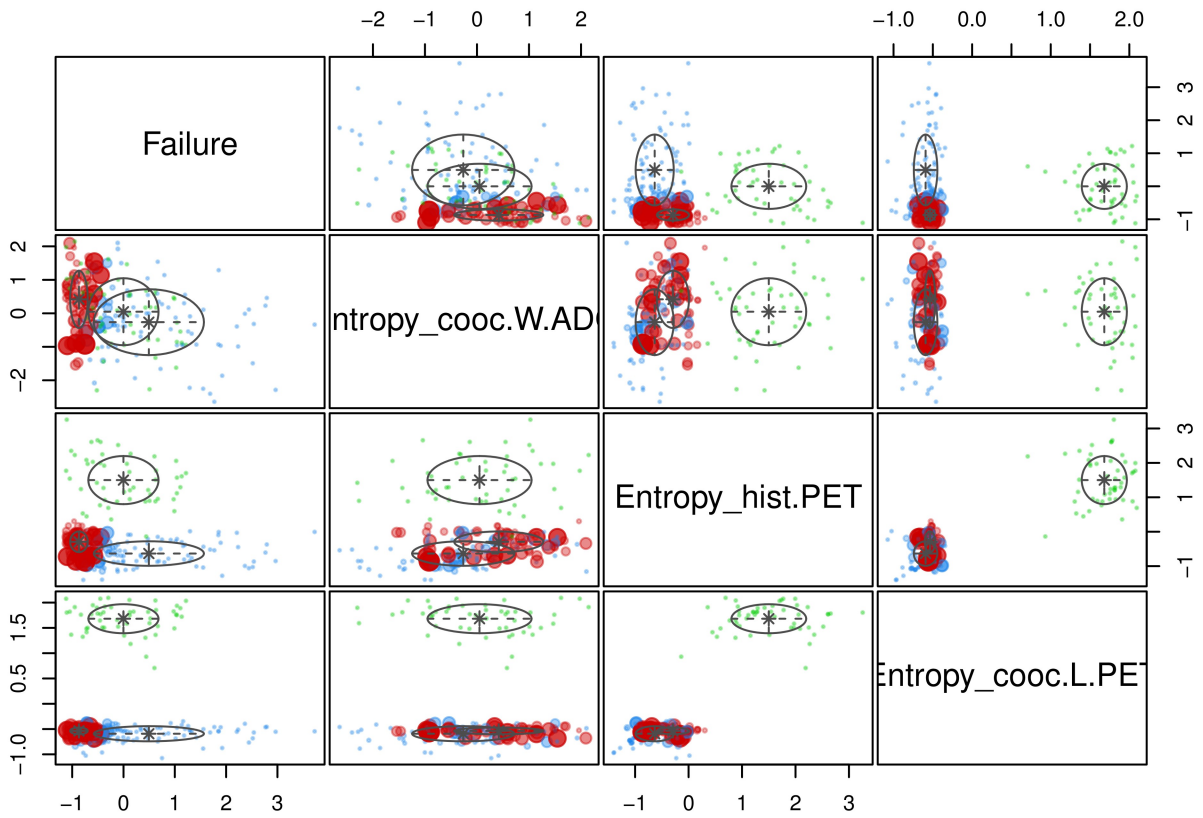
Reference code from week10 assignment

```
# Apply GMM model with 3 components
df_mdCluster <- select(final_data, Failure, Entropy_cooc.W.ADC, Entropy_hist.PET, Entropy_cooc.L.PET)
Md_cluster <- Mclust(df_mdCluster, G = 3)

# Plot results
```

```
par(mar=c(1,1,1,1))
plot(Md_cluster, what = "density")
```




```
plot(Md_cluster, what = "uncertainty")
```




```
# Observations with high uncertainty
sort(Md_cluster$uncertainty, decreasing = TRUE) %>% head()
```

```
##       116       101        79        43         4       139
## 0.4391194 0.3439769 0.3436707 0.3222617 0.3128399 0.3021089
```

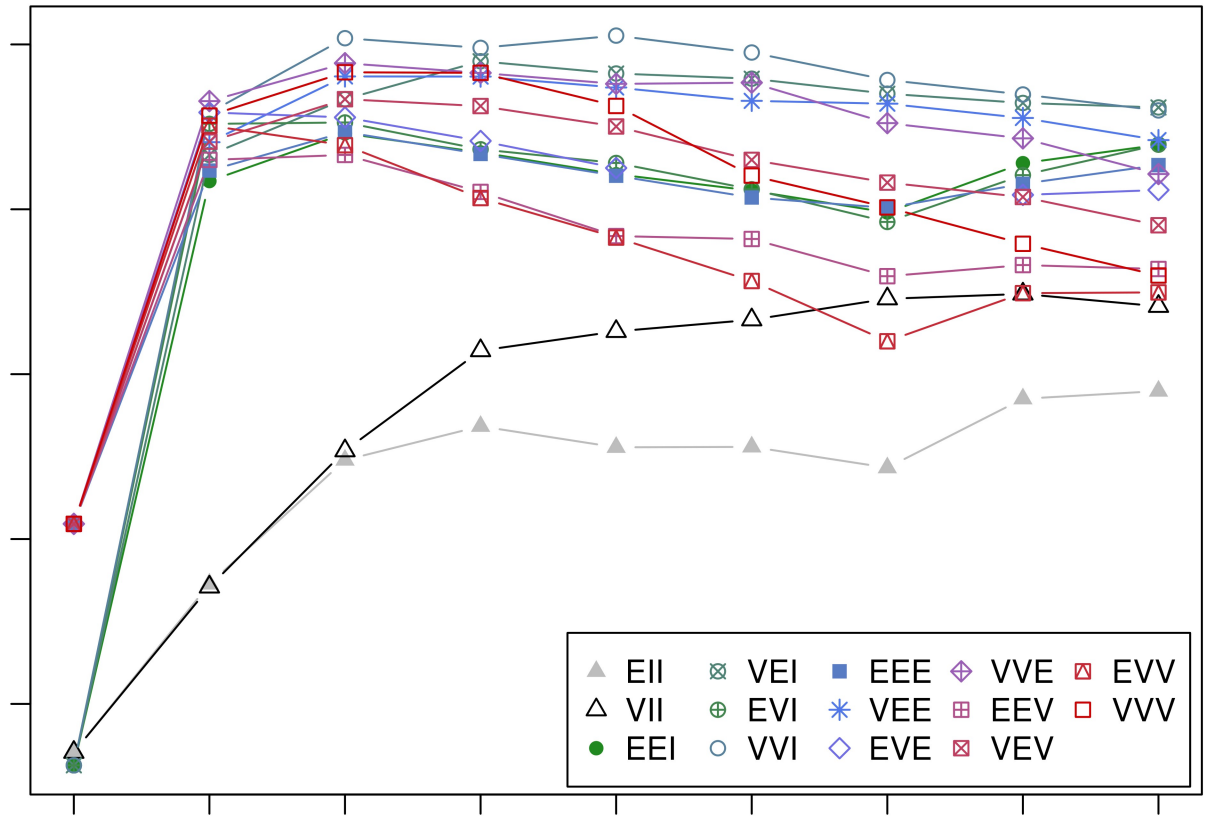
```
summary(Md_cluster)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVI (diagonal, varying volume and shape) model with 3 components:
##
##  log-likelihood   n df       BIC       ICL
##       -627.5309 197 26 -1392.425 -1407.834
##
## Clustering table:
##  1  2  3
## 89 58 50
```

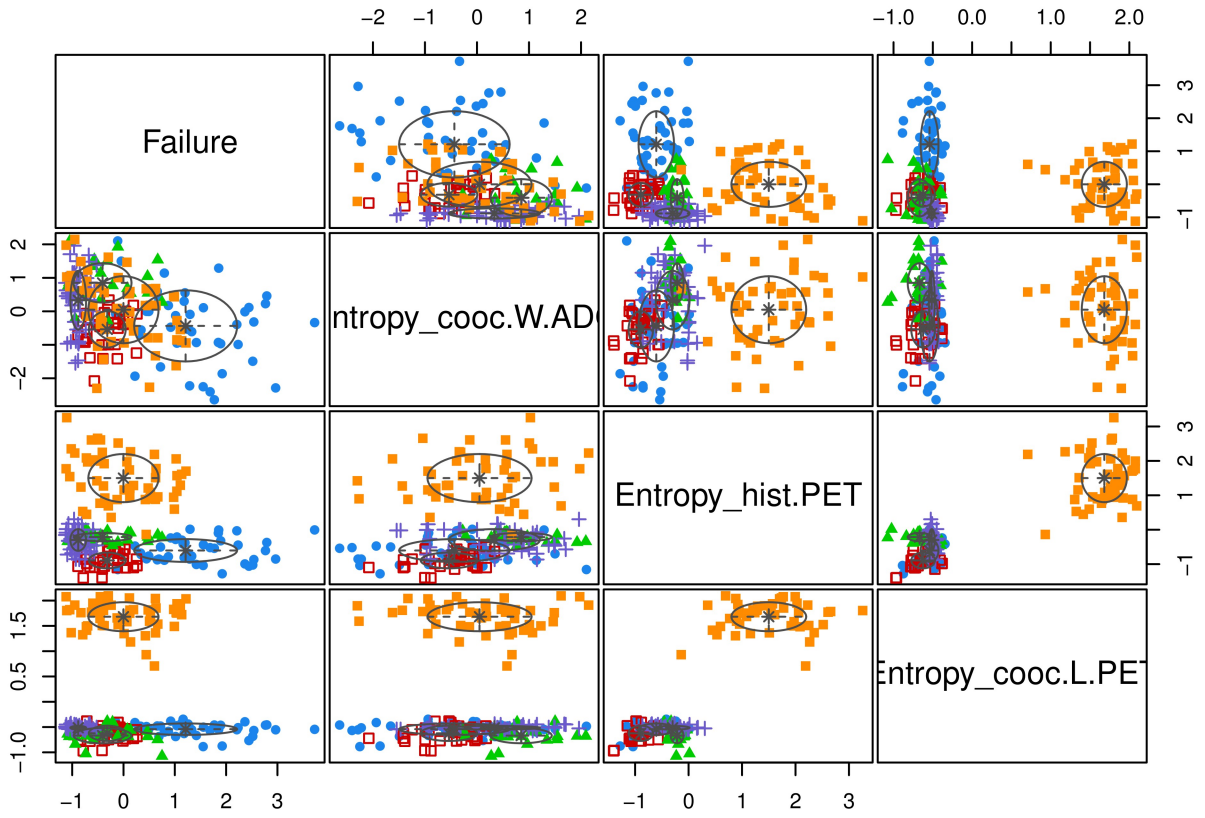
```
opt_mdCluster <- Mclust(df_mdCluster)

summary(opt_mdCluster)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVI (diagonal, varying volume and shape) model with 5 components:
##
##  log-likelihood   n df       BIC       ICL
##       -578.4222 197 44 -1389.305 -1422.034
##
## Clustering table:
##  1  2  3  4  5
## 43 33 24 47 50
```

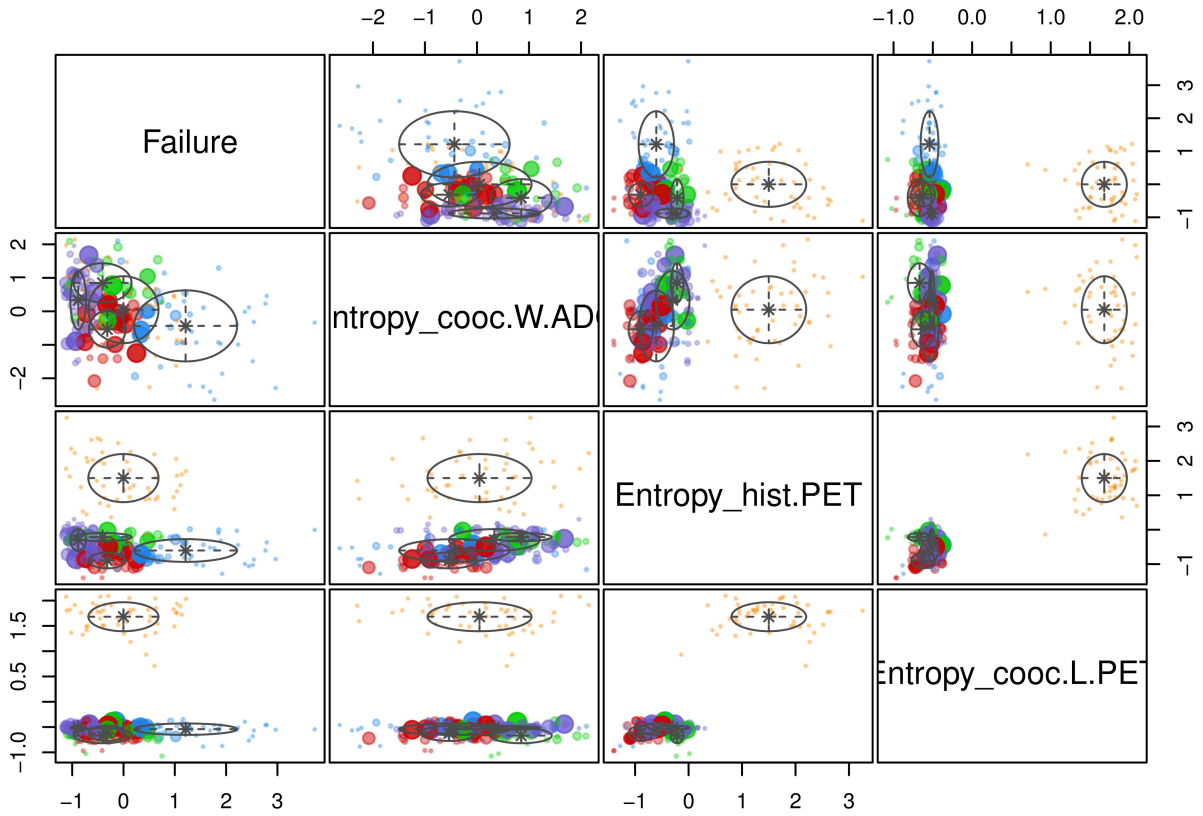
```r
# Model selection BIC
legend_args <- list(x = "bottomright", ncol = 5)
plot(opt_mdCluster, what = 'BIC', legendArgs = legend_args)
```




```r
plot(opt_mdCluster, what = 'classification')
```



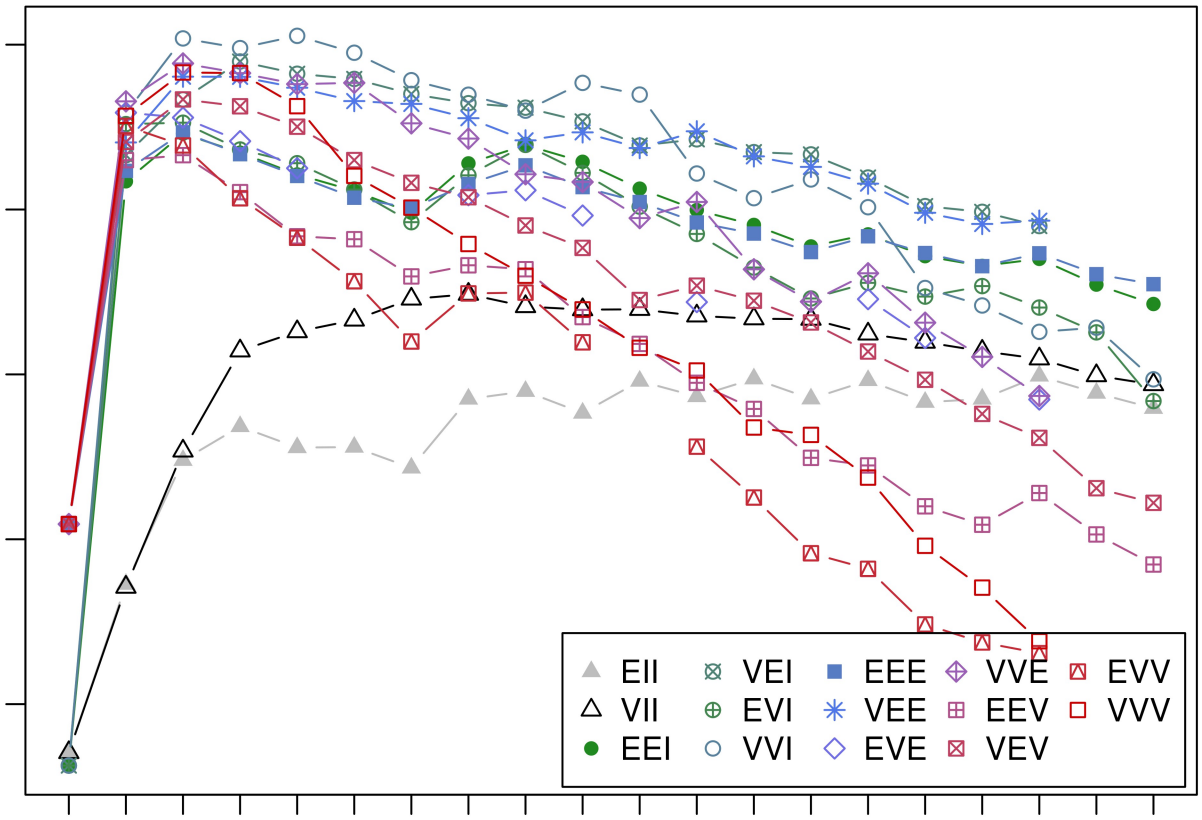

```
plot(opt_mdCluster, what = 'uncertainty')
```




```
df_mc <- Mclust(df_mdCluster, 1:20)

df_final <- Mclust(final_data, 1:20)

summary(df_mc)
```

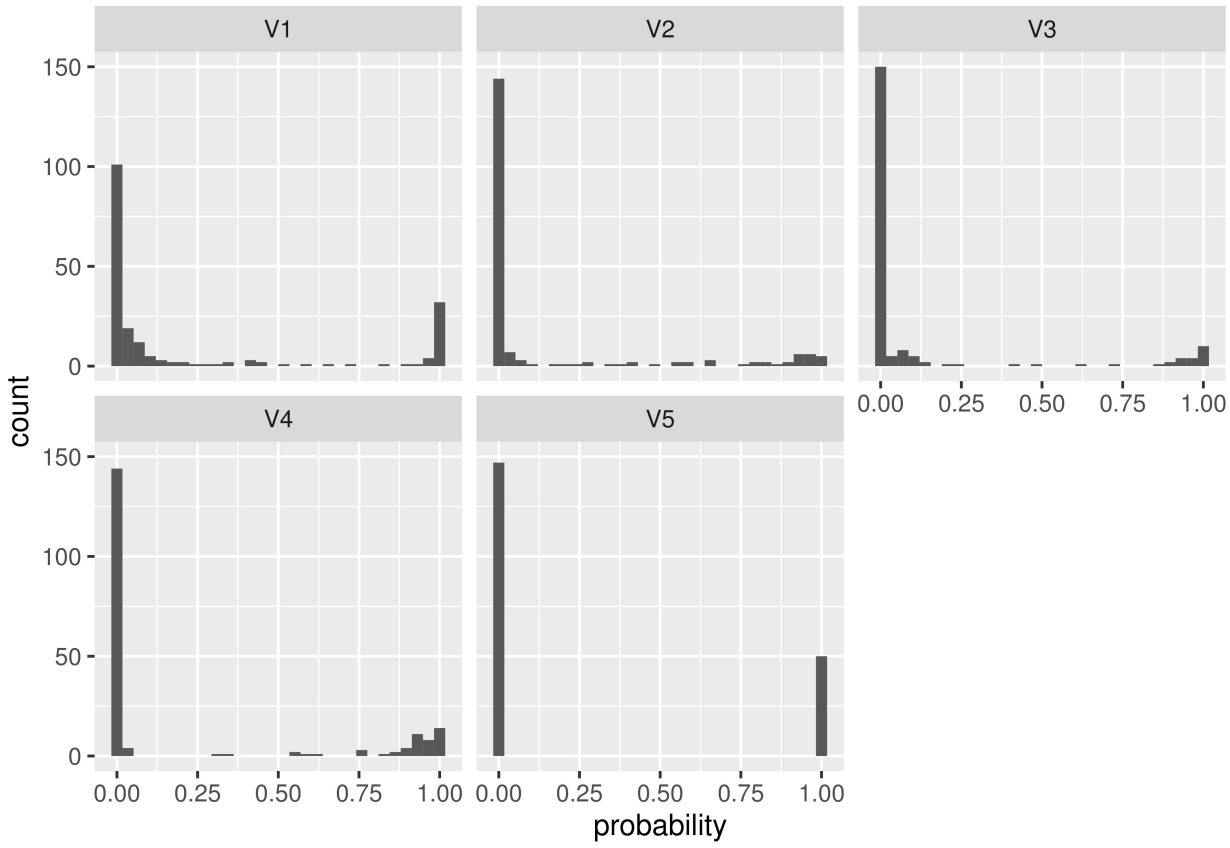
```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVI (diagonal, varying volume and shape) model with 5 components:
##
##  log-likelihood   n df       BIC       ICL
##       -578.4222 197 44 -1389.305 -1422.034
##
## Clustering table:
##  1  2  3  4  5
## 43 33 24 47 50
```

```
plot(df_mc, what = 'BIC',
     legendArgs = list(x = "bottomright", ncol = 5))
```

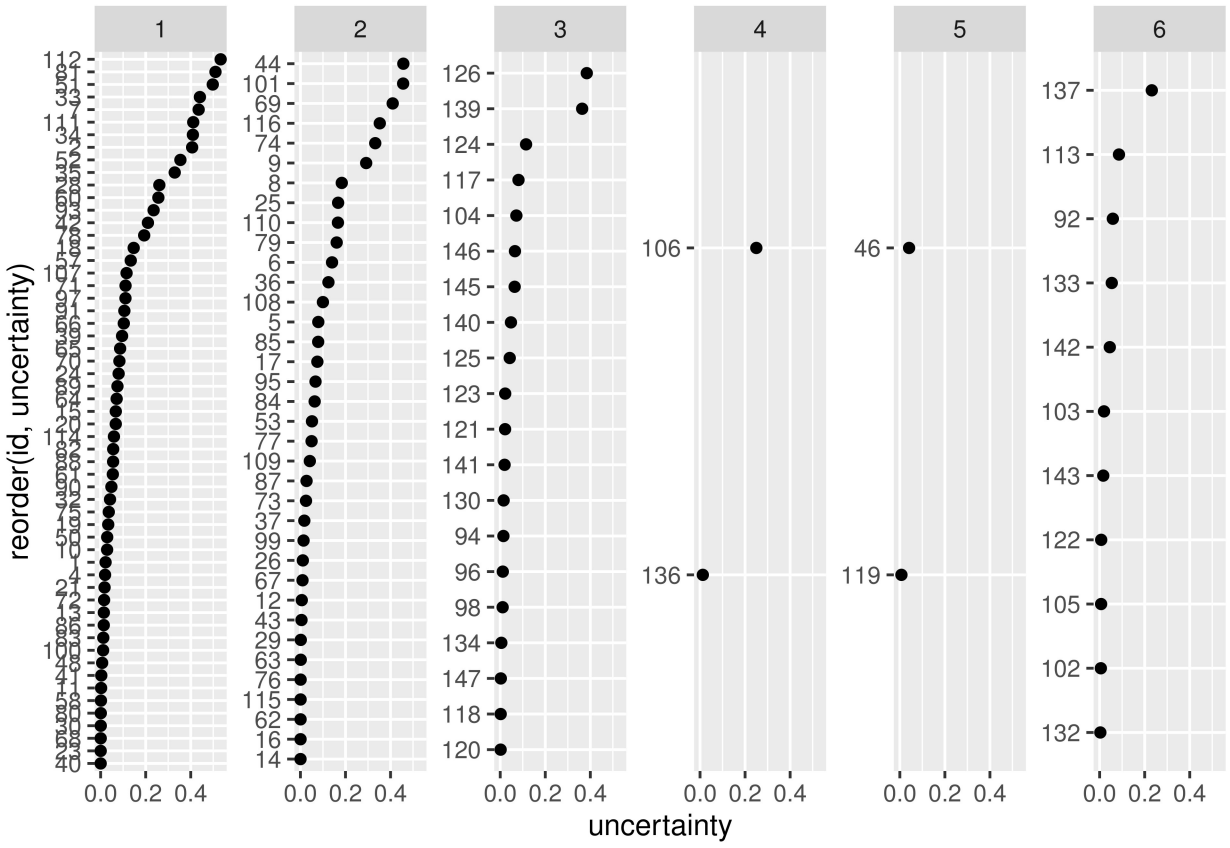



```
probabilities <- df_mc$z

probabilities <- probabilities %>%
  as.data.frame() %>%
  mutate(id = row_number()) %>%
  tidyr::gather(cluster, probability, -id)

ggplot(probabilities, aes(probability)) +
  geom_histogram() +
  facet_wrap(~ cluster, nrow = 2)
```

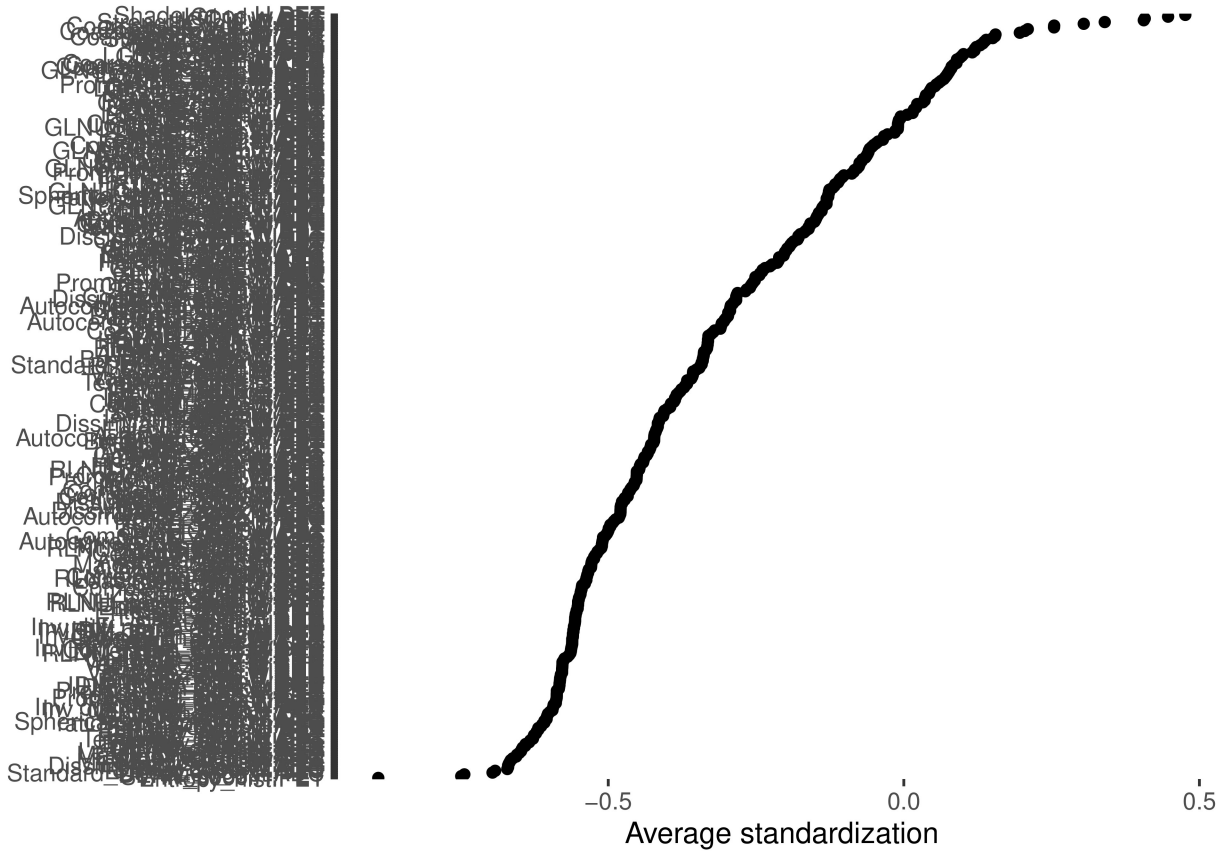
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
uncertainty <- data.frame(
  id = 1:nrow(final_data),
  cluster = df_final$classification,
  uncertainty = df_mc$uncertainty
)

uncertainty %>%
  group_by(cluster) %>%
  filter(uncertainty > 0.0001) %>%
  ggplot(aes(uncertainty, reorder(id, uncertainty))) +
  geom_point() +
  facet_wrap(~ cluster, scales = 'free_y', nrow = 1)
```




```
cluster2 <- final_data %>%
  scale() %>%
  as.data.frame() %>%
  mutate(cluster = df_mc$classification) %>%
  filter(cluster == 2) %>%
  select(-cluster)

cluster2 %>%
  tidyr::gather(product, std_count) %>%
  group_by(product) %>%
  summarize(avg = mean(std_count)) %>%
  ggplot(aes(avg, reorder(product, avg))) +
  geom_point() +
  labs(x = "Average standardization", y = NULL)
```


−0.5
0.0
0.5
Average standardization