Team_project

Javier Merino Meyliani Sanjaya Angeli De los Reyes Nay Zaw Lin

2025-03-30

Contents

1 Objective Questions:

- What distinct clusters can be identified based on lifestyle choices, and how do these clusters correlate with Dry Eye Disease outcomes?
- Does BMI and physical health contribute to the development of Dry Eye Disease, and can clustering analysis reveal subgroups at higher risk?

2 Data Source:

The dataset was sourced from Kaggle (in CSV format) and is intended for predictive modelling and diagnostic analysis of Dry Eye Disease based on key attributes such as sleep quality, sleep duration, eye redness, itchiness, screen time, blue-light filter usage and eye strain.

Link: https://www.kaggle.com/datasets/dakshnagra/dry-eye-disease/data

3 Initial Transformation:

```
df = read.csv("/Users/javiermerino/Documents/Langara/3.DANA4840/Project/Dry_Eye_Dataset.csv", header = T
```

3.1 Missing Values

```
colSums(is.na(df))
```

```
##                     Gender                        Age
##                          0                          0
##             Sleep.duration             Sleep.quality
##                          0                          0
##               Stress.level            Blood.pressure
##                          0                          0
##                 Heart.rate               Daily.steps
##                          0                          0
##          Physical.activity                     Height
##                          0                          0
```

```
##                       Weight              Sleep.disorder
##                            0                           0
##         Wake.up.during.night      Feel.sleepy.during.day
##                            0                           0
##         Caffeine.consumption         Alcohol.consumption
##                            0                           0
##                      Smoking               Medical.issue
##                            0                           0
##           Ongoing.medication     Smart.device.before.bed
##                            0                           0
##          Average.screen.time           Blue.light.filter
##                            0                           0
##        Discomfort.Eye.strain              Redness.in.eye
##                            0                           0
## Itchiness.Irritation.in.eye             Dry.Eye.Disease
##                            0                           0
```

3.2 Data Type

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
df_factor = df%>%mutate(across(c(Gender,Sleep.disorder,Wake.up.during.night,Feel.sleepy.during.day,Caff
```

```
df_bloodp = df_factor %>%
  separate(Blood.pressure, into = c("Systolic", "Diastolic"), sep = "/", convert = TRUE)
```

```
summary(df_bloodp)
```

```
##  Gender          Age         Sleep.duration   Sleep.quality Stress.level
##  F: 9972   Min.   :18.00   Min.   : 4.000   1:4003        1:4012
##  M:10028   1st Qu.:24.00   1st Qu.: 5.500   2:3995        2:3929
##            Median :31.00   Median : 7.000   3:4024        3:4131
##            Mean   :31.42   Mean   : 6.998   4:4010        4:4028
##            3rd Qu.:39.00   3rd Qu.: 8.500   5:3968        5:3900
##            Max.   :45.00   Max.   :10.000
##     Systolic     Diastolic    Heart.rate      Daily.steps    Physical.activity
##  Min.   : 90   Min.   :60   Min.   : 60.00   Min.   : 1000   Min.   :  0.00
##  1st Qu.:102   1st Qu.:67   1st Qu.: 70.00   1st Qu.: 6000   1st Qu.: 45.00
##  Median :115   Median :75   Median : 80.00   Median :11000   Median : 91.00
##  Mean   :115   Mean   :75   Mean   : 79.91   Mean   :10537   Mean   : 90.07
##  3rd Qu.:128   3rd Qu.:83   3rd Qu.: 90.00   3rd Qu.:16000   3rd Qu.:135.00
##  Max.   :140   Max.   :90   Max.   :100.00   Max.   :20000   Max.   :180.00
##      Height          Weight       Sleep.disorder Wake.up.during.night
##  Min.   :150.0   Min.   : 50.00   N:10069        N:10000
```

```
##  1st Qu.:162.0   1st Qu.: 62.00   Y: 9931        Y:10000
##  Median :175.0   Median : 75.00
##  Mean   :174.9   Mean   : 74.89
##  3rd Qu.:188.0   3rd Qu.: 88.00
##  Max.   :200.0   Max.   :100.00
##  Feel.sleepy.during.day Caffeine.consumption Alcohol.consumption Smoking
##  N:10178                N: 9911              N: 9991             N:10017
##  Y: 9822                Y:10089              Y:10009             Y: 9983
##
##
##
##
##  Medical.issue Ongoing.medication Smart.device.before.bed Average.screen.time
##  N:10111       N: 9918            N: 9997                 Min.   : 1.00
##  Y: 9889       Y:10082            Y:10003                 1st Qu.: 3.30
##                                                           Median : 5.50
##                                                           Mean   : 5.52
##                                                           3rd Qu.: 7.80
##                                                           Max.   :10.00
##  Blue.light.filter Discomfort.Eye.strain Redness.in.eye
##  N:10016           N: 9963               N:10129
##  Y: 9984           Y:10037               Y: 9871
##
##
##
##
##  Itchiness.Irritation.in.eye Dry.Eye.Disease
##  N:10063                     N: 6963
##  Y: 9937                     Y:13037
##
##
##
##
```

```
df_bmi = df_bloodp %>%
  mutate(BMI = (Weight/(Height*Height))*10000)
```

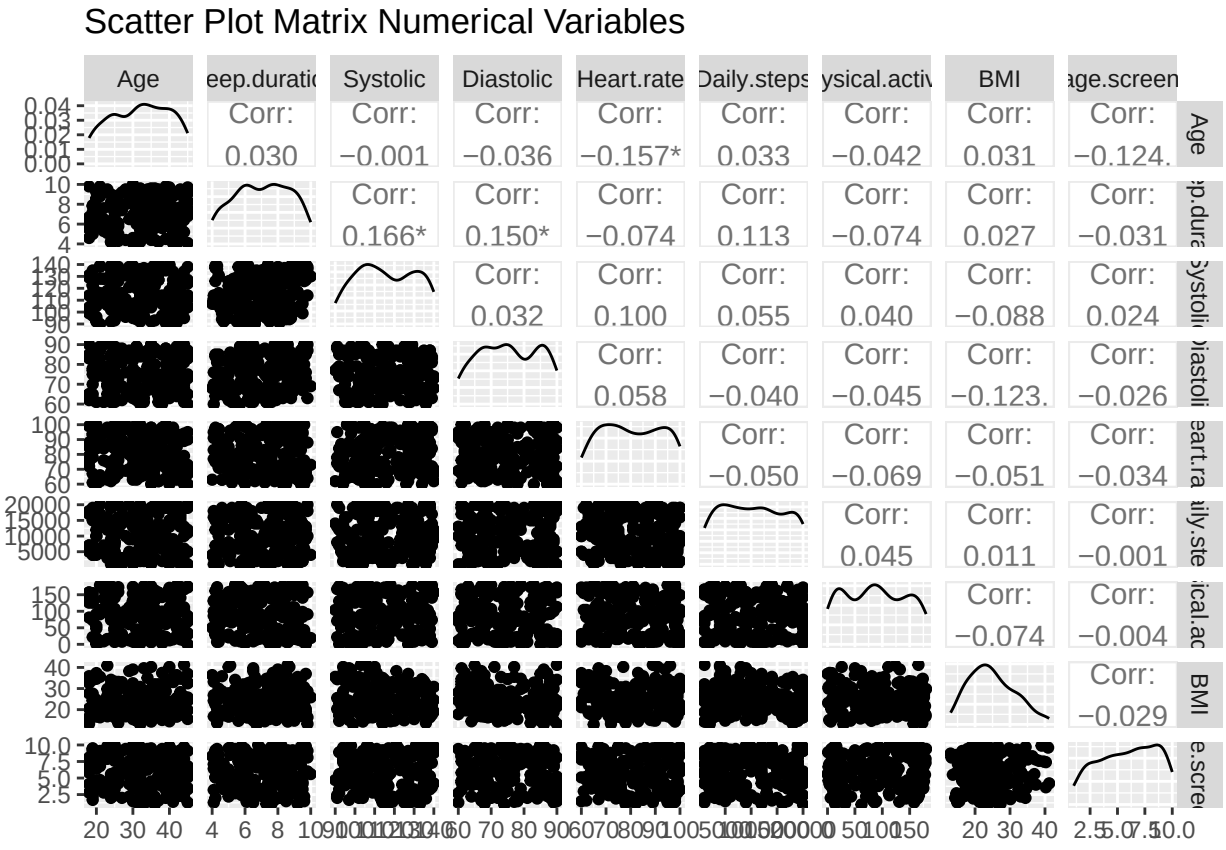
3.3 Data Sample

```
library(rsample)

set.seed(123)  # Ensure reproducibility

# Perform stratified sampling
sample = df_bmi %>%
  group_by(Dry.Eye.Disease) %>%  # Replace "Gender" with your categorical variable
  sample_frac(size = 200 / nrow(df_bloodp)) %>%
  ungroup()
```

4 Data Exploration

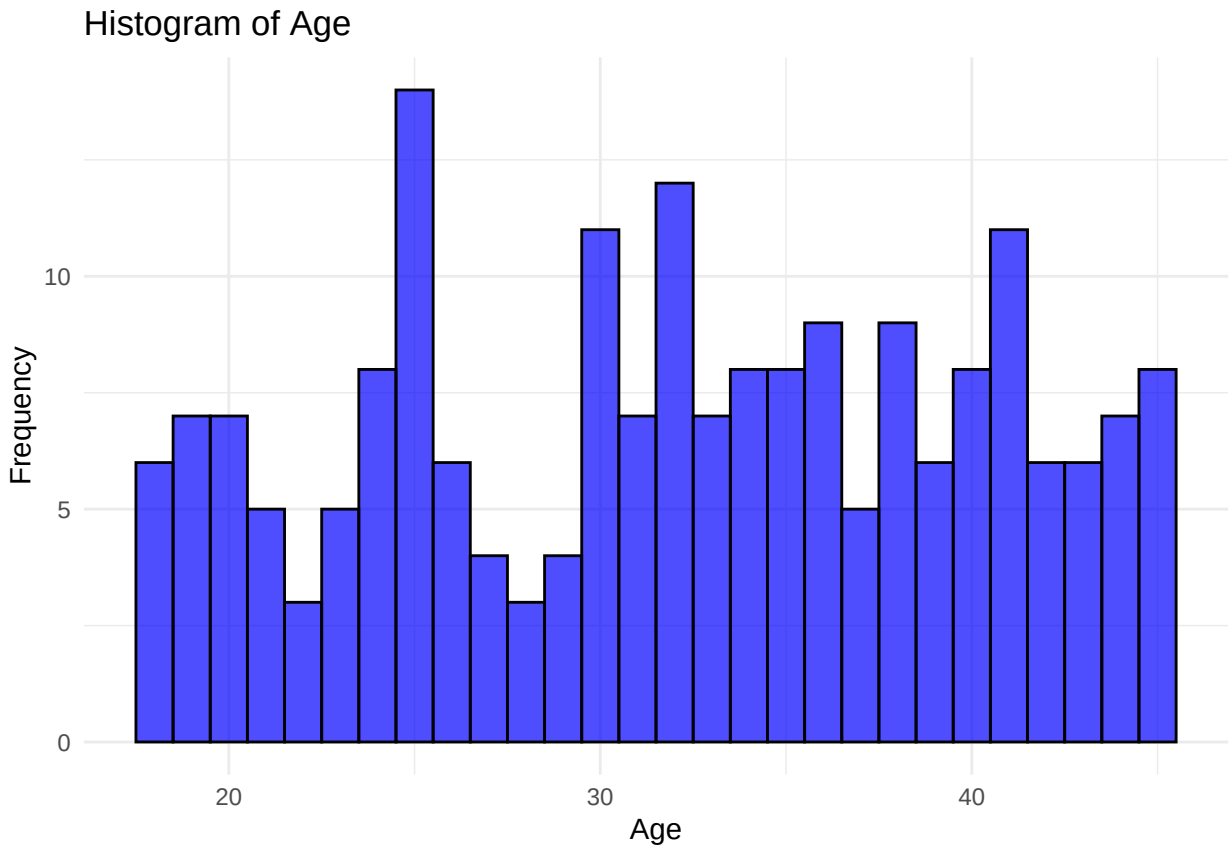
4.1 Numerical Data

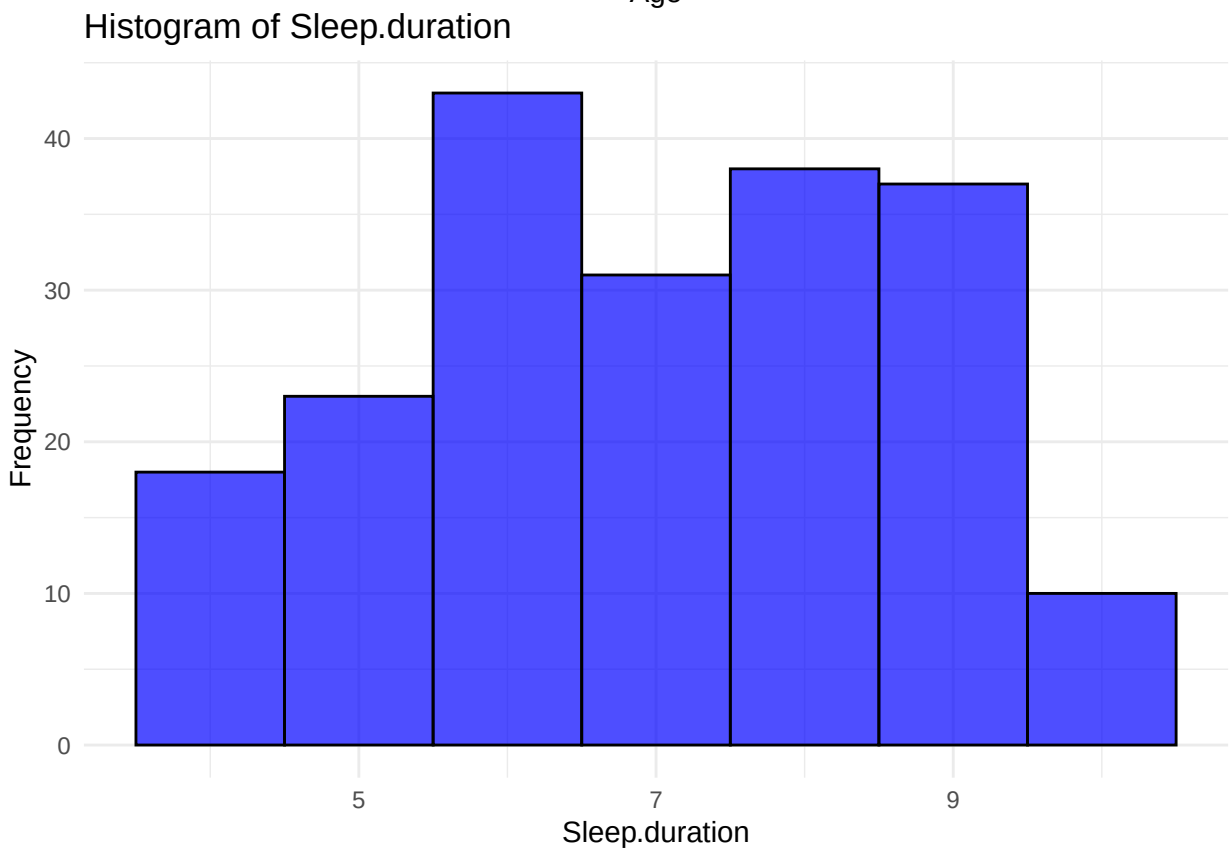
```
sample.numeric = sample %>% select(Age, Sleep.duration, Systolic, Diastolic, Heart.rate, Daily.steps, Pl
```

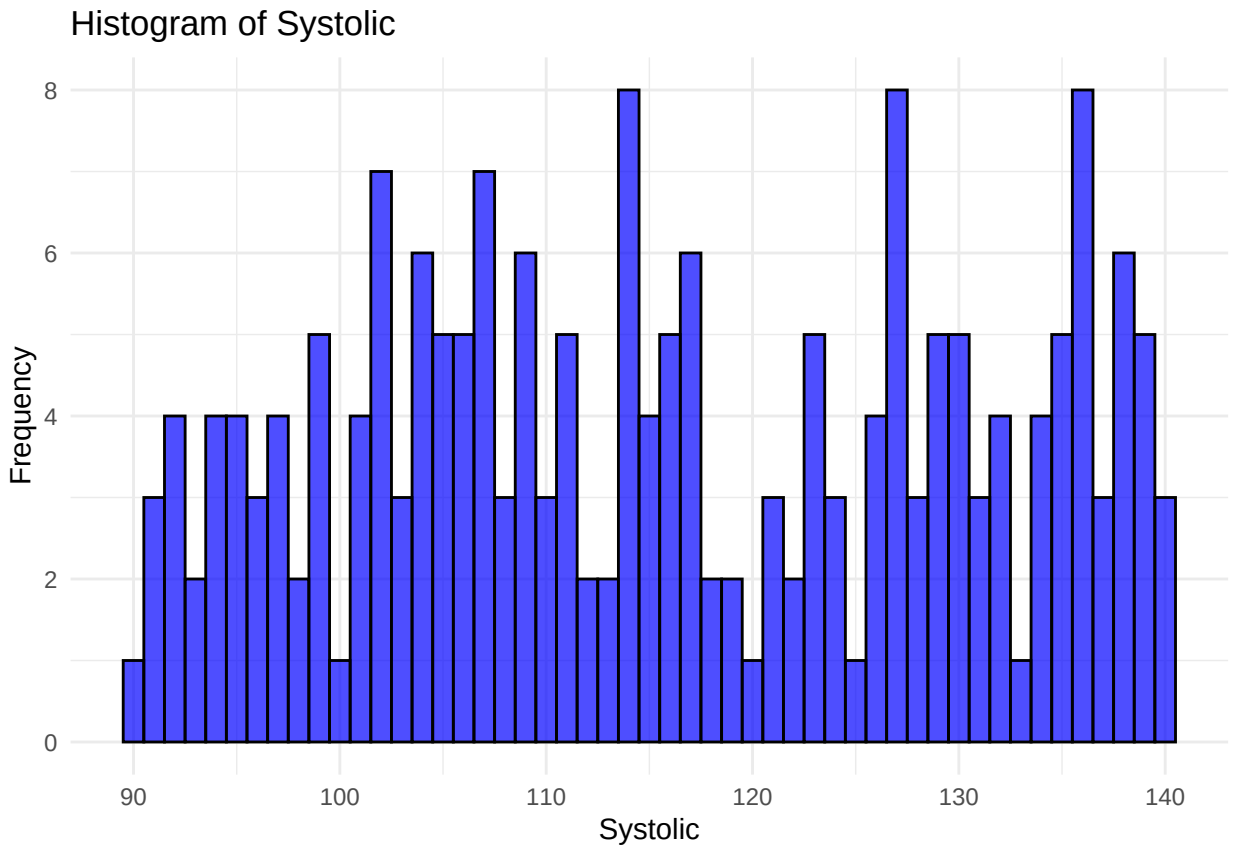


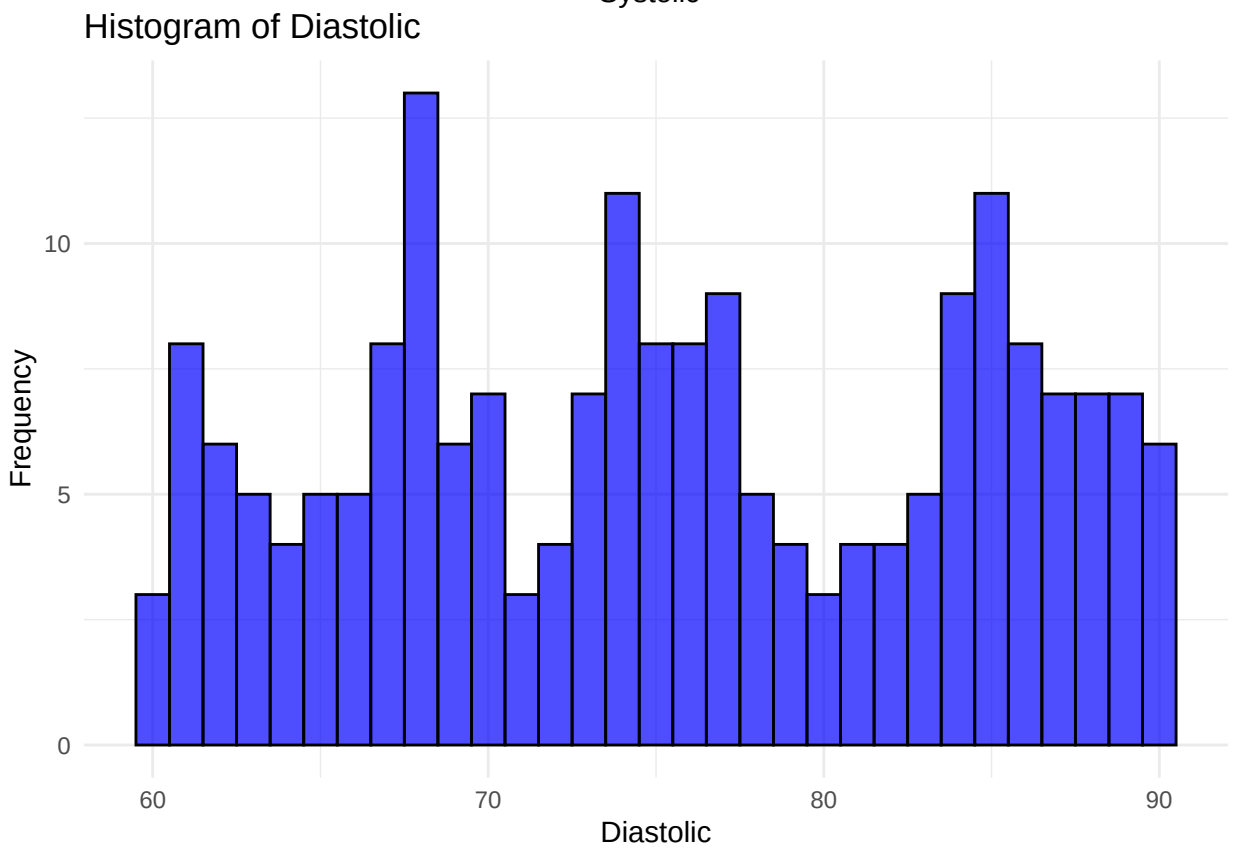

```
plot_histograms <- function(df) {
  numeric_cols <- df %>% select(where(is.numeric))

  for (col in colnames(numeric_cols)) {
    p = ggplot(df, aes_string(x = col)) +
      geom_histogram(binwidth = 1, fill = "blue", color = "black", alpha = 0.7) +
      labs(title = paste("Histogram of", col), x = col, y = "Frequency") +
      theme_minimal()
    print(p)
  }
}

plot_histograms(sample.numeric)
```

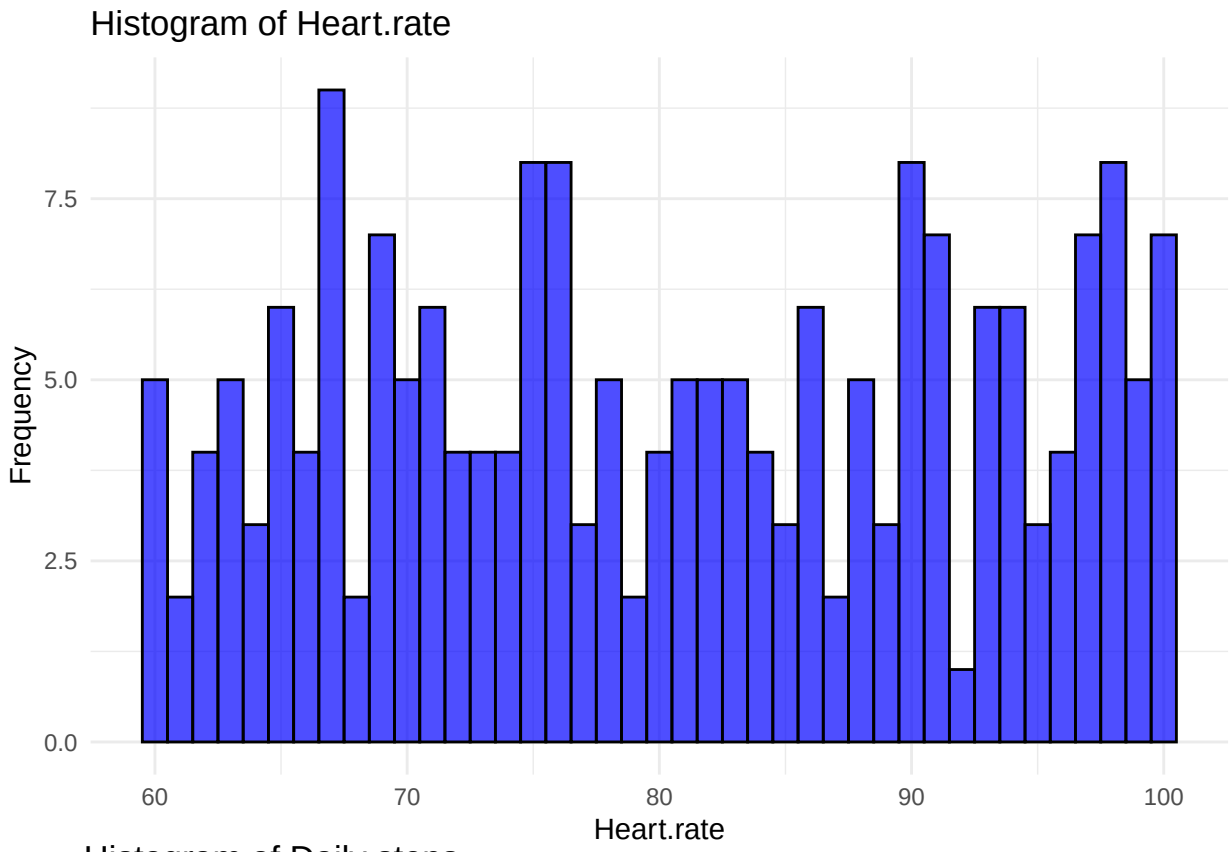
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

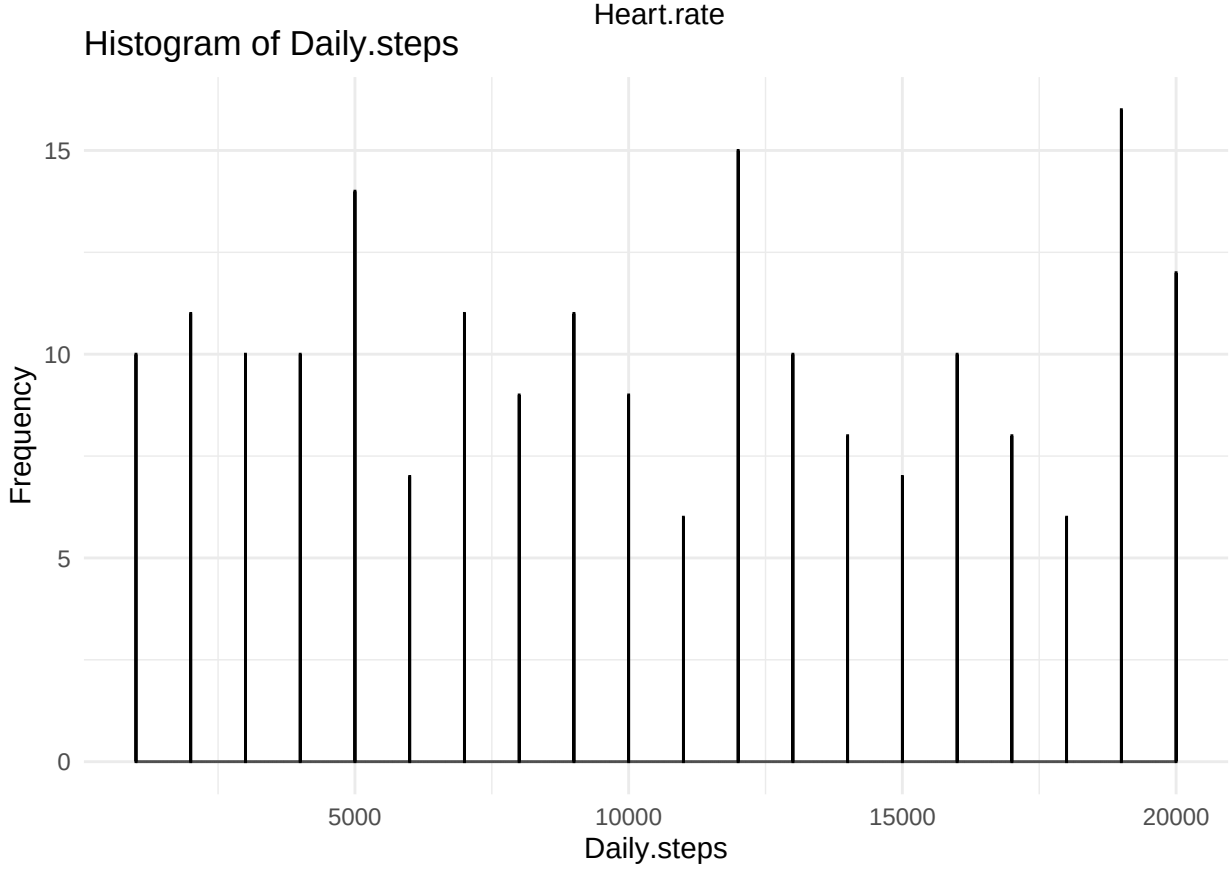

Histogram of Age
Frequency
0
5
10
20
30
40
Age

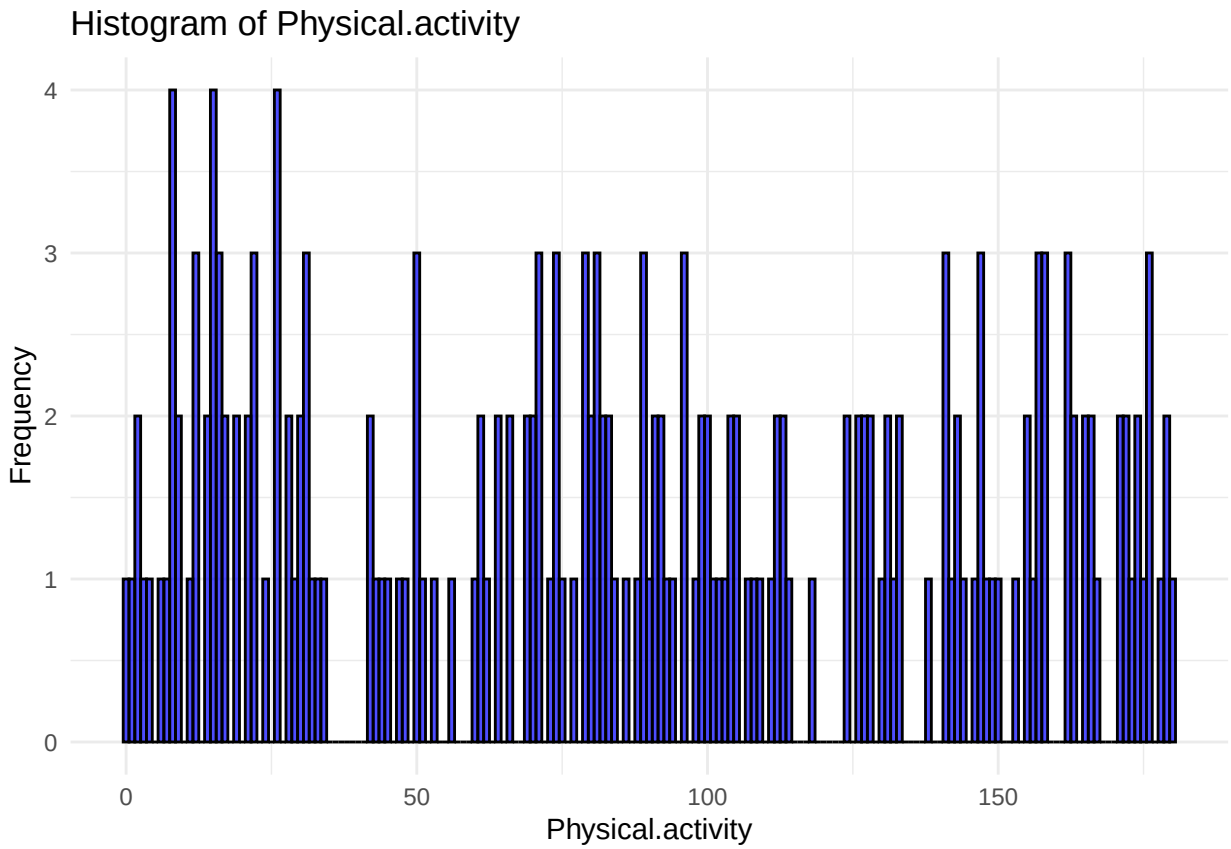

Histogram of Sleep.duration
Frequency
0
10
20
30
40
5
7
9
Sleep.duration

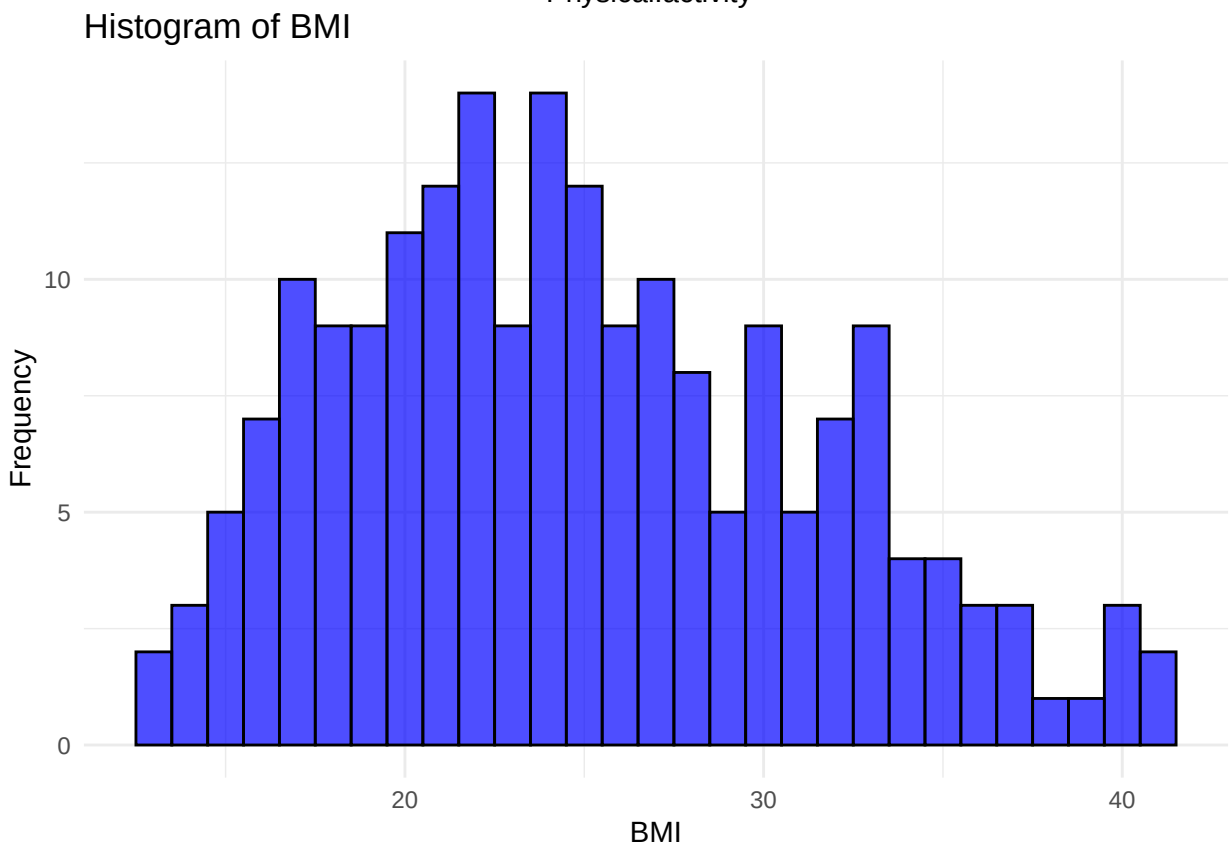

Histogram of Systolic
Frequency
0
2
4
6
8
90
100
110
120
130
140
Systolic

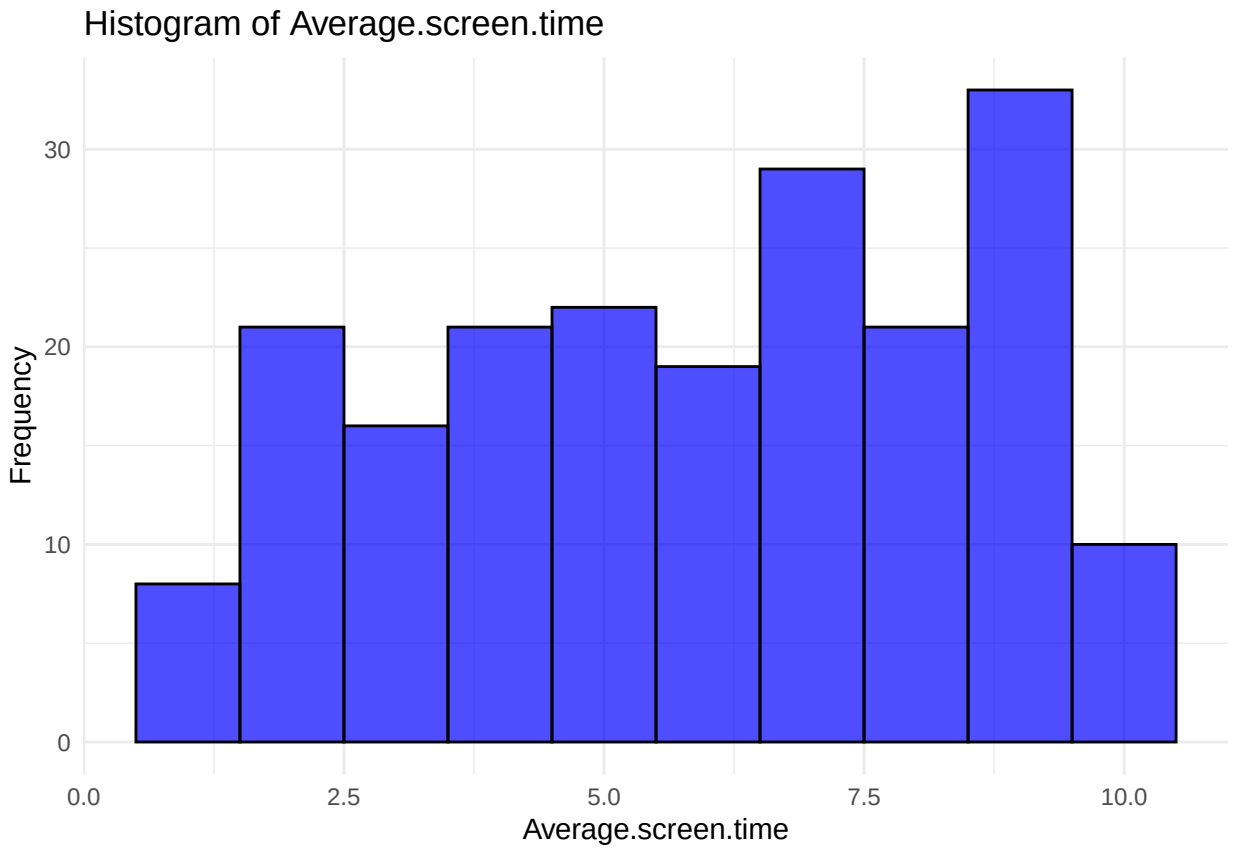

Histogram of Diastolic
Frequency
0
5
10
60
70
80
90
Diastolic


Histogram of Heart.rate
Frequency
0.0
2.5
5.0
7.5
60
70
80
90
100
Heart.rate

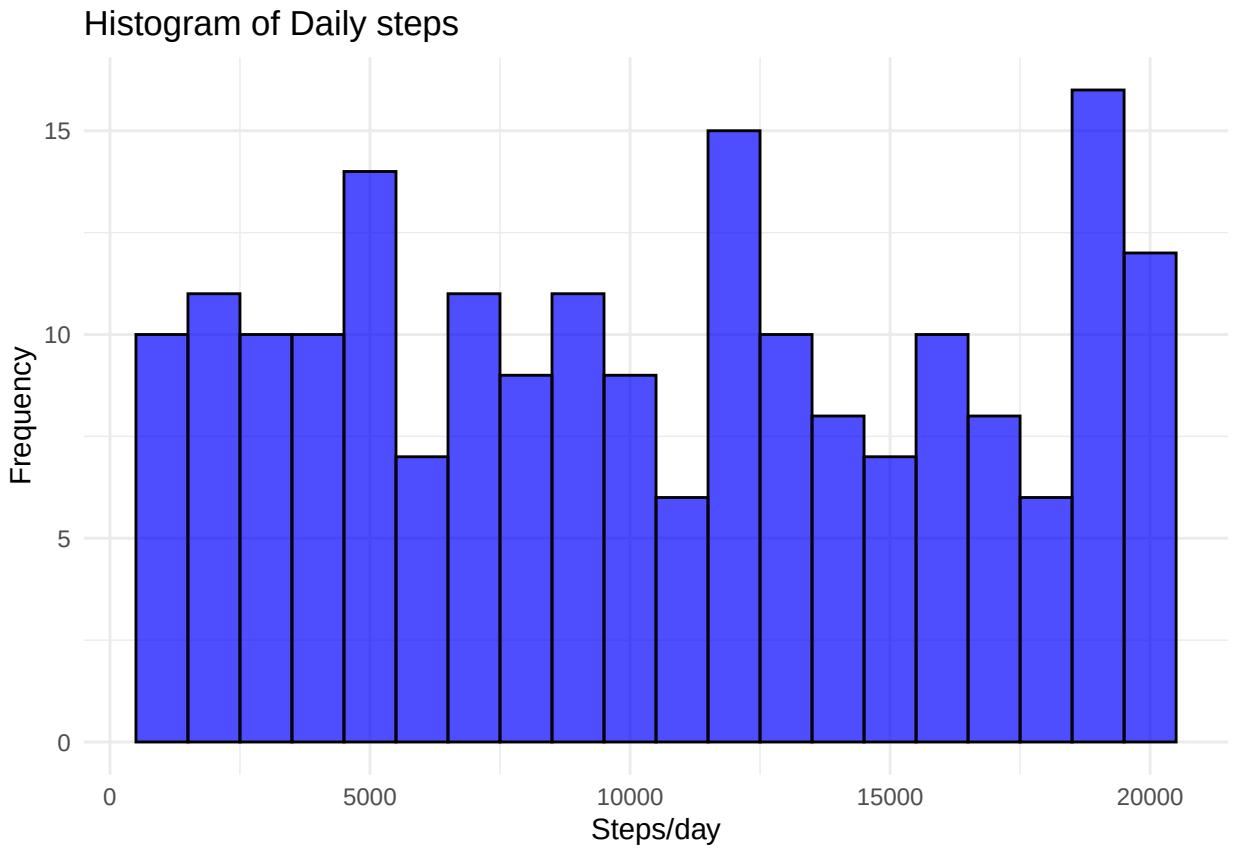

Histogram of Daily.steps
Frequency
0
5
10
15
5000
10000
15000
20000
Daily.steps


Histogram of Physical.activity
Frequency
Physical.activity
0
1
2
3
4
0
50
100
150

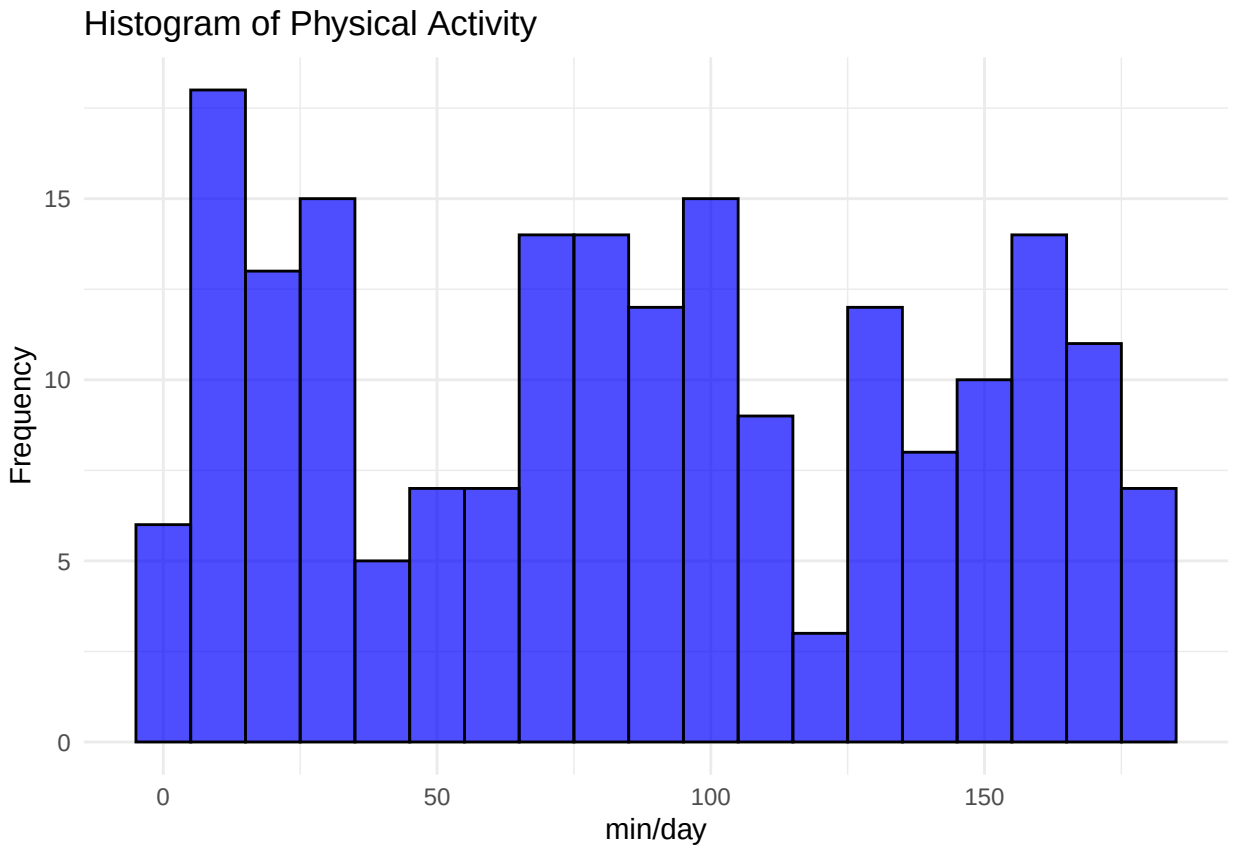

Histogram of BMI
Frequency
BMI
0
5
10
20
30
40




```
ggplot(sample.numeric, aes_string(x = sample.numeric$Daily.steps)) +
      geom_histogram(binwidth = 1000, fill = "blue", color = "black", alpha = 0.7) +
      labs(title = "Histogram of Daily steps", x = "Steps/day", y = "Frequency") +
      theme_minimal()
```



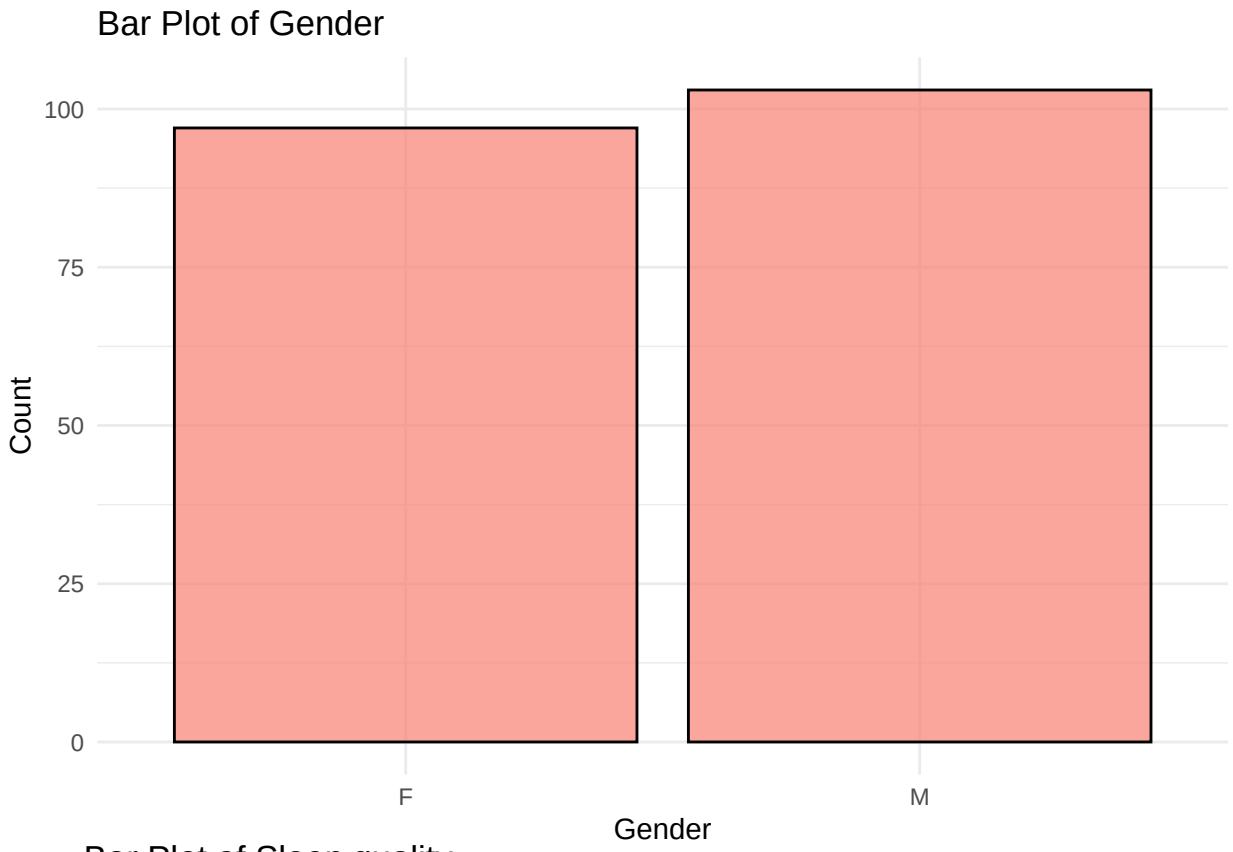

```
ggplot(sample.numeric, aes_string(x = sample.numeric$Physical.activity)) +
      geom_histogram(binwidth = 10, fill = "blue", color = "black", alpha = 0.7) +
      labs(title = "Histogram of Physical Activity", x = "min/day", y = "Frequency") +
      theme_minimal()
```

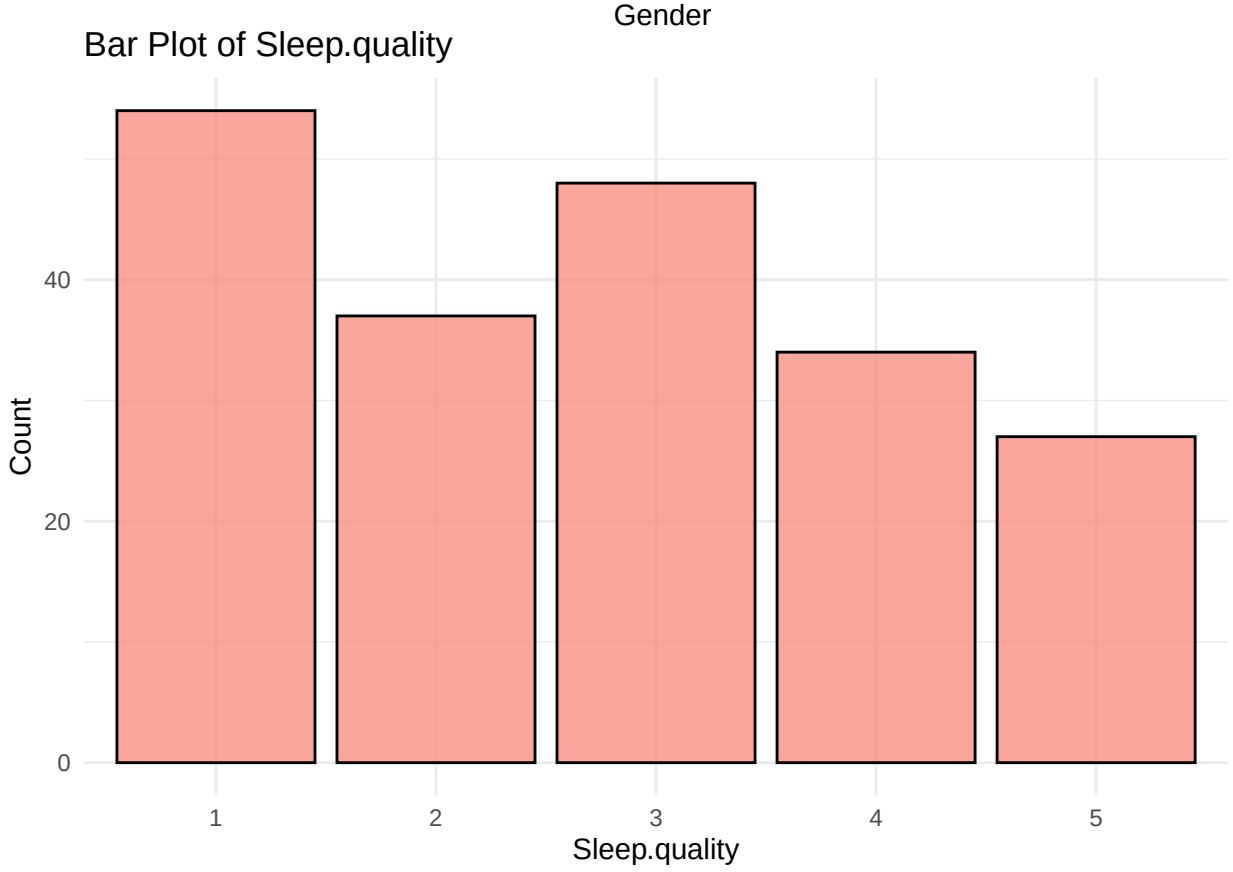


4.2 Categorical Data

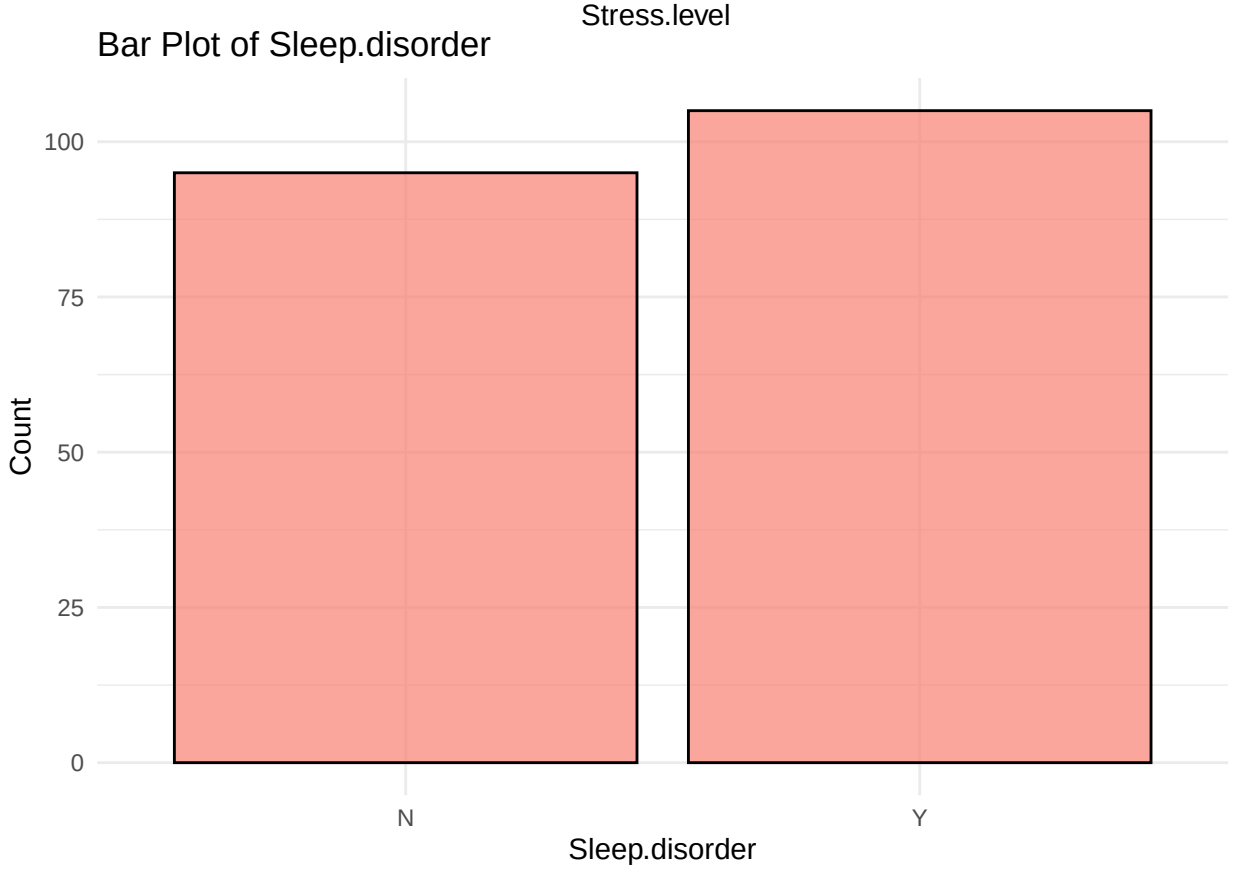
```
sample.categorical = sample %>%select(Gender,Sleep.quality,Stress.level, Sleep.disorder,Wake.up.during.
```

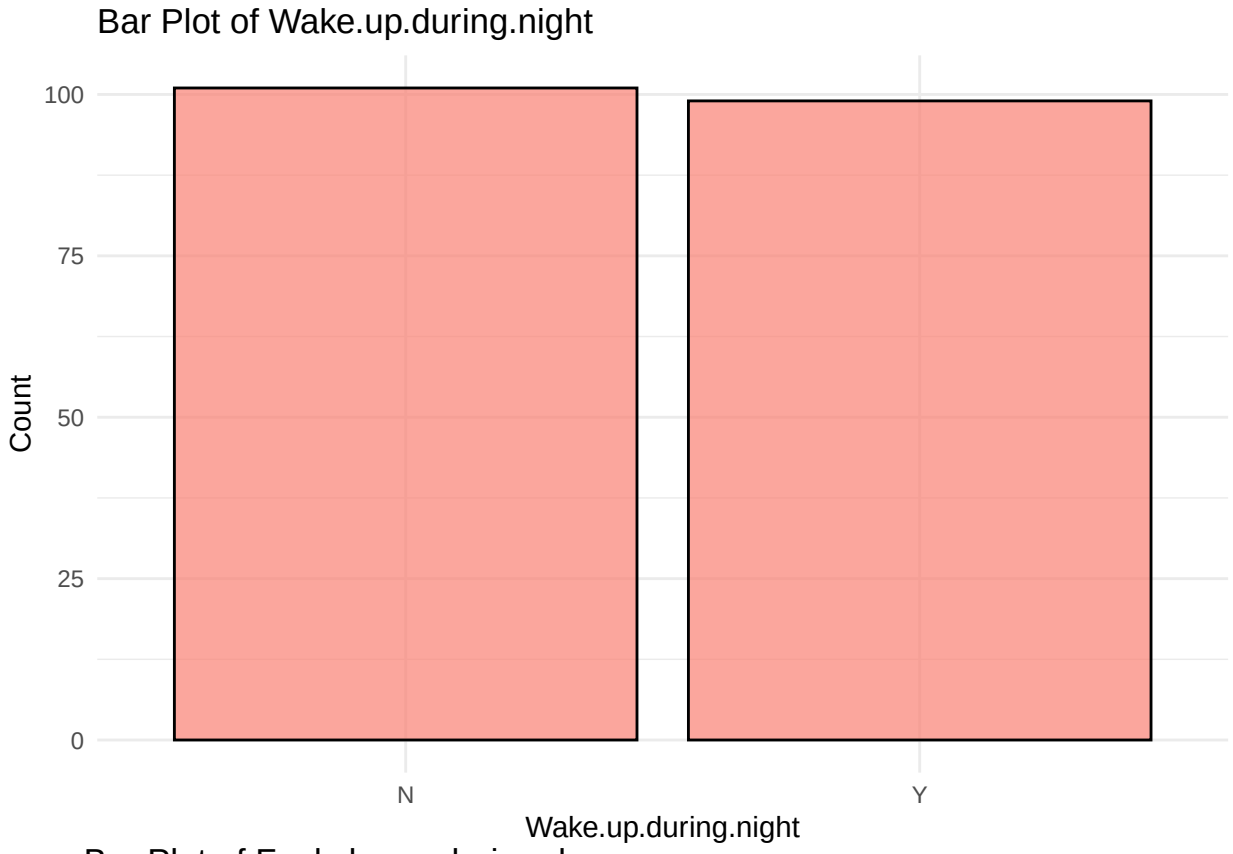
```
plot_bar_plots <- function(df) {
  categorical_cols <- df %>% select(where(is.factor))

  for (col in colnames(categorical_cols)) {
    p <- ggplot(df, aes_string(x = col)) +
      geom_bar(fill = "salmon", color = "black", alpha = 0.7) +
      labs(title = paste("Bar Plot of", col), x = col, y = "Count") +
      theme(axis.text.x = element_text(angle = 45, hjust = 1)) +  # Rotate x-axis labels
      theme_minimal()

    print(p)  # Print the plot
  }
}

# Call the function on your categorical data frame
plot_bar_plots(sample.categorical)
```

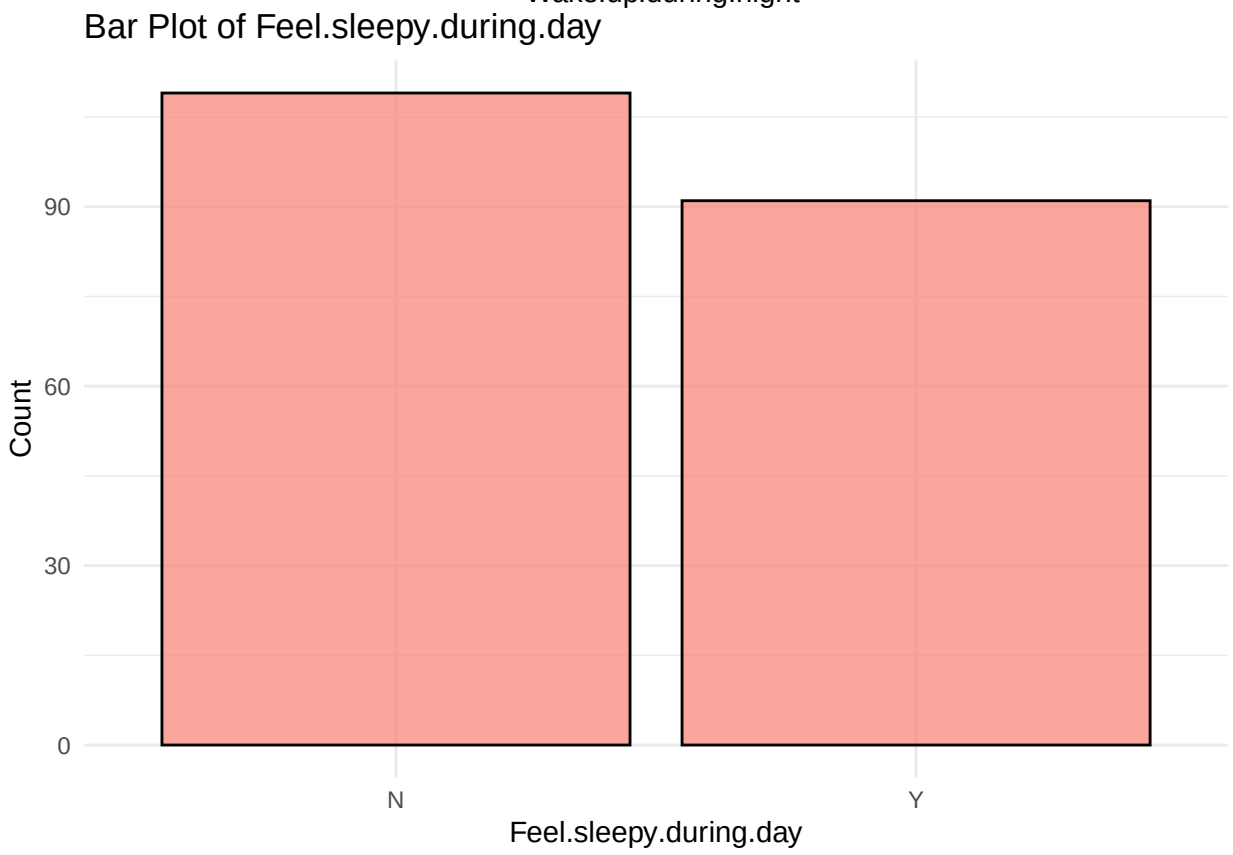

Bar Plot of Gender
100
75
50
25
0
Count
F
M
Gender

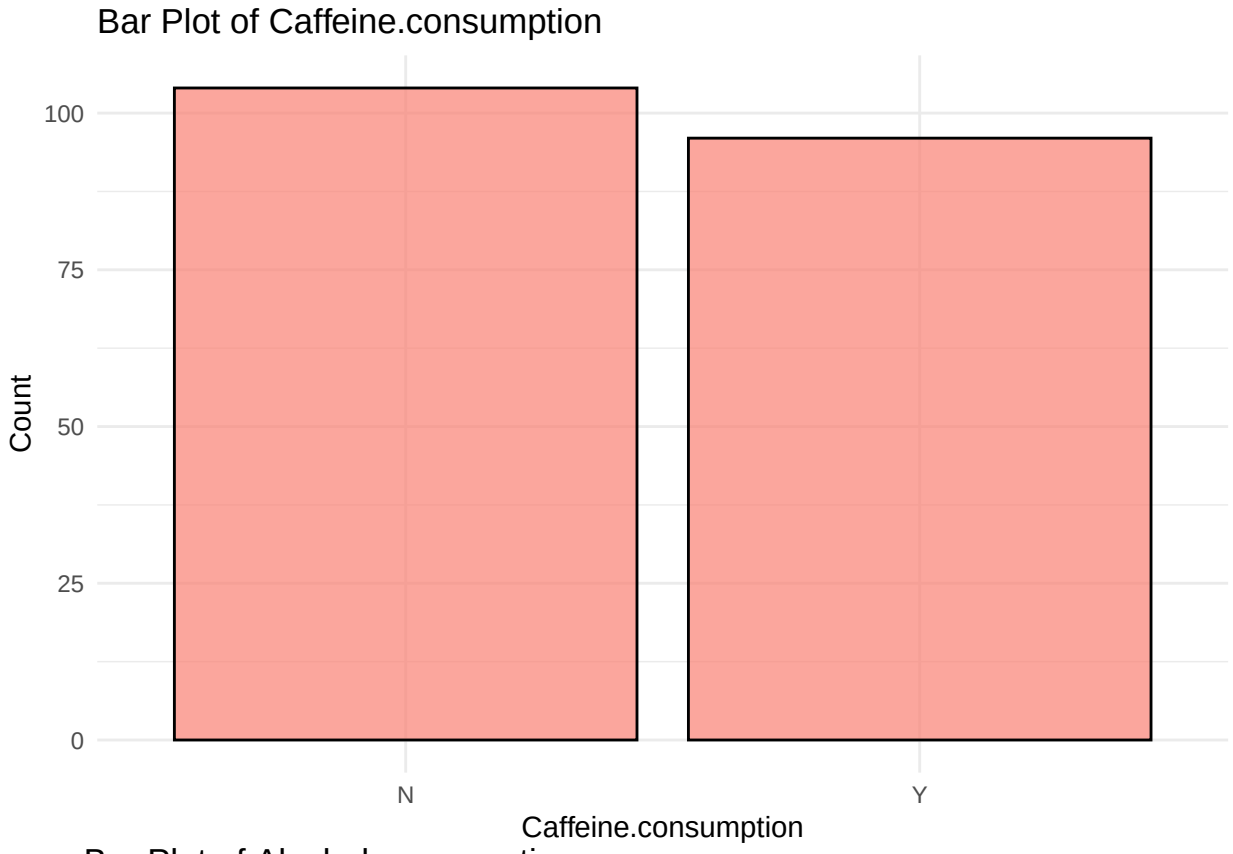

Bar Plot of Sleep.quality
40
20
0
Count
1
2
3
4
5
Sleep.quality

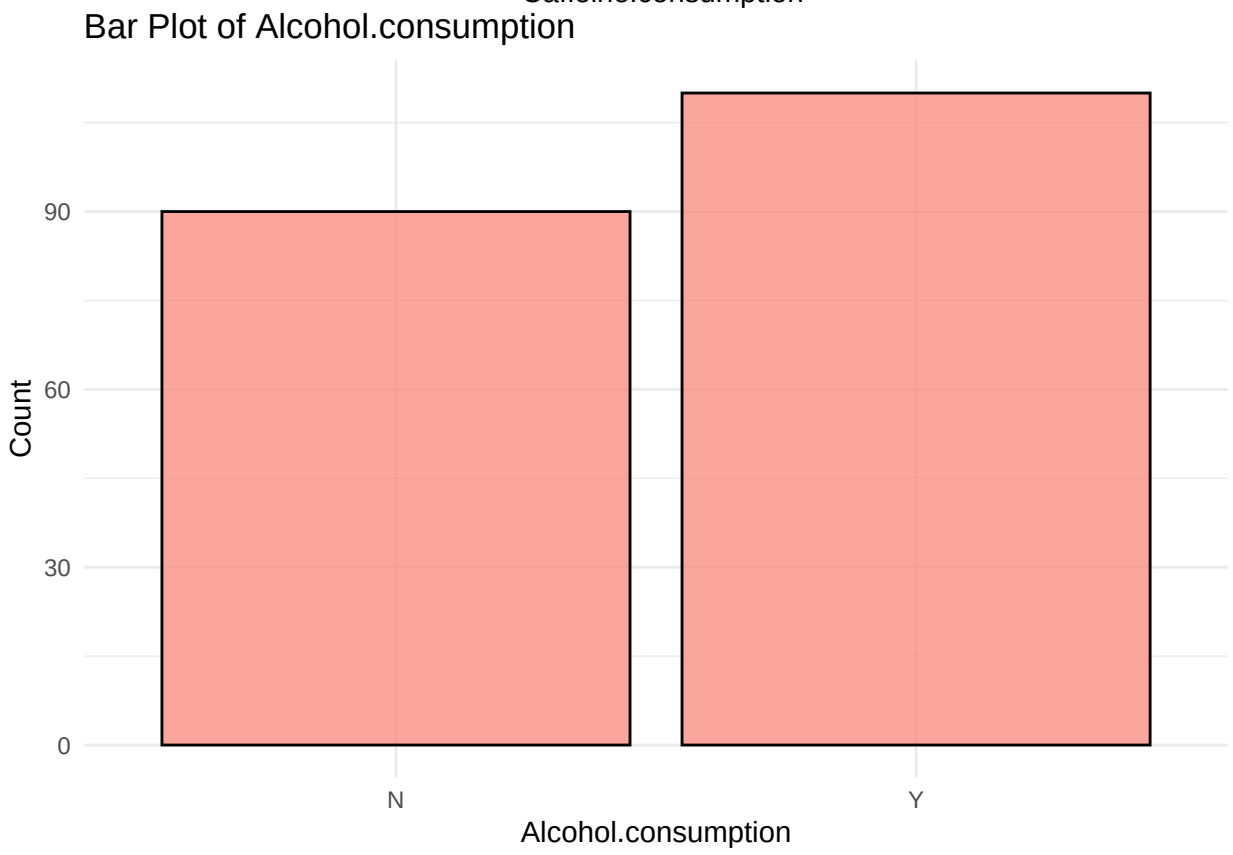

Bar Plot of Stress.level
Count
50
40
30
20
10
0
1
2
3
4
5
Stress.level

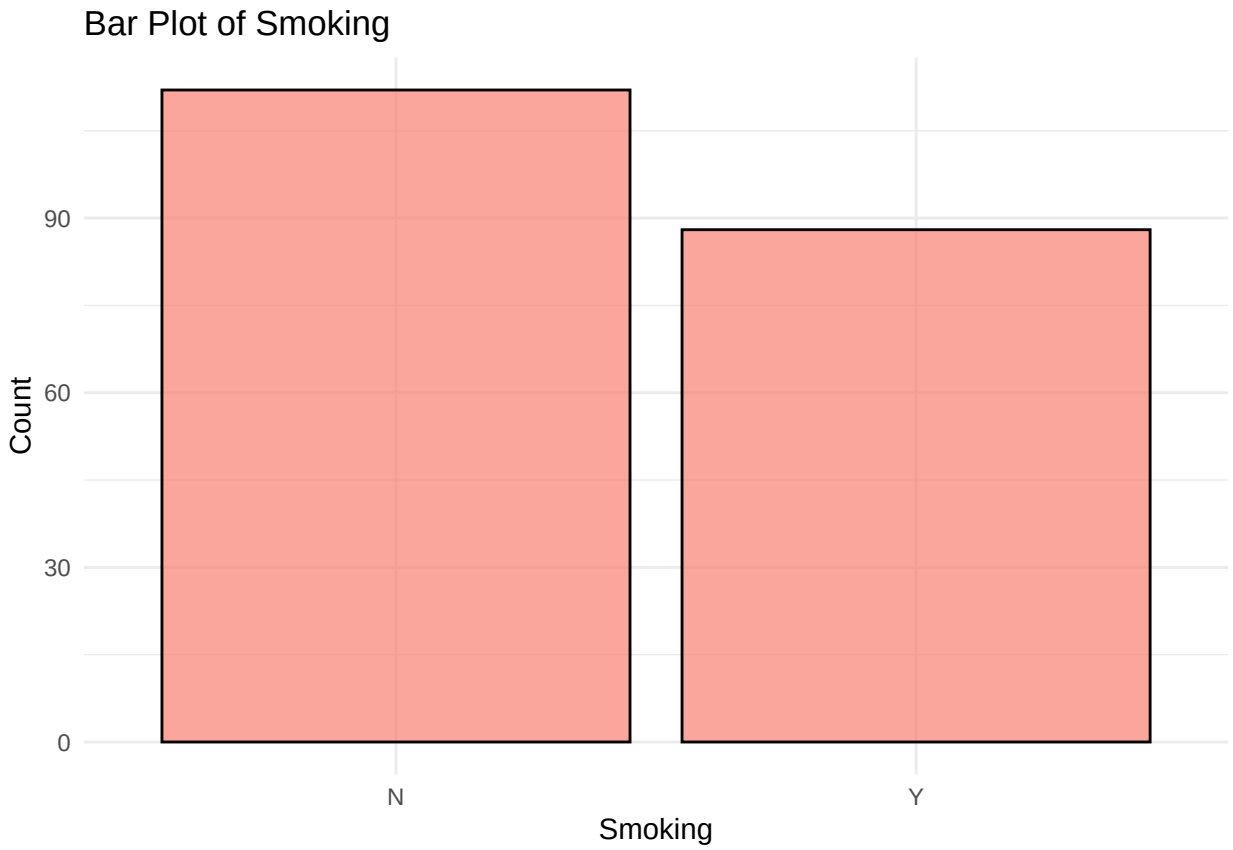

Bar Plot of Sleep.disorder
Count
100
75
50
25
0
N
Y
Sleep.disorder

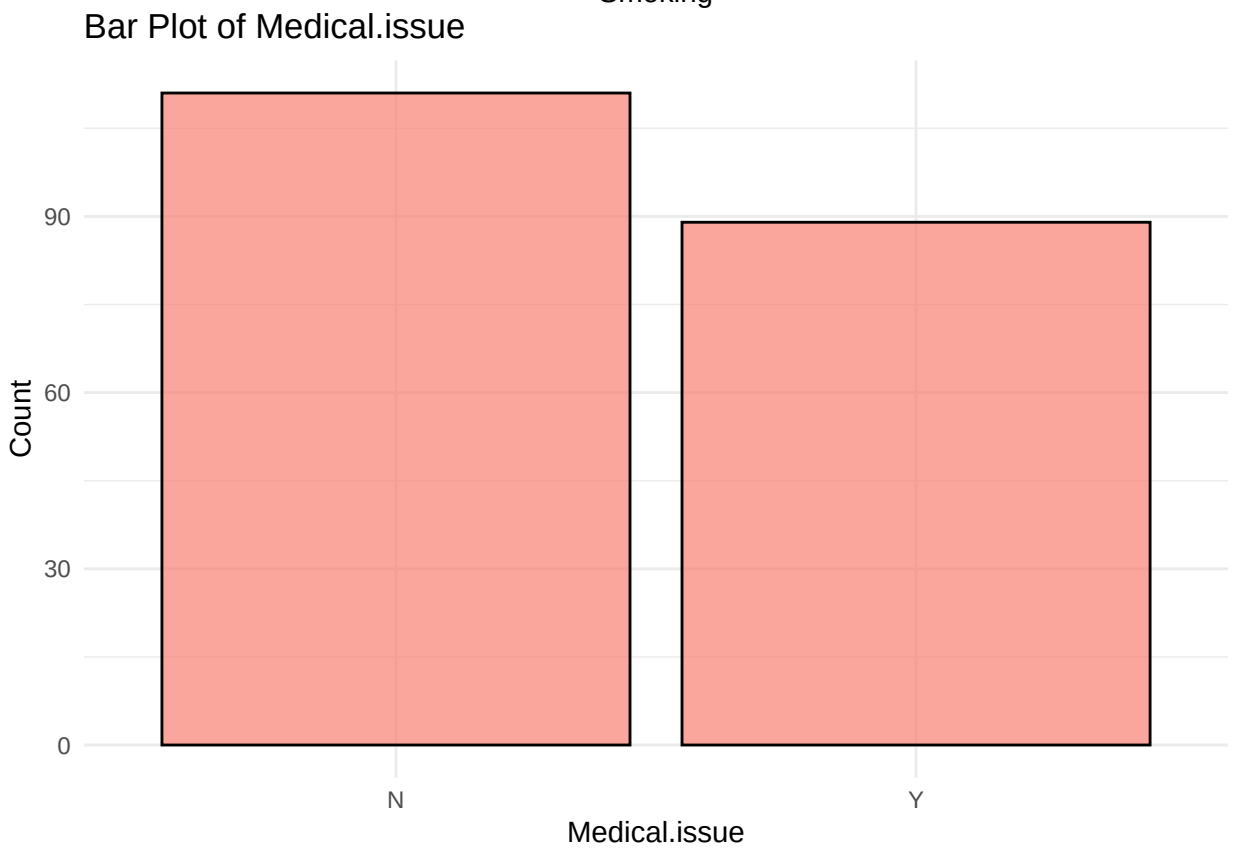

Bar Plot of Wake.up.during.night
Count
100
75
50
25
0
N
Y
Wake.up.during.night

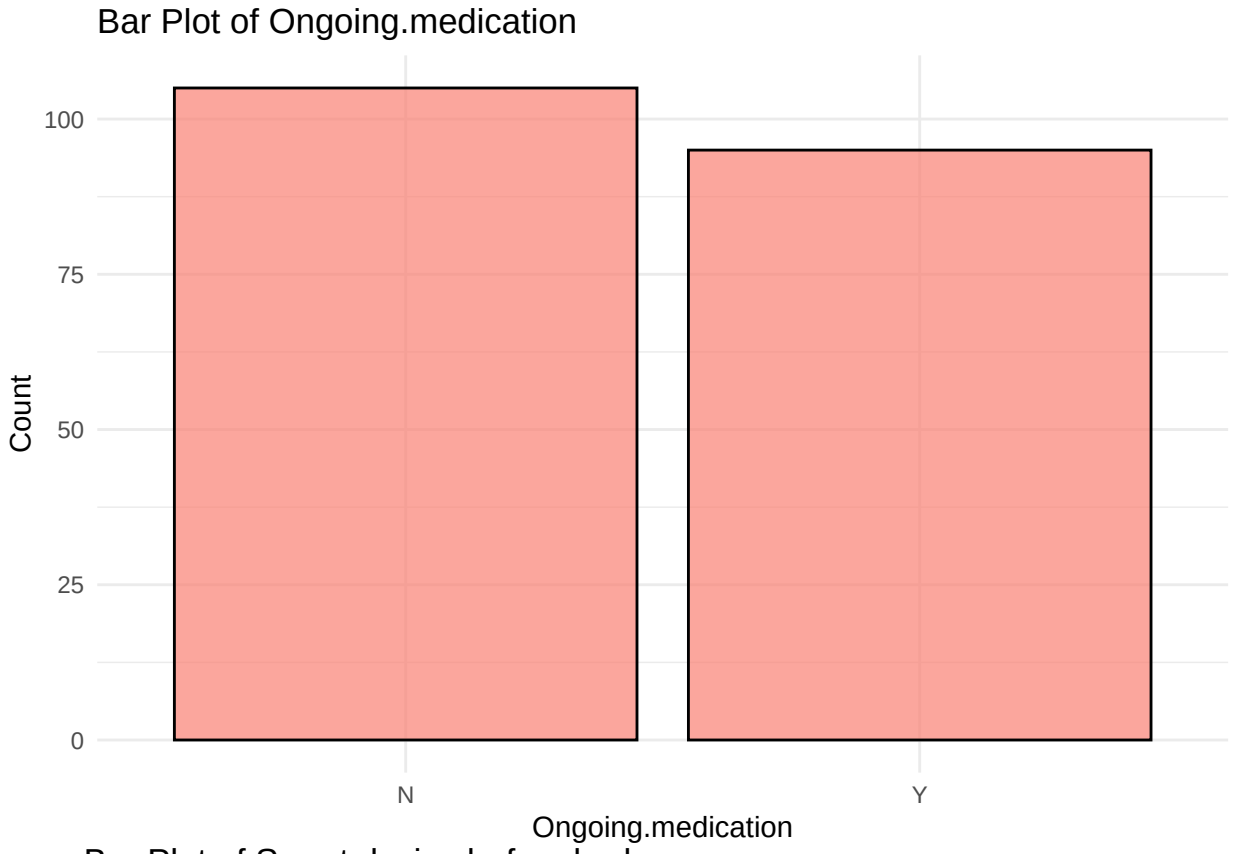

Bar Plot of Feel.sleepy.during.day
Count
90
60
30
0
N
Y
Feel.sleepy.during.day

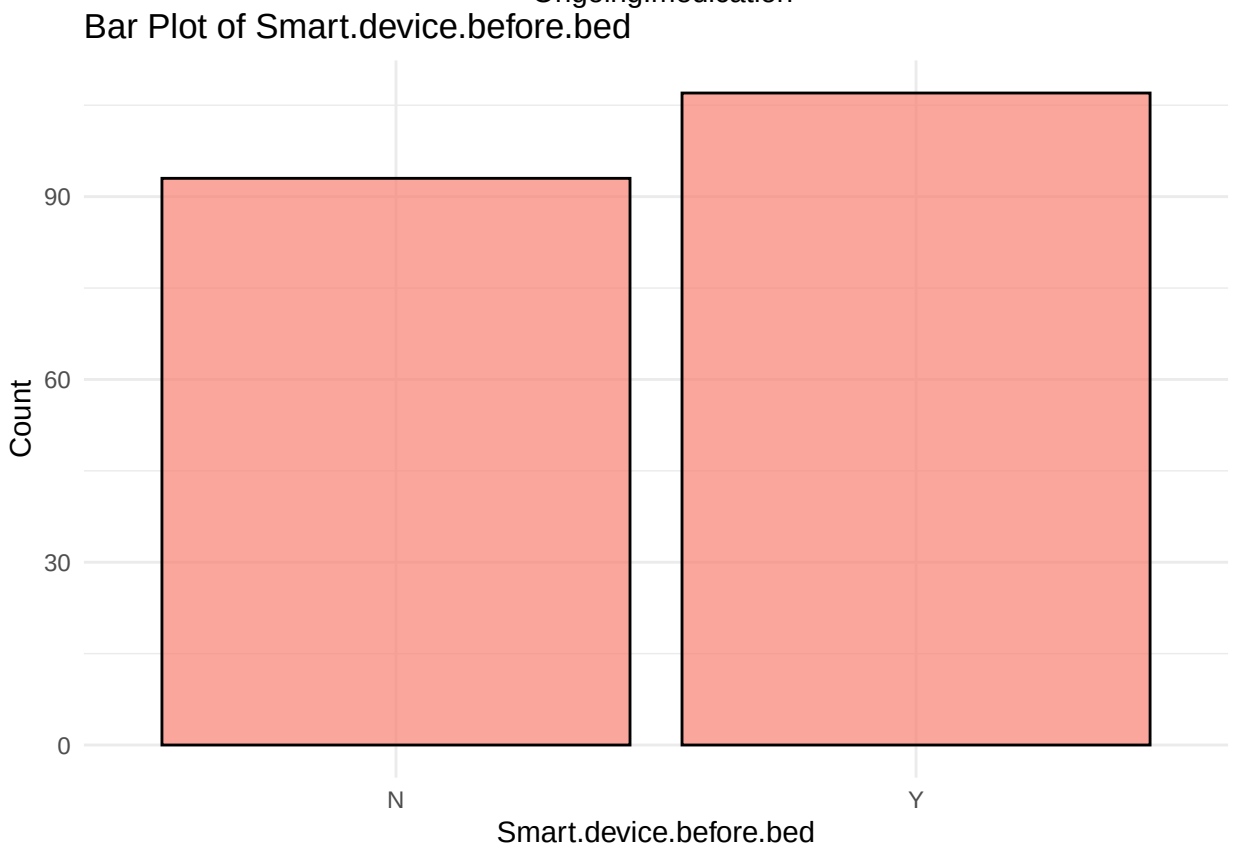

Bar Plot of Caffeine.consumption
100
75
50
25
0
Count
N
Y
Caffeine.consumption

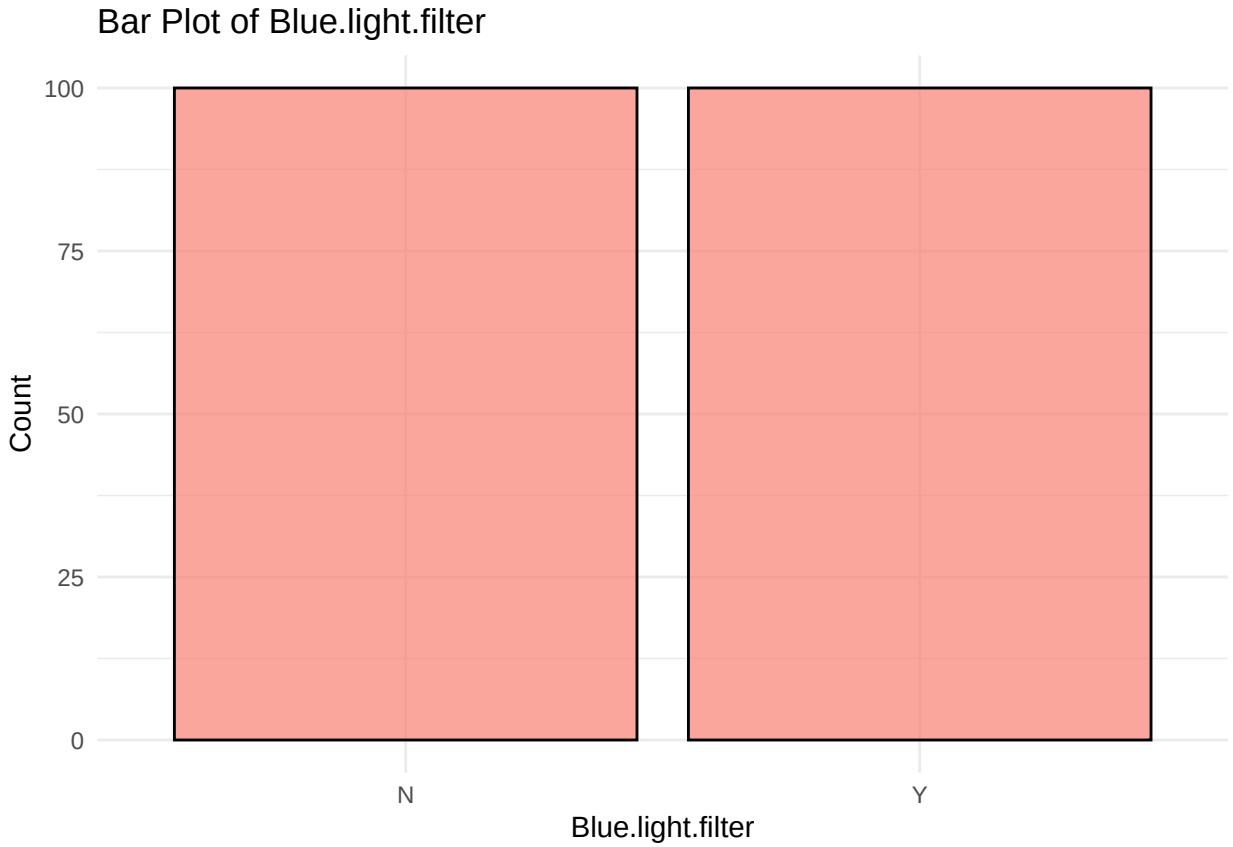

Bar Plot of Alcohol.consumption
90
60
30
0
Count
N
Y
Alcohol.consumption

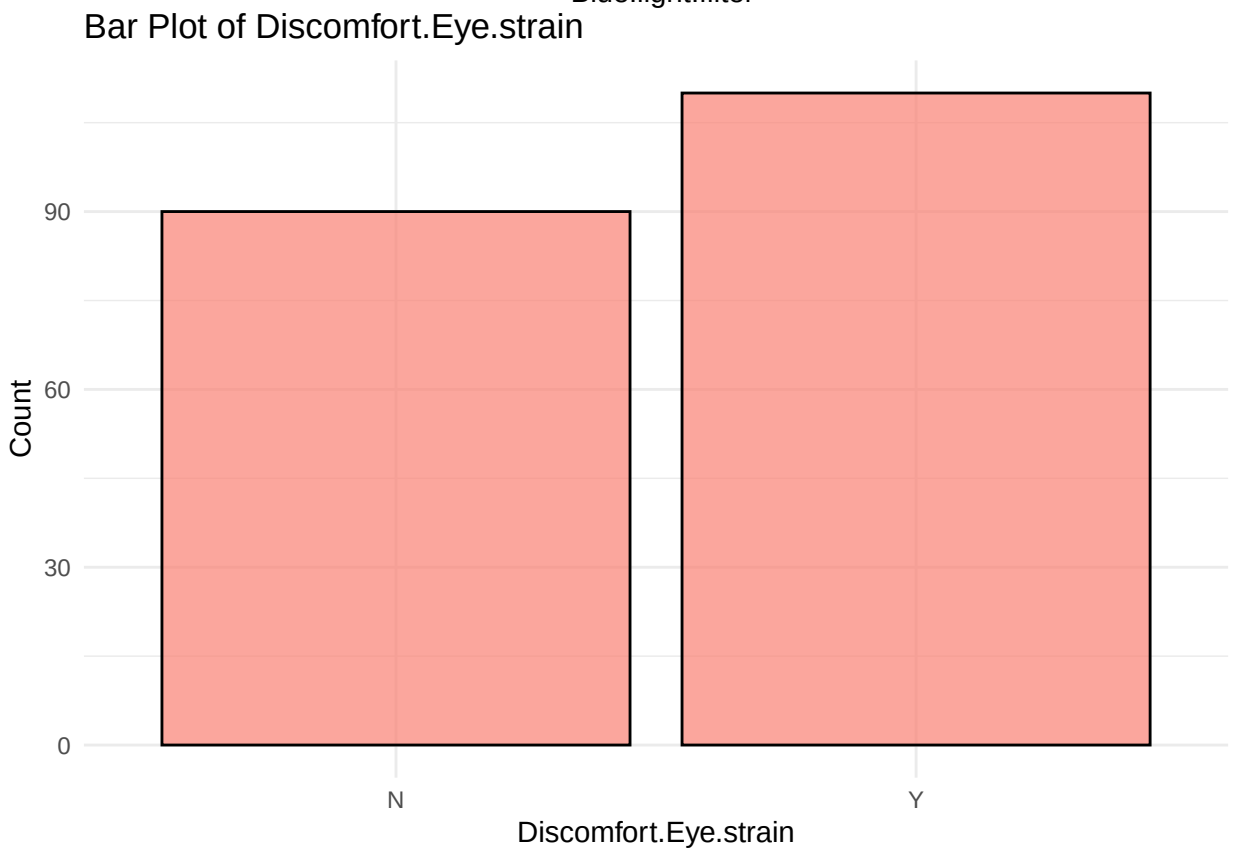

Bar Plot of Smoking
Count
90
60
30
0
N
Y
Smoking

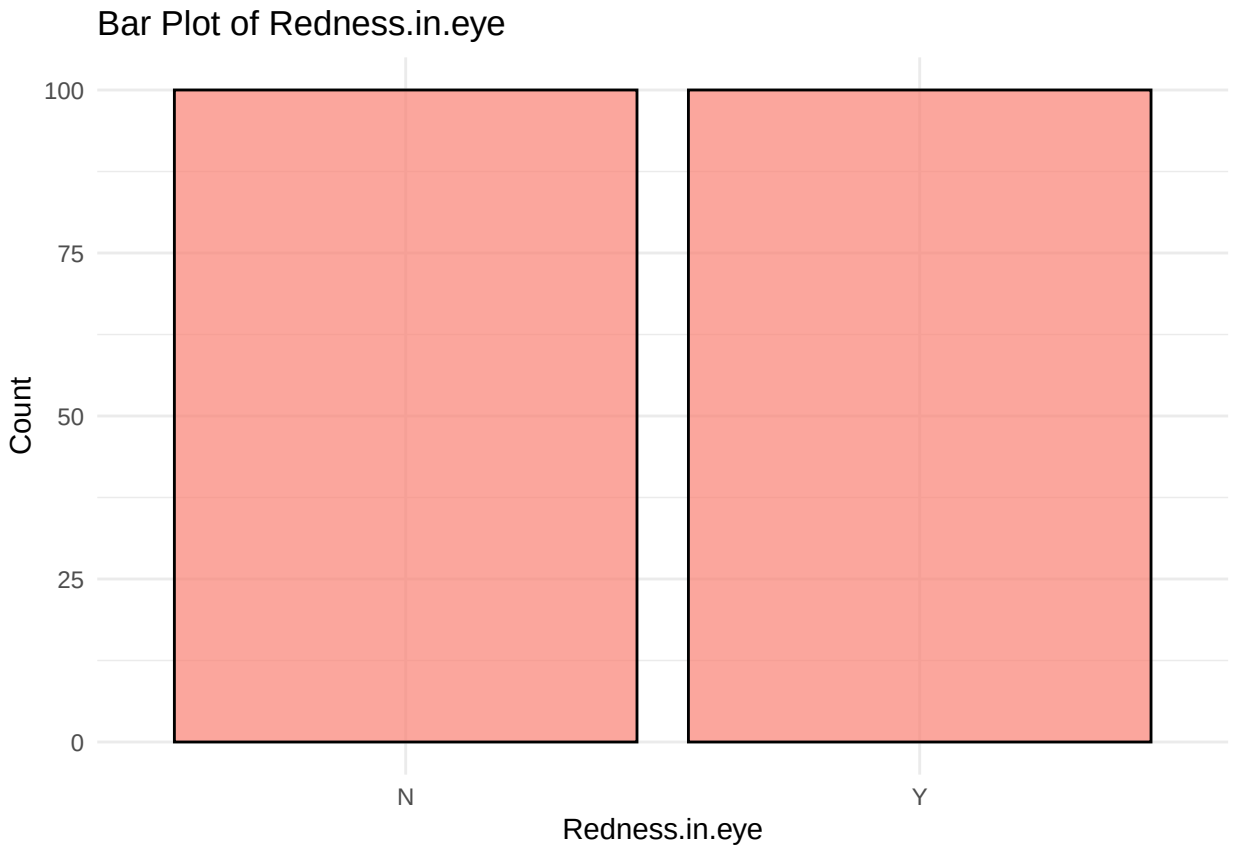

Bar Plot of Medical.issue
Count
90
60
30
0
N
Y
Medical.issue

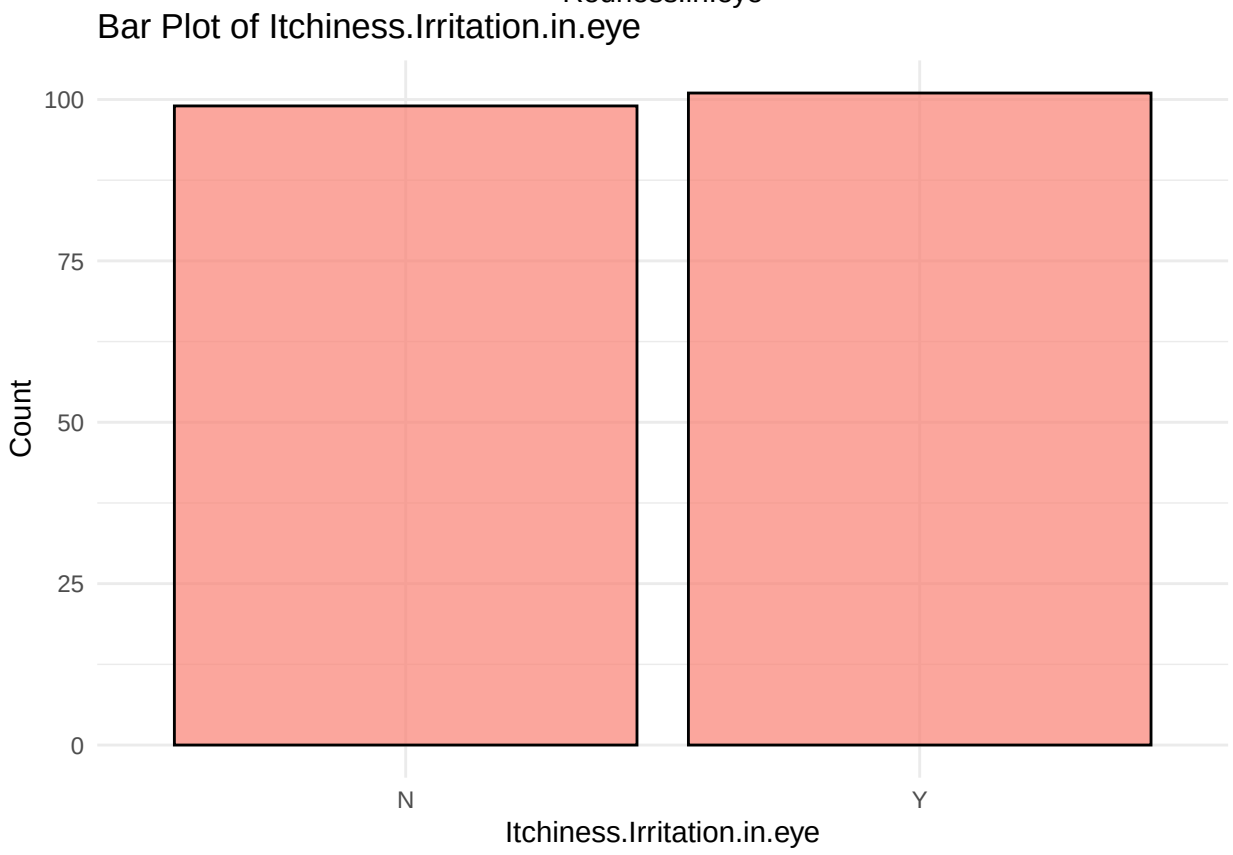

Bar Plot of Ongoing.medication
Count
100
75
50
25
0
N
Y
Ongoing.medication

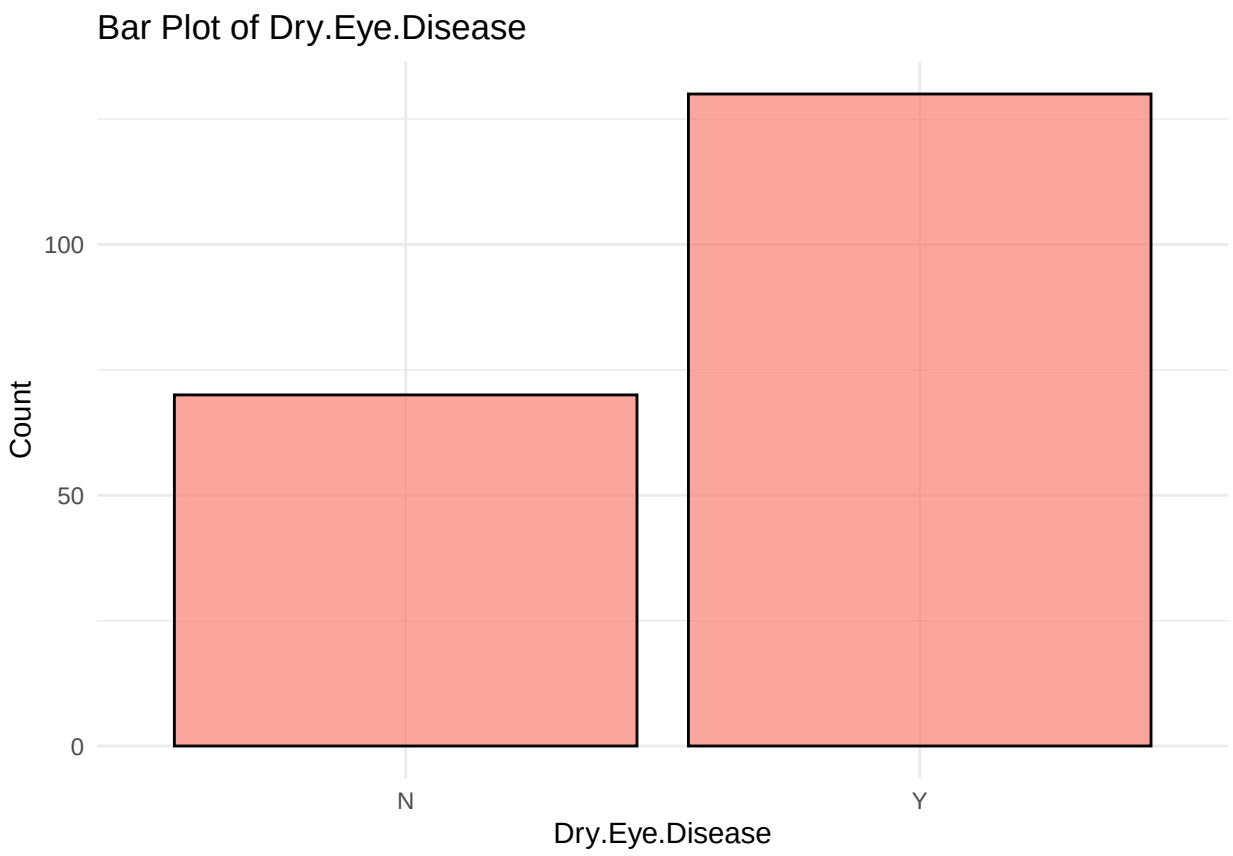

Bar Plot of Smart.device.before.bed
Count
90
60
30
0
N
Y
Smart.device.before.bed


Bar Plot of Blue.light.filter
100
75
50
25
0
Count
N
Y
Blue.light.filter


Bar Plot of Discomfort.Eye.strain
90
60
30
0
Count
N
Y
Discomfort.Eye.strain


Bar Plot of Redness.in.eye
Count
100
75
50
25
0
N
Y
Redness.in.eye


Bar Plot of Itchiness.Irritation.in.eye
Count
100
75
50
25
0
N
Y
Itchiness.Irritation.in.eye



5 Cluster Tendency

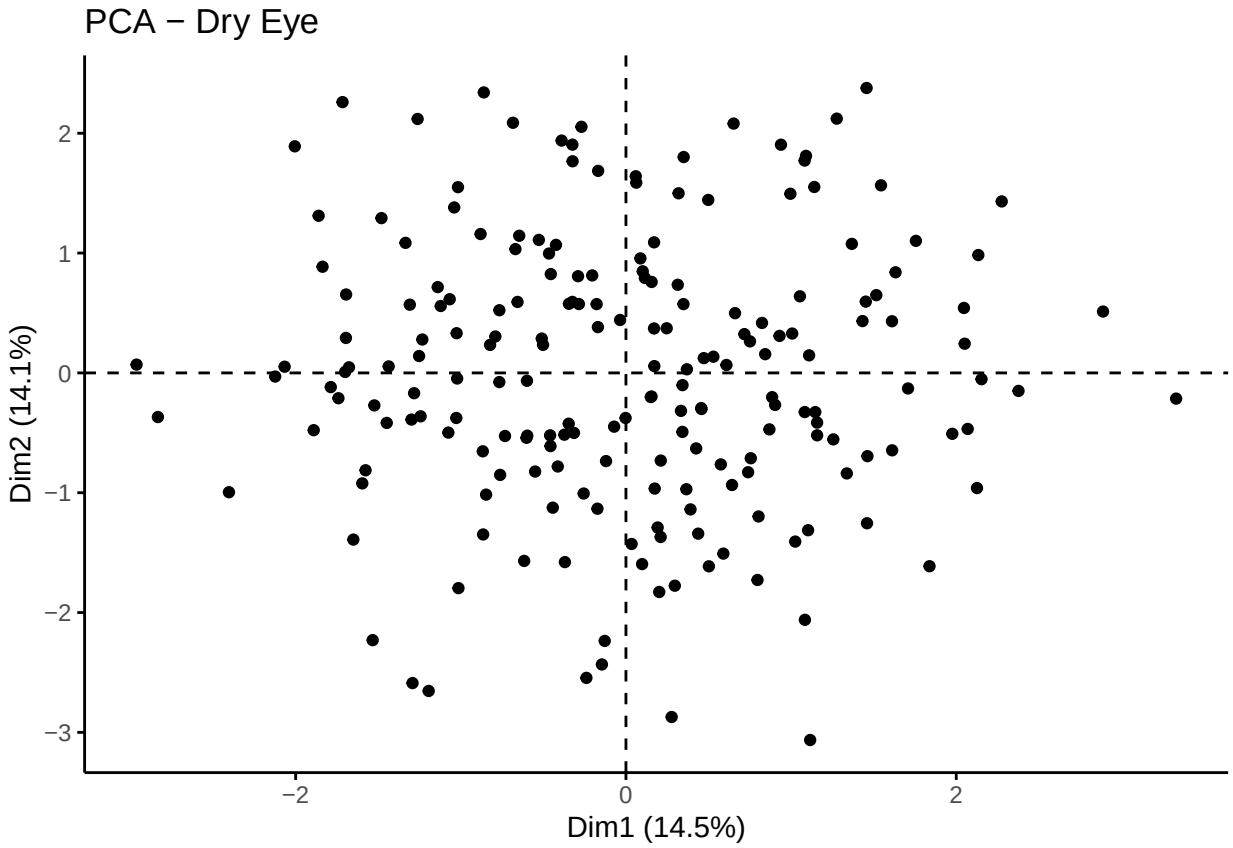
5.1 PCA Plot

```
sample.scale = scale(sample.numeric)
```

```
library(factoextra)
```

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

```
fviz_pca_ind(prcomp(sample.scale), title = "PCA - Dry Eye",
             palette = "jco",
             geom = "point", ggtheme = theme_classic(),
             legend = "bottom")
```

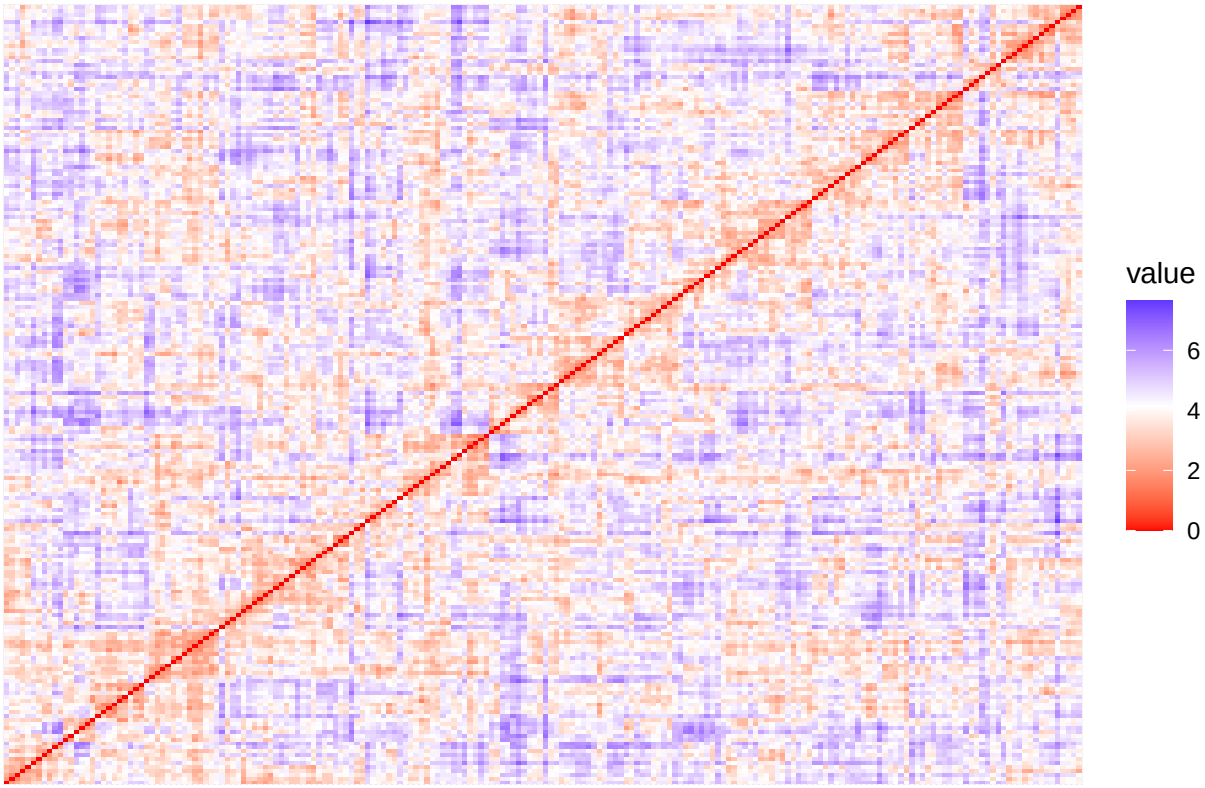


5.2 Hopkins stat

```
res <- get_clust_tendency(sample.scale, n = nrow(sample.scale)-1, graph = TRUE)
res$hopkins_stat
```

```
## [1] 0.4982412
```

```
fviz_dist(dist(sample.scale,method = "euclidean"), show_labels = FALSE) + labs(title = "Dry eye data")
```



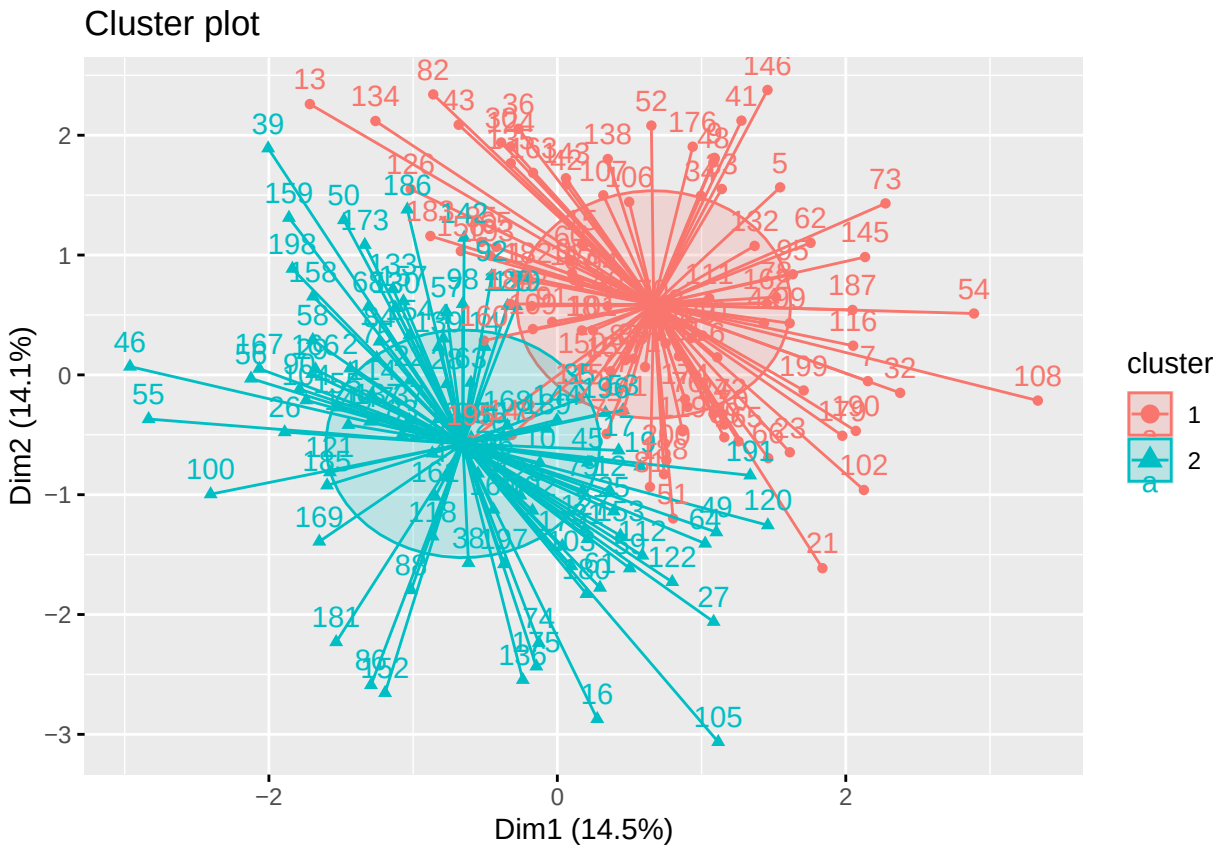
6 Cluster Exploration

6.1 Kmeans

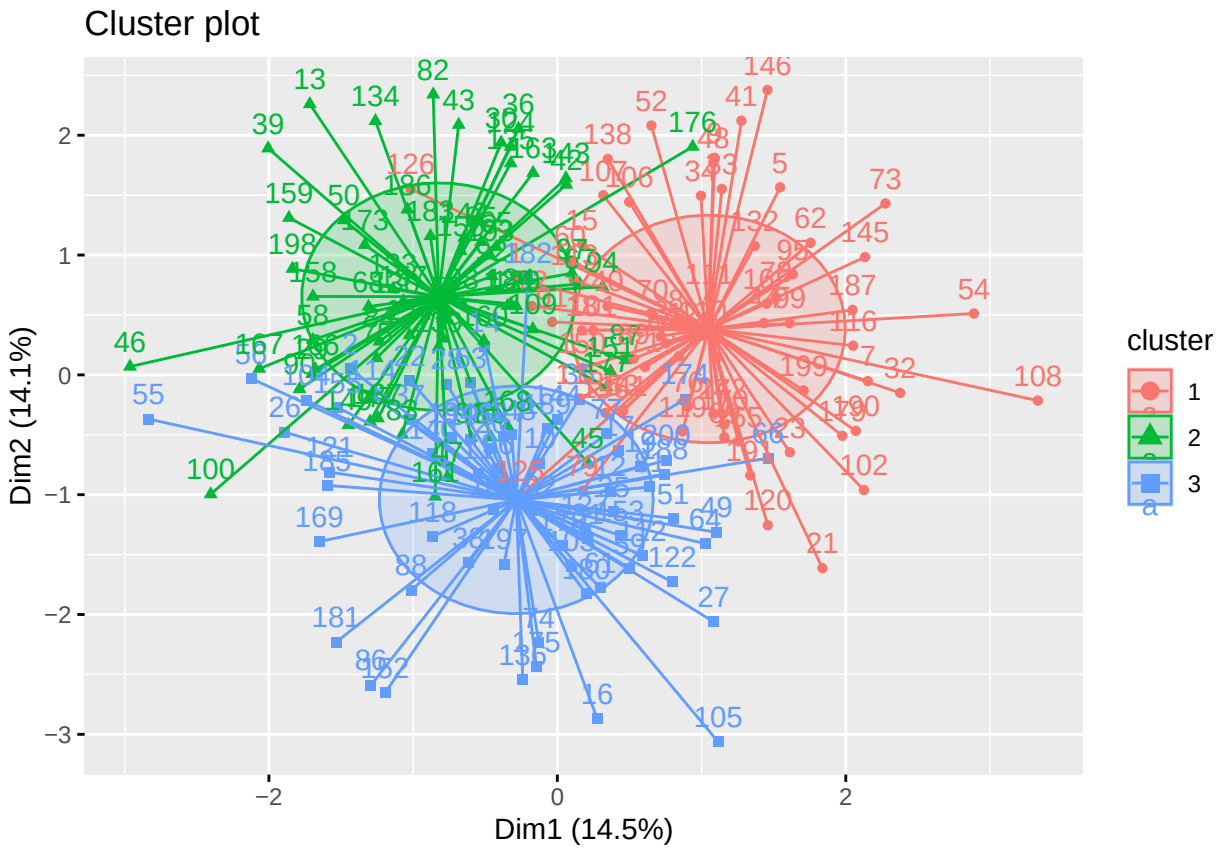
```
km2 <- kmeans(sample.scale, 2, nstart = 25)
km3 <- kmeans(sample.scale, 3, nstart = 25)
km4 <- kmeans(sample.scale, 4, nstart = 25)
```

6.1.1 Plots

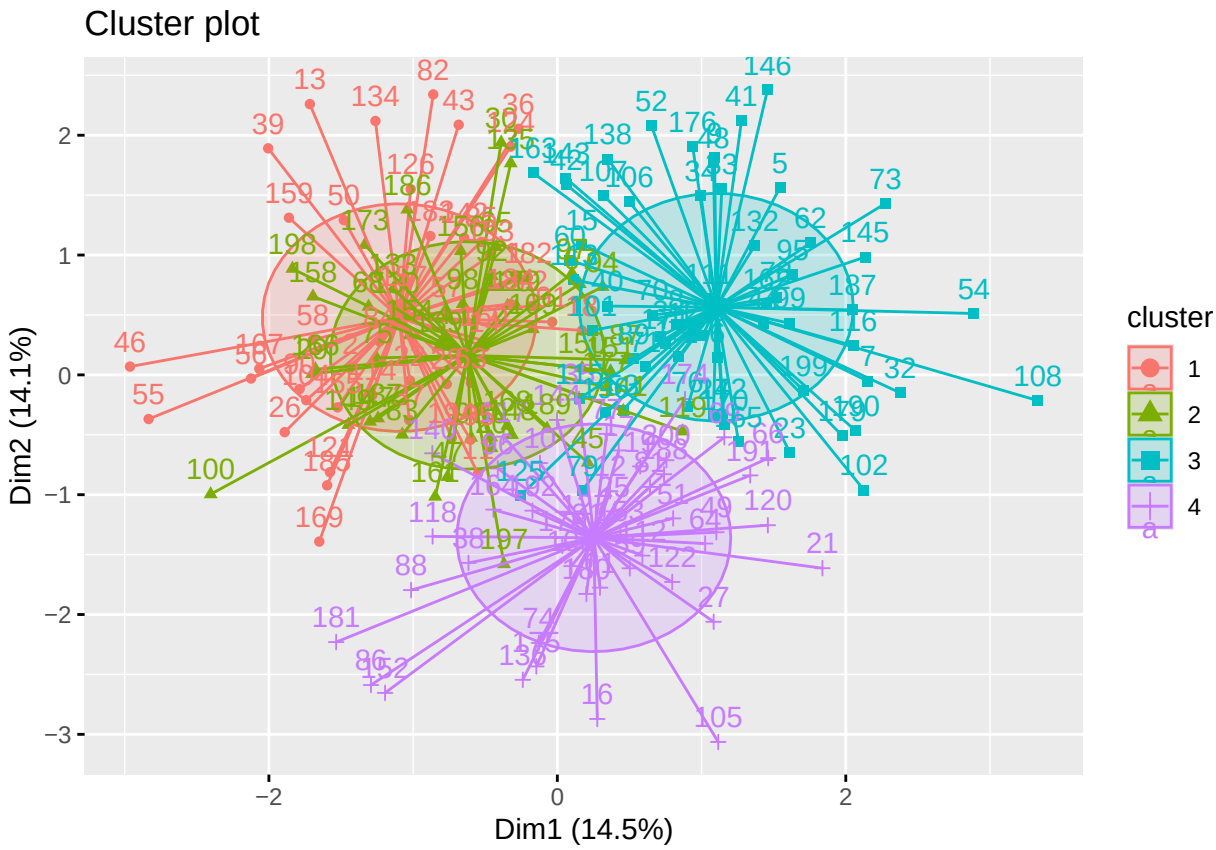
```
fviz_cluster(km2, data = sample.scale, ellipse.type = "euclid", star.plot = TRUE)
```




```
fviz_cluster(km3, data = sample.scale, ellipse.type = "euclid",star.plot = TRUE)
```




```
fviz_cluster(km4, data = sample.scale, ellipse.type = "euclid",star.plot = TRUE)
```

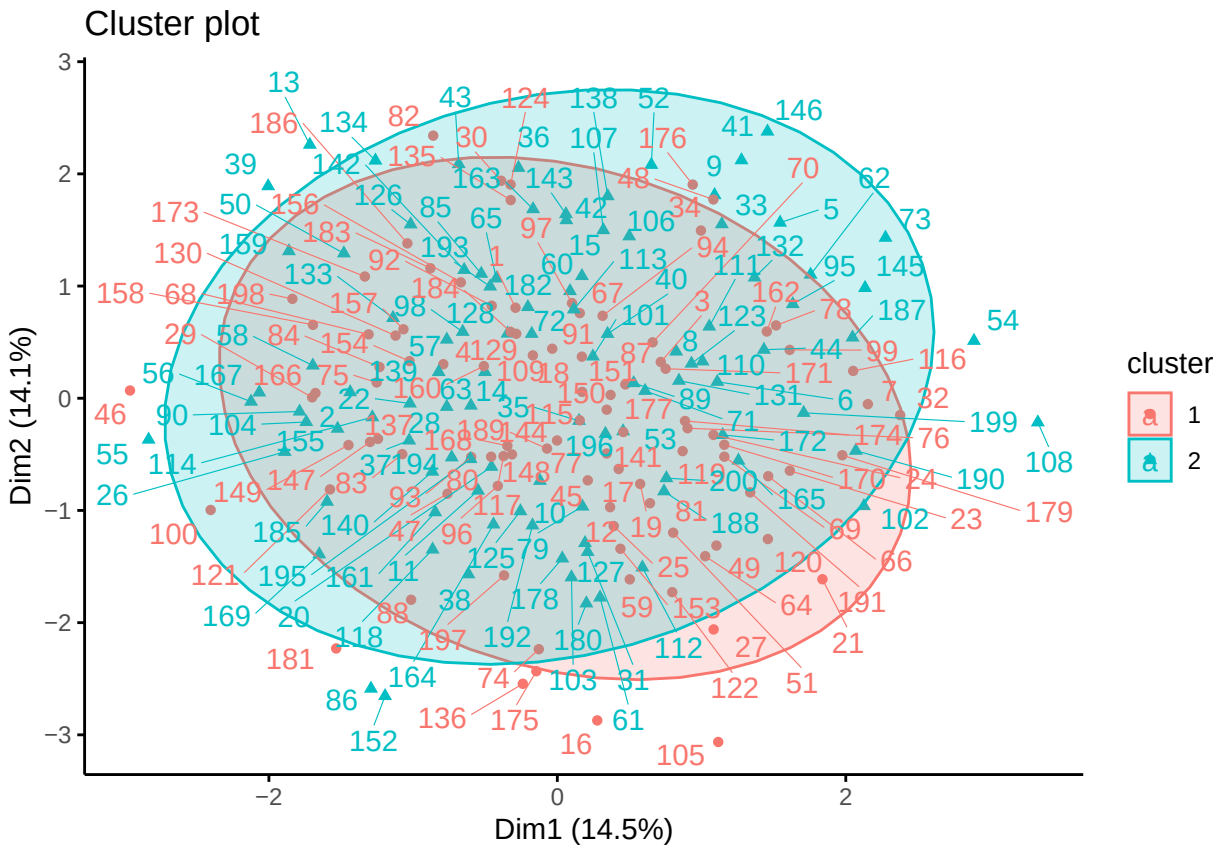


6.2 Kmedoids

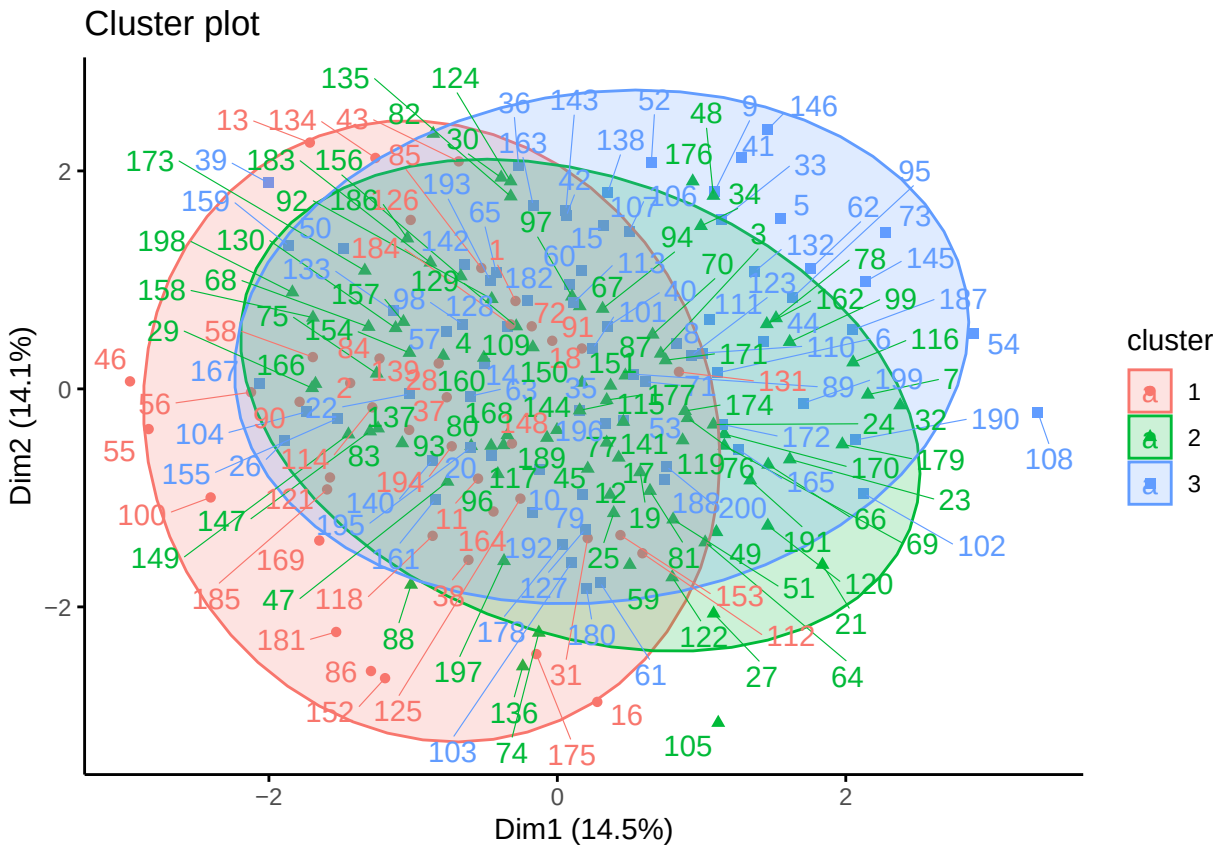
```
library(cluster)
pam.res2 <- pam(sample.scale, 2)
pam.res3 <- pam(sample.scale, 3)
pam.res4 <- pam(sample.scale, 4)
```

6.2.1 Plots

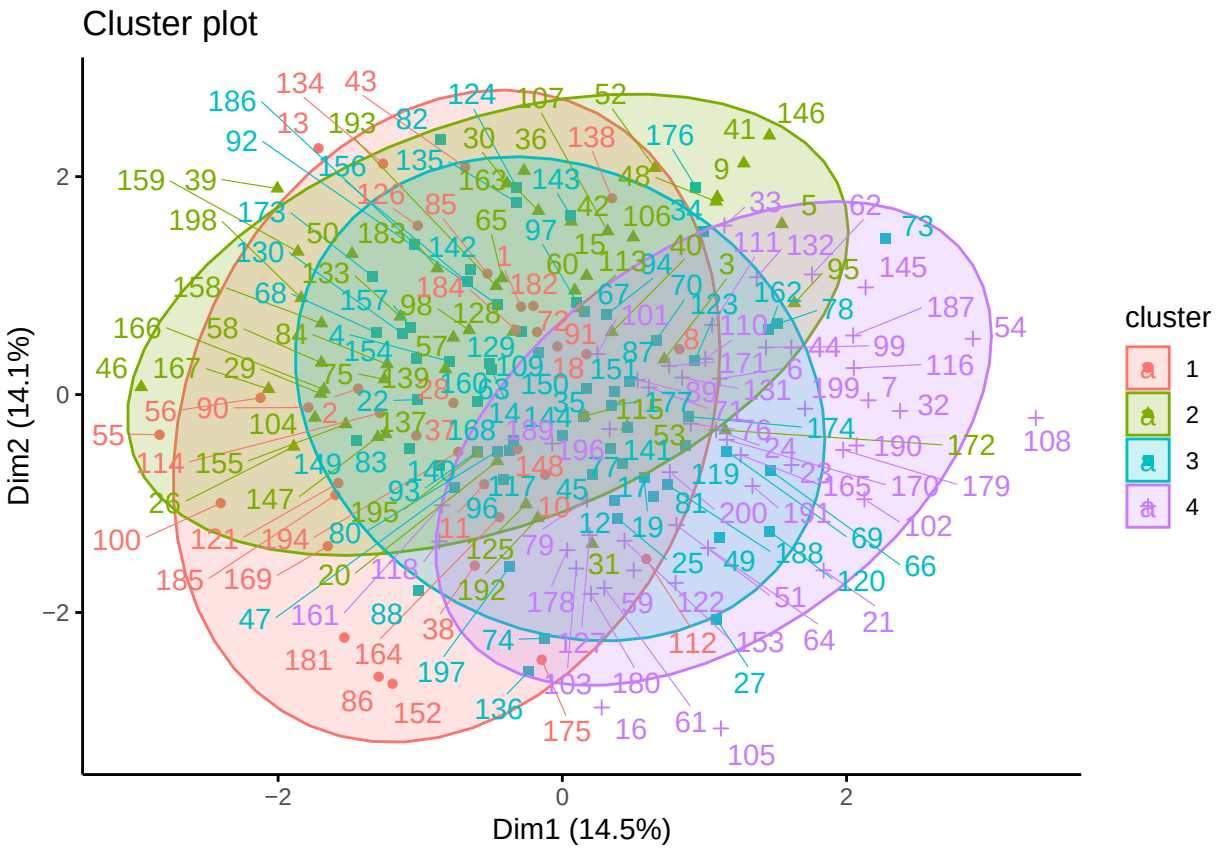
```
fviz_cluster(pam.res2,
             data = sample.scale,
             ellipse.type = "t", # Concentration ellipse
             repel = TRUE, # Avoid label overplotting (slow)
             ggtheme = theme_classic()
)
```




```
fviz_cluster(pam.res3,
             ellipse.type = "t", # Concentration ellipse
             repel = TRUE, # Avoid label overplotting (slow)
             ggtheme = theme_classic()
)
```




```
fviz_cluster(pam.res4,
             ellipse.type = "t", # Concentration ellipse
             repel = TRUE, # Avoid label overplotting (slow)
             ggtheme = theme_classic()
)
```



6.3 Hierarchical Clustering

```
res.dist <- dist(sample.scale, method = "euclidean")
```

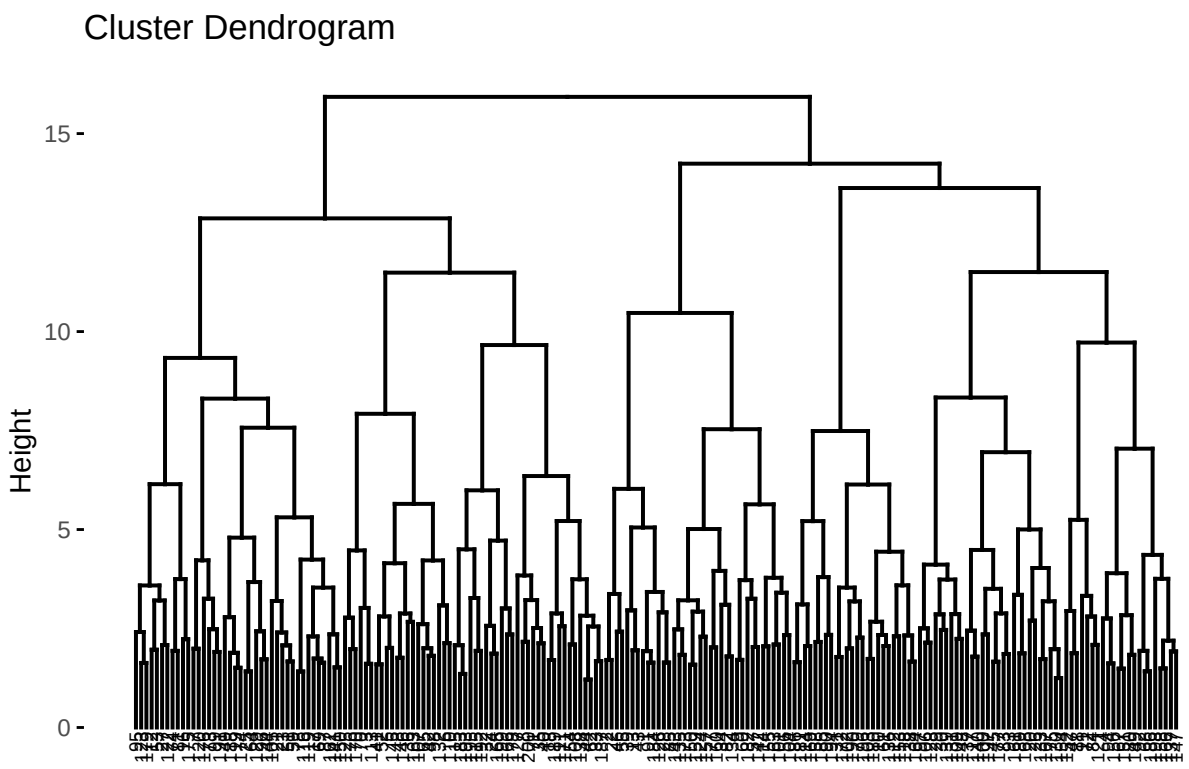
6.3.1 Ward

```
res.hc.ward <- hclust(d = res.dist, method = "ward.D2")
```

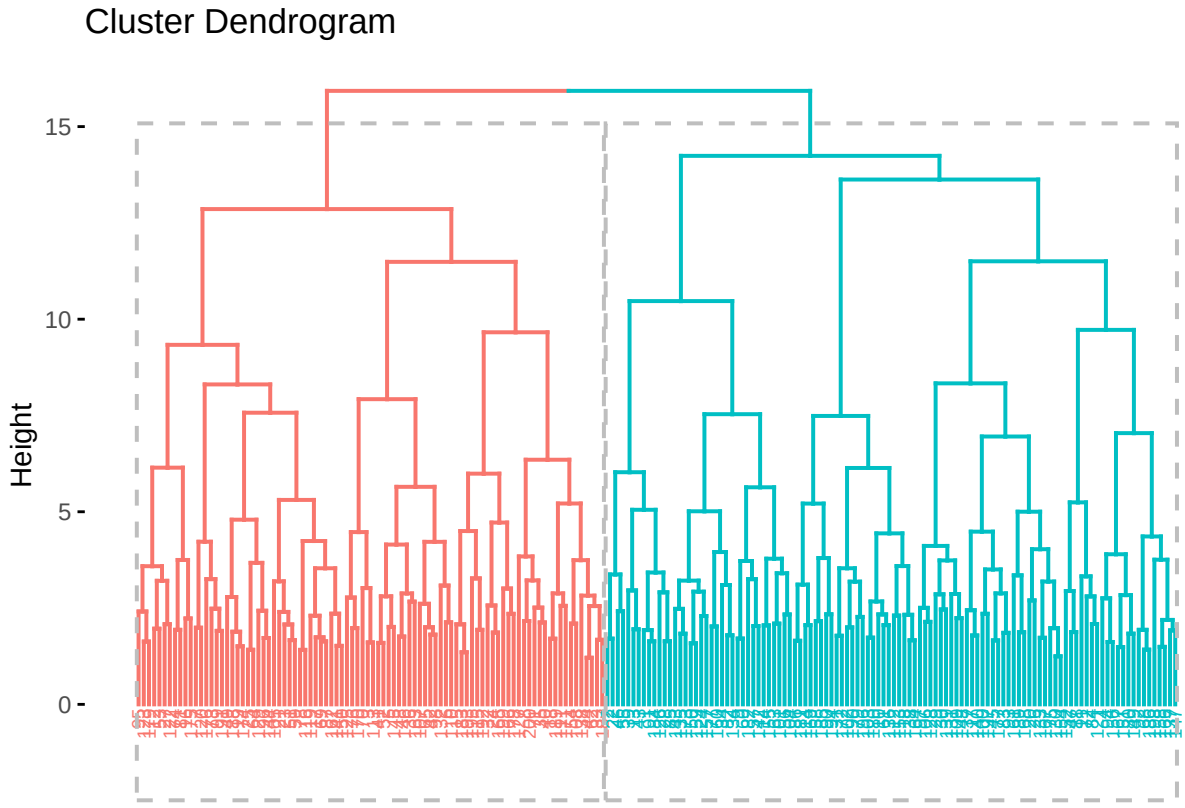
```
fviz_dend(res.hc.ward, cex = 0.5)
```

6.3.1.1 Plot

```
## 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.
```




```
fviz_dend(res.hc.ward, k = 2, # Cut in four groups
          cex = 0.5, # label size
          color_labels_by_k = TRUE, # color labels by groups
          rect = TRUE # Add rectangle around groups, "#FC4E07"
)
```

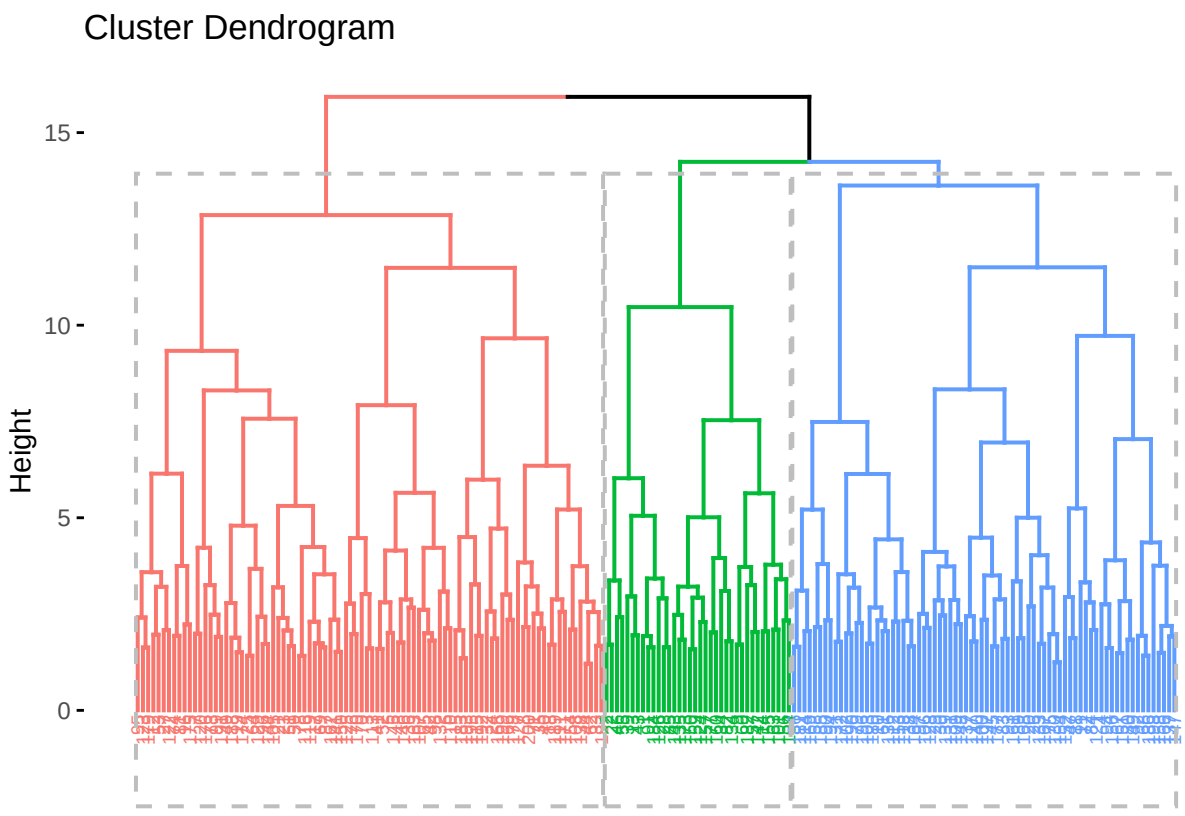


6.3.1.2 Cut

```
grp <- cutree(res.hc.ward, k = 2)
table(grp)
```

```
## grp
##   1   2
## 110  90
```

```
fviz_dend(res.hc.ward, k = 3, # Cut in four groups
          cex = 0.5, # label size
          color_labels_by_k = TRUE, # color labels by groups
          rect = TRUE # Add rectangle around groups, "#FC4E07"
)
```




```
grp <- cutree(res.hc.ward, k = 3)
table(grp)
```

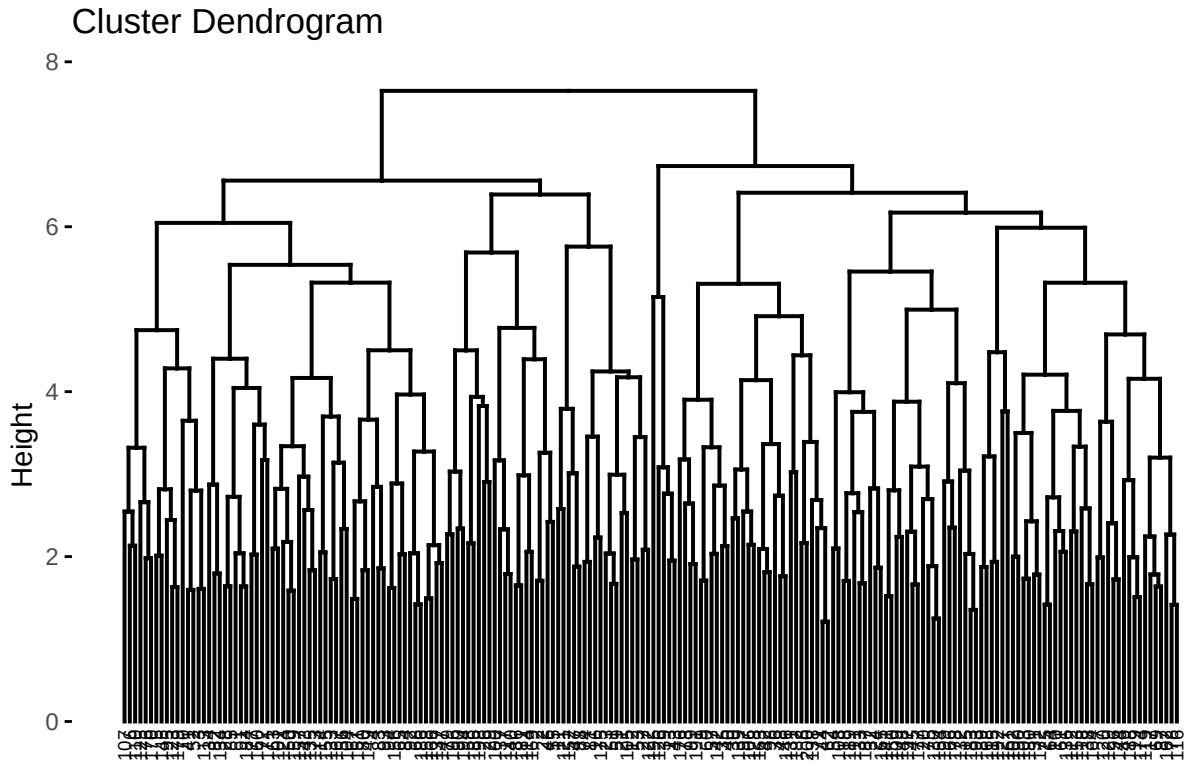
```
## grp
##  1  2  3
## 36 90 74
```

6.3.2 Complete

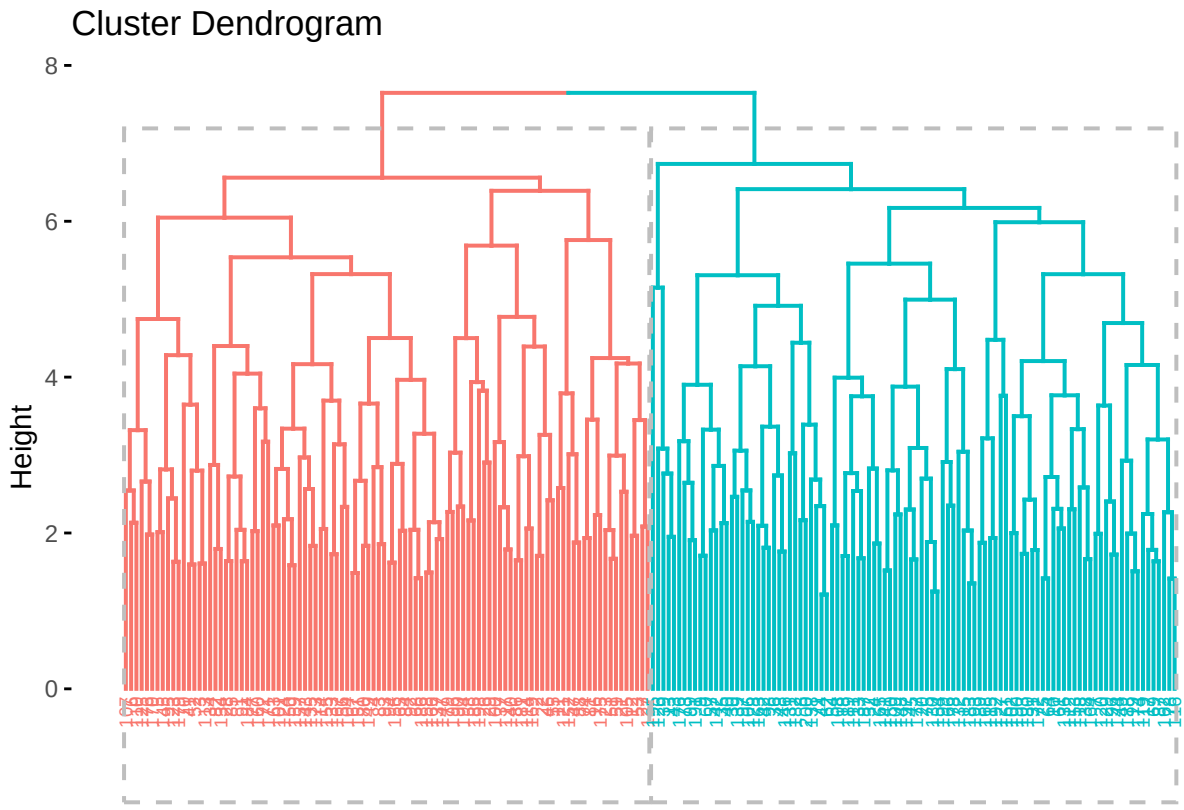
```
res.hc.com <- hclust(d = res.dist, method = "complete")
```

```
fviz_dend(res.hc.com, cex = 0.5)
```

6.3.2.1 Plot




```
fviz_dend(res.hc.com, k = 2, # Cut in four groups
          cex = 0.5, # label size
          color_labels_by_k = TRUE, # color labels by groups
          rect = TRUE # Add rectangle around groups, "#FC4E07"
)
```

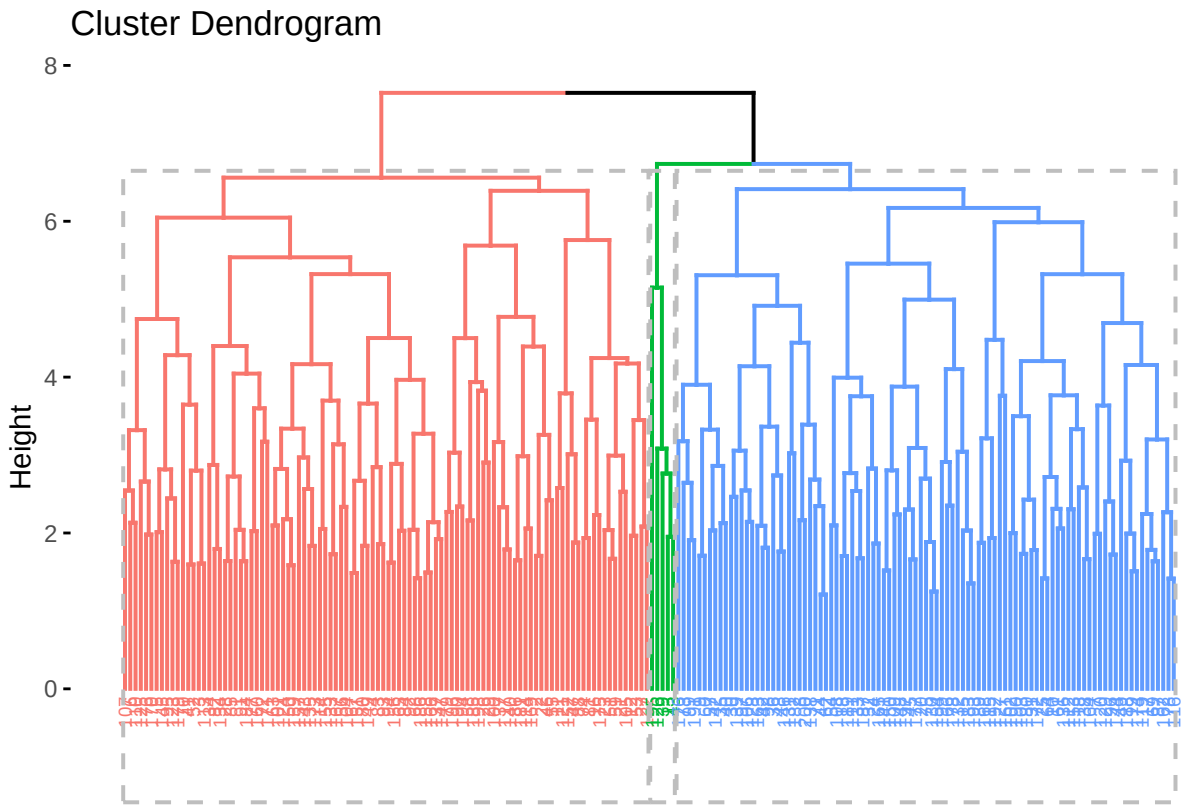


6.3.2.2 Cut

```
grp <- cutree(res.hc.com, k = 2)
table(grp)
```

```
## grp
##   1   2
## 100 100
```

```
fviz_dend(res.hc.com, k = 3, # Cut in four groups
          cex = 0.5, # label size
          color_labels_by_k = TRUE, # color labels by groups
          rect = TRUE # Add rectangle around groups, "#FC4E07"
)
```




```
grp <- cutree(res.hc.com, k = 3)
table(grp)
```

```
## grp
##   1   2   3
## 100  95   5
```

6.3.3 Comparing

```
library(dendextend)
```

```
## Warning: package 'dendextend' was built under R version 4.4.1
```

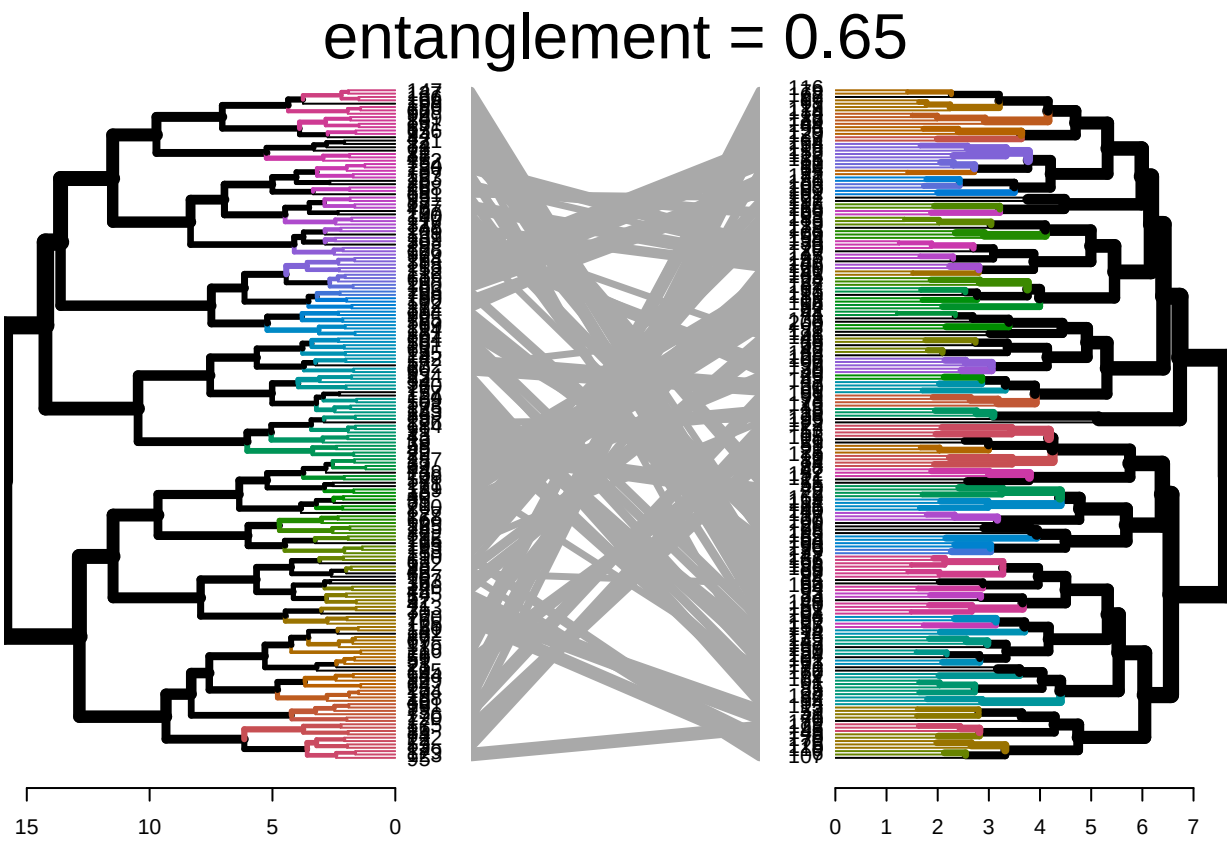
```
##
## ---------------------
## Welcome to dendextend version 1.19.0
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
dend1 <- as.dendrogram (res.hc.ward)
dend2 <- as.dendrogram (res.hc.com)
dend_list <- dendlist(dend1, dend2)
```

```
tanglegram(dend1, dend2,
highlight_distinct_edges = FALSE, # Turn-off dashed lines
common_subtrees_color_lines = FALSE, # Turn-off line colors
common_subtrees_color_branches = TRUE, # Color common branches
main = paste("entanglement =", round(entanglement(dend_list), 2))
)
```

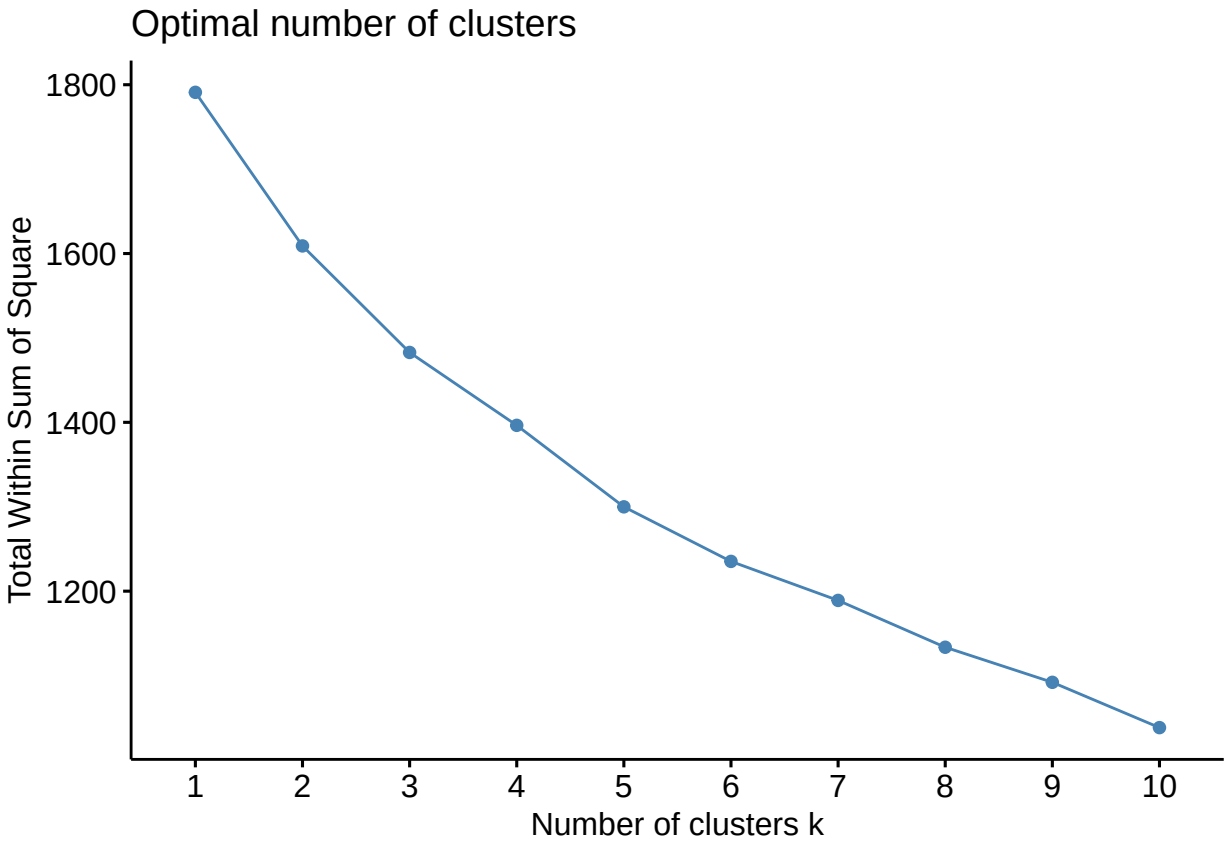


7 Choosing the Number of Clusters

7.1 Plotting

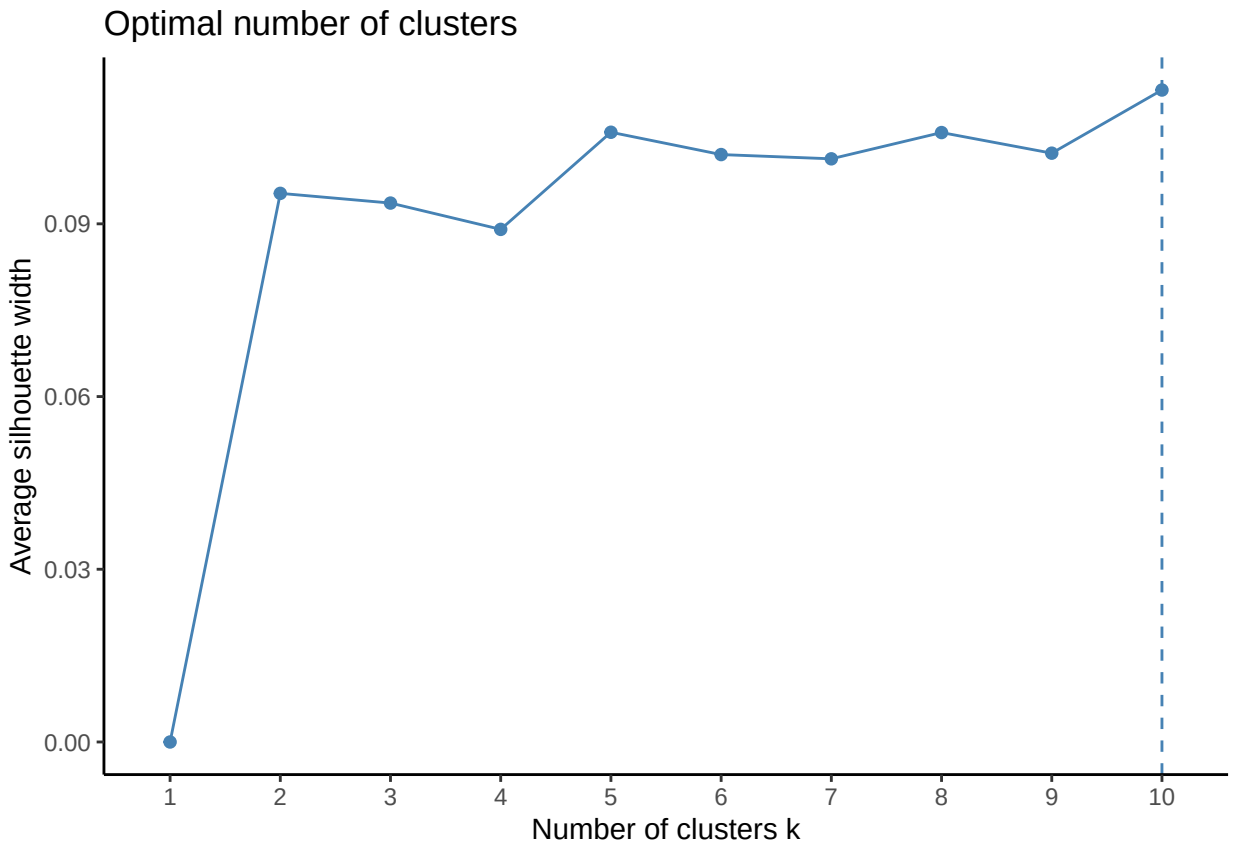
7.1.1 Within Sum of Square error

```
fviz_nbclust(sample.scale, kmeans, method = "wss")
```



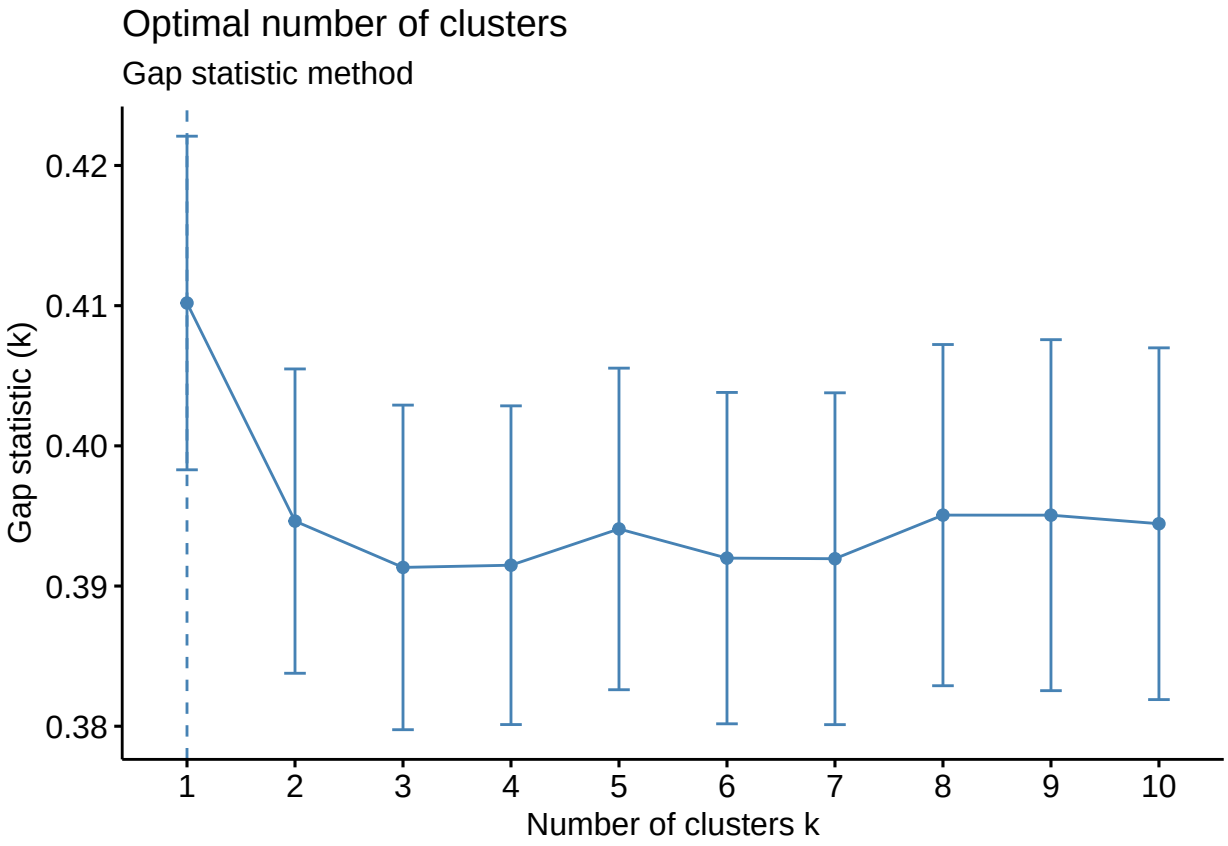
7.1.2 Silhoutte

```
fviz_nbclust(sample.scale, kmeans, method = "silhouette")+
  theme_classic()
```



7.1.3 Gap Statistic

```
set.seed(123)
fviz_nbclust(sample.scale, kmeans, nstart = 25, method = "gap_stat", nboot = 50)+
labs(subtitle = "Gap statistic method")
```



7.2 Internal Measures

```
library(clValid)
clmethods = c("hierarchical","kmeans","pam")
internal = clValid(sample.scale,
                   nClust = 2:4,
                   clMethods = clmethods,
                   validation = "internal",
                   method ="complete")
summary(internal)
```

```
##
## Clustering Methods:
##  hierarchical kmeans pam
##
## Cluster sizes:
##  2 3 4
##
## Validation Measures:
##                                  2        3        4
##
## hierarchical Connectivity  141.6738 147.0984 180.2702
##              Dunn            0.2445   0.2511   0.2568
##              Silhouette      0.0422   0.0311   0.0326
## kmeans       Connectivity  108.9905 149.9813 178.9163
##              Dunn            0.2115   0.2235   0.2451
##              Silhouette      0.0939   0.0884   0.1023
```

```
## pam          Connectivity  147.4833 184.8659 223.8028
##              Dunn            0.1769   0.1769   0.1970
##              Silhouette      0.0655   0.0667   0.0644
##
## Optimal Scores:
##
##              Score    Method       Clusters
## Connectivity 108.9905 kmeans       2
## Dunn           0.2568 hierarchical 4
## Silhouette     0.1023 kmeans       4
```

7.3 Stability Measures

```
stability = clValid(sample.scale,
                    nClust = 2:4,
                    clMethods = clmethods,
                    validation = "stability",
                    method = "complete")
summary(stability)
```

```
##
## Clustering Methods:
##  hierarchical kmeans pam
##
## Cluster sizes:
##  2 3 4
##
## Validation Measures:
##                          2      3      4
##
## hierarchical APN  0.4000 0.5995 0.6452
##              AD   4.1315 4.1167 4.0777
##              ADM  0.8868 1.1681 1.3771
##              FOM  1.0001 1.0008 0.9987
## kmeans       APN  0.4270 0.5511 0.5237
##              AD   4.0929 4.0387 3.9138
##              ADM  1.1371 1.4104 1.4000
##              FOM  1.0013 1.0004 1.0011
## pam          APN  0.3235 0.5192 0.5404
##              AD   4.0695 4.0504 3.9724
##              ADM  0.7583 1.1892 1.2582
##              FOM  1.0004 1.0032 1.0010
##
## Optimal Scores:
##
##     Score  Method       Clusters
## APN 0.3235 pam          2
## AD  3.9138 kmeans       4
## ADM 0.7583 pam          2
## FOM 0.9987 hierarchical 4
```

8 Cluster Validation

8.1 Silhoutte coefficient

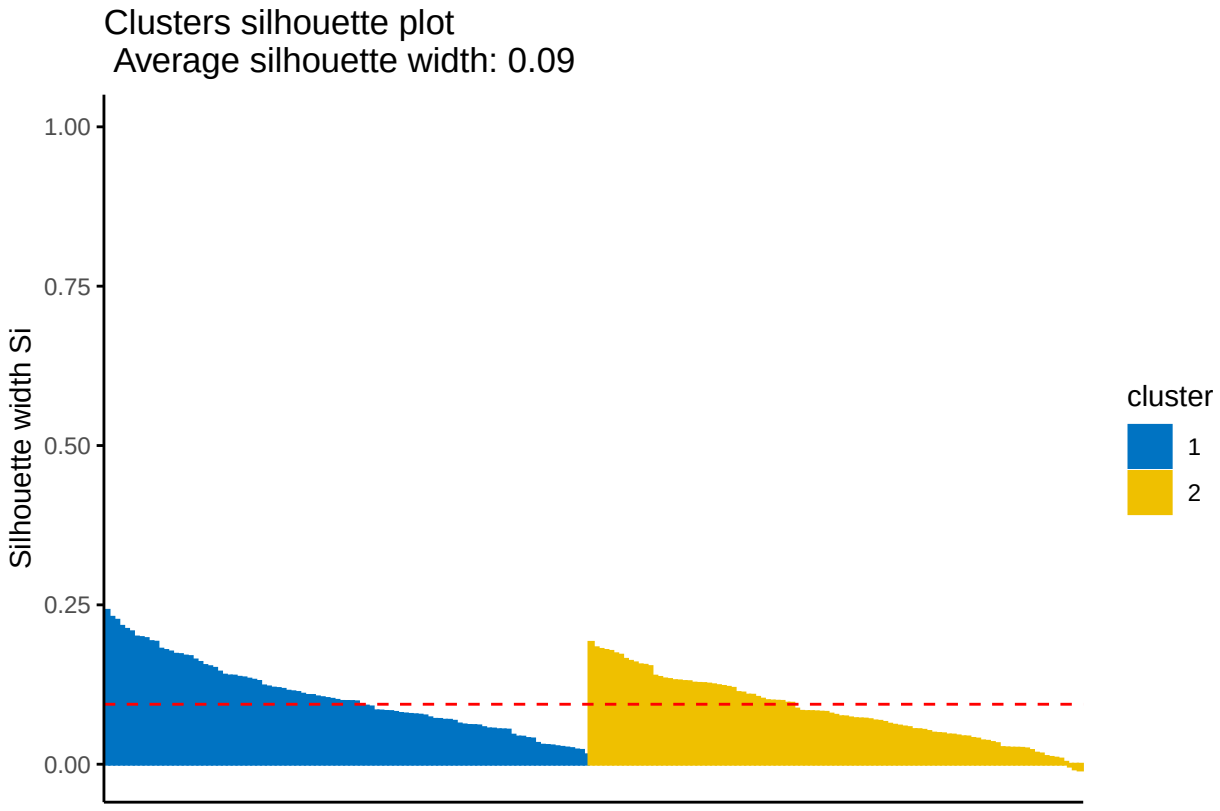
```
# K-means clustering
km.res2 <- eclust(sample.scale, "kmeans", k = 2, nstart = 25, graph = FALSE)
#km.res3 <- eclust(sample.scale, "kmeans", k = 3, nstart = 25, graph = FALSE)
km.res4 <- eclust(sample.scale, "kmeans", k = 4, nstart = 25, graph = FALSE)
# Visualize k-means clusters
#fviz_cluster(km.res, geom = "point", ellipse.type = "euclid", palette = "jco", ggtheme = theme_minimal
#ellipse.type = "euclid, t"
#star.plot=TRUE
```

```
hc.res2 <- eclust(sample.scale, "hclust", k = 2, hc_metric = "euclidean", hc_method = "complete", graph
#hc.res3 <- eclust(sample.scale, "hclust", k = 3, hc_metric = "euclidean", hc_method = "complete", grap
hc.res4 <- eclust(sample.scale, "hclust", k = 4, hc_metric = "euclidean", hc_method = "complete", graph
# Visualize dendrograms
#fviz_dend(hc.res, show_labels = FALSE, palette = "jco", as.ggplot = TRUE)
```

```
pam.res2 <- eclust(sample.scale, "pam", k = 2, nstart = 25, graph = FALSE)
#pam.res3 <- eclust(sample.scale, "pam", k = 3, nstart = 25, graph = FALSE)
pam.res4 <- eclust(sample.scale, "pam", k = 4, nstart = 25, graph = FALSE)
```

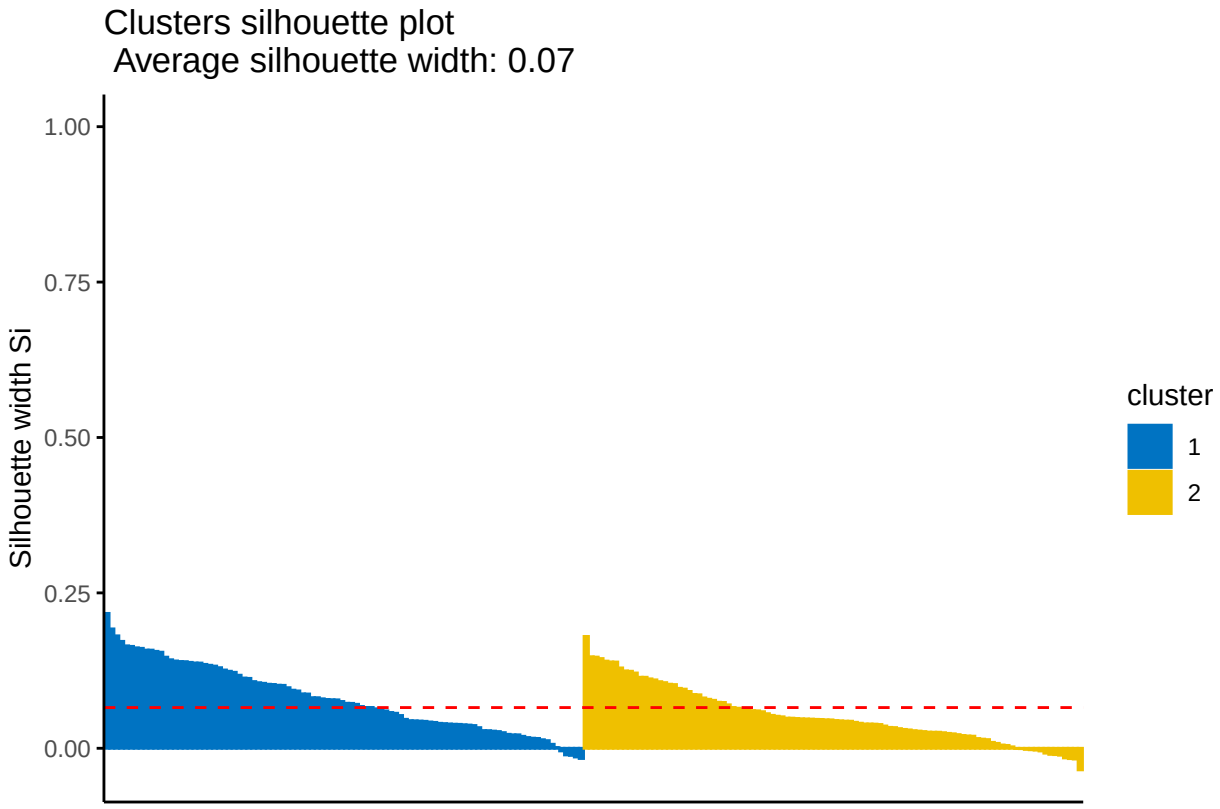
```
fviz_silhouette(km.res2, palette = "jco", ggtheme = theme_classic())
```

```
##   cluster size ave.sil.width
## 1       1   99          0.11
## 2       2  101          0.08
```



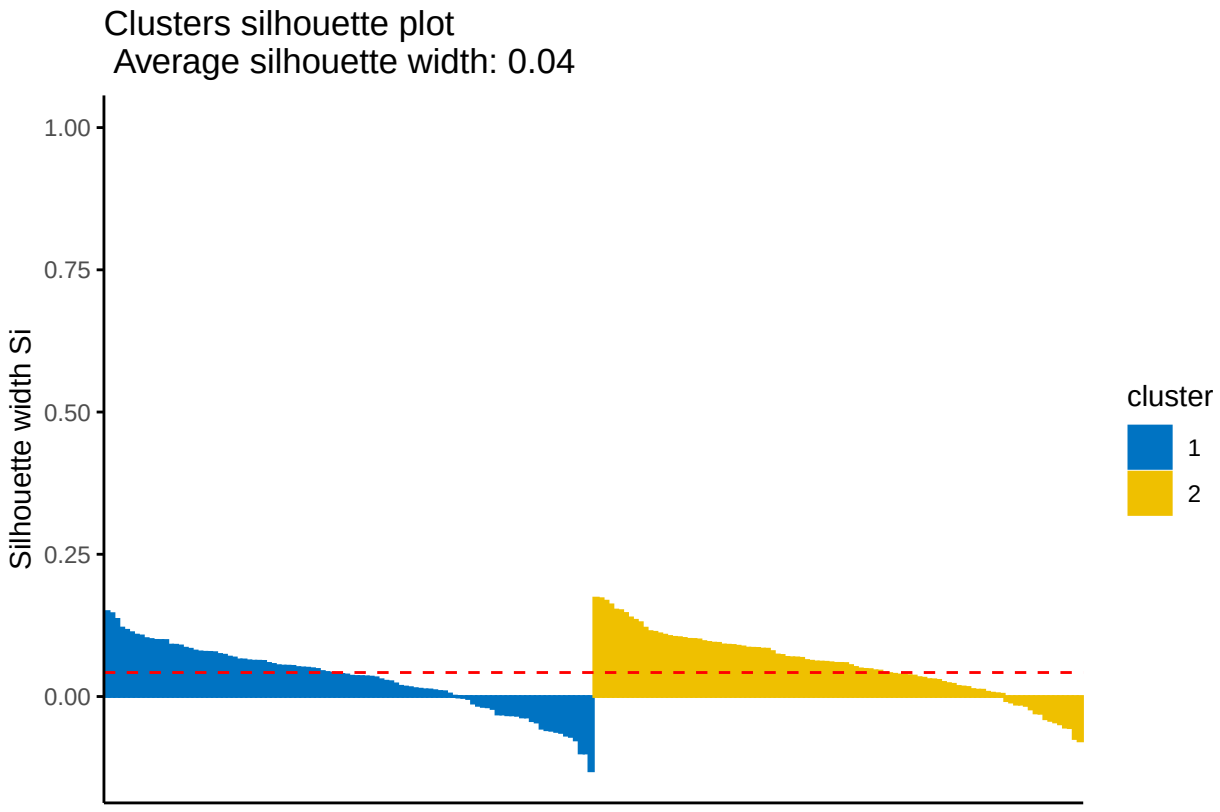
```
fviz_silhouette(pam.res2, palette = "jco", ggtheme = theme_classic())
```

```
##   cluster size ave.sil.width
## 1       1   98          0.08
## 2       2  102          0.05
```



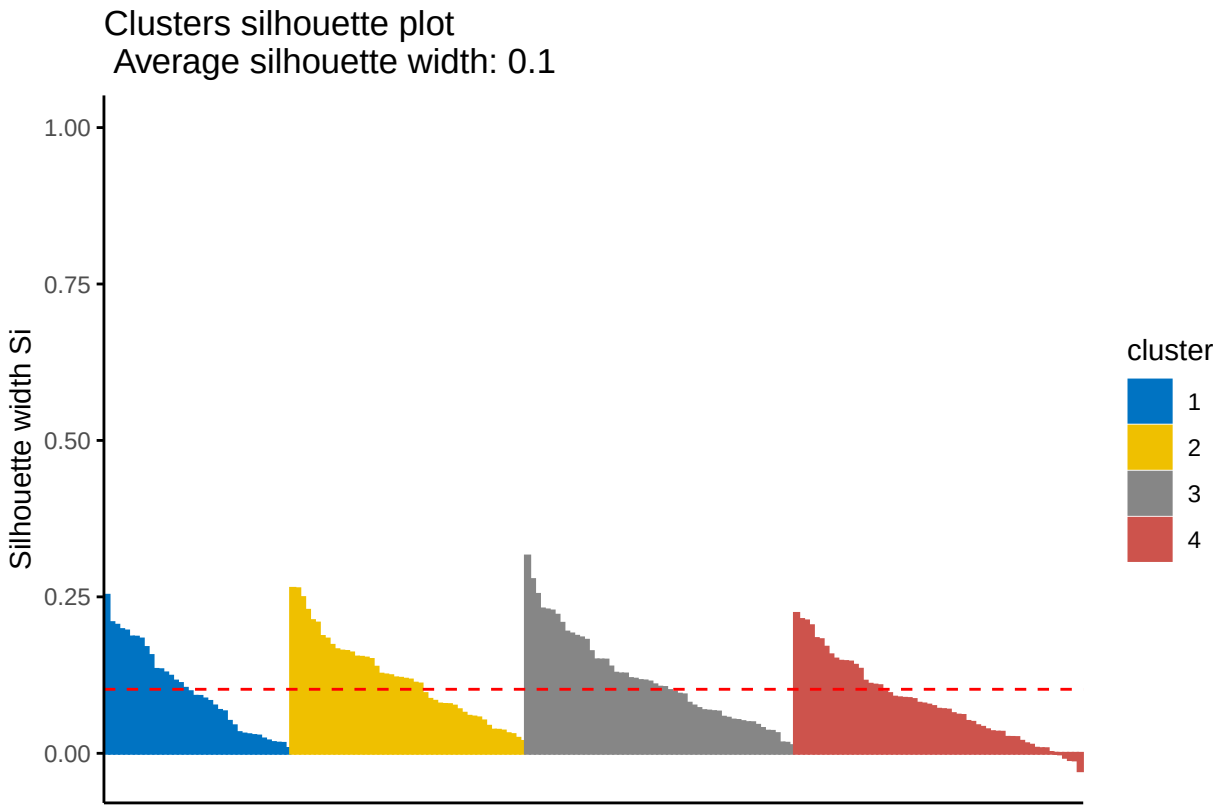

```
fviz_silhouette(hc.res2, palette = "jco", ggtheme = theme_classic())
```

```
##   cluster size ave.sil.width
## 1       1  100          0.03
## 2       2  100          0.05
```



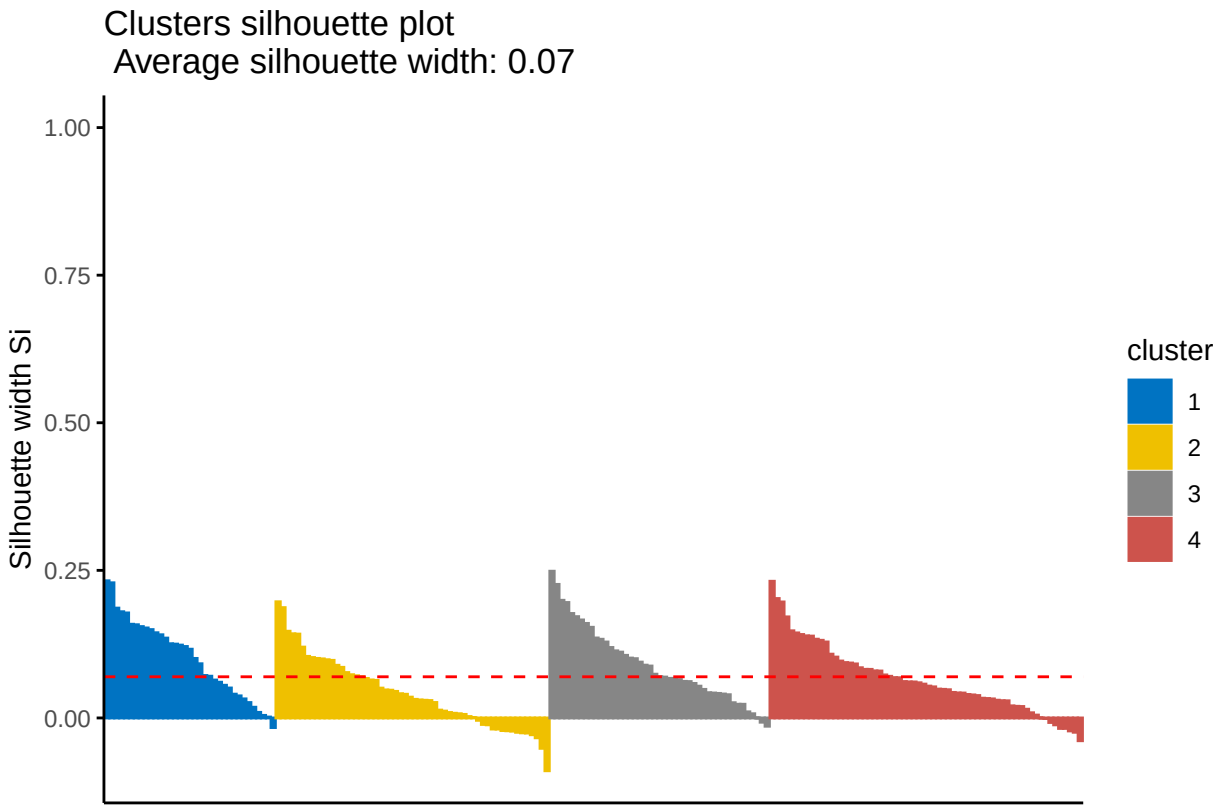

```
#fviz_silhouette(km.res3, palette = "jco", ggtheme = theme_classic())
#fviz_silhouette(pam.res3, palette = "jco", ggtheme = theme_classic())
#fviz_silhouette(hc.res3, palette = "jco", ggtheme = theme_classic())

fviz_silhouette(km.res4, palette = "jco", ggtheme = theme_classic())
```

```
##   cluster size ave.sil.width
## 1       1   38          0.10
## 2       2   48          0.12
## 3       3   55          0.12
## 4       4   59          0.08
```



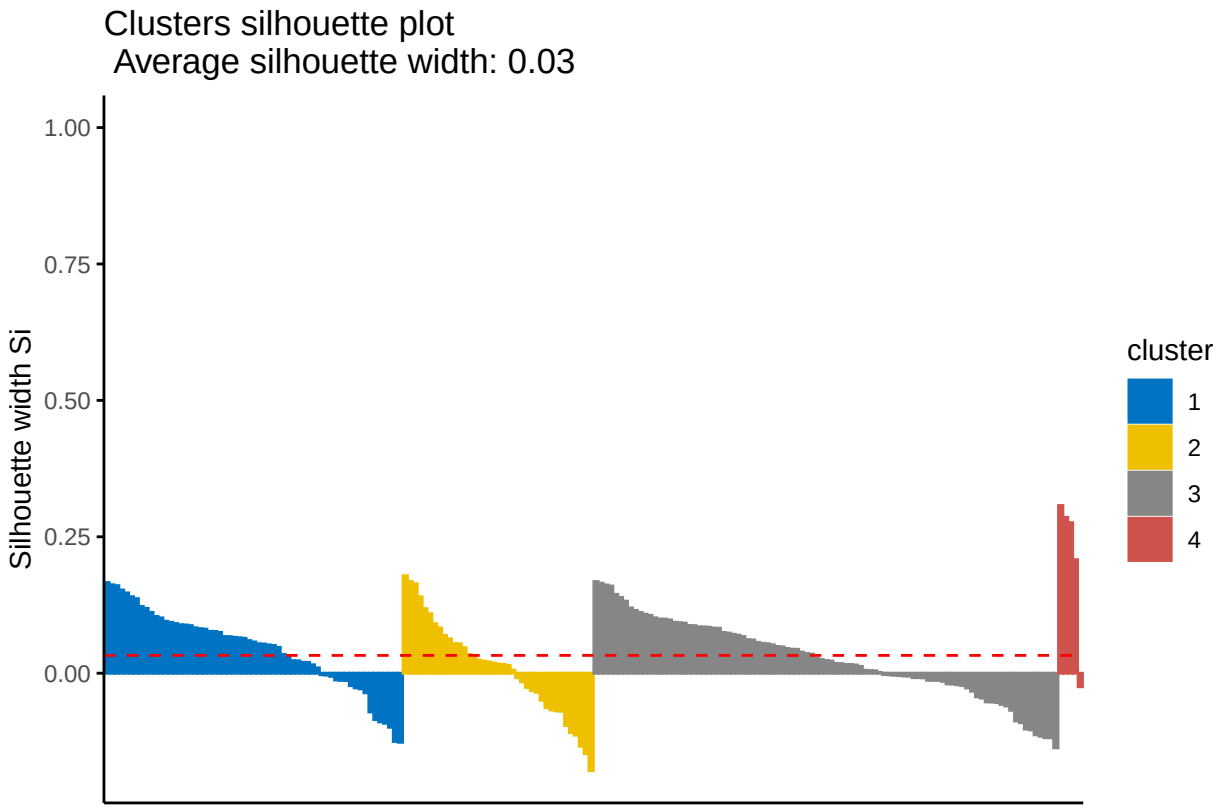

```
fviz_silhouette(pam.res4, palette = "jco", ggtheme = theme_classic())
```

```
##   cluster size ave.sil.width
## 1       1   35          0.10
## 2       2   56          0.04
## 3       3   45          0.09
## 4       4   64          0.06
```




```
fviz_silhouette(hc.res4, palette = "jco", ggtheme = theme_classic())
```

```
##   cluster size ave.sil.width
## 1       1   61          0.04
## 2       2   39          0.01
## 3       3   95          0.03
## 4       4    5          0.21
```




```
compute_negative_silhouette = function(km_res) {

  sil = as.data.frame(km_res$silinfo)
  neg_sil = which(sil[, "widths.sil_width"] < 0)
  percentage = round(length(neg_sil) / 200 * 100, 2)

  return(percentage)
  print(percentage)
}
```

```
compute_negative_silhouette(km.res2)
```

```
## [1] 1.5
```

```
compute_negative_silhouette(km.res4)
```

```
## [1] 2.5
```

```
compute_negative_silhouette(pam.res2)
```

```
## [1] 9.5
```

```
compute_negative_silhouette(pam.res4)
```

```
## [1] 14
```

```
compute_negative_silhouette(hc.res2)
```

```
## [1] 22
```

```
compute_negative_silhouette(hc.res4)
```

```
## [1] 35
```

```
sil= as.data.frame(km.res2$silinfo)
neg_sil = which(sil[, 'widths.sil_width'] <0)
neg_sil
```

```
## [1] 198 199 200
```

```
percentage = round(length(neg_sil)/200*100,2)
percentage
```

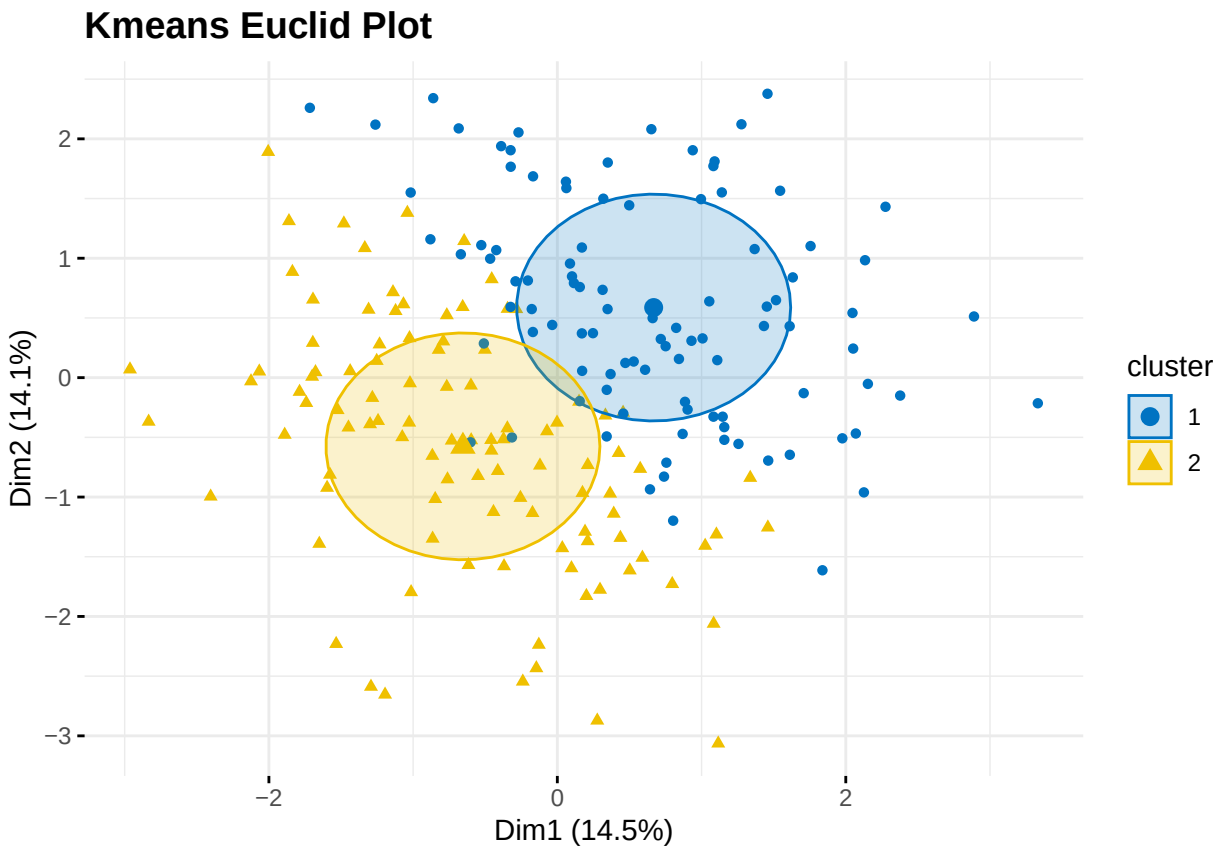
```
## [1] 1.5
```

```
sil2=as.data.frame(km.res4$silinfo)
neg_sil2 = which(sil2[, 'widths.sil_width'] <0)
percentage = round(length(neg_sil2)/200*100,2)
percentage
```

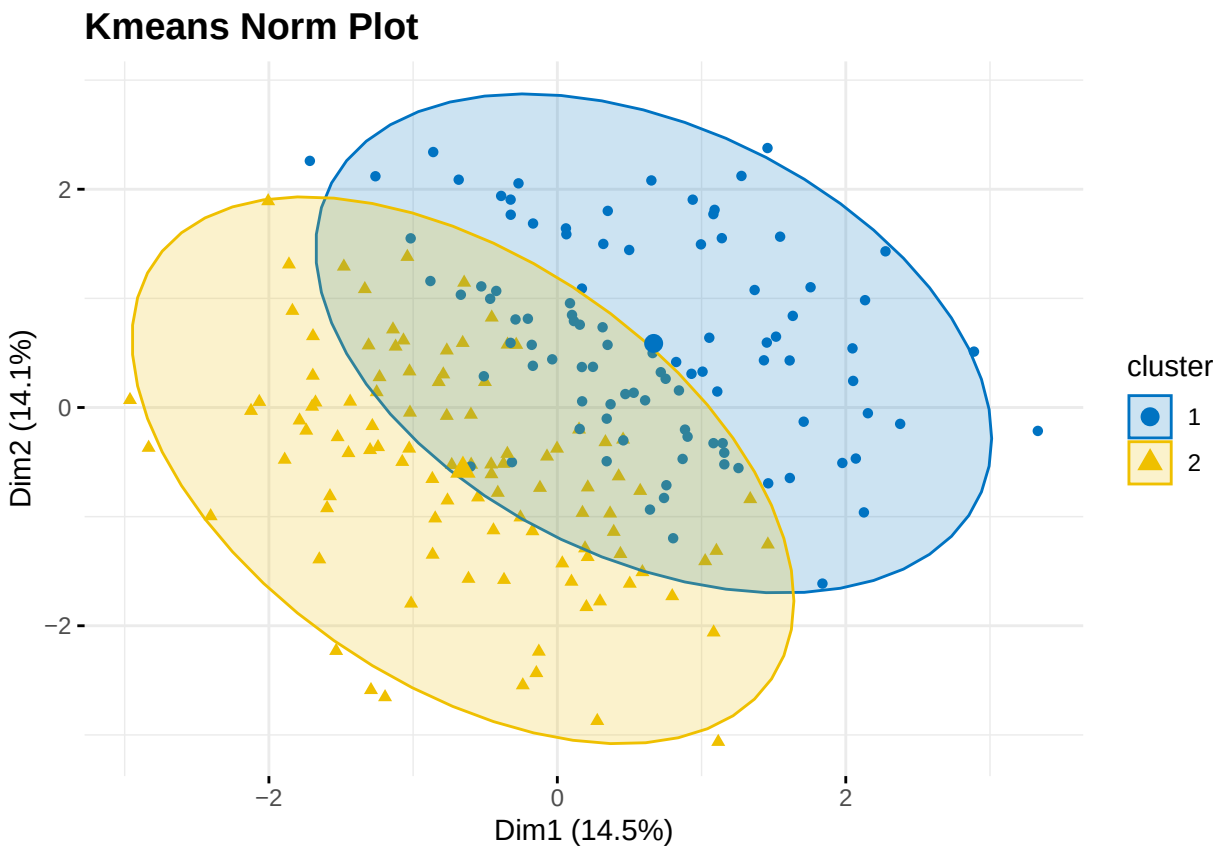
```
## [1] 2.5
```

8.2 Plots

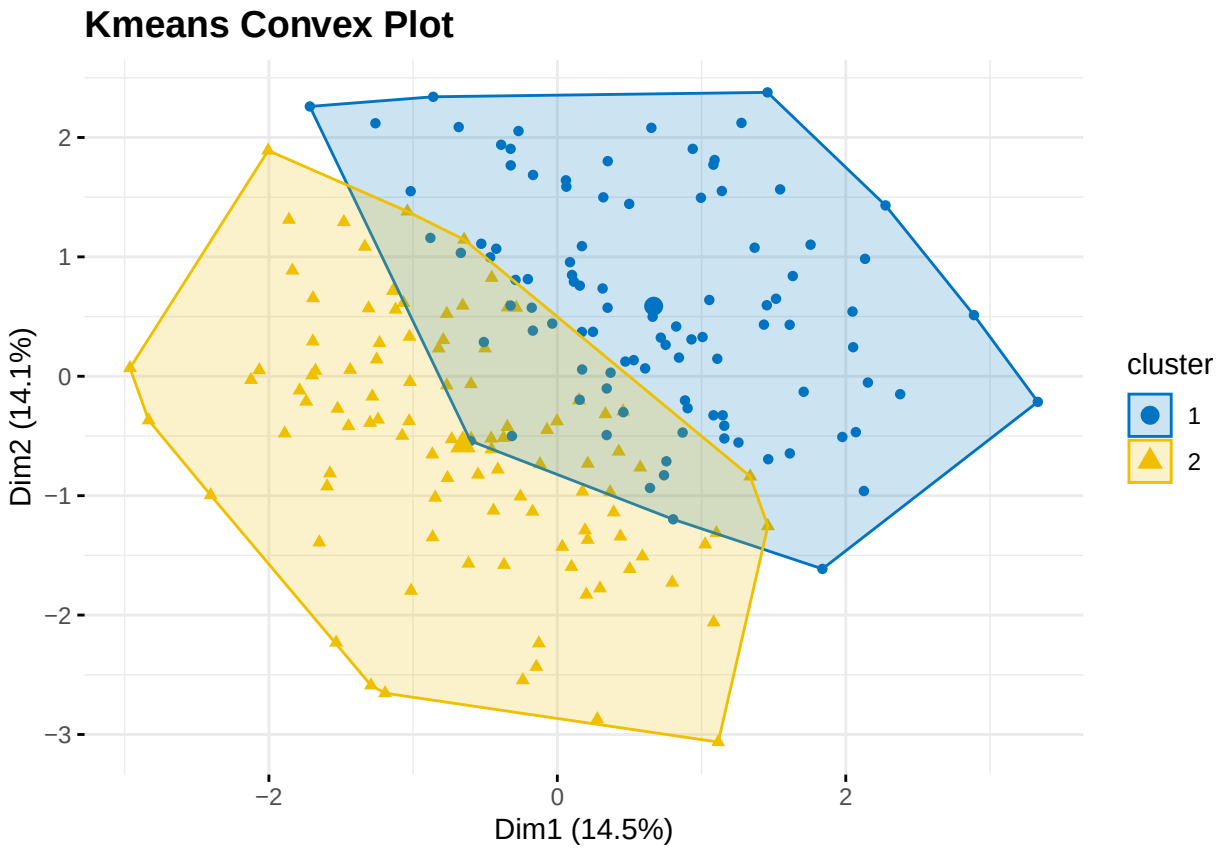
```
fviz_cluster(km.res2, geom = "point", ellipse.type = "euclid", palette = "jco", ggtheme = theme_minimal
```



```
fviz_cluster(km.res2, geom = "point", ellipse.type = "norm", palette = "jco", ggtheme = theme_minimal()
```




```
fviz_cluster(km.res2, geom = "point", ellipse.type = "convex", palette = "jco", ggtheme = theme_minimal
```



8.3 External Cluster Validation

```
compute_clustering_metrics = function(data, true_labels, cluster_labels) {
  library(fpc)
  distance_matrix =  dist(data)
  true_labels_numeric = as.numeric(true_labels)
  cluster_labels_cluster = cluster_labels$cluster

  clust_stats = cluster.stats(distance_matrix, true_labels_numeric, cluster_labels_cluster)

  table_labels = table(true_labels, cluster_labels_cluster)

  return(list(table_labels,
              clust_stats$corrected.rand,
              clust_stats$vi))
}
```

8.3.1 Rand and VI

```
compute_clustering_metrics(sample.scale, sample$Dry.Eye.Disease, km.res2)
```

```
## Warning: package 'fpc' was built under R version 4.4.1
```

```
## [[1]]
##            cluster_labels_cluster
## true_labels  1  2
##           N 34 36
```

```
##           Y 65 65
##
## [[2]]
## [1] -0.004502033
##
## [[3]]
## [1] 1.340358
```

```
compute_clustering_metrics(sample.scale, sample$Sleep.quality, km.res2)
```

```
## [[1]]
##            cluster_labels_cluster
## true_labels  1  2
##           1 28 26
##           2 17 20
##           3 24 24
##           4 16 18
##           5 14 13
##
## [[2]]
## [1] -0.007118655
##
## [[3]]
## [1] 2.2706
```

```
compute_clustering_metrics(sample.scale, sample$Caffeine.consumption, km.res2)
```

```
## [[1]]
##            cluster_labels_cluster
## true_labels  1  2
##           N 54 50
##           Y 45 51
##
## [[2]]
## [1] -0.002529433
##
## [[3]]
## [1] 1.382899
```

```
compute_clustering_metrics(sample.scale, sample$Alcohol.consumption, km.res2)
```

```
## [[1]]
##            cluster_labels_cluster
## true_labels  1  2
##           N 43 47
##           Y 56 54
##
## [[2]]
## [1] -0.004095755
##
## [[3]]
## [1] 1.380265
```

```
compute_clustering_metrics(sample.scale, sample$Smoking, km.res2)
```

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

```
## true_labels  1  2
##           N 54 58
##           Y 45 43
##
## [[2]]
## [1] -0.004073889
##
## [[3]]
## [1] 1.378185
```

```
compute_clustering_metrics(sample.scale, sample$Smart.device.before.bed, km.res2)
```

```
## [[1]]
##            cluster_labels_cluster
## true_labels  1  2
##           N 44 49
##           Y 55 52
##
## [[2]]
## [1] -0.003417582
##
## [[3]]
## [1] 1.382127
```

```
compute_clustering_metrics(sample.scale, sample$Blue.light.filter, km.res2)
```

```
## [[1]]
##            cluster_labels_cluster
## true_labels  1  2
##           N 48 52
##           Y 51 49
##
## [[2]]
## [1] -0.004145452
##
## [[3]]
## [1] 1.385344
```

```
compute_clustering_metrics(sample.scale, sample$Sleep.quality, km.res2)
```

```
## [[1]]
##            cluster_labels_cluster
## true_labels  1  2
##           1 28 26
##           2 17 20
##           3 24 24
##           4 16 18
##           5 14 13
##
## [[2]]
## [1] -0.007118655
##
## [[3]]
## [1] 2.2706
```

```
compute_clustering_metrics(sample.scale, sample$Gender, km.res2)
```

```
## [[1]]
##            cluster_labels_cluster
## true_labels  1  2
##           F 48 49
##           M 51 52
##
## [[2]]
## [1] -0.00504552
##
## [[3]]
## [1] 1.385794
```

9 Advanced Clustering

9.1 Model Clustering

```
library(mclust)
```

```
## Warning: package 'mclust' was built under R version 4.4.1
```

```
## Package 'mclust' version 6.1.1
## Type 'citation("mclust")' for citing this R package in publications.
```

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

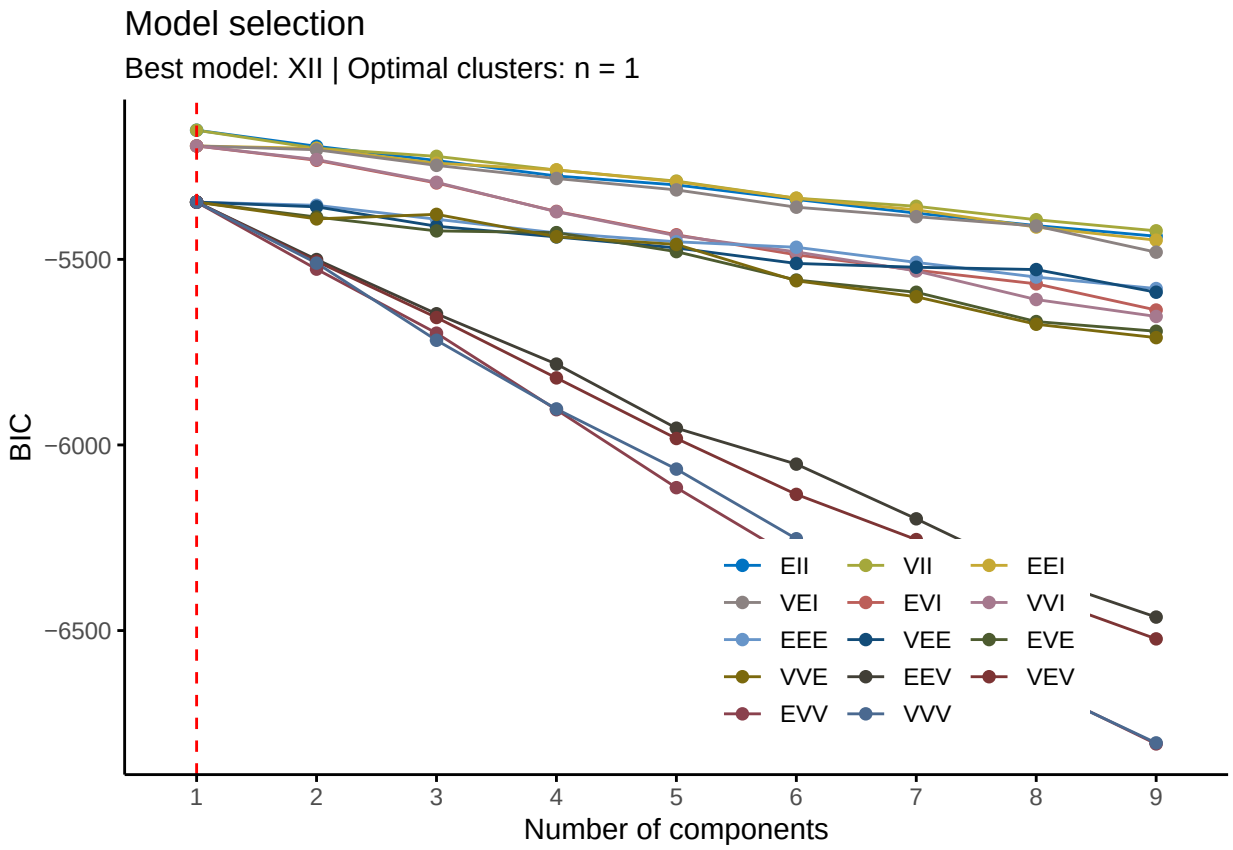
```
## The following object is masked from 'package:purrr':
##
##     map
```

```
mc = Mclust(sample.scale)            # Model-based-clustering
summary(mc)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust XII (spherical multivariate normal) model with 1 component:
##
##  log-likelihood   n df       BIC       ICL
##       -2549.578 200 10 -5152.139 -5152.139
##
## Clustering table:
##   1
## 200
```

```
fviz_mclust(mc, "BIC", palette = "jco")
```

```
## Warning: `gather_()` was deprecated in tidyr 1.2.0.
## i Please use `gather()` instead.
## 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.
```



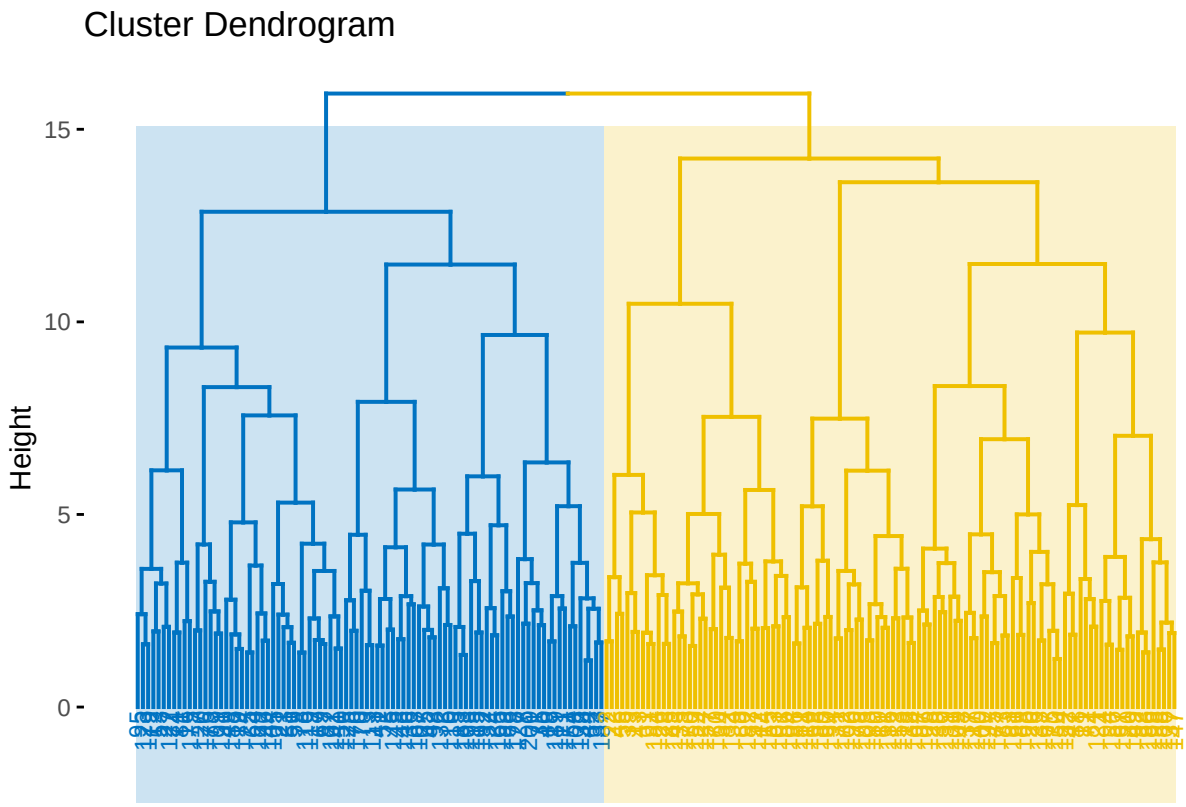
9.2 Hibrid HKmeans

```
hk.res = hkmeans(sample.scale,2)

hk.res
```

```
## Hierarchical K-means clustering with 2 clusters of sizes 103, 97
##
## Cluster means:
##          Age Sleep.duration   Systolic  Diastolic Heart.rate Daily.steps
## 1  0.2104530     -0.3913723 -0.3090731 -0.5913692 -0.1490198  0.09007813
## 2 -0.2234708      0.4155809  0.3281910  0.6279488  0.1582375 -0.09564997
##   Physical.activity        BMI Average.screen.time
## 1         0.3079067  0.2379895          -0.1218283
## 2        -0.3269525 -0.2527105           0.1293641
##
## Clustering vector:
##   [1] 2 1 2 1 2 2 2 2 2 1 1 2 1 1 2 2 2 1 2 1 2 1 2 2 1 1 2 1 1 1 2 2 2 2 1 1 1
##  [38] 1 1 2 2 2 1 2 2 1 1 2 2 1 2 2 2 2 1 1 1 1 2 1 1 2 1 2 1 2 2 1 2 2 2 2 2 2
##  [75] 1 2 2 2 1 2 2 1 1 1 1 1 2 1 2 1 2 1 1 2 2 1 1 1 2 1 1 2 1 1 2 1 2 2 1 2 2
## [112] 1 2 1 2 2 1 1 2 2 1 2 2 1 2 1 2 1 1 1 1 2 1 1 1 1 1 1 1 1 2 1 2 1 2 2 1 1
## [149] 1 2 2 1 1 1 1 2 1 1 1 1 1 2 2 1 2 1 1 1 1 2 2 2 1 2 2 2 2 2 2 1 1 1 1 1 1
## [186] 1 2 2 1 2 2 2 2 1 2 1 1 1 2 2
##
## Within cluster sum of squares by cluster:
## [1] 829.5194 783.2343
##  (between_SS / total_SS =  10.0 %)
```

```
##
## Available components:
##
##  [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
##  [6] "betweenss"    "size"         "iter"         "ifault"       "data"
## [11] "hclust"
```

```
fviz_dend(hk.res, cex = 0.6, palette = "jco",
          rect = TRUE, rect_border = "jco", rect_fill = TRUE)
```

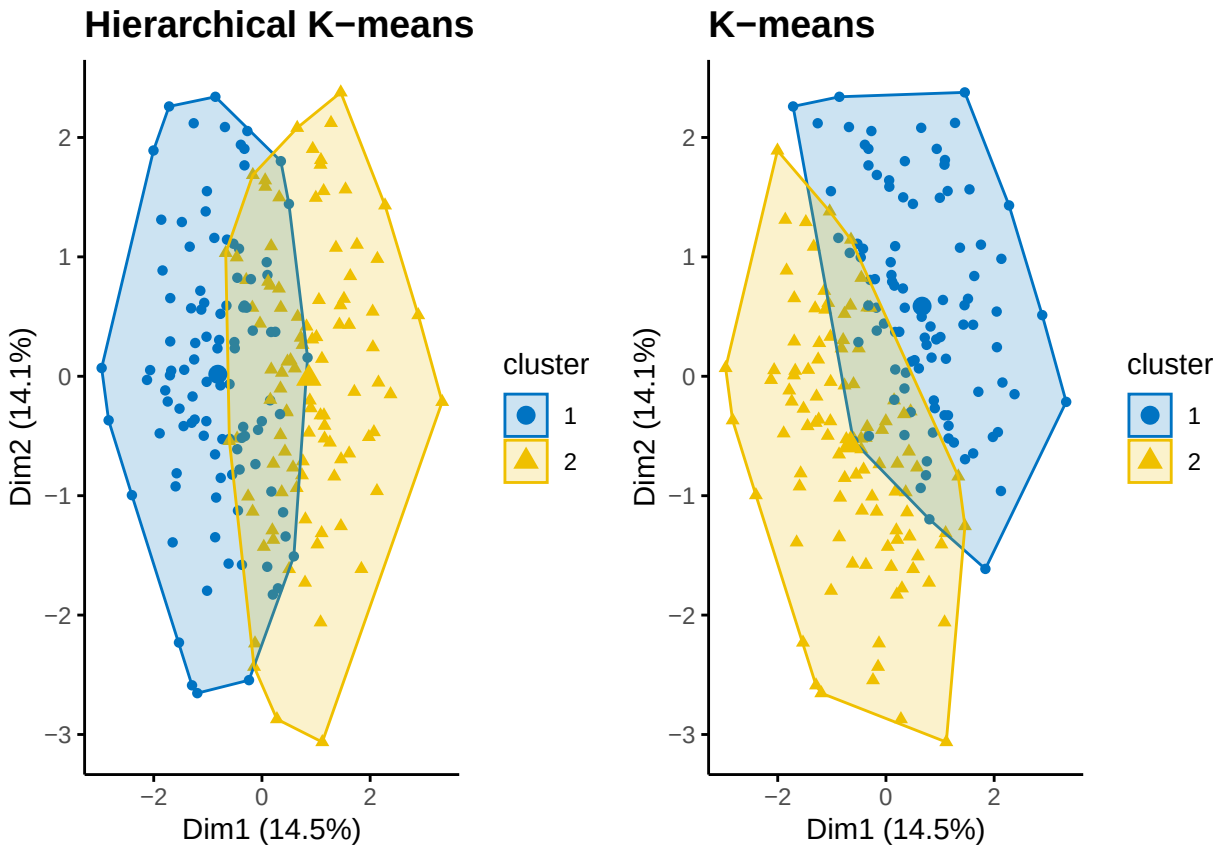



```
library(gridExtra)
```

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

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

```
p1=fviz_cluster(hk.res, palette = "jco", repel = TRUE, ggtheme = theme_classic(), geom = 'point', ellips

p2=fviz_cluster(km.res2, palette = "jco", repel = TRUE, ggtheme = theme_classic(), geom = c("point"), el

grid.arrange(p1, p2, ncol = 2, widths = c(2, 2))
```



Fuzzy Clustering

```
fanny.res <- fanny(sample.scale, 2)
```

```
## Warning in fanny(sample.scale, 2): the memberships are all very close to 1/k.
## Maybe decrease 'memb.exp' ?
```

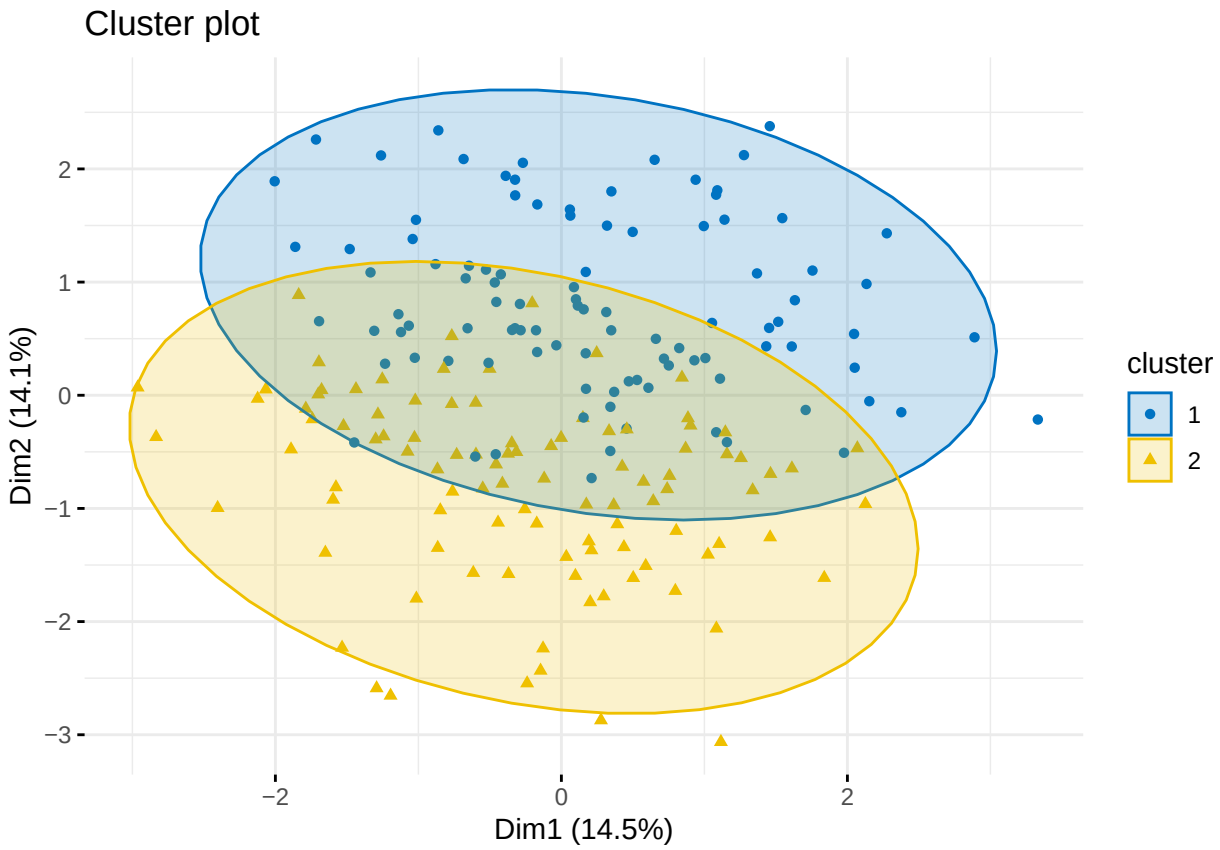
```
head(fanny.res$membership, 10)
```

```
##       [,1] [,2]
##  [1,]  0.5  0.5
##  [2,]  0.5  0.5
##  [3,]  0.5  0.5
##  [4,]  0.5  0.5
##  [5,]  0.5  0.5
##  [6,]  0.5  0.5
##  [7,]  0.5  0.5
##  [8,]  0.5  0.5
##  [9,]  0.5  0.5
## [10,]  0.5  0.5
```

```
fanny.res$coeff
```

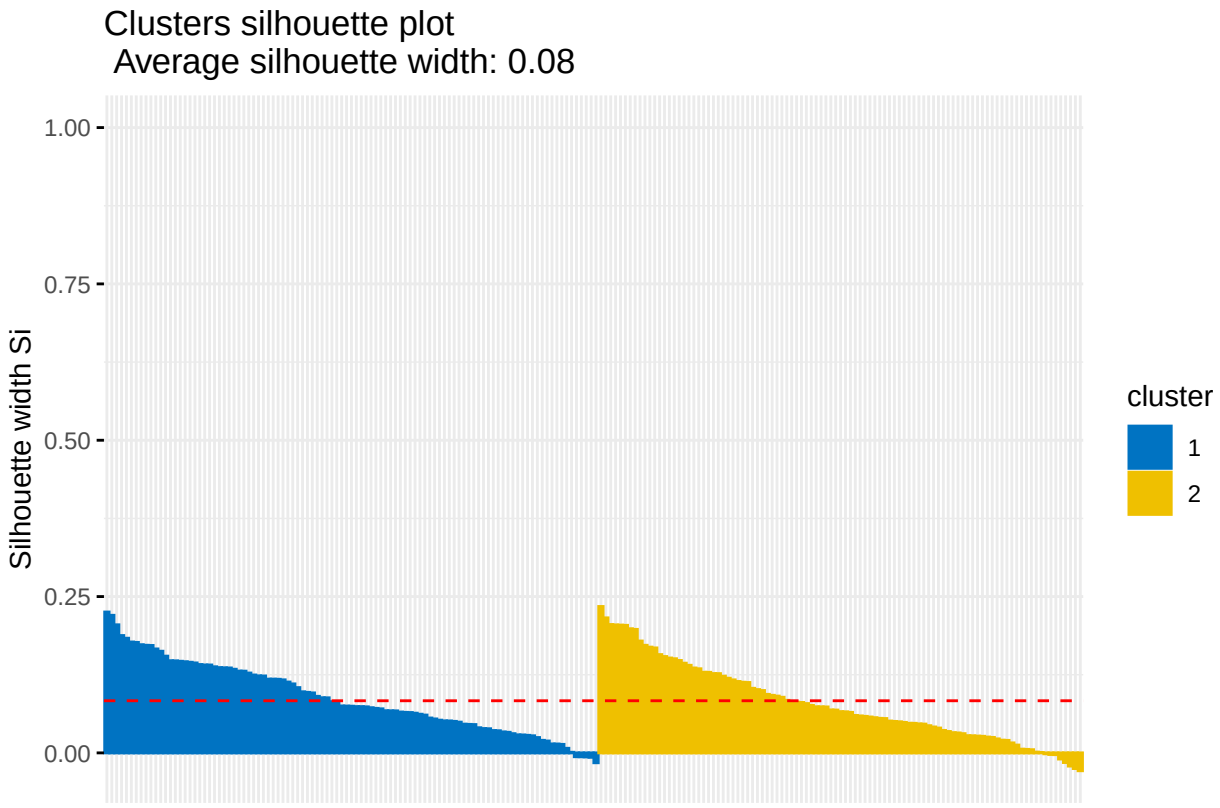
```
##   dunn_coeff   normalized
## 5.000000e-01 8.437695e-15
```

```
fviz_cluster(fanny.res, ellipse.type = "norm", repel = TRUE,
             palette = "jco", ggtheme = theme_minimal(), geom = "point",
             legend = "right")
```



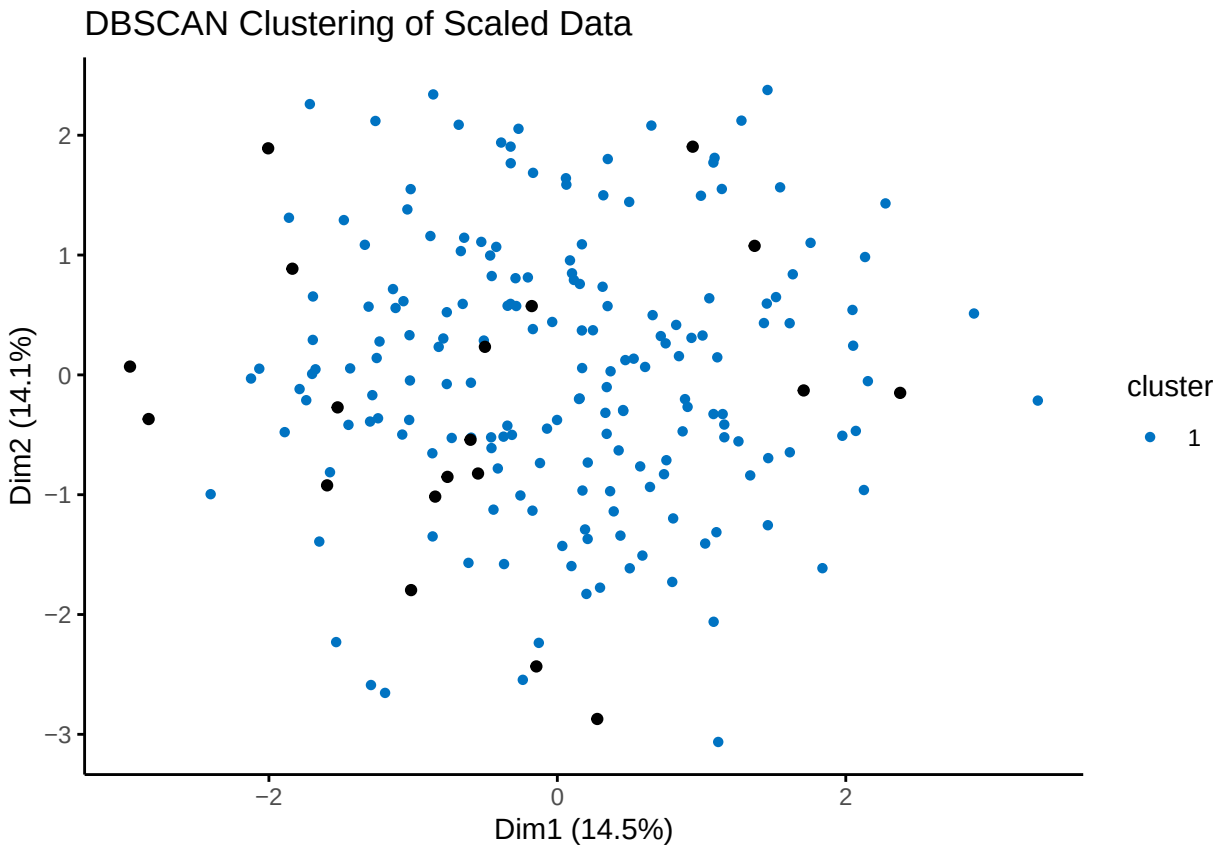

```
fviz_silhouette(fanny.res, palette = "jco",
                ggtheme = theme_minimal())
```

```
##   cluster size ave.sil.width
## 1       1  101          0.09
## 2       2   99          0.08
```



9.3 DBSCAN

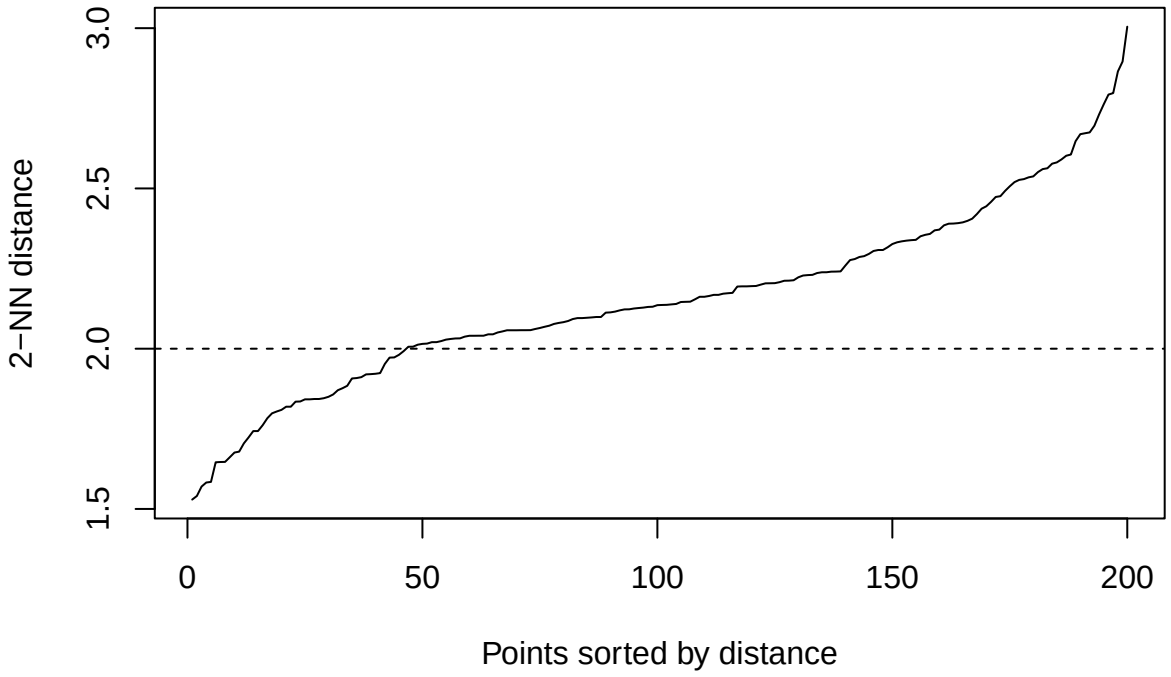
```
set.seed(123)
db <- fpc::dbscan(sample.scale, eps = 2.5, MinPts = 5)
fviz_cluster(db, data = sample.scale, stand = FALSE,
             ellipse = FALSE, show.clust.cent = FALSE,
             geom = "point",palette = "jco", ggtheme = theme_classic())+
  ggtitle("DBSCAN Clustering of Scaled Data")
```




```
print(db)
```

```
## dbscan Pts=200 MinPts=5 eps=2.5
##         0   1
## border 19  55
## seed    0 126
## total  19 181
```

```
dbscan::kNNdistplot(sample.scale, k =  2)
abline(h = 2, lty = 2)
```



10 Statistical Analysis

For this statistical analysis, ideas from "A Study of Clustered Data and Approaches to Its Analysis" by Sally Galbraith, James A. Daniel, and Bryce Vissel, and "Cluster analysis and its application to healthcare claims data: a study of end-stage renal disease patients who initiated hemodialysis" by Minlei Liao, Yunfeng Li, Farid Kianifard, Engels Obi and Stephen Arcona. In the first the Wilcoxon rank test was performed to evaluate difference in clusters, and in the second, boxplots are performed to see difference in distributions.

```
#Merging
sample.cluster = cbind(sample, cluster=km.res2$cluster)
sample.cluster_df = as.data.frame(sample.cluster)
```

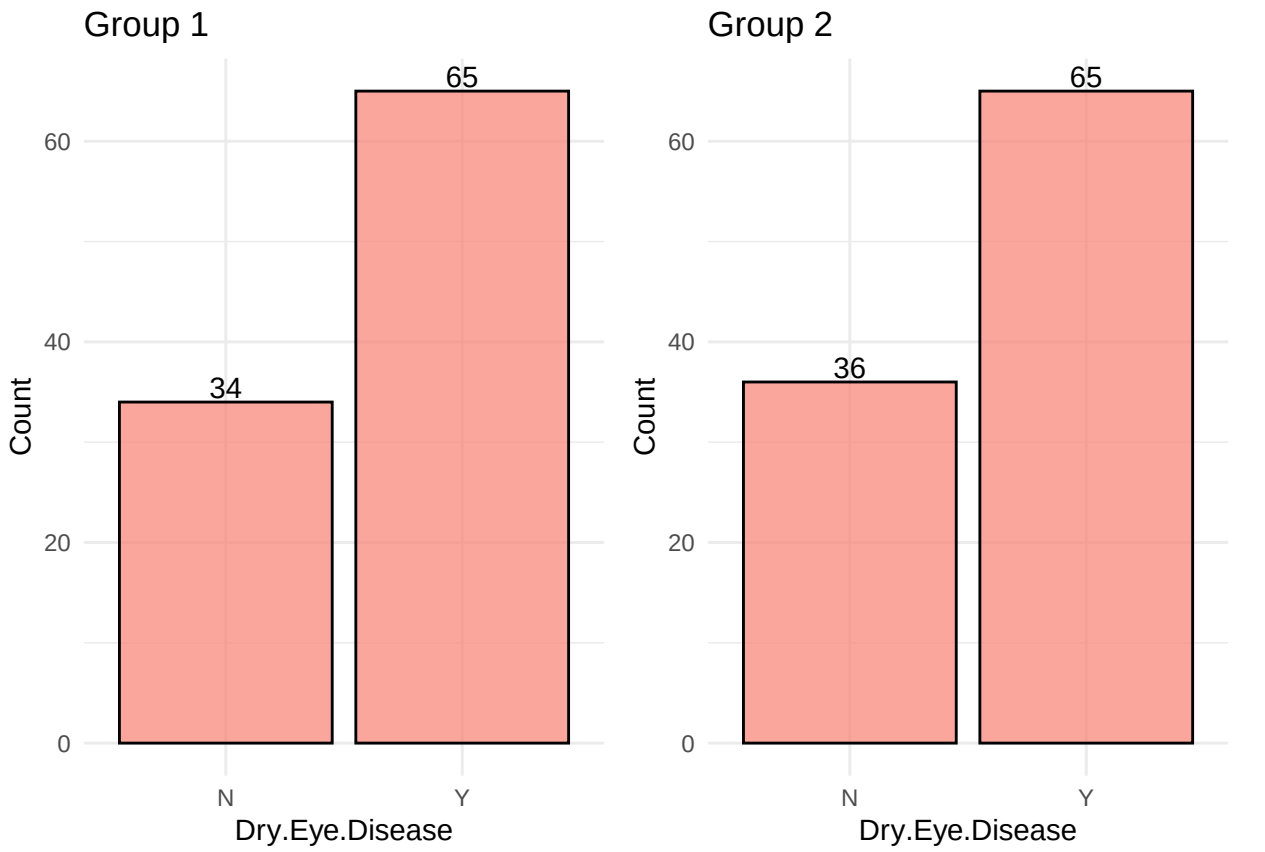
```
#groups
group1 = sample.cluster_df %>% filter(cluster==1)
group2 = sample.cluster_df %>% filter(cluster==2)
```

10.1 Categorical

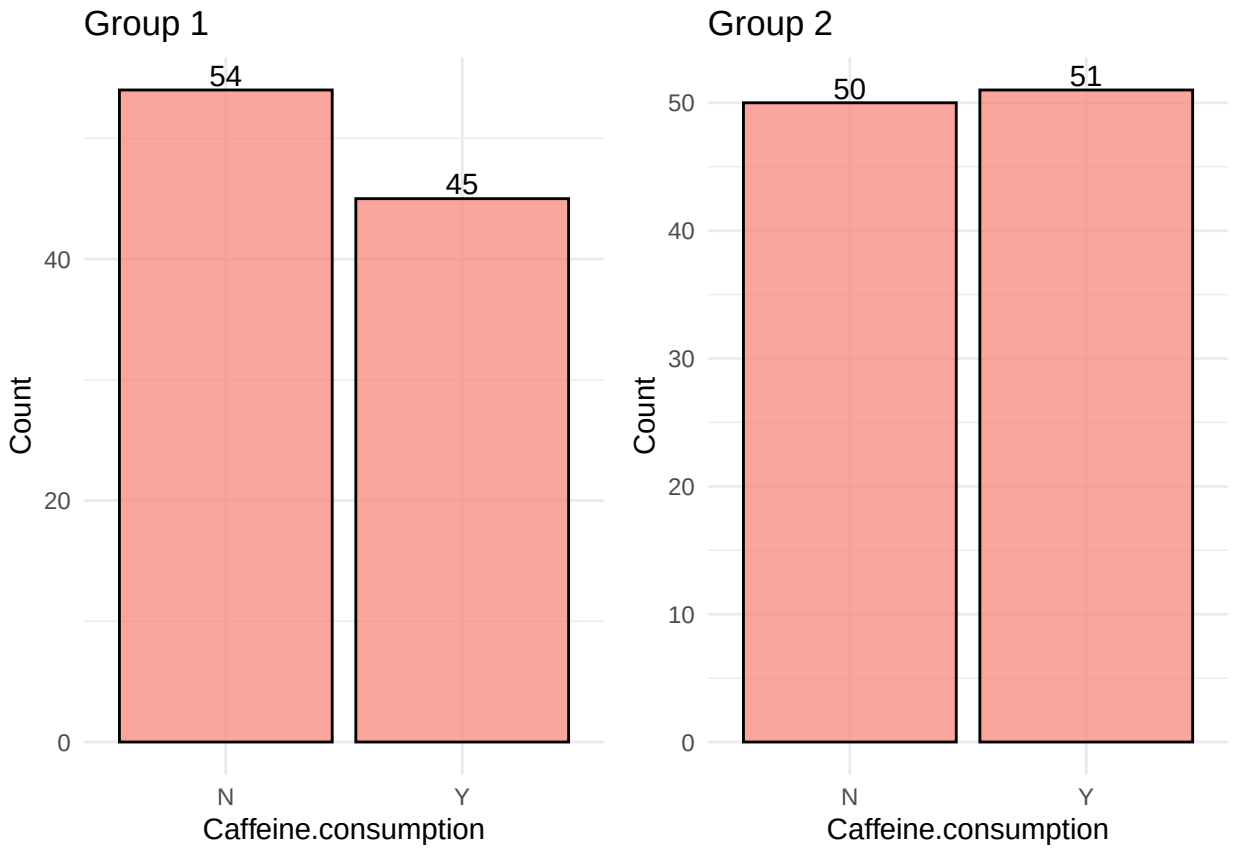
```
plot_groups <- function(column) {
  # Convert column to a string if it's passed as a symbol
  column <- as.character(substitute(column))

  # Create the first plot
  plot1 <- ggplot(group1, aes_string(x = column)) +
    geom_bar(fill = "salmon", color = "black", alpha = 0.7) +
    geom_text(stat = "count", aes(label = after_stat(count)), vjust = -0.2, size = 4) +
    labs(title = "Group 1", x = column, y = "Count") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
    theme_minimal()

  # Create the second plot
  plot2 <- ggplot(group2, aes_string(x = column)) +
```

```
    geom_bar(fill = "salmon", color = "black", alpha = 0.7) +
    geom_text(stat = "count", aes(label = after_stat(count)), vjust = -0.2, size = 4) +
    labs(title = "Group 2", x = column, y = "Count") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
    theme_minimal()

  # Arrange the plots in a grid
  grid.arrange(plot1, plot2, ncol = 2, widths = c(1, 1))
}
```

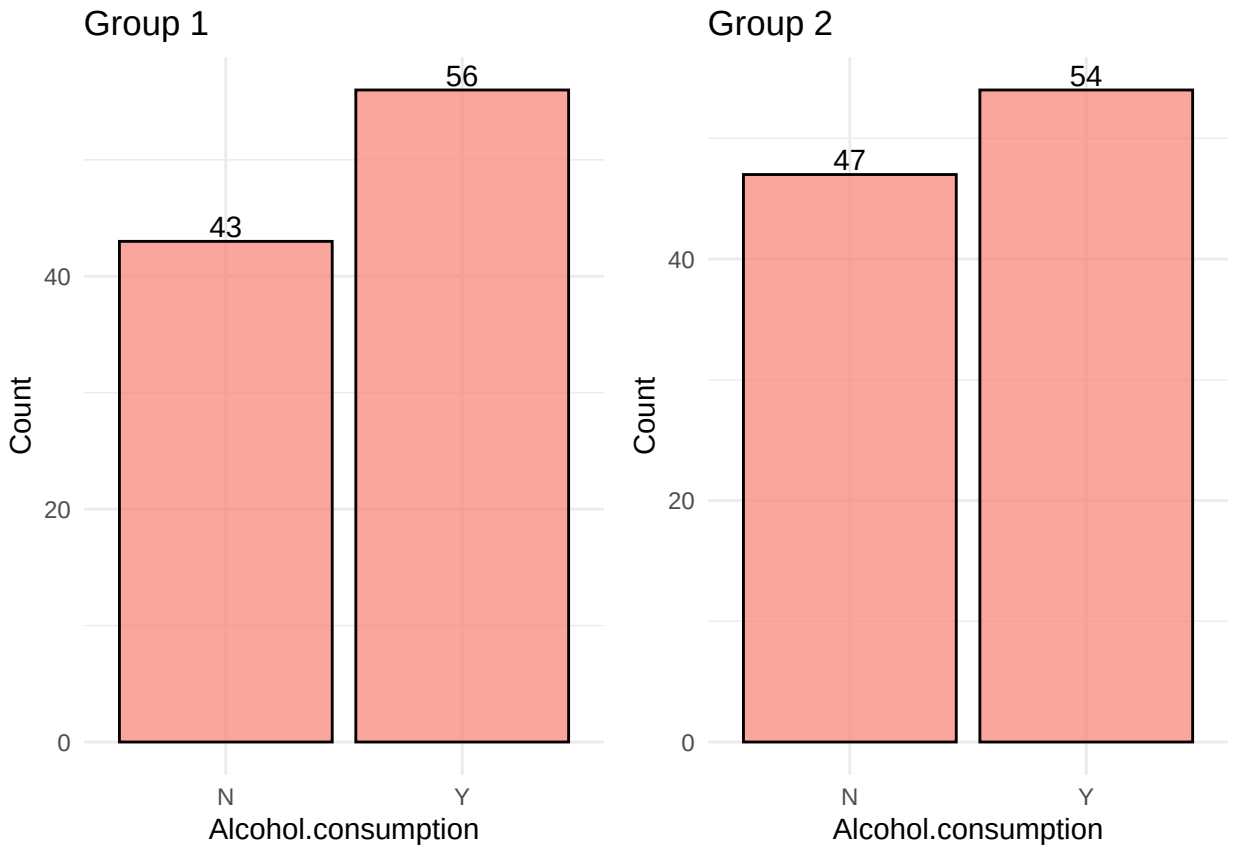
```
plot_groups(Dry.Eye.Disease)
```



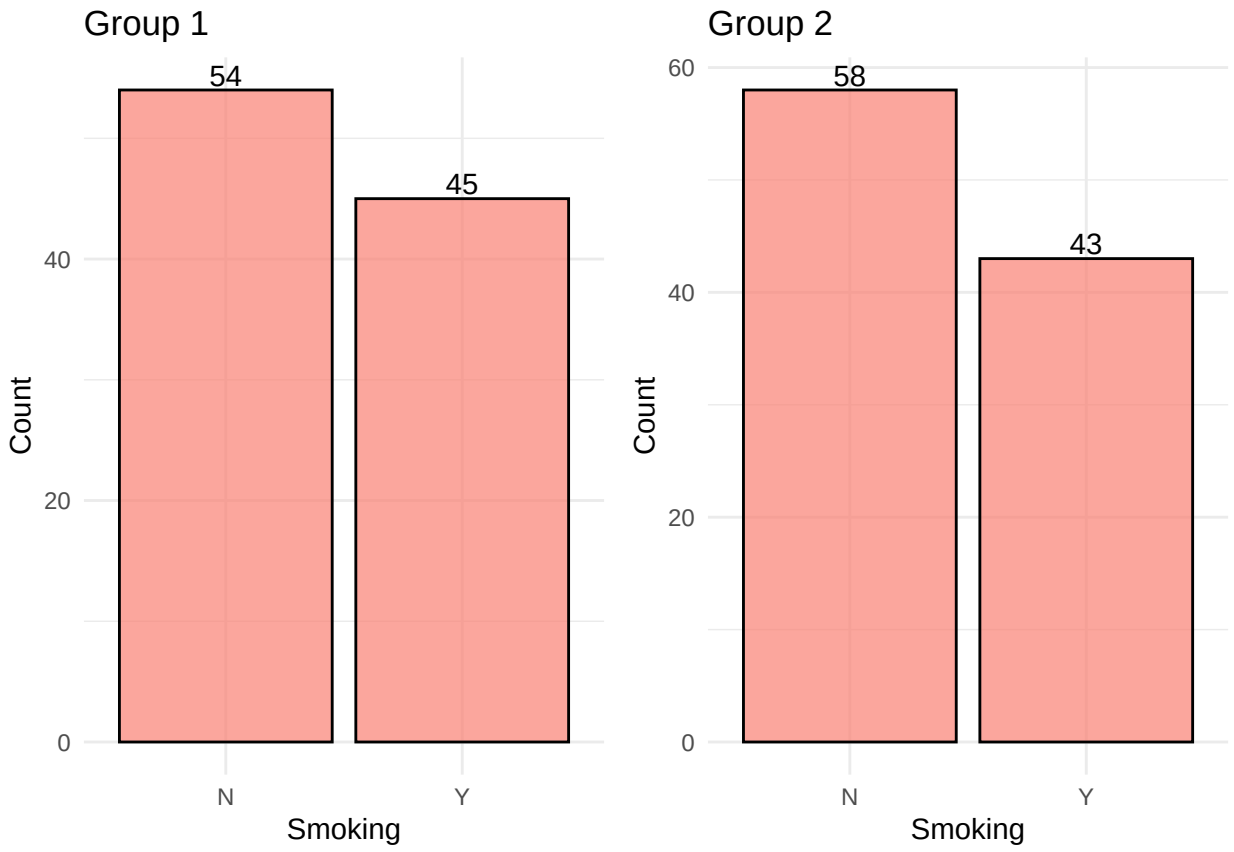

```
plot_groups(Caffeine.consumption)
```



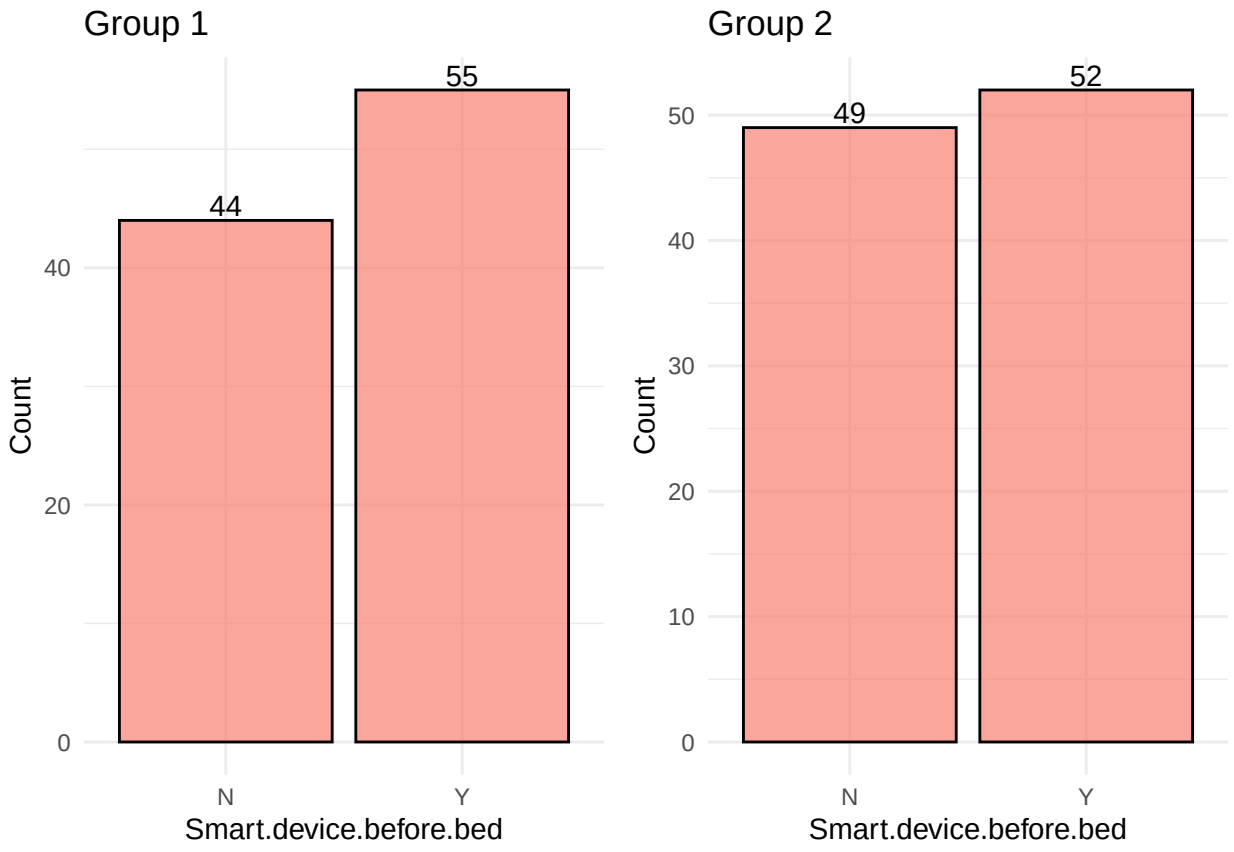

```
plot_groups(Alcohol.consumption)
```



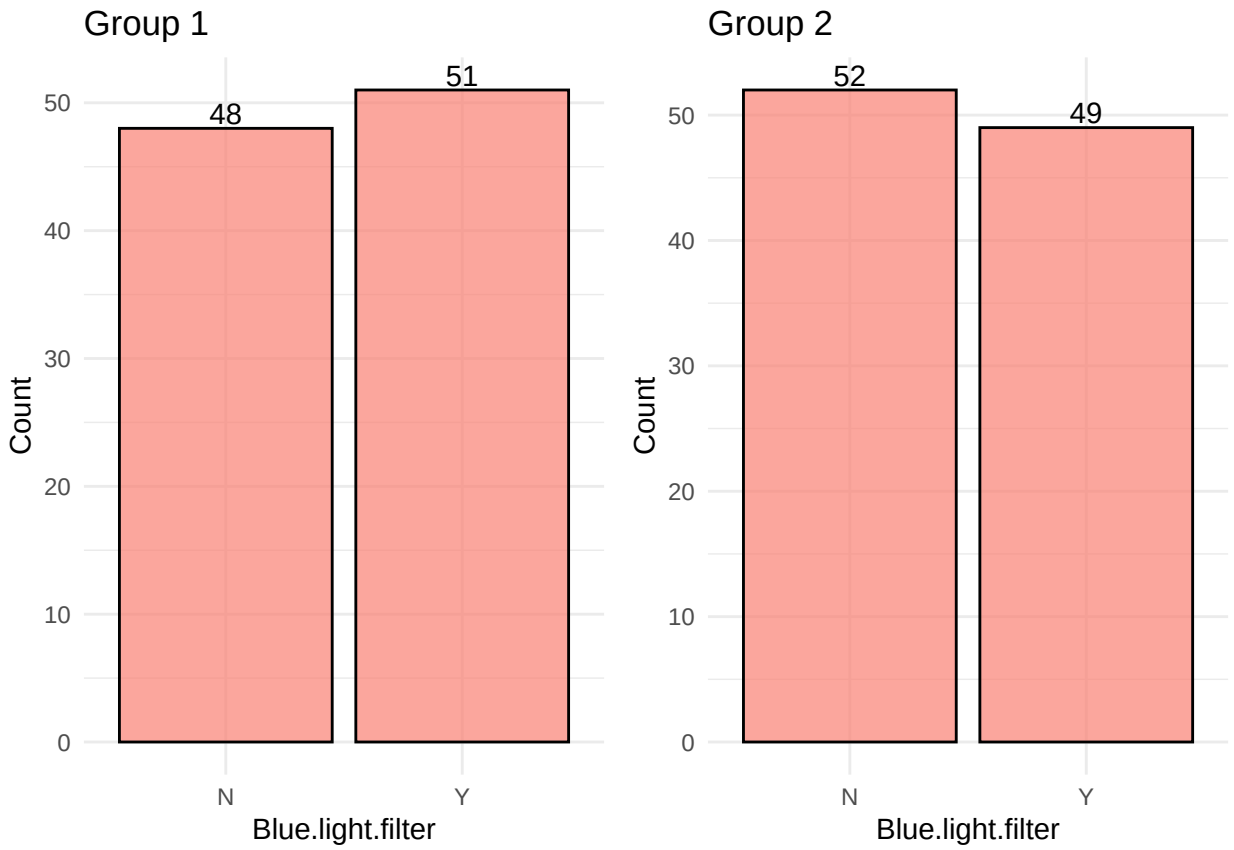

```
plot_groups(Smoking)
```



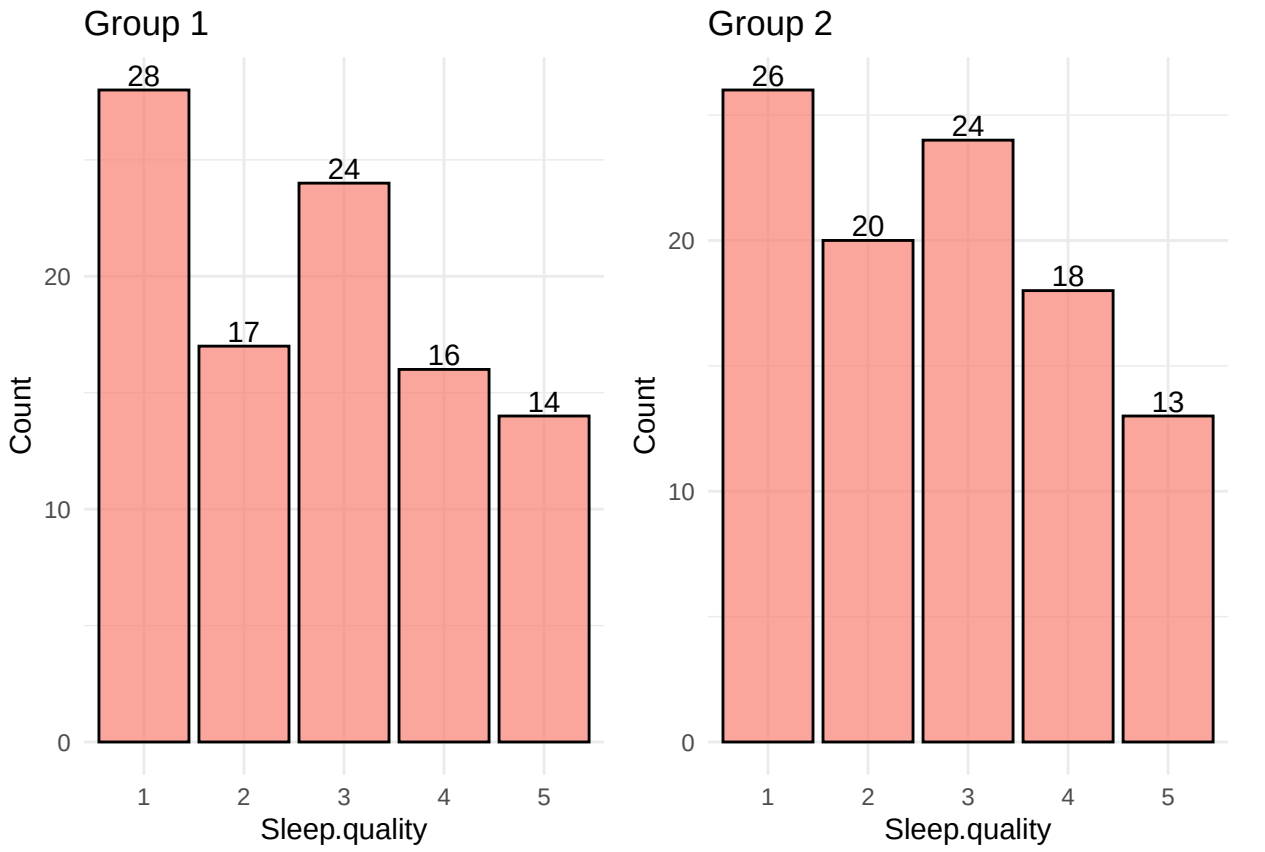

```
plot_groups(Smart.device.before.bed)
```



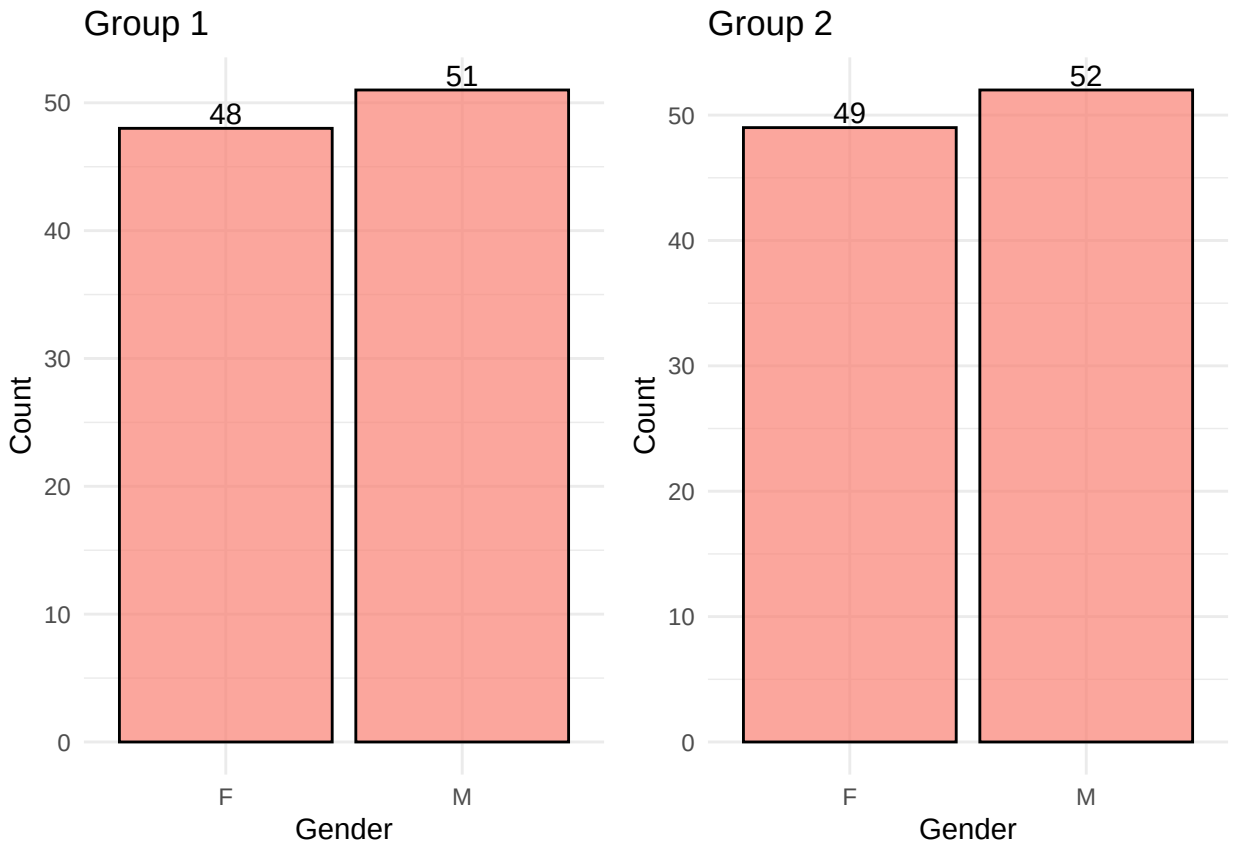

```
plot_groups(Blue.light.filter)
```




```
plot_groups(Sleep.quality)
```



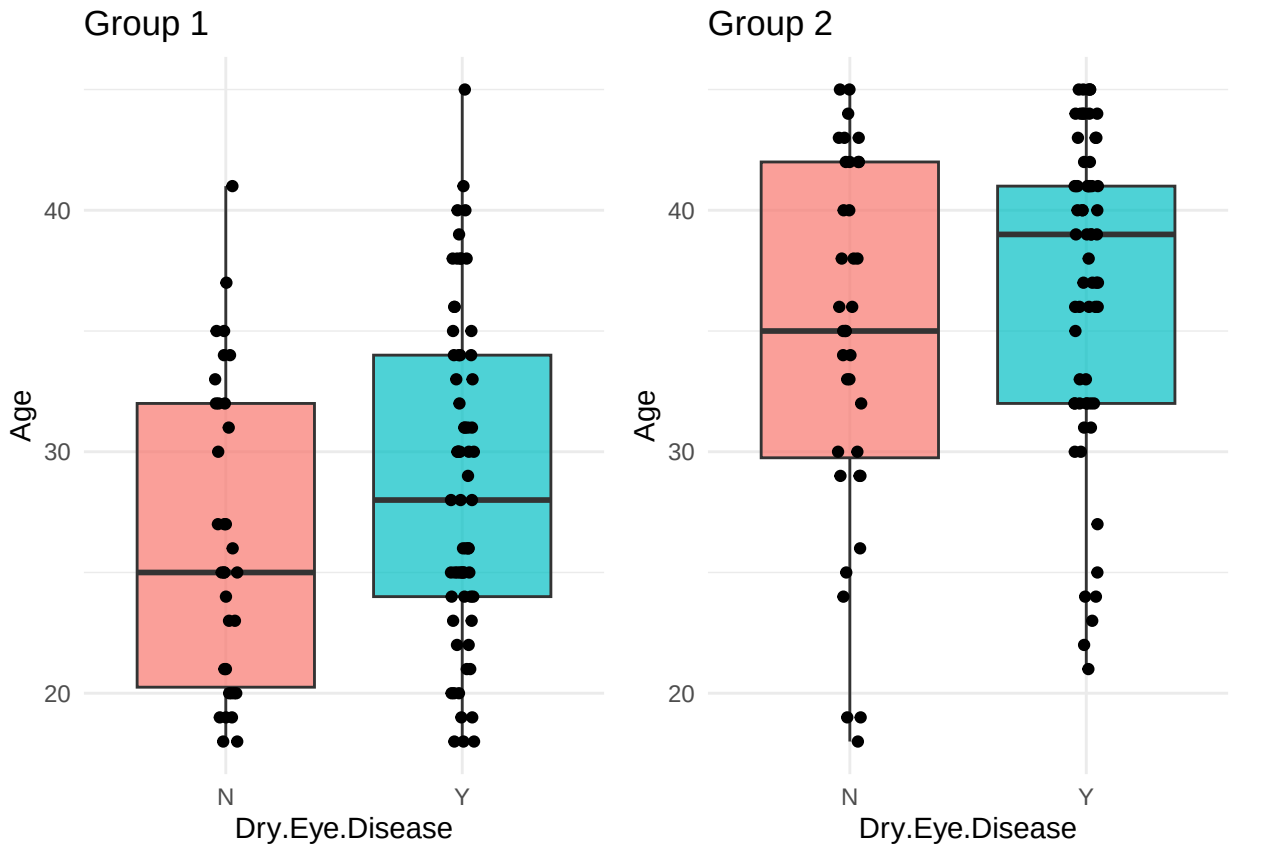

```
plot_groups(Gender)
```



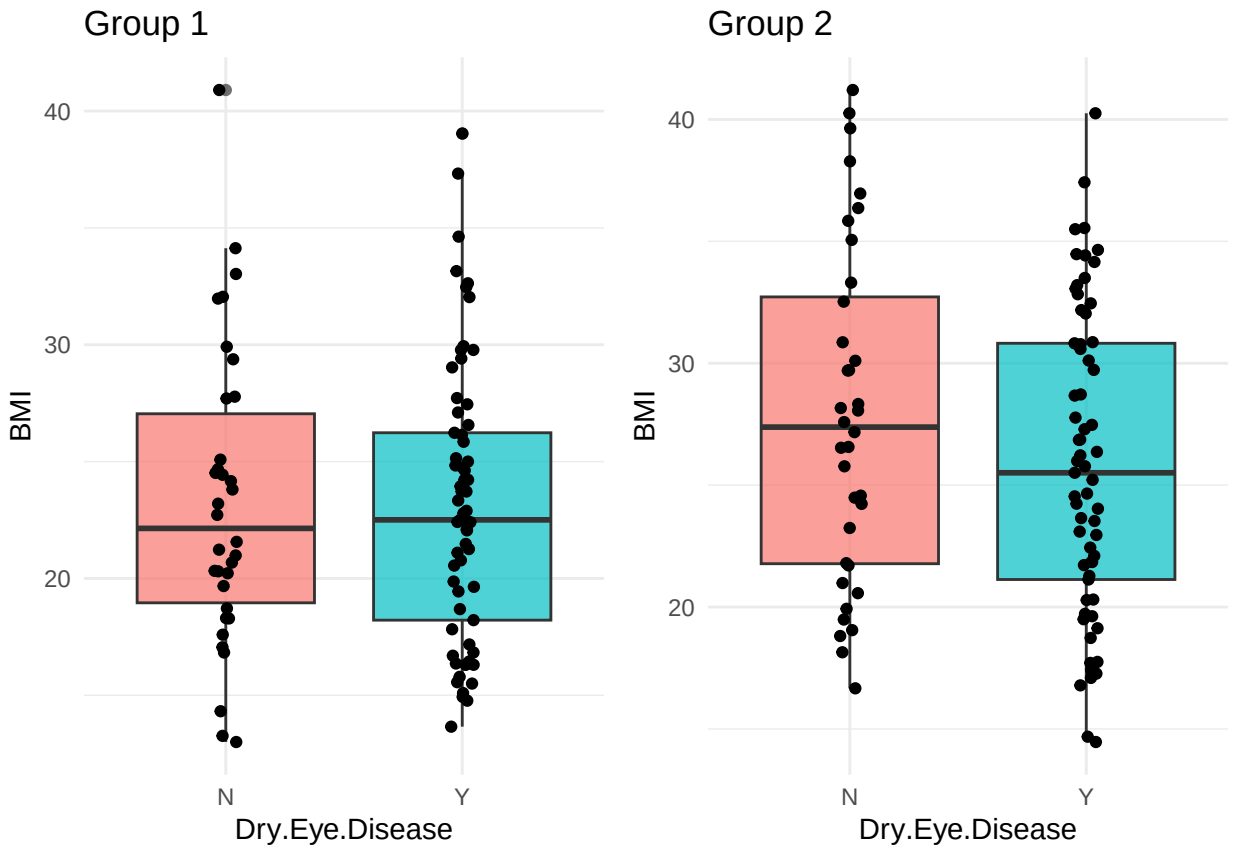
10.2 Numerical Data

```
plot_boxplots <- function(column_name) {

  column_name <- as.character(substitute(column_name))
  # Create plot for group 1
  plot1 <- ggplot(group1, aes_string(x = "Dry.Eye.Disease", y = column_name, fill = "Dry.Eye.Disease"))
    geom_boxplot(alpha = 0.7) +
    geom_jitter(height = 0, width = .05) +
    labs(title = "Group 1", y = column_name) +
    theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
    theme_minimal() +
    theme(legend.position = "none")

  # Create plot for group 2
  plot2 <- ggplot(group2, aes_string(x = "Dry.Eye.Disease", y = column_name, fill = "Dry.Eye.Disease"))
    geom_boxplot(alpha = 0.7) +
    geom_jitter(height = 0, width = .05) +
    labs(title = "Group 2", y = column_name) +
    theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
    theme_minimal() +
    theme(legend.position = "none")

  # Arrange and display both plots side by side
  grid.arrange(plot1, plot2, ncol = 2, widths = c(1, 1))
}
```

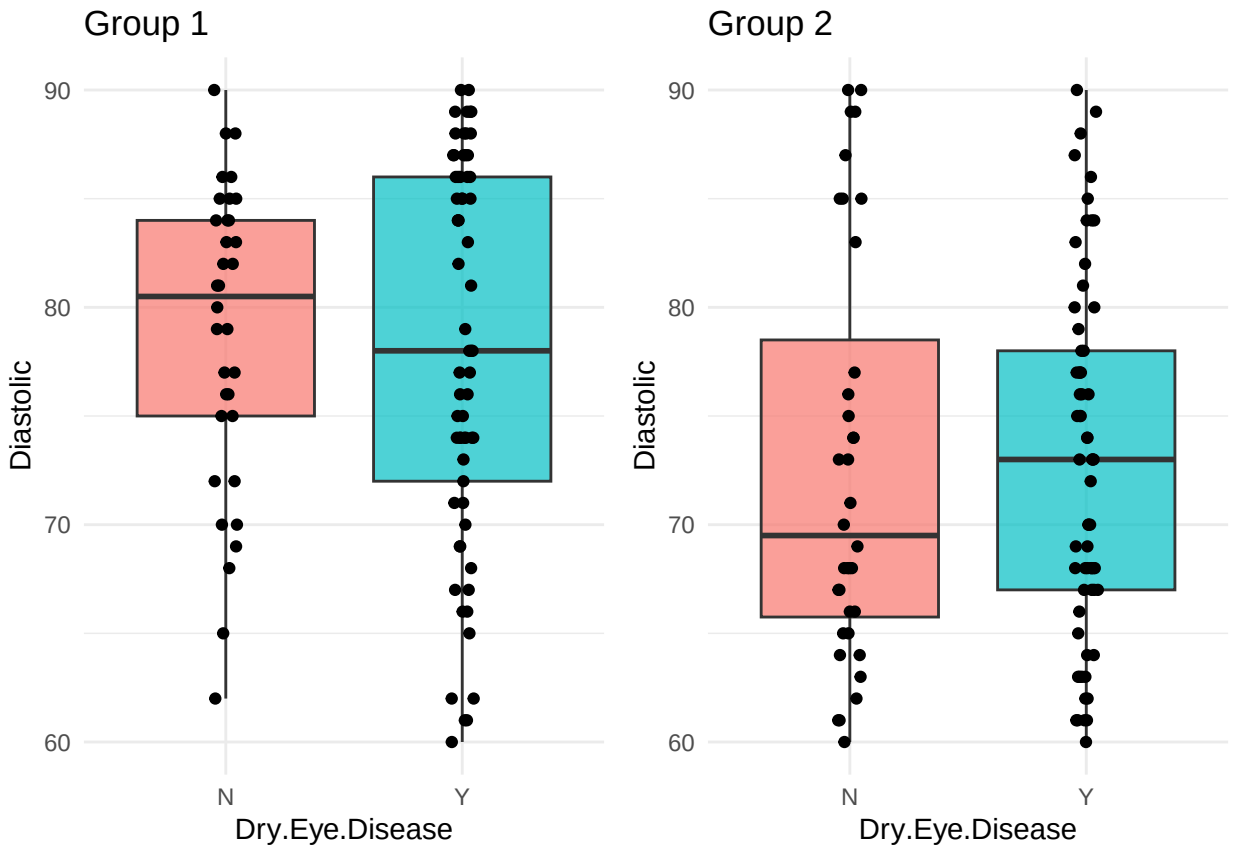
```
plot_boxplots(Age)
```



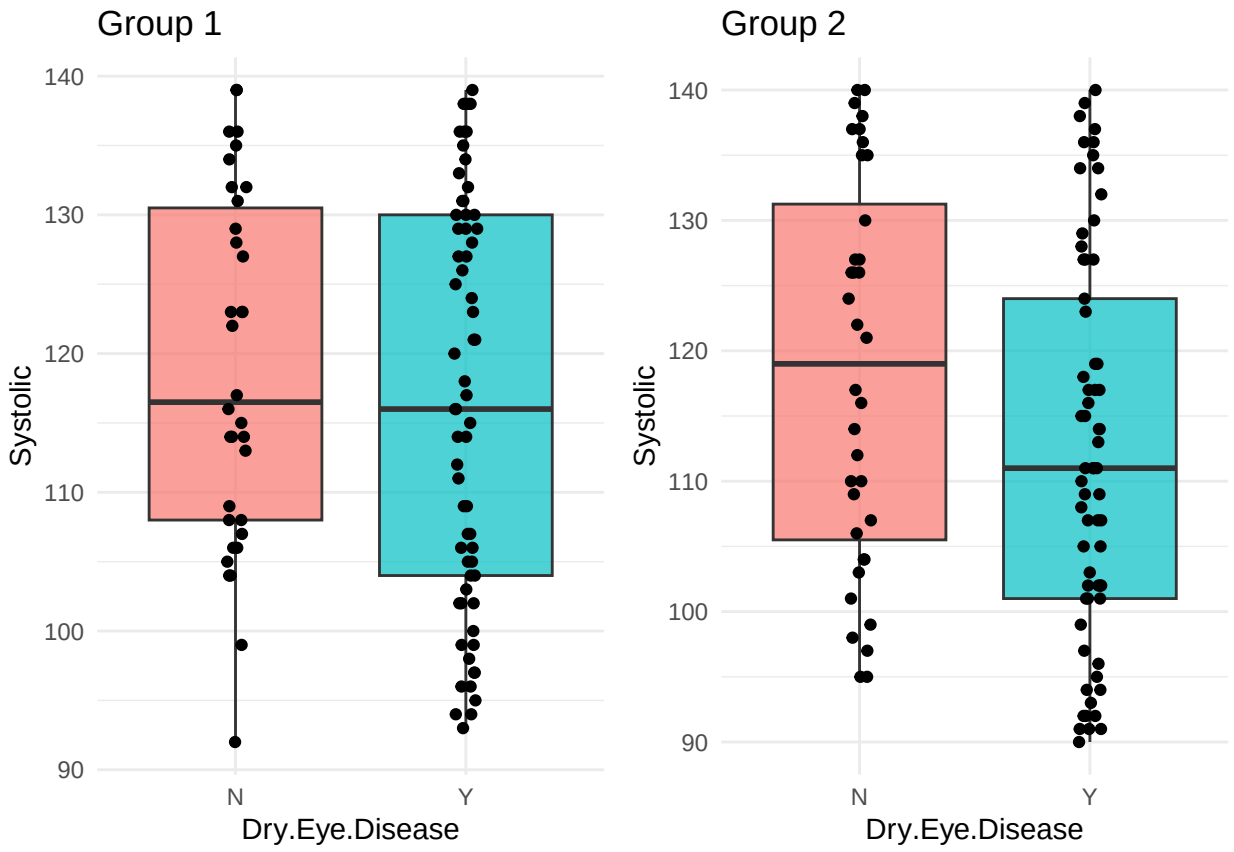

```
plot_boxplots(BMI)
```



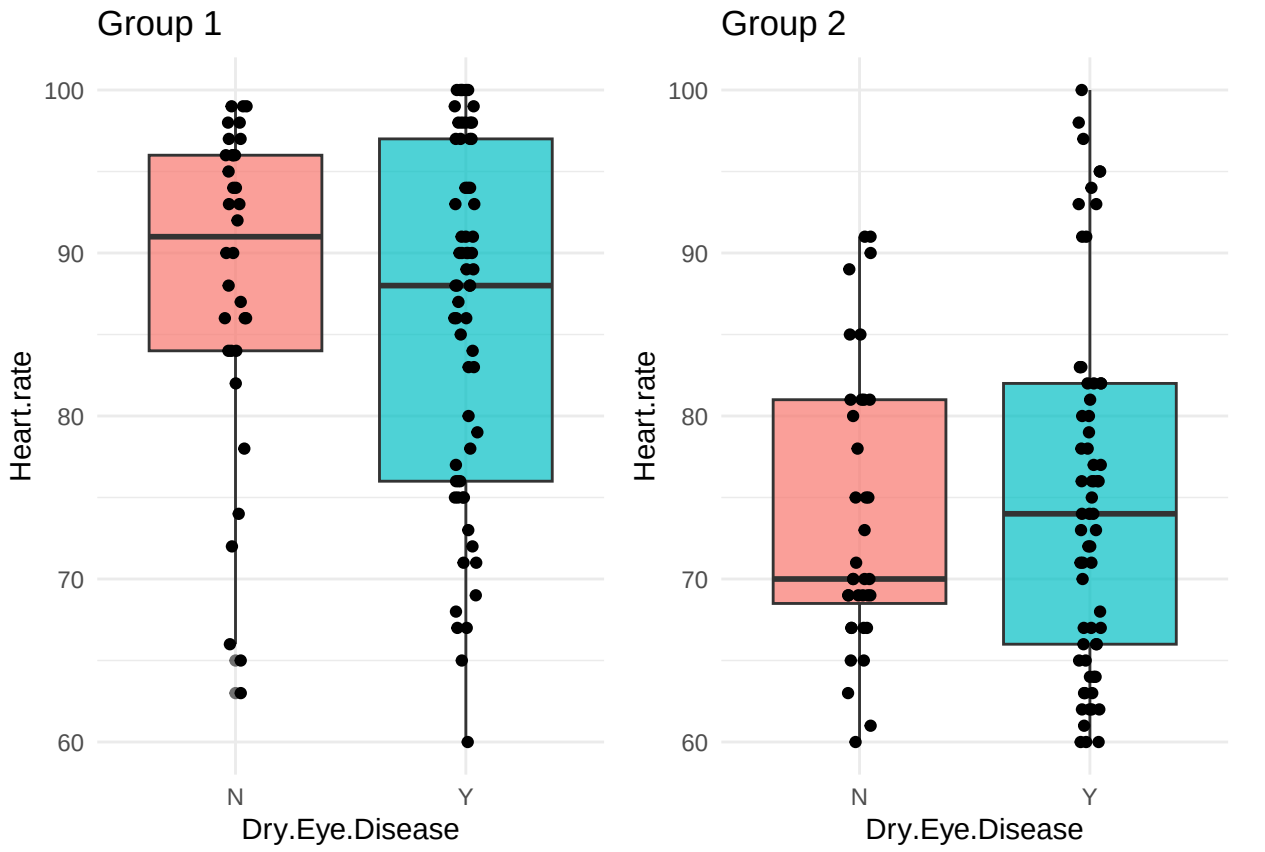

```
plot_boxplots(Diastolic)
```



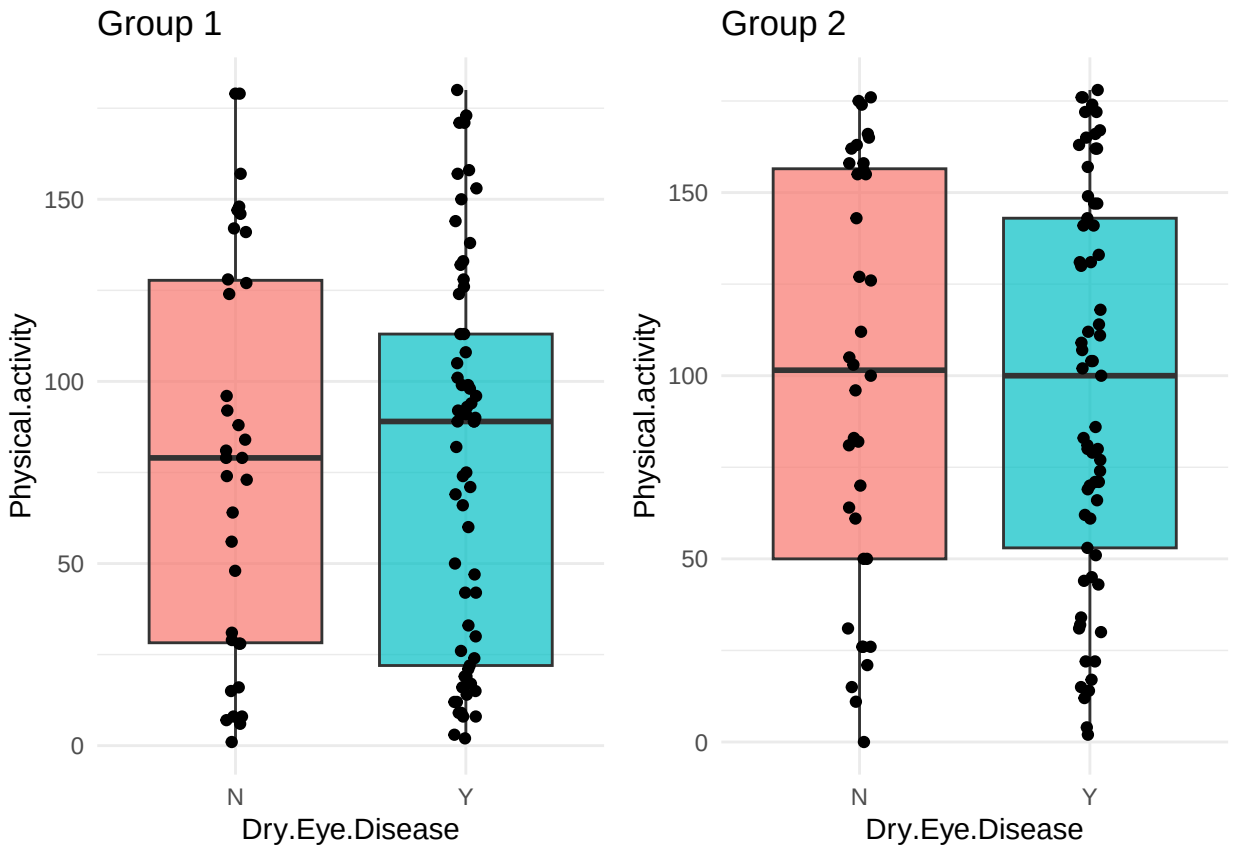

```
plot_boxplots(Systolic)
```



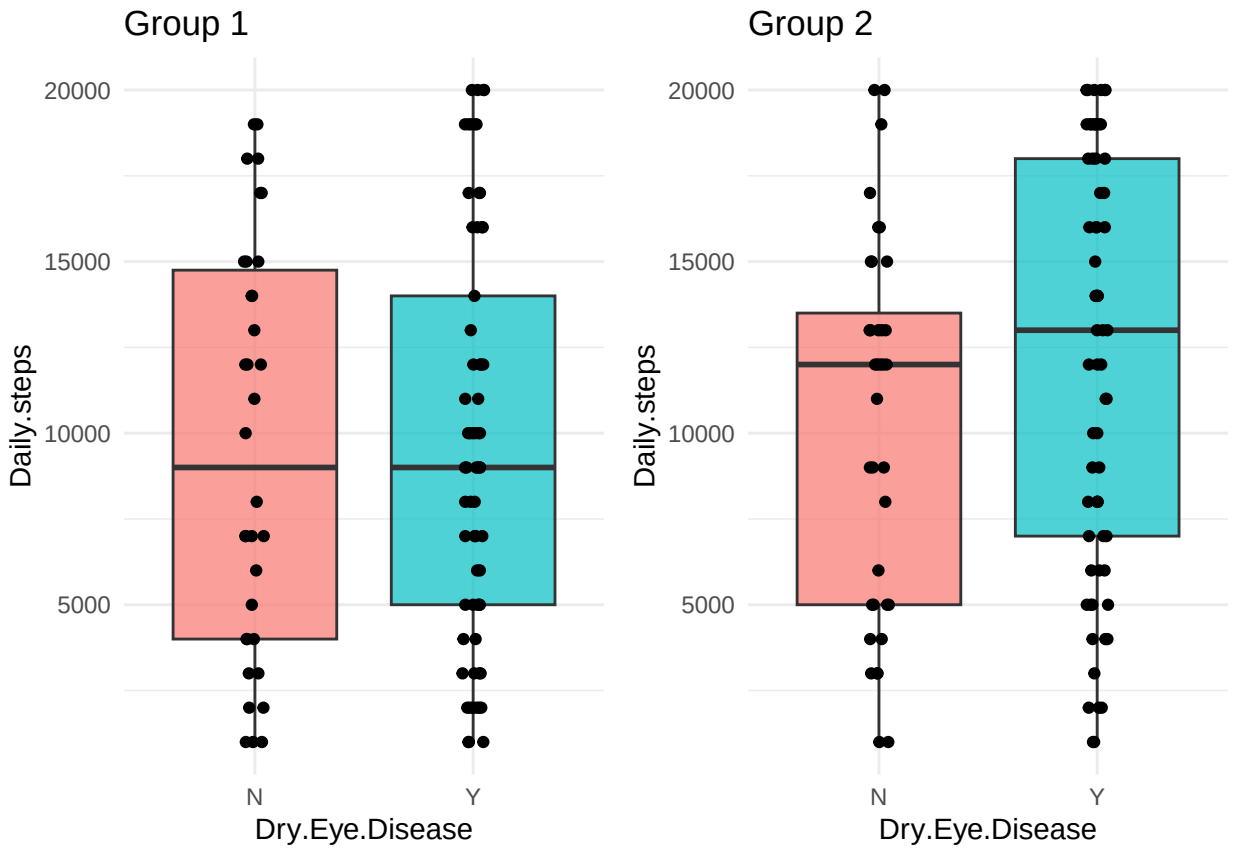

```
plot_boxplots(Heart.rate)
```



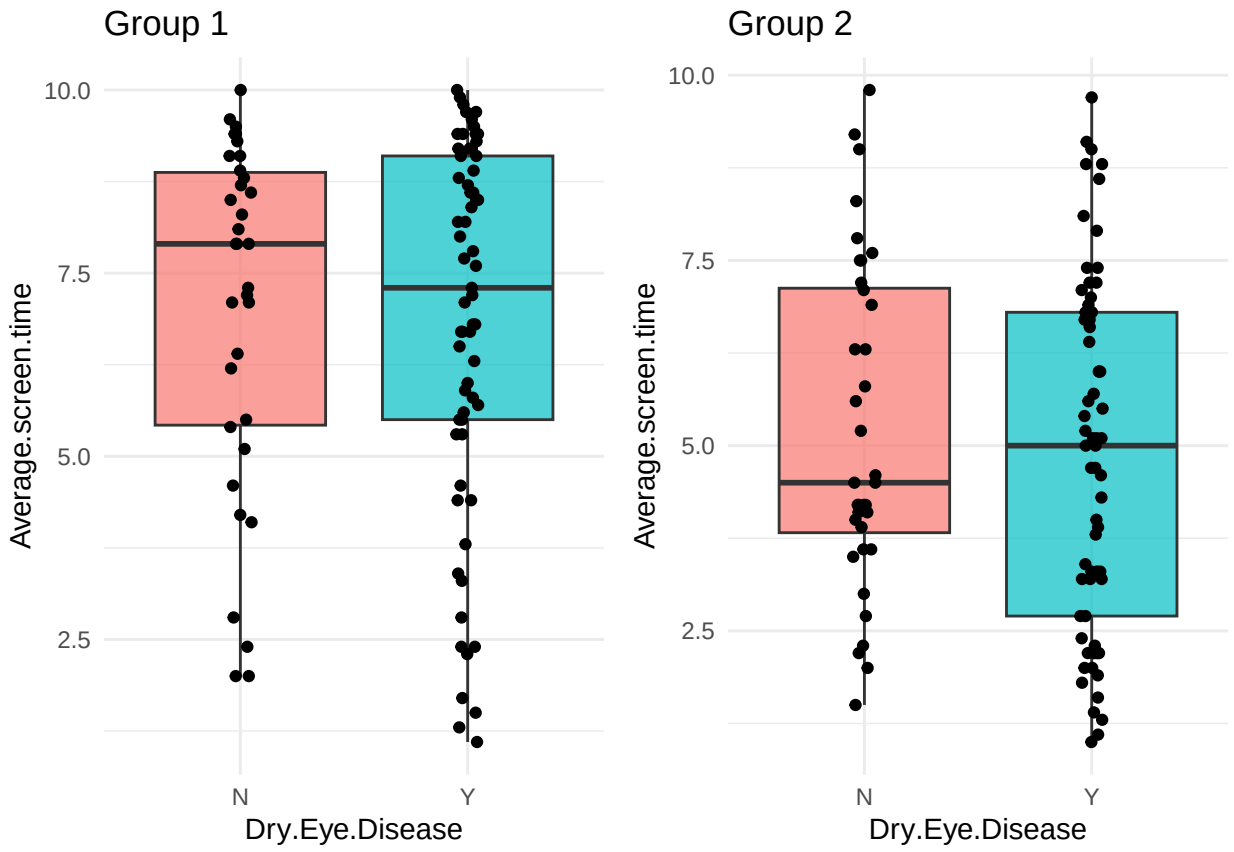

```
plot_boxplots(Physical.activity)
```



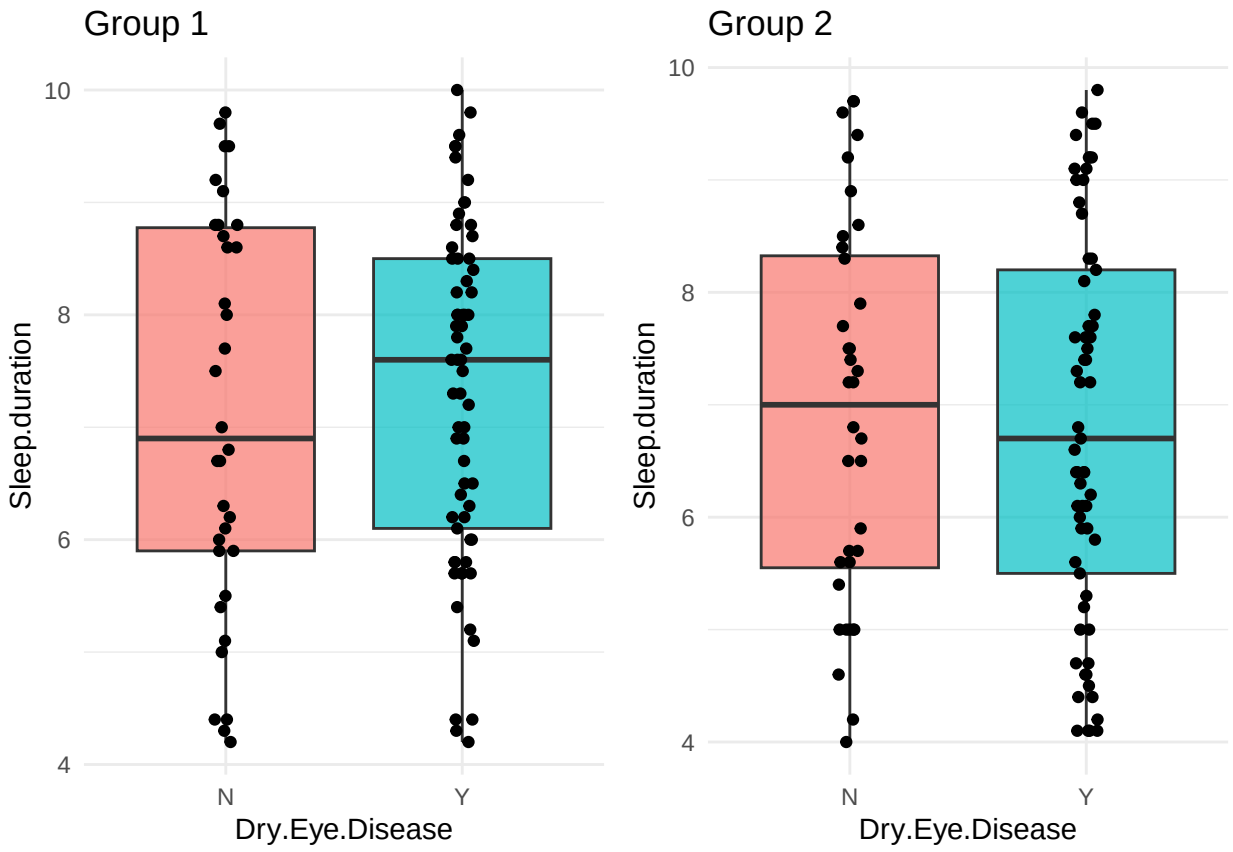

```
plot_boxplots(Daily.steps)
```




```
plot_boxplots(Average.screen.time)
```




```
plot_boxplots(Sleep.duration)
```



10.2.1 Normality Shapiro Test

```
group1.Yes = group1 %>% filter(Dry.Eye.Disease == "Y")
group2.Yes = group2 %>% filter(Dry.Eye.Disease == "Y")
```

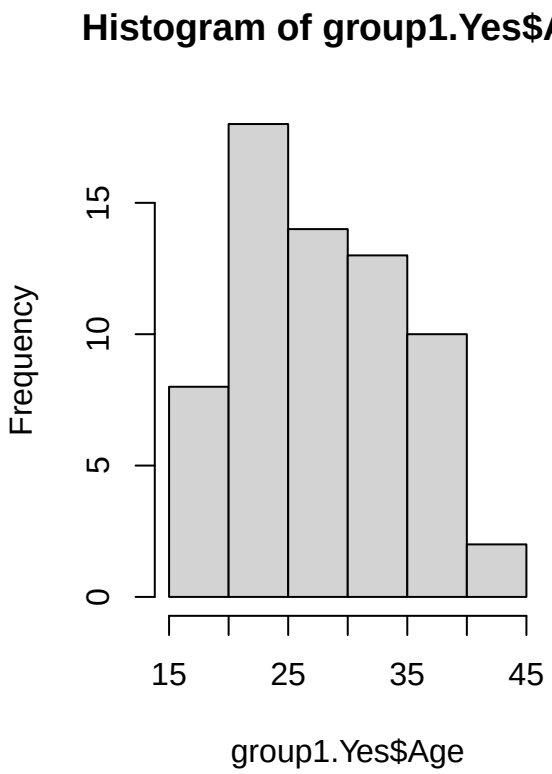
```
variables <- c("Age", "Diastolic", "Systolic", "Heart.rate", "Physical.activity", "Daily.steps", "Averag
```

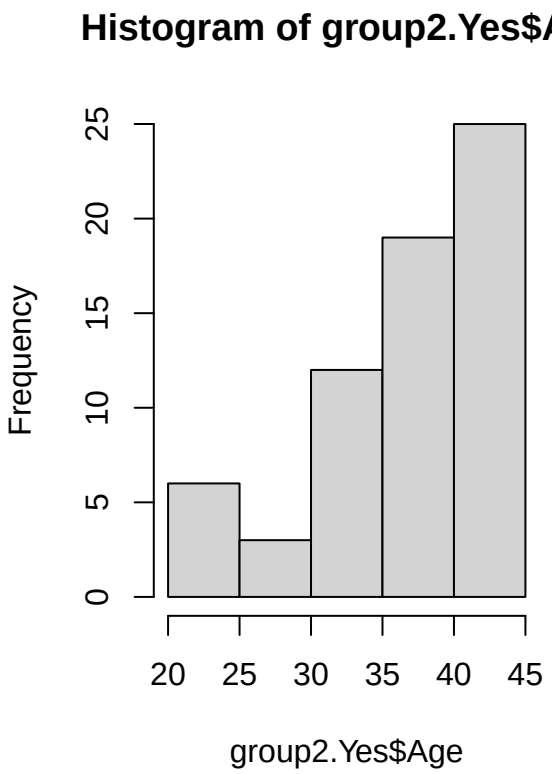
```
shapiro_results <- list()

for (var in variables) {
  test_group1 <- shapiro.test(group1.Yes[[var]])
  test_group2 <- shapiro.test(group2.Yes[[var]])

  # Store results in a data frame
  shapiro_results[[var]] <- data.frame(
    Variable = var,
    Group = c("Group 1", "Group 2"),
    W_statistic = c(test_group1$statistic, test_group2$statistic),
    P_value = c(test_group1$p.value, test_group2$p.value),
    Normality = ifelse(c(test_group1$p.value, test_group2$p.value) > 0.05, "Normal", "Not Normal")
  )
}

shapiro_table <- do.call(rbind, shapiro_results)
print(shapiro_table)
```

```
##                                 Variable    Group W_statistic      P_value
```

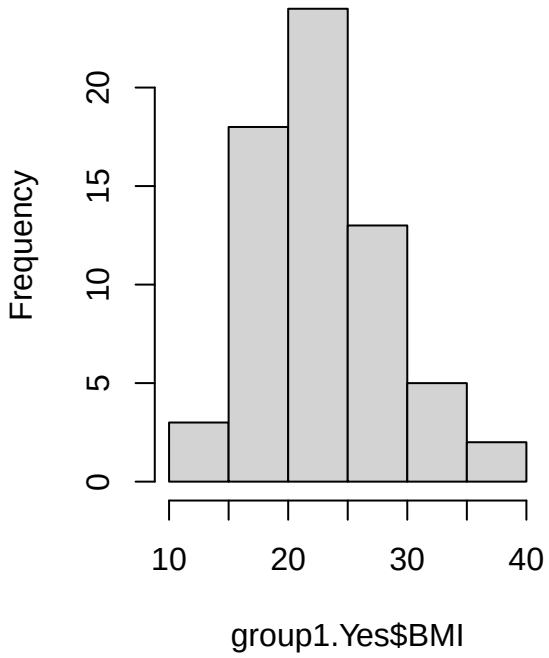
```
## Age.1                                Age Group 1   0.9674074 0.0841411393
## Age.2                                Age Group 2   0.9159313 0.0003022393
## Diastolic.1                    Diastolic Group 1   0.9237392 0.0006442693
## Diastolic.2                    Diastolic Group 2   0.9583149 0.0280301630
## Systolic.1                      Systolic Group 1   0.9280037 0.0009876772
## Systolic.2                      Systolic Group 2   0.9482551 0.0086870560
## Heart.rate.1                  Heart.rate Group 1   0.9314896 0.0014112047
## Heart.rate.2                  Heart.rate Group 2   0.9342165 0.0018748071
## Physical.activity.1     Physical.activity Group 1   0.9349362 0.0020222390
## Physical.activity.2     Physical.activity Group 2   0.9464739 0.0071035876
## Daily.steps.1                Daily.steps Group 1   0.9324305 0.0015557705
## Daily.steps.2                Daily.steps Group 2   0.9232402 0.0006132529
## Average.screen.time.1 Average.screen.time Group 1   0.9067543 0.0001291212
## Average.screen.time.2 Average.screen.time Group 2   0.9606688 0.0371546886
## Sleep.duration.1           Sleep.duration Group 1   0.9713946 0.1367184498
## Sleep.duration.2           Sleep.duration Group 2   0.9486250 0.0090598818
##                         Normality
## Age.1                      Normal
## Age.2                  Not Normal
## Diastolic.1            Not Normal
## Diastolic.2            Not Normal
## Systolic.1             Not Normal
## Systolic.2             Not Normal
## Heart.rate.1           Not Normal
## Heart.rate.2           Not Normal
## Physical.activity.1    Not Normal
## Physical.activity.2    Not Normal
## Daily.steps.1          Not Normal
## Daily.steps.2          Not Normal
## Average.screen.time.1  Not Normal
## Average.screen.time.2  Not Normal
## Sleep.duration.1           Normal
## Sleep.duration.2       Not Normal
```

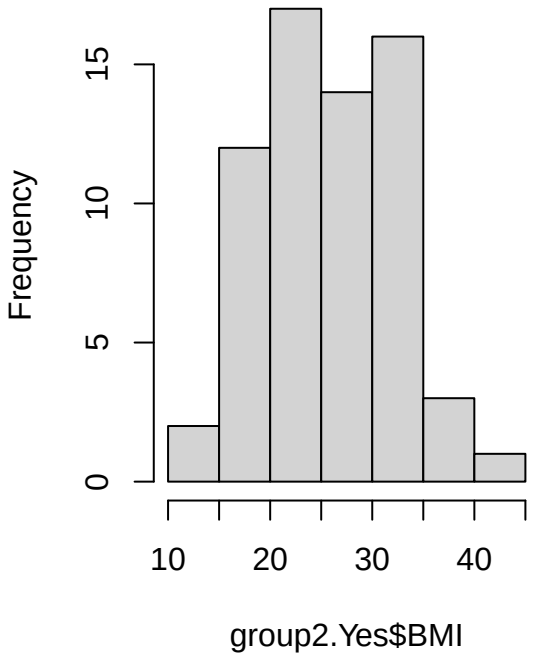
```r
par(mfrow=c(1,2))
hist(group1.Yes$Age)
hist(group2.Yes$Age)
```



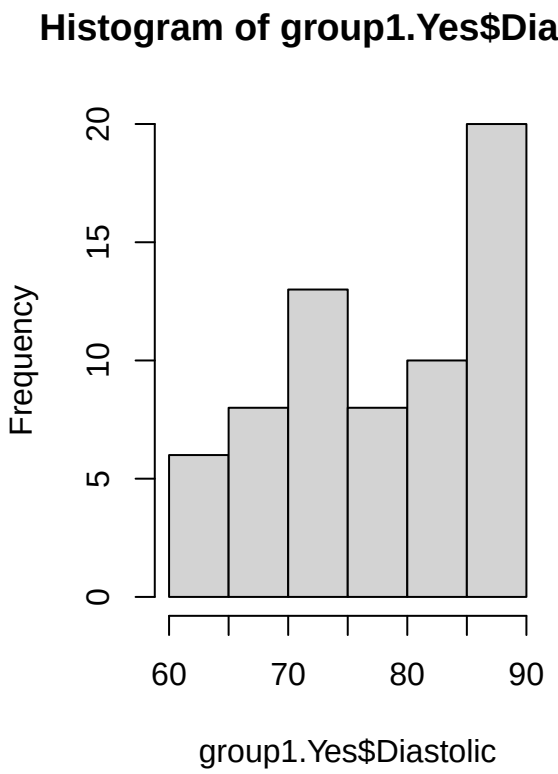


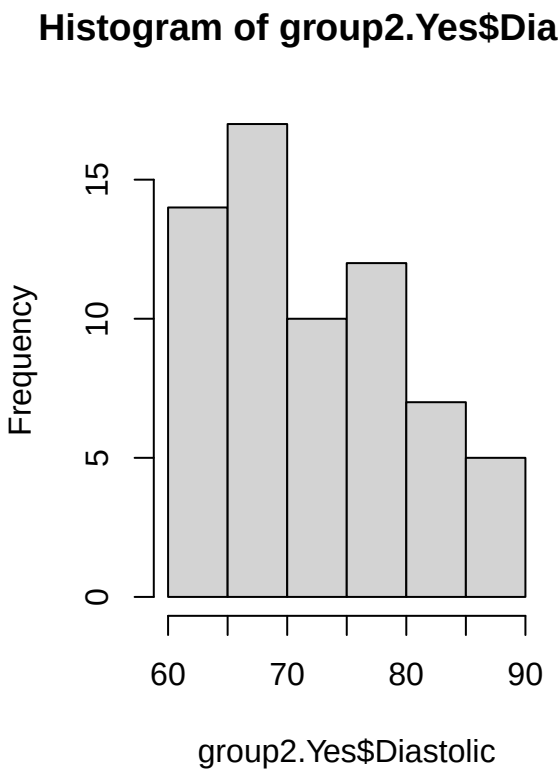

```
hist(group1.Yes$BMI)
hist(group2.Yes$BMI)
```



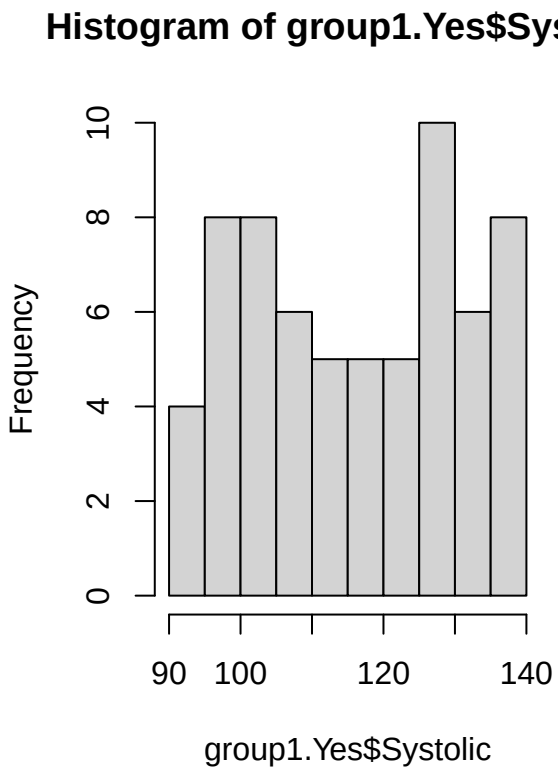


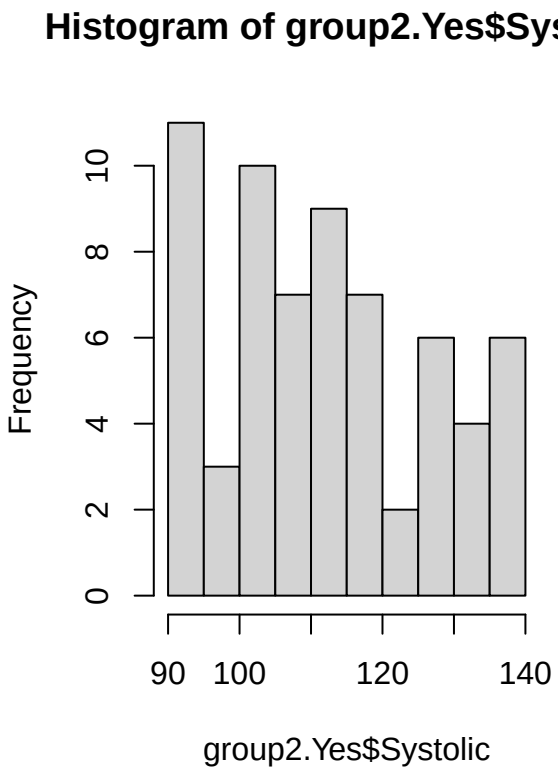

```
hist(group1.Yes$Diastolic)
hist(group2.Yes$Diastolic)
```



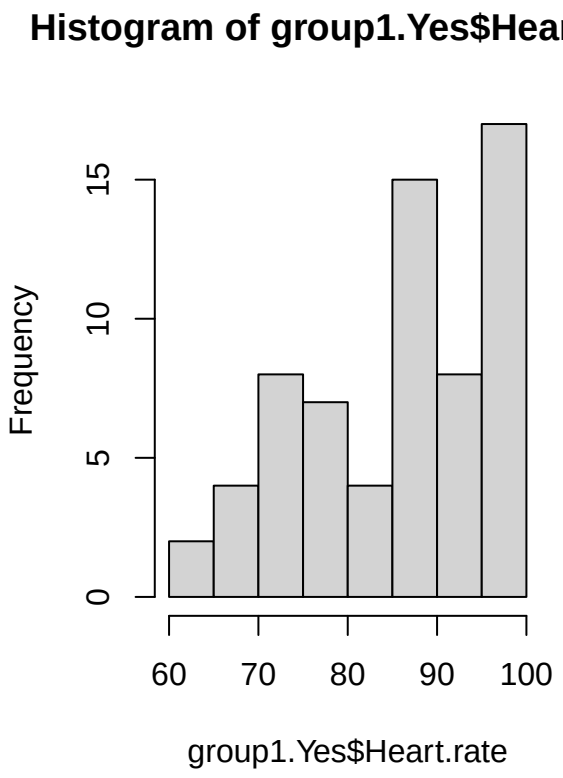


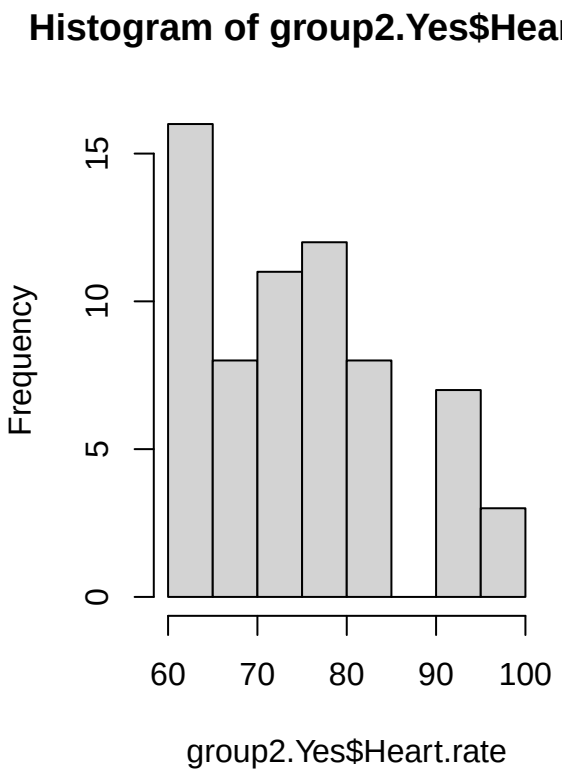

```
hist(group1.Yes$Systolic)
hist(group2.Yes$Systolic)
```



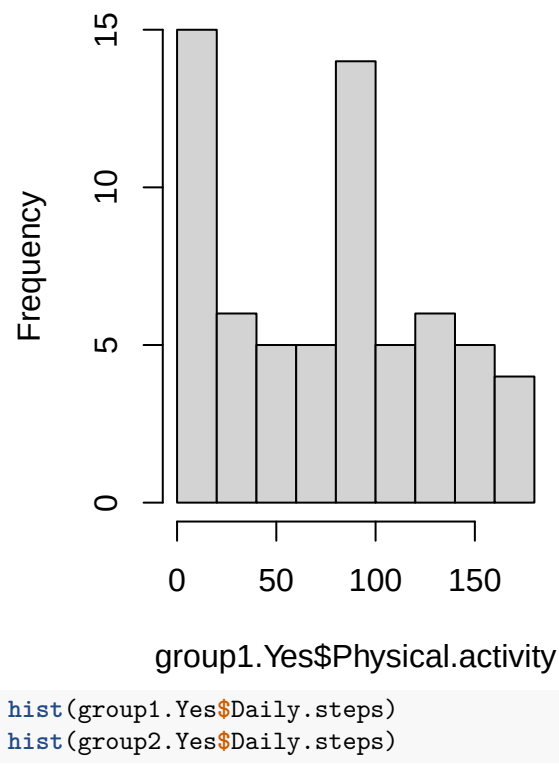


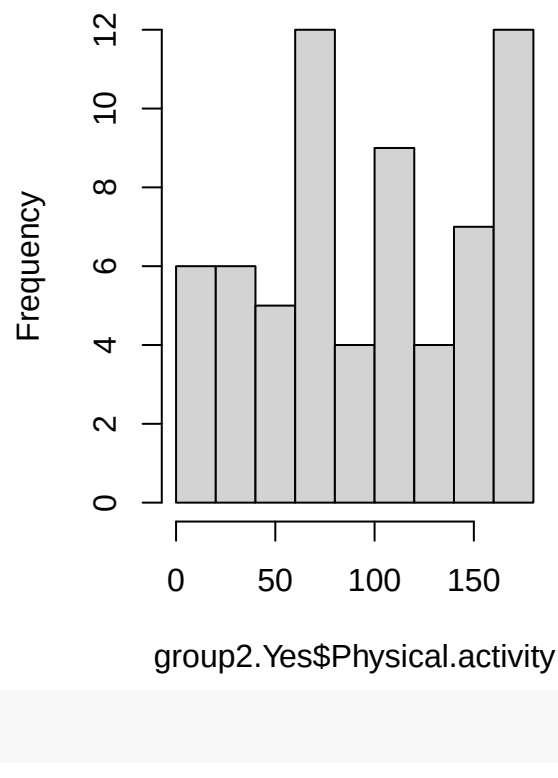

```
hist(group1.Yes$Heart.rate)
hist(group2.Yes$Heart.rate)
```



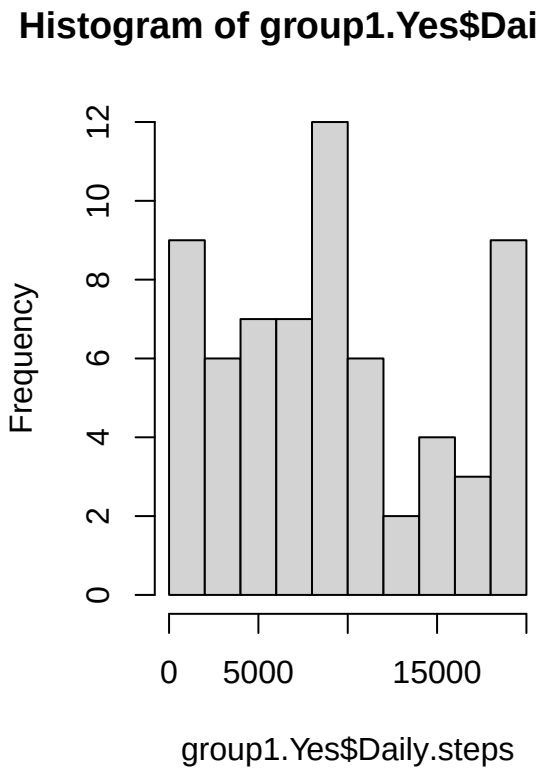


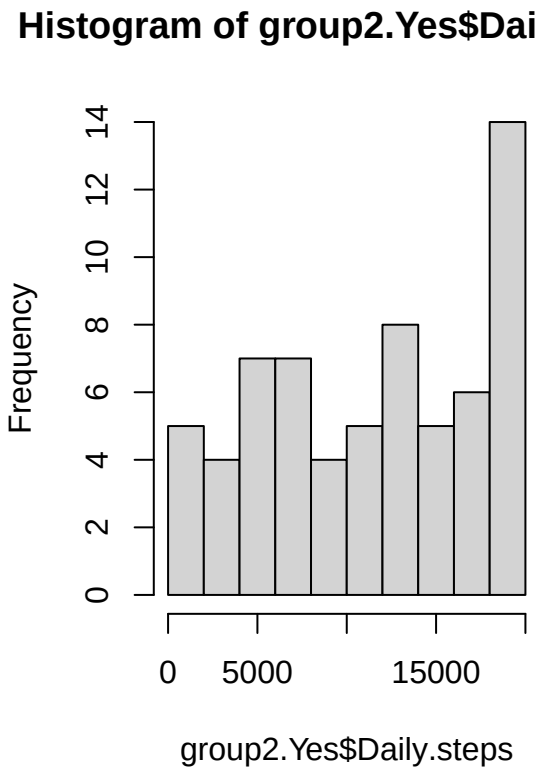

```
hist(group1.Yes$Physical.activity)
hist(group2.Yes$Physical.activity)
```



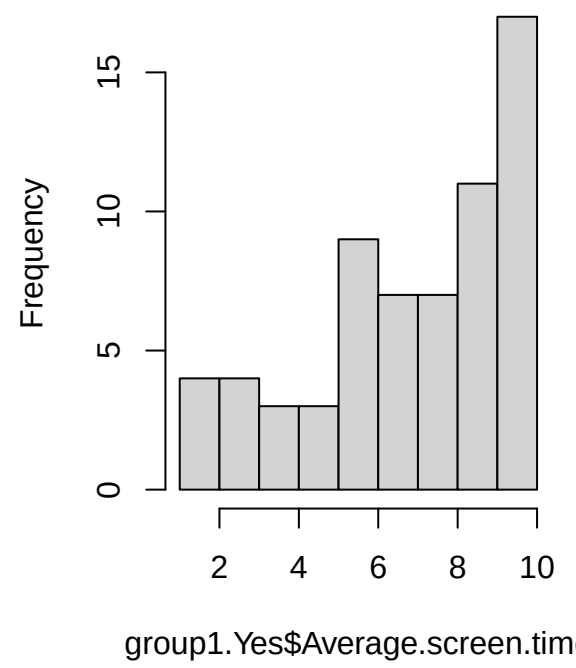


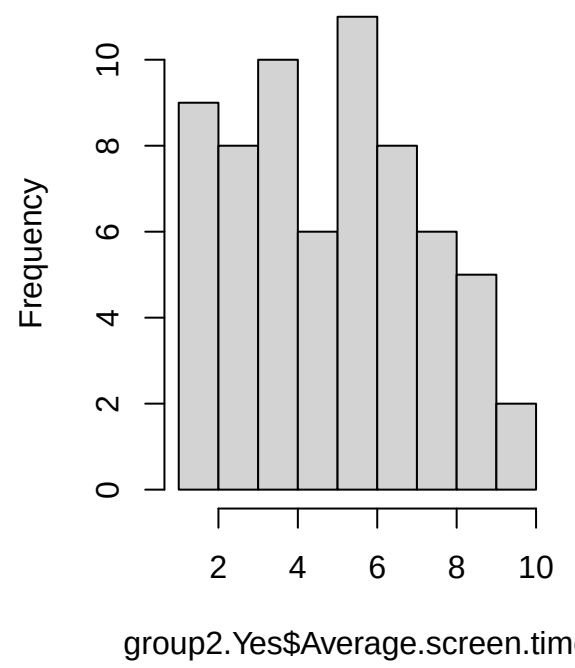

```
hist(group1.Yes$Daily.steps)
hist(group2.Yes$Daily.steps)
```



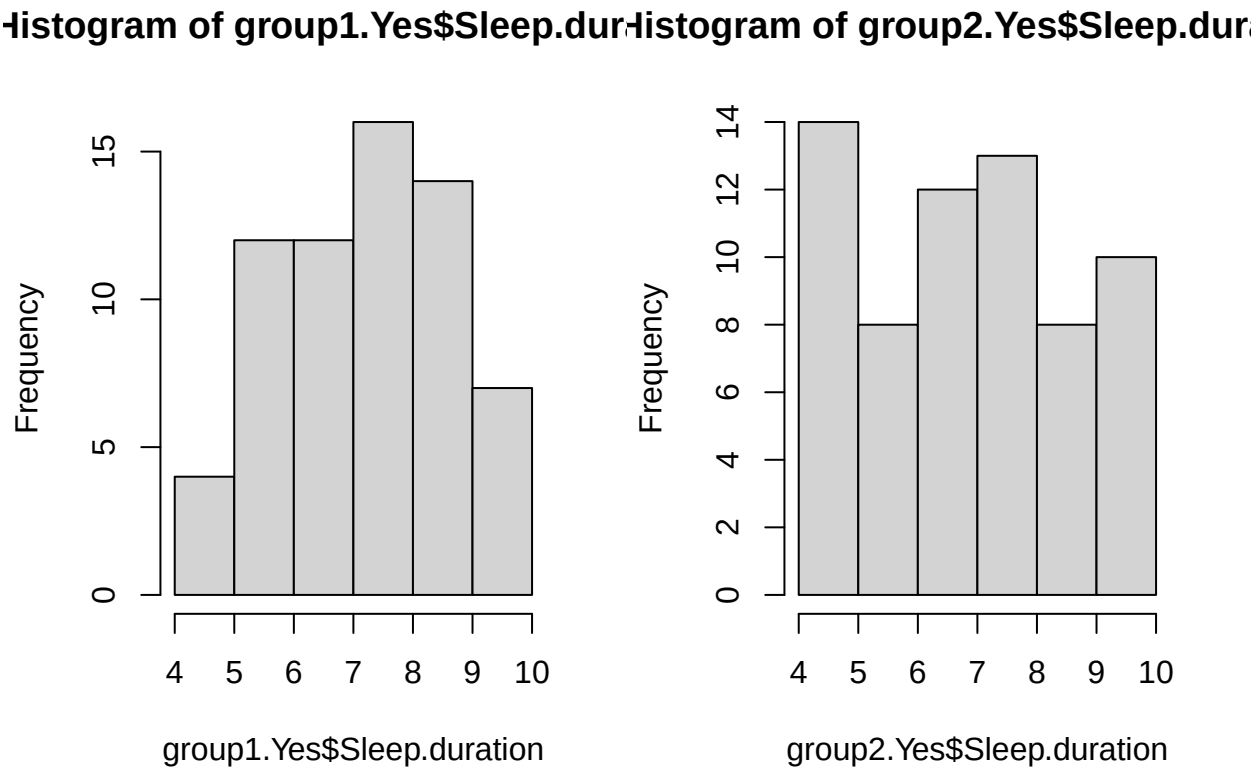



```r
hist(group1.Yes$Average.screen.time)
hist(group2.Yes$Average.screen.time)
```






```r
hist(group1.Yes$Sleep.duration)
hist(group2.Yes$Sleep.duration)
```



10.2.2 Wilcoxon test

```
# Initialize an empty list to store results
wilcoxon_results = list()

# Significance level
alpha = 0.05

# Perform Wilcoxon test for each variable
for (var in variables) {
  test = wilcox.test(group1.Yes[[var]], group2.Yes[[var]], exact = FALSE)

  # Store results in a data frame
  wilcoxon_results[[var]] <- data.frame(
    Variable = var,
    W_statistic = test$statistic,
    P_value = test$p.value,
    Decision = ifelse(test$p.value < alpha, "Reject H0", "Do Not Reject H0")
  )
}

# Combine results into a single data frame
wilcoxon_table <- do.call(rbind, wilcoxon_results)

# Print results
print(wilcoxon_table)
```

```
##                                Variable W_statistic      P_value
## Age                                 Age       798.5 9.183743e-10
```

```
## Diastolic                     Diastolic      2858.0 5.168820e-04
## Systolic                       Systolic      2455.5 1.106253e-01
## Heart.rate                   Heart.rate      3224.0 2.265052e-07
## Physical.activity     Physical.activity      1679.0 4.376868e-02
## Daily.steps                 Daily.steps      1660.0 3.498317e-02
## Average.screen.time Average.screen.time      3047.5 1.348412e-05
## Sleep.duration           Sleep.duration      2446.5 1.203346e-01
##                             Decision
## Age                        Reject H0
## Diastolic                  Reject H0
## Systolic            Do Not Reject H0
## Heart.rate                 Reject H0
## Physical.activity          Reject H0
## Daily.steps                Reject H0
## Average.screen.time        Reject H0
## Sleep.duration      Do Not Reject H0
```

11 Objective Question

```
calculate_mode <- function(x) {
  uniq_x <- unique(x)
  uniq_x[which.max(tabulate(match(x, uniq_x)))] # Most frequent value
}
```

```
sample.cluster_df_yes = sample.cluster_df %>% filter(Dry.Eye.Disease=="Y")

# Summarize data
summary_table_yes <- sample.cluster_df_yes %>%
  group_by(cluster) %>% # Group by cluster
  summarise(
    across(where(is.numeric), mean, na.rm = TRUE),  # Median for numeric
    across(where(is.character), calculate_mode),      # Mode for categorical
    across(where(is.factor), calculate_mode)         # Mode for factors
  )
```

```
## Warning: There was 1 warning in `summarise()`.
## i In argument: `across(where(is.numeric), mean, na.rm = TRUE)`.
## i In group 1: `cluster = 1`.
## Caused by warning:
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
## Supply arguments directly to `.fns` through an anonymous function instead.
##
##   # Previously
##   across(a:b, mean, na.rm = TRUE)
##
##   # Now
##   across(a:b, \(x) mean(x, na.rm = TRUE))
```

```
print(summary_table_yes)
```

```
## # A tibble: 2 x 29
##   cluster   Age Sleep.duration Systolic Diastolic Heart.rate Daily.steps
##     <int> <dbl>          <dbl>    <dbl>     <dbl>      <dbl>       <dbl>
## 1       1  28.8           7.30     117.      78.1       86.2       9615.
```

```
## 2       2  37            6.83     112.      72.6       74.9      11938.
## # i 22 more variables: Physical.activity <dbl>, Height <dbl>, Weight <dbl>,
## #   Average.screen.time <dbl>, BMI <dbl>, Gender <fct>, Sleep.quality <fct>,
## #   Stress.level <fct>, Sleep.disorder <fct>, Wake.up.during.night <fct>,
## #   Feel.sleepy.during.day <fct>, Caffeine.consumption <fct>,
## #   Alcohol.consumption <fct>, Smoking <fct>, Medical.issue <fct>,
## #   Ongoing.medication <fct>, Smart.device.before.bed <fct>,
## #   Blue.light.filter <fct>, Discomfort.Eye.strain <fct>, ...
```

```r
sample.cluster_df_no = sample.cluster_df %>% filter(Dry.Eye.Disease=="N")

summary_table_no <- sample.cluster_df_no %>%
  group_by(cluster) %>% # Group by cluster
  summarise(
    across(where(is.numeric), mean, na.rm = TRUE),
    across(where(is.character), calculate_mode),
    across(where(is.factor), calculate_mode)
  )

print(summary_table_no)
```

```
## # A tibble: 2 x 29
##   cluster   Age Sleep.duration Systolic Diastolic Heart.rate Daily.steps
##     <int> <dbl>          <dbl>    <dbl>     <dbl>      <dbl>       <dbl>
## 1       1  26.4           7.13     119.      78.8       88.1        9500
## 2       2  34.6           6.89     118.      72.7       73.8       10250
## # i 22 more variables: Physical.activity <dbl>, Height <dbl>, Weight <dbl>,
## #   Average.screen.time <dbl>, BMI <dbl>, Gender <fct>, Sleep.quality <fct>,
## #   Stress.level <fct>, Sleep.disorder <fct>, Wake.up.during.night <fct>,
## #   Feel.sleepy.during.day <fct>, Caffeine.consumption <fct>,
## #   Alcohol.consumption <fct>, Smoking <fct>, Medical.issue <fct>,
## #   Ongoing.medication <fct>, Smart.device.before.bed <fct>,
## #   Blue.light.filter <fct>, Discomfort.Eye.strain <fct>, ...
```

```r
library(writexl)

# Save to an Excel file
write_xlsx(summary_table_no, "summary_table_no.xlsx")
write_xlsx(summary_table_yes, "summary_table_yes.xlsx")
```

12 Results

For identifying risk groups among the clusters, was considered a comparison of the average (for numerical) and modes (for categorical) grouped by Dry.Eye.Disease and cluster. If there was not difference in the average considering both clusters between each classes of Dry.Eye.Disease, then it was considered that the feature had not influence in the diagnosis. The same with the mode for categorical features. The results were as follow:

- For life choices: people who drink alcohol and use Blue light filter might be at higher risk of suffering a positive diagnostic of Dry Eye Disease.
- For physical health aspects: Females who have been diagnosed with sleep disorder, eye strain discomfort and eye irritation might be at higher risk of having a positive diagnose of Dry Eye Disease.

13 Conclusions

- The Hopkins statistic revealed low clustering tendency, and the Rand and VI indexes also revealed values that suggests a random clustering assignment for this data set.
- Neither of the advanced clustering algorithms learned in class provided a better result than the ones explored in this study: kmeans, pam, and hierarchical clustering.
- The wilcoxon test revealed that there was a statistical difference between the clusters numerical features, nevertheless the classes for the categorical features appeared to have similar distributions for both clusters.

14 Next Steps

- Run supervised ML algorithms in the data grouped in each cluster and compare the results obtained.
- Run the analysis used in this report in a bigger sample of the data set.
- Find new sources of data related to dry eye disease and compare results.