Data Analysis and Statistical Models - Project Work

Master in Artificial Intelligence and Data Science a.a. 2024/2025

Marco Longo - Francesca Ricci - Maria Rotella

2025-03-25

Data Analysis and Statistical Models

Case Study: Coffee Composition

Indice

Abstract

Il presente studio, basato sulle competenze acquisite nel corso **Data Analysis and Statistical Models**, analizza il dataset **coffee** del pacchetto *pgmm* di R con l'obiettivo di identificare gruppi di osservazioni simili.

Dopo un'analisi preliminare delle variabili e della tendenza al clustering, vengono applicati il clustering partizionale, quello gerarchico e i modelli di misture gaussiane. Questi metodi consentono di esplorare la struttura latente dei dati e di individuare relazioni tra i gruppi, mettendo in evidenza le caratteristiche principali che distinguono ciascun cluster. Lo studio si propone di offrire una visione accessibile del pattern dei dati, fornendo informazioni utili per comprendere le dinamiche che emergono nel dataset.

Analisi dei dati del caffè

Dataset

Il dataset contiene 43 osservazioni, ognuna riferita ad un particolare tipo di caffè.
Per ogni caffè viene indicato, mediante variabili categoriali, il **paese** di provenienza e la **varietà** che lo caratterizza, nella fattispecie "robusta" o "arabica". Le altre 12 variabili numeriche fanno riferimento alle caratteristiche chimico-fisiche del caffè.

Analisi preliminare delle variabili

Di seguito sono elencati in dettaglio le variabili presenti nel dataset.

- paese di provenienza del caffè
- varietà
- acqua
- peso del chicco
- prodotto estratto
- valore del ph
- acido libero
- contenuto minerale
- grasso
- caffeina
- trigonellina
- acido clorogenico
- acido neoclorogenico
- acido isoclorogenico

```
library(pgmm)
data(coffee)
head(coffee)
```

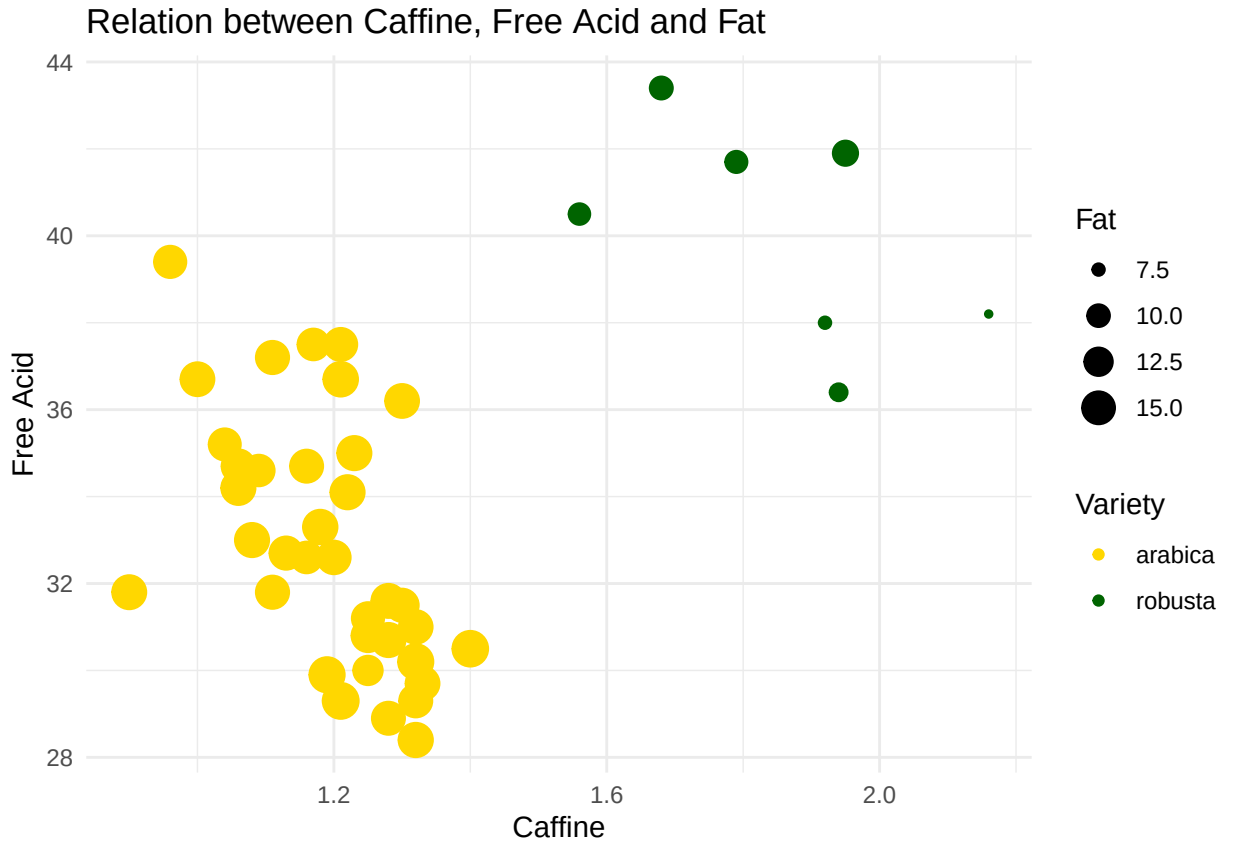
```
##   Variety  Country     Water Bean Weight Extract Yield ph Value Free Acid
## 1       1   mexico  8.939999       156.6          33.5     5.80      32.7
## 2       1   mexico  7.400000       157.3          32.1     5.81      30.8
## 3       1 guatemal  9.740000       152.9          33.1     5.26      36.7
## 4       1 honduras 10.400000       174.0          31.5     5.61      34.2
## 5       1 salvador 10.540000       145.1          35.2     5.77      31.8
## 6       1 salvador 10.000000       156.4          34.5     5.83      32.6
##   Mineral Content  Fat Caffine Trigonelline Chlorogenic Acid
## 1            3.80 15.2    1.13         1.03             5.38
## 2            3.71 15.0    1.25         1.01             5.13
## 3            4.15 16.1    1.21         1.05             5.94
## 4            3.94 15.8    1.06         0.94             5.87
## 5            4.09 15.2    1.11         0.99             5.09
## 6            3.88 15.4    1.20         0.81             5.30
##   Neochlorogenic Acid Isochlorogenic Acid
## 1                0.40                0.79
## 2                0.32                0.97
## 3                0.24                0.76
## 4                0.39                0.59
## 5                0.49                0.72
## 6                0.43                0.69
```

Poichè l'analisi è mirata alla composizione del caffè, verranno escluse le due variabili categoriali. In particolare, il paese di origine non è determinante ai fini dello studio e verrà indicato, in fase di preprocessing, nell'identificativo della tupla.

La varietà di un caffè è definita dalla combinazione delle caratteristiche intrinseche del chicco, come il contenuto di caffeina, i grassi, il pH e i minerali presenti. E' piuttosto noto tra i bevitori di caffè, che la varietà arabica è più delicata e dolce, mentre la robusta è caratterizzata da un gusto più deciso e forte.

Il grafico seguente mostra come anche solo 3 componenti del caffè possono già essere distintive rispetto alla varietà.

```
library(ggplot2)
df <- coffee
df$Variety <- factor(coffee$Variety, levels = c(1, 2), labels = c("arabica", "robusta"))
ggplot(df, aes(x = Caffine, y = `Free Acid`, color = Variety, size = Fat)) +
  geom_point() +
  scale_color_manual(values = colors2) +
  labs(title = "Relation between Caffine, Free Acid and Fat",
       x = "Caffine",
       y = "Free Acid") +
  theme_minimal()
```



Per tale ragione la varietà verrà esclusa dallo studio, che considererà le 12 componenti chimico-fisiche.

Preprocessing

Si modifica il row name per ogni osservazione, al fine di avere una migliore visualizzazione nei grafici che verranno mostrati nel project work. Il row name sarà formato dalla sigla del paese di orgine (3 char), dal progressivo del caffè per quel paese e da un suffisso indicante la varietà ("a" o "r" per "arabica" o "robusta", rispettivamente). Ad esempio il caffè "bra2_a" indica il secondo caffè del Brasile (secondo l'ordine del dataset) di tipo Arabica.

Successivamente, si crea un nuovo dataframe con le sole numeriche.

```
df$Country <- as.character(df$Country)
df$Country[df$Country == "Costadav"] <- "cdavorio"
df$Country_Count <- ave(df$Country, df$Country, FUN = seq_along)
row.names(df) <- with(df,
                      paste0(
                        tolower(substr(Country, 1, 3)),    # prefisso paese
                        Country_Count,                     # progressivo
                        ifelse(Variety == "arabica", "_a", "_r")   # Arabica o Robusta
                      ))
df$Country_Count <- NULL

df.to.scale <- df[, -c(1, 2)]
head(df.to.scale)
```

```
##            Water Bean Weight Extract Yield ph Value Free Acid Mineral Content
## mex1_a  8.939999       156.6          33.5     5.80      32.7            3.80
## mex2_a  7.400000       157.3          32.1     5.81      30.8            3.71
## gua1_a  9.740000       152.9          33.1     5.26      36.7            4.15
## hon1_a 10.400000       174.0          31.5     5.61      34.2            3.94
## sal1_a 10.540000       145.1          35.2     5.77      31.8            4.09
## sal2_a 10.000000       156.4          34.5     5.83      32.6            3.88
##         Fat Caffine Trigonelline Chlorogenic Acid Neochlorogenic Acid
## mex1_a 15.2    1.13         1.03             5.38                0.40
## mex2_a 15.0    1.25         1.01             5.13                0.32
## gua1_a 16.1    1.21         1.05             5.94                0.24
## hon1_a 15.8    1.06         0.94             5.87                0.39
## sal1_a 15.2    1.11         0.99             5.09                0.49
## sal2_a 15.4    1.20         0.81             5.30                0.43
##        Isochlorogenic Acid
## mex1_a                0.79
## mex2_a                0.97
## gua1_a                0.76
## hon1_a                0.59
## sal1_a                0.72
## sal2_a                0.69
```

La fase finale del preprocessing è la standardizzazione dei dati mediante la funzione *scale* di R, essendo le variabili numeriche di grandezze differenti.

```
df.num <- scale(df.to.scale)
```

Tendenza al clustering

La *Hopkins Statistic* è una misura utilizzata per valutare se un dataset è "clusterizzabile", se presenta, cioè, una struttura che può essere ben rappresentata da cluster.

Usiamo la funzione **hopkins** della libreria **clustertend**.

```
library(clustertend)
set.seed(11)

random.df <- apply(df.num, 2, function(x){runif(length(x), min(x), max(x))})
random.df <- as.data.frame(random.df)

hopkins(random.df, nrow(random.df)-1)
```

```
## $H
## [1] 0.5023185
```

```
hopkins(df.num, nrow(df.num)-1)
```

```
## $H
## [1] 0.3531155
```

Considerando i risultati della funzione (0.5 sul campione casuale indica assenza di struttura), il valore di Hopkins pari a 0.35 che i dati potrebbero avere una moderata struttura di cluster.

Tuttavia, è da considerare che la Hopkins Statistic tende a sottovalutare la clusterizzabilità con dati non uniformemente distribuiti, oppure in presenza di interazioni tra variabili che invece formano cluster ben definiti.

Analisi dei componenti principali (PCA)

In questa sezione viene effettuata l'analisi dei componenti principali, ritenuta necessaria nel presente studio, dato l'elevato numero di variabili contenute nel dataset **coffee**. La PCA, infatti, permette di ridurre la dimensionalità del dataset, identificando nuove variabili (componenti principali, appunto) che sono combinazioni lineari delle variabili originali. Queste componenti massimizzano la varianza *spiegata*, permettendo di rappresentare i dati in uno spazio più compatto senza perdere informazioni essenziali.

Utilizziamo la funzione **PCA** del pacchetto **FactoMineR**, che effettua, tra l'altro, una standardizzazione preliminare dei dati.

```
library(FactoMineR)

pca.res <- PCA(df.to.scale, ncp = 5, scale.unit = TRUE, graph = FALSE)
df.pca <- as.data.frame(pca.res$ind$coord)
```

La funzione **get_eigenvalue** di **factoextra** permette di recuperare ed analizzare gli autovalori, ossia le quantità di varianza associata a ciascuna componente principale.

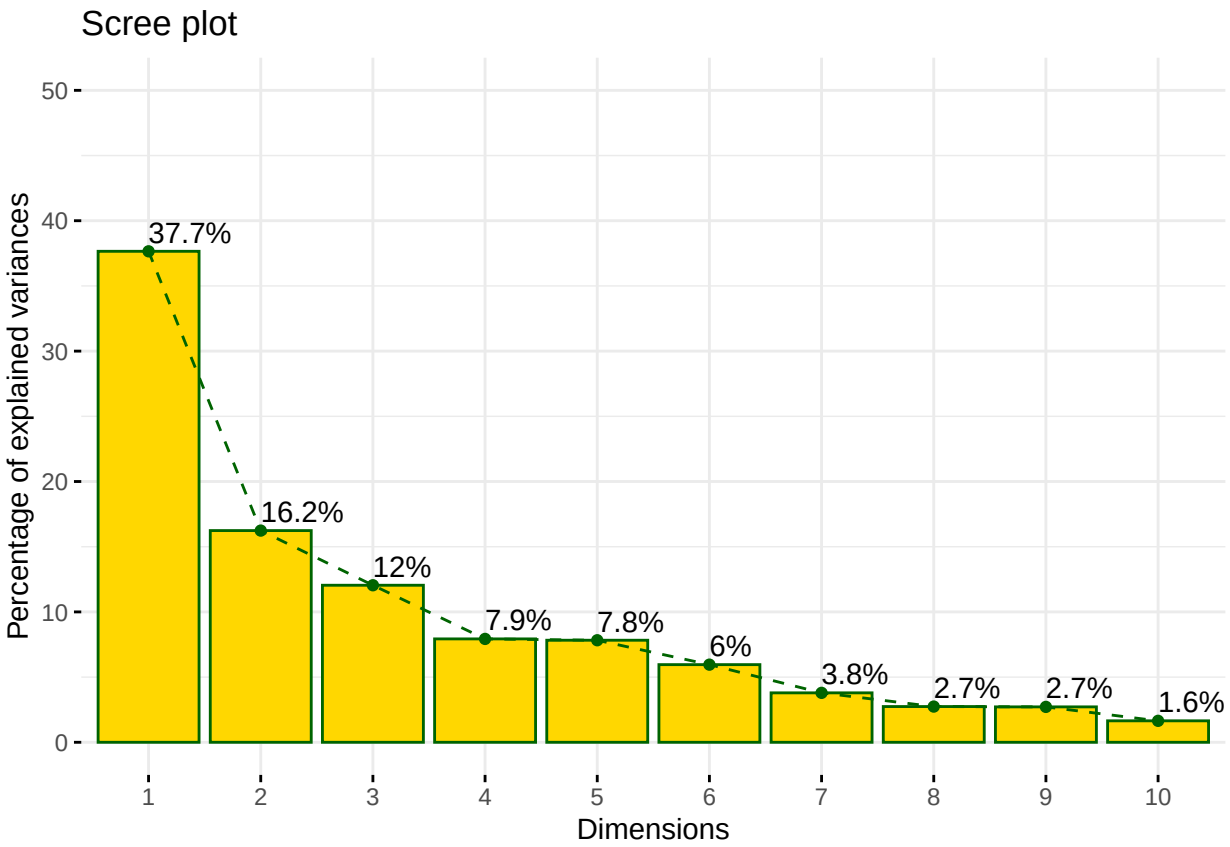
```
library(factoextra)
eig.val <- get_eigenvalue(pca.res)
eig.val
```

```
##        eigenvalue variance.percent cumulative.variance.percent
## Dim.1  4.51857884       37.6548237                    37.65482
## Dim.2  1.94835527       16.2362939                    53.89112
## Dim.3  1.44522963       12.0435802                    65.93470
## Dim.4  0.95151424        7.9292854                    73.86398
## Dim.5  0.93917393        7.8264494                    81.69043
## Dim.6  0.71472589        5.9560491                    87.64648
## Dim.7  0.45503111        3.7919259                    91.43841
## Dim.8  0.32894870        2.7412392                    94.17965
## Dim.9  0.32588053        2.7156711                    96.89532
## Dim.10 0.19751351        1.6459459                    98.54126
## Dim.11 0.09223152        0.7685960                    99.30986
## Dim.12 0.08281683        0.6901402                   100.00000
```

La tabella mostra le dimensioni in ordine di grandezza dell'autovalore, ossia di *varianza spiegata*. Quindi per le prime componenti principali è grande (37.%, 16.2%, ...) e diminuisce man mano per le successive. L'ultima colonna ci permette di determinare quante dimensioni considerare, al fine di raggiungere una percentuale accettabile di varianza cumulata. Si nota infatti che le prime 5 dimensioni permettono di spiegare più dell'80% di varianza. Pertanto è lecito considerare un numero di componenti principali pari a 5.

La funzione *fviz_eig* permette di avere una visione immediata della distribuzione della varianza spiegata per componente.

```
fviz_eig(pca.res,
         addlabels = TRUE, ylim = c(0, 50),
         barfill = "gold", barcolor = "darkgreen", linecolor = "darkgreen",
         linetype = "dashed", ggtheme = theme_minimal()
)
```



Di seguito si riporta la correlazione tra le componenenti principali e le variabili originarie del dataset.

```
var <- get_pca_var(pca.res)
var$contrib
```

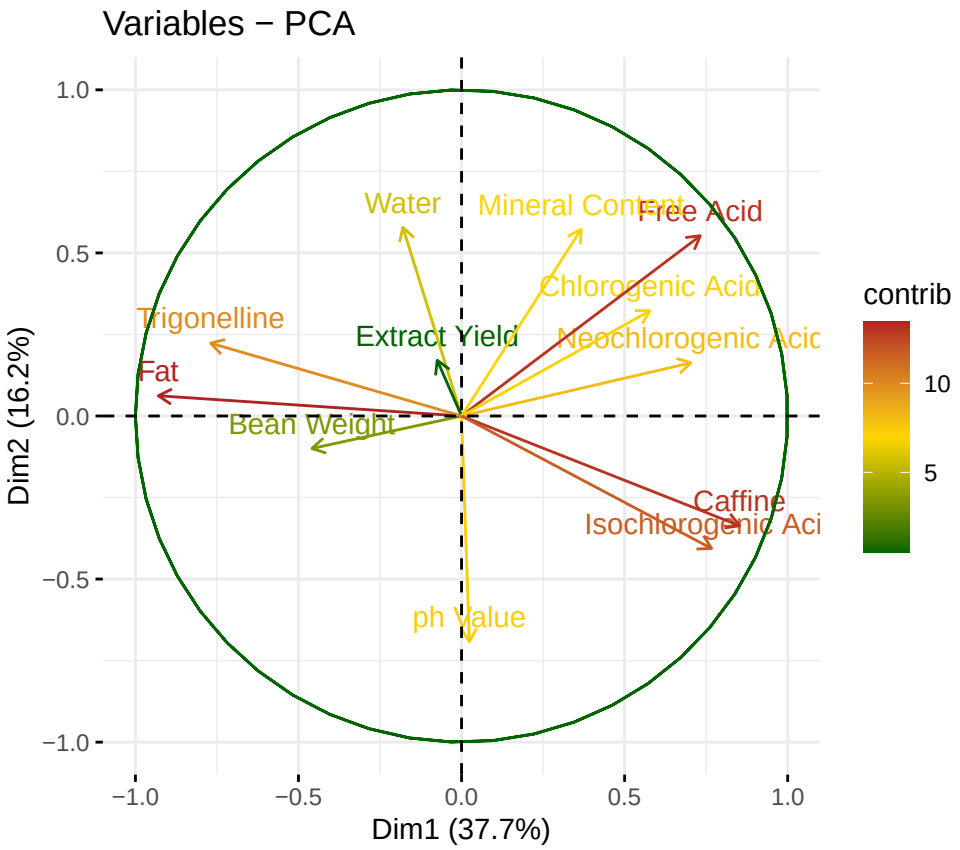
```
##                          Dim.1      Dim.2       Dim.3       Dim.4        Dim.5
## Water                0.71578169 17.1721183  0.86085377  0.54606881 49.308874899
## Bean Weight          4.65879266  0.5094312  0.04574555 72.45829447  0.196742429
## Extract Yield        0.12291866  1.4964519 55.33400102  1.93480404  1.410221358
## ph Value             0.01280497 24.5928279 15.03249701  1.92585475  5.041710570
## Free Acid           11.85920915 15.6817448  0.59148590  0.05177707  0.106350813
## Mineral Content      2.97530396 16.8077643  6.16514396  5.45495616 13.084874038
## Fat                 19.12684175  0.1977564  0.88479854  0.37392196  0.876459268
## Caffine             16.08588760  5.8081702  0.28987436  2.88219232  0.001828468
## Trigonelline        13.10292692  2.5613398  3.29609545  1.68467186  6.243741400
## Chlorogenic Acid     7.38460301  5.3401437 13.59057311  4.42093338 15.997827467
## Neochlorogenic Acid 10.94963144  1.3642738  0.92808904  7.94717975  6.447280229
## Isochlorogenic Acid 13.00529818  8.4679778  2.98084230  0.31934541  1.284089060
```

Le variabili con il maggior contributo alla prima componente principale risultano essere il grasso (19.13) e la caffeina (16.08), mentre sulla seconda il ph (24.59), l'acqua (17.17) e i minerali (16.80).

La funzione **fviz_pca_var** permette di plottare i contributi visti in tabella. Si tratta di un grafico bidimensionale che mostra la correlazione tra le variabili e le prime due componenti principali, che vengono utilizzate come coordinate nel grafico.

Le PC possono essere interpretate in base alle variabili con cui mostrano la maggiore correlazione, sia in direzione positiva che negativa.

```
fviz_pca_var(pca.res, col.var = "contrib", repel = FALSE,
             col.circle = "darkgreen", labelsize = 4, gradient.cols = colors3)
```

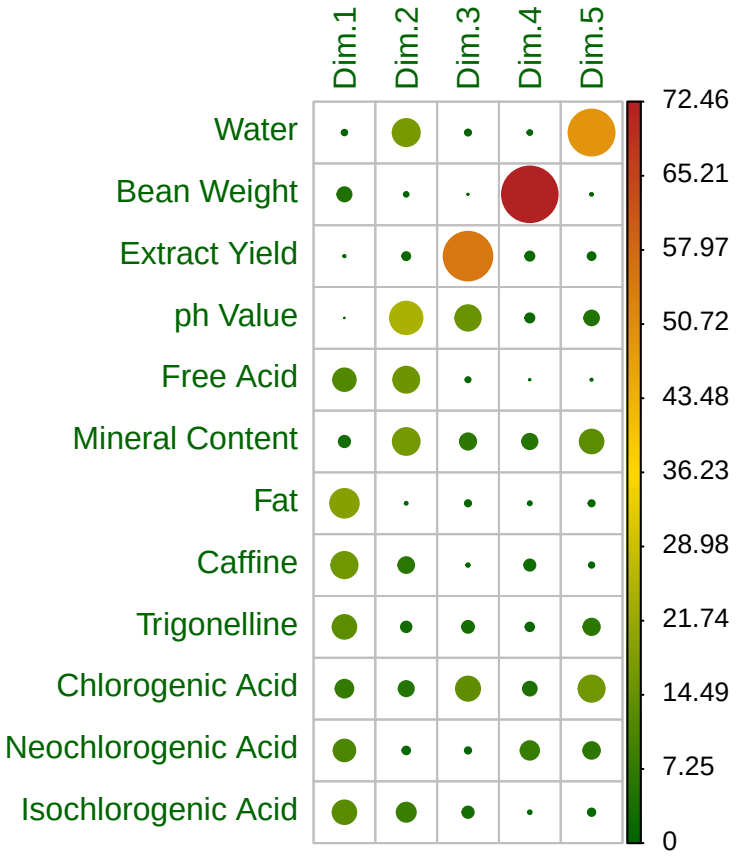


Gli assi del grafico sono dati dalle prime due componenti principali, che insieme spiegano circa il 54% della varianza totale.

I vettori che si allontanano dall'origine rappresentano le variabili originali, e la loro lunghezza indica quanto sono ben descritte dalle prime due componenti principali. L'angolo tra i vettori è un'approssimazione della correlazione tra le variabili: un angolo piccolo indica correlazioni positive (come tra acido libero e acido neoclorogenico), un angolo retto indica una non correlazione (come tra acido isoclorogenico e minerali), angoli piatti indicano variabili negativamente correlate (come peso del chicco e trigonellina). Si distinguono in rosso le variabili più lunghe, quindi con un maggior contributo alla componente, come il grasso, la caffeina e l'acido libero.

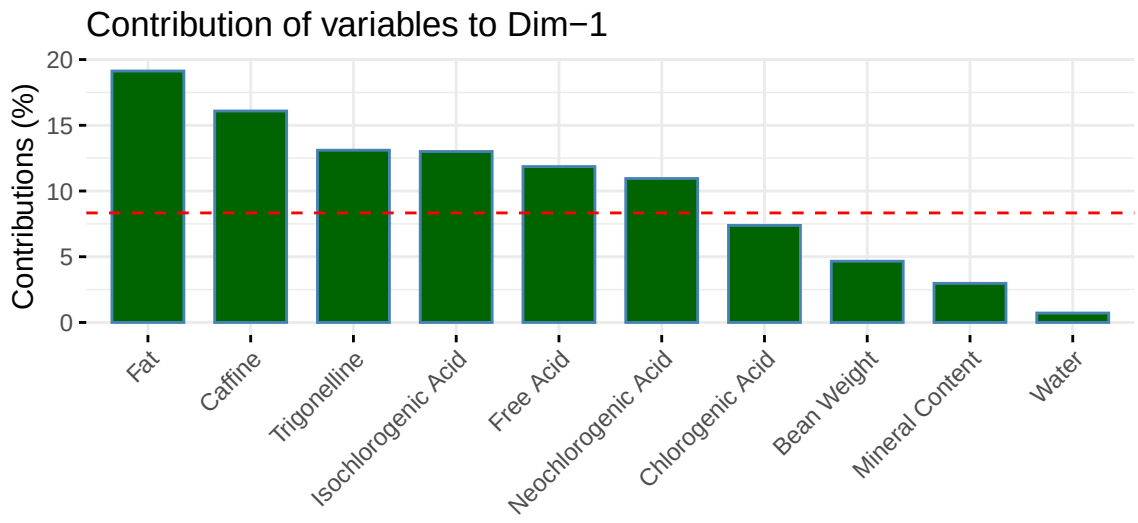
Con il *corrplot* di seguito, è possibile evidenziare meglio le variabili che contribuiscono di più per ciascuna dimensione.

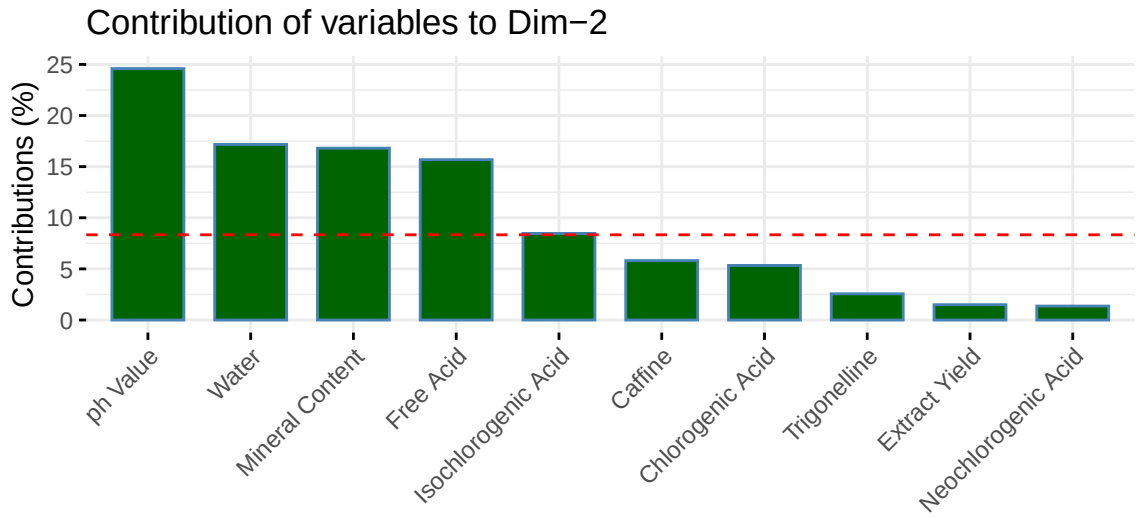
```
library(corrplot)
colors.palette <- colorRampPalette(colors3)(200)
corrplot(var$contrib,
         is.corr = FALSE,
         col = colors.palette,
         tl.col = "darkgreen",
         cl.ratio = 0.2,
         cl.align = "l")
```

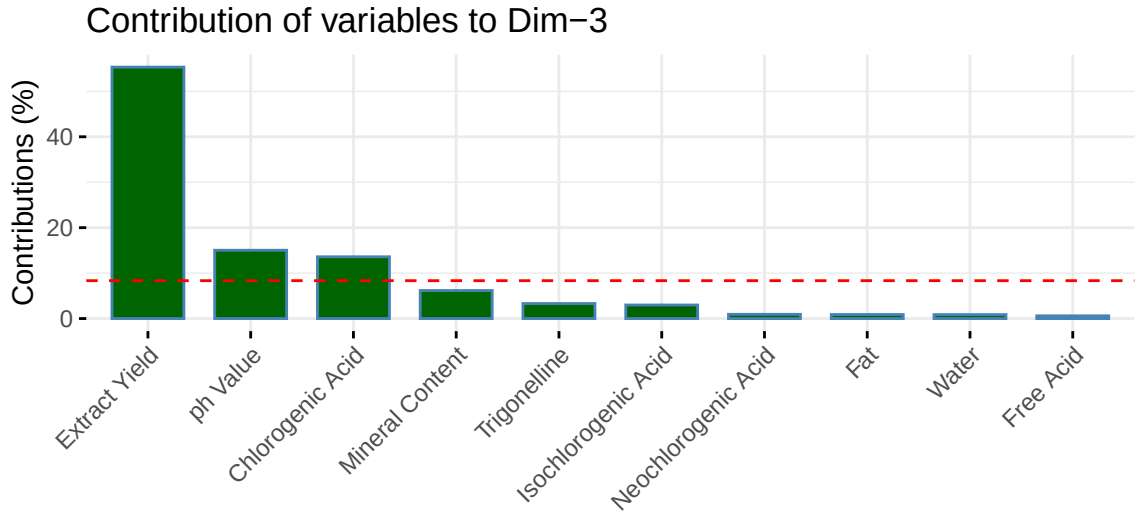


Oltre a quanto già scritto sopra, si evidenziano i contributi di variabili quali prodotto estratto, peso del chicco e acqua rispettivamente nella terza, quarta e quinta componente.

Tramite **fviz_contrib** si visualzzano i contributi delle variabili sulle componenti principali.







Misure di distanza per il clustering

In questa sezione viene analizzata la *similarità* tra i caffè in base alle componenti chimico-fisiche, indispensabile per la Clustering Analysis che è finalizzata a determinare gruppi di elementi "simili", ossia non *distanti*.

Si calcola la distanza euclidea tra gli elementi del dataframe mediante la funzione R *dist*:

```
dist.eucl <- dist(df.num, method = "euclidean")
round(as.matrix(dist.eucl)[1:9, 1:9], 1)
```

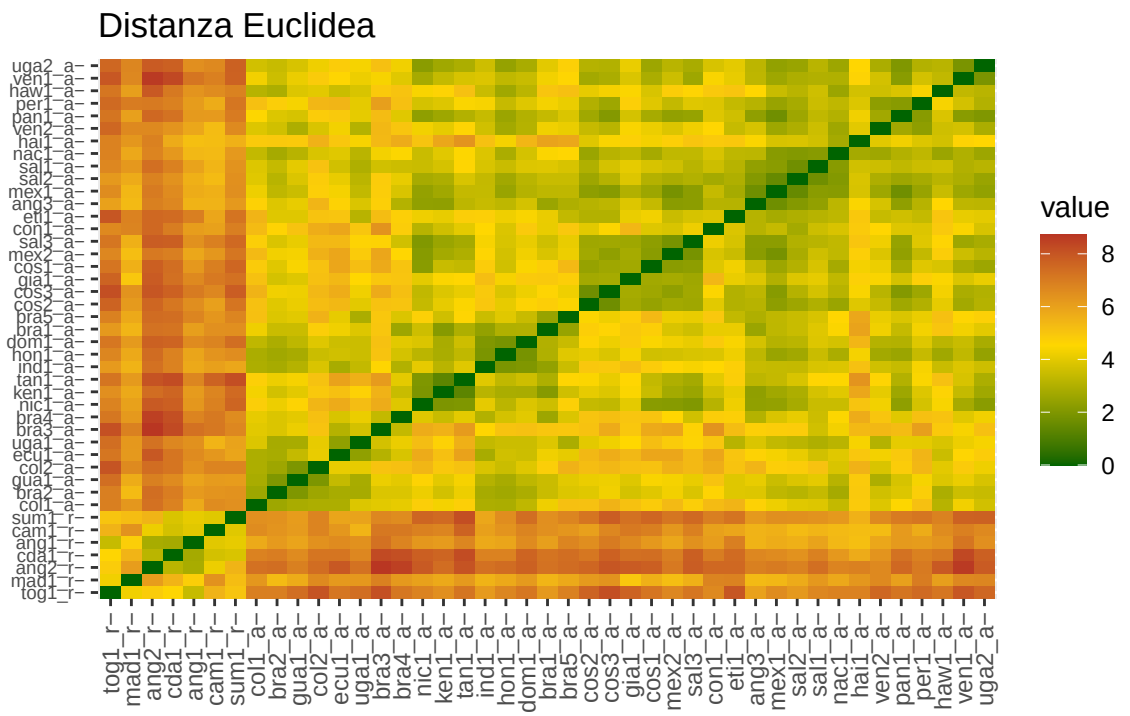
```
##        mex1_a mex2_a gua1_a hon1_a sal1_a sal2_a sal3_a nic1_a nac1_a
## mex1_a    0.0    1.8    3.5    2.5    2.2    1.5    2.3    2.4    2.2
## mex2_a    1.8    0.0    4.3    3.7    3.6    2.9    1.8    2.1    3.3
## gua1_a    3.5    4.3    0.0    2.8    4.0    3.8    4.1    4.6    3.1
## hon1_a    2.5    3.7    2.8    0.0    3.5    2.7    3.8    3.2    2.9
## sal1_a    2.2    3.6    4.0    3.5    0.0    1.7    3.2    3.5    2.2
## sal2_a    1.5    2.9    3.8    2.7    1.7    0.0    3.0    2.8    2.1
## sal3_a    2.3    1.8    4.1    3.8    3.2    3.0    0.0    2.0    3.3
## nic1_a    2.4    2.1    4.6    3.2    3.5    2.8    2.0    0.0    3.5
## nac1_a    2.2    3.3    3.1    2.9    2.2    2.1    3.3    3.5    0.0
```

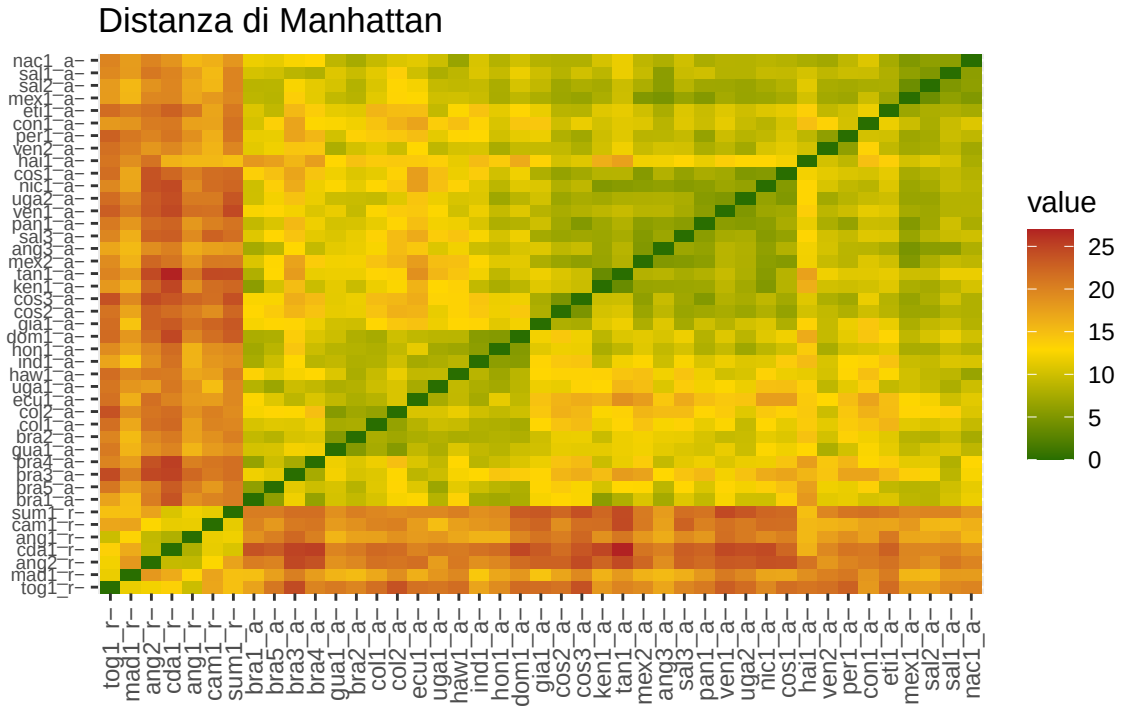
Stesso procedimento utilizzando la distanza di Manhattan, che calcola la somma dei valori assoluti delle differenze lungo ciascuna dimensione.

```
dist.manh <- dist(df.num, method = "manhattan")
round(as.matrix(dist.manh)[1:9, 1:9], 1)
```

```
##        mex1_a mex2_a gua1_a hon1_a sal1_a sal2_a sal3_a nic1_a nac1_a
## mex1_a    0.0    5.2    9.1    7.6    6.1    3.9    6.6    6.8    5.1
## mex2_a    5.2    0.0   12.1   11.0    9.4    7.8    5.3    5.8    8.9
## gua1_a    9.1   12.1    0.0    8.6   10.8   10.4   10.7   12.7    8.9
## hon1_a    7.6   11.0    8.6    0.0    9.3    7.7   10.3    8.8    8.3
## sal1_a    6.1    9.4   10.8    9.3    0.0    5.0    9.6    9.3    5.9
## sal2_a    3.9    7.8   10.4    7.7    5.0    0.0    8.5    7.5    6.0
## sal3_a    6.6    5.3   10.7   10.3    9.6    8.5    0.0    5.5    9.9
## nic1_a    6.8    5.8   12.7    8.8    9.3    7.5    5.5    0.0    8.7
## nac1_a    5.1    8.9    8.9    8.3    5.9    6.0    9.9    8.7    0.0
```

Si visualizzano graficamente i risultati mediante *fviz_dist* di factoextra:





Graficamente non si apprezzano significative differenze tra i due metodi. Per entrambi è evidente una maggiore distanza tra i caffè distinti per varietà: infatti la "striscia" rossa è in corrispondenza dei caffè il cui suffisso è "_r" (Robusta) rispetto a quelli con "_a" (Arabica), come anticipato in fase di analisi preliminare dei dati.

Clustering partizionale

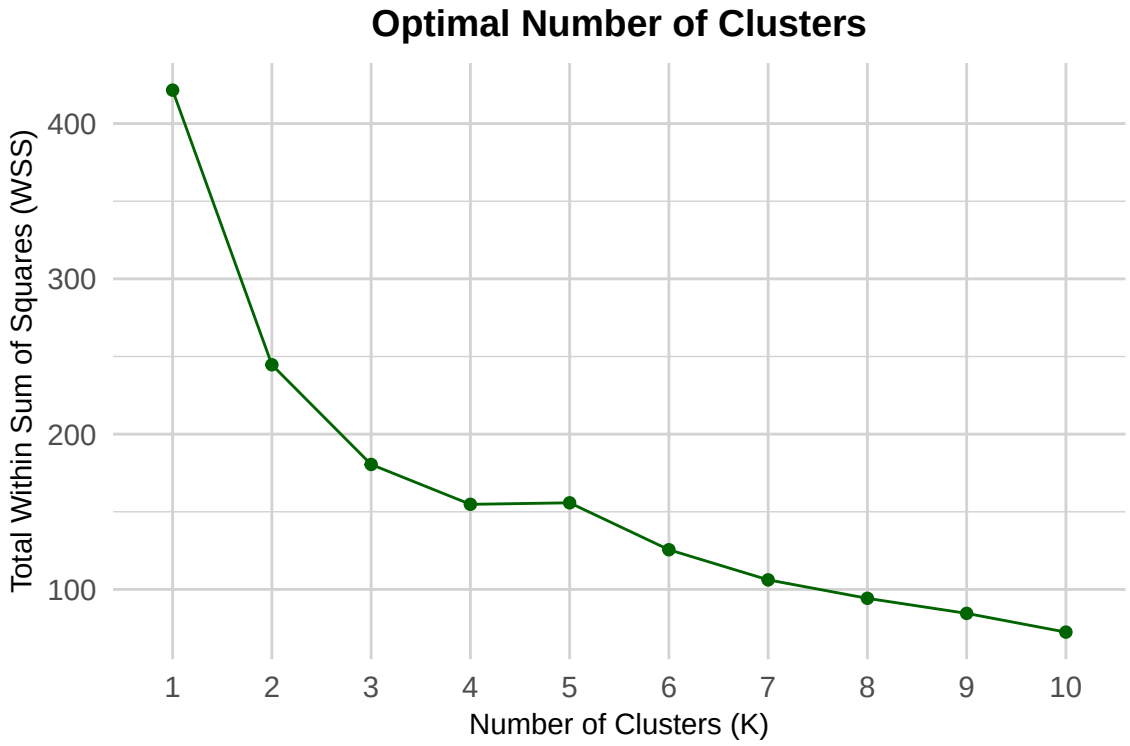
I metodi di Partitioning Clustering sono utilizzati per classificare le osservazioni del dataset in gruppi in base alla loro similarità. In questa sezione utilizzeremo due metodi:

- **K-means**, in cui ogni cluster è rappresentato dal centroide, ovvero un punto ideale che corrisponde alla media delle coordinate di tutti i punti appartenenti al cluster;
- **K-medoids**, in cui ogni cluster è rappresentato da un elemento specifico del cluster, chiamato *medoid*, che è l'osservazione più rappresentativa, ossia, il punto con la distanza complessiva minima rispetto agli altri punti del cluster.

Entrambi prevedono che si conosca a priori il numero K di cluster da determinare, pertanto nella sezione successiva si valuterà il numero ottimale di cluster.

Numero ottimale dei cluster

Poichè il numero di cluster che si vuole ottenere è da specificare come input degli algoritmi che seguono, si utilizza la funzione **fviz_nbclust** di **factoextra** col metodo **WSS** (Within Sum of Squares), che mostra, per ogni k cluster, la somma dei quadrati delle distanze euclidee tra ogni punto del dataset ed il centroide assegnato del cluster di appartenza.

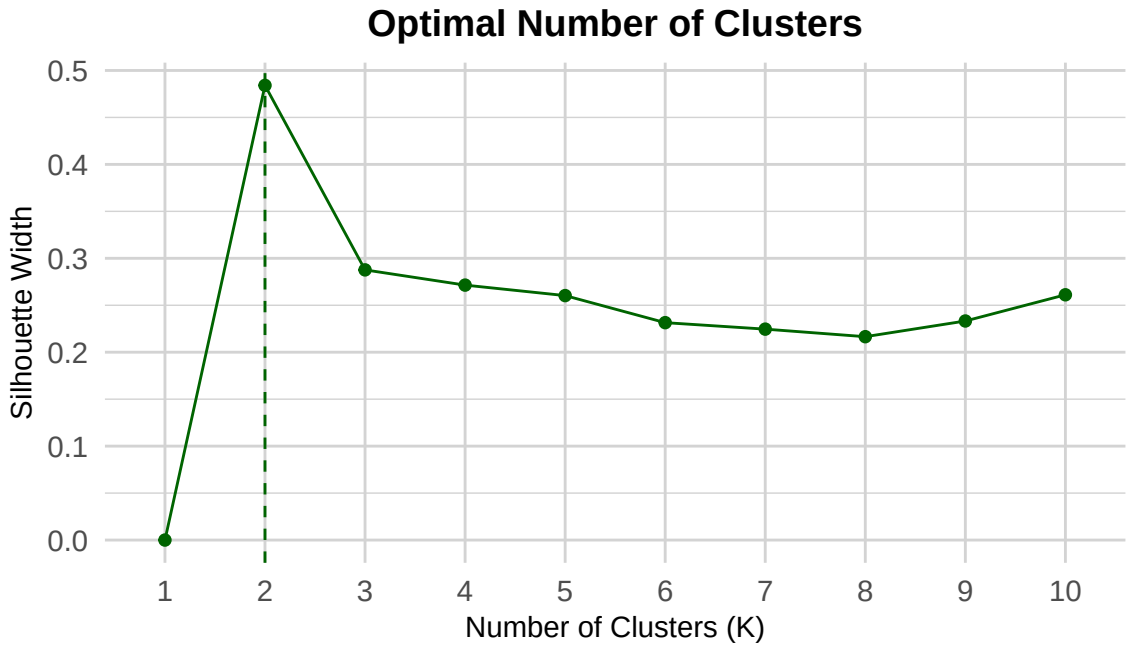


Si nota un "gomito" formato dalla linea in corrispondenza di K=3, indice che potrebbe essere il numero ottimale di cluster da considerare.

Il metodo **Gap Statistic** confronta la variabilità interna dei cluster con quella di un dataset casuale.

Il risultato conferma k = 3.

In ultima istanza, si applica il metodo **Silhouette Width**, che analizza quanto un oggetto è simile al cluster a cui appartiene rispetto agli altri cluster, fornendo un'indicazione della qualità della separazione tra i cluster.



In questo caso il grafico evidenzia come il numero più adatto per gli algoritmi partizionali sia k = 2.

K-Means

Il clustering K-means è l'algoritmo di machine learning non supervisionato più comunemente utilizzato per suddividere un dataset in un insieme di k cluster. L'idea di base del metodo consiste nel definire cluster tali da minimizzare la variazione totale tra item all'interno dello stesso cluster.

Di seguito è eseguita la funzione **K-means** del pacchetto **stats**, applicata alle componenti principali calcolate nelle sezioni precedenti, per k = 3. Successivamente i risultati della stessa funzione saranno valutati per K = 2 e K = 4.

```
set.seed(123)
km3.res <- kmeans(df.pca, 3, nstart = 50)
```

Di seguito i risultati relativi ai cluster, ai rispetti *centri* ed alla loro numerosità:

```
km3.res$cluster
```

```
## mex1_a mex2_a gua1_a hon1_a sal1_a sal2_a sal3_a nic1_a nac1_a cos1_a cos2_a
##      3      3      2      2      3      3      3      3      3      3      3
## cos3_a pan1_a hai1_a dom1_a ven1_a ven2_a col1_a col2_a ecu1_a per1_a bra1_a
##      3      3      3      2      3      3      2      2      2      3      2
## bra2_a bra3_a bra4_a bra5_a cda1_r tog1_r cam1_r con1_a ang1_r ang2_r ang3_a
##      2      2      2      2      1      1      1      3      1      1      3
## eti1_a uga1_a uga2_a ken1_a tan1_a mad1_r ind1_a sum1_r gia1_a haw1_a
##      3      2      3      3      3      1      2      1      3      2
```

```
km3.res$centers
```

```
##        Dim.1     Dim.2      Dim.3       Dim.4       Dim.5
## 1  4.5710659 -0.382535 -0.2712533  0.01078269 -0.19208277
## 2 -0.6253584  1.651997 -0.4142204  0.07468266  0.17887748
## 3 -1.0564747 -0.929555  0.3499027 -0.05095619 -0.05271388
```

```
km3.res$size
```

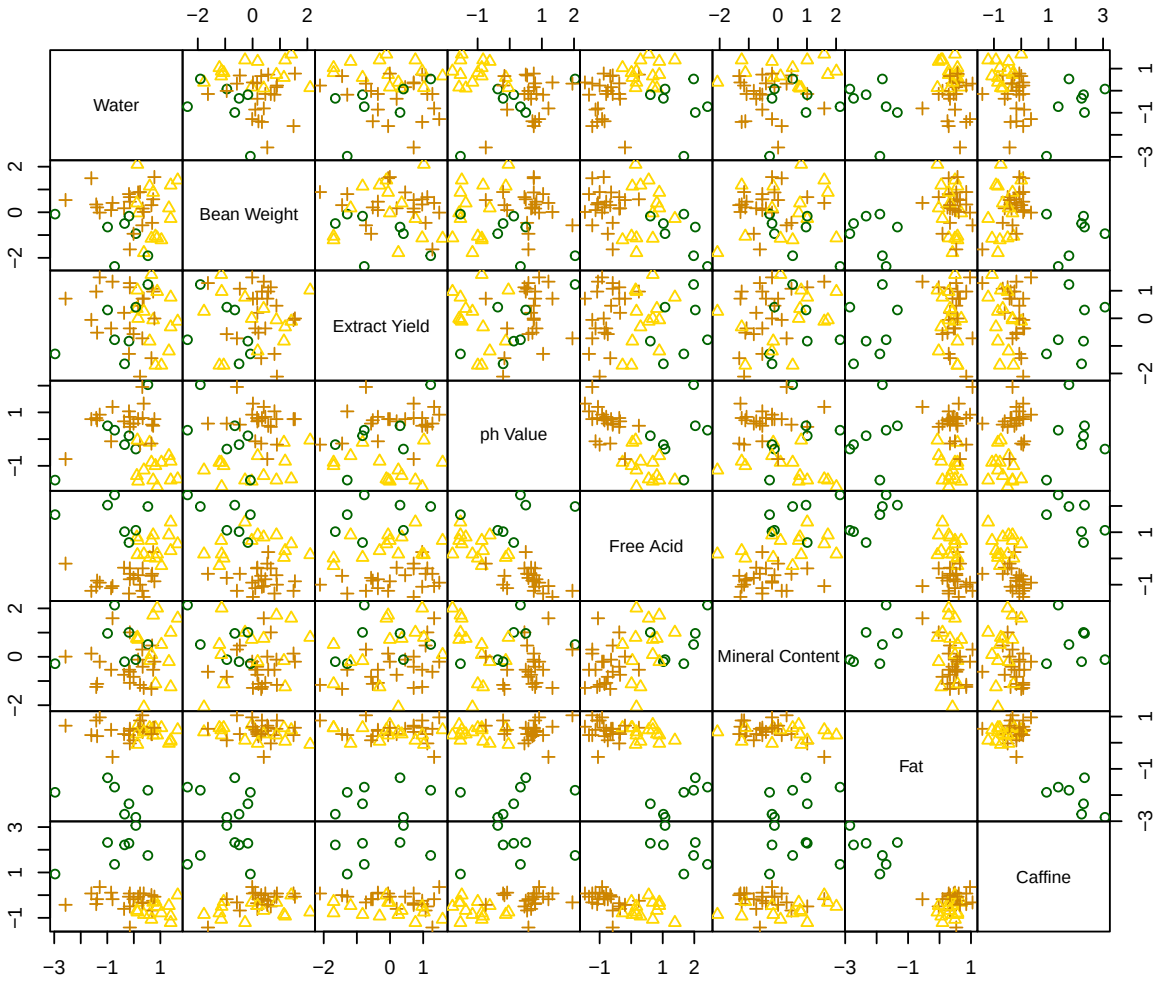
```
## [1]  7 14 22
```

I valori relativi al clustering possono essere ricondotti al dataframe iniziale.

```
df.c.kmeans <- cbind(df, cluster = km3.res$cluster)
head(df.c.kmeans[, c(1, 2, 4, 8, 9, 10, ncol(df.c.kmeans))], 14)
```

```
##        Variety  Country Bean Weight Mineral Content  Fat Caffine cluster
## mex1_a arabica   mexico       156.6            3.80 15.2    1.13       3
## mex2_a arabica   mexico       157.3            3.71 15.0    1.25       3
## gua1_a arabica guatemal       152.9            4.15 16.1    1.21       2
## hon1_a arabica honduras       174.0            3.94 15.8    1.06       2
## sal1_a arabica salvador       145.1            4.09 15.2    1.11       3
## sal2_a arabica salvador       156.4            3.88 15.4    1.20       3
## sal3_a arabica salvador       155.2            3.85 15.6    1.33       3
## nic1_a arabica nicaragu       167.8            3.85 15.1    1.28       3
## nac1_a arabica nacaragu       165.4            4.22 14.3    1.16       3
## cos1_a arabica costaric       180.3            4.01 15.1    1.32       3
## cos2_a arabica costaric       153.2            3.93 16.8    1.40       3
## cos3_a arabica costaric       159.6            3.68 16.5    1.19       3
## pan1_a arabica   panama       161.8            3.72 15.5    1.32       3
## hai1_a arabica    haiti       160.8            4.36 13.0    1.25       3
```

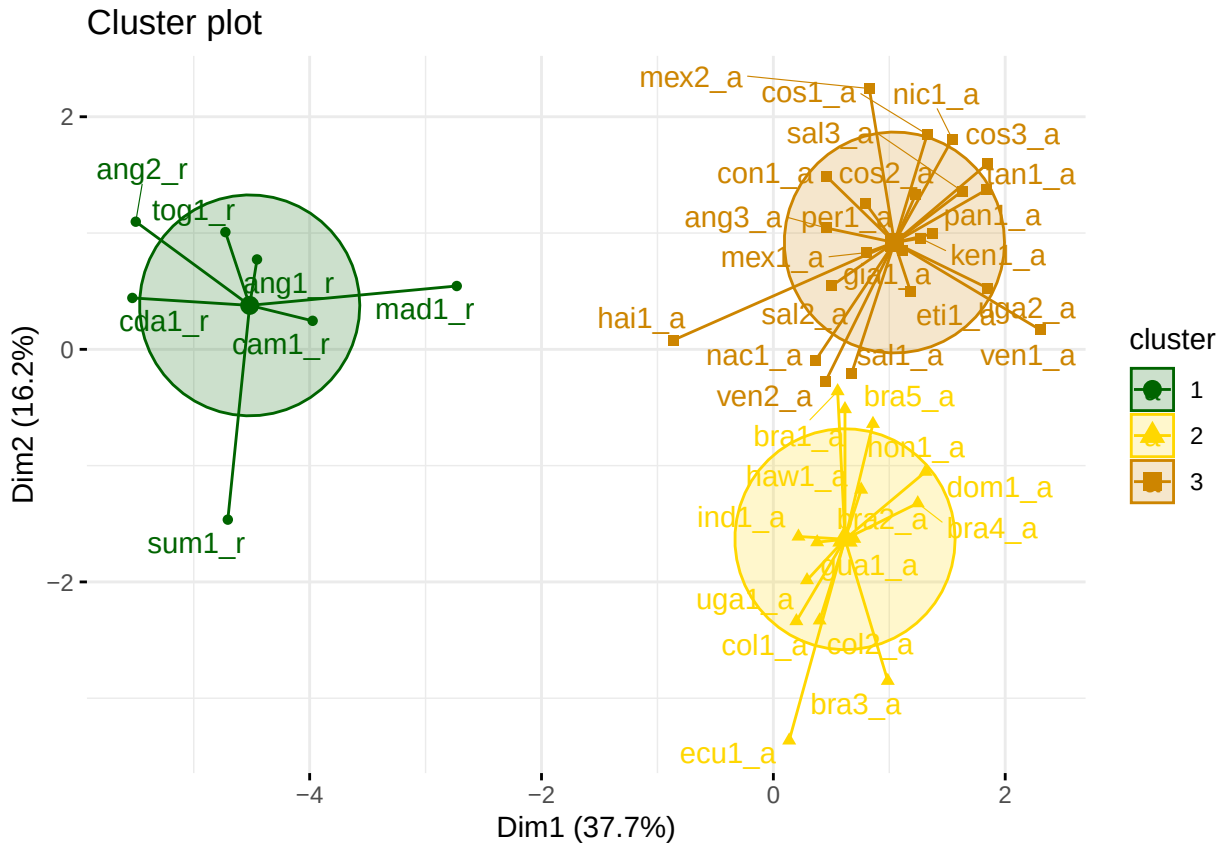
Il seguente grafico mostra uno scatterplot per ogni coppia di variabili del dataset. Per una migliore visualizzazione si considerano solo 8 variabili su 12.



Per alcune coppie di variabili la distribuzione dei cluster risulta marcata (nonostante la scarsa leggibilità del grafico), come ad esempio Fat, Mineral Content e Caffine, che sembrano essere discriminanti nella suddivisione dei cluster.

Mediante la funzione **fviz_cluster** è possibile avere una visualizzazione più chiara della suddivisione in cluster di ogni caffè. Tale funzione prende in input il risultato di k-means ed il dataframe originale delle numeriche, visto che al suo interno effettua la PCA quando il numero delle variabili è maggiore di 2.

```
fviz_cluster(km3.res, df.num, ellipse.type = "euclid", star.plot = TRUE, repel = TRUE,
             palette = colors6, ggtheme = theme_minimal()
)
```



Il grafico mostra una discreta separazione tra i cluster, con la prima componente principale che sembra essere più efficace nel distinguere i cluster rispetto alla seconda componente, distinzione evidente soprattutto tra il cluster 1 e gli altri due, che risultano invece sovrapposti.

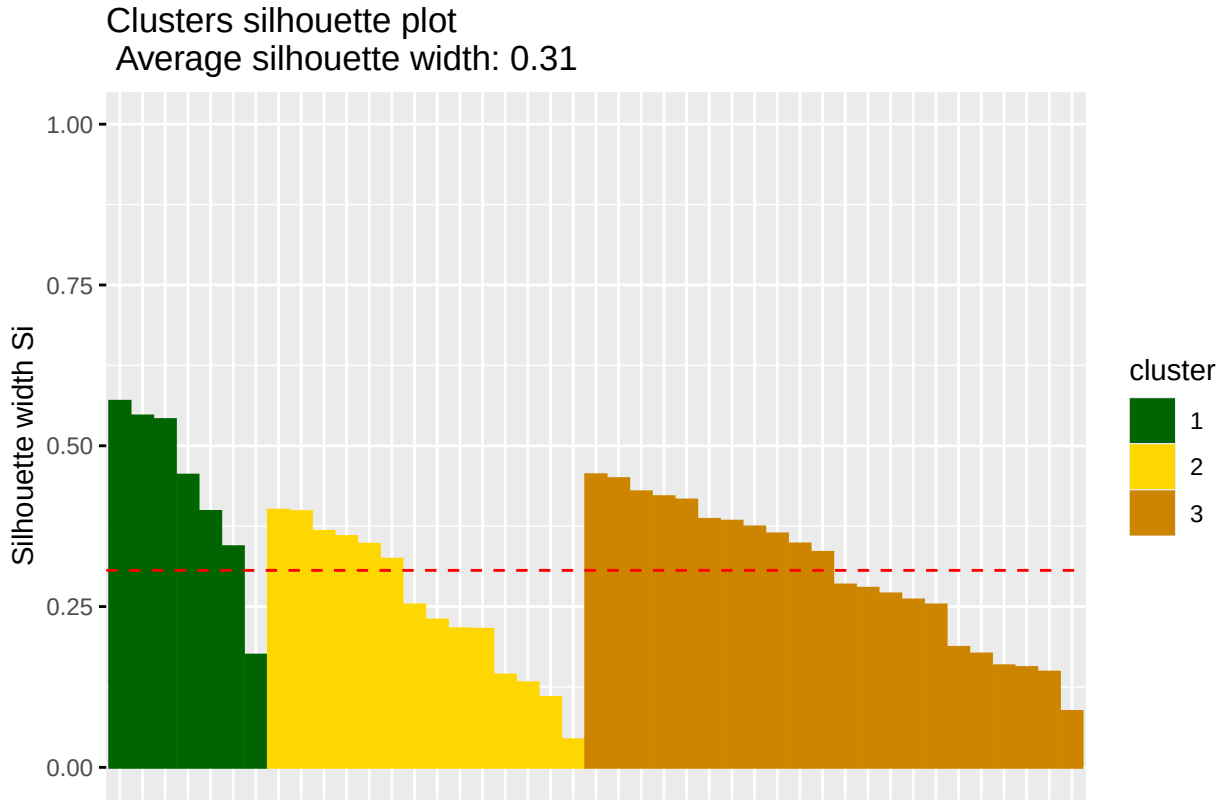
Si può notare dal nome del caffè che al primo cluster appartengono tutte e sole le varietà Robusta, indice del caffè che le caratteristiche che distinguono tale varietà sono determinanti ai fini della classificazione.

Di seguito viene richiamata la funzione **silhouette** di **cluster** che permette di calcolare l'*average silhouette width*, per misurare il grado di appartenenza di un punto al proprio cluster rispetto agli altri.

```
library(cluster)
sil3 <- silhouette(km3.res$cluster, dist(df.pca))
p.sil3 <- fviz_silhouette(sil3, palette = colors6)
```

```
##   cluster size ave.sil.width
## 1       1    7          0.43
## 2       2   14          0.25
## 3       3   22          0.30
```

La funzione **fviz_silhouette** permette di visualizzare tali score distinguendoli graficamente per cluster.



Anche da questo grafico si evince una migliore aderenza dei caffè del primo cluster.

Si vuole applicare ora K-means con k = 2.

```
set.seed(123)
km2.res <- kmeans(df.pca, 2, nstart = 50)
```

Di seguito i risultati relativi ai cluster:

```
km2.res$centers
```

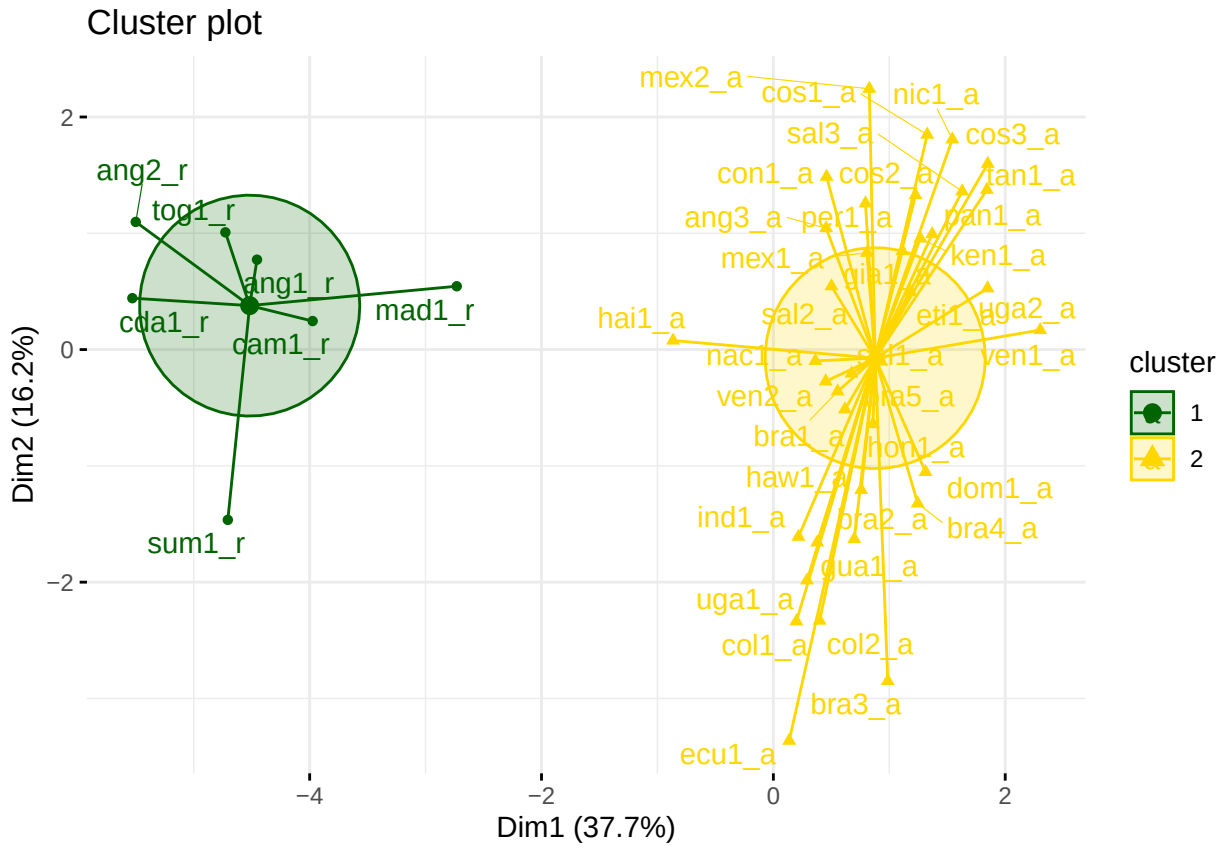
```
##        Dim.1      Dim.2      Dim.3        Dim.4       Dim.5
## 1  4.5710659 -0.3825350 -0.2712533  0.010782690 -0.19208277
## 2 -0.8888184  0.0743818  0.0527437 -0.002096634  0.03734943
```

```
km2.res$size
```

```
## [1]  7 36
```

Nuovamente, mostriamo i cluster mediante **fviz_cluster**.

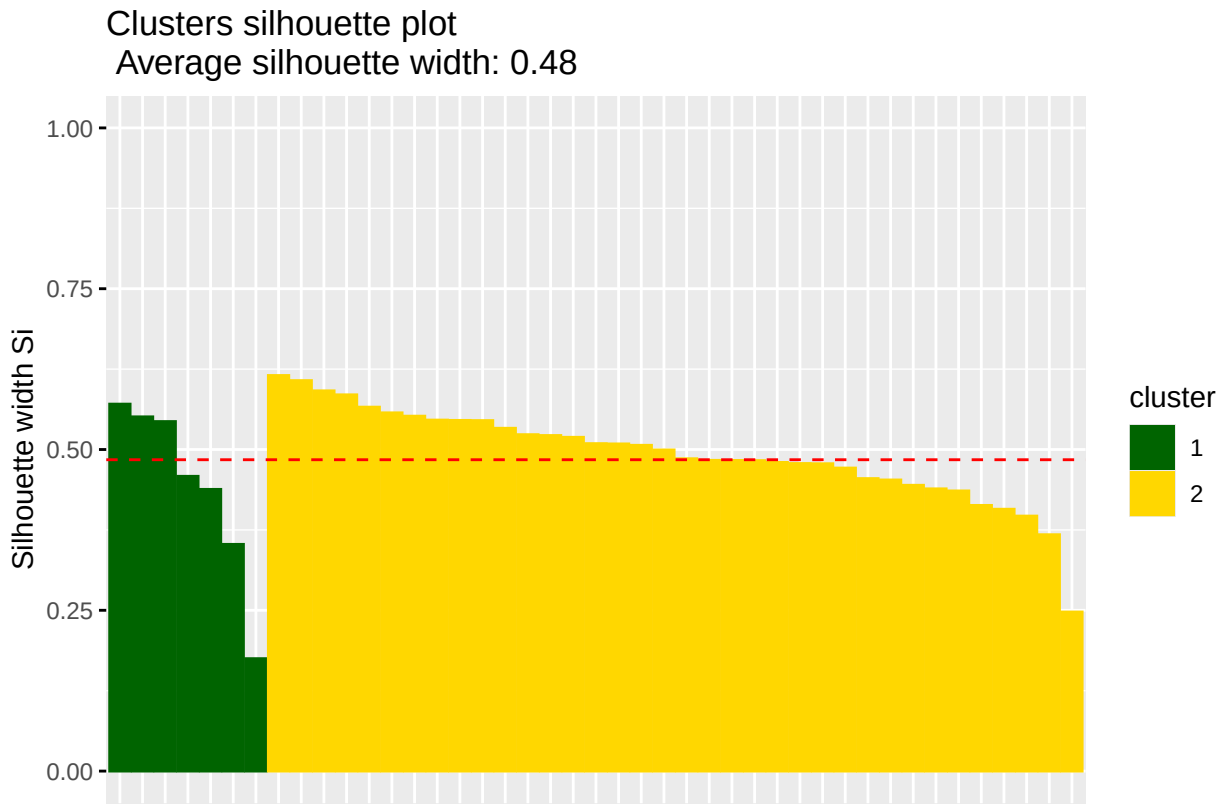
```
fviz_cluster(km2.res, df.num, ellipse.type = "euclid", star.plot = TRUE, repel = TRUE,
             palette = colors6, ggtheme = theme_minimal()
)
```



Con 2 cluster la separazione è netta, soprattutto in riferimento alla prima componente principale. I due cluster, come si evince dall'identificativo dei caffè, coincidono con le due varietà.

```
sil2 <- silhouette(km2.res$cluster, dist(df.pca))
p.sil2 <- fviz_silhouette(sil2, palette=colors6)
```

```
##   cluster size ave.sil.width
## 1       1    7          0.44
## 2       2   36          0.49
```



Rispetto al precedente grafico, con 2 cluster si registra una *average silhouette width* più alta.

Procediamo con k = 4.

```
set.seed(123)
km4.res <- kmeans(df.pca, 4, nstart = 50)
```

Di seguito i risultati relativi ai cluster:

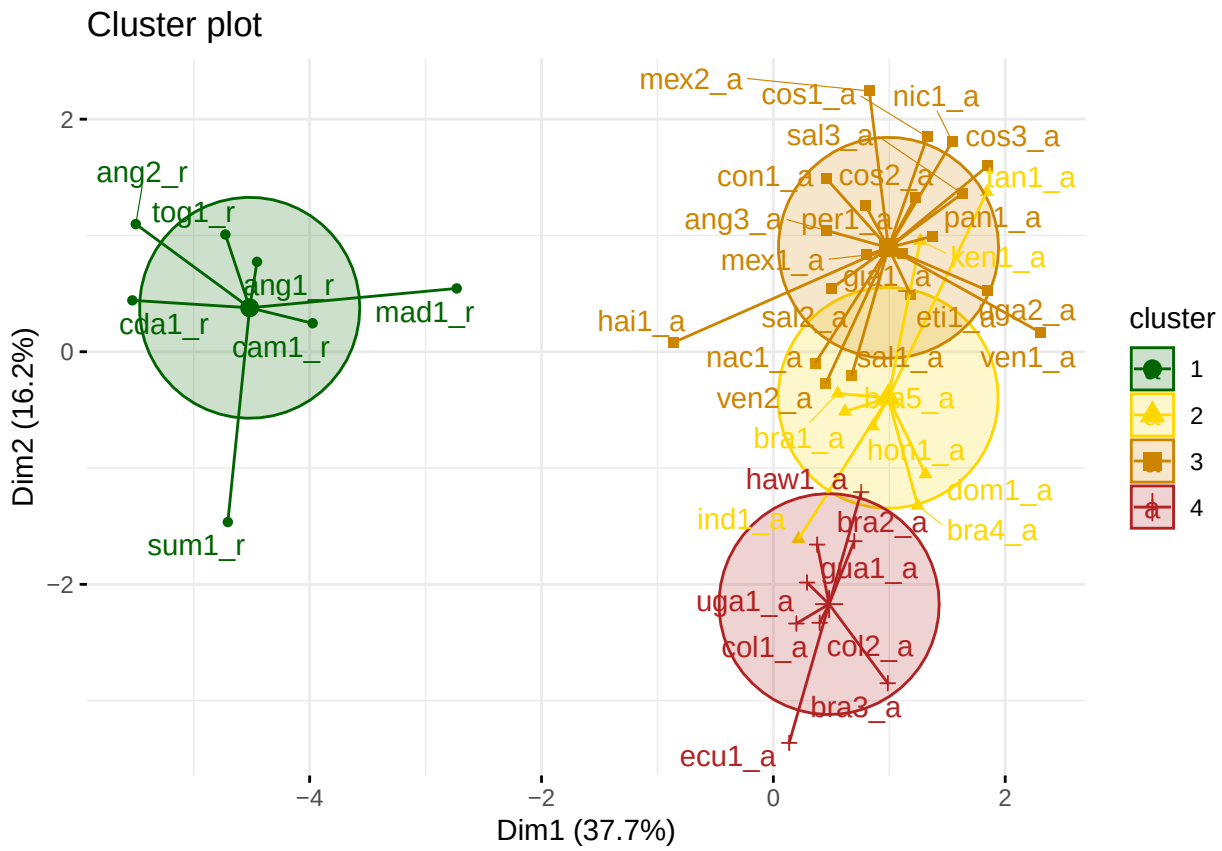
```
km4.res$centers
```

```
##        Dim.1      Dim.2      Dim.3       Dim.4      Dim.5
## 1  4.5710659 -0.3825350 -0.2712533  0.01078269 -0.1920828
## 2 -1.0013382  0.4008409 -1.4759459 -0.21441979  0.7622009
## 3 -1.0047793 -0.9046536  0.5879964 -0.10523604 -0.1408332
## 4 -0.4863962  2.1955112  0.2433015  0.46807505 -0.2420455
```

```
km4.res$size
```

```
## [1]  7  8 20  8
```

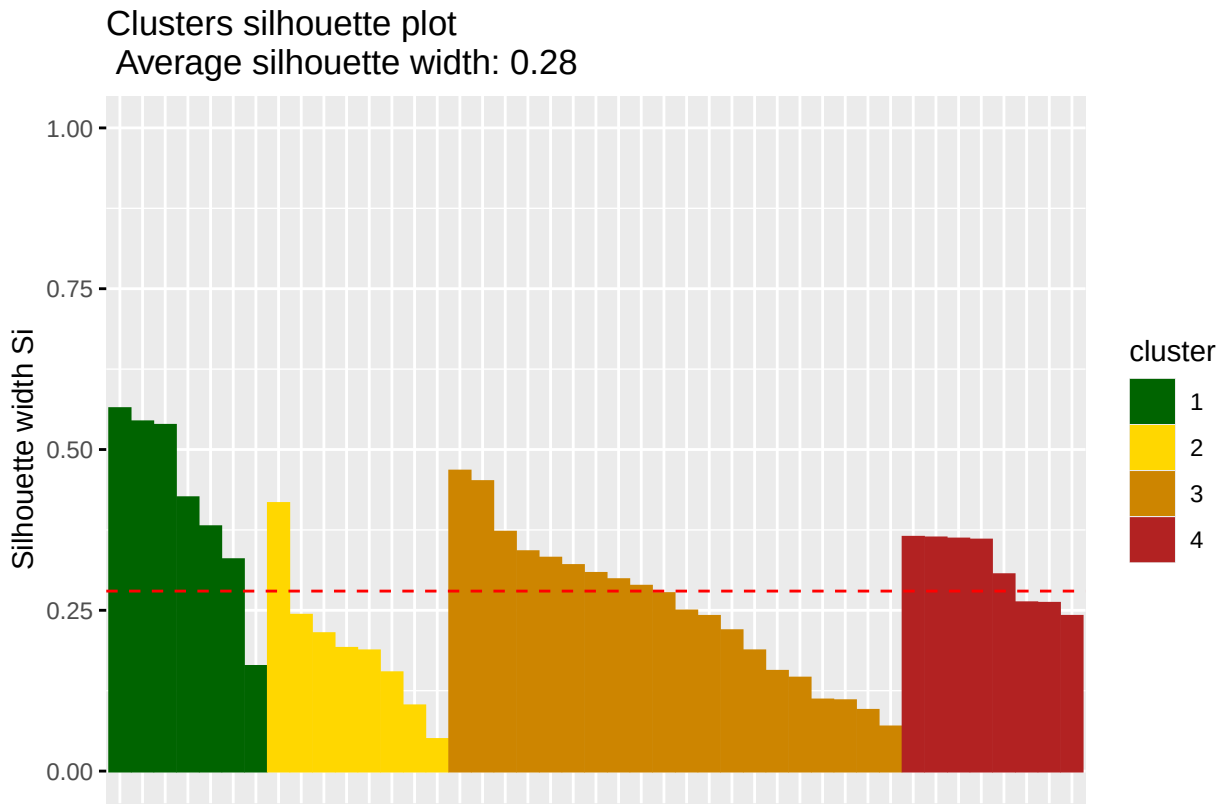
```
fviz_cluster(km4.res, df.num, ellipse.type = "euclid",star.plot = TRUE,repel = TRUE,
             palette = colors6, ggtheme = theme_minimal()
)
```



Anche con 4 cluster il primo risulta ben distinto dagli altri tre, sui quali si evidenzia maggiore complessità e sovrapposizione.

```
sil4 <- silhouette(km4.res$cluster, dist(df.pca))
p.sil4 <- fviz_silhouette(sil4, palette=colors6)
```

```
##   cluster size ave.sil.width
## 1       1    7          0.42
## 2       2    8          0.19
## 3       3   20          0.25
## 4       4    8          0.31
```



Rispetto ai precedenti, come atteso, con 4 cluster si registra una *average silhouette width* leggermente più bassa rispetto a K = 3.

K-Medoids

L'algoritmo k-medoids è un approccio di clustering che suddivide il dataset in k cluster. Ciascun cluster, in questo caso, è rappresentato da uno degli elementi appartenenti al cluster, appunto il medoide, per il quale la dissimilarità media rispetto a tutti gli altri item del cluster è minima. Esso corrisponde al punto più centralmente localizzato nel cluster.

Si esegue la funzione **pam** (Partitioning Around Medoids) di **cluster**, sia per K = 2 che per k = 3.

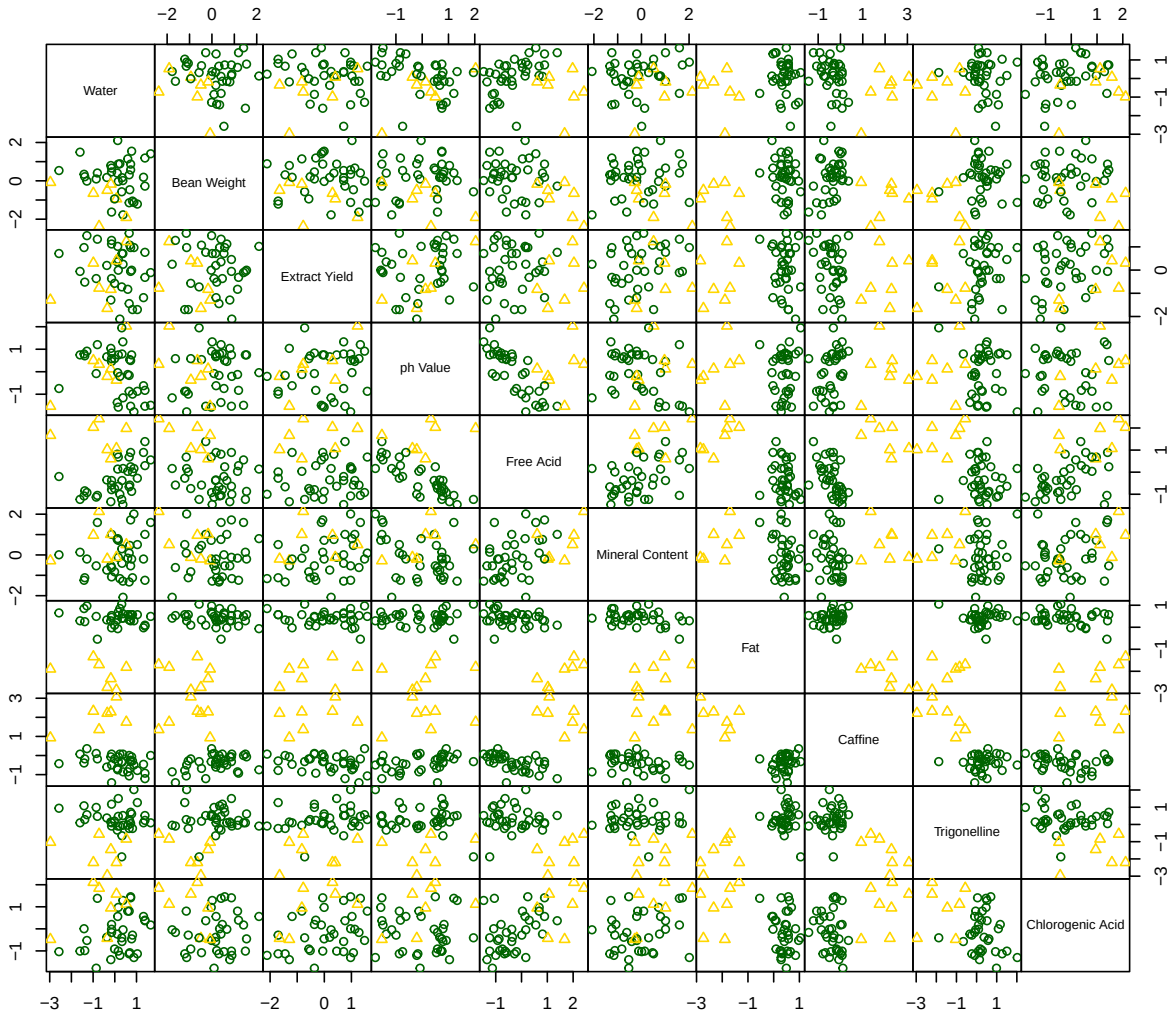
```
pam2.res <- pam(df.num, 2)
```

```
##        Variety  Country Bean Weight Mineral Content  Fat Caffine cluster
## mex1_a arabica   mexico       156.6            3.80 15.2    1.13       1
## mex2_a arabica   mexico       157.3            3.71 15.0    1.25       1
## gua1_a arabica guatemal       152.9            4.15 16.1    1.21       1
## hon1_a arabica honduras       174.0            3.94 15.8    1.06       1
## sal1_a arabica salvador       145.1            4.09 15.2    1.11       1
## sal2_a arabica salvador       156.4            3.88 15.4    1.20       1
## sal3_a arabica salvador       155.2            3.85 15.6    1.33       1
## nic1_a arabica nicaragu       167.8            3.85 15.1    1.28       1
## nac1_a arabica nacaragu       165.4            4.22 14.3    1.16       1
## cos1_a arabica costaric       180.3            4.01 15.1    1.32       1
```

Di seguito sono riportati i medoidi.

```
##             Water Bean Weight Extract Yield  ph Value  Free Acid
## mex1_a -0.3674808   0.1748259     0.1995409 0.6647434 -0.3526718
## ang1_r -0.1825883  -0.1704229    -0.8280348 0.1254233  0.6064758
##        Mineral Content        Fat    Caffine Trigonelline Chlorogenic Acid
## mex1_a      -0.7434088  0.3322506 -0.6048584    0.3771739       -0.3812287
## ang1_r       1.0093279 -2.3340837  2.2850204   -1.4562192        0.9485734
##        Neochlorogenic Acid Isochlorogenic Acid
## mex1_a            0.029026           0.1857075
## ang1_r            1.277144           2.3399150
```

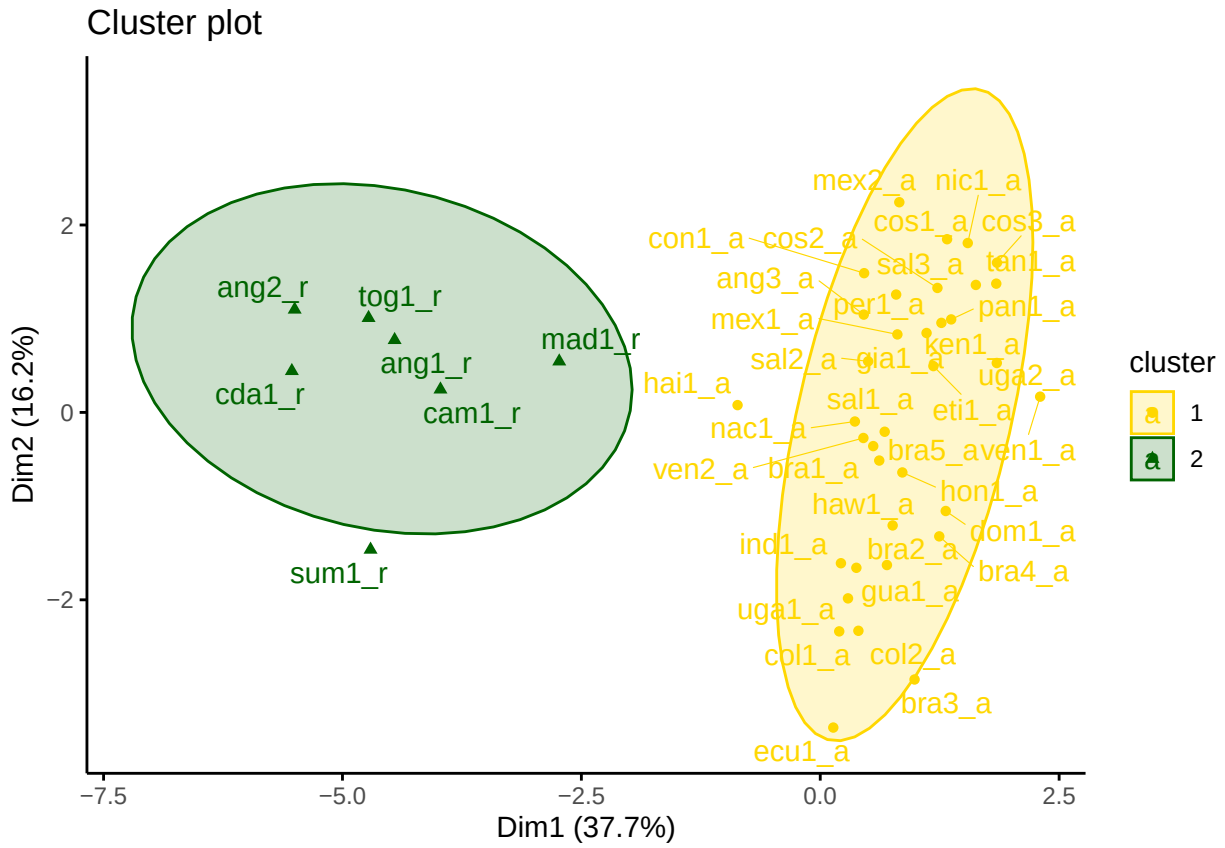
Analogamente a quanto fatto in precedenza, si riporta la correlazione di alcune variabili rispetto ai cluster determinati.



Anche in questo caso si nota una maggiore distinzione dei cluster per alcune variabili, come caffeina, grasso e trigonellina, rispetto ad altre per cui la distribuzione nei cluster è meno netta, come l'acido clogenico e l'estratto.

La funzione **fviz_cluster** permette di visualizzare la distinzione in cluster.

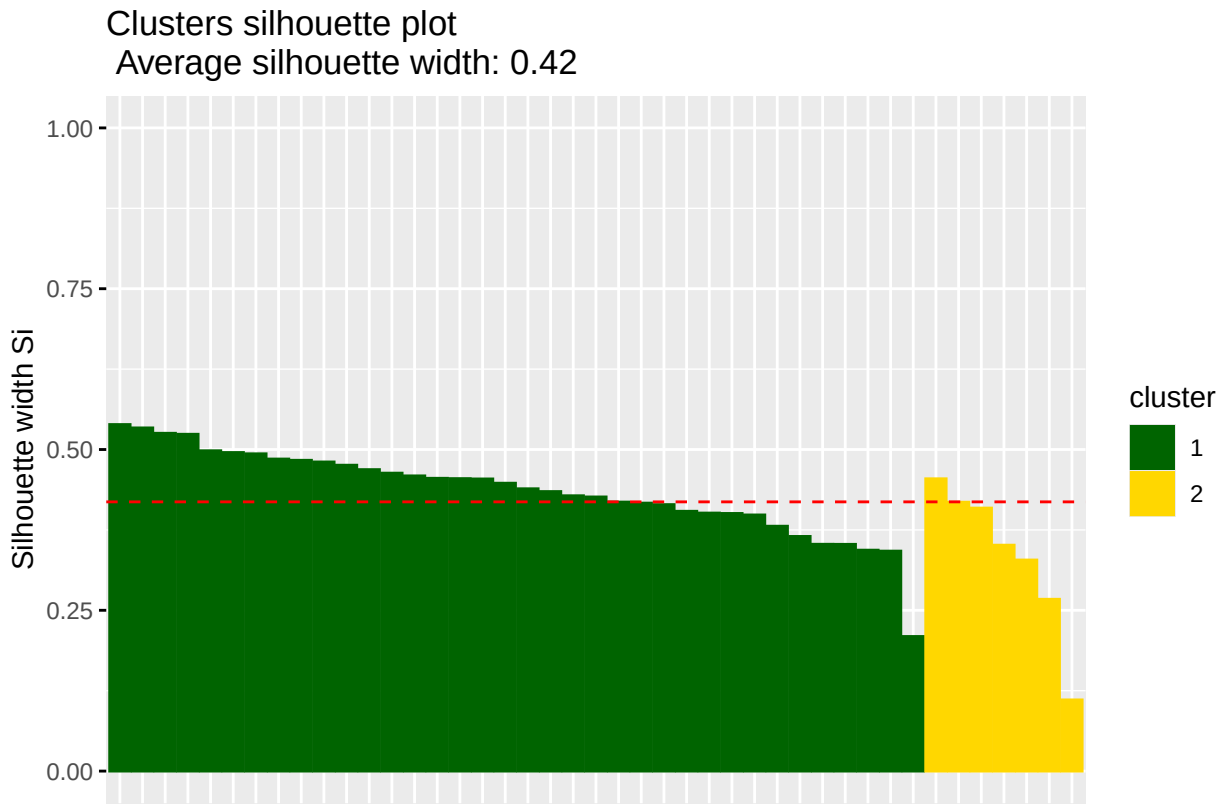
```
fviz_cluster(pam2.res, palette = colors2, ellipse.type = "t", repel = TRUE,
             ggtheme = theme_classic()
)
```



Il grafico evidenzia una netta separazione tra i cluster che corrispondono esattamente alla distinzione dei caffè per varietà: cluster 1 contiene tutti i caffè di Arabica, cluster 2 di Robusta.

```
sil.2 <- silhouette(pam2.res$cluster, dist(df.num))
p.sil.2 <- fviz_silhouette(sil.2, palette=colors6)
```

```
##   cluster size ave.sil.width
## 1       1   36          0.44
## 2       2    7          0.33
```



Come da aspettative risultato simile al Kmeans per K = 2 ma leggermente più basso.

Si applica lo stesso procedimento per k = 3.

```
pam3.res <- pam(df.num, 3)
```

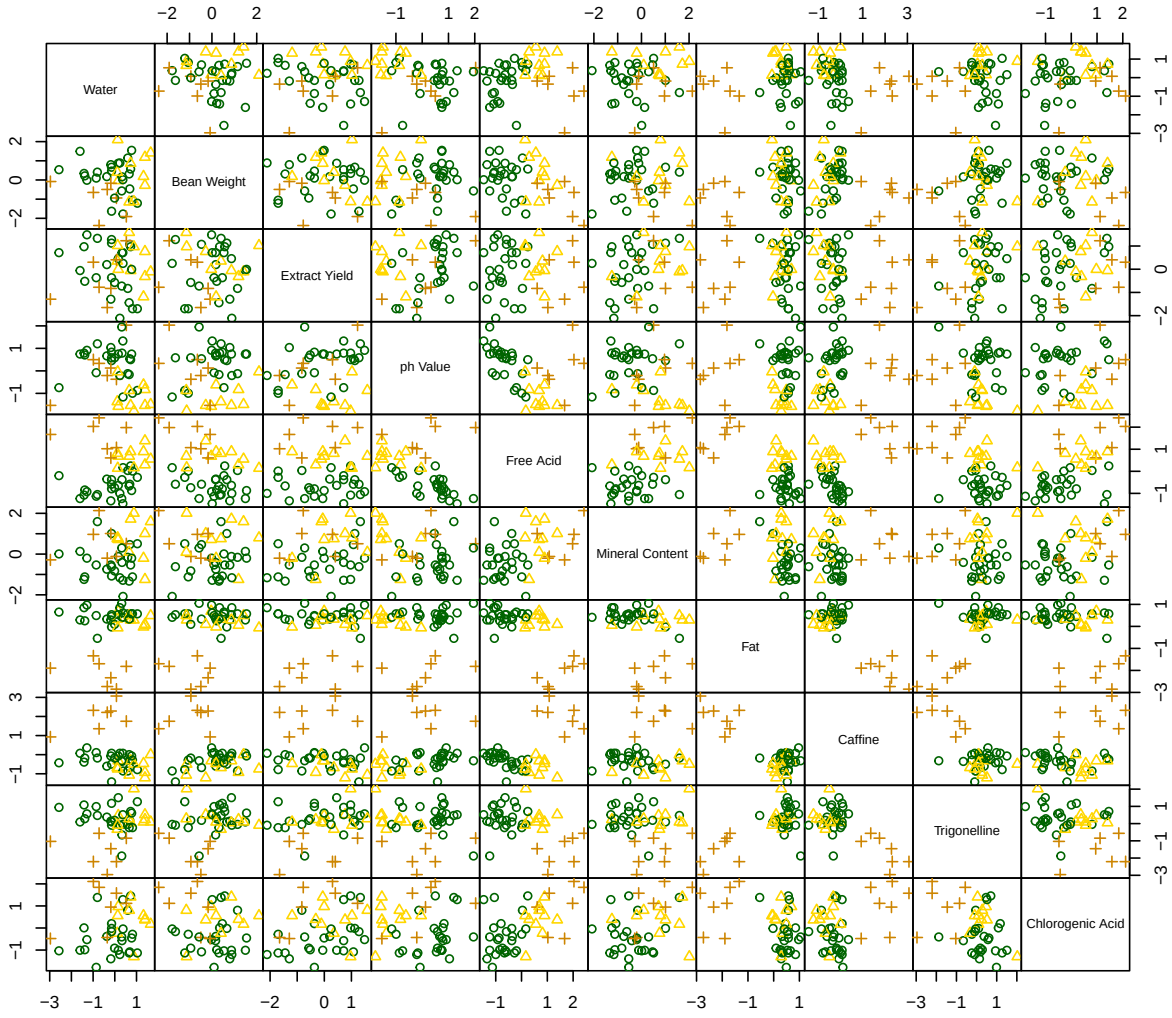
```
##        Variety  Country Bean Weight Mineral Content  Fat Caffine cluster
## mex1_a arabica   mexico       156.6            3.80 15.2    1.13       1
## mex2_a arabica   mexico       157.3            3.71 15.0    1.25       1
## gua1_a arabica guatemal       152.9            4.15 16.1    1.21       2
## hon1_a arabica honduras       174.0            3.94 15.8    1.06       1
## sal1_a arabica salvador       145.1            4.09 15.2    1.11       1
## sal2_a arabica salvador       156.4            3.88 15.4    1.20       1
## sal3_a arabica salvador       155.2            3.85 15.6    1.33       1
## nic1_a arabica nicaragu       167.8            3.85 15.1    1.28       1
## nac1_a arabica nacaragu       165.4            4.22 14.3    1.16       1
## cos1_a arabica costaric       180.3            4.01 15.1    1.32       1
```

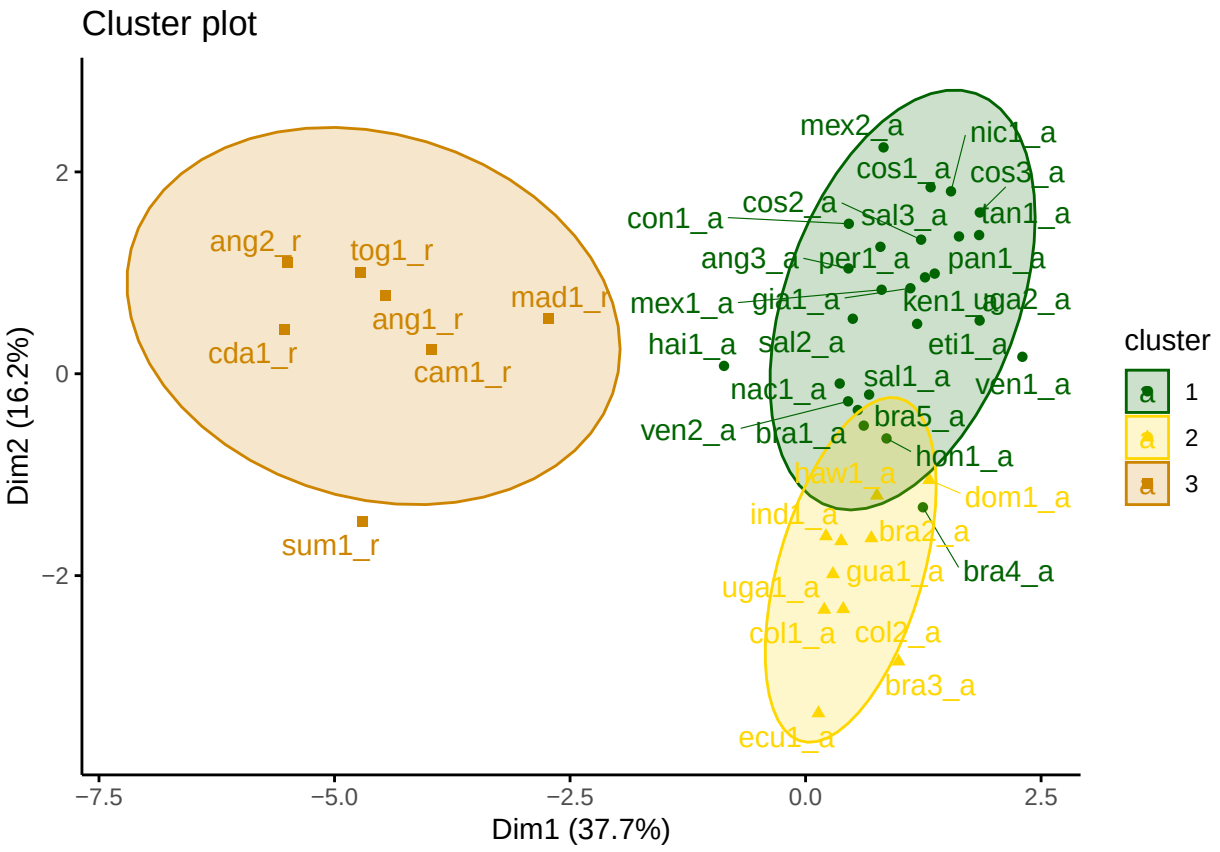
Di seguito sono riportati i medoidi.

```
##             Water Bean Weight Extract Yield   ph Value  Free Acid
## mex1_a -0.3674808   0.1748259     0.1995409  0.6647434 -0.3526718
## bra2_a  0.1145592   0.4031346     0.3536768 -1.3680785  0.8138587
## ang1_r -0.1825883  -0.1704229    -0.8280348  0.1254233  0.6064758
##        Mineral Content        Fat    Caffine Trigonelline Chlorogenic Acid
## mex1_a      -0.7434088  0.3322506 -0.6048584    0.3771739       -0.3812287
```

```
## bra2_a       0.7589370  0.3322506 -0.6762134   -0.3279773       -0.2361594
## ang1_r       1.0093279 -2.3340837  2.2850204   -1.4562192        0.9485734
##        Neochlorogenic Acid Isochlorogenic Acid
## mex1_a           0.0290260           0.1857075
## bra2_a          -0.8290552          -0.9656792
## ang1_r           1.2771441           2.3399150
```

Analogamente a quanto fatto in precedenza, si riporta la correlazione di alcune variabili rispetto ai cluster determinati e la distizione in cluster.

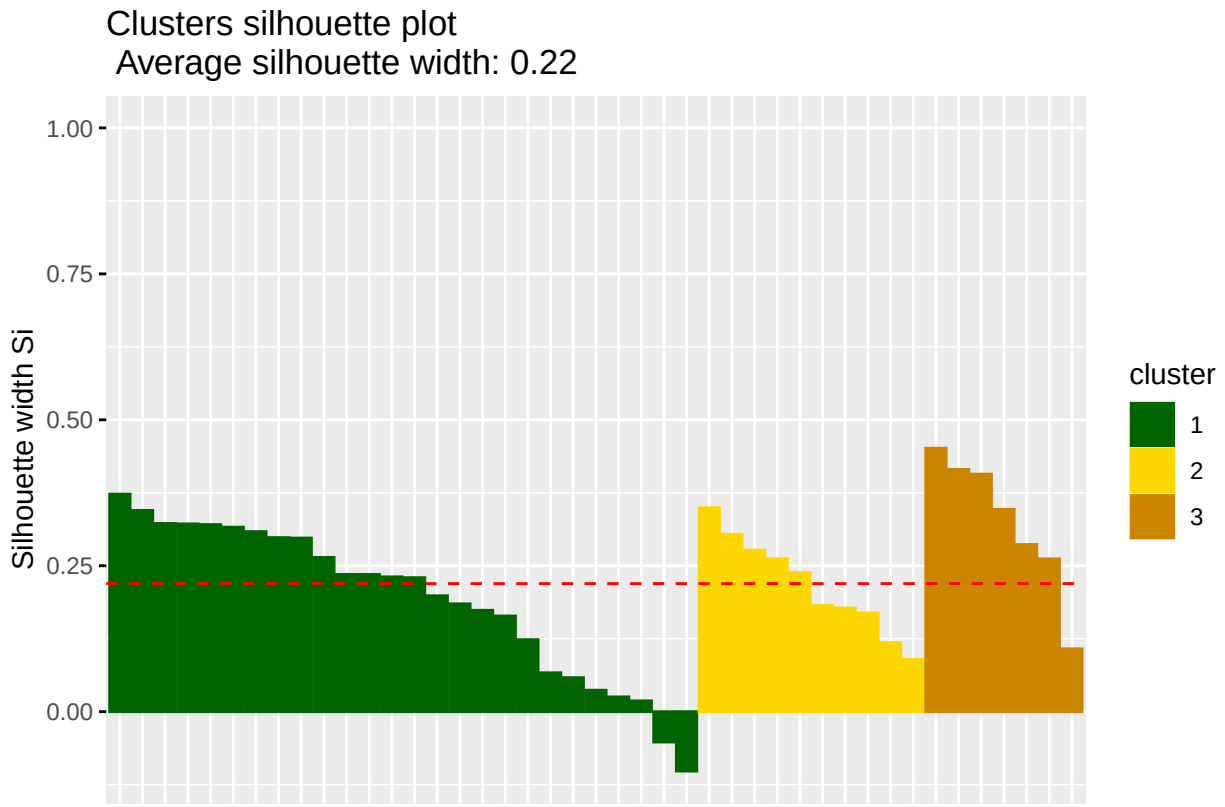




Nonostante ci sia una discreta separazione in cluster, in questo caso i caffè di varietà Arabica sono leggermente soprapposti, denotando una certa complessità.

```
sil.3 <- silhouette(pam3.res$cluster, dist(df.num))
p.sil.3 <- fviz_silhouette(sil.3, palette=colors6)
```

```
##   cluster size ave.sil.width
## 1       1   26          0.19
## 2       2   10          0.22
## 3       3    7          0.33
```



L'Average Silhouette Width pari a 0.22, inferiore al precedente, conferma che per k-medoids il numero ideale di cluster è pari a 2.

Clustering gerarchico

Il clustering gerarchico è un metodo di partizionamento dei dati che si basa sulla similarità tra gli oggetti. A differenza del clustering partizionale, non richiede di specificare in anticipo il numero di cluster in cui suddividere il dataset. Esistono due approcci principali:

- Clustering **agglomerativo**: ogni osservazione inizia come un cluster singolo (foglia) e, attraverso un processo iterativo, i cluster più simili vengono combinati tra loro, fino a formare un unico cluster che rappresenta l'intero dataset (radice).
- Clustering **divisivo**: l'opposto del metodo agglomerativo; si parte da un unico cluster contenente tutte le osservazioni (radice) e, progressivamente, si suddivide in cluster più piccoli sulla base delle dissimilarità, fino a quando ogni osservazione appartiene al proprio cluster individuale.

In questa sezione sarà adottato il metodo agglomerativo, il più comunemente usato tra i due principali approcci, poiché meno complesso dal punto di vista computazionale.

Clustering agglomerativo

L'algoritmo consiste nei seguenti step, a valle della preparazione dei dati che consideriamo già effettuata nelle sezioni precedenti:

- si calcolano le informazioni di (dis)similarità tra ogni coppia di item del dataset;
- si usa una funzione *linkage* per raggruppare gli item in un albero gerarchico di cluster, basandosi sulle informazioni di distanza di cui al punto precedente;
- si determina dove *tagliare* l'albero gerarchico per ottenere i cluster al fine di creare la partizione finale dei dati.

Come visto in una delle precedenti sezioni, la funzione **dist** di **stats** permette di determinare le distanze tra le osservazioni. Di seguito viene applicata sia la distanza euclidea che la distanza di Manhattan sul dataframe delle componenti principali.

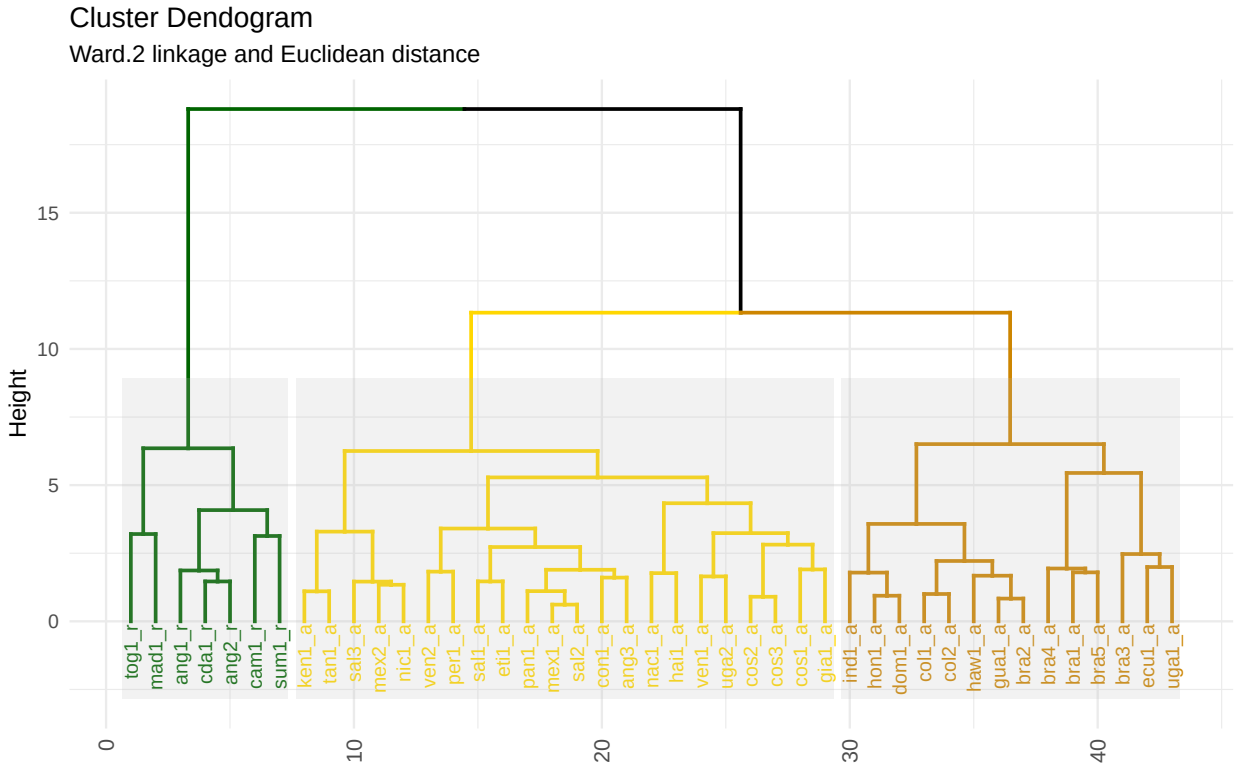
Distanza euclidea su PCA:

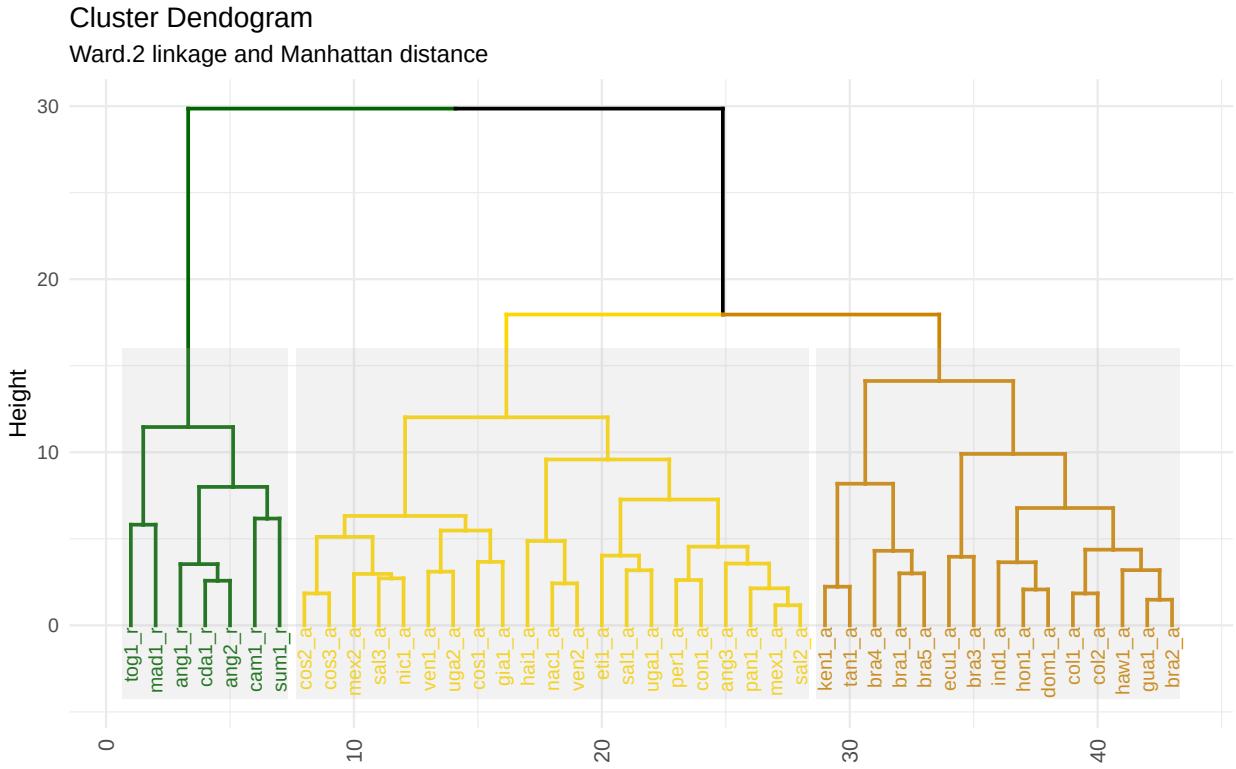
```
##        mex1_a mex2_a gua1_a hon1_a sal1_a sal2_a sal3_a nic1_a nac1_a
## mex1_a   0.00   1.63   2.81   2.36   1.41   0.61   1.75   1.97   1.86
## mex2_a   1.63   0.00   4.14   3.41   2.90   2.10   1.41   1.34   3.04
## gua1_a   2.81   4.14   0.00   1.49   2.57   2.55   3.78   3.90   2.13
## hon1_a   2.36   3.41   1.49   0.00   2.78   2.13   3.21   2.83   2.45
## sal1_a   1.41   2.90   2.57   2.78   0.00   1.33   2.61   3.24   1.95
## sal2_a   0.61   2.10   2.55   2.13   1.33   0.00   2.32   2.35   1.78
## sal3_a   1.75   1.41   3.78   3.21   2.61   2.32   0.00   1.45   3.07
## nic1_a   1.97   1.34   3.90   2.83   3.24   2.35   1.45   0.00   3.38
## nac1_a   1.86   3.04   2.13   2.45   1.95   1.78   3.07   3.38   0.00
```

Distanza di Manhattan su PCA:

```
##        mex1_a mex2_a gua1_a hon1_a sal1_a sal2_a sal3_a nic1_a nac1_a
## mex1_a   0.00   2.78   4.46   4.60   2.64   1.16   3.48   3.90   3.88
## mex2_a   2.78   0.00   6.35   5.60   4.94   3.94   2.88   2.94   5.29
## gua1_a   4.46   6.35   0.00   3.07   4.73   4.02   7.45   6.36   3.66
## hon1_a   4.60   5.60   3.07   0.00   5.14   4.31   6.28   5.19   4.67
## sal1_a   2.64   4.94   4.73   5.14   0.00   2.69   5.65   6.54   3.08
## sal2_a   1.16   3.94   4.02   4.31   2.69   0.00   4.64   4.91   3.64
## sal3_a   3.48   2.88   7.45   6.28   5.65   4.64   0.00   2.72   6.39
## nic1_a   3.90   2.94   6.36   5.19   6.54   4.91   2.72   0.00   6.25
## nac1_a   3.88   5.29   3.66   4.67   3.08   3.64   6.39   6.25   0.00
```

I risultati dei due calcoli vengono di seguito utilizzati come input per la funzione **hclust** di **stats**, che serve a determinare un albero gerarchico di cluster.





Un modo per valutare quanto il dendogramma generato rappresenti bene i dati, consiste nel calcolare la correlazione tra le distanze cophenetiche e i dati originali sulle distanze calcolate. La distanza cophenetica tra due oggetti è data dall'altezza del ramo comune più basso nel dendrogramma che li unisce. Più il valore del coefficiente di correlazione si avvicina a 1, più la soluzione di clustering riflette accuratamente i dati.

Di seguito si utilizza la funzione R **cophenetic** per distanze precedentemente calcolate, per i rispettivi dendogrammi.

```
# Ward.2 evaluation with Euclidean
res.coph.eu <- cophenetic(res.hc.eu.1)
cor(res.dist.eu, res.coph.eu)
```

```
## [1] 0.8645509
```

```
# Ward.2 evaluation with Manhattan
res.coph.ma <- cophenetic(res.hc.ma.1)
cor(res.dist.ma, res.coph.ma)
```

```
## [1] 0.8027389
```

Si evince che il dendogramma basato sulla distanza euclidea riflette meglio la partizione dei dati rispetto a quello basato sulla distanza di Manhattan. Pertanto d'ora in avanti si considererà come distanza di riferimento.

La funzione **cutree** di **stats** permette di tagliare l'albero in un numero di gruppi specificato. Nel caso in esame si considera un numero di cluster pari a 2 ed a 3, numeri esaminati nelle precedenti sezioni.

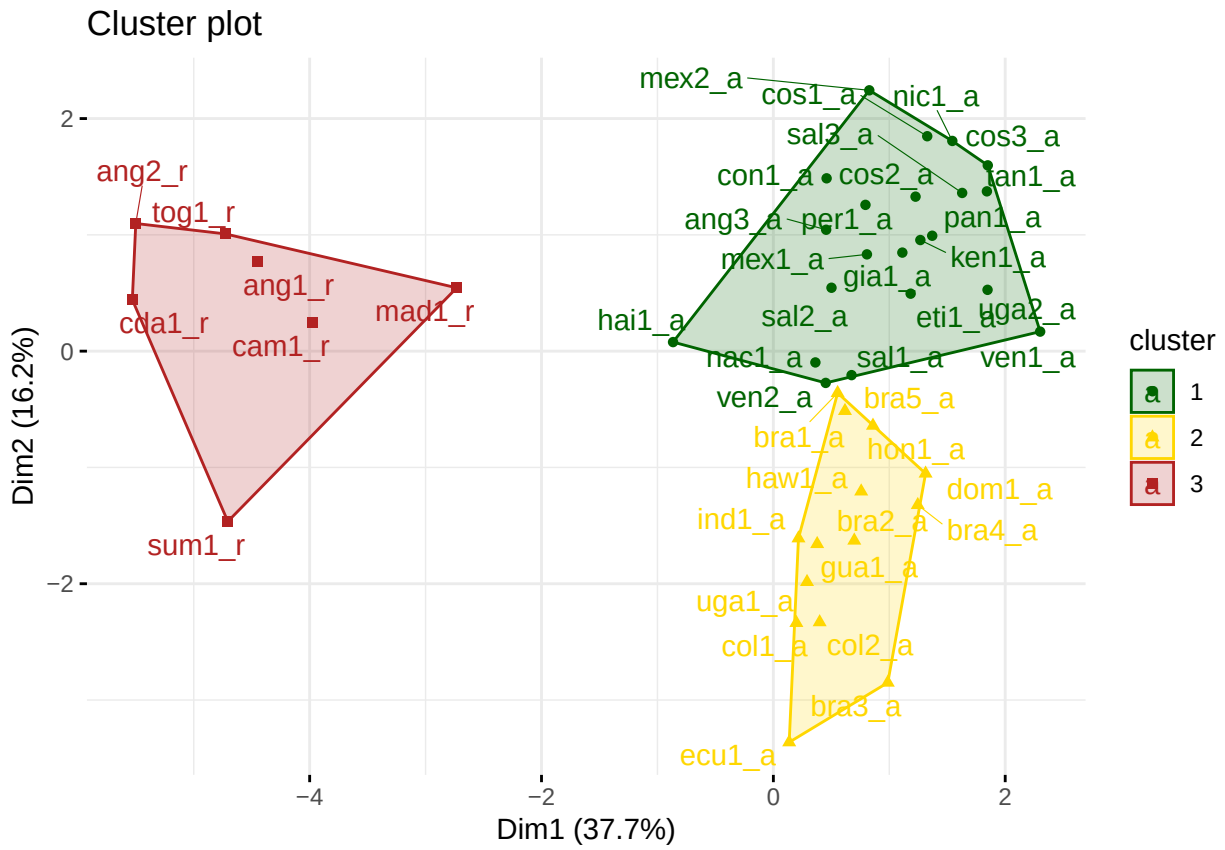
```
grp23 <- cutree(res.hc.eu.1, k = c(2,3))
table(grp2 = grp23[,"2"], grp3 = grp23[,"3"])
```

```
##     grp3
## grp2  1  2  3
##    1 22 14  0
##    2  0  0  7
```

Dalla tabella si nota che la funzione taglia il dendogramma, in entrambi i casi, all'altezza dei caffè di varietà Robusta, lasciando in un caso tutti quelli con varietà Arabica nell'altro cluster (7|36) o, in caso di k = 3, in due cluster ben evidenziati da entrambi i dendogrammi (7|22|14).

Di seguito il grafico ottenuto con fviz_cluster, considerando il clustering con K=3.

```
grp1 <- cutree(res.hc.eu.1, k = 3)
fviz_cluster(list(data = df.num, cluster = grp1),
             palette = colors3, ellipse.type = "convex", repel = TRUE,
             show.clust.cent = FALSE, ggtheme = theme_minimal())
```



I cluster risultano ben distinti, soprattutto per la prima componente principale.

Di seguito sono valutati anche altri metodi di *linkage*, come fatto per Ward.2, in particolare i metodi **complete**, **average** e **centroid**.

```
res.hc2 <- hclust(res.dist.eu, method = "complete")
cor(res.dist.eu, cophenetic(res.hc2))
```

```
## [1] 0.8274959
```

```
res.hc3 <- hclust(res.dist.eu, method = "average")
cor(res.dist.eu, cophenetic(res.hc3))
```

```
## [1] 0.8920207
```

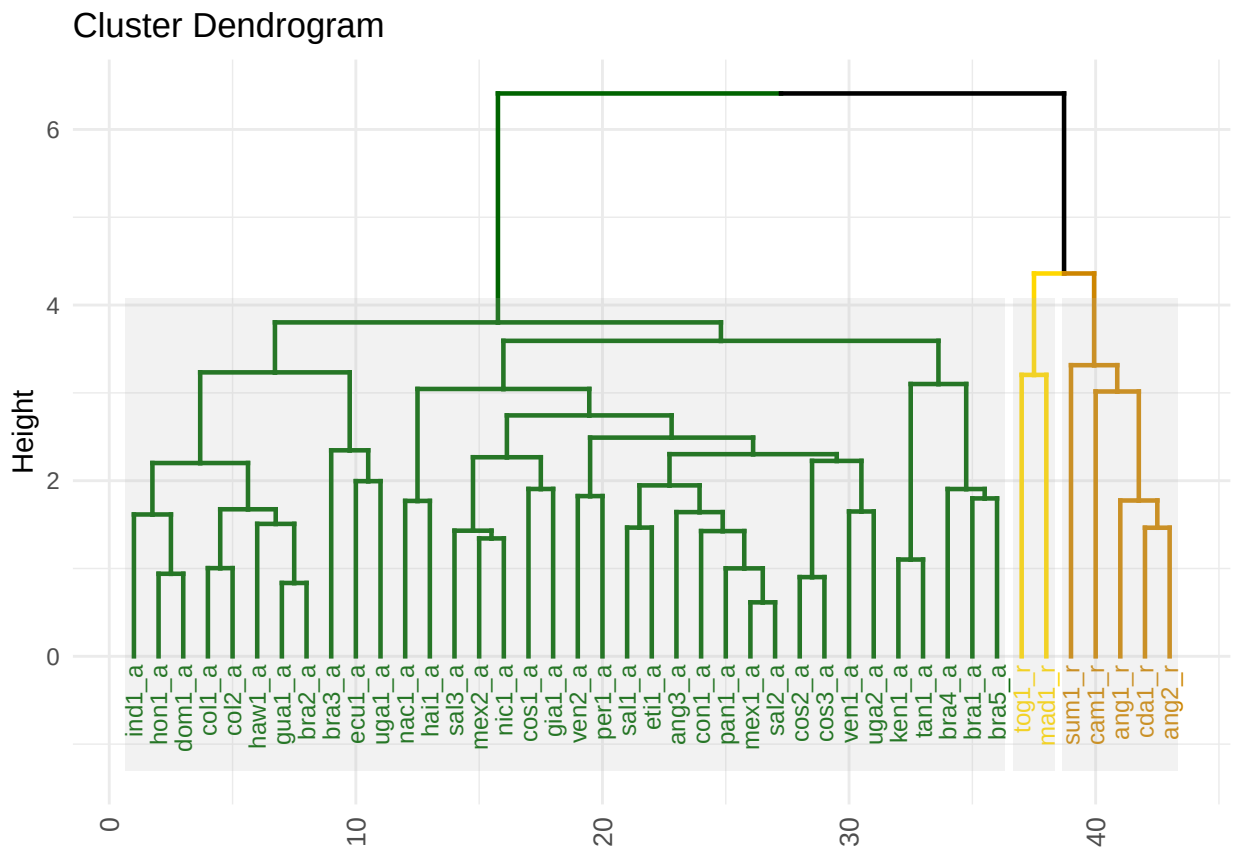
```
res.hc4 <- hclust(res.dist.eu, method = "centroid")
cor(res.dist.eu, cophenetic(res.hc4))
```

```
## [1] 0.8540007
```

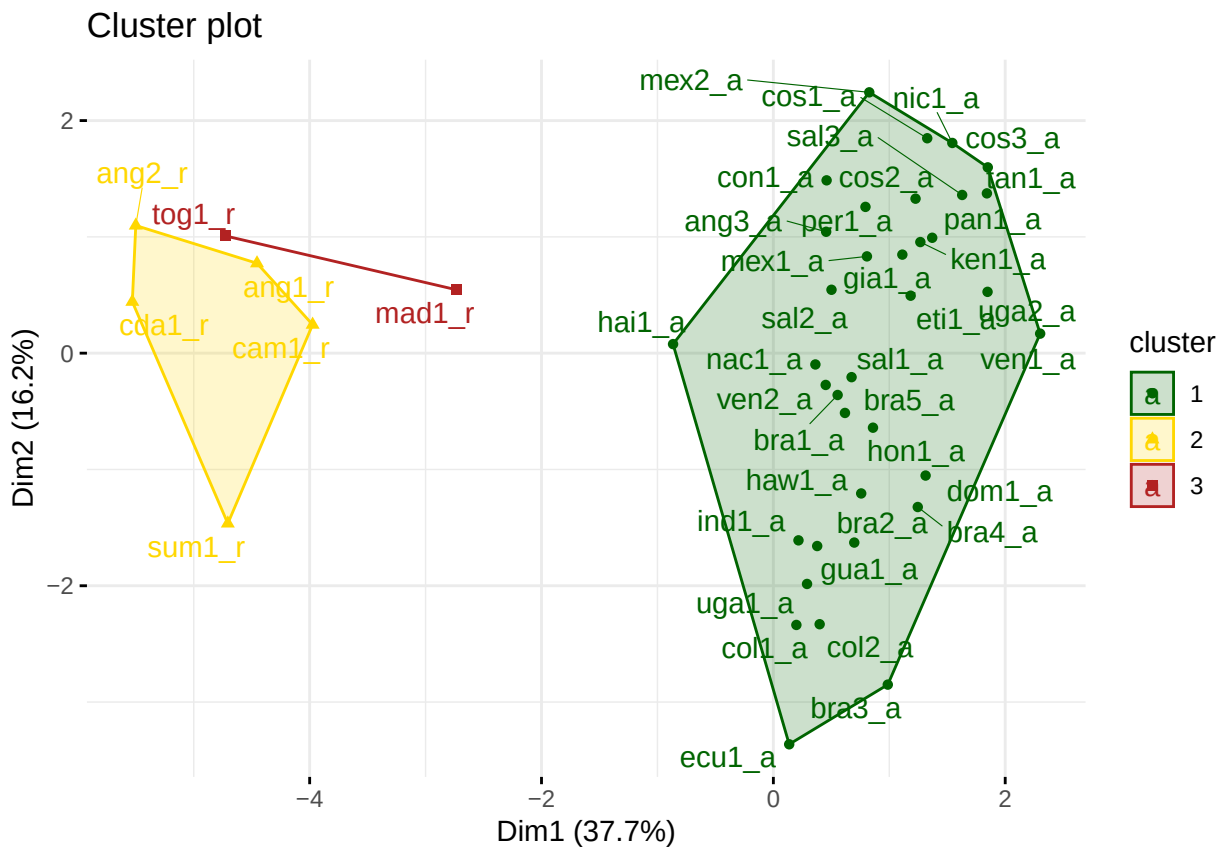
Il metodo di linkage **average** mostra il risultato migliore in termini di distanze cophenetiche.

Di seguito i grafici corrispondenti.

```
fviz_dend(res.hc3, cex = 0.65, k = 3,
          k_colors = colors6, rect = TRUE, rect_border = "grey", rect_fill = TRUE,
          horiz = FALSE, lwd = 0.8) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1, size = 10))
```



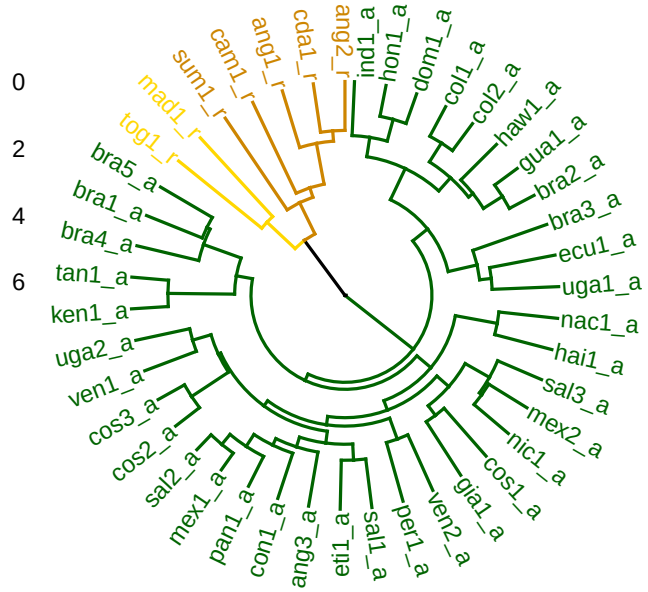

```
grp3 <- cutree(res.hc3, k = 3)
fviz_cluster(list(data = df.num, cluster = grp3),
             palette = colors3, ellipse.type = "convex", repel = TRUE,
             show.clust.cent = FALSE, ggtheme = theme_minimal())
```



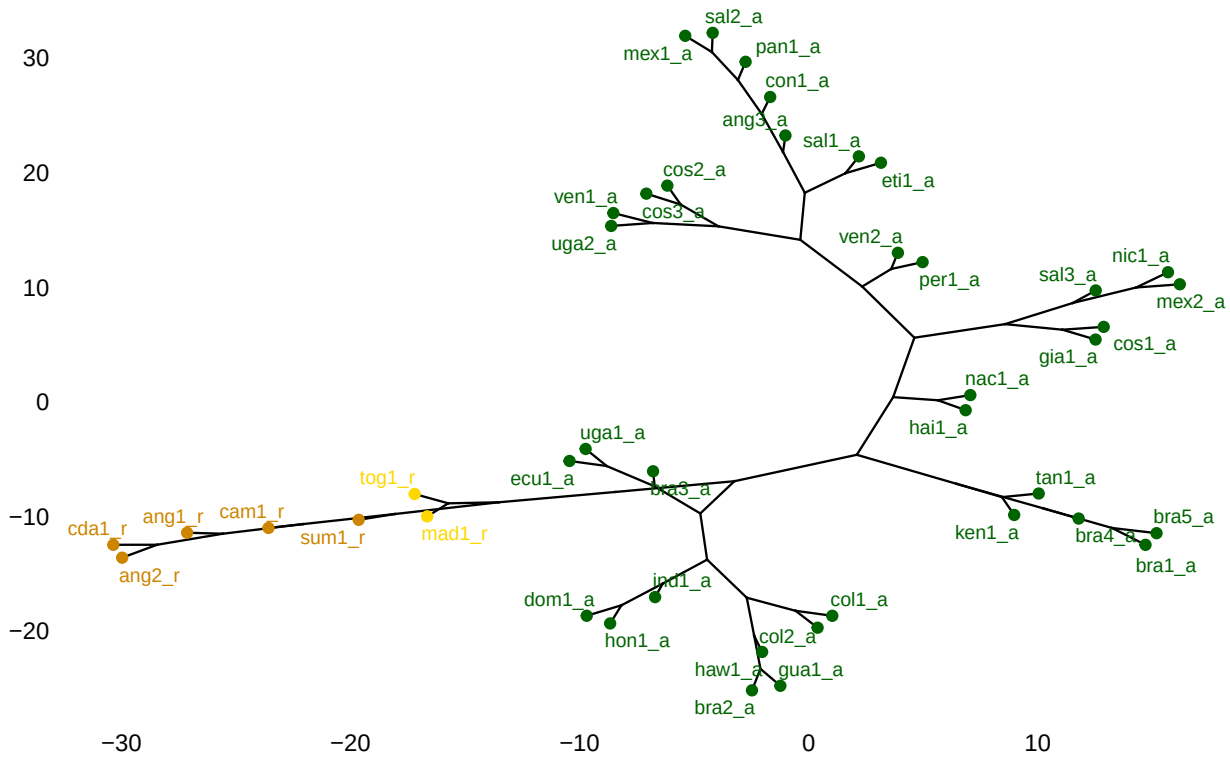
In questo caso si nota come il gruppo relativo ai caffè di varietà Robusta siano divisi in due cluster, mentre sia unico quello relativo ai caffè di Arabica.

Di seguito è mostrato lo stesso dendogramma con strutture grafiche differenti.

```
fviz_dend(res.hc3, cex = 0.8, k = 3, k_colors = colors6, type = "circular")
```



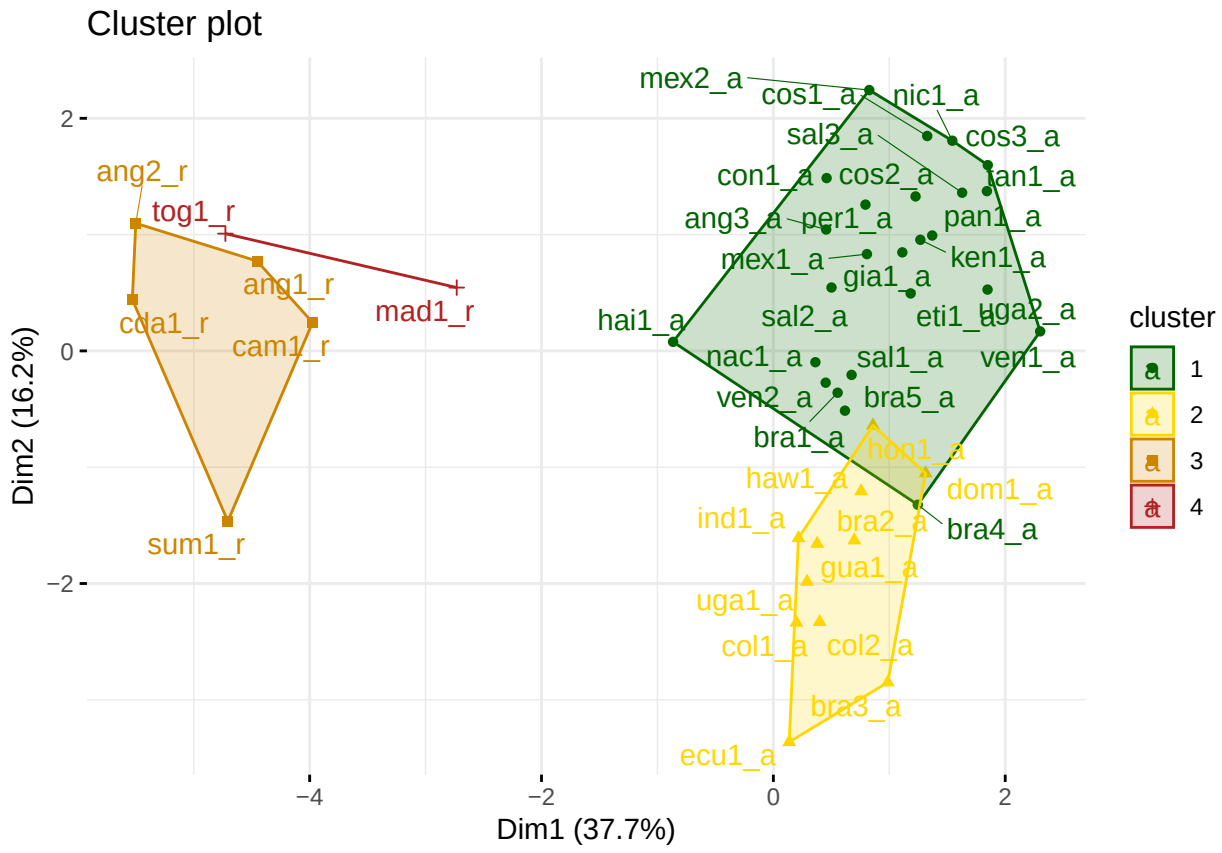

```
fviz_dend(res.hc3, k = 3, k_colors = colors6, type = "phylogenic", repel = TRUE)
```



Così come fatto per il clustering partizionale, anche in questo caso viene mostrata la suddivisione in 4 cluster.

```
grp2 <- cutree(res.hc3, k = 4)
table(grp2)
```

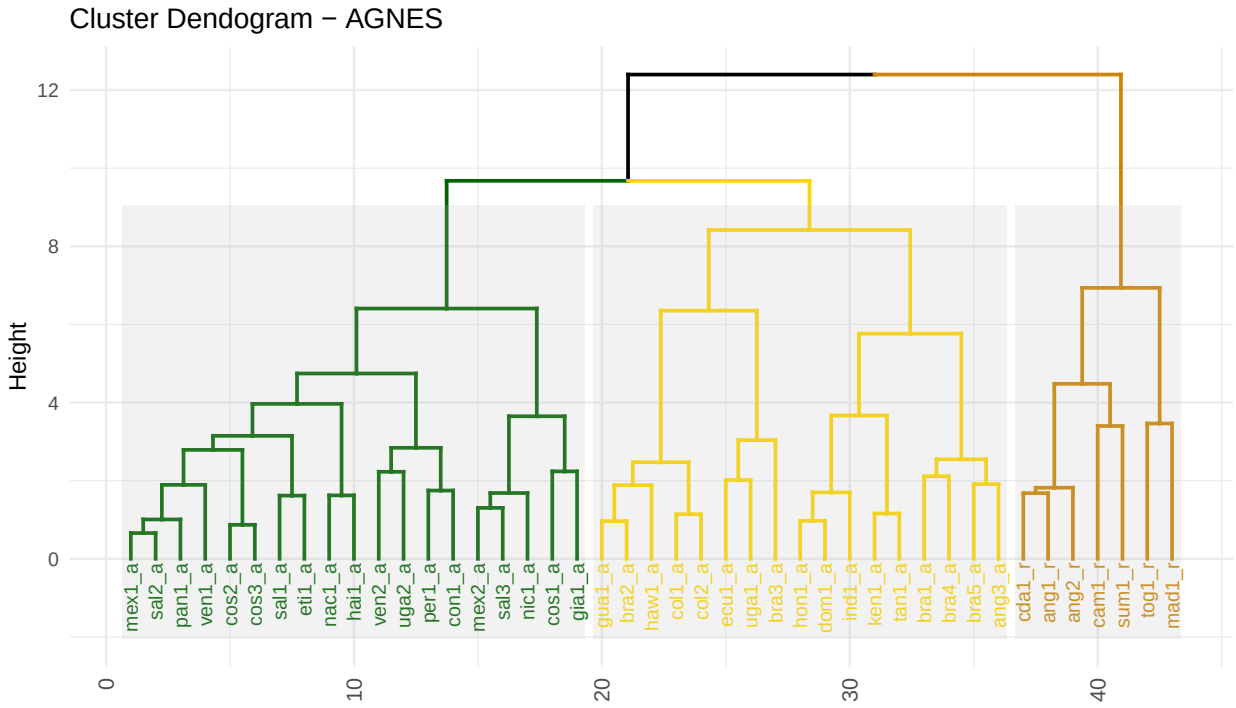
```
## grp2
##  1  2  3  4
## 25 11  5  2
```

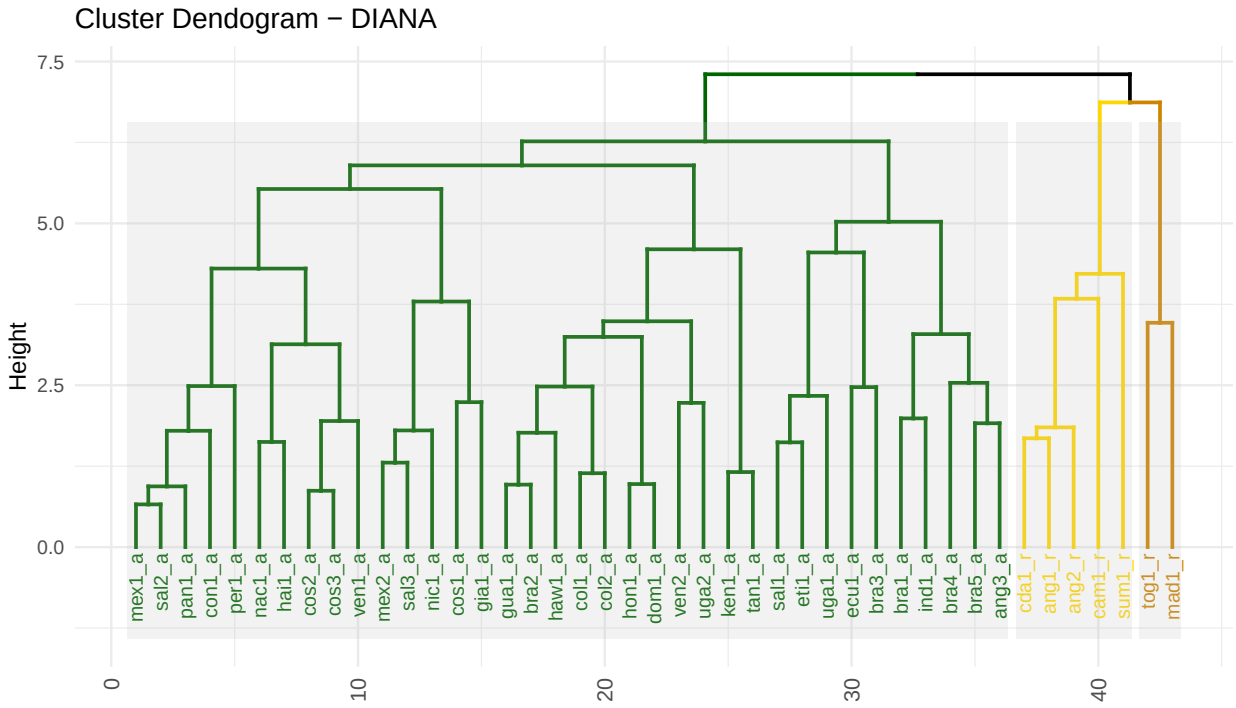


Le valutazioni dei risultati dei vari metodi applicati saranno effettuate alla fine della presente sezione.

Le funzioni AGNES e DIANA

R mette a disposizioni le funzioni legate agli algoritmi agglomerativi e divisivi, rispettivamente implementati dalle funzioni **agnes** (**AG**glomerative **NES**ting) and **diana** (**DI**visive **ANA**lysis). Le funzioni effettuano gli step visti singolarmente nelle sezioni precedenti: *scale*, *dist* e *hclust*, consentendo di scegliere il metodo per la distanza e per il linkage. Di seguito i rispettivi dendogrammi.





Confronto tra dendogramma

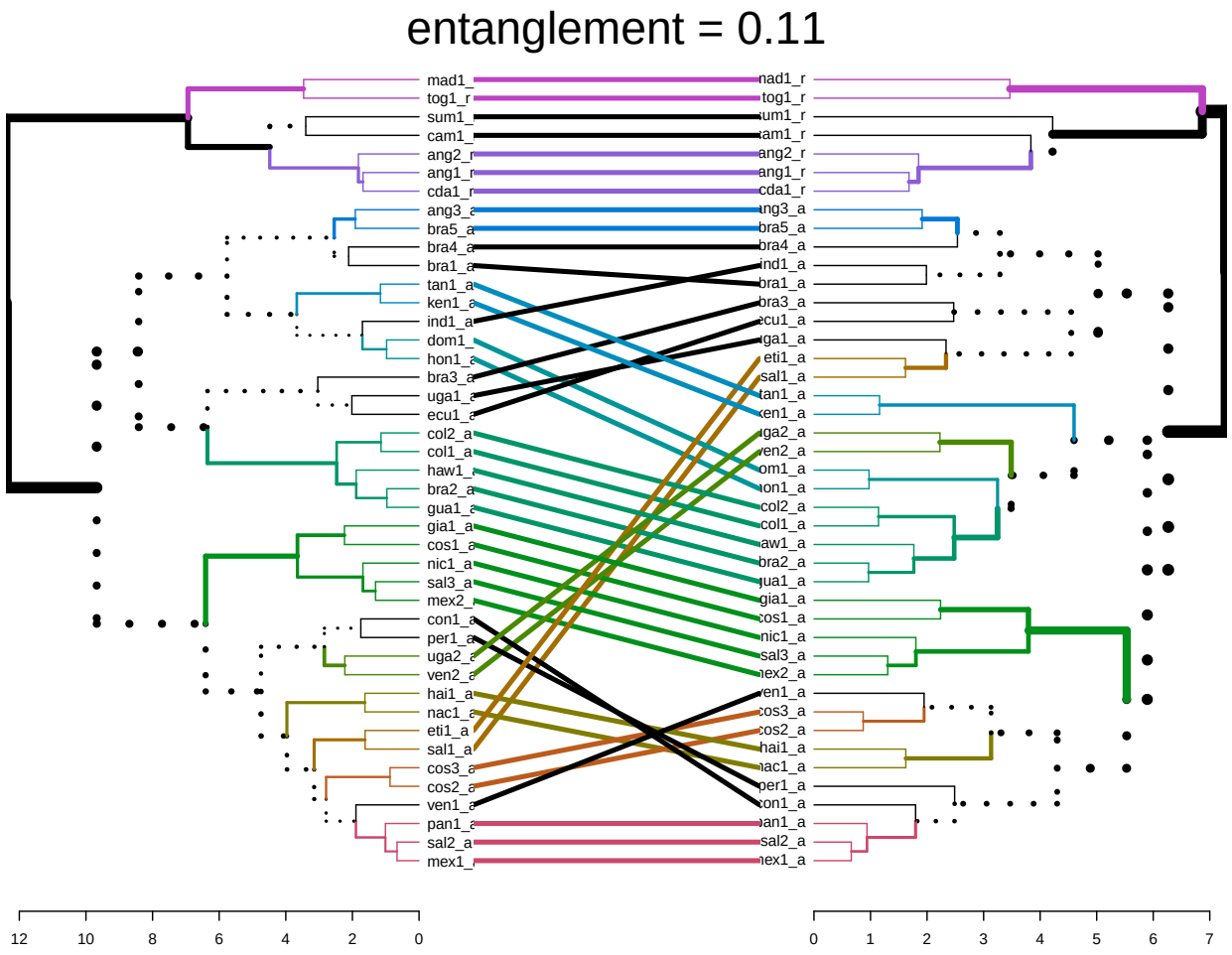
La funzione **tanglegram** del pacchetto **dendextend** permette di mettere a confronto graficamente due dendogrammi messi insieme mediante la funzione **dendlist** dello stesso pacchetto.

Di seguito confrontiamo i due dendogrammi ottenuti nella sezione precedente dalle funzioni AGNES e DIANA.

```
library(dendextend)

dend1 <- as.dendrogram (res.agnes)
dend2 <- as.dendrogram (res.diana)
dend_list <- dendlist(dend1, dend2)

tanglegram(dend1, dend2,
    highlight_distinct_edges = TRUE,
    common_subtrees_color_lines = TRUE,
    common_subtrees_color_branches = TRUE,
    main = paste("entanglement =", round(entanglement(dend_list), 2))
)
```



Il valore **entanglement** assume valori da 0 ad 1. Il valore ottenuto pari a 0.11 è indice del fatto che i due dendogrammi rappresentano in modo abbastanza simile i dati del dataset.

Metodi di validazione

Per valutare la qualità dei cluster ottenuti con i diversi algoritmi vengono applicate alcune metriche di validazione. Tali metriche si suddividono in tre categorie: interna, che misura la coesione e la separazione dei cluster senza riferimento a etichette esterne; esterna, che confronta i cluster con una partizione di riferimento nota; relativa, che confronta più soluzioni di clustering per identificare la più stabile.

Validazione Interna

Di seguito due metodi per la validazione interna.

- **Silhouette**: misura quanto un punto è simile ai punti del proprio cluster rispetto a quelli degli altri cluster; assume valori tra -1 e 1, valori più alti indicano cluster ben separati.
- **Indice di Dunn**: è dato dal rapporto tra la distanza minima tra cluster e la dimensione massima di un cluster; un valore alto indica cluster ben separati e compatti.

Per il calcolo di questi indici usiamo la funzione **cluster.stats** del pacchetto **fpc**:

```
##                                     silhouette dunn_index
## k-means (k=2)                        0.4186062  0.7748133
## k-means (k=3)                        0.2291961  0.3334320
## k-means (k=4)                        0.1986312  0.3038764
## k-medoids (K=2)                      0.4186062  0.7748133
## k-medoids (K=3)                      0.2194148  0.2879368
## Hierarchical - Euclidean Dist (K=2)  0.4186062  0.7748133
## Hierarchical - Euclidean Dist (K=3)  0.2291961  0.3334320
## Hierarchical - Manhattan Dist (K=2)  0.4186062  0.7748133
## Hierarchical - Manhattan Dist (K=3)  0.1997713  0.3038764
## Hierarchical - Diana (k=2)           0.4186062  0.7748133
## Hierarchical - Diana (k=3)           0.3804807  0.5351811
## Hierarchical - Agnes (K=2)           0.4186062  0.7748133
## Hierarchical - Agnes (K=3)           0.1941655  0.2619362
```

Rispetto al numero di cluster, i valori più alti si ottengono per k = 2 per tutti gli algoritmi di clustering, probabilmente per la ormai nota separazione delle caratteristiche che distinguono le varietà. Per k = 3, k-means ha score più alti rispetto a k-medoids. Per il clustering gerarchico, l'algoritmo DIANA presenta score più alti di Silhouette e Dunn.

Validazione Esterna

Di seguito due metodi per la validazione esterna

- **Corrected Rand Index (ARI)**: Valuta la somiglianza tra due partizionamenti, correggendo per l'**accordo atteso**. Varia tra -1 e 1, con 1 che indica una corrispondenza perfetta.
- **Meila Variation of Information (VI)**: Misura la distanza tra due clusterizzazioni basata sull'entropia. Valori più bassi indicano partizionamenti più simili.

Ad esempio, possiamo studiare se la suddivisione creata da k-medoids (con k = 2) riflette la varietà di caffè. Osserviamo per prima cosa la matrice di confusione:

```
##
##      1  2
##   1 36  0
##   2  0  7
```

In questo caso, l'algoritmo ha identificato un matching perfetto con la varietà di caffè. Osserviamo il **corrected rand index** e il **Meila Variation of Information (VI)**:

```
stats_kmed2 <- cluster.stats(dist(df.num), coffee$Variety, pam2.res$clustering)
stats_kmed2$corrected.rand
```

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

```
stats_kmed2$vi
```

```
## [1] 0
```

I valori ottenuti confermano la corretta classificazione delle varietà, in quanto l'*accordo* tra le varietà dei caffè e la soluzione dei cluster è pari ad 1, utilizzando l'indice Rand corretto, e il *disaccordo* tra le varietà e la soluzione dei cluster è 0, utilizzando l'indice VI di Meila.

Validazione Relativa

Questi metodi riguardano misure di stabilità, una versione particolare delle misure interne, che valutano la stabilità di un risultato di clustering confrontandolo con i cluster ottenuti dopo che ogni variabile è stata rimossa, uno alla volta.

Le misure di interesse riguardano: - **Average Proportion of Non-Overlap (APN)**: indica la percentuale media di osservazioni che cambiano cluster tra diverse partizioni; valori più bassi indicano maggiore stabilità. - **Average Distance (AD)**: indica la distanza media tra partizioni ottenute con diversi metodi o parametri. - **Average Distance Between Means (ADM)**: indica la distanza media tra le medie delle distanze intra-cluster. - **Figure of Merit (FOM)**: è una metrica utilizzata per misurare la compattezza interna dei cluster.

Per calcolarli usiamo la funzione **clValid** del pacchetto **clValid**:

```
##
## Clustering Methods:
##  hierarchical kmeans pam
##
## Cluster sizes:
##  2 3 4 5
##
## Validation Measures:
##                         2      3      4      5
##
## hierarchical APN  0.0000 0.0096 0.0275 0.0766
##              AD   3.7109 3.5775 3.4973 3.4121
##              ADM  0.0000 0.0906 0.2017 0.3484
##              FOM  0.7943 0.7883 0.7825 0.7822
## kmeans       APN  0.0000 0.0370 0.0532 0.1610
##              AD   3.7109 3.5610 3.2272 3.1616
##              ADM  0.0000 0.1314 0.3788 0.5984
##              FOM  0.7943 0.7811 0.7378 0.7340
## pam          APN  0.0000 0.0659 0.1150 0.1779
##              AD   3.7109 3.3731 3.2191 3.1389
##              ADM  0.0000 0.2480 0.3698 0.5672
##              FOM  0.7943 0.7481 0.7435 0.7444
##
## Optimal Scores:
##
```

```
##     Score  Method       Clusters
## APN 0.0000 hierarchical 2
## AD  3.1389 pam          5
## ADM 0.0000 hierarchical 2
## FOM 0.7340 kmeans       5
```

Dai risultati ottenuti, il clustering gerarchico risulta essere il più stabile e mantiene una coerenza interna maggiore sia con 2 che con 3 cluster. Il k-means risulta migliore con 3 cluster anziché con 2 (FOM più basso). Il k-medoids denota un compromesso, per k=3 migliora la qualità complessiva (FOM) ma con una dispersione maggiore (ADM).

Modelli Avanzati di Clustering

Dopo aver condotto una prima analisi con metodi di clustering partizionali, si procede lo studio con modelli più avanzati che permettono di catturare maggiormente la complessità delle distribuzioni e delle relazioni latenti tra le variabili. In particolare, verranno applicati i seguenti metodi:

- **Gaussian Mixture Model (GMM)**, che consente di modellare la distribuzione dei dati come una combinazione di distribuzioni gaussiane, permettendo l'identificazione di sottogruppi omogenei all'interno del dataset.
- **Parsimonious Gaussian Mixture Model (PGMM)**, che estende l'approccio del GMM introducendo una struttura parametrica più parsimoniosa, con l'obiettivo di ridurre il numero di parametri da stimare. Questa semplificazione li rende utili per analizzare dati ad alta dimensionalità o con limitati campioni, bilanciando precisione e complessità.
- **Finite Mixture of Regressions (FMR)**, che combina le potenzialità dei modelli di regressione con le tecniche di clustering, consentendo di modellare relazioni diverse tra variabili indipendenti e dipendenti in sottogruppi latenti.
- **Finite Mixture of Regressions with Concomitant variables (FMRC)**, che aggiunge ulteriori informazioni tramite covariate, migliorando la capacità di spiegare la variabilità tra gruppi e fornendo una comprensione più approfondita dei fattori che influenzano il fenomeno studiato.

Applicazione dei modelli GMM

GMM consente di modellare la distribuzione dei dati osservati mediante un insieme di "cluster," ciascuno descritto da una distribuzione gaussiana. Si utilizza la funzione **Mclust** di **mclust**, che si basa sull'algoritmo EM (Expectation-Maximization) per stimare i parametri del modello. Una volta stimati i parametri, il pacchetto seleziona il modello migliore in base al BIC.

```
library(mclust)
gmm_model <- Mclust(df.num)
summary(gmm_model)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEI (diagonal, equal shape) model with 3 components:
##
##  log-likelihood  n df       BIC       ICL
##       -551.1776 43 52 -1297.938 -1298.114
##
## Clustering table:
##  1  2  3
## 22 14  7
```

L'algoritmo ha selezionato un modello di tipo **VEI** (volume variabile, forma uguale, orientazione isotropa), suggerendo che i cluster individuati presentano varianze diverse ma forme simili.

Controlliamo il numero di cluster creati e aggiungiamo la colonna “Cluster” al dataframe”:

```
table(gmm_model$classification)
```

```
##
##  1  2  3
## 22 14  7
```

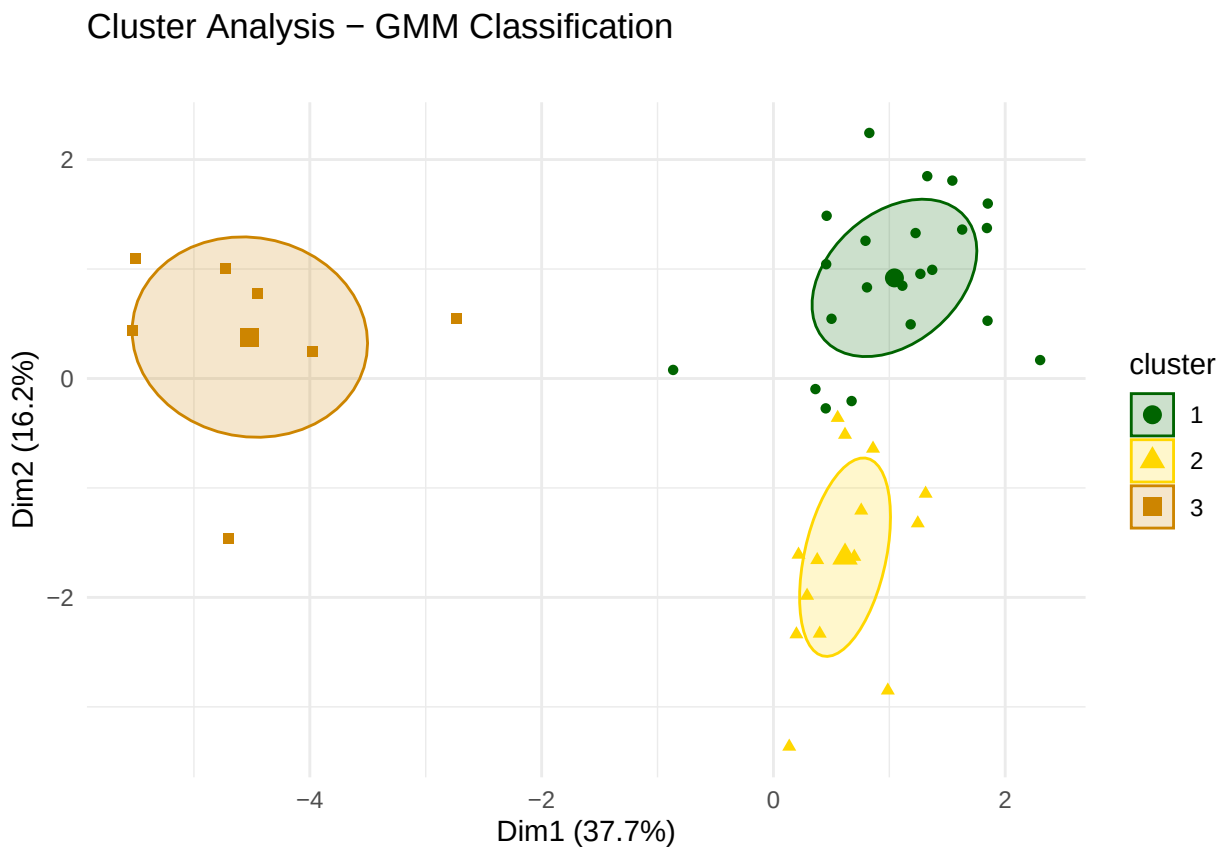
```
df.scaled <- as.data.frame(df.num)
df.scaled$Cluster <- gmm_model$classification
```

Osserviamo le probabilità a posteriori per il modello:

```
##                [,1]         [,2]         [,3]
## mex1_a 9.998966e-01 1.033651e-04 3.841226e-19
## mex2_a 1.000000e+00 5.880600e-09 7.038173e-18
## gua1_a 6.496995e-06 9.999935e-01 3.158828e-18
## hon1_a 2.560372e-02 9.743963e-01 3.001385e-20
## sal1_a 9.982961e-01 1.703882e-03 6.770879e-19
## sal2_a 9.997689e-01 2.310858e-04 5.236902e-18
```

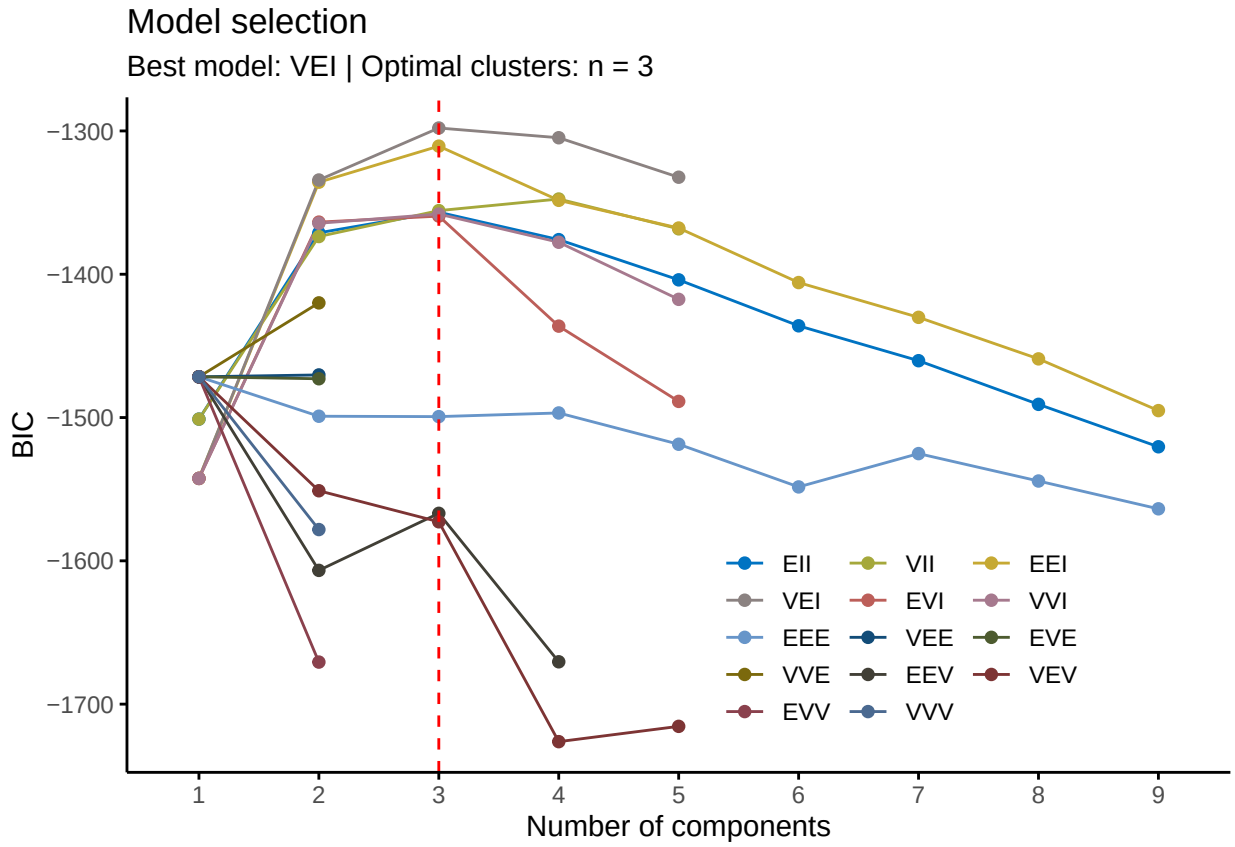
Si nota che molte osservazioni hanno una probabilità alta (vicina a 1) di appartenere ad un cluster specifico. Questo indica che i cluster sono ben separati.

Visualizziamo graficamente i cluster mediante **fviz_mclust** per verificare se esistono overlap tra gli stessi.



Dal grafico emerge che i punti di ciascun colore formano gruppi ben distinti e abbastanza lontani tra loro, il che significa che il modello sta identificando cluster ben definiti. Le ellissi rappresentano la covarianza dei cluster; il cluster 2 presenta una ellissi allungata, il che indica una variabilità maggiore in una specifica direzione. Mentre i cluster 1 e 3 presentano ellissi più compatte, ad indicare cluster più coerenti e densi.

Osserviamo ora il grafico "BIC", che mostra il valore del Bayesian Information Criterion (BIC) per diversi numeri di cluster e modelli di covarianza.



Il modello con il valore di BIC più alto è quello scelto da Mclust; una linea tratteggiata evidenzia il numero di componenti scelte per questo modello. Si osserva che il BIC decresce all'aumentare del numero di cluster, quindi l'aggiunta di più cluster non migliora il modello. In questo scenario, un numero di cluster pari a tre sembra una buona scelta.

Il grafico "uncertainty" di seguito mostra il livello di incertezza nell'assegnazione delle osservazioni ai cluster. Ogni punto rappresenta un'osservazione e il colore riflette il grado di incertezza (più scuro e più grande indicano più incertezza).

```
fviz_mclust(gmm_model, "uncertainty", palette = colors6) + theme_minimal()
```

Dal grafico emerge che la maggior parte delle osservazioni ha bassa incertezza (colori chiari e piccoli), mentre solo pochi punti (tra il cluster 1 e il cluster 2) presentano aree scure e grandi; tali osservazioni hanno probabilità simili di appartenere a più cluster.

Valutazione delle prestazioni

Osserviamo il confronto tra la suddivisione in cluster ottenuto tramite modelli GMM rispetto a quello ottimale ottenuto nello studio precedente (con metodo K-means e k=3).

Verranno utilizzate tre metriche principali: accuracy, Adjusted Rand Index (ARI) e la matrice di confusione.

- **Matrice di confusione**: offre una visione dettagliata delle performance del modello, mostrando il numero di osservazioni correttamente ed erroneamente assegnate per ciascun cluster.
- **Accuracy**: misura la percentuale di osservazioni correttamente assegnate al cluster di appartenenza.
- **Adjusted Rand Index (ARI)**: è una metrica più robusta che confronta l'assegnazione dei cluster con la vera partizione (o con una partizione di riferimento), tenendo conto delle assegnazioni casuali. Assume valori compresi tra -1 e 1, dove un valore vicino a 1 indica una forte corrispondenza tra la partizione ottenuta e quella attesa.

Creiamo una matrice di confusione e usiamo la funzione **confmatrix** per riordinare le righe della matrice in modo tale da avere sulla diagonale principale i valori numerici più grandi:

```
confmatrix <- function(A) {
  cf <- A
  G <- nrow(A)
  for (i in 1:(G - 1)) {
    max_row <- which.max(cf[i:G, i]) + (i - 1)
    if (max_row != i) {
      cf[c(i, max_row), ] <- cf[c(max_row, i), ]
      rownames(cf)[c(i, max_row)] <- rownames(cf)[c(max_row, i)]
    }
  }
  return(cf)
}

# Creazione della matrice di confusione tra GMM e K-means
optimal_classification <- km3.res$cluster
conf_matrix <- confmatrix(table(df.scaled$Cluster, optimal_classification))

conf_matrix
```

```
##    optimal_classification
##      1  2  3
##   3  7  0  0
##   2  0 14  0
##   1  0  0 22
```

Sebbene i numeri assegnati ai cluster differiscano tra GMM e k-means, si nota che la suddivisione risulta essere la stessa.

```
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
print(paste("Accuracy:", round(accuracy * 100, 2), "%"))
```

```
## [1] "Accuracy: 100 %"
```

```
ari <- adjustedRandIndex(df.scaled$Cluster, optimal_classification)
print(paste("Adjusted Rand Index (ARI):", round(ari, 2)))
```

```
## [1] "Adjusted Rand Index (ARI): 1"
```

Un aspetto interessante emerso dall'analisi è il disallineamento delle etichette tra i cluster individuati dai modelli GMM e K-means. Sebbene entrambi i metodi abbiano identificato esattamente gli stessi cluster, l'assegnazione delle etichette è risultata diversa (ad esempio, il cluster 1 di GMM corrisponde al cluster 3 di K-means). Tuttavia, si nota una perfetta corrispondenza tra le partizioni, ottenendo un'accuracy del 100% e un Adjusted Rand Index (ARI) pari a 1.

Parsimonious Gaussian Mixture Model (PGMM)

Usiamo la funzione **pgmmEM** del pacchetto **pgmm** per applicare i Parsimonious Gaussian Mixture Models. Questo tipo di modelli combina l'approccio dei mixture models con l'analisi della struttura della covarianza, permettendo di identificare gruppi latenti in dati multivariati. La funzione implementa l'algoritmo Expectation-Maximization (EM) per stimare i parametri del modello. Questo consente di trovare la migliore partizione possibile dei dati tenendo conto sia della struttura dei cluster che della parsimonia del modello.

```
df.scaled <- df.scaled[, -ncol(df.scaled)]
```

Come parametri del modello possono essere impostati: - **zstart**, che definisce il metodo di inizializzazione delle probabilità a posteriori per l'assegnazione dei cluster; il valore 1 per usare una partizione iniziale casuale; il valore 2 per utilizzare l'output di un altro metodo di clustering (K-means) - **loop**: Specifica quanti diversi avvii casuali devono essere eseguiti.

Usiamo il modello con inizializzazione casuale delle probabilità a posteriori e 10 differenti punti iniziali:

```
pgmm_model <- pgmmEM(x = df.scaled, zstart = 1, loop=10)
```

```
## Based on 10 random starts, the best model (BIC) for the range of factors and components used is a UCU
## The BIC for this model is -1365.492.
```

```
summary(pgmm_model)
```

```
## Based on 10 random starts, the best model (BIC) for the range of factors and components used is a UCU
## The BIC for this model is -1365.492.
```

Il miglior modello selezionato in base al Bayesian Information Criterion (BIC) è risultato essere il modello UCU, con q = 1 fattore latente e G = 2 componenti (cluster). Il valore del BIC per questo modello è -1365.492, indicando che tra i modelli testati, questo offre il miglior compromesso tra bontà di adattamento e complessità del modello. L'identificazione di G = 2 cluster suggerisce la presenza di due gruppi latenti distinti nel dataset, ciascuno caratterizzato da una propria struttura delle relazioni tra le variabili osservate.

Avendo identificato due gruppi, proviamo a confrontare i risultati ottenuti con la colonna "varietà" del dataset originale, per vedere se le classificazioni del modello riflettono la varietà di caffè:

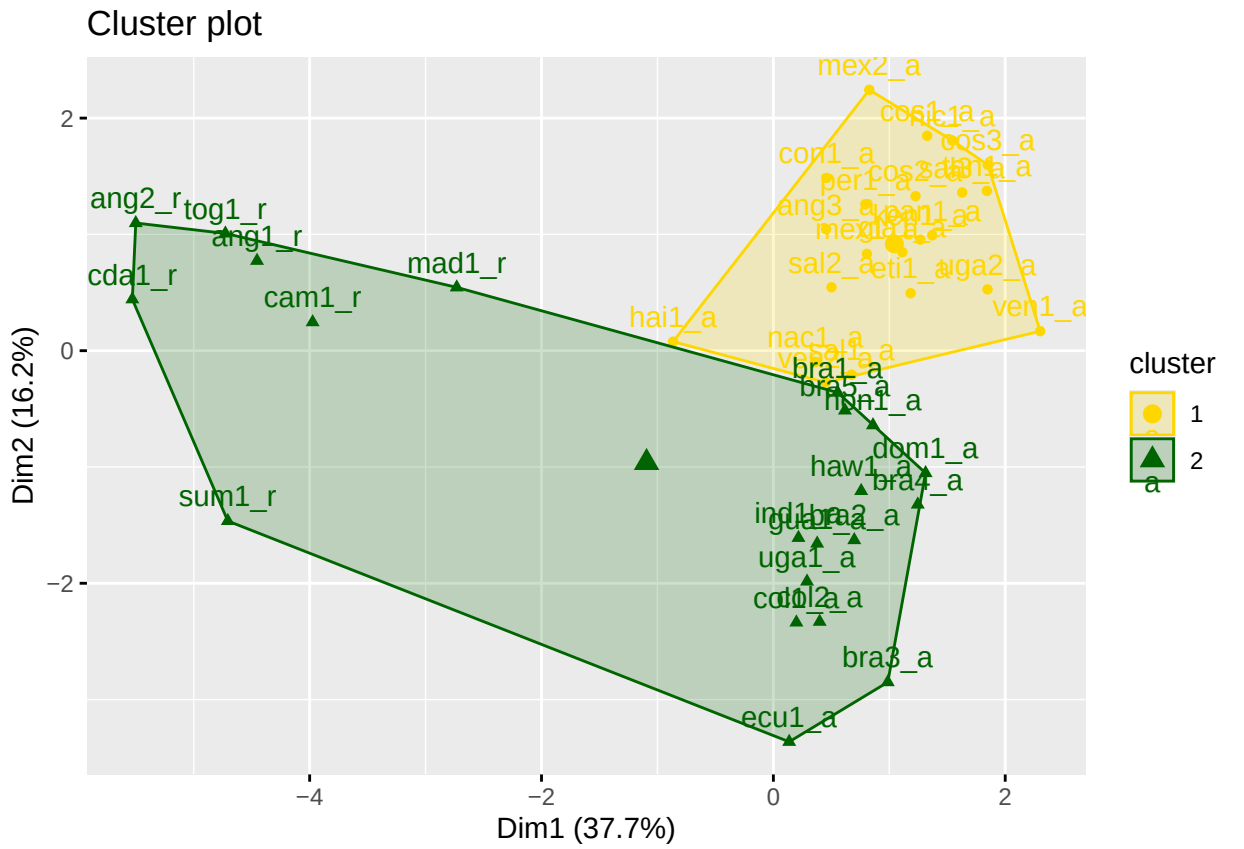
```
res <- table(coffee[,1], pgmm_model$map)
res
```

```
##
##      1  2
##   1 22 14
##   2  0  7
```

```
accuracy <- sum(diag(res)) / sum(res)
print(paste("Accuracy:", round(accuracy * 100, 2), "%"))
```

```
## [1] "Accuracy: 67.44 %"
```

```
fviz_cluster(list(data = df.scaled, cluster = pgmm_model$map), palette=colors2)
```



Il modello classifica correttamente le 22 osservazioni della classe 1 nel cluster 1 e 7 osservazioni della classe 2 nel cluster 2, ma commette un errore su 14 osservazioni della classe 1 che vengono assegnate al cluster 2.

Proviamo ora a creare un secondo modello in cui il punto di partenza (zstart) è k-means e il modello scelto è UUU:

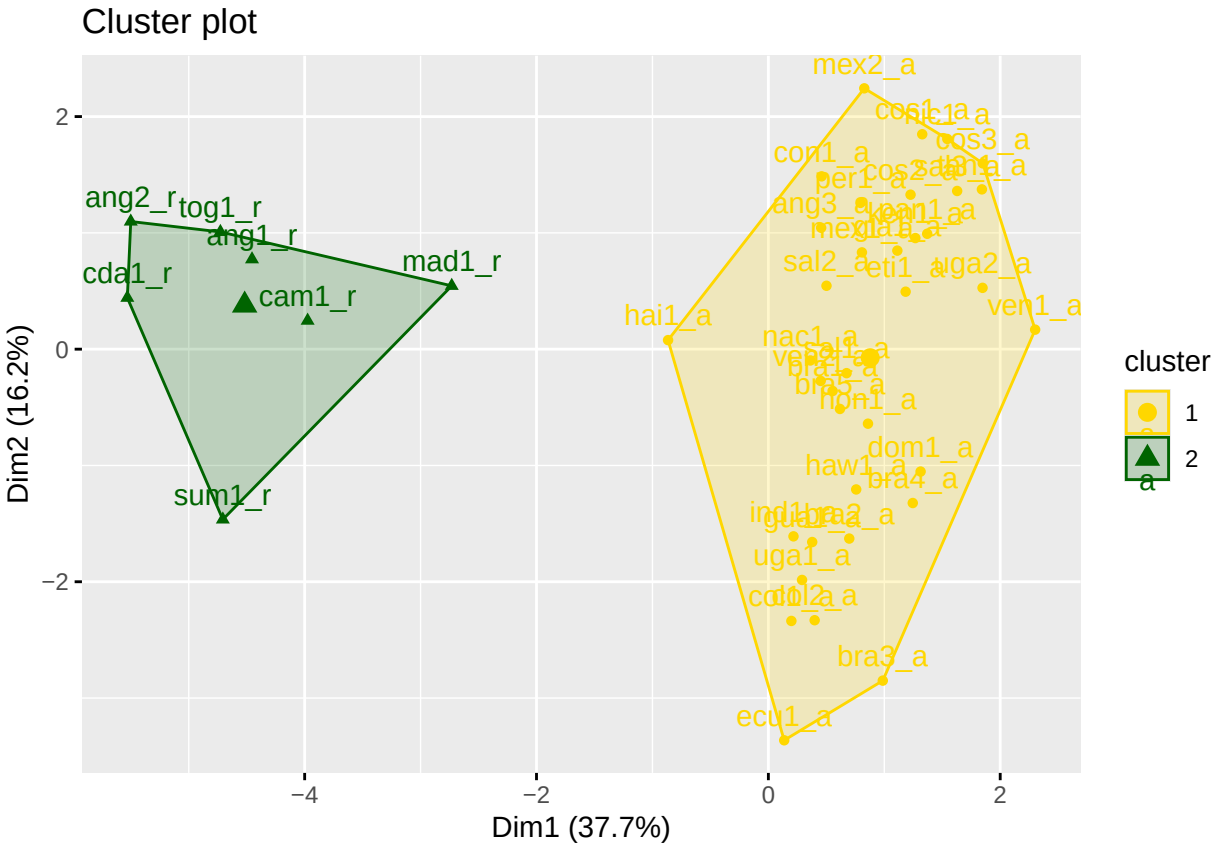
```
pgmm_model2<-pgmmEM(x=df.scaled,zstart=2,modelSubset=c("UUU"))
```

```
## Based on k-means starting values, the best model (BIC) for the range of factors and components used i
## The BIC for this model is -1339.23.
```

```
table(coffee[,1],pgmm_model2$map)
```

```
##
##      1  2
##   1 36  0
##   2  0  7
```

```
fviz_cluster(list(data = df.scaled, cluster = pgmm_model2$map), palette=colors2)
```

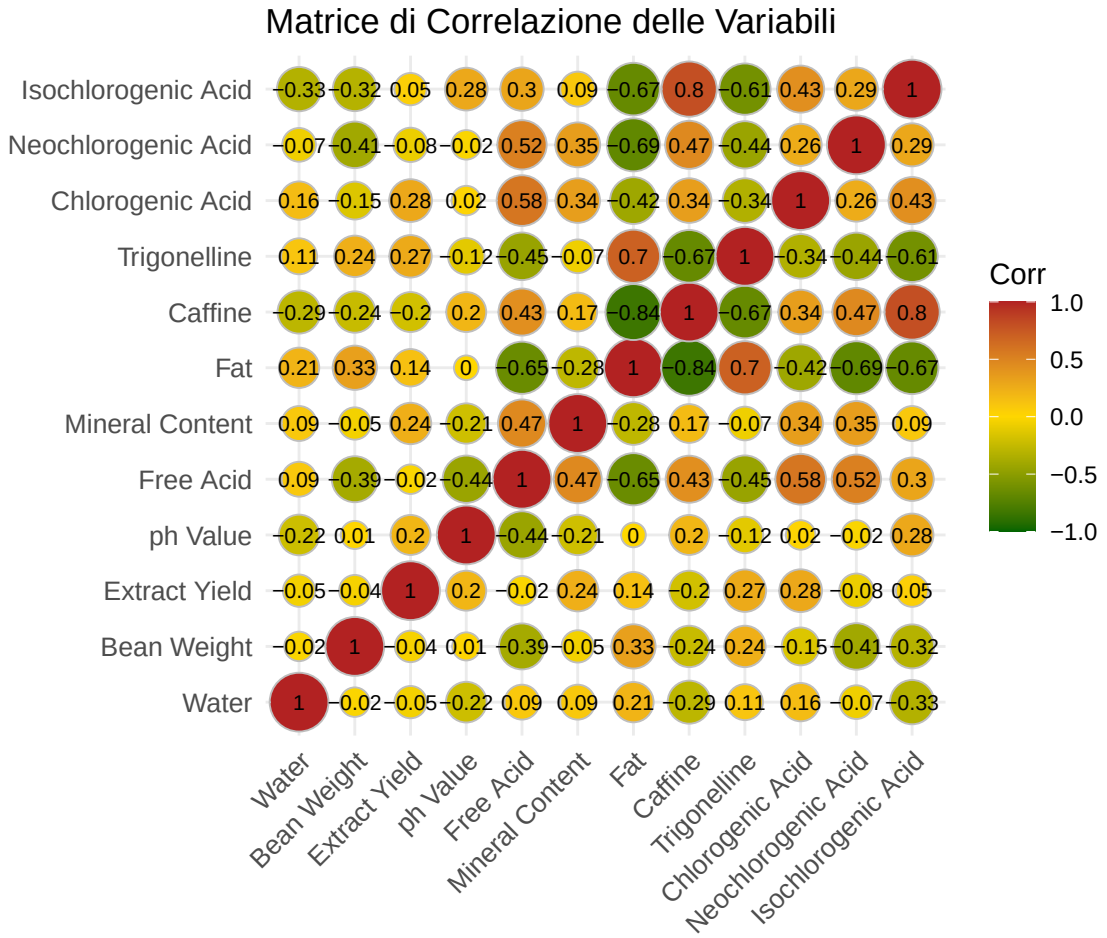


Per questo modello, tutte le osservazioni sono classificate correttamente e rispecchiano la varietà di caffè.

Misture di Modelli di Regressione Lineare

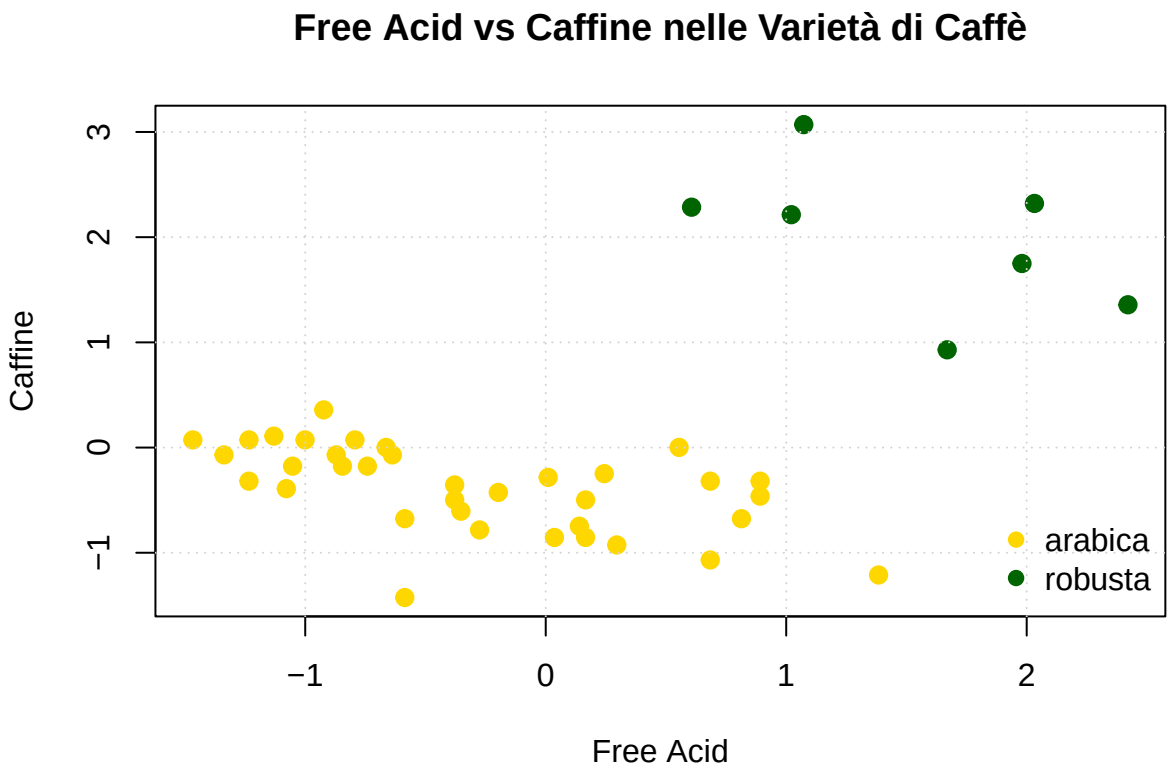
I Mixture of Linear Regression Models (FMR) rappresentano un approccio statistico flessibile per analizzare dati eterogenei in cui si sospetta la presenza di sottogruppi latenti con relazioni diverse tra le variabili. A differenza di un semplice modello di regressione lineare, che assume una relazione unica tra la variabile dipendente e le variabili indipendenti, i modelli FMR permettono di modellare più relazioni contemporaneamente, ciascuna specifica per un sottogruppo di dati.

In questa analisi, vogliamo studiare la caffeina come variabile target, con l'obiettivo di capire quali fattori chimici e fisici del caffè influenzano la quantità presente. L'utilizzo di modelli statistici avanzati ci permette di identificare le variabili più rilevanti e comprendere meglio le dinamiche che determinano il contenuto di caffeina. Studiamo, quindi, la correlazione tra variabili per capire quali sono quelle maggiormente correlate con la caffeina:



Riprendendo l'analisi effettuata inizialmente, focalizziamo lo studio sulla relazione tra la caffeina (**Caffine**) e l'acidità del caffè (**Free Acid**).

Visualizziamo uno scatterplot per capire la relazione tra la caffeina e la variabile di interesse:



Questa visualizzazione mette in evidenza la presenza di due raggruppamenti distinti, suggerendo potenziali differenze strutturali tra le varietà di caffè in relazione al contenuto di caffeina e all'acidità libera.

Questa osservazione preliminare motiva l'utilizzo di un Finite Mixture Regression Model (FRM), che permette di modellare le relazioni lineari differenti tra le due variabili nei sottogruppi latenti. L'obiettivo è comprendere come i fattori chimici e fisici del caffè, in particolare l'acidità libera, influenzino il contenuto di caffeina nelle diverse varietà.

Finite Mixture of Regressions (FMR)

Per approfondire l'analisi della relazione tra caffeina e Free Acid, è stato utilizzato il metodo **stepFlexmix**. In particolare, si è specificato:

- *k=2* per indicare al modello di individuare due componenti (o cluster) distinti nei dati, coerentemente con i due raggruppamenti osservati nello scatterplot iniziale
- *nrep=10* per definire il numero di ripetizioni del processo di fitting con diversi valori iniziali.

In particolare, il modello viene eseguito 10 volte e viene mantenuta la soluzione con la massima verosimiglianza.

```
set.seed(1000)
X.fmr <- stepFlexmix(caffine ~ free_acid, k=2, nrep=10)
```

```
## 2 : * * * * * * * * * *
```

```
summary(X.fmr)
```

```
##
## Call:
## stepFlexmix(caffine ~ free_acid, k = 2, nrep = 10)
##
##        prior size post>0 ratio
## Comp.1 0.287    9     43 0.209
## Comp.2 0.713   34     36 0.944
##
## 'log Lik.' -34.57109 (df=7)
## AIC: 83.14219   BIC: 95.47059
```

I risultati del modello riportano le seguenti informazioni:

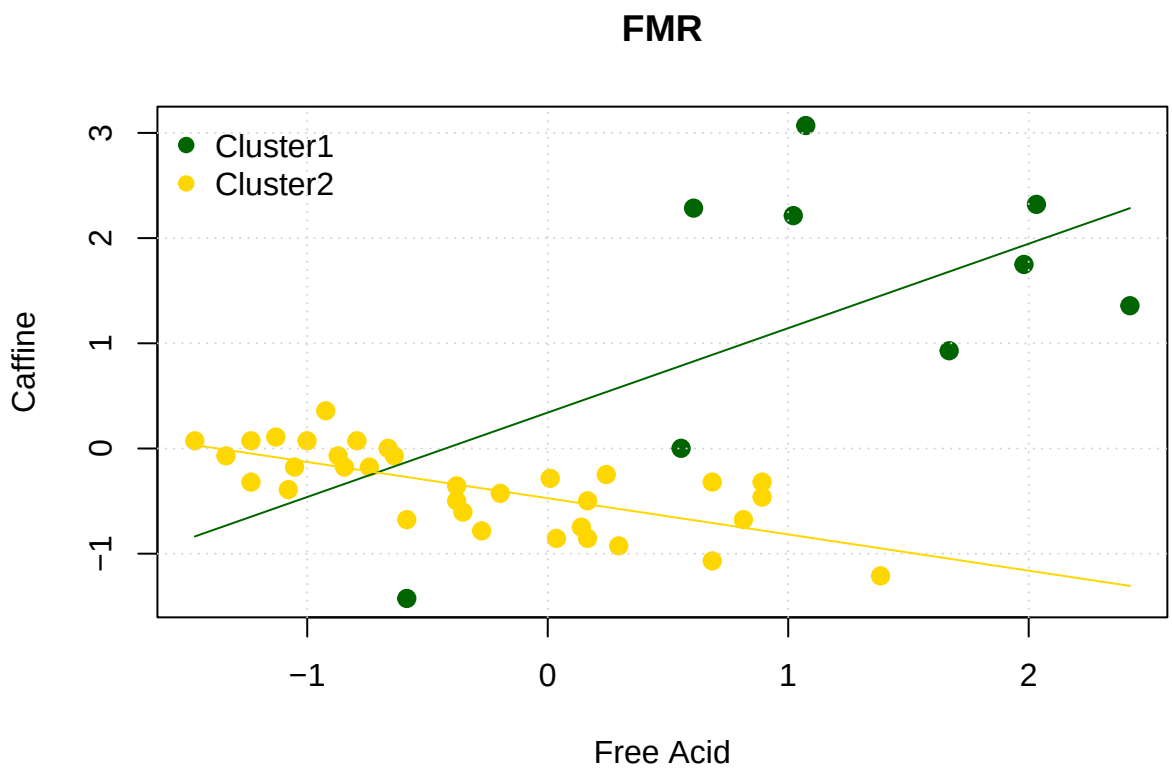
- **prior**: rappresenta le probabilità a priori associate a ciascun componente (o cluster) del modello. Indica la proporzione prevista di osservazioni appartenenti a ogni cluster prima di osservare i dati. In questo caso, otteniamo circa il 29% delle osservazioni nel primo componente e circa il 71% nel secondo.
- **size**: indica il numero di osservazioni assegnate a ciascun cluster dopo l'addestramento del modello. Questo valore riflette la dimensione effettiva dei gruppi, che nel caso in questione risulta essere di 9 elementi nel primo componente e 34 nel secondo, che a grandi linee conferma la suddivisione delle prior.
- **post $>$ 0**: mostra il numero di osservazioni che hanno una probabilità a posteriori maggiore di 0 di appartenere a ciascun cluster.
- **ratio**: rappresenta il rapporto tra il numero di osservazioni con probabilità a posteriori maggiore di zero e la dimensione totale del cluster. È una misura della "certezza" nell'assegnazione, infatti un valore vicino a 1 indica che quasi tutte le osservazioni sono chiaramente attribuibili al cluster, mentre valori più bassi suggeriscono una maggiore sovrapposizione tra cluster. In questo caso, la seconda componente presenta una ratio molto elevata, mentre la prima un valore più basso.

```r
pred.X.fmr <- clusters(X.fmr)

plot(free_acid, caffine, col = colors6[pred.X.fmr], pch = 19, cex = 1.2,
     xlab = "Free Acid", ylab = "Caffine", main = "FMR")
grid(col = "lightgray", lty = "dotted")
legend("topleft", legend = c("Cluster1", "Cluster2"), col=colors6, pch=19, bty="n")

FMR1<-parameters(X.fmr)[1:2,1]
FMR2<-parameters(X.fmr)[1:2,2]

curve(FMR1[1]+FMR1[2]*x,col="darkgreen",add=TRUE)
curve(FMR2[1]+FMR2[2]*x,col="gold",add=TRUE)
```



Il grafico presenta due distinti raggruppamenti, facilmente identificabili grazie alla colorazione e attribuibili alle varietà di caffè (arabica e robusta). Come emerso dalla fase di analisi preliminare, la varietà arabica si distingue per un basso contenuto di caffeina e acido libero, mentre la varietà robusta è caratterizzata da concentrazioni elevate di entrambi.

Analizzando le rette di regressione, si osserva che quella di colore giallo (probabilmente associabile al raggruppamento "arabica") presenta una pendenza negativa, mentre l'altra mostra una pendenza positiva. Da tale osservazione si deduce che il modello suggerisce una relazione differente tra caffeina e acido libero nelle due categorie.

Di seguito sono riportati alcuni indici di valutazione delle prestazioni. In particolare, si confronta il raggruppamento ottenuto tramite il metodo **stepFlexmix** con la suddivisione effettiva delle varietà di caffè, come indicato nella colonna **Variety** del dataset **coffee**.

```
cf.pred1.fmr<-confmatrix(table(coffee$Variety, clusters(X.fmr)))
cf.pred1.fmr
```

```
##
##      1  2
##   2  7  0
##   1  2 34
```

```
accuracy.fmr <-sum(diag(cf.pred1.fmr))/nrow(coffee)*100
accuracy.fmr
```

```
## [1] 95.34884
```

```
ARI.fmr<-adjustedRandIndex(coffee$Variety,clusters(X.fmr))
ARI.fmr
```

```
## [1] 0.7882282
```

I risultati ottenuti dal modello **stepFlexmix** mostrano una buona capacità di distinguere tra i due cluster presenti nei dati. L'Adjusted Rand Index (ARI), pari a 0.7882, indica un'elevata concordanza tra i cluster individuati dal modello e le etichette reali (o attese). Dalla matrice di confusione si evince che il modello ha identificato correttamente la struttura latente del dataset, con una bassa percentuale di misclassificazioni.

Finite Mixture of Regressions with Concomitant variables (FMRC)

I modelli di regressione con variabili concomitanti (FMRC) rappresentano un'estensione dei modelli di regressione a misture, in cui le variabili concomitanti influenzano la probabilità di appartenenza ai diversi cluster. A differenza dei modelli standard, in cui i cluster sono determinati esclusivamente dalle variabili predittive, l'inclusione delle variabili concomitanti permette di modellare la probabilità di cluster membership in funzione di caratteristiche aggiuntive.

Nel nostro studio, proviamo ad arricchire il modello aggiungendo come variabili concomitanti alcune delle caratteristiche chimico-fisiche maggiormente correlate con la quantità di caffeina. L'obiettivo è valutare se l'inclusione di queste variabili possa migliorare la capacità predittiva del modello e fornire informazioni aggiuntive sui fattori che influenzano l'assegnazione ai diversi cluster. In particolare, verranno considerate come variabili concomitanti: **Free Acid**, **Fat**, **Trigonelline**, **Water**.

```
set.seed(1000)

X.fmrc <- stepFlexmix(caffine ~ free_acid, k=2, nrep=10,
                      concomitant = FLXPmultinom(~ free_acid+trigonelline+fat+water))
```

```
## 2 : * * * * * * * * * *
```

```
summary(X.fmrc)
```

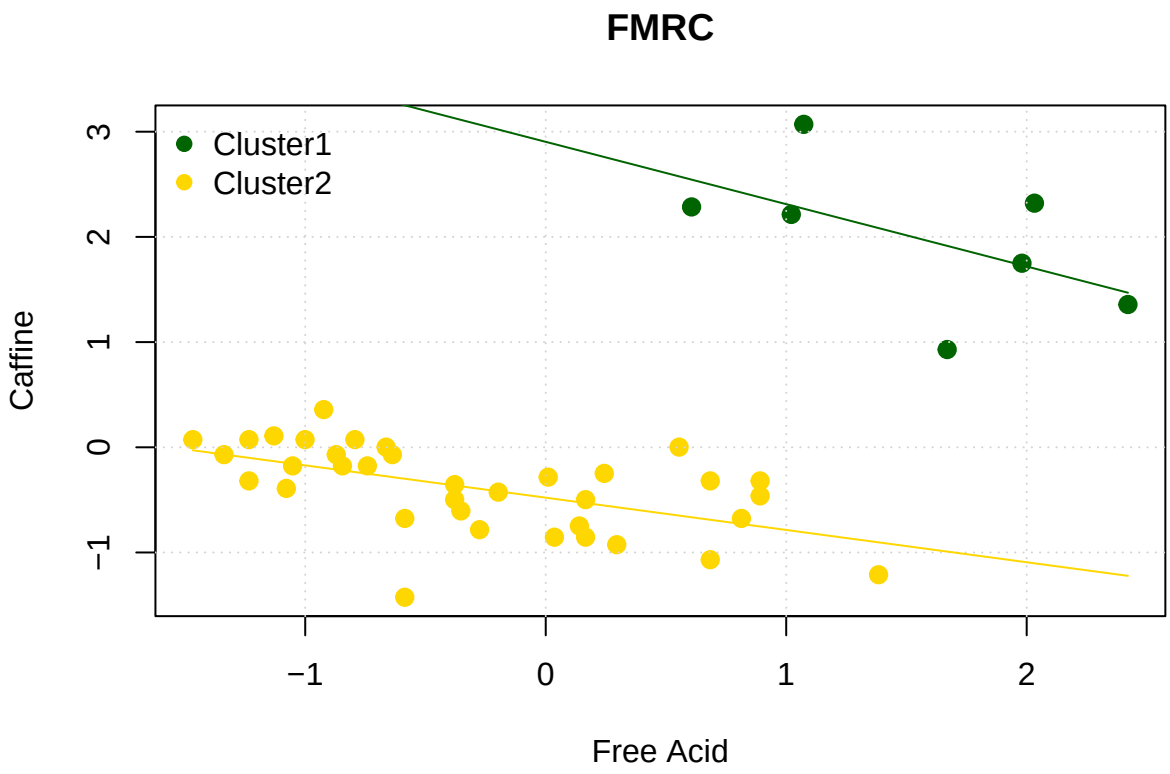
```
##
## Call:
## stepFlexmix(caffine ~ free_acid, concomitant = FLXPmultinom(~free_acid +
##     trigonelline + fat + water), k = 2, nrep = 10)
##
##        prior size post>0 ratio
## Comp.1 0.163    7      7     1
## Comp.2 0.837   36     36     1
##
## 'log Lik.' -17.59427 (df=11)
## AIC: 57.18855   BIC: 76.56175
```

```
pred.X.fmrc <- clusters(X.fmrc)

plot(free_acid, caffine,  col = colors6[pred.X.fmrc], pch = 19, cex = 1.2,
     xlab = "Free Acid", ylab = "Caffine", main = "FMRC")
grid(col = "lightgray", lty = "dotted")
legend("topleft", legend = c("Cluster1", "Cluster2"), col=colors6, pch=19, bty="n")

FMR1<-parameters(X.fmrc)[1:2,1]
FMR2<-parameters(X.fmrc)[1:2,2]

curve(FMR1[1]+FMR1[2]*x,col="darkgreen",add=TRUE)
curve(FMR2[1]+FMR2[2]*x,col="gold",add=TRUE)
```



Il grafico evidenzia un raggruppamento più netto e definito rispetto al precedente, in cui i due gruppi, associabili alle varietà di caffè, sono chiaramente separati in base alla quantità di caffeina e acido libero.

È interessante osservare che, nei due modelli, la pendenza della retta di regressione verde (probabilmente riferibile al raggruppamento della varietà "robusta") presenta un comportamento differente rispetto alla retta gialla (associabile al raggruppamento della varietà "arabica"). Infatti, mentre nel primo modello la retta mostrava una pendenza positiva, indicando una relazione diretta tra caffeina e acido libero, in questo caso la pendenza risulta negativa, suggerendo una relazione inversa tra le due variabili.

Questo cambiamento nella tendenza può essere attribuito all'introduzione delle variabili concomitanti, che probabilmente hanno consentito al modello di cogliere in modo più accurato i fattori che influenzano il contenuto di caffeina nella varietà robusta.

Come in precedenza, osserviamo gli indici di valutazione delle prestazioni:

```
cf.pred1.fmrc<-confmatrix(table(coffee$Variety, clusters(X.fmrc)))
cf.pred1.fmrc
```

```
##
##       1  2
##   2   7  0
##   1   0 36
```

```
accuracy.fmrc <-sum(diag(cf.pred1.fmrc))/nrow(coffee)*100
accuracy.fmrc
```

```
## [1] 100
```

```
ARI.fmrc <-adjustedRandIndex(coffee$Variety,clusters(X.fmrc))
ARI.fmrc
```

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

L'aggiunta delle variabili concomitanti ha modificato notevolmente le prestazioni del modello, portando a un significativo miglioramento nella capacità di classificazione.

Riportiamo ora uno schema riassuntivo per confrontare le prestazioni dei due modelli:

```
error_MR <- (1-accuracy.fmr/100)*100
error_MRC <- (1-accuracy.fmrc/100)*100

table <- data.frame(
  MR = c(accuracy.fmr, error_MR, ARI.fmr),
  MRC = c(accuracy.fmrc, error_MRC, ARI.fmrc)
)

rownames(table) <- c("Accuracy (%)", "Misclassification Error (%)", "Adjusted Rand Index")
print(table)
```

```
##                                    MR MRC
## Accuracy (%)                95.3488372 100
## Misclassification Error (%)  4.6511628   0
## Adjusted Rand Index          0.7882282   1
```

Conclusioni

Lo studio dei dati del caffè ha offerto l'opportunità di esplorare diverse tecniche di analisi statistica e modelli di clustering, consentendo di comprendere meglio i fattori che influenzano la composizione chimico-fisica del caffè.

L'**analisi preliminare** e il **preprocessing** dei dati hanno permesso di identificare le principali variabili rilevanti, evidenziando la necessità di trasformazioni e normalizzazioni per garantire la robustezza dei modelli applicati. L'utilizzo della **PCA** ha fornito una visione semplificata della struttura dei dati, riducendo la dimensionalità senza perdere informazioni rilevanti.

I metodi di **clustering partizionale** (K-means e K-medoids) e **gerarchico** hanno rappresentato un primo approccio per individuare sottogruppi omogenei all'interno del dataset. In particolare, K-means ha mostrato una buona capacità di separare i dati in gruppi distinti, ma con alcune limitazioni nel catturare la complessità delle distribuzioni.

L'introduzione di **modelli misti gaussiani** (GMM) ha permesso di modellare meglio la struttura probabilistica dei dati, evidenziando la presenza di due cluster principali. Il passaggio ai **modelli parsimoniosi GMM** (PGMM) ha migliorato l'interpretabilità, consentendo di individuare modelli più semplici e stabili.

Con l'applicazione di **misture di modelli di regressione** (FMR e FMRC), l'attenzione si è spostata sull'analisi della relazione tra la quantità di **caffeina** e le variabili chimico-fisiche del caffè. L'aggiunta di variabili **concomitanti** ha notevolmente migliorato le prestazioni del modello, con un ARI di 1 e un'accuracy del 100%, dimostrando come fattori aggiuntivi possano influenzare non solo la regressione ma anche l'assegnazione ai cluster.

In conclusione, questo studio ha evidenziato l'importanza di combinare metodi di clustering e modelli di regressione per comprendere fenomeni complessi. L'approccio progressivo adottato — partendo dai metodi partizionali fino ai modelli misti con variabili concomitanti — ha permesso di ottenere risultati significativi e coerenti, migliorando sia la **capacità predittiva** sia la **comprensione delle relazioni** tra le variabili analizzate.