```r
library(readr)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v purrr     1.0.2
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## -- 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 errors
```

```r
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(knitr)
library(class)
library(ggplot2)
library(ggcorrplot)
library(dplyr)
library(e1071)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```r
library(caret)
library(factoextra)
```

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

```r
library(cluster)
library(cowplot)
```

```
##
## Attaching package: 'cowplot'
##
## The following object is masked from 'package:lubridate':
##
##     stamp
```

```r
library(pander)
library(kernlab)
```

```
##
## Attaching package: 'kernlab'
##
## The following object is masked from 'package:purrr':
##
##     cross
##
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```r
library(tidyr)
library(fastDummies)
```

```
## Thank you for using fastDummies!
## To acknowledge our work, please cite the package:
## Kaplan, J. & Schlegel, B. (2023). fastDummies: Fast Creation of Dummy (Binary) Columns and Rows from
```

```r
library(FactoMineR)
```

```r
data = read.csv("/Users/srikanthgembali/Downloads/Cereals.csv")
head(data)
```

```
##                        name mfr type calories protein fat sodium fiber carbo
## 1                 100%_Bran   N    C       70       4   1    130  10.0   5.0
## 2          100%_Natural_Bran   Q    C      120       3   5     15   2.0   8.0
## 3                   All-Bran   K    C       70       4   1    260   9.0   7.0
## 4 All-Bran_with_Extra_Fiber   K    C       50       4   0    140  14.0   8.0
## 5             Almond_Delight   R    C      110       2   2    200   1.0  14.0
## 6   Apple_Cinnamon_Cheerios   G    C      110       2   2    180   1.5  10.5
##   sugars potass vitamins shelf weight cups   rating
## 1      6    280       25     3      1 0.33 68.40297
## 2      8    135        0     3      1 1.00 33.98368
## 3      5    320       25     3      1 0.33 59.42551
## 4      0    330       25     3      1 0.50 93.70491
## 5      8     NA       25     3      1 0.75 34.38484
## 6     10     70       25     1      1 0.75 29.50954
```

```r
summary(data)
```

```
##      name               mfr                type              calories
##  Length:77          Length:77          Length:77          Min.   : 50.0
##  Class :character   Class :character   Class :character   1st Qu.:100.0
##  Mode  :character   Mode  :character   Mode  :character   Median :110.0
##                                                           Mean   :106.9
##                                                           3rd Qu.:110.0
##                                                           Max.   :160.0
```

```
##
##     protein           fat            sodium          fiber
##  Min.   :1.000   Min.   :0.000   Min.   :  0.0   Min.   : 0.000
##  1st Qu.:2.000   1st Qu.:0.000   1st Qu.:130.0   1st Qu.: 1.000
##  Median :3.000   Median :1.000   Median :180.0   Median : 2.000
##  Mean   :2.545   Mean   :1.013   Mean   :159.7   Mean   : 2.152
##  3rd Qu.:3.000   3rd Qu.:2.000   3rd Qu.:210.0   3rd Qu.: 3.000
##  Max.   :6.000   Max.   :5.000   Max.   :320.0   Max.   :14.000
##
##      carbo           sugars           potass          vitamins
##  Min.   : 5.0   Min.   : 0.000   Min.   : 15.00   Min.   :  0.00
##  1st Qu.:12.0   1st Qu.: 3.000   1st Qu.: 42.50   1st Qu.: 25.00
##  Median :14.5   Median : 7.000   Median : 90.00   Median : 25.00
##  Mean   :14.8   Mean   : 7.026   Mean   : 98.67   Mean   : 28.25
##  3rd Qu.:17.0   3rd Qu.:11.000   3rd Qu.:120.00   3rd Qu.: 25.00
##  Max.   :23.0   Max.   :15.000   Max.   :330.00   Max.   :100.00
##  NA's   :1      NA's   :1        NA's   :2
##      shelf           weight          cups           rating
##  Min.   :1.000   Min.   :0.50   Min.   :0.250   Min.   :18.04
##  1st Qu.:1.000   1st Qu.:1.00   1st Qu.:0.670   1st Qu.:33.17
##  Median :2.000   Median :1.00   Median :0.750   Median :40.40
##  Mean   :2.208   Mean   :1.03   Mean   :0.821   Mean   :42.67
##  3rd Qu.:3.000   3rd Qu.:1.00   3rd Qu.:1.000   3rd Qu.:50.83
##  Max.   :3.000   Max.   :1.50   Max.   :1.500   Max.   :93.70
##
```

```
#removing missing values

cereals_data = na.omit(data)
summary(cereals_data)
```

```
##      name               mfr                type              calories
##  Length:74          Length:74          Length:74          Min.   : 50
##  Class :character   Class :character   Class :character   1st Qu.:100
##  Mode  :character   Mode  :character   Mode  :character   Median :110
##                                                           Mean   :107
##                                                           3rd Qu.:110
##                                                           Max.   :160
##     protein           fat        sodium          fiber            carbo
##  Min.   :1.000   Min.   :0   Min.   :  0.0   Min.   : 0.000   Min.   : 5.00
##  1st Qu.:2.000   1st Qu.:0   1st Qu.:135.0   1st Qu.: 0.250   1st Qu.:12.00
##  Median :2.500   Median :1   Median :180.0   Median : 2.000   Median :14.50
##  Mean   :2.514   Mean   :1   Mean   :162.4   Mean   : 2.176   Mean   :14.73
##  3rd Qu.:3.000   3rd Qu.:1   3rd Qu.:217.5   3rd Qu.: 3.000   3rd Qu.:17.00
##  Max.   :6.000   Max.   :5   Max.   :320.0   Max.   :14.000   Max.   :23.00
##      sugars           potass          vitamins          shelf
##  Min.   : 0.000   Min.   : 15.00   Min.   :  0.00   Min.   :1.000
##  1st Qu.: 3.000   1st Qu.: 41.25   1st Qu.: 25.00   1st Qu.:1.250
##  Median : 7.000   Median : 90.00   Median : 25.00   Median :2.000
##  Mean   : 7.108   Mean   : 98.51   Mean   : 29.05   Mean   :2.216
##  3rd Qu.:11.000   3rd Qu.:120.00   3rd Qu.: 25.00   3rd Qu.:3.000
##  Max.   :15.000   Max.   :330.00   Max.   :100.00   Max.   :3.000
##      weight           cups            rating
##  Min.   :0.500   Min.   :0.2500   Min.   :18.04
```

```
##  1st Qu.:1.000   1st Qu.:0.6700   1st Qu.:32.45
##  Median :1.000   Median :0.7500   Median :40.25
##  Mean   :1.031   Mean   :0.8216   Mean   :42.37
##  3rd Qu.:1.000   3rd Qu.:1.0000   3rd Qu.:50.52
##  Max.   :1.500   Max.   :1.5000   Max.   :93.70
```

```
cereals_data = as.data.frame(cereals_data)
cereals_data = cereals_data[, c(4:12,14:16)] #selecting only numerical values
cereals_data = scale(cereals_data)
head(cereals_data)
```

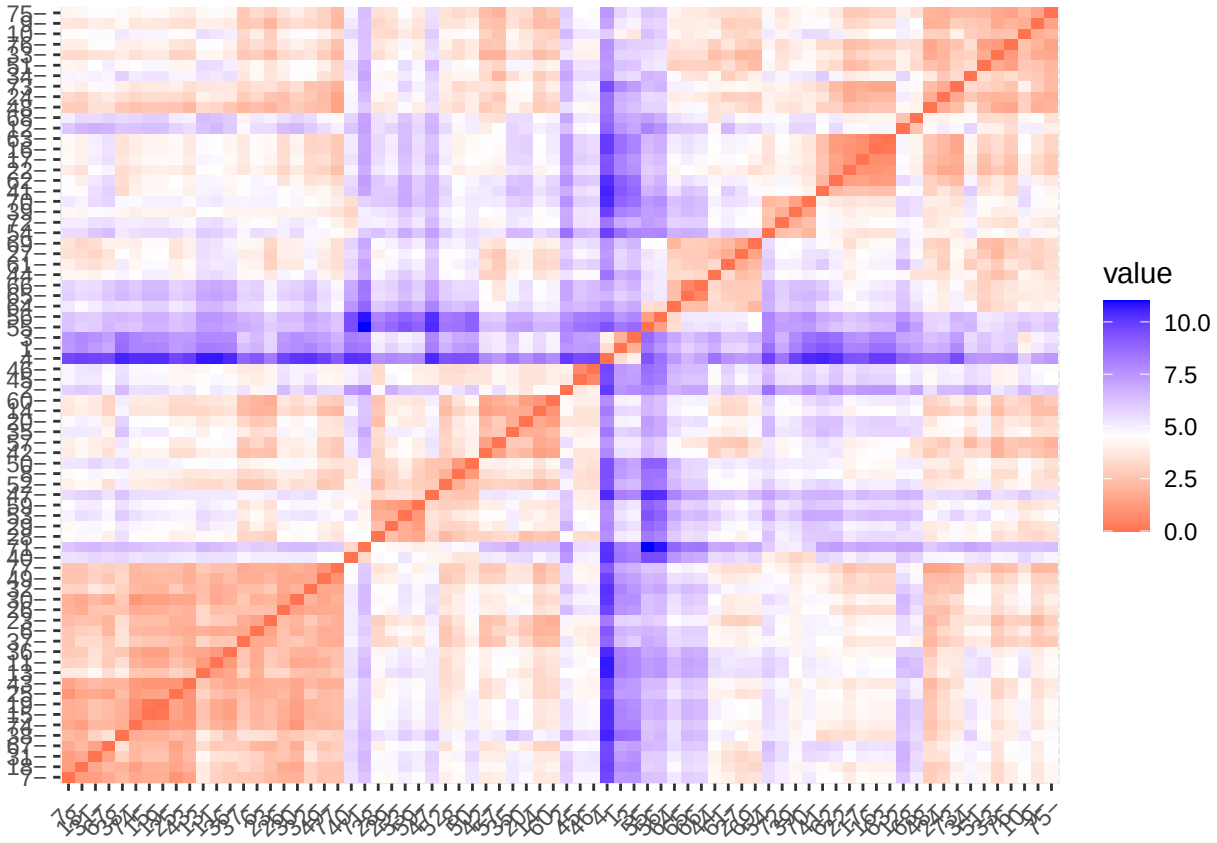
```
##     calories    protein        fat     sodium       fiber      carbo     sugars
## 1 -1.8659155  1.3817478  0.0000000 -0.3910227  3.22866747 -2.5001396 -0.2542051
## 2  0.6537514  0.4522084  3.9728810 -1.7804186 -0.07249167 -1.7292632  0.2046041
## 3 -1.8659155  1.3817478  0.0000000  1.1795987  2.81602258 -1.9862220 -0.4836096
## 4 -2.8737823  1.3817478 -0.9932203 -0.2702057  4.87924705 -1.7292632 -1.6306324
## 6  0.1498180 -0.4773310  0.9932203  0.2130625 -0.27881412 -1.0868662  0.6634132
## 7  0.1498180 -0.4773310 -0.9932203 -0.4514312 -0.48513656 -0.9583868  1.5810314
##       potass   vitamins     weight       cups     rating
## 1  2.5605229 -0.1818422 -0.2008324 -2.0856582  1.8549038
## 2  0.5147738 -1.3032024 -0.2008324  0.7567534 -0.5977113
## 3  3.1248675 -0.1818422 -0.2008324 -2.0856582  1.2151965
## 4  3.2659536 -0.1818422 -0.2008324 -1.3644493  3.6578436
## 6 -0.4022862 -0.1818422 -0.2008324 -0.3038480 -0.9165248
## 7 -0.9666308 -0.1818422 -0.2008324  0.7567534 -0.6553998
```

```
dim(cereals_data)
```

```
## [1] 74 12
```

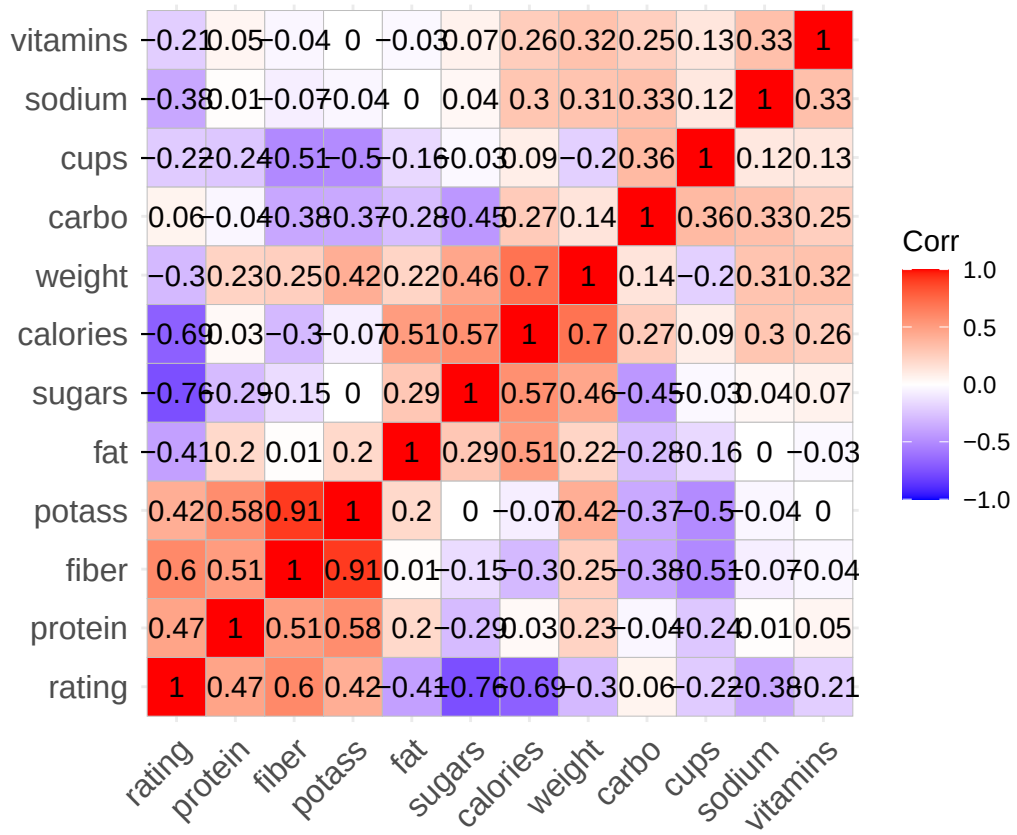
1. Apply hierarchical clustering to the data using Euclidean distance to the normalized measurements.

```
distance_table <- get_dist(cereals_data)
fviz_dist(distance_table)
```



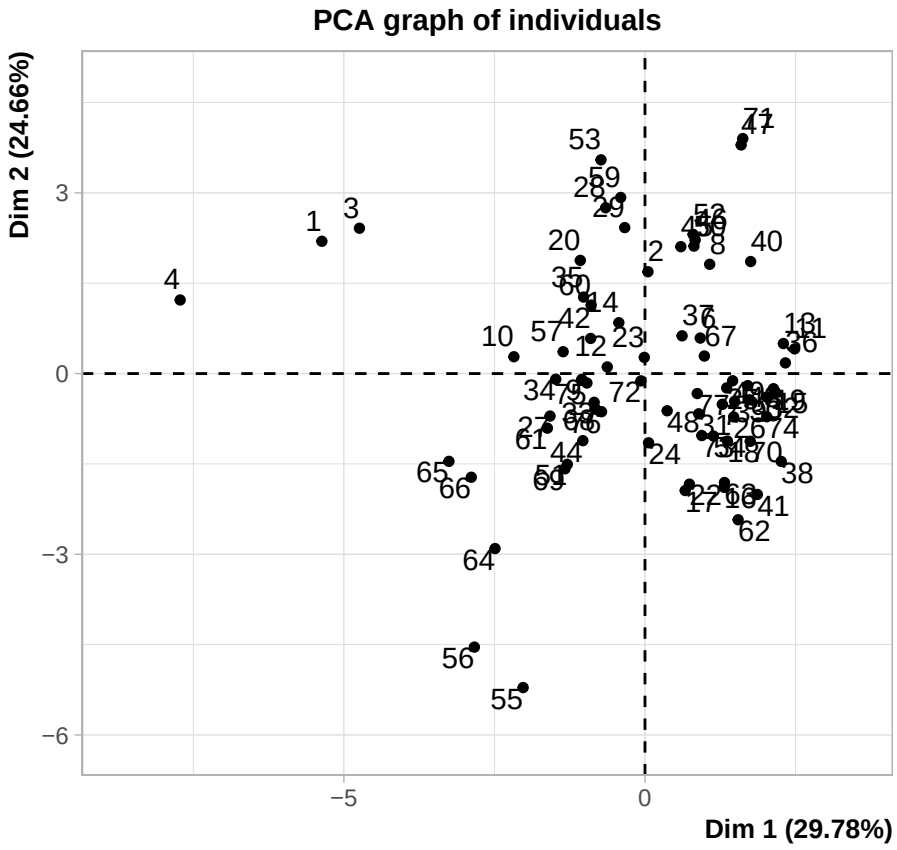
As we can see, the diagonal values are zeros (dark orange) because they represent the distance between each point and itself. The purple and blue colors represent the furthest distance between any pair of observations.

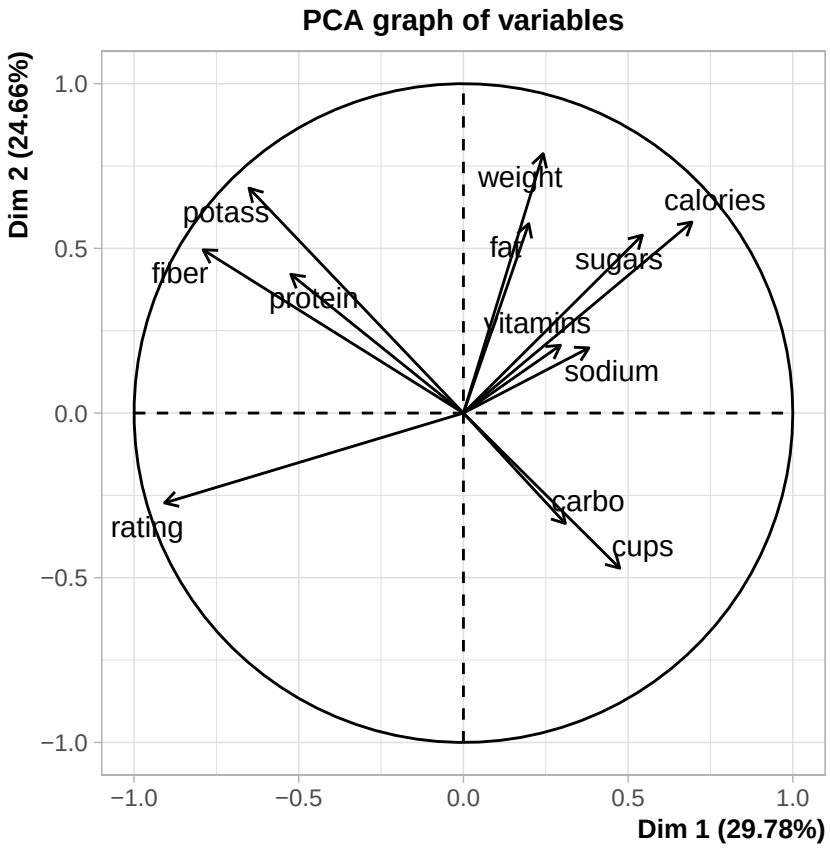
```
corr <- cor(cereals_data)
ggcorrplot(corr, lab = TRUE, hc.order = TRUE, type = "full")
```



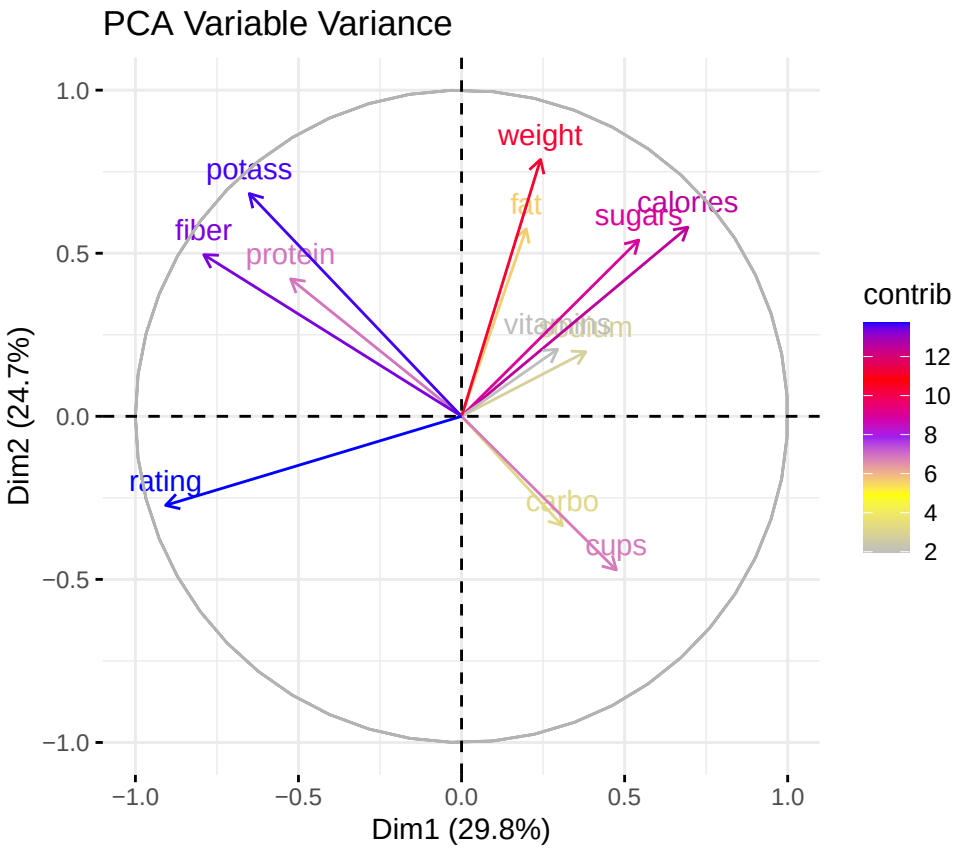
calories, sugar and fats are highly negatively correlated with rating, while Potass is highly positively correlated with fiber and protein.

```
#Trying to Understand the variable variance by performing PCA

pca_cereal <- PCA(cereals_data)
```


PCA graph of individuals
Dim 2 (24.66%)
Dim 1 (29.78%)
3
0
−3
−6
−5
0

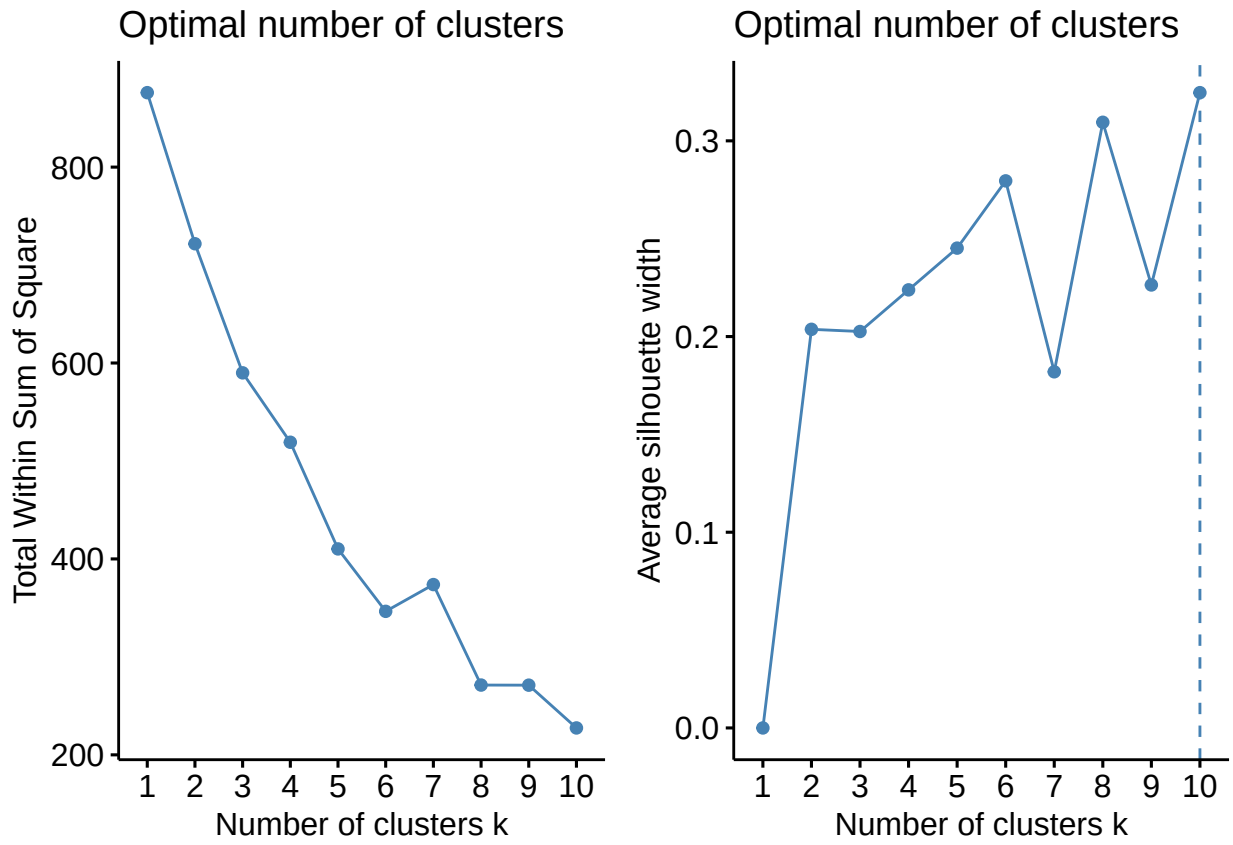



```
var <- get_pca_var(pca_cereal)
fviz_pca_var(pca_cereal, col.var="contrib",
gradient.cols = c("grey","yellow","purple","red","blue"),ggrepel = TRUE ) + labs( title = "PCA Variable
```



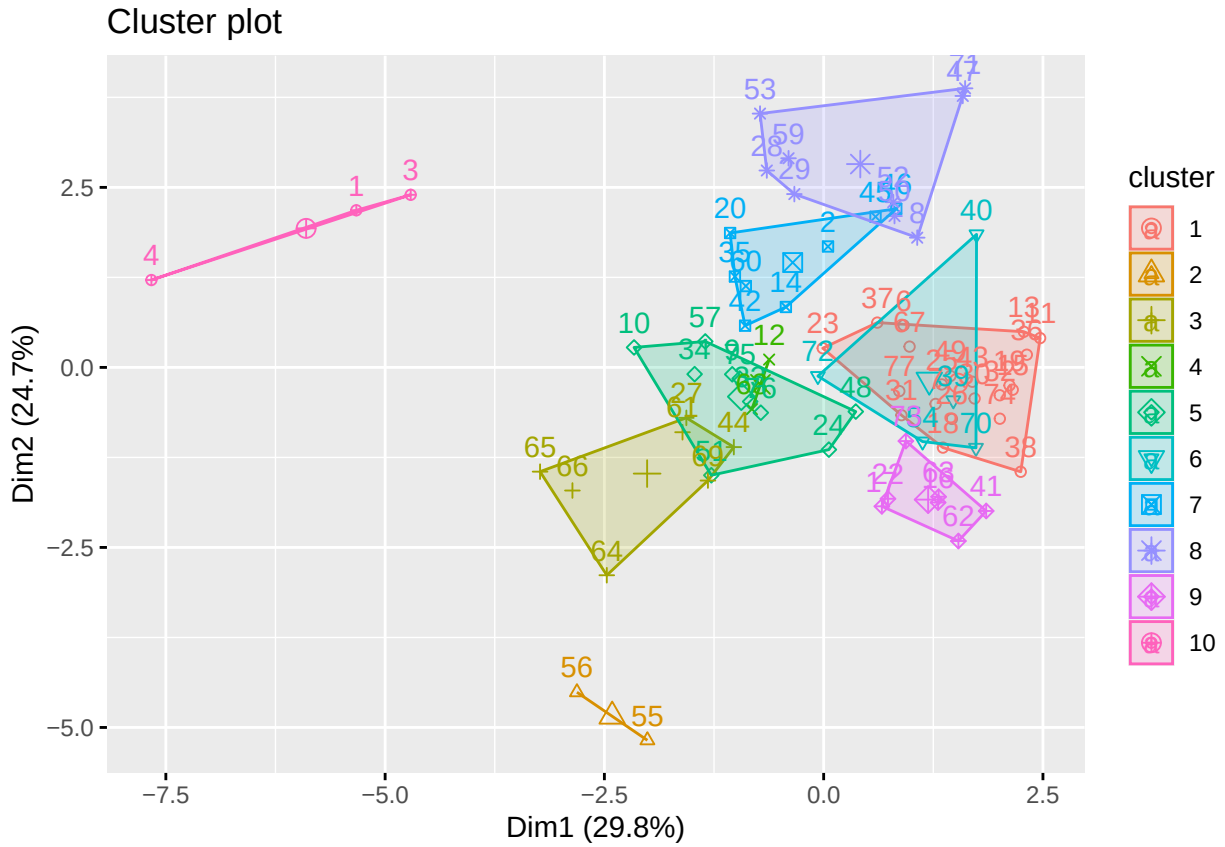
From PCA Variable Variance, we can infer that sugar, calories, protein, potass, and fiber contribute more to the two PCA components/dimensions (variables).

```
Elbow_method = fviz_nbclust(cereals_data, kmeans, method = "wss")
Silhouette = fviz_nbclust(cereals_data, kmeans, method = "silhouette")
plot_grid(Elbow_method, Silhouette, nrow = 1)
```



Optimal number of clusters, K = 10.

```
set.seed(123)
k10 = kmeans(cereals_data, centers = 10, nstart = 25)
fviz_cluster(k10, data = cereals_data)
```



After applying both the silhouette method and elbow method, we obtained a K value of 10, which we used to plot the 10 clusters. However, upon observing the plot, we noticed that some clusters were overlapping, indicating that using only K-means clustering may not be the best option for optimization. Therefore, we will apply hierarchical clustering to determine an optimal number of clusters.

Use Agnes to compare the clustering from single linkage, complete linkage, average linkage, and Ward. Choose the best method

```
hc_single = agnes(distance_table, method = "single")
hc_complete = agnes(distance_table, method = "complete")
hc_average = agnes(distance_table, method = "average")
hc_ward = agnes(distance_table, method = "ward")

print(hc_single$ac)
```

```
## [1] 0.6072384
```

```
print(hc_complete$ac)
```

```
## [1] 0.8469328
```

```
print(hc_average$ac)
```

```
## [1] 0.7881955
```

```r
print(hc_ward$ac)
```

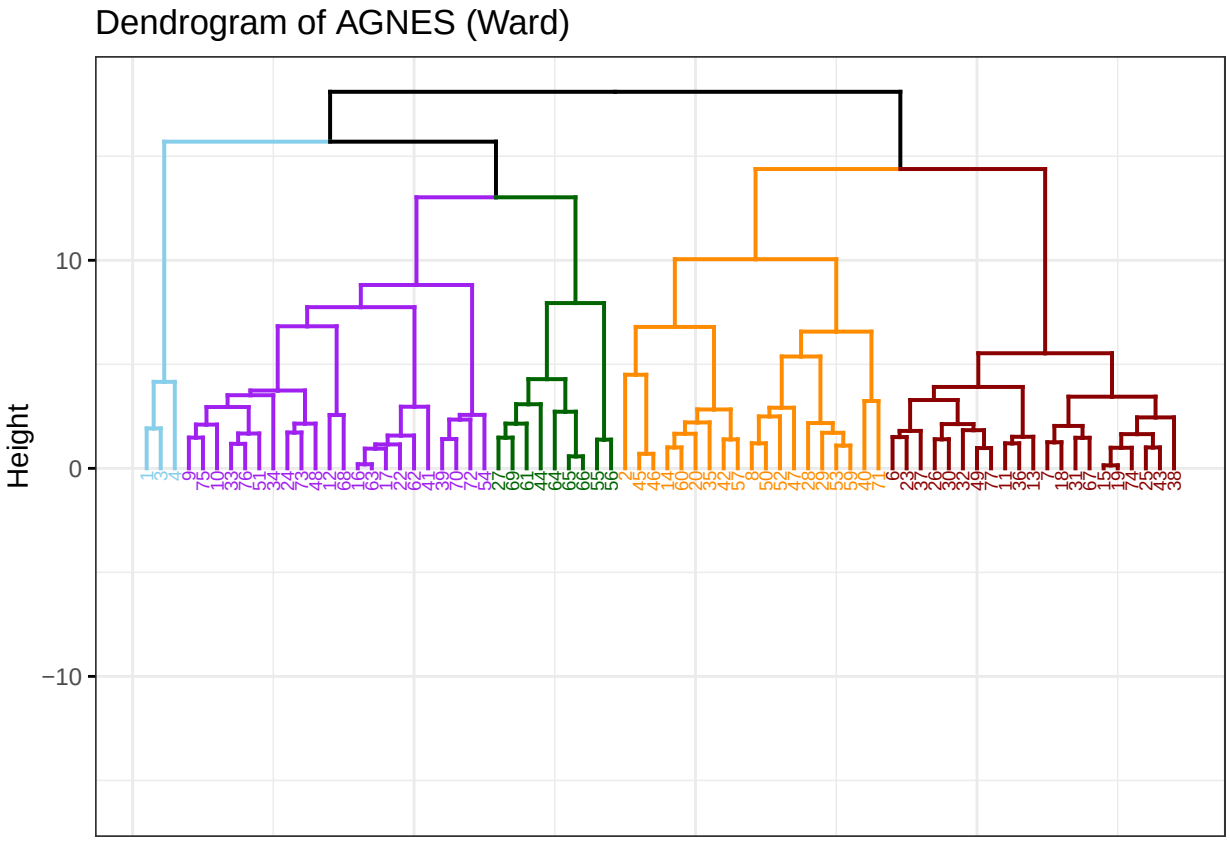
```
## [1] 0.9087265
```

The best agglomerative (AGNES) linkage to use is the Ward linkage, which gives 90.87% accuracy.

2. How many clusters would you choose?

```r
#Utilizing the Ward linkage, 5 clusters seem to be a good number to group the data
fviz_dend(hc_ward, k = 5, main = "Dendrogram of AGNES (Ward)",cex = 0.5, k_colors = c("skyblue", "purple
```

```
## 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.
```




```r
cereals_data_2 = cutree(hc_ward, k = 5)
clustered_df = as.data.frame(cbind (cereals_data, cereals_data_2))
```

3. Comment on the structure of the clusters and on their stability. Hint: To check stability, partition the data and see how well clusters formed based on one part apply to the other part.

Cluster partition A

```
#We will partition the data into two groups
A = cereals_data [1:50,]
summary(A)
```

```
##     calories          protein             fat              sodium
##  Min.   :-2.8738   Min.   :-1.40687   Min.   :-0.9932   Min.   :-1.9616
##  1st Qu.:-0.3541   1st Qu.:-0.47733   1st Qu.:-0.9932   1st Qu.:-0.2702
##  Median : 0.1498   Median :-0.01256   Median : 0.0000   Median : 0.1527
##  Mean   : 0.1700   Mean   :-0.01256   Mean   : 0.1986   Mean   : 0.1055
##  3rd Qu.: 0.6538   3rd Qu.: 0.45221   3rd Qu.: 0.9932   3rd Qu.: 0.6661
##  Max.   : 2.6695   Max.   : 3.24083   Max.   : 3.9729   Max.   : 1.5420
##      fiber              carbo              sugars            potass
##  Min.   :-0.89778   Min.   :-2.5001   Min.   :-1.6306   Min.   :-1.10772
##  1st Qu.:-0.79462   1st Qu.:-0.7014   1st Qu.:-0.4836   1st Qu.:-0.80791
##  Median :-0.17565   Median :-0.2518   Median : 0.3193   Median :-0.12011
##  Mean   : 0.03067   Mean   :-0.1541   Mean   : 0.2046   Mean   : 0.01251
##  3rd Qu.: 0.34015   3rd Qu.: 0.5191   3rd Qu.: 0.8928   3rd Qu.: 0.35605
##  Max.   : 4.87925   Max.   : 1.8682   Max.   : 1.8104   Max.   : 3.26595
##     vitamins           weight             cups              rating
##  Min.   :-1.30320   Min.   :-0.2008   Min.   :-2.42505   Min.   :-1.7336
##  1st Qu.:-0.18184   1st Qu.:-0.2008   1st Qu.:-0.64324   1st Qu.:-0.8384
##  Median :-0.18184   Median :-0.2008   Median :-0.30385   Median :-0.3765
##  Mean   :-0.06971   Mean   : 0.0977   Mean   :-0.05015   Mean   :-0.1857
##  3rd Qu.:-0.18184   3rd Qu.:-0.2008   3rd Qu.: 0.75675   3rd Qu.: 0.2479
##  Max.   : 3.18224   Max.   : 3.0583   Max.   : 2.87796   Max.   : 3.6578
```

```
B = cereals_data [51:74,]
summary(B)
```

```
##     calories          protein             fat              sodium
##  Min.   :-2.8738   Min.   :-1.40687   Min.   :-0.9932   Min.   :-1.9616
##  1st Qu.:-0.8580   1st Qu.:-0.47733   1st Qu.:-0.9932   1st Qu.:-1.8257
##  Median :-0.3541   Median :-0.01256   Median : 0.0000   Median : 0.3943
##  Mean   :-0.3541   Mean   : 0.02617   Mean   :-0.4138   Mean   :-0.2199
##  3rd Qu.: 0.1498   3rd Qu.: 0.45221   3rd Qu.: 0.0000   3rd Qu.: 0.6359
##  Max.   : 1.6616   Max.   : 3.24083   Max.   : 0.9932   Max.   : 1.9045
##      fiber              carbo              sugars            potass
##  Min.   :-0.89778   Min.   :-1.4723   Min.   :-1.6306   Min.   :-1.17826
##  1st Qu.:-0.58830   1st Qu.:-0.2518   1st Qu.:-0.9998   1st Qu.:-0.77264
##  Median :-0.07249   Median : 0.3264   Median :-0.9424   Median :-0.08484
##  Mean   :-0.06389   Mean   : 0.3211   Mean   :-0.4263   Mean   :-0.02606
##  3rd Qu.: 0.34015   3rd Qu.: 1.1615   3rd Qu.: 0.2046   3rd Qu.: 0.25024
##  Max.   : 1.57809   Max.   : 2.1251   Max.   : 1.8104   Max.   : 2.27835
##     vitamins           weight             cups              rating
##  Min.   :-1.3032   Min.   :-3.4600   Min.   :-1.3644   Min.   :-1.0417
##  1st Qu.:-0.1818   1st Qu.:-0.2008   1st Qu.:-0.6432   1st Qu.:-0.2374
##  Median :-0.1818   Median :-0.2008   Median : 0.7568   Median : 0.1395
##  Mean   : 0.1452   Mean   :-0.2035   Mean   : 0.1045   Mean   : 0.3869
##  3rd Qu.:-0.1818   3rd Qu.:-0.2008   3rd Qu.: 0.7568   3rd Qu.: 0.9954
##  Max.   : 3.1822   Max.   : 3.0583   Max.   : 1.3083   Max.   : 2.2874
```

```r
# Computing the distances
distance_A = get_dist(A)

# Compute with AGNES and with different linkage methods For A data
hc_single_A = agnes(distance_A, method = "single")
hc_complete_A = agnes(distance_A, method = "complete")
hc_average_A = agnes(distance_A, method = "average")
hc_ward_A = agnes(distance_A, method = "ward")

print(hc_single_A$ac)
```

```
## [1] 0.573335
```

```r
print(hc_complete_A$ac)
```

```
## [1] 0.8315788
```

```r
print(hc_average_A$ac)
```

```
## [1] 0.7602929
```

```r
print(hc_ward_A$ac)
```

```
## [1] 0.8892351
```

The best linkage is Ward with 88.92% accuracy for A

```r
# Computing the distances
distance_B = get_dist(B)

# Compute with AGNES and with different linkage methods For A data
hc_single_B = agnes(distance_B, method = "single")
hc_complete_B = agnes(distance_B, method = "complete")
hc_average_B = agnes(distance_B, method = "average")
hc_ward_B = agnes(distance_B, method = "ward")

print(hc_single_B$ac)
```

```
## [1] 0.5445689
```

```r
print(hc_complete_B$ac)
```

```
## [1] 0.8224932
```

```r
print(hc_average_B$ac)
```

```
## [1] 0.7089309
```

```
print(hc_ward_B$ac)
```

```
## [1] 0.857288
```

The best linkage is Ward with 85.72% accuracy for B

Use the cluster centroids from A to assign each record in partition B (each record is assigned to the cluster with the closest centroid).

```
Clustered_df_A = cutree (hc_ward_A, k=5)
Clusters_A = as.data.frame(cbind(A, Clustered_df_A))
nrow(Clusters_A)
```

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

```
Clust_1 = colMeans (Clusters_A [Clusters_A$ Clustered_df_A == "1" ,])

Clustered_df_B = cutree (hc_ward_B, k=5)
Clusters_B = as.data.frame(cbind(B, Clustered_df_B))
nrow(Clusters_B)
```

```
## [1] 24
```

```
Clust_2 = colMeans (Clusters_B [Clusters_B$ Clustered_df_B == "1" ,])
```

```
Centroid = rbind(Clust_1, Clust_2)
Centroid
```

```
##          calories   protein        fat    sodium    fiber      carbo     sugars
## Clust_1 -2.201871 1.3817478 -0.3310734 0.1727901 3.641312 -2.0718749 -0.7894824
## Clust_2  0.989707 0.4522084  0.0000000 0.4546965 1.165443 -0.3588163  1.4280950
##           potass   vitamins     weight        cups    rating Clustered_df_A
## Clust_1 2.983781 -0.1818422 -0.2008324 -1.84525525  2.242648              1
## Clust_2 2.043207  0.9395180  2.3195559 -0.06344498 -0.508841              1
```

At an overall level, both clusters seem fine, but there is also a slight difference. Cluster_1 has higher fiber, protein and potassium content compared to Cluster_2, which may suggest that cereals in this cluster are healthier or more nutrient-dense. Cluster_2 has a higher sugar content compared to Cluster_1, which may suggest that cereals in this cluster are less healthy or have more added sugars.

Assess how consistent the cluster assignments are compared to the assignments based on all the data.

We are comparing the mean values of each feature for the two clusters identified in the data. These centroids can be used to compare the features of the two clusters and explore differences or similarities between them. Here, we observe that Cluster_1 has higher fiber, protein and potassium content compared to Cluster_2, suggesting that cereals in this cluster are healthier or more nutrient-dense. Conversely, Cluster_2 exhibits a higher sugar content compared to Cluster_1, implying that cereals in this cluster are less healthy or contain more added sugars, hence the lower rating of Cluster 2 compared to Cluster 1.

4. The elementary public schools would like to choose a set of cereals to include in their daily cafeterias. Every day a different cereal is offered, but all cereals should support a healthy diet. For this goal, you are requested to find a cluster of "healthy cereals." Should the data be normalized? If not, how should they be used in the cluster analysis?

```r
#To analyze which group of cereals are healthier to distribute daily in cafeterias in elementary public

data = na.omit(data)

Healthy_data = as.data.frame(cbind (data, cereals_data_2))
Healthy_data_sort = Healthy_data[order(Healthy_data$cereals_data_2),c(1,17)]
Count_cluster = Healthy_data_sort %>% group_by(cereals_data_2) %>% summarise(count = n())
print(Count_cluster)
```

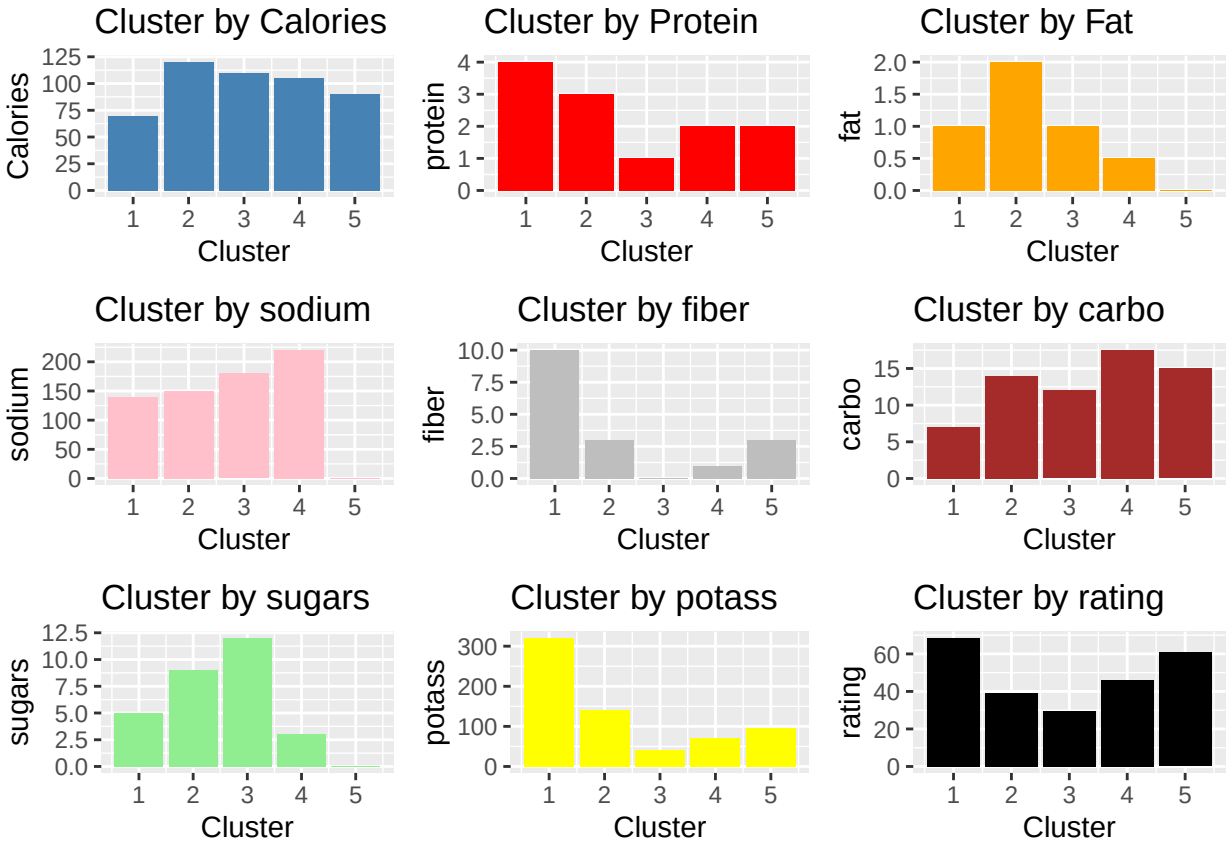
```
## # A tibble: 5 x 2
##   cereals_data_2 count
##            <int> <int>
## 1              1     3
## 2              2    19
## 3              3    21
## 4              4    22
## 5              5     9
```

```r
#Summary table showing the median of each variable
Healthy_data_Var = Healthy_data [,4:17]
cluster_table = Healthy_data_Var %>% group_by(cereals_data_2) %>% summarize(across(.cols = everything()
print(cluster_table)
```

```
## # A tibble: 5 x 14
##   cereals_data_2 calories protein   fat sodium fiber carbo sugars potass
##            <int>    <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>  <dbl>
## 1              1       70       4   1      140    10   7        5    320
## 2              2      120       3   2      150     3  14        9    140
## 3              3      110       1   1      180     0  12       12     40
## 4              4      105       2   0.5    220     1  17.5      3     70
## 5              5       90       2   0        0     3  15        0     95
## # i 5 more variables: vitamins <dbl>, shelf <dbl>, weight <dbl>, cups <dbl>,
## #   rating <dbl>
```

```r
calories = ggplot(cluster_table, aes(x = cereals_data_2, y = calories)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(x = "Cluster", y = "Calories") +
  ggtitle("Cluster by Calories")

protein = ggplot(cluster_table, aes(x = cereals_data_2, y = protein)) +
  geom_bar(stat = "identity", fill = "red") +
  labs(x = "Cluster", y = "protein") +
  ggtitle("Cluster by Protein")

fat = ggplot(cluster_table, aes(x = cereals_data_2, y = fat)) +
  geom_bar(stat = "identity", fill = "orange") +
  labs(x = "Cluster", y = "fat") +
  ggtitle("Cluster by Fat")

sodium = ggplot(cluster_table, aes(x = cereals_data_2, y = sodium)) +
  geom_bar(stat = "identity", fill = "pink") +
  labs(x = "Cluster", y = "sodium") +
```

```
  ggtitle("Cluster by sodium")

fiber = ggplot(cluster_table, aes(x = cereals_data_2, y = fiber)) +
  geom_bar(stat = "identity", fill = "gray") +
  labs(x = "Cluster", y = "fiber") +
  ggtitle("Cluster by fiber")

carbo = ggplot(cluster_table, aes(x = cereals_data_2, y = carbo)) +
  geom_bar(stat = "identity", fill = "brown") +
  labs(x = "Cluster", y = "carbo") +
  ggtitle("Cluster by carbo")

sugars = ggplot(cluster_table, aes(x = cereals_data_2, y = sugars)) +
  geom_bar(stat = "identity", fill = "lightgreen") +
  labs(x = "Cluster", y = "sugars") +
  ggtitle("Cluster by sugars")

potass = ggplot(cluster_table, aes(x = cereals_data_2, y = potass)) +
  geom_bar(stat = "identity", fill = "yellow") +
  labs(x = "Cluster", y = "potass") +
  ggtitle("Cluster by potass")

rating = ggplot(cluster_table, aes(x = cereals_data_2, y = rating)) +
  geom_bar(stat = "identity", fill = "black") +
  labs(x = "Cluster", y = "rating") +
  ggtitle("Cluster by rating")

plot_grid(calories, protein, fat, sodium, fiber, carbo, sugars, potass, rating)
```



Based on the graphs, we can see that Cluster 1 has the lowest values for calories, fat, and sugars, and the highest values for protein, fiber, and vitamins, which suggests that it may contain cereals that are generally considered healthier options. This is reflected in its very high rating as well. However, Cluster 1 does not satisfy the need for a different cereal per day, as per our client's request. Therefore, we also recommend Cluster 5 to fulfill this requirement. Cluster 5 has zero fats, zero sugars, and the second-lowest number of calories after Cluster 1. Additionally, it boasts a good amount of proteins and fiber.

On the other hand, Cluster 3 exhibits the highest values for calories and sugars, and the lowest values for protein, fiber, and vitamins, suggesting that it may contain cereals that are generally considered less healthy. We observed a similar insight from our correlation plot: higher sugar content correlates with lower ratings, indicating lower healthiness. However, it's important to note that this is just a general observation, and individual cereals within each cluster may vary in terms of their nutritional value.