Métodos No Supervisados

Métodos No Supervisados

Análisis exploratorio de datos

Se importa la data y se visualiza su composición, se cambian variables y sus nombres y se quitan outliers:

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.1     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.0
v ggplot2   3.4.2     v tibble    3.2.1
v lubridate 1.9.2     v tidyr     1.3.0
v purrr     1.0.1
-- 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 beco
```

```
library(caret)
```

```
Loading required package: lattice

Attaching package: 'caret'

The following object is masked from 'package:purrr':

    lift
```

```r
library(ggplot2)
library(e1071)
library(DataExplorer)
library(caTools)
library(plotly)
```

```
Attaching package: 'plotly'

The following object is masked from 'package:ggplot2':

    last_plot

The following object is masked from 'package:stats':

    filter

The following object is masked from 'package:graphics':

    layout
```

```r
library(readr)
library(pROC)
```

```
Type 'citation("pROC")' for a citation.

Attaching package: 'pROC'

The following objects are masked from 'package:stats':

    cov, smooth, var
```

```r
library(MASS)
```

```
Attaching package: 'MASS'

The following object is masked from 'package:plotly':
```

```
    select

The following object is masked from 'package:dplyr':

    select
```

```
library(dplyr)
library(readr)
library(factoextra)
```

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

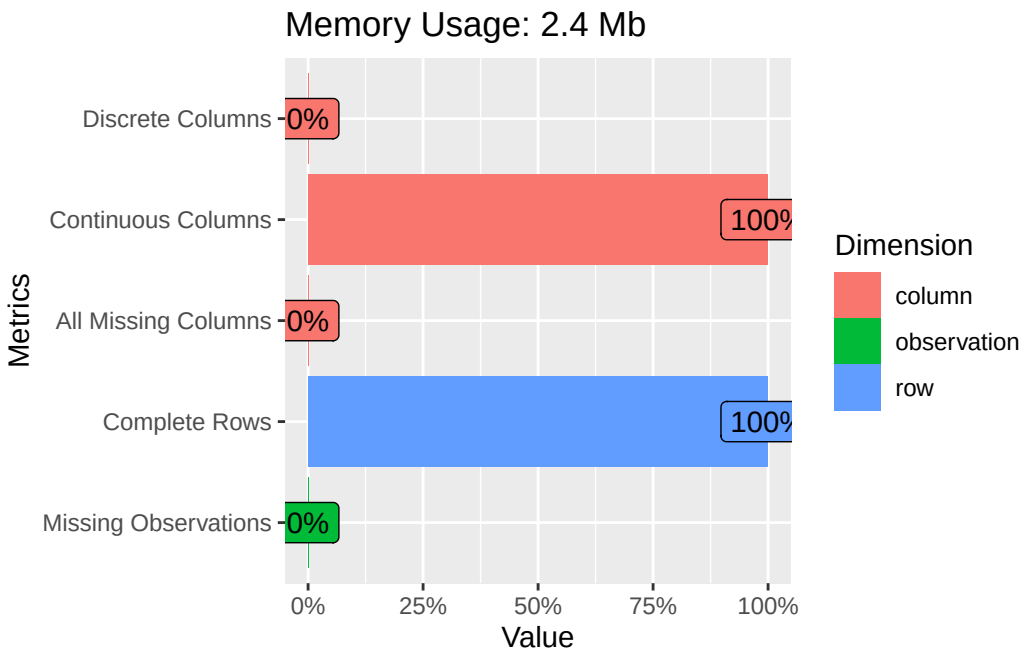
```
library(ggpubr)
library(scatterplot3d)
library(FactoMineR)

# Importación de los datos

data <- read_csv("Steel_industry_data.csv",
                 col_types = cols(date = col_datetime(format = "%d/%m/%Y %H:%M")))
View(data)

data <- as.data.frame(data)

data <- data[, -c(10)]

data <- data[,-c(1)]

data <- data %>%
  mutate(Load_Type = as.character(Load_Type)) %>%
  mutate(Load_Type = case_when(
    Load_Type == "Light_Load" ~ as.numeric("1"),
    Load_Type == "Medium_Load" ~ as.numeric("2"),
    Load_Type == "Maximum_Load" ~ as.numeric("3"),
    TRUE ~ as.numeric(Load_Type)
  ))
```

```
Warning: There was 1 warning in `mutate()`.
i In argument: `Load_Type = case_when(...)`.
Caused by warning:
! NAs introducidos por coerción
```

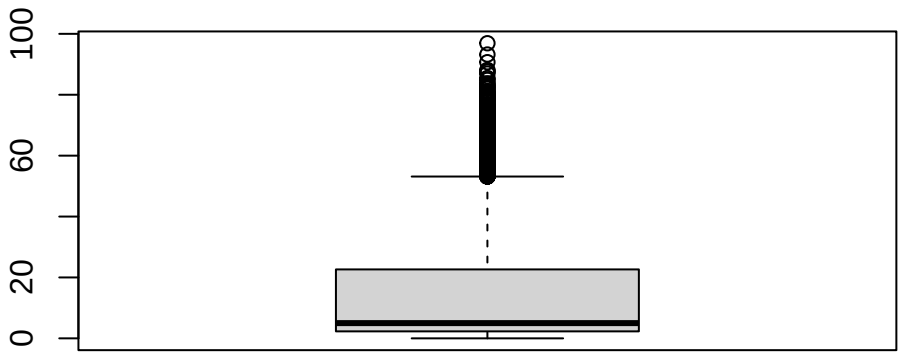
```r
data <- data %>%
  mutate(WeekStatus = as.character(WeekStatus)) %>%
  mutate( WeekStatus = case_when(
    WeekStatus == "Weekday" ~ as.numeric("1"),
    WeekStatus == "Weekend" ~ as.numeric("0"),
    TRUE ~ as.numeric(WeekStatus)
  ))
```

```
Warning: There was 1 warning in `mutate()`.
i In argument: `WeekStatus = case_when(...)`.
Caused by warning:
! NAs introducidos por coerción
```

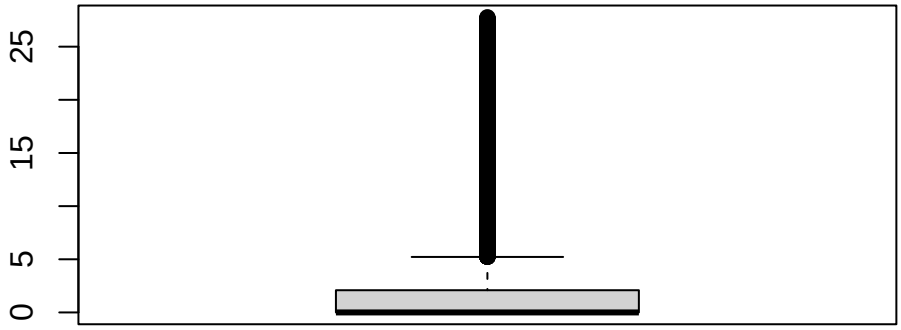
```r
plot_intro(data)
```



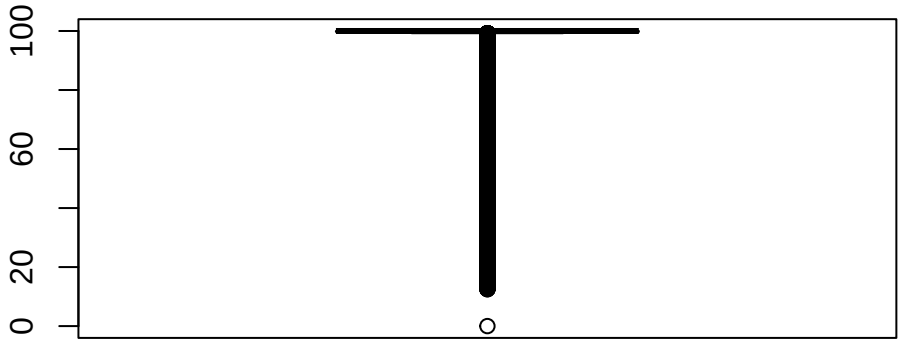

```r
boxplot(data$Lagging_Current_Reactive.Power_kVarh)
```



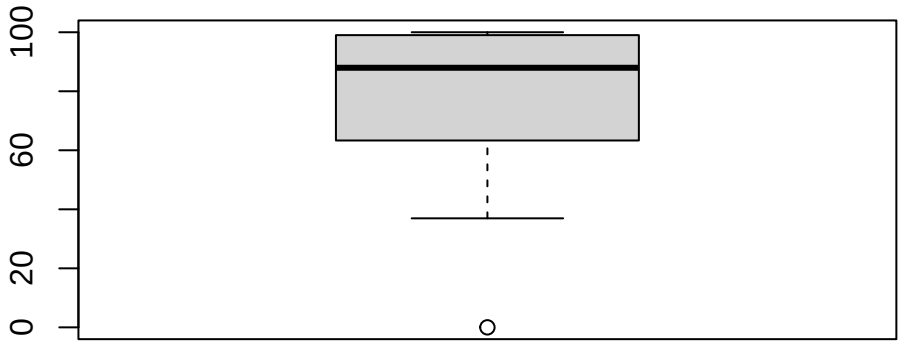

```
boxplot(data$Leading_Current_Reactive_Power_kVarh)
```



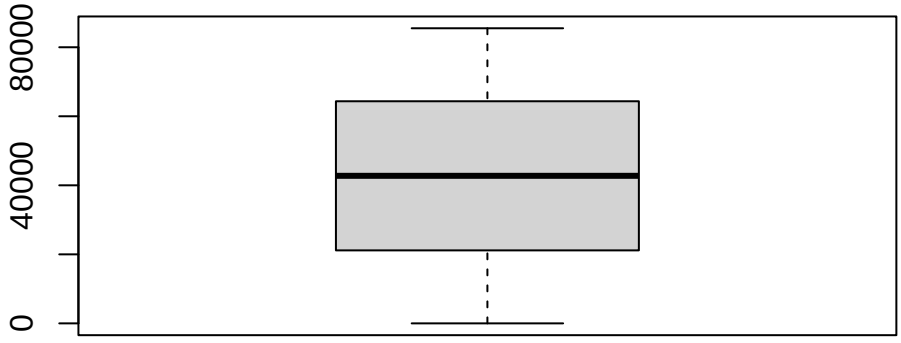
```
boxplot(data$Leading_Current_Power_Factor)
```



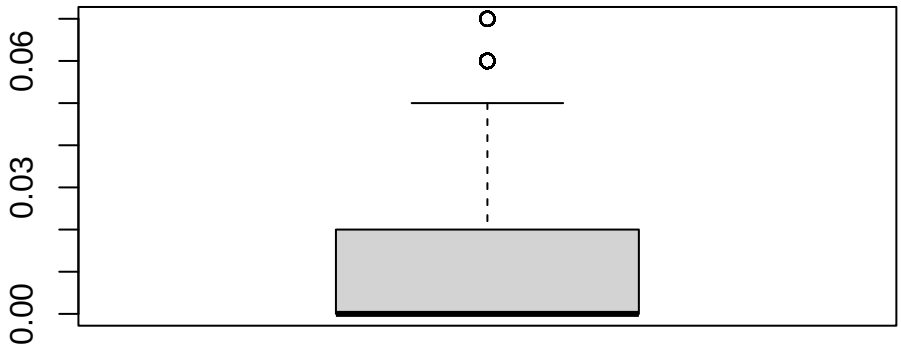

```
boxplot(data$Lagging_Current_Power_Factor)
```



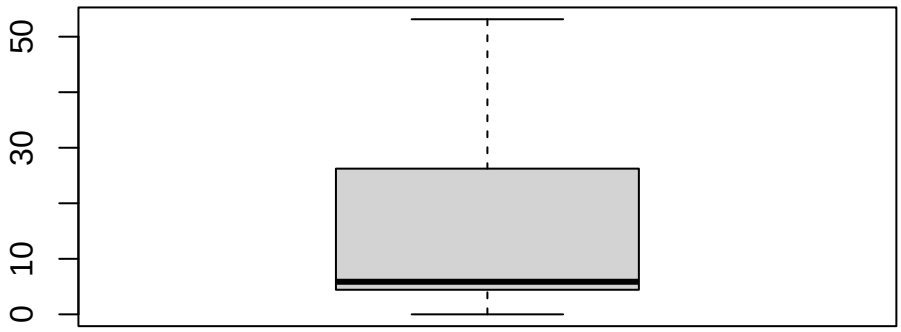

```
boxplot(data$NSM)
```



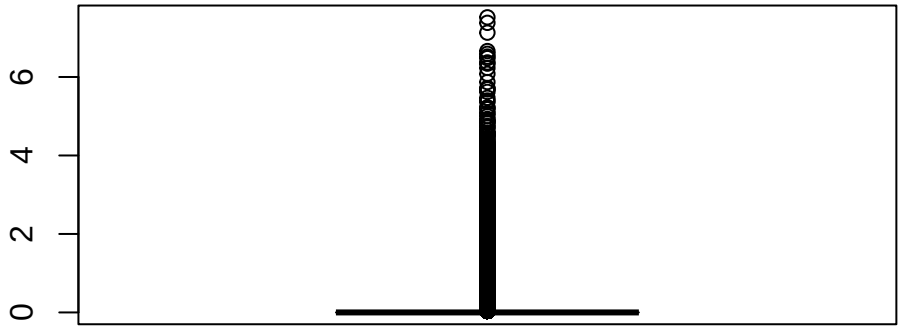
```r
boxplot(data$`CO2(tCO2)`)
```



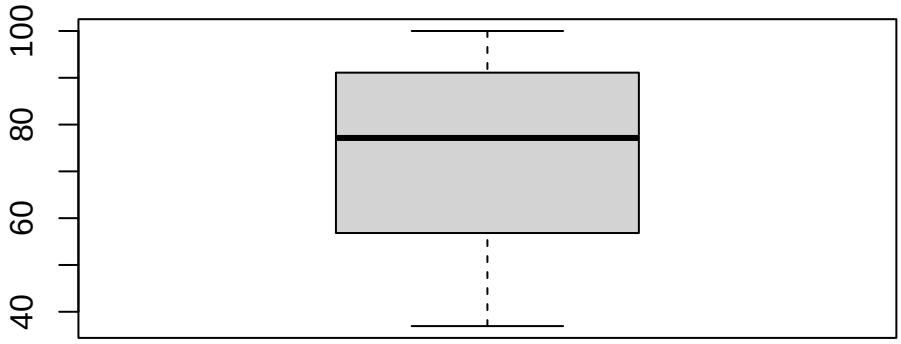

```r
for (i in c( "Lagging_Current_Reactive.Power_kVarh", "Leading_Current_Power_Factor","Laggi
{
  outliers <- boxplot.stats(data[[i]])$out
  data[[i]][data[[i]] %in% outliers] <- NA
}
data <- filter_if(data, is.numeric , all_vars(!is.na(.)))


boxplot(data$Lagging_Current_Reactive.Power_kVarh)
```



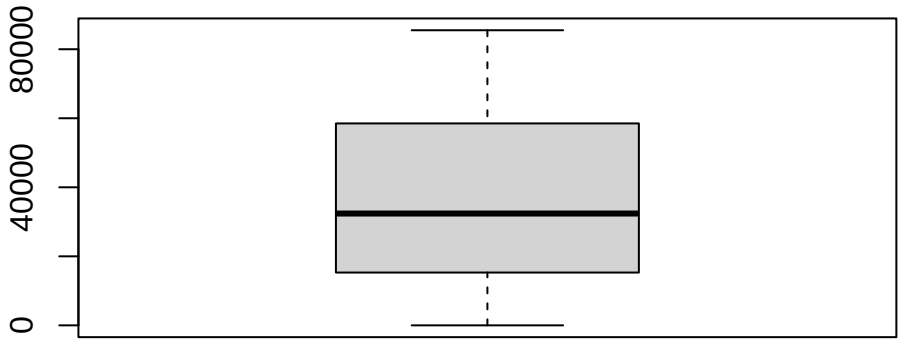

```
boxplot(data$Leading_Current_Reactive_Power_kVarh)
```



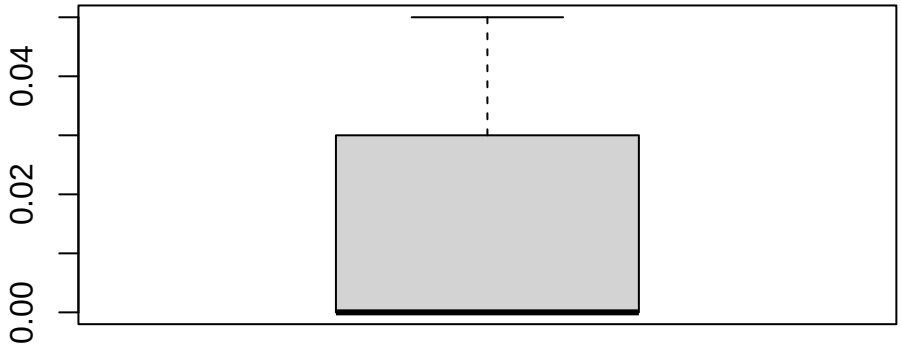
```
boxplot(data$Lagging_Current_Power_Factor)
```




```
boxplot(data$NSM)
```



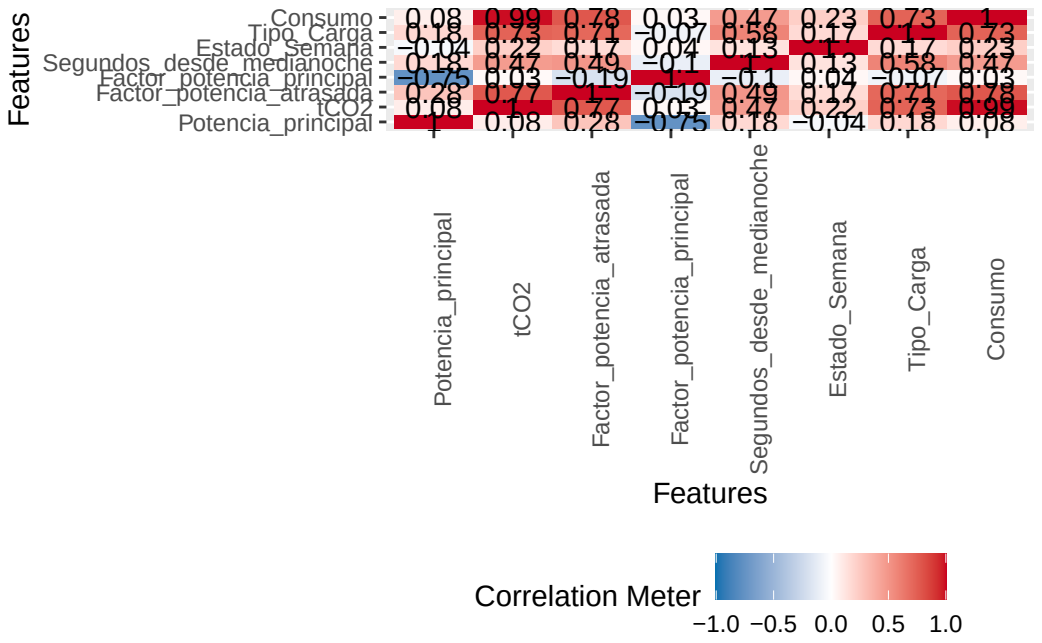

```
boxplot(data$`CO2(tCO2)`)
```



```
data <- as.data.frame(cbind(scale(data[ ,c(2:9)]), data$Usage_kWh))

data_sc <-data

colnames(data_sc) <- c("Potencia_atrasada",
                       "Potencia_principal",
                       "tCO2",
                       "Factor_potencia_atrasada",
                       "Factor_potencia_principal",
                       "Segundos_desde_medianoche",
                       "Estado_Semana","Tipo_Carga","Consumo")
View(data_sc)
```

También, se añade al análisis exploratorio el PCA:

```
#ELIMINACIÓN DE VARIABLE "CONSUMO"
features <- data_sc[,2:9]
consumo <- data_sc[1,]


view(features)

#CORRELACIÓN ENTRE LOS ATRIBUTOS


plot_correlation(features)
```




```
#PCA estandarización
```

```
apply(X = features, MARGIN = 2, FUN = mean)
```

```
      Potencia_principal                    tCO2  Factor_potencia_atrasada
           -2.240994e-17           -2.313641e-17           -2.624214e-17
Factor_potencia_principal Segundos_desde_medianoche            Estado_Semana
            3.161361e-14            -1.203713e-16           -1.748616e-17
              Tipo_Carga                  Consumo
            4.273954e-17            3.036808e+01
```

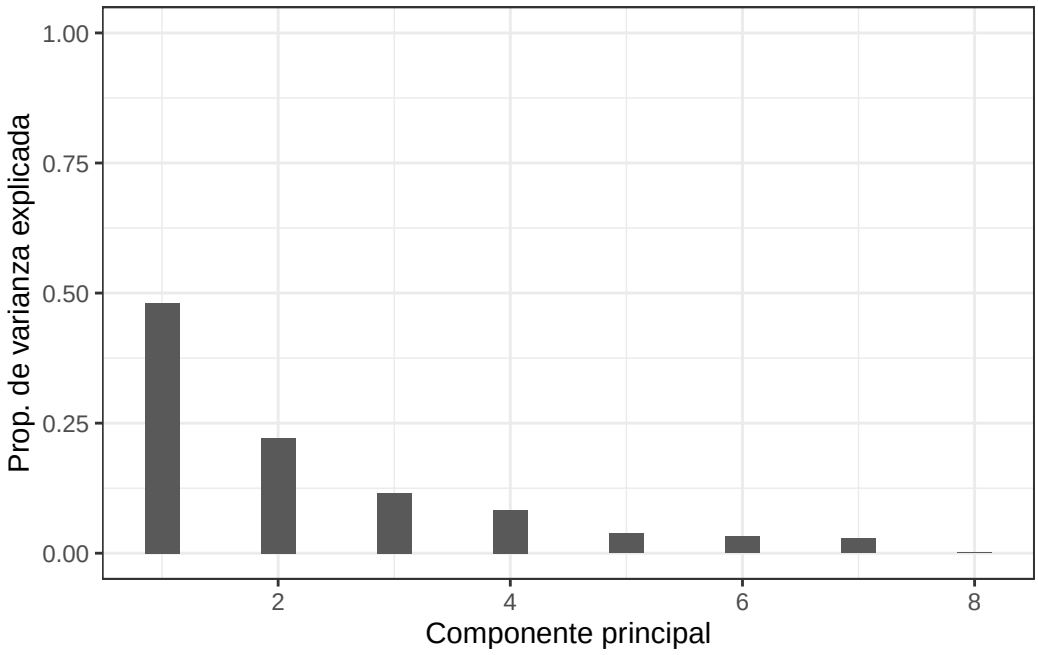
```
apply(X = features, MARGIN = 2, FUN = var)
```

```
      Potencia_principal                    tCO2  Factor_potencia_atrasada
                   1.000                   1.000                   1.000
Factor_potencia_principal Segundos_desde_medianoche            Estado_Semana
                   1.000                   1.000                   1.000
              Tipo_Carga                  Consumo
                   1.000                1048.184
```

```r
#DEFINICIÓN DE MODELO PCA

pca <- prcomp(features, scale = TRUE)
names(pca)
```

```
[1] "sdev"     "rotation" "center"   "scale"    "x"
```

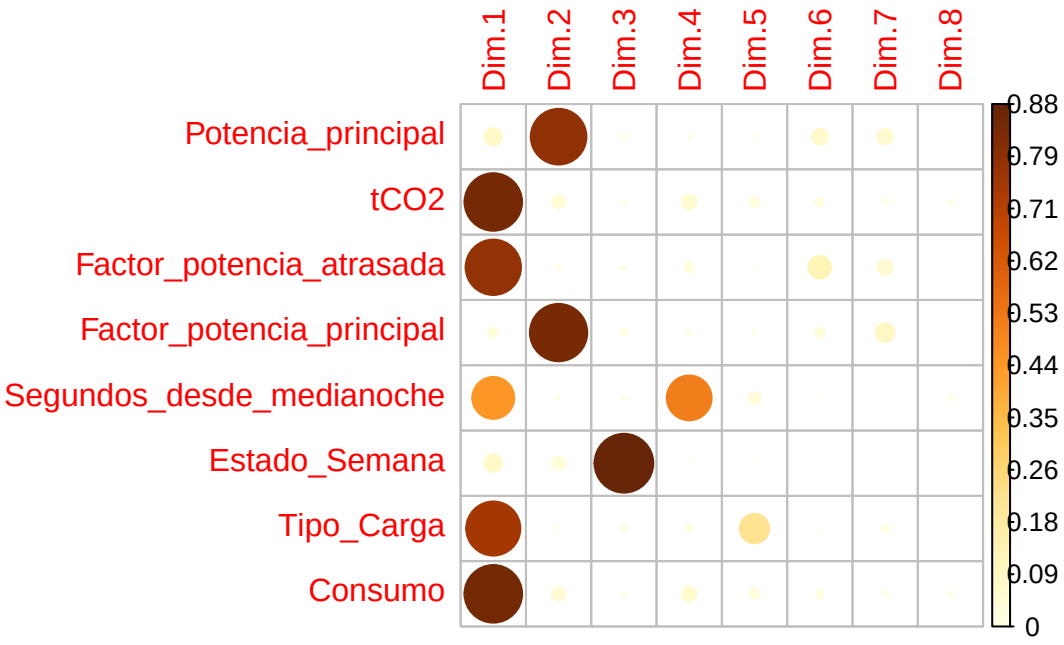
```r
#Varianza explicada por componente

prop_varianza <- pca$sdev^2 / sum(pca$sdev^2)

pca_var<-ggplot(data = data.frame(prop_varianza, pc = 1:length(prop_varianza)), aes(x = pc
  geom_col(width = 0.3) +  scale_y_continuous(limits = c(0,1)) +  theme_bw() +
  labs(x = "Componente principal", y = "Prop. de varianza explicada")
pca_var
```



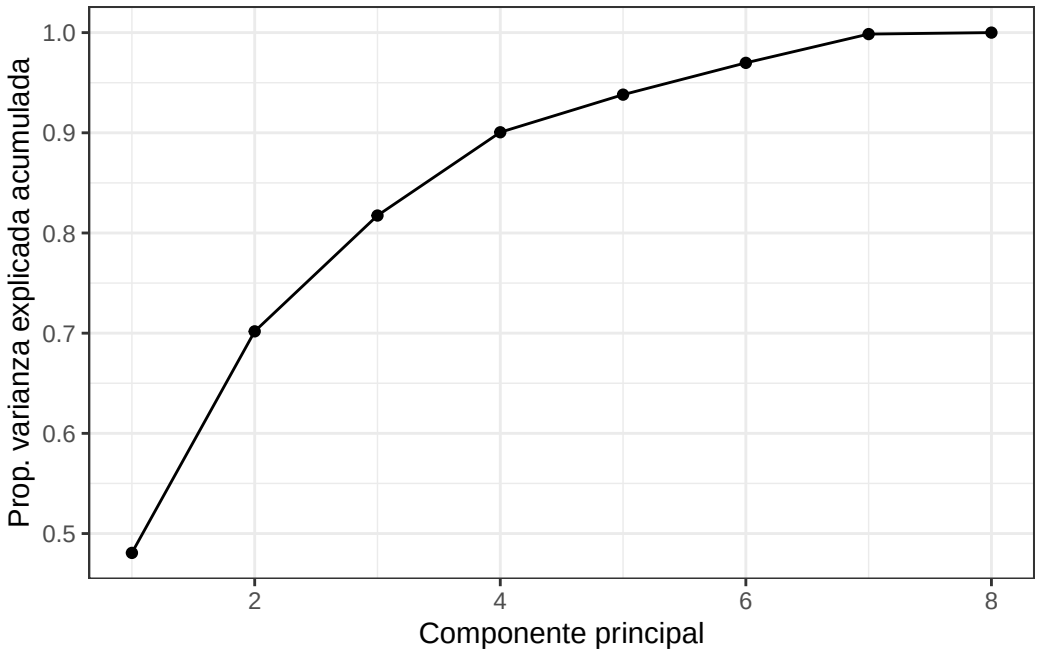

```r
# Importancia de cada variable

library(corrplot)
```

```
corrplot 0.92 loaded
```

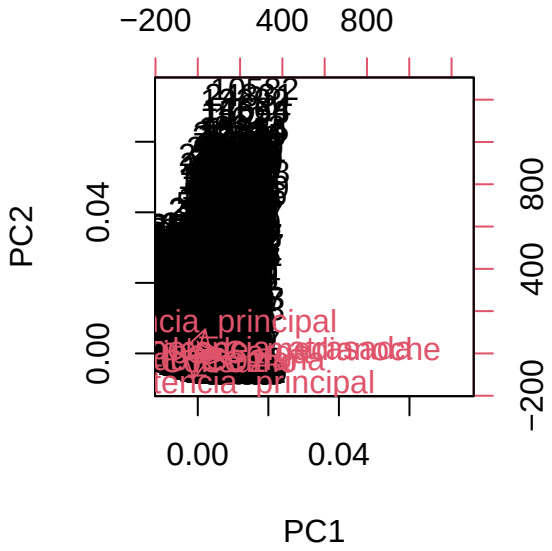
```
var <- get_pca_var(pca)
corrplot(var$cos2, is.corr = FALSE)
```



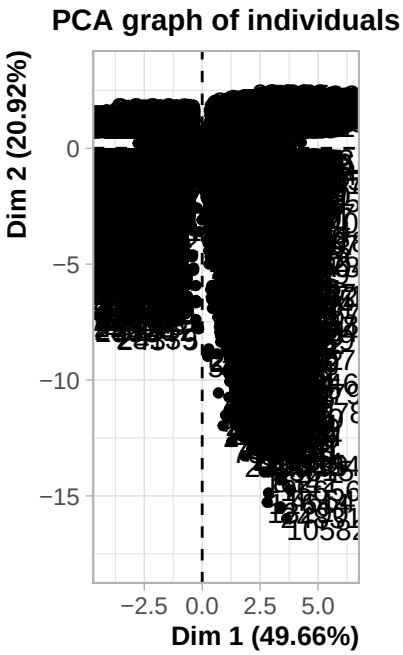

```
# Varianza explicada acumulada

prop_varianza_acum <- cumsum(prop_varianza)

pca_var_acum<-ggplot(data = data.frame(prop_varianza_acum, pc = 1:length(prop_varianza)),
  geom_point() +  geom_line() +  theme_bw() +  labs(x = "Componente principal", y = "Prop.

pca_var_acum
```

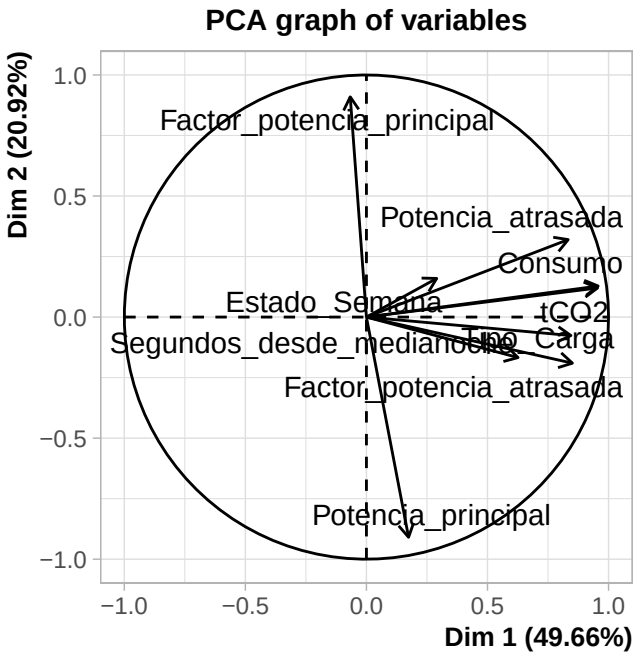

Prop. varianza explicada acumulada
1.0
0.9
0.8
0.7
0.6
0.5
2
4
6
8
Componente principal

```
biplot(pca)
```


−200
400
800
PC2
0.04
0.00
800
400
−200
0.00
0.04
PC1

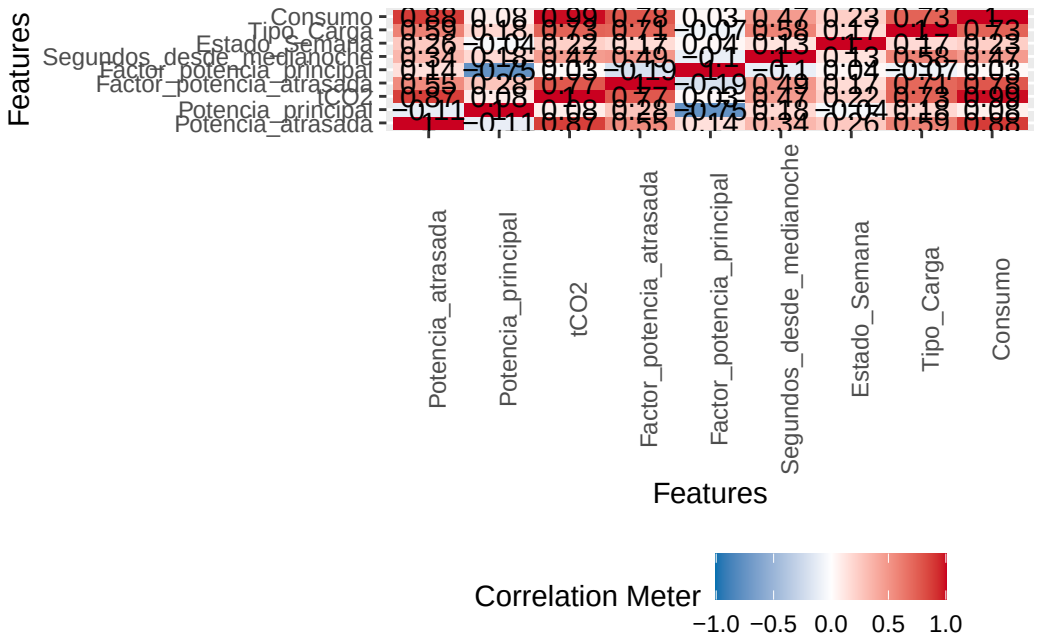
```r
#Gráfico 2d

pca=PCA(data_sc)
```



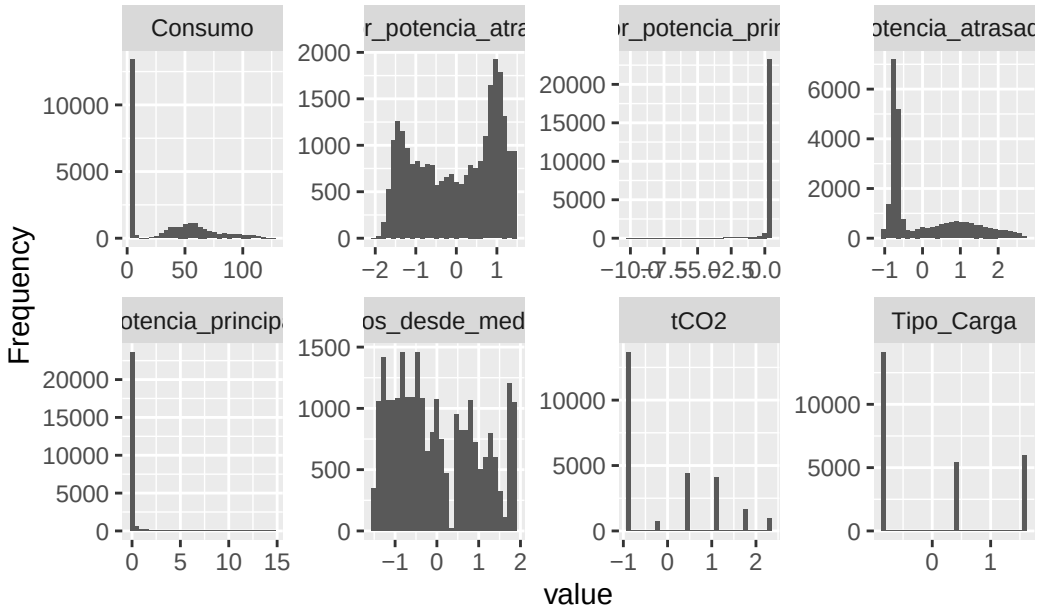


K-means

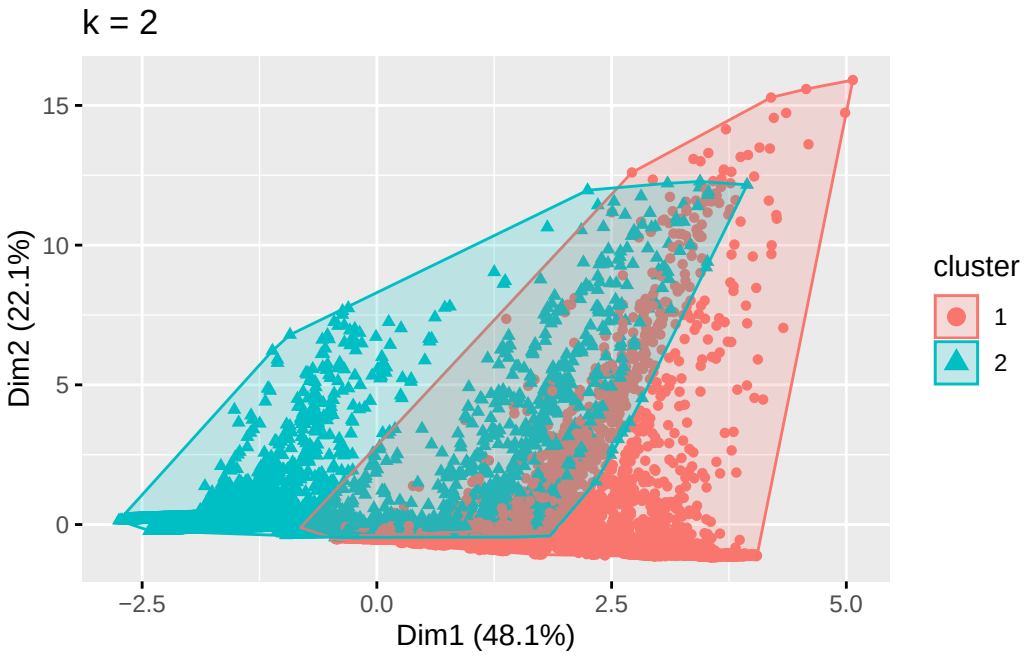
```
# Separación entre datos de train y test

sample <- sample.split(data_sc$Consumo, SplitRatio = 0.75)

train <- subset(data_sc, sample == TRUE)
test <- subset(data_sc, sample == FALSE)

View(train)

plot_correlation(data_sc)
```



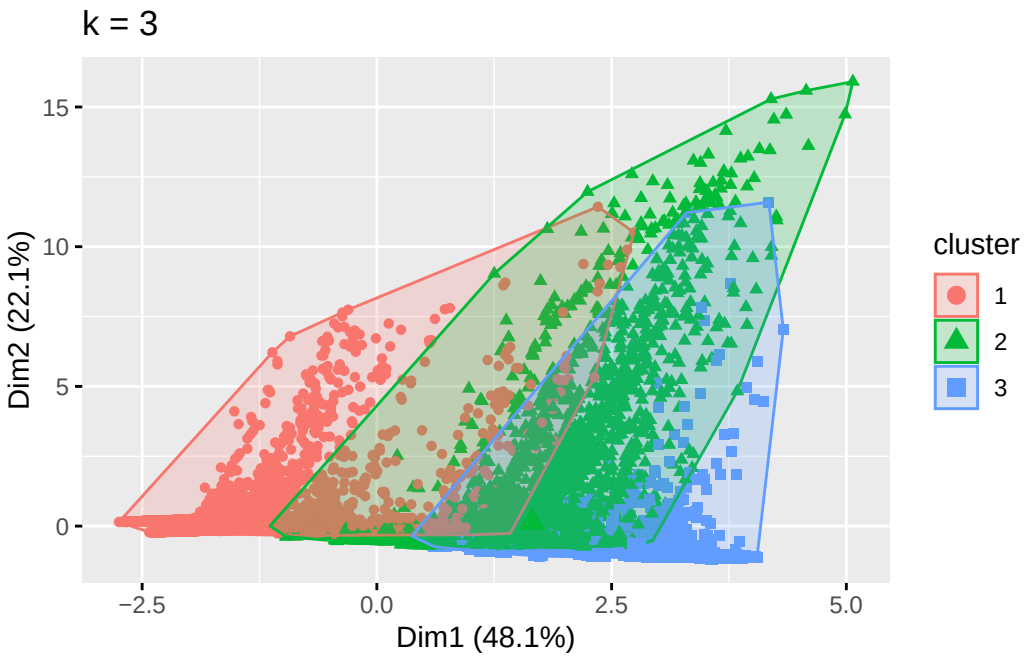

```
plot_histogram(data_sc)
```



```r
#K-MEANS

#Semilla de reproducibilidad

set.seed(411)
#Definición de modelos de K-Means para diferentes números de clústeres
k2 <- kmeans(features, centers = 2, nstart = 25)
k3 <- kmeans(features, centers = 3, nstart = 25)
k4 <- kmeans(features, centers = 4, nstart = 25)
k5 <- kmeans(features, centers = 5, nstart = 25)
#Definición de gráficos de los clústeres
p2 <- fviz_cluster(k2, geom = "point", data = features) + ggtitle("k = 2")
p3 <- fviz_cluster(k3, geom = "point", data = features) + ggtitle("k = 3")

#Muestra los gráficos ya definidos
print(p2)
```

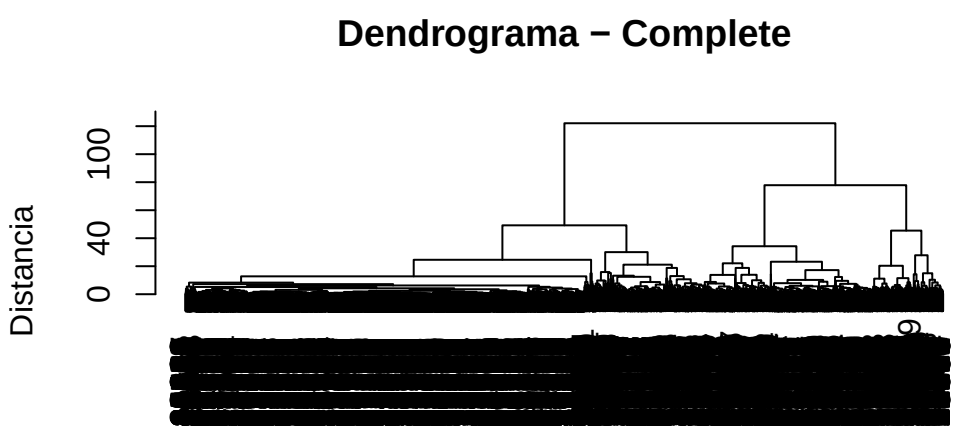



```r
print(p3)
```

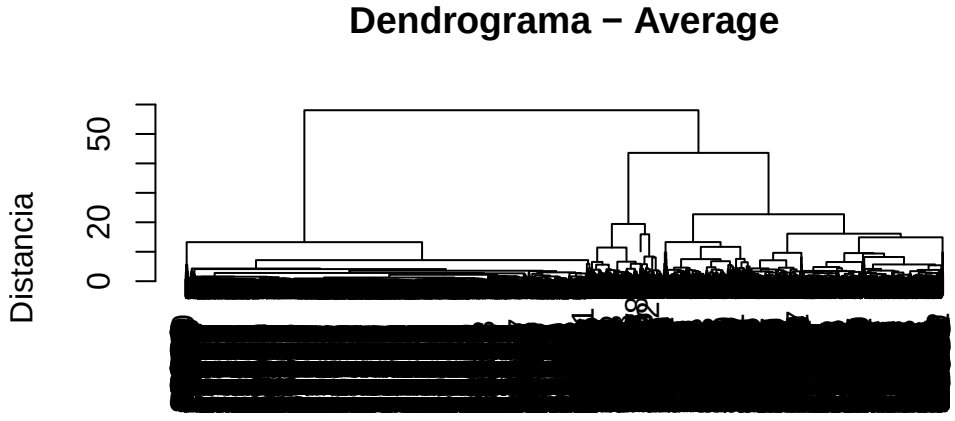


Agrupamiento jeráquico - Aglomerative

```
#Agrupamiento jerárquico
#Aglomerativa

set.seed(587)
dist_matrix <- dist(data_sc)

hc1 <- hclust(dist_matrix, method = "complete")
hc2 <- hclust(dist_matrix, method = "average")

plot(hc1, main = "Dendrograma - Complete", xlab = "Índices", ylab = "Distancia")
```




```
plot(hc2, main = "Dendrograma - Average", xlab = "Índices", ylab = "Distancia")
```



Red neuronal

```r
library(neuralnet)
library(NeuralNetTools)
red_neuronal <- neuralnet(
  formula = Consumo ~ .,
  data = train,
  hidden = c(6,4),
  stepmax = 0.5e+06,
  linear.output = TRUE,
  act.fct = "logistic",
)

# Predicciones

data_pred <- compute(red_neuronal,test)

test_df <- as.data.frame(cbind(round(data_pred,digits = 2),test$Consumo))

colnames(test_df) <- c("Predicción",
                       "Real")

# Dibujo de la red neuronal

plotnet(red_neuronal)
olden(red_neuronal)
```