Laboratorio 1

Ricardo Kaleb Flores Alfonso

2024-10-12

```
library(mnormt)
library(MVN)
library(ggplot2)
library(psych)
```

```
##
## Adjuntando el paquete: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(performance)
library(GPArotation)
```

```
##
## Adjuntando el paquete: 'GPArotation'
```

```
## The following objects are masked from 'package:psych':
##
##     equamax, varimin
```

```
library(mnormt)
library(datasets)
```

Problema 1

Descomposición espectral

```
mat <- matrix(c(4.4,0.8,0.8,5.6),ncol=2)
mat
```

```
##      [,1] [,2]
## [1,]  4.4  0.8
## [2,]  0.8  5.6
```

```
eigenvalues <- eigen(mat)$values
eigenvectors <- eigen(mat)$vectors
print("Valores")
```

```
## [1] "Valores"
```

```
eigenvalues
```

```
## [1] 6 4
```

```r
print("Vectores")
```

```
## [1] "Vectores"
```

```r
eigenvectors
```

```
##           [,1]       [,2]
## [1,] 0.4472136 -0.8944272
## [2,] 0.8944272  0.4472136
```

Reconstruida

```r
# A debería ser igual a Q * Lambda * inv(Q)
A_reconstructed <- eigenvectors %*% diag(eigenvalues) %*% solve(eigenvectors)

cat("\nMatriz reconstruida A (verificación):\n")
```

```
##
## Matriz reconstruida A (verificación):
```

```r
print(A_reconstructed)
```

```
##      [,1] [,2]
## [1,]  4.4  0.8
## [2,]  0.8  5.6
```

Problema 2

Hallar la probabilidad

```r
miu <- c(2.5,4)
cov <- matrix(c(1.2,0,0,2.3),ncol=2)
x_1 <- c(2,3)
prob <- pmnorm(x_1,mean=miu,varcov=cov)
print("La probabilidad es")
```

```
## [1] "La probabilidad es"
```

```r
prob
```

```
## [1] 0.08257333
```

Problema 3

calcular las distancias de Mahalanobis y hallar las proporciones de datos por debajo de los percentiles de Chi-cuadrada corespondientes a 10, 20, 30, 40, 50, 60, 70, 80 y 90.

```r
df <- read.csv("datosX1X2X3.csv")
```

```r
media <- colMeans(df)
cov_x <- cov(df)
dist_mahalanobis <- mahalanobis(df, center = media, cov = cov_x)
head(dist_mahalanobis)
```

```
## [1] 1.060850 4.804832 4.599162 2.823013 1.301073 1.650721
```

```r
# Definir los niveles de confianza (1 - alfa)
niveles_confianza <- c(0.10, 0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.80, 0.90)
```

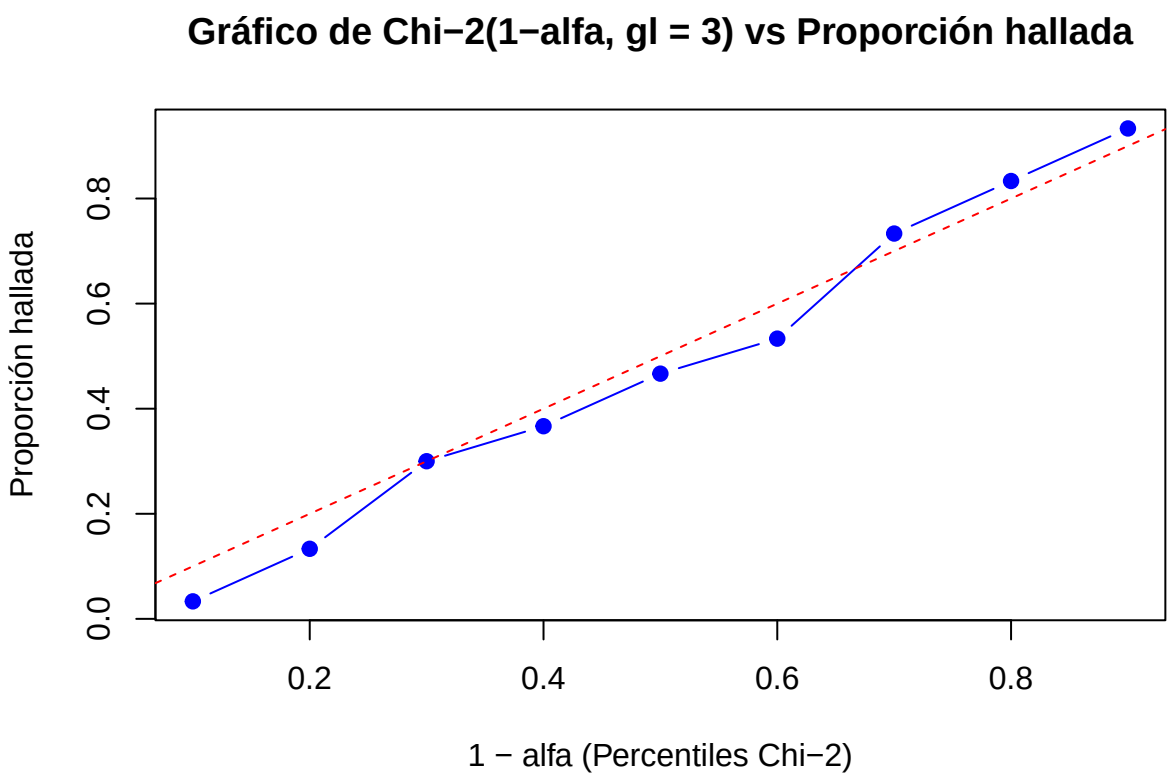
```r
# Calcular los percentiles de la distribución Chi-cuadrada con gl = 3
percentiles_chi2 <- qchisq(niveles_confianza, df = 3)

# Mostrar los percentiles
print(percentiles_chi2)
```

```
## [1] 0.5843744 1.0051740 1.4236522 1.8691684 2.3659739 2.9461661 3.6648708
## [8] 4.6416277 6.2513886
```

```r
# Calcular las proporciones de datos por debajo de cada percentil de Chi-cuadrada
proporciones <- sapply(percentiles_chi2, function(p) mean(dist_mahalanobis <= p))

# Mostrar las proporciones
print(proporciones)
```

```
## [1] 0.03333333 0.13333333 0.30000000 0.36666667 0.46666667 0.53333333 0.73333333
## [8] 0.83333333 0.93333333
```

```r
# Graficar los percentiles de Chi-2 vs la proporción hallada
plot(niveles_confianza, proporciones, type = "b", col = "blue", pch = 19,
     xlab = "1 - alfa (Percentiles Chi-2)", ylab = "Proporción hallada",
     main = "Gráfico de Chi-2(1-alfa, gl = 3) vs Proporción hallada")

# Agregar línea de referencia de identidad
abline(a = 0, b = 1, col = "red", lty = 2)
```



Se observa que las proporciones halladas están cerca de los percentiles teóricos de la distribución Chi-cuadrada, podríamos decir que x sigue una distribución normal multivariada.

Problema 4

A los datos numéricos del problema 3 plantee las hipótesis de la Prueba de normalidad Hipótesis nula H_0: Los datos siguen una distribución normal multivariada.

Hipótesis alternativa H_1: Los datos no siguen una distribución normal multivariada.

```
df <- read.csv("datosX1X2X3.csv")
mvn_mardia <- mvn(df, mvnTest = "mardia")
mvn_hz <- mvn(df, mvnTest = "hz")

p_mardia <- mvn_mardia$multivariateNormality$`p value`
p_hz <- mvn_hz$multivariateNormality$`p value`


print(paste("P-Value Sesgo Mardia:", p_mardia[1]))
```

```
## [1] "P-Value Sesgo Mardia: 0.659972337630838"
```

```
print(paste("P-Value Curtosis Mardia:", p_mardia[2]))
```

```
## [1] "P-Value Curtosis Mardia: 0.240490081413617"
```

```
# ¿Cual es el valor p de la prueba de normalidad multivaridad de Henze-Zirkler's?
print(paste("P-Value HZ:", p_hz))
```

```
## [1] "P-Value HZ: 0.503368731490781"
```

El test de sesgo y curtosis es mayor a 0.05, por lo que ambos pasan el supuesto de normalidad. De igual manera el test de Henze-Zirkler obtuvo un valor de 0.503, por lo que se cumple el supuesto de normalidad

Problema 5

A) Realice la prubea de normalidad de mardia

```
olmos <- read.csv("olmos.csv")
df <- olmos[c("Longitud","Diametro","Altura")]
mardia <- mvn(df, mvnTest = "mardia")
mardia$multivariateNormality
```

```
##              Test         Statistic            p value Result
## 1 Mardia Skewness  17.0312537859682 0.0736752675872285    YES
## 2 Mardia Kurtosis 0.848747898502705  0.396021587220212    YES
## 3             MVN              <NA>               <NA>    YES
```

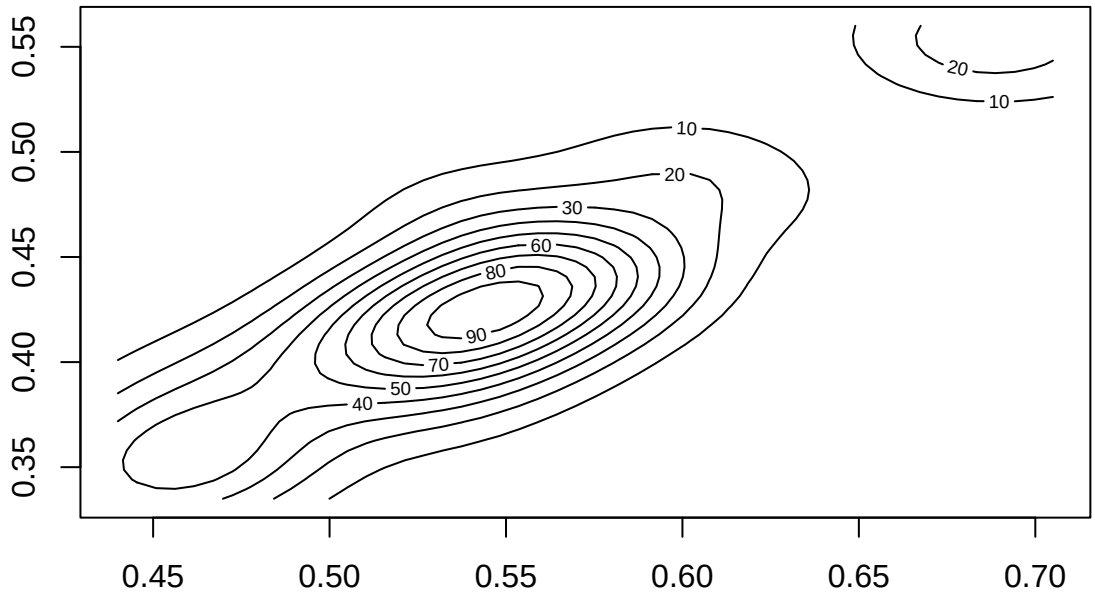
```
mardia$univariateNormality
```

```
##               Test  Variable Statistic   p value Normality
## 1 Anderson-Darling Longitud     0.6693    0.0725    YES
## 2 Anderson-Darling Diametro     0.6223    0.0955    YES
## 3 Anderson-Darling  Altura      0.2248    0.8041    YES
```

B) Elabore la gráfica de contorno

```
# Generar la matriz de covarianza y el vector de medias
media <- colMeans(df)
cov_matrix <- cov(df)
```

```r
# Graficar el contorno de la normal multivariada
library(MASS)
contour_plot <- kde2d(df$Longitud, df$Diametro, n = 50)
contour(contour_plot)
```



C) Con el vector de medias y la matriz de covarianza de la normal multivariada en en el inciso A, calcule la probabilidad de que P(X <= (0.25, 0.25, 0.25))

```r
library(MVN)

prob <- pmnorm( x = c(0.25, 0.25, 0.25), mean = media, varcov = cov_matrix)
prob
```

```
## [1] 6.623513e-06
```

D) Calcula la distancia de Mahalanobis de cada observación al centroide

```r
media <- colMeans(olmos)
cov_matrix <- cov(olmos)
# Calcular la distancia de Mahalanobis para cada observación
dist_mahalanobis <- mahalanobis(olmos, center = media, cov = cov_matrix)

# Identificar las observaciones más cercanas y más alejadas al centroide
max_dist <- which.max(dist_mahalanobis)
min_dist <- which.min(dist_mahalanobis)
```

```r
cat("Observación más alejada:", max_dist, "\n")
```

```
## Observación más alejada: 14
```

```r
cat("Observación más cercana:", min_dist, "\n")
```

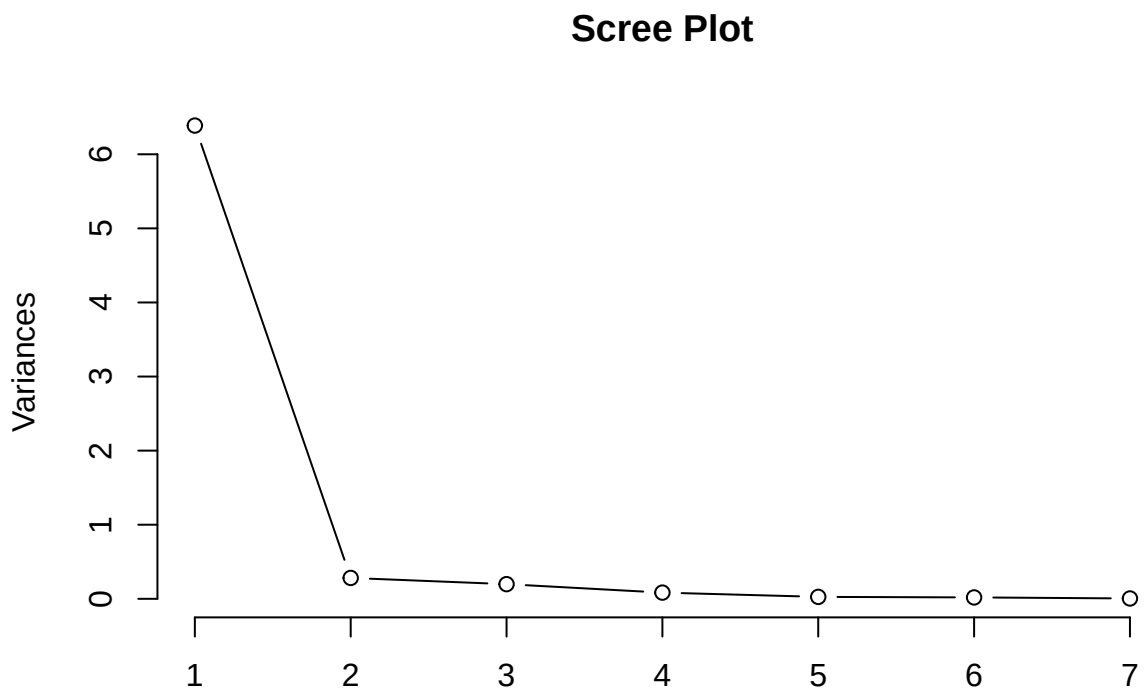
```
## Observación más cercana: 6
```

E) Aplica un análisis de componentes principales a los datos y con base en al menos tres criterios

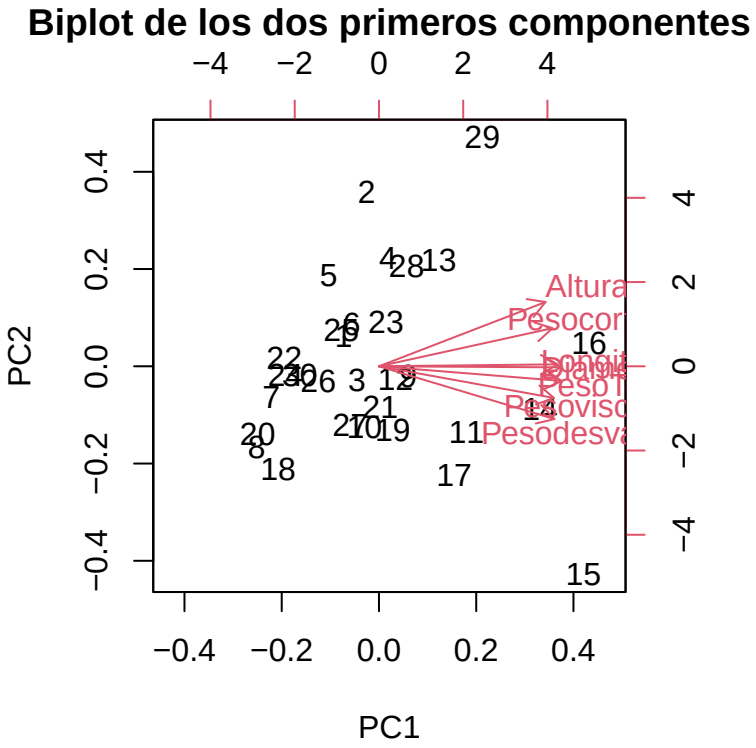
```r
# Aplicar PCA
pca_result <- prcomp(olmos, scale = TRUE)

# Mostrar la varianza acumulada
summary(pca_result)
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.5274 0.53058 0.44469 0.28935 0.16163 0.13661 0.06806
## Proportion of Variance 0.9125 0.04022 0.02825 0.01196 0.00373 0.00267 0.00066
## Cumulative Proportion  0.9125 0.95273 0.98098 0.99294 0.99667 0.99934 1.00000
```

```r
# Gráfico de Scree
plot(pca_result, type = "l", main = "Scree Plot")
```



```r
# Graficar las puntuaciones de los dos primeros componentes
biplot(pca_result, main = "Biplot de los dos primeros componentes")
```




```r
# Obtener las combinaciones lineales de los componentes
pca_result$rotation
```

```
##                       PC1         PC2         PC3         PC4         PC5
## Longitud        0.3829206  0.01984410 -0.11178144 -0.82798216 -0.2777313
## Diametro        0.3895735 -0.01072829 -0.23266157 -0.19008711  0.6094553
## Altura          0.3571618  0.65634059  0.55239439  0.08997344  0.2658877
## PesoTotal       0.3913471 -0.14929602 -0.20861352  0.19596466 -0.0575013
## Pesodesvainado  0.3762967 -0.53901550  0.01823927  0.25053286  0.3175768
## Pesovisceras    0.3752590 -0.32334612  0.55274938  0.10746252 -0.4720819
## Pesocorteza     0.3721137  0.38900954 -0.52806230  0.39686290 -0.3920595
##                        PC6         PC7
## Longitud        -0.27298918 -0.05719002
## Diametro         0.61556971 -0.08655058
## Altura          -0.23308811  0.05823544
## PesoTotal       -0.09146657  0.85496423
## Pesodesvainado  -0.52753646 -0.35442067
## Pesovisceras     0.45211736 -0.10137310
## Pesocorteza      0.03390893 -0.34495301
```

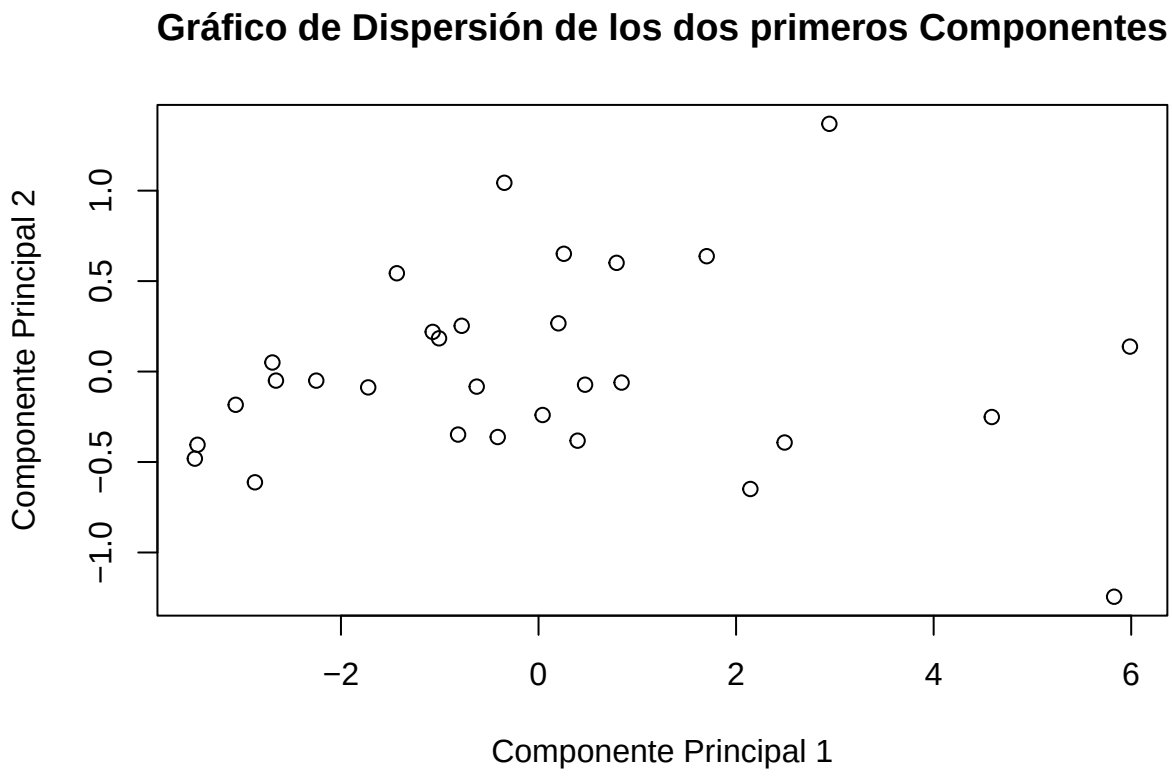
F) Escribir las combinaciones lineales

```
pca_result$rotation[,c(1,2)]
```

```
##                     PC1         PC2
## Longitud       0.3829206  0.01984410
## Diametro       0.3895735 -0.01072829
## Altura         0.3571618  0.65634059
## PesoTotal      0.3913471 -0.14929602
## Pesodesvainado 0.3762967 -0.53901550
## Pesovisceras   0.3752590 -0.32334612
## Pesocorteza    0.3721137  0.38900954
```

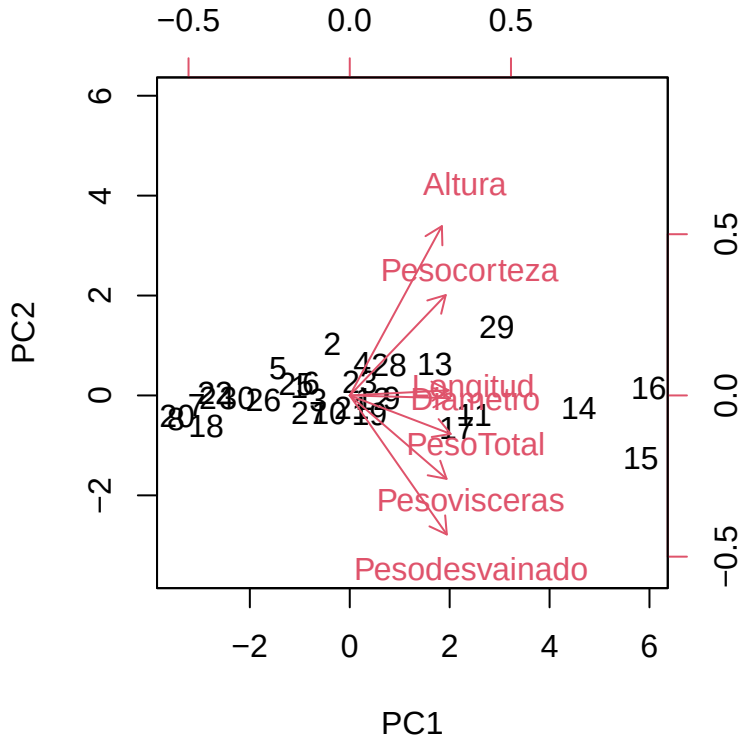
Utilizando los dos primeros componentes hacer una gráfica de dispersión de las puntuaciones. Comentar el gráfico en función de la variabilidad.

```
# Obtener puntuaciones de los dos primeros componentes
scores <- pca_result$x[, 1:2]

# Gráfico de dispersión
plot(scores, xlab = "Componente Principal 1", ylab = "Componente Principal 2", main = "Gráfico de Dispe
```



H) Hacer un gráfico vectorial de las variables e interpretar sus relaciones.

```
# Gráfico biplot
biplot(pca_result, scale = 0)
```



Problema 6 ## A) Justifique por qué es adecuado el uso del Análisis factorial (hacer la prueba de esfericidad de Bartlett y KMO).

```
# Matriz de correlación
corr.test(olmos)
```

```
## Call:corr.test(x = olmos)
## Correlation matrix
##                Longitud Diametro Altura PesoTotal Pesodesvainado Pesovisceras
## Longitud           1.00     0.96   0.86      0.95           0.90         0.90
## Diametro           0.96     1.00   0.86      0.98           0.93         0.91
## Altura             0.86     0.86   1.00      0.84           0.77         0.85
## PesoTotal          0.95     0.98   0.84      1.00           0.97         0.93
## Pesodesvainado     0.90     0.93   0.77      0.97           1.00         0.95
## Pesovisceras       0.90     0.91   0.85      0.93           0.95         1.00
## Pesocorteza        0.90     0.94   0.86      0.94           0.84         0.81
##                Pesocorteza
## Longitud              0.90
## Diametro              0.94
## Altura                0.86
## PesoTotal             0.94
## Pesodesvainado        0.84
## Pesovisceras          0.81
## Pesocorteza           1.00
## Sample Size
## [1] 30
## Probability values (Entries above the diagonal are adjusted for multiple tests.)
##                Longitud Diametro Altura PesoTotal Pesodesvainado Pesovisceras
```

```
## Longitud              0        0      0         0              0            0
## Diametro              0        0      0         0              0            0
## Altura                0        0      0         0              0            0
## PesoTotal             0        0      0         0              0            0
## Pesodesvainado        0        0      0         0              0            0
## Pesovisceras          0        0      0         0              0            0
## Pesocorteza           0        0      0         0              0            0
##                Pesocorteza
## Longitud                 0
## Diametro                 0
## Altura                   0
## PesoTotal                0
## Pesodesvainado           0
## Pesovisceras             0
## Pesocorteza              0
##
##  To see confidence intervals of the correlations, print with the short=FALSE option
```

```
# Prueba de esfericidad de Bartlett
check_sphericity_bartlett(olmos)
```

```
## # Test of Sphericity
##
## Bartlett's test of sphericity suggests that there is sufficient significant correlation in the data f
```

```
# Medida de adecuación muestral de Kaiser-Meyer-Olkin (KMO)
KMO(cor(olmos))
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = cor(olmos))
## Overall MSA =  0.84
## MSA for each item =
##       Longitud       Diametro         Altura      PesoTotal Pesodesvainado
##           0.92           0.93           0.82           0.78           0.83
##   Pesovisceras    Pesocorteza
##           0.83           0.76
```

B) Justifique el número de factores principales que se utilizarán en el modelo

```
pca_result = prcomp(olmos)
# Eigenvalores
pca_result$sdev
```

```
## [1] 0.427139183 0.053734410 0.027774291 0.020152956 0.014667729 0.009313184
## [7] 0.007361492
```

```
# Aportacion acumulada
cumsum(pca_result$sdev) / sum(pca_result$sdev)
```

```
## [1] 0.7625535 0.8584832 0.9080675 0.9440457 0.9702314 0.9868578 1.0000000
```

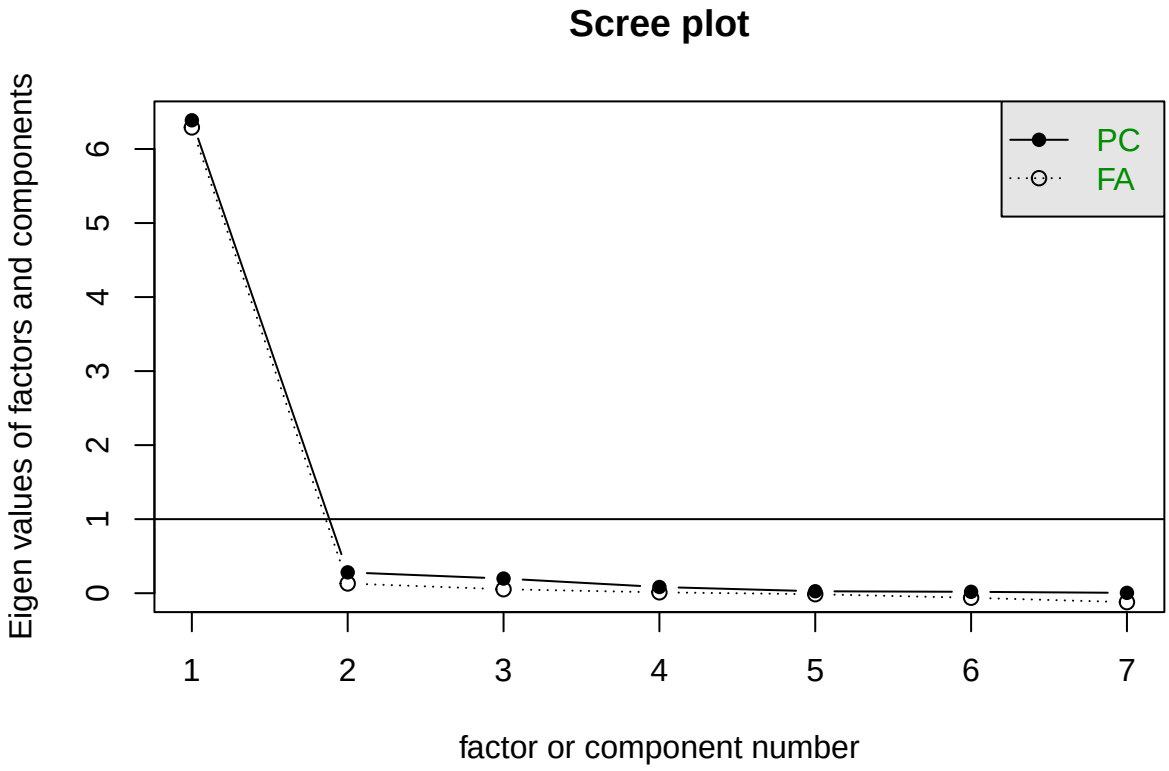
```
# Combinaciones Lineales
pca_result$rotation
```

```
##                      PC1         PC2          PC3         PC4         PC5
## Longitud       0.14890586  0.07635415  0.468234796 -0.75503238 -0.18936524
## Diametro       0.13388178  0.07957222  0.137250190 -0.28098170 -0.18290039
```

```
## Altura          0.05072929  0.08235490  0.357662557  0.18274943 -0.19340250
## PesoTotal       0.86583167  0.13198558 -0.144383206 -0.01669354  0.44900012
## Pesodesvainado 0.34778797 -0.64252207 -0.319294494 -0.01535904 -0.59799027
## Pesovisceras    0.18441539 -0.35875918  0.714928360  0.45095235  0.07241479
## Pesocorteza     0.22956321  0.64968687 -0.008832536  0.33720238 -0.57345127
##                         PC6         PC7
## Longitud         0.382533847  0.02209150
## Diametro        -0.862871702 -0.31615736
## Altura          -0.258193127  0.85161971
## PesoTotal       -0.003157374  0.10089134
## Pesodesvainado   0.053929652  0.05935724
## Pesovisceras     0.070030767 -0.33563973
## Pesocorteza      0.186101870 -0.21896115
```

```r
cor_olmos = cor(olmos)
scree(cor_olmos)
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```



C) Identifique las comunalidades de los factores del modelo propuesto, y los errores: interprete si se necesita un nuevo factor.

```r
# Realizar análisis factorial con el número de factores seleccionado
fa_result <- fa(cor_olmos, nfactors = 2, rotate = "none")
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
fa_varimax = fa(cor_olmos, nfactors = 2, rotate = "varimax")
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
fa_oblimin = fa(cor_olmos, nfactors = 2, rotate = "oblimin")
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
data.frame(Normal = fa_result$communalities, VARIMAX = fa_varimax$communalities, OBLIMIN = fa_oblimin$c
```

```
##                   Normal   VARIMAX   OBLIMIN
## Longitud       0.9268256 0.9268256 0.9268256
## Diametro       0.9739740 0.9739740 0.9739740
## Altura         0.7955799 0.7955799 0.7955799
## PesoTotal      0.9847066 0.9847066 0.9847066
## Pesodesvainado 0.9950000 0.9950000 0.9950000
## Pesovisceras   0.9106034 0.9106034 0.9106034
## Pesocorteza    0.9350870 0.9350870 0.9350870
```

```
cbind(fa_result$residual, fa_varimax$residual, fa_oblimin$residual)
```

```
##                    Longitud     Diametro       Altura    PesoTotal
## Longitud        0.073112915  0.013436555  0.006250406 -0.004305453
## Diametro        0.013436555  0.025976323 -0.011865472  0.002282749
## Altura          0.006250406 -0.011865472  0.204282020 -0.022090788
## PesoTotal      -0.004305453  0.002282749 -0.022090788  0.015204450
## Pesodesvainado -0.004886504  0.005052948 -0.026187436  0.009167933
## Pesovisceras    0.001597368 -0.012805684  0.052925590 -0.007121473
## Pesocorteza    -0.012008070  0.002662614  0.003202091  0.020638272
##                Pesodesvainado Pesovisceras  Pesocorteza     Longitud
## Longitud         -0.004886504  0.001597368 -0.012008070  0.073112915
## Diametro          0.005052948 -0.012805684  0.002662614  0.013436555
## Altura           -0.026187436  0.052925590  0.003202091  0.006250406
## PesoTotal         0.009167933 -0.007121473  0.020638272 -0.004305453
## Pesodesvainado    0.002672349  0.001084834  0.015843843 -0.004886504
## Pesovisceras      0.001084834  0.089346080 -0.031329496  0.001597368
## Pesocorteza       0.015843843 -0.031329496  0.064856327 -0.012008070
##                    Diametro       Altura    PesoTotal Pesodesvainado
## Longitud        0.013436555  0.006250406 -0.004305453   -0.004886504
## Diametro        0.025976323 -0.011865472  0.002282749    0.005052948
## Altura         -0.011865472  0.204282020 -0.022090788   -0.026187436
## PesoTotal       0.002282749 -0.022090788  0.015204450    0.009167933
## Pesodesvainado  0.005052948 -0.026187436  0.009167933    0.002672349
## Pesovisceras   -0.012805684  0.052925590 -0.007121473    0.001084834
## Pesocorteza     0.002662614  0.003202091  0.020638272    0.015843843
##                Pesovisceras  Pesocorteza     Longitud     Diametro       Altura
## Longitud        0.001597368 -0.012008070  0.073112915  0.013436555  0.006250406
## Diametro       -0.012805684  0.002662614  0.013436555  0.025976323 -0.011865472
## Altura          0.052925590  0.003202091  0.006250406 -0.011865472  0.204282020
```

```
## PesoTotal      -0.007121473  0.020638272 -0.004305453  0.002282749 -0.022090788
## Pesodesvainado  0.001084834  0.015843843 -0.004886504  0.005052948 -0.026187436
## Pesovisceras    0.089346080 -0.031329496  0.001597368 -0.012805684  0.052925590
## Pesocorteza    -0.031329496  0.064856327 -0.012008070  0.002662614  0.003202091
##                   PesoTotal Pesodesvainado Pesovisceras  Pesocorteza
## Longitud       -0.004305453   -0.004886504  0.001597368 -0.012008070
## Diametro        0.002282749    0.005052948 -0.012805684  0.002662614
## Altura         -0.022090788   -0.026187436  0.052925590  0.003202091
## PesoTotal       0.015204450    0.009167933 -0.007121473  0.020638272
## Pesodesvainado  0.009167933    0.002672349  0.001084834  0.015843843
## Pesovisceras   -0.007121473    0.001084834  0.089346080 -0.031329496
## Pesocorteza     0.020638272    0.015843843 -0.031329496  0.064856327
```

```
mr1 = data.frame(MR1_NORMAL = fa_result$loadings[,1], MR1_VARIMAX = fa_varimax$loadings[,1], MR1_OBLIMI

mr2 = data.frame(MR2_NORMAL = fa_result$loadings[,2], MR2_VARIMAX = fa_varimax$loadings[,2], MR1_OBLIMI

mr1
```

```
##                MR1_NORMAL MR1_VARIMAX MR1_OBLIMIN
## Longitud        0.9615699   0.7311096   0.9647391
## Diametro        0.9856530   0.7502973   0.9889956
## Altura          0.8780582   0.7458081   0.8893439
## PesoTotal       0.9919153   0.6997754   0.9893457
## Pesodesvainado  0.9555358   0.4944097   0.9337753
## Pesovisceras    0.9399186   0.5692442   0.9274121
## Pesocorteza     0.9355710   0.8476442   0.9532822
```

```
mr2
```

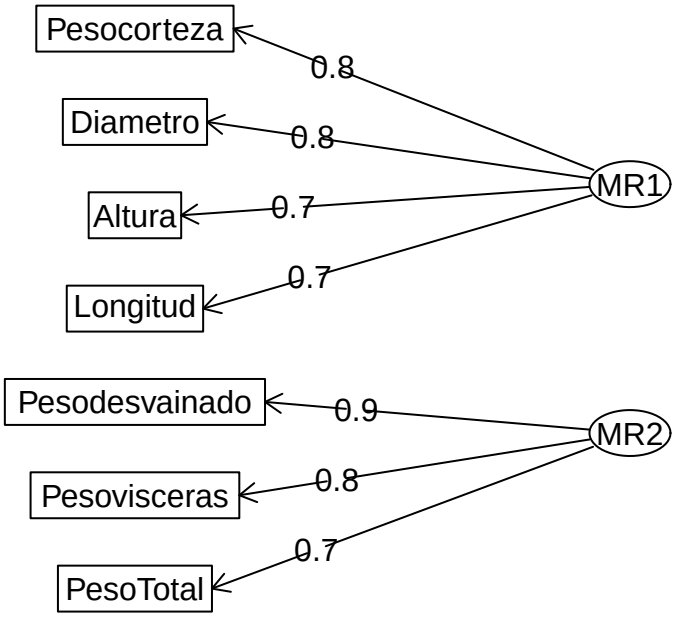
```
##                 MR2_NORMAL MR2_VARIMAX MR1_OBLIMIN
## Longitud        0.04764969   0.6263912 -0.04289352
## Diametro        0.05011844   0.6411533 -0.04524613
## Altura          0.15726341   0.4893754 -0.15318715
## PesoTotal      -0.02999342   0.7036405  0.03508535
## Pesodesvainado -0.29030847   0.8676904  0.29582688
## Pesovisceras   -0.16494507   0.7659079  0.17009034
## Pesocorteza     0.24464386   0.4654492 -0.24048141
```

D) Encuentre con ayuda de un gráfico de variables qué conviene más sin rotación o con rotación varimax.

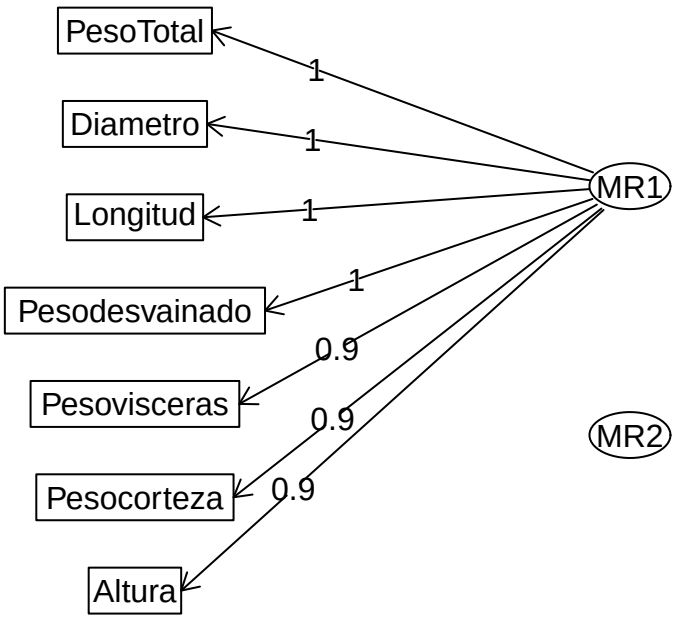
```
# Realizar análisis factorial con rotación varimax
fa_varimax <- fa(olmos, nfactors = 2, rotate = "varimax")
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
# Gráfico de diagrama de factores
fa.diagram(fa_varimax)
```




```
fa.diagram(fa_result)
```



¿Tienen interpretación en el contexto del problema los factores encontrados?

Problema 7

```
library(dplyr)
```

```
##
## Adjuntando el paquete: 'dplyr'
## The following object is masked from 'package:MASS':
##
##     select
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

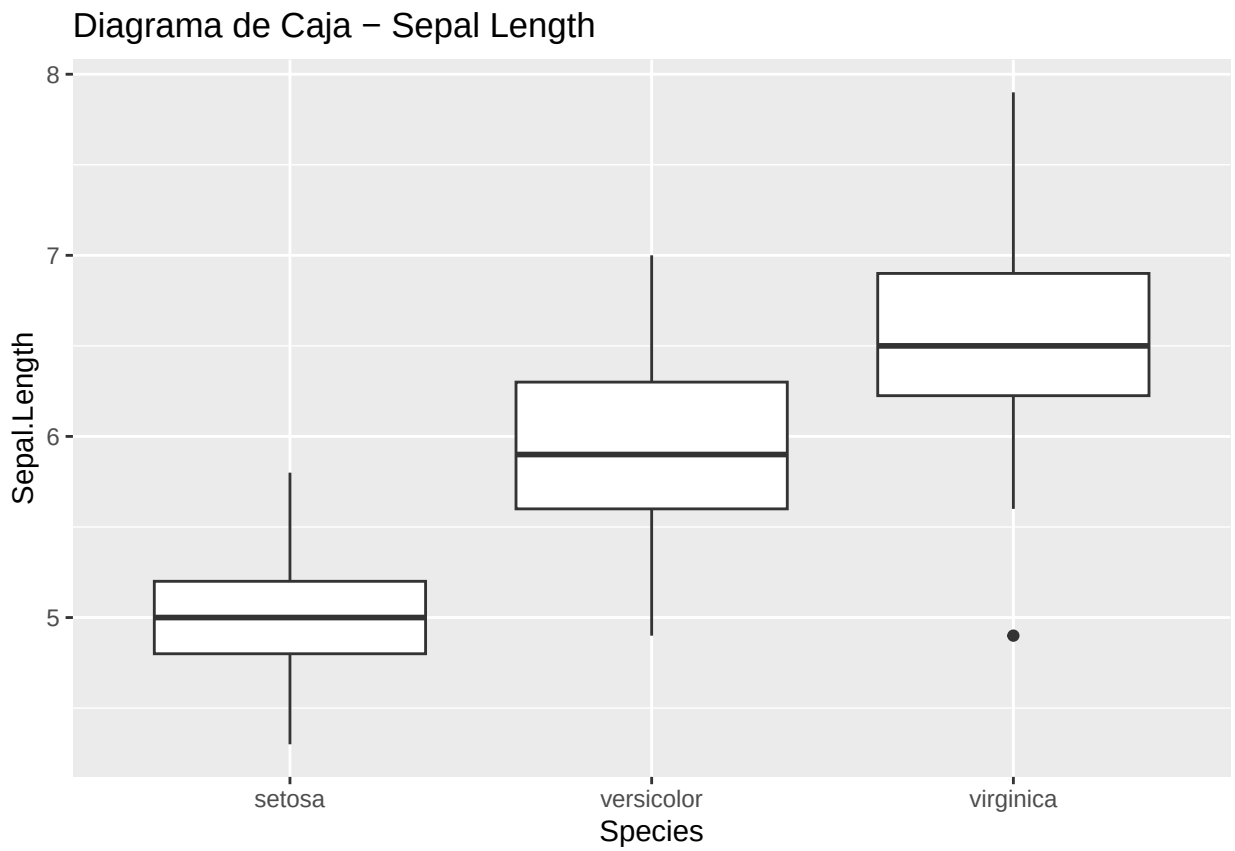
```
library(ggplot2)
```

A) Haz un leve análisis descriptivo para cada variable por especie: media, desviación estándar, diagramas de caja y bigote

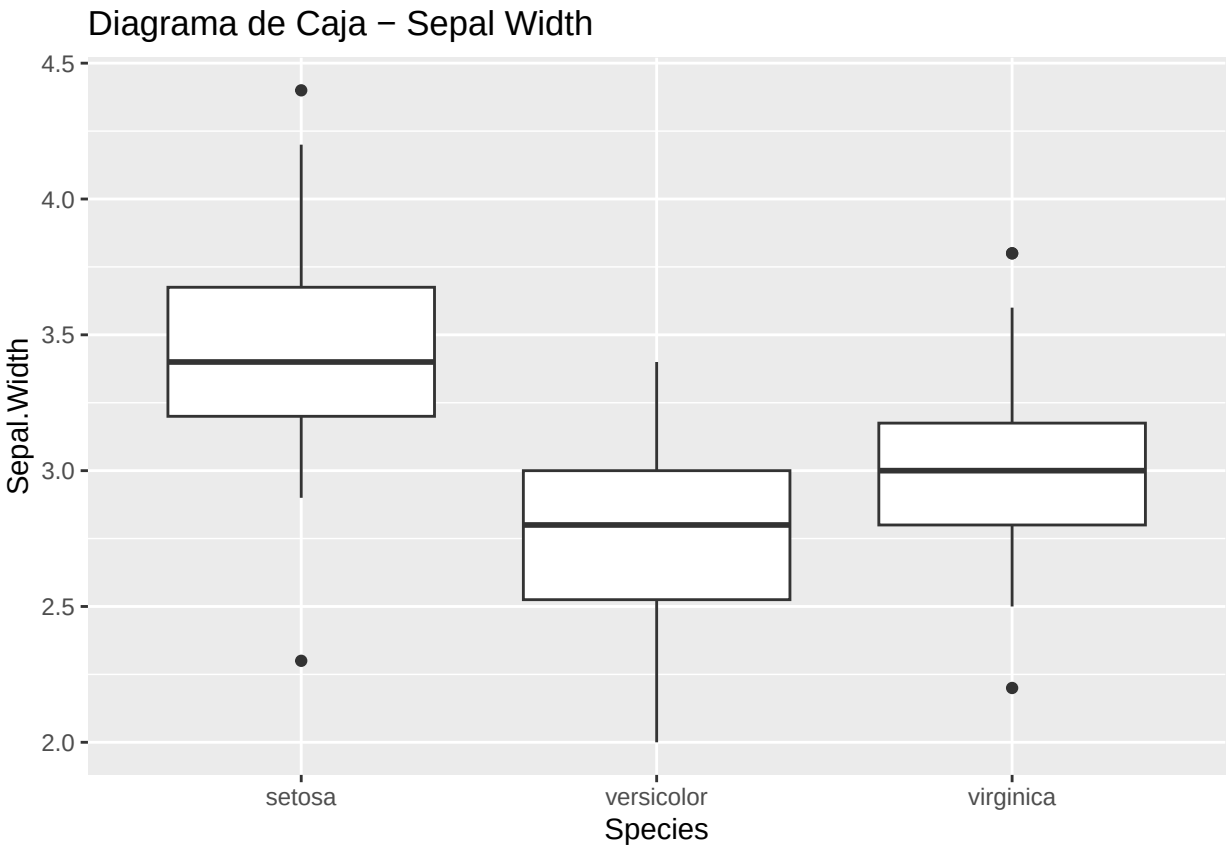
```
descriptive_stats <- iris %>%
  group_by(Species) %>%
  summarise(
    Sepal.Length_promedio = mean(Sepal.Length),
    Sepal.Length_desviacion = sd(Sepal.Length),
    Sepal.Width_promedio = mean(Sepal.Width),
    Sepal.Width_desviacion = sd(Sepal.Width),
    Petal.Length_promedio = mean(Petal.Length),
    Petal.Length_desviacion = sd(Petal.Length),
    Petal.Width_promedio = mean(Petal.Width),
    Petal.Width_desviacion = sd(Petal.Width)
  )
descriptive_stats
```

```
## # A tibble: 3 x 9
##   Species    Sepal.Length_promedio Sepal.Length_desviacion Sepal.Width_promedio
##   <fct>                      <dbl>                   <dbl>                <dbl>
## 1 setosa                      5.01                   0.352                 3.43
## 2 versicolor                  5.94                   0.516                 2.77
## 3 virginica                   6.59                   0.636                 2.97
## # i 5 more variables: Sepal.Width_desviacion <dbl>,
## #   Petal.Length_promedio <dbl>, Petal.Length_desviacion <dbl>,
## #   Petal.Width_promedio <dbl>, Petal.Width_desviacion <dbl>
```

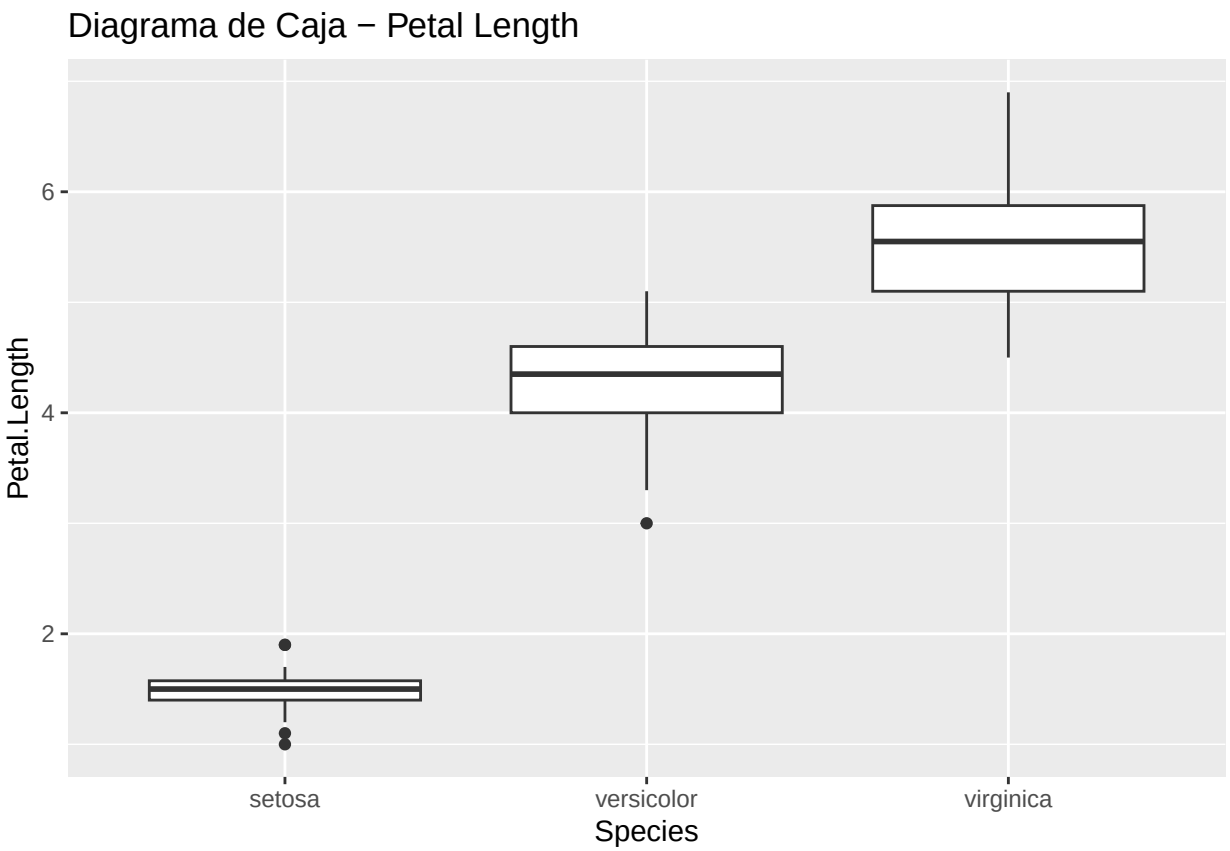
```
ggplot(iris, aes(x = Species, y = Sepal.Length)) + geom_boxplot() + ggtitle("Diagrama de Caja - Sepal L
```



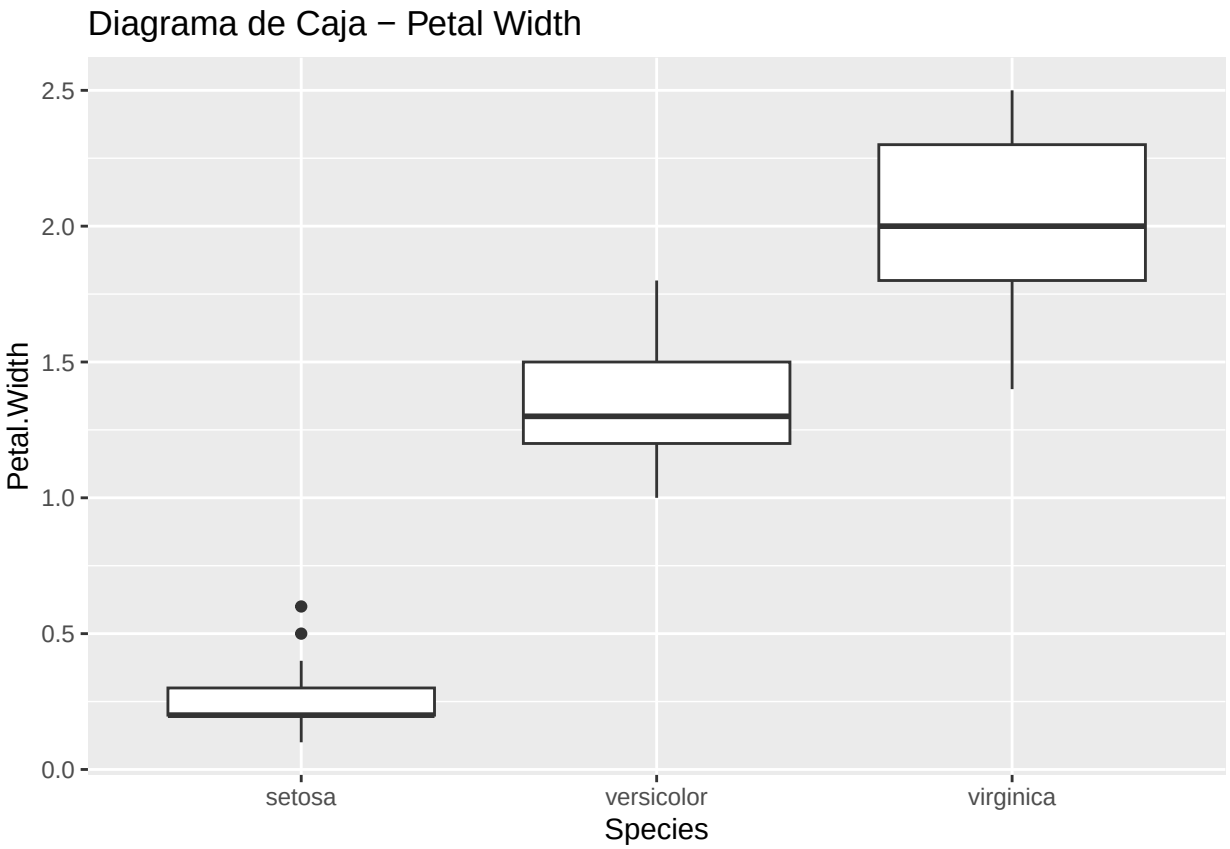

```
ggplot(iris, aes(x = Species, y = Sepal.Width)) + geom_boxplot() + ggtitle("Diagrama de Caja - Sepal Wi
```




```
ggplot(iris, aes(x = Species, y = Petal.Length)) + geom_boxplot() + ggtitle("Diagrama de Caja - Petal L
```



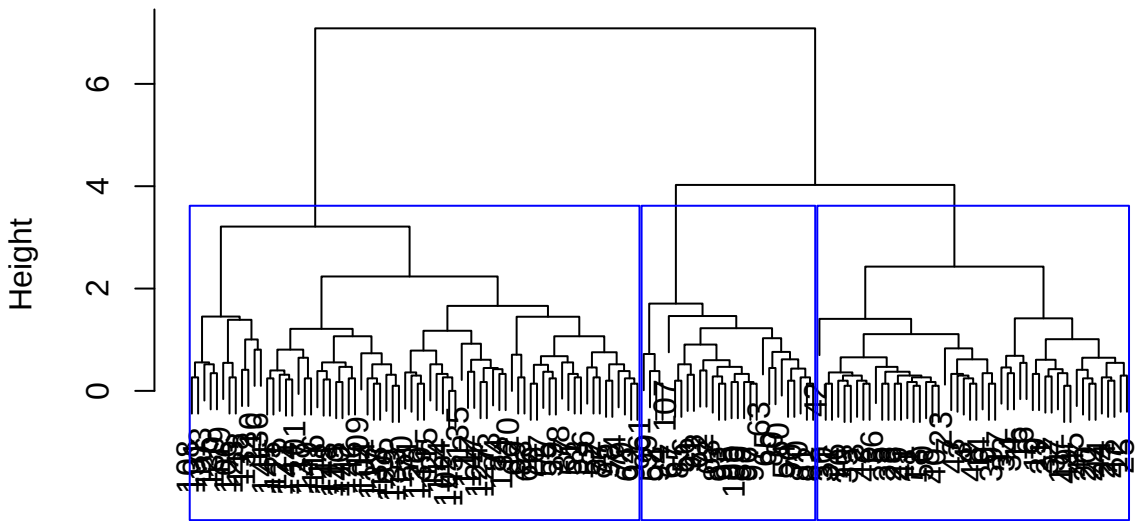

```
ggplot(iris, aes(x = Species, y = Petal.Width)) + geom_boxplot() + ggtitle("Diagrama de Caja - Petal Wi
```



B) Realiza dos análisis clouster jerárquicos usando dos distintas distancias y métodos de aglomeración. Sigue los siguientes puntos para cada uno de ellos:

Distance euclidiana

```r
x <- iris[1:4]
dist_iris <- dist(x,method="euclidean")

# Clúster jerárquico usando el método "complete"
jerarquico1 <- hclust(dist_iris, method = "complete")

# Dendograma
plot(jerarquico1, main = "Dendograma (Distancia Euclidiana, Complete)")

# Seleccionar número óptimo de grupos y colorear el dendograma
rect.hclust(jerarquico1, k = 3, border = "blue")
```




```r
# Asignación de grupos a cada observación
grupo1 <- cutree(jerarquico1, k = 3)

# Resultado: combinación de iris con el grupo asignado
resultado1 <- cbind(iris, Grupo = grupo1)

# Contar observaciones mal clasificadas
table(resultado1$Species, resultado1$Grupo)
```

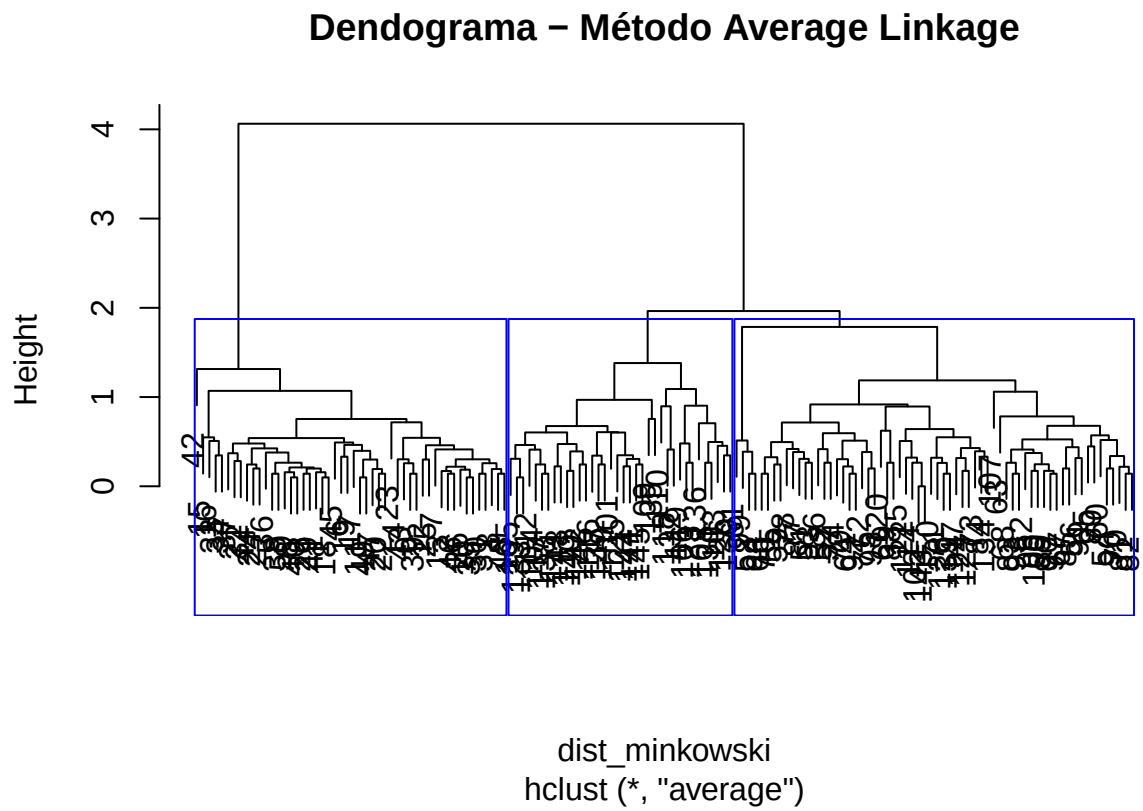
```
##
##                1  2  3
##   setosa      50  0  0
##   versicolor   0 23 27
##   virginica    0 49  1
```

```r
# Calcular medias por grupo de clasificación
aggregate(. ~ Grupo, data = resultado1[, -5], FUN = mean)
```

```
##   Grupo Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     1     5.006000    3.428000     1.462000    0.246000
## 2     2     6.545833    2.963889     5.273611    1.850000
## 3     3     5.532143    2.635714     3.960714    1.228571
```

```r
# Clustering jerárquico usando la distancia minkowski y el método de enlace promedio (average linkage)
dist_minkowski <- dist(x, method = "minkowski")
jerarquico2 <- hclust(dist_minkowski, method = "average")

# Dendograma y selección del número óptimo de grupos
```

```r
plot(jerarquico2, main = "Dendograma - Método Average Linkage")
rect.hclust(jerarquico2, k = 3, border = "blue")
```




```r
# Asignar las observaciones a los grupos
grupos2 <- cutree(jerarquico2, k = 3)
resultado2 <- cbind(iris, Grupo = grupos2)

# Contar las observaciones mal clasificadas
table(iris$Species, grupos2)
```

```
##             grupos2
##               1  2  3
##   setosa     50  0  0
##   versicolor  0 50  0
##   virginica   0 14 36
```

```r
# Calcula la media para cada grupo de clasificación por el método
aggregate(. ~ Grupo, data = resultado2[, -5], FUN = mean)
```

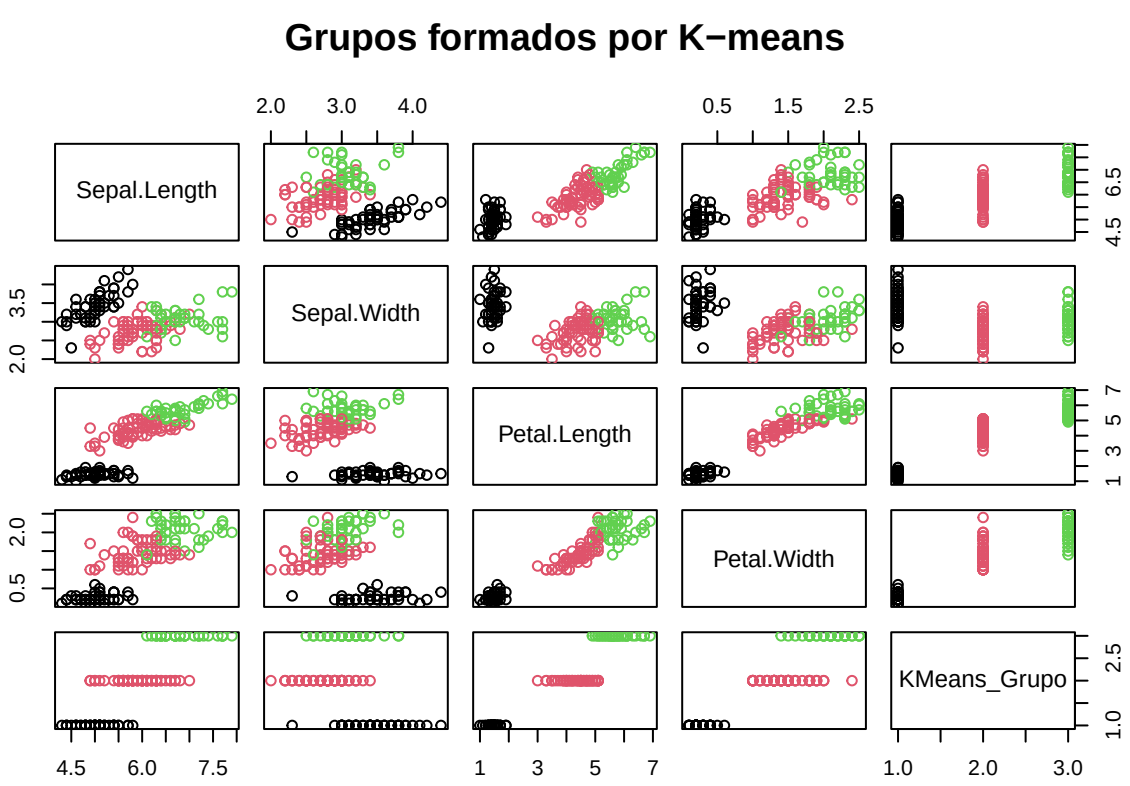
```
##   Grupo Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     1     5.006000    3.428000     1.462000    0.246000
## 2     2     5.929688    2.757812     4.410938    1.439062
## 3     3     6.852778    3.075000     5.786111    2.097222
```

Interpreta el dendograma obtenido y concluye para los dos métodos. Indica cuál es mejor y por qué.

Ambos dendogramas funcionan de una buena manera, sin embargo el obtenido a traves de la distancia de minkowski logró clasificar de mejor manera las clases de plantas

C) Hacer el gráfico de agromeración no-jerárquica con el método de k-medias para las especies de Iris.

```
set.seed(123)
kmeans_result <- kmeans(x, centers = 3)

# Asignar las observaciones a los grupos formados por K-means
iris$KMeans_Grupo <- kmeans_result$cluster

# Gráfico de los grupos formados por K-means
pairs(iris[, -5], col = kmeans_result$cluster, main = "Grupos formados por K-means")
```




```
# Contar las observaciones mal clasificadas
table(iris$Species, iris$KMeans_Grupo)
```

```
##
##                1  2  3
##   setosa      50  0  0
##   versicolor   0 48  2
##   virginica    0 14 36
```

D) ¿Cuál de los dos métodos resultó mejor par la clasificación de acuerdo a la clasificacion de cada observación en las especies y en los grupos.

El mejor metodo fue el jerarquico con distancia de minkowski, este tuvo menos errores al clasificar los datos obtenidos.