Actividad Integradora

Luis Maximiliano López Ramírez

2024-08-20

```
# Especificar el nuevo directorio
nuevo_directorio <- "C:/Users/luism/Escritorio/Documetos_2/Actividades
Concentración"

# Cambiar al nuevo directorio
setwd(nuevo_directorio)
```

Variable seleccionada: **5. Agua**

```
# Carga los datos desde el archivo datosRes.csv
datos <- read.csv("food_data_g.csv")
X <- datos$Water
```

 # Punto 1. Análisis descriptivo de la variable

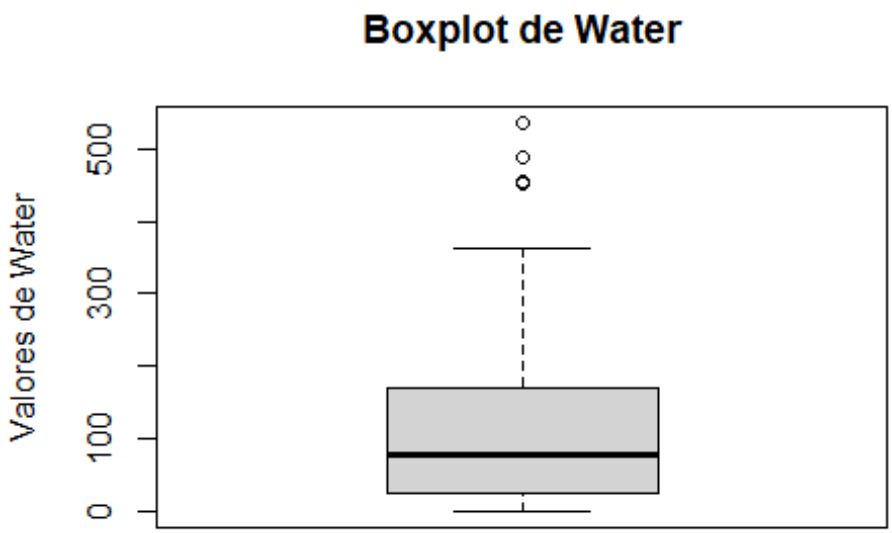
Analiza una de las siguientes variables en cuanto a sus datos atípicos y normalidad. La variable que te corresponde analizar te será asignada por tu profesora al inicio de la actividad:

1. Calorías
2. Grasas saturadas
3. Grasas monosaturadas
4. Sodio **5. Agua**
5. Sodio
6. Densidad Nutricional

1. Para analizar datos atípicos se te sugiere:

A. Graficar el diagrama de caja y bigote

```
boxplot(X, main = "Boxplot de Water", ylab = "Valores de Water")
```



B. Calcula las principales medidas que te ayuden a identificar datos atípicos (utilizar summary te puede abreviar el cálculo): Cuartil 1, Cuartil 3, Media, Cuartil 3, Rango intercuartílico y Desviación estándar

```
summary(X)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0    25.9    76.7   101.7   169.1   535.8
```

```
Q1 <- quantile(X, 0.25)  # Primer cuartil (Q1)
Q3 <- quantile(X, 0.75)  # Tercer cuartil (Q3)
media <- mean(X)         # Media
rango_intercuartilico <- IQR(X)  # Rango intercuartílico (Q3 - Q1)
desviacion_estandar <- sd(X)  # Desviación estándar

cat("Primer Cuartil (Q1):", Q1, "\n")
```

```
## Primer Cuartil (Q1): 25.9
```

```
cat("Tercer Cuartil (Q3):", Q3, "\n")
```

```
## Tercer Cuartil (Q3): 169.05
```

```
cat("Media:", media, "\n")
```

```
## Media: 101.6587
```

```
cat("Rango Intercuartílico:", rango_intercuartilico, "\n")
```

```
## Rango Intercuartílico: 143.15
```

```
cat("Desviación Estándar:", desviacion_estandar, "\n")
```

```
## Desviación Estándar: 88.50171
```

C. Identifica la cota de 1.5 rangos intercuartílicos para datos atípicos, ¿hay datos atípicos de acuerdo con este criterio? ¿cuántos son?

```
Q1 <- quantile(X, 0.25)  # Primer cuartil (Q1)
Q3 <- quantile(X, 0.75)  # Tercer cuartil (Q3)
IQR_value <- IQR(X)      # Rango intercuartílico (IQR)

# Calcular las cotas inferior y superior
lower_bound <- Q1 - 1.5 * IQR_value
upper_bound <- Q3 + 1.5 * IQR_value

# Identificar los datos atípicos
outliers <- X[X < lower_bound | X > upper_bound]

num_outliers <- length(outliers)

cat("Cota inferior:", lower_bound, "\n")
```

```
## Cota inferior: -188.825
```

```
cat("Cota superior:", upper_bound, "\n")
```

```
## Cota superior: 383.775
```

```
cat("Número de datos atípicos:", num_outliers, "\n")
```

```
## Número de datos atípicos: 4
```

```
cat("Datos atípicos:", outliers, "\n")
```

```
## Datos atípicos: 451.7 489.3 453.8 535.8
```

D. Identifica la cota de 3 desviaciones estándar alrededor de la media, ¿hay datos atípicos de acuerdo con este criterio? ¿cuántos son?

```
media <- mean(X)
desviacion_estandar <- sd(X)

# Calcular las cotas de 3 desviaciones estándar
lower_bound <- media - 3 * desviacion_estandar
upper_bound <- media + 3 * desviacion_estandar

# Identificar los datos atípicos
outliers <- X[X < lower_bound | X > upper_bound]

num_outliers <- length(outliers)
```

```r
cat("Cota inferior:", lower_bound, "\n")
```

```
## Cota inferior: -163.8464
```

```r
cat("Cota superior:", upper_bound, "\n")
```

```
## Cota superior: 367.1638
```

```r
cat("Número de datos atípicos:", num_outliers, "\n")
```

```
## Número de datos atípicos: 4
```

```r
cat("Datos atípicos:", outliers, "\n")
```

```
## Datos atípicos: 451.7 489.3 453.8 535.8
```

E. Identifica la cota de 3 rangos intercuartílicos para datos extremos, ¿hay datos extremos de acuerdo con este criterio? ¿cuántos son?

```r
Q1 <- quantile(X, 0.25)
Q3 <- quantile(X, 0.75)
IQR_value <- IQR(X)

# Calcular las cotas de 3 veces el rango intercuartílico
lower_bound <- Q1 - 3 * IQR_value
upper_bound <- Q3 + 3 * IQR_value

# Identificar los datos extremos
extremes <- X[X < lower_bound | X > upper_bound]

num_extremes <- length(extremes)

cat("Cota inferior para datos extremos:", lower_bound, "\n")
```

```
## Cota inferior para datos extremos: -403.55
```

```r
cat("Cota superior para datos extremos:", upper_bound, "\n")
```

```
## Cota superior para datos extremos: 598.5
```

```r
cat("Número de datos extremos:", num_extremes, "\n")
```

```
## Número de datos extremos: 0
```

```r
cat("Datos extremos:", extremes, "\n")
```

```
## Datos extremos:
```

F. Interpreta los resultados obtenidos y argumenta sobre el comportamiento de los datos atípicos y extremos en la variable seleccionada

Los resultados indican que, según los criterios de 1.5 veces el rango intercuartílico (IQR) y 3 desviaciones estándar, hay 4 datos que se consideran atípicos en la variable seleccionada. Sin embargo, no se identificaron datos extremos utilizando la cota más estricta de 3 veces el IQR. Esto sugiere que, aunque hay algunos valores que se desvían significativamente del resto de los datos, estos no son lo suficientemente extremos como para ser considerados fuera de lo esperado en un rango mucho más amplio, lo que indica una distribución con algunas desviaciones moderadas pero sin outliers extremos.

2. Para analizar normalidad se te sugiere:

1. Realiza pruebas de normalidad univariada para la variable (utiliza las pruebas de Anderson-Darling y de Jarque Bera). No olvides incluir H0 y H1 para la prueba de normalidad.

H0 (Hipótesis nula): Los datos siguen una distribución normal. H1 (Hipótesis alternativa): Los datos no siguen una distribución normal.

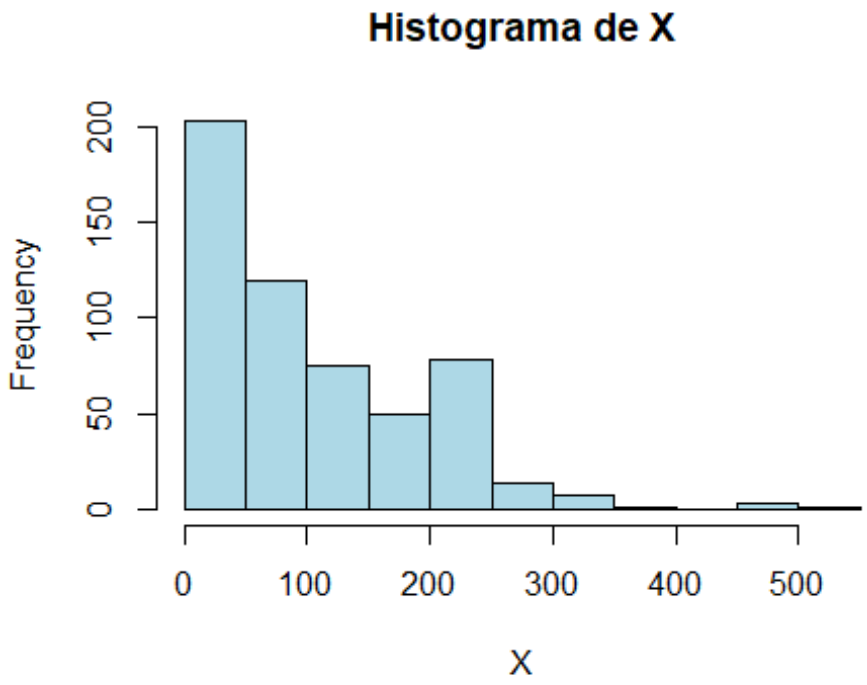
```
library(nortest)
library(tseries)

## Warning: package 'tseries' was built under R version 4.3.3

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

# Prueba de Anderson-Darling
ad_test <- ad.test(X)

# Prueba de Jarque-Bera
jb_test <- jarque.bera.test(X)

cat("Resultado de la prueba de Anderson-Darling:\n")

## Resultado de la prueba de Anderson-Darling:

print(ad_test)

##
##  Anderson-Darling normality test
##
## data:  X
## A = 15.968, p-value < 2.2e-16

cat("\nResultado de la prueba de Jarque-Bera:\n")

##
## Resultado de la prueba de Jarque-Bera:

print(jb_test)
```

```
##
##  Jarque Bera Test
##
## data:  X
## X-squared = 153.58, df = 2, p-value < 2.2e-16
```

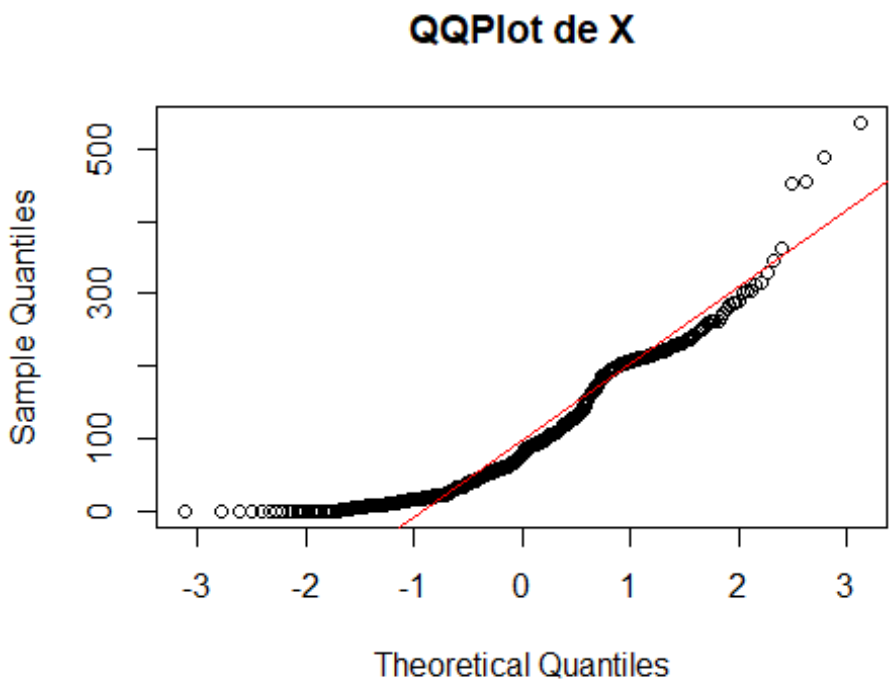
Si el p-valor es menor que un nivel de significancia (por ejemplo, 0.05), se rechaza la hipótesis nula H0, indicando que los datos no siguen una distribución normal. Dado que se obtuvieron valores p de 2.2e-16 entonces se rechaza H0 y los datos no siguen una distribución normal.

2. Grafica los datos y su respectivo QQPlot: qqnorm(datos) y qqline(datos)

```
# Graficar los datos
hist(X, main = "Histograma de X", xlab = "X", col = "lightblue", border =
"black")
```




```
# Crear un QQPlot
qqnorm(X, main = "QQPlot de X")
qqline(X, col = "red")
```



3. Calcula el coeficiente de sesgo y el coeficiente de curtosis

```
library(e1071)

# Calcular el coeficiente de sesgo (skewness)
sesgo <- skewness(X)

# Calcular el coeficiente de curtosis (kurtosis)
curtosis <- kurtosis(X)

cat("Coeficiente de sesgo:", sesgo, "\n")

## Coeficiente de sesgo: 1.080845

cat("Coeficiente de curtosis:", curtosis, "\n")

## Coeficiente de curtosis: 1.395062
```

4. Compara las medidas de media, mediana y rango medio de cada variable

```
media <- mean(X)
mediana <- median(X)

minimo <- min(X)
maximo <- max(X)
rango_medio <- (minimo + maximo) / 2

cat("Media:", media, "\n")
```

```
## Media: 101.6587
```

```
cat("Mediana:", mediana, "\n")
```

```
## Mediana: 76.7
```

```
cat("Rango Medio:", rango_medio, "\n")
```

```
## Rango Medio: 267.9
```

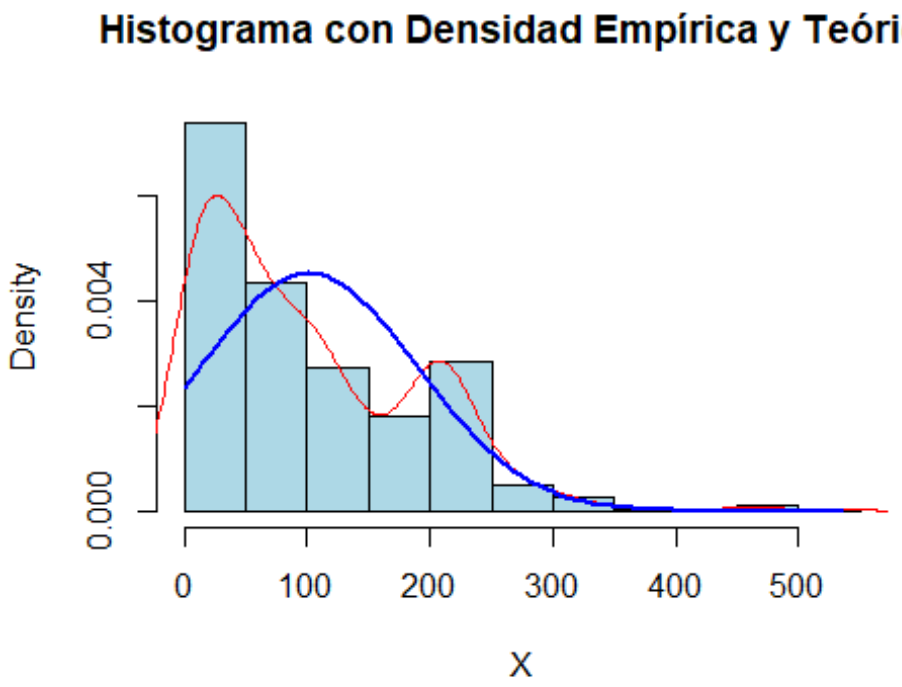
5. Realiza el gráfico de densidad empírica y teórica suponiendo normalidad en la variable. Adapta el código:

hist(datos,freq=FALSE)

lines(density(datos),col="red")

curve(dnorm(x,mean=mean(datos,sd=sd(datos)), from=-6, to=6, add=TRUE, col="blue",lwd=2)

```
hist(X, freq = FALSE, main = "Histograma con Densidad Empírica y
Teórica", xlab = "X", col = "lightblue", border = "black")

lines(density(X),col="red")

curve(dnorm(x, mean = mean(X), sd = sd(X)), from = min(X), to = max(X),
add = TRUE, col = "blue", lwd = 2)
```



6. Interpreta los gráficos y los resultados obtenidos en cada punto con vías a indicar si hay normalidad de los datos. Comenta las características encontradas:

Los datos de X no siguen una distribución normal, como lo sugieren varios indicadores. La media (101.6587) es considerablemente mayor que la mediana (76.7), y el sesgo positivo (1.083794) junto con una alta curtosis (4.411058) indican una distribución sesgada a la derecha con colas más pesadas. Las pruebas de normalidad, Anderson-Darling y Jarque-Bera, tienen p-valores extremadamente bajos (menores que 0.05), lo que lleva a rechazar la hipótesis de normalidad. Estos resultados, junto con las posibles discrepancias en los gráficos comparativos, confirman que los datos no se ajustan a una distribución normal.

A. Considera alejamientos de normalidad por simetría, curtosis

Los datos de X muestran alejamientos significativos de la normalidad en términos de simetría y curtosis. La media (101.6587) es notablemente mayor que la mediana (76.7), lo que indica una asimetría positiva. El coeficiente de sesgo (1.083794) y la curtosis elevada (4.411058) refuerzan esta asimetría y sugieren colas más pesadas en comparación con una distribución normal. Las pruebas estadísticas de normalidad, Anderson-Darling y Jarque-Bera, proporcionan p-valores extremadamente bajos, confirmando que los datos no siguen una distribución normal.

B. Comenta si hay aparente influencia de los datos atípicos en la normalidad de los datos

La presencia de datos atípicos puede estar influyendo en estos resultados. Los datos atípicos, que se identificaron usando criterios de 1.5 IQR y 3 desviaciones estándar, pueden estar contribuyendo a la asimetría y a la alta curtosis observadas, afectando así la normalidad de la distribución.

C. Emite una conclusión sobre la normalidad de los datos. Se debe argumentar en términos de los 3 puntos analizados: las pruebas de normalidad, los gráficos y las medidas.

En conclusión, los datos de X no presentan una distribución normal. Las pruebas de normalidad, los gráficos y las medidas de tendencia central y dispersión indican claramente que los datos están sesgados y tienen colas más pesadas, con una posible influencia de datos atípicos en estos resultados.

 # Punto 2. Transformación a normalidad

1. Encuentra la mejor transformación de los datos para lograr normalidad. Puedes hacer uso de la transformación Box-Cox o de Yeo Johnson o el comando powerTransform para encontrar la mejor lambda para la transformación. Utiliza el modelo exacto y el aproximado de acuerdo con las sugerencias de Box y Cox para la transformación.

```
# Verificar si hay valores iguales a 0 en X
hay_cero <- any(X == 0)

# Imprimir el resultado
if (hay_cero) {
```

```r
  cat("Hay valores iguales a 0 en X.\n")
} else {
  cat("No hay valores iguales a 0 en X.\n")
}
```

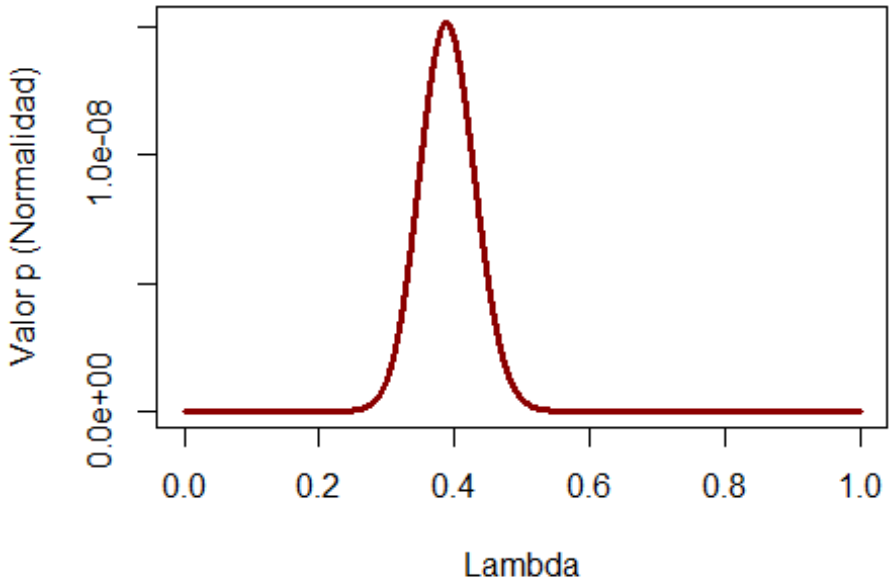
```
## Hay valores iguales a 0 en X.
```

```r
library(VGAM)
```

```
## Loading required package: stats4
```

```
## Loading required package: splines
```

```r
library(nortest)

lp <- seq(0, 1, 0.001)  # Valores de lambda propuestos
nlp <- length(lp)
n <- length(X)
D <- matrix(as.numeric(NA), ncol=2, nrow=nlp)
d <- NA

for (i in 1:nlp) {
  d <- yeo.johnson(X, lambda = lp[i])
  p <- ad.test(d)
  D[i, ] <- c(lp[i], p$p.value)
}

# Convertir la matriz en un data frame y asignar nombres a las columnas
N <- as.data.frame(D)
colnames(N) <- c("Lambda", "P_value")

# Graficar
plot(N$Lambda, N$P_value, type="l",
     col="darkred", lwd=3,
     xlab="Lambda",
     ylab="Valor p (Normalidad)")
```




```r
# Encontrar el índice del máximo valor p
max_index <- which.max(N$P_value)

# Obtener el valor de lambda correspondiente
best_lambda <- N$Lambda[max_index]
max_p_value <- N$P_value[max_index]

# Imprimir los resultados
cat("El valor de lambda que maximiza el valor p es:", best_lambda, "\n")

## El valor de lambda que maximiza el valor p es: 0.388

cat("El valor p máximo es:", max_p_value, "\n")

## El valor p máximo es: 1.517219e-08
```

2. Escribe las ecuaciones de los modelos de transformación encontrados.

Yeo Johnson $X_2 = \frac{(x+1)^{0.388}-1}{0.388}$

Modelo aproximado $X_1 = \sqrt{X+1}$

3. Analiza la normalidad de las transformaciones obtenidas con los datos originales. Utiliza como argumento de normalidad:

3.1 Compara las medidas: Mínimo, máximo, media, mediana, cuartil 1 y cuartil 3, sesgo y curtosis.

```r
X_aprox=sqrt(X+1)
X_yeo=yeo.johnson(X, best_lambda)

# Función para calcular las medidas
calculate_stats <- function(x) {
  c(
    Min = min(x, na.rm = TRUE),
    Max = max(x, na.rm = TRUE),
    Mean = mean(x, na.rm = TRUE),
    Median = median(x, na.rm = TRUE),
    Q1 = quantile(x, 0.25, na.rm = TRUE),
    Q3 = quantile(x, 0.75, na.rm = TRUE),
    Skewness = skewness(x, na.rm = TRUE),
    Kurtosis = kurtosis(x, na.rm = TRUE)
  )
}

# Aplicar la función a cada variable
stats_X <- calculate_stats(X)
stats_X1 <- calculate_stats(X_aprox)
stats_X2 <- calculate_stats(X_yeo)

print("Estadísticas con datos originales:")
```

```
## [1] "Estadísticas con datos originales:"
```

```r
print(stats_X)
```

```
##        Min        Max       Mean     Median     Q1.25%     Q3.75%
Skewness
##   0.000000 535.800000 101.658699  76.700000  25.900000 169.050000
1.080845
##   Kurtosis
##   1.395062
```

```r
print("Estadísticas con trasnformacion aproximada:")
```

```
## [1] "Estadísticas con trasnformacion aproximada:"
```

```r
print(stats_X1)
```

```
##        Min        Max       Mean     Median     Q1.25%     Q3.75%
Skewness
##  1.0000000 23.1689447  9.0571767  8.8147603  5.1865120 13.0402036
0.1533543
##   Kurtosis
## -0.7661745
```

```r
print("Estadísticas con trasnformacion Yeo Johnson:")
```

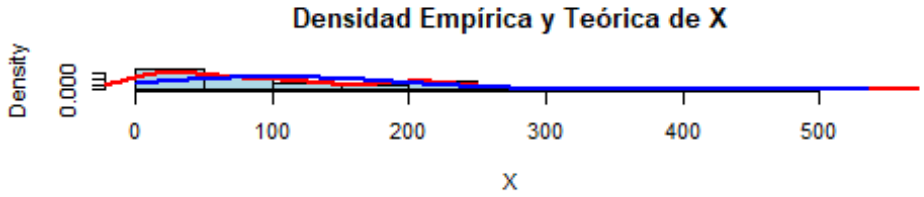
```
## [1] "Estadísticas con trasnformacion Yeo Johnson:"
```

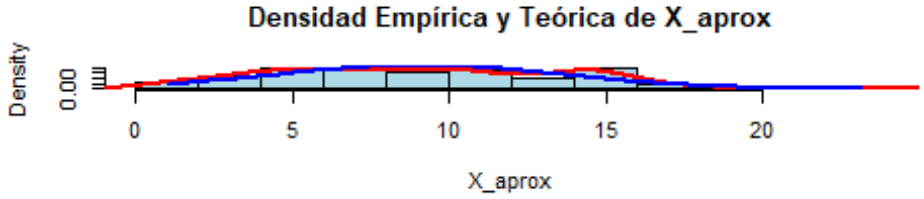
```r
print(stats_X2)
```

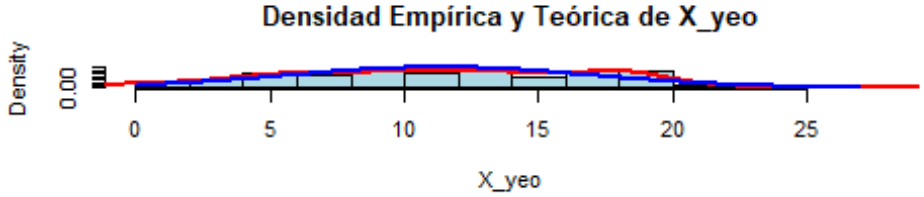
```
##         Min         Max        Mean      Median      Q1.25%
Q3.75%
##  0.00000000 26.95758194 11.31922399 11.37516751  6.66777687
16.32992577
##    Skewness    Kurtosis
## -0.07864157 -0.78686238
```

3.2 Grafica las funciones de densidad empírica y teórica de los 2 modelos obtenidos (exacto y aproximado) y los datos originales.

```r
library(MASS)

par(mfrow = c(3, 1))

# Graficar la densidad empírica y teórica para X
hist(X, freq = FALSE, main = "Densidad Empírica y Teórica de X", xlab =
"X", col = "lightblue", border = "black")
lines(density(X), col = "red", lwd = 2)
curve(dnorm(x, mean = mean(X), sd = sd(X)), from = min(X), to = max(X),
add = TRUE, col = "blue", lwd = 2)

# Graficar la densidad empírica y teórica para X_aprox
hist(X_aprox, freq = FALSE, main = "Densidad Empírica y Teórica de
X_aprox", xlab = "X_aprox", col = "lightblue", border = "black")
lines(density(X_aprox), col = "red", lwd = 2)
curve(dnorm(x, mean = mean(X_aprox), sd = sd(X_aprox)), from =
min(X_aprox), to = max(X_aprox), add = TRUE, col = "blue", lwd = 2)

# Graficar la densidad empírica y teórica para X_yeo
hist(X_yeo, freq = FALSE, main = "Densidad Empírica y Teórica de X_yeo",
xlab = "X_yeo", col = "lightblue", border = "black")
lines(density(X_yeo), col = "red", lwd = 2)
curve(dnorm(x, mean = mean(X_yeo), sd = sd(X_yeo)), from = min(X_yeo), to
= max(X_yeo), add = TRUE, col = "blue", lwd = 2)
```







3.3 Realiza la prueba de normalidad de Anderson-Darling y de Jarque Bera para los datos transformados y los originales

```
# Prueba para datos originales
cat("Prueba de Anderson-Darling para X:\n")

## Prueba de Anderson-Darling para X:

print(ad.test(X))

##
##  Anderson-Darling normality test
##
## data:  X
## A = 15.968, p-value < 2.2e-16

cat("\nPrueba de Jarque-Bera para X:\n")

##
## Prueba de Jarque-Bera para X:

jb_test_X <- jarque.bera.test(X)
print(jb_test_X)

##
##  Jarque Bera Test
##
```

```
## data:  X
## X-squared = 153.58, df = 2, p-value < 2.2e-16

# Prueba para datos transformados X_aprox
cat("\nPrueba de Anderson-Darling para X_aprox:\n")

##
## Prueba de Anderson-Darling para X_aprox:

print(ad.test(X_aprox))

##
##  Anderson-Darling normality test
##
## data:  X_aprox
## A = 4.0333, p-value = 4.785e-10

cat("\nPrueba de Jarque-Bera para X_aprox:\n")

##
## Prueba de Jarque-Bera para X_aprox:

jb_test_X_aprox <- jarque.bera.test(X_aprox)
print(jb_test_X_aprox)

##
##  Jarque Bera Test
##
## data:  X_aprox
## X-squared = 15.364, df = 2, p-value = 0.000461

# Prueba para datos transformados X_yeo
cat("\nPrueba de Anderson-Darling para X_yeo:\n")

##
## Prueba de Anderson-Darling para X_yeo:

print(ad.test(X_yeo))

##
##  Anderson-Darling normality test
##
## data:  X_yeo
## A = 3.4136, p-value = 1.517e-08

cat("\nPrueba de Jarque-Bera para X_yeo:\n")

##
## Prueba de Jarque-Bera para X_yeo:

jb_test_X_yeo <- jarque.bera.test(X_yeo)
print(jb_test_X_yeo)
```

```
##
##  Jarque Bera Test
##
## data:  X_yeo
## X-squared = 14.496, df = 2, p-value = 0.0007115
```

4. Detecta anomalías y corrige tu base de datos (datos atípicos, ceros anámalos, etc).

```r
# Calcular cuartiles y rango intercuartílico
Q1 <- quantile(X, 0.25, na.rm = TRUE)
Q3 <- quantile(X, 0.75, na.rm = TRUE)
IQR <- Q3 - Q1

# Definir los límites para identificar atípicos según IQR
lower_bound_iqr <- Q1 - 1.5 * IQR
upper_bound_iqr <- Q3 + 1.5 * IQR

# Detectar valores atípicos
atipicos_iqr <- X < lower_bound_iqr | X > upper_bound_iqr

# Calcular media y desviación estándar
mean_X <- mean(X, na.rm = TRUE)
sd_X <- sd(X, na.rm = TRUE)

# Definir los límites para 3 desviaciones estándar
lower_bound_sd <- mean_X - 3 * sd_X
upper_bound_sd <- mean_X + 3 * sd_X

# Detectar valores atípicos según desviaciones estándar
atipicos_sd <- X < lower_bound_sd | X > upper_bound_sd

# Combinar los criterios para detectar atípicos
atipicos_combined <- atipicos_iqr | atipicos_sd

# Crear X_clean sin datos atípicos
X_clean <- X[!atipicos_combined]

# Imprimir resultados
cat("Número total de datos antes de la limpieza:", length(X), "\n")
```

```
## Número total de datos antes de la limpieza: 551
```

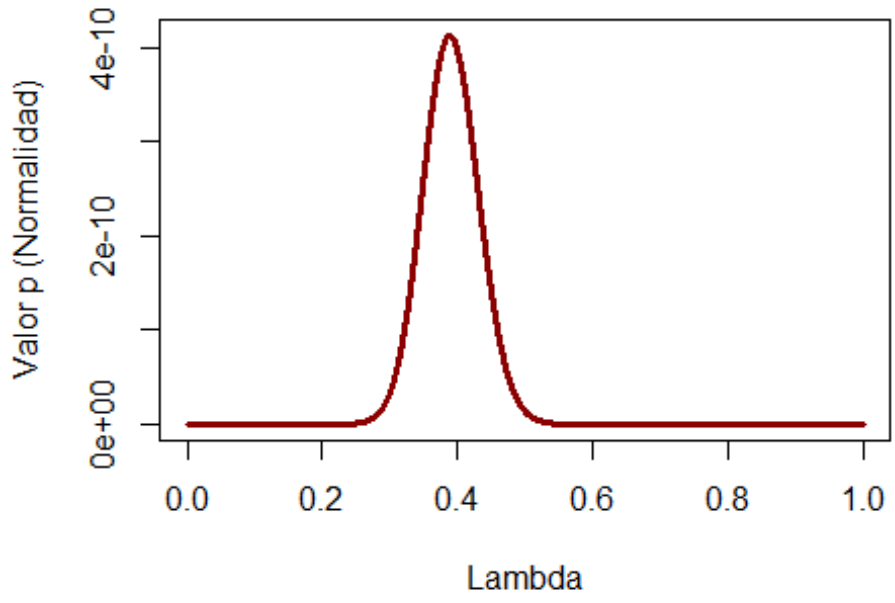
```r
cat("Número total de datos después de la limpieza:", length(X_clean),
"\n")
```

```
## Número total de datos después de la limpieza: 547
```

5. Comenta la normalidad de las transformaciones obtenidas. Utiliza como argumento de normalidad:

5.1 Compara las medidas: Mínimo, máximo, media, mediana, cuartil 1 y cuartil 3, sesgo y curtosis.

```
lp <- seq(0, 1, 0.001)  # Valores de lambda propuestos
nlp <- length(lp)
n <- length(X_clean)
D <- matrix(as.numeric(NA), ncol=2, nrow=nlp)
d <- NA

for (i in 1:nlp) {
  d <- yeo.johnson(X_clean, lambda = lp[i])
  p <- ad.test(d)
  D[i, ] <- c(lp[i], p$p.value)
}

# Convertir la matriz en un data frame y asignar nombres a las columnas
N <- as.data.frame(D)
colnames(N) <- c("Lambda", "P_value")

# Graficar
plot(N$Lambda, N$P_value, type="l",
     col="darkred", lwd=3,
     xlab="Lambda",
     ylab="Valor p (Normalidad)")
```




```
# Encontrar el índice del máximo valor p
max_index <- which.max(N$P_value)
```

```r
# Obtener el valor de lambda correspondiente
best_lambda <- N$Lambda[max_index]
max_p_value <- N$P_value[max_index]

# Imprimir los resultados
cat("El valor de lambda que maximiza el valor p es:", best_lambda, "\n")
```

```
## El valor de lambda que maximiza el valor p es: 0.388
```

```r
cat("El valor p máximo es:", max_p_value, "\n")
```

```
## El valor p máximo es: 4.120814e-10
```

```r
X1_aprox=sqrt(X_clean+1)
X1_yeo=yeo.johnson(X_clean, best_lambda)

# Función para calcular las medidas
calculate_stats <- function(x) {
  c(
    Min = min(x, na.rm = TRUE),
    Max = max(x, na.rm = TRUE),
    Mean = mean(x, na.rm = TRUE),
    Median = median(x, na.rm = TRUE),
    Q1 = quantile(x, 0.25, na.rm = TRUE),
    Q3 = quantile(x, 0.75, na.rm = TRUE),
    Skewness = skewness(x, na.rm = TRUE),
    Kurtosis = kurtosis(x, na.rm = TRUE)
  )
}

# Aplicar la función a cada variable
stats_X <- calculate_stats(X_clean)
stats_X1 <- calculate_stats(X1_aprox)
stats_X2 <- calculate_stats(X1_yeo)

print("Estadísticas con datos originales sin atípicos:")
```

```
## [1] "Estadísticas con datos originales sin atípicos:"
```

```r
print(stats_X)
```

```
##         Min         Max        Mean      Median      Q1.25%
Q3.75%
##   0.0000000 363.6000000  98.8726563  74.4000000  24.8500000
166.2000000
##    Skewness    Kurtosis
##   0.7068126  -0.5434420
```

```r
print("Estadísticas con trasnformacion aproximada sin atípicos:")
```

```
## [1] "Estadísticas con trasnformacion aproximada sin atípicos:"
```

```r
print(stats_X1)
```

```
##         Min         Max        Mean      Median      Q1.25%
Q3.75%
##  1.00000000 19.09450183  8.96268724  8.68331734  5.08343081
12.93058159
##    Skewness    Kurtosis
##  0.04738701 -1.03556229
```

```r
print("Estadísticas con trasnformacion Yeo Johnson sin atípicos:")
```

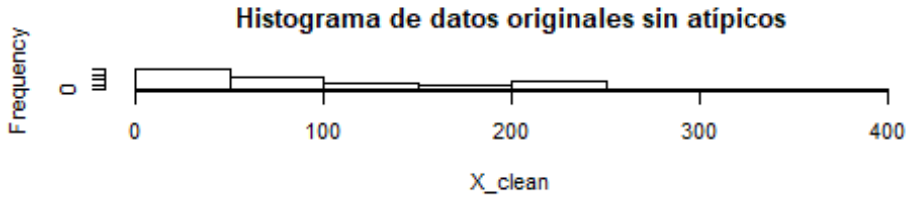
```
## [1] "Estadísticas con trasnformacion Yeo Johnson sin atípicos:"
```

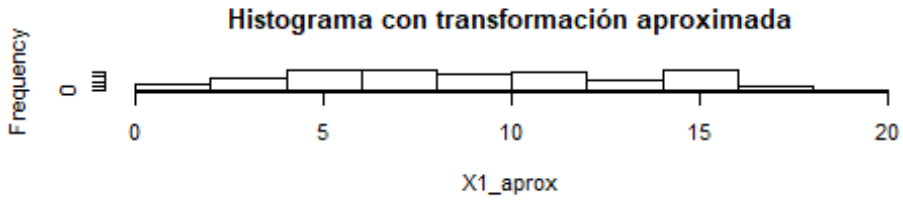
```r
print(stats_X2)
```

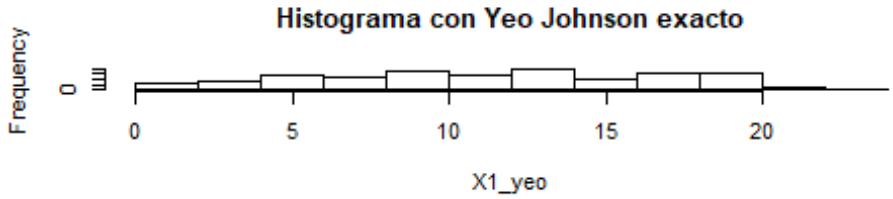
```
##        Min        Max       Mean     Median     Q1.25%     Q3.75%
Skewness
##  0.0000000 22.8413690 11.2135497 11.2134454  6.5246061 16.2064985 -
0.1561737
##   Kurtosis
## -0.9294953
```

5.2 Obten el histograma de los 2 modelos obtenidos (exacto y aproximado) y de los datos originales.

```r
par(mfrow=c(3,1))
hist(X_clean,col=0,main="Histograma de datos originales sin atípicos")
hist(X1_aprox,col=0,main="Histograma con transformación aproximada")
hist(X1_yeo,col=0,main="Histograma con Yeo Johnson exacto")
```







5.3 Interpreta la prueba de normalidad de Anderson-Darling y Jarque Bera para los datos transformados y los originales

```
# Prueba para datos originales
cat("Prueba de Anderson-Darling para X sin atípicos:\n")

## Prueba de Anderson-Darling para X sin atípicos:

print(ad.test(X_clean))

##
##  Anderson-Darling normality test
##
## data:  X_clean
## A = 16.478, p-value < 2.2e-16

cat("\nPrueba de Jarque-Bera para X:\n")

##
## Prueba de Jarque-Bera para X:

jb_test_X <- jarque.bera.test(X_clean)
print(jb_test_X)

##
##  Jarque Bera Test
##
```

```
## data:  X_clean
## X-squared = 52.306, df = 2, p-value = 4.384e-12

# Prueba para datos transformados X_aprox
cat("\nPrueba de Anderson-Darling para X_aprox sin atípicos:\n")

##
## Prueba de Anderson-Darling para X_aprox sin atípicos:

print(ad.test(X1_aprox))

##
##  Anderson-Darling normality test
##
## data:  X1_aprox
## A = 4.6714, p-value = 1.382e-11

cat("\nPrueba de Jarque-Bera para X_aprox:\n")

##
## Prueba de Jarque-Bera para X_aprox:

jb_test_X_aprox <- jarque.bera.test(X1_aprox)
print(jb_test_X_aprox)

##
##  Jarque Bera Test
##
## data:  X1_aprox
## X-squared = 24.309, df = 2, p-value = 5.266e-06

# Prueba para datos transformados X_yeo
cat("\nPrueba de Anderson-Darling para X_yeo sin atípicos:\n")

##
## Prueba de Anderson-Darling para X_yeo sin atípicos:

print(ad.test(X1_yeo))

##
##  Anderson-Darling normality test
##
## data:  X1_yeo
## A = 4.0601, p-value = 4.121e-10

cat("\nPrueba de Jarque-Bera para X_yeo:\n")

##
## Prueba de Jarque-Bera para X_yeo:

jb_test_X_yeo <- jarque.bera.test(X1_yeo)
print(jb_test_X_yeo)
```

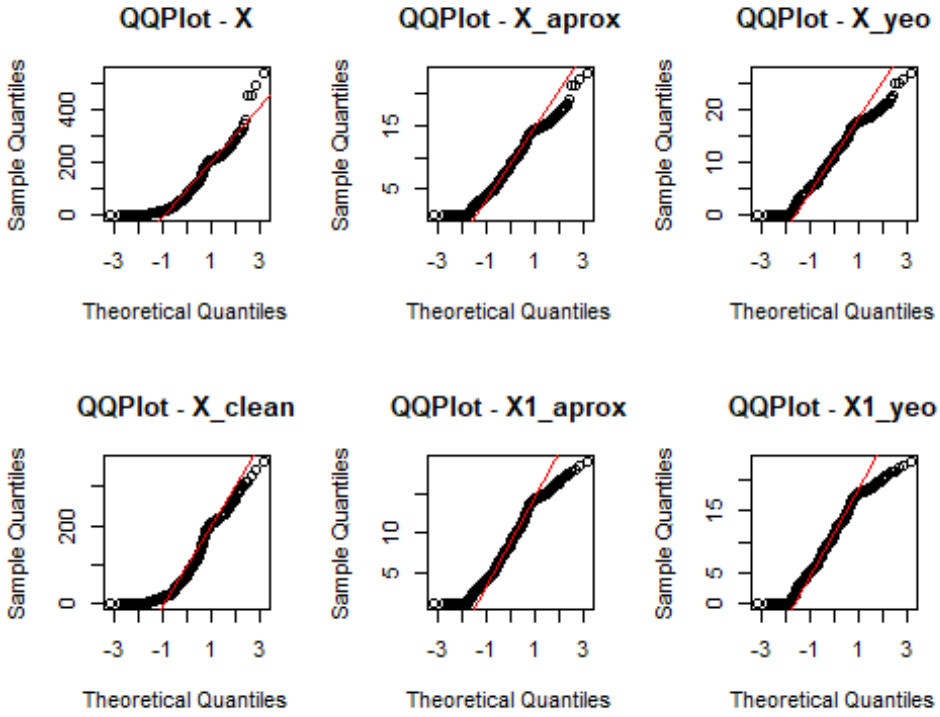
```
##
##  Jarque Bera Test
##
## data:  X1_yeo
## X-squared = 21.607, df = 2, p-value = 2.033e-05
```

5.4 Indica posibilidades de motivos de alejamiento de normalidad (sesgo, curtosis, datos atípicos, etc

Las transformaciones aplicadas (aproximada y Yeo-Johnson) y la eliminación de datos atípicos han mejorado la simetría y la distribución de los datos, pero las pruebas de normalidad siguen indicando que los datos no son completamente normales. Esto sugiere que, a pesar de las mejoras, podría haber características inherentes en los datos que causan desviaciones persistentes de la normalidad, como colas pesadas o sesgo residual.

6. Define la mejor transformación de los datos de acuerdo a las características de los modelos que encontraste. Toma en cuenta los criterios del inciso anterior para analizar normalidad y la economía del modelo.

```r
# Crear gráficos QQPlot para cada variable en base R

# Definir una función para crear el QQPlot
plot_qq <- function(data, title) {
  qqnorm(data, main = title)
  qqline(data, col = "red")
}

# Configurar la ventana de gráficos para mostrar 3x2 gráficos
par(mfrow = c(2, 3))

# Crear los gráficos QQPlot
plot_qq(X, "QQPlot - X")
plot_qq(X_aprox, "QQPlot - X_aprox")
plot_qq(X_yeo, "QQPlot - X_yeo")
plot_qq(X_clean, "QQPlot - X_clean")
plot_qq(X1_aprox, "QQPlot - X1_aprox")
plot_qq(X1_yeo, "QQPlot - X1_yeo")
```




```
# Restaurar la configuración de gráficos por defecto
par(mfrow = c(1, 1))
```

La mejor transformación de acuerdo a los criterios de normalidad y economía del modelo es la Transformación Yeo-Johnson. Aunque ambas transformaciones (aproximada y Yeo-Johnson) muestran mejoras en sesgo y curtosis, la Yeo-Johnson:

Mejor Normalización: Aunque no elimina completamente la desviación de normalidad, ofrece una distribución más cercana a la normalidad que la aproximada según las medidas obtenidas como sesgo y curtosis.

Economía: Aunque puede ser más compleja, la transformación Yeo-Johnson proporciona una normalización más robusta en contextos diversos.

Recomendación: Utilizar la transformación Yeo-Johnson para tus datos, ya que, a pesar de las pruebas de normalidad indicando desviaciones, esta transformación ofrece una mejor aproximación a la normalidad y es más flexible para diferentes tipos de datos.