A3-Regresión Múltiple-Detección datos atípicos

Luis Maximiliano López Ramírez

2024-09-24

En la base de datos Al corte se describe un experimento realizado para evaluar el impacto de las variables: fuerza, potencia, temperatura y tiempo sobre la resistencia al corte. Indica cuál es la mejor relación entre estas variables que describen la resistencia al corte.

```
datos <- read.csv("AlCorte.csv")
```

 # 1. Haz un análisis descriptivo de los datos: medidas principales y gráficos

```
# Medidas principales
summary(datos)  # Resumen estadístico de todas las columnas
```

```
##      Fuerza      Potencia    Temperatura      Tiempo    Resistencia   
##  Min.   :25   Min.   : 45   Min.   :150   Min.   :10   Min.   :22.70  
##  1st Qu.:30   1st Qu.: 60   1st Qu.:175   1st Qu.:15   1st Qu.:34.67  
##  Median :35   Median : 75   Median :200   Median :20   Median :38.60  
##  Mean   :35   Mean   : 75   Mean   :200   Mean   :20   Mean   :38.41  
##  3rd Qu.:40   3rd Qu.: 90   3rd Qu.:225   3rd Qu.:25   3rd Qu.:42.70  
##  Max.   :45   Max.   :105   Max.   :250   Max.   :30   Max.   :58.70
```

```
# Calcular estadísticas descriptivas para variables numéricas
library(dplyr)  # Para manipulación de datos
```

```
## Warning: package 'dplyr' was built under R version 4.3.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(psych)  # Para funciones de análisis descriptivo más detalladas
```

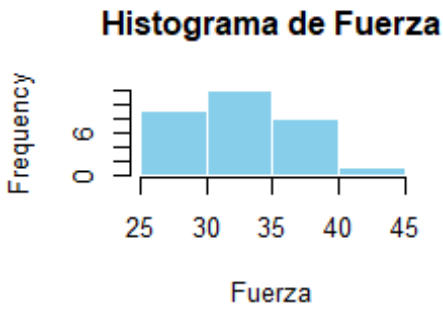
```
## Warning: package 'psych' was built under R version 4.3.2
```

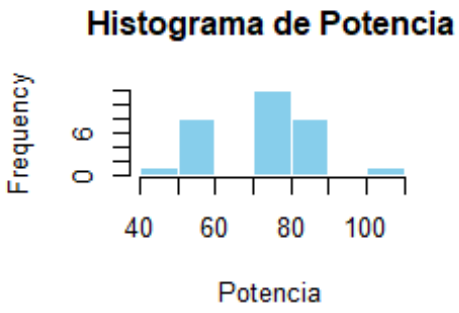
```
# Seleccionar solo las columnas numéricas
numericos <- datos %>% select(where(is.numeric))

# Estadísticas descriptivas detalladas
```

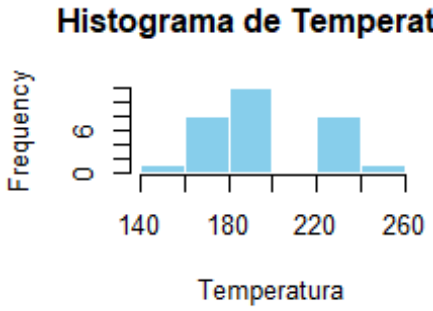
```r
descriptivas <- psych::describe(numericos)
print(descriptivas)
```

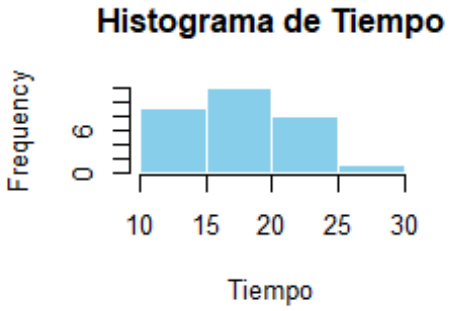
```
##             vars  n   mean    sd median trimmed   mad   min   max
range skew
## Fuerza         1 30  35.00  4.55   35.0   35.00  7.41  25.0  45.0
20 0.00
## Potencia       2 30  75.00 13.65   75.0   75.00 22.24  45.0 105.0
60 0.00
## Temperatura    3 30 200.00 22.74  200.0  200.00 37.06 150.0 250.0
100 0.00
## Tiempo         4 30  20.00  4.55   20.0   20.00  7.41  10.0  30.0
20 0.00
## Resistencia    5 30  38.41  8.95   38.6   38.05  6.30  22.7  58.7
36 0.22
##             kurtosis   se
## Fuerza         -0.66 0.83
## Potencia       -0.66 2.49
## Temperatura    -0.66 4.15
## Tiempo         -0.66 0.83
## Resistencia    -0.35 1.63
```

```r
# Gráficos

# Histograma de todas las variables numéricas
par(mfrow = c(2, 2))  # Dividir la ventana gráfica en una cuadrícula de
2x2
for(col in colnames(numericos)) {
  hist(numericos[[col]], main = paste("Histograma de", col), xlab = col,
col = "skyblue", border = "white")
}
```

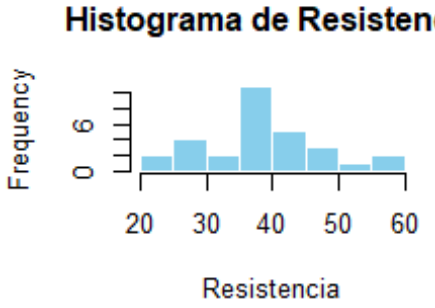




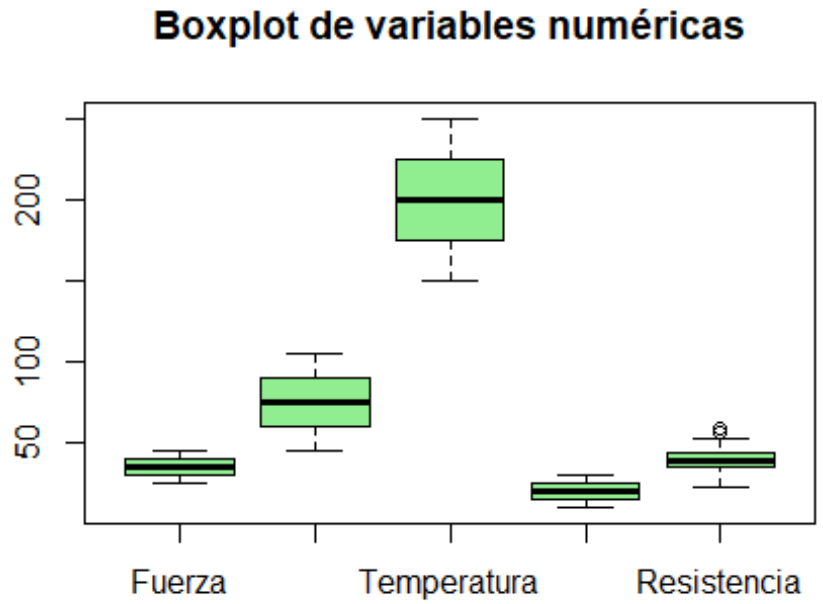




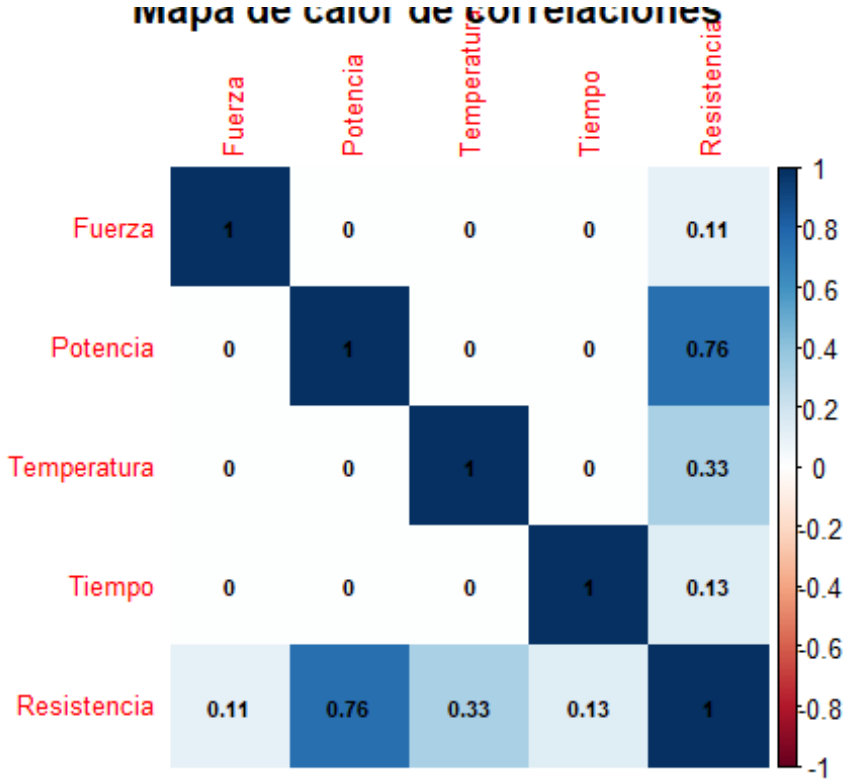

```r
# Boxplot de todas las variables numéricas
par(mfrow = c(1, 1))  # Restablecer la ventana gráfica
```



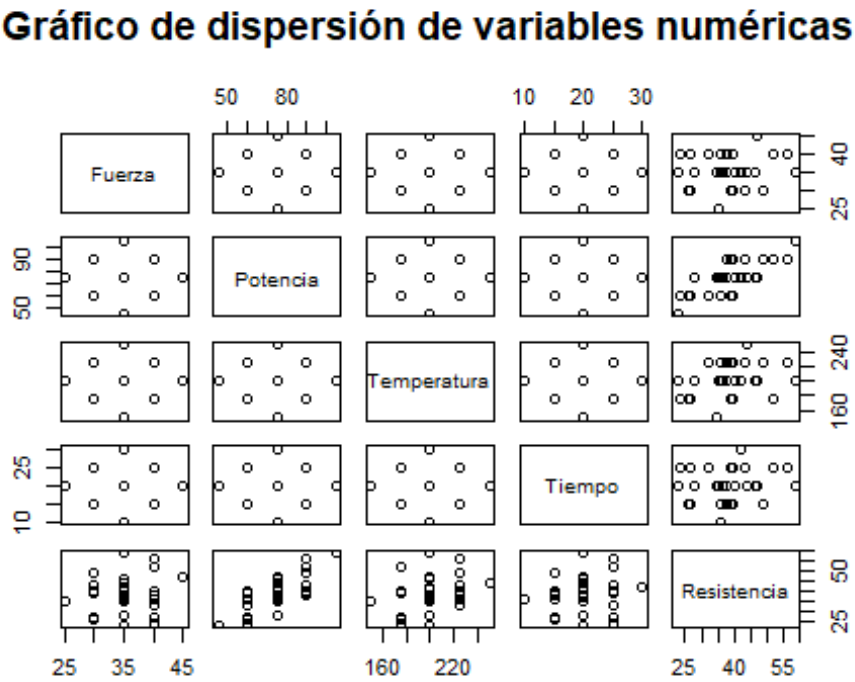
```r
boxplot(numericos, main = "Boxplot de variables numéricas", col =
"lightgreen")
```




```r
# Mapa de calor de correlaciones
library(corrplot)

## Warning: package 'corrplot' was built under R version 4.3.2

## corrplot 0.92 loaded

correlaciones <- cor(numericos, use = "complete.obs")
corrplot(correlaciones, method = "color", addCoef.col = "black", tl.cex =
0.8, number.cex = 0.7, main = "Mapa de calor de correlaciones")
```




```r
# Gráfico de dispersión para explorar relaciones entre variables
pairs(numericos, main = "Gráfico de dispersión de variables numéricas")
```



 # 2. Encuentra el mejor modelo de regresión que explique la variable Resistencia. Analiza el modelo basándote en significancia del modelo

```
# Definir el modelo nulo (sin variables predictoras) y el modelo completo
(con todas las variables predictoras)
modelo_nulo <- lm(Resistencia ~ 1, data = datos)  # Modelo nulo con solo
la media
modelo_completo <- lm(Resistencia ~ Fuerza + Potencia + Temperatura +
Tiempo, data = datos)  # Modelo completo con todas las variables

# Selección de variables utilizando el método "forward" (de abajo hacia
arriba)
Model_forward <- step(modelo_nulo, scope = list(lower = modelo_nulo,
upper = modelo_completo), direction = "forward")
```

```
## Start:  AIC=132.51
## Resistencia ~ 1
##
##               Df Sum of Sq     RSS    AIC
## + Potencia     1   1341.01  984.24 108.72
## + Temperatura  1    252.20 2073.06 131.07
## <none>                     2325.26 132.51
## + Tiempo       1     40.04 2285.22 133.99
## + Fuerza       1     26.88 2298.38 134.16
##
## Step:  AIC=108.72
## Resistencia ~ Potencia
##
##               Df Sum of Sq    RSS    AIC
## + Temperatura  1   252.202 732.04 101.84
## <none>                     984.24 108.72
## + Tiempo       1    40.042 944.20 109.47
## + Fuerza       1    26.882 957.36 109.89
##
## Step:  AIC=101.84
## Resistencia ~ Potencia + Temperatura
##
##          Df Sum of Sq    RSS    AIC
## <none>                 732.04 101.84
## + Tiempo  1    40.042 692.00 102.15
## + Fuerza  1    26.882 705.16 102.72
```

```
summary(Model_forward)
```

```
##
## Call:
## lm(formula = Resistencia ~ Potencia + Temperatura, data = datos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.3233  -2.8067  -0.8483   3.1892   9.4600
```

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -24.90167   10.07207  -2.472  0.02001 *
## Potencia      0.49833    0.07086   7.033 1.47e-07 ***
## Temperatura   0.12967    0.04251   3.050  0.00508 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.207 on 27 degrees of freedom
## Multiple R-squared:  0.6852, Adjusted R-squared:  0.6619
## F-statistic: 29.38 on 2 and 27 DF,  p-value: 1.674e-07
```

```
# Selección de variables utilizando el método "backward" (de arriba hacia abajo)
Model_backward <- step(modelo_completo, direction = "backward")
```

```
## Start:  AIC=102.96
## Resistencia ~ Fuerza + Potencia + Temperatura + Tiempo
##
##               Df Sum of Sq     RSS    AIC
## - Fuerza       1     26.88  692.00 102.15
## - Tiempo       1     40.04  705.16 102.72
## <none>                      665.12 102.96
## - Temperatura  1    252.20  917.32 110.61
## - Potencia     1   1341.01 2006.13 134.08
##
## Step:  AIC=102.15
## Resistencia ~ Potencia + Temperatura + Tiempo
##
##               Df Sum of Sq     RSS    AIC
## - Tiempo       1     40.04  732.04 101.84
## <none>                      692.00 102.15
## - Temperatura  1    252.20  944.20 109.47
## - Potencia     1   1341.02 2033.02 132.48
##
## Step:  AIC=101.84
## Resistencia ~ Potencia + Temperatura
##
##               Df Sum of Sq     RSS    AIC
## <none>                      732.04 101.84
## - Temperatura  1     252.2  984.24 108.72
## - Potencia     1    1341.0 2073.06 131.07
```

```
summary(Model_backward)
```

```
##
## Call:
## lm(formula = Resistencia ~ Potencia + Temperatura, data = datos)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -11.3233  -2.8067  -0.8483   3.1892   9.4600
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -24.90167   10.07207  -2.472  0.02001 *
## Potencia      0.49833    0.07086   7.033 1.47e-07 ***
## Temperatura   0.12967    0.04251   3.050  0.00508 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.207 on 27 degrees of freedom
## Multiple R-squared:  0.6852, Adjusted R-squared:  0.6619
## F-statistic: 29.38 on 2 and 27 DF,  p-value: 1.674e-07

# Selección de variables utilizando el método "both" (combinación de
"forward" y "backward")
Model_both <- step(modelo_completo, direction = "both", trace = 1)

## Start:  AIC=102.96
## Resistencia ~ Fuerza + Potencia + Temperatura + Tiempo
##
##               Df Sum of Sq     RSS    AIC
## - Fuerza       1     26.88  692.00 102.15
## - Tiempo       1     40.04  705.16 102.72
## <none>                      665.12 102.96
## - Temperatura  1    252.20  917.32 110.61
## - Potencia     1   1341.01 2006.13 134.08
##
## Step:  AIC=102.15
## Resistencia ~ Potencia + Temperatura + Tiempo
##
##               Df Sum of Sq     RSS    AIC
## - Tiempo       1     40.04  732.04 101.84
## <none>                      692.00 102.15
## + Fuerza       1     26.88  665.12 102.96
## - Temperatura  1    252.20  944.20 109.47
## - Potencia     1   1341.02 2033.02 132.48
##
## Step:  AIC=101.84
## Resistencia ~ Potencia + Temperatura
##
##               Df Sum of Sq     RSS    AIC
## <none>                      732.04 101.84
## + Tiempo       1     40.04  692.00 102.15
## + Fuerza       1     26.88  705.16 102.72
## - Temperatura  1    252.20  984.24 108.72
## - Potencia     1   1341.01 2073.06 131.07

summary(Model_both)
```

```
##
## Call:
## lm(formula = Resistencia ~ Potencia + Temperatura, data = datos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.3233  -2.8067  -0.8483   3.1892   9.4600
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -24.90167   10.07207  -2.472  0.02001 *
## Potencia      0.49833    0.07086   7.033 1.47e-07 ***
## Temperatura   0.12967    0.04251   3.050  0.00508 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.207 on 27 degrees of freedom
## Multiple R-squared:  0.6852, Adjusted R-squared:  0.6619
## F-statistic: 29.38 on 2 and 27 DF,  p-value: 1.674e-07
```

1. Economía de las variables

```r
# Calcular AIC y BIC para cada modelo
calcular_aic_bic <- function(modelo) {
  aic <- extractAIC(modelo)[2]                # AIC con k = 2
  bic <- extractAIC(modelo, k = log(n))[2]   # BIC con k = log(n)
  return(c(AIC = aic, BIC = bic))
}

# Definir el número de observaciones (n)
n <- nrow(datos)  # Asegúrate de que 'datos' es el nombre de tu DataFrame


# Obtener los valores de AIC y BIC para cada modelo
aic_bic_forward <- calcular_aic_bic(Model_forward)
aic_bic_backward <- calcular_aic_bic(Model_backward)
aic_bic_both <- calcular_aic_bic(Model_both)
aic_bic_completo <- calcular_aic_bic(modelo_completo)

# Crear la tabla comparativa
comparacion_modelos <- data.frame(
  Modelo = c("Forward", "Backward", "Both", "Completo"),
  R2 = c(summary(Model_forward)$r.squared,
         summary(Model_backward)$r.squared,
         summary(Model_both)$r.squared,
         summary(modelo_completo)$r.squared),
  AIC = c(aic_bic_forward["AIC"],
          aic_bic_backward["AIC"],
          aic_bic_both["AIC"],
          aic_bic_completo["AIC"]),
```

```
  BIC = c(aic_bic_forward["BIC"],
          aic_bic_backward["BIC"],
          aic_bic_both["BIC"],
          aic_bic_completo["BIC"])
)

# Mostrar la tabla comparativa
print(comparacion_modelos)
```

```
##     Modelo        R2      AIC      BIC
## 1  Forward 0.6851783 101.8392 106.0428
## 2 Backward 0.6851783 101.8392 106.0428
## 3     Both 0.6851783 101.8392 106.0428
## 4 Completo 0.7139593 102.9630 109.9690
```

Observando los resultados se puede observar que el mejor modelo por economía es el de Resistencia = Potencia + Temperatura ya que solamente tiene 2 variables predictoras y si bien es cierto que tiene menor coeficiente de determinación que el modelo completo, éste tiene menor AIC y BIC, por lo que lo que resulta mejor el modelo anterior.

2. Significación global (Prueba para el modelo)

```
# Ajustar el modelo lineal
modelo <- lm(formula = Resistencia ~ Potencia + Temperatura, data =
datos)

# Resumen del modelo
resumen_modelo <- summary(modelo)

# Extraer la significancia global (valor p asociado al estadístico F)
p_valor_global <- resumen_modelo$fstatistic[1]
df1 <- resumen_modelo$fstatistic[2]
df2 <- resumen_modelo$fstatistic[3]

# Calcular el valor p a partir de la estadística F
p_valor <- pf(p_valor_global, df1, df2, lower.tail = FALSE)
cat("Significancia global del modelo (valor p):", p_valor, "\n")
```

```
## Significancia global del modelo (valor p): 1.674456e-07
```

```
# Regla de decisión con un nivel de significancia de 0.05
nivel_significancia <- 0.05

if (p_valor < nivel_significancia) {
  cat("El modelo es significativo a un nivel de", nivel_significancia,
"\n")
} else {
  cat("El modelo NO es significativo a un nivel de", nivel_significancia,
```

```
"\n")
}

## El modelo es significativo a un nivel de 0.05
```

3. Significación individual (Prueba para cada □□)

```
# Extraer los valores p para cada coeficiente
p_valores_coef <- resumen_modelo$coefficients[, 4]  # Los valores p están
en la cuarta columna de los coeficientes

# Nivel de significancia
nivel_significancia <- 0.05

# Probar la significancia de cada coeficiente
for (i in 1:length(p_valores_coef)) {
  if (p_valores_coef[i] < nivel_significancia) {
    cat(rownames(resumen_modelo$coefficients)[i], "es significativo con
p-valor de", p_valores_coef[i], "\n")
  } else {
    cat(rownames(resumen_modelo$coefficients)[i], "NO es significativo
con p-valor de", p_valores_coef[i], "\n")
  }
}

## (Intercept) es significativo con p-valor de 0.02001412
## Potencia es significativo con p-valor de 1.46543e-07
## Temperatura es significativo con p-valor de 0.005082118
```

4. Variación explicada por el modelo

```
# Obtener el R² directamente del resumen
r_cuadrado <- resumen_modelo$r.squared
cat("El coeficiente de determinación (R²) es:", r_cuadrado, "\n")

## El coeficiente de determinación (R²) es: 0.6851783
```

 # 3. Analiza la validez del modelo encontrado:

1. Análisis de residuos (homocedasticidad, independencia, etc)

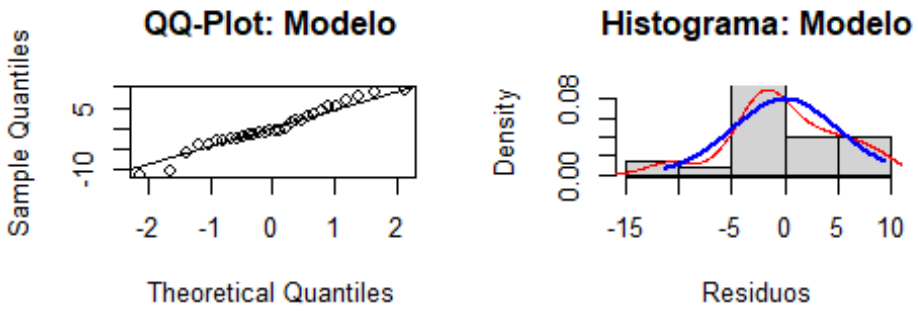
1. Normalidad de los residuos

% Hipótesis de Normalidad (Anderson-Darling)

$$\begin{cases} H_0: \text{Los residuos siguen una distribución normal.} \\ H_1: \text{Los residuos no siguen una distribución normal.} \end{cases}$$

```
# Cargar la librería para la prueba Anderson-Darling
library(nortest)

# Realizar la prueba de Anderson-Darling sobre los residuos de los
```

```r
modelos
resultado_modelo <- ad.test(modelo$residuals)

# Mostrar los resultados de las pruebas
cat("Resultados de la prueba de normalidad Anderson-Darling:\n")

## Resultados de la prueba de normalidad Anderson-Darling:

cat("\nModelo:\n")

##
## Modelo:

print(resultado_modelo)

##
##  Anderson-Darling normality test
##
## data:  modelo$residuals
## A = 0.41149, p-value = 0.3204

# Verificar normalidad en base al valor p
if (resultado_modelo$p.value > 0.05) {
  cat("\nSe tiene normalidad en el Modelo.\n")
} else {
  cat("\nNo se tiene normalidad en el Modelo.\n")
}

##
## Se tiene normalidad en el Modelo.

# Gráficos de normalidad y distribuciones para Modelo 1
par(mfrow = c(2, 2))  # Configurar para mostrar 4 gráficos en una
cuadrícula 2x2

# Calcular los límites del eje Y para los histogramas
ylim_1 <- range(density(modelo$residuals)$y)

# QQ-Plot del Modelo 1
qqnorm(modelo$residuals, main = "QQ-Plot: Modelo")
qqline(modelo$residuals)

# Histograma del Modelo 1
hist(modelo$residuals, freq = FALSE, ylim = ylim_1,
     main = "Histograma: Modelo", xlab = "Residuos")
lines(density(modelo$residuals), col = "red")
curve(dnorm(x, mean = mean(modelo$residuals), sd = sd(modelo$residuals)),
      from = min(modelo$residuals), to = max(modelo$residuals), add =
TRUE, col = "blue", lwd = 2)
```



2. Verificación de media cero

% Hipótesis de Media Diferente de Cero (t de Student)

$$\begin{cases} H_0: \mu = 0 & \text{(El promedio de los residuos es igual a cero)} \\ H_1: \mu \neq 0 & \text{(El promedio de los residuos es diferente de cero)} \end{cases}$$

```
# Realiza la prueba t para los residuos de los modelos sin y con
interacción
resultado_t_modelo <- t.test(modelo$residuals)

# Mostrar los resultados de las pruebas t
cat("Resultados de la prueba t de Student para los residuos:\n")

## Resultados de la prueba t de Student para los residuos:

cat("\nModelo:\n")

##
## Modelo:

print(resultado_t_modelo)

##
##  One Sample t-test
##
## data:  modelo$residuals
## t = 8.8667e-17, df = 29, p-value = 1
```

```
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -1.876076  1.876076
## sample estimates:
##    mean of x
## 8.133323e-17

# Verificar si el promedio de los residuos es significativamente
diferente de cero
if (resultado_t_modelo$p.value > 0.05) {
  cat("\nEl promedio de los residuos del Modelo no es significativamente
diferente de cero.\n")
} else {
  cat("\nEl promedio de los residuos del Modelo es significativamente
diferente de cero.\n")
}

##
## El promedio de los residuos del Modelo no es significativamente
diferente de cero.
```

3. Homocedasticidad, linealidad e independencia

% Hipótesis de Autocorrelación (Durbin-Watson y Breusch-Godfrey)

$$\begin{cases} H_0: \text{Los errores no están autocorrelacionados (independencia).} \\ H_1: \text{Los errores están autocorrelacionados.} \end{cases}$$

% Hipótesis de Homocedasticidad (Breusch-Pagan y Goldfeld-Quandt)

$$\begin{cases} H_0: \text{La varianza de los errores es constante (homocedasticidad).} \\ H_1: \text{La varianza de los errores no es constante (heterocedasticidad).} \end{cases}$$

```
# Cargar la librería necesaria
library(lmtest)

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

# Gráficos de residuos contra valores ajustados para ambos modelos
par(mfrow = c(2, 2))  # Configurar para mostrar 4 gráficos en una
cuadrícula 2x2

# Modelo 1: Sin Interacción
plot(modelo$fitted.values, modelo$residuals,
     main = "Residuos vs. Ajustados: Modelo 1 (Sin Interacción)",
```

```r
     xlab = "Valores Ajustados", ylab = "Residuos", pch = 19, col =
"blue")
abline(h = 0, col = "red", lwd = 2)

# Pruebas de autocorrelación de errores
cat("Pruebas de autocorrelación de errores o independencia:\n")
```

```
## Pruebas de autocorrelación de errores o independencia:
```

```r
# Modelo 1: Sin Interacción
dw_modelo <- dwtest(modelo)
bg_modelo <- bgtest(modelo)

cat("\nModelo:\n")
```

```
##
## Modelo:
```

```r
# Verificar autocorrelación en base al valor p
if (dw_modelo$p.value > 0.05) {
  cat("\nNo se rechaza H₀: Los errores no están autocorrelacionados
(Durbin-Watson).\nTiene independencia.\n")
} else {
  cat("\nSe rechaza H₀: Los errores están autocorrelacionados (Durbin-
Watson).\nNo tiene independencia.\n")
}
```

```
##
## No se rechaza H₀: Los errores no están autocorrelacionados (Durbin-
Watson).
## Tiene independencia.
```

```r
if (bg_modelo$p.value > 0.05) {
  cat("No se rechaza H₀: Los errores no están autocorrelacionados
(Breusch-Godfrey).\nTiene independencia.\n")
} else {
  cat("Se rechaza H₀: Los errores están autocorrelacionados (Breusch-
Godfrey).\nNo tiene independencia.\n")
}
```

```
## No se rechaza H₀: Los errores no están autocorrelacionados (Breusch-
Godfrey).
## Tiene independencia.
```

```r
# Pruebas de homocedasticidad
cat("\nPruebas de homocedasticidad:\n")
```

```
##
## Pruebas de homocedasticidad:
```

```r
# Modelo 1: Sin Interacción
bp_modelo <- bptest(modelo)
```

```r
gq_modelo <- gqtest(modelo)

cat("\nModelo:\n")
```

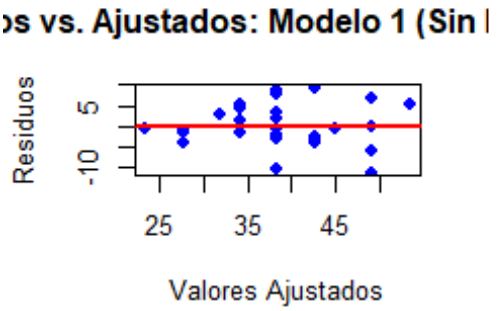
```
##
## Modelo:
```

```r
# Verificar homocedasticidad en base al valor p
if (bp_modelo$p.value > 0.05) {
  cat("\nNo se rechaza H₀: La varianza de los errores es constante
(Breusch-Pagan).\nTiene homocedasticidad.\n")
} else {
  cat("\nSe rechaza H₀: La varianza de los errores no es constante
(Breusch-Pagan).\nNo tiene homocedasticidad.\n")
}
```

```
##
## No se rechaza H₀: La varianza de los errores es constante (Breusch-
Pagan).
## Tiene homocedasticidad.
```

```r
if (gq_modelo$p.value > 0.05) {
  cat("No se rechaza H₀: La varianza de los errores es constante
(Goldfeld-Quandt).\nTiene homocedasticidad.\n")
} else {
  cat("Se rechaza H₀: La varianza de los errores no es constante
(Goldfeld-Quandt).\nNo tiene homocedasticidad.\n")
}
```

```
## No se rechaza H₀: La varianza de los errores es constante (Goldfeld-
Quandt).
## Tiene homocedasticidad.
```



2. No multicolinealidad de Xi

```
# Cargar el paquete 'car'
library(car)

## Warning: package 'car' was built under R version 4.3.2

## Loading required package: carData

## Warning: package 'carData' was built under R version 4.3.2

##
## Attaching package: 'car'

## The following object is masked from 'package:psych':
##
##     logit

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

# Calcular el VIF
vif_valores <- vif(modelo)

# Mostrar los valores de VIF
print(vif_valores)
```

```
##    Potencia Temperatura
##           1           1
```

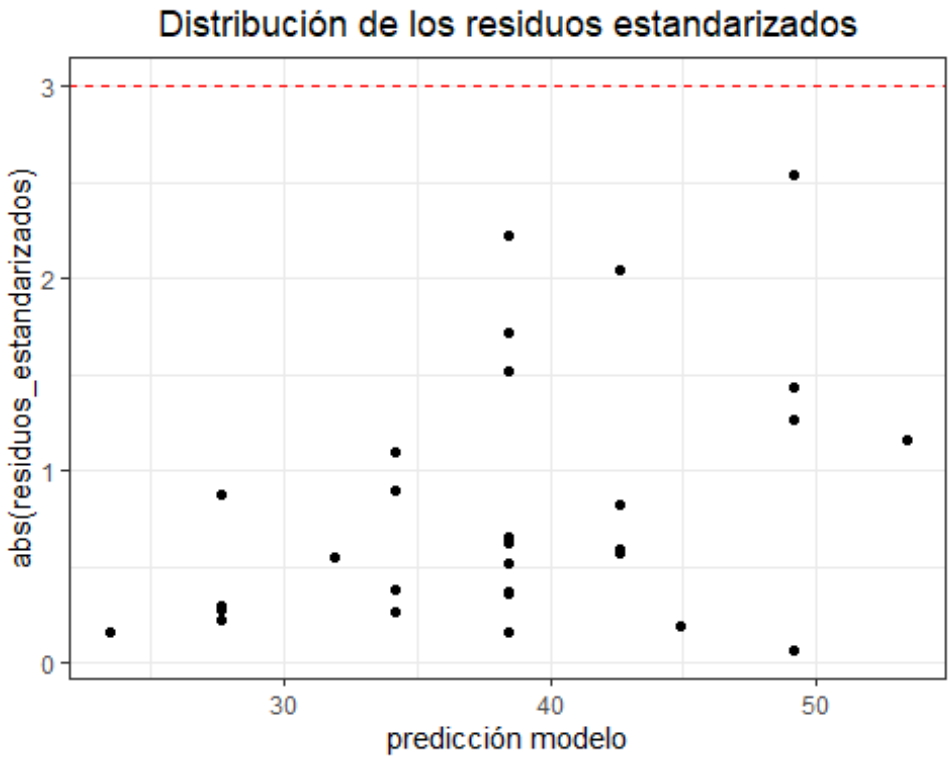
VIF < 5: No hay una multicolinealidad significativa entre las variables. Los valores en este rango suelen ser aceptables.

 # 4. Haz el análisis de datos atípicos e influyentes del mejor modelo encontrado

1. Datos atípicos

1. Estandarización extrema de los residuos

```
Datos = datos

library(dplyr)
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.3.2

##
## Attaching package: 'ggplot2'

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

Datos$residuos_estandarizados <- rstudent(modelo)
#Introduce una columna en Datos con los residuos estandarizados de los n
datos

#Gráfico auxiliar:
ggplot(data = Datos, aes(x = predict(modelo), y =
abs(residuos_estandarizados))) +
geom_hline(yintercept = 3, color = "red", linetype = "dashed") +
# se identifican en rojo observaciones con residuos estandarizados
absolutos > 3
geom_point(aes(color = ifelse(abs(residuos_estandarizados) > 3, 'red',
'black'))) +
scale_color_identity() +
labs(title = "Distribución de los residuos estandarizados", x =
"predicción modelo") +
theme_bw() + theme(plot.title = element_text(hjust = 0.5))
```

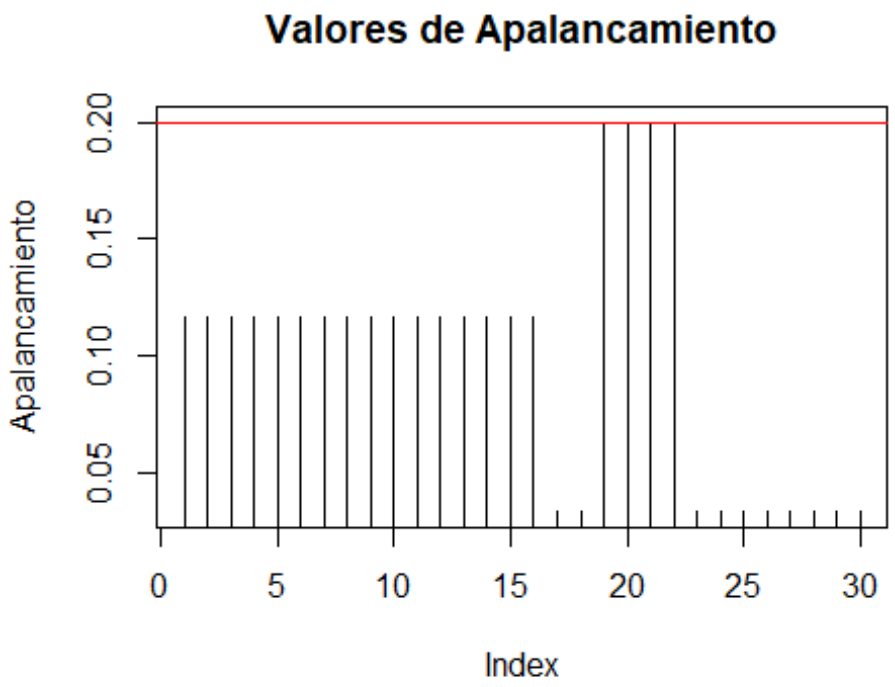



```
#Cuenta e identifica cuántos datos atípicos hay:
Atipicos = which(abs(Datos$residuos_estandarizados)>3)

#Muestra las observaciones con altos residuos estandarizados
Datos[Atipicos, ]
```

```
## [1] Fuerza                  Potencia                Temperatura
## [4] Tiempo                  Resistencia
residuos_estandarizados
## <0 rows> (or 0-length row.names)
```

2. Distancia de Leverage

```
leverage = hatvalues(modelo)
#Calcula el leverage de los n datos

# Gráfico auxiliar:
plot(leverage, type="h", main="Valores de Apalancamiento",
ylab="Apalancamiento")
abline(h = 2*mean(leverage), col="red") # Límite comúnmente usado
```




```
# Cuenta e identifica cuántos datos atípicos hay:
high_leverage_points = which(leverage > 1.99*mean(leverage))

#Muestra las observaciones con alto leverage
Datos[high_leverage_points, ]
```

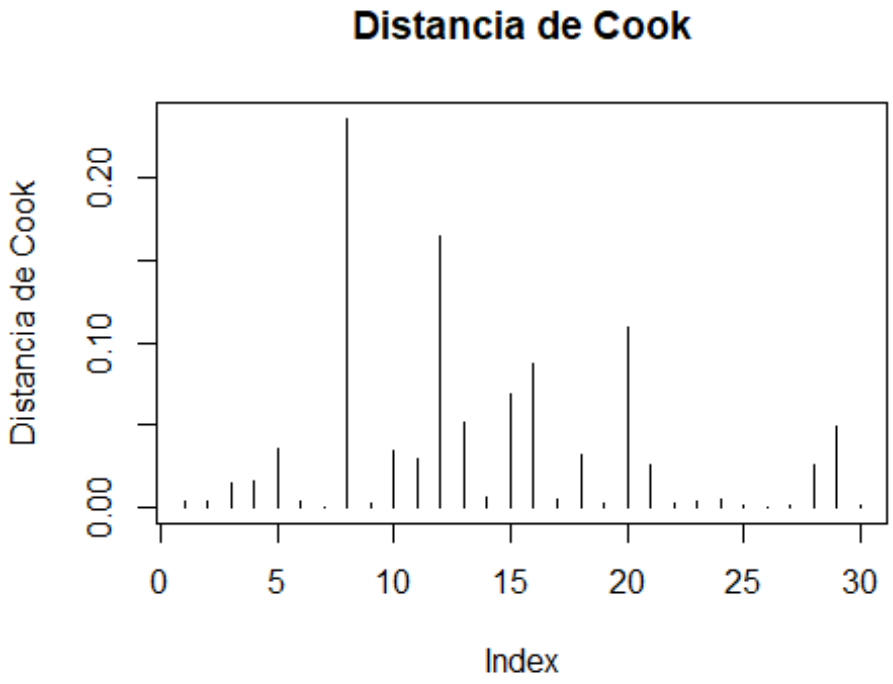
```
##    Fuerza Potencia Temperatura Tiempo Resistencia
residuos_estandarizados
## 19     35       45         200     20        22.7              -
0.1595110
## 20     35      105         200     20        58.7
1.1543546
## 21     35       75         150     20        34.5
0.5460190
## 22     35       75         250     20        44.0              -
0.1876539
```

1. Datos influyentes

1. Distancia de Cook

```
cooksdistance <- cooks.distance(modelo)
#Calcula la distancia de Cook de los n datos

#Gráfico auxiliar:
plot(cooksdistance, type="h", main="Distancia de Cook", ylab="Distancia
```

```
de Cook")
abline(h = 1, col="red") # Límite comúnmente usado
```

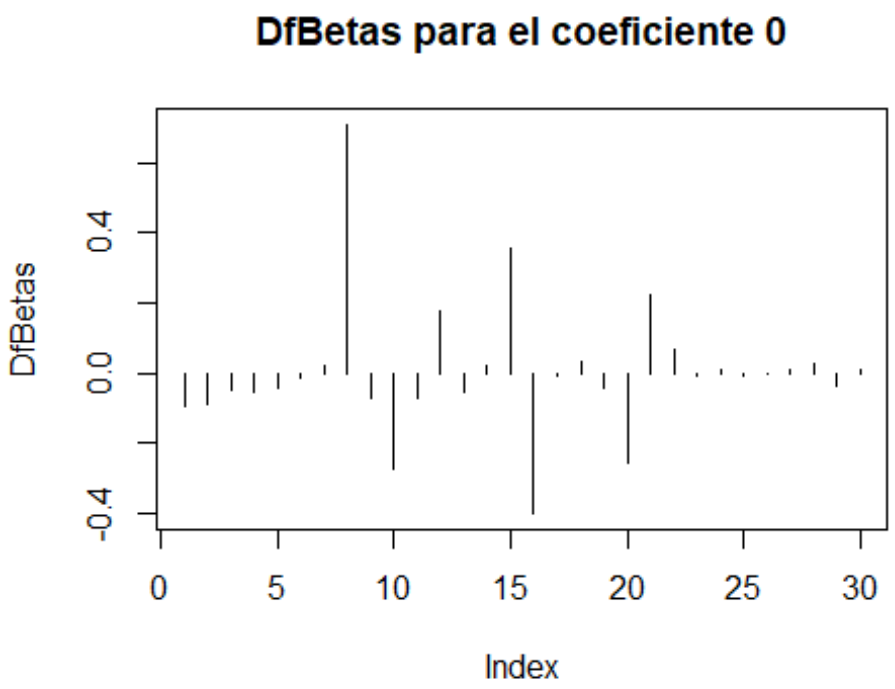



```
#Cuenta e identifica cuántos datos atípicos hay:
puntos_influyentes = which(cooksdistance > 1)

#Muestra las observaciones influyentes
Datos[puntos_influyentes, ]
```

```
## [1] Fuerza                  Potencia                Temperatura
## [4] Tiempo                  Resistencia
residuos_estandarizados
## <0 rows> (or 0-length row.names)
```

2. DfBetas

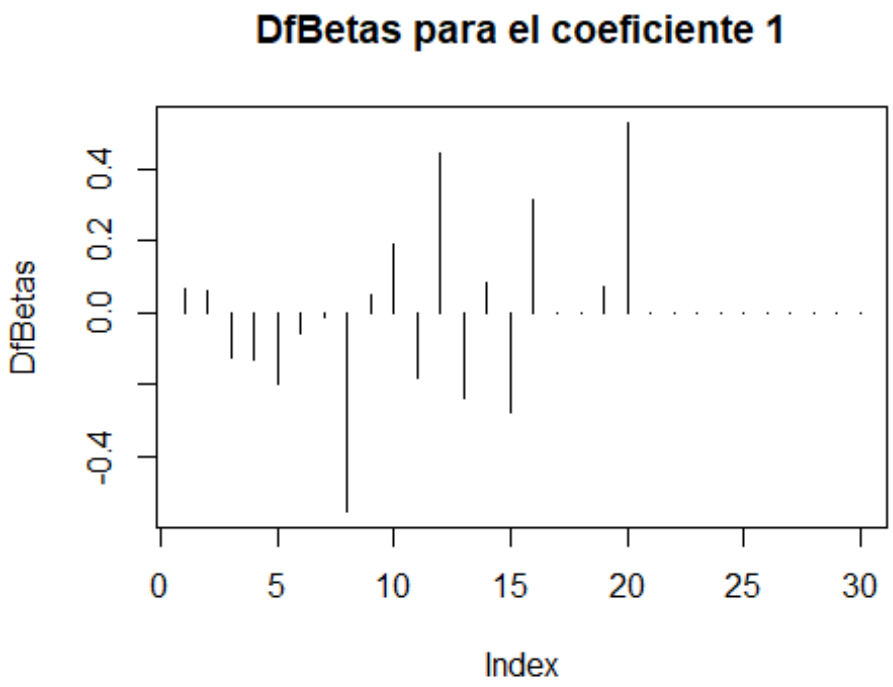
```
dfbetas_values = dfbetas(modelo)
#Calcula la DfBeta de los n datos para cada βj

plot(dfbetas_values[, 1], type="h", main='DfBetas para el coeficiente 0',
ylab="DfBetas")
abline(h = c(-1, 1), col="red") # Límites comunes
```




```r
#Cuenta e identifica cuántos datos atípicos hay:
puntos_influyentes = which(abs(dfbetas_values[, 2]) > 1)

#Muestra las observaciones influyentes
Datos[puntos_influyentes, ]
```

```
## [1] Fuerza                  Potencia                Temperatura
## [4] Tiempo                  Resistencia
residuos_estandarizados
## <0 rows> (or 0-length row.names)
```

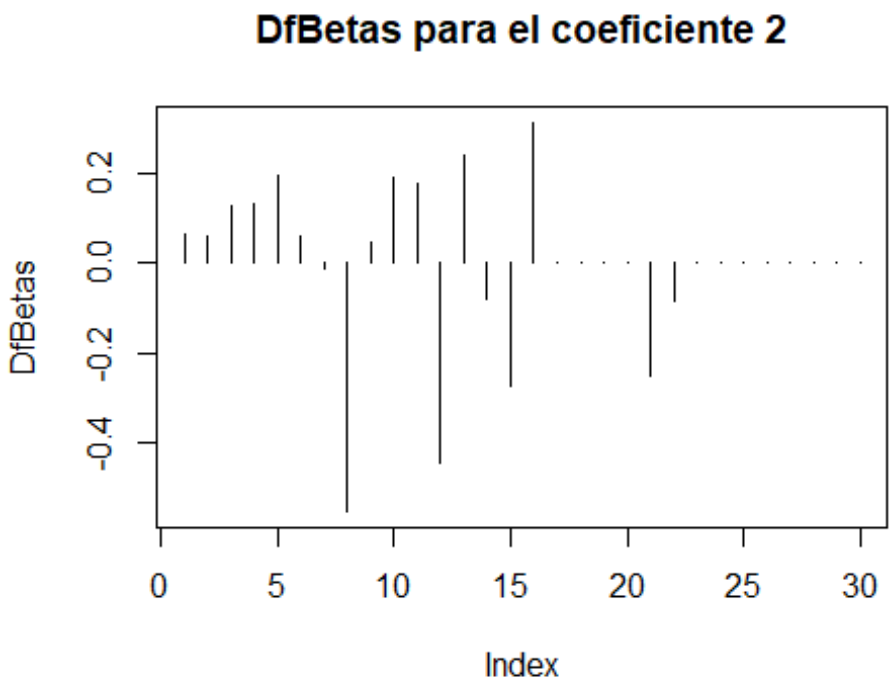
```r
plot(dfbetas_values[, 2], type="h", main='DfBetas para el coeficiente 1',
ylab="DfBetas")
abline(h = c(-1, 1), col="red") # Límites comunes
```




```r
#Cuenta e identifica cuántos datos atípicos hay:
puntos_influyentes = which(abs(dfbetas_values[, 2]) > 1)

#Muestra las observaciones influyentes
Datos[puntos_influyentes, ]
```

```
## [1] Fuerza                  Potencia                Temperatura
## [4] Tiempo                  Resistencia
residuos_estandarizados
## <0 rows> (or 0-length row.names)
```

```r
#Gráfico auxiliar, para la variable 2:

plot(dfbetas_values[, 3], type="h", main='DfBetas para el coeficiente 2',
ylab="DfBetas")
abline(h = c(-1, 1), col="red") # Límites comunes
```




```
#Cuenta e identifica cuántos datos atípicos hay:
puntos_influyentes = which(abs(dfbetas_values[, 2]) > 1)

#Muestra las observaciones influyentes
Datos[puntos_influyentes, ]
```

```
## [1] Fuerza                  Potencia                Temperatura
## [4] Tiempo                  Resistencia
residuos_estandarizados
## <0 rows> (or 0-length row.names)
```

3. Resúmenes de los datos atípicos e influyentes

1. Influence measures

Calculan: • Distancia de leverange (hii) • Distancia de Cook • DfBetas

```
influencia = influence.measures(modelo)
#Calcula las medidas de los n datos

#Resumen de datos influyentes:
summary(influencia)
```

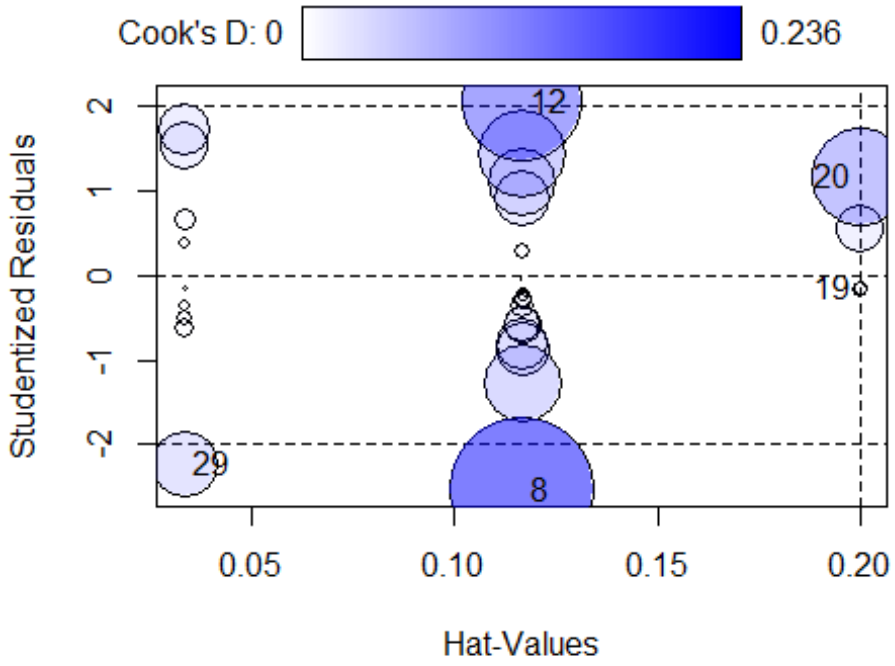
```
## Potentially influential observations of
##   lm(formula = Resistencia ~ Potencia + Temperatura, data = datos) :
##
##    dfb.1_ dfb.Ptnc dfb.Tmpr dffit cov.r   cook.d hat
## 8   0.71  -0.55    -0.55    -0.92  0.65_*  0.24   0.12
```

```
## 19 -0.04   0.07     0.00    -0.08  1.40_*  0.00   0.20
## 21  0.22   0.00    -0.25     0.27  1.35_*  0.03   0.20
## 22  0.07   0.00    -0.09    -0.09  1.39_*  0.00   0.20

# Detecta los datos con posible influencia
```

2. Influence Plot

Calcula: • Distancia de leverange (hii) • Distancia de Cook • Residuos estandarizados

```
library(car)
influencePlot(modelo)
```

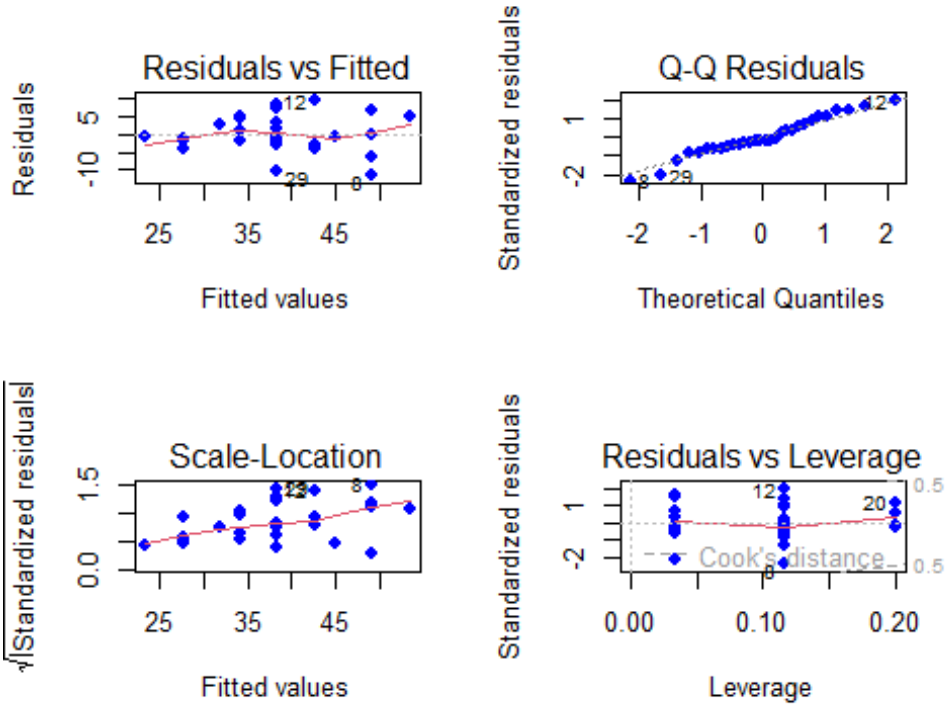



```
##      StudRes        Hat       CookD
## 8  -2.535832 0.11666667 0.235696235
## 12  2.043589 0.11666667 0.164507739
## 19 -0.159511 0.20000000 0.002199712
## 20  1.154355 0.20000000 0.109693544
## 29 -2.216952 0.03333333 0.049338917

# grafica los residuos con estandarización

#extrema, el laverage y la distancia de cook
#Muestra las observaciones influyentes
```

3. Plot del modelo

Gráfica y detecta atípicos o influyentes en los gráficos: • Residuos vs valores ajustados • Qqplot de los residuos • Residuos estandarizados vs valores ajustados • Residuos estandarizados vs Distancia de Leverage y de Cook

```
par(mfrow=c(2, 2))
plot(modelo, col='blue', pch=19)
```



Los datos atípicos e influyentes que tenemos en as distintas pruebas son los siguientes:

Datos atípicos

Estandarización extrema de los residuos

Se buscaron datos atípicos cuyos residuos están a más de 3 dessviaciones estándar pero no encontró ni uno.

Distancia de Leverage

Usando que los datos estuvieran a más de 1.99*media se encontraron 4 datos atípicos siendo estos los datos cuyos índices son 19, 20, 21 y 22

Datos influyentes

Distancia de cook

No se encontraron datos cuya distancia de cook fuera mayor a 1 sin embargo, el dato con el úndice 8 es el que tiene mayor distancia de cook

DfBetas

Con la prueba de DfBetas en cada coeficiente no se encontraron datos influyentes cuyos dfbetas_values fueran mayores a 1 sin embargo, en la gráfica de cada coeficiente el dato con el índice 8 es el que tiene mayor valor de DfBetas

Datos atípicos e influyentes

Influence Measures

Según esta prueba los datos POTENCIALMENTE influyentes son los datos 8, 19, 21, 22, los últimos 2 se debieron por la distancia de leverage y el dato 8 se debió por la distancia de cook y las dfbetas

Influence Plot

Con esta gráfica y prueba los posibles datos influyentes fueron los datos 8, 12, 19, 20, 29. Los datos 8, 12, 29 se debieron a la estandarización extrema de los residuos pero la razón por la que nos salieron previamente fue porque se bajó el umbral de desviaciones estándar a 2, por lo que fue más flexible en ese sentido y los datos 19 y 20 se debieron a la distancia de Leverage

Conclusión: Parece que los datos atípicos e influyenes que más se repiten son justamente los de la prueba de Influence Measures ya que dicta posibles datos influeyentes en nuestro modelo