A3-Regresión Múltiple-Detección datos atípicos

Ricardo Salinas

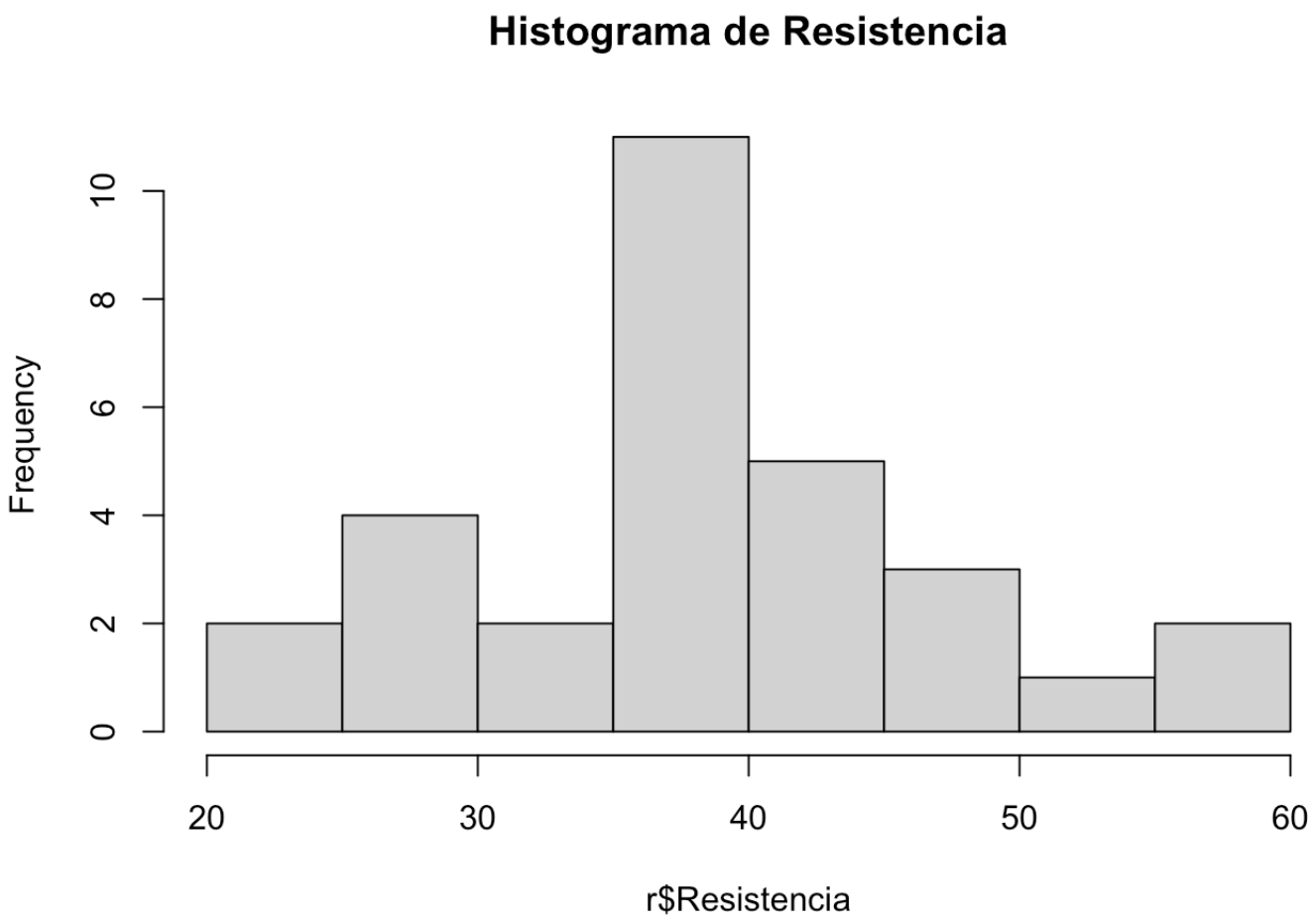
2024-09-24

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

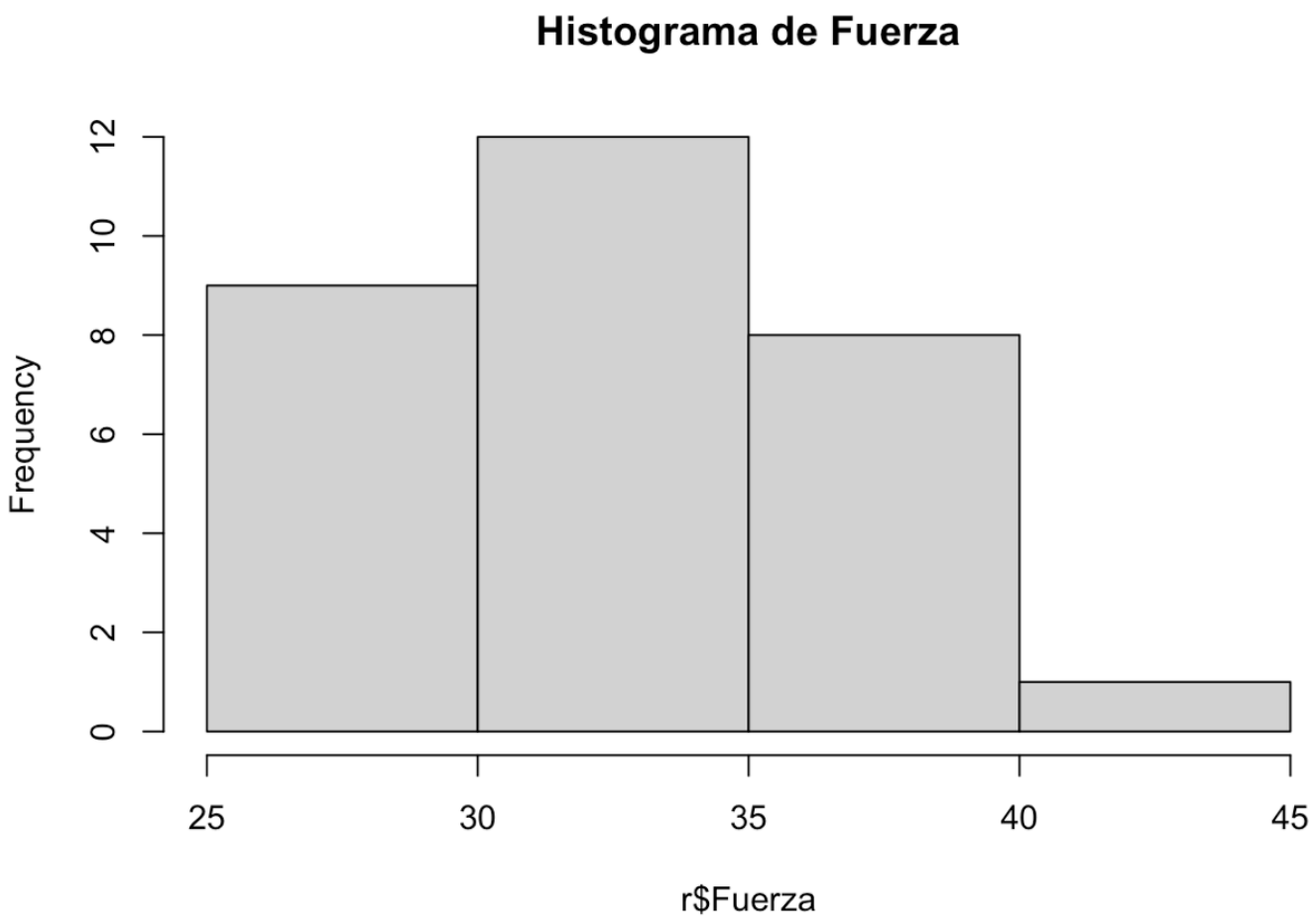
```
r = read.csv("AlCorte.csv")
summary(r)
```

```
##      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
```

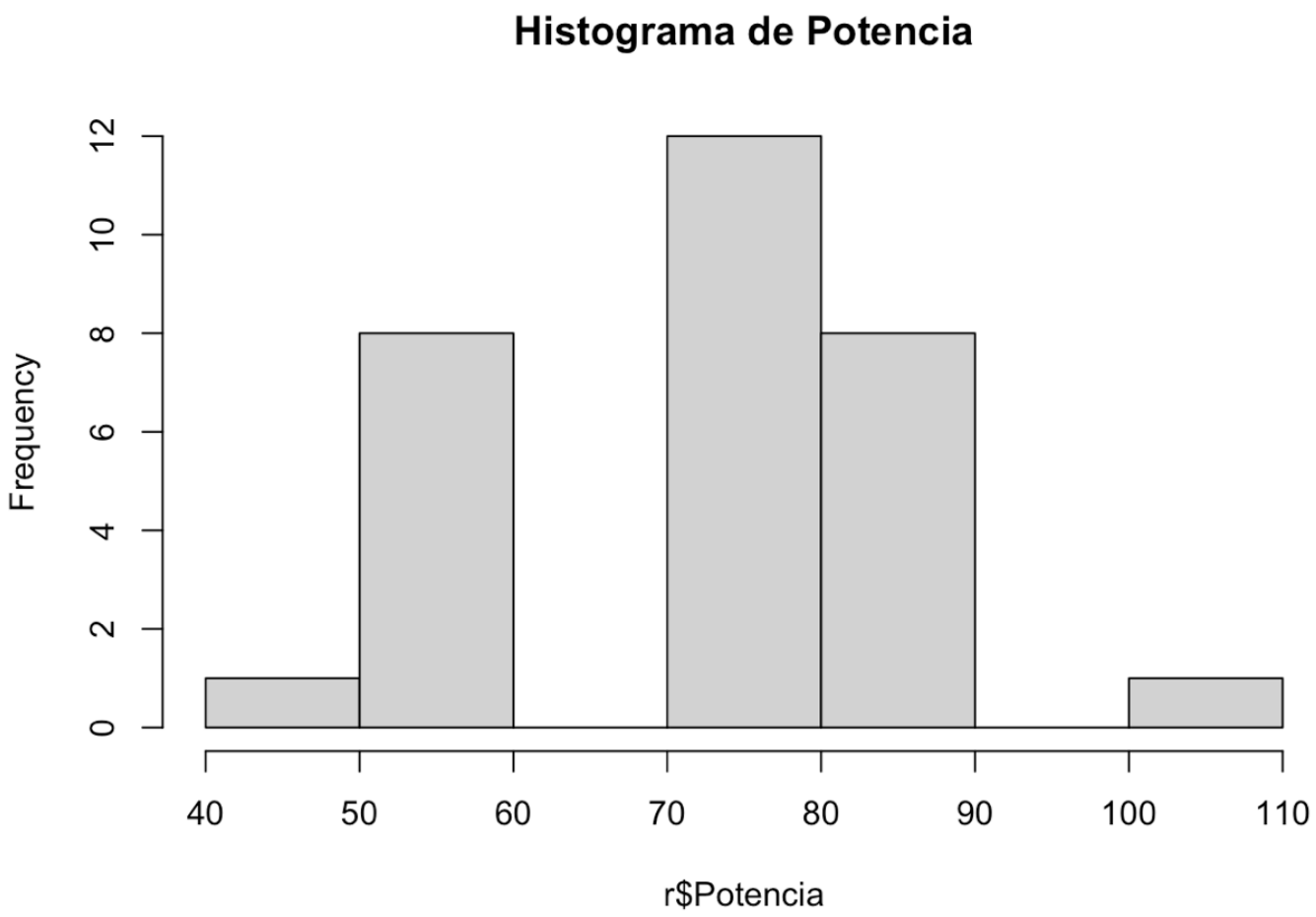
```
hist(r$Resistencia, main = "Histograma de Resistencia")
```



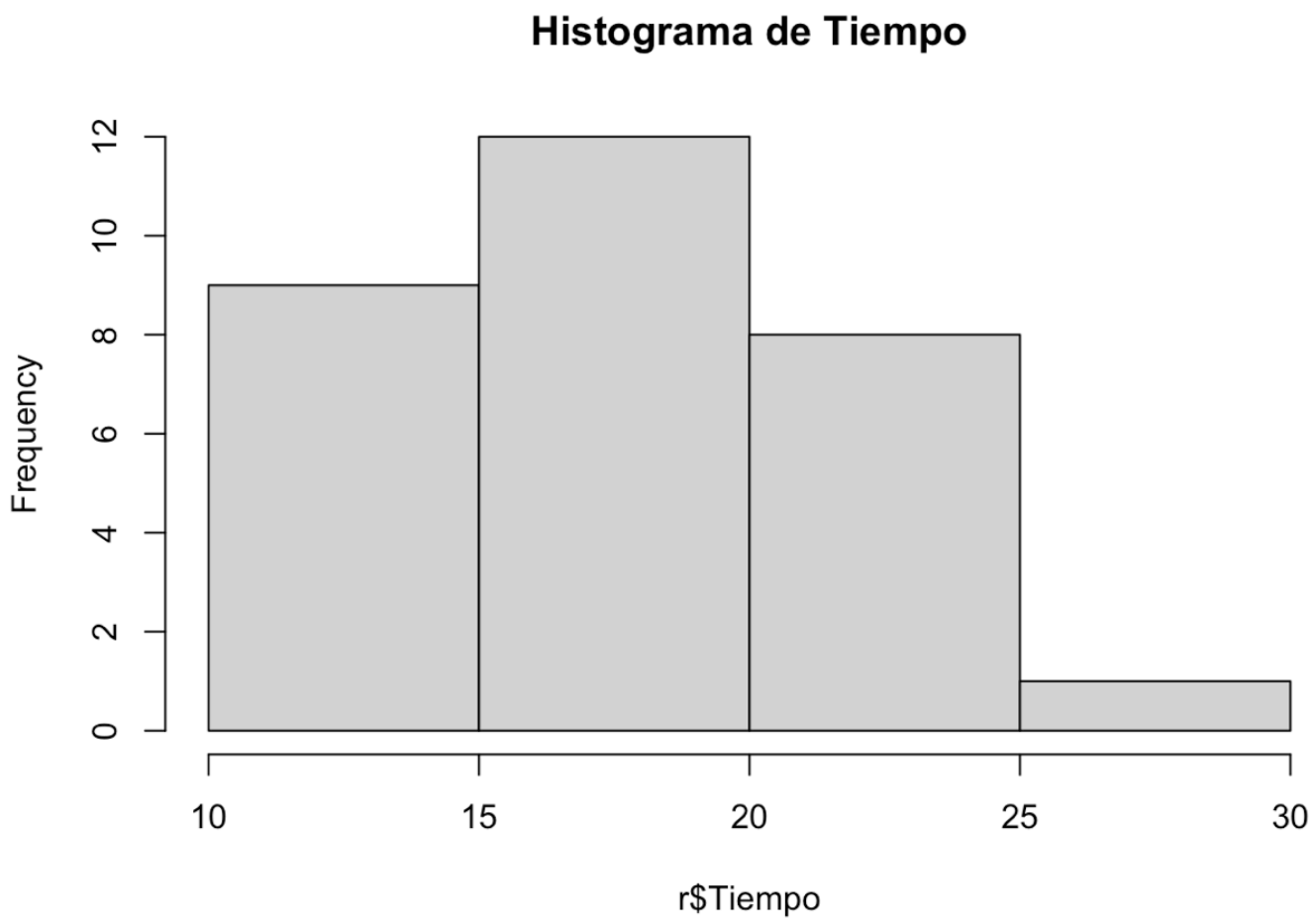

```
hist(r$Fuerza, main = "Histograma de Fuerza")
```



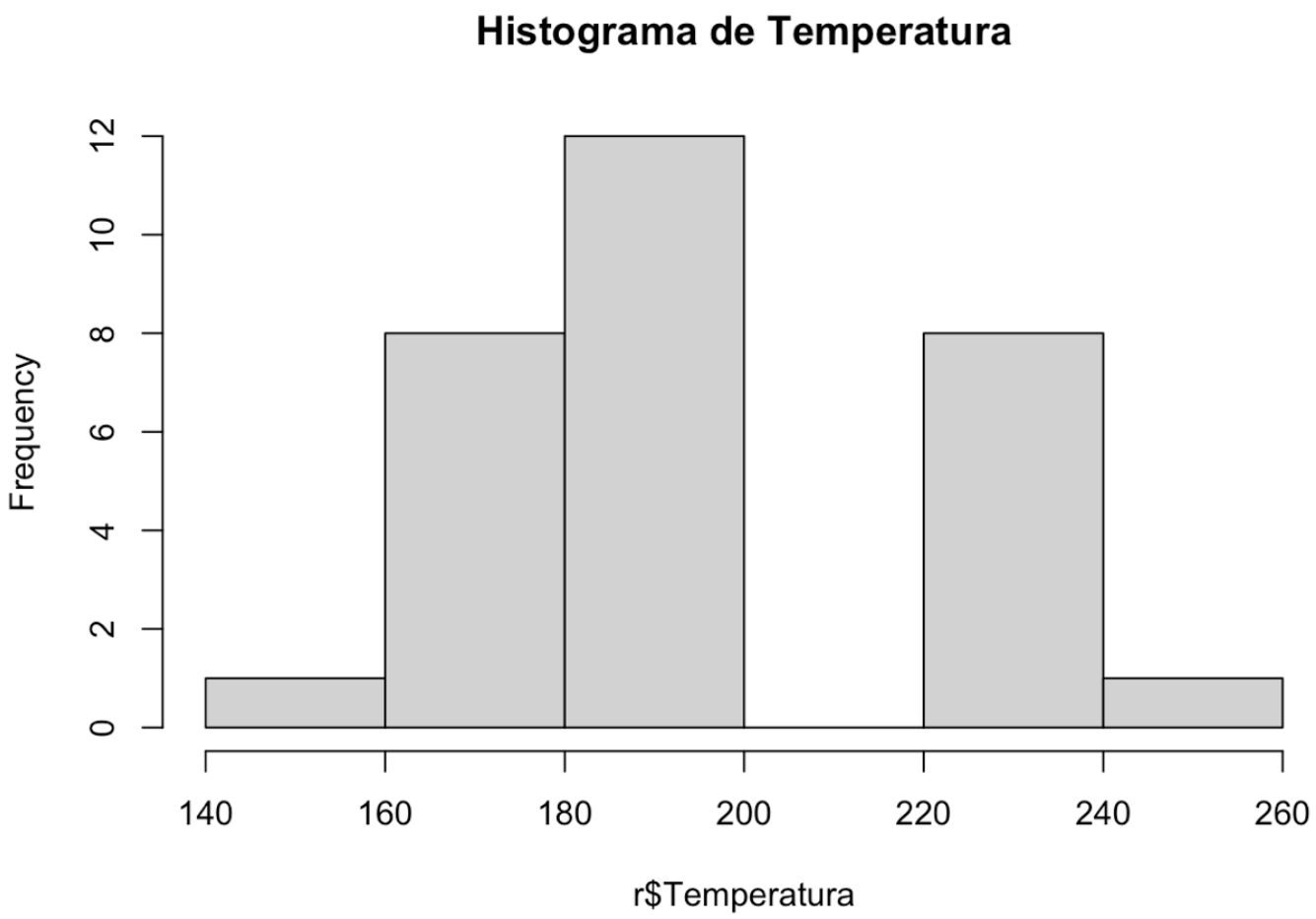

```
hist(r$Potencia, main = "Histograma de Potencia")
```



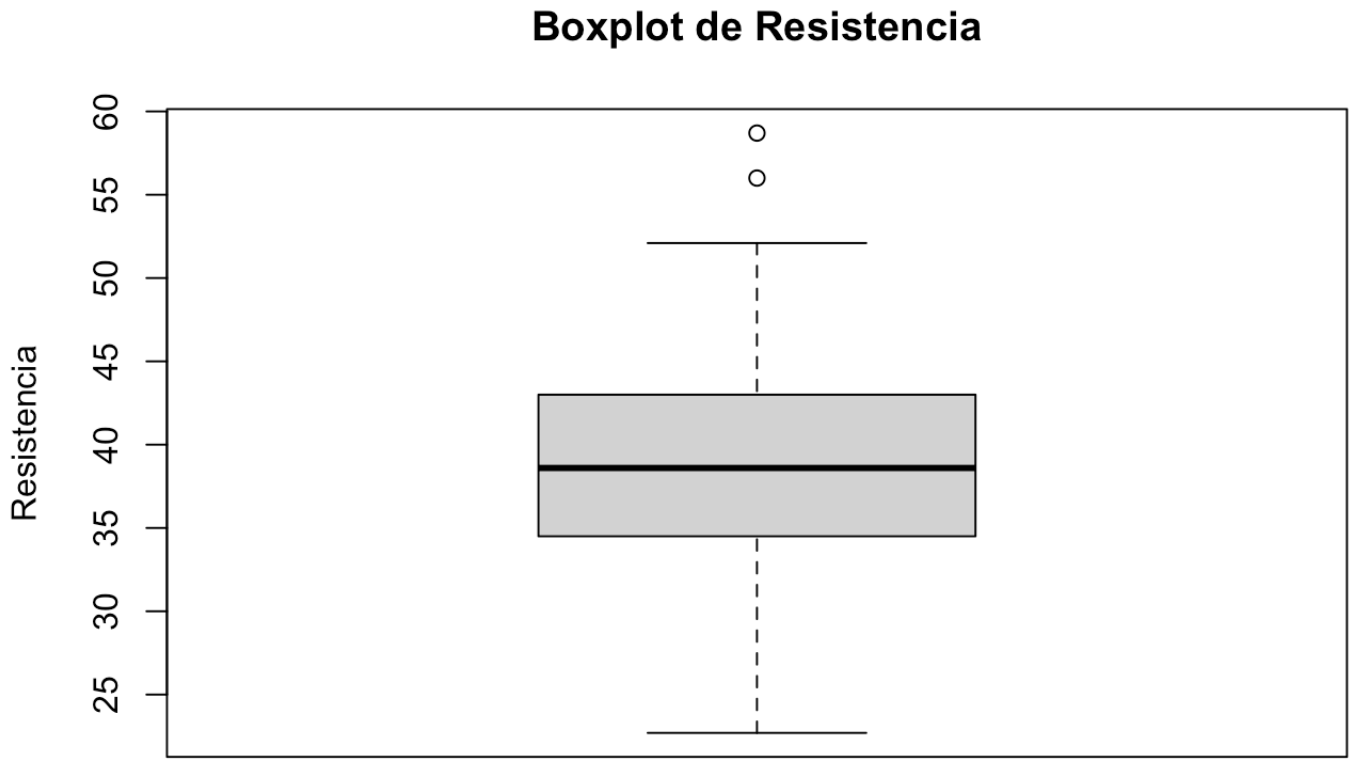

```
hist(r$Tiempo, main = "Histograma de Tiempo")
```



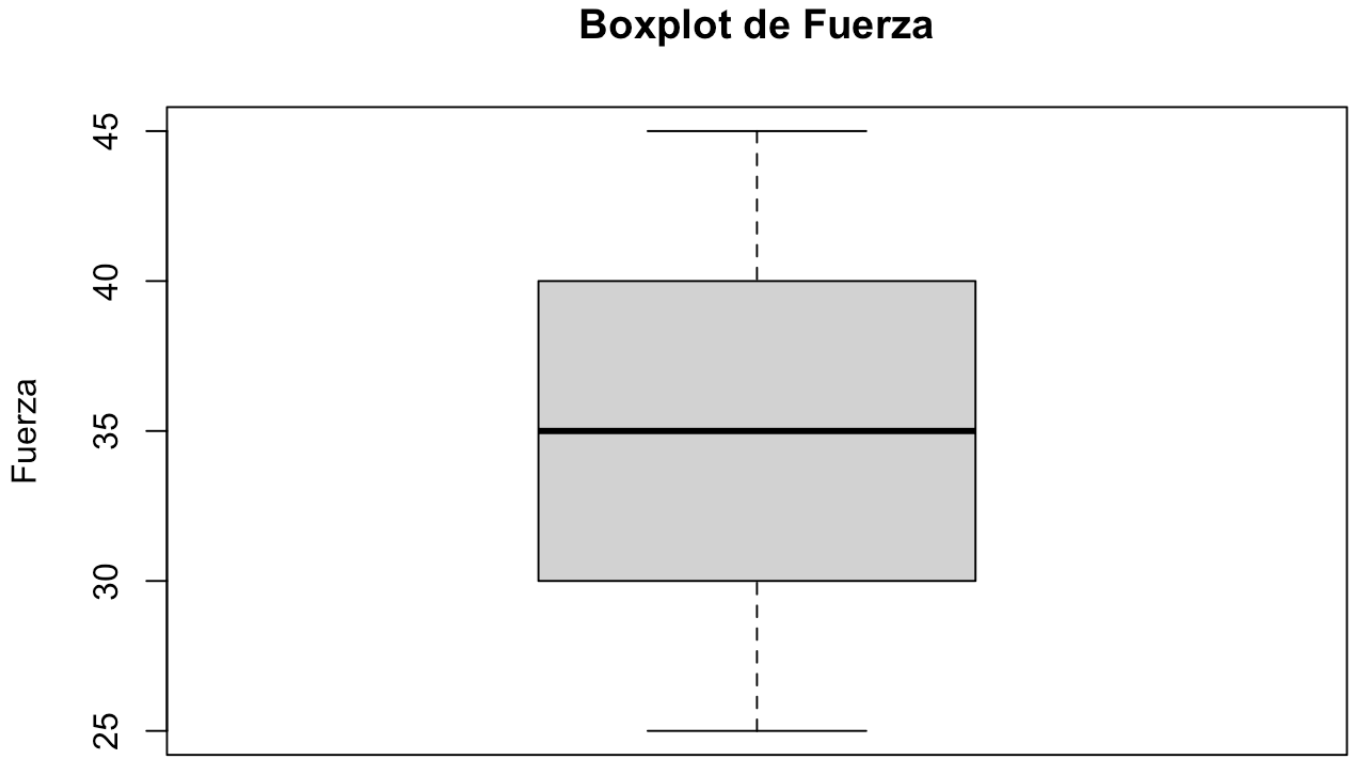

```
hist(r$Temperatura, main = "Histograma de Temperatura")
```



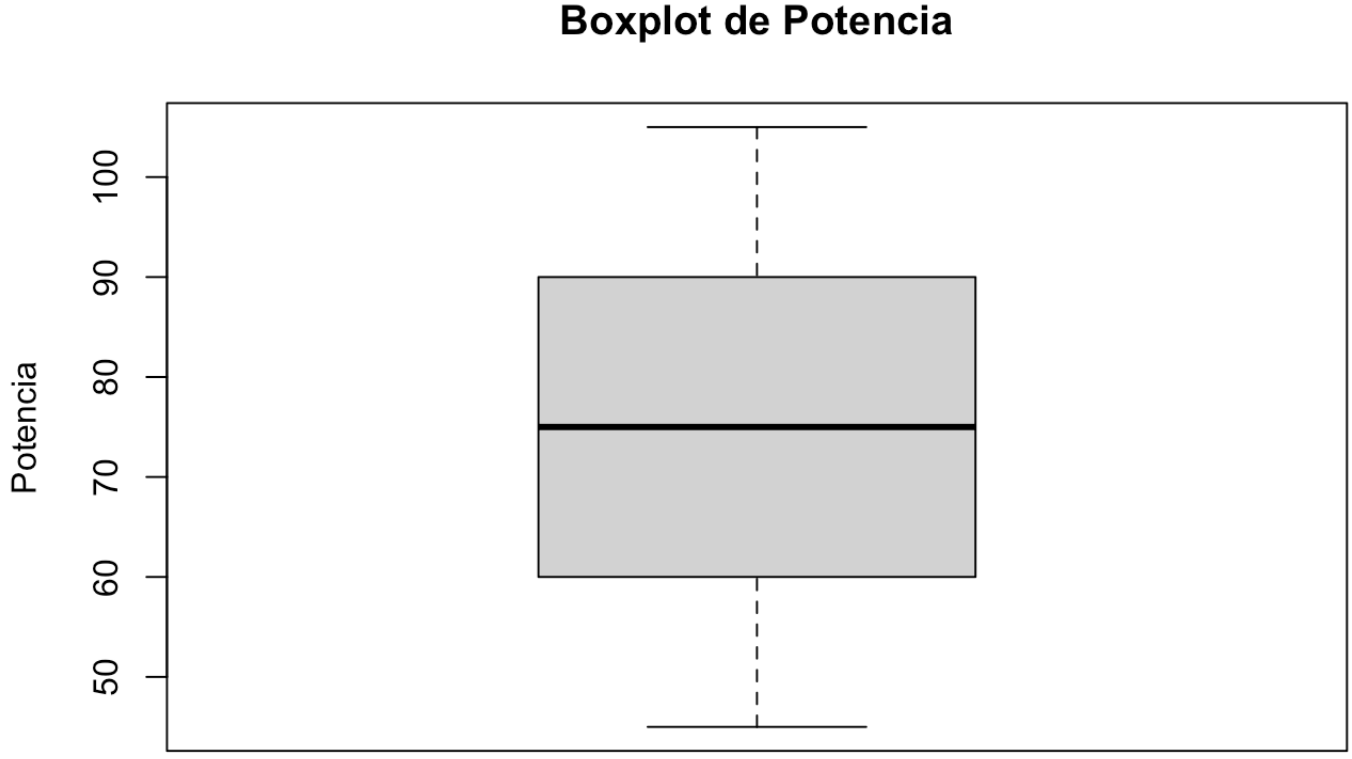

```
boxplot(r$Resistencia, main = "Boxplot de Resistencia", ylab = "Resistencia")
```



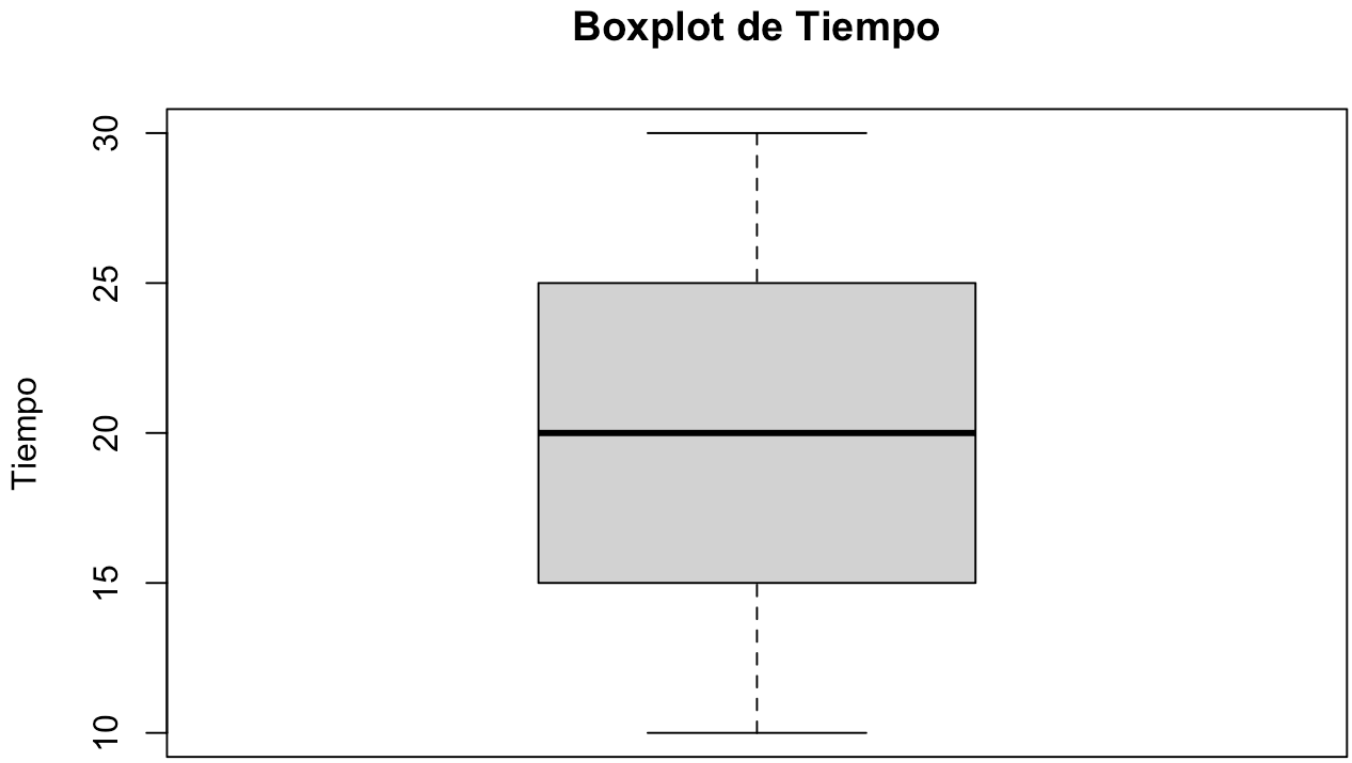

```
boxplot(r$Fuerza, main = "Boxplot de Fuerza", ylab = "Fuerza")
```



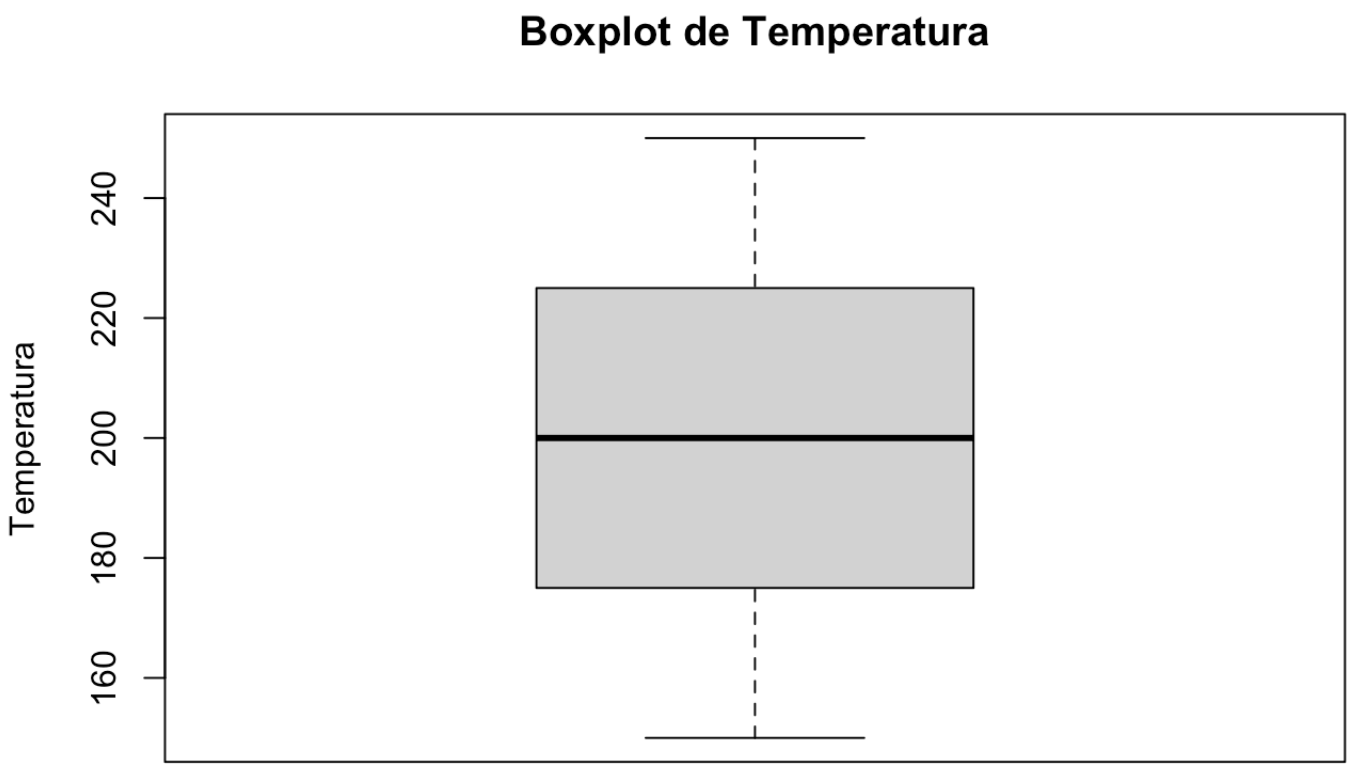

```
boxplot(r$Potencia, main = "Boxplot de Potencia", ylab = "Potencia")
```




```
boxplot(r$Tiempo, main = "Boxplot de Tiempo", ylab = "Tiempo")
```




```
boxplot(r$Temperatura, main = "Boxplot de Temperatura", ylab = "Temperatura")
```



2.Encuentra el mejor modelo de regresión que explique la variable Resistencia

```
modelo = lm(Resistencia~., data = r)

#Significancia del modelo:

#Economía de las variables
pasos = step(modelo, 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   1341.01 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
```

```
modelo_nulo = lm(Resistencia~1, data=r)

pasos2 = step(modelo_nulo, scope = list(lower = modelo_nulo, upper = modelo), directi
on = "forward", data = r)
```

```
## 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
```

```
n = length(r$Resistencia)
pasos3 = step(modelo, direction="both", k = log(n))
```

```
## Start:  AIC=109.97
## Resistencia ~ Fuerza + Potencia + Temperatura + Tiempo
##
##               Df Sum of Sq     RSS    AIC
## - Fuerza       1     26.88  692.00 107.76
## - Tiempo       1     40.04  705.16 108.32
## <none>                      665.12 109.97
## - Temperatura  1    252.20  917.32 116.21
## - Potencia     1   1341.01 2006.13 139.69
##
## Step:  AIC=107.76
## Resistencia ~ Potencia + Temperatura + Tiempo
##
##               Df Sum of Sq     RSS    AIC
## - Tiempo       1     40.04  732.04 106.04
## <none>                      692.00 107.76
## + Fuerza       1     26.88  665.12 109.97
## - Temperatura  1    252.20  944.20 113.68
## - Potencia     1   1341.01 2033.02 136.69
##
## Step:  AIC=106.04
## Resistencia ~ Potencia + Temperatura
##
##               Df Sum of Sq     RSS    AIC
## <none>                      732.04 106.04
## + Tiempo       1     40.04  692.00 107.76
## + Fuerza       1     26.88  705.16 108.32
## - Temperatura  1    252.20  984.24 111.52
## - Potencia     1   1341.01 2073.06 133.87
```

```
k = log(n)

print(pasos)
```

```
##
## Call:
## lm(formula = Resistencia ~ Potencia + Temperatura, data = r)
##
## Coefficients:
## (Intercept)     Potencia  Temperatura
##    -24.9017       0.4983       0.1297
```

```
print(pasos2)
```

```
##
## Call:
## lm(formula = Resistencia ~ Potencia + Temperatura, data = r)
##
## Coefficients:
## (Intercept)     Potencia  Temperatura
##    -24.9017       0.4983       0.1297
```

```
print(pasos3)
```

```
##
## Call:
## lm(formula = Resistencia ~ Potencia + Temperatura, data = r)
##
## Coefficients:
## (Intercept)     Potencia  Temperatura
##    -24.9017       0.4983       0.1297
```

```
#Significación global (Prueba para el modelo)
modelo = lm(Resistencia ~., data = r)
summary(modelo)
```

```
##
## Call:
## lm(formula = Resistencia ~ ., data = r)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.0900  -1.7608  -0.3067   2.4392   7.5933
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -37.47667   13.09964  -2.861  0.00841 **
## Fuerza        0.21167    0.21057   1.005  0.32444
## Potencia      0.49833    0.07019   7.100 1.93e-07 ***
## Temperatura   0.12967    0.04211   3.079  0.00499 **
## Tiempo        0.25833    0.21057   1.227  0.23132
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.158 on 25 degrees of freedom
## Multiple R-squared:  0.714,  Adjusted R-squared:  0.6682
## F-statistic:  15.6 on 4 and 25 DF,  p-value: 1.592e-06
```

```
#Significación individual (Prueba para cada βi)
confint(modelo)
```

```
##                    2.5 %      97.5 %
## (Intercept) -64.45588404 -10.4974493
## Fuerza       -0.22201780   0.6453511
## Potencia      0.35377185   0.6428948
## Temperatura   0.04292977   0.2164036
## Tiempo       -0.17535113   0.6920178
```

```
#Variación explicada por el modelo
summary(modelo)
```

```
##
## Call:
## lm(formula = Resistencia ~ ., data = r)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.0900  -1.7608  -0.3067   2.4392   7.5933
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -37.47667   13.09964  -2.861  0.00841 **
## Fuerza        0.21167    0.21057   1.005  0.32444
## Potencia      0.49833    0.07019   7.100 1.93e-07 ***
## Temperatura   0.12967    0.04211   3.079  0.00499 **
## Tiempo        0.25833    0.21057   1.227  0.23132
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.158 on 25 degrees of freedom
## Multiple R-squared:  0.714,  Adjusted R-squared:  0.6682
## F-statistic:  15.6 on 4 and 25 DF,  p-value: 1.592e-06
```

```
print("Se tiene un R cuadrado de : 0.714")
```

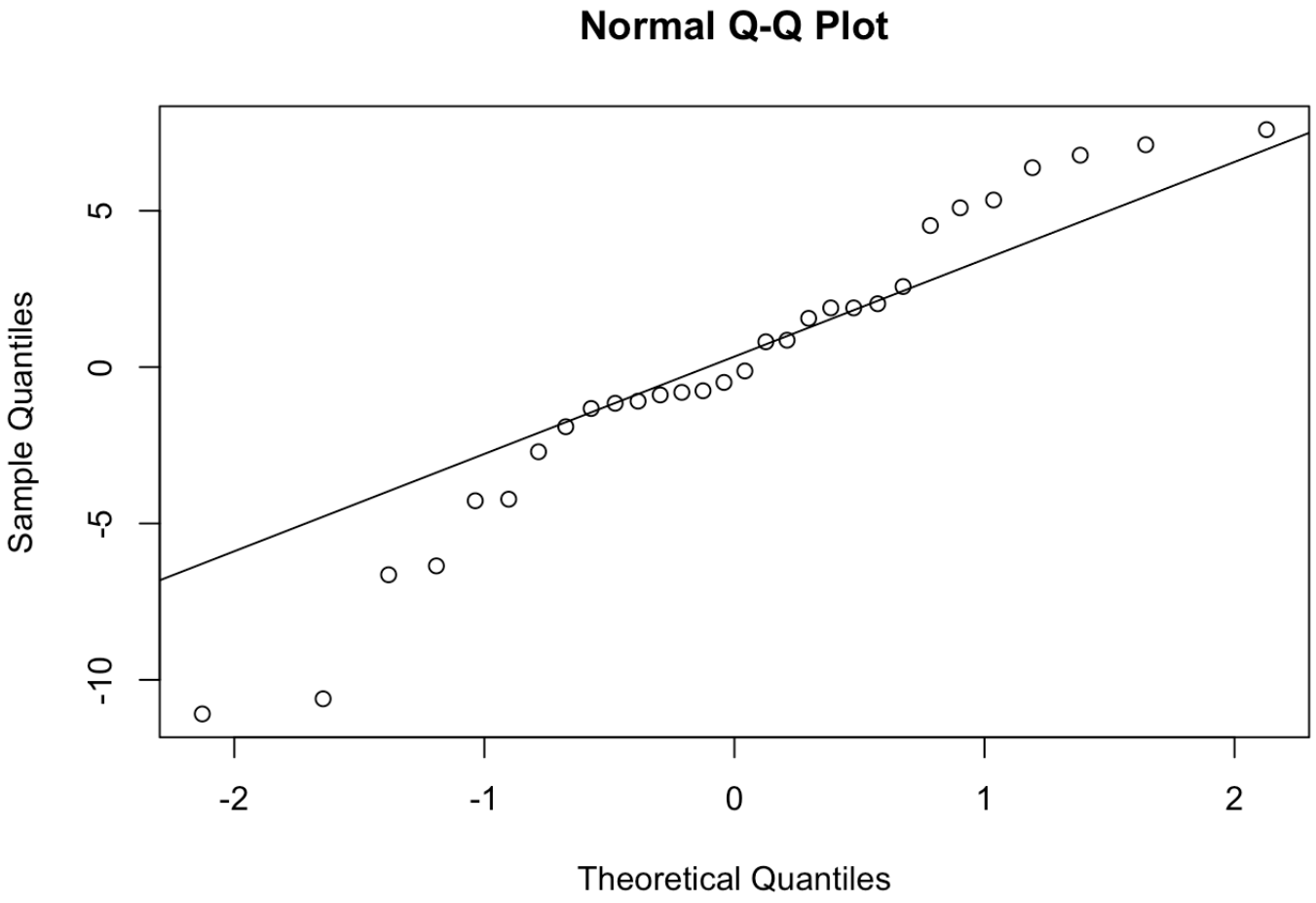
```
## [1] "Se tiene un R cuadrado de : 0.714"
```

3. Analiza la validez del modelo encontrado

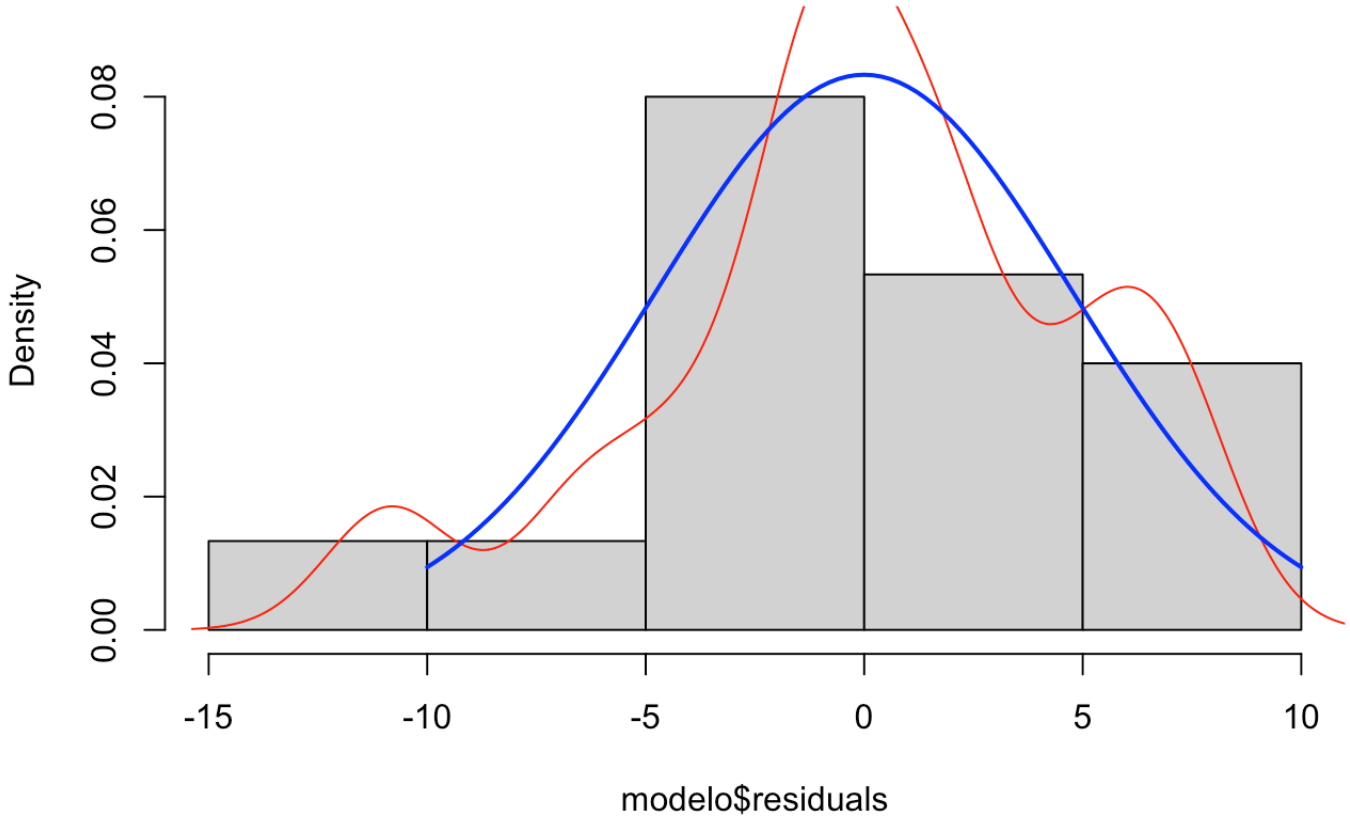
```
#Análisis de residuos (homocedasticidad, independencia, etc)
library(nortest)
ad.test(modelo$residuals)
```

```
##
##  Anderson-Darling normality test
##
## data:  modelo$residuals
## A = 0.40259, p-value = 0.3367
```

```
qqnorm(modelo$residuals)
qqline(modelo$residuals)
```




```
hist(modelo$residuals,freq=FALSE, ylim=c(0,0.09))
lines(density(modelo$residuals),col="red")
curve(dnorm(x,mean=mean(modelo$residuals),sd=sd(modelo$residuals)), from=-10, to=10,
add=TRUE, col="blue",lwd=2)
```




```
#No multicolinealidad de Xi
library(car)
```

```
## Loading required package: carData
```

```
vif(modelo)
```

```
##      Fuerza    Potencia Temperatura      Tiempo
##           1           1           1           1
```

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

```
library(dplyr)
```

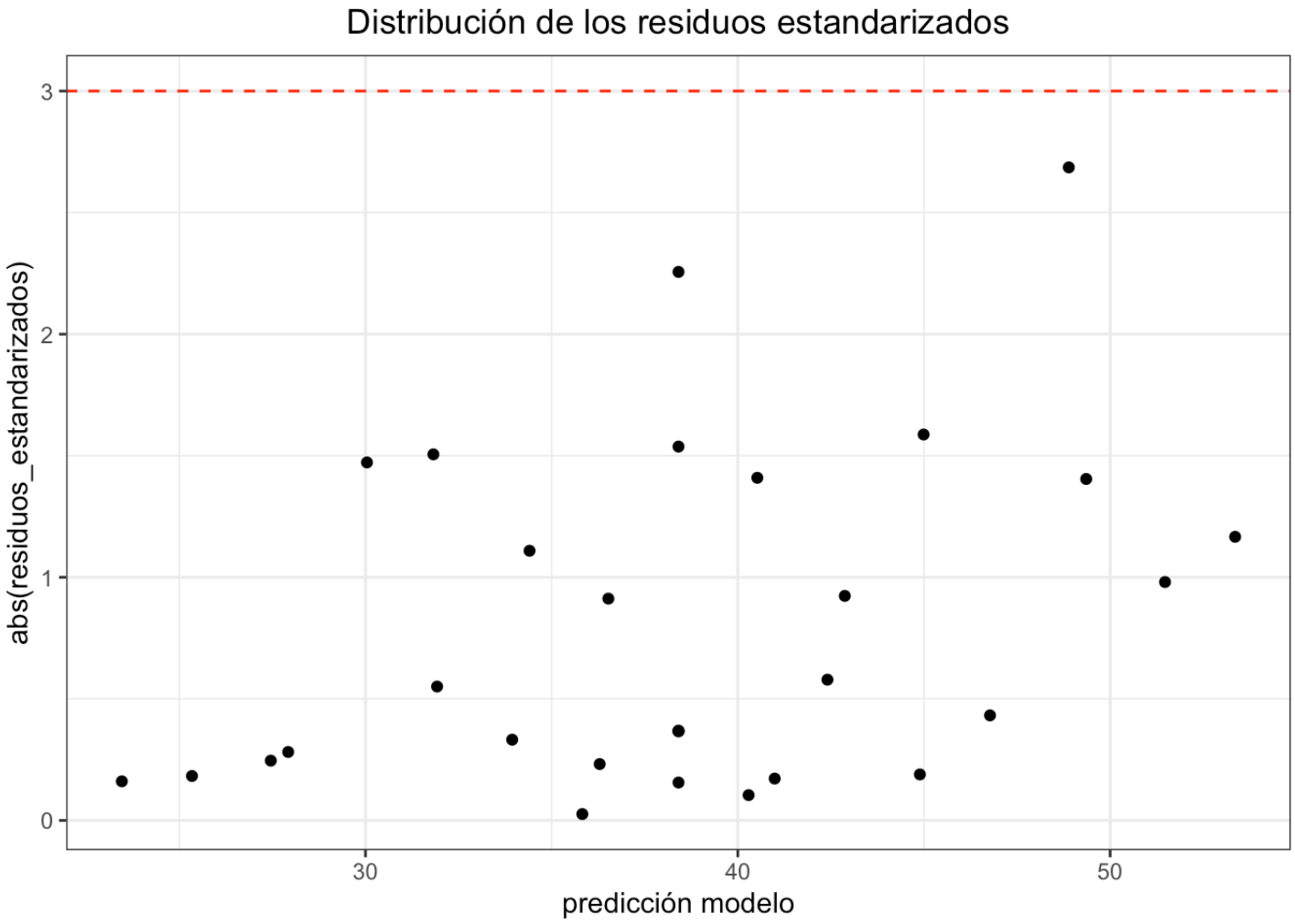
```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:car':
##
##     recode
```

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

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

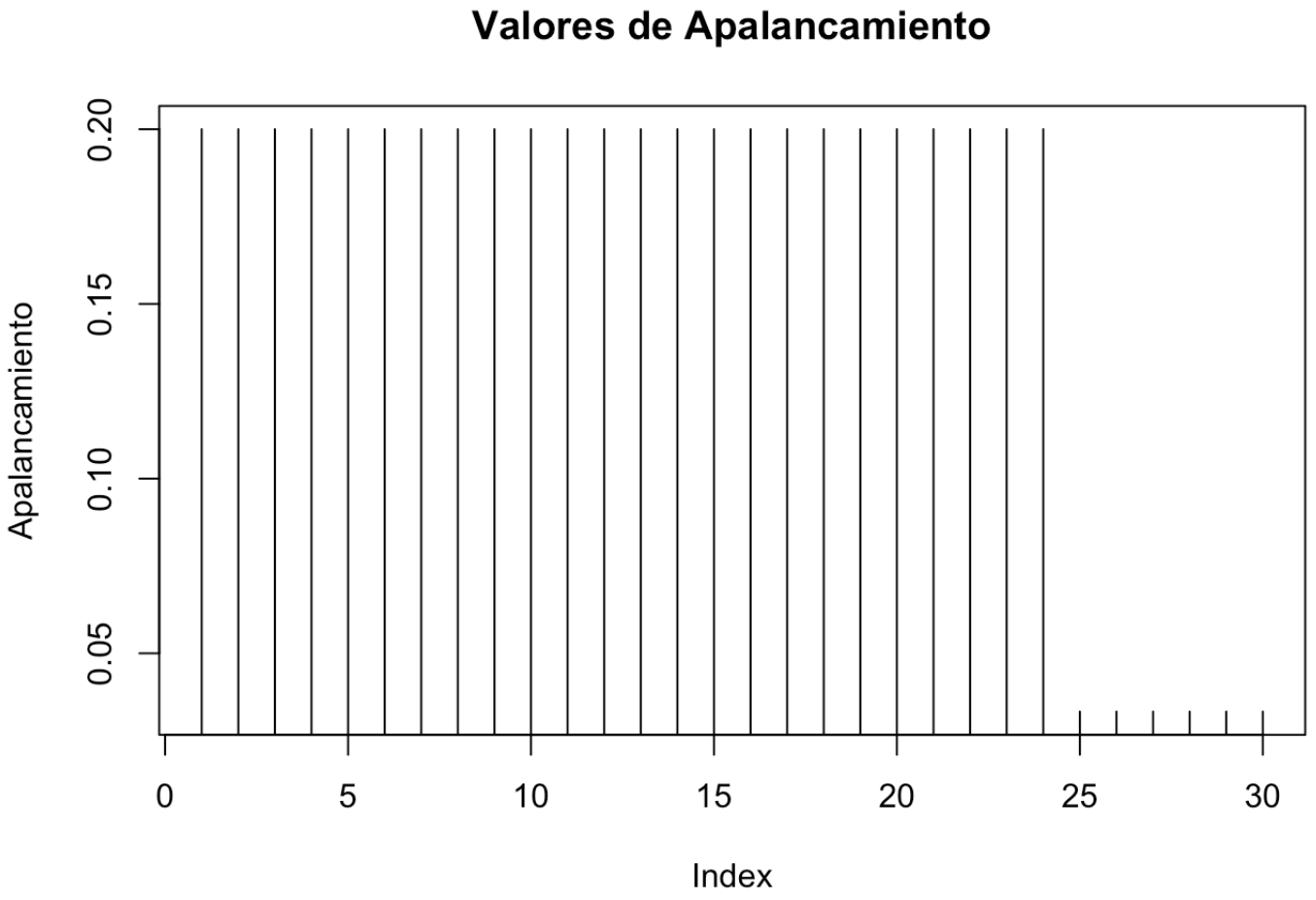
```
r$residuos_estandarizados = rstudent(modelo)

library(ggplot2)

ggplot(data = r, 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))
```




```
atipicos = which(abs(r$residuos_estandarizados)>3)
r[atipicos, ]
```

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

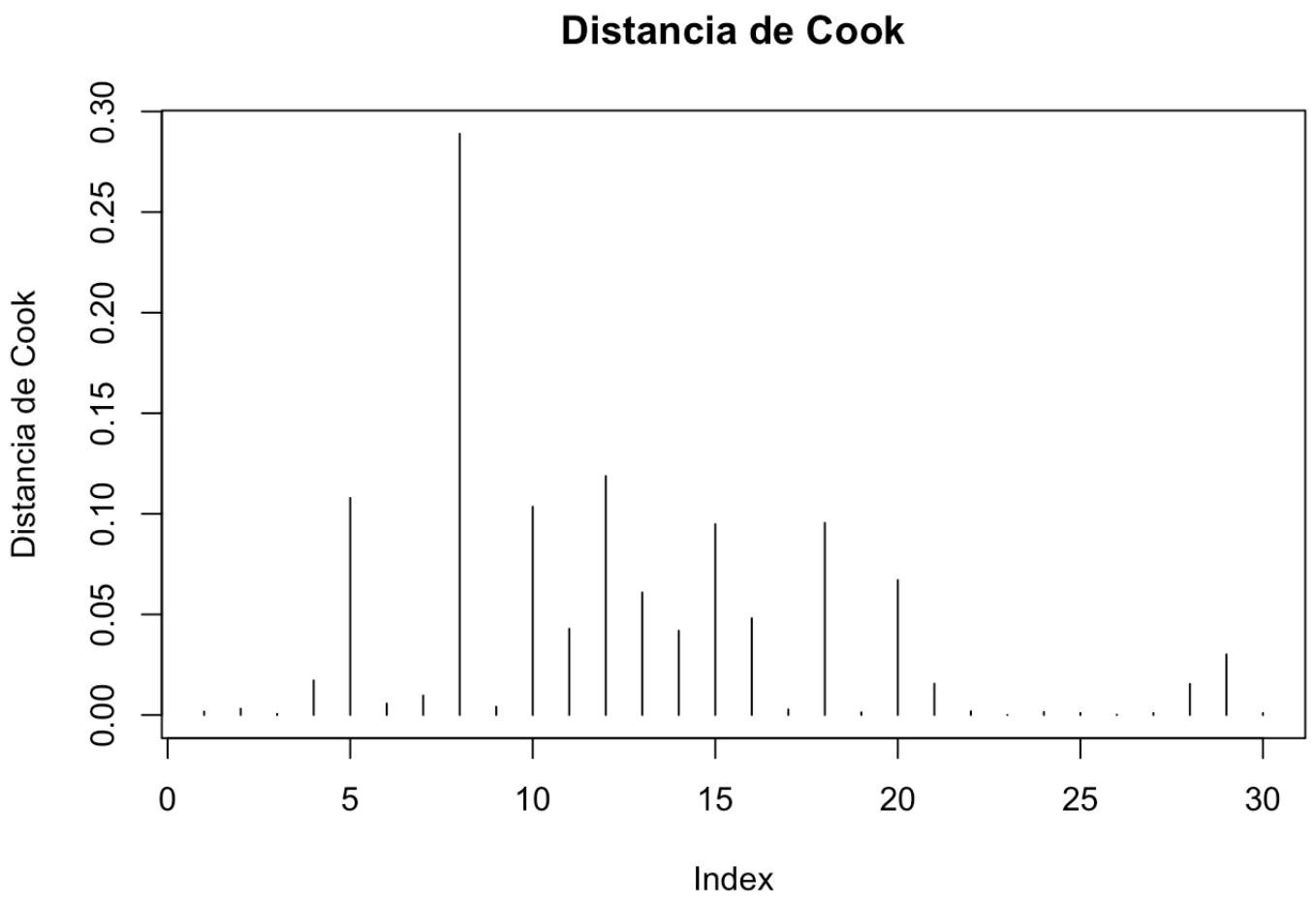
```
leverage = hatvalues(modelo)
plot(leverage, type="h", main="Valores de Apalancamiento", ylab="Apalancamiento")
abline(h = 2*mean(leverage), col="red") # Límite comúnmente usado
```




```
high_leverage_points = which(leverage > 2*mean(leverage))
r[high_leverage_points, ]
```

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

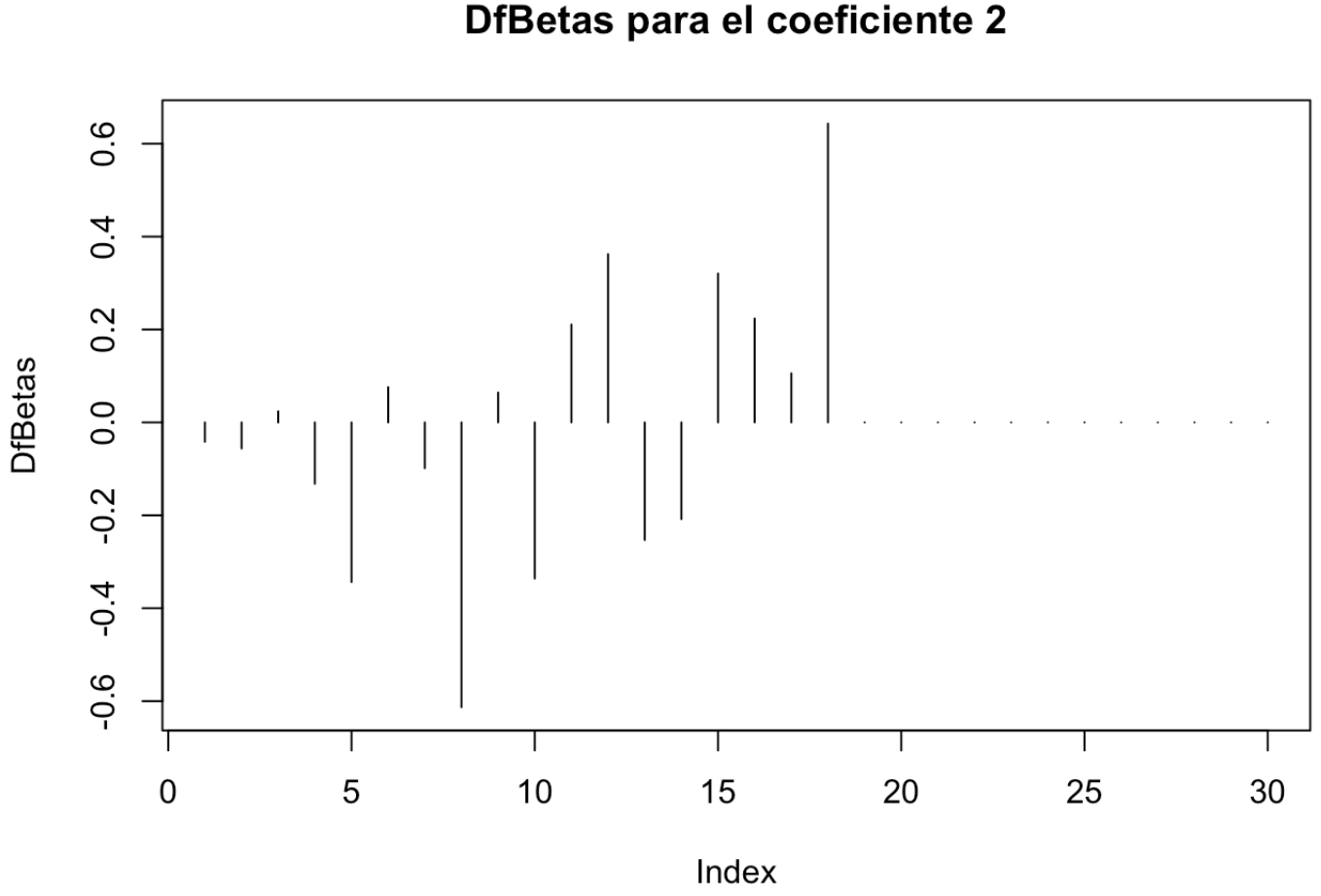
```
cooksd = cooks.distance(modelo)
plot(cooksd, type="h", main="Distancia de Cook", ylab="Distancia de Cook")
abline(h = 1, col="red") # Límite comúnmente usado
```




```
puntos_influyentes = which(cooksd > 1)
r[puntos_influyentes, ]
```

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

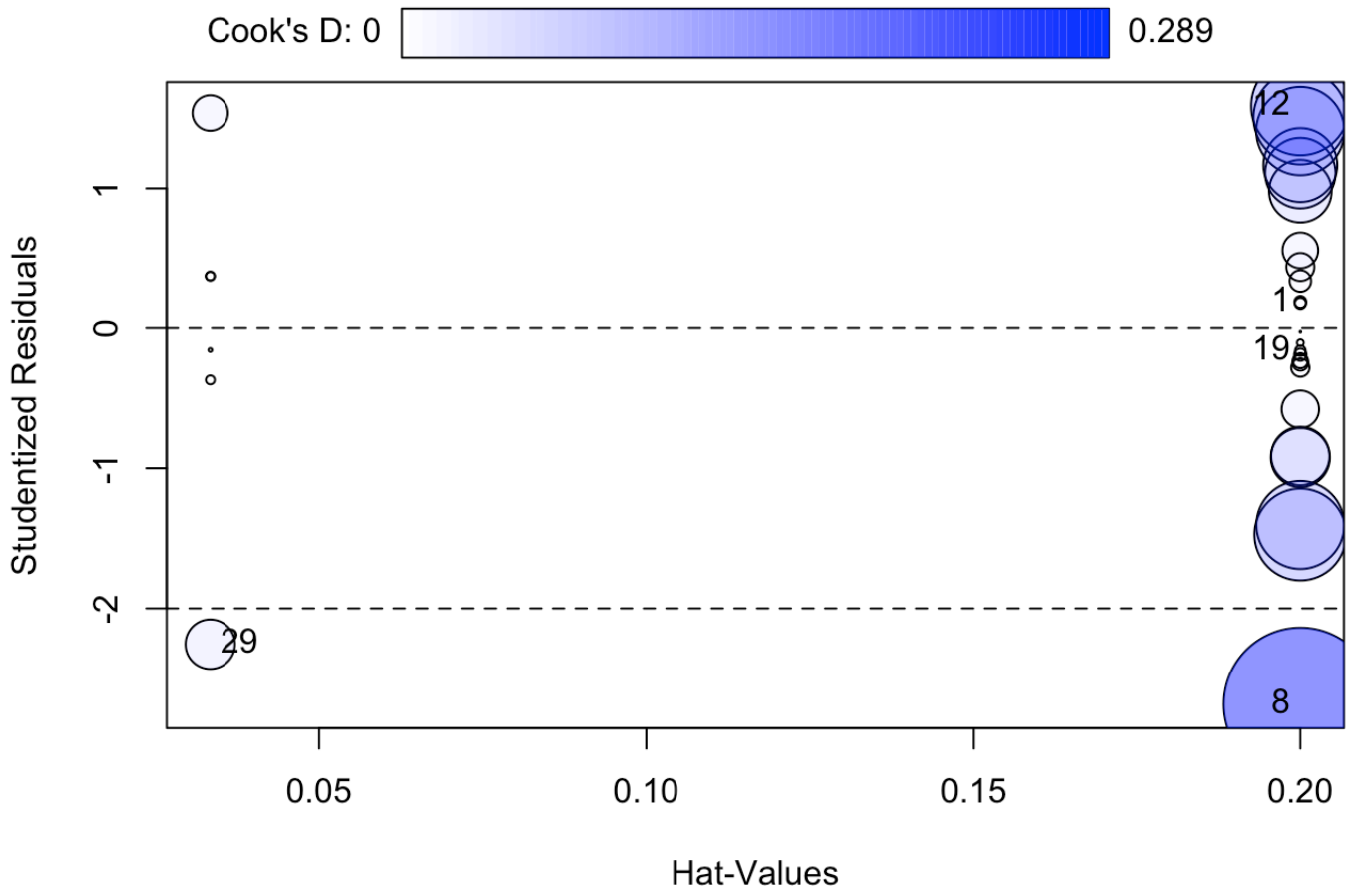
```
dfbetas_values = dfbetas(modelo)
plot(dfbetas_values[, 2], type="h", main="DfBetas para el coeficiente 2",
ylab="DfBetas")
abline(h = c(-1, 1), col="red") # Límites comunes
```




```
puntos_influyentes = which(abs(dfbetas_values[, 2]) > 1)


influencia = influence.measures(modelo)
summary(influencia)
```

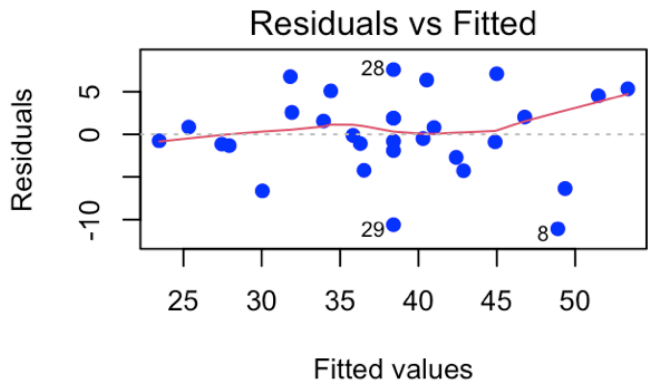
```
## Potentially influential observations of
##   lm(formula = Resistencia ~ ., data = r) :
##
##   dfb.1_ dfb.Furz dfb.Ptnc dfb.Tmpr dfb.Timp dffit   cov.r cook.d hat
## 8  0.75  -0.61    -0.61    -0.61     0.61    -1.34_*  0.41  0.29   0.20
```

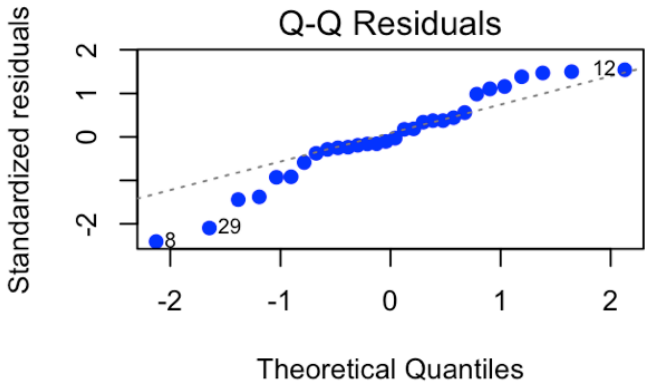
```
library(car)
influencePlot(modelo)
```

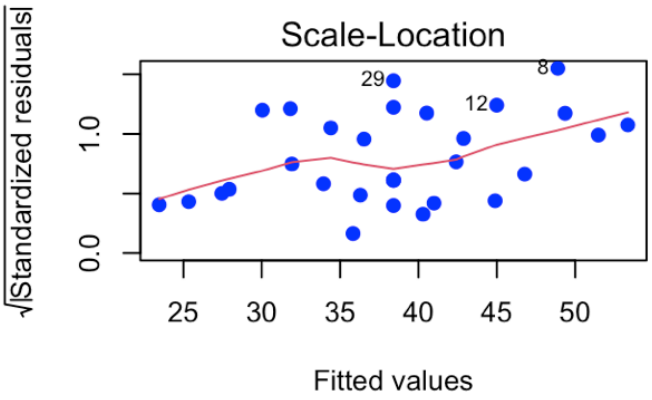


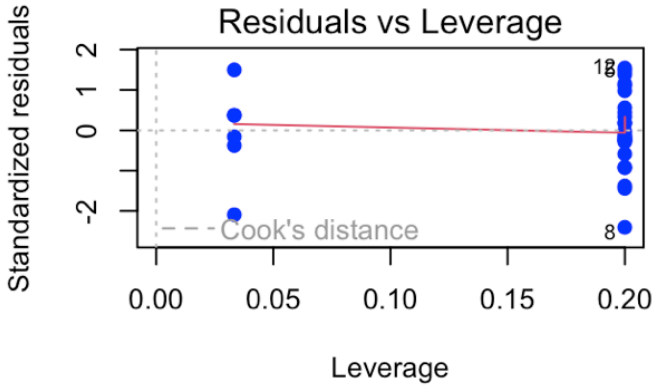

```
##       StudRes        Hat       CookD
## 1   0.1827728 0.20000000 0.001737472
## 8  -2.6860791 0.20000000 0.288924241
## 12  1.5872954 0.20000000 0.118757239
## 19 -0.1607864 0.20000000 0.001345024
## 29 -2.2561277 0.03333333 0.030168517
```

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


Residuals vs Fitted
Residuals
Fitted values
28
29
8
25
30
35
40
45
50
-10
0
5


Q-Q Residuals
Standardized residuals
Theoretical Quantiles
12
29
8
-2
-1
0
1
2


Scale-Location
√|Standardized residuals|
Fitted values
29
12
8
25
30
35
40
45
50
0.0
1.0


Residuals vs Leverage
Standardized residuals
Leverage
Cook's distance
8
0.00
0.05
0.10
0.15
0.20
-2
0
1
2