Solución Ejercicios 1 y 2, Tema 3. Diagnóstico y validación en Regresión lineal múltiple.

Máster en Ciencia de Datos. Módulo: Análisis exploratorio de datos

Ana Navarro Quiles

Curso 2022/2023

```
load('datosTema3.Rdata')
```

Ejercicio 1

Los sistemas de entrega de productos son de vital importancia para las empresas. En particular, les suele interesar predecir el *tiempo* necesario para realizar los pedidos. Supongamos que la persona responsable de analizar los datos a cargo de una empresa sólo tiene acceso rápido a información sobre la distancia y el número de cajas que ha de distribuirse en cada pedido. En el fichero **cervezas** tenemos unos datos que representan las tres variables nombradas.

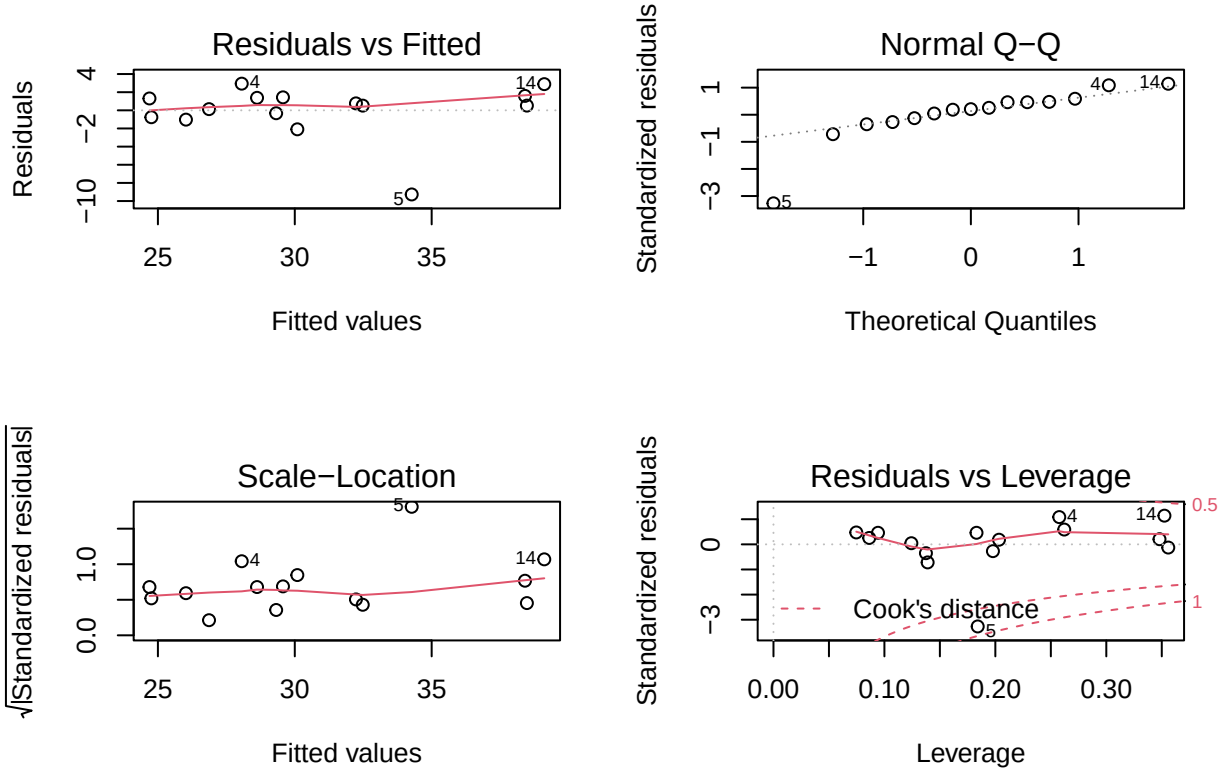
a) ¿Cuál es el porcentaje de varianza explicada por tu modelo?¿Qué variables son relevantes?

```
mod1 <- lm(tiempo ~ ., data=cervezas, na.action=na.exclude)
summary(mod1)
```

```
##
## Call:
## lm(formula = tiempo ~ ., data = cervezas, na.action = na.exclude)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.2716 -0.5405  0.5212  1.4051  2.9381
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.3112     5.8573   0.395  0.70007
## cajas         0.8772     0.1530   5.732 9.43e-05 ***
## distancia     0.4559     0.1468   3.107  0.00908 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.141 on 12 degrees of freedom
## Multiple R-squared:  0.7368, Adjusted R-squared:  0.6929
## F-statistic:  16.8 on 2 and 12 DF,  p-value: 0.0003325
```

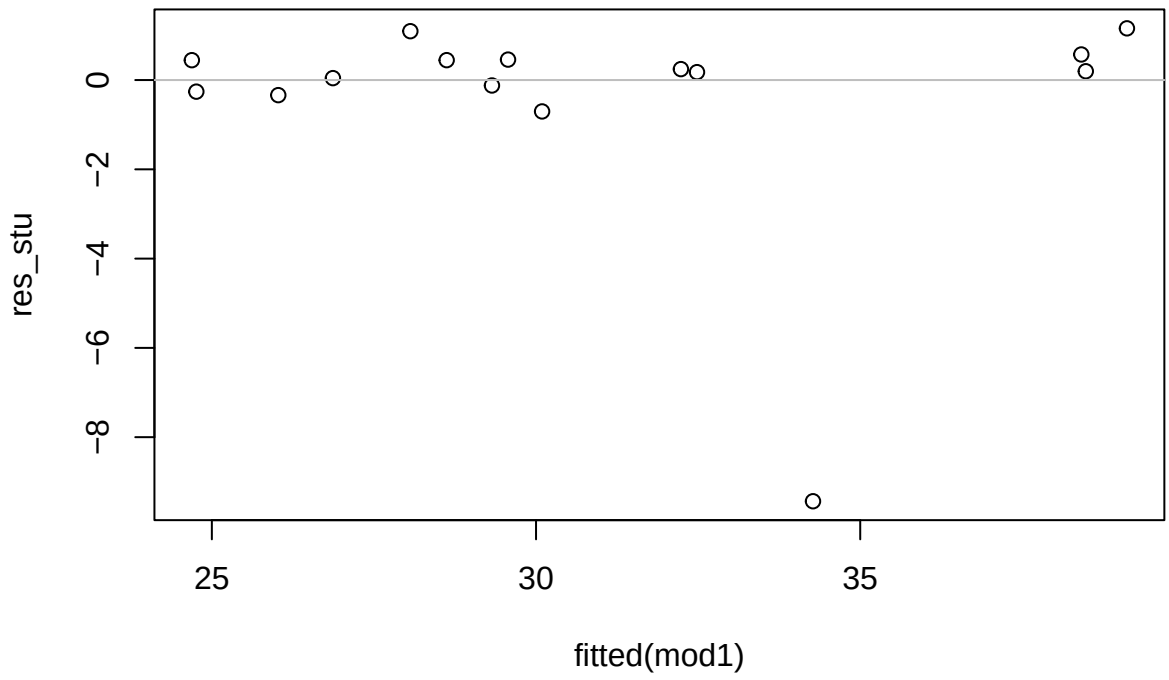
b) Diagnostica el modelo ¿Qué observas?¿ Puedes mejorar tu modelo solucionando el o los problemas observados?

```
par(mfrow=c(2,2))
plot(mod1)
```




```
res_stu<-rstudent(mod1)
par(mfrow=c(1,1))
plot(fitted(mod1),res_stu)
 abline(h=0, col="gray")


 library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.3
```

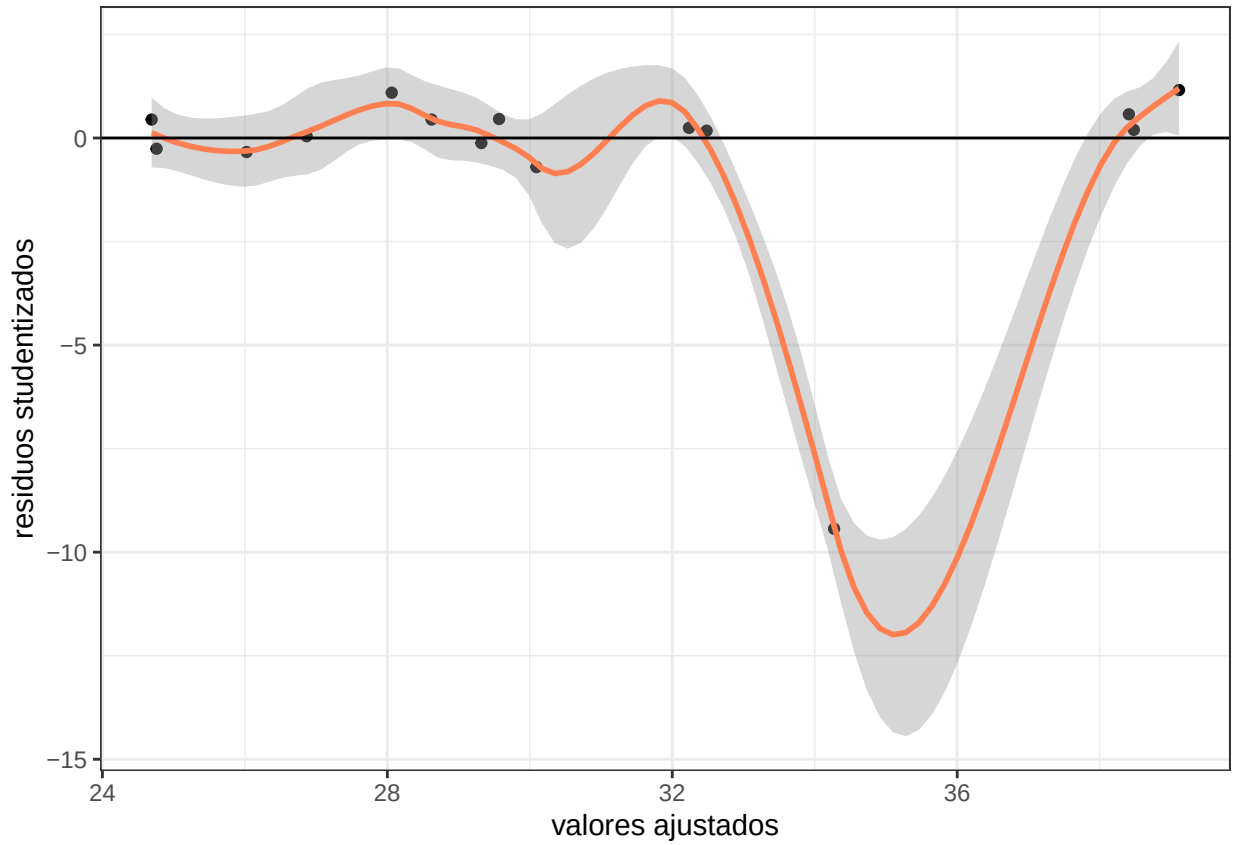



```
  library(gridExtra)
```

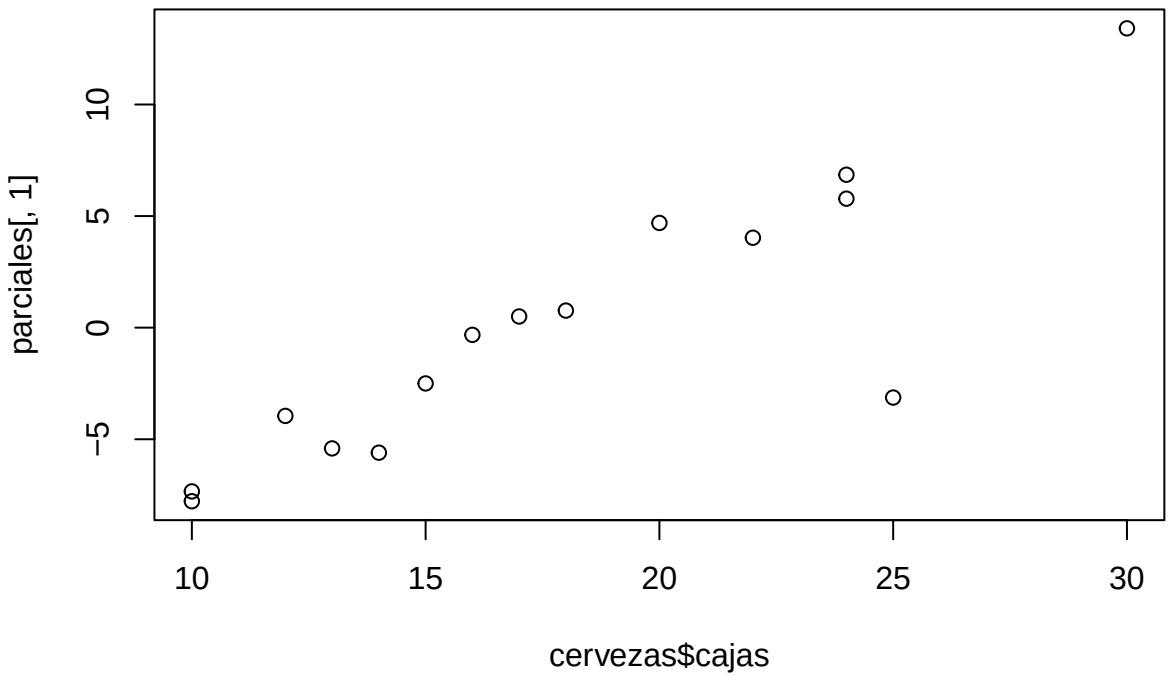
```
## Warning: package 'gridExtra' was built under R version 4.1.3
```

```
  ggplot(data = cervezas, aes(x =fitted(mod1), y = res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  labs(y = "residuos studentizados",
       x = "valores ajustados") +
  theme_bw()
```

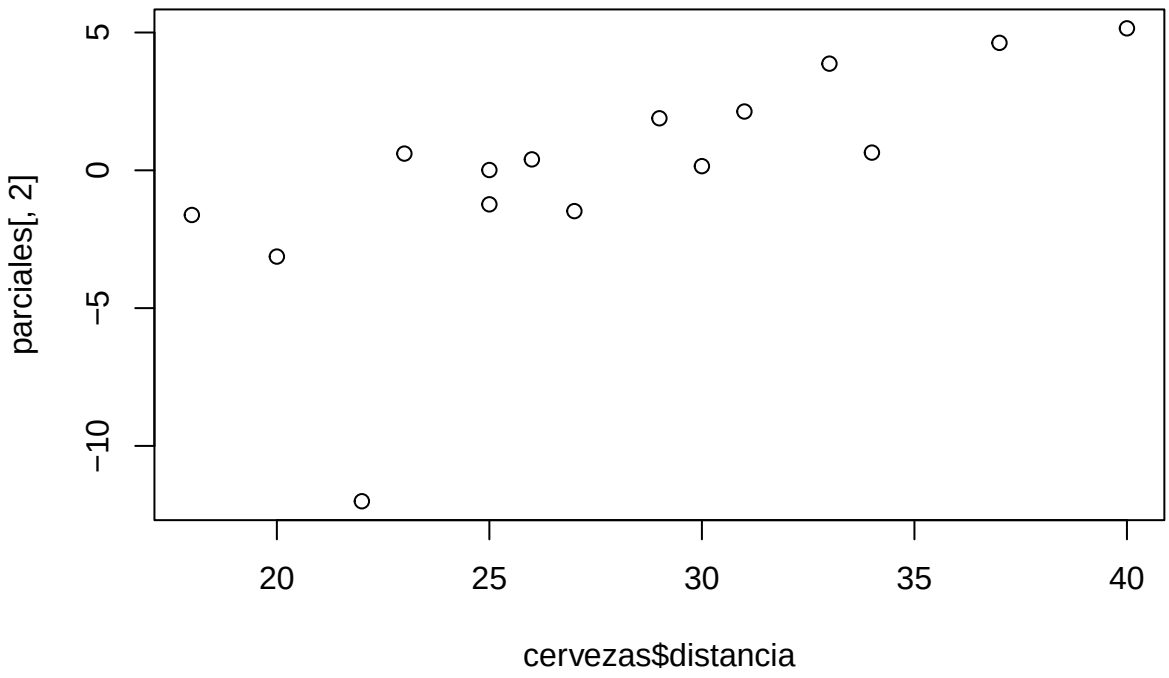
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
  parciales <- residuals(mod1,type="partial")
plot(cervezas$cajas,parciales[,1])
```

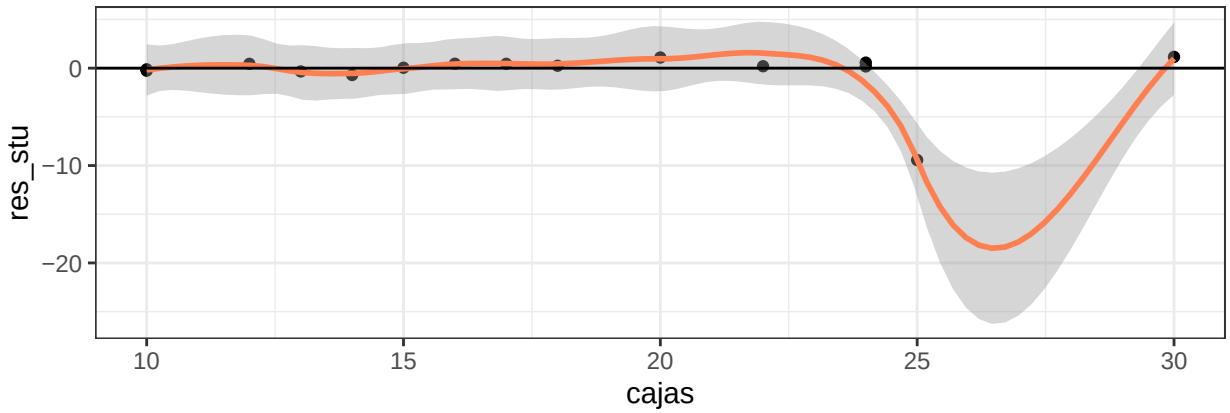


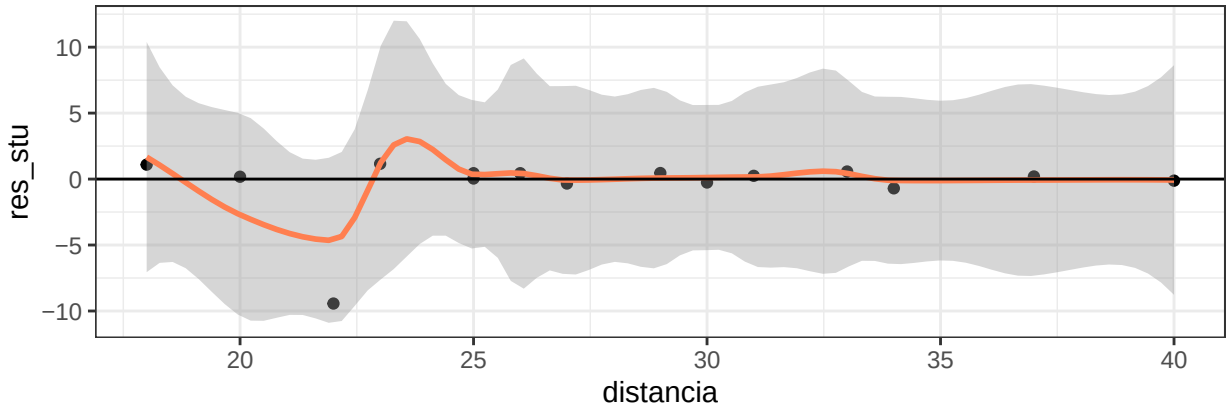

```
plot(cervezas$distancia,parciales[,2])
```




```
  plot1 <- ggplot(data = cervezas, aes(cajas, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  plot2 <- ggplot(data = cervezas, aes(distancia, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  grid.arrange(plot1, plot2)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```






```
  # Vemos que falla la linealidad, pero parece consecuencia de outliers. Luego
  # veremos si estos outliers son valores influyentes también.
```

```
library(lmtest)
```

```
## Warning: package 'lmtest' was built under R version 4.1.3
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 4.1.3
```

```
##
## Attaching package: 'zoo'
```

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

```
    bptest(mod1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mod1
## BP = 2.6416, df = 2, p-value = 0.2669
```

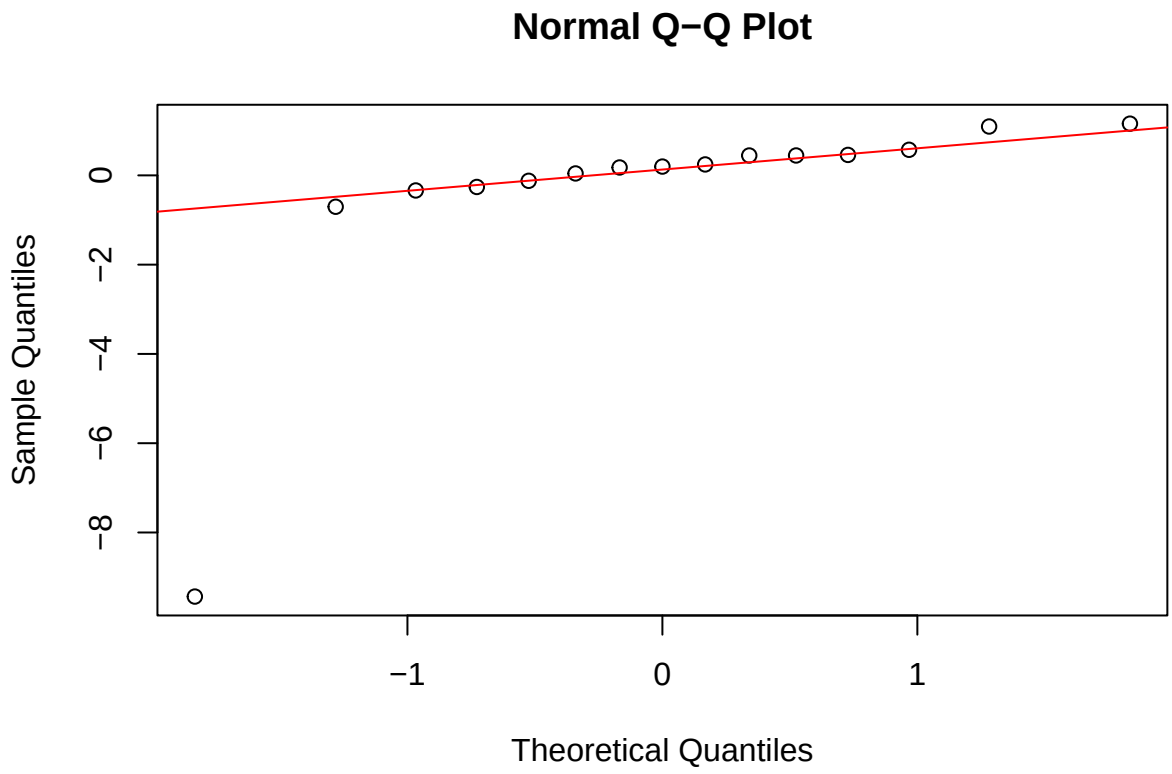
```
    # No hay problema con la homocedasticidad.

library(car)
```

```
## Loading required package: carData
## Warning: package 'carData' was built under R version 4.1.3
##
## Attaching package: 'carData'
## The following object is masked _by_ '.GlobalEnv':
##
##     Duncan
```

```
vif(mod1)  # No hay problema con la colinealidad.
```

```
##     cajas distancia
##  1.196553  1.196553
```

```
qqnorm(res_stu)
qqline(res_stu,col="red")
```




```
shapiro.test(res_stu) #Falla la normalidad.
```

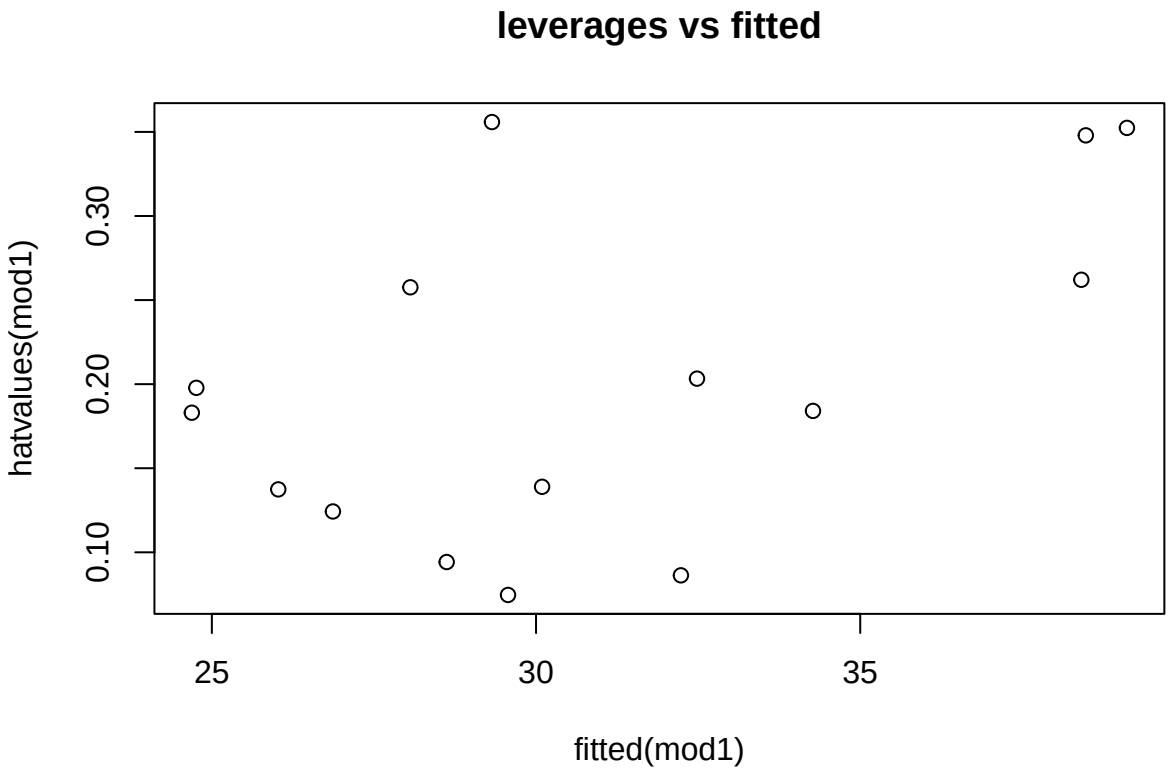
```
##
##  Shapiro-Wilk normality test
##
## data:  res_stu
## W = 0.46864, p-value = 1.985e-06
```

```
cervezas[abs(res_stu) > 3,] #El 5 es un outlier, veamos si también influyente.
```

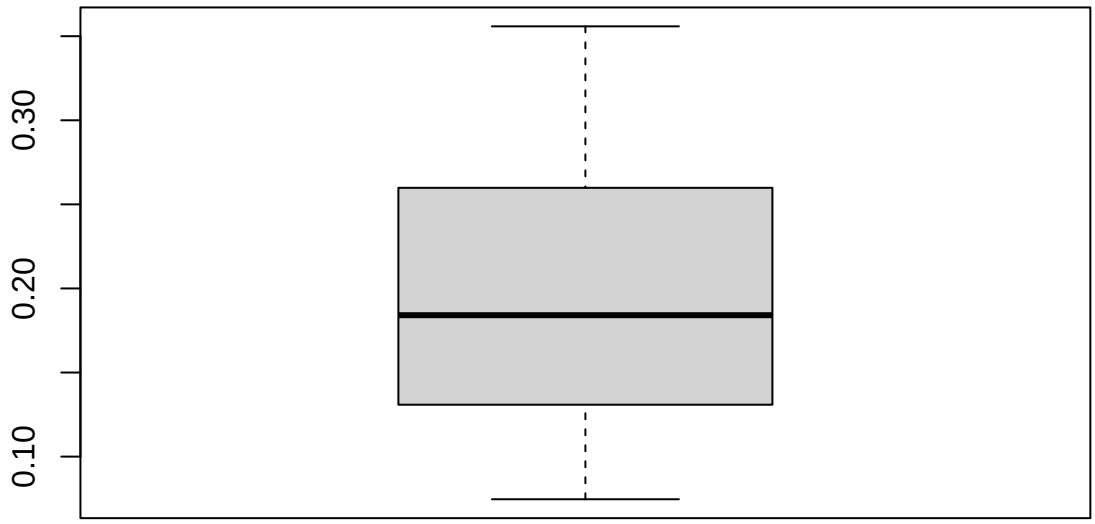
```
##   cajas distancia tiempo
```

```
## 5    25        22      25
```

```
n<-nrow(cervezas)
p<-ncol(cervezas)-1
plot(fitted(mod1),hatvalues(mod1),main="leverages vs fitted")
abline(h=2*3/n,col="red",lwd=1);
abline(h=3*3/n,col="red",lwd=3);
```



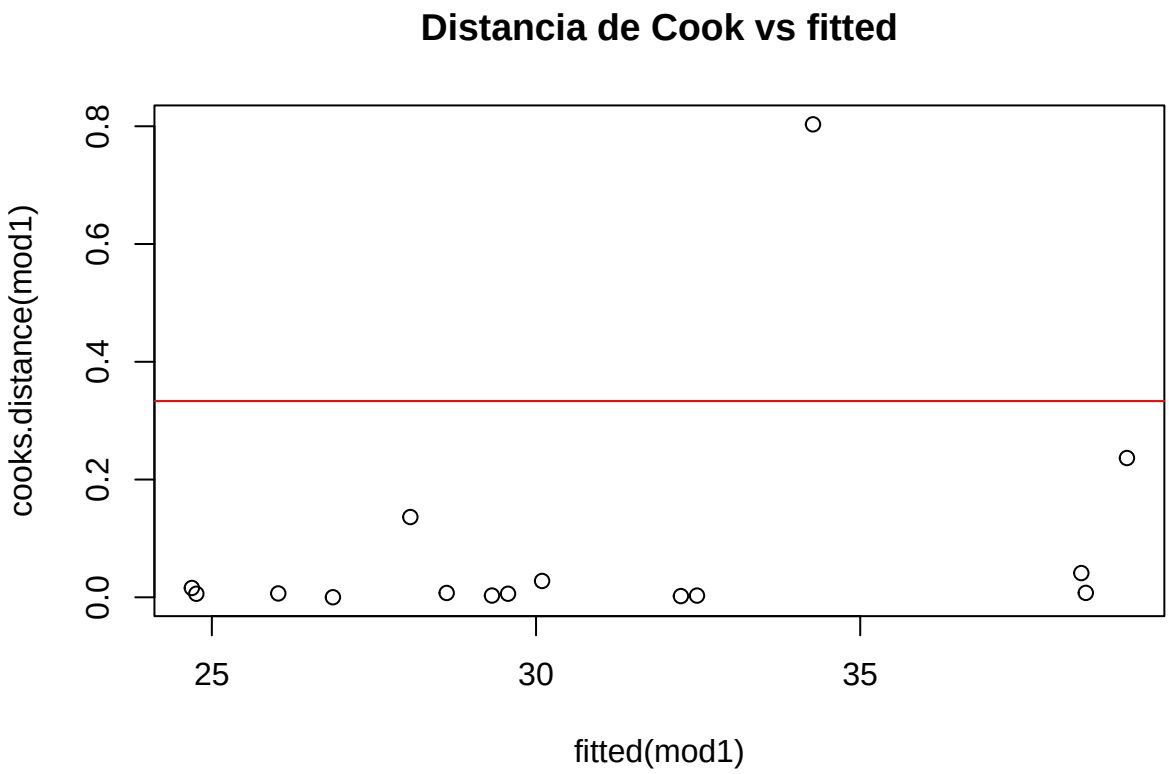

```
boxplot(hatvalues(mod1))
```



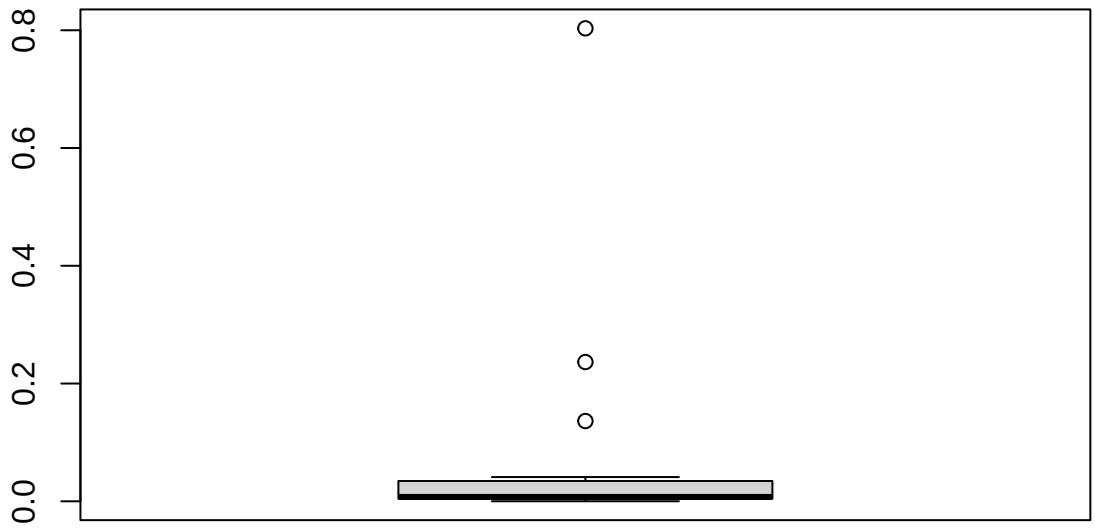

```
  summary(hatvalues(mod1))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.07466 0.13086 0.18409 0.20000 0.25984 0.35586
```

```
   plot(fitted(mod1),cooks.distance(mod1),main="Distancia de Cook vs fitted")
  abline(h=4/(n-p-1),col="red",lwd=1);
```



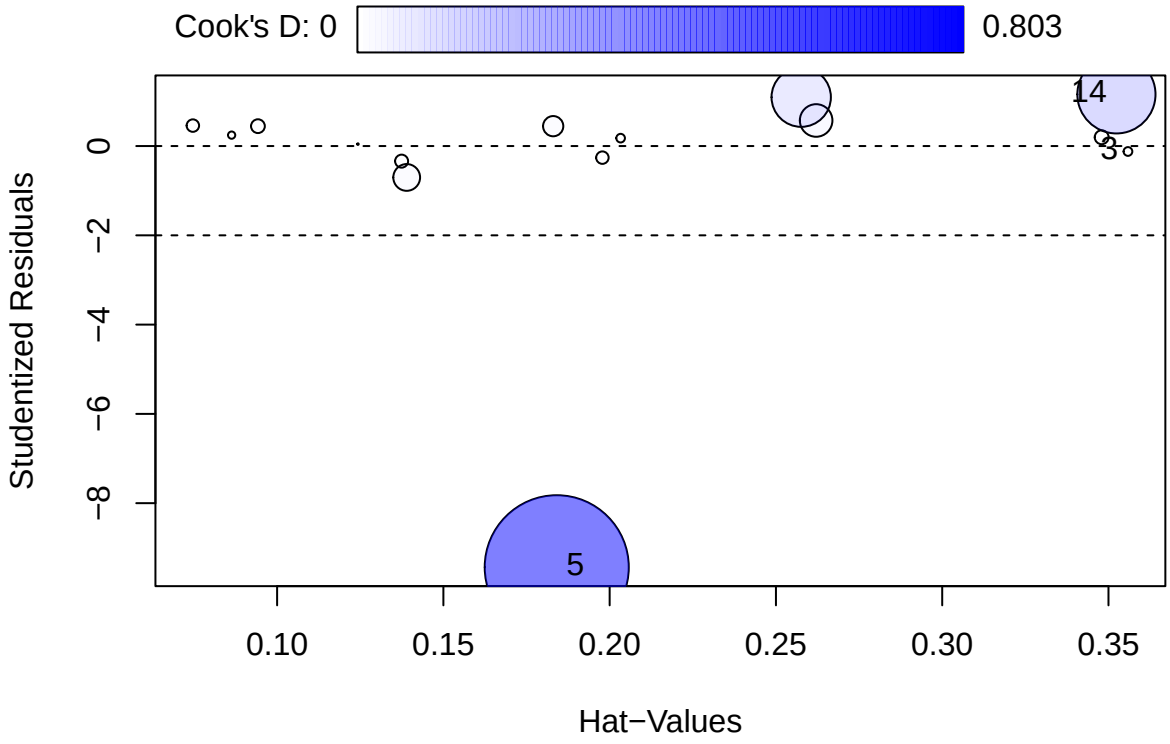

```
boxplot(cooks.distance(mod1))
```




```
  summary(cooks.distance(mod1))
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000965 0.0044914 0.0073729 0.0868279 0.0344502 0.8032552
```

```
   library(car)
  influencePlot(mod1)
```




```
##       StudRes       Hat       CookD
## 3  -0.1216537 0.3558580 0.002969127
## 5  -9.4355293 0.1840900 0.803255189
## 14  1.1581473 0.3523921 0.236559562
```

```
  summary(influence.measures(mod1))
```

```
## Potentially influential observations of
##   lm(formula = tiempo ~ ., data = cervezas, na.action = na.exclude) :
##
##    dfb.1_ dfb.cajs dfb.dstn dffit   cov.r   cook.d hat
## 3   0.03   0.02    -0.06    -0.09    2.01_*  0.00   0.36
## 5  -0.30  -2.38_*   1.48_*  -4.48_*  0.00_*  0.80   0.18
## 10 -0.12   0.09     0.12     0.14    1.97_*  0.01   0.35
```

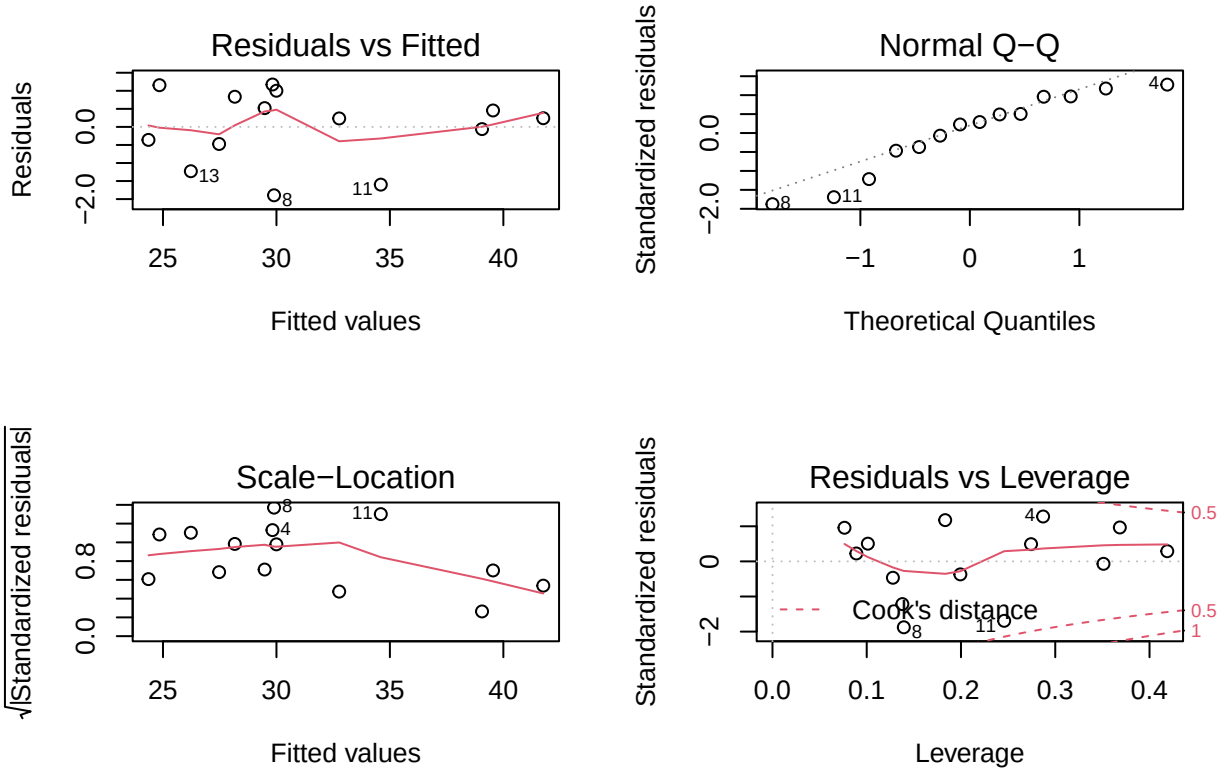
```
# La única solución sería eliminar el valor influyente. Siempre y cuando la investigación
# lo permita.
cervezas2<-cervezas[-5,]
```

```
mod2 <- lm(tiempo ~ ., data=cervezas2, na.action=na.exclude)
summary(mod2)
```

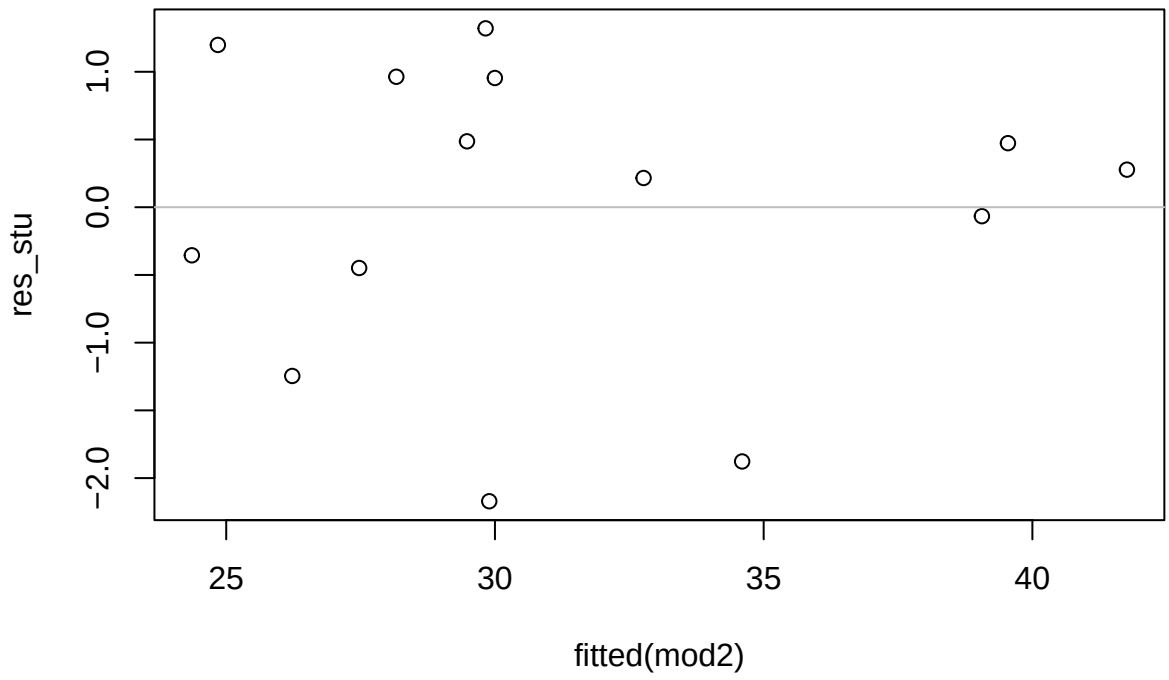
```
##
## Call:
## lm(formula = tiempo ~ ., data = cervezas2, na.action = na.exclude)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -1.8944 -0.4451  0.2374  0.7567  1.1740
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.91517    2.02974   1.436    0.179
## cajas        1.00314    0.05466  18.352 1.34e-09 ***
## distancia    0.38045    0.05146   7.393 1.37e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.088 on 11 degrees of freedom
## Multiple R-squared:  0.9685, Adjusted R-squared:  0.9627
## F-statistic: 168.9 on 2 and 11 DF,  p-value: 5.533e-09
```

```
par(mfrow=c(2,2))
plot(mod2)
```

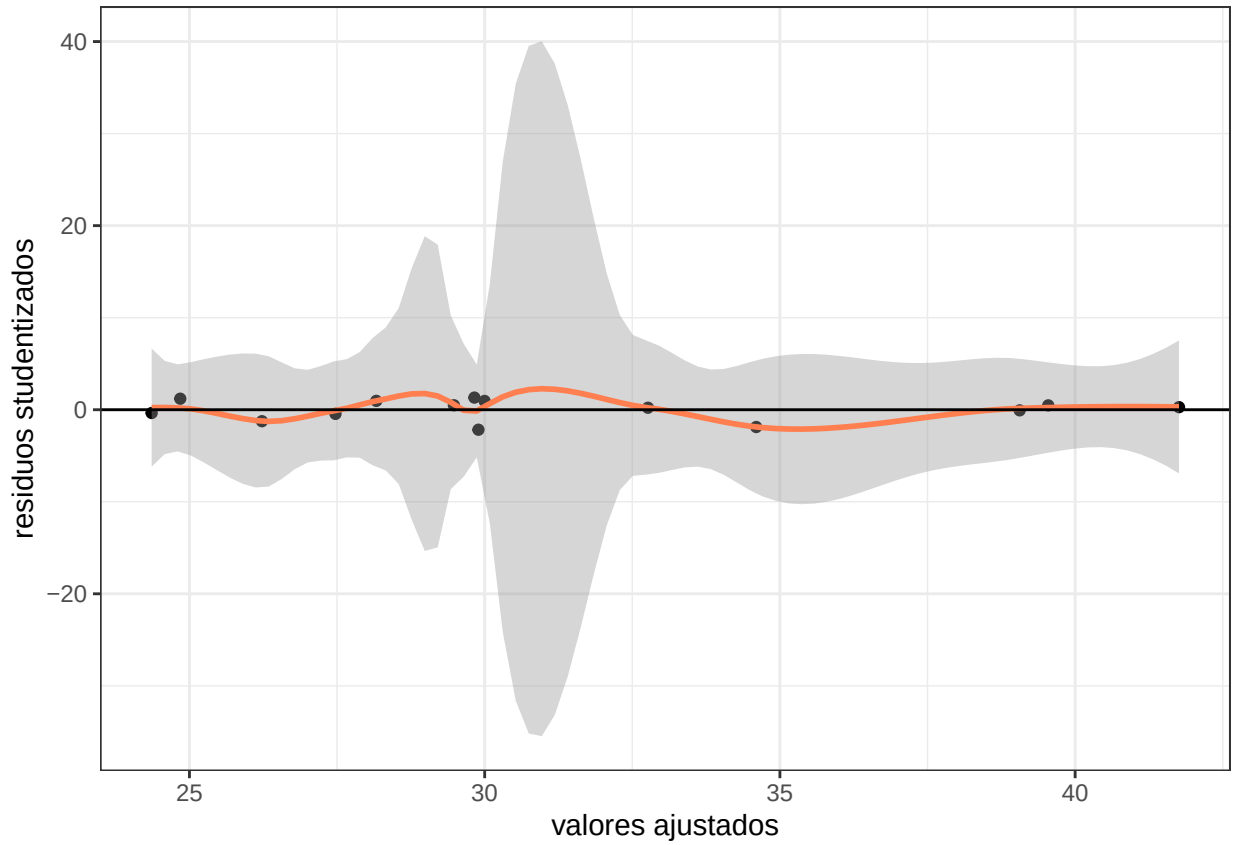



```
res_stu<-rstudent(mod2)
par(mfrow=c(1,1))
plot(fitted(mod2),res_stu)
 abline(h=0, col="gray")
```

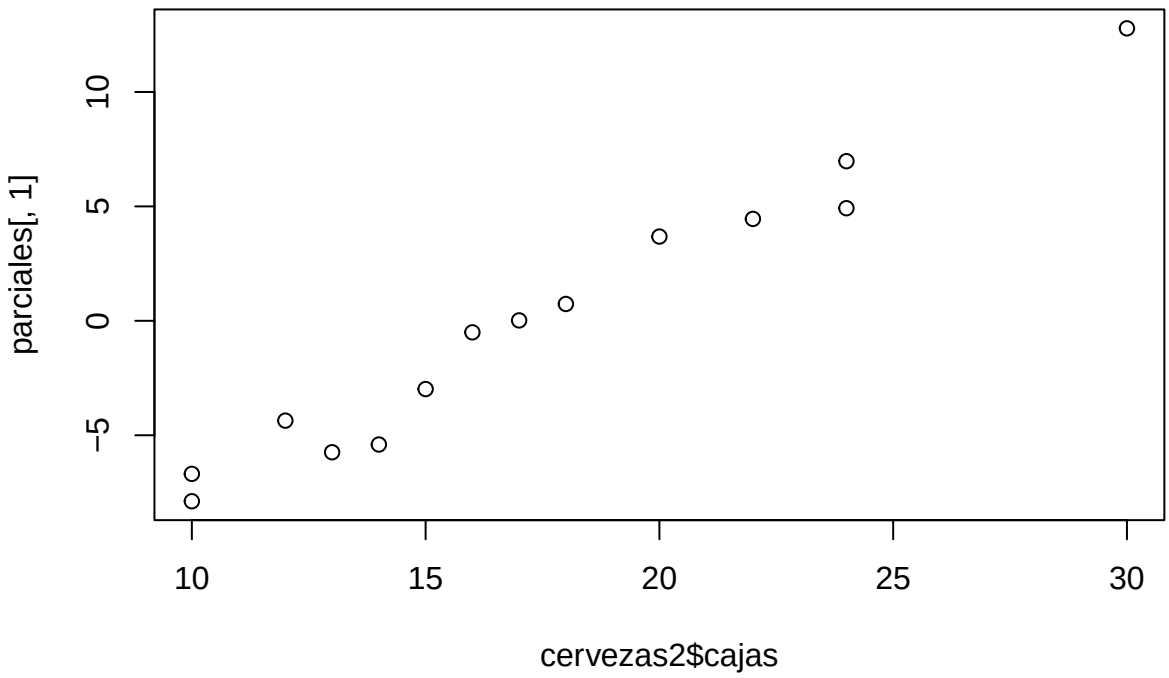



```
library(ggplot2)
 library(gridExtra)
 ggplot(data = cervezas2, aes(x =fitted(mod2), y = res_stu)) +
 geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
 labs(y = "residuos studentizados",
      x = "valores ajustados") +
 theme_bw()
```

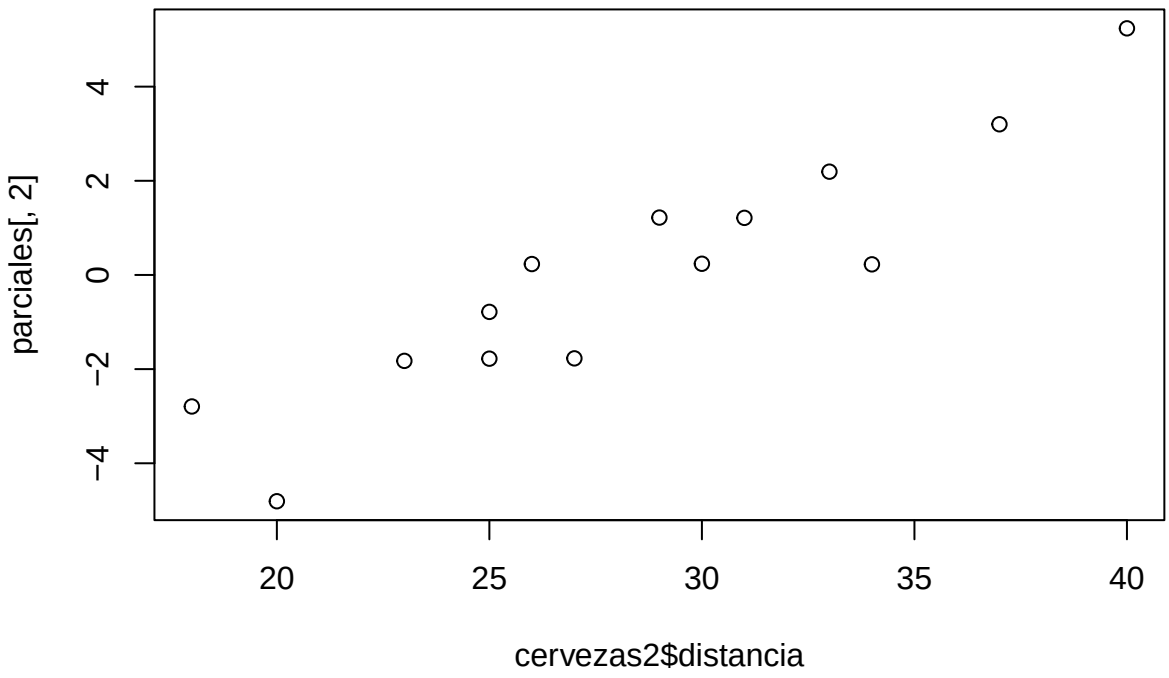
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
  parciales <- residuals(mod2,type="partial")
plot(cervezas2$cajas,parciales[,1])
```




```
plot(cervezas2$distancia,parciales[,2])
```




```
  plot1 <- ggplot(data = cervezas2, aes(cajas, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  plot2 <- ggplot(data = cervezas2, aes(distancia, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  grid.arrange(plot1, plot2)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 12

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 2

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 12

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 2

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
```

```
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 27
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 2
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

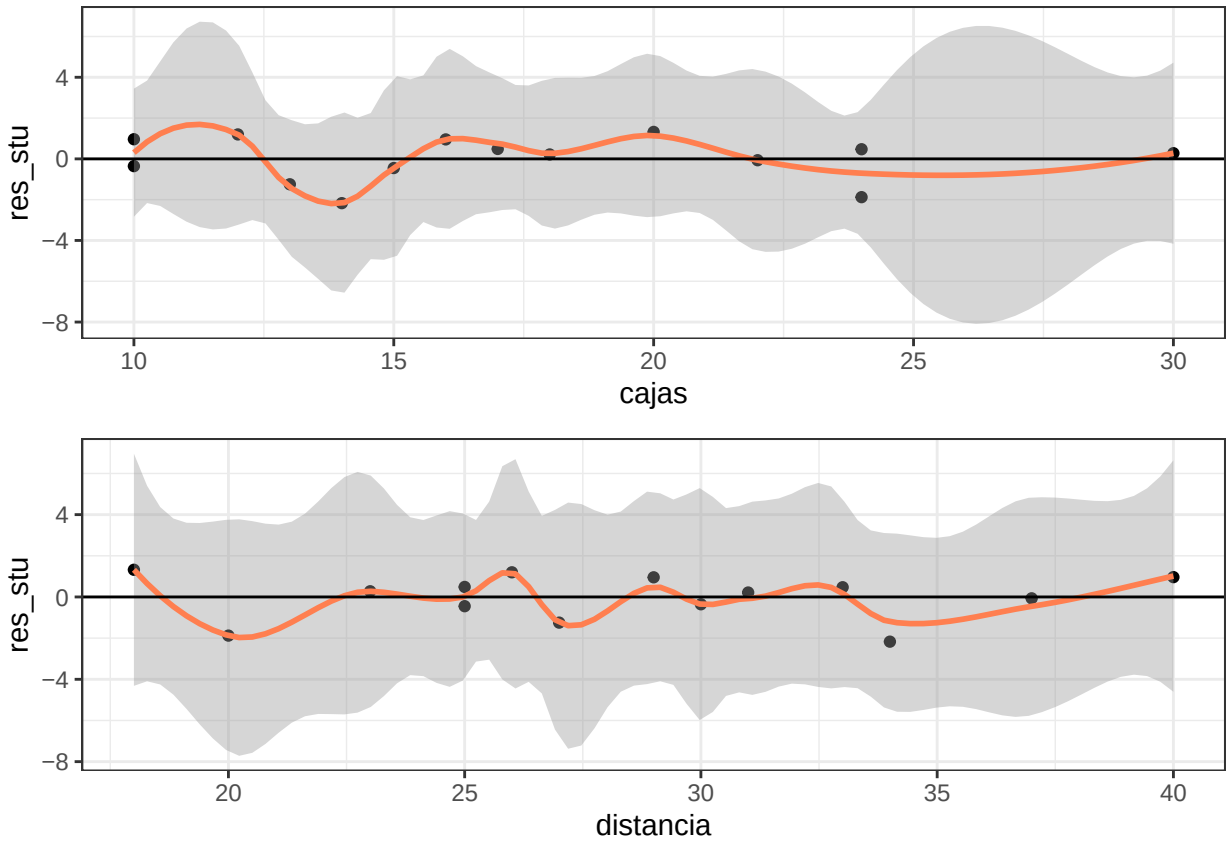
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 4
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at 27
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 2
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 4
```




```
  # Parece que ya no tenemos problemas con la linealidad
```

```
library(lmtest)
    bptest(mod2)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mod2
## BP = 1.038, df = 2, p-value = 0.5951
```

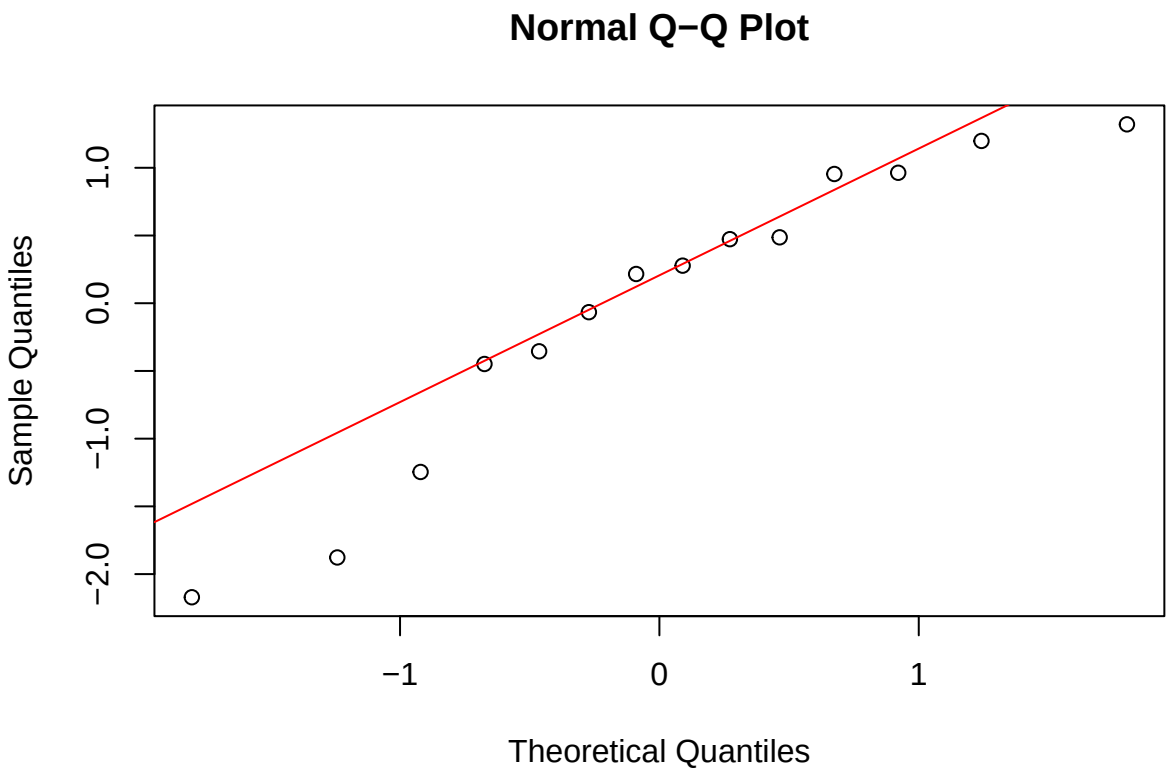
```
    # No hay problema con la homocedasticidad.

library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 4.1.3
```

```
vif(mod2)  # No hay problema con la colinealidad.
```

```
##     cajas distancia
##  1.139873  1.139873
```

```
qqnorm(res_stu)
qqline(res_stu,col="red")
```



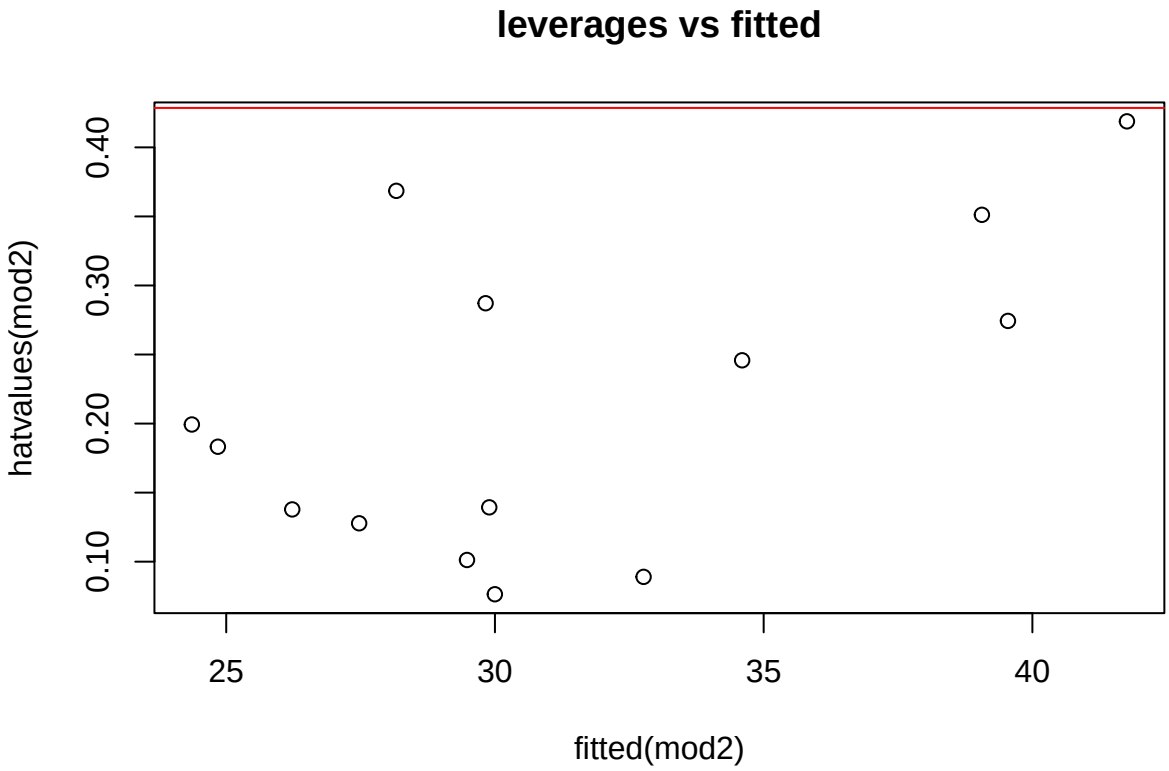

```
shapiro.test(res_stu) #No hay problema con la normalidad.
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res_stu
## W = 0.91628, p-value = 0.1944
```

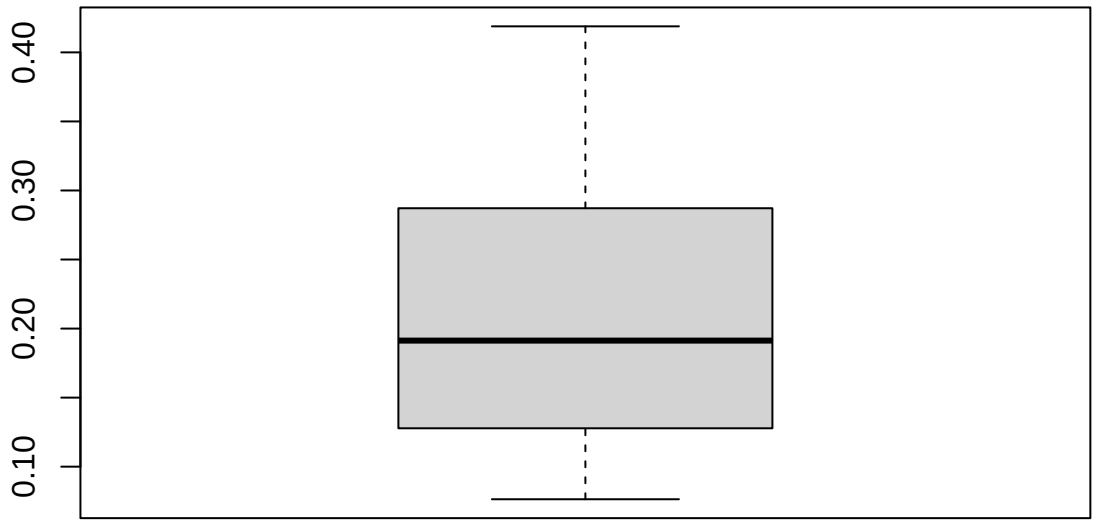
```
cervezas2[abs(res_stu) > 3,] #No hay outliers
```

```
## [1] cajas     distancia tiempo
## <0 rows> (or 0-length row.names)
```

```
  n<-nrow(cervezas2)
  p<-ncol(cervezas2)
  plot(fitted(mod2),hatvalues(mod2),main="leverages vs fitted")
  abline(h=2*3/n,col="red",lwd=1);
  abline(h=3*3/n,col="red",lwd=3);
```



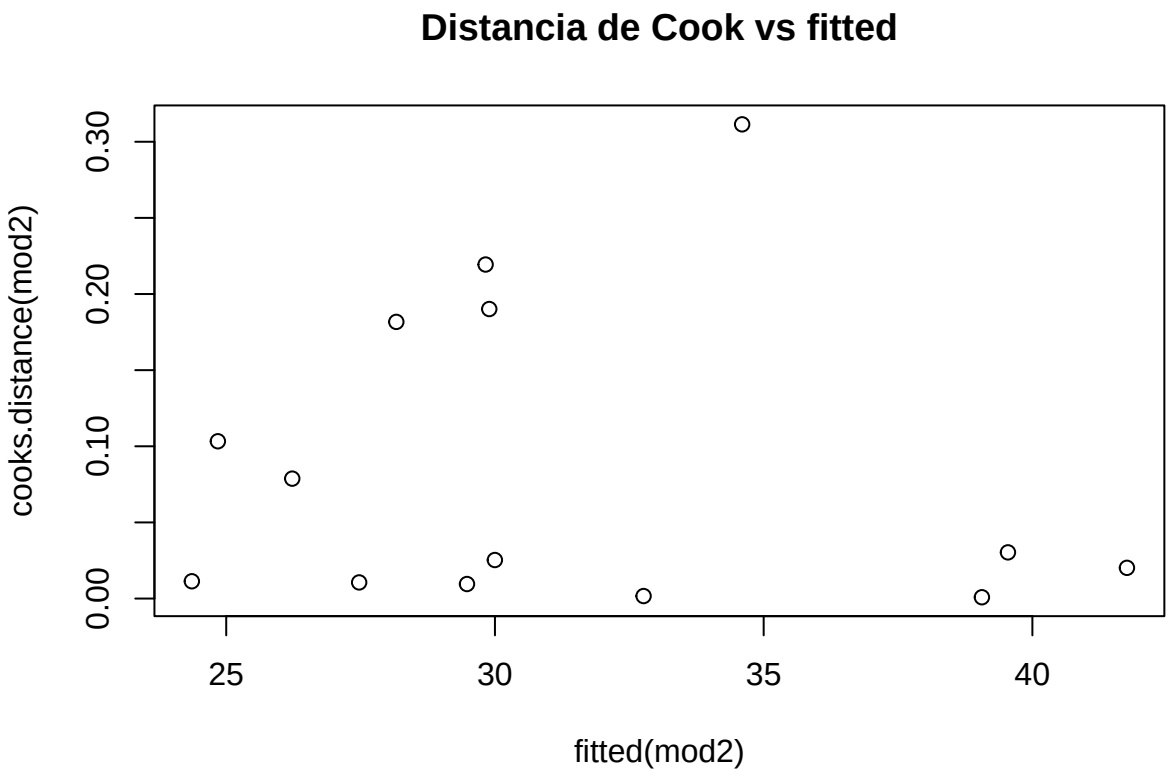

```
boxplot(hatvalues(mod2))
```



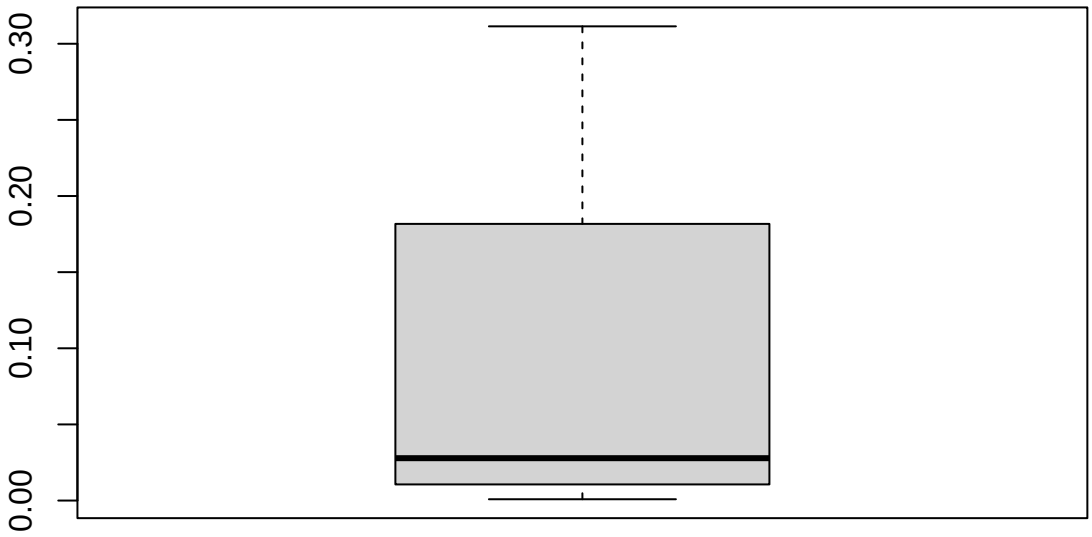

```
  summary(hatvalues(mod2))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.07641 0.13029 0.19129 0.21429 0.28394 0.41885
```

```
   plot(fitted(mod2),cooks.distance(mod2),main="Distancia de Cook vs fitted")
  abline(h=4/(n-p-1),col="red",lwd=1);
```



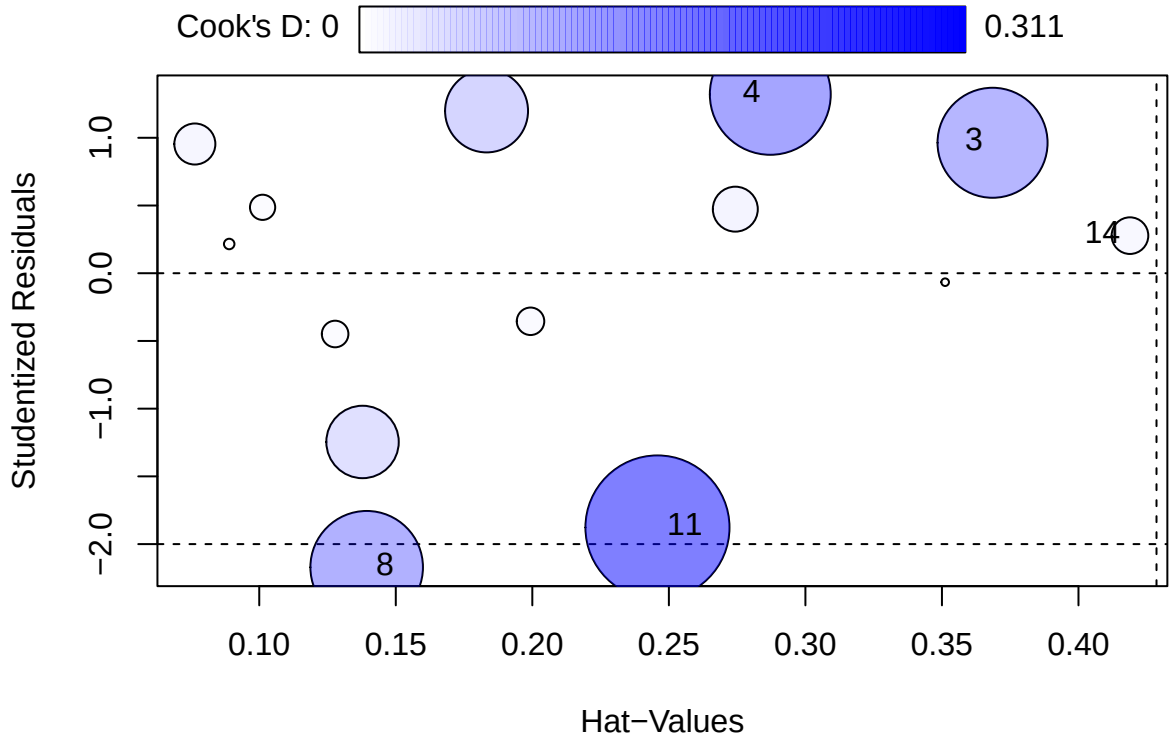

```
boxplot(cooks.distance(mod2))
```




```
  summary(cooks.distance(mod2))
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0008695 0.0107933 0.0278257 0.0853285 0.1620988 0.3114330
```

```
   library(car)
  influencePlot(mod2)
```



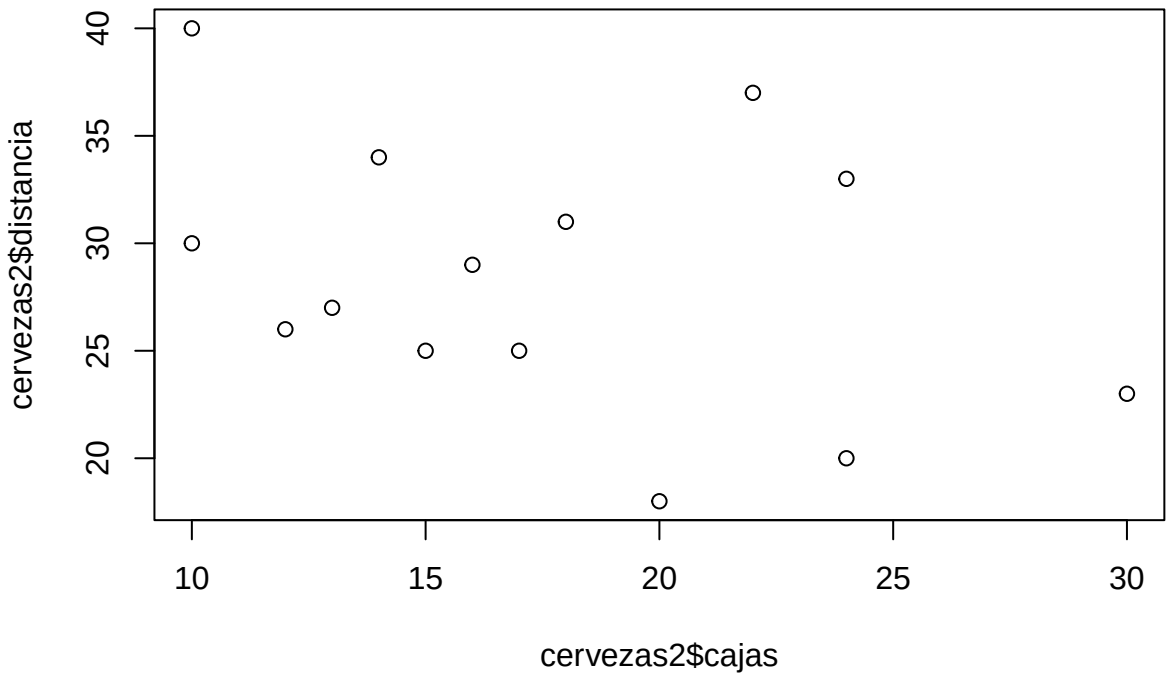

```
##       StudRes       Hat      CookD
## 3   0.9632676 0.3685313 0.18169872
## 4   1.3207490 0.2871398 0.21936684
## 8  -2.1709007 0.1393403 0.19015201
## 11 -1.8772551 0.2458253 0.31143297
## 14  0.2774491 0.4188495 0.02018726
```

```
summary(influence.measures(mod2))
```

```
## Potentially influential observations of
##   lm(formula = tiempo ~ ., data = cervezas2, na.action = na.exclude) :
##
##    dfb.1_ dfb.cajs dfb.dstn dffit cov.r   cook.d hat
## 10  0.04  -0.03    -0.04    -0.05  2.05_*  0.00   0.35
## 14 -0.07   0.20    -0.01     0.24  2.24_*  0.02   0.42
```

c) Puedes realizar una predicción para un nuevo reparto que consiste en llevar 20 cajas a 40 km de distancia y dar su error de predicción. ¿Y si hay que llevarlas a 70km?

```
plot(cervezas2$cajas,cervezas2$distancia)
```




```
x0 <- data.frame(cajas=20, distancia=40)
predict(mod2,newdata=x0,interval='prediction')
```

```
##        fit      lwr      upr
## 1 38.19591 35.32773 41.06409
```

```
# A 70km no se puede hacer la predicción, dado que la muestra que hemos tomado
# llega únicamente a los 40km
```

EXTRA: Vamos a calcular el error de predicción con las distintas metodologías vistas en clase

```
### Validación
set.seed(12345)
seleccion <- sample(nrow(cervezas2),round(nrow(cervezas2)*3/4))
entrenamiento <- cervezas2[seleccion,]
prueba <- cervezas2[-seleccion,]


ajuste <- lm(tiempo ~ .,data=entrenamiento)
summary(ajuste)
```

```
##
## Call:
## lm(formula = tiempo ~ ., data = entrenamiento)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.32115 -0.52952  0.07806  0.66350  1.10200
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.69026    2.01925   0.837     0.43
## cajas        0.98530    0.05166  19.073 2.71e-07 ***
## distancia    0.43785    0.05234   8.365 6.85e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9455 on 7 degrees of freedom
## Multiple R-squared:  0.9815, Adjusted R-squared:  0.9762
## F-statistic: 185.3 on 2 and 7 DF,  p-value: 8.673e-07
```

```
prediccion <- predict(ajuste,prueba)
(ecm <- mean((prueba$tiempo-prediccion)^2))
```

```
## [1] 2.262805
```

```
### Validación (variabilidad)
resultados <- NULL
for (i in 1:20) {
  seleccion <- sample(nrow(cervezas2),round(nrow(cervezas2)*3/4))
entrenamiento <- cervezas2[seleccion,]
prueba <- cervezas2[-seleccion,]
ajuste <- lm(tiempo ~ .,data=entrenamiento)
summary(ajuste)
prediccion <- predict(ajuste,prueba)
  resultados <- c(resultados,mean((prueba$tiempo-prediccion)^2))
}
summary(resultados)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.7322  0.8978  1.6467  1.6915  2.1846  3.0272
```

```
### Validación cruzada
library(boot)
```

```
##
## Attaching package: 'boot'
```

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

```
ajuste <- glm(tiempo ~ ., data=cervezas2)
set.seed(12345)
ecm <- cv.glm(cervezas2,ajuste,K=5)
ecm$delta[1]
```

```
## [1] 2.032126
```

```
### Validación cruzada (variabilidad)

resultadosK <- NULL
for (i in 1:20) {
   resultadosK <- c(resultadosK,cv.glm(cervezas2,ajuste,K=5)$delta[1])
}
summary(resultadosK)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.105   1.389   1.436   1.501   1.622   2.141
```

```
# Bootstrap de la estimación de los coeficientes de la regresión lineal múltiple
B <- 1000
boot.fun <- function(datos,indice) {
  coeficientes <- coef(lm(tiempo~.,
                          data=datos,subset=indice))
}
set.seed(12345)
boot(cervezas2,boot.fun,B)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = cervezas2, statistic = boot.fun, R = B)
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 2.9151688  0.1369284374  1.92204539
## t2* 1.0031356  0.0004652882  0.05042081
## t3* 0.3804508 -0.0057697726  0.05743081
```

```
### Bootstrap de una nueva predicción
xcajas <- 12
xdistancia <- 33
ajuste <- lm(tiempo ~ ., data=cervezas2)
predict(ajuste,newdata = data.frame(cajas=xcajas,distancia=xdistancia), se.fit=TRUE,interval = 'confider
```

```
## $fit
##        fit      lwr      upr
## 1 27.50767 26.57207 28.44328
##
## $se.fit
## [1] 0.4250839
##
## $df
## [1] 11
##
## $residual.scale
## [1] 1.087843
```

```
B <- 1000
boot.fun <- function(datos,indice,x1=x1,x2=x2) {
  coeficientes <- coef(lm(tiempo~.,
                          data=datos,subset=indice))
  return(coeficientes[1]+coeficientes[2]*x1+coeficientes[3]*x2)
}
set.seed(12345)
boot(cervezas2,boot.fun,B,x1=xcajas,x2=xdistancia)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
```

```
##
##
## Call:
## boot(data = cervezas2, statistic = boot.fun, R = B, x1 = xcajas,
##     x2 = xdistancia)
##
##
## Bootstrap Statistics :
##     original     bias    std. error
## t1* 27.50767 -0.0478906   0.4553085
```

Ejercicio 2

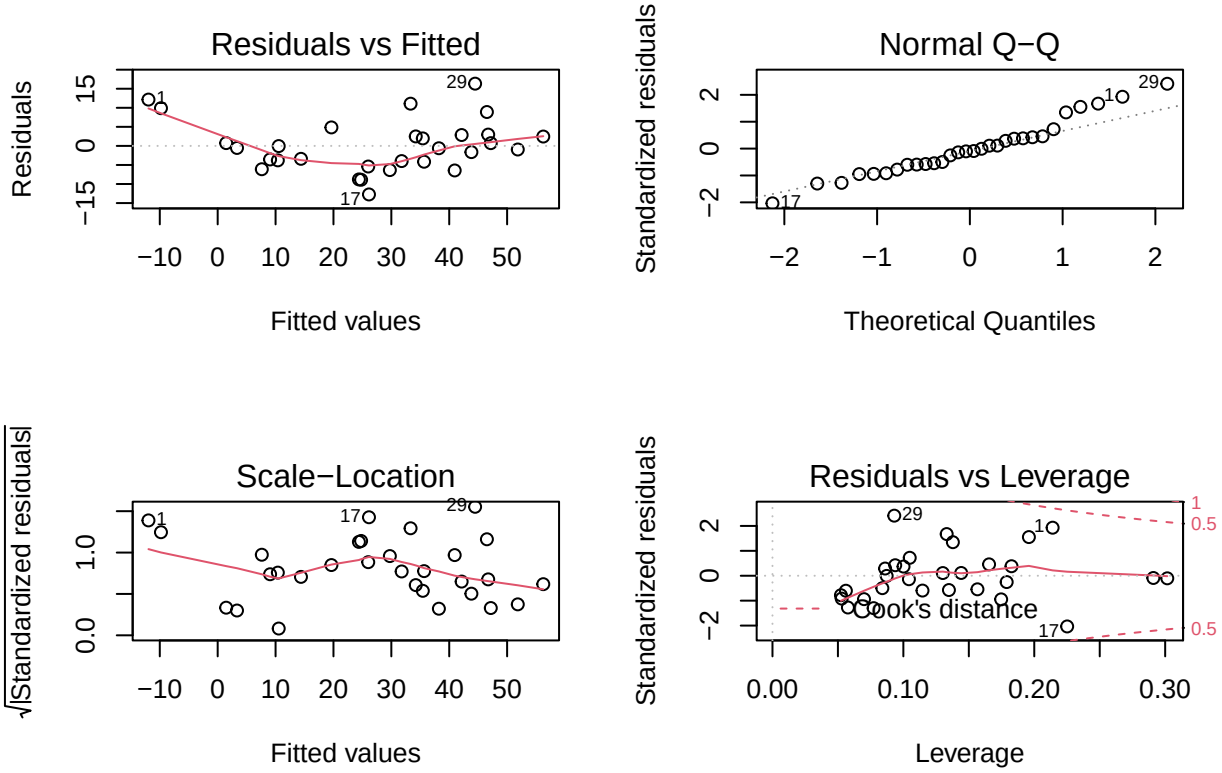
En los procesos de producción, hay bastante confusión sobre cuáles son las partes del proceso que hacen que ocurran desviaciones del resultado final que se está buscando. Existen diferentes factores que pueden influir: la temperatura del proceso de producción, la densidad del producto, o la propia tasa de producción. El fichero **defectos** contiene información sobre el número medio de defectos (en cada lote analizado) encontrados en 30 pruebas, junto con el valor de las covariables antes comentadas.

a) Ajusta un modelo para predecir el número medio de defectos con la información disponible, diagnostica el modelo y evalúa la efectividad de posibles soluciones.

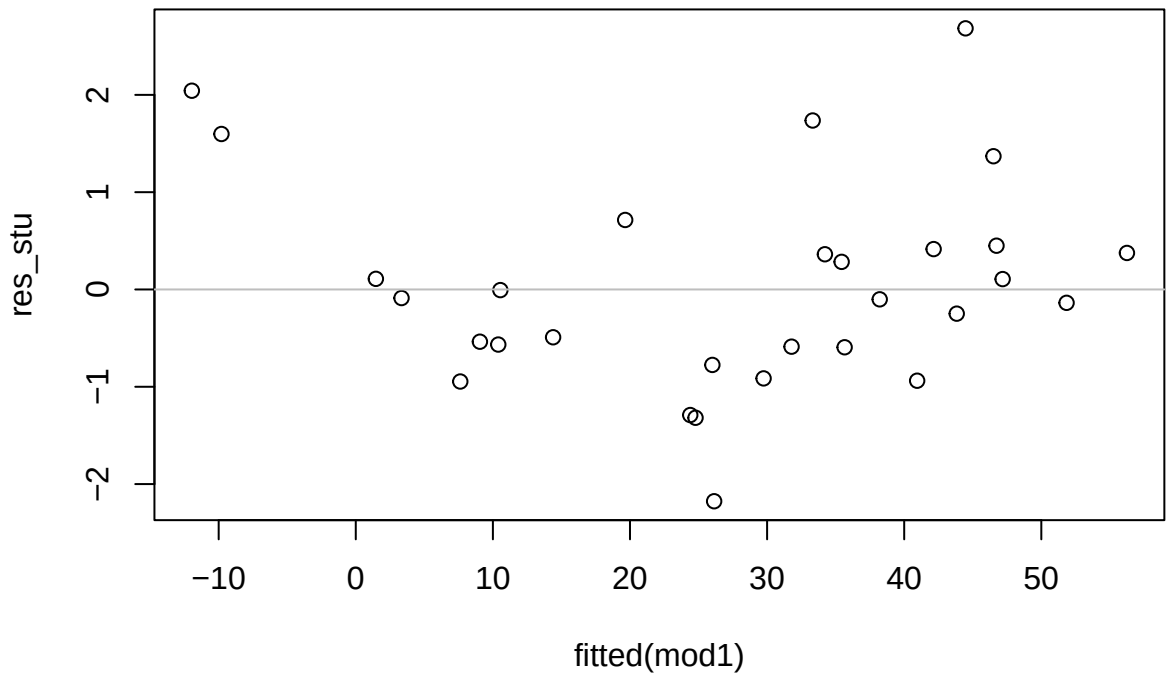
```
mod1 <- lm(defectuosos ~ temperatura+densidad+tasa, data=defectos, na.action=na.exclude)
summary(mod1)
```

```
##
## Call:
## lm(formula = defectuosos ~ temperatura + densidad + tasa, data = defectos,
##     na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.7367  -4.1116  -0.5755   2.7617  16.3279
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.3244    65.9265   0.157   0.8768
## temperatura  16.0779     8.2941   1.938   0.0635 .
## densidad     -1.8273     1.4971  -1.221   0.2332
## tasa          0.1167     0.1306   0.894   0.3797
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.11 on 26 degrees of freedom
## Multiple R-squared:  0.8797, Adjusted R-squared:  0.8658
## F-statistic: 63.36 on 3 and 26 DF,  p-value: 4.371e-12
```

```
# Observamos que el modelo es significativo, aunque por separado sus coeficientes no lo son.
# Vamos a diagnosticarlo

par(mfrow=c(2,2))
plot(mod1)
```

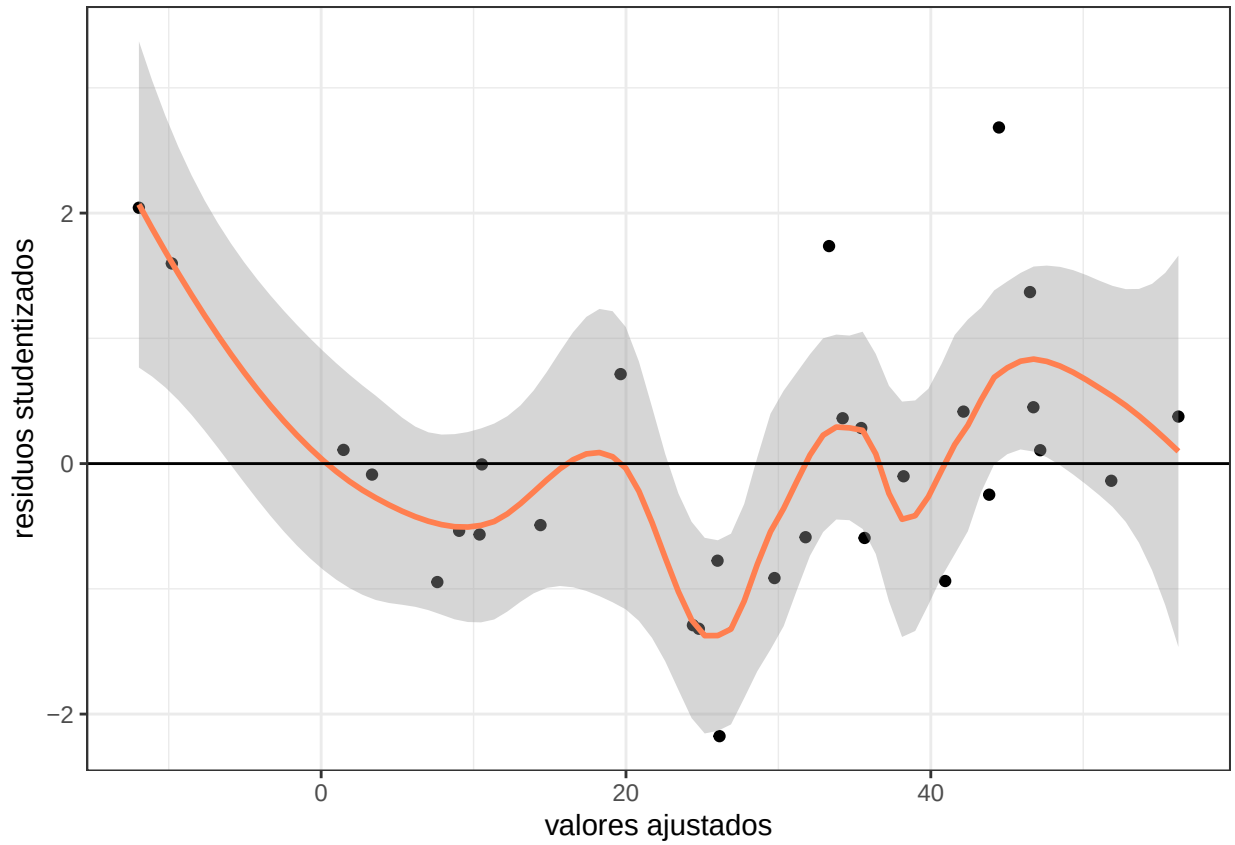



```
res_stu<-rstudent(mod1)
par(mfrow=c(1,1))
plot(fitted(mod1),res_stu)
 abline(h=0, col="gray")
```

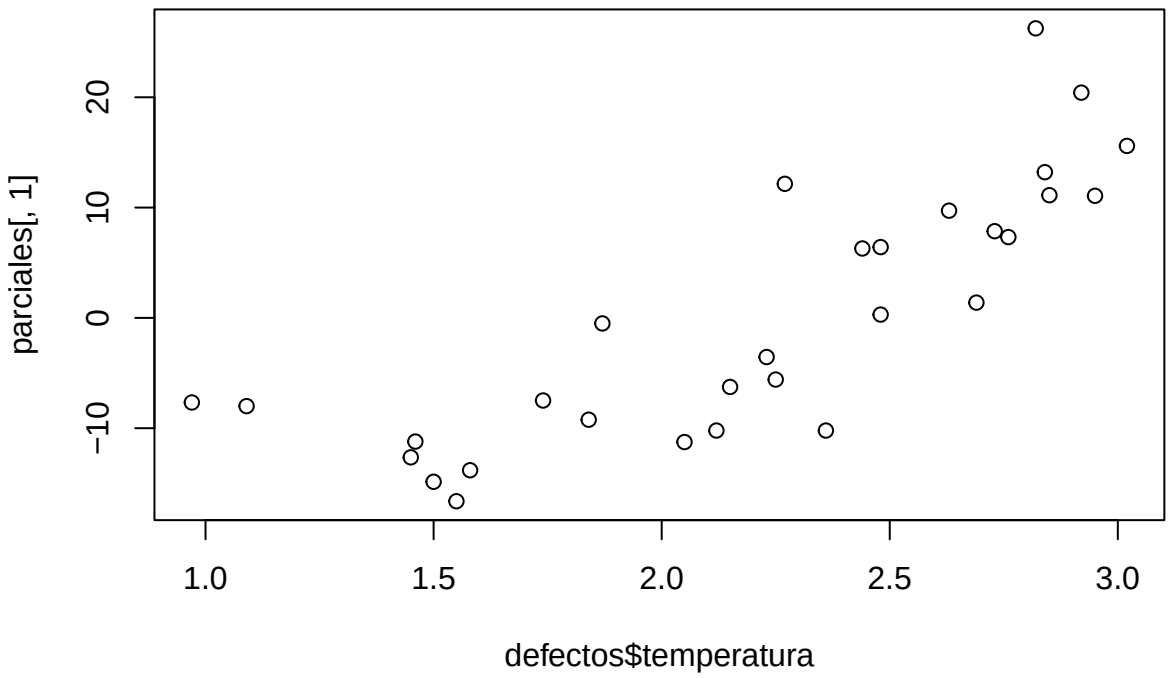



```
library(ggplot2)
 library(gridExtra)
 ggplot(data = defectos, aes(x =fitted(mod1), y = res_stu)) +
 geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
 labs(y = "residuos studentizados",
      x = "valores ajustados") +
 theme_bw()
```

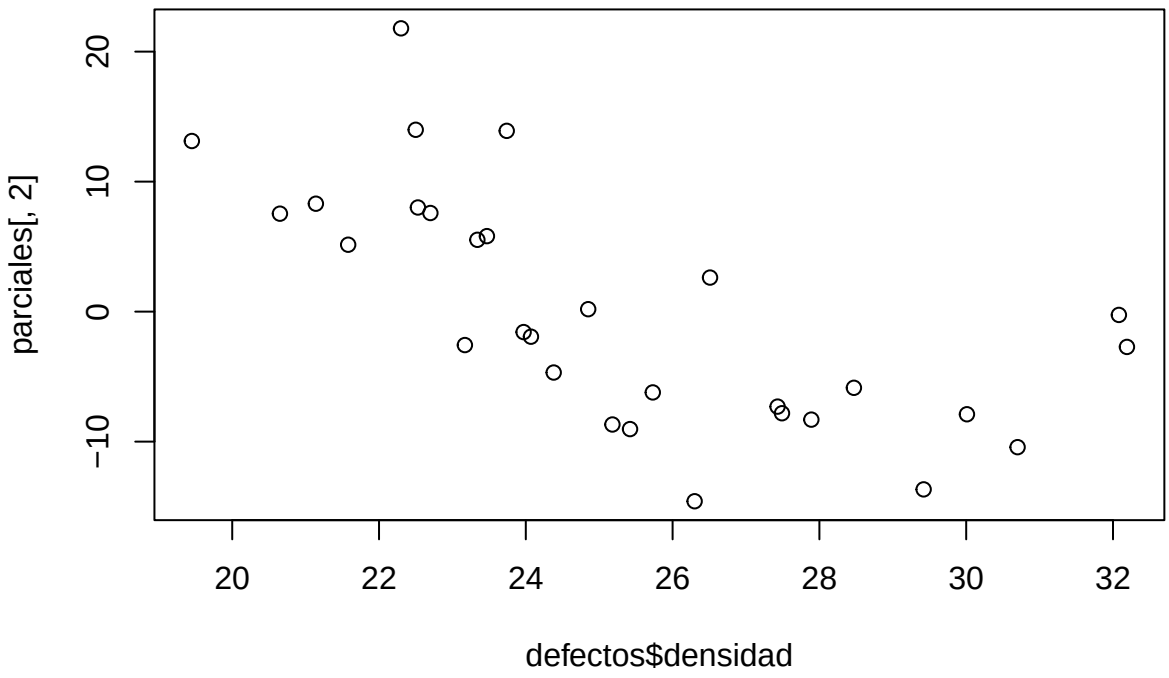
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



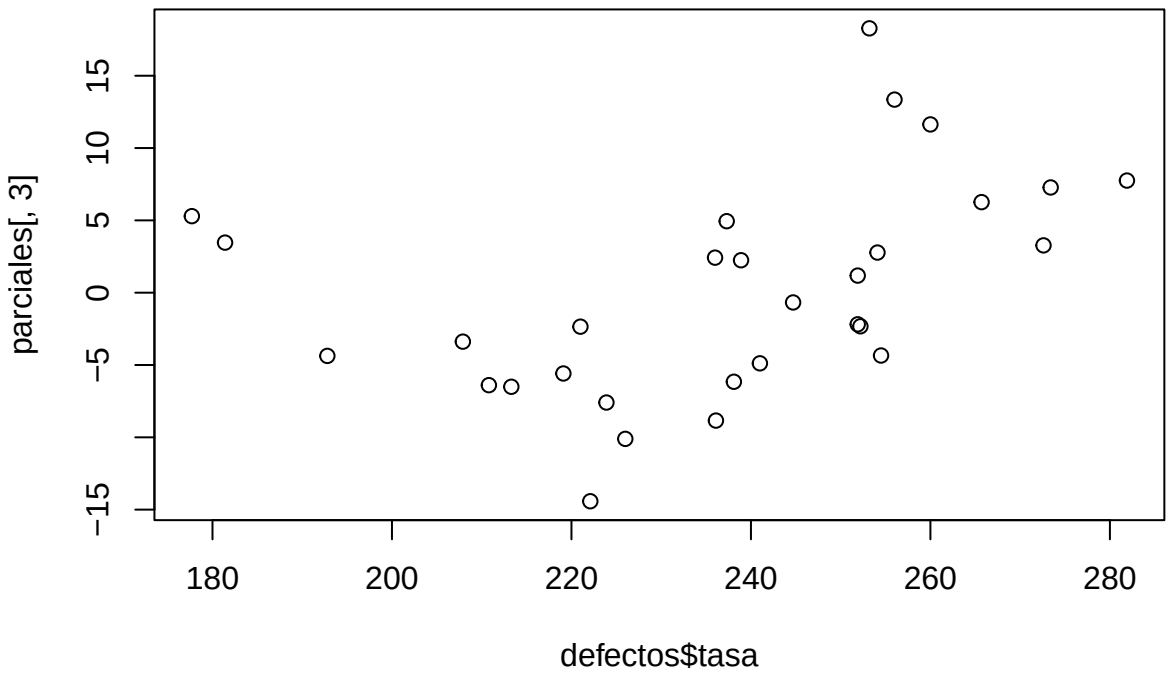

```
  parciales <- residuals(mod1,type="partial")
plot(defectos$temperatura,parciales[,1])
```




```
plot(defectos$densidad,parciales[,2])
```

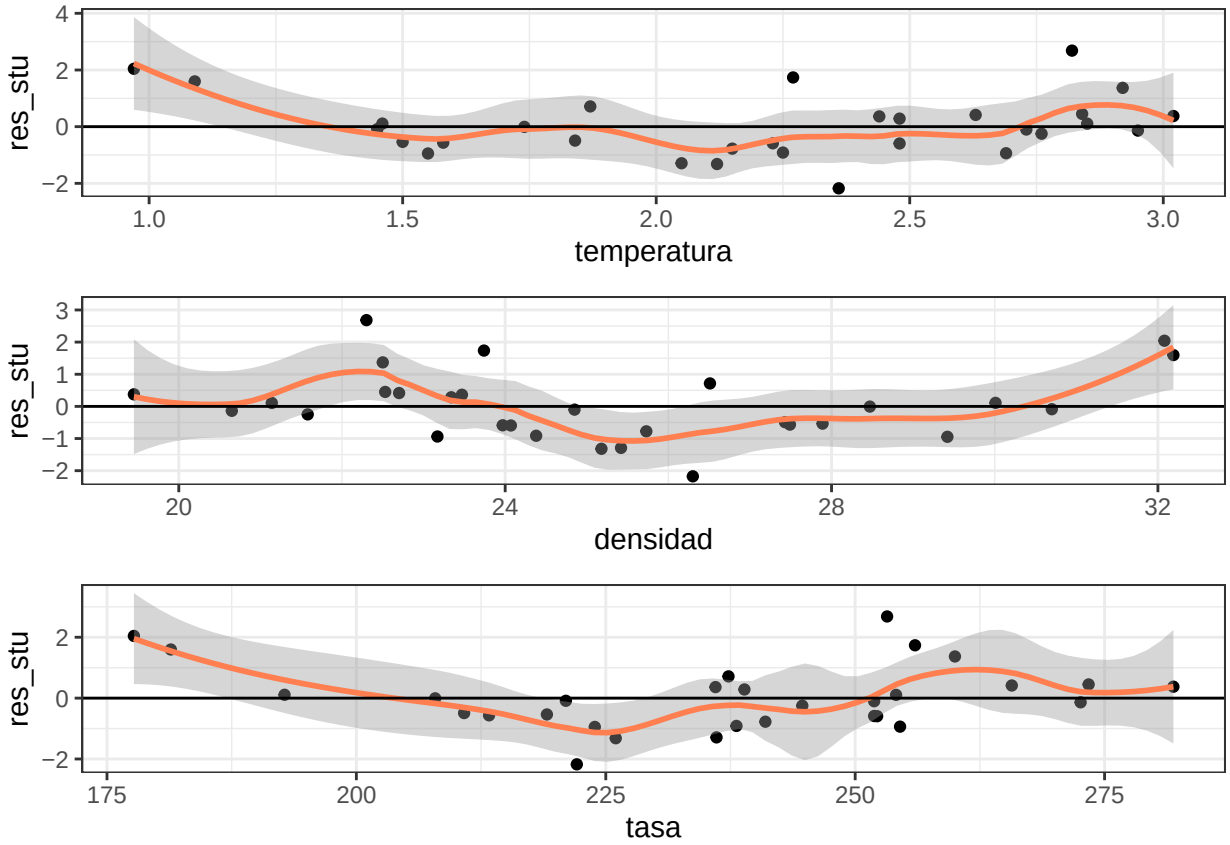



```
plot(defectos$tasa,parciales[,3])
```




```
  plot1 <- ggplot(data = defectos, aes(temperatura, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  plot2 <- ggplot(data = defectos, aes(densidad, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  plot3 <- ggplot(data = defectos, aes(tasa, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  grid.arrange(plot1, plot2,plot3)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



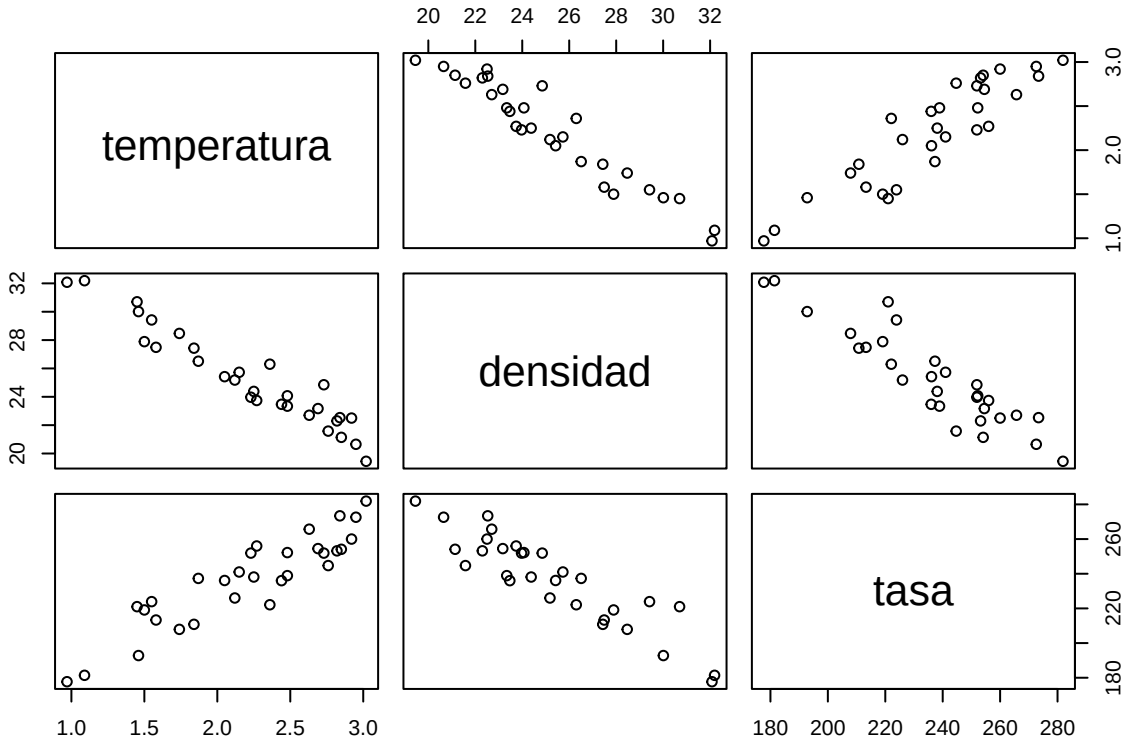

```
  library(lmtest)
    bptest(mod1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mod1
## BP = 2.1985, df = 3, p-value = 0.5322
```

```
    # No hay problema con la homocedasticidad.

library(car)
vif(mod1)  # Hay problema con la multicolinealidad
```

```
## temperatura    densidad        tasa
##   13.431614   14.508872    6.642619
```

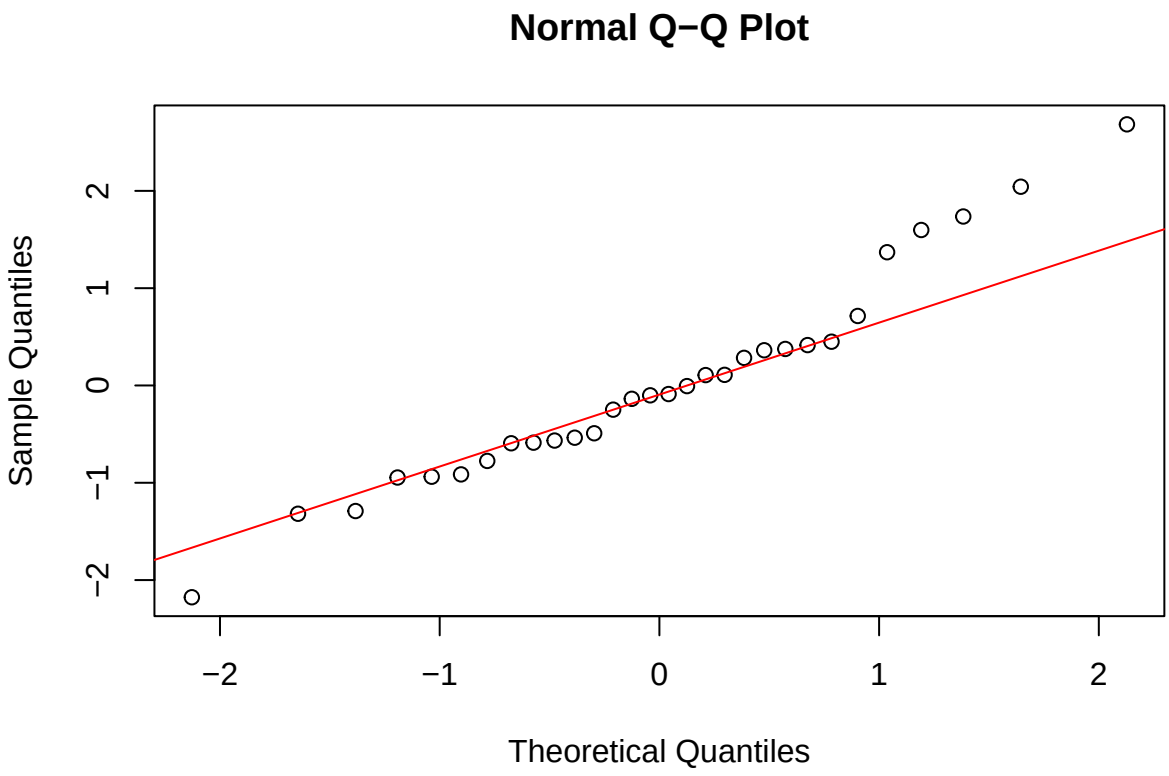
```
pairs(defectos[,c('temperatura','densidad','tasa')])
```




```
cor(defectos[,c('defectuosos','temperatura','densidad','tasa')])
```

```
##              defectuosos temperatura   densidad       tasa
## defectuosos    1.0000000   0.9289640 -0.9232497  0.8851935
## temperatura    0.9289640   1.0000000 -0.9591021  0.9082920
## densidad      -0.9232497  -0.9591021  1.0000000 -0.9154169
## tasa           0.8851935   0.9082920 -0.9154169  1.0000000
```

```
# Como soluciones sería mejorar el diseño del experimento o obtener una muestra
# más grande. En este caso no es posible y lo único que podemos hacer es
# eliminar aquella variable que pensamos que está causando el problema (En este
# caso parece que es tasa: dado que es la que menos relación tiene con Y=defectuosos
# pero tiene mucha relación con temperatura y densidad. Nos interesa quitar las
# que tengan menos relación con Y. Además es la que tiene el p-valor más alto, es la menos significativ


qqnorm(res_stu)
qqline(res_stu,col="red")
```



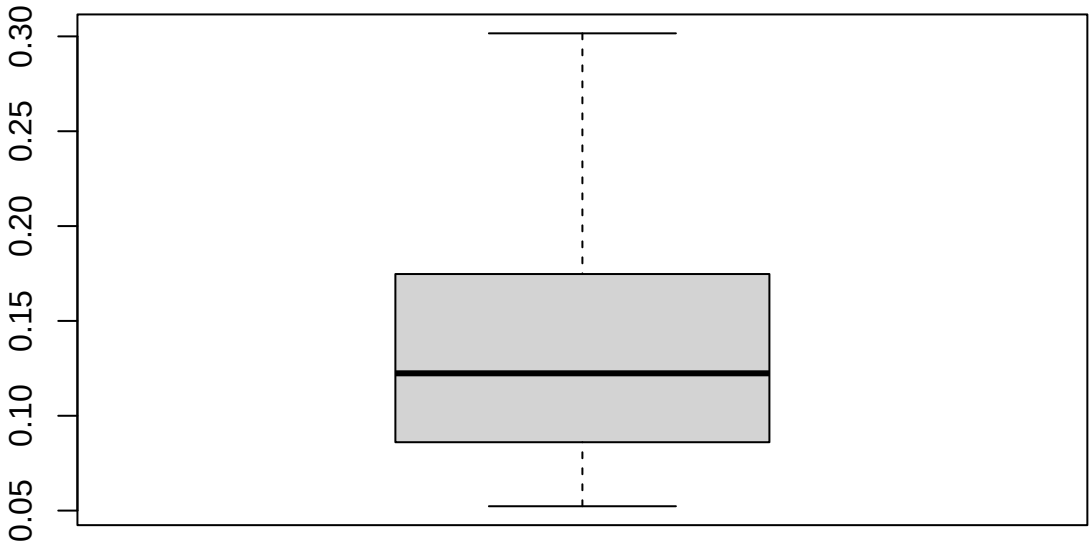

```
shapiro.test(res_stu) #No falla la normalidad.
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res_stu
## W = 0.95986, p-value = 0.3072
```

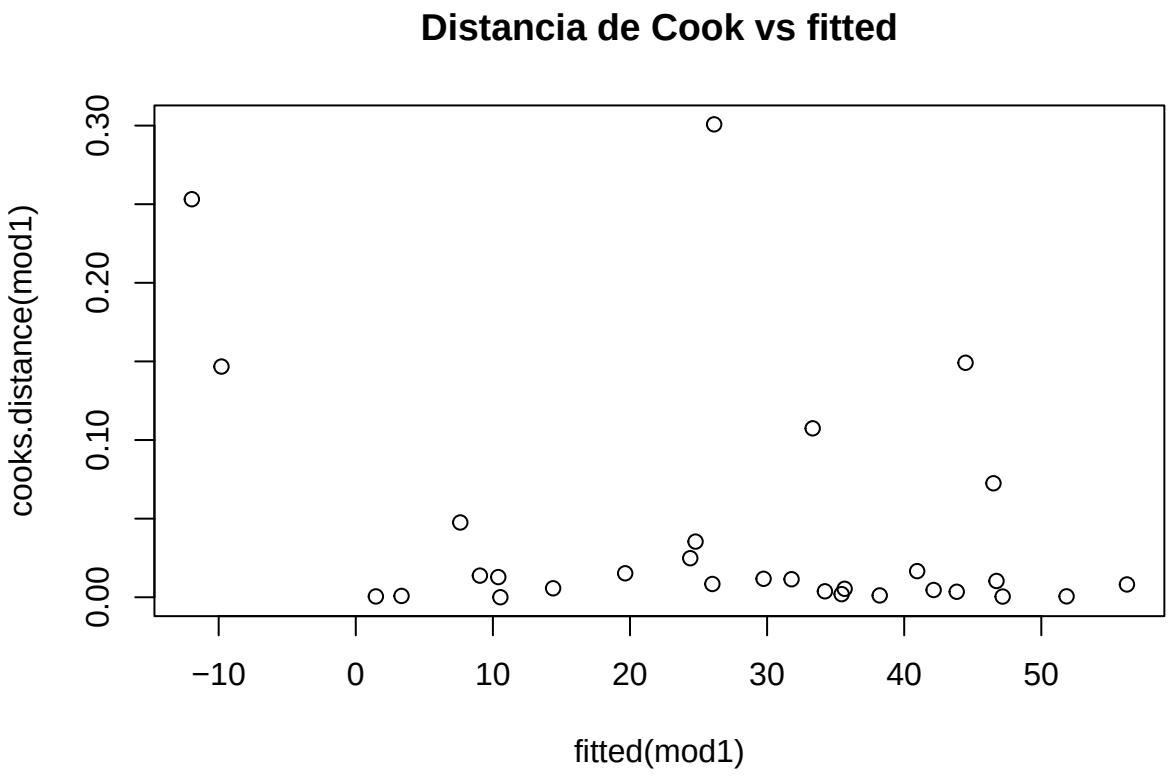
```
defectos[abs(res_stu) > 3,] # No hay outliers
```

```
## [1] caso        temperatura densidad    tasa        defectuosos
## <0 rows> (or 0-length row.names)
```

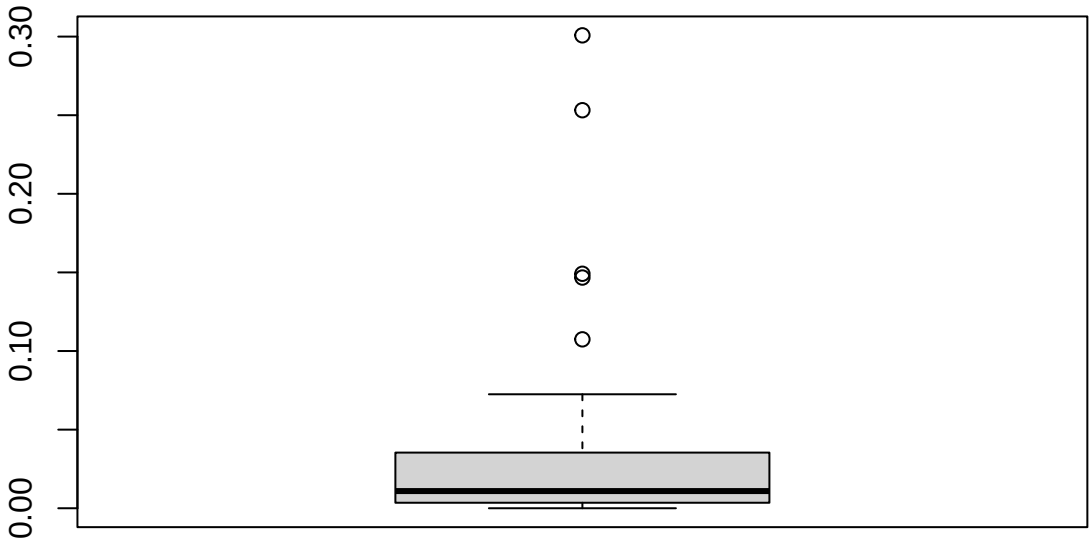
```
  boxplot(hatvalues(mod1))
```



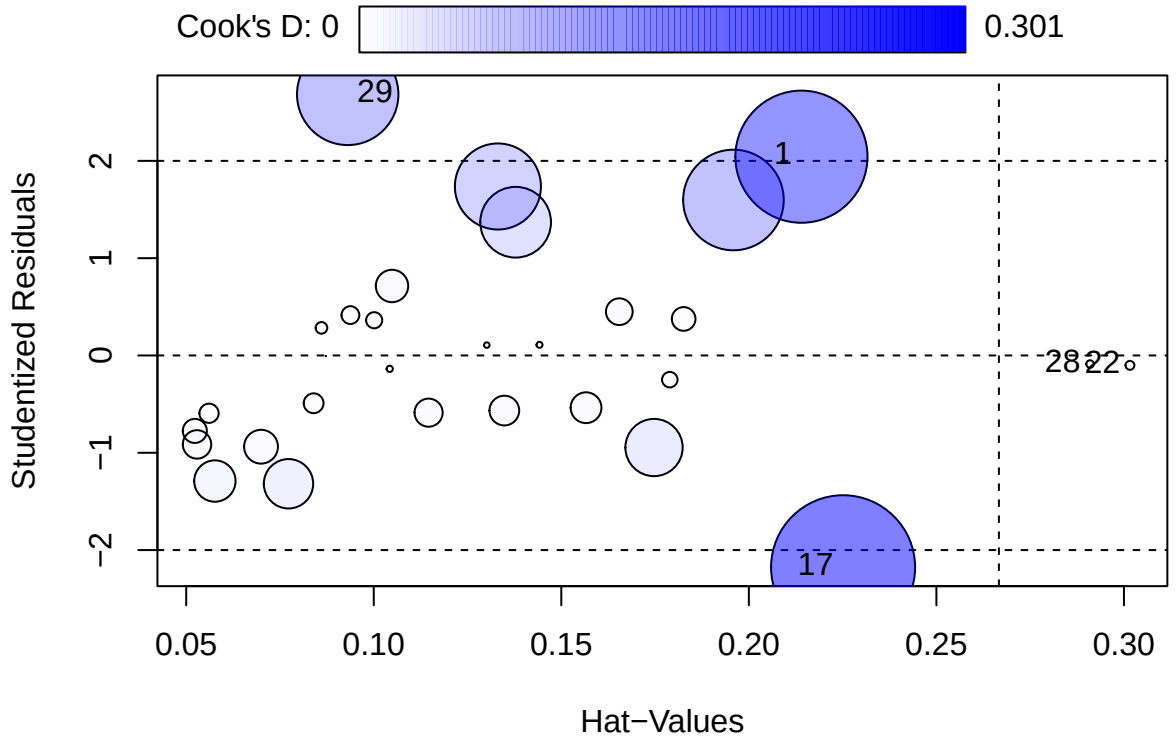

```
 plot(fitted(mod1),cooks.distance(mod1),main="Distancia de Cook vs fitted")
abline(h=0.33,col="red",lwd=1);
```




```
boxplot(cooks.distance(mod1))
```




```
 library(car)
influencePlot(mod1)
```




```
##         StudRes       Hat        CookD
## 1   2.04275197 0.2140211 0.2531696165
## 17 -2.17628088 0.2251284 0.3007865317
## 22 -0.10129576 0.3016046 0.0011516333
## 28 -0.08801137 0.2909679 0.0008262195
## 29  2.68354448 0.0930517 0.1491412636
```

```
  summary(influence.measures(mod1)) # .
```

```
## Potentially influential observations of
##   lm(formula = defectuosos ~ temperatura + densidad + tasa, data = defectos,      na.action = na.excl
##
##    dfb.1_ dfb.tmpr dfb.dnsd dfb.tasa dffit cov.r   cook.d hat
## 17   0.36  -1.03_*  -0.62     0.58    -1.17  0.75    0.30   0.23
## 22   0.05  -0.06    -0.06    -0.01    -0.07  1.67_*  0.00   0.30
## 28   0.04  -0.01    -0.04    -0.04    -0.06  1.65_*  0.00   0.29
## 29  -0.01   0.45     0.09    -0.34     0.86  0.47_*  0.15   0.09
```

```
# Vamos a definir el nuevo modelo quitando la variable tasa
mod0 <- lm(defectuosos ~ temperatura+densidad+tasa, data=defectos, na.action=na.exclude)
summary(mod0)
```

```
##
## Call:
## lm(formula = defectuosos ~ temperatura + densidad + tasa, data = defectos,
##     na.action = na.exclude)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -12.7367  -4.1116  -0.5755   2.7617  16.3279
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.3244    65.9265   0.157   0.8768
## temperatura  16.0779     8.2941   1.938   0.0635 .
## densidad     -1.8273     1.4971  -1.221   0.2332
## tasa          0.1167     0.1306   0.894   0.3797
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.11 on 26 degrees of freedom
## Multiple R-squared:  0.8797, Adjusted R-squared:  0.8658
## F-statistic: 63.36 on 3 and 26 DF,  p-value: 4.371e-12
```

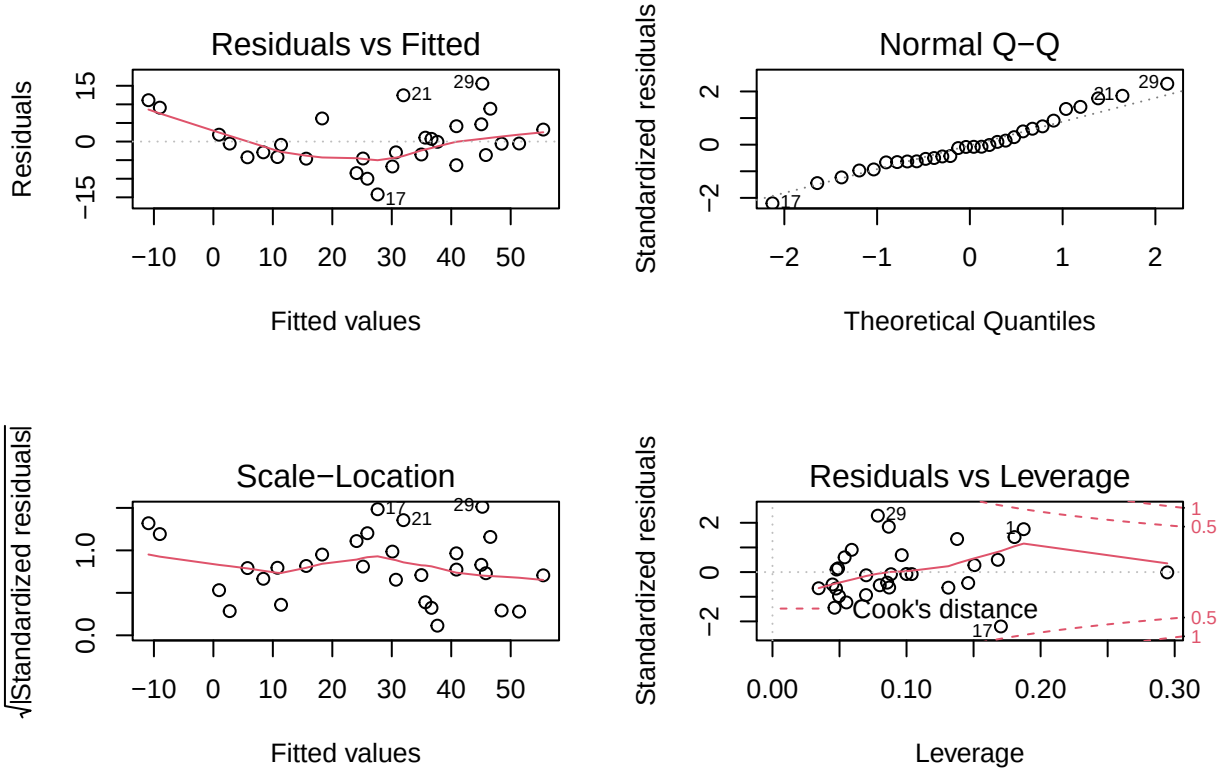
```
mod1 <- lm(defectuosos ~ temperatura+densidad, data=defectos, na.action=na.exclude)
summary(mod1)
```

```
##
## Call:
## lm(formula = defectuosos ~ temperatura + densidad, data = defectos,
##     na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.2233  -4.2245  -0.5605   3.8984  15.5638
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   46.238     52.066   0.888   0.3823
## temperatura   18.050      7.965   2.266   0.0317 *
## densidad      -2.327      1.383  -1.683   0.1040
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.084 on 27 degrees of freedom
## Multiple R-squared:  0.876,  Adjusted R-squared:  0.8668
## F-statistic: 95.35 on 2 and 27 DF,  p-value: 5.784e-13
```

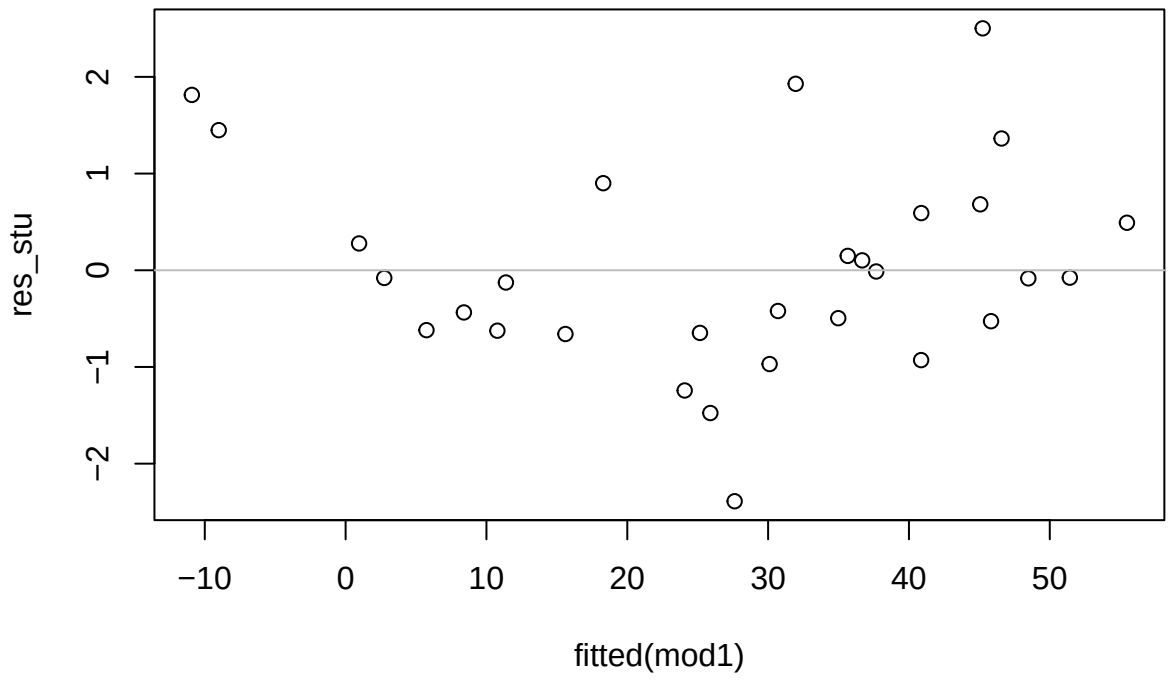
```
anova(mod0,mod1) # La variable tasa no es significativa
```

```
## Analysis of Variance Table
##
## Model 1: defectuosos ~ temperatura + densidad + tasa
## Model 2: defectuosos ~ temperatura + densidad
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     26 1314.4
## 2     27 1354.8 -1   -40.372 0.7986 0.3797
```

```
# Observamos que el modelo es significativo, aunque por separado la temperatura sí que lo es pero la de

# Vamos a diagnosticarlo

par(mfrow=c(2,2))
plot(mod1)
```

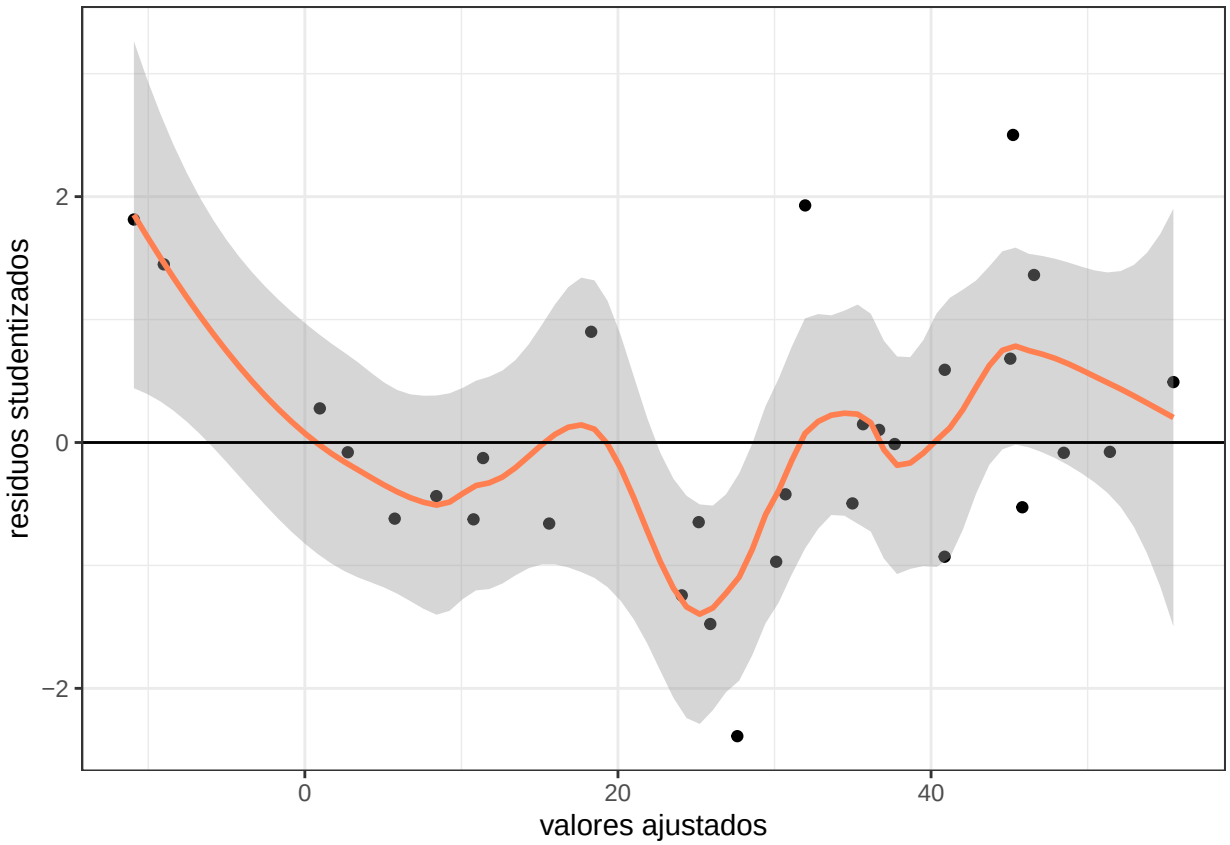



```
res_stu<-rstudent(mod1)
par(mfrow=c(1,1))
plot(fitted(mod1),res_stu)
 abline(h=0, col="gray")
```

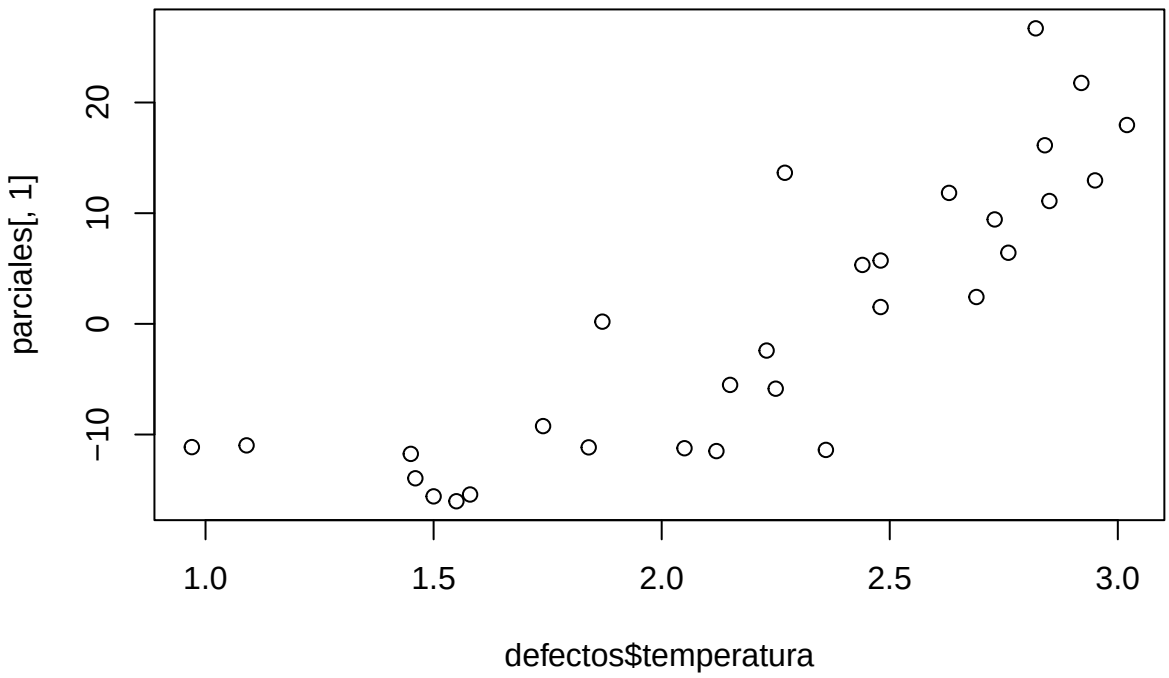



```
library(ggplot2)
 library(gridExtra)
 ggplot(data = defectos, aes(x =fitted(mod1), y = res_stu)) +
 geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
 labs(y = "residuos studentizados",
      x = "valores ajustados") +
 theme_bw()
```

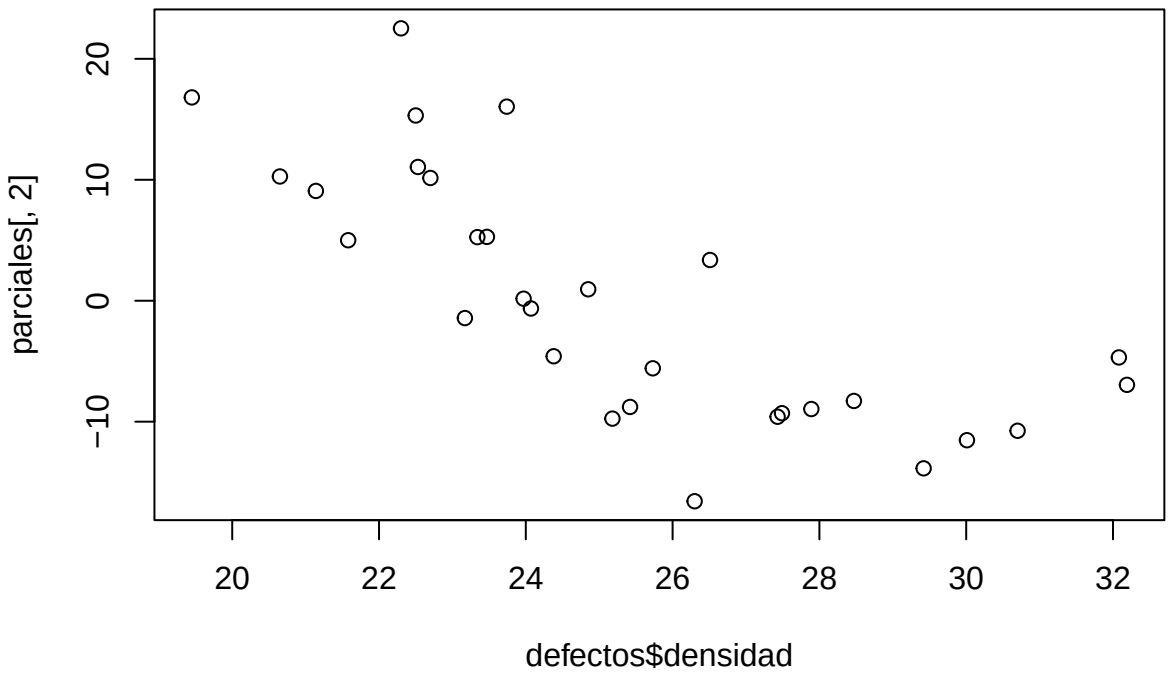
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
  parciales <- residuals(mod1,type="partial")
plot(defectos$temperatura,parciales[,1])
```

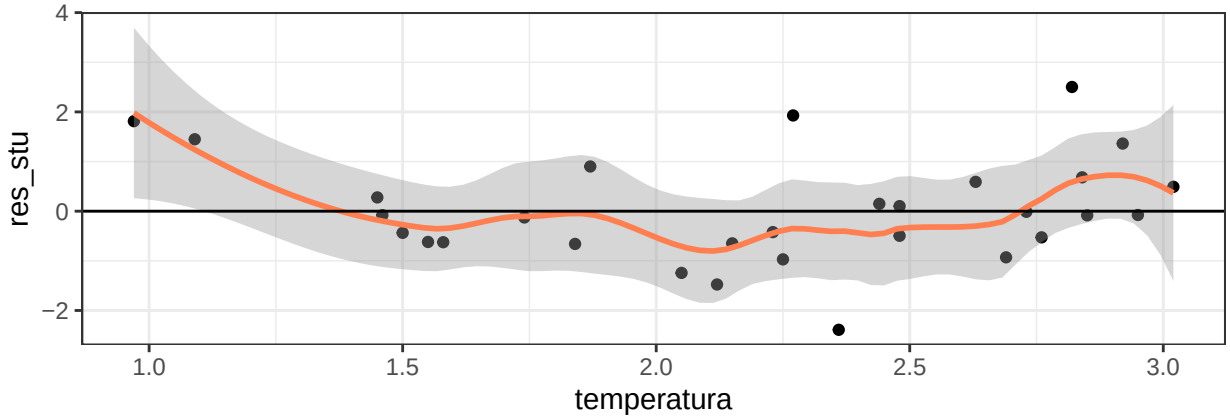


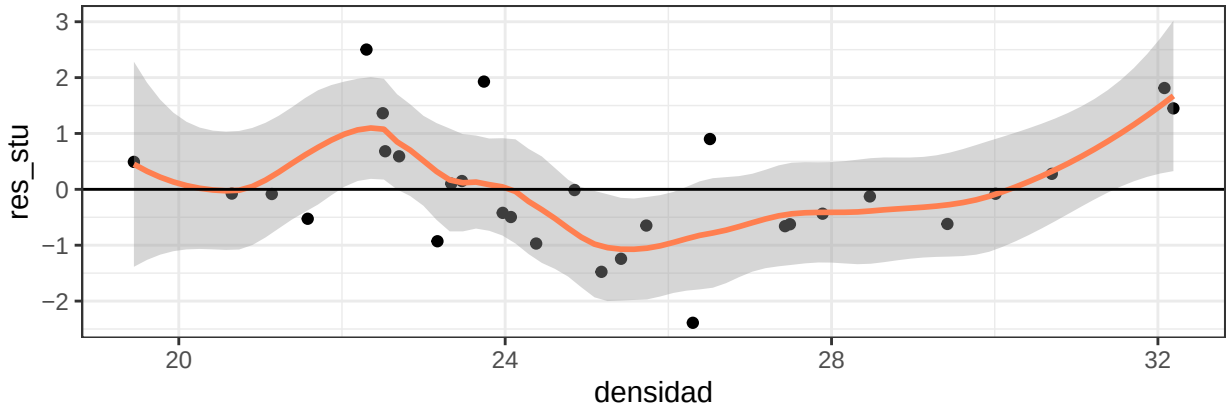

```
plot(defectos$densidad,parciales[,2])
```




```
  plot1 <- ggplot(data = defectos, aes(temperatura, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  plot2 <- ggplot(data = defectos, aes(densidad, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  grid.arrange(plot1, plot2)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```





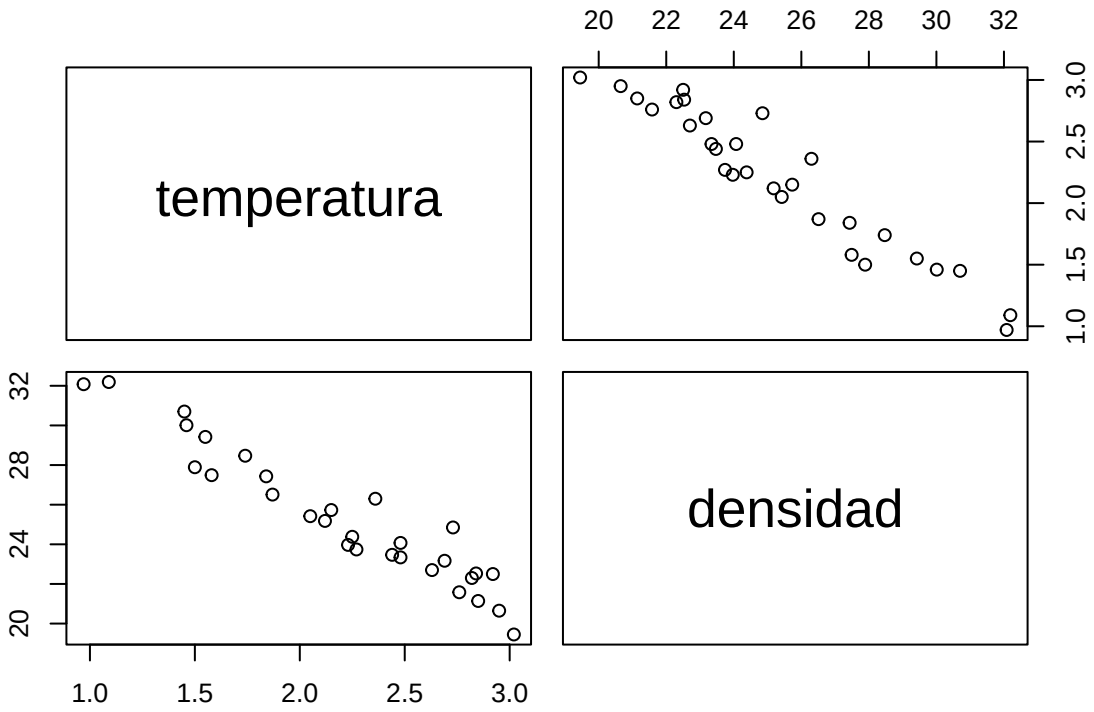

```
  #Parece que falla un poquito más la linealidad, pero no se ve claramente la forma

  library(lmtest)
    bptest(mod1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mod1
## BP = 0.81142, df = 2, p-value = 0.6665
```

```
    # No hay problema con la homocedasticidad.

library(ISLR)
vif(mod1)  # Siguen habiendo problema con la multicolinealidad (lo esperábamos!)
```

```
## temperatura    densidad
##    12.48079    12.48079
```

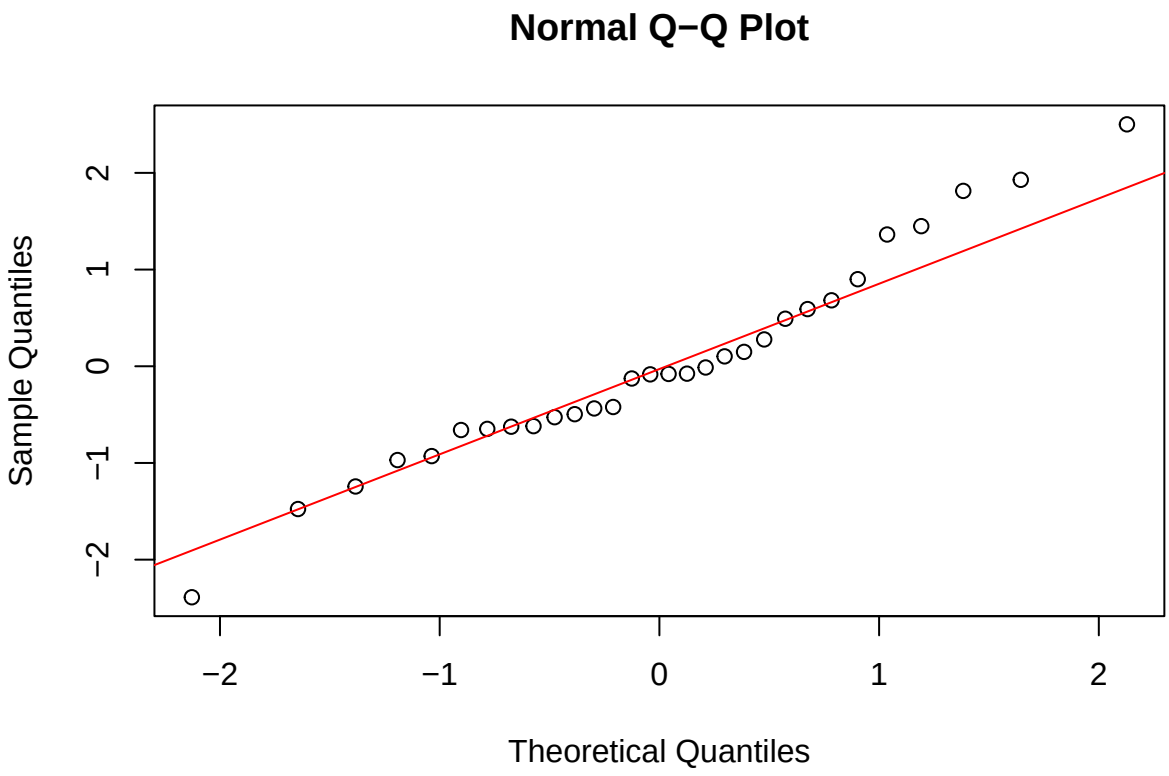
```
pairs(defectos[,c('temperatura','densidad')])
```




```
cor(defectos[,c('defectuosos','temperatura','densidad')])
```

```
##              defectuosos temperatura   densidad
## defectuosos    1.0000000   0.9289640 -0.9232497
## temperatura    0.9289640   1.0000000 -0.9591021
## densidad      -0.9232497  -0.9591021  1.0000000
```

```
# Como soluciones sería mejorar el diseño del experimento o obtener una muestra
# más grande. En este caso no es posible y lo único que podemos hacer es
# eliminar aquella variable que pensamos que está causando el problema: densidad.


qqnorm(res_stu)
qqline(res_stu,col="red")
```



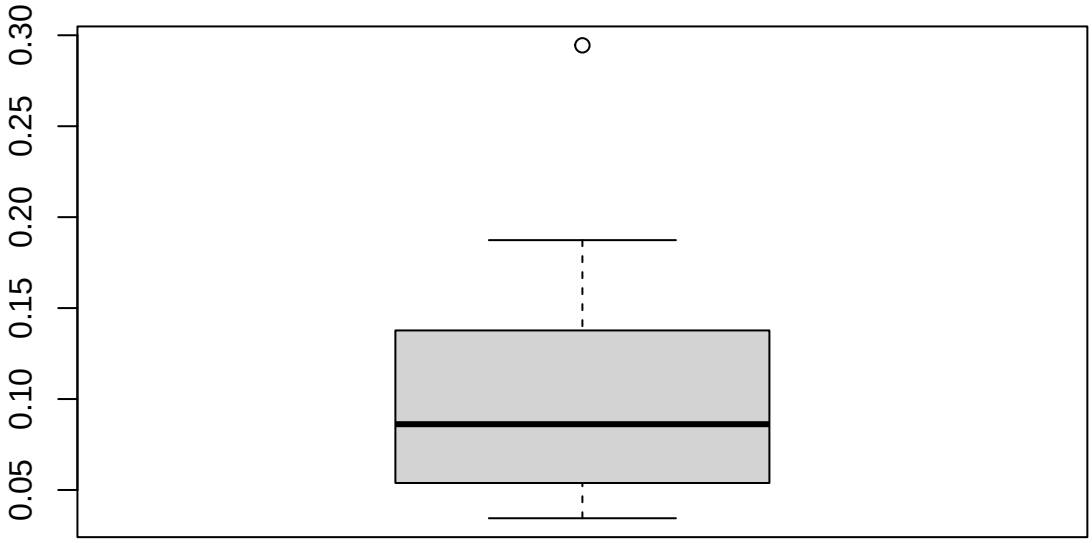

```
shapiro.test(res_stu) #No falla la normalidad.
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res_stu
## W = 0.96968, p-value = 0.5304
```

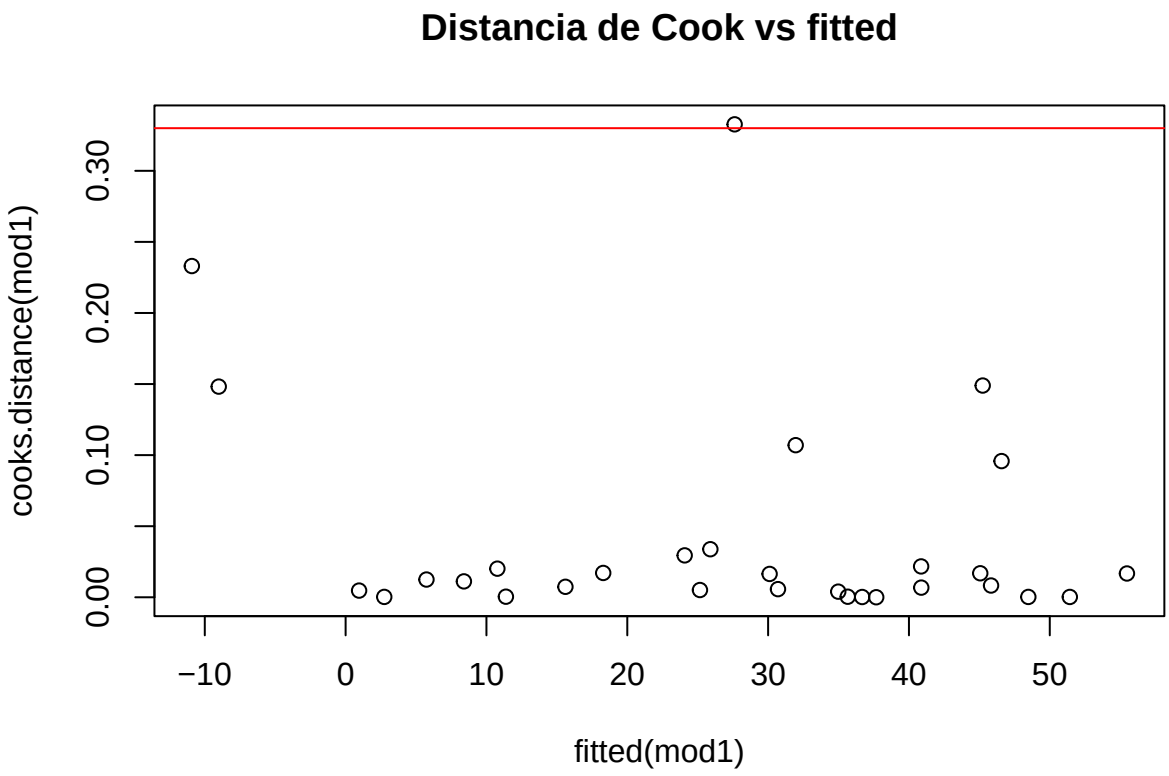
```
defectos[abs(res_stu) > 3,] # No hay outliers
```

```
## [1] caso        temperatura densidad    tasa        defectuosos
## <0 rows> (or 0-length row.names)
```

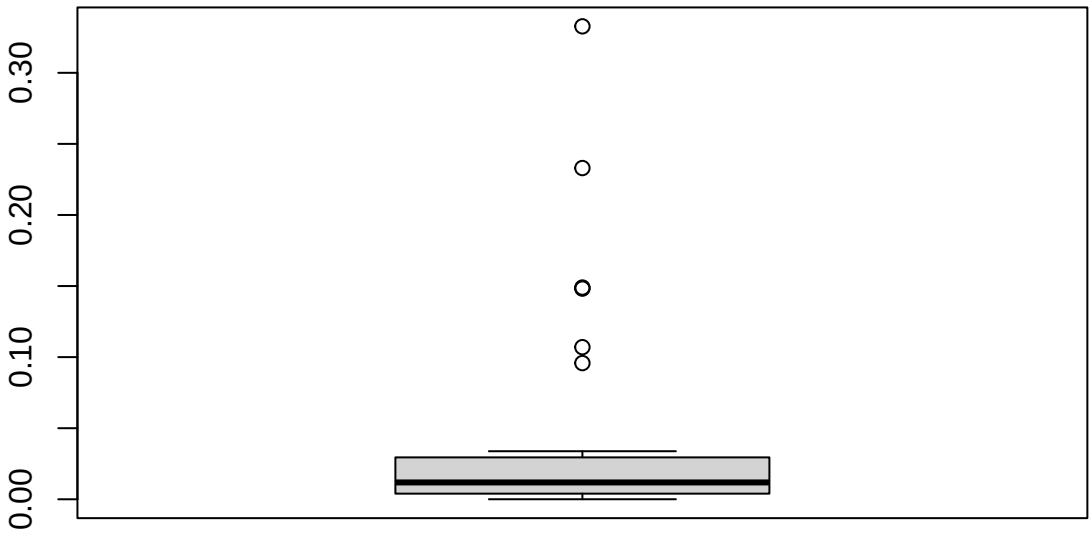
```
  boxplot(hatvalues(mod1))
```



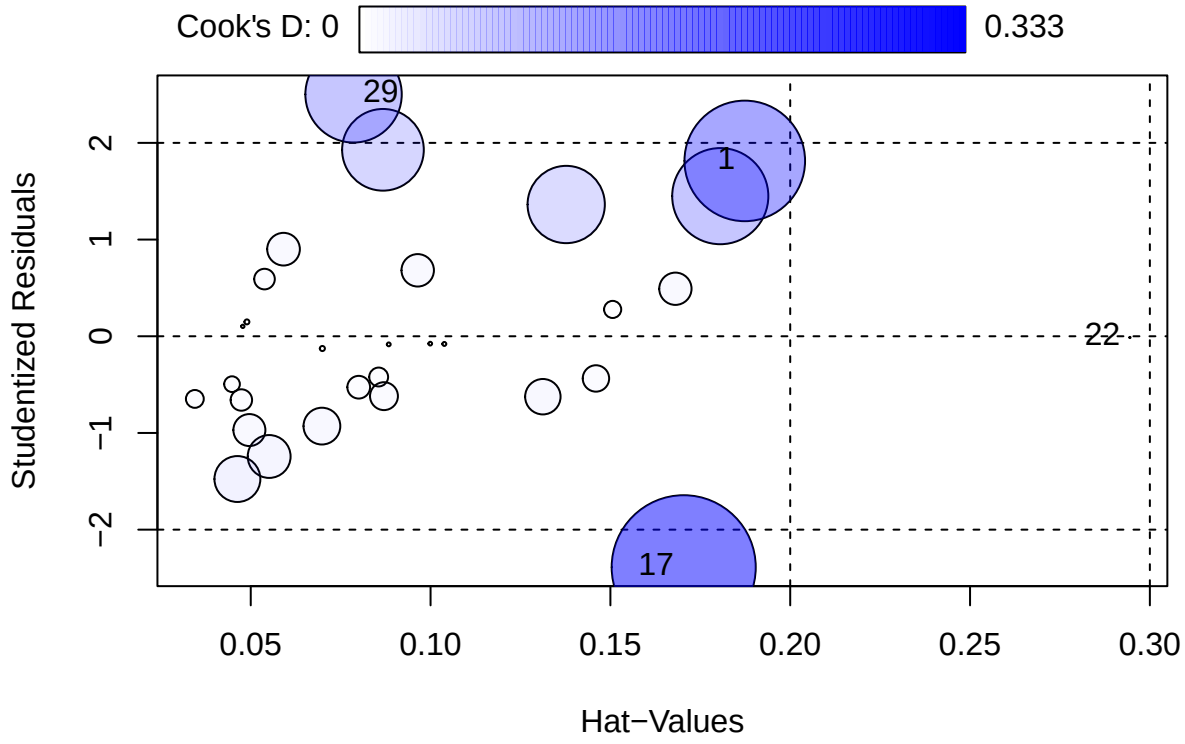

```
 plot(fitted(mod1),cooks.distance(mod1),main="Distancia de Cook vs fitted")
abline(h=0.33,col="red",lwd=1);
```




```
boxplot(cooks.distance(mod1))
```




```
 library(car)
influencePlot(mod1)
```




```
##         StudRes         Hat        CookD
## 1    1.81353806 0.18736413 2.330145e-01
## 17 -2.38892049 0.17038447 3.326938e-01
## 22 -0.01263928 0.29446027 2.307896e-05
## 29  2.50197214 0.07858884 1.489540e-01
```

```
  summary(influence.measures(mod1)) #tenemos valores influyentes, pero menos que antes.
```

```
## Potentially influential observations of
##   lm(formula = defectuosos ~ temperatura + densidad, data = defectos,      na.action = na.exclude) :
##
##    dfb.1_ dfb.tmpr dfb.dnsd dffit   cov.r   cook.d hat
## 17  0.96  -0.96    -0.96    -1.08_*  0.74    0.33   0.17
## 22  0.01  -0.01    -0.01    -0.01    1.59_*  0.00   0.29
## 29 -0.25   0.35     0.21     0.73    0.64_*  0.15   0.08
```

```
# Vamos a definir el nuevo modelo quitando la variable densidad
mod0 <- lm(defectuosos ~ temperatura+densidad, data=defectos, na.action=na.exclude)
summary(mod0)
```

```
##
## Call:
## lm(formula = defectuosos ~ temperatura + densidad, data = defectos,
##     na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.2233  -4.2245  -0.5605   3.8984  15.5638
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   46.238     52.066   0.888   0.3823
## temperatura   18.050      7.965   2.266   0.0317 *
## densidad      -2.327      1.383  -1.683   0.1040
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.084 on 27 degrees of freedom
## Multiple R-squared:  0.876,  Adjusted R-squared:  0.8668
## F-statistic: 95.35 on 2 and 27 DF,  p-value: 5.784e-13
```

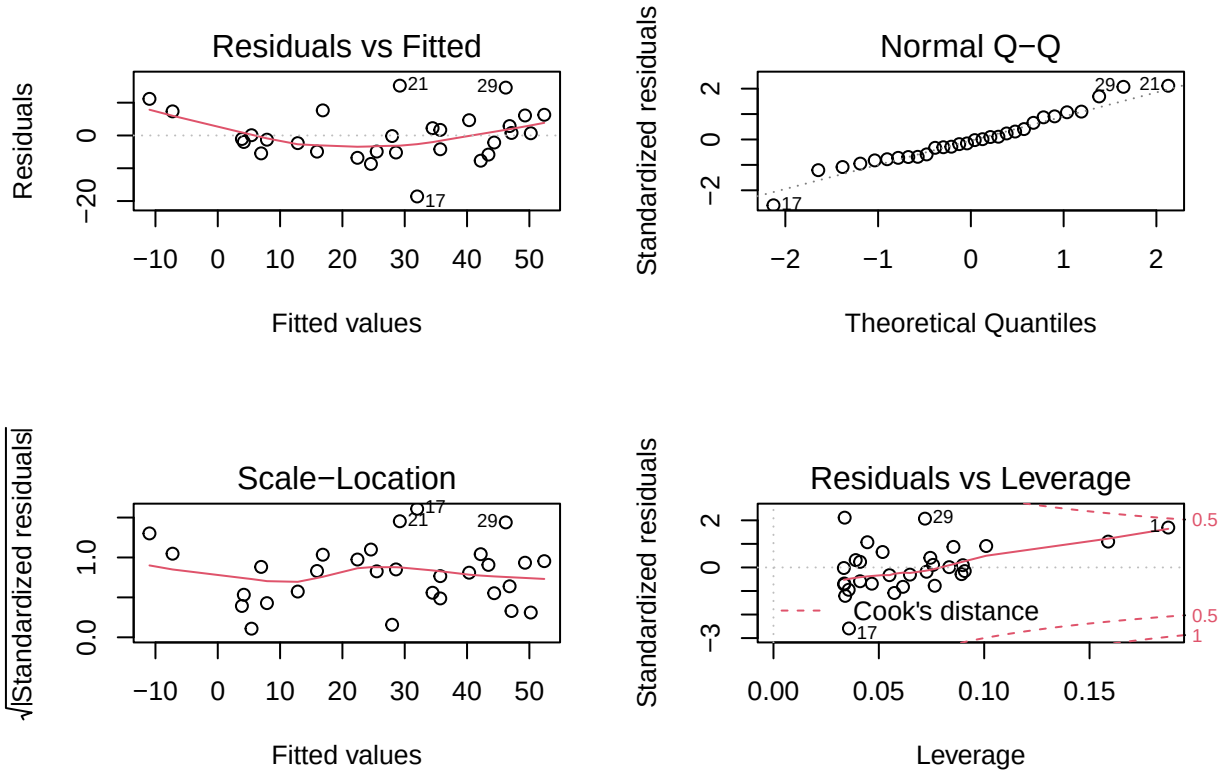
```
mod1 <- lm(defectuosos ~ temperatura, data=defectos, na.action=na.exclude)
summary(mod1)
```

```
##
## Call:
## lm(formula = defectuosos ~ temperatura, data = defectos, na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.5952  -4.9203  -0.6253   4.2133  15.1861
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -40.938      5.298  -7.727 2.04e-08 ***
## temperatura   30.904      2.327  13.279 1.32e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.312 on 28 degrees of freedom
## Multiple R-squared:  0.863,  Adjusted R-squared:  0.8581
## F-statistic: 176.3 on 1 and 28 DF,  p-value: 1.317e-13
```

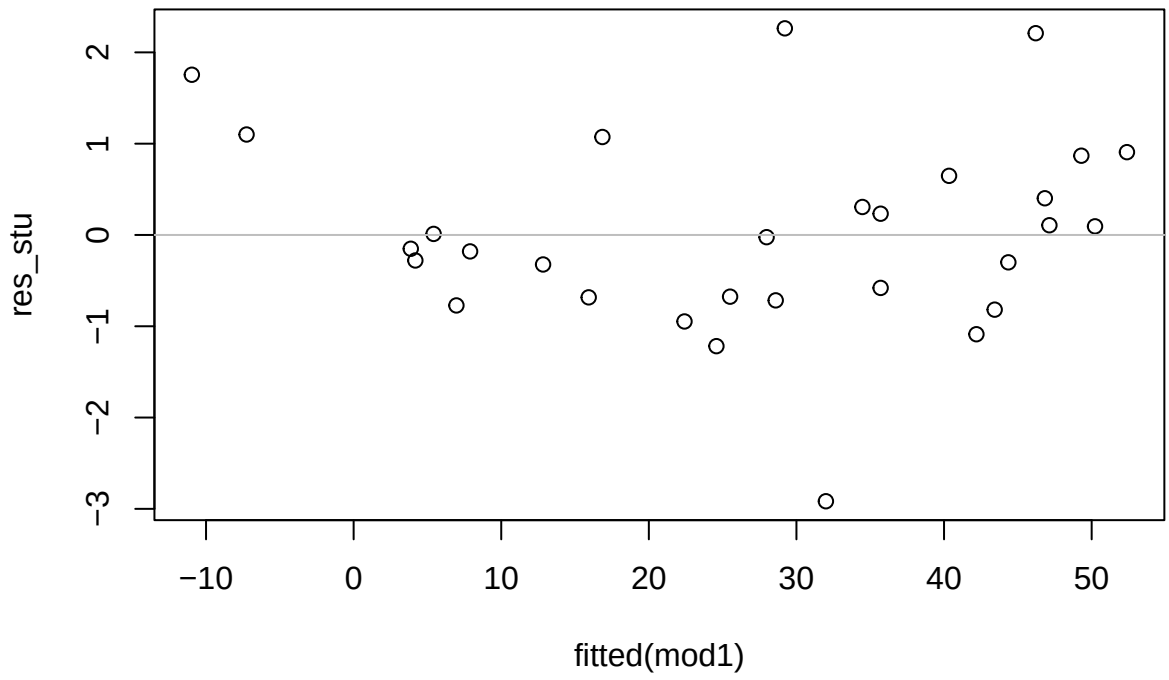
```
anova(mod0,mod1) # La variable densidad no es significativa
```

```
## Analysis of Variance Table
##
## Model 1: defectuosos ~ temperatura + densidad
## Model 2: defectuosos ~ temperatura
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     27 1354.8
## 2     28 1496.8 -1   -142.05 2.8309  0.104
```

```
# Observamos que el modelo es significativo y la temperatura también (esto era esperable y lo que habrí

# Vamos a diagnosticarlo

par(mfrow=c(2,2))
plot(mod1)
```

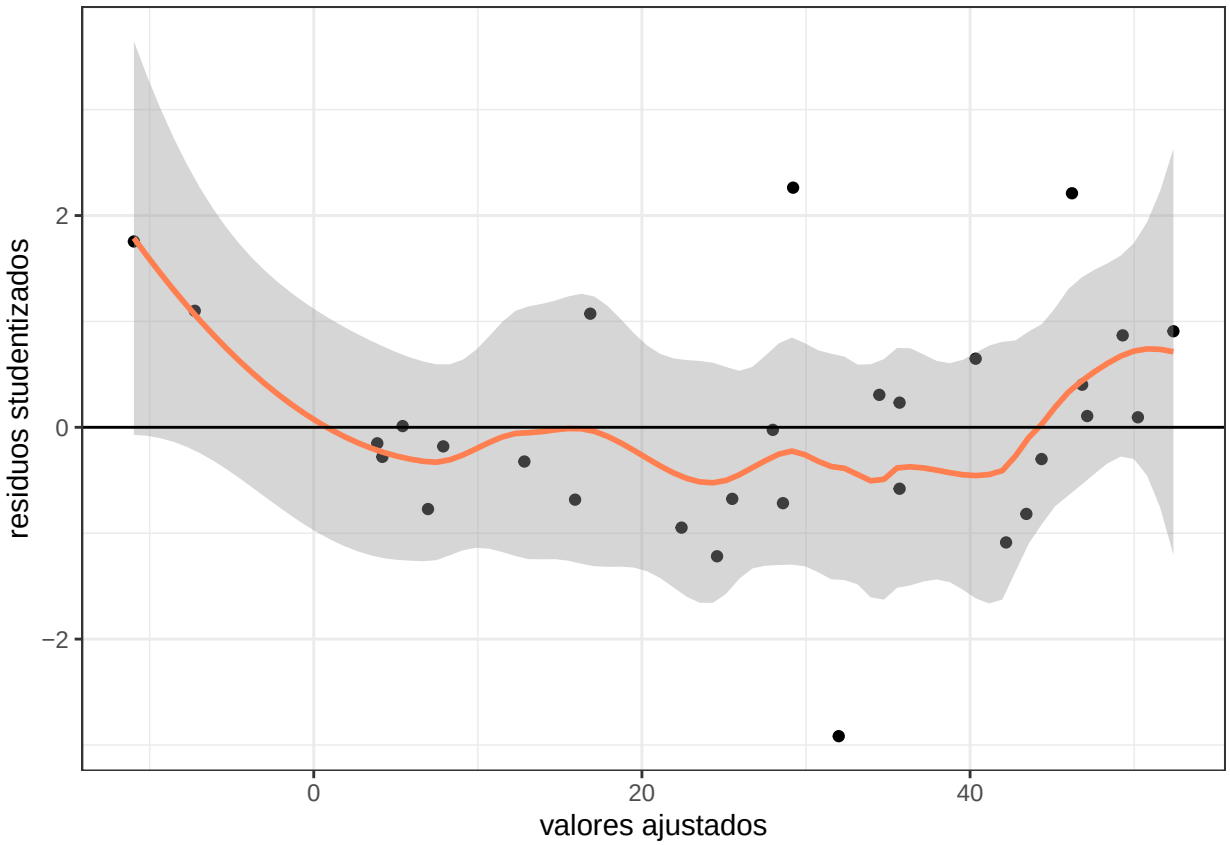



```
res_stu<-rstudent(mod1)
par(mfrow=c(1,1))
plot(fitted(mod1),res_stu)
 abline(h=0, col="gray")
```



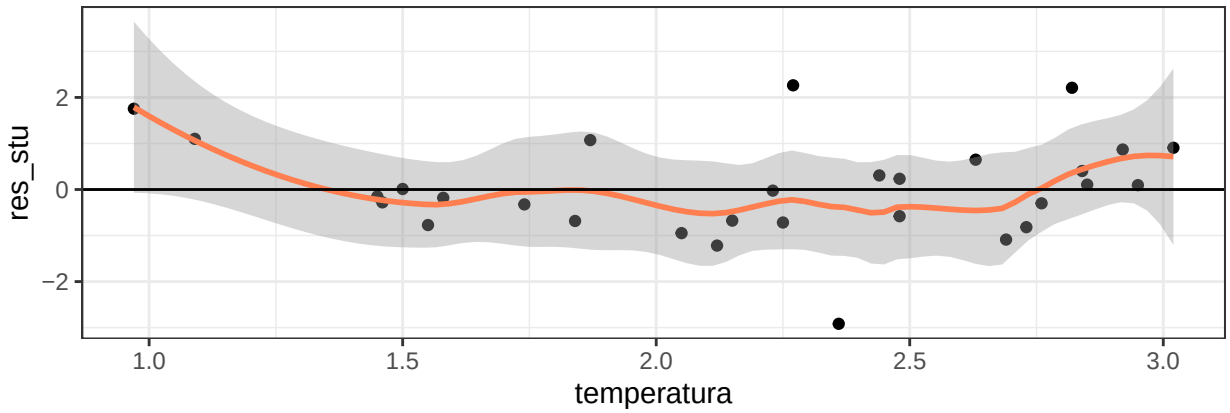

```
library(ggplot2)
 library(gridExtra)
 ggplot(data = defectos, aes(x =fitted(mod1), y = res_stu)) +
 geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
 labs(y = "residuos studentizados",
      x = "valores ajustados") +
 theme_bw()
```

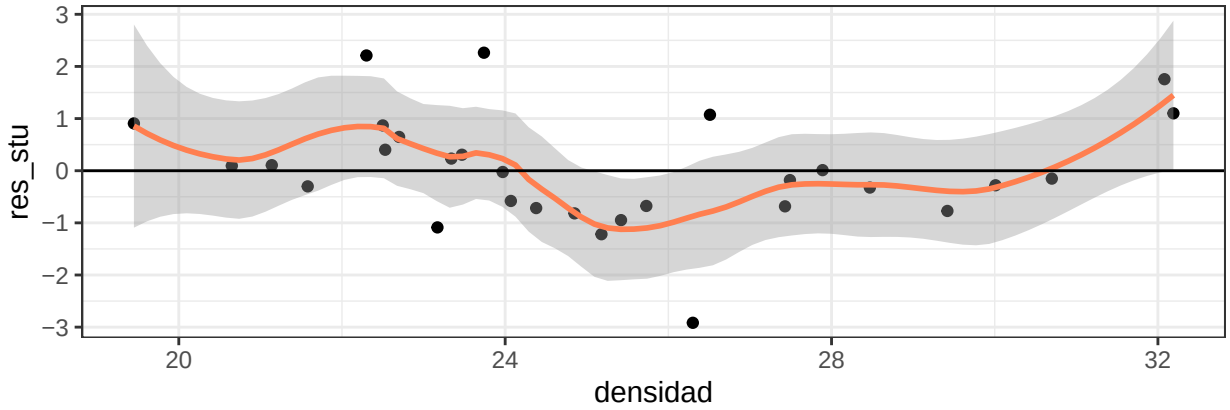
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
  plot1 <- ggplot(data = defectos, aes(temperatura, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  plot2 <- ggplot(data = defectos, aes(densidad, res_stu)) +
  geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
  theme_bw()
  grid.arrange(plot1, plot2)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

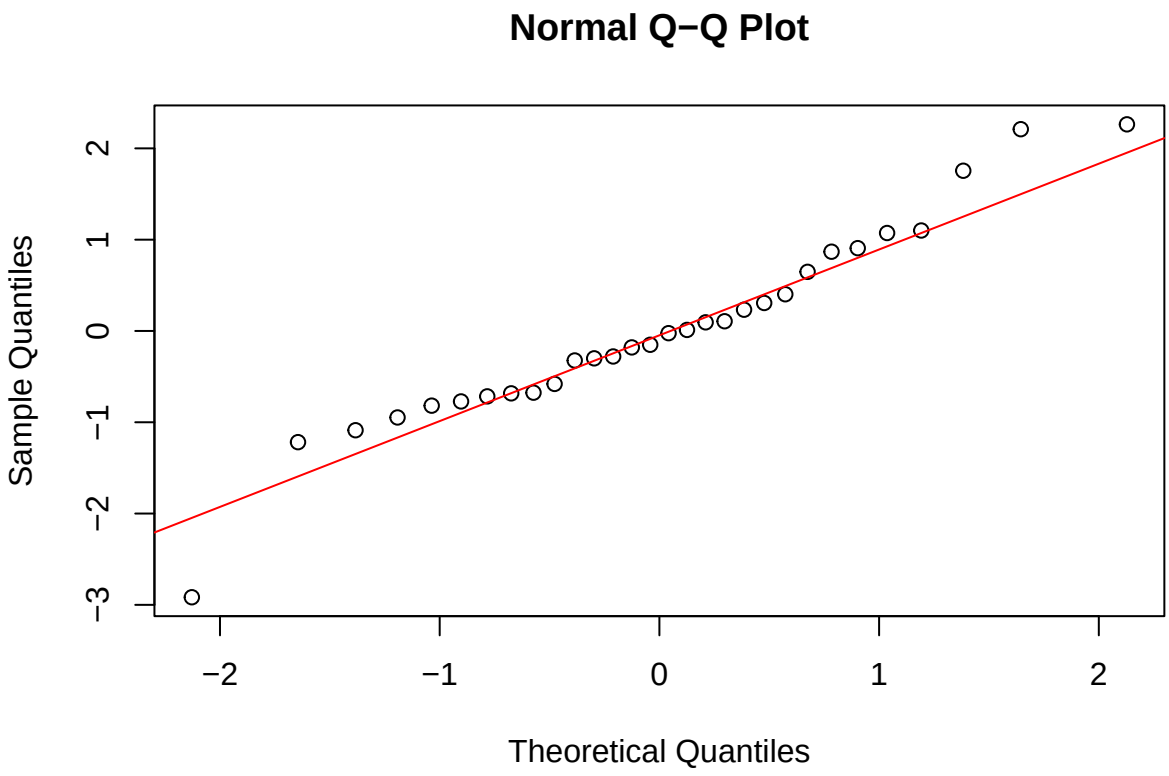





```
  #No veo un gran problema con la linealidad

  library(lmtest)
    bptest(mod1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mod1
## BP = 0.040278, df = 1, p-value = 0.8409
```

```
    # No hay problema con la homocedasticidad.

#ahora ya no tiene sentido ver la multicolineadidad, dado que solo hay una variable.


qqnorm(res_stu)
qqline(res_stu,col="red")
```



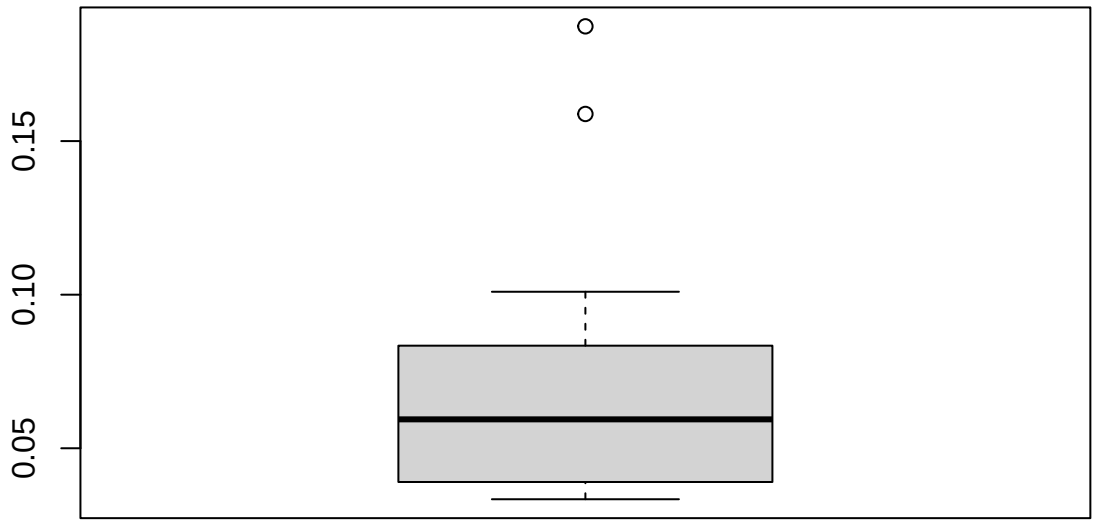

```
shapiro.test(res_stu) #No falla la normalidad.
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res_stu
## W = 0.9567, p-value = 0.2546
```

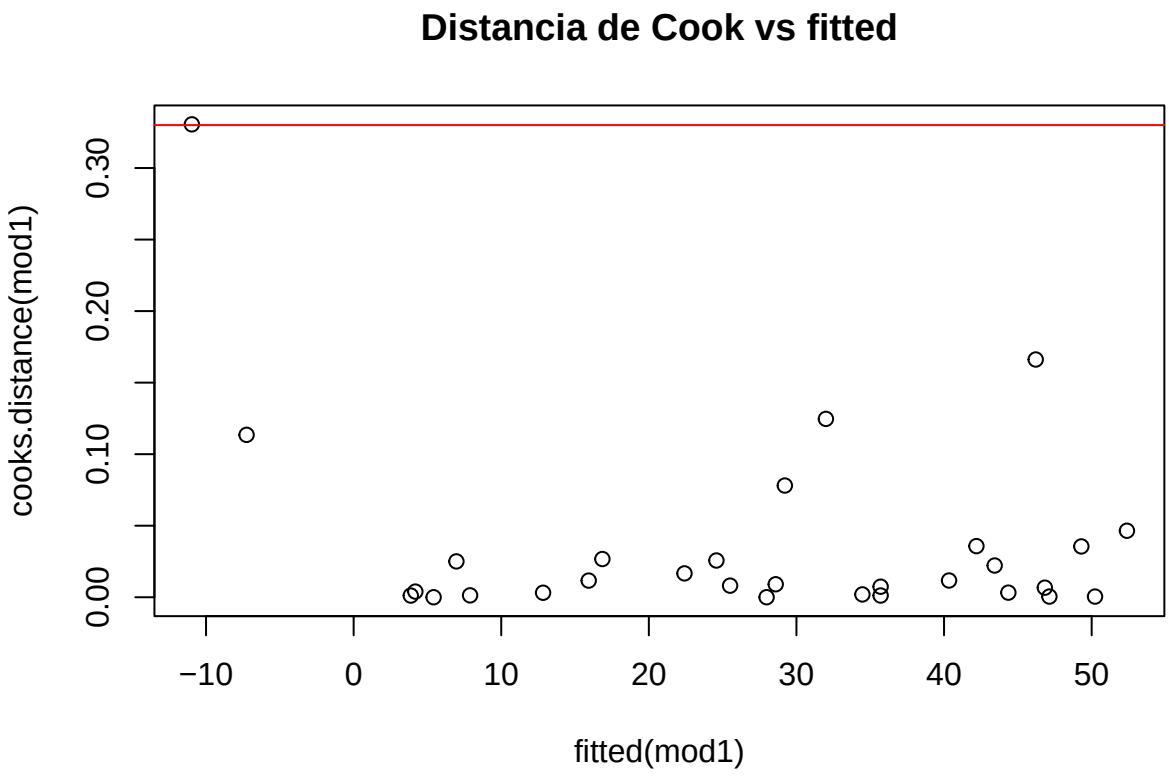
```
defectos[abs(res_stu) > 3,] # No hay outliers
```

```
## [1] caso        temperatura densidad    tasa        defectuosos
## <0 rows> (or 0-length row.names)
```

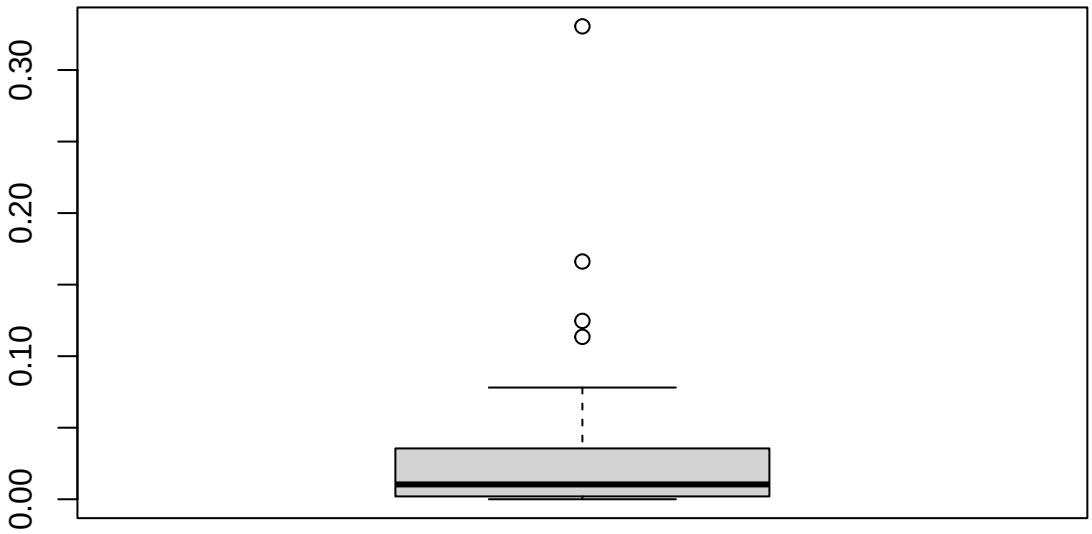
```
  boxplot(hatvalues(mod1))
```



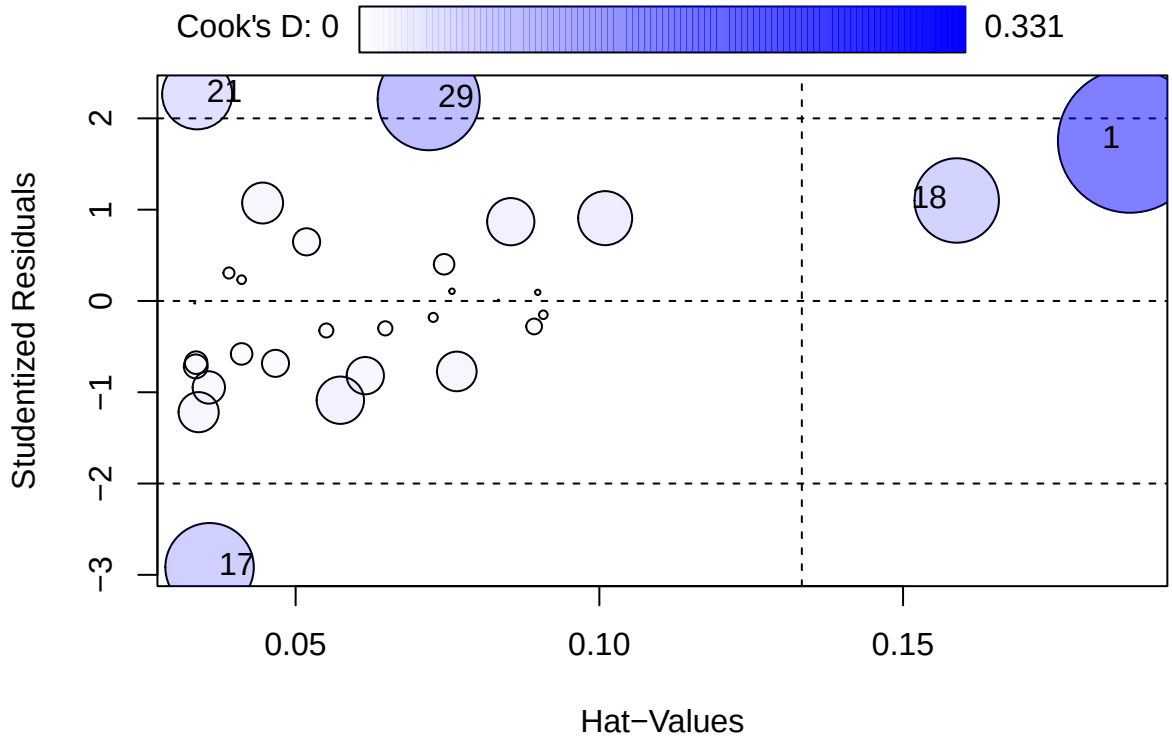

```
 plot(fitted(mod1),cooks.distance(mod1),main="Distancia de Cook vs fitted")
abline(h=0.33,col="red",lwd=1);
```




```
boxplot(cooks.distance(mod1))
```




```
 library(car)
influencePlot(mod1)
```




```
##      StudRes        Hat      CookD
## 1   1.755095 0.18735169 0.33052253
## 17 -2.916708 0.03583049 0.12465135
## 18  1.100559 0.15883129 0.11349740
## 21  2.263199 0.03378811 0.07806591
## 29  2.210216 0.07190040 0.16616792
```

```
  summary(influence.measures(mod1)) # los puntos 1 y 17 aparecen como influyentes, pero realmente el 17
```

```
## Potentially influential observations of
##   lm(formula = defectuosos ~ temperatura, data = defectos, na.action = na.exclude) :
##
##    dfb.1_ dfb.tmpr dffit   cov.r   cook.d hat
## 1    0.83  -0.76     0.84_*  1.07    0.33   0.19
## 17   0.01  -0.15    -0.56    0.64_*  0.12   0.04
```

```
# Realmente únicamente hemos visto problemas con la linealidad. Vamos a quitar el influyente y ver que
mod0 <- lm(defectuosos ~ temperatura, data=defectos, na.action=na.exclude)
summary(mod0)
```

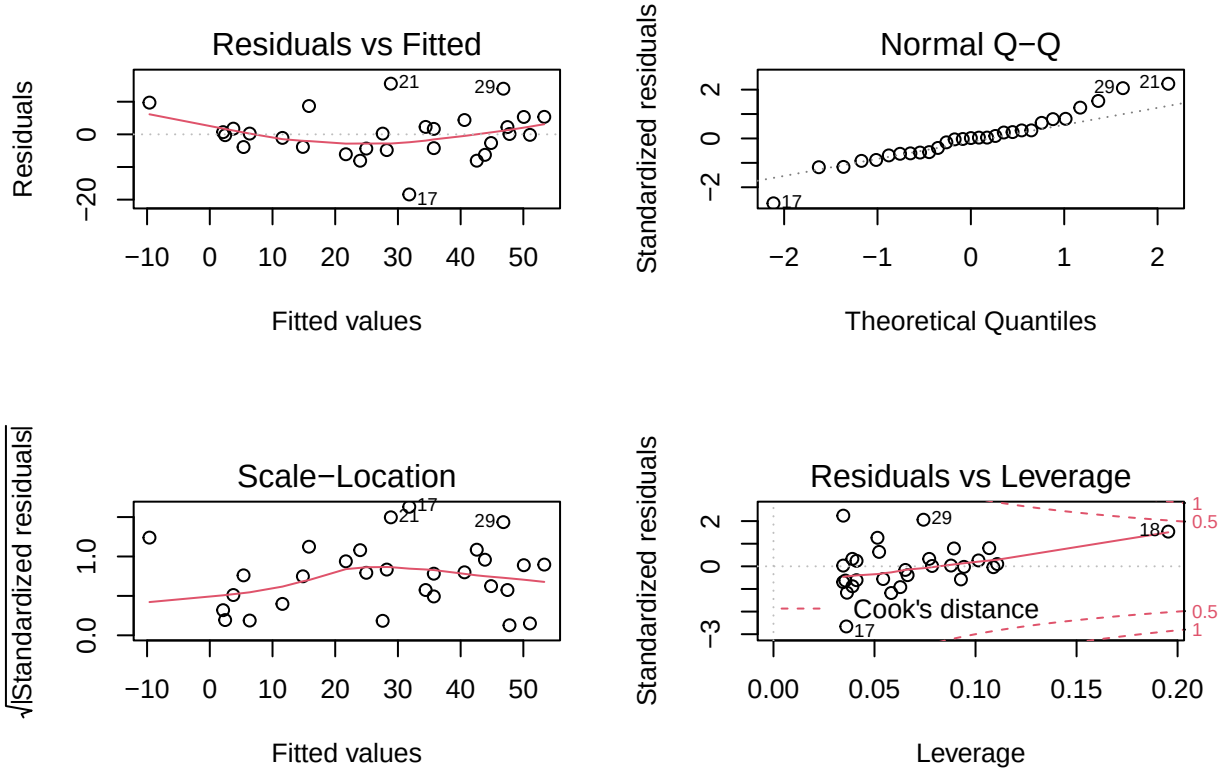
```
##
## Call:
## lm(formula = defectuosos ~ temperatura, data = defectos, na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.5952  -4.9203  -0.6253   4.2133  15.1861
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -40.938      5.298  -7.727 2.04e-08 ***
## temperatura   30.904      2.327  13.279 1.32e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.312 on 28 degrees of freedom
## Multiple R-squared:  0.863,  Adjusted R-squared:  0.8581
## F-statistic: 176.3 on 1 and 28 DF,  p-value: 1.317e-13
```

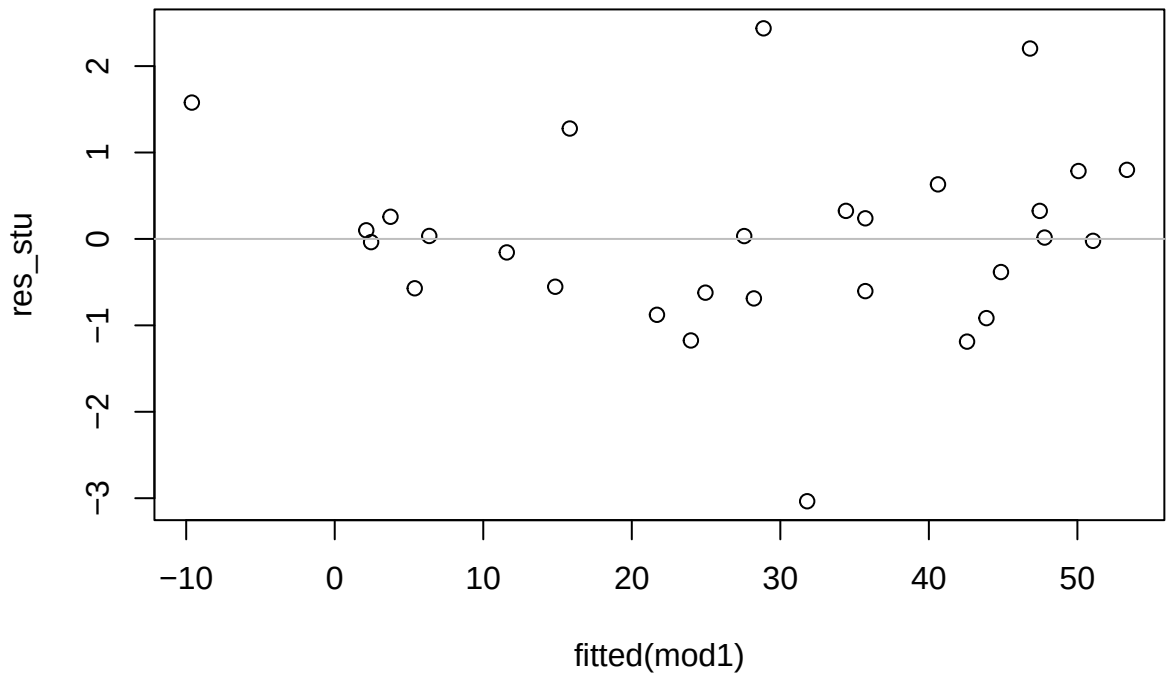
```
defectos2<-defectos[-1,]
mod1 <- lm(defectuosos ~ temperatura, data=defectos2, na.action=na.exclude)
summary(mod1)
```

```
##
## Call:
## lm(formula = defectuosos ~ temperatura, data = defectos2, na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.4068  -4.2211   0.1098   2.2837  15.5290
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -45.175      5.653  -7.992 1.37e-08 ***
## temperatura   32.619      2.449  13.320 2.20e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.054 on 27 degrees of freedom
## Multiple R-squared:  0.8679, Adjusted R-squared:  0.863
## F-statistic: 177.4 on 1 and 27 DF,  p-value: 2.197e-13
```

```
# Podemos observar como varían los coeficientes del modelo. Pero no se pueden comparar con los criterio

# Vamos a diagnosticarlo

par(mfrow=c(2,2))
plot(mod1)
```

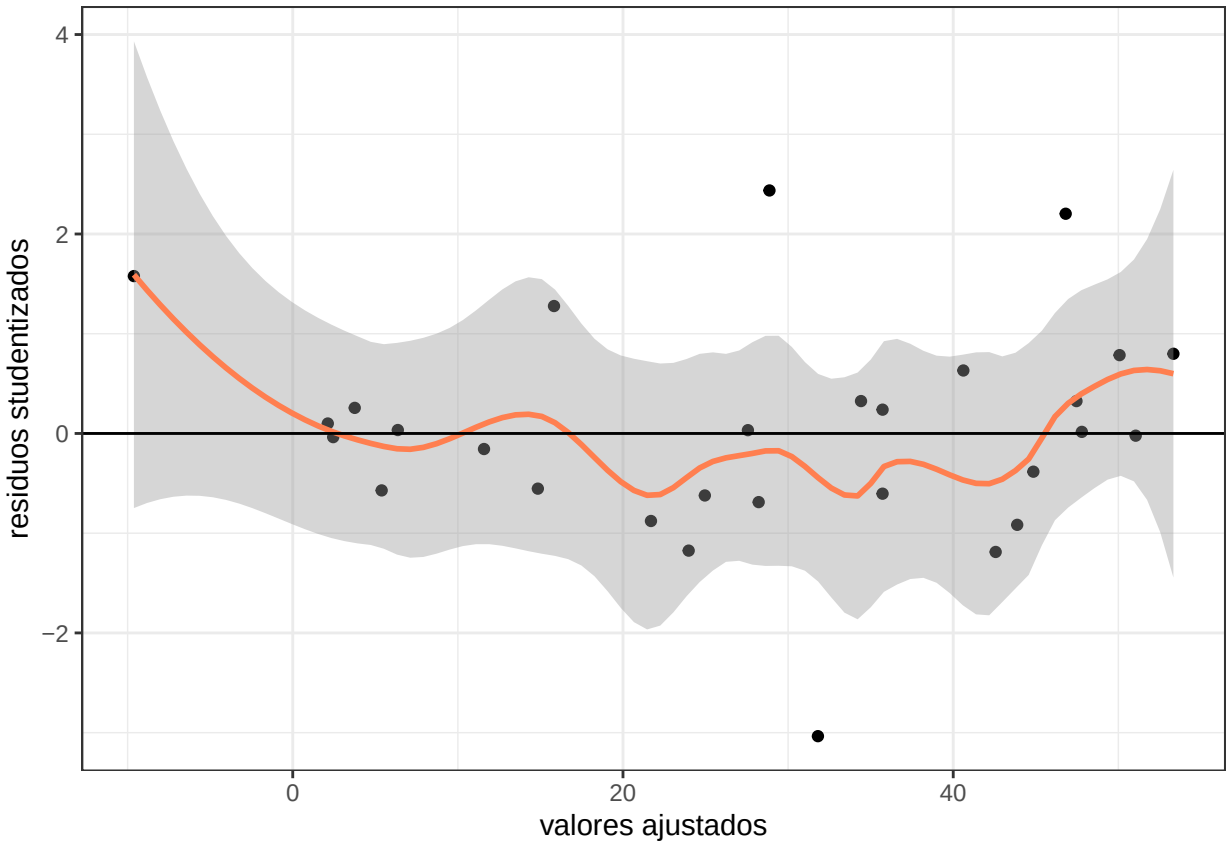



```
res_stu<-rstudent(mod1)
par(mfrow=c(1,1))
plot(fitted(mod1),res_stu)
 abline(h=0, col="gray")
```




```
library(ggplot2)
 library(gridExtra)
 ggplot(data = defectos2, aes(x =fitted(mod1), y = res_stu)) +
 geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
 labs(y = "residuos studentizados",
      x = "valores ajustados") +
 theme_bw()
```

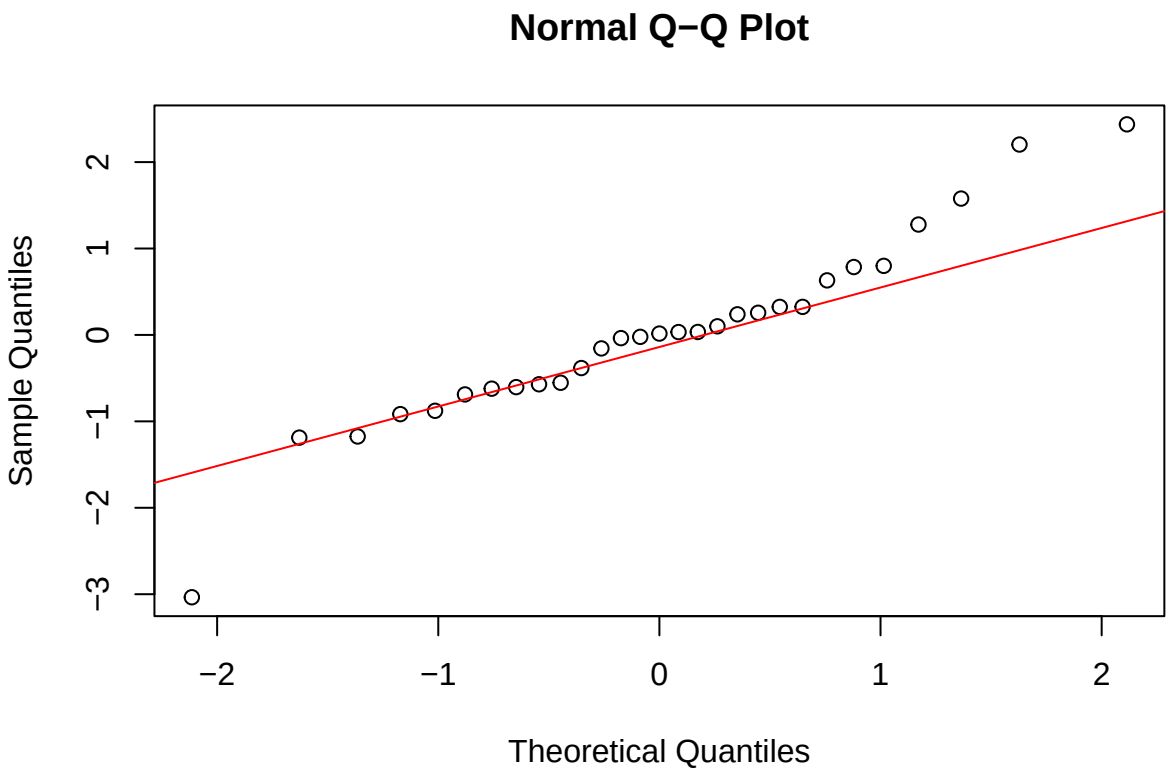
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
  #La linealidad se ve así así, podemos intentar arreglarlo.

  library(lmtest)
    bptest(mod1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mod1
## BP = 0.16554, df = 1, p-value = 0.6841
```

```
    # No hay problema con la homocedasticidad.

qqnorm(res_stu)
qqline(res_stu,col="red")
```



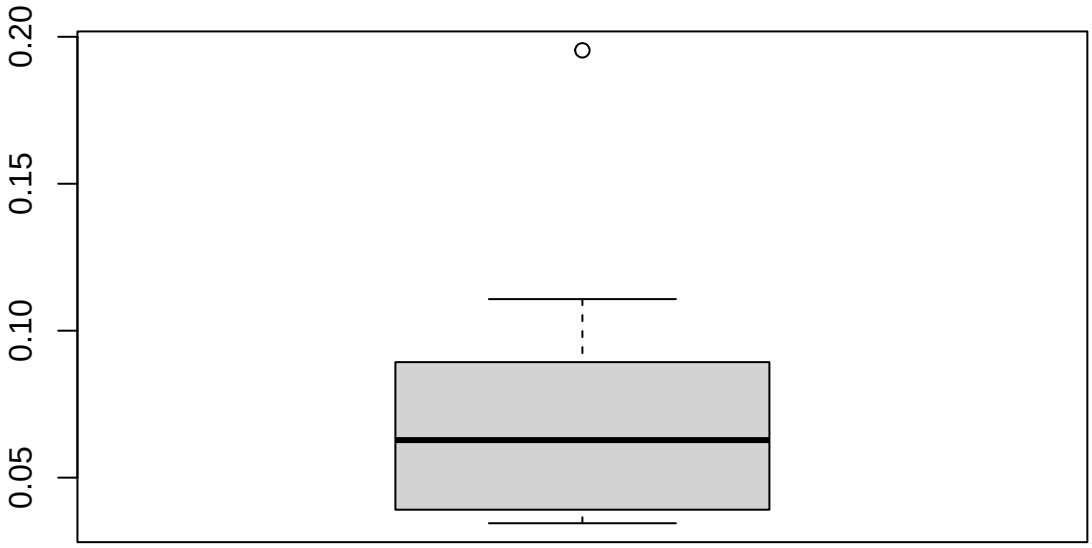

```
shapiro.test(res_stu) #No falla la normalidad.
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res_stu
## W = 0.94758, p-value = 0.1584
```

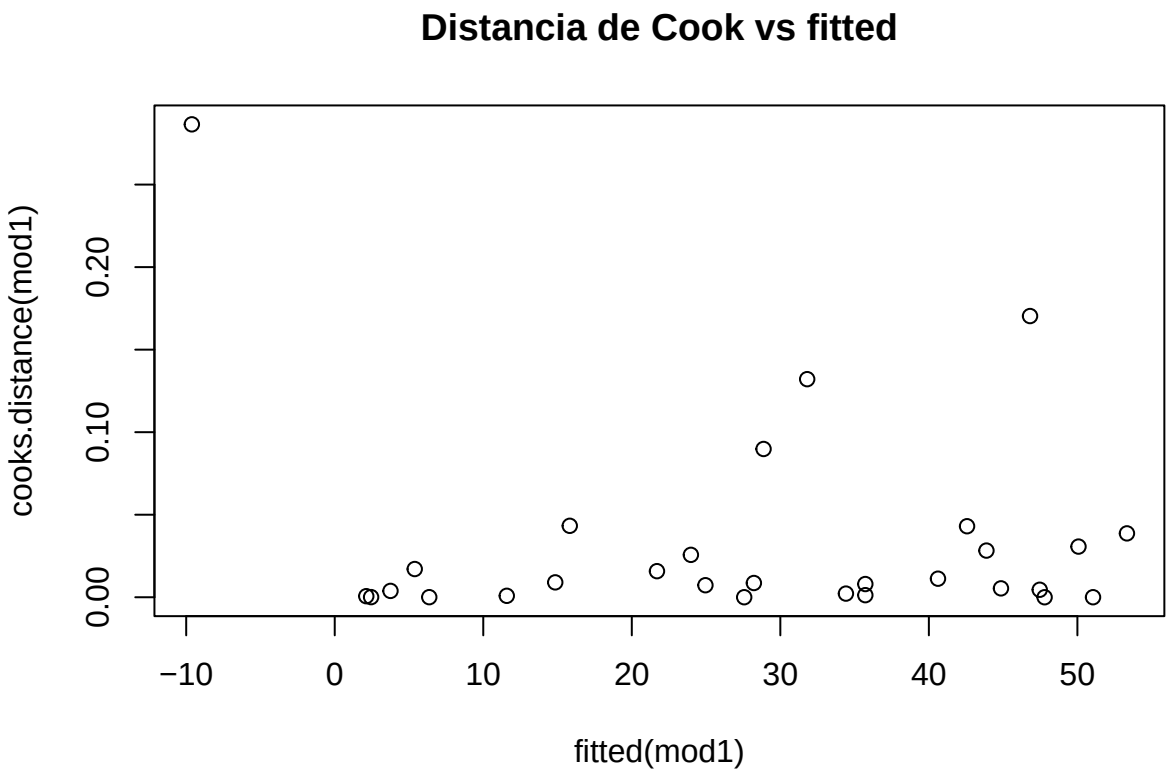
```
defectos2[abs(res_stu) > 3,] # Aparece un outlier, el 17
```

```
##    caso temperatura densidad  tasa defectuosos
## 17   17        2.36     26.3 222.1        13.4
```

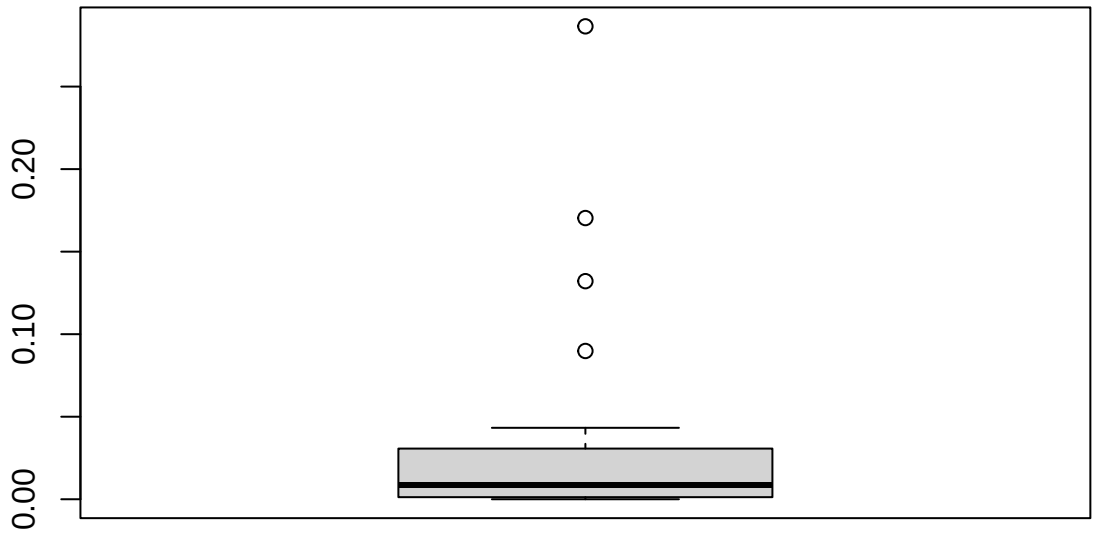
```
  boxplot(hatvalues(mod1))
```



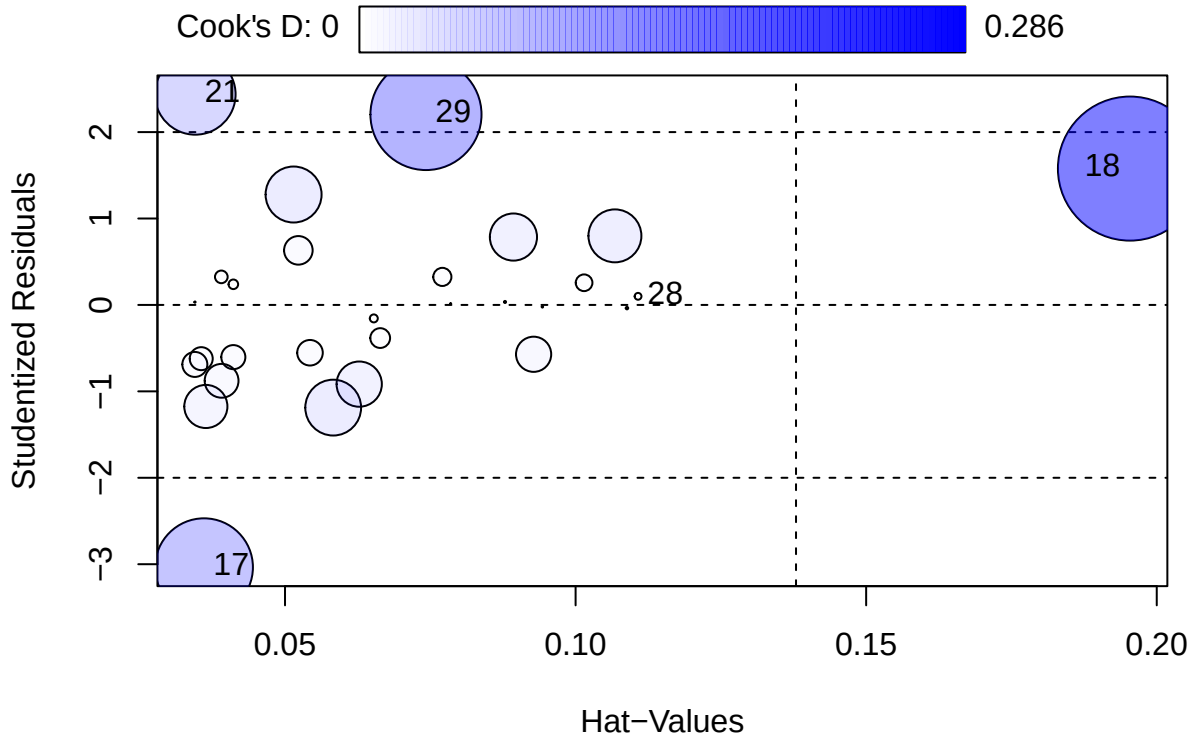

```
 plot(fitted(mod1),cooks.distance(mod1),main="Distancia de Cook vs fitted")
abline(h=0.33,col="red",lwd=1);
```




```
boxplot(cooks.distance(mod1))
```




```
 library(car)
influencePlot(mod1)
```




```
##         StudRes         Hat        CookD
## 17 -3.03504335 0.03606219 0.1321241664
## 18  1.57787941 0.19538915 0.2864890420
## 21  2.43668256 0.03455499 0.0898286937
## 28  0.09985421 0.11074677 0.0006445139
## 29  2.20316508 0.07425448 0.1703524302
```

```
  summary(influence.measures(mod1)) #El punto que era outlier, no es super influyente. Podríamos quitar
```

```
## Potentially influential observations of
##   lm(formula = defectuosos ~ temperatura, data = defectos2, na.action = na.exclude) :
##
##    dfb.1_ dfb.tmpr dffit cov.r   cook.d hat
## 17 -0.01  -0.12    -0.59  0.61_*  0.13   0.04
## 21  0.09   0.02     0.46  0.74_*  0.09   0.03
```

b) Ajusta un modelo con el que se pretende explicar la relación entre la temperatura y el número medio de defectos, con la información disponible, diagnostica el modelo y evalúa la efectividad de las posibles soluciones.

```
mod1 <- lm(defectuosos ~ temperatura, data=defectos, na.action=na.exclude)
summary(mod1)
```

```
##
## Call:
## lm(formula = defectuosos ~ temperatura, data = defectos, na.action = na.exclude)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -18.5952  -4.9203  -0.6253   4.2133  15.1861
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -40.938      5.298  -7.727 2.04e-08 ***
## temperatura   30.904      2.327  13.279 1.32e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.312 on 28 degrees of freedom
## Multiple R-squared:  0.863,  Adjusted R-squared:  0.8581
## F-statistic: 176.3 on 1 and 28 DF,  p-value: 1.317e-13
```

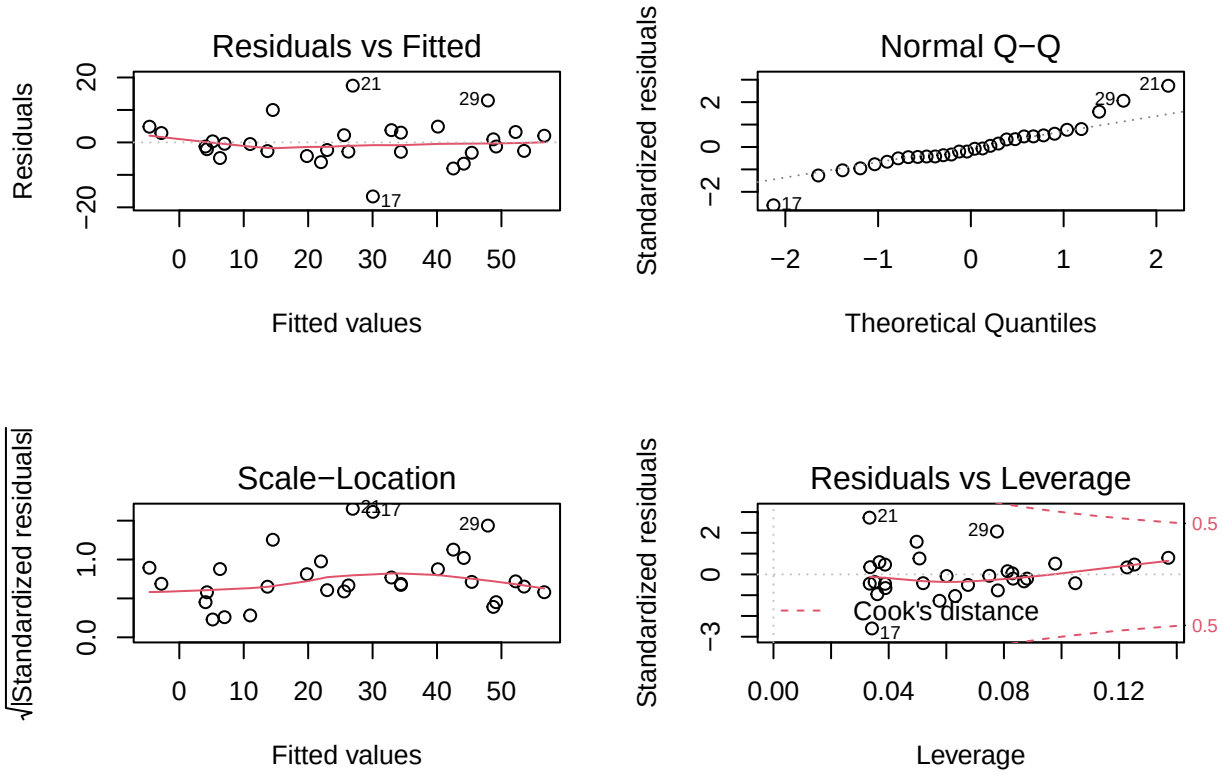
```
# Este modelo ya lo hemos diagnosticado antes. Vamos a ponerle una transformación a ver que tal:

mod2 <- lm(defectuosos ~ I(temperatura^2), data=defectos, na.action=na.exclude)
summary(mod2)
```

```
##
## Call:
## lm(formula = defectuosos ~ I(temperatura^2), data = defectos,
##     na.action = na.exclude)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.646  -2.885  -0.884   2.973  17.477
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -11.6847     2.8248  -4.137 0.000291 ***
## I(temperatura^2)   7.4925     0.4945  15.153 5.06e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.512 on 28 degrees of freedom
## Multiple R-squared:  0.8913, Adjusted R-squared:  0.8874
## F-statistic: 229.6 on 1 and 28 DF,  p-value: 5.063e-15
```

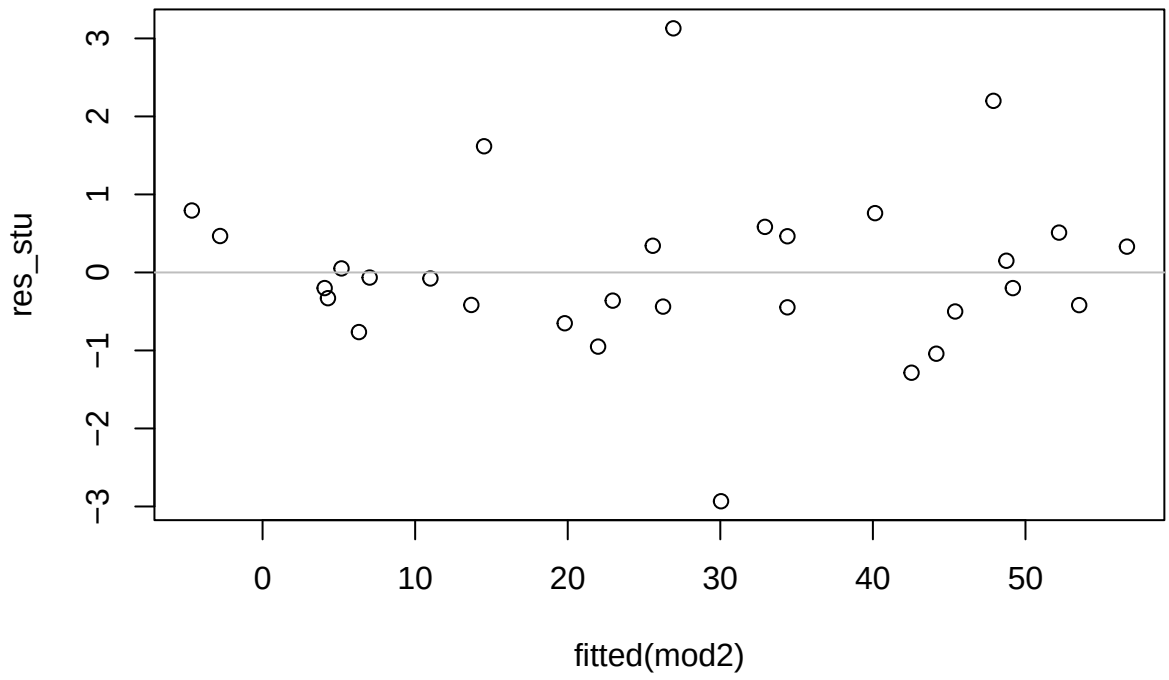
```
AIC(mod1,mod2) #Hemos mejorado un poquito
```

```
##      df     AIC
## mod1  3 208.434
## mod2  3 201.484
```

```
par(mfrow=c(2,2))
plot(mod2) # Parece que la linealidad en nuestros residuos ha mejorado!
```

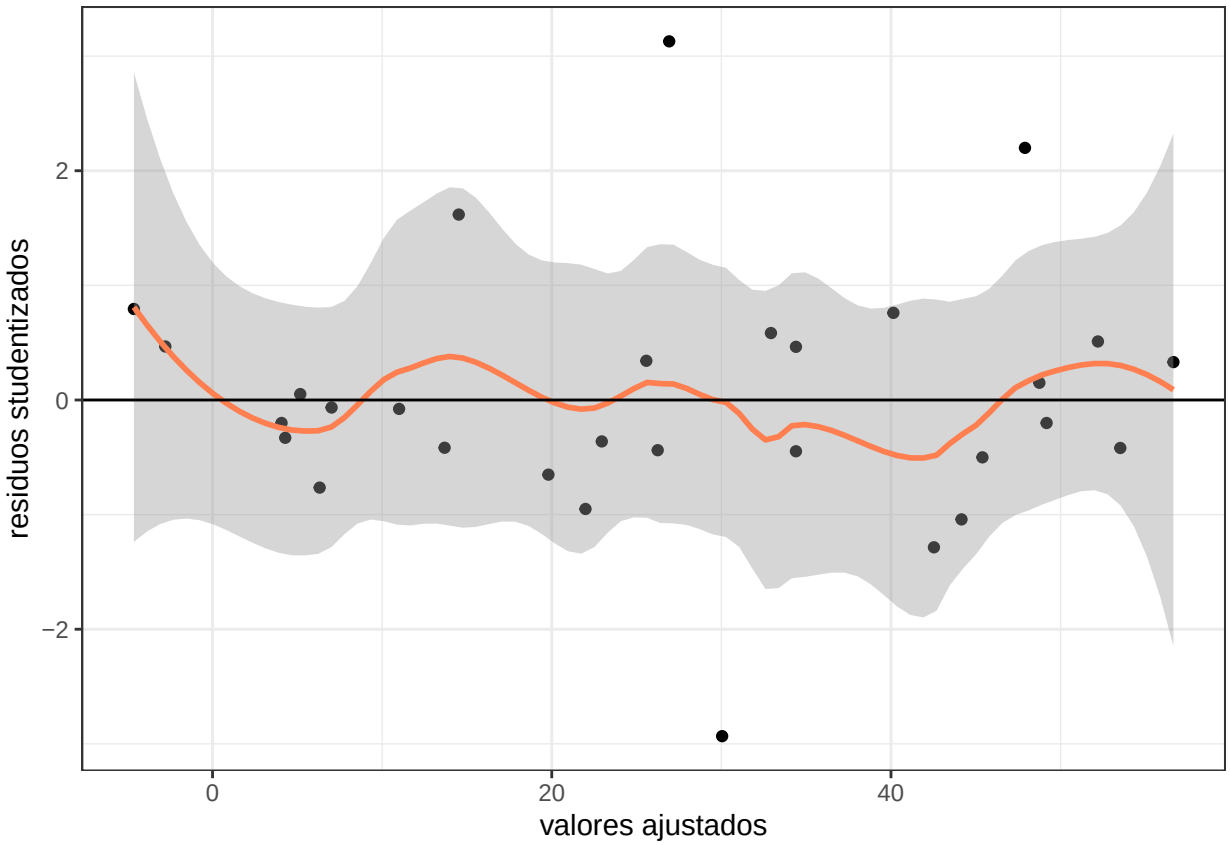



```
res_stu<-rstudent(mod2)
par(mfrow=c(1,1))
plot(fitted(mod2),res_stu)
 abline(h=0, col="gray")
```




```
library(ggplot2)
 library(gridExtra)
 ggplot(data = defectos, aes(x =fitted(mod2), y = res_stu)) +
 geom_point() + geom_smooth(color = "coral",span=0.4) + geom_hline(yintercept = 0) +
 labs(y = "residuos studentizados",
      x = "valores ajustados") +
 theme_bw()
```

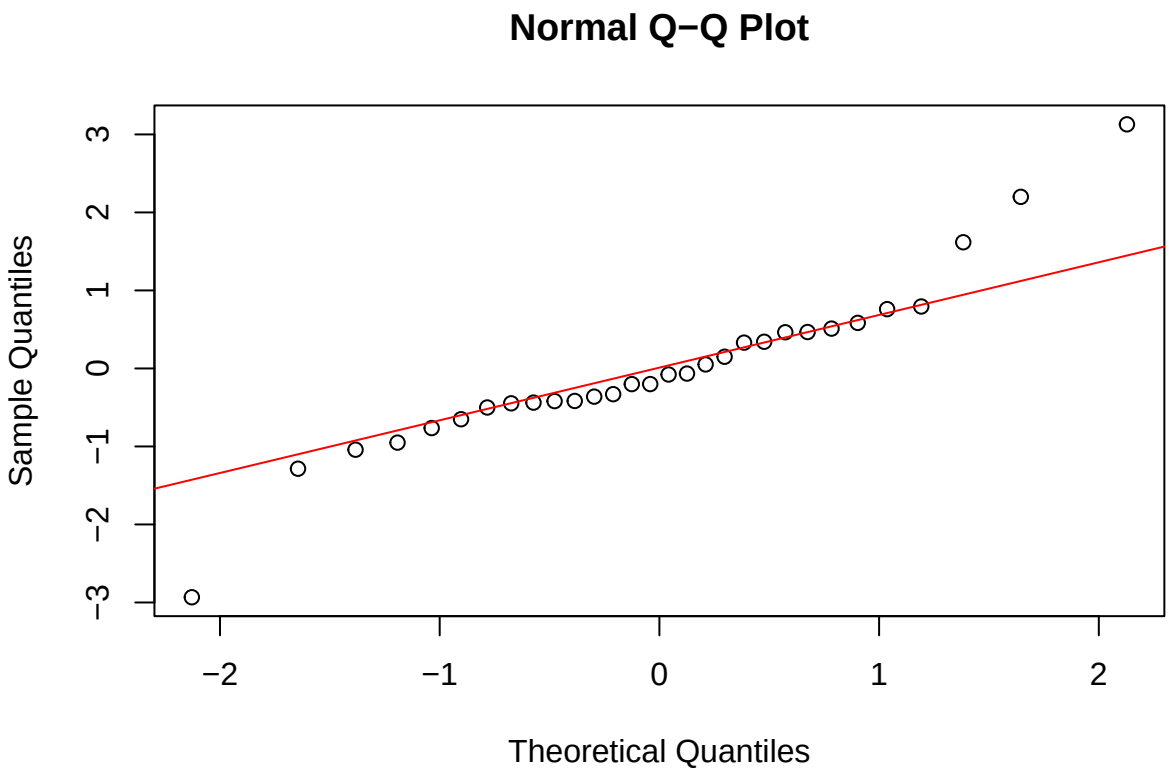
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
  library(lmtest)
    bptest(mod2)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mod2
## BP = 0.27391, df = 1, p-value = 0.6007
```

```
    # No hay problema con la homocedasticidad.



qqnorm(res_stu)
qqline(res_stu,col="red")
```



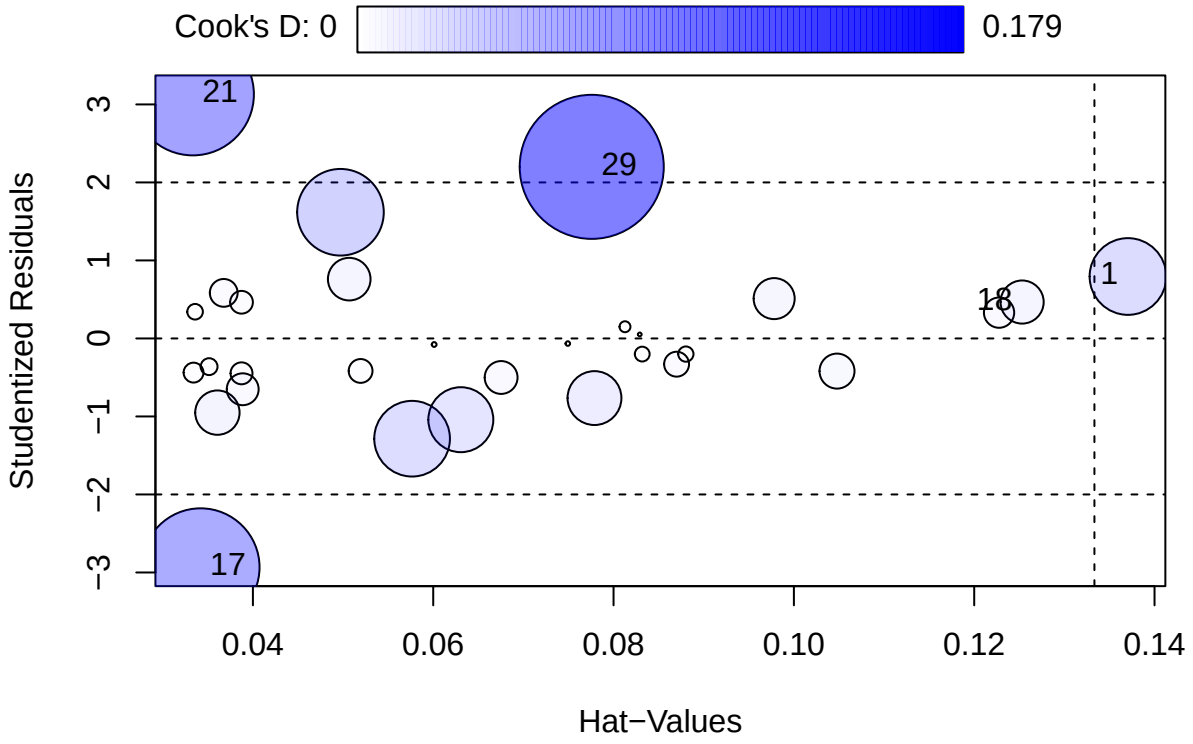

```
shapiro.test(res_stu) #Ahora falla la normalidad.
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res_stu
## W = 0.92163, p-value = 0.02957
```

```
defectos[abs(res_stu) > 3,] # Hay outliers, el punto 21
```

```
##    caso temperatura densidad tasa defectuosos
## 21   21        2.27    23.74  256        44.4
```

```
   library(car)
  influencePlot(mod2)
```




```
##        StudRes         Hat      CookD
## 1    0.7939912 0.13705211 0.05073096
## 17 -2.9330144 0.03419848 0.11978265
## 18   0.4667032 0.12531425 0.01605110
## 21   3.1289599 0.03333830 0.12848799
## 29   2.1996601 0.07757817 0.17893550
```

```
  summary(influence.measures(mod2))
```

```
## Potentially influential observations of
##   lm(formula = defectuosos ~ I(temperatura^2), data = defectos,      na.action = na.exclude) :
##
##    dfb.1_ dfb.I(^2 dffit cov.r   cook.d hat
## 10 -0.07   0.11     0.12  1.22_*  0.01   0.12
## 17 -0.15  -0.09    -0.55  0.64_*  0.12   0.03
## 21  0.25  -0.01     0.58  0.60_*  0.13   0.03
```

```
  # Observamos en el mod1 un pequeño problema de linealidad, que podemos arreglar
  # haciendo una transformación ^2 de la variable explicativa.
  # Si lo hacemos añadiendo el término cuadrático, nos falla la normalidad y nos
  # aparece un outlier aunque no muy influyente. .
```