Producción empresarial

J. Ramajo

2020

library(readr)  
library(car)

## Loading required package: carData

library(MASS)  
library(effects)

## Registered S3 methods overwritten by 'lme4':  
## method from  
## cooks.distance.influence.merMod car   
## influence.merMod car   
## dfbeta.influence.merMod car   
## dfbetas.influence.merMod car

## lattice theme set by effectsTheme()  
## See ?effectsTheme for details.

library(tidyverse)

## ── Attaching packages ──────────────────────────────────────────────────────────────────────────── tidyverse 1.3.0 ──

## ✓ ggplot2 3.3.1 ✓ dplyr 1.0.0  
## ✓ tibble 3.0.1 ✓ stringr 1.4.0  
## ✓ tidyr 1.1.0 ✓ forcats 0.5.0  
## ✓ purrr 0.3.4

## ── Conflicts ─────────────────────────────────────────────────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()  
## x dplyr::recode() masks car::recode()  
## x dplyr::select() masks MASS::select()  
## x purrr::some() masks car::some()

library(gvlma)  
library(tseries)

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

library(stats)  
library(lmtest)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

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

#  
PROD\_EMP <- read\_csv("PROD\_EMP.csv")

## Parsed with column specification:  
## cols(  
## K = col\_double(),  
## L = col\_double(),  
## Y = col\_double()  
## )

head(PROD\_EMP, n=10)

## # A tibble: 10 x 3  
## K L Y  
## <dbl> <dbl> <dbl>  
## 1 0.0102 3 0.0263  
## 2 0.0994 2 0.0811  
## 3 0.0139 3 0.129   
## 4 0.0900 4 0.273   
## 5 0.442 2 0.289   
## 6 0.243 5 0.350   
## 7 0.00223 2 0.386   
## 8 0.430 3 0.432   
## 9 0.669 8 0.454   
## 10 0.715 11 0.519

dim(PROD\_EMP)

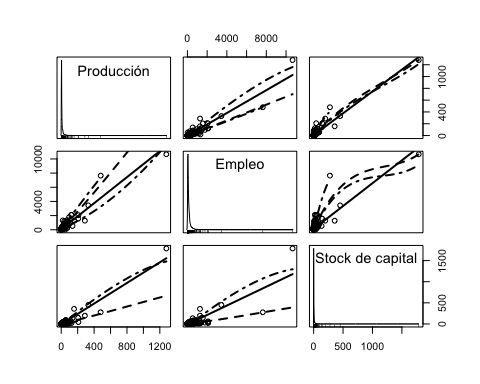
## [1] 569 3

summary(PROD\_EMP)

## K L Y   
## Min. : 0.0022 Min. : 1.0 Min. : 0.0263   
## 1st Qu.: 0.9834 1st Qu.: 45.0 1st Qu.: 2.6663   
## Median : 2.1993 Median : 86.0 Median : 4.6026   
## Mean : 11.5154 Mean : 201.1 Mean : 14.7192   
## 3rd Qu.: 6.2146 3rd Qu.: 176.0 3rd Qu.: 9.7421   
## Max. :1786.8992 Max. :10661.0 Max. :1279.3717

#  
# Matriz 'scatterplot' de los datos

#  
scatterplotMatrix(~Y + L + K, data=PROD\_EMP, var.labels=c("Producción", "Empleo", "Stock de capital"), col="black")



# Función Cobb-Douglas  
S(lm\_cd <- lm(log(Y) ~ log(L) + log(K), data = PROD\_EMP))

## Call: lm(formula = log(Y) ~ log(L) + log(K), data = PROD\_EMP)  
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.71146 0.09671 -17.70 <2e-16 \*\*\*  
## log(L) 0.71485 0.02314 30.89 <2e-16 \*\*\*  
## log(K) 0.20757 0.01719 12.08 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard deviation: 0.4781 on 566 degrees of freedom  
## Multiple R-squared: 0.8378  
## F-statistic: 1462 on 2 and 566 DF, p-value: < 2.2e-16   
## AIC BIC   
## 779.89 797.27

confint(lm\_cd)

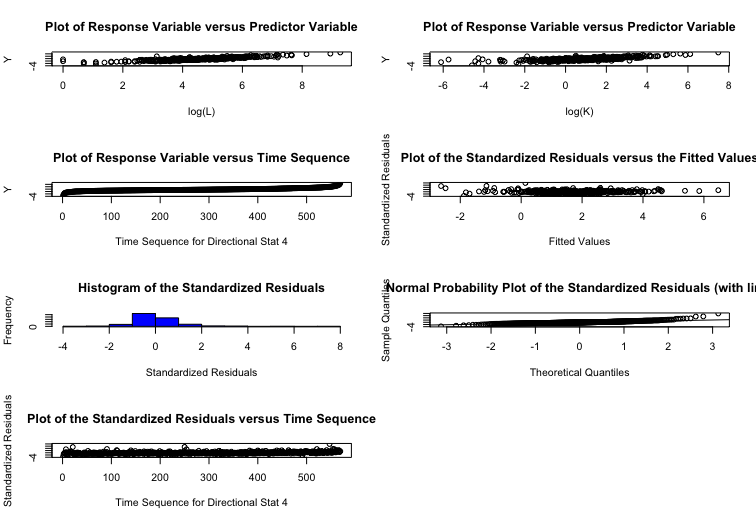
## 2.5 % 97.5 %  
## (Intercept) -1.9014148 -1.5215040  
## log(L) 0.6693927 0.7603008  
## log(K) 0.1738111 0.2413295

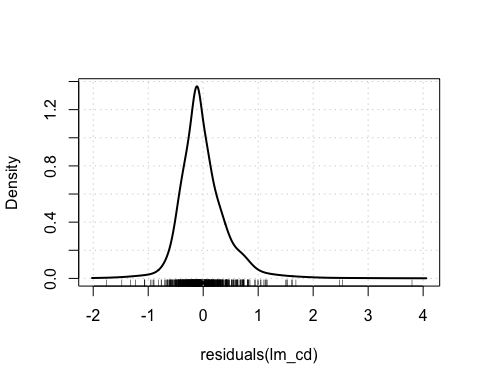
#  
# Diagnósticos  
#  
# Validación general del modelo (gvlma)

#  
gvmodel <- gvlma(lm\_cd)  
summary(gvmodel)

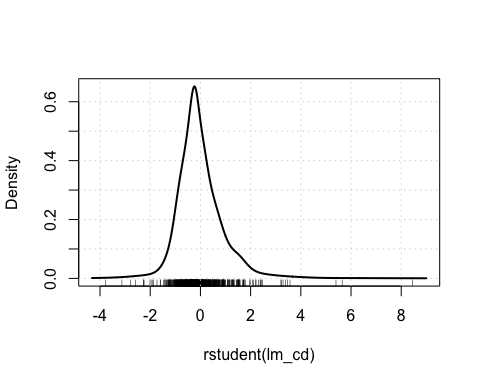
##   
## Call:  
## lm(formula = log(Y) ~ log(L) + log(K), data = PROD\_EMP)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.7604 -0.2665 -0.0694 0.1926 3.7975   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.71146 0.09671 -17.70 <2e-16 \*\*\*  
## log(L) 0.71485 0.02314 30.89 <2e-16 \*\*\*  
## log(K) 0.20757 0.01719 12.08 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4781 on 566 degrees of freedom  
## Multiple R-squared: 0.8378, Adjusted R-squared: 0.8373   
## F-statistic: 1462 on 2 and 566 DF, p-value: < 2.2e-16  
##   
##   
## ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS  
## USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:  
## Level of Significance = 0.05   
##   
## Call:  
## gvlma(x = lm\_cd)   
##   
## Value p-value Decision  
## Global Stat 2722.128 0.000e+00 Assumptions NOT satisfied!  
## Skewness 285.104 0.000e+00 Assumptions NOT satisfied!  
## Kurtosis 2392.189 0.000e+00 Assumptions NOT satisfied!  
## Link Function 43.343 4.594e-11 Assumptions NOT satisfied!  
## Heteroscedasticity 1.492 2.219e-01 Assumptions acceptable.

plot(gvmodel)

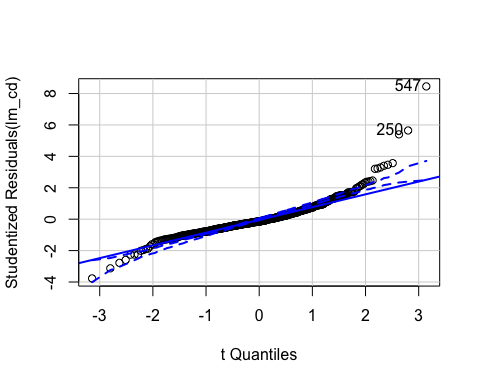
  
# gvmodel.del <- deletion.gvlma(gvmodel)  
# summary(gvmodel.del)  
#  
# 'residuals' -errores estimados y `rstudent()` - residuos estudentizados  
# `densityPlot()` chequeo de la distribución de los errores (densidades estimadas)  
densityPlot(residuals(lm\_cd))



densityPlot(rstudent(lm\_cd))



# `qqPlot()` chequeo de errores no-normales (comparación de los residuos estudentizados con una distribution t)  
qqPlot(lm\_cd)



## [1] 250 547

#  
# Chequeo de 'outliers' en la regresión  
#  
max(hatvalues(lm\_cd))

## [1] 0.07352641

which.max(hatvalues(lm\_cd))

## 250   
## 250

#  
outlierTest(lm\_cd)

## rstudent unadjusted p-value Bonferroni p  
## 547 8.453471 2.4146e-16 1.3739e-13  
## 250 5.647423 2.5824e-08 1.4694e-05  
## 21 5.400574 9.7990e-08 5.5757e-05

#  
max(cooks.distance(lm\_cd))

## [1] 0.8000356

which.max(cooks.distance(lm\_cd))

## 250   
## 250

#  
max(abs(dffits(lm\_cd)))

## [1] 1.590946

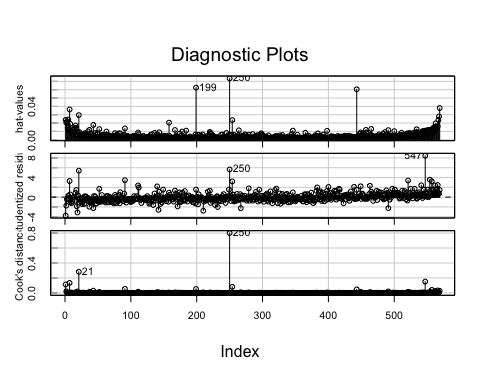
which.max(abs(dffits(lm\_cd)))

## 250   
## 250

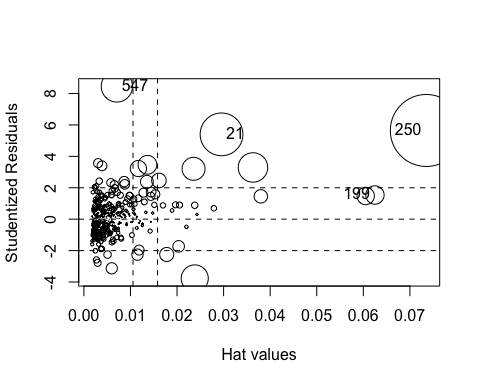
#  
# Medidas de influencia  
S(influence.measures(lm\_cd))

## Potentially influential observations of  
## lm(formula = log(Y) ~ log(L) + log(K), data = PROD\_EMP) :  
##   
## dfb.1\_ dfb.l(L) dfb.l(K) dffit cov.r cook.d hat   
## 1 -0.22 0.12 0.33 -0.59\_\* 0.96\_\* 0.11 0.02\_\*  
## 2 -0.21 0.19 -0.02 -0.25\_\* 1.01 0.02 0.02\_\*  
## 3 -0.03 0.02 0.04 -0.07 1.03\_\* 0.00 0.02\_\*  
## 4 -0.01 0.01 0.00 -0.02 1.02\_\* 0.00 0.01   
## 5 0.05 -0.04 0.02 0.05 1.03\_\* 0.00 0.02\_\*  
## 6 -0.04 0.03 0.00 -0.05 1.02\_\* 0.00 0.01   
## 7 0.15 -0.05 -0.42 0.64\_\* 0.99 0.13 0.04\_\*  
## 8 0.07 -0.07 0.03 0.08 1.02\_\* 0.00 0.02\_\*  
## 12 -0.01 0.01 0.00 -0.01 1.02\_\* 0.00 0.01   
## 15 0.01 -0.02 0.08 -0.09 1.02\_\* 0.00 0.01   
## 16 0.02 0.00 -0.08 0.12 1.02\_\* 0.00 0.02\_\*  
## 19 -0.21 0.19 -0.18 -0.24\_\* 0.96\_\* 0.02 0.01   
## 21 0.61 -0.49 -0.24 0.94\_\* 0.89\_\* 0.28 0.03\_\*  
## 43 -0.25 0.25 -0.27 -0.30\_\* 1.00 0.03 0.02\_\*  
## 91 0.14 -0.07 -0.23 0.41\_\* 0.96\_\* 0.05 0.01   
## 111 0.09 -0.05 -0.11 0.22\_\* 0.98\_\* 0.02 0.01   
## 142 -0.03 0.02 -0.07 -0.13 0.97\_\* 0.01 0.00   
## 158 -0.06 0.08 -0.12 0.13 1.02\_\* 0.01 0.02\_\*  
## 184 0.03 -0.03 0.03 0.03 1.02\_\* 0.00 0.01   
## 199 -0.22 0.27 -0.39 0.40\_\* 1.06\_\* 0.05 0.06\_\*  
## 210 0.03 -0.05 -0.03 -0.15 0.97\_\* 0.01 0.00   
## 250 1.52\_\* -1.54\_\* 1.26\_\* 1.59\_\* 0.92\_\* 0.80\_\* 0.07\_\*  
## 254 -0.14 0.22 -0.46 0.50\_\* 0.98\_\* 0.08 0.02\_\*  
## 267 0.11 -0.13 0.09 -0.16 0.98\_\* 0.01 0.01   
## 443 -0.26 0.31 -0.36 0.38\_\* 1.06\_\* 0.05 0.06\_\*  
## 457 0.02 -0.02 0.02 -0.03 1.02\_\* 0.00 0.01   
## 463 0.14 -0.13 0.14 0.18 0.98\_\* 0.01 0.01   
## 491 0.18 -0.19 0.05 -0.24\_\* 0.99 0.02 0.01   
## 521 0.14 -0.13 0.15 0.21 0.95\_\* 0.01 0.00   
## 534 0.03 -0.03 0.01 -0.03 1.02\_\* 0.00 0.01   
## 541 0.00 0.01 0.06 0.14 0.98\_\* 0.01 0.00   
## 547 0.14 0.01 -0.46 0.71\_\* 0.71\_\* 0.15 0.01   
## 549 0.00 0.00 0.01 0.01 1.02\_\* 0.00 0.01   
## 552 0.00 0.00 0.00 0.00 1.02\_\* 0.00 0.01   
## 553 -0.01 0.01 0.00 0.02 1.02\_\* 0.00 0.01   
## 554 -0.06 0.08 0.01 0.20 0.94\_\* 0.01 0.00   
## 557 -0.28 0.32 -0.24 0.35\_\* 0.96\_\* 0.04 0.01   
## 558 -0.03 0.04 0.00 0.05 1.02\_\* 0.00 0.01   
## 560 -0.03 0.03 0.01 0.05 1.02\_\* 0.00 0.01   
## 561 -0.09 0.09 -0.02 0.11 1.02\_\* 0.00 0.02   
## 562 0.02 -0.03 0.21 0.28\_\* 0.99 0.03 0.01   
## 563 -0.01 0.01 0.09 0.13 1.02\_\* 0.01 0.02\_\*  
## 566 -0.06 0.05 0.18 0.32\_\* 0.99 0.03 0.02\_\*  
## 567 -0.05 0.04 0.06 0.14 1.03\_\* 0.01 0.02\_\*  
## 568 -0.07 0.07 0.02 0.12 1.03\_\* 0.00 0.03\_\*  
## 569 -0.12 0.11 0.12 0.29\_\* 1.03\_\* 0.03 0.04\_\*

influenceIndexPlot(lm\_cd, vars=c("hat", "Studentized","Cook"))

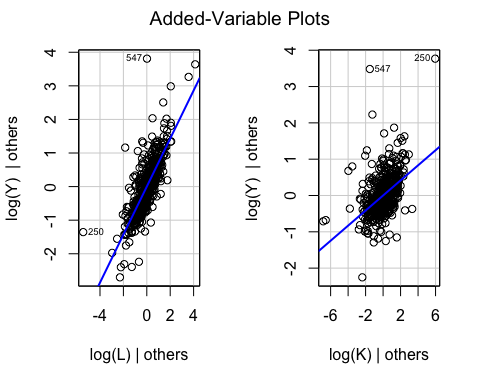


influencePlot(lm\_cd, xlab="Hat values")

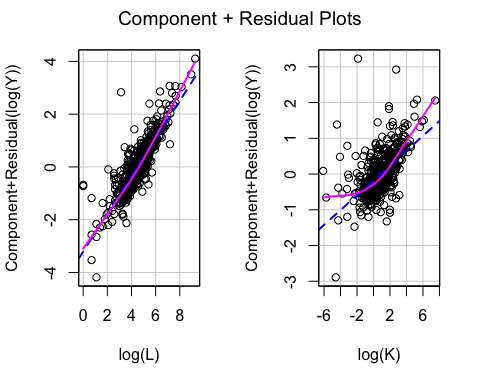


## StudRes Hat CookD  
## 21 5.400574 0.029552088 0.28202200  
## 199 1.537692 0.062489021 0.05240819  
## 250 5.647423 0.073526406 0.80003559  
## 547 8.453471 0.007080278 0.15105305

# Gráficos de variable añadida, buscando casos influyentes  
avPlots(lm\_cd, id=list(cex=0.60, method="mahal"))



# Chequeo de no linealidad: gráficos de componente+residuo  
crPlots(lm\_cd, smooth=list(span=0.7))



#  
ncvTest(lm\_cd, var.formula= ~ log(L) + log(K))

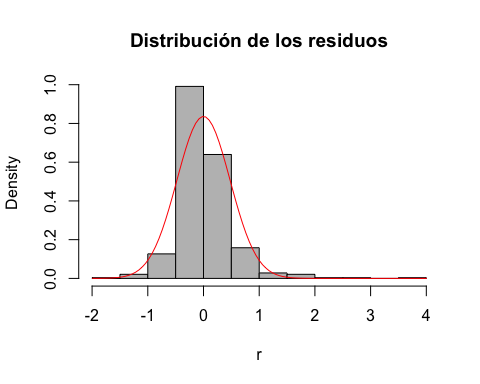
## Non-constant Variance Score Test   
## Variance formula: ~ log(L) + log(K)   
## Chisquare = 115.8566, Df = 2, p = < 2.22e-16

#  
# Normalidad de los residuos  
# Distribución de los residuos  
r <- resid(lm\_cd)  
rbar <- mean(r)  
sdr <- sd(r)  
rbar ; sdr

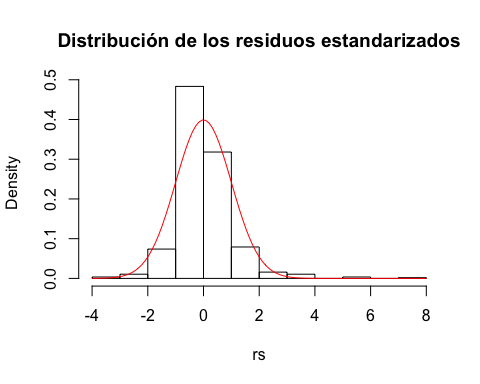
## [1] 2.55161e-17

## [1] 0.4772243

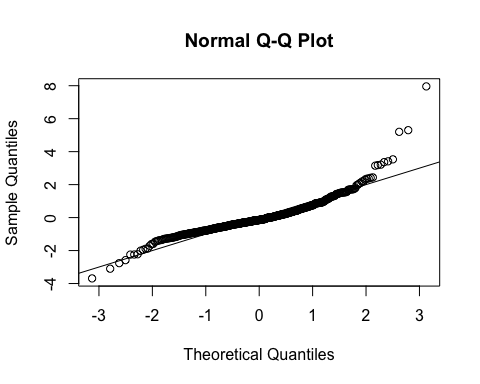
hist(r, col="grey", freq=FALSE, main="Distribución de los residuos",  
 ylab="Density", xlab="r")  
curve(dnorm(x, rbar, sdr), col=2, add=TRUE,  
 ylab="Density", xlab="r")



#   
# Residuos estandarizados  
rs<-((r-rbar)/sdr)   
hist(rs, freq=FALSE, main="Distribución de los residuos estandarizados",  
 ylab="Density", xlab="rs")  
curve(dnorm(x, 0, 1), col=2, add=TRUE,  
 ylab="Density", xlab="rs")



#  
qqnorm(rs)   
abline(0,1)



#  
jarque.bera.test(r) #(package 'tseries')

##   
## Jarque Bera Test  
##   
## data: r  
## X-squared = 2677.3, df = 2, p-value < 2.2e-16

shapiro.test(r)

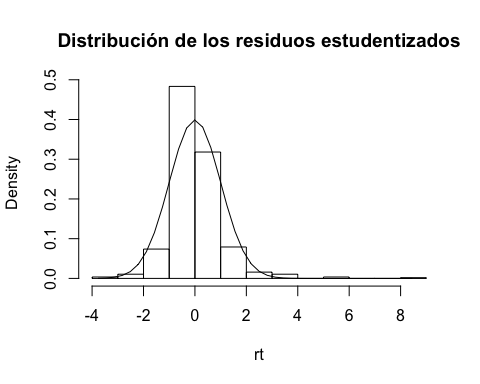
##   
## Shapiro-Wilk normality test  
##   
## data: r  
## W = 0.88577, p-value < 2.2e-16

#  
# Distribución de los residuos estudentizados  
#   
n <- nobs(lm\_cd)  
k <- n-df.residual(lm\_cd)  
n ; k

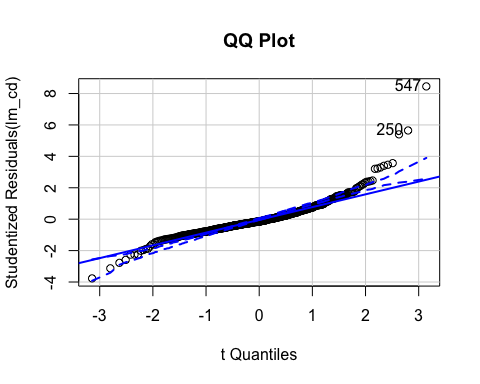
## [1] 569

## [1] 3

rt <- studres(lm\_cd)  
hist(rt, freq=FALSE,  
 main="Distribución de los residuos estudentizados")  
xfit<-seq(min(rt),max(rt),length=40)  
yfit<-dt(xfit,n-k-2)  
lines(xfit, yfit)

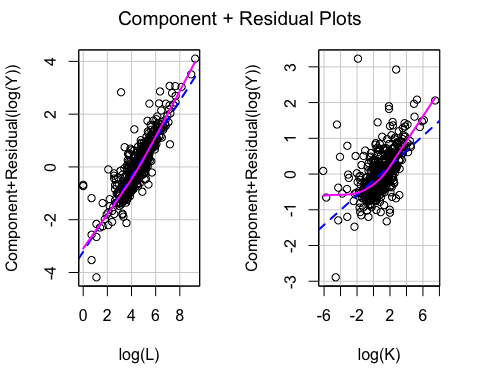


#  
# QQ plot de los residuos estudentizados  
qqPlot(lm\_cd, main="QQ Plot")

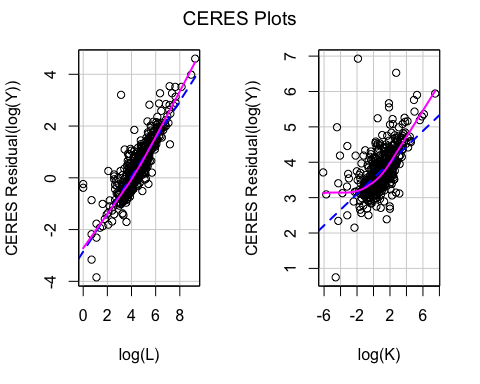


## [1] 250 547

#  
# No linealidad  
# Component+R plots  
crPlots(lm\_cd)



# Ceres plots  
ceresPlots(lm\_cd)



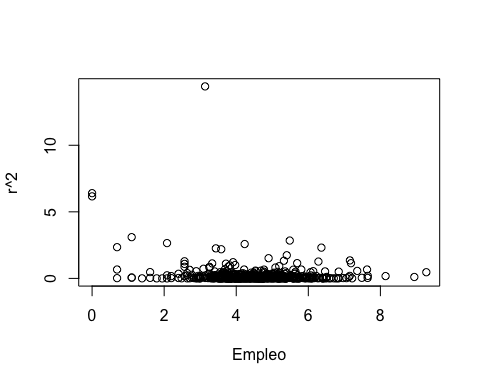
# Test RESET  
resettest(lm\_cd, power=2, type="fitted")

##   
## RESET test  
##   
## data: lm\_cd  
## RESET = 46.587, df1 = 1, df2 = 565, p-value = 2.266e-11

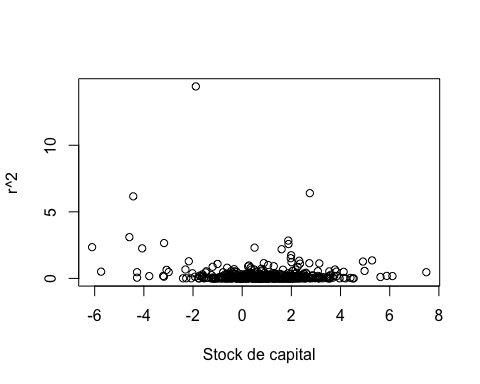
resettest(lm\_cd, power=2:3, type="fitted")

##   
## RESET test  
##   
## data: lm\_cd  
## RESET = 24.493, df1 = 2, df2 = 564, p-value = 6.303e-11

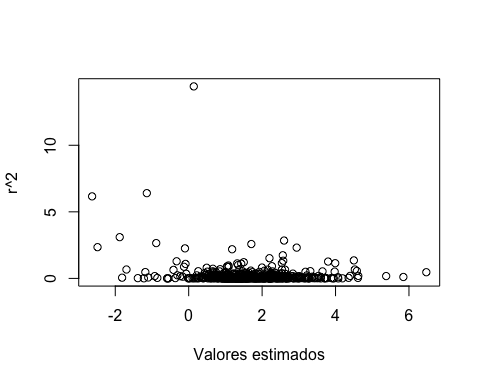
#  
# Chequeo de varianza no constante: heteroscedasticidad  
r2 <- resid(lm\_cd)^2  
yhat <- fitted(lm\_cd)  
l\_L <- log(PROD\_EMP$L)  
l\_K <- log(PROD\_EMP$K)  
plot(l\_L,r2, xlab="Empleo", ylab="r^2")



plot(l\_K,r2, xlab="Stock de capital", ylab="r^2")

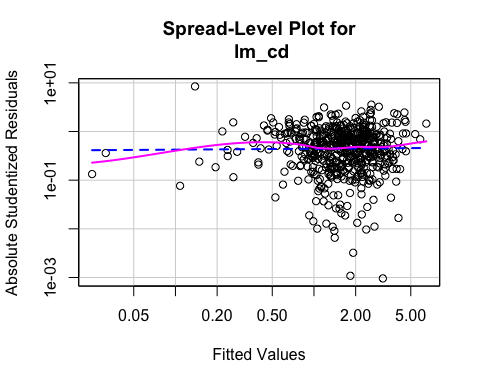


#  
plot(yhat,r2, xlab="Valores estimados", ylab="r^2")



#  
spreadLevelPlot(lm\_cd)

## Warning in spreadLevelPlot.lm(lm\_cd):   
## 28 negative fitted values removed



##   
## Suggested power transformation: 0.9813464

#  
ncvTest(lm\_cd)

## Non-constant Variance Score Test   
## Variance formula: ~ fitted.values   
## Chisquare = 113.3136, Df = 1, p = < 2.22e-16

ncvTest(lm\_cd, ~ log(L) + log(K) + I((log(L))^2) + I((log(K))^2) + I(log(L)\*log(K)) ) # Test de White (score test)

## Non-constant Variance Score Test   
## Variance formula: ~ log(L) + log(K) + I((log(L))^2) + I((log(K))^2) + I(log(L) \* log(K))   
## Chisquare = 491.2026, Df = 5, p = < 2.22e-16

ncvTest(lm\_cd, ~ log(L) + log(K) ) # Test de Breusch-Pagan (las variables Zs pueden ser externas al modelo) (score test)

## Non-constant Variance Score Test   
## Variance formula: ~ log(L) + log(K)   
## Chisquare = 115.8566, Df = 2, p = < 2.22e-16

#  
bptest(lm\_cd) # Breusch-Pagan robusto (variante robusta de Koenker)

##   
## studentized Breusch-Pagan test  
##   
## data: lm\_cd  
## BP = 19.237, df = 2, p-value = 6.647e-05

bptest(lm\_cd, studentize = FALSE) # Breusch-Pagan estándar (escalado)

##   
## Breusch-Pagan test  
##   
## data: lm\_cd  
## BP = 115.86, df = 2, p-value < 2.2e-16

bptest(lm\_cd, studentize = FALSE, varformula = ~ log(L), data = PROD\_EMP ) #

##   
## Breusch-Pagan test  
##   
## data: lm\_cd  
## BP = 115.18, df = 1, p-value < 2.2e-16

#  
# Corrección de la heteroscedasticidad  
#  
# Errores estándar (SEs) robustos  
S(lm\_cd, vcov.=hccm(lm\_cd, type = "hc1")) # hc1 (corrección de White)

## Call: lm(formula = log(Y) ~ log(L) + log(K), data = PROD\_EMP)  
## Standard errors computed by hccm(lm\_cd, type = "hc1")   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.71146 0.17500 -9.780 < 2e-16 \*\*\*  
## log(L) 0.71485 0.04137 17.279 < 2e-16 \*\*\*  
## log(K) 0.20757 0.03025 6.862 1.78e-11 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard deviation: 0.4781 on 566 degrees of freedom  
## Multiple R-squared: 0.8378  
## F-statistic: 510.4 on 2 and 566 DF, p-value: < 2.2e-16   
## AIC BIC   
## 779.89 797.27

#  
# Modelo translog  
#  
S(lm\_tl <- lm(log(Y) ~ log(L) + log(K) + I((log(L))^2) + I((log(K))^2) + I(log(L)\*log(K)), data = PROD\_EMP))

## Call: lm(formula = log(Y) ~ log(L) + log(K) + I((log(L))^2) + I((log(K))^2) +  
## I(log(L) \* log(K)), data = PROD\_EMP)  
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.957857 0.211377 -4.531 7.16e-06 \*\*\*  
## log(L) 0.315355 0.100434 3.140 0.001779 \*\*   
## log(K) 0.359393 0.053041 6.776 3.13e-11 \*\*\*  
## I((log(L))^2) 0.047207 0.012236 3.858 0.000128 \*\*\*  
## I((log(K))^2) 0.035384 0.006159 5.745 1.50e-08 \*\*\*  
## I(log(L) \* log(K)) -0.043268 0.012381 -3.495 0.000512 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard deviation: 0.4478 on 563 degrees of freedom  
## Multiple R-squared: 0.8585  
## F-statistic: 683.1 on 5 and 563 DF, p-value: < 2.2e-16   
## AIC BIC   
## 708.35 738.76

S(lm\_tl, vcov.=hccm(lm\_tl, type = "hc1")) # hc1 (corrección de White)

## Call: lm(formula = log(Y) ~ log(L) + log(K) + I((log(L))^2) + I((log(K))^2) +  
## I(log(L) \* log(K)), data = PROD\_EMP)  
## Standard errors computed by hccm(lm\_tl, type = "hc1")   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.957857 0.400962 -2.389 0.01723 \*   
## log(L) 0.315355 0.188766 1.671 0.09535 .   
## log(K) 0.359393 0.111510 3.223 0.00134 \*\*   
## I((log(L))^2) 0.047207 0.022171 2.129 0.03367 \*   
## I((log(K))^2) 0.035384 0.008899 3.976 7.92e-05 \*\*\*  
## I(log(L) \* log(K)) -0.043268 0.024302 -1.780 0.07554 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard deviation: 0.4478 on 563 degrees of freedom  
## Multiple R-squared: 0.8585  
## F-statistic: 593.8 on 5 and 563 DF, p-value: < 2.2e-16   
## AIC BIC   
## 708.35 738.76