Actividad 13

Oskar Arturo Gamboa Reyes

2024-09-10

```r
data = cars
print(data)
```

```
##    speed dist
## 1      4    2
## 2      4   10
## 3      7    4
## 4      7   22
## 5      8   16
## 6      9   10
## 7     10   18
## 8     10   26
## 9     10   34
## 10    11   17
## 11    11   28
## 12    12   14
## 13    12   20
## 14    12   24
## 15    12   28
## 16    13   26
## 17    13   34
## 18    13   34
## 19    13   46
## 20    14   26
## 21    14   36
## 22    14   60
## 23    14   80
## 24    15   20
## 25    15   26
## 26    15   54
## 27    16   32
## 28    16   40
## 29    17   32
## 30    17   40
## 31    17   50
## 32    18   42
## 33    18   56
## 34    18   76
## 35    18   84
## 36    19   36
## 37    19   46
## 38    19   68
```

```
## 39    20   32
## 40    20   48
## 41    20   52
## 42    20   56
## 43    20   64
## 44    22   66
## 45    23   54
## 46    24   70
## 47    24   92
## 48    24   93
## 49    24  120
## 50    25   85
```

Análisis de normalidad

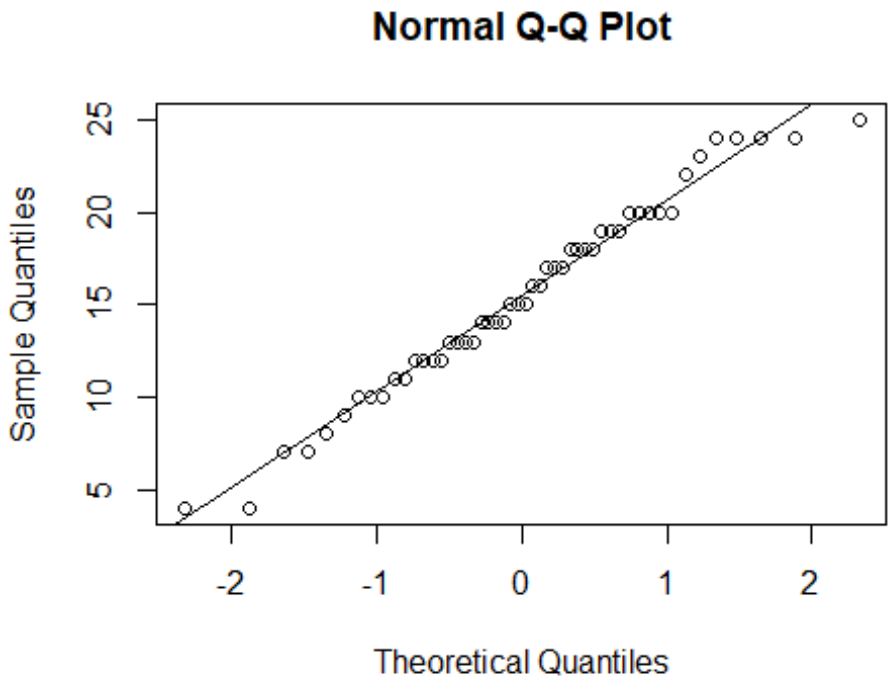
Pruebas de normalidad

```
library(nortest)
lillie.test(data$speed)

##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  data$speed
## D = 0.068539, p-value = 0.8068

shapiro.test(data$speed)

##
##  Shapiro-Wilk normality test
##
## data:  data$speed
## W = 0.97765, p-value = 0.4576

lillie.test(data$dist)

##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  data$dist
## D = 0.12675, p-value = 0.04335

shapiro.test(data$dist)

##
##  Shapiro-Wilk normality test
##
## data:  data$dist
## W = 0.95144, p-value = 0.0391
```

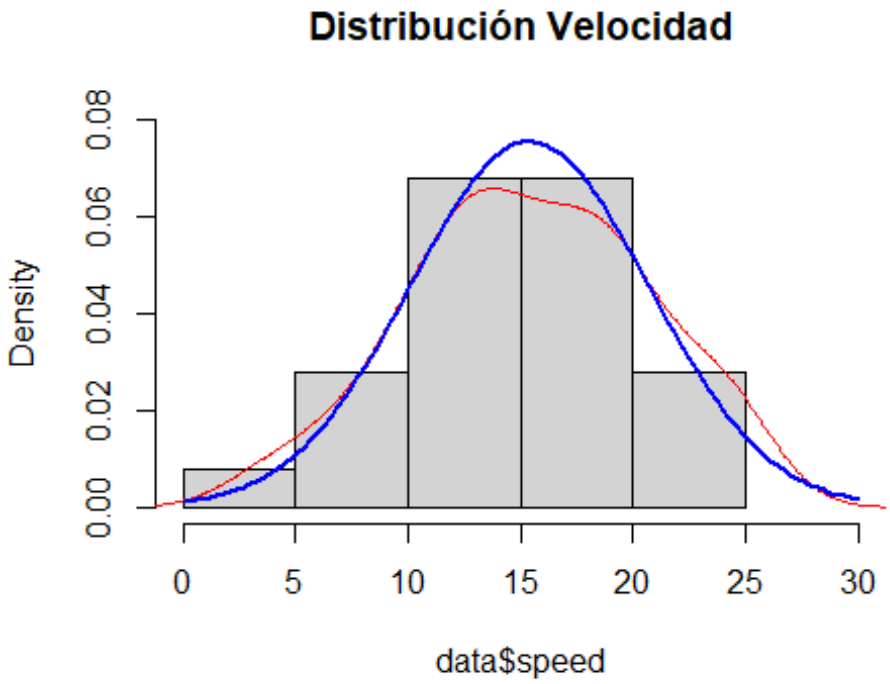
Gráficas

Velocidad

```
qqnorm(data$speed)
qqline(data$speed)
```



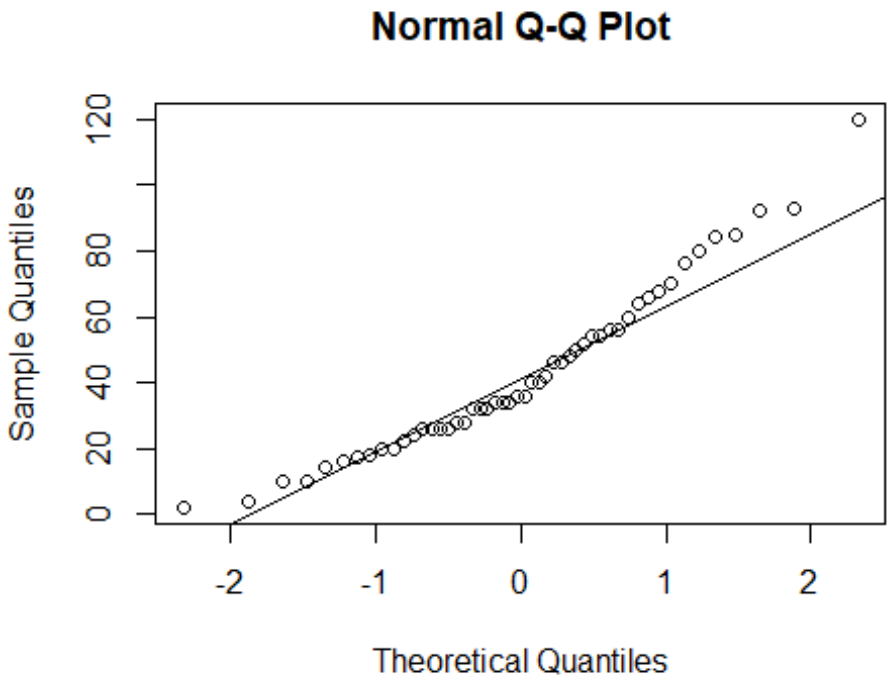

```
hist(data$speed,freq=FALSE, xlim=c(0,30), ylim=c(0,0.08),main="Distribución
Velocidad")
lines(density(data$speed),col="red")
curve(dnorm(x,mean=mean(data$speed),sd=sd(data$speed)), from=0, to=30,
add=TRUE, col="blue",lwd=2)
```



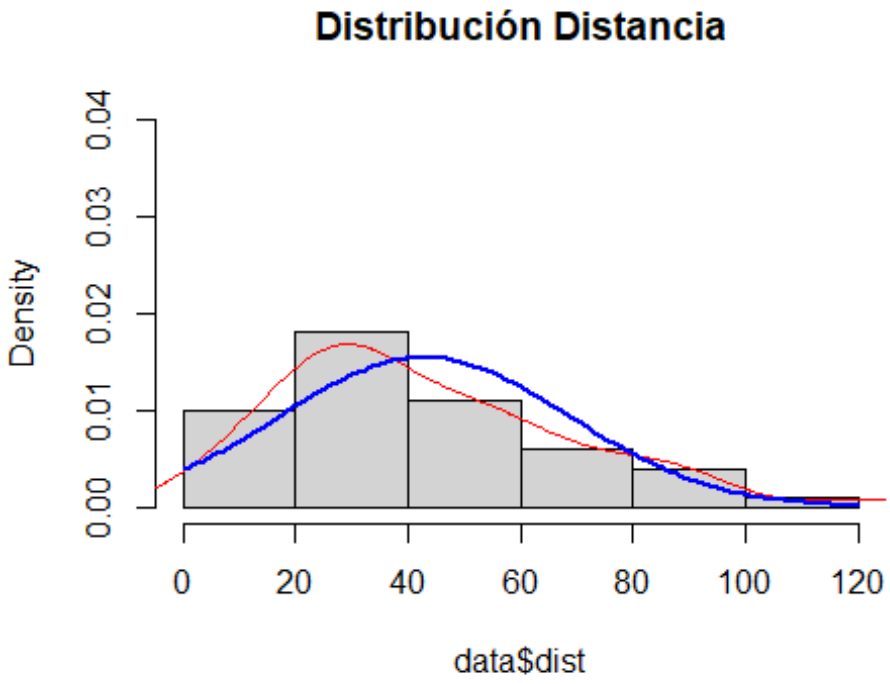

```
library(e1071)
cat("Sesgo", skewness(data$speed)," ")

## Sesgo -0.1105533

cat("Curtosis", kurtosis(data$speed))

## Curtosis -0.6730924
```

Distancia

```
qqnorm(data$dist)
qqline(data$dist)
```




```r
hist(data$dist,freq=FALSE, xlim=c(0,120), ylim=c(0,0.04),main="Distribución Distancia")
lines(density(data$dist),col="red")
curve(dnorm(x,mean=mean(data$dist),sd=sd(data$dist)), from=0, to=120, add=TRUE, col="blue",lwd=2)
```




```
library(e1071)
cat("Sesgo", skewness(data$dist)," ")

## Sesgo 0.7591268

cat("Curtosis", kurtosis(data$dist))

## Curtosis 0.1193971
```

Conclusiones

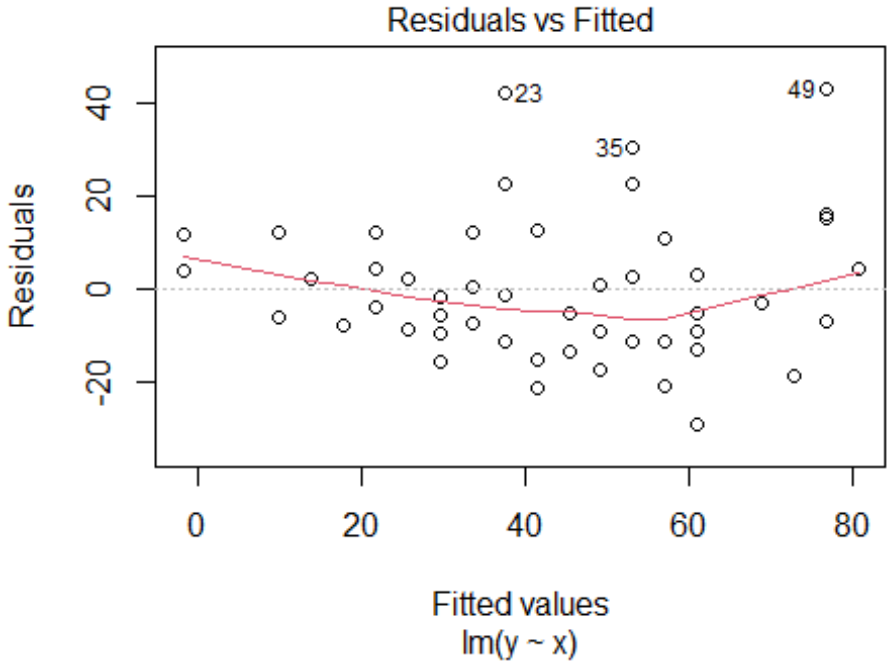
La variable de velocidad tiene una curva bastante normal, pasa todos las pruebas y no se notan muchos casos atípicos, relamente no hay que modificarla para tener normalidad. Al contrario, la variable de distancia podemos ver que no se comporta de manera normal, tiene un gran sesgo a la derecha ya que tiene datos atípicos altos y una curtosis baja ya que los datos estan muy homogenizados.

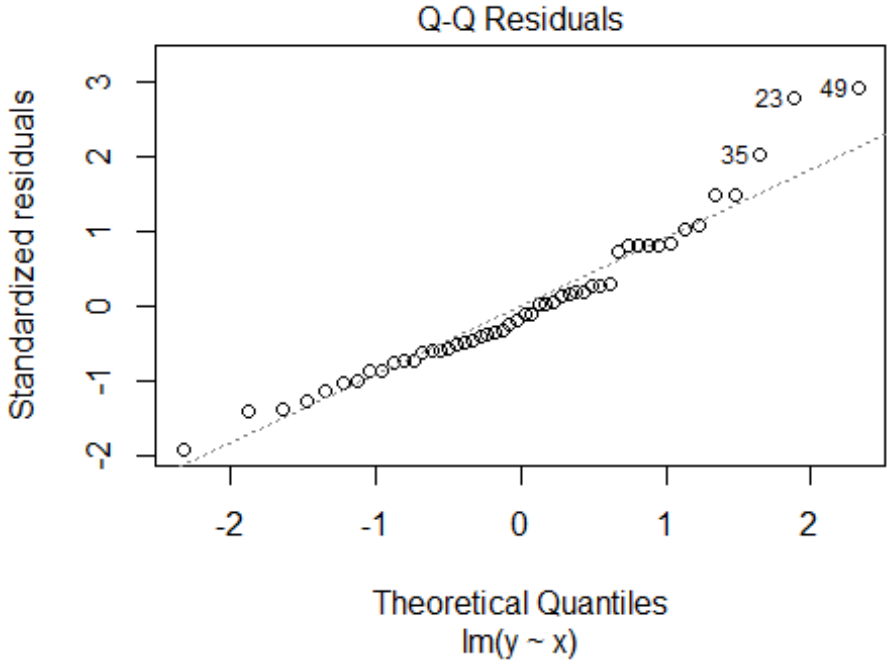
Regresión lineal

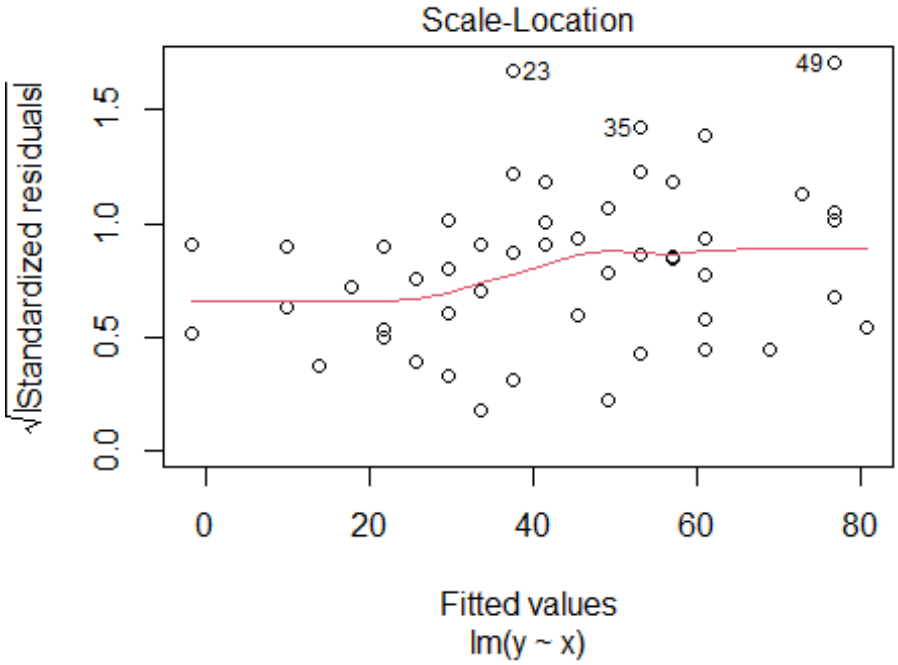
```
x = data$speed
y = data$dist

modeloLineal = lm(y~x)

summary(modeloLineal)
```

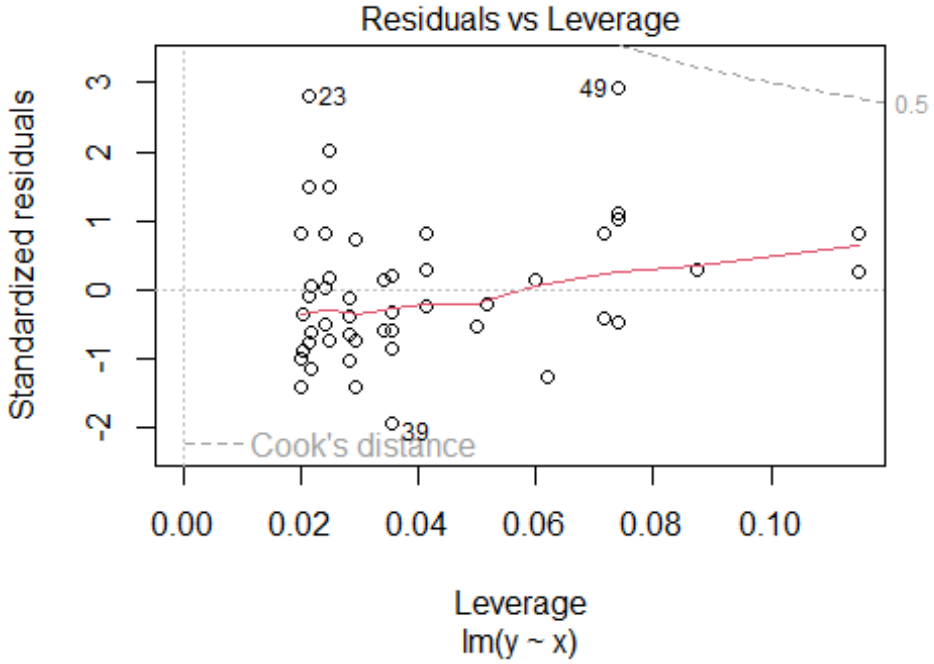
```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## x             3.9324     0.4155   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
## Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

```
plot(modeloLineal)
```


Residuals vs Fitted
Residuals
Fitted values
lm(y ~ x)
23
49
35
-20
0
20
40
0
20
40
60
80


Q-Q Residuals
Standardized residuals
Theoretical Quantiles
lm(y ~ x)
23
49
35
-2
-1
0
1
2
-2
-1
0
1
2
3





Homocedasticidad

```
library(lmtest)
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

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

bptest(modeloLineal)

##
##  studentized Breusch-Pagan test
##
## data:  modeloLineal
## BP = 3.2149, df = 1, p-value = 0.07297

gqtest(modeloLineal)

##
##  Goldfeld-Quandt test
##
## data:  modeloLineal
## GQ = 1.5512, df1 = 23, df2 = 23, p-value = 0.1498
## alternative hypothesis: variance increases from segment 1 to 2
```

Independencia

```
dwtest(modeloLineal)

##
##  Durbin-Watson test
##
## data:  modeloLineal
## DW = 1.6762, p-value = 0.09522
## alternative hypothesis: true autocorrelation is greater than 0

bgtest(modeloLineal)

##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  modeloLineal
## LM test = 1.2908, df = 1, p-value = 0.2559
```

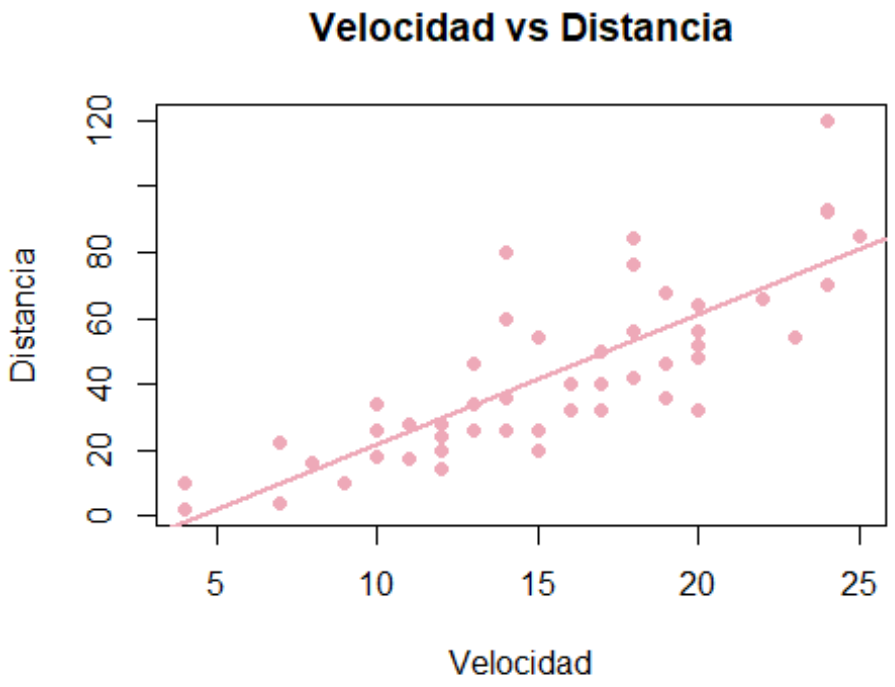
Linealidad

```
library(lmtest)
resettest(modeloLineal)

##
##  RESET test
##
```

```
## data:  modeloLineal
## RESET = 1.5554, df1 = 2, df2 = 46, p-value = 0.222
```

Gráfica

```
plot(x,y, col="pink2", main="Velocidad vs Distancia", ylab="Distancia", xlab
= "Velocidad", pch=19)
abline(lm(y~x), col="pink2", lwd=2)
```



Conclusiones

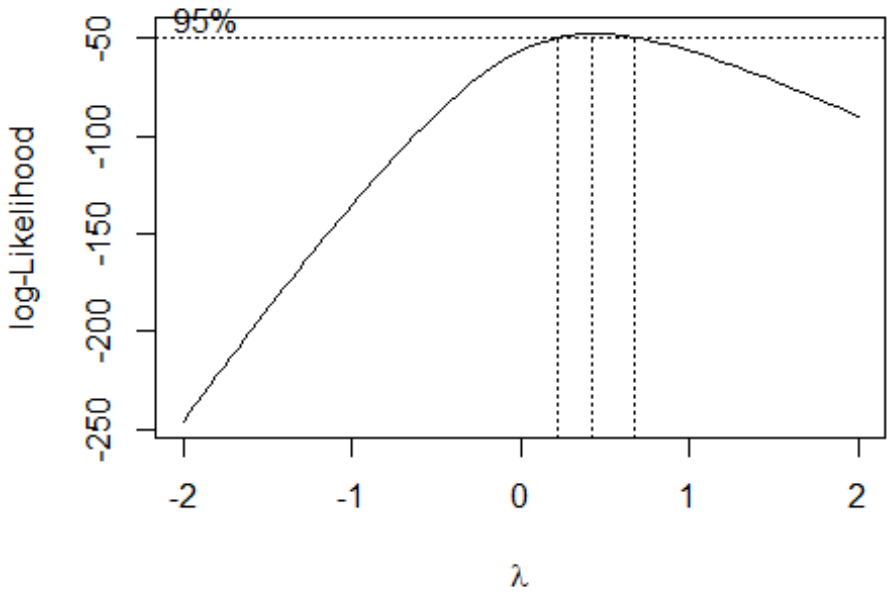
Aunque podemos ver que este modelo si explica ciertos comportamientos de nuestros datos, no es muy precisa, al solamente tener 65% de ajuste. Aunque paso las pruebas de homocestidad e independencia no obtuvo un buen resultado en la linealidad, al ver las gráficas podemos ver que hay datos atípicos que nuestro modelo no logra explicar, sería necesario hacer otro modelo para ver si se encuentra una mejor predicción.

Regresión no lineal

La variable con menos normalidad fue la distancia.

Normalización

```
library(MASS)
bc<-boxcox(lm(y~x))
```




```
l=bc$x[which.max(bc$y)]
print(paste("lambda: ", l))

## [1] "lambda:  0.424242424242424"
```

Aproximado = $\sqrt{y}$ Exacto = $\frac{(y)^{0.42}}{0.42}$

```
aprox = sqrt(y)
exacto = (((y)^0.42))/0.42
```

Pruebas de Normalidad

```
library(e1071)
cat("Sesgo Modelo Aproximado", skewness(aprox)," ")

## Sesgo Modelo Aproximado -0.01902765

cat("Curtosis Modelo Aproximado", kurtosis(aprox),"\n")

## Curtosis Modelo Aproximado -0.3144682

cat("Sesgo Modelo Exacto", skewness(exacto)," ")

## Sesgo Modelo Exacto -0.1790006

cat("Curtosis Modelo Exacto", kurtosis(exacto),"\n")

## Curtosis Modelo Exacto -0.1774661
```

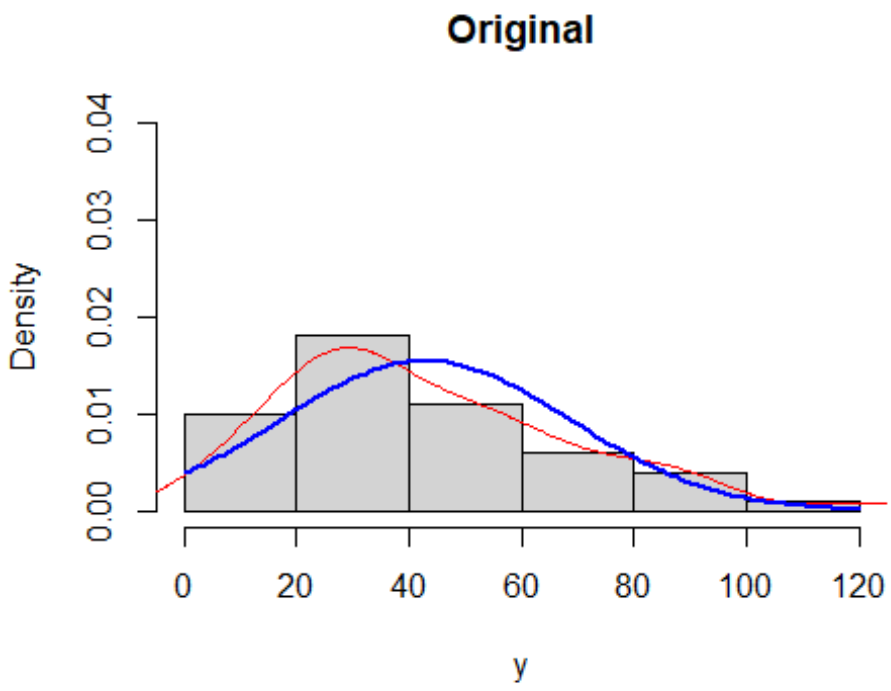
```r
cat("Sesgo", skewness(y)," ")
```

```
## Sesgo 0.7591268
```

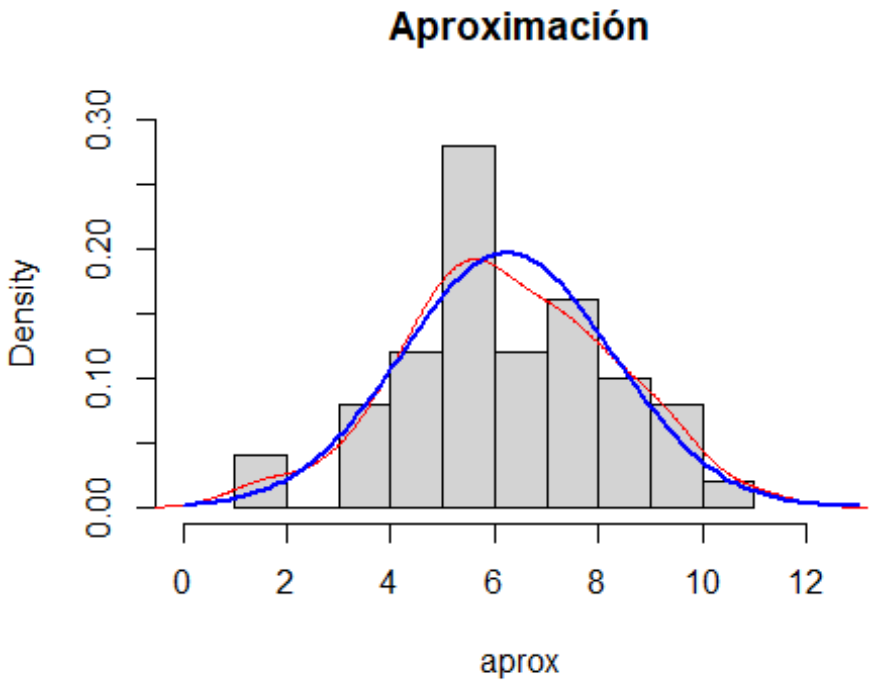
```r
cat("Curtosis", kurtosis(y))
```

```
## Curtosis 0.1193971
```

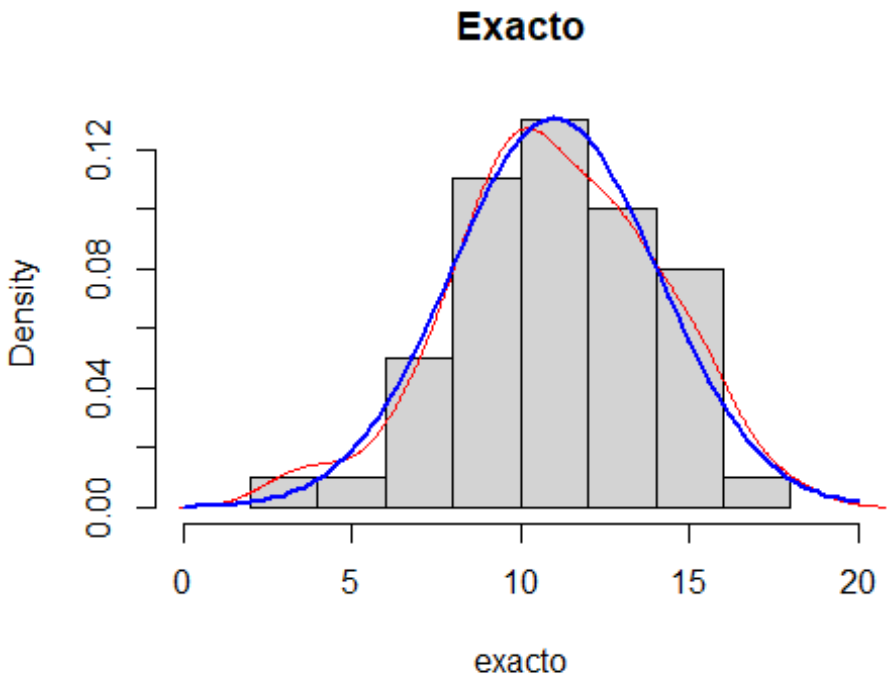
```r
hist(y,freq=FALSE, xlim=c(0,120), ylim=c(0,0.04),main="Original")
lines(density(y),col="red")
curve(dnorm(x,mean=mean(y),sd=sd(y)), from=0, to=120, add=TRUE,
col="blue",lwd=2)
```



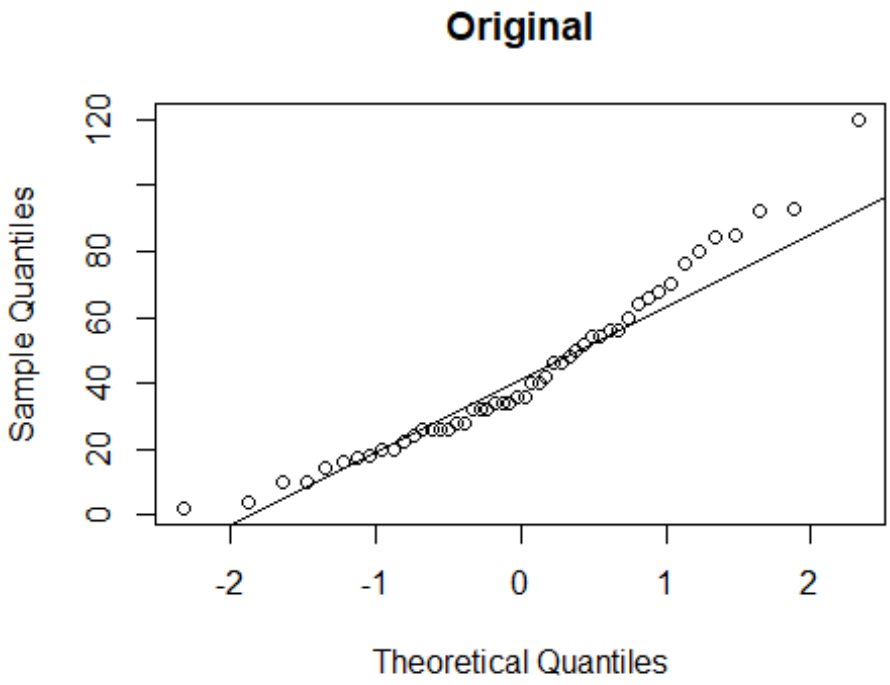

```r
hist(aprox,freq=FALSE, xlim=c(0,13), ylim=c(0,0.30),main="Aproximación")
lines(density(aprox),col="red")
curve(dnorm(x,mean=mean(aprox),sd=sd(aprox)), from=0, to=13, add=TRUE,
col="blue",lwd=2)
```



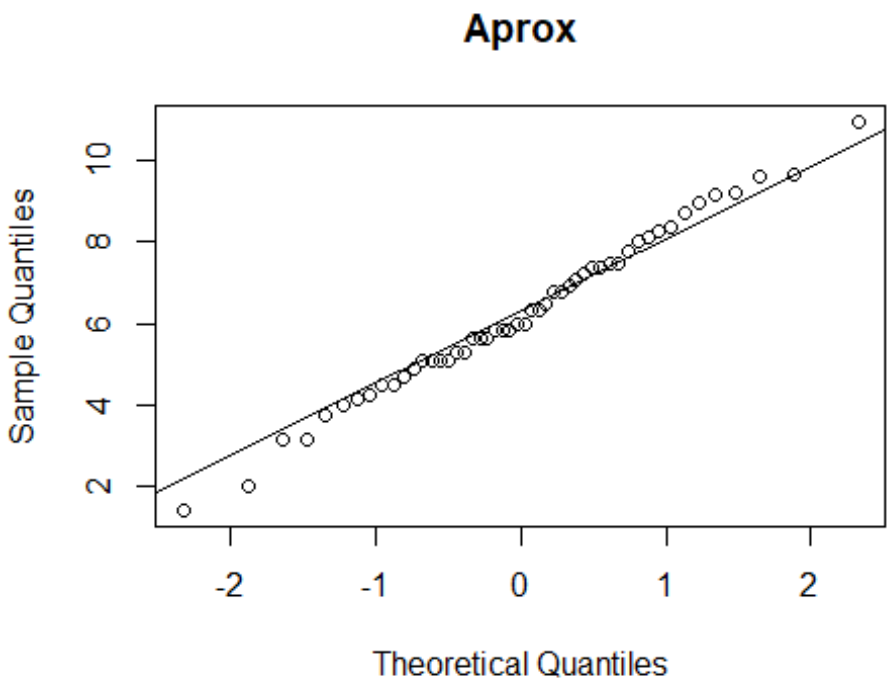

```
hist(exacto,freq=FALSE, xlim=c(0,20), ylim=c(0,0.13),main="Exacto")
lines(density(exacto),col="red")
curve(dnorm(x,mean=mean(exacto),sd=sd(exacto)), from=0, to=20, add=TRUE,
col="blue",lwd=2)
```



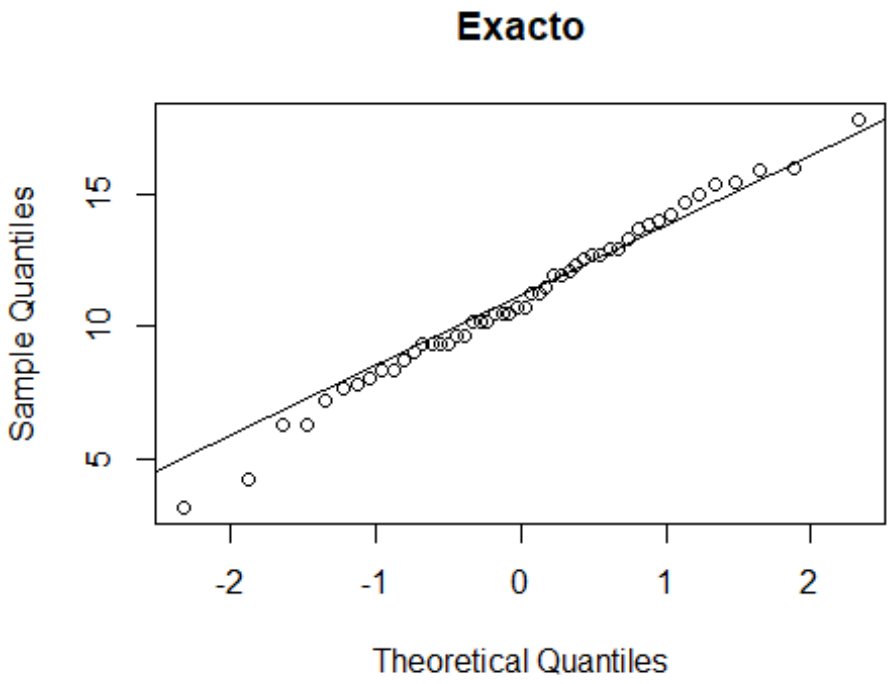

```r
qqnorm(y, main="Original")
qqline(y)
```



```r
qqnorm(aprox, main="Aprox")
qqline(aprox)
```




```r
qqnorm(exacto, main="Exacto")
qqline(exacto)
```




```
print("Original")
```

```
## [1] "Original"
```

```
lillie.test(y)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  y
## D = 0.12675, p-value = 0.04335
```

```
shapiro.test(y)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  y
## W = 0.95144, p-value = 0.0391
```

```
print("Aprox")
```

```
## [1] "Aprox"
```

```
lillie.test(aprox)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
```

```
## data:  aprox
## D = 0.067624, p-value = 0.8218
```

```
shapiro.test(aprox)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  aprox
## W = 0.99347, p-value = 0.9941
```

```
print("Exacto")
```

```
## [1] "Exacto"
```

```
lillie.test(exacto)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  exacto
## D = 0.056406, p-value = 0.9567
```

```
shapiro.test(exacto)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  exacto
## W = 0.99148, p-value = 0.9745
```

Conclusiones

Aunque los dos modelos pasan todas las pruebas de normalidad, yo diria que el que se deberia usar es el modelo exacto ya que tiene mejores pruebas de Lillie y Shapiro en conjunto, además tiene un histograma que más se apega a la apariencia de una distribución normal y finalmente los datos en la gráfica qqplot se apegan un poco más a la linea, dejando una menor cantidad de datos atípicos.

Regresión con modelo transformado

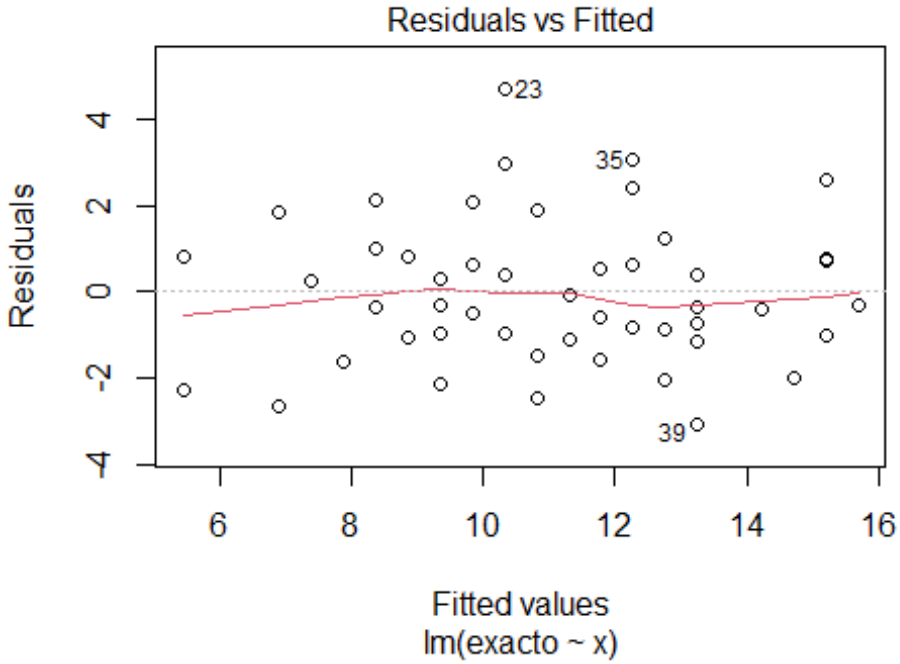
```
modeloTrans = lm(exacto~x)

summary(modeloTrans)
```

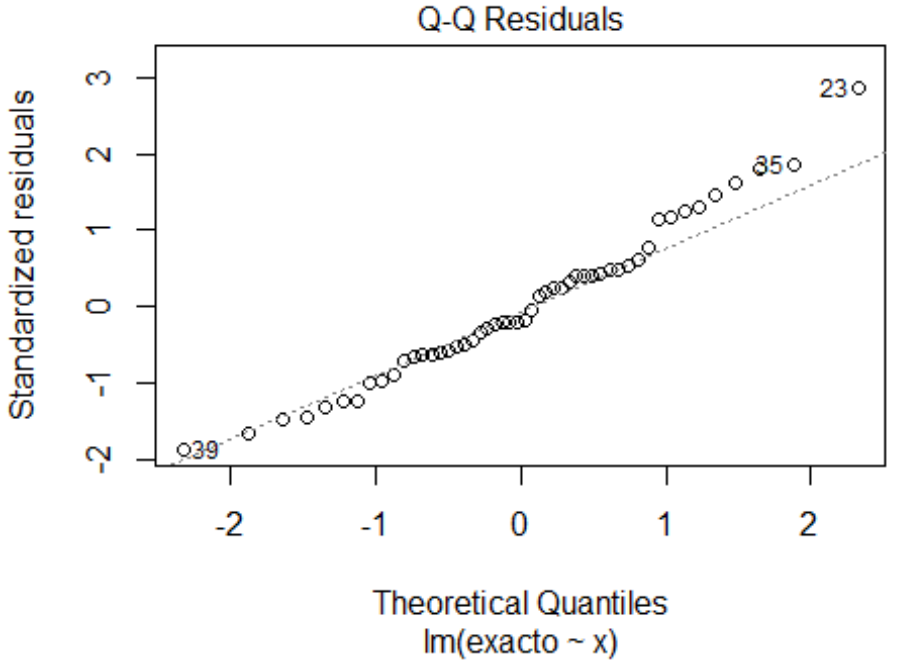
```
##
## Call:
## lm(formula = exacto ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0428 -1.0284 -0.3023  0.7894  4.6776
##
```

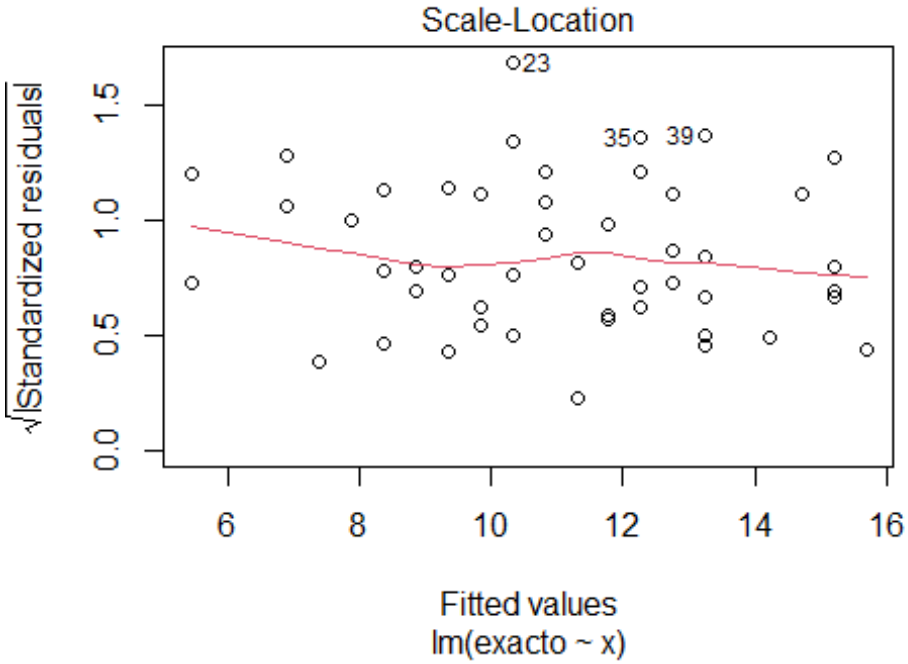
```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.48625    0.72761   4.791 1.63e-05 ***
## x            0.48819    0.04473  10.913 1.34e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.656 on 48 degrees of freedom
## Multiple R-squared:  0.7127, Adjusted R-squared:  0.7068
## F-statistic: 119.1 on 1 and 48 DF,  p-value: 1.341e-14
```

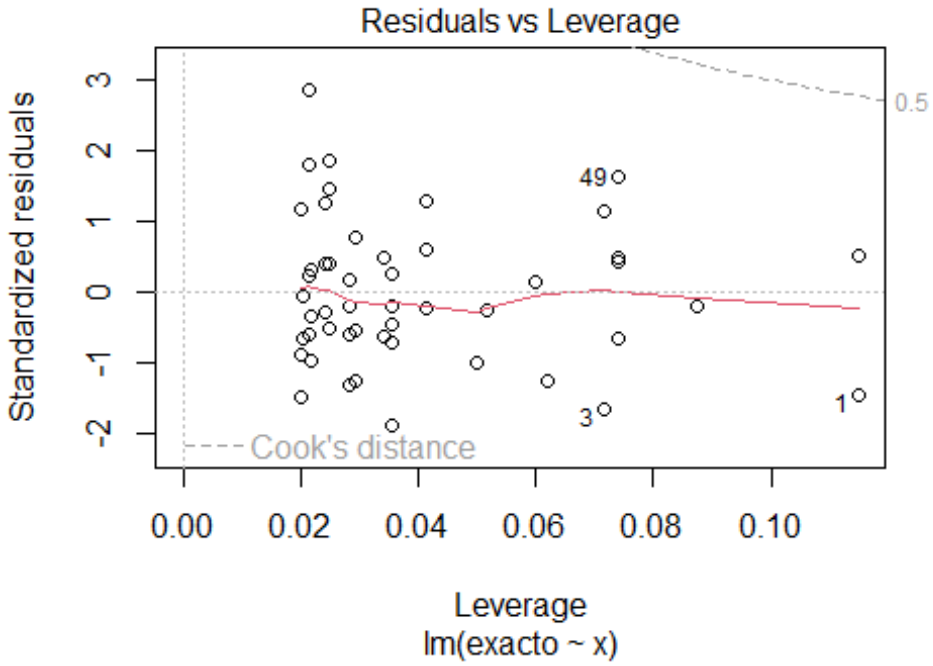
Análisis de normalidad

```
plot(modeloTrans)
```


Residuals vs Fitted
Residuals
Fitted values
lm(exacto ~ x)
23
35
39
-4
-2
0
2
4
6
8
10
12
14
16


Q-Q Residuals
Standardized residuals
Theoretical Quantiles
lm(exacto ~ x)
23
35
39
-2
-1
0
1
2
3





Homocedasticidad

```
library(lmtest)
bptest(modeloTrans)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  modeloTrans
## BP = 0.16186, df = 1, p-value = 0.6874
```

```
gqtest(modeloTrans)
```

```
##
##  Goldfeld-Quandt test
##
## data:  modeloTrans
## GQ = 0.73256, df1 = 23, df2 = 23, p-value = 0.7694
## alternative hypothesis: variance increases from segment 1 to 2
```

Independencia

```
dwtest(modeloTrans)
```

```
##
##  Durbin-Watson test
##
## data:  modeloTrans
## DW = 1.9613, p-value = 0.3874
## alternative hypothesis: true autocorrelation is greater than 0
```

```
bgtest(modeloTrans)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  modeloTrans
## LM test = 5.2401e-06, df = 1, p-value = 0.9982
```

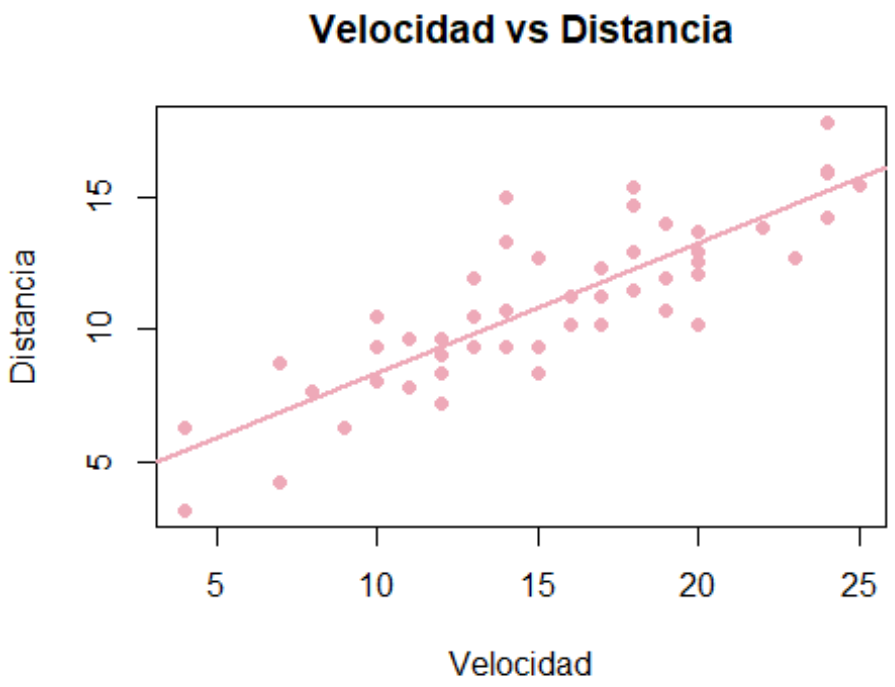
Linealidad

```
library(lmtest)
resettest(modeloTrans)
```

```
##
##  RESET test
##
## data:  modeloTrans
## RESET = 0.70165, df1 = 2, df2 = 46, p-value = 0.501
```

Gráfica

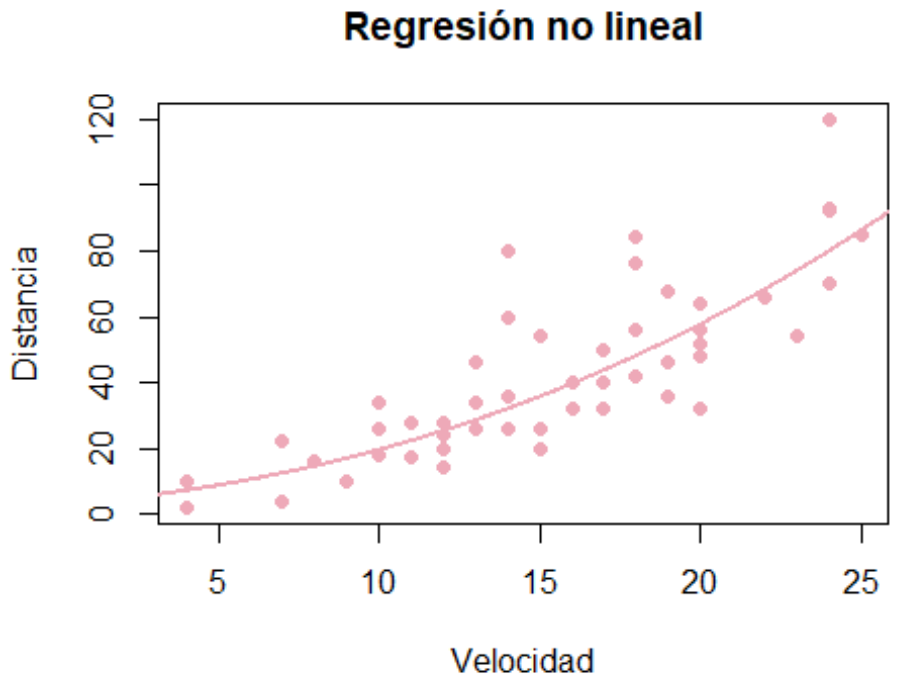
```
plot(x,exacto, col="pink2", main="Velocidad vs Distancia", ylab="Distancia",
xlab = "Velocidad", pch=19)
abline(lm(exacto~x), col="pink2", lwd=2)
```



Modelo no lineal

$$y = (0.2016x + 1.4616)^{\frac{50}{21}}$$

```
Y = function(x){(0.2016*x+1.4616)^(50/21)}



plot(x, y, col = "pink2", pch=19, xlab = "Velocidad", ylab = "Distancia",
main="Regresión no lineal")

x = seq(0,26,0.01)
lines(x, Y(x), col="pink2",lwd=2)
```



Este modelo mejora mucho su significancia, toma en cuenta que mientras mas velocidad la distancia no va a crecer de manera lineal, va a tener cierta exponencialidad. Todos los resultados mejoraron, tenemos un modelo que explica una mayor cantidad de datos y crea menos datos atípicos. Las pruebas de homocedasticidad e independencia mejorar significativamente, podemos ver que los datos son más constantes y que los errores no estan correlacionados.

Conclusión

En efecto el mejor modelo para describir estos datos es el no lineal, simplemente se adapta de mejor manera a los datos y nos proporciona residuos más normalizados con menos datos atípicos. Sin embargo todavía podemos ver algunos datos que no se adaptan al modelo ya que son muy extremos, además es un modelo más tardado de encontrar, tenemos que hacer una transformación de datos, que requiere todo un análisis para comprobar su efectividad antes de poder empezar a hacer el modelo, sin embargo para casos donde la exactitud es una prioridad creo que desarrollar modelos de este tipo son vitales para cumplir los objetivos.