act13

Facundo Colasurdo Caldironi

2024-09-10

El objetivo es encontrar el mejor modelo que relacione la velocidad de los automóviles y las distancias necesarias para detenerse en autos de modelos existentes en 1920 (base de datos car). La ecuación encontrada no sólo deberá ser el mejor modelo obtenido sino también deberá ser el más económico en terminos de la complejidad del modelo.

##Parte 1: Análisis de normalidad Accede a los datos de cars en R (data = cars)

```
m = data(cars)
head(cars)
```

```
##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10
```

```
speed = cars$speed
dist = cars$dist
```

Prueba normalidad univariada de la velocidad y distancia (prueba con dos de las pruebas vistas en clase)

```
library(nortest)
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

```
ad.test(speed)
```

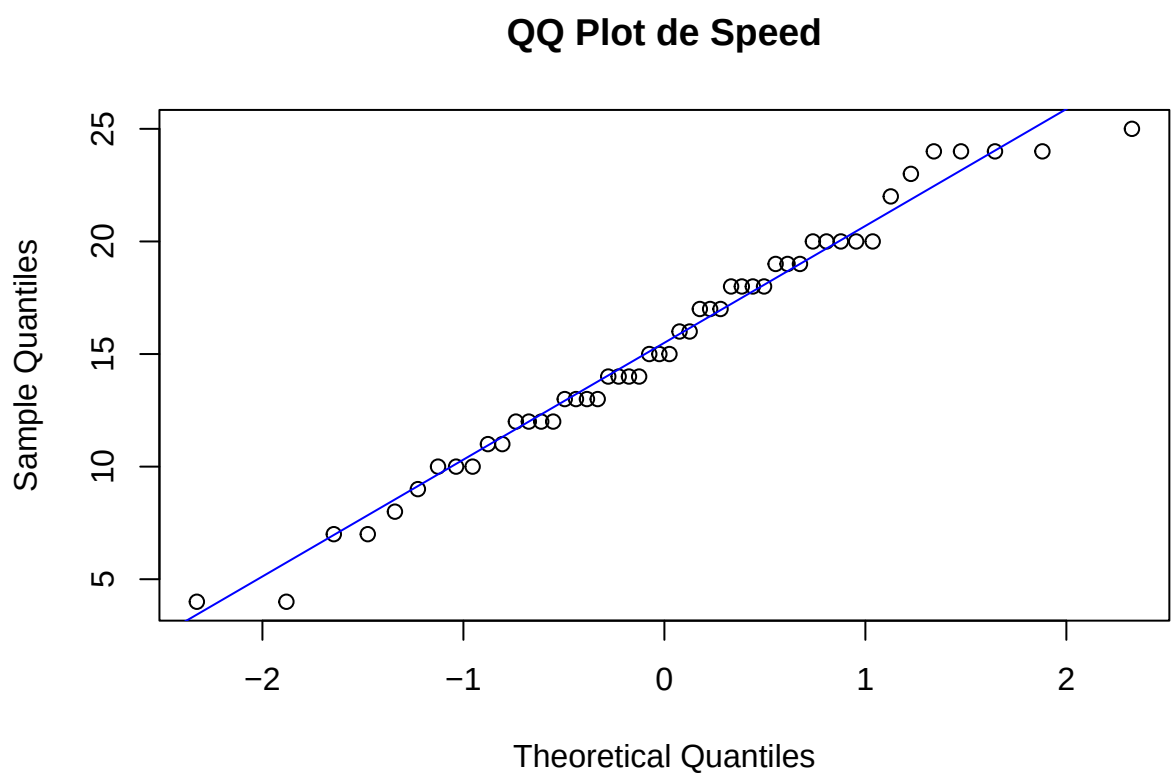
```
##
##  Anderson-Darling normality test
##
## data:  speed
## A = 0.26143, p-value = 0.6927
```

```r
ad.test(dist)
```

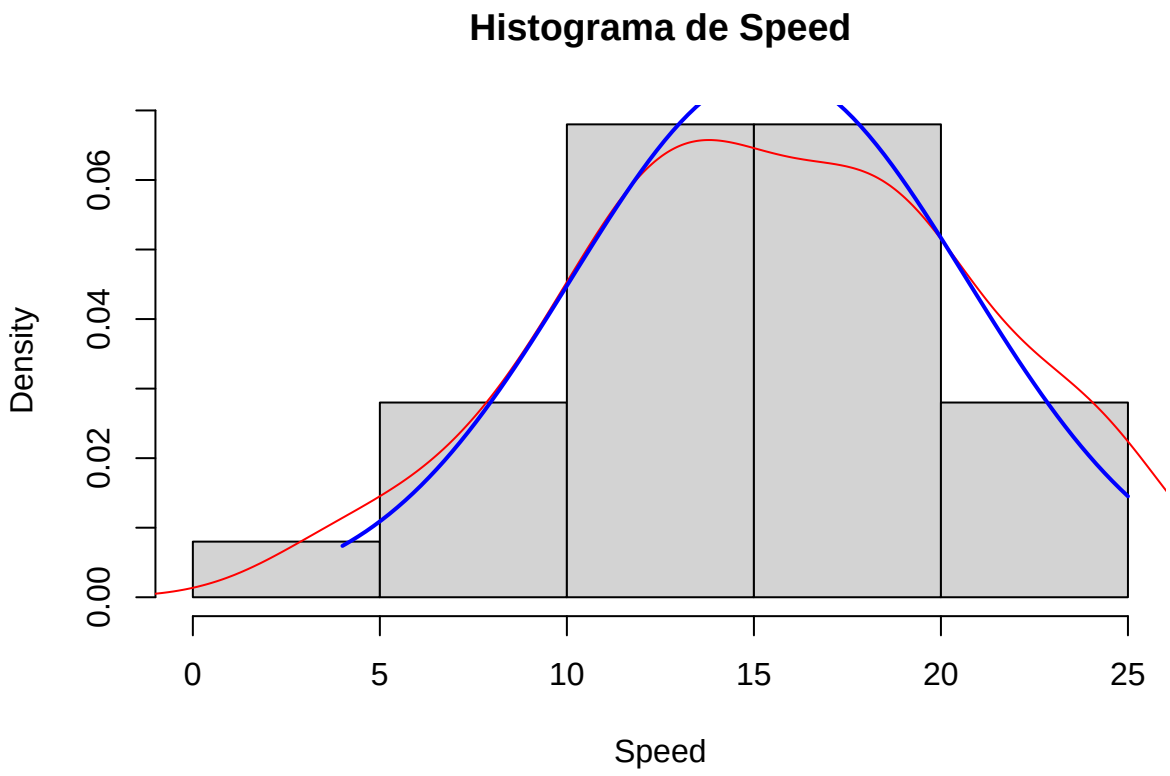
```
##
##  Anderson-Darling normality test
##
## data:  dist
## A = 0.74067, p-value = 0.05021
```

Realiza gráficos que te ayuden a identificar posibles alejamientos de normalidad: los datos y su respectivo QQPlot: qqnorm(datos) y qqline(datos) para cada variable Realiza el histograma y su distribución teórica de probabilidad (sugerencia, adapta el código: hist(datos,freq=FALSE) lines(density(datos),col="red") curve(dnorm(x,mean=mean(datos),sd=sd(datos)), from=min(datos), to=max(datos), add=TRUE, col="blue",lwd=2) Se te sugiere usar par(mfrow=c(1,2)) para graficar el QQ plot y el histograma de una variable en un mismo espacio.

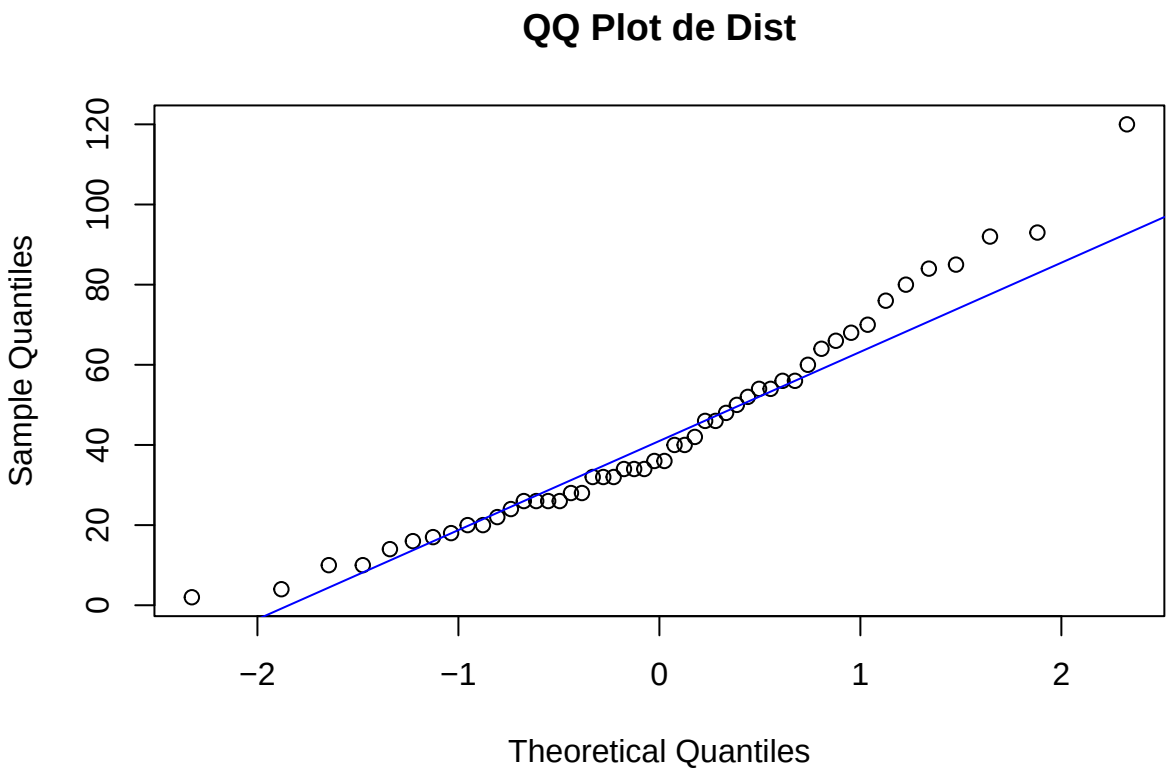
```r
# QQ Plot para la variable speed
qqnorm(cars$speed, main="QQ Plot de Speed")
qqline(cars$speed, col="blue")
```



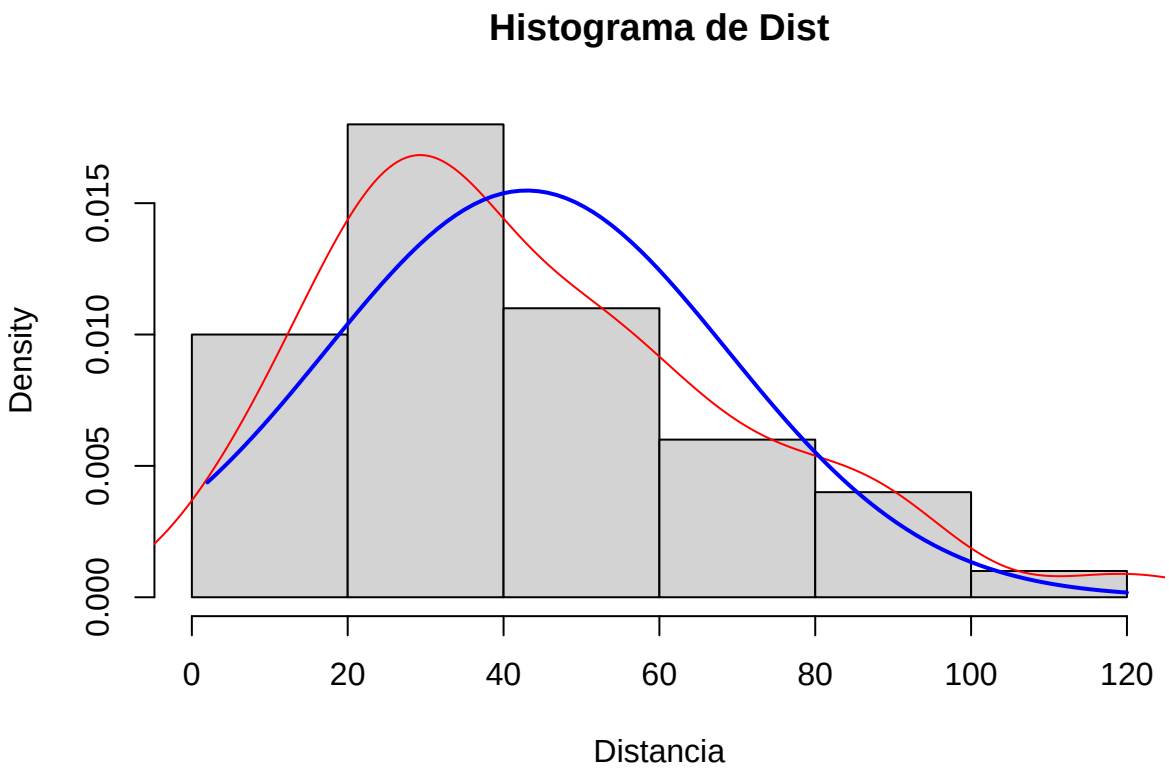

```r
# Histograma para la variable speed con densidad y curva normal teórica
hist(cars$speed, freq=FALSE, main="Histograma de Speed", xlab="Speed")
lines(density(cars$speed), col="red")  # Densidad empírica
curve(dnorm(x, mean=mean(cars$speed), sd=sd(cars$speed)),
      from=min(cars$speed), to=max(cars$speed), add=TRUE, col="blue", lwd=2)  # Curva normal teórica
```




```r
# QQ Plot para la variable dist
qqnorm(cars$dist, main="QQ Plot de Dist")
qqline(cars$dist, col="blue")
```




```r
# Histograma para la variable dist con densidad y curva normal teórica
hist(cars$dist, freq=FALSE, main="Histograma de Dist", xlab="Distancia")
lines(density(cars$dist), col="red")  # Densidad empírica
curve(dnorm(x, mean=mean(cars$dist), sd=sd(cars$dist)),
      from=min(cars$dist), to=max(cars$dist), add=TRUE, col="blue", lwd=2)  # Curva normal teórica
```



QQ Plot de speed: Muestra que los puntos se desvían un poco de la línea de referencia normal, especialmente en los extremos. Esto sugiere que la distribución de dist puede no ser perfectamente normal.

Histograma de speed: La distribución parece más sesgada y presenta colas más pesadas comparadas con la normal teórica, indicando una desviación más notable de la normalidad

QQ Plot de Dist: Muestra que los puntos se desvían un poco de la línea de referencia normal, especialmente en los extremos.

Histograma de dist: Muestra la densidad empírica y la curva normal teórica. La distribución parece más sesgada y presenta colas más pesadas comparadas con la normal teórica, indicando una desviación más notable de la normalidad.

Calcula el coeficiente de sesgo y el coeficiente de curtosis (sugerencia: usar la librería e1071, usar: skeness y kurtosis) para cada variable.

```
library(e1071)


print("Sesgo de speed:")
```

```
## [1] "Sesgo de speed:"
```

```
print(skewness(cars$speed))
```

```
## [1] -0.1105533
```

```r
print("Curtosis de speed:")
```

```
## [1] "Curtosis de speed:"
```

```r
print(kurtosis(cars$speed))
```

```
## [1] -0.6730924
```

```r
print("Sesgo de dist:")
```

```
## [1] "Sesgo de dist:"
```

```r
print(skewness(cars$dist))
```

```
## [1] 0.7591268
```

```r
print("Curtosis de dist:")
```

```
## [1] "Curtosis de dist:"
```

```r
print(kurtosis(cars$dist))
```

```
## [1] 0.1193971
```

Comenta cada gráfico y resultado que hayas obtenido.

Emite una conclusión final sobre la normalidad de los datos. Argumenta basándote en todos los análisis realizados en esta parte. Incluye posibles motivos de alejamiento de normalidad.

La variable speed muestra una distribución que se ajusta bastante bien a la normalidad, tanto en los gráficos como en la prueba de Anderson-Darling. Aunque el histograma y el QQ Plot indican ligeras desviaciones, el p-valor alto en la prueba de normalidad sugiere que la distribución de speed es aproximadamente normal. La variable dist muestra desviaciones más significativas de la normalidad. El QQ Plot y el histograma muestran una distribución con colas más pesadas y una asimetría hacia la derecha. La prueba de Anderson-Darling proporciona evidencia mixta, con un p-valor en el límite que sugiere que la distribución puede no ser completamente normal.

Las posibles razones para el alejamiento de la normalidad incluyen la presencia de valores extremos (outliers) o una distribución inherentemente sesgada y con colas más largas. Estos aspectos pueden deberse a características naturales del proceso que genera estos datos, donde las distancias pueden variar significativamente. Aunque speed parece aproximadamente normal, las ligeras desviaciones pueden deberse a la variabilidad inherente en los datos o a una muestra que no cubre todos los posibles rangos de valores.

##Parte 2: Parte 2: Regresión lineal

Prueba regresión lineal simple entre distancia y velocidad. Usa lm(y~x). Escribe el modelo lineal obtenido. Grafica los datos y el modelo (ecuación) que obtuviste. Analiza significancia del modelo: individual, conjunta y coeficiente de determinación. Usa summary(Modelo)

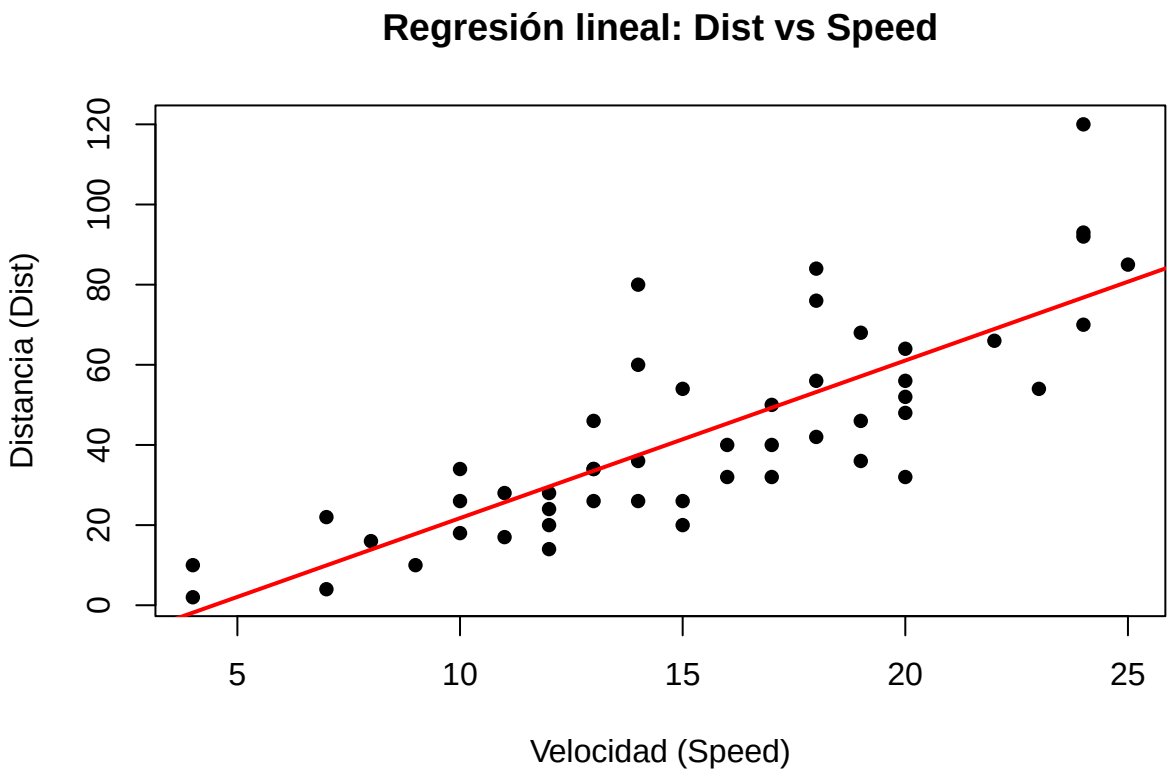
```r
# Ajustar el modelo de regresión lineal simple
modelo <- lm(dist ~ speed, data = cars)

# Mostrar el resumen del modelo para analizar significancia
summary(modelo)
```

```
##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## 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 *
## speed         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
```

```r
# Graficar los datos originales y la línea de regresión ajustada
plot(cars$speed, cars$dist, main="Regresión lineal: Dist vs Speed",
     xlab="Velocidad (Speed)", ylab="Distancia (Dist)",
     pch=16)

# Agregar la línea de regresión al gráfico
abline(modelo, col="red", lwd=2)
```



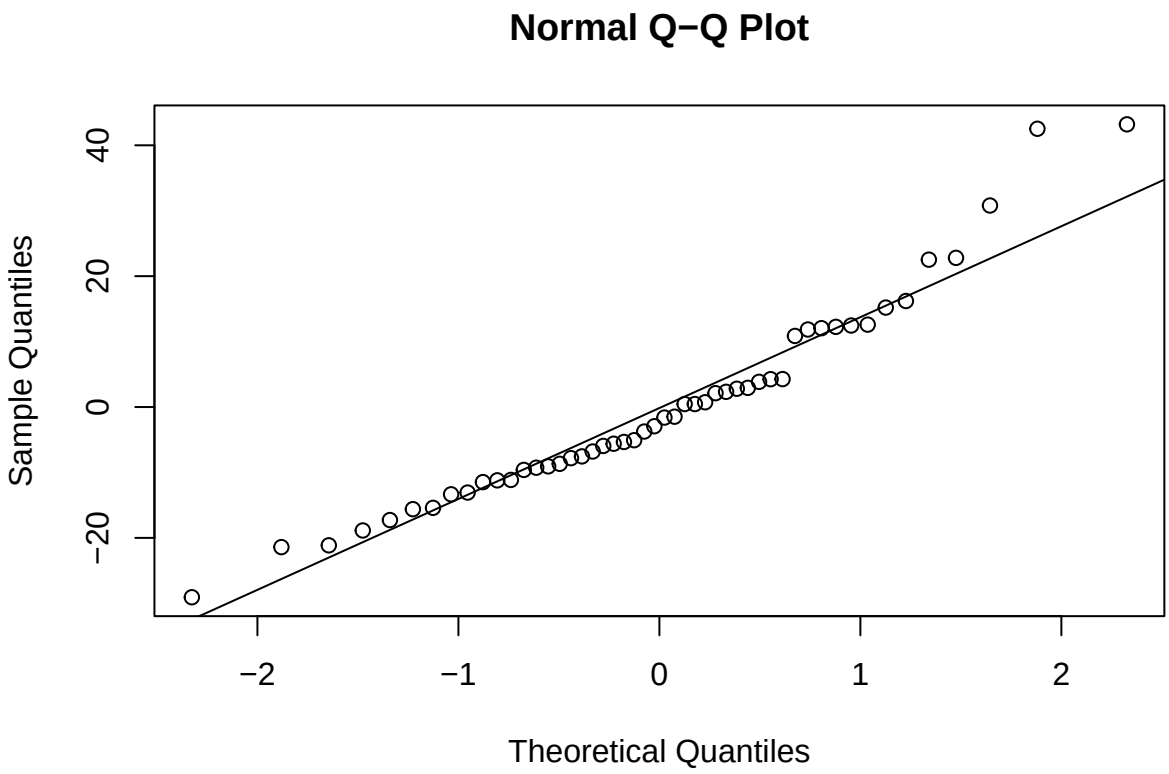
Modelo lineal: dist = -17.57909 + 3.932409 * speed

Analiza validez del modelo. Residuos con media cero Normalidad de los residuos Homocedasticidad, independencia y linealidad. Usa plot(Modelo) para los gráficos y añade pruebas de hipótesis. Grafica los datos y el modelo de la distancia en función de la velocidad.

```
library (nortest)
ad.test (modelo$residuals)
```

```
##
##  Anderson-Darling normality test
##
## data:  modelo$residuals
## A = 0.79406, p-value = 0.0369
```

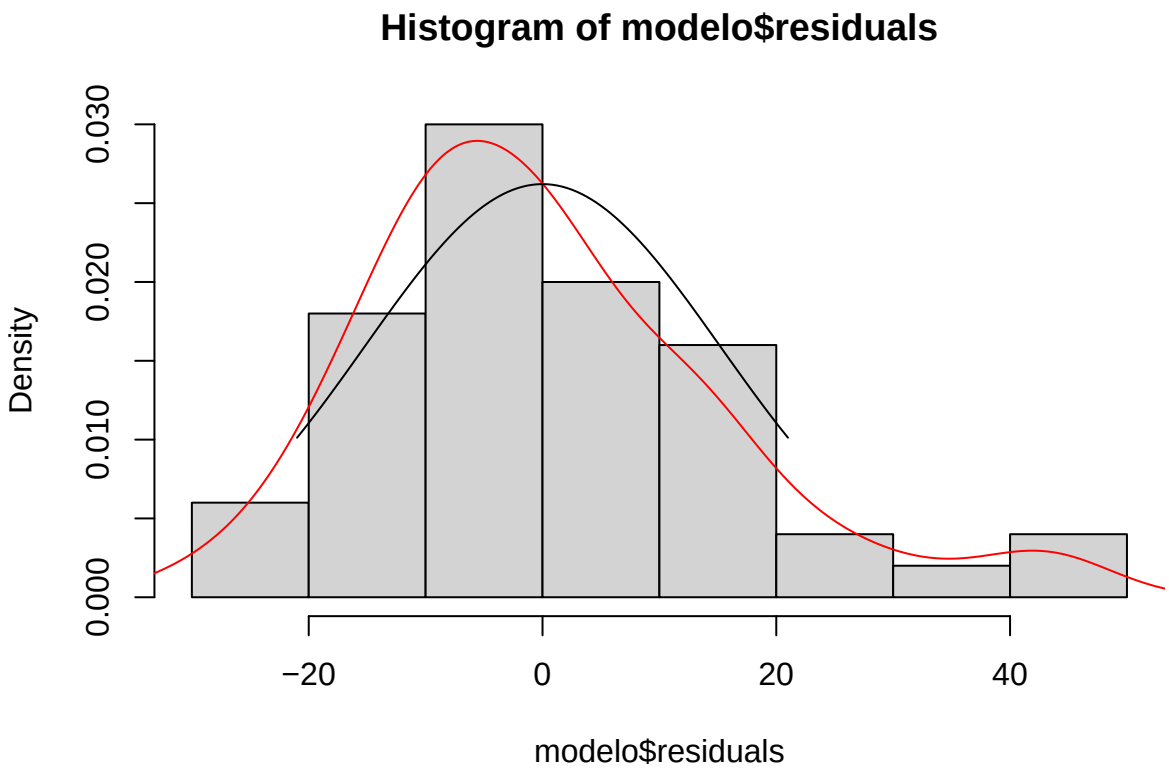
```
qqnorm(modelo$residuals)
qqline (modelo$residuals)
```




```
hist (modelo$residuals, freq=FALSE)
lines (density (modelo$residuals),col="red")

curve (dnorm(x,mean=mean(modelo$residuals),sd=sd(modelo$residuals)), from=-21, to=21, add=TRUE, col="bla
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): "Iwd" is not a graphical
## parameter
```




```
t.test (modelo$residuals)
```

```
##
##  One Sample t-test
##
## data:  modelo$residuals
## t = 1.0315e-16, df = 49, p-value = 1
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -4.326  4.326
## sample estimates:
##    mean of x
## 2.220446e-16
```

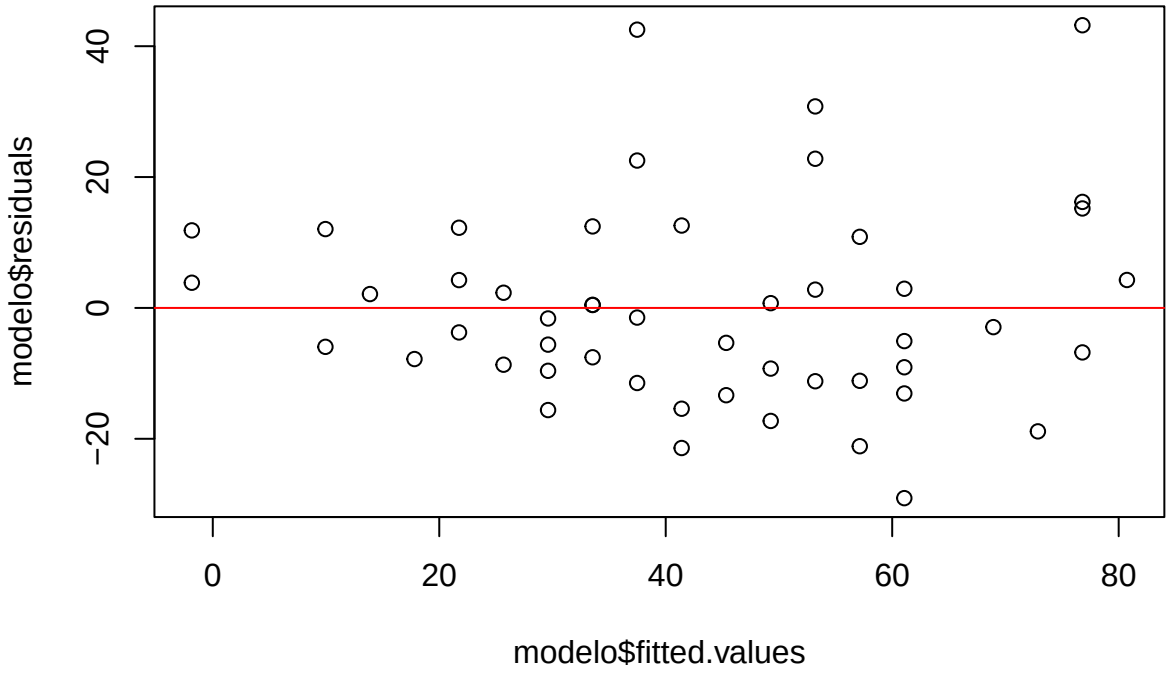
```
plot (modelo$fitted.values, modelo$residuals)
abline(h=0, col="red")

library(lmtest)
```

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

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

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




```
bptest (modelo)
```

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

```
dwtest (modelo)
```

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

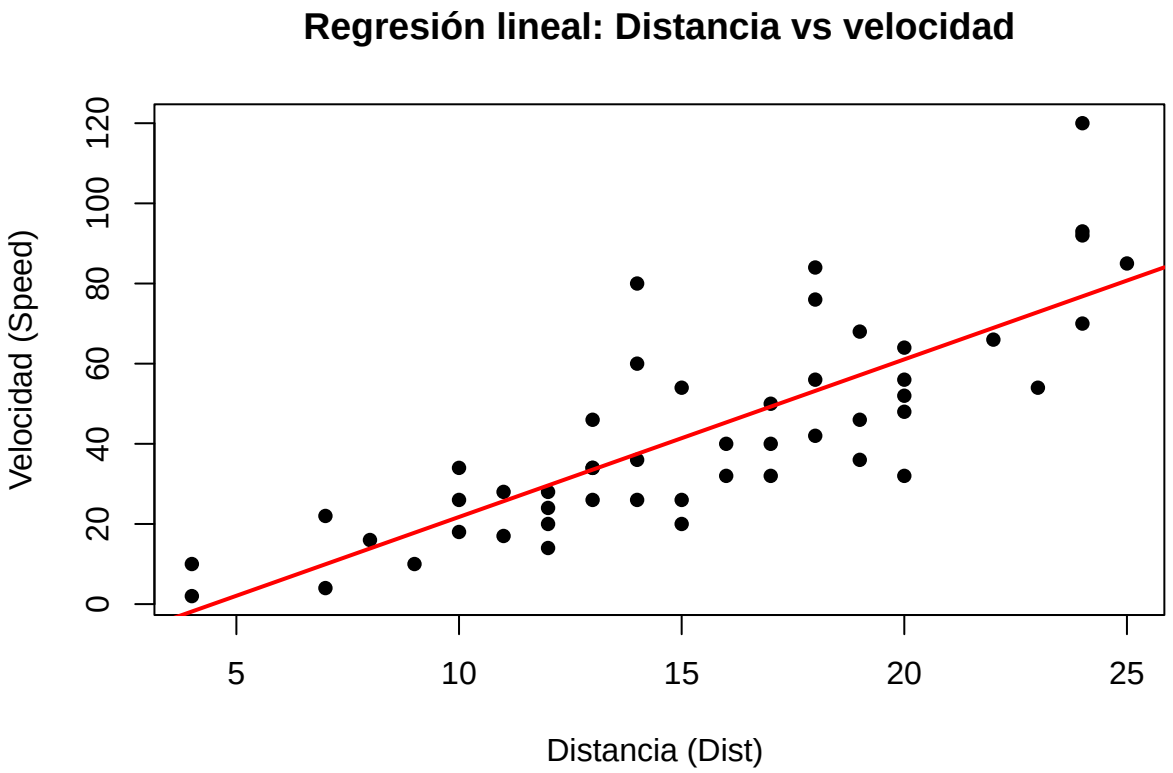
```
resettest (modelo)
```

```
##
```

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

```r
# Graficar los datos originales y la línea de regresión ajustada
plot(cars$speed, cars$dist, main="Regresión lineal: Distancia vs velocidad", xlab="Distancia (Dist) ", y

# Agregar la línea de regresión al gráfico
abline(modelo, col="red", lwd=2)
```



Comenta sobre la idoneidad del modelo en función de su significancia y validez

Los residuos no siguen completamente una distribución normal, lo que puede afectar las pruebas de significancia y los intervalos de confianza, tambien, la posible heterocedasticidad sugiere que la varianza de los errores no es constante a lo largo de los niveles de speed. No hay evidencia fuerte de autocorrelación, pero la prueba sugiere una ligera posibilidad y por ultimo, no hay evidencia significativa de que el modelo esté mal especificado.

##Parte 3: Regresión no lineal

Con el objetivo de probar un modelo no lineal que explique la relación entre la distancia y la velocidad, haz una transformación con la base de datos car que te garantice normalidad en ambas variables (ojo: concentrate solo en la variable que tiene más alejamiento de normalidad).

Encuentra el valor de en la transformación Box-Cox para el modelo lineal: donde Y sea la distancia y X la velocidad. Aprovecha que el comando de boxcox en R te da la oportunidad de trabajar con el modelo líneal: Utiliza: boxcox(lm(Distancia~Velocidad)) si la variable con más alejamiento de normalidad es la distancia Utiliza: boxcox(lm(Velocidad~Distancia)) si la variable con más alejamiento de normalidad es la velocidad

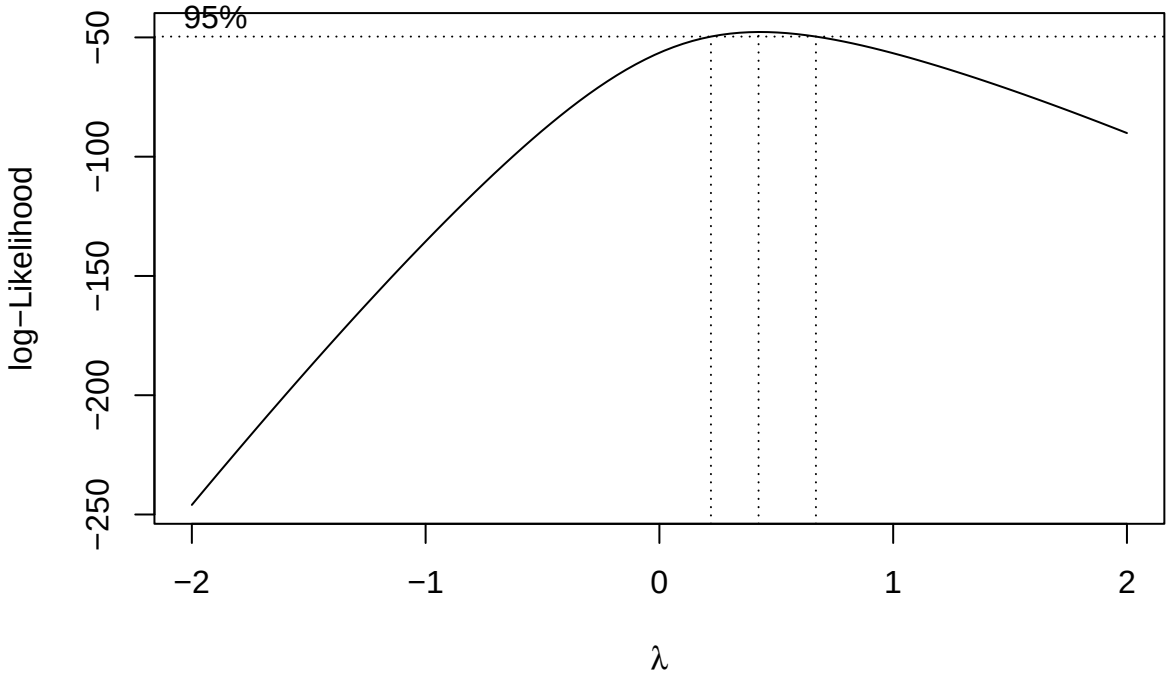
La transformación se hará sobre la variable que usas como dependiente en el comando lm(y~x) Define la transformación exacta y el aproximada de acuerdo con el valor de que encontraste en la transformación de Box y Cox. Escribe las ecuaciones de las dos transformaciones encontradas.

DISTANCIA es la que tiene mayor alejamiento de la normalidad

```
# Cargar el paquete MASS
library(MASS)

# Ajustar un modelo de regresión lineal: Distancia ~ Velocidad
modeloDist <- lm(dist ~ speed, data = cars)

# Aplicar la transformación Box-Cox sobre el modelo
boxcox_transformDist <- boxcox(modeloDist)
```




```
# Ver el valor de lambda que maximiza la verosimilitud
lambda_optimoDist <- boxcox_transformDist$x[which.max(boxcox_transformDist$y)]
cat("El valor óptimo de lambda es:", lambda_optimoDist, "\n")
```

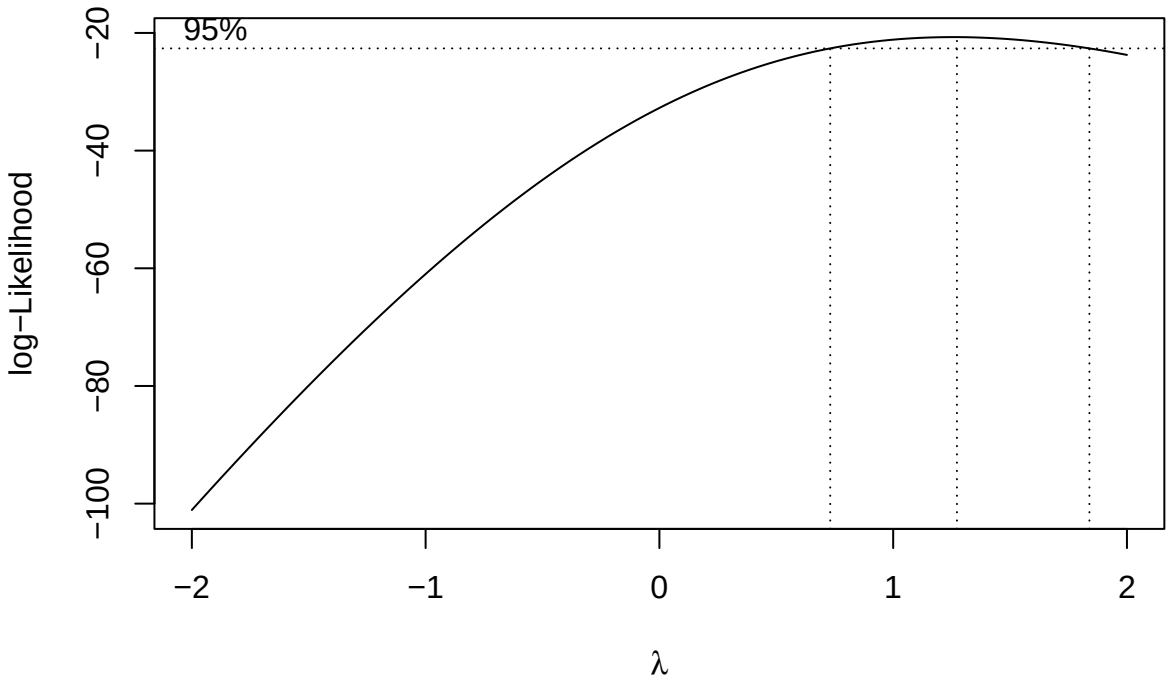
```
## El valor óptimo de lambda es: 0.4242424
```

```
# Cargar el paquete MASS
library(MASS)

# Ajustar un modelo de regresión lineal: velocidad ~ Distancia
```

```r
modeloVel <- lm(speed ~ dist, data = cars)

# Aplicar la transformación Box-Cox sobre el modelo
boxcox_transform1 <- boxcox(modeloVel)
```




```r
# Ver el valor de lambda que maximiza la verosimilitud
lambda_optimoVEL <- boxcox_transform1$x[which.max(boxcox_transform1$y)]
cat("El valor óptimo de lambda es:", lambda_optimoVEL, "\n")
```

```
## El valor óptimo de lambda es: 1.272727
```

Distancia: $Y(\lambda) = log(y)$

Velocidad $Y(\lambda) = \frac{Y^{0}.4242424-1}{0.4242424}$

Analiza la normalidad de las transformaciones obtenidas. Utiliza como argumento de normalidad: Compara las medidas: sesgo y curtosis. Obten el histograma de los 2 modelos obtenidos (exacto y aproximado) y los datos originales. Realiza algunas pruebas de normalidad para los datos transformados.

```r
# Aplicar las transformaciones
transf_dist <- log(cars$dist)  # Transformación exacta para Distancia
transf_speed <- (cars$speed^lambda_optimoVEL - 1) / lambda_optimoVEL  # Transformación Box-Cox para Vel

# Comparación de medidas: Sesgo y Curtosis
original_dist_skew <- skewness(cars$dist)
```

```r
original_dist_kurt <- kurtosis(cars$dist)
transf_dist_skew <- skewness(transf_dist)
transf_dist_kurt <- kurtosis(transf_dist)


cat("Para Distancia:\n")
```

```
## Para Distancia:
```

```r
cat("Sesgo original:", original_dist_skew, "\n")
```

```
## Sesgo original: 0.7591268
```

```r
cat("Curtosis original:", original_dist_kurt, "\n")
```

```
## Curtosis original: 0.1193971
```

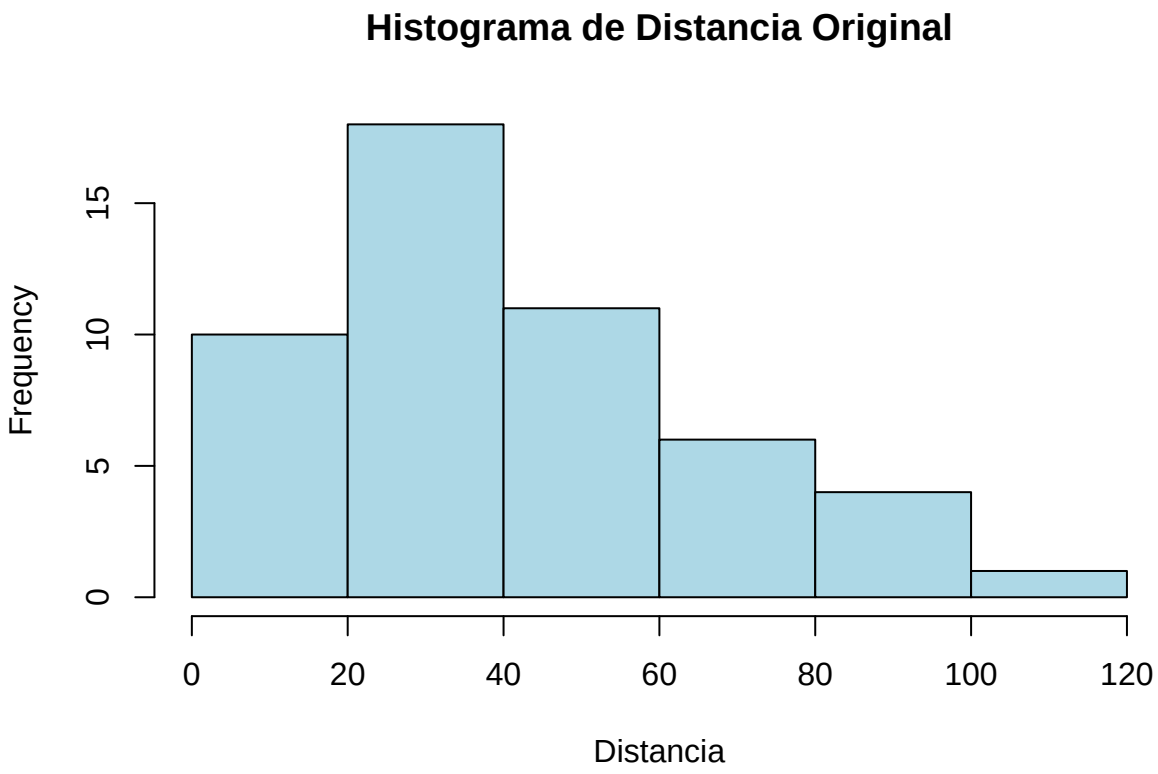
```r
cat("Sesgo transformado:", transf_dist_skew, "\n")
```

```
## Sesgo transformado: -1.302538
```

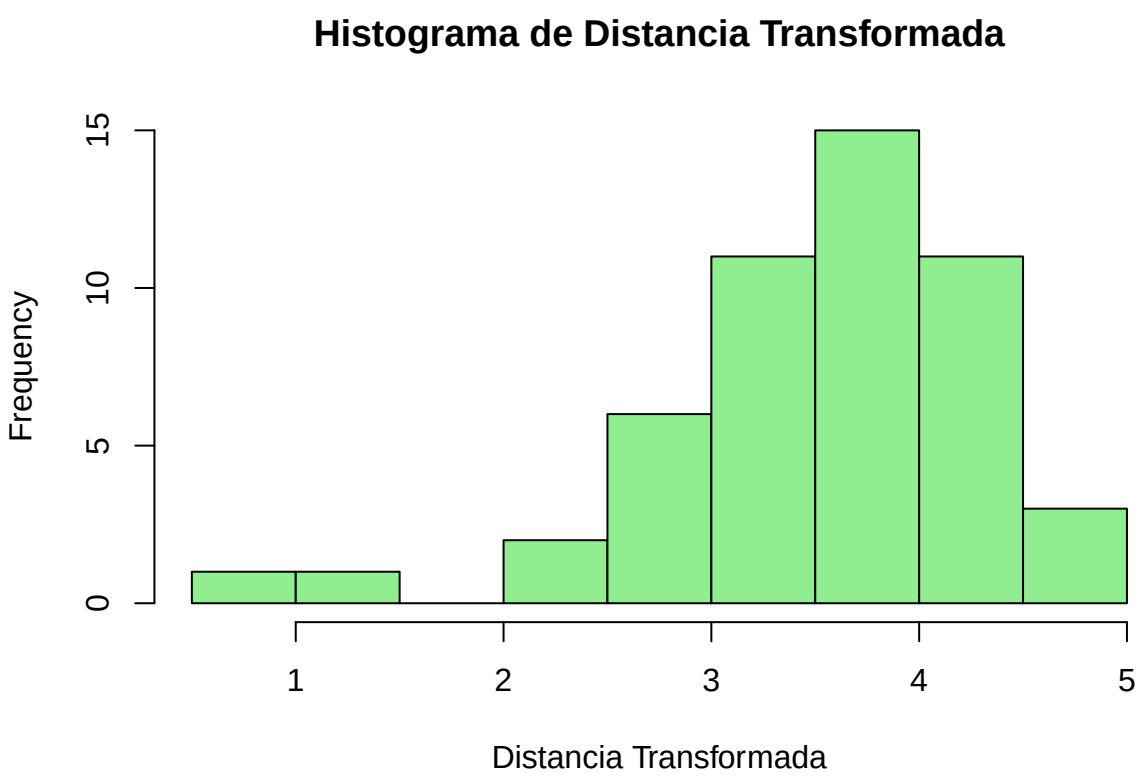
```r
cat("Curtosis transformado:", transf_dist_kurt, "\n")
```

```
## Curtosis transformado: 2.543008
```

```r
hist(cars$dist, main = "Histograma de Distancia Original", xlab = "Distancia", col = "lightblue")
```




```
hist(transf_dist, main = "Histograma de Distancia Transformada", xlab = "Distancia Transformada", col =
```




```
# Pruebas de normalidad
shapiro_test_dist <- shapiro.test(cars$dist)
shapiro_test_transf_dist <- shapiro.test(transf_dist)


cat("Prueba de normalidad para Distancia original:\n")
```

```
## Prueba de normalidad para Distancia original:
```

```
print(shapiro_test_dist)
```

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

```
cat("Prueba de normalidad para Distancia transformada:\n")
```

```
## Prueba de normalidad para Distancia transformada:
```

```
print(shapiro_test_transf_dist)
```

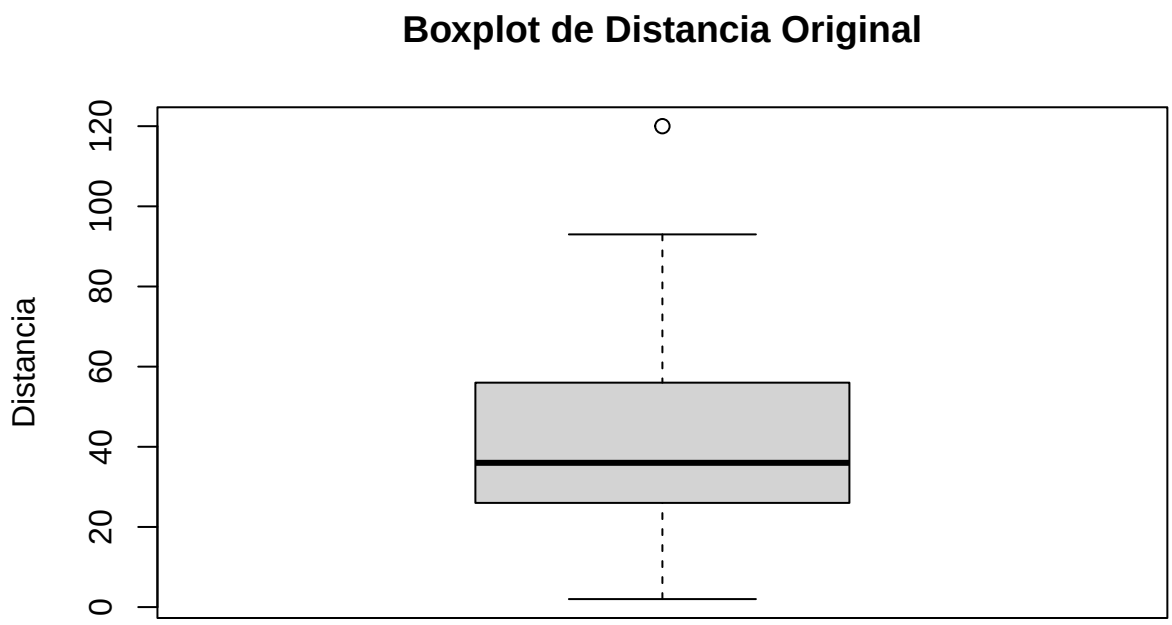
```
##
##  Shapiro-Wilk normality test
##
## data:  transf_dist
## W = 0.91024, p-value = 0.001066
```

Detecta anomalías y corrige tu base de datos tranformado (datos atípicos, ceros anámalos, etc): solo en caso de no tener normalidad en las transformaciones. En caso de corrección de los datos por anomalías, vuelve a buscar la para tus nuevos datos.

```
# Revisar datos atípicos y ceros anómalos
summary(cars$dist)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00   26.00   36.00   42.98   56.00  120.00
```

```
boxplot(cars$dist, main = "Boxplot de Distancia Original", ylab = "Distancia")
```



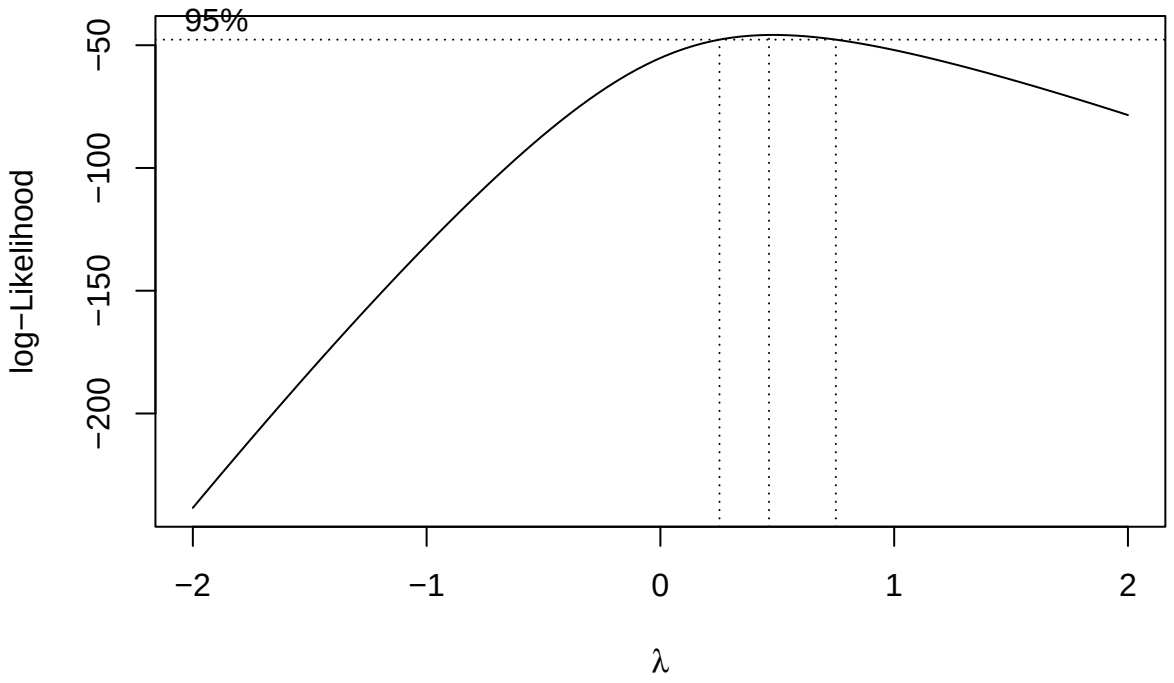

```
# Encontrar valores atípicos
quantiles <- quantile(cars$dist, probs = c(0.25, 0.75))
IQR <- quantiles[2] - quantiles[1]
lower_bound <- quantiles[1] - 1.5 * IQR
upper_bound <- quantiles[2] + 1.5 * IQR
```

```r
# Identificar valores atípicos
anomalous_dist <- cars$dist[cars$dist < lower_bound | cars$dist > upper_bound]
cat("Valores atípicos en Distancia:", anomalous_dist, "\n")
```

```
## Valores atípicos en Distancia: 120
```

```r
# Filtrar datos atípicos y ceros anómalos
cleaned_data <- subset(cars, dist >= lower_bound & dist <= upper_bound)

# Aplicar la transformación Box-Cox nuevamente
modeloDist_cleaned <- lm(dist ~ speed, data = cleaned_data)
boxcox_transformDist_cleaned <- boxcox(modeloDist_cleaned)
```




```r
lambda_optimoDist_cleaned <- boxcox_transformDist_cleaned$x[which.max(boxcox_transformDist_cleaned$y)]
cat("El nuevo valor óptimo de lambda para Distancia (datos limpios) es:", lambda_optimoDist_cleaned, "\n
```

```
## El nuevo valor óptimo de lambda para Distancia (datos limpios) es: 0.4646465
```

```r
# Cargar los paquetes necesarios
library(MASS)

# Aplicar las transformaciones a los datos limpios
transf_dist_cleaned <- (cleaned_data$dist^0.4646465 - 1) / 0.4646465  # Transformación Box-Cox con nuev
```

```r
# Comparación de medidas: Sesgo y Curtosis para datos limpios
original_dist_cleaned_skew <- skewness(cleaned_data$dist)
original_dist_cleaned_kurt <- kurtosis(cleaned_data$dist)
transf_dist_cleaned_skew <- skewness(transf_dist_cleaned)
transf_dist_cleaned_kurt <- kurtosis(transf_dist_cleaned)

cat("Para Distancia (Datos Limpiados):\n")
```

```
## Para Distancia (Datos Limpiados):
```

```r
cat("Sesgo original:", original_dist_cleaned_skew, "\n")
```

```
## Sesgo original: 0.5002547
```

```r
cat("Curtosis original:", original_dist_cleaned_kurt, "\n")
```

```
## Curtosis original: -0.6409882
```

```r
cat("Sesgo transformado:", transf_dist_cleaned_skew, "\n")
```

```
## Sesgo transformado: -0.2381771
```

```r
cat("Curtosis transformado:", transf_dist_cleaned_kurt, "\n")
```

```
## Curtosis transformado: -0.3566407
```

```r
# Histogramas
par(mfrow=c(2,2))  # Configura la ventana gráfica para mostrar 2x2 gráficos
hist(cleaned_data$dist, main = "Histograma de Distancia Original (Limpio)", xlab = "Distancia", col = "
hist(transf_dist_cleaned, main = "Histograma de Distancia Transformada (Limpio)", xlab = "Distancia Tra

# Pruebas de normalidad
shapiro_test_dist_cleaned <- shapiro.test(cleaned_data$dist)
shapiro_test_transf_dist_cleaned <- shapiro.test(transf_dist_cleaned)

cat("Prueba de normalidad para Distancia original (Datos Limpios):\n")
```

```
## Prueba de normalidad para Distancia original (Datos Limpios):
```

```r
print(shapiro_test_dist_cleaned)
```

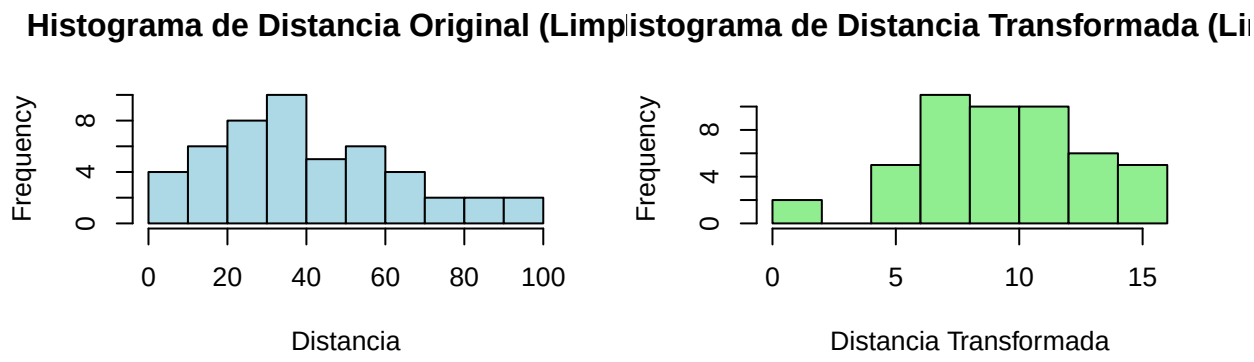
```
##
##  Shapiro-Wilk normality test
##
## data:  cleaned_data$dist
## W = 0.95909, p-value = 0.08692
```

```r
cat("Prueba de normalidad para Distancia transformada (Datos Limpios):\n")
```

```
## Prueba de normalidad para Distancia transformada (Datos Limpios):
```

```r
print(shapiro_test_transf_dist_cleaned)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  transf_dist_cleaned
## W = 0.98369, p-value = 0.7253
```



Concluye sobre las dos transformaciones realizadas: Define la mejor transformación de los datos de acuerdo a las características de las dos transformaciones encontradas (exacta o aproximada). Toman en cuenta la normalidad de los datos y la economía del modelo.

La transformacion aproximada, ya que tiene un mejor ajuste de normalidad

Con la mejor transformación (punto 2), realiza la regresión lineal simple entre la mejor transformación (exacta o aproximada) y la variable velocidad:

Escribe el modelo lineal para la transformación.

```r
# Cargar el paquete necesario
library(MASS)
```

```r
# Aplicar la transformación Box-Cox a Distancia con el valor de lambda óptimo
lambda_optimo <- 0.4646465
transf_dist_cleaned <- (cars$dist^lambda_optimo - 1) / lambda_optimo

# Ajustar el modelo de regresión lineal: Distancia Transformada ~ Velocidad
modelo_regresion <- lm(transf_dist_cleaned ~ speed, data = cars)

# Resumen del modelo
summary(modelo_regresion)
```

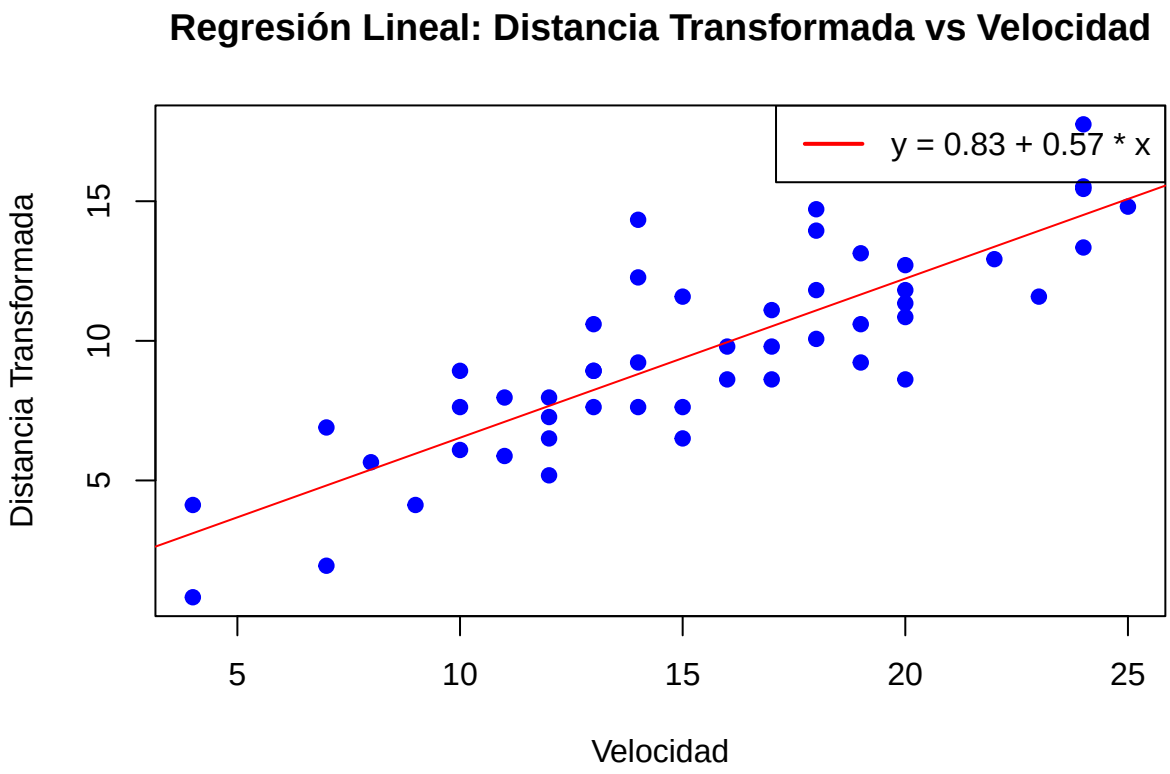
```
##
## Call:
## lm(formula = transf_dist_cleaned ~ speed, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6111 -1.2135 -0.3362  0.9916  5.5251
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.83012    0.85271   0.974    0.335
## speed        0.56997    0.05243  10.872 1.53e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.94 on 48 degrees of freedom
## Multiple R-squared:  0.7112, Adjusted R-squared:  0.7052
## F-statistic: 118.2 on 1 and 48 DF,  p-value: 1.528e-14
```

Grafica los datos y el modelo lineal (ecuación) de la transformación elegida vs velocidad.

```r
# Graficar los datos
plot(cars$speed, transf_dist_cleaned, main = "Regresión Lineal: Distancia Transformada vs Velocidad",
     xlab = "Velocidad", ylab = "Distancia Transformada", pch = 19, col = "blue")

# Añadir la línea de regresión
abline(modelo_regresion, col = "red")

# Mostrar la ecuación del modelo en la gráfica
eq <- paste("y =", round(coef(modelo_regresion)[1], 3), "+", round(coef(modelo_regresion)[2], 3), "* x")
legend("topright", legend = eq, col = "red", lwd = 2)
```



Analiza significancia del modelo (individual, conjunta y coeficiente de correlación)

```
# Resumen del modelo para significancia
summary(modelo_regresion)
```

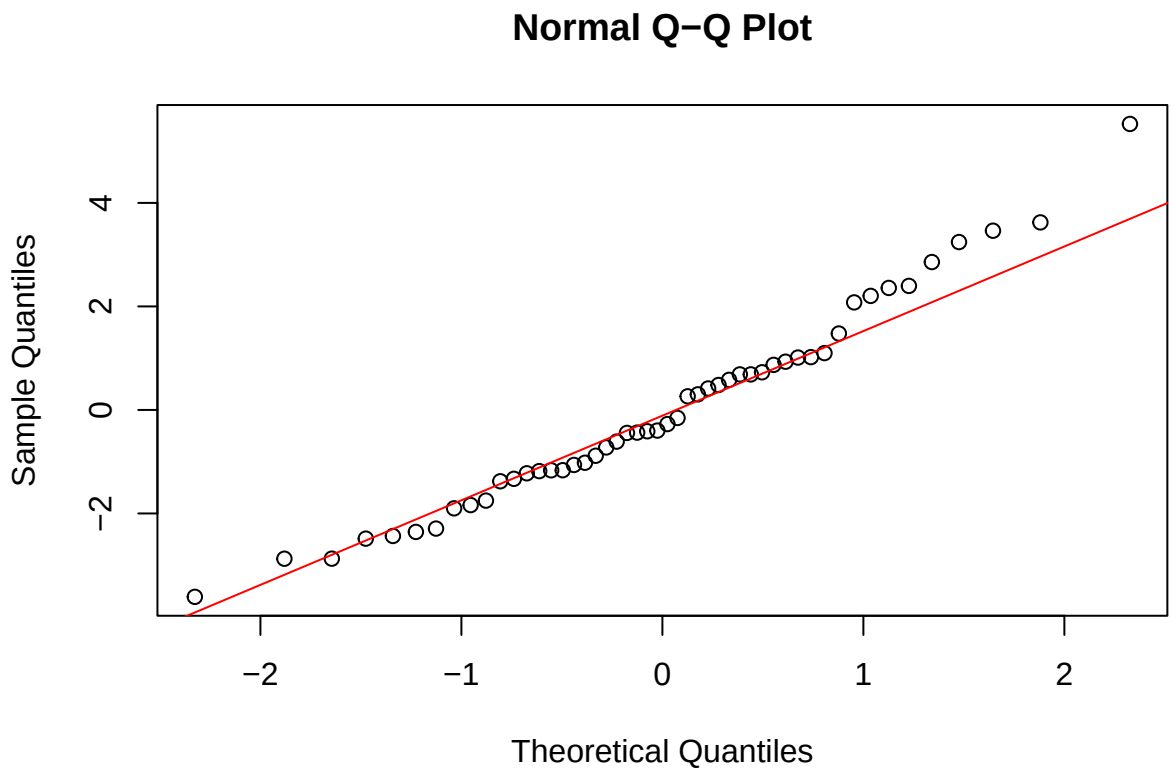
```
##
## Call:
## lm(formula = transf_dist_cleaned ~ speed, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6111 -1.2135 -0.3362  0.9916  5.5251
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.83012    0.85271   0.974    0.335
## speed        0.56997    0.05243  10.872 1.53e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.94 on 48 degrees of freedom
## Multiple R-squared:  0.7112, Adjusted R-squared:  0.7052
## F-statistic: 118.2 on 1 and 48 DF,  p-value: 1.528e-14
```

```r
# Calcular el coeficiente de correlación
cor(cars$speed, transf_dist_cleaned)
```

```
## [1] 0.8433221
```

Analiza validez del modelo: normalidad de los residuos, homocedasticidad e independencia. Indica si hay candidatos a datos atípicos o influyentes en la regresión. Usa plot(Modelo) para los gráficos y añade pruebas de hipótesis.

```r
# Gráfico Q-Q de los residuos
qqnorm(residuals(modelo_regresion))
qqline(residuals(modelo_regresion), col = "red")
```




```r
# Prueba de normalidad para los residuos
shapiro_test_residuos <- shapiro.test(residuals(modelo_regresion))
cat("Prueba de normalidad para los residuos:\n")
```

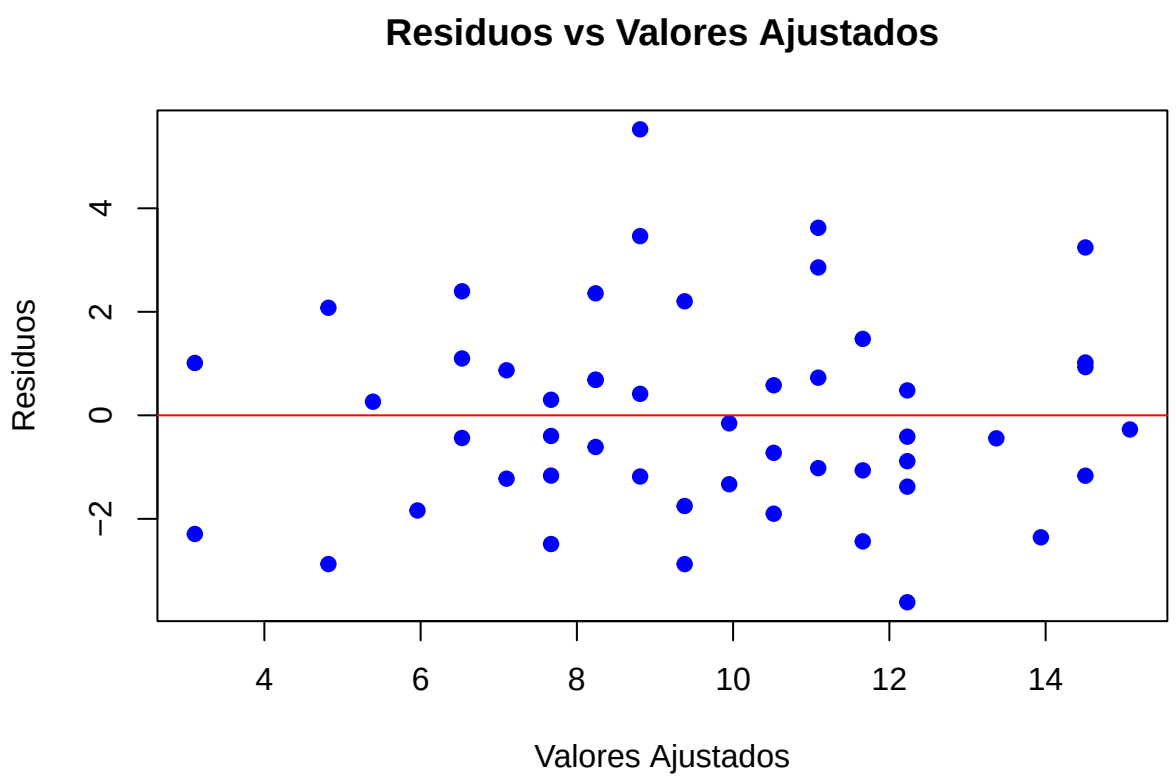
```
## Prueba de normalidad para los residuos:
```

```r
print(shapiro_test_residuos)
```

```
##
##  Shapiro-Wilk normality test
```

```
##
## data:  residuals(modelo_regresion)
## W = 0.97474, p-value = 0.3567
```

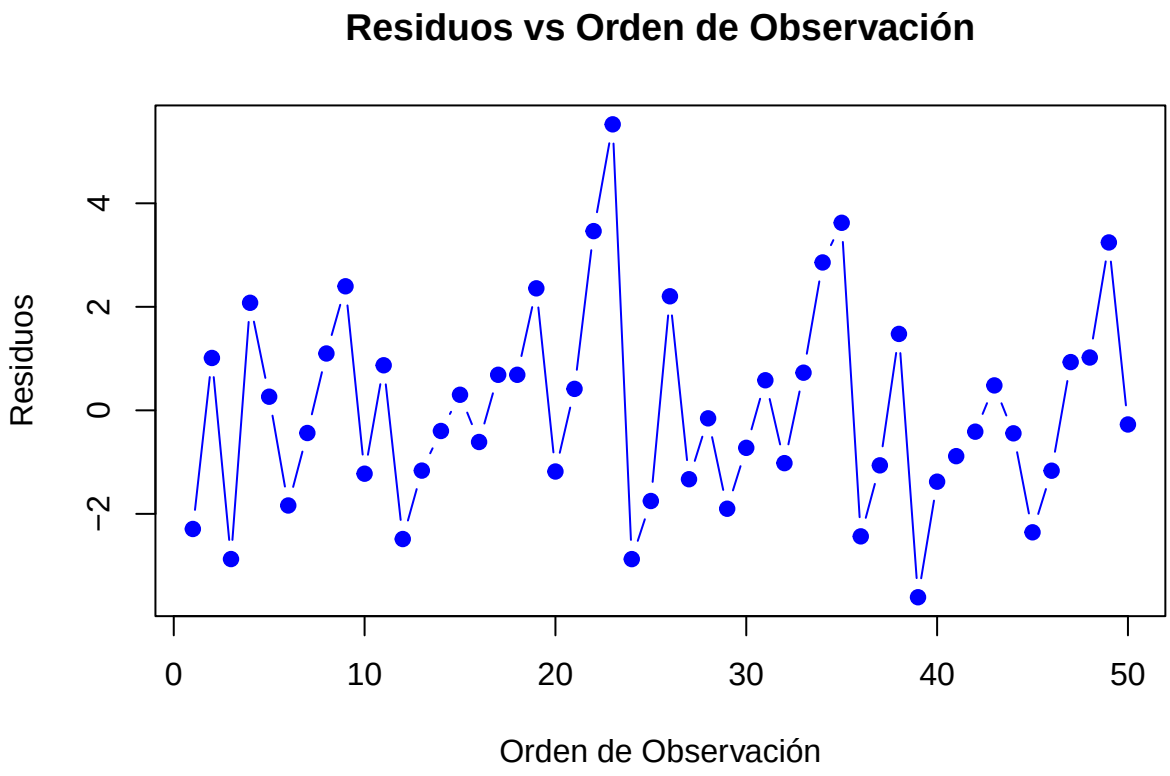
```r
# Gráfico de residuos vs valores ajustados
plot(fitted(modelo_regresion), residuals(modelo_regresion),  main = "Residuos vs Valores Ajustados", xla
abline(h = 0, col = "red")
```




```r
# Prueba de Breusch-Pagan para homocedasticidad
library(lmtest)
bptest(modelo_regresion)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  modelo_regresion
## BP = 0.011715, df = 1, p-value = 0.9138
```

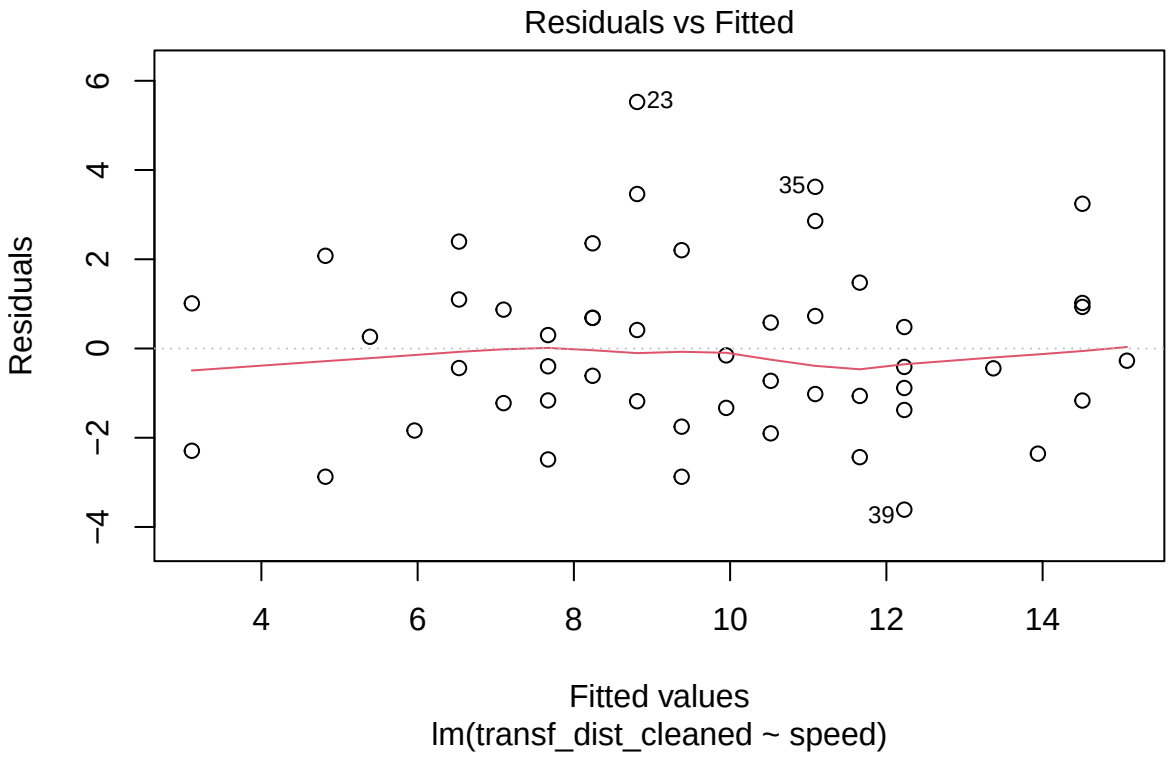
```r
# Gráfico de residuos vs orden de observación
plot(residuals(modelo_regresion), type = "b", main = "Residuos vs Orden de Observación", xlab = "Orden
```

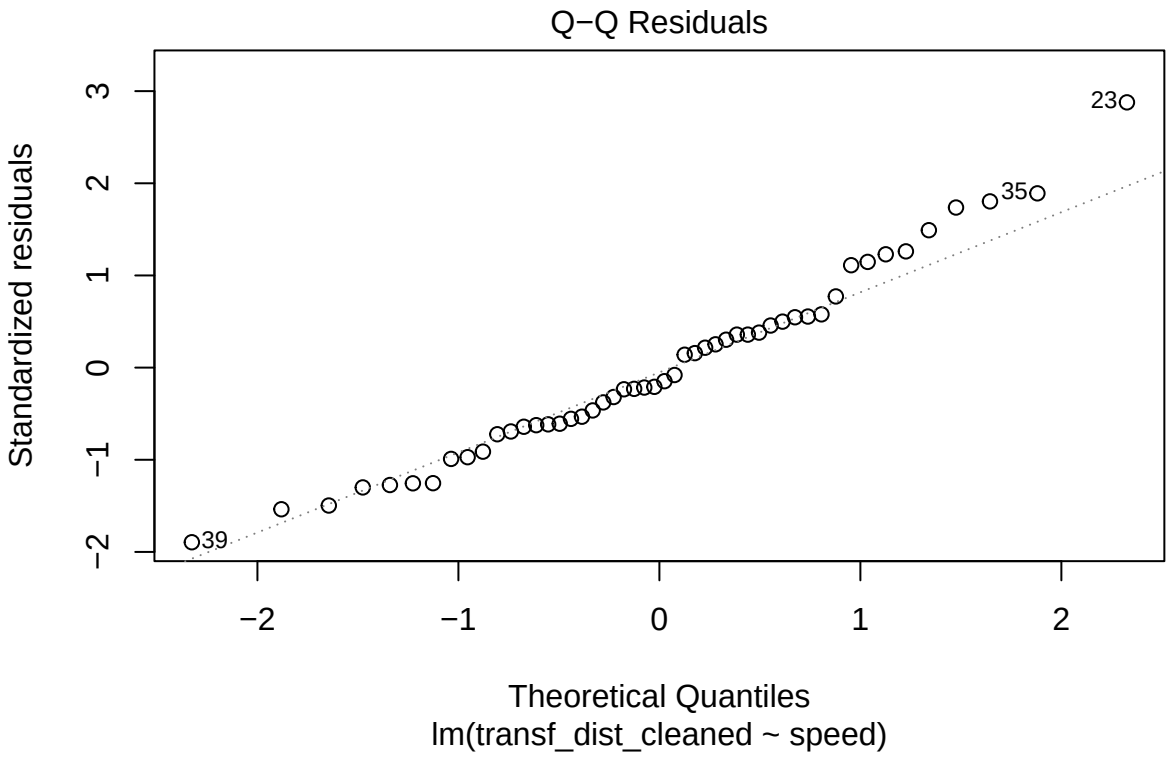


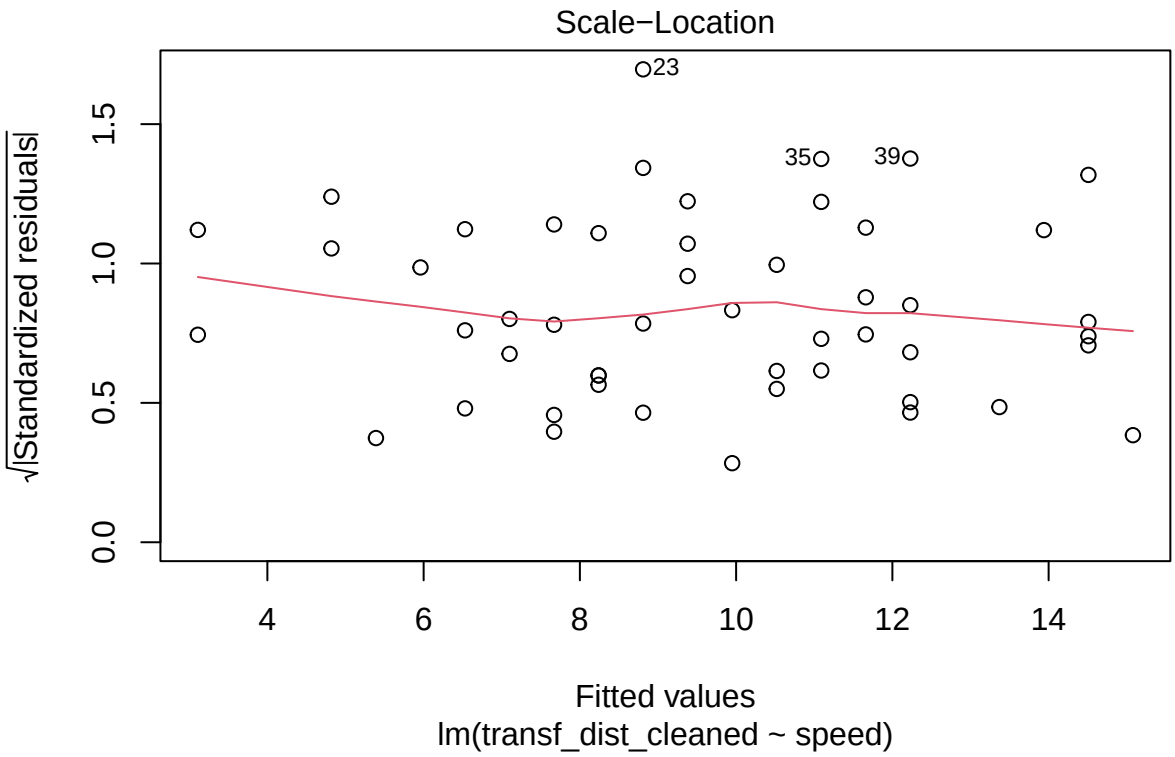

```
# Prueba de Durbin-Watson para autocorrelación de residuos
dwtest(modelo_regresion)
```

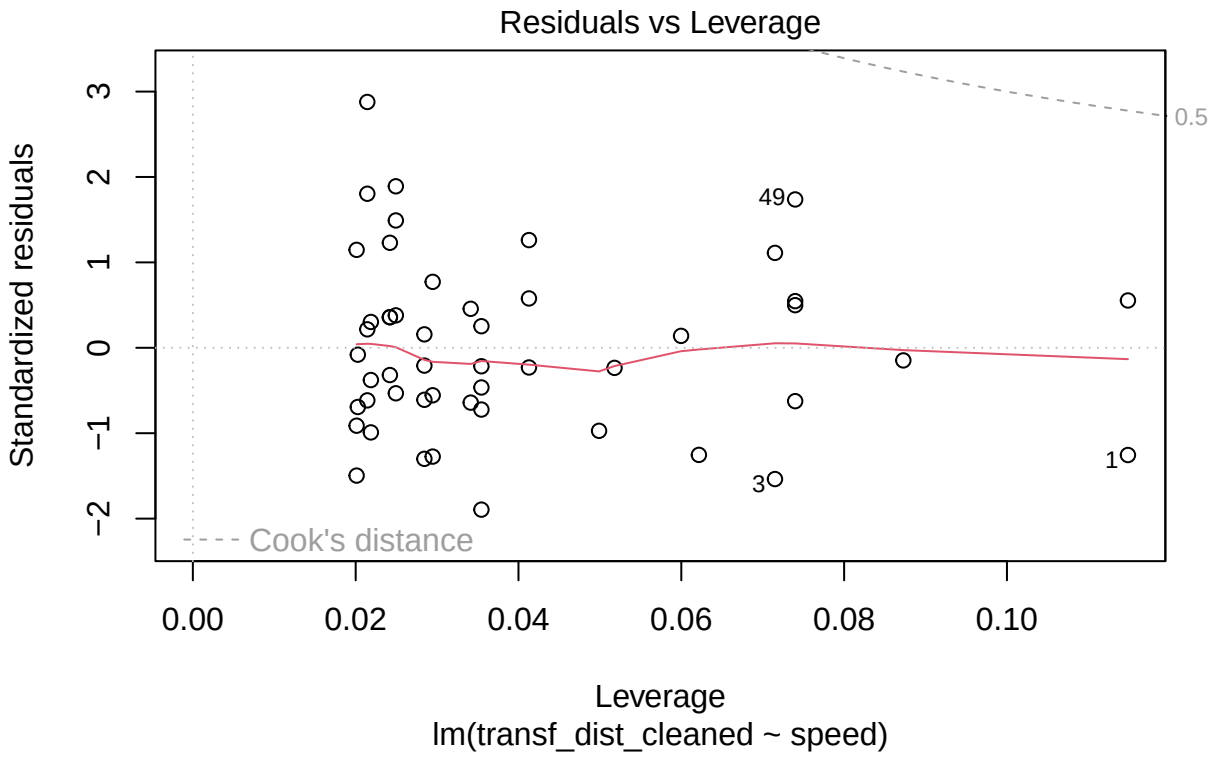
```
##
##  Durbin-Watson test
##
## data:  modelo_regresion
## DW = 1.9519, p-value = 0.3745
## alternative hypothesis: true autocorrelation is greater than 0
```

```
plot(modelo_regresion)
```


Residuals vs Fitted
Residuals
Fitted values
lm(transf_dist_cleaned ~ speed)
23
35
39
6
4
2
0
−2
−4
4
6
8
10
12
14


Q−Q Residuals
Standardized residuals
3
2
1
0
−1
−2
23
35
39
−2
−1
0
1
2
Theoretical Quantiles
lm(transf_dist_cleaned ~ speed)


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
23
35
39
4
6
8
10
12
14
Fitted values
lm(transf_dist_cleaned ~ speed)



Despeja la distancia del modelo lineal obtenido entre la transformación y la velocidad. Obtendrás el modelo no lineal que relaciona la distancia con la velocidad directamente (y no con su transformación).

```
# Obtener los coeficientes del modelo
coef_intercepto <- coef(modelo_regresion)[1]
coef_pendiente <- coef(modelo_regresion)[2]

# Sustituir en la fórmula
modelo_no_lineal <- function(velocidad) {
  (0.4646465 * (coef_intercepto + coef_pendiente * velocidad) + 1)^(1 / 0.4646465)
}
```

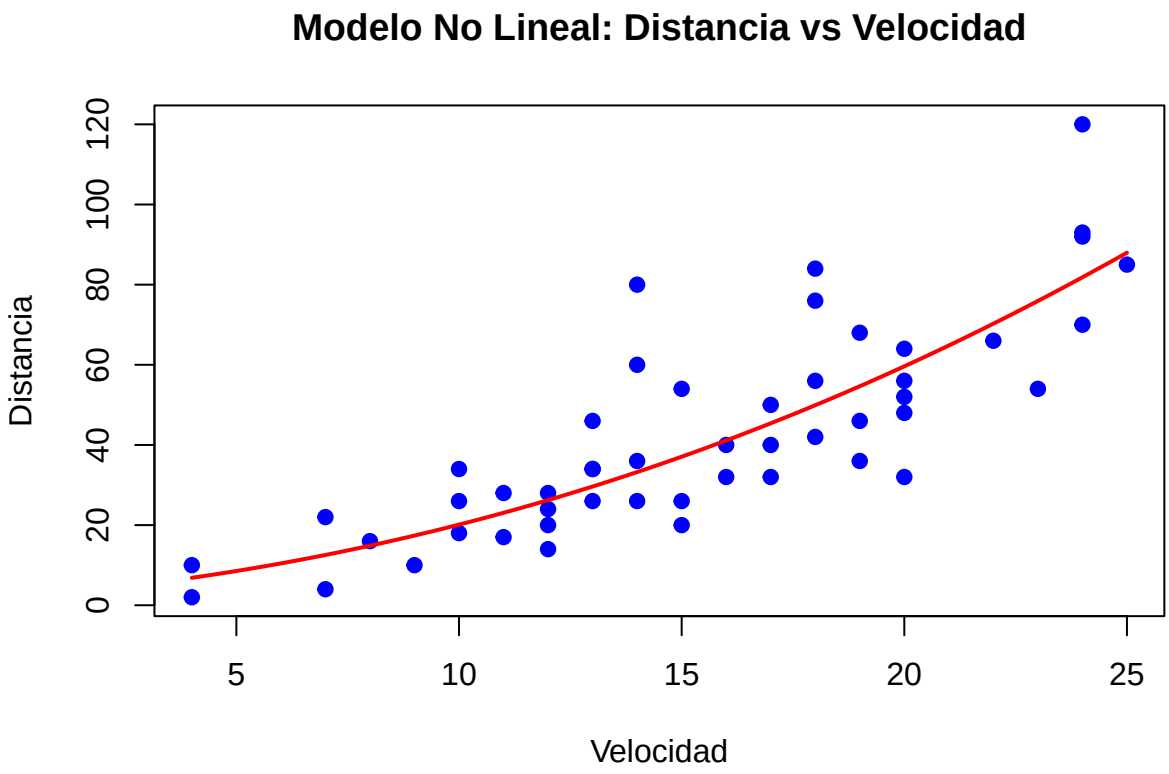
Grafica los datos y el modelo de la distancia en función de la velocidad.

```
# Crear una secuencia de velocidades para la gráfica
velocidades <- seq(min(cars$speed), max(cars$speed), length.out = 100)

# Calcular las distancias usando el modelo no lineal
distancias_no_lineales <- modelo_no_lineal(velocidades)

# Graficar los datos originales
plot(cars$speed, cars$dist, main = "Modelo No Lineal: Distancia vs Velocidad",
     xlab = "Velocidad", ylab = "Distancia", pch = 19, col = "blue")

# Añadir la línea del modelo no lineal
lines(velocidades, distancias_no_lineales, col = "red", lwd = 2)
```



Comenta sobre la idoneidad del modelo en función de su significancia y validez. En resumen, la idoneidad del modelo es alta en función de la significancia y la validez de los residuos. El modelo parece ser una buena aproximación para describir la relación entre la distancia y la velocidad, basada en los resultados obtenidos. Si se identifican puntos atípicos o influyentes, se deberían investigar más a fondo para asegurar que no afectan la validez del modelo.

##Parte 4: Conclusión

Define cuál de los dos modelos analizados (Punto 1 o Punto 2) es el mejor modelo para describir la relación entre la distancia y la velocidad. El modelo exacto es el mejor modelo para poder describir la relacion que existe entre la diistancia y la velocidad de los autos

Comenta sobre posibles problemas del modelo elegido (datos atípicos, alejamiento de los supuestos, dificultad de cálculo o interpretación) Los principales problemas de este modelo son la sensibilidad a valores atipicos grandes, lo cual pueden afectar los analisis del modelo-