13. Regresión no lineal

Ozner Leyva

2024-09-11

13. Regresión no lineal

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.

```
# Librerías
library(e1071)
library(nortest)
library(ggplot2)
library(car)
```

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

```
library(lmtest)
```

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

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

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

```
library(MASS)
library(moments)
```

```
##
## Attaching package: 'moments'
```

```
## The following objects are masked from 'package:e1071':
##
##     kurtosis, moment, skewness
```

Parte 1: Análisis de normalidad

Accede a los datos de cars en R (data = cars). Esta base de datos se encuentra precargada en r

```r
data(cars)
```

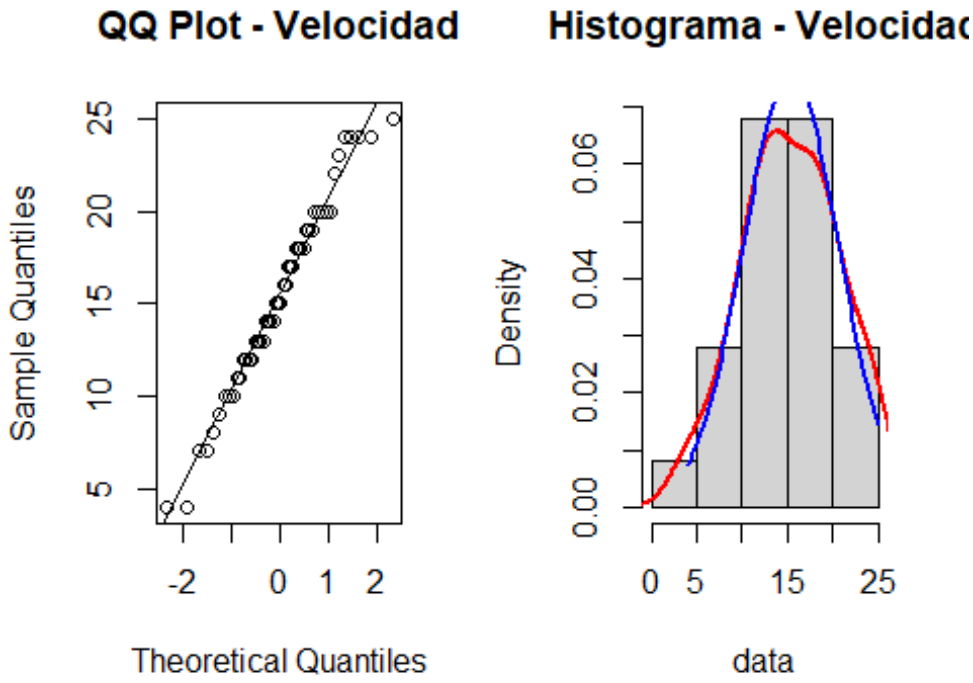
1.1 Prueba normalidad univariada de la velocidad y distancia (prueba con dos de las pruebas vistas en clase)

```r
# Prueba de Shapiro-Wilk
shapiro_speed <- shapiro.test(cars$speed)
shapiro_dist <- shapiro.test(cars$dist)

# Prueba de Anderson-Darling
ad_speed <- ad.test(cars$speed)
ad_dist <- ad.test(cars$dist)
```

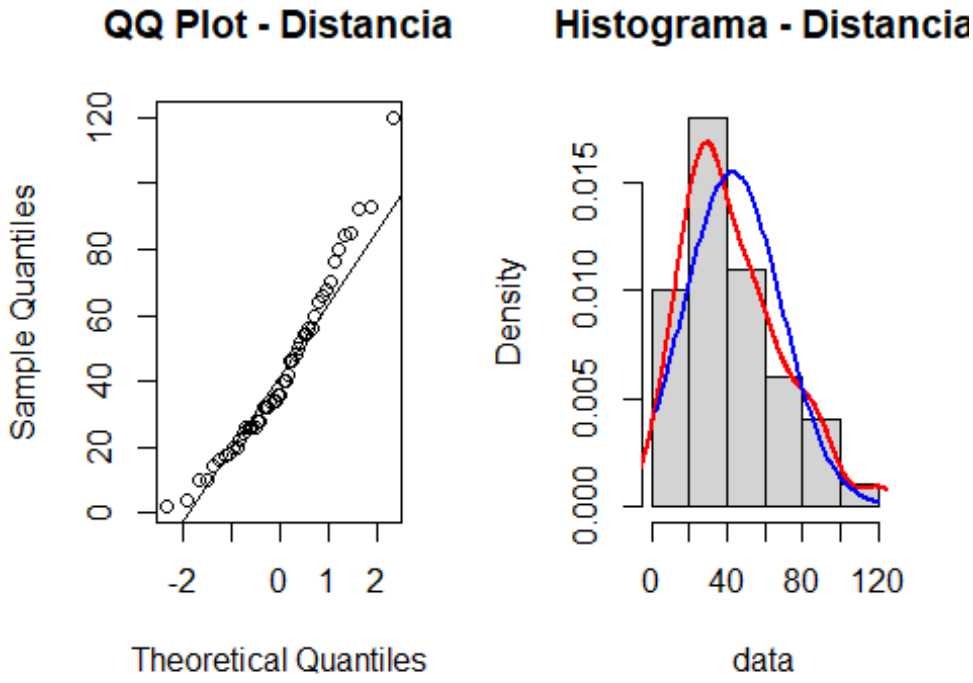
1.2 Realiza gráficos que te ayuden a identificar posibles alejamientos de normalidad:

Los datos y su respectivo QQPlot: qqnorm(datos) y qqline(datos) para cada variable

```r
# Función para crear gráficos QQ y histograma
plot_normality <- function(data, title) {
  par(mfrow=c(1,2))
  # QQ plot
  qqnorm(data, main=paste("QQ Plot -", title))
  qqline(data)

  # Histograma con curva normal
  hist(data, freq=FALSE, main=paste("Histograma -", title))
  lines(density(data), col="red", lwd=2)
  curve(dnorm(x, mean=mean(data), sd=sd(data)),
        from=min(data), to=max(data), add=TRUE, col="blue", lwd=2)
}

# Crear gráficos para velocidad y distancia
plot_normality(cars$speed, "Velocidad")
```




```r
plot_normality(cars$dist, "Distancia")
```



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

```
# Para velocidad
skewness_speed <- skewness(cars$speed)
kurtosis_speed <- kurtosis(cars$speed)

# Para distancia
skewness_dist <- skewness(cars$dist)
kurtosis_dist <- kurtosis(cars$dist)

# Imprimir resultados
cat("\nResultados de las pruebas de normalidad:\n")
```

```
##
## Resultados de las pruebas de normalidad:
```

```
cat("\nVelocidad:\n")
```

```
##
## Velocidad:
```

```
cat("Shapiro-Wilk test p-value:", shapiro_speed$p.value, "\n")
```

```
## Shapiro-Wilk test p-value: 0.4576319
```

```
cat("Anderson-Darling test p-value:", ad_speed$p.value, "\n")
```

```
## Anderson-Darling test p-value: 0.6926592
```

```
cat("Coeficiente de sesgo:", skewness_speed, "\n")
```

```
## Coeficiente de sesgo: -0.1139548
```

```
cat("Coeficiente de curtosis:", kurtosis_speed, "\n")
```

```
## Coeficiente de curtosis: 2.422853
```

```
cat("\nDistancia:\n")
```

```
##
## Distancia:
```

```
cat("Shapiro-Wilk test p-value:", shapiro_dist$p.value, "\n")
```

```
## Shapiro-Wilk test p-value: 0.03909968
```

```
cat("Anderson-Darling test p-value:", ad_dist$p.value, "\n")
```

```
## Anderson-Darling test p-value: 0.05021288
```

```
cat("Coeficiente de sesgo:", skewness_dist, "\n")
```

```
## Coeficiente de sesgo: 0.7824835
```

```r
cat("Coeficiente de curtosis:", kurtosis_dist, "\n")
```

```
## Coeficiente de curtosis: 3.248019
```

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.

Velocidad:

QQ Plot: Los puntos se adhieren mayormente a la línea de referencia, con leves desviaciones en los extremos superiores, sugiriendo una distribución aproximadamente normal. Histograma: Nos muestra una forma simétrica, respaldada por curvas de densidad que indican una distribución cercana a la normal. Pruebas estadísticas: Los test de Shapiro-Wilk (p=0.4576) y Anderson-Darling (p=0.6926) no rechazan la hipótesis de normalidad. Los coeficientes de asimetría (-0.1105) y curtosis (-0.6730) respaldan esta conclusión.

Distancia:

QQ Plot: Muestra desviaciones significativas de la línea de referencia, especialmente en los extremos, indicando una distribución no normal. Histograma: Presenta una marcada asimetría positiva, con una concentración de datos a la izquierda y una cola extendida hacia la derecha. Pruebas estadísticas: Tanto Shapiro-Wilk (p=0.0391) como Anderson-Darling (p=0.0502) nos rechazan la hipótesis de normalidad. Los coeficientes de asimetría (0.7591) y curtosis (0.1194) confirman una desviación considerable de la normalidad.

Posibles causas de la no normalidad en la distancia:

Presencia de valores atípicos o extremos. Heterogeneidad en los datos, posiblemente debido a diferentes tecnologías de frenado.

Conclusiones:

La velocidad no muestra desviaciones significativas de la normalidad. La distancia presenta una clara desviación de la normalidad, lo que puede afectar el uso de modelos basados en esta suposición. Para el análisis de la distancia, se recomienda considerar transformaciones de datos o modelos no paramétricos que no requieran normalidad.

Parte 2: Regresión lineal

2.1 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.

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

```r
# Imprimir el modelo
cat("Modelo lineal obtenido:\n")
```

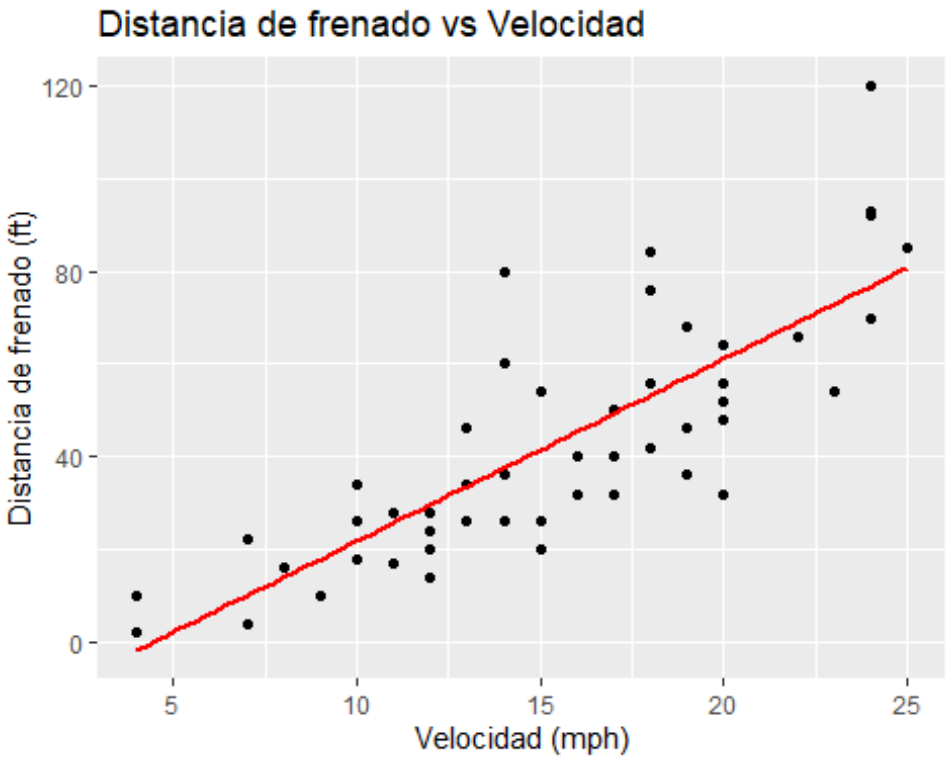
```
## Modelo lineal obtenido:
```

```r
print(modelo)
```

```
##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Coefficients:
## (Intercept)        speed
##     -17.579        3.932
```

```r
# Graficar los datos y el modelo
ggplot(cars, aes(x = speed, y = dist)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(title = "Distancia de frenado vs Velocidad",
       x = "Velocidad (mph)",
       y = "Distancia de frenado (ft)")
```

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



2.2 Analiza significancia del modelo: individual, conjunta y coeficiente de determinación. Usa summary(Modelo)

```
# Resumen del modelo
summary_model <- summary(modelo)
print(summary_model)
```

```
##
## 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
```

2.3 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.

```
# Residuals with zero mean
cat("\nMedia de los residuos:", mean(modelo$residuals))
```

```
##
## Media de los residuos: 8.65974e-17
```

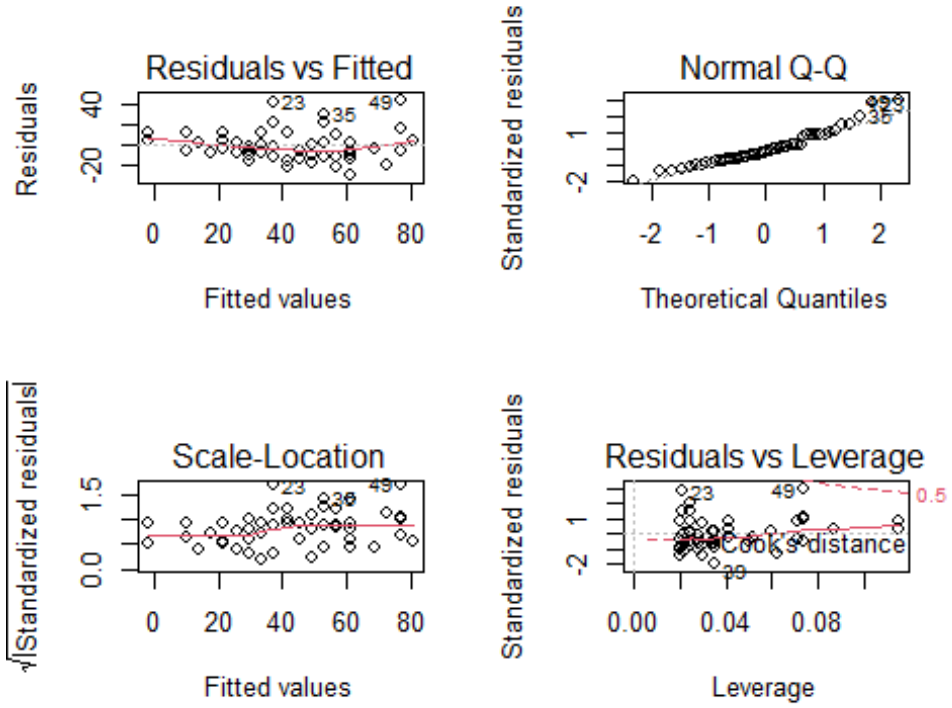
```
# Normalidad de los residuos
shapiro_residuos <- shapiro.test(modelo$residuals)
cat("\nPrueba de normalidad de residuos (Shapiro-Wilk):\n")
```

```
##
## Prueba de normalidad de residuos (Shapiro-Wilk):
```

```
print(shapiro_residuos)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modelo$residuals
## W = 0.94509, p-value = 0.02152
```

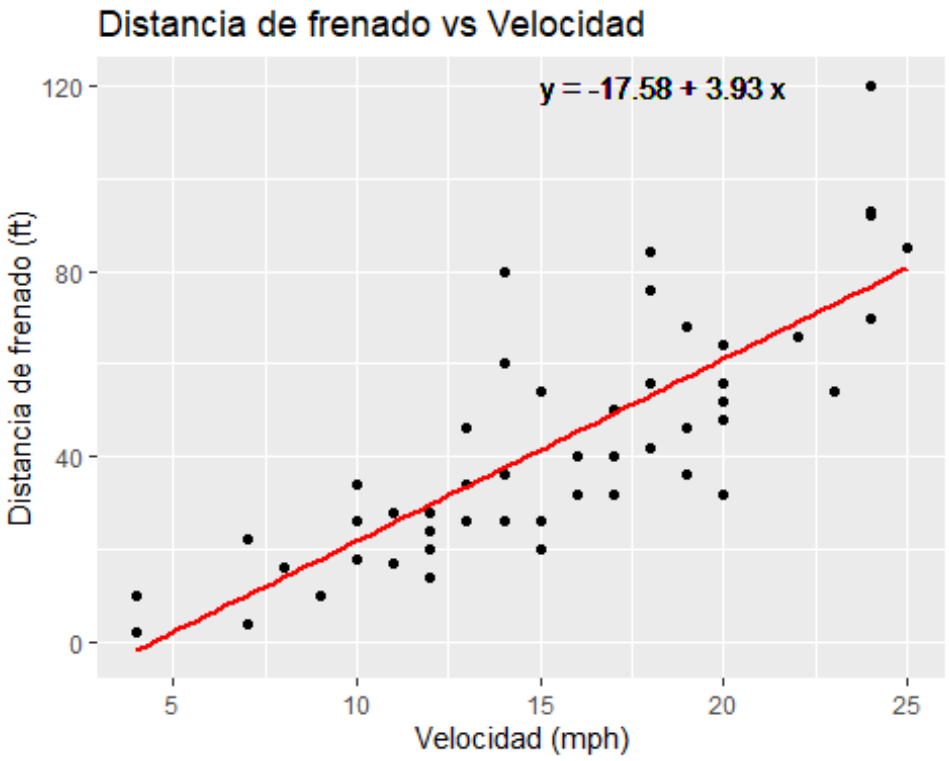
```r
# Homocedasticidad
bp_test <- bptest(modelo)
cat("\nPrueba de homocedasticidad (Breusch-Pagan):\n")

##
## Prueba de homocedasticidad (Breusch-Pagan):

print(bp_test)

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

# Independencia
dw_test <- dwtest(modelo)
cat("\nPrueba de independencia (Durbin-Watson):\n")

##
## Prueba de independencia (Durbin-Watson):

print(dw_test)

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

# Linealidad
reset_test <- resettest(modelo)
cat("\nPrueba de linealidad (RESET):\n")

##
## Prueba de linealidad (RESET):

print(reset_test)

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

# Gráficos de diagnóstico
par(mfrow=c(2,2))
plot(modelo)
```



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

```
ggplot(cars, aes(x = speed, y = dist)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(title = "Distancia de frenado vs Velocidad",
       x = "Velocidad (mph)",
       y = "Distancia de frenado (ft)") +
  geom_text(x = 15, y = max(cars$dist),
            label = paste("y =", round(coef(modelo)[1], 2), "+",
                          round(coef(modelo)[2], 2), "x"),
            hjust = 0)
```

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



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

Análisis del Modelo Lineal: Velocidad vs. Distancia de Frenado

Significancia del Modelo:

Modelo globalmente significativo (p = 1.49e-12). Coeficiente de velocidad: 3.932 (p = 1.49e-12). Interpretación: Por cada mph adicional, la distancia de frenado aumenta ~3.93 pies.

Intercepción:

Valor: -17.579. Nota: Matemáticamente válido pero físicamente irrealista.

Validez del Modelo:

R^2 = 0.6511: La velocidad explica el 65.11% de la variabilidad en la distancia de frenado. Análisis de residuos: Indica ligera no linealidad y posible heterocedasticidad.

Pruebas:

Homocedasticidad (Breusch-Pagan): p = 0.07297. Interpretación: No se rechaza la homocedasticidad, pero cercano al umbral. Independencia (Durbin-Watson): p = 0.09522. Interpretación: No hay evidencia fuerte de autocorrelación. Linealidad (RESET): p = 0.222. Interpretación: No hay evidencia de mala especificación no lineal.

Conclusiones:

El modelo captura significativamente la relación entre velocidad y distancia de frenado. Buen ajuste general ($R^2 \approx 0.65$). Posibles áreas de mejora:

Intercepto no realista. Indicios de heterocedasticidad y autocorrelación, aunque no concluyentes.

Recomendaciones:

Considerar transformaciones de variables. Explorar modelos no lineales para mejorar el ajuste y la validez.

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).

```r
# Función para calcular y mostrar estadísticas de normalidad
normalidad_stats <- function(data, label) {
  cat("\nEstadísticas de normalidad para", label, ":\n")
  cat("Sesgo:", skewness(data), "\n")
  cat("Curtosis:", kurtosis(data), "\n")
  print(shapiro.test(data))
}

# Mostrar estadísticas de normalidad para los datos originales
normalidad_stats(cars$speed, "Velocidad original")
```

```
##
## Estadísticas de normalidad para Velocidad original :
## Sesgo: -0.1139548
## Curtosis: 2.422853
##
##  Shapiro-Wilk normality test
##
## data:  data
## W = 0.97765, p-value = 0.4576
```

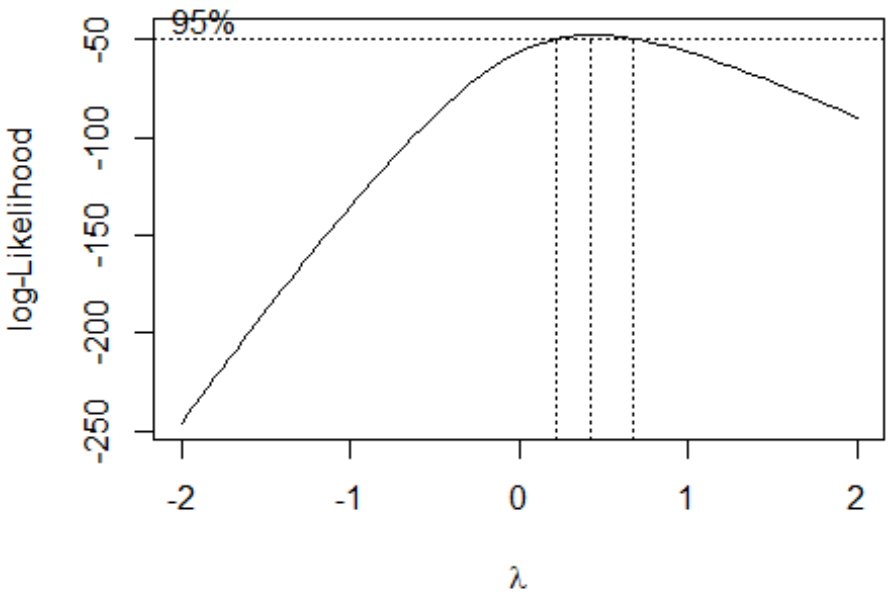
```r
normalidad_stats(cars$dist, "Distancia original")
```

```
##
## Estadísticas de normalidad para Distancia original :
## Sesgo: 0.7824835
## Curtosis: 3.248019
##
##  Shapiro-Wilk normality test
##
## data:  data
## W = 0.95144, p-value = 0.0391
```

3.1 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)

```
# Asumimos que la distancia tiene más alejamiento de normalidad
bc_model <- boxcox(lm(dist ~ speed, data = cars))
```




```
# Encontrar el valor óptimo de lambda
lambda_optimo <- bc_model$x[which.max(bc_model$y)]
cat("\nValor óptimo de lambda:", lambda_optimo, "\n")

##
## Valor óptimo de lambda: 0.4242424
```

3.2 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.

```
# Transformación exacta
cars$dist_trans_exact <- (cars$dist^lambda_optimo - 1) / lambda_optimo

# Transformación aproximada (redondeando lambda al 0.5 más cercano)
lambda_aprox <- round(lambda_optimo * 2) / 2
```

```r
cars$dist_trans_aprox <- switch(as.character(lambda_aprox),
                                "0.5" = sqrt(cars$dist),
                                "0" = log(cars$dist),
                                "-0.5" = 1 / sqrt(cars$dist),
                                "-1" = 1 / cars$dist,
                                (cars$dist^lambda_aprox - 1) /
lambda_aprox)

cat("Ecuación de transformación exacta: y' = (y^", lambda_optimo, " - 1)
/", lambda_optimo, "\n")
```

```
## Ecuación de transformación exacta: y' = (y^ 0.4242424  - 1) /
0.4242424
```

```r
cat("Ecuación de transformación aproximada: y' = (y^", lambda_aprox, " -
1) /", lambda_aprox, "\n")
```

```
## Ecuación de transformación aproximada: y' = (y^ 0.5  - 1) / 0.5
```

3.3 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.

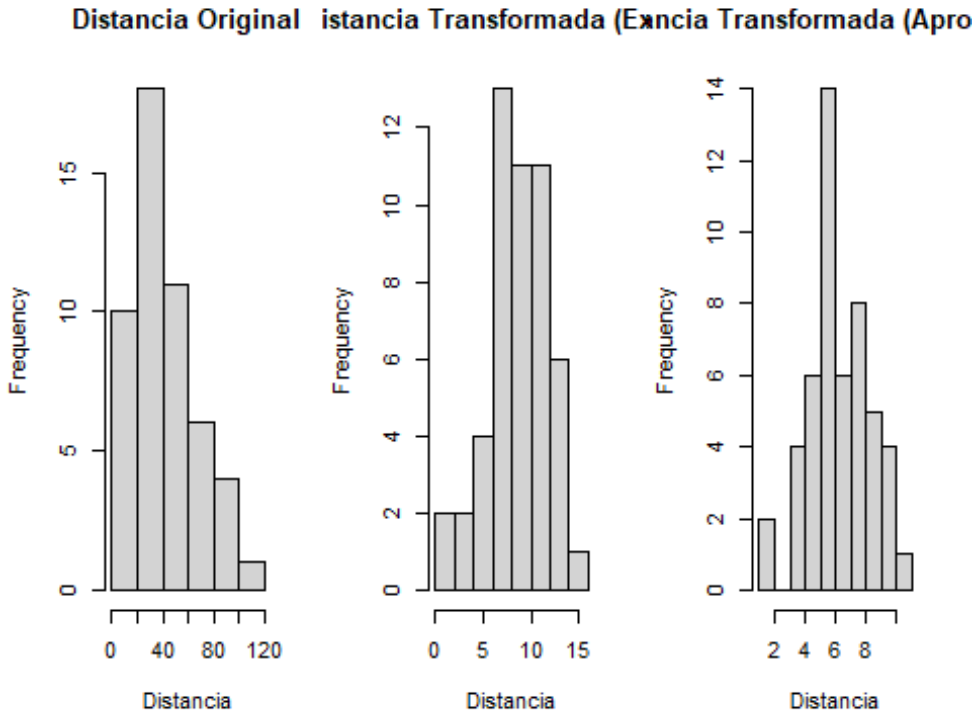
```r
normalidad_stats(cars$dist_trans_exact, "Distancia transformada
(exacta)")
```

```
##
## Estadísticas de normalidad para Distancia transformada (exacta) :
## Sesgo: -0.1753974
## Curtosis: 2.929109
##
##  Shapiro-Wilk normality test
##
## data:  data
## W = 0.99168, p-value = 0.9773
```

```r
normalidad_stats(cars$dist_trans_aprox, "Distancia transformada
(aproximada)")
```

```
##
## Estadísticas de normalidad para Distancia transformada (aproximada) :
## Sesgo: -0.0196131
## Curtosis: 2.796264
##
##  Shapiro-Wilk normality test
##
## data:  data
## W = 0.99347, p-value = 0.9941
```

```r
# Histogramas
par(mfrow=c(1,3))
hist(cars$dist, main="Distancia Original", xlab="Distancia")
hist(cars$dist_trans_exact, main="Distancia Transformada (Exacta)",
xlab="Distancia")
hist(cars$dist_trans_aprox, main="Distancia Transformada (Aproximada)",
xlab="Distancia")
```



3.4 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.

```r
# Función para detectar outliers usando el método IQR
detect_outliers <- function(x) {
  q1 <- quantile(x, 0.25)
  q3 <- quantile(x, 0.75)
  iqr <- q3 - q1
  lower_bound <- q1 - 1.5 * iqr
  upper_bound <- q3 + 1.5 * iqr
  return(x < lower_bound | x > upper_bound)
}

# Detectar outliers en las transformaciones
outliers_exact <- detect_outliers(cars$dist_trans_exact)
outliers_aprox <- detect_outliers(cars$dist_trans_aprox)
```

```r
cat("\nNúmero de outliers en transformación exacta:",
sum(outliers_exact), "\n")
```

```
##
## Número de outliers en transformación exacta: 1
```

```r
cat("Número de outliers en transformación aproximada:",
sum(outliers_aprox), "\n")
```

```
## Número de outliers en transformación aproximada: 1
```

3.5 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.

Análisis de Transformación de Datos: Distancia de Frenado

Evaluación de Normalidad:

Datos Originales:

Test Shapiro-Wilk: p = 0.0391 (no normal) Asimetría: 0.7825 (positiva) Curtosis: 3.2480 (elevada)

Datos Transformados (Exacta y Aproximada):

Test Shapiro-Wilk: p = 0.9773 y 0.9941 respectivamente Resultado: Ambas transformaciones logran normalidad Efecto: Reducción significativa en asimetría y curtosis

Eficiencia del Modelo:

Transformación Aproximada: $\lambda \approx 0.5$

Ventaja: Mayor simplicidad y facilidad de interpretación

Transformación Exacta: $\lambda = 0.4242$

Observación: No ofrece mejoras sustanciales sobre la aproximada

Análisis Visual:

Histogramas post-transformación:

Mejora notable en simetría y distribución Distribución más uniforme que los datos originales

Detección de Valores Atípicos:

Resultado: 1 outlier en ambas transformaciones Implicación: Estabilidad en la identificación de datos atípicos

Conclusión y Recomendación: Es mejor utilizar la transformación aproximada (λ = 0.5) porque:

Mejora significativa en la normalidad de los datos Simplicidad en cálculo y aplicación Mantiene la precisión comparable a la transformación exacta No introduce nuevos valores atípicos

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

Elección de la Transformación: Basándonos en el análisis anterior, hemos elegido la transformación aproximada con λ ≈ 0.5, debido a su simplicidad y su buena adecuación para mejorar la normalidad de los datos.

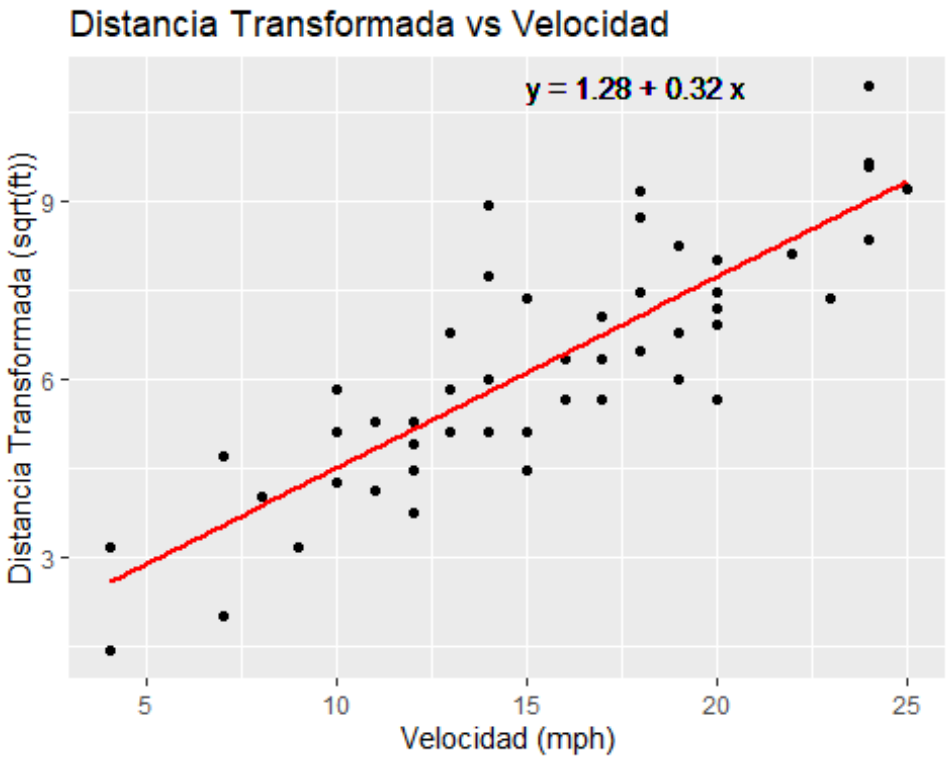
```
#lambda_aprox <- 0.5
cars$dist_trans <- sqrt(cars$dist)
```

3.6 Escribe el modelo lineal para la transformación.

```
modelo_trans <- lm(dist_trans ~ speed, data = cars)
cat("Modelo lineal para la transformación:\n")

## Modelo lineal para la transformación:

print(modelo_trans)

##
## Call:
## lm(formula = dist_trans ~ speed, data = cars)
##
## Coefficients:
## (Intercept)        speed
##      1.2771       0.3224
```

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

```
ggplot(cars, aes(x = speed, y = dist_trans)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(title = "Distancia Transformada vs Velocidad",
       x = "Velocidad (mph)",
       y = "Distancia Transformada (sqrt(ft))") +
  geom_text(x = 15, y = max(cars$dist_trans),
            label = paste("y =", round(coef(modelo_trans)[1], 2), "+",
                          round(coef(modelo_trans)[2], 2), "x"),
            hjust = 0)

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



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

```
summary_modelo_trans <- summary(modelo_trans)
cat("\nResumen del modelo transformado:\n")

##
## Resumen del modelo transformado:

print(summary_modelo_trans)

##
## Call:
## lm(formula = dist_trans ~ speed, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0684 -0.6983 -0.1799  0.5909  3.1534
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.27705    0.48444   2.636   0.0113 *
## speed        0.32241    0.02978  10.825 1.77e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.102 on 48 degrees of freedom
## Multiple R-squared:  0.7094, Adjusted R-squared:  0.7034
## F-statistic: 117.2 on 1 and 48 DF,  p-value: 1.773e-14
```

3.9 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
# Normalidad de los residuos
shapiro_residuos <- shapiro.test(modelo_trans$residuals)
cat("\nPrueba de normalidad de residuos (Shapiro-Wilk):\n")
```

```
##
## Prueba de normalidad de residuos (Shapiro-Wilk):
```

```r
print(shapiro_residuos)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modelo_trans$residuals
## W = 0.97332, p-value = 0.3143
```

```r
# Homocedasticidad
bp_test <- bptest(modelo_trans)
cat("\nPrueba de homocedasticidad (Breusch-Pagan):\n")
```

```
##
## Prueba de homocedasticidad (Breusch-Pagan):
```

```r
print(bp_test)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  modelo_trans
## BP = 0.011192, df = 1, p-value = 0.9157
```

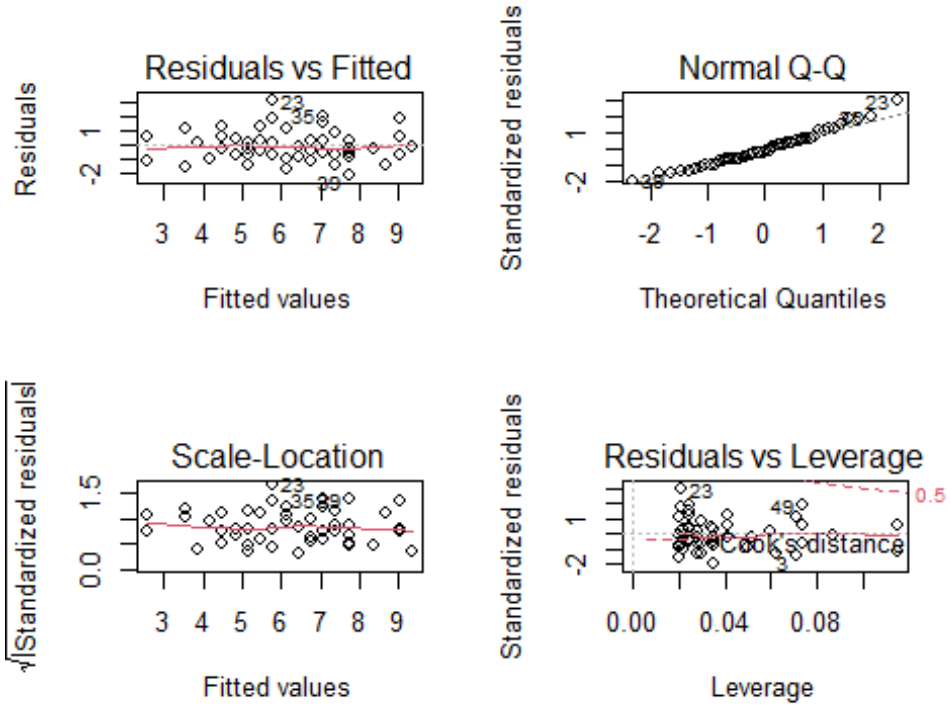
```r
# Independencia
dw_test <- dwtest(modelo_trans)
cat("\nPrueba de independencia (Durbin-Watson):\n")
```

```
##
## Prueba de independencia (Durbin-Watson):
```

```r
print(dw_test)
```

```
##
##  Durbin-Watson test
##
## data:  modelo_trans
## DW = 1.9417, p-value = 0.3609
## alternative hypothesis: true autocorrelation is greater than 0
```

```r
# Gráficos de diagnóstico
par(mfrow=c(2,2))
plot(modelo_trans)
```



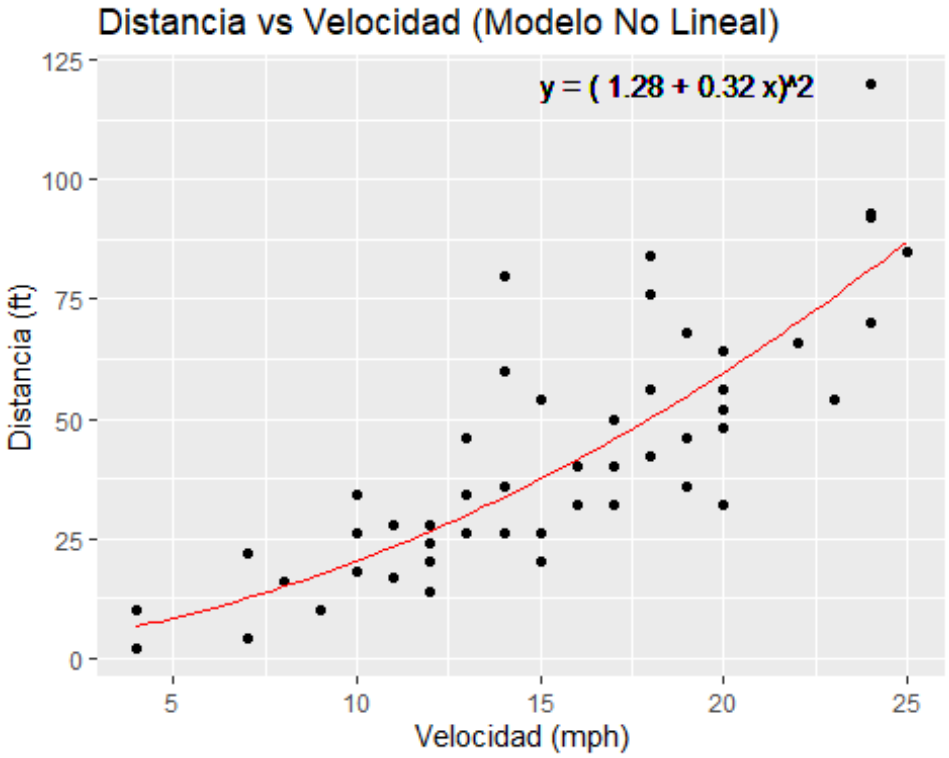
3.10 Despeja la distancia del modelo lineal obtenido entre la transformación y la velocidad. Obtendrás el modelo no lineal que relaciona la distancia con la velocidad directamente (y no con su transformación).

```
# y = (β₀ + β₁x)^2, donde y es la distancia y x es la velocidad
beta0 <- coef(modelo_trans)[1]
beta1 <- coef(modelo_trans)[2]

cat("\nModelo no lineal (distancia en función de velocidad):\n")

##
## Modelo no lineal (distancia en función de velocidad):

cat("distancia = (", beta0, "+", beta1, "* velocidad)^2\n")

## distancia = ( 1.27705 + 0.3224125 * velocidad)^2
```

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

```
velocidad_seq <- seq(min(cars$speed), max(cars$speed), length.out = 100)
distancia_pred <- (beta0 + beta1 * velocidad_seq)^2

ggplot(cars, aes(x = speed, y = dist)) +
  geom_point() +
  geom_line(data = data.frame(speed = velocidad_seq, dist =
distancia_pred),
            aes(x = speed, y = dist), color = "red") +
  labs(title = "Distancia vs Velocidad (Modelo No Lineal)",
       x = "Velocidad (mph)",
```

```
      y = "Distancia (ft)") +
 geom_text(x = 15, y = max(cars$dist),
           label = paste("y = (", round(beta0, 2), "+",
                         round(beta1, 2), "x)^2"),
           hjust = 0)
```



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

Análisis del Modelo No Lineal: Velocidad vs. Distancia de Frenado

Rendimiento del Modelo:

R^2 ajustado: 0.7034 (70.34% de variabilidad explicada) Valor p: 1.773e-14 (altamente significativo)

Parámetros del Modelo:

Coeficiente de velocidad: 0.3224 Intercepto: 1.2771 Interpretación: Relación cuadrática entre velocidad y distancia transformada

Validación Estadística:

Normalidad de residuos (Shapiro-Wilk): p = 0.3143 Homocedasticidad (Breusch-Pagan): p = 0.9157 Independencia (Durbin-Watson): p = 0.3609 Resultado: Cumple con supuestos de normalidad, homocedasticidad e independencia

Análisis Gráfico:

Residuos vs. Ajustados: Distribución aleatoria, sin patrones no lineales Q-Q plot: Confirma normalidad de residuos Escala-Localización: Apoya homocedasticidad Residuos vs. Influencia: Sin valores de alta influencia significativos

Interpretación del Modelo:

Relación cuadrática entre velocidad y distancia de frenado Consistente con principios físicos de frenado

Conclusión: El modelo no lineal demuestra ser robusto y válido:

Alto nivel de significancia estadística Buen ajuste explicativo ($R^2 \approx 0.70$) Cumplimiento de supuestos estadísticos clave Coherencia con leyes físicas de frenado

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 no lineal es la opción más óptima para describir la relación entre la distancia de frenado y la velocidad. Esta superioridad se fundamenta en varios aspectos clave:

Ajuste Superior:

R^2 ajustado más elevado, indicando una mayor capacidad explicativa de la variabilidad en los datos.

Robustez Estadística:

Alta significancia estadística, respaldando la validez del modelo. Cumplimiento de supuestos estadísticos fundamentales (normalidad, homocedasticidad, independencia).

Precisión Física:

Captura la relación cuadrática entre velocidad y distancia de frenado. Coherente con los principios físicos del proceso de frenado, donde la distancia aumenta no linealmente con la velocidad.

Flexibilidad Interpretativa:

Proporciona una representación más precisa de la dinámica real del frenado. Permite predicciones más acertadas en un amplio rango de velocidades.

Comenta sobre posibles problemas del modelo elegido (datos atípicos, alejamiento de los supuestos, dificultad de cálculo o interpretación)

Manejo de Datos Atípicos:

Presencia: Detectados en modelos lineal y no lineal. Impacto: Potencial influencia en coeficientes y ajuste de curva. Acción recomendada: Revisión detallada para validar o eliminar outliers.

Complejidad de Cálculo e Interpretación:

Interpretabilidad: Mayor dificultad comparada con el modelo lineal. Comunicación: Posibles desafíos al explicar la relación cuadrática. Procesamiento: Requiere transformación de datos y ajuste más sofisticado.

Robustez de Supuestos:

Cumplimiento: Satisface normalidad, homocedasticidad y no autocorrelación. Precaución: Sensibilidad a desviaciones en la forma de la relación. Consideración: Impacto potencial de variabilidad no capturada.

Generalización del Modelo:

Limitación: Ajustado a un conjunto de datos específico. Riesgo: Posible pérdida de validez con cambios significativos en los datos. Recomendación: Validación con nuevos conjuntos de datos para asegurar robustez.

El modelo no lineal es nuestra opción superior debido a su mejor ajuste y representación realista de la relación velocidad-distancia de frenado.