Regresión lineal simple

Agustín Castro

2023-10-14

Contents

Inicio

Píldoras_R. Material de formación

En esta práctica trabajemos con la **regresión lineal simple**

[mi_blog] https://agustincastro.es

La **regresión lineal simple** es un método estadístico que se utiliza para **modelar la relación entre una variable dependiente (o respuesta) y una variable independiente**. La idea principal detrás de la regresión lineal simple es comprender **cómo cambia la variable dependiente cuando lo hace la variable independiente**. El objetivo es encontrar la mejor línea recta que se ajuste a los datos observados.

Vamos a utilizar las siguientes librerías en esta práctica.

```
library(tidyverse)
library(ggplot2)
library(hrbrthemes) # para el theme_ipsum()
library(lmtest) # para el test de Breusch-Pagan
library(psych) # para pairs.panels
```

mtcars

Trabajaremos el dataset **mtcars** que contiene datos de diferentes modelos de coches. El conjunto de datos mtcars en R consta de **32 observaciones** y **11 variables**.

```
data(mtcars)
```

Nos centraremos ahora en las variables **mpg** (millas por galón = consumo) y **wt** (peso).

```
head(mtcars, 5)
```

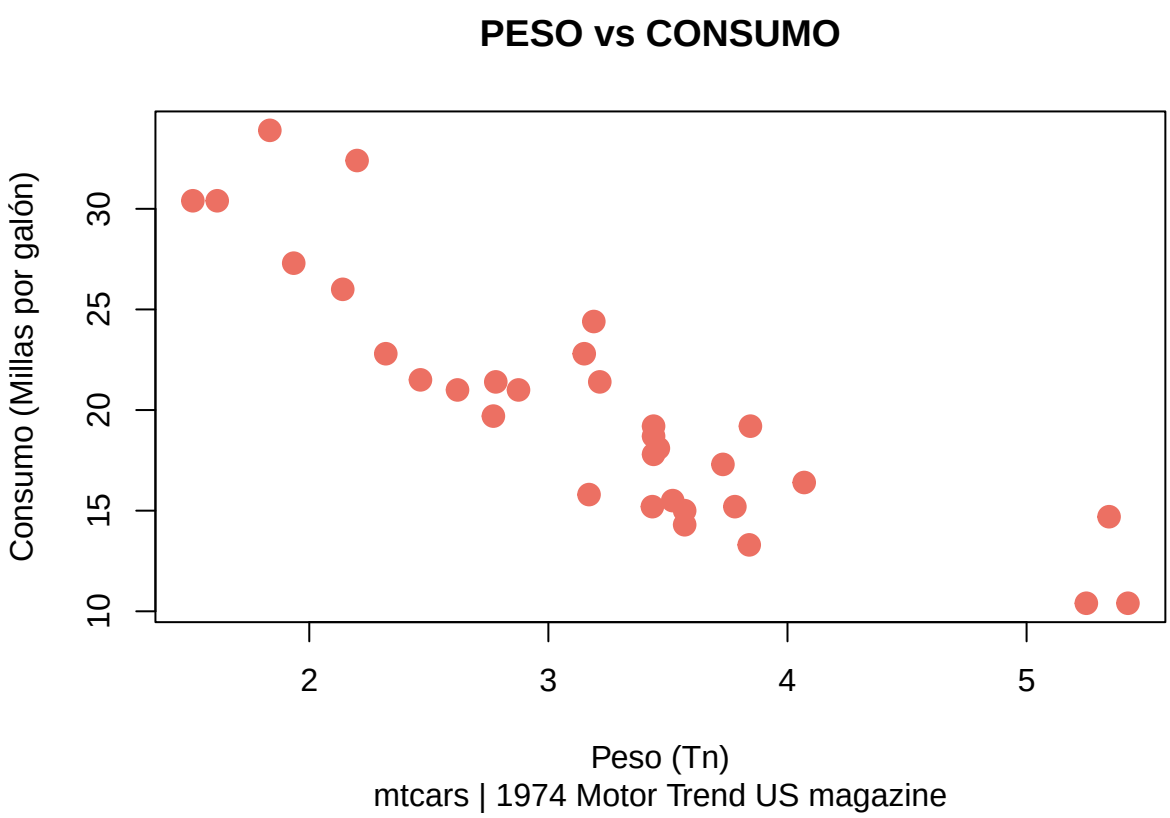
```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
```

```
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

Dibujamos con **plot** la relación entre las variables **mpg** y **wt**. Esta función viene de base con R y es sencilla de usar para echar un vistazo rápido. Se observa que hay una relación negativa, o inversa, entre las dos variables. **A medida que aumenta el peso, disminuye la cantidad de millas que podemos recorrer por galón de combustible**.

```
plot(mtcars$wt, mtcars$mpg, main = "PESO vs CONSUMO",
     sub = "mtcars | 1974 Motor Trend US magazine",
     xlab = "Peso (Tn)", ylab = "Consumo (Millas por galón)",
     col = "#EC7063", pch = 19, type = "p", cex = 1.5)
```



Correlación lineal Pearson

Podemos estudiar como es la correlación **lineal** entre ambas variables utilizando **cor**. El coeficiente de correlación de Pearson (r) es de **-0.868**, y el coeficiente de determinación (R) es **0.7528**. Podríamos decir que el 75% de la variabilidad de una variable es explicada por la otra.

```
r <- cor(mtcars$wt, mtcars$mpg)
R <- r^2 # coeficiente de determinación
print (r)
```

```
## [1] -0.8676594
```

```
print (R)
```

```
## [1] 0.7528328
```

Podemos obtener información detallada del test aplicado para estudiar la correlación con la función **cor.test**.

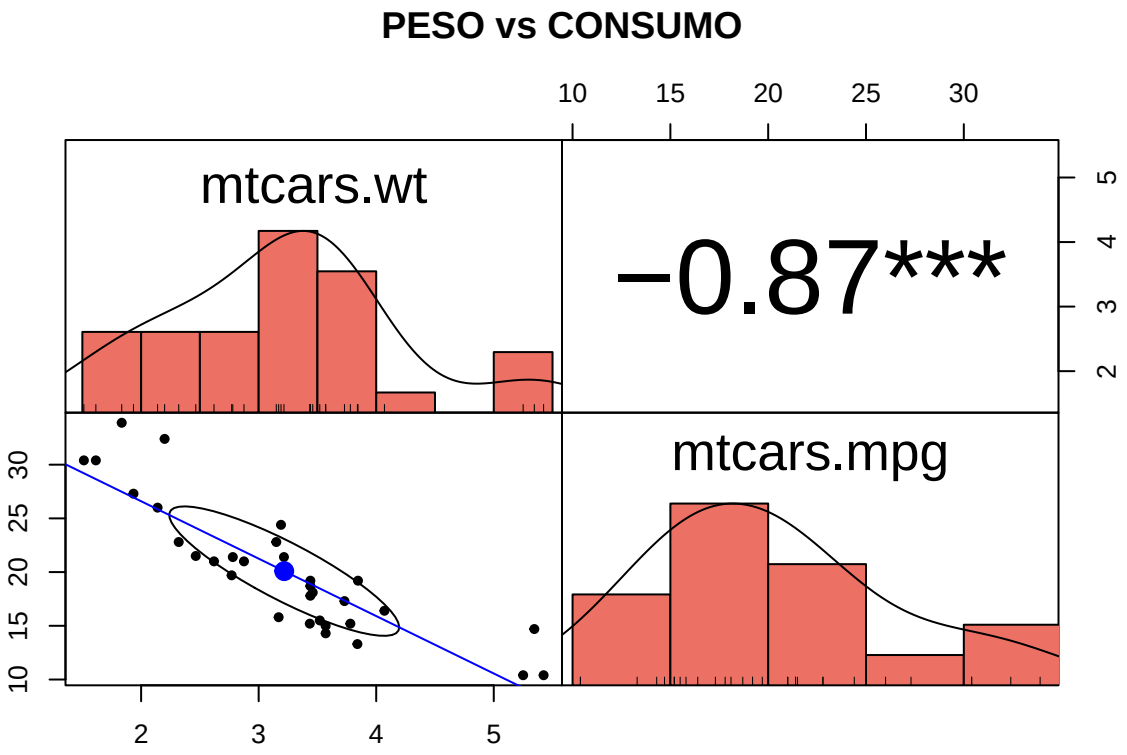
```
cor.test(mtcars$wt, mtcars$mpg)
```

```
##
##  Pearson's product-moment correlation
```

```
##
## data:  mtcars$wt and mtcars$mpg
## t = -9.559, df = 30, p-value = 1.294e-10
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.9338264 -0.7440872
## sample estimates:
##        cor
## -0.8676594
```

Gráfico de correlación También podemos representar gráficamente la correlación con **pairs.panels**, una función de la librería **pysch**. Hay otras muchas funciones para representar gráficamente la correlación, como por ejemplo **corrplot**, aunque lo veremos en otras prácticas.

```
corr_grafico <- data.frame(mtcars$wt, mtcars$mpg)
pairs.panels(corr_grafico, pch = 20, stars = TRUE, gap = 0,
             lm = TRUE, col = "blue",
             hist.col = "#EC7063",
             main = "PESO vs CONSUMO")
```



Regresión lineal

Para crear el modelo de regresión lineal utilizamos la función **lm**, que creará el objeto **lm_mtcars**. En este objeto es donde se guardarán todos los resultados. La variable dependiente, o respuesta, será **mpg** y la

independiente, o predictora, **wt**. Lo que nos interesa es conocer *como el peso del vehículo va a influir en el consumo de combustible*. De hecho, se intuye una relación inversa.

El objeto creado contiene información de dos valores, el **intercepto** y la **pendiente** de la recta (y = a + bX, donde y = es la variable respuesta (mpg), a = intercepto y b = pendiente (wt)). El modelo sería el siguiente **mpg = 37.2851 - 5.3445 * wt**.

```
lm_mtcars <- lm(mpg ~ wt, data = mtcars)
lm_mtcars
```

```
##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Coefficients:
## (Intercept)           wt
##      37.285       -5.344
```

Si queremos más informción sobre el modelo creado, podemos acceder a un resumen utilizando la función **summary**.

```
summary(lm_mtcars)
```

```
##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5432 -2.3647 -0.1252  1.4096  6.8727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
## wt           -5.3445     0.5591  -9.559 1.29e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.046 on 30 degrees of freedom
## Multiple R-squared:  0.7528, Adjusted R-squared:  0.7446
## F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

[importante] Para poder dar por bueno el modelo de regresión lineal es necesario que se cumplan una serie de supuestos. El primero de ellos es que la relación entre las variables sea **lineal**. Por otro lado, es necesario que los **residuos sean normales**, es decir, que sigan una **distribución normal**. Por último, es necesario que la **varianza de los residuos** sea homogenea, es decir, que exista **homocedasticidad**.

Shapiro-Wilk

Para medir la normalidad de los residuos podemos utilizar el test de **Shapiro-Wilk**. Como el p-valor (0.1044) es mayor que el nivel de significancia (0.05), concluimos que no hay evidencia para rechazar la hipótesis nula de normalidad. Por lo tanto, **los residuos se distribuyen normalmente**.

```r
shapiro.test(residuals(lm_mtcars))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(lm_mtcars)
## W = 0.94508, p-value = 0.1044
```

```r
# shapiro.test(lm_mtcarars$residuals))
# otra forma correcta de calcularlo
```

Breusch-Pagan

La homocedasticidad se puede comprobar visualmente con un gráfico de dispersión o, con el test de **Breusch-Pagan**. Para realizar este test **es necesario haber instalado** previamente la librería **lmtest**. El valor p (0.8406) es mayor que 0.05, lo que sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de homocedasticidad. Esto implica que **la varianza de los residuos es constante en todas las variables independientes**.

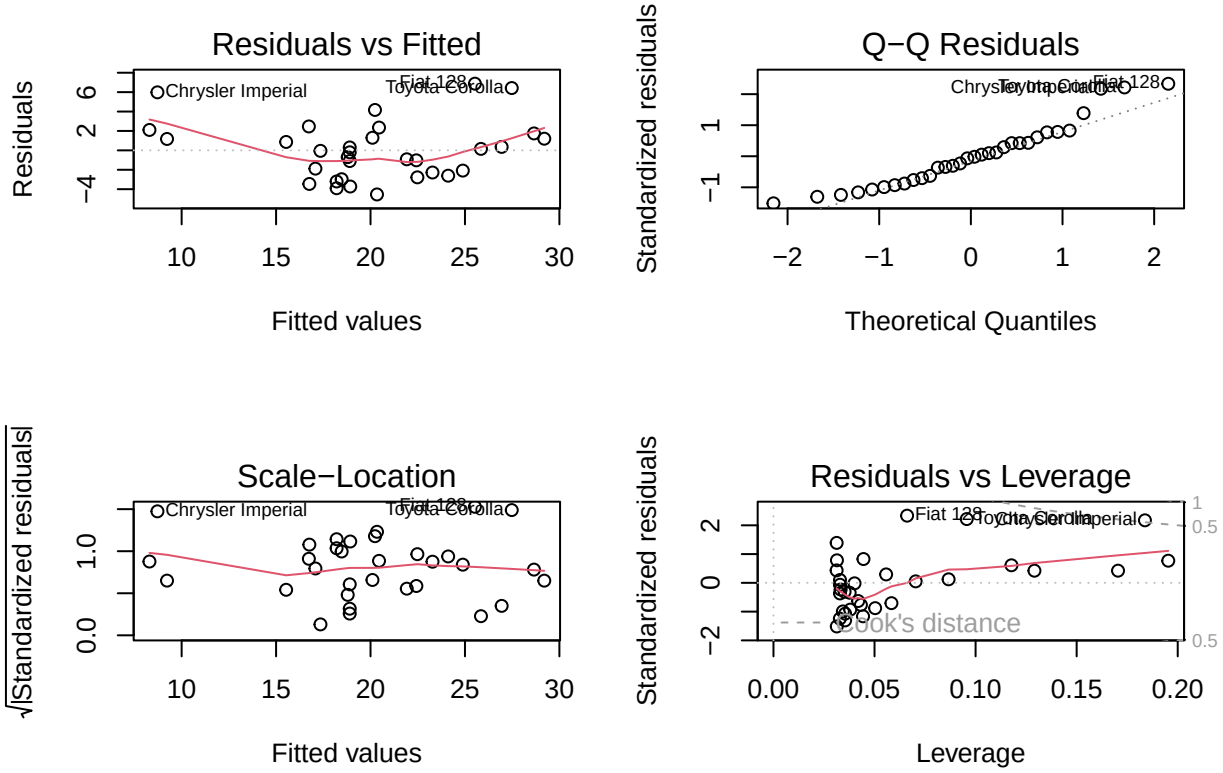
```r
bptest(lm_mtcars)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  lm_mtcars
## BP = 0.040438, df = 1, p-value = 0.8406
```

Gráficas de residuos

Para comprobar la **normalidad** y la **homocedasticidad** podemos utilizar también estas gráficas de residuos. En las dos gráficas de la izquierda se observa que estos se distribuyen de forma aleatoria, sin seguir ningún patrón. La aparición de dispersiones importantes al inicio o final del gráfico son indicadores de lo contrario. Por otro lado, en el gráfico situado en la zona superior derecha (Q-Q Plot) se observa visualmente que los residuos se distribuyen de forma normal.

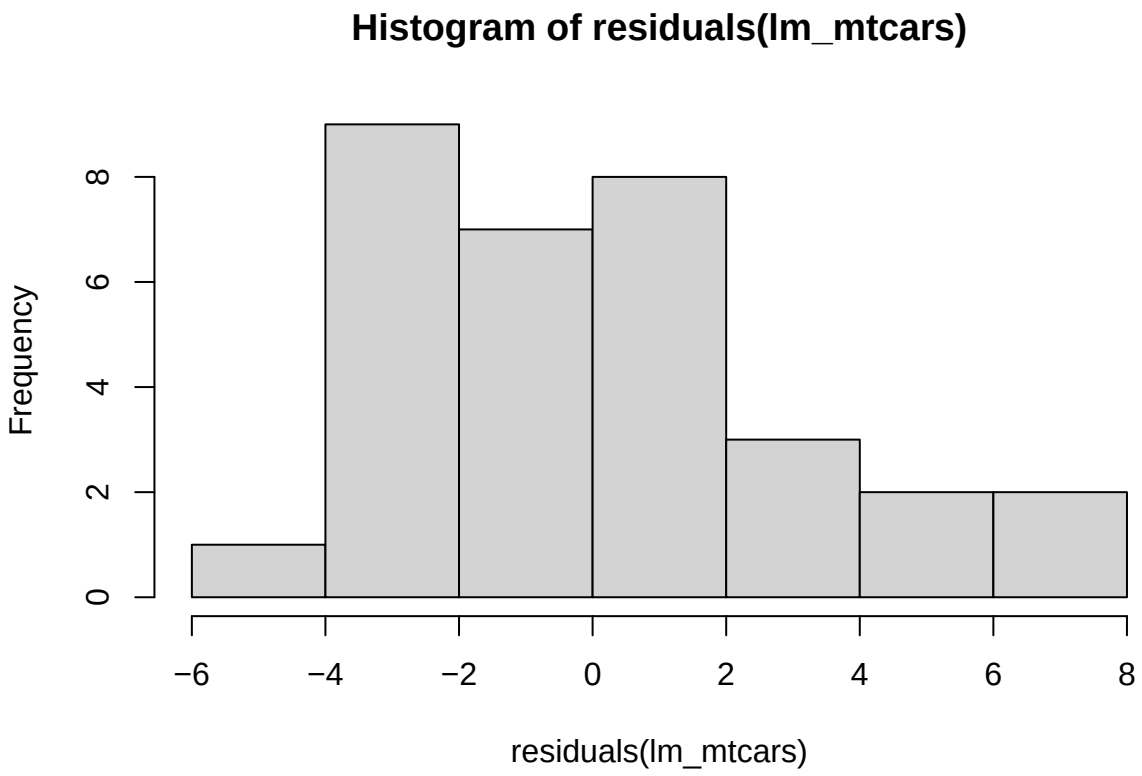
```r
par(mfrow = c(2,2)) # ventana en 2 filas y 2 columnas
plot(lm_mtcars)
```



Histograma de residuos

Podemos observar también el histograma de los residuos para comprobar visualmente si se distribuyen de forma **normal**.

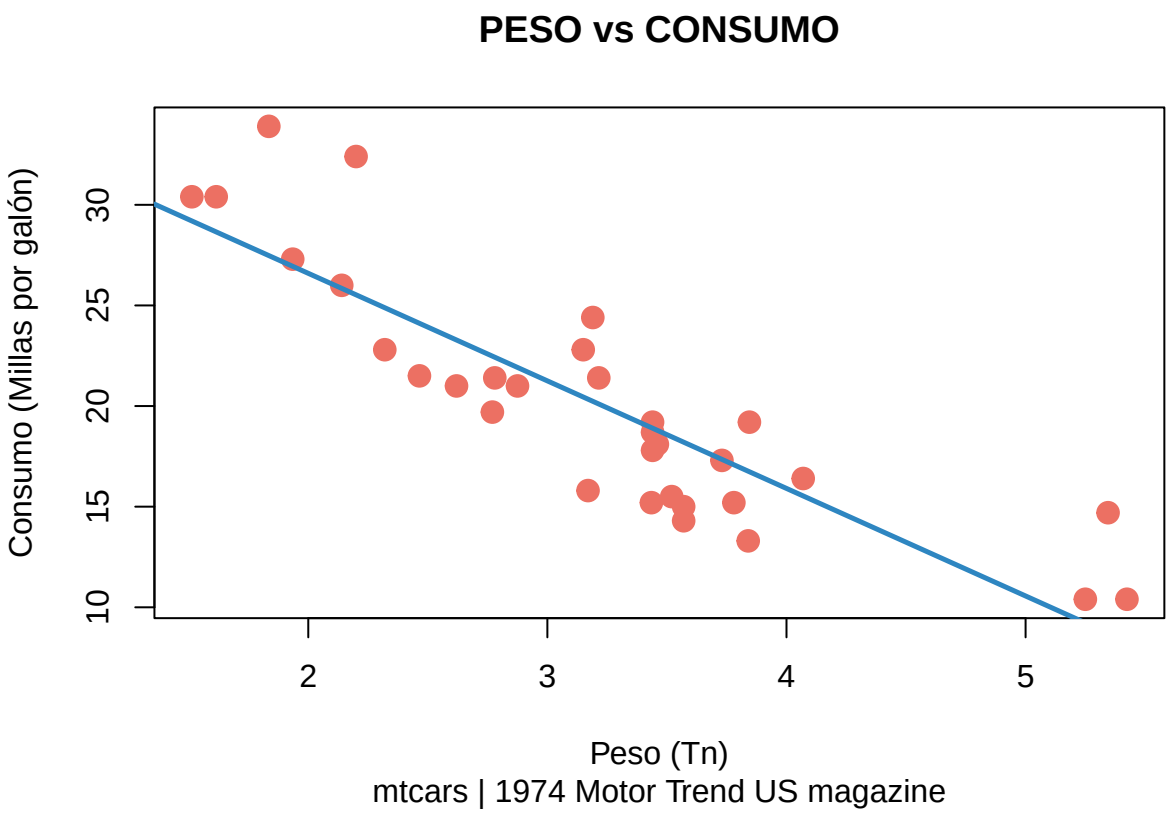
```
hist(residuals(lm_mtcars))
```



Gráfica de la regresión (plot)

Una vez que damos por bueno el modelo podemos, por ejemplo, representar la **recta de regresión** en el gráfico anterior. Para ello utilizamos la función **abline**, que superpondrá la línea el gráfico generado por **plot**. Ambas funciones están de forma nativa en R.

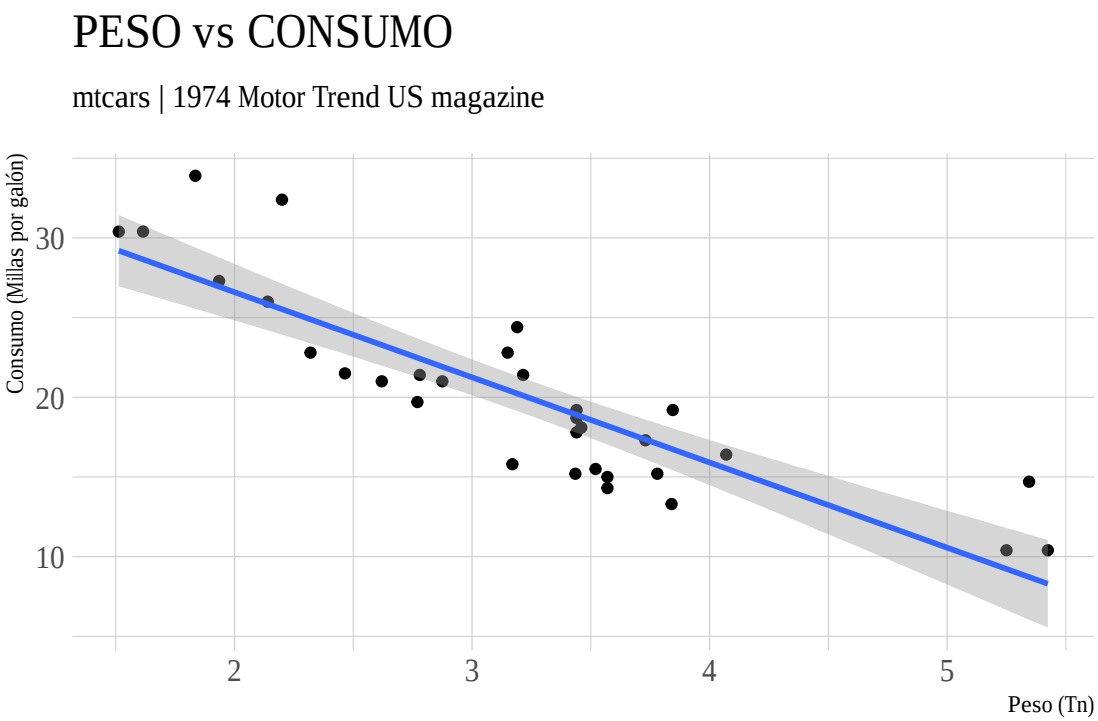
```
par(mfrow = c(1,1))
plot(mtcars$wt, mtcars$mpg, main = "PESO vs CONSUMO",
     sub = "mtcars | 1974 Motor Trend US magazine",
     xlab = "Peso (Tn)", ylab = "Consumo (Millas por galón)",
     col = "#EC7063", pch = 19, type = "p", cex = 1.5)
abline(lm_mtcars, col = "#2E86C1", lwd = 2.5)
```



Gráfica de la regresión (ggplot)

Podemos utilizar la librería **ggplot2** para representar la recta de regresión. En este caso utilizamos la función **geom_smooth**, seleccionando el metodo **lm** para que, con el método de mínimos cuadrados ordinarios (MCO), ajuste una línea de regresión lineal a los datos.

```
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "PESO vs CONSUMO",
       subtitle = "mtcars | 1974 Motor Trend US magazine",
       x = "Peso (Tn)", y = "Consumo (Millas por galón)") +
  theme_ipsum()
```



Estimación

por último, podemos hacer el cáñculo de una predicción de millas por galón para un peso de vehículo de 3.5 toneladas. El resultado sería de **18.58 millas por galón**.

```
wt = 3.5
mpg = 37.2851 - 5.3445 * wt
print(mpg)
```

```
## [1] 18.57935
```

eof, 13/10/2023