

# Laboratorio 3 – Regresión lineal, transformaciones no lineales e interpretación

## Table of contents

Primeros pasos . . . . .	1
Relación entre gasto y tasa de aprobación en matemáticas . . . . .	2
Coefficientes de regresión “a mano” . . . . .	3
Visualización de los estimadores . . . . .	3
Transformaciones no lineales . . . . .	4
Errores estándar y salida de regresión . . . . .	6
Cálculo de errores estándar “a mano” (opcional) . . . . .	6

El propósito de este laboratorio es practicar: estimar **regresiones**, calcular **fórmulas** de MCO “a mano”, **visualizar** la *Función de Regresión Muestral* (Sample Regression Function), usar **transformaciones no lineales** e **interpretar coeficientes**.

## Primeros pasos

Abre un nuevo script de R y carga los paquetes

```
# Instalar paquetes solo si faltan (estilo ECO/EPG)
pkgs <- c("tidyverse", "modelsummary", "broom", "wooldridge", "kableExtra")
to_install <- pkgs[!pkgs %in% rownames(installed.packages())]
if (length(to_install) > 0) {
  install.packages(to_install, repos = "http://cran.us.r-project.org")
}

library(tidyverse)
library(modelsummary)
library(broom)
library(wooldridge)
```

```
library(kableExtra)

# Receta ECO/EPG: forzar tablas estables (LaTeX clásico)
options(modelsummary_factory_default = "kableExtra")
```

Para este laboratorio usaremos datos sobre **gasto escolar** y **tasas de aprobación** de matemáticas en Michigan. Esto está en el conjunto `meap93` del paquete `wooldridge`. Cada observación corresponde a un **distrito escolar** de Michigan.

```
df <- as_tibble(meap93)
```

## Relación entre gasto y tasa de aprobación en matemáticas

Estima el siguiente modelo de regresión:

$$\text{math10} = \beta_0 + \beta_1 \text{expend} + u$$

El código para hacerlo es:

```
est <- lm(math10 ~ expend, data = df)
tidy(est)
```

```
# A tibble: 2 x 5
  term      estimate std.error statistic    p.value
  <chr>      <dbl>    <dbl>    <dbl>    <dbl>
1 (Intercept) 13.4      2.93      4.55 0.00000700
2 expend      0.00246  0.000660    3.72 0.000227
```

```
glance(est)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df logLik   AIC   BIC
  <dbl>      <dbl>    <dbl>    <dbl>    <dbl> <dbl> <dbl> <dbl> <dbl>
1  0.0330      0.0306  10.3      13.8 0.000227     1 -1531. 3067. 3079.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Deberías obtener un coeficiente cercano a 0.00246 para `expend`. **Interpreta** este coeficiente. (Puedes escribir la interpretación como un comentario en tu script `.R`). ¿Te parece un número pequeño, considerando las unidades en que están `math10` y `expend`?

## Coeficientes de regresión “a mano”

Verifica que los coeficientes estimados en **est** coinciden con las fórmulas del libro:

$$\hat{\beta}_0 = \overline{math10} - \hat{\beta}_1 \overline{expend}, \quad \hat{\beta}_1 = \frac{\widehat{cov}(math10, expend)}{\widehat{var}(expend)}$$

Puedes hacerlo escribiendo:

```
beta1 <- cov(df$math10, df$expend) / var(df$expend)
beta0 <- mean(df$math10) - beta1 * mean(df$expend)
beta0
```

```
[1] 13.35923
```

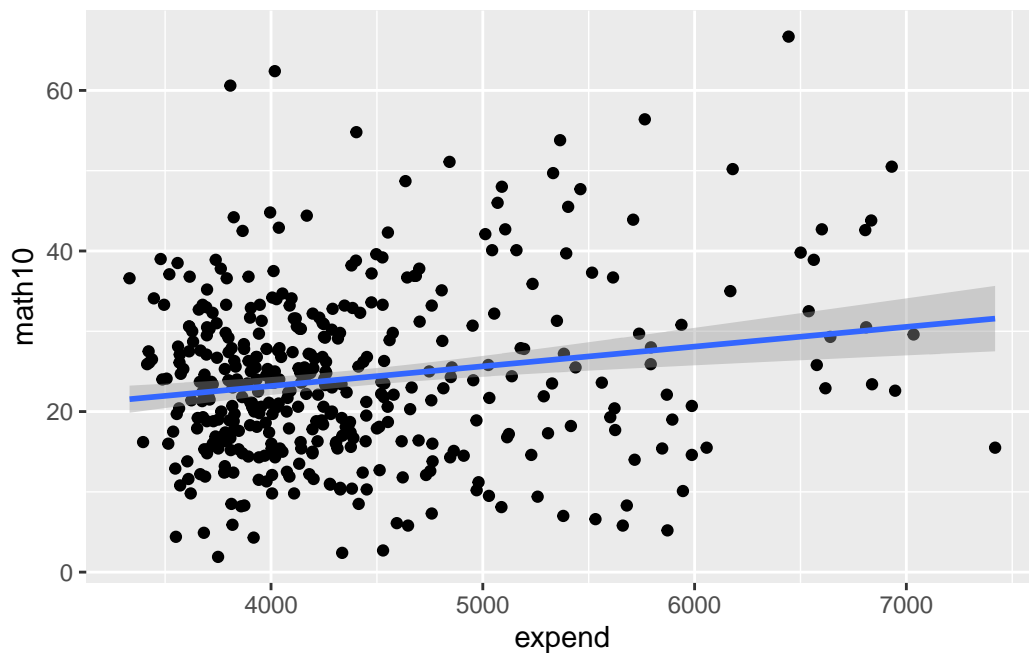
```
beta1
```

```
[1] 0.002455715
```

## Visualización de los estimadores

A menudo es útil visualizar el modelo estimado. Wooldridge lo llama la **Función de Regresión Muestral** (*Sample Regression Function*). Podemos hacerlo con:

```
ggplot(df, aes(expend, math10)) +
  geom_point() +
  geom_smooth(method = "lm")
```



## Transformaciones no lineales

Considera una versión modificada del modelo donde usamos el **logaritmo** del gasto en vez del gasto en niveles. ¿Por qué querríamos usar  $\log(\text{gasto})$ ?

Una razón típica es que pensamos que **cada dólar adicional no tiene el mismo efecto** sobre la tasa de aprobación: los retornos marginales podrían ser **decrecientes** (ver también: *Ley de rendimientos marginales decrecientes*).

Crea la variable logarítmica usando `mutate()`:

```
df <- df %>% mutate(logexpend = log(expend))
```

Ahora estima el modelo nuevamente y vuelve a visualizar (mostrando ambas formas funcionales juntas):

```
est <- lm(math10 ~ logexpend, data = df)
tidy(est)
```

```
# A tibble: 2 x 5
  term      estimate std.error statistic  p.value
  <chr>         <dbl>    <dbl>     <dbl>   <dbl>
```

```
1 (Intercept)    -69.3    26.5    -2.61 0.00929
2 logexpend      11.2     3.17     3.52 0.000475
```

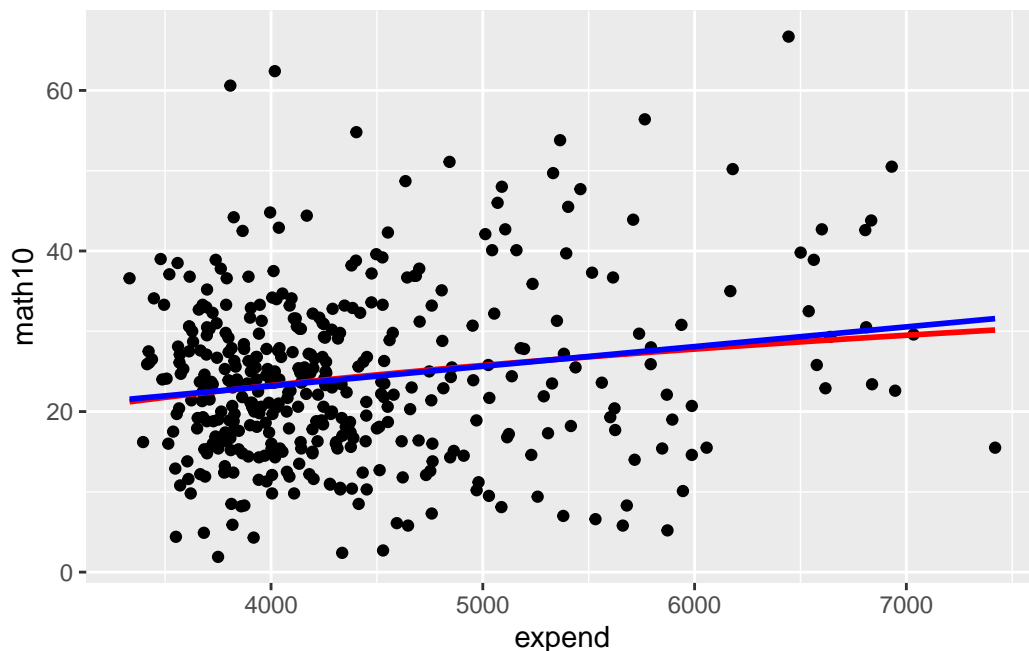
```
glance(est)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df logLik   AIC   BIC
    <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl>
1  0.0297         0.0273  10.3      12.4 0.000475     1 -1531. 3069. 3081.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
# Tabla del modelo (estable para PDF)
modelsummary(est, output = "kableExtra") |>
  kable_styling(full_width = FALSE, position = "center", latex_options = "hold_position")
```

	(1)
(Intercept)	−69.341 (26.530)
logexpend	11.164 (3.169)
Num.Obs.	408
R2	0.030
R2 Adj.	0.027
AIC	3068.8
BIC	3080.8
Log.Lik.	−1531.396
F	12.411
RMSE	10.32

```
# Visualización: forma logarítmica vs lineal (ambas sobre el mismo gráfico)
ggplot(df, aes(expend, math10)) +
  geom_point() +
  stat_smooth(method = "lm", se = FALSE, formula = y ~ log(x), col = "red") +
  stat_smooth(method = "lm", se = FALSE, col = "blue")
```



¿Cuál es la interpretación de  $\beta_1$  en este nuevo modelo? (Agrégala como comentario en tu script).

### Errores estándar y salida de regresión

Finalmente, mira el **error estándar**, el **estadístico t** y los **p-values** asociados a los parámetros  $\beta_0$  y  $\beta_1$ . El **p.value** reportado en `tidy(est)` prueba la hipótesis:

$$H_0 : \beta_1 = 0 \quad \text{vs} \quad H_a : \beta_1 \neq 0$$

¿Un mayor gasto escolar incrementa significativamente la tasa de aprobación en matemáticas?

### Cálculo de errores estándar “a mano” (opcional)

Si te queda tiempo, intenta calcular los errores estándar “a mano” según las fórmulas del texto. Para ello necesitamos calcular: **sig** (la desviación estándar de  $u$ ), **n** (tamaño muestral), **SST<sub>x</sub>** ( $(n - 1)$  veces la varianza de `logexpend`), y la suma de cuadrados de `logexpend`:

```
n <- dim(df)[1]
sig <- sqrt(sum(est$residuals^2) / (n - 2)) # alternativa: glance(est)$sigma
SSTx <- (n - 1) * var(df$logexpend)
sumx2 <- sum(df$logexpend^2)
```

El error estándar del intercepto se calcula con:

```
sqrt((sig^2 * (1/n) * sumx2) / SSTx)
```

```
[1] 26.53013
```

Y el error estándar de la pendiente (para logexpend) es:

```
sqrt(sig^2 / SSTx)
```

```
[1] 3.169011
```