Modelo Lineal Simple

Camilo Vega

2023-04-01

```r
# Carga de paquetes necesarios para el código
library(tidyverse) # Conjunto de paquetes para manipulación de datos
library(ggside) # Extiende ggplot2 con gráficos adicionales
library(GGally) # Extiende ggplot2 con gráficos de matriz
library(ggdist) # Extiende ggplot2 con gráficos de distribución
library(tidyquant) # Paquete de finanzas para análisis cuantitativo de datos
library(paqueteMET) # Paquete para el análisis de series temporales
library(skimr) # Paquete para el análisis exploratorio de datos
library(knitr) # Paquete para creación de tablas en formato de salida
library(bestNormalize) # Busca la mejor transformación para noralización
library(rlang)
library(broom)
library(qqplotr)
library(gridExtra)
library(grid)

# Se crea un objeto "datos_vivienda" que contiene los datos de vivienda4
datos_vivienda <- vivienda4

source("funciones_personalizadas.R")
```

```r
datos_vivienda_mod <- datos_vivienda |>
    filter(zona == "Zona Sur",
           tipo == "Apartamento")
```

```r
set.seed(4321)
norm_preciom <- bestNormalize(datos_vivienda_mod$preciom, allow_orderNorm = FALSE)

norm_preciom$chosen_transform
```

```
## Standardized Box Cox Transformation with 1065 nonmissing obs.:
##  Estimated statistics:
##  - lambda = -0.567132
##  - mean (before standardization) = 1.672926
##  - sd (before standardization) = 0.0144479
```

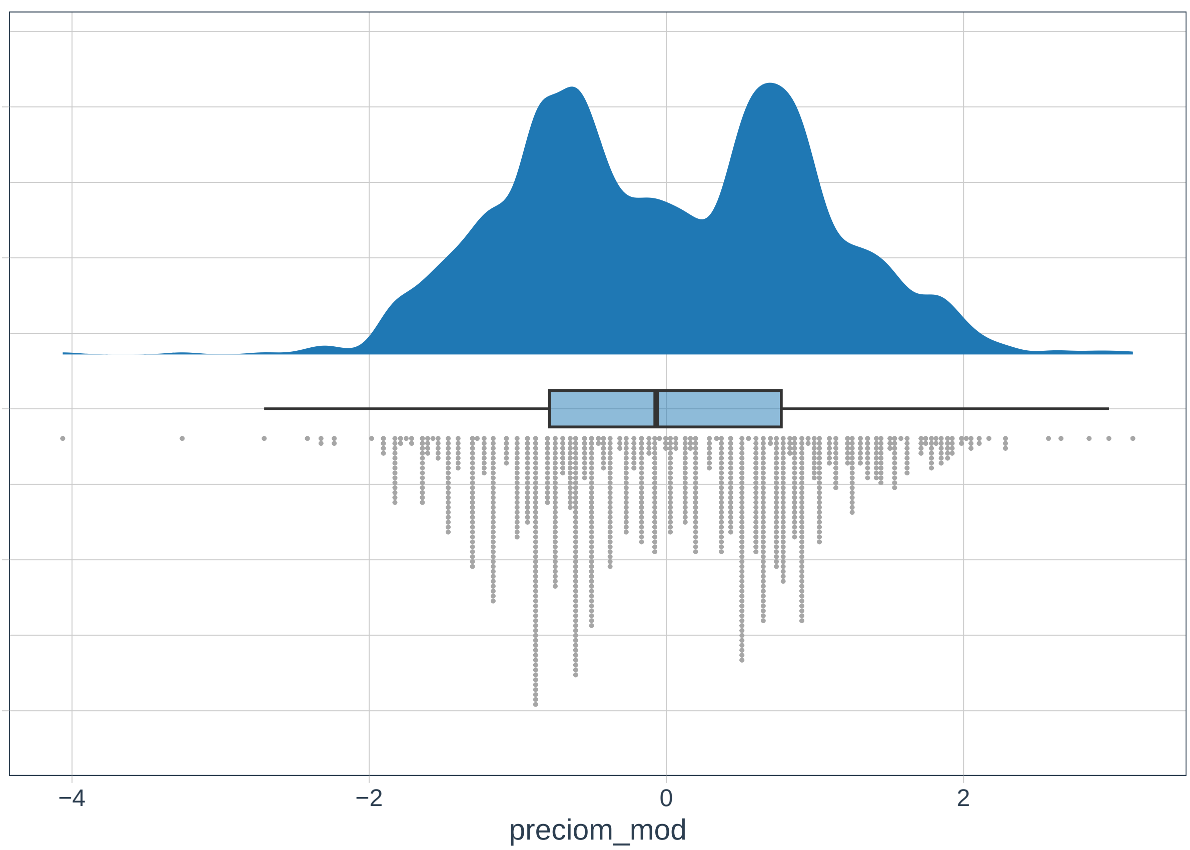
```r
set.seed(4321)
norm_areaconst <- bestNormalize(datos_vivienda_mod$areaconst, allow_orderNorm = FALSE)

norm_areaconst$chosen_transform
```

```
## Standardized Box Cox Transformation with 1065 nonmissing obs.:
##  Estimated statistics:
##  - lambda = -0.9999576
##  - mean (before standardization) = 0.9857638
##  - sd (before standardization) = 0.003228487
```

$$y(\lambda) = \begin{cases} \frac{y^\lambda - 1}{\lambda}, & \text{if } \lambda \neq 0 \\ \ln(y), & \text{if } \lambda = 0 \end{cases}$$

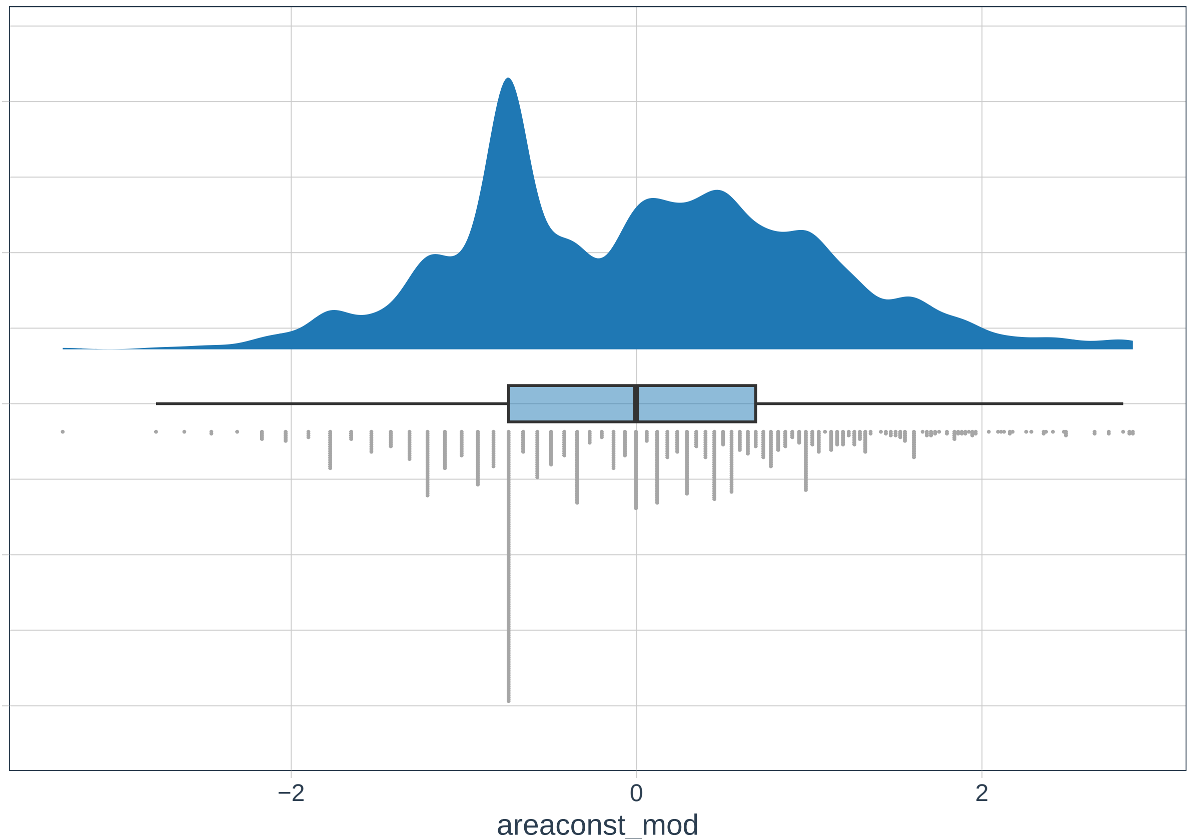
```
datos_vivienda_mod <- datos_vivienda_mod |>
    mutate(preciom_mod = norm_preciom$x.t,
           areaconst_mod = norm_areaconst$x.t)
```

```
gg_rain_cloud(
    datos_vivienda_mod,
    preciom_mod); summary_table(
        datos_vivienda_mod,
        preciom_mod)
```



min	q1	median	mean	q3	max	skewness
-4.06	-0.79	-0.07	0	0.77	3.14	0.02

```
gg_rain_cloud(
    datos_vivienda_mod,
    areaconst_mod); summary_table(
        datos_vivienda_mod,
        areaconst_mod)
```



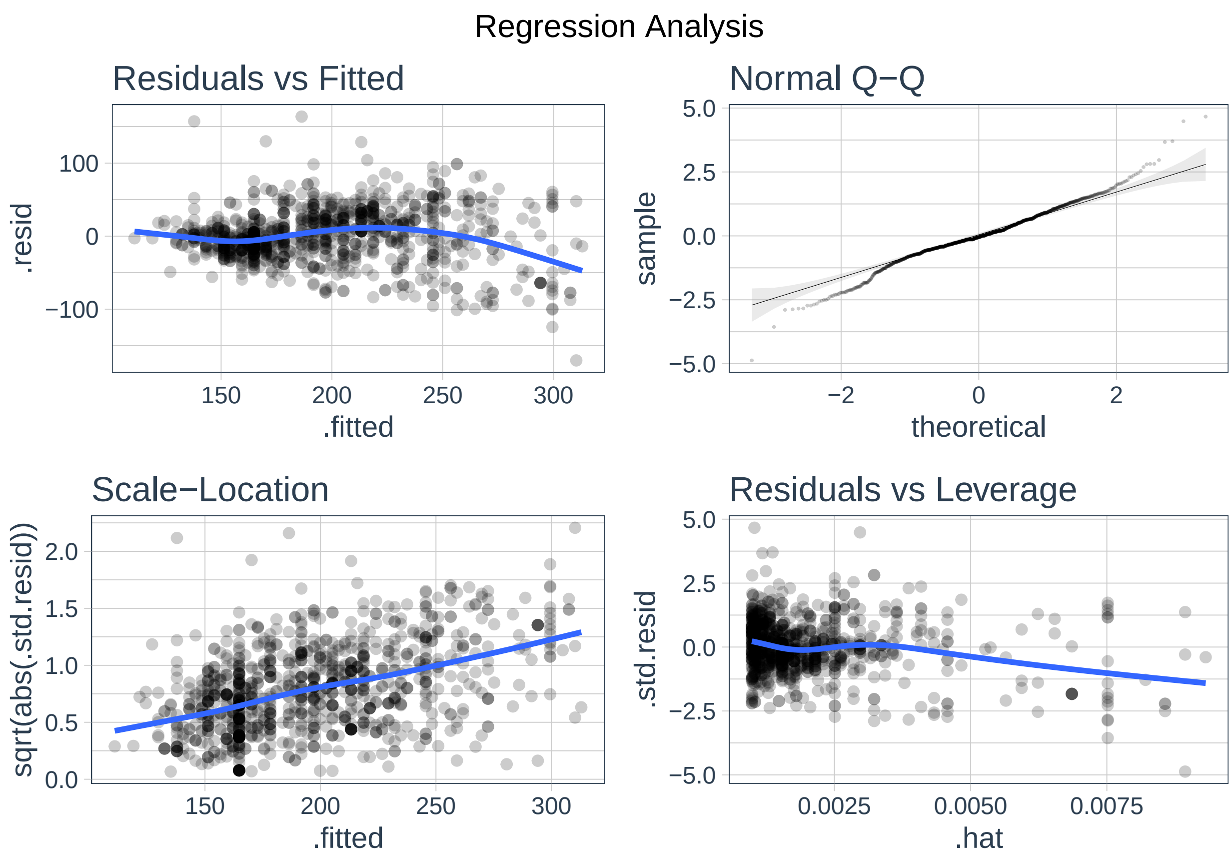
min	q1	median	mean	q3	max	skewness
-3.32	-0.74	0	0	0.69	2.87	0.2

```
datos_lin_lin <- datos_vivienda_mod |>
    select(preciom, areaconst) |>
    remove_outliers(preciom) |>
    remove_outliers(areaconst)

lm_lin_lin <- lm(preciom ~ areaconst, datos_lin_lin)

glance(lm_lin_lin) |>
    mutate_all(~ round(.,4)) |>
    select(-10, -2) |>
    kable() ; tidy(lm_lin_lin,
                conf.int = TRUE) |>
    mutate(across(where(is.numeric), ~ round(.,4))) |>
    kable()
```

r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	df.residual	nobs
0.5708	35.1033	1348.28	0	1	-5055.869	10117.74	10132.51	1014	1016

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	3.1770	5.3467	0.5942	0.5525	-7.3149	13.6689
areaconst	2.6935	0.0734	36.7189	0.0000	2.5496	2.8374

```
datos_lin_lin_aug <- augment(lm_lin_lin)

reg_analysis(datos_lin_lin_aug)
```



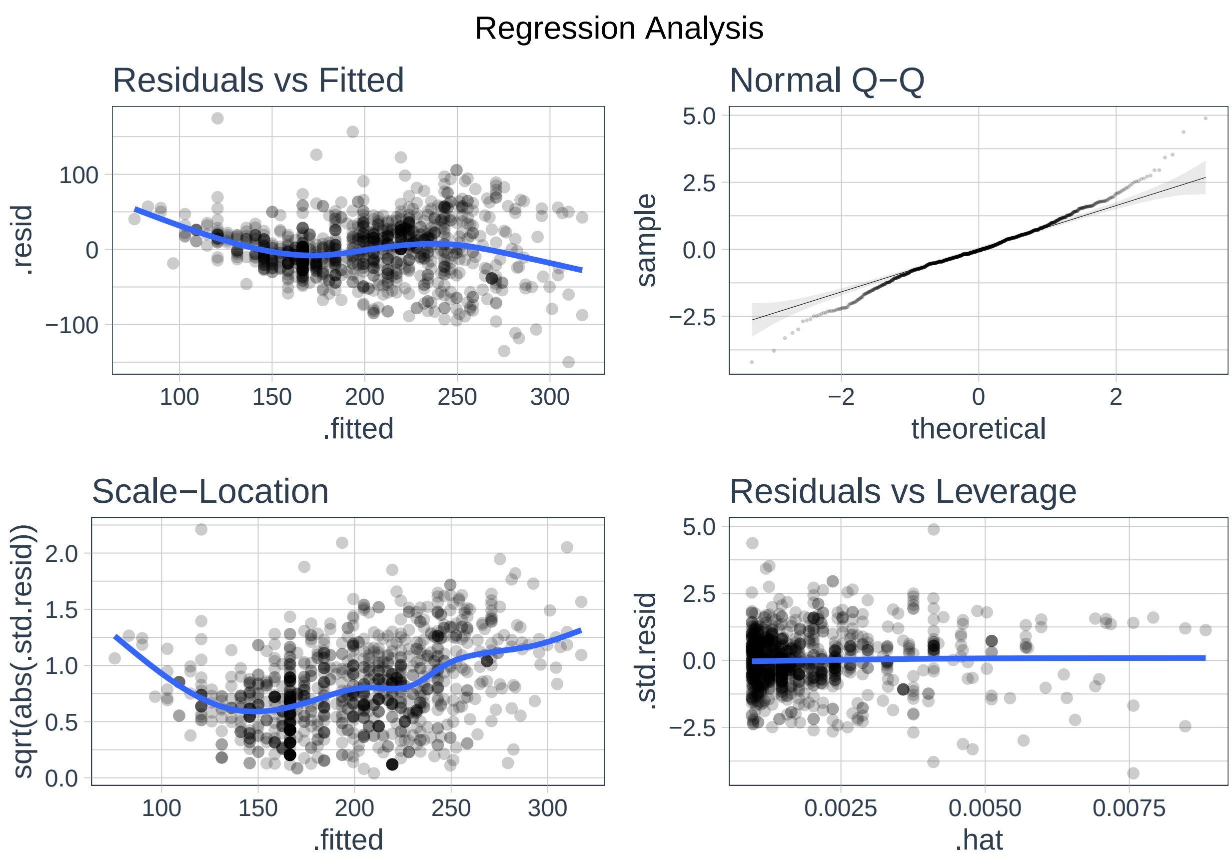

```
datos_lin_box <- datos_vivienda_mod |>
    select(preciom, areaconst_mod) |>
    remove_outliers(preciom) |>
    remove_outliers(areaconst_mod)

lm_lin_box <- lm(preciom ~ areaconst_mod, datos_lin_box)

glance(lm_lin_box) |>
    mutate_all(~ round(.,4)) |>
    select(-10, -2) |>
    kable(); tidy(lm_lin_lin,
```

```
                   conf.int = TRUE) |>
    mutate(across(where(is.numeric), ~ round(.,4))) |>
    kable()
```

r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	df.residual	nobs
0.5866	35.7896	1482.879	0	1	-5230.436	10466.87	10481.73	1045	1047

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	3.1770	5.3467	0.5942	0.5525	-7.3149	13.6689
areaconst	2.6935	0.0734	36.7189	0.0000	2.5496	2.8374

```
datos_lin_box_aug <- augment(lm_lin_box)

reg_analysis(datos_lin_box_aug)
```



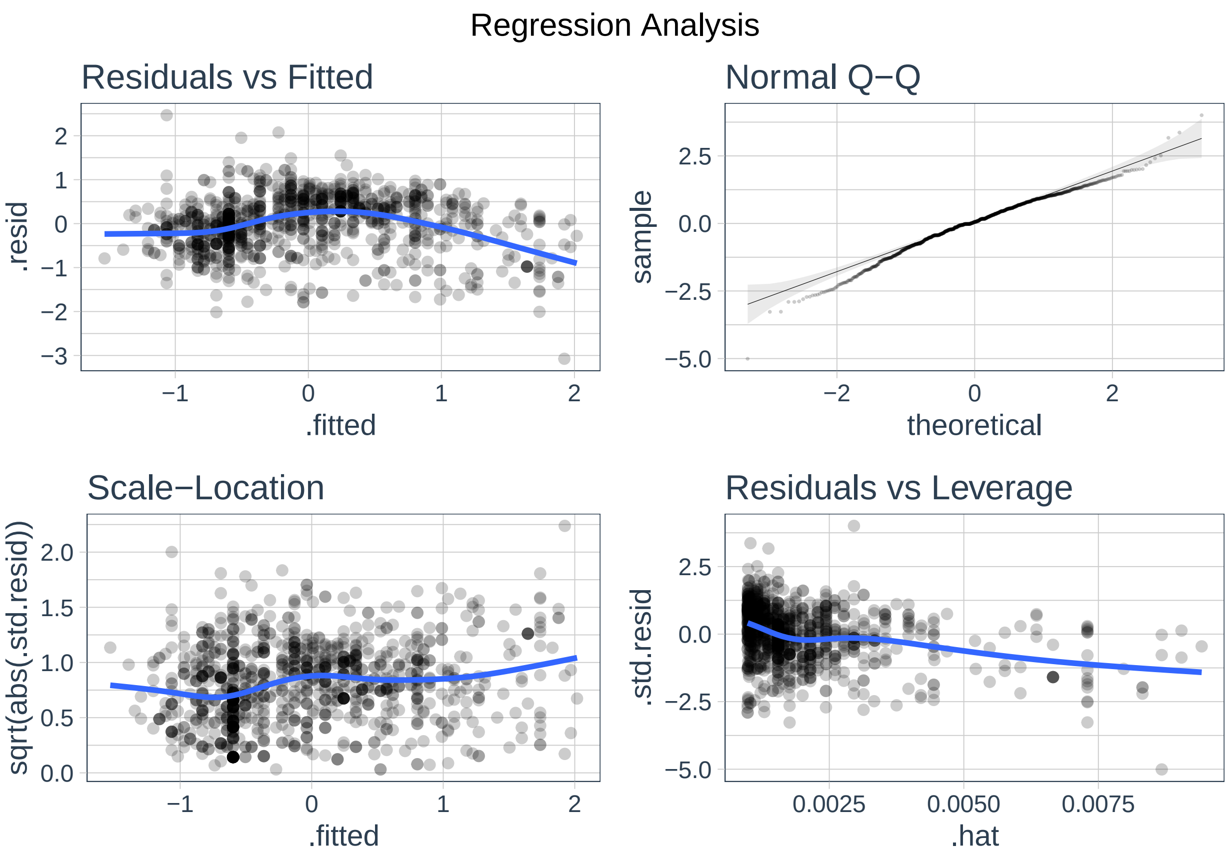

```
datos_box_lin <- datos_vivienda_mod |>
    select(preciom_mod, areaconst) |>
    remove_outliers(preciom_mod) |>
    remove_outliers(areaconst)

lm_box_lin <- lm(preciom_mod ~ areaconst, datos_box_lin)
```

```
glance(lm_box_lin) |>
    mutate_all(~ round(.,4)) |>
    select(-10, -2) |>
    kable(); tidy(lm_box_lin,
                  conf.int = TRUE) |>
    mutate(across(where(is.numeric), ~ round(.,4))) |>
    kable()
```

r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	df.residual	nobs
0.5697	0.6165	1344.944	0	1	-951.1484	1908.297	1923.074	1016	1018

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	-3.3997	0.0931	-36.5140	0	-3.5824	-3.2170
areaconst	0.0467	0.0013	36.6735	0	0.0442	0.0492

```
datos_box_lin_aug <- augment(lm_box_lin)

reg_analysis(datos_box_lin_aug)
```

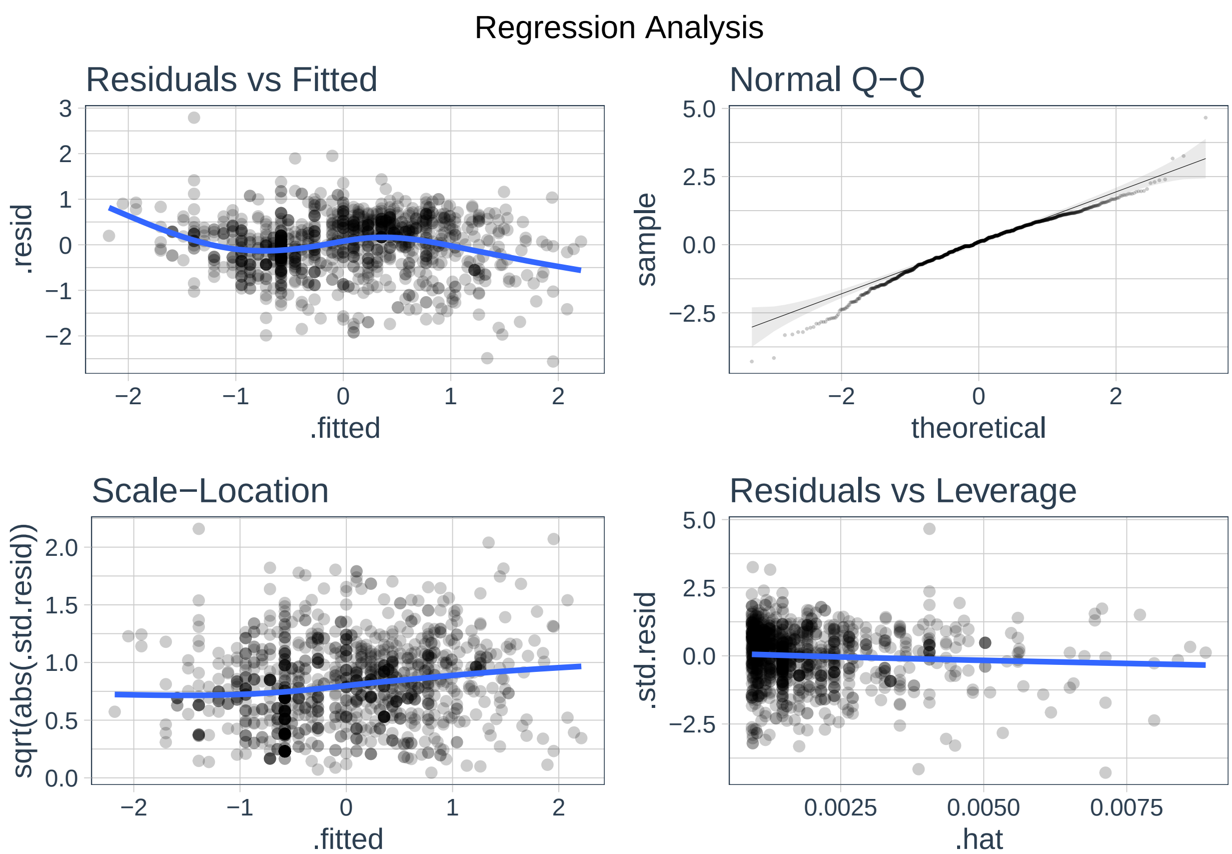


```
datos_box_box <- datos_vivienda_mod |>
    select(preciom_mod, areaconst_mod) |>
    remove_outliers(preciom_mod) |>
    remove_outliers(areaconst_mod)

lm_box_box <- lm(preciom_mod ~ areaconst_mod, datos_box_box)

glance(lm_box_box) |>
    mutate_all(~ round(.,4)) |>
    select(-10, -2) |>
    kable(); tidy(lm_box_box,
                  conf.int = TRUE) |>
    mutate(across(where(is.numeric), ~ round(.,4))) |>
    kable()
```

r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	df.residual	nobs
0.6195	0.5998	1717.925	0	1	-958.6084	1923.217	1938.106	1055	1057

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.0023	0.0185	0.1238	0.9015	-0.0339	0.0385
areaconst_mod	0.7842	0.0189	41.4479	0.0000	0.7470	0.8213

```
datos_box_box_aug <- augment(lm_box_box)

reg_analysis(datos_box_box_aug)
```




```
# library(tidymodels)
# library(tidyr)
# library(dplyr)
#
# # Definir el conjunto de datos
# data(mtcars)
#
# # Dividir los datos en un conjunto de entrenamiento y un conjunto de prueba
# set.seed(123)
# car_split <- initial_split(mtcars, prop = 0.8)
# car_train <- training(car_split)
# car_test <- testing(car_split)
#
# # Especificar los motores y grids de hiperparámetros
# models <- list(
#   lm = linear_reg() %>% set_engine("lm"),
#   brulee = linear_reg() %>% set_engine("brulee"),
#   gls = linear_reg() %>% set_engine("gls"),
#   keras = linear_reg() %>% set_engine("keras"),
#   stan = linear_reg() %>% set_engine("stan")
# )
#
# param_grid <- list(
#   lm = list(
#     penalty = c(0, 0.01, 0.1),
#     mixture = c(1, 0.5, 0)
```

```
#   ),
#   brulee = list(
#     lambda = c(0.01, 0.1, 1),
#     dropout = c(0.1, 0.2, 0.5)
#   ),
#   gls = list(
#     correlation = c("corAR1", "corARMA"),
#     p = c(1, 2)
#   ),
#   keras = list(
#     epochs = c(10, 20, 30),
#     dropout = c(0.1, 0.2, 0.5)
#   ),
#   stan = list(
#     chains = c(2, 4, 8),
#     iter = c(200, 500, 1000)
#   )
# )
#
# # Ajustar los modelos con 10 combinaciones de hiperparámetros
# results <- models %>%
#   tune_grid(
#     resamples = vfold_cv(car_train, v = 10),
#     grid = param_grid,
#     metrics = metric_set(rmse),
#     control = control_grid(save_pred = TRUE)
#   ) %>%
#   fit_resamples()
#
# # Comparar los resultados
# all_results <- results %>%
#   collect_metrics() %>%
#   mutate(model = map_chr(.metrics, ~.x$recipe)) %>%
#   separate(model, c("model", "recipe"), sep = "_on_") %>%
#   mutate_at(vars(model, recipe), as.factor)
#
# # Ver el resumen de los resultados
# summary_results <- all_results %>%
#   group_by(model, recipe) %>%
#   summarize(mean_rmse = mean(.estimate), .groups = "drop") %>%
#   arrange(mean_rmse)
#
# print(summary_results)
#
# # Seleccionar el mejor modelo
# best_model <- all_results %>%
#   filter(mean_rmse == min(mean_rmse))
#
# print(best_model)
#
# # Evaluar el modelo seleccionado en el conjunto de prueba
# final_model <- best_model %>%
#   select(-c(recipe, mean_rmse)) %>%
```

```
#   pull(model) %>%
#   extract_model(models) %>%
#   set_args(maxit = 1000) %>%
#   fit(car_train)
#
# final_predictions <- final_model %>%
#   predict(new_data = car_test)
#
# # Calcular la métrica de evaluación en el conjunto de prueba
# final_rmse <- final_predictions %>%
#   bind_cols(car_test) %>%
#   metrics(truth = mpg, estimate = .pred) %>%
#   select(.metric, .estimator, .estimate) %>%
#   filter(.metric == "rmse")
#
# print(final_rmse)
```