Modelo Lineal Simple

Camilo Vega

Contents

Introduccion

Este documento presenta un análisis destinado a la construcción de un modelo lineal simple utilizando los datos del dataframe "vivienda4" del paquete "paqueteMET". El objetivo es determinar el modelo más adecuado para explicar la relación entre los metros cuadrados de una vivienda y su precio en millones de pesos colombianos.

Carga de librerias y funciones personalizadas

En este análisis, se utilizan las librería listadas en el siguiente código. Además, para facilitar la creación de visualizaciones, se emplean las funciones personalizadas, las cuales se encuentran en el archivo funciones_personalizadas.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) # Conjunto de herramientas para programación en R
library(broom) # Convierte resultados de modelos en tablas y gráficos
library(qqplotr) # Paquete para gráficos QQ-plot
library(gridExtra) # Paquete para la combinación de gráficos
library(grid) # Paquete para la manipulación de grillas de gráficos
library(tidymodels) # Conjunto de paquetes para modelado estadístico
library(multilevelmod) # Paquete para ajuste de modelos de efectos mixtos

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

# Carga de funciones personalizadas
source("funciones_personalizadas.R")
```

Ingenieria de caracteristicas

Teniendo en cuenta las recomendaciones del análisis exploratorio inicial detallado en el documento `analisis_exploratorio.pdf`, se han realizado las siguientes transformaciones en los datos del dataframe `vivienda4` para construir un modelo lineal simple adecuado.

Filtrado

Se ha llevado a cabo un filtrado de los datos del dataframe para incluir únicamente aquellos que corresponden a apartamentos ubicados en la zona sur. De esta manera, se podrá trabajar exclusivamente con la información relevante para el análisis y construcción del modelo lineal simple.

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

Transformación de datos

Para transformar los datos y lograr una aproximación lo más cercana posible a una distribución normal, se utilizará el paquete de R llamado `bestNormalize`. De esta manera, se podrá mejorar la calidad del análisis y la construcción del modelo lineal simple.

Transformación preciom

```r
# Fijar la semilla para reproducibilidad
set.seed(4321)

# Aplicar bestNormalize a los datos de precios de vivienda
norm_preciom <- bestNormalize(datos_vivienda_mod$preciom, allow_orderNorm = FALSE)

# Observar la mejor transformación
norm_preciom$chosen_transform
```

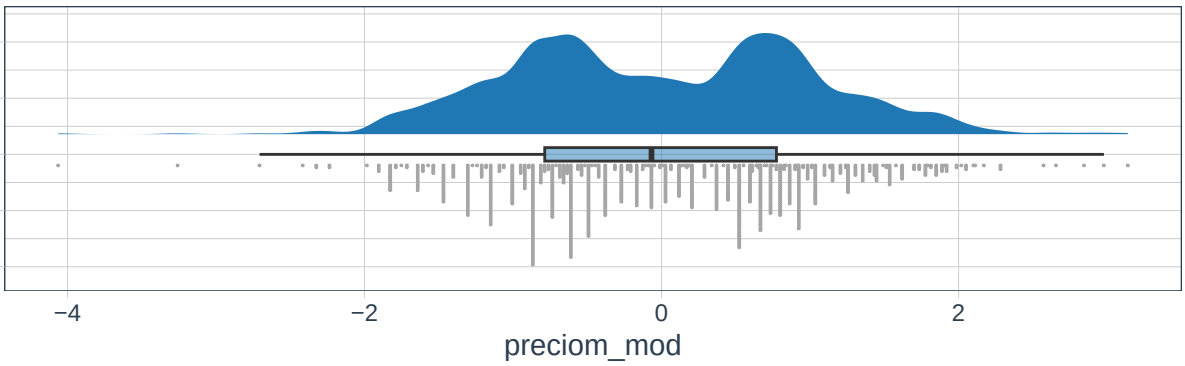
```
## Standardized Box Cox Transformation with 1065 nonmissing obs.:
##  Estimated statistics:
##  - lambda = -0.567132
##  - mean (before standardization) = 1.672926
##  - sd (before standardization) = 0.0144479
```

Luego de aplicar diferentes métodos de transformación en la variable `norm_preciom` se ha determinado que la mejor opción es el método de Box Cox, con un valor de λ de -0.567132. Esta información puede ser representada mediante la siguiente fórmula:

$$norm\_preciom\_mod = \frac{y^{-0.567132} - 1}{-0.567132}$$

Se procede a graficar la distribución resultante y a calcular las medidas de resumen correspondientes. De esta manera, se podrá visualizar y comprender mejor la distribución de los datos transformados.

```r
# Agregando preciom_mod al a los datos
datos_vivienda_mod <- datos_vivienda_mod |>
    mutate(preciom_mod = norm_preciom$x.t)

# Grafica de densidad y tabla de resumen
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

Al observar las medidas de tendencia central de la distribución transformada, se puede apreciar que se acercan a cero y que la asimetría desaparece. Sin embargo, es importante mencionar que la transformación de Box Cox hace más evidente la presencia de una bimodalidad en los datos, lo que sugiere que hay variables que no han sido consideradas y que pueden estar afectando la variable en cuestión.

Transformación areaconst

```
# Fijar la semilla para reproducibilidad
set.seed(4321)

# Aplicar bestNormalize a los datos de área de vivienda
norm_areaconst <- bestNormalize(datos_vivienda_mod$areaconst, allow_orderNorm = FALSE)

# Observar la mejor transformación
norm_areaconst$chosen_transform
```

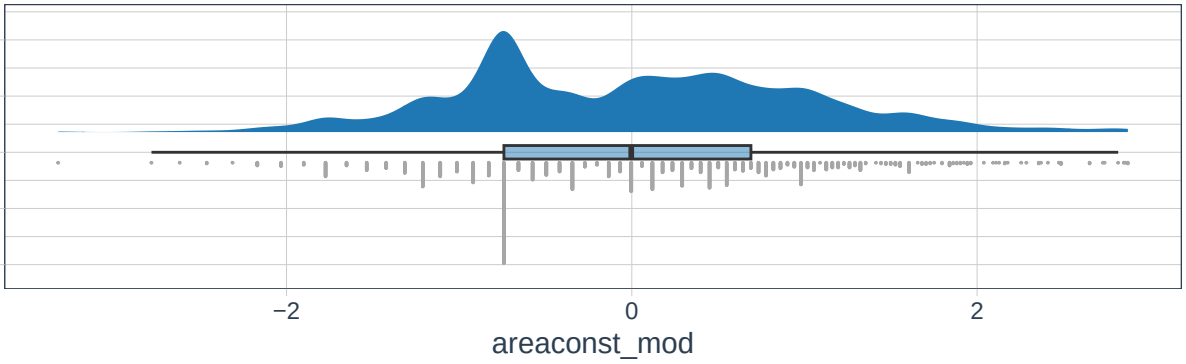
```
## Standardized Box Cox Transformation with 1065 nonmissing obs.:
##  Estimated statistics:
##  - lambda = -0.9999576
##  - mean (before standardization) = 0.9857638
##  - sd (before standardization) = 0.003228487
```

Luego de aplicar diferentes métodos de transformación en la variable `areaconst` se ha determinado que la mejor opción es el método de Box Cox, con un valor de λ de -0.9999576. Esta información puede ser representada mediante la siguiente fórmula:

$$areaconst\_mod = \frac{y^{-0.9999576} - 1}{-0.9999576}$$

Se procede a graficar la distribución resultante y a calcular las medidas de resumen correspondientes. De esta manera, se podrá visualizar y comprender mejor la distribución de los datos transformados.

```r
# Agregando datos_vivienda_mod al a los datos
datos_vivienda_mod <- datos_vivienda_mod |>
    mutate(areaconst_mod = norm_areaconst$x.t)

# Grafica de densidad y tabla de resumen
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

Al observar las medidas de tendencia central de la distribución transformada, se puede apreciar que se acercan a cero y que la asimetría desaparece. Sin embargo, es importante mencionar que la transformación de Box Cox hace más evidente la presencia de una bimodalidad en los datos, lo que sugiere que hay variables que no han sido consideradas y que pueden estar afectando la variable en cuestión.

Construcción de Modelos iniciales

En esta primera fase, se crearán un total de cuatro modelos lineales simples, los cuales se basarán en distintas combinaciones de transformaciones de los datos de precio y área, identificadas de la siguiente manera:

- lin_lin: Precio sin transformar, Área sin transformar
- lin_box: Precio sin transformar, Área transformada con el método de Box Cox
- box_lin: Precio transformado con el método de Box Cox, Área sin transformar
- box_box: Precio transformado con el método de Box Cox, Área transformada con el método de Box Cox

Para cada una de las anteriores se aplicara el siguiente procedimiento:

1. Producción de objetos en R para creación de modelo analisis del mismo.
    - 1.A Creación de un data frame con los datos.
    - 1.B Eliminación de outliers usando metodo de rango intercuartil ($\pm 1.5 * IQR$)
    - 1.C Creación del modelo lineal simple.

- 1.D Creación tablas resumen del modelo (coeficientes, intervalos de confianza (95%), p-value coeficientes, p-value, R^2)
- 1.E Creación graficas supuesto residuales.
- 1.F Visualización del modelo.

2. Interpretación del modelo.

lin_lin

Creación Objetos R lin_lin

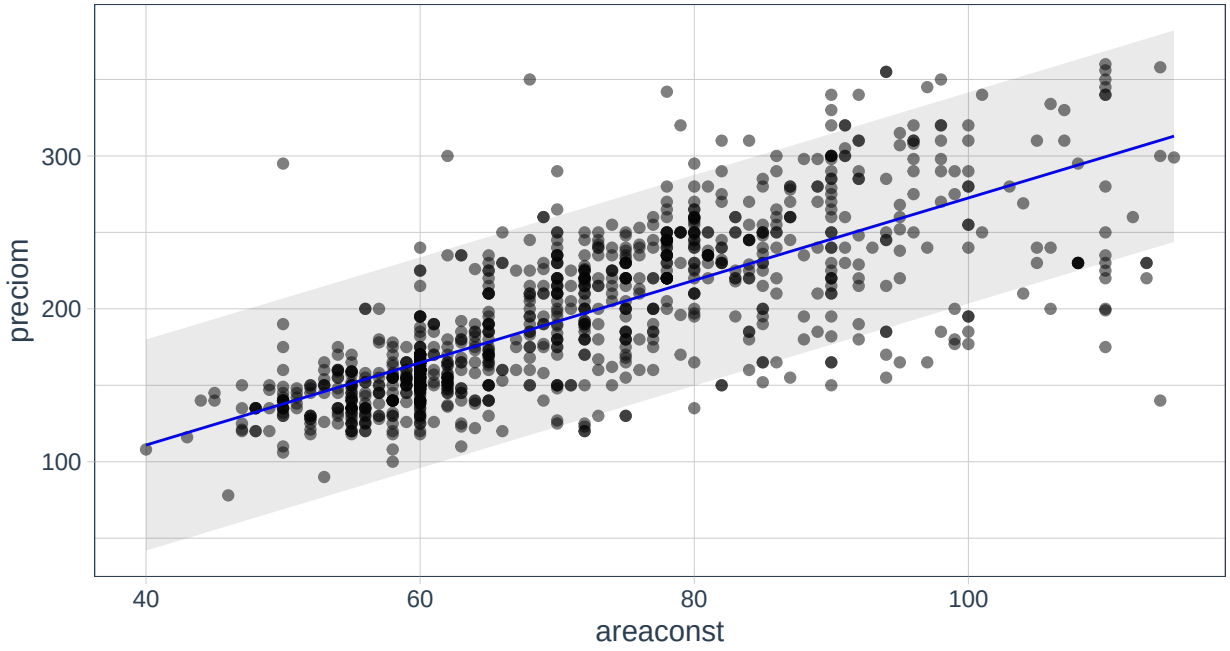
```
# Seleccionando datos y removiendo outliers
datos_lin_lin <- datos_vivienda_mod |>
    select(preciom, areaconst) |>
    remove_outliers(preciom) |>
    remove_outliers(areaconst)

# Creando modelo
lm_lin_lin <- lm(preciom ~ areaconst, datos_lin_lin)

# Agregando al data frame datos del modelo
datos_lin_lin_aug <- augment(lm_lin_lin) |>
  bind_cols(predict(lm_lin_lin, datos_lin_lin, interval = "prediction") |>
              as_tibble() |>
              select(-fit))

# Tabla resumen modelo
tll1 <- glance(lm_lin_lin) |>
    mutate_all(~ round(.,4)) |>
    select(-10, -2) |>
    kable()

# Tabla resumen coeficientes
tll2 <- tidy(lm_lin_lin,
                  conf.int = TRUE) |>
    mutate(across(where(is.numeric), ~ round(.,4))) |>
    kable()

# Grafica modelo
pll1 <- gg_lm_plot(datos_lin_lin_aug, areaconst, preciom, .fitted, lwr, upr)
```

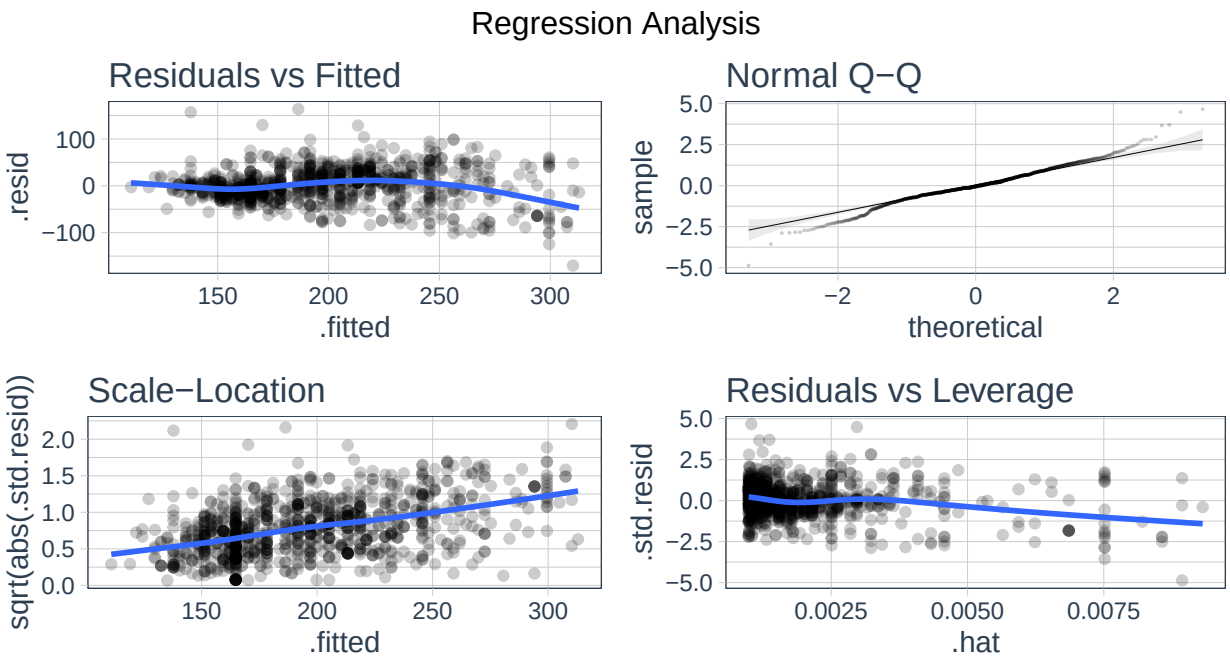
Tablas y Graficas lin_lin

```
tll1; tll2; pll1; reg_analysis(
  datos_lin_lin_aug)
```

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





Interpretación modelo lin_lin

Se observa que el intercepto no difiere significativamente de cero, lo cual es coherente ya que a medida que el área de una vivienda se aproxima a cero, su precio también tiende a disminuir.

Respecto al coeficiente del área construida, se aprecia que el p-valor tiende a cero, lo que sugiere que existe una relación entre el área de una vivienda y su precio. Además, se observa que los valores del coeficiente son positivos, tanto para el valor del coeficiente como para los intervalos de confianza, lo que indica que a medida que aumenta el área de una vivienda, también lo hace su precio.

En cuanto al coeficiente de determinación R^2, se puede notar que su valor no es muy alto, lo que indica que el modelo es capaz de explicar aproximadamente el 57% de la variación en los precios, igualmente el valor p del modelo tiende a cero lo cual nos indica que el modelo a pesar de no explicar en gran medida la variación se puede usar.

Por último, se ha realizado un análisis gráfico en el que se ha observado que a medida que las predicciones son más altas, los residuos tienden a ser más dispersos. Además, se ha detectado la presencia de valores extremos que afectan la regresión y que generan residuos no normales en los valores extremos.

lin_box

Creación Objetos R lin_box

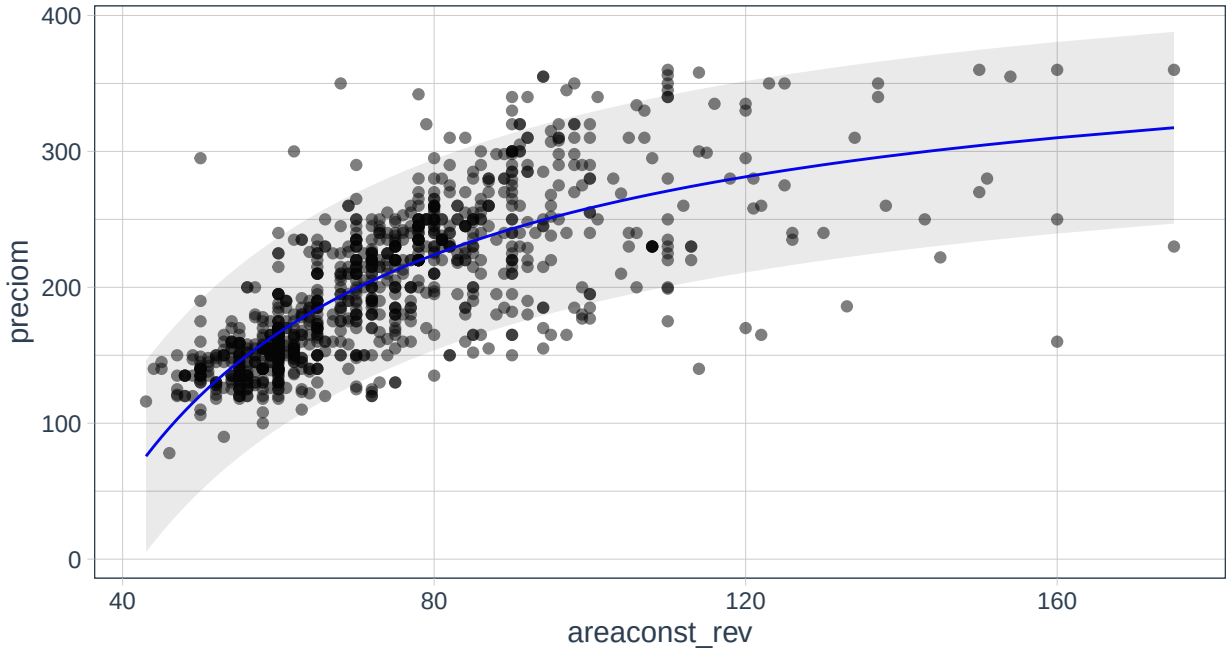
```
# Seleccionando datos y removiendo outliers
datos_lin_box <- datos_vivienda_mod |>
    select(preciom, areaconst_mod) |>
    remove_outliers(preciom) |>
    remove_outliers(areaconst_mod)

# Creando modelo
lm_lin_box <- lm(preciom ~ areaconst_mod, datos_lin_box)

# Tabla resumen modelo
tlb1 <- glance(lm_lin_box) |>
    mutate_all(~ round(.,4)) |>
    select(-10, -2) |>
    kable()

# Tabla resumen coeficientes
tlb2 <-tidy(lm_lin_box,
     conf.int = TRUE) |>
  mutate(across(where(is.numeric), ~ round(.,4))) |>
  kable()

# Agregando al data frame datos del modelo
datos_lin_box_aug <- augment(lm_lin_box) |>
  bind_cols(predict(lm_lin_box, datos_lin_box, interval = "prediction") |>
              as_tibble() |>
              select(-fit)) |>
    mutate(areaconst_rev = predict(norm_areaconst, areaconst_mod, inverse = TRUE))



# Grafica modelo
plb1 <- gg_lm_plot(datos_lin_box_aug, areaconst_rev, preciom, .fitted, lwr, upr)
```

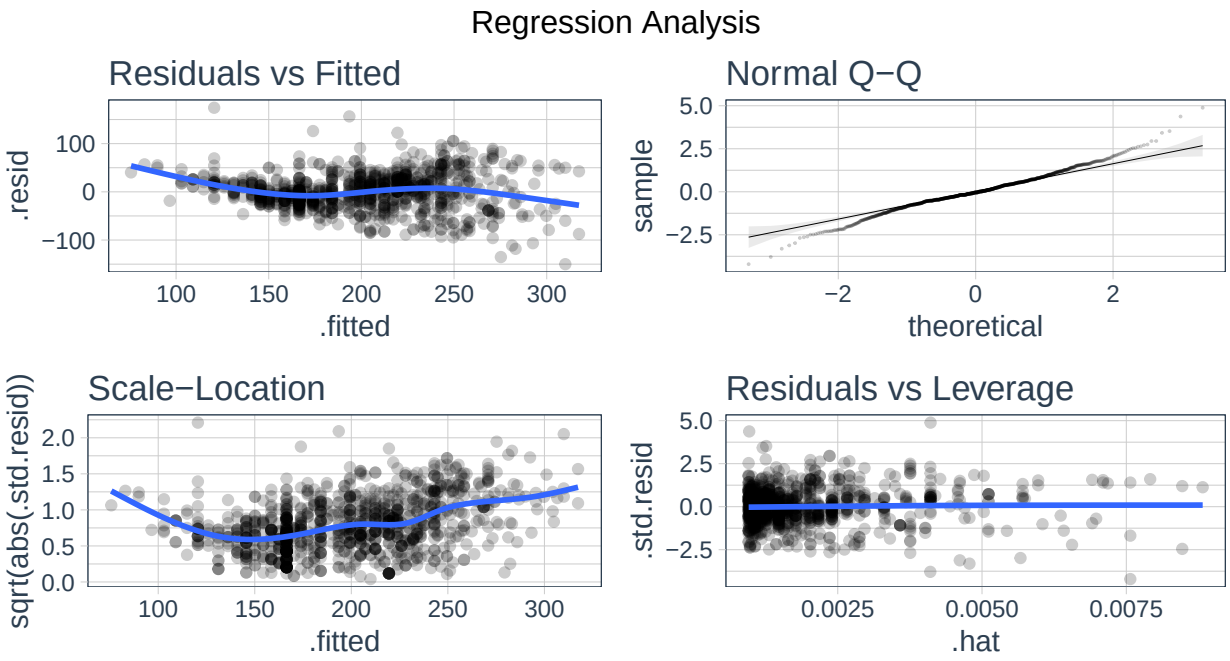
Tablas y Graficas lin_box

```
tlb1; tlb2; plb1; reg_analysis(
  datos_lin_box_aug)
```

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)	199.4434	1.1068	180.2055	0	197.2717	201.6152
areaconst_mod	44.4797	1.1551	38.5082	0	42.2132	46.7462





Interpretación modelo lin_box

Debido a la transformación de los datos, los coeficientes del modelo pierden un poco de interpretabilidad ya que el área se encuentra en otra escala. No obstante, se puede notar que tanto el intercepto como el coeficiente del área tienen un valor p que tiende a cero, lo cual indica que la relación entre las variables se mantiene después de la transformación.

Además, los intervalos de confianza del coeficiente del área sugieren una relación positiva con el precio. En cuanto al coeficiente de determinación R^2, este mejora ligeramente respecto al modelo sin transformaciones y su valor p sigue siendo cercano a 0.

La gráfica del modelo muestra una tendencia logarítmica, lo que sugiere que la relación del área sobre el precio es alta al principio, pero a medida que el área aumenta, pierde efecto sobre el precio.

Por último, en el análisis gráfico de los residuales, se puede observar que el efecto de los valores extremos que afectaban a la regresión ha desaparecido. Sin embargo, se sigue notando una tendencia en los residuales y su normalidad empeora especialmente en los valores altos.

box_lin

Creación Objetos R box_lin

```
# Seleccionando datos y removiendo outliers
datos_box_lin <- datos_vivienda_mod |>
    select(preciom_mod, areaconst) |>
    remove_outliers(preciom_mod) |>
    remove_outliers(areaconst)

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

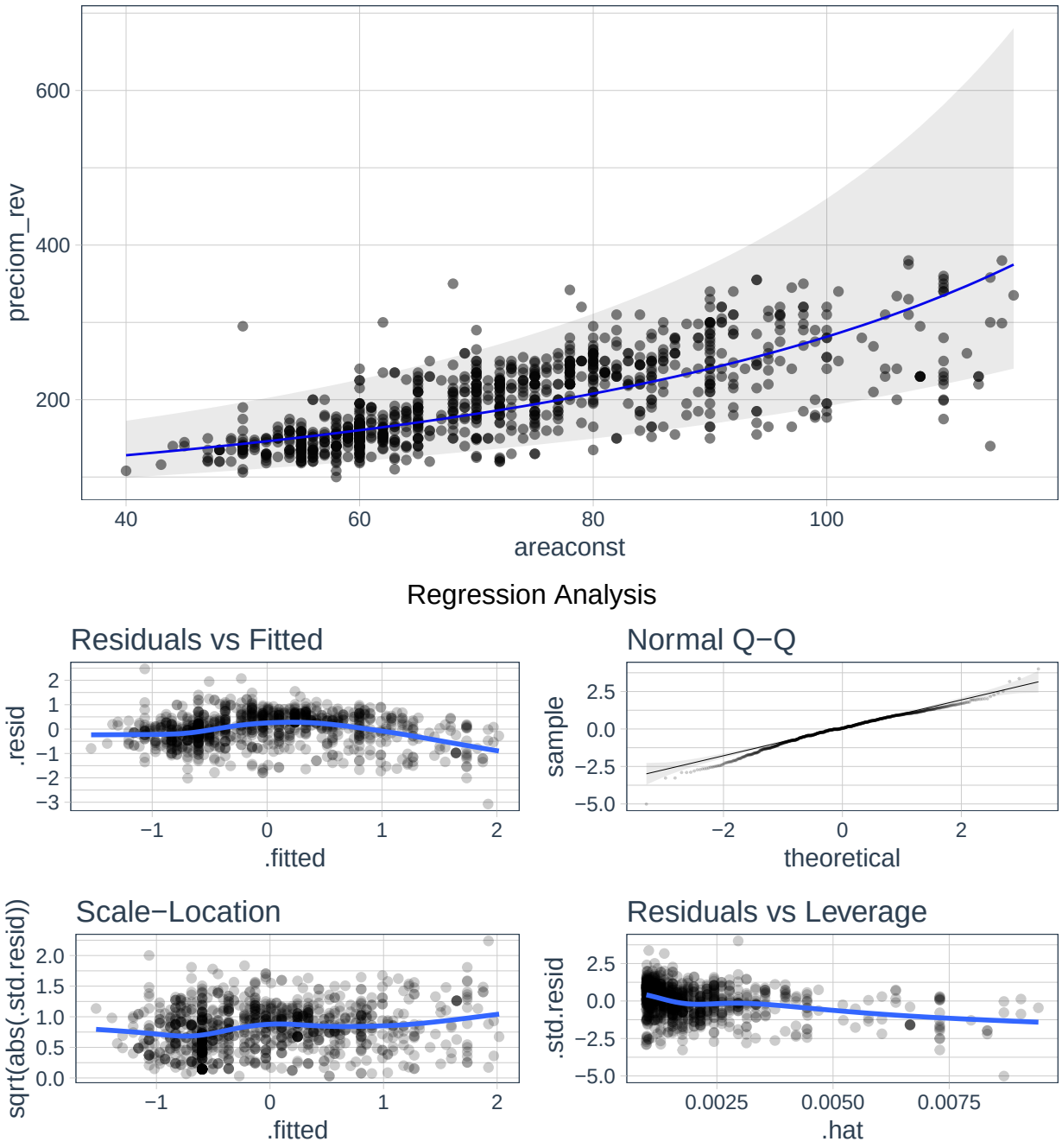
```
# Tabla resumen modelo
tbl1 <- glance(lm_box_lin) |>
    mutate_all(~ round(.,4)) |>
    select(-10, -2) |>
    kable()

# Tabla resumen coeficientes
tbl2 <-  tidy(lm_box_lin,
     conf.int = TRUE) |>
  mutate(across(where(is.numeric), ~ round(.,4))) |>
  kable()

# Agregando al data frame datos del modelo
datos_box_lin_aug <- augment(lm_box_lin) |>
   bind_cols(predict(lm_box_lin, datos_box_lin, interval = "prediction") |>
               as_tibble() |>
               select(-fit)) |>
    mutate(preciom_rev = predict(norm_preciom, preciom_mod, inverse = TRUE),
           .fitted_rev = predict(norm_preciom, .fitted, inverse = TRUE),
         lwr_rev = predict(norm_preciom, lwr, inverse = TRUE),
         upr_rev = predict(norm_preciom, upr, inverse = TRUE))

# Grafica modelo
pbl1 <- gg_lm_plot(datos_box_lin_aug, areaconst, preciom_rev, .fitted_rev, lwr_rev, upr_rev)
```

Tablas y Graficas box_lin

```
tbl1; tbl2; pbl1; reg_analysis(
  datos_box_lin_aug)
```

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



Interpretación modelo box_lin

Debido a la transformación de los datos, los coeficientes del modelo pierden parte de su interpretabilidad, ya que el precio se encuentra en otra escala. Sin embargo, es posible observar que tanto el intercepto como el coeficiente del área tienen un valor p cercano a cero, lo que indica que la relación entre las variables se mantiene después de la transformación.

Además, los intervalos de confianza del coeficiente del área sugieren una relación positiva con el precio. En cuanto al coeficiente de determinación R^2, este empeora con respecto al modelo sin transformaciones y su valor p sigue siendo cercano a 0.

La gráfica del modelo muestra una tendencia exponencial, lo que sugiere que la relación del área sobre el precio es baja al principio, pero aumenta a medida que el área aumenta.

Por último, en el análisis gráfico de los residuos, se puede observar que en comparación con el modelo sin transformaciones, la normalidad de los residuos mejora en los valores altos, pero siguen presentándose tendencias y valores extremos que afectan a la regresión.

box_box

Creación Objetos R box_box

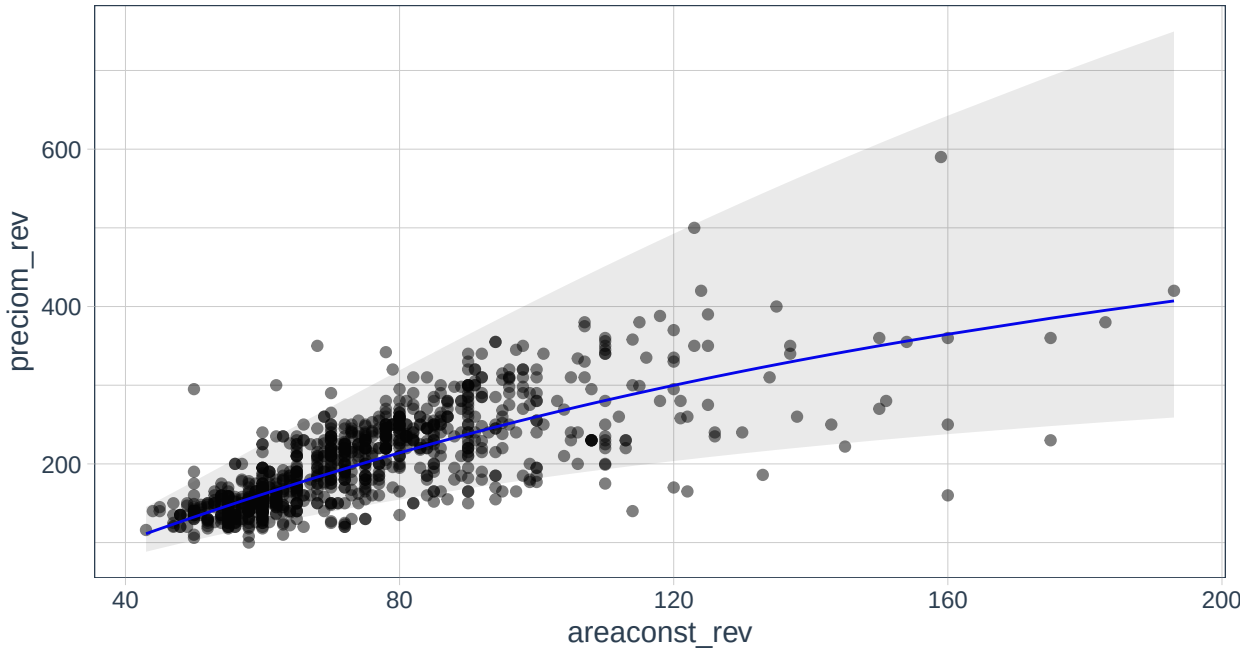
```
# Seleccionando datos y removiendo outliers
datos_box_box <- datos_vivienda_mod |>
    select(preciom_mod, areaconst_mod) |>
    remove_outliers(preciom_mod) |>
    remove_outliers(areaconst_mod)

# Creando modelo
lm_box_box <- lm(preciom_mod ~ areaconst_mod, datos_box_box)

# Tabla resumen modelo
tbb1 <- glance(lm_box_box) |>
    mutate_all(~ round(.,4)) |>
    select(-10, -2) |>
    kable()

# Tabla resumen coeficientes
tbb2 <- tidy(lm_box_box,
     conf.int = TRUE) |>
  mutate(across(where(is.numeric), ~ round(.,4))) |>
  kable()

# Agregando al data frame datos del modelo
datos_box_box_aug <- augment(lm_box_box) |>
  bind_cols(predict(lm_box_box, datos_box_box, interval = "prediction") |>
              as_tibble() |>
              select(-fit)) |>
  mutate(preciom_rev = predict(norm_preciom, preciom_mod, inverse = TRUE),
         .fitted_rev = predict(norm_preciom, .fitted, inverse = TRUE),
         areaconst_rev = predict(norm_areaconst, areaconst_mod, inverse = TRUE),
         lwr_rev = predict(norm_preciom, lwr, inverse = TRUE),
         upr_rev = predict(norm_preciom, upr, inverse = TRUE))

# Grafica modelo
pbb1 <- gg_lm_plot(datos_box_box_aug, areaconst_rev, preciom_rev, .fitted_rev, lwr_rev, upr_rev)
```

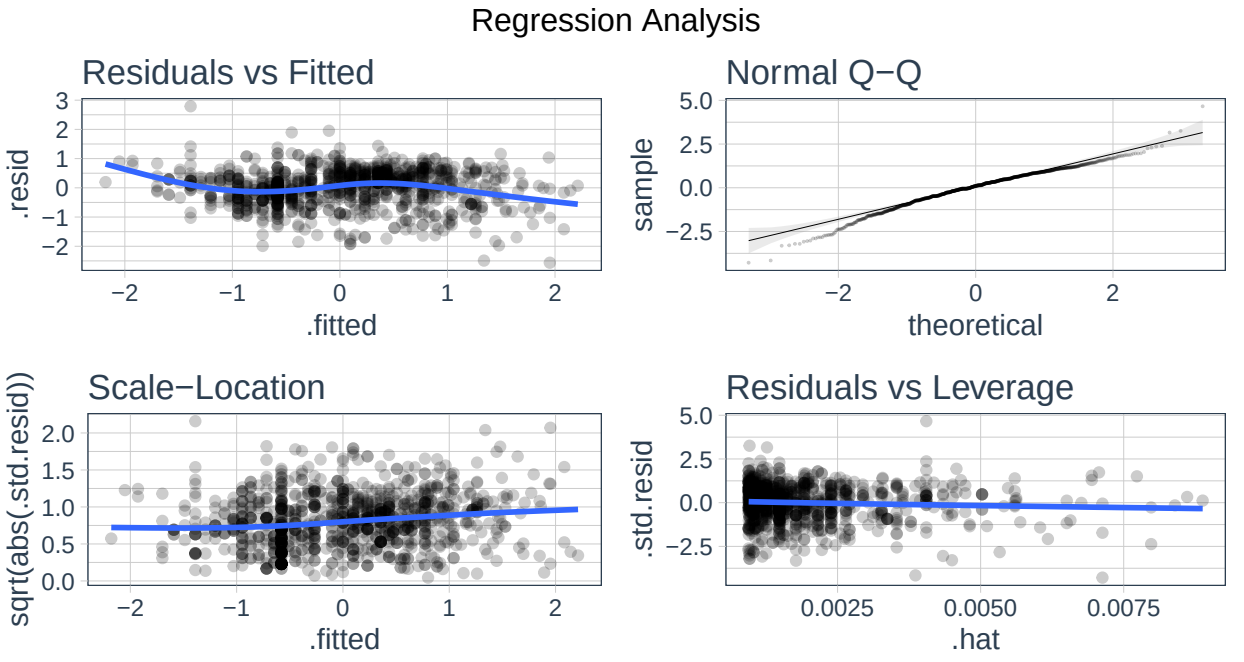
Tablas y Graficas box_box

```
tbb1; tbb2; pbb1; reg_analysis(
  datos_box_box_aug)
```

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





Interpretación modelo box_box

Debido a la transformación de los datos, los coeficientes del modelo pierden parte de su interpretabilidad, ya que el precio y el área se encuentran en otra escala. Al realizar una transformación adicional, el intercepto deja de ser significativo y se aproxima a cero. En cuanto al coeficiente del área, su valor p tiende a cero, lo que indica una fuerte relación con el precio.

Además, los intervalos de confianza del coeficiente del área sugieren una relación positiva con el precio. En cuanto al coeficiente de determinación R^2, este es el mejor de los cuatro modelos, con una explicación del 62% de la variabilidad del precio. Esto convierte al modelo en un buen candidato para inferencia.

La gráfica del modelo muestra una tendencia logarítmica, lo que sugiere que la relación del área sobre el precio es alta al principio, pero disminuye a medida que el área aumenta.

Por último, en el análisis gráfico de los residuos, se puede observar que todavía existen tendencias en los residuos, aunque son menos marcadas en comparación con los otros modelos. En cuanto a la normalidad de los residuos, esta solo es deficiente en los valores bajos. Además, se puede observar que el efecto de los valores extremos que afectaban a la regresión ha desaparecido.

Predicción sobre los modelos.

Para hacernos una idea del desempeño de estos cuatro modelos realizaremos una predicción para cada uno sobre una vivienda de $110m^2$

```
# Datos para modelos transfdormados y no transformados

data_lin <- tibble(areaconst = 110)
data_box <- predict(norm_areaconst, data_lin) |> set_names("areaconst_mod")

#Predicciones modelos
pred_lin_lin <- predict(lm_lin_lin, data_lin, interval = "prediction") |>
  as_tibble()
pred_lin_box <- predict(lm_lin_box, data_box, interval = "prediction") |>
  as_tibble()
pred_box_lin <- predict(lm_box_lin, data_lin, interval = "prediction") |>
  as_tibble() |>
  mutate(across(everything(), ~ predict(norm_preciom, ., inverse = TRUE)))
pred_box_box <- predict(lm_box_box, data_box, interval = "prediction") |>
  as_tibble() |>
  mutate(across(everything(), ~ predict(norm_preciom, ., inverse = TRUE)))

# Data frame con los relustados de los modelos
bind_rows(list(lin_lin = pred_lin_lin,
               lin_box = pred_lin_box,
               box_lin = pred_box_lin,
               box_box = pred_box_box),
          .id = 'Modelo') |>
  kable()
```

Modelo	fit	lwr	upr
lin_lin	299.4624	230.3206	368.6043
lin_box	270.8850	200.5255	341.2445
box_lin	334.7524	219.9544	581.5748

Modelo	fit	lwr	upr
box_box	280.5805	193.0348	451.3031

Los resultados indican que la elección del modelo tiene un gran impacto en la predicción del precio de un apartamento con un área determinada. Sin embargo, independientemente del modelo elegido, un apartamento de $110m^2$ con un precio de 200 millones de pesos sería una oferta llamativa en los datos del modelo Apartamentos de la Zona Sur.

Modelo escojido

En base a los resultados obtenidos, se concluye que el modelo seleccionado es el box_box. Este modelo presenta un coeficiente de determinación R^2 del 62%, lo que significa que puede explicar el 62% de la variabilidad en el precio de los apartamentos. Además, la evaluación gráfica de los residuos muestra que este modelo presenta una distribución más uniforme y se acerca a la normalidad en comparación con los otros modelos. Por lo tanto, se puede concluir que el modelo box_box es el más adecuado para analizar la relación entre el área y el precio de los apartamentos en este conjunto de datos.

Construcción modelo para inferencia.

El modelo elegido, box_box, fue entrenado utilizando el conjunto completo de datos, lo que podría resultar en un sobreajuste y limitar su capacidad para hacer inferencias en nuevos datos. Por lo tanto, nuestro próximo paso será construir un modelo que pueda ser utilizado para inferir sobre nuevos datos. Para lograr esto, dividiremos nuestro conjunto de datos en un conjunto de entrenamiento y otro de prueba, y utilizaremos una validación cruzada con k = 10 pliegues en el conjunto de entrenamiento.

Utilizando el paquete `tidymodels`, modelaremos estos datos con tres motores diferentes de regresión lineal, lm, gls y stan, y seleccionaremos el mejor modelo basado en su coeficiente de determinación (R^2) en la validación cruzada.

División de los datos

Se dividirán los datos en un conjunto de entrenamiento y otro de prueba utilizando una proporción del 70% y 30%, respectivamente. Además, se aplicará una técnica de validación cruzada con 10 pliegues sobre el conjunto de datos de prueba. De esta manera, se podrá evaluar el rendimiento de los modelos de regresión lineal en datos no vistos y evitar sobreajuste.

```
# Fijar la semilla para reproducibilidad
set.seed(1234)

# Split 70/30
box_box_split <- datos_box_box_aug |>
    select(preciom_mod, areaconst_mod) |>
    initial_split(prop = 0.7,
                  strata = preciom_mod)

#Conjunto de entrenamiento, prueba y k=10 prlieges
box_box_train <- training(box_box_split)
box_box_test <- testing(box_box_split)
box_box_folds <- vfold_cv(box_box_train, strata = preciom_mod, v = 10)
```

Creación de modelos

Usando la función `recipe`, especificaremos la fórmula `preciom_mod ~ areaconst_mod` para definir nuestra receta de transformación de datos. Dado que la transformación de datos ya se aplicó en la fase inicial, no agregaremos pasos adicionales. Posteriormente, procederemos a definir los modelos que serán utilizados.

```
# Receta y definición del la formula
box_box_rec <- recipe(preciom_mod ~ areaconst_mod, data = box_box_train)

# Iniciando los modelos a usar
models_box_box <- list(
    lm_reg = linear_reg(),
    gls_reg = linear_reg(engine = "gls"),
    stan_reg = linear_reg(engine = "stan")

)

#
workflow_set_box_box <- workflow_set(preproc = list(formula = box_box_rec)
                                     , models = models_box_box)
```

Entremamiento modelos

Se entrenarán los modelos usando validación cruzada con k=10 y se guardarán los resultados en `grid_results`.

```
# Fijar la semilla para reproducibilidad
set.seed(1234)

# Definiendo que se guarden los pasos del flujo
grid_ctrl <-
   control_grid(
      save_pred = TRUE,
      save_workflow = TRUE
   )

# Entrenando los modelos con  k=10 pliegues y guardando los resultados
grid_results <-
   workflow_set_box_box |>
   workflow_map(
      seed = 1234,
      resamples = box_box_folds,
      control = grid_ctrl
   )
```

Selección del mejor modelo

Primero miraremos los resultados de los modelos

```
# Ranking de medidas
grid_results |>
  rank_results() |>
```

```
  select(-.config, -n, - preprocessor, -model) |>
  kable()
```

wflow_id	.metric	mean	std_err	rank
formula_stan_reg	rmse	0.5810449	0.0158259	1
formula_stan_reg	rsq	0.6398633	0.0206219	1
formula_lm_reg	rmse	0.5810577	0.0158290	2
formula_lm_reg	rsq	0.6398633	0.0206219	2
formula_gls_reg	rmse	0.5810577	0.0158290	3
formula_gls_reg	rsq	0.6398633	0.0206219	3

Observamos que todos los modelos tienen el mismo coeficiente de determinación (R^2), por lo que para desempatar utilizaremos el error de raíz cuadrada media (RMSE por sus siglas en inglés). De esta manera, el modelo seleccionado será la regresión lineal `stan`.

```
# Seleccionado mejor modelo
best_results <-
   grid_results |>
   extract_workflow_set_result("formula_stan_reg") |>
   select_best(metric = "rmse")

# Reentrenando sobre el conjunto completo de datos
box_box_test_results <- grid_results |>
  extract_workflow("formula_stan_reg") |>
  finalize_workflow(best_results) |>
  last_fit(split = box_box_split)
```

Coeficientes y R^2 modelo inferencia final

Este es el R^2, coeficientes del modelo e intervalos de confianza

```
collect_metrics(box_box_test_results) |>
  filter(.metric == "rsq") |>
  select(.metric, .estimate
         ) |>
  kable(); box_box_test_results |>
  extract_workflow() |>
  extract_model() |>
  posterior_interval() |>
  as.data.frame() |>
  rownames_to_column(var = "term") |>
  slice(-3) |>
  mutate(fit = box_box_test_results$.workflow[[1]][2]$fit$fit$fit$coefficients) |>
  select(term, fit, everything()) |>
  kable()
```

.metric	.estimate
rsq	0.5894217

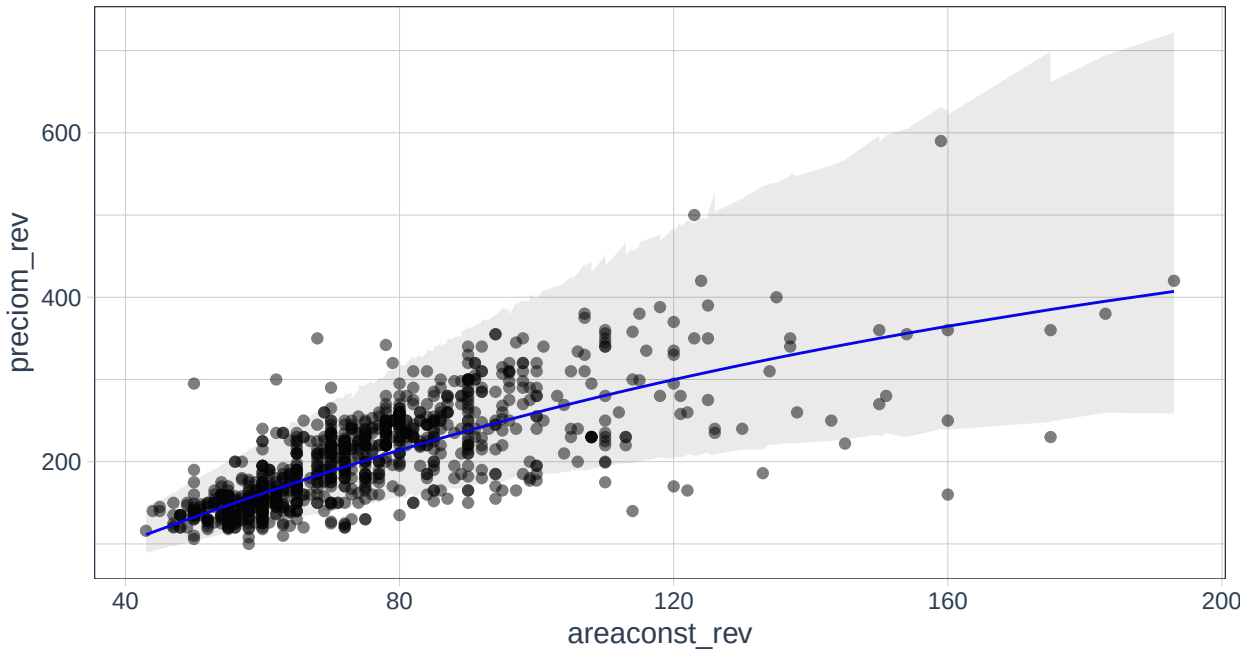
term	fit	5%	95%
(Intercept)	0.0065604	-0.0290646	0.0423074
areaconst_mod	0.7787920	0.7421385	0.8146302

Observamos que los coeficientes y sus intervalos de confianza en el modelo ajustado mediante validación cruzada son similares a los del modelo inicial box_box. A pesar de esto, el coeficiente de determinación (R^2) del modelo ajustado es menor. Sin embargo, esta reducción era esperada, ya que la validación cruzada implica una mayor generalización del modelo, lo que puede resultar en una disminución del R^2. La ventaja de este enfoque es que el modelo ajustado estará menos propenso al sobreajuste y, por lo tanto, será más adecuado para la predicción de nuevos datos.

Gráfica modelo de inferencia final.

A continuación la representación grafica del modelo evaluado sobre todos los datos de Apartamentos de la Zona Sur de estrato 4.

```
box_box_test_results_aug <- datos_box_box |>
  bind_cols(tibble(.fitted = box_box_test_results |>
                      extract_workflow() |>
                      extract_model() |>
                      predict(type = "response", newdata = datos_box_box)),
            as_tibble(box_box_test_results |>
                        extract_workflow() |>
                        extract_model() |>
                        predictive_interval (newdata = datos_box_box, prob = 0.95)) |>
                        set_names(c("lwr","upr"))) |>
  mutate(preciom_rev = predict(norm_preciom, preciom_mod, inverse = TRUE),
         .fitted_rev = predict(norm_preciom, .fitted, inverse = TRUE),
         areaconst_rev = predict(norm_areaconst, areaconst_mod, inverse = TRUE),
         lwr_rev = predict(norm_preciom, lwr, inverse = TRUE),
         upr_rev = predict(norm_preciom, upr, inverse = TRUE),
         )


ggplot(box_box_test_results_aug, aes(areaconst_rev, preciom_rev,)) +
  geom_point(alpha = 0.5) +
  geom_line(data = datos_box_box_aug, aes(areaconst_rev, .fitted_rev), color = "blue") +
  geom_ribbon(aes(ymin = lwr_rev, ymax = upr_rev ),alpha = 0.1)+
    theme_tq()
```



Predicción sobre modelo de inferencia final.

Para hacernos una idea del desempeño del modelo realizaremos una predicción para cada uno sobre una vivienda de $110m^2$

```
tibble(.fited = box_box_test_results |>
         extract_workflow() |>
         extract_model() |>
         predict(type = "response", newdata = data_box)) |>
  bind_cols(as_tibble(box_box_test_results |>
                        extract_workflow() |>
                        extract_model() |>
                        predictive_interval (newdata = data_box, prob = 0.95)) |>
              set_names(c("lwr","upr"))) |>
  mutate(across(everything(), ~ predict(norm_preciom, ., inverse = TRUE))) |>
  kable()
```

.fited	lwr	upr
280.095	195.4612	448.4461

Grabando modelo final

Para su futura implementación, se guardará el modelo en un archivo de documento .RDS. Esto permitirá una fácil recuperación del modelo y su uso en diferentes contextos. El archivo .RDS contendrá el objeto del modelo guardado como un archivo binario, lo que asegurará que todas las características del modelo, como los coeficientes y los parámetros, se mantengan intactos. Además, la extensión .RDS es ampliamente reconocida en el ecosistema de R como un formato de archivo para la persistencia de objetos, lo que garantiza que el modelo pueda ser utilizado por otros usuarios sin problemas.

Además, se almacenan los objetos `norm_areaconst` y `norm_preciom` como complemento del modelo, con el fin de poder transformar y revertir la transformación de datos para el modelado. De esta manera, se logra mostrar los datos en su escala original.

```
saveRDS(box_box_test_results |>
         extract_workflow() |>
         extract_model(),
        "modelo_lineal_simple.rds")

saveRDS(norm_areaconst,
        "area_box_cox.rds")

saveRDS(norm_preciom,
        "precio_box_cox.rds")
```