Modelos Predictivos

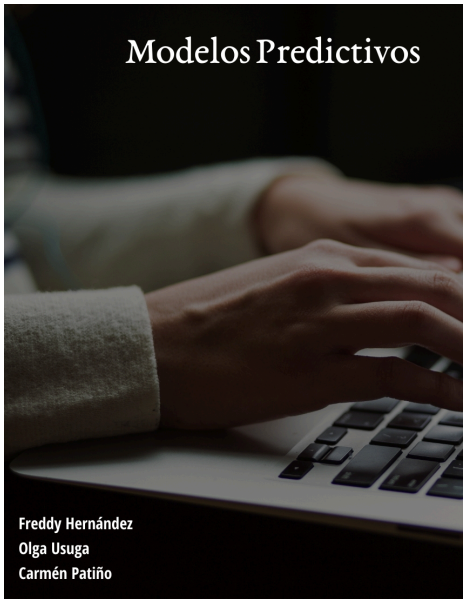
Freddy Hernández
Olga Usuga
Carmen Patiño

2019-11-25

Gracias a Dios por todo lo que me ha dado.

Índice general

Bienvenido



Este libro está destinado para estudiantes de ingeniería y estadística que deseen aprender sobre modelos de regresión y la forma de aplicarlos por medio del lenguaje de programación R.

Freddy Hernández[1]
Olga Usuga[2]
Carmen Patiño[3]

[1] https://fhernanb.github.io/

[2] http://scienti.colciencias.gov.co:8081/cvlac/visualizador/generarCurriculoCv.do?cod_rh=0001417591

[3] http://scienti.colciencias.gov.co:8081/cvlac/visualizador/generarCurriculoCv.do?cod_rh=0000756008

Estructura del libro

En el capítulo se presentan los árboles de regresión.

Software y convenciones

Para realizar este libro usamos los paquetes **knitr** (Xie, 2015) y **bookdown** (Xie, 2019) que permiten unir la ventajas de LaTeX y R en un mismo archivo.

En todo el libro se presentarán códigos que el lector puede copiar y pegar en su consola de R para obtener los mismos resultados aquí del libro. Los códigos se destacan en una caja de color similar a la mostrada a continuación.

```
4 + 6
a <- c(1, 5, 6)
5 * a
1:10
```

Los resultados o salidas obtenidos de cualquier código se destacan con dos símbolos de númeral (**##**) al inicio de cada línea o renglón, esto quiere decir que todo lo que inicie con **##** son resultados obtenidos y **NO** los debe copiar. Abajo se muestran los resultados obtenidos luego de correr el código anterior.

```
## [1] 10
```

```
## [1]  5 25 30
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

Bloques informativos

En varias partes del libro usaremos bloques informativos para resaltar algún aspecto importante. Abajo se encuentra un ejemplo de los bloques y su significado.

Nota aclaratoria.

Sugerencia.

Advertencia.

Árboles de regresión

Los árboles de regresión/clasificación fueron propuestos par Leo Breiman[4] en el libro (Breiman et al., 1984) y son árboles de decisión que tienen como objetivo asignar un valor de $\hat{y}$ dependiendo de los valores de las covariables.

Los árboles se pueden clasificar en dos tipos que son:

1. Árboles de regresión en los cuales la variable respuesta y es cuantitativa.
2. Árboles de clasificación en los cuales la variable respuesta y es cualitativa.

El presente capítulo está destinado a árboles de regresión, los árboles de clasificación se explican en el capítulo .

Un árbol de regresión consiste en hacer preguntas de tipo ¿$x_k \leq c$? para cada una de las covariables, de esta forma el espacio de las covariables es divido en hiper-rectángulos y todas las observaciones que queden dentro de un hiper-rectángulo tendrán el mismo valor estimado $\hat{y}$.

En la siguiente figura se ilustra el árbol en el lado izquierdo y la partición del espacio en el lado derecho. La partición del espacio se hace de manera repetitiva para encontrar las variables y los valores de corte c de tal manera que se minimice la función de costos $\sum_{i=1}^{i=n}(y_i - \hat{y}_i)^2$.

Los pasos para realizar la partición del espacio son:

1. Dado un conjunto de covariables (características), encontrar la covariable que permita predecir mejor la variable respuesta.
2. Encontrar el punto de corte c sobre esa covariable que permita predecir mejor la variable respuesta.
3. Repetir los pasos anteriores hasta que se alcance el criterio de parada.

Algunas de las ventajas de los árboles de regresión son:

- Fácil de entender e intrepretar.
- Requiere poca preparación de los datos.
- Las covariables pueden ser cualitativas o cuantitativas.
- No exige supuestos distribucionales.

[4] `https://en.wikipedia.org/wiki/Leo_Breiman`

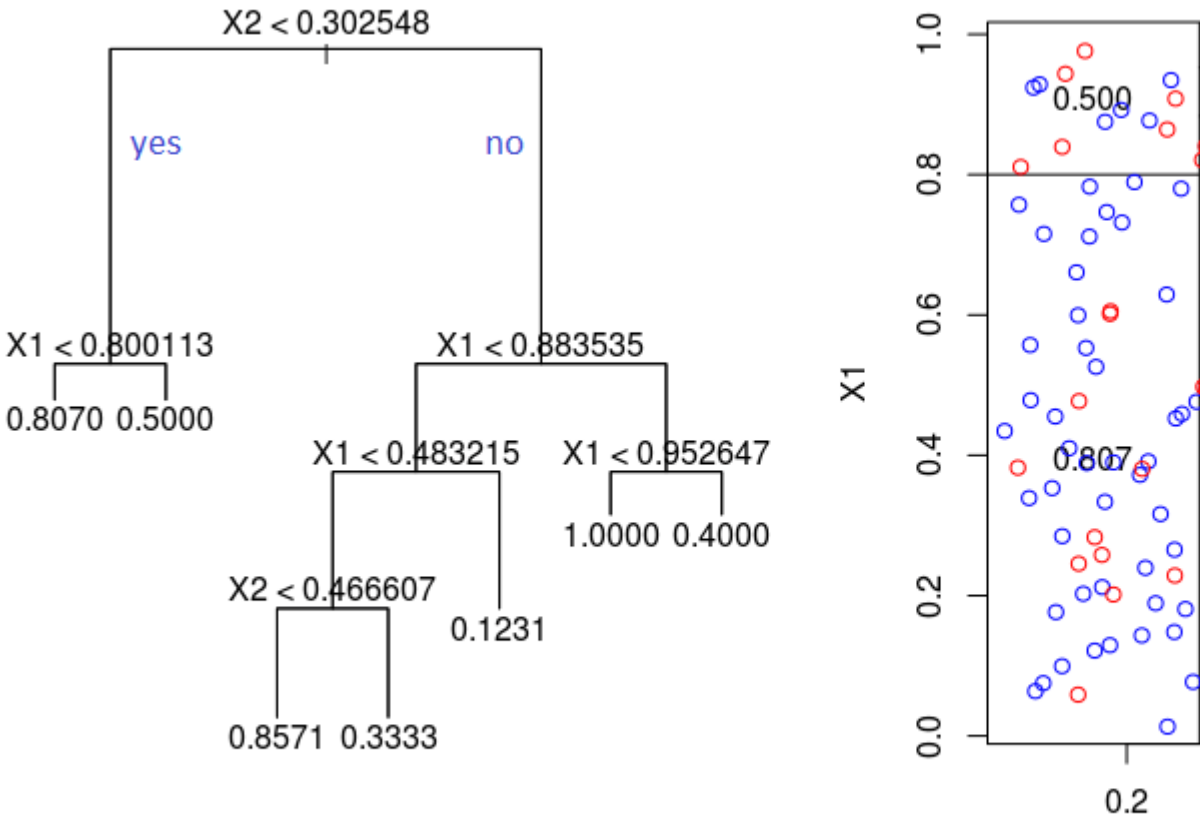


Figura 1: Ilustración de la técnica Árboles de Regresión. A la izquierda el árbol y a la derecha la partición del espacio.

Para explicaciones más detalladas sobre las técnicas basadas en árboles recomendamos consultar el capítulo 8 de (James et al., 2013). Se recomienda también ver este video[5] con una explicación sencilla sobre árboles.

[5] `https://www.youtube.com/watch?v=7VeUPuFGJHk`

Paquetes

En esta sección se mencionan algunos de los paquetes más comunes para implementar árboles de regresión.

El paquete **rpart** (Therneau and Atkinson, 2019) es uno de los paquetes que se pueden usar para crear árboles de regresión. La función para crear un árbol de regresión es `rpart`, a continuación la estructura de la función.

```
rpart(formula, data, weights, subset, na.action = na.rpart, method,
      model = FALSE, x = FALSE, y = TRUE, parms, control, cost, ...)
```

Otro paquete útil para árboles de regresión es **tree** (Ripley, 2019). La función para crear un árbol de regresión es `tree`, a continuación la estructura de la función.

```
tree(formula, data, weights, subset,
     na.action = na.pass, control = tree.control(nobs, ...),
     method = "recursive.partition",
     split = c("deviance", "gini"),
     model = FALSE, x = FALSE, y = TRUE, wts = TRUE, ...)
```

Se recomienda al lector que consulte la ayuda de las funciones anteriores para que pueda entender las posibilidades que se tienen con cada una de ellas.

Ejemplo con el paquete rpart

En este ejemplo se busca encontrar un modelo de regresion lineal que explique la variable respuesta y en función de las covariables x_1 a x_{11}, los datos provienen del ejercicio 9.5 del libro de Montgomery, Peck and Vining (2003)[6]. El paquete **MPV** (Braun, 2019) contiene todos los datos que acompañan al libro.

A continuación se muestra el encabezado de la base de datos y la definición de las variables.

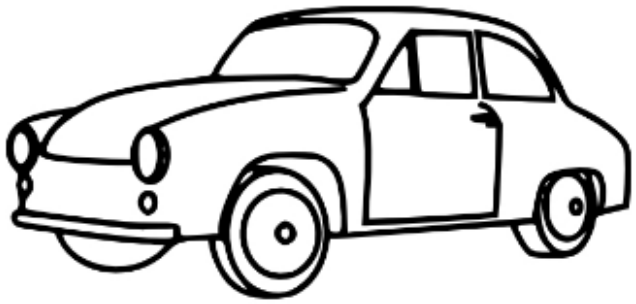
Nota: Type of transmission (1=automatic, 0=manual).

Antes de iniciar es necesario revisar si hay `NA's` y eliminarlos.

[6] `https://www.amazon.com/Introduccion-analisis-regresion-lineal-Spanish/dp/9702403278`

TABLE B.3 Gasoline Mileage Performance for 32 Antomo

Automobile	y	x_1	x_2	x_3	x_4	x_5	x_6
Apollo	18.90	350	165	260	8.0:1	2.56:1	4
Omega	17.00	350	170	275	8.5:1	2.56:1	4
Nova	20.00	250	105	185	8.25:1	2.73:1	1
Monarch	18.25	351	143	255	8.0:1	3.00:1	2
Duster	20.07	225	95	170	8.4:1	2.76:1	1



Definición de las variables

y: Miles/gallon
x_1: Displacement (cubic in.)
x_2: Horsepower (ft-lb)
x_3: Torqne (ft-lb)
x_4: Compression ratio
x_5: Rear axle ratio
Source: *Motor Trend*, 1975.

x_6: Carburetor (l
x_7: No. of transm
x_8: Overall lengt
x_9: Width (in.)
x_{10}: Weight (lb)
x_{11}: Type of trans

Figura 2: Ilustración de la base de datos.

```
library(MPV)  # Aqui estan los datos
table.b3[22:26, ] # Can you see the missing values?
```

```
##        y    x1  x2  x3  x4   x5 x6 x7    x8   x9  x10 x11
## 22 21.47 360.0 180 290 8.4 2.45  2  3 214.2 76.3 4250   1
## 23 16.59 400.0 185  NA 7.6 3.08  4  3 196.0 73.0 3850   1
## 24 31.90  96.9  75  83 9.0 4.30  2  5 165.2 61.8 2275   0
## 25 29.40 140.0  86  NA 8.0 2.92  2  4 176.4 65.4 2150   0
## 26 13.27 460.0 223 366 8.0 3.00  4  3 228.0 79.8 5430   1
```

```
datos <- table.b3[-c(23, 25), ]
```

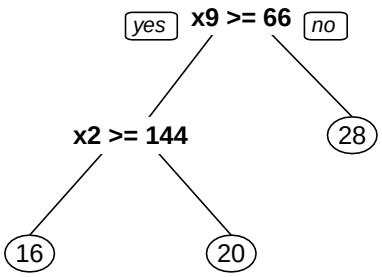
El objeto `datos` tiene la base de datos sin las líneas con `NA`, lo mismo se hubiese podido realizar usando la función `na.omit`. La base de datos tiene 30 filas y 12 columnas.

```
library(rpart)
library(rpart.plot)
```

```
mod1 <- rpart(y ~ ., data=datos)
```

Dibjuemos el árbol con `prp` que es una función del paquete **rpart.plot** (Milborrow, 2019).

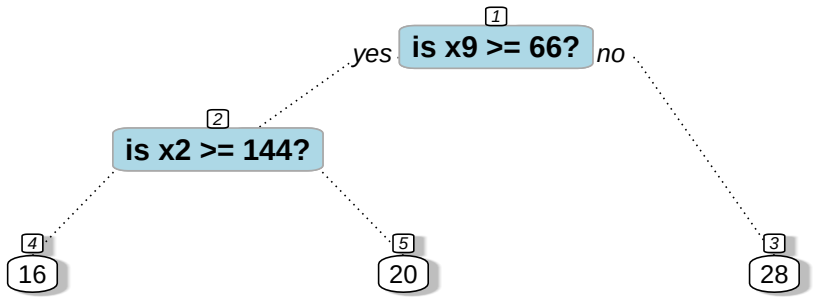
```
prp(mod1)
```



Construyamos nuevamente el árbol pero explorando todas las opciones de la función `prp`.

```
prp(mod1, main="",
    nn = TRUE,              # display the node numbers
    fallen.leaves = TRUE,   # put the leaves on the bottom of the page
    shadow.col = "gray",    # shadows under the leaves
    branch.lty = 3,         # draw branches using dotted lines
    branch = .5,            # change angle of branch lines
    faclen = 0,             # faclen = 0 to print full factor names
    trace = 1,              # print the auto calculated cex, xlim, ylim
    split.cex = 1.2,        # make the split text larger than the node text
    split.prefix = "is ",   # put "is " before split text
    split.suffix = "?",     # put "?" after split text
    split.box.col = "lightblue",    # lightgray split boxes (default is white)
    split.border.col = "darkgray", # darkgray border on split boxes
    split.round = 0.5)              # round the split box corners a tad
```

```
## cex 1   xlim c(0, 1)   ylim c(0, 1)
```



Usando la información del árbol anterior es posible predecir el valor de y. Por ejemplo:

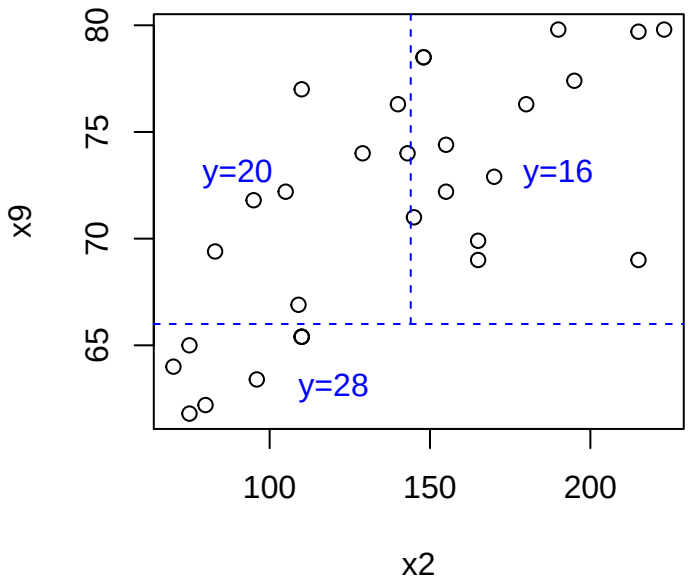
1. Si una nueva observación tiene $x_9 = 70$ y $x_2 = 100$, entonces $\hat{y} = 20$.
2. Si otra observación tiene $x_9 = 60$ y $x_2 = 150$, entonces $\hat{y} = 28$.

Como en el árbol anterior solo aparecen las variables x_2 y x_9 se recomienda volver a construir el árbol sólo con ellas.

```
mod1 <- rpart(y ~ x2 + x9, data=datos)
```

Este árbol por tener solo dos covariables se puede representar de la siguiente forma:

```
with(datos, plot(x=x2, y=x9))
abline(h=66, lty='dashed', col='blue')
segments(x0=144, y0=66, x1=144, y1=82, lty='dashed', col='blue')
text(x=120, y=63, labels='y=28', col=4)
text(x=90, y=73, labels='y=20', col=4)
text(x=190, y=73, labels='y=16', col=4)
```



Para predecir los valores de y se puede usar la función `predict`. A continuación el código para predecir la respuesta en los dos casos anteriores.

```
nuevos_datos <- data.frame(x2=c(100, 150), x9=c(70, 60))
predict(object=mod1, newdata=nuevos_datos)
```

```
##        1        2
```

```
## 19.66875 28.06625
```

En este ejemplo los datos originales se usaron como conjunto de entrenamiento y prueba debido a que solo se cuentan con 30 observaciones.

Entre más cerca estén las $\hat{y}$ de los y observados se puede decir que el modelo es mejor. A continuación la correlación entre $\hat{y}$ y y.

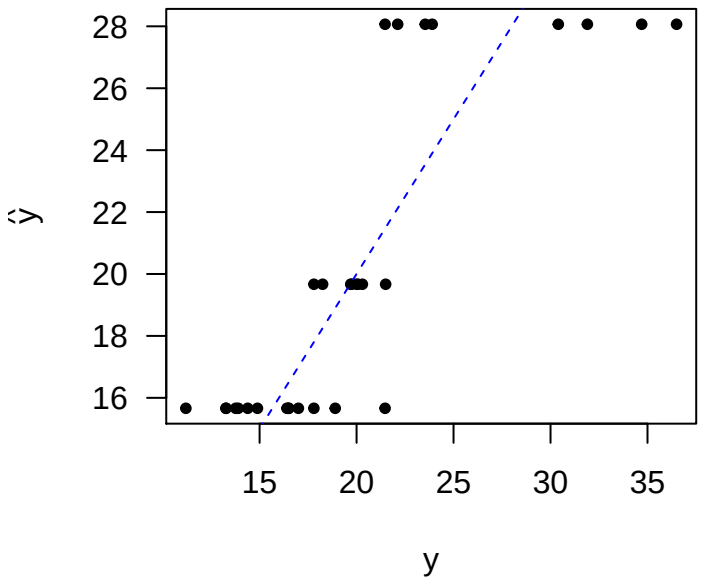
```
y_hat <- predict(object=mod1, newdata=datos)
cor(y_hat, datos$y)
```

```
## [1] 0.8300304
```

¿Qué opina de este valor?

A continuación un diagrama de dispersión entre $\hat{y}$ y y.

```
plot(x=datos$y, y=y_hat, pch=20, las=1, xlab='y', ylab=expression(hat(y)))
abline(a=0, b=1, lty="dashed", col="blue")
```



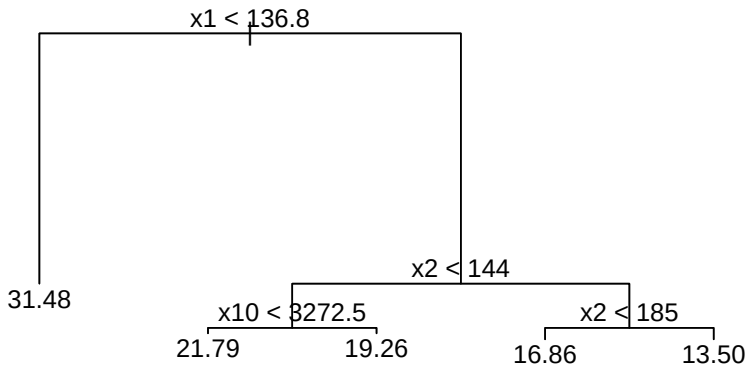
Ejemplo con el paquete tree

Aquí vamos a repetir el ejemplo anterior con otro paquete.

```
library(tree)
mod2 <- tree(y ~ ., data=datos)
```

Para dibujar el árbol se puede usar las siguientes instrucciones.

```
plot(mod2)
text(mod2, pretty=0)
```



Entre más cerca estén las $\hat{y}$ de los y observados se puede decir que el modelo es mejor. A continuación la correlación entre $\hat{y}$ y y.

```
y_hat <- predict(object=mod2, newdata=datos)
cor(y_hat, datos$y)
```

```
## [1] 0.9265051
```

Árboles de clasificación

Los árboles de regresión/clasificación fueron propuestos par Leo Breiman[7] en el libro (Breiman et al., 1984) y son árboles de decisión que tienen como objetivo asignar un valor de $\hat{y}$ dependiendo de los valores de las covariables.

Los árboles se pueden clasificar en dos tipos que son:

1. Árboles de regresión en los cuales la variable respuesta y es cuantitativa.
2. Árboles de clasificación en los cuales la variable respuesta y es cualitativa.

El presente capítulo está destinado a árboles de clasificación, los árboles de regresión se explican en el capítulo .

[7] `https://en.wikipedia.org/wiki/Leo_Breiman`

Support Vector Machines

Cortes and Vapnik (1995) propusieron las máquinas de soporte vectorial para el problema de clasificación.

Suponga que deseamos tenemos dos grupos de objetos, unos de color rojo y otros de color amarillo como se muestran en la siguiente figura.

El objetivo es dibujar una línea recta que separe los dos grupos, sin embargo, muchas líneas se podrían dibujar, a continuación se muestran tres posibles líneas con las cuales se consigue el objetivo.

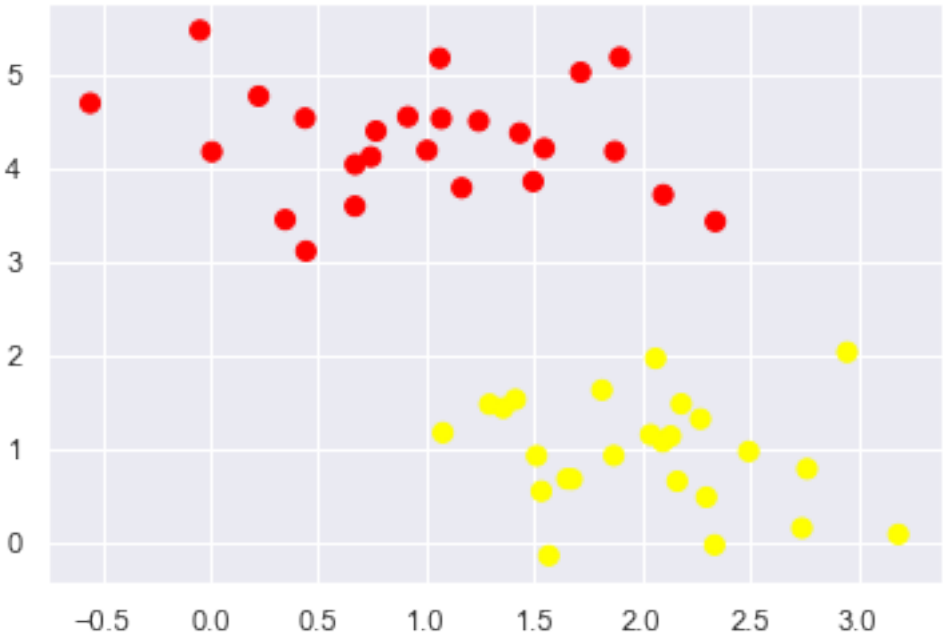


Figura 3: Ilustración de la técnica Árboles de Regresión. A la izquierda el árbol y a la derecha la partición del espacio.

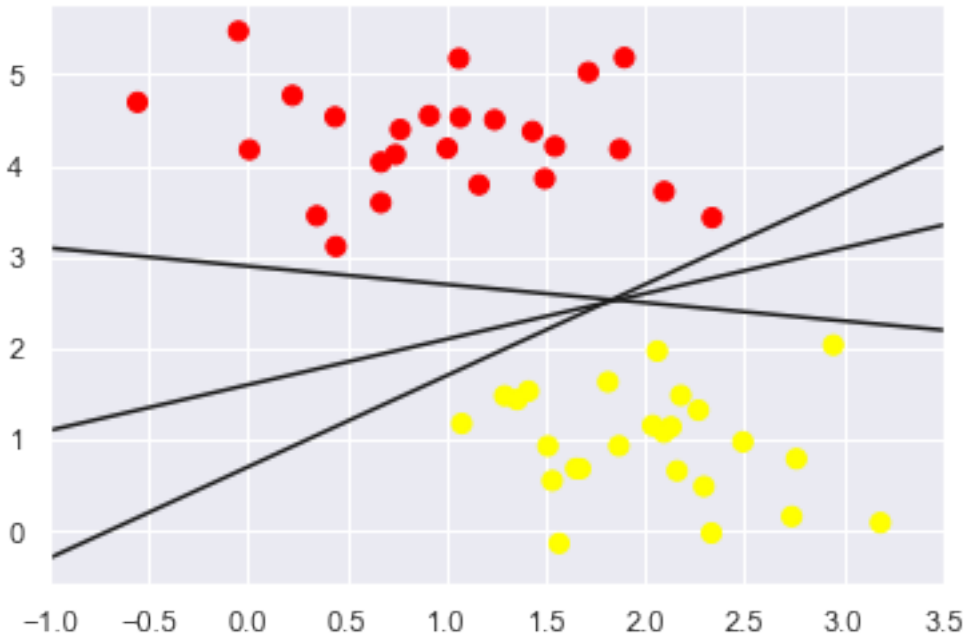


Figura 4: espacio2.

Imaginemos que cada línea es como una carretera que se puede ampliar a ambos lados hasta que toque el punto más cercano, ya se amarillo o rojo. Al hacer esto vamos a obtener las tres carreteras que se muestran a continuación.

De todas las carreteras nos interesa aquella que tenga el mayor ancho o margen, con esa carretera es que se pueden clasificar nuevas observaciones en el grupo rojo o grupo amarillo. A continuación se muestra la figura sólo con la línea de separación que tiene el mayor ancho.

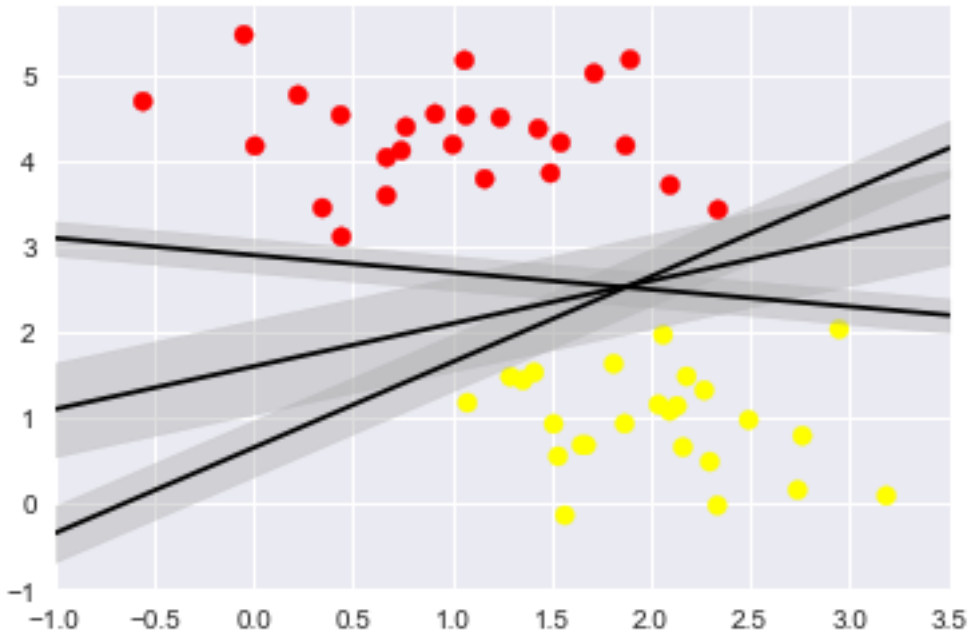


Figura 5: espacio3.

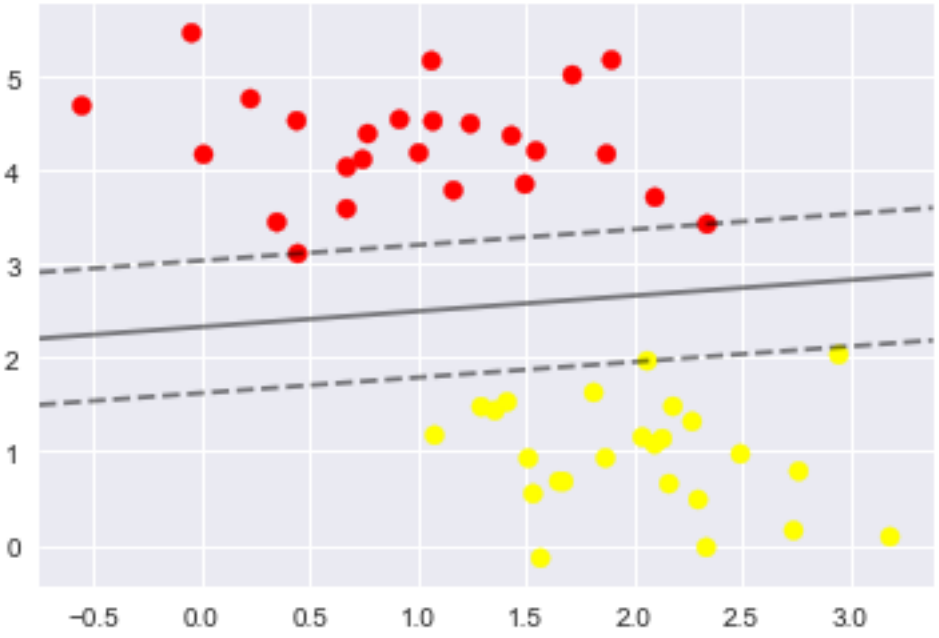


Figura 6: espacio4.

Regresión lineal versus árboles de regresión

En este capítulo se muestra una comparación entre modelos de regresión y árboles de regresion.

Regresión lineal

El modelo de regresión lineal simple es uno de los más populares en modelación. Este modelo se puede resumir a continuación.

$$y_i \sim N(\mu_i, \sigma^2), \tag{1}$$
$$\mu_i = \beta_0 + \beta_1 x_i, \tag{2}$$
$$\sigma^2 = \text{constante} \tag{3}$$

Arboles de regresión

Una explicación de los árboles de regresión puede ser consultada en el capítulo .

Las librerías en R para implementar árboles de regresión son:

```
library(rpart)
library(rpart.plot)
```

Ejemplo

Como ilustración vamos a usar los datos del ejemplo 2.1 del libro de Montgomery, Peck and Vining (2003)[8]. En el ejemplo 2.1 los autores ajustaron un modelo de regresión lineal simple para explicar la Resistencia de una soldadura en función de la Edad de la misma.

A continuación el código para cargar los datos y una muestra de las 6 primeras observaciones de la base de datos, en total tenemos 20 observaciones.

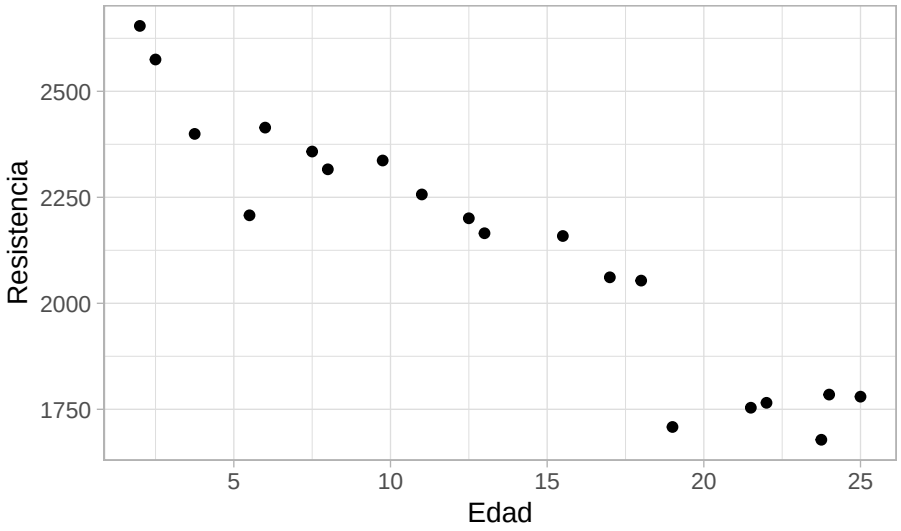
```
file <- "https://raw.githubusercontent.com/fhernanb/datos/master/propelente"
datos <- read.table(file=file, header=TRUE)
head(datos) # shows the first 6 rows
```

```
##   Resistencia  Edad
## 1     2158.70 15.50
## 2     1678.15 23.75
## 3     2316.00  8.00
## 4     2061.30 17.00
## 5     2207.50  5.50
## 6     1708.30 19.00
```

Para crear un diagrama de dispersión que nos muestre la relación entre las dos variables usamos las siguientes instrucciones.

```
library(ggplot2)
ggplot(datos, aes(x=Edad, y=Resistencia)) +
  geom_point() + theme_light()
```

[8] https://www.amazon.com/Introduccion-analisis-regresion-lineal-Spanish/dp/9702403278



De la figura anterior se ve claramente que a medida que aumenta la edad de la soldadura, la resistencia que ella ofrece disminuye. Adicionalmente, se observa que la relación entre las variables es lineal con una dispersión que parece constante.

¿Quién estima mejor? ¿un modelo de regresión lineal simple o un árbol?

```
rls <- lm(Resistencia ~ Edad, data=datos)
arb <- rpart(Resistencia ~ Edad, data=datos)
```

```
arb <- rpart(Resistencia ~ Edad, data=datos, method="anova")
```

¿Qué hay dentro de modelo de regresión lineal simple?

```
summary(rls)
```

```
##
## Call:
## lm(formula = Resistencia ~ Edad, data = datos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -215.98  -50.68   28.74   66.61  106.76
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 2627.822     44.184   59.48  < 2e-16 ***
## Edad         -37.154      2.889  -12.86 1.64e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 96.11 on 18 degrees of freedom
## Multiple R-squared:  0.9018, Adjusted R-squared:  0.8964
## F-statistic: 165.4 on 1 and 18 DF,  p-value: 1.643e-10
```

¿Qué hay dentro de modelo del arbol?

```
summary(arb)
```

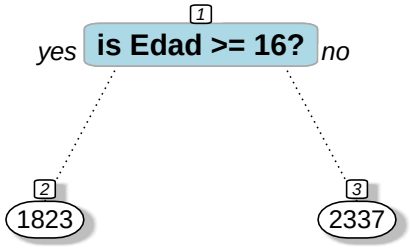
```
## Call:
## rpart(formula = Resistencia ~ Edad, data = datos, method = "anova")
##   n= 20
##
##          CP nsplit rel error   xerror      xstd
## 1 0.7480619      0 1.0000000 1.057314 0.2269184
## 2 0.0100000      1 0.2519381 1.057314 0.2269184
##
## Variable importance
## Edad
##  100
##
## Node number 1: 20 observations,    complexity param=0.7480619
##   mean=2131.358, MSE=84686.88
##   left son=2 (8 obs) right son=3 (12 obs)
##   Primary splits:
##       Edad < 16.25 to the right, improve=0.7480619, (0 missing)
##
## Node number 2: 8 observations
##   mean=1823.094, MSE=19439.95
##
## Node number 3: 12 observations
##   mean=2336.867, MSE=22599.79
```

Construyamos nuevamente el árbol pero explorando todas las opciones de la función `prp`.

```
prp(arb, main = "",
    nn = TRUE,              # display the node numbers
    fallen.leaves = TRUE,   # put the leaves on the bottom of the page
    shadow.col = "gray",    # shadows under the leaves
    branch.lty = 3,         # draw branches using dotted lines
```

```
    branch = .5,           # change angle of branch lines
    faclen = 0,            # faclen = 0 to print full factor names
    trace = 1,             # print the auto calculated cex, xlim, ylim
    split.cex = 1.2,       # make the split text larger than the node text
    split.prefix = "is ",  # put "is " before split text
    split.suffix = "?",    # put "?" after split text
    split.box.col = "lightblue",   # lightgray split boxes (default is white)
    split.border.col = "darkgray", # darkgray border on split boxes
    split.round = 0.5)             # round the split box corners a tad
```

```
## cex 1   xlim c(-0.65, 1.65)   ylim c(-0.15, 1.15)
```

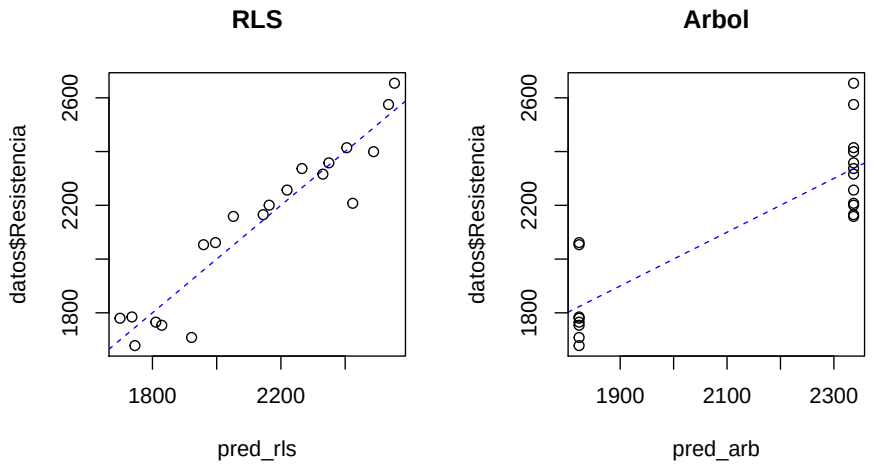


A continuación las predicciones con ambos modelos.

```
pred_rls <- predict(object=rls, newdata=datos)
pred_arb <- predict(object=arb, newdata=datos)
```

Dibujemos y_i versus $\hat{y}_i$.

```
par(mfrow=c(1, 2))
plot(x=pred_rls, y=datos$Resistencia, main="RLS")
abline(a=0, b=1, lty="dashed", col="blue")
plot(x=pred_arb, y=datos$Resistencia, main="Arbol")
abline(a=0, b=1, lty="dashed", col="blue")
```



Vamos a calcular $Cor(y_i, \hat{y}_i)$.

```
cor(datos$Resistencia, pred_rls)
```

```
## [1] 0.9496533
```

```
cor(datos$Resistencia, pred_arb)
```

```
## [1] 0.8649057
```

Calculemos ahora el Error Cuadrático Medio $ECM = \frac{1}{n}\sum(y_i - \hat{y}_i)^2$.

```
mean((datos$Resistencia - pred_rls)^2)
```

```
## [1] 8312.743
```

```
mean((datos$Resistencia - pred_arb)^2)
```

```
## [1] 21335.85
```

¿Cuál método prefiere usted?

Estudio de simulación para comparar ambos métodos

El objetivo es comparar ambos modelos repetidas veces, para esto vamos a simular conjuntos de datos que tengan un comportamiento lineal y parecido a los datos del ejemplo. El modelo que vamos a considerar es el siguiente:

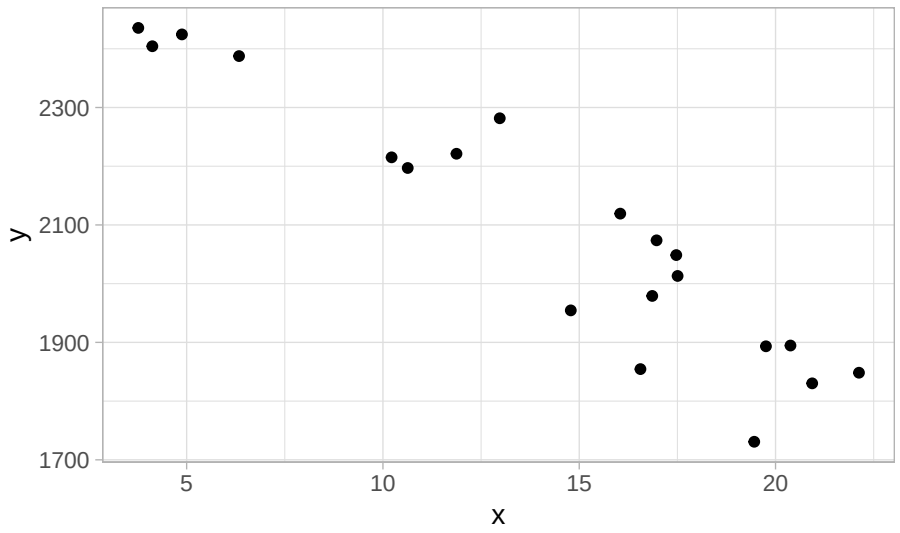
$$y_i \sim N(\mu_i, \sigma^2), \tag{4}$$
$$\mu_i = 2627 - 37x_i, \tag{5}$$
$$\sigma = 96, \tag{6}$$
$$x \sim U(2, 25) \tag{7}$$

Vamos a crear una función generadora de datos.

```
gen_dat <- function(n) {
  x <- runif(n=n, min=2, max=25)
  media <- 2627 - 37 * x
  y <- rnorm(n=n, mean=media, sd=96)
  data.frame(x=x, y=y)
}
```

Generemos unos datos de prueba y graficamos los datos.

```
datos_train <- gen_dat(n=20)
ggplot(datos_train, aes(x=x, y=y)) +
  geom_point() + theme_light()
```



Usando los datos de prueba vamos a ajustar los modelos y luego calcularemos los indicadores.

```
datos_train <- gen_dat(n=20)  # Para entrenar
datos_test  <- gen_dat(n=20)  # Para validar
rls <- lm(y ~ x, data=datos_train)
arb <- rpart(y ~ x, data=datos_train)
pred_rls <- predict(object=rls, newdata=datos_test)
pred_arb <- predict(object=arb, newdata=datos_test)
cor(datos_test$y, pred_rls)
```

```
## [1] 0.9615395
```

```
cor(datos_test$y, pred_arb)
```

```
## [1] 0.8500365
```

```
mean((datos_test$y - pred_rls)^2)
```

```
## [1] 8010.614
```

```
mean((datos_test$y - pred_arb)^2)
```

```
## [1] 23925.86
```

Al observar los resultados anteriores vemos que el modelo de regresión lineal se comporta mejor que el árbol de regresión, esto se debe a que los datos están siendo generados con un modelo de regresión lineal.

Ahora vamos a realizar el estudio de simulación para explorar el efecto de $n = 10, 20, 40$ sobre el ECM usando 5 réplicas para cada n, este es un estudio de simulación "naive" pero ilustrativo.

```
n <- c(10, 20, 40)
nrep <- 5
result <- numeric()
for (i in n) {
  for(k in 1:nrep) {
    datos_train <- gen_dat(n=i)  # Para entrenar
    datos_test  <- gen_dat(n=i)  # Para validar
    rls <- lm(y ~ x, data=datos_train)
    arb <- rpart(y ~ x, data=datos_train)
    pred_rls <- predict(object=rls, newdata=datos_test)
    pred_arb <- predict(object=arb, newdata=datos_test)
    ecm1 <- mean((datos_test$y - pred_rls)^2)
    ecm2 <- mean((datos_test$y - pred_arb)^2)
    result <- rbind(result, c(i, ecm1, ecm2)) # No eficiente pero sirve
  }
}
colnames(result) <- c("n", "ecm_lrs", "ecm_arb")
result <- as.data.frame(result)
result
```

```
##     n   ecm_lrs   ecm_arb
## 1  10 22342.892  67143.51
## 2  10 28186.080 109455.80
## 3  10 18074.550  72013.92
## 4  10  5747.585  79585.71
## 5  10 12592.402  60357.27
## 6  20  7624.788  27244.90
## 7  20  7825.550  29318.18
## 8  20  5721.970  20488.30
## 9  20  9683.974  32764.16
## 10 20 13750.203  21107.22
## 11 40  8436.121  27969.58
## 12 40 12162.325  20550.88
## 13 40 14748.927  14208.98
## 14 40  9615.399  22187.77
## 15 40 11055.010  14160.77
```

El objeto `result` tiene los resultados de la simulación, vamos a calcular el ECM promedio para rls y árboles diferenciando por n.

```
library(dplyr)
result %>% group_by(n) %>% summarise(ecm_medio_lrs=mean(ecm_lrs),
                                     ecm_medio_arb=mean(ecm_arb))
```

```
## # A tibble: 3 x 3
##       n ecm_medio_lrs ecm_medio_arb
##   <dbl>         <dbl>         <dbl>
## 1    10        17389.        77711.
## 2    20         8921.        26185.
## 3    40        11204.        19816.
```

Retos

A continuación los retos que usted debe aceptar.

1. Extienda el estudio de simulación para otros valores de n y aumentando el número de repeticiones `nrep`, decida usted los valores.
2. Con los resultados anteriores haga un gráfico de ECM promedio versus n para rls y árboles en la misma figura.
3. ¿Se iguala ECM promedio del árbol con el de regresión para algún valor de n?
4. ¿Cuál técnica presenta el ECM menor?
5. ¿Es posible encontrar un $ECM = 0$ para algún valor de n?
6. ¿Para qué sirve el paquete `dplyr`?
7. ¿Qué es un `tibble`?

Bibliografía

Braun, W. (2019). *MPV: Data Sets from Montgomery, Peck and Vining*. R package version 1.55.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Chapman and Hall/CRC, Boca Raton, Florida, 1nd edition. ISBN 978-0412048418.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning – with Applications in R*, volume 103 of *Springer Texts in Statistics*. Springer, New York.

Milborrow, S. (2019). *rpart.plot: Plot 'rpart' Models: An Enhanced Version of 'plot.rpart'*. R package version 3.0.7.

Ripley, B. (2019). *tree: Classification and Regression Trees*. R package version 1.0-40.

Therneau, T. and Atkinson, B. (2019). *rpart: Recursive Partitioning and Regression Trees*. R package version 4.1-15.

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2019). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.12.