Tarea 3

Estudiante

John Daniel hoyos Arias
Ivan Santiago Rojas Martinez
Genaro Alfonso Aristizabal Echeverri

Docente

Cesar Augusto Gomez Velez

Asignatura

Analitica de datos

Sede Medellín
15 de Noviembre del 2022

Índice

1. Ejercicio1 **4**

1.1. a) Use la función poly() para ajustar una regresión polinomial cúbica y con esta predecir la variable nox usando dis. Reporte el resultado de la regresión, luego grafique los datos resultantes y los ajustes polinómicos. 4

1.2. b) Grafique los ajustes polinómicos para un rango de polinomios de diferentes grados (digamos, de 1 a 10), y reporte la suma de cuadrados de los residuales asociada. 6

1.3. c) Realice una validación cruzada o algún otro enfoque para seleccionar el óptimo grado para el polinomio y explique sus resultados. 7

1.3.1. K-fold Cross-Validation 7

1.4. d) Use la función bs() para ajustar una spline de regresión para predecir nox usando dis. Reporte la salida para el ajuste usando cuatro grados de libertad. ¿Cómo ubicó los nodos?. Grafique el ajuste resultante. 8

1.5. e) Ahora ajuste una spline de regresión para un rango de grados de libertad, y grafique los ajustes resultantes e informe el RSS resultante. Describa los resultados obtenidos. 9

1.5.1. Analisis de RSS 11

1.6. f) Realice una validación cruzada o algún otro enfoque para seleccionar los mejores grados de libertad para una spline de regresión sobre estos datos. Describa sus resultados. 11

2. Ejercicio2 **14**

2.1. a)Conjunto de entrenamiento y prueba 15

2.1.1. Conjunto de entrenamiento 15

2.1.2. Conjunto de prueba 16

2.2. b)Arbol de regresión 17

2.2.1. Interpretación 18

2.3. c) Validadación cruzada y poda del arbol de regresión 19

2.3.1. Interpretación 20

2.4. d) Bagging 21

2.5. e) Bosque aleatorio 23

1. Ejercicio1

1.1. a) Use la función poly() para ajustar una regresión polinomial cúbica y con esta predecir la variable nox usando dis. Reporte el resultado de la regresión, luego grafique los datos resultantes y los ajustes polinómicos.

```
attach(Boston)
```

```
fit1 <- lm(nox ~ poly(dis,4), data = Boston)
coef(summary(fit1))
```

```
##                     Estimate  Std. Error     t value      Pr(>|t|)
## (Intercept)    0.55469506 0.002761339 200.8790240  0.000000e+00
## poly(dis, 4)1 -2.00309590 0.062114782 -32.2482963 2.540459e-124
## poly(dis, 4)2  0.85632995 0.062114782  13.7862506  6.924872e-37
## poly(dis, 4)3 -0.31804899 0.062114782  -5.1203430  4.356581e-07
## poly(dis, 4)4  0.03354668 0.062114782   0.5400757  5.893848e-01
```

Al ajustar la regresión correspondiente, se pudo determinar que es suficiente considerar un polinomio de grado 3. Mientras que con un polinomio de grado 4, se comienzan a presentar problemas de sobreparametrización y, por lo tanto, algunos de los parámetros dejan de ser significativos para el modelo.

```
fit <- lm(nox ~ poly(dis,3), data = Boston)
(summary(fit))
```

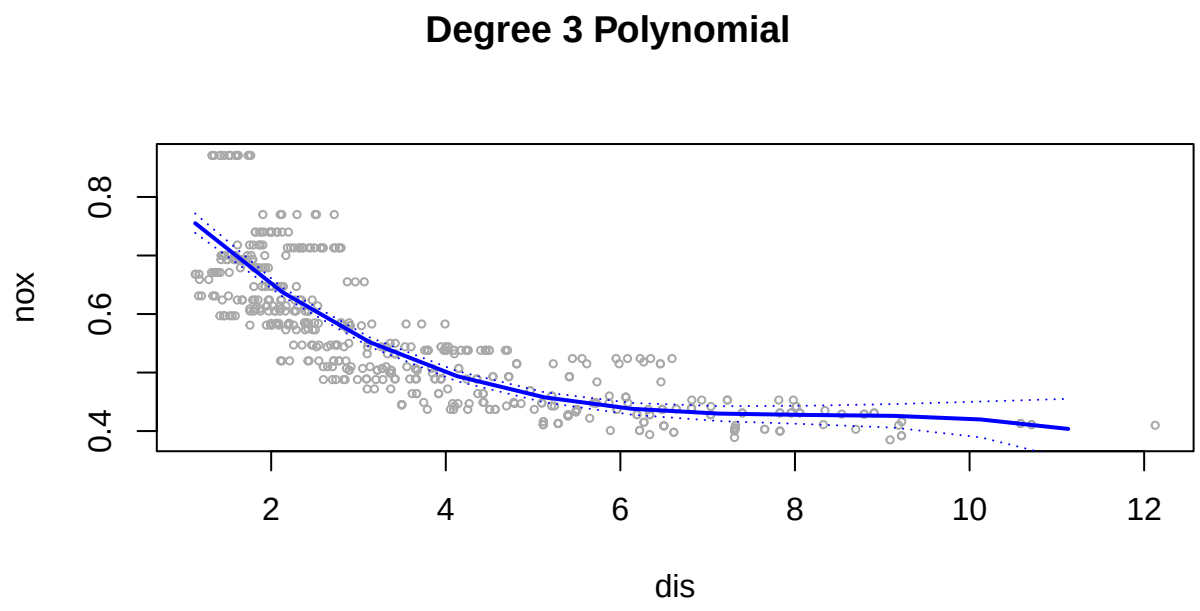
```
##
## Call:
## lm(formula = nox ~ poly(dis, 3), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.121130 -0.040619 -0.009738  0.023385  0.194904
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.554695   0.002759 201.021  < 2e-16 ***
## poly(dis, 3)1 -2.003096   0.062071 -32.271  < 2e-16 ***
## poly(dis, 3)2  0.856330   0.062071  13.796  < 2e-16 ***
## poly(dis, 3)3 -0.318049   0.062071  -5.124 4.27e-07 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06207 on 502 degrees of freedom
## Multiple R-squared:  0.7148, Adjusted R-squared:  0.7131
## F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

Todas las variables en este modelo de regresión cúbica parecen tener valores p que son estadísticamente significativos. El error estándar residual es bajo pero los grados de libertad son altos. Los valores de R-Squared son relativamente altos en 0.71 para explicar la varianza en el modelo.

```
dislims <- range(dis)
dis.grid <- seq(from = dislims[1], to = dislims [2])
preds <- predict(fit , newdata = list(dis = dis.grid),se = TRUE)
se.bands <- cbind(preds$fit + 2 * preds$se.fit ,preds$fit - 2 * preds$se.fit)
```

```
par(mfrow = c(1, 1), mar = c(4.5 , 4.5, 1, 1),
oma = c(0, 0, 4, 0))
plot(dis , nox , xlim = dislims , cex = .5, col = "darkgrey")
title("Degree 3 Polynomial", outer = T)
lines(dis.grid, preds$fit , lwd = 2, col = "blue")
matlines(dis.grid , se.bands, lwd = 1, col = "blue", lty = 3)
```



Graficamente, observamos un ajuste bastante bueno al considerar un polinomio de grado 3.

1.2. b) Grafique los ajustes polinómicos para un rango de polinomios de diferentes grados (digamos, de 1 a 10), y reporte la suma de cuadrados de los residuales asociada.

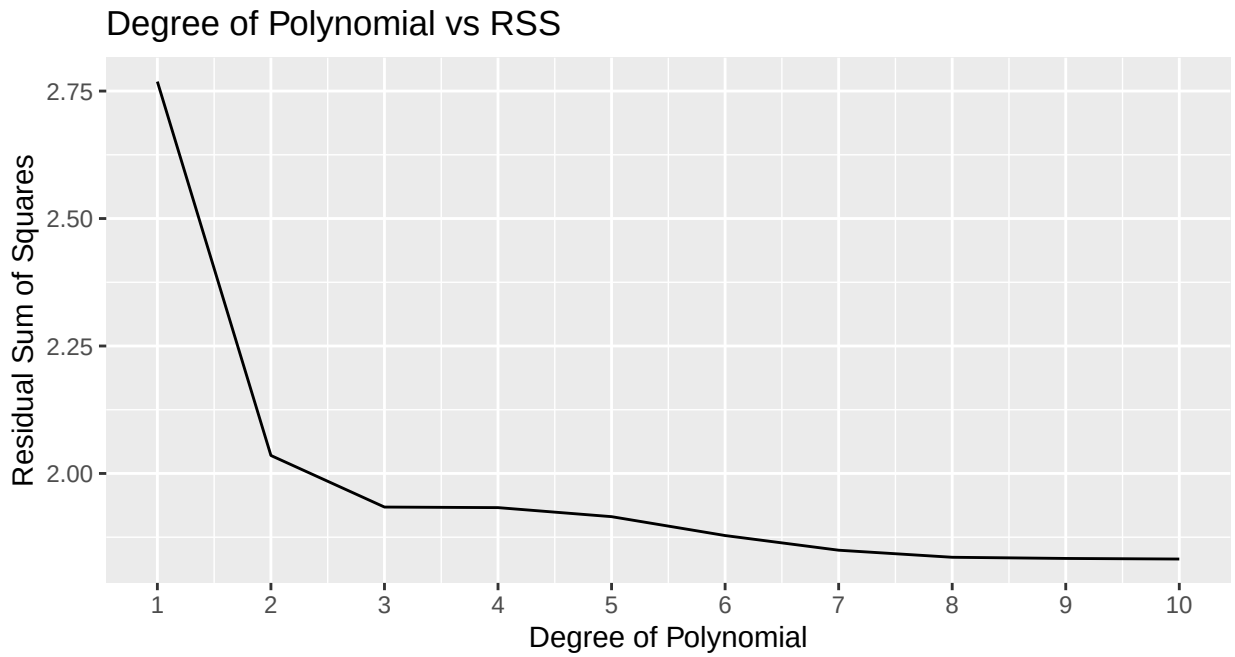
Se ajustarán modelos que van desde uno lineal hasta un polinomio de grado 5, para determinar el modelo más simple que sea suficiente para explicar la relación entre nox y dis.

```
## Analysis of Variance Table
##
## Model 1: nox ~ dis
## Model 2: nox ~ poly(dis, 2)
## Model 3: nox ~ poly(dis, 3)
## Model 4: nox ~ poly(dis, 4)
## Model 5: nox ~ poly(dis, 5)
##   Res.Df    RSS Df Sum of Sq        F    Pr(>F)
## 1    504 2.7686
## 2    503 2.0353  1   0.73330 191.4334 < 2.2e-16 ***
## 3    502 1.9341  1   0.10116  26.4073 3.972e-07 ***
## 4    501 1.9330  1   0.00113   0.2938   0.58804
## 5    500 1.9153  1   0.01769   4.6185   0.03211 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##                   Estimate  Std. Error     t value      Pr(>|t|)
## (Intercept)    0.55469506 0.002751421 201.6031400  0.000000e+00
## poly(dis, 5)1 -2.00309590 0.061891679 -32.3645429 9.417911e-125
## poly(dis, 5)2  0.85632995 0.061891679  13.8359464  4.303120e-37
## poly(dis, 5)3 -0.31804899 0.061891679  -5.1388005  3.971902e-07
## poly(dis, 5)4  0.03354668 0.061891679   0.5420225  5.880445e-01
## poly(dis, 5)5  0.13300890 0.061891679   2.1490594  3.210905e-02
```

Parece ser más apropiado utilizar el modelo cúbico donde El p-valor que compara el polinomio de grado 4, Model3 y Model4, es aproximadamente 59 % mientras que el polinomio de grado 5 Model5 parece innecesario por qué su p-valor es 0.03. Por lo tanto, un modelo cuadrático parece proporcionar un ajuste razonable a los datos en comparación a modelos de grados inferiores y superiores.

```
#library(data.table)
rss <- data.table(seq(1:10), rss, keep.rownames = TRUE)
ggplot(rss, aes(V1, rss)) +
  geom_line() +
  scale_x_continuous(breaks=c(1:10)) +
  labs(x='Degree of Polynomial', y='Residual Sum of Squares', title='Degree of Polynomia
```




```
rss[c(2,3),]
```

```
##    V1      rss
## 1:  2 2.035262
## 2:  3 1.934107
```

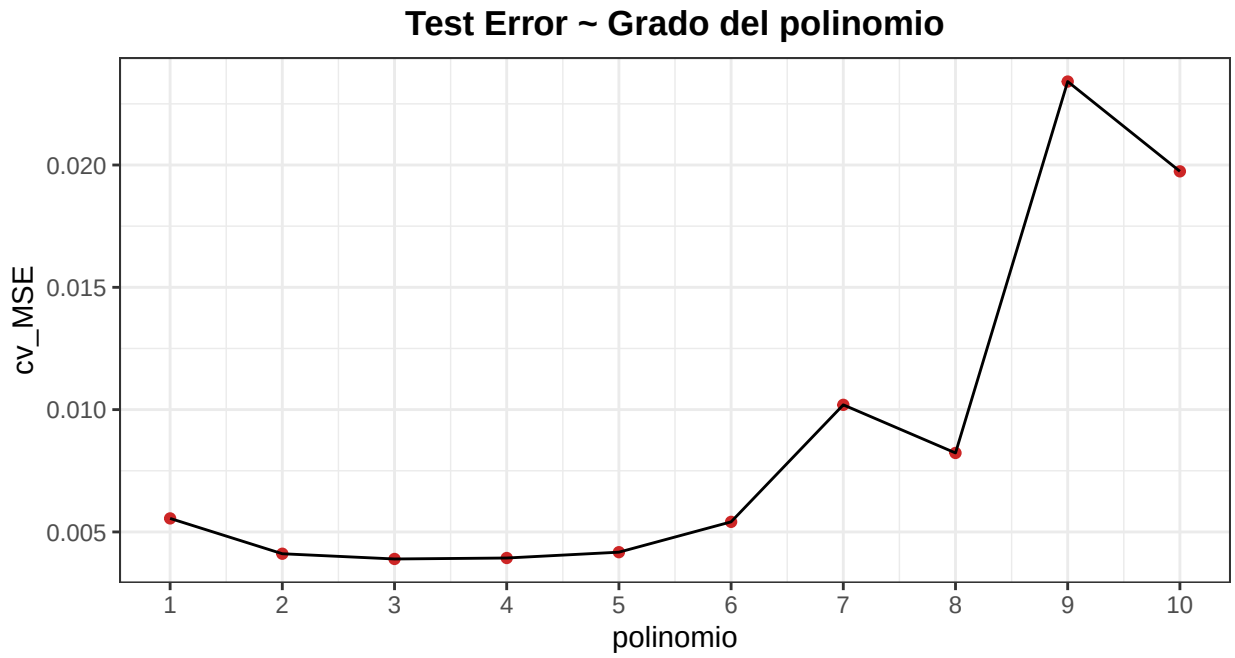
el menor Rss se obtiene con 10 grados de libertad.

1.3. c) Realice una validación cruzada o algún otro enfoque para seleccionar el óptimo grado para el polinomio y explique sus resultados.

1.3.1. K-fold Cross-Validation

```
#library(boot)
#library(ggplot2)
cv_MSE_k10 <- rep(NA,10)

for (i in 1:10) {
  modelo <- glm(nox ~ poly(dis, i), data = Boston)
  set.seed(17)
  cv_MSE_k10[i] <- cv.glm(data = Boston, glmfit = modelo, K = 10)$delta[1]
}
```

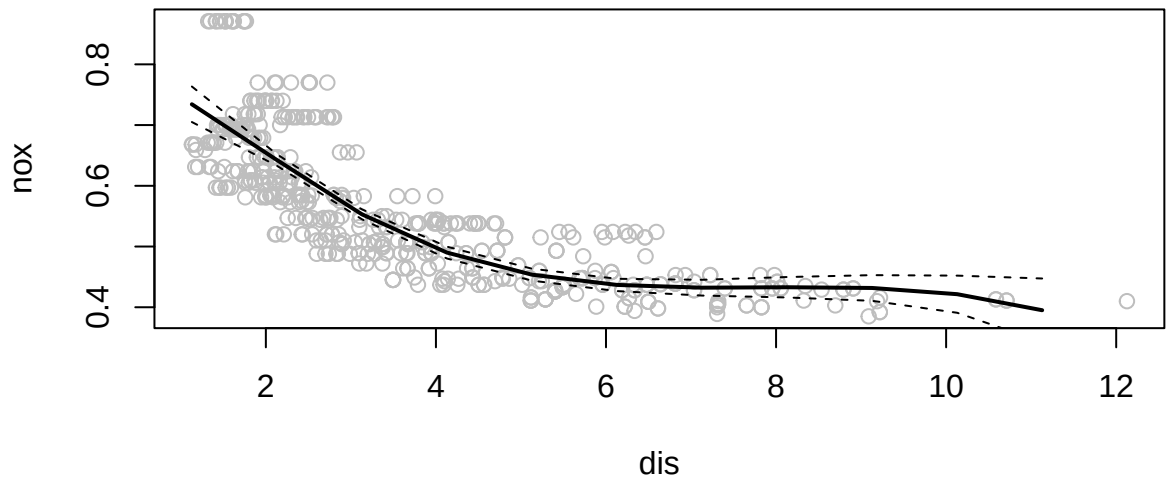
```
ggplot(data = data.frame(polinomio = 1:10, cv_MSE = cv_MSE_k10),
       aes(x = polinomio, y = cv_MSE)) +
geom_point(colour = c("firebrick3")) +
geom_path() +
scale_x_continuous(breaks = c(0:10)) +
theme_bw() +
labs(title  =  'Test Error ~ Grado del polinomio') +
theme(plot.title = element_text(hjust = 0.5, face = 'bold'))
```



Tras el proceso de validación cruzada, se determina que el mejor grado del polinomio es efectivamente el de grado 3, dado que es donde se presenta un menor MSE.

1.4. d) Use la función bs() para ajustar una spline de regresión para predecir nox usando dis. Reporte la salida para el ajuste usando cuatro grados de libertad. ¿Cómo ubicó los nodos?. Grafique el ajuste resultante.

```
#library(splines)
fit <- lm(nox ~ bs(dis , knots = c(3.2)), data = Boston)
pred <- predict(fit , newdata = list(dis = dis.grid), se = T)
plot(dis , nox , col = "gray")
lines(dis.grid, pred$fit , lwd = 2)
lines(dis.grid , pred$fit + 2 * pred$se, lty = "dashed")
lines(dis.grid , pred$fit - 2 * pred$se, lty = "dashed")
```




```
#library(ggplot2)

attr(bs(dis , df = 4), "knots")
```

```
##     50%
## 3.20745
```

Para ajustar una spline de regresión con 4 grados de libertad, se utilizó la función attr la cual nos permite determinar la posición de los nodos según los grados de libertad, en este caso, como acabamos de observar, para 4 grados de libertad, solo permite un nodo que representa la mediana en 3.2.

1.5. e) Ahora ajuste una spline de regresión para un rango de grados de libertad, y grafique los ajustes resultantes e informe el RSS resultante. Describa los resultados obtenidos.

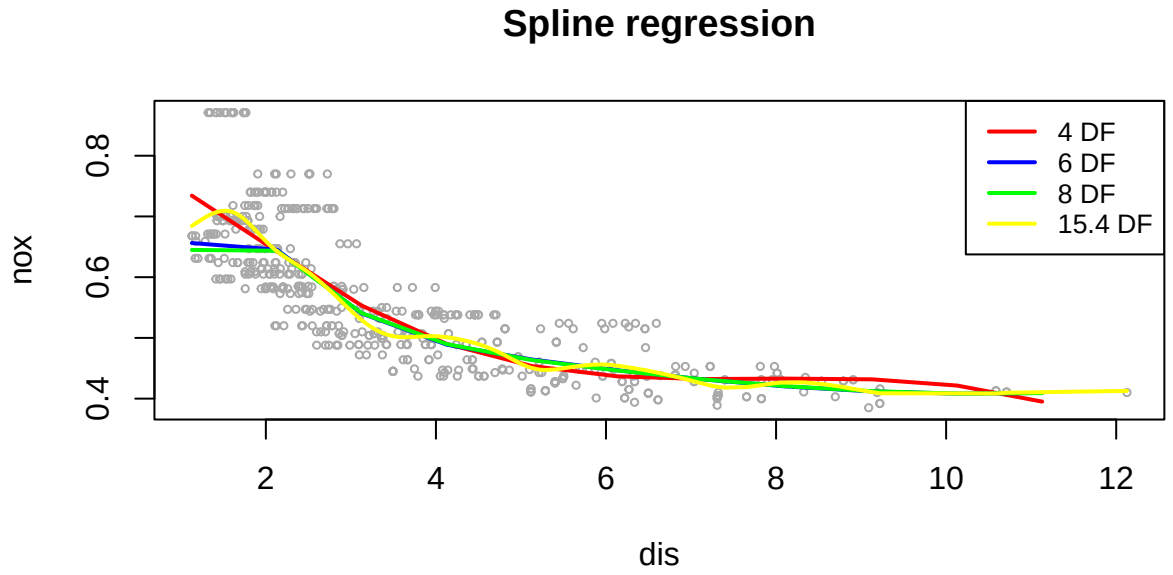
```
plot.new()
plot(dis , nox , xlim = dislims , cex = .5, col = "darkgrey")
title("Spline regression")
# 4grados de libertad
fit.1 <- lm(nox ~ bs(dis , knots = c(3.2)), data = Boston)
# 6grados de libertad
attr(bs(dis , df = 6), "knots")
```

```
##       25%       50%       75%
## 2.100175 3.207450 5.188425
```

```
fit.2 <- lm(nox ~ bs(dis , knots = c(2.1,3.2,5.18)), data = Boston)
# 7grados de libertad
attr(bs(dis , df = 7), "knots")
```

```
##    20%    40%    60%    80%
## 1.9512 2.6403 3.8750 5.6150
```

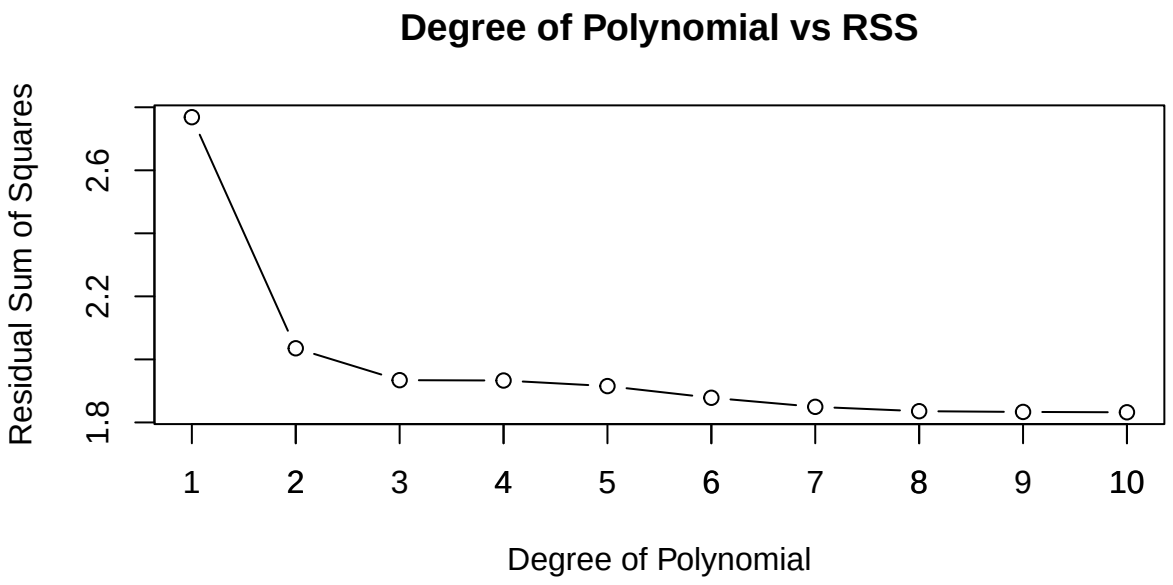
```
fit.3 <- lm(nox ~ bs(dis , knots = c(1.95,2.6,3.87,5.61)), data = Boston,cv=T)
#spline suave, con grados de libertad obtenidos a partir de cv
fit.4 <- smooth.spline(dis , nox , cv = TRUE)

pred1 <- predict(fit.1 , newdata = list(dis = dis.grid), se = T)
pred2 <- predict(fit.2, newdata = list(dis = dis.grid), se = T)
pred3<- predict(fit.3, newdata = list(dis = dis.grid), se = T)

lines(dis.grid, pred1$fit ,col='red', lwd = 2)
lines(dis.grid, pred2$fit ,col='blue', lwd = 2)
lines(dis.grid, pred3$fit ,col='green', lwd = 2)
lines(fit.4 , col = "yellow", lwd = 2)

legend("topright", legend = c("4 DF", "6 DF",'8 DF','15.4 DF'),col = c("red", "blue",'gr
```



Se observan splines de regresión con 4, 6, 7 y un spline suave con 15.4 gl, claramente el mejor ajuste se observa para un spline cúbico con 4 grados de libertad, a medida que se aumentan los grados de libertad, la tendencia del ajuste se deja llevar por el ruido y esto genera problemas de predicción.

1.5.1. Analisis de RSS

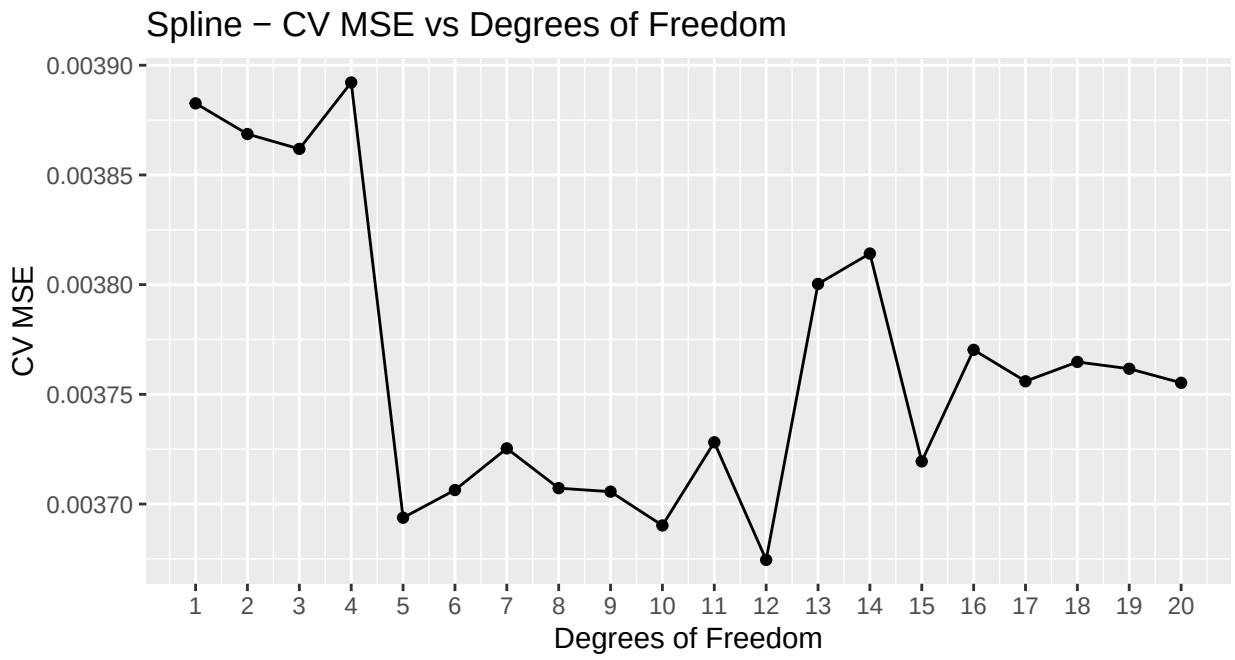
```
rss <- rep(NA, 10)
for (i in 1:10) {
  poly.fit <- lm(nox ~ poly(dis, i), data=Boston)
  rss[i] <- sum(poly.fit$residuals^2)
}

plot(1:10, rss, xlab='Degree of Polynomial', ylab='Residual Sum of Squares', type='b', m
axis(1, at = seq(1,10, by=1))
```



1.6. f) Realice una validación cruzada o algún otro enfoque para seleccionar los mejores grados de libertad para una spline de regresión sobre estos datos. Describa sus resultados.

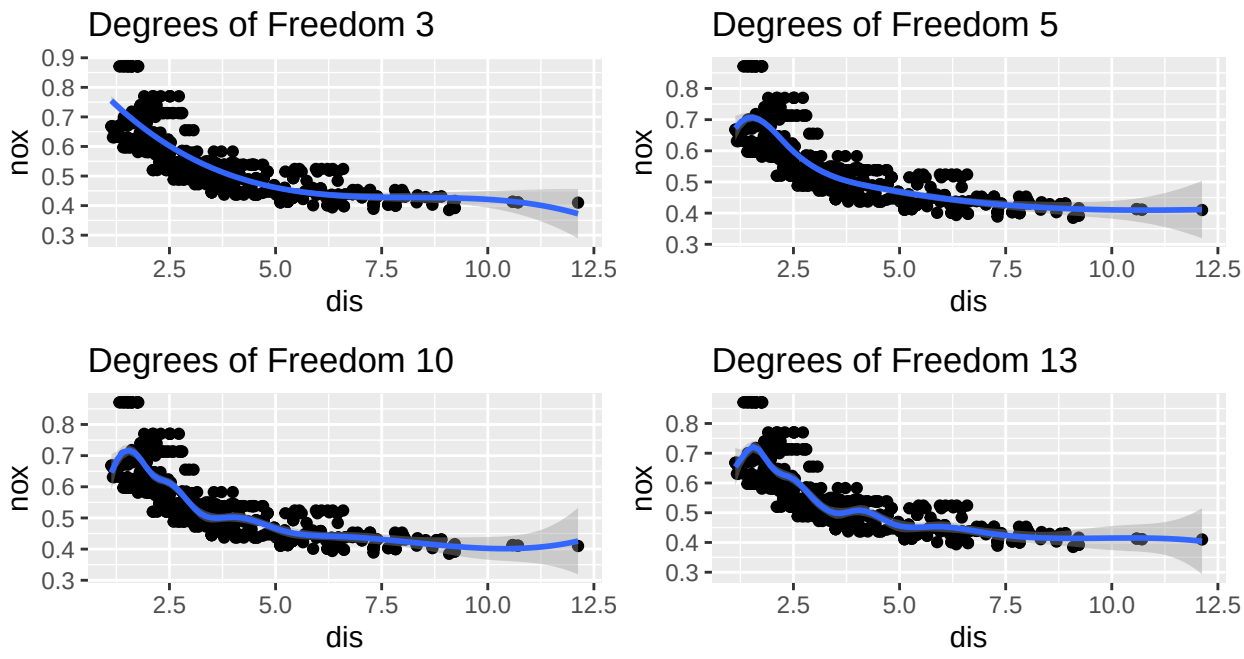
```
#library(boot)
#library(data.table)
```

```
#library(ggplot2)
cv.spline.fun <- function(i) {
  fit <- glm(nox ~ bs(dis, df=i), data=Boston)
  cv.error <- cv.glm(Boston, fit, K=10)$delta[2]
}

cv.err <- sapply(1:20, cv.spline.fun)

df <- seq(1:20)

dt <- data.table(df, cv.err)

#plot.new()
ggplot(dt, aes(df, cv.err)) +
  geom_point() +
  geom_line() +
  scale_x_continuous(breaks=seq(1:20)) +
  labs(x='Degrees of Freedom', y='CV MSE', title='Spline - CV MSE vs Degrees of Freedom'
```




```
#library(gridExtra)

p1 <- ggplot(Boston, aes(x=dis, y=nox)) +
    geom_point(colour='black') +
    stat_smooth(method='lm', formula= y ~ bs(x,df=3)) +
  labs(x='dis',y='nox', title='Degrees of Freedom 3')
```

```
p2 <- ggplot(Boston, aes(x=dis, y=nox)) +
    geom_point(colour='black') +
    stat_smooth(method='lm', formula= y ~ bs(x,df=5)) +
  labs(x='dis',y='nox', title='Degrees of Freedom 5')


p3 <- ggplot(Boston, aes(x=dis, y=nox)) +
    geom_point(colour='black') +
    stat_smooth(method='lm', formula= y ~ bs(x,df=10)) +
  labs(x='dis',y='nox', title='Degrees of Freedom 10')


p4 <- ggplot(Boston, aes(x=dis, y=nox)) +
    geom_point(colour='black') +
    stat_smooth(method='lm', formula= y ~ bs(x,df=13)) +
  labs(x='dis',y='nox', title='Degrees of Freedom 13')


grid.arrange(p1,p2,p3,p4, ncol=2, nrow=2)
```



Se observa que el ajuste puede darse entre 5 y 10 grados de libertad, aunque según el RSS el que mejor ajuste tiene es el de 10 grados de libertad.

2. Ejercicio2

En este ejercicio se utilizarán arboles de regresión para predecir los valores de la variable **sales** en la base de datos **Carseats** de la libreria **ISLR2**, tratando dicha variable como continua:

Primeramente, se cargan los datos y se examinan sus caracteristicas:

```
head(Carseats)
```

```
##   Sales CompPrice Income Advertising Population Price ShelveLoc Age Education
## 1  9.50       138     73          11        276   120       Bad  42        17
## 2 11.22       111     48          16        260    83      Good  65        10
## 3 10.06       113     35          10        269    80    Medium  59        12
## 4  7.40       117    100           4        466    97    Medium  55        14
## 5  4.15       141     64           3        340   128       Bad  38        13
## 6 10.81       124    113          13        501    72       Bad  78        16
##   Urban  US
## 1   Yes Yes
## 2   Yes Yes
## 3   Yes Yes
## 4   Yes Yes
## 5   Yes  No
## 6    No Yes
```

```
str(Carseats)
```

```
## 'data.frame':    400 obs. of  11 variables:
##  $ Sales      : num  9.5 11.22 10.06 7.4 4.15 ...
##  $ CompPrice  : num  138 111 113 117 141 124 115 136 132 132 ...
##  $ Income     : num  73 48 35 100 64 113 105 81 110 113 ...
##  $ Advertising: num  11 16 10 4 3 13 0 15 0 0 ...
##  $ Population : num  276 260 269 466 340 501 45 425 108 131 ...
##  $ Price      : num  120 83 80 97 128 72 108 120 124 124 ...
##  $ ShelveLoc  : Factor w/ 3 levels "Bad","Good","Medium": 1 2 3 3 1 1 3 2 3 3 ...
##  $ Age        : num  42 65 59 55 38 78 71 67 76 76 ...
##  $ Education  : num  17 10 12 14 13 16 15 10 10 17 ...
##  $ Urban      : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 2 2 1 1 ...
##  $ US         : Factor w/ 2 levels "No","Yes": 2 2 2 2 1 2 1 2 1 2 ...
```

La base **Carseats** constituye un conjunto de datos simulados que contiene las ventas de sillas de coche para niños en 400 tiendas diferentes.

Dicha base contiene las siguientes 11 variables:

- **Sales:** Ventas unitarias (en miles) en cada ubicación.
- **CompPrice:** Precio cobrado por el competidor en cada ubicación.
- **Income:** Nivel de ingresos de la comunidad (en miles de dólares).
- **Advertising:** Presupuesto de publicidad local para la empresa en cada ubicación (en miles de dólares).
- **Population:** Tamaño de la población en la región (en miles).
- **Price:** Precio de los cargos de la compañía por los asientos de seguridad en cada sitio.
- **ShelveLoc:** Un factor con niveles en el que **Bad** indica la calidad de la ubicación de las estanterías para los asientos de automóvil en cada **sitioGoodMedium**.
- **Age:** Edad media de la población local.
- **Education:** Nivel de educación en cada lugar.
- **Urban:** Un factor con niveles **No** y **Yes** para indicar si la tienda está en una zona urbana o rural.
- **US:** Un factor con niveles **No** y **Yes** para indicar si la tienda está en USA o no.

2.1. a)Conjunto de entrenamiento y prueba

Se procede a dividir el conjunto de observaciones en un conjunto de entrenamiento y un conjunto de prueba. De la siguiente manera:

```
# cargo de libreria para arboles de regresión
library(tree)
```

De forma aleatoria se realiza un *sample* del conjunto de datos para posteriormente crear los conjuntos de prueba y de entrenamiento:

```
set.seed (123)
train = sample(1: nrow(Carseats), nrow(Carseats)*0.7)
```

2.1.1. Conjunto de entrenamiento

```
# datos de entrenamiento para la base Carseats
train_car = Carseats[train, ]
dim(train_car) # dimensiones de los datos de entrenamiento
```

```
## [1] 280  11
```

```
head(train_car) # encabezado de los datos de entrenamiento
```

```
##     Sales CompPrice Income Advertising Population Price ShelveLoc Age Education
## 179 10.66       104     71          14         89    81    Medium  25        14
## 14  10.96       115     28          11         29    86      Good  53        18
## 195  7.23       112     98          18        481   128    Medium  45        11
## 306  8.03       115     29          26        394   132    Medium  33        13
## 118  8.80       145     53           0        507   119    Medium  41        12
## 299 10.98       148     63           0        312   130      Good  63        15
##     Urban  US
## 179    No Yes
## 14    Yes Yes
## 195   Yes Yes
## 306   Yes Yes
## 118   Yes  No
## 299   Yes  No
```

2.1.2. Conjunto de prueba

```
# datos de prueba para la base Carseats
test_car = Carseats[-train, ]
dim(test_car) # dimensiones de los datos de prueba
```

```
## [1] 120  11
```

```
head(test_car) # encabezado de los datos de prueba
```

```
##    Sales CompPrice Income Advertising Population Price ShelveLoc Age Education
## 1   9.50       138     73          11        276   120       Bad  42        17
## 3  10.06       113     35          10        269    80    Medium  59        12
## 6  10.81       124    113          13        501    72       Bad  78        16
## 8  11.85       136     81          15        425   120      Good  67        10
## 12 11.96       117     94           4        503    94      Good  50        13
## 15 11.17       107    117          11        148   118      Good  52        18
##    Urban  US
## 1    Yes Yes
## 3    Yes Yes
## 6     No Yes
## 8    Yes Yes
## 12   Yes Yes
## 15   Yes Yes
```

Los datos de entrenamiento y prueba se dividieron en proporciones de 0.7 y 0.3 con respecto a la base **Carseats**. Es decir, el 70 % de dicha base corresponden con datos de entrenamiento (280 registros) y el 30 % con los datos de prueba (120 registros).

2.2. b)Arbol de regresión

Ahora se ajusta un árbol de regressión en el conjunto de entrenamiento.

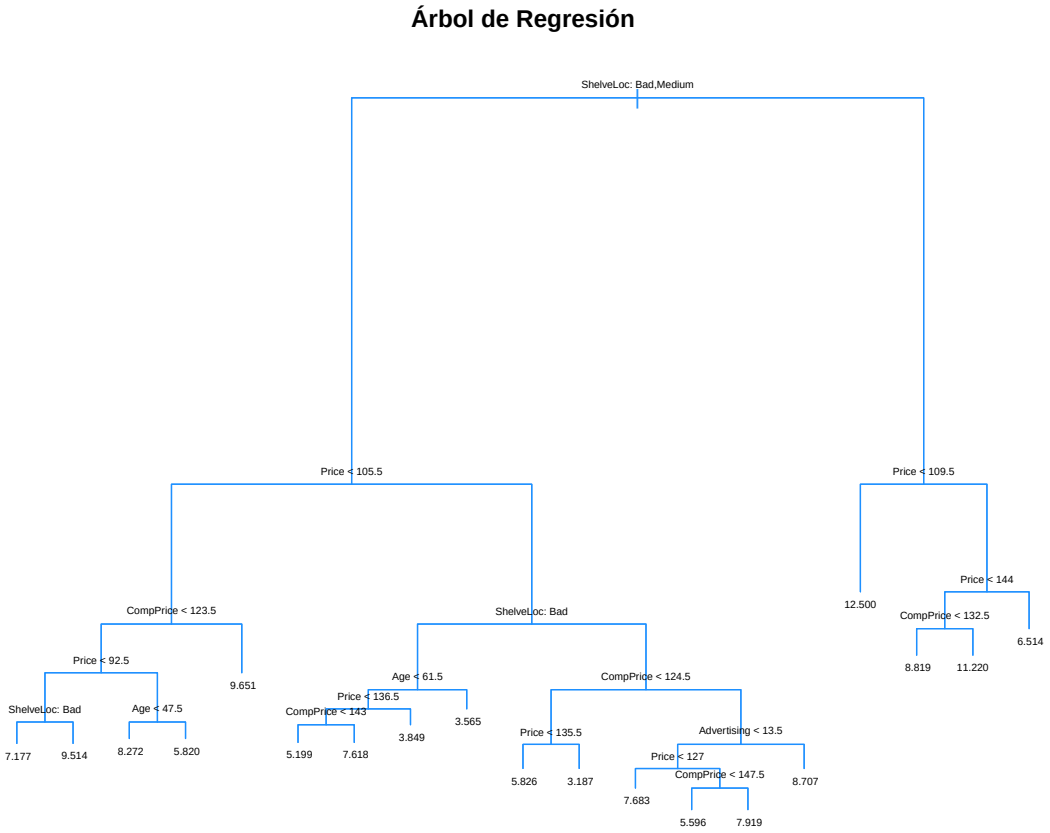
```
# ajuste del arbol de regresión
tree_carseats <-  tree(Sales~.,Carseats,subset = train)

# resumen del modelo
summary(tree_carseats)
```

```
##
## Regression tree:
## tree(formula = Sales ~ ., data = Carseats, subset = train)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "CompPrice"   "Age"         "Advertising"
## Number of terminal nodes:  19
## Residual mean deviance:  2.373 = 619.2 / 261
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -4.1570 -1.0160  0.1123  0.0000  0.8903  4.0310
```

Se observa como el modelo de regresión seleccionó 5 variables las cuales son: **“ShelveLoc”**, **“Price”**, **“CompPrice”**, **“Age”** y **“Advertising”**. Además, dicho árbol estableció 19 nodos terminales.

Luego, se Gráfica dicho árbol y se interpretan los resultados.

```
plot(tree_carseats, col = '#1E90FF')
text(tree_carseats, pretty = 0, cex = 0.5)
title('Árbol de Regresión', cex = 2)
```



2.2.1. Interpretación

Se observa del árbol de regresión como en su nivel más alto, separa por el factor **ShelveLoc**, en los niveles **bad** y **medium**. En su segundo nivel más alto separa por la variable **Price** esto índica que los precios de los cargos de la compañía por los asientos de seguridad en cada sitio. En este caso, cuando **ShelveLoc** está en un nivel **bad** son menores a 105.5. Mientras que, cuando **ShelveLoc** está en un nivel **medium** son menores a 109.5. También se observa como los precios varían según la edad y el precio cobrado por cada competidor en cada ubicación.

También se observa como, el indicador más importante de **Sales** parece ser un "nivel de calidad de ubicación" (**ShelveLoc**) debido a que la primera rama separa la categoría **Good** de las categorías **Bad** y **Medium**. Cuando el precio de la silla de bebé para carro (**Price**), tiende a ser más alto, las ventas son en promedio menores.

Por otro lado, cuando la calidad de ubicación es buena (**ShelveLoc**), las ventas son en promedio más altas que cuando la calidad de ubicación es mala. No obstante, cuando la inversión de publicidad es más alta (**Advertising**), las ventas son mayores.

¿Qué valor para el MSE de prueba se obtiene?

Para dar respuesta a esta pregunta se realiza lo siguiente:

```
yhat <- predict(tree_carseats, newdata = test_car)
mean((yhat - test_car$Sales)^2)
```

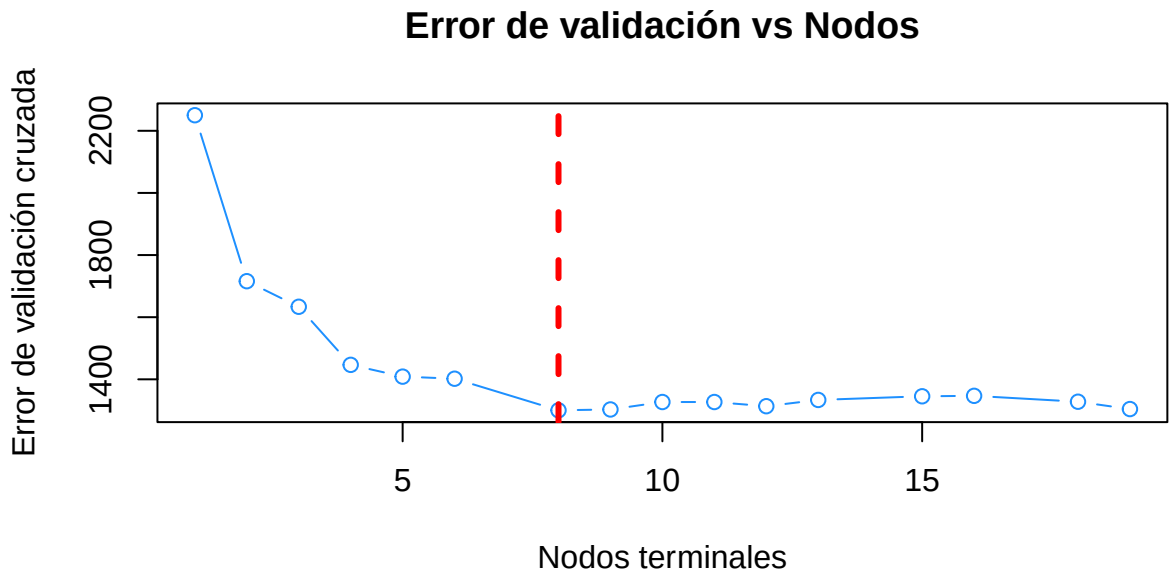
```
## [1] 3.602818
```

Se obtiene un error de prueba de 3.602818; la raiz cuadrada de dicho valor es 1.898109; indicando que en este modelo las ventas están alrededor de $1.898109 del verdadero valor.

2.3. c) Validadación cruzada y poda del arbol de regresión

Se utiliza la validación cruzada usando la función **cv.tree()** para ver si una poda del árbol mejora su desempeño y a fin de determinar el grado óptimo de complejidad del árbol.

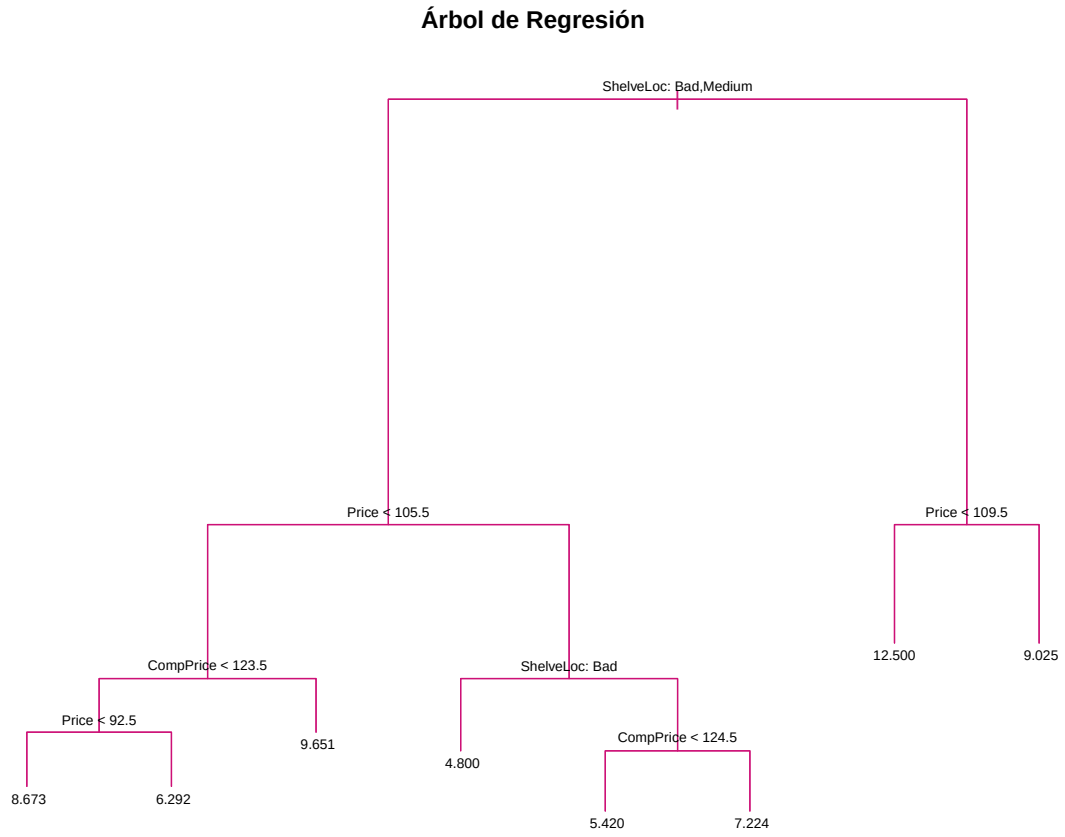
```
# validación cruzada con el árbol de regresión de entrenamiento
set.seed(123)
cv_carseats =cv.tree(tree_carseats)
plot(cv_carseats$size ,cv_carseats$dev,type='b', col = "#1E90FF",
      ylab = 'Error de validación cruzada', xlab = 'Nodos terminales',
     main = 'Error de validación vs Nodos')
abline(v=8, col="red", lwd=3, lty=2)
```



Según lo evidencia el gráfico, el grado óptimo de complejidad del árbol, es aquel que cuenta con 8 nodos.

Ahora bien si se quiere podar el árbol, se utiliza la función **prune.tree()**.

```
# Se poda el arbol de regresión
prune_carseats = prune.tree(tree_carseats ,best = 8)
plot(prune_carseats, col = "#CD1076")
text(prune_carseats ,pretty =0, cex = 0.7)
title('Árbol de Regresión', cex = 2)
```



2.3.1. Interpretación

De este arbol de regresion se observa como, cuando los ingresos de la comunidad (**income**) son menores, las ventas son menores. El indicador más importante de **Sales** parece ser un "nivel de calidad de ubicación" (**ShelveLoc**) debido a que la primera rama separa la categoría **Good** de las categorías **Bad** y **Medium**.

Además, cuando el precio de la silla de bebé para carro (**Price**), tiende a ser más alto, las ventas son en promedio menores. Cuando la calidad de ubicación es bueno (**ShelveLoc**), las ventas son en promedio más altas que cuando la calidad de ubicación es mala. Por otro lado, cuando la inversión de publicidad es más alta (**Advertising**), las ventas son mayores.

¿Qué valor para el MSE de prueba se obtiene?

Para dar respuesta a esta pregunta se realiza lo siguiente:

```
yhat_cv <- predict(prune_carseats, newdata = test_car)
mean((yhat_cv - test_car$Sales)^2)
```

```
## [1] 4.353022
```

Se observa como a pesar de que se disminuyo la complejidad de los nodos terminales, el MSE obtenido aumento, en este caso paso de ser de 3.602818 a 4.353022 con la poda. Por lo cual en este caso la poda del árbol no consiguió mejorar el MSE de prueba.

2.4. d) Bagging

Se utiliza el método **bagging** para analizar estos datos **Carseats** con la función **randomForest()** y usando la función **importance()** para determinar cuál de las variables es la más importante. Dicho procedimiento se muestra como sigue:

```
# cargando el paquete para realizar Random Forest y Bagging
library(randomForest)
set.seed(1)
bag_carseats = randomForest(Sales~.,data=Carseats ,subset =train,mtry=10, importance =TR
bag_carseats
```

```
##
## Call:
##  randomForest(formula = Sales ~ ., data = Carseats, mtry = 10,      importance = TRUE
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 10
##
##           Mean of squared residuals: 2.574801
##                     % Var explained: 67.63
```

Se utilizan 500 árboles de regresión, para ajustar un modelo bagging. Ahora se el MSE de prueba con el método **bagging**:

```
# MSe con el método bagging
yhat.bag <- predict(bag_carseats, newdata = test_car)
mean((yhat.bag - test_car$Sales)^2)
```

```
## [1] 2.272119
```

- Se observa que con el método **bagging** se obtiene un MSE de 2.272119, menor al de los árboles anteriores con y sin la poda.

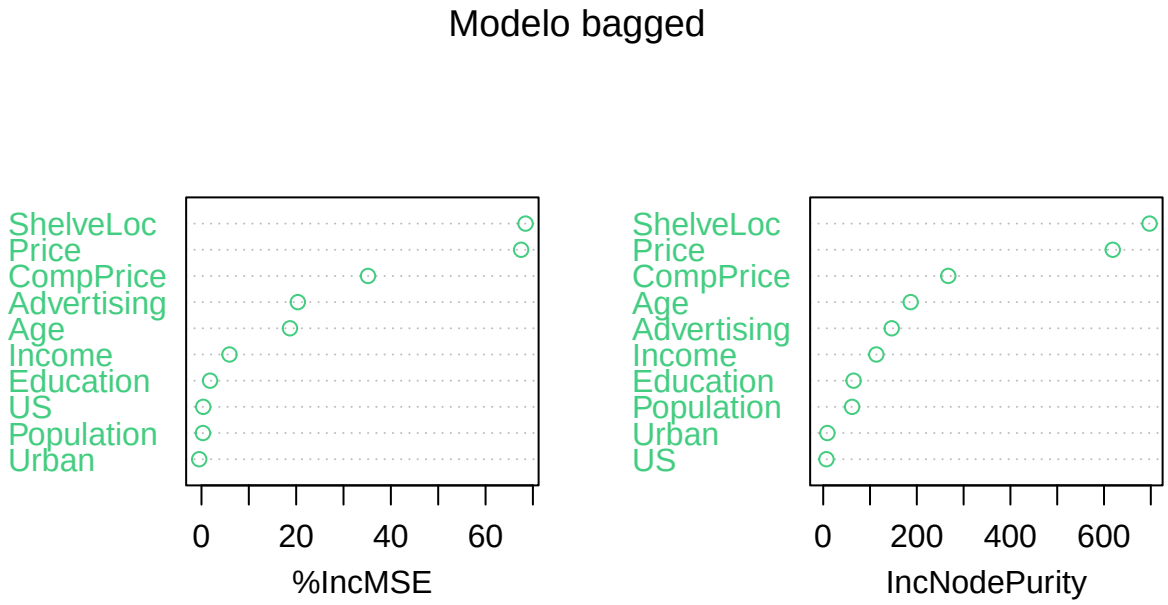
Utilizando la función "importance()", se indaga por la importancia de cada variable:

```
importance(bag_carseats)
```

```
##                %IncMSE IncNodePurity
## CompPrice   35.1770243    267.204047
## Income       5.9175775    113.497425
## Advertising 20.3537856    146.375789
## Population   0.3062319     61.711257
## Price       67.5118343    618.672228
## ShelveLoc   68.4483441    697.271381
## Age         18.6917816    186.902308
## Education    1.8144653     64.970061
## Urban       -0.4783088      8.909692
## US           0.3661361      6.959288
```

- Cuando las variables **ShelveLoc** y **Price** se excluyen, se presenta la mayor disminución media en la precisión de las ventas, en las muestras **bagging**, estas medidas son de 50.78 y 46.38, respectivamente.
- Cuando las variables **ShelveLoc** y **Price** se incluyen, se presenta el mayor incremento total de la pureza del nodo que resulta de divisiones sobre la variable, promediada sobre todos los árboles.

```
varImpPlot(bag_carseats, col = '#43CD80', main = 'Modelo bagged')
```



Los resultados indican que en todos los árboles considerados en el **bagging**, la calidad de ubicación (**ShelveLoc**) y el precio de las silla de bebé(**Price**), son las dos variables más importantes.

2.5. e) Bosque aleatorio

Ahora se utiliza un bosque aleatorio (Random-Forest) para analizar estos datos Carseats y predecir las ventas.

```
set.seed(6270)
rf_car <- randomForest(Sales ~ ., data = train_car, mtry = 4, importance=TRUE)
rf_car
```

```
##
## Call:
##  randomForest(formula = Sales ~ ., data = train_car, mtry = 4,      importance = TRUE
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##           Mean of squared residuals: 2.776799
##                     % Var explained: 65.09
```

Se utilizaron 500 arboles para construir el bosque aleatorio, cada uno con 5 variables. Ahora se el MSE de prueba con el método **random forest**:

```
# MSe con el método random forest
yhat.rf<- predict(rf_car, newdata = test_car)
mean((yhat.rf - test_car$Sales)^2)
```

```
## [1] 2.44811
```

- Se observa que con el método **random forest** se obtiene un MSE de 2.44811, un poco superior al obtenido con el método **bagging** pero menor al obtenido de los árboles anteriores con y sin la poda.

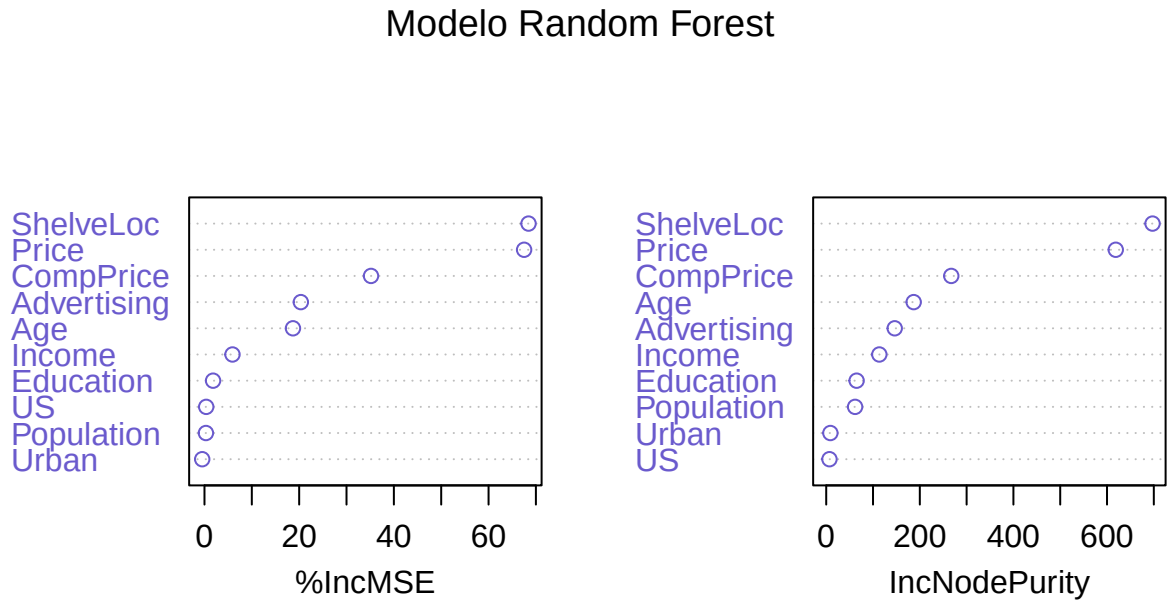
Utilizando la función "importance()", se indaga por la importancia de cada variable:

```
importance(rf_car)
```

```
##                %IncMSE IncNodePurity
## CompPrice   21.1920671     237.12205
## Income       5.3036663     148.36651
## Advertising 18.6790761     176.48525
## Population  -0.2717863     105.93086
## Price       43.5117901     517.15672
## ShelveLoc   52.8933891     573.75583
## Age         17.7737416     251.35542
## Education    1.7547294      92.02680
## Urban        2.0481489      16.07192
## US           4.0258450      18.06178
```

- Cuando las variables **ShelveLoc** y **Price** se excluyen, se presenta la mayor disminución media en la precisión de las ventas, en las muestras **bagging**, estas medidas son de 52.89 y 43.51, respectivamente.
- Cuando las variables **ShelveLoc** y **Price** se incluyen, se presenta el mayor incremento total de la pureza del nodo que resulta de divisiones sobre la variable, promediada sobre todos los árboles.

```
varImpPlot(bag_carseats, col = '#6A5ACD', main = 'Modelo Random Forest')
```



Los resultados indican que en todos los árboles considerados en el **Random Forest**, la calidad de ubicación (**ShelveLoc**) y el precio de las silla de bebé(**Price**), son las dos variables más importantes.

Describa el efecto de m el número de variables consideradas en cada subdivisión, en la tasa de error obtenida.

Para la construcción del modelo random forest se utilizó m=4, que corresponde a la función techo de 10/3, y se obtiene un error de prueba muy similar al utilizado con bagged, pero con este se garantiza que hay una menor correlación entre árboles.

Se considera m=5, para ver si este mejora las estimaciones en el conjunto de prueba.

```
set.seed(6270)
rf_car2 <- randomForest(Sales ~ ., data = train_car, mtry = 5, importance=TRUE)
rf_car2
```

```
##
## Call:
##  randomForest(formula = Sales ~ ., data = train_car, mtry = 5,      importance = TRUE
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 2.629496
##                     % Var explained: 66.94
```

Luego se cálcula su respectivo MSE de prueba:

```
# MSe con el método random forest
yhat.rf2<- predict(rf_car2, newdata = test_car)
mean((yhat.rf2 - test_car$Sales)^2)
```

```
## [1] 2.346424
```

Cuando se utiliza m=5, la tasa de error de prueba disminuye con respecto a bagged, hay menor correlación entre los árboles debido a que no se consideran todos los predictores y por lo tanto se puede disminuir la varianza.