Seleccion de modelo Pt 2

Fernando Anorve

3/25/2020

¿Cómo elegir “sabiamente” nuestro modelo?

Supongamos que tenemos una respuesta $\mathbf{Y}$ y potenciales predictores $X_1, X_2, \ldots, X_p$. Nuestro objetivo es que nuestro modelo sea lo más simple posible (i.e. que utilice la menor cantidad de predictores), sin que se ponga entredicho buen ajuste a los datos.

¿Cuál es el universo completo de modelos que se pueden ajustar? 2^p

Selección de subconjuntos de predictores

Algunas técnicas que se utilizan para elegir modelo incluyen elegir un subconjunto apropiado de los p posibles predictores. Dichas técnicas incluyen:

- Selección Exhaustiva
- Selección Forward
- Selección Backwward

Aunque dichos métodos están algo desactualizados, son sencillos de entender y podemos mostrar algunos ejemplos.

Posteriormente cubriremos métodos más interesantes y de uso más actual, como *Ridge* o *Lasso*

Conjuntos de datos

Consideremos el conjunto Hitters de la librería ISLR, que contiene datos de las ligas mayores de baseball de las temporadas de 1986 y 1987. El propósito de este ejemplo es tratar de predecir el salario de un jugador de baseball basado en varias estadísticas asociadas con el performance del año anterior.

Podemos eliminar las observaciones que tienen NA en el campo de salario

```
sum(is.na(Hitters$Salary))
```

```
## [1] 59
```

```
Hitters = na.omit(Hitters)
```

Vemos que hay 19 posibles predictores que pueden entrar en el modelo

```
names(Hitters)
```

```
##  [1] "AtBat"     "Hits"      "HmRun"     "Runs"      "RBI"
##  [6] "Walks"     "Years"     "CAtBat"    "CHits"     "CHmRun"
## [11] "CRuns"     "CRBI"      "CWalks"    "League"    "Division"
## [16] "PutOuts"   "Assists"   "Errors"    "Salary"    "NewLeague"
```

```
dim(Hitters)
```

```
## [1] 263  20
```

Nuestro objetivo: Hallar un modelo menos complejo que el que usa los 19 predictores, que tenga un ajuste "decente".

Selección exhaustiva

El primer método de selección funciona por fuerza bruta. Como habíamos mencionado, cuando se cuenta con un número de predictores p relativamente bajo, podemos darnos el lujo de evaluar los $2^p - 1$ posibles modelos, y obtener de ellos los "mejores" modelos.

El algoritmo funciona como sigue:

1. Sea $\mathcal{M}_0$ el modelo naíf $\mathbb{E}[Y] = \beta_0$
2. Para $k = 1, 2, \ldots, p$:
    a. Ajustar todos los modelos que contienen exactamente k predictores. En total son $\binom{p}{k}$ posibles modelos.
    b. Entre los $\binom{p}{k}$ posibles modelos, se ecoge el que tenga el RSS más pequeño (a esto nos referimos con el "mejor" modelo)
3. Entre los modelos $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$, se selecciona el mejor modelo en términos del método de evaluación de nuestra preferencia: C_p, AIC, BIC, o R^2 ajustada

¿Cómo utilizar este modelo en R?

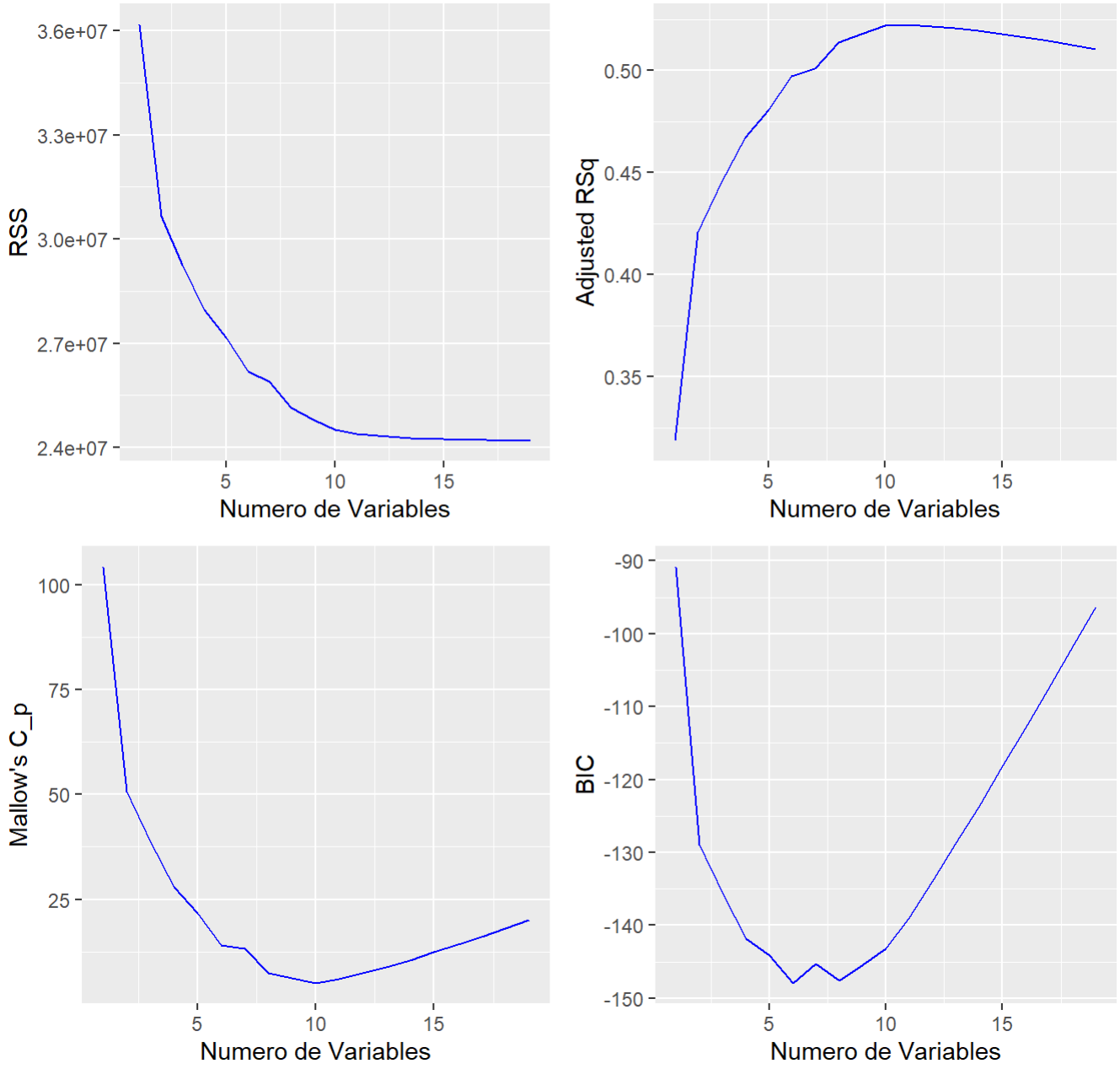
Librería leaps

```
library(leaps)
n = 19

regfit.full = regsubsets(Salary ~ . , data = Hitters, nvmax = n, method = "exhaustive")

reg.summary = summary(regfit.full)
```

¿Cómo podemos interpretar los resultados?

La función ya regresa algunos estadísticos que podemos utilizar para evaluar el modelo
$RSS, R^2, R^2_{adj}, C_p, BIC$

```
rss_plt <- ggplot(data = NULL, aes(x = 1:n,y = reg.summary$rss)) +
  geom_line(color = "blue") + xlab("Numero de Variables") +  ylab("RSS")

adjr2_plt <- ggplot(data = NULL, aes(x = 1:n,y = reg.summary$adjr2)) + geom_line(color = "blue")
+ xlab("Numero de Variables") +
  ylab("Adjusted RSq")

cp_plt <- ggplot(data = NULL, aes(x = 1:n,y = reg.summary$cp)) + geom_line(color = "blue") + xla
b("Numero de Variables") +
  ylab("Mallow's C_p")

bic_plt <- ggplot(data = NULL, aes(x = 1:n,y = reg.summary$bic)) + geom_line(color = "blue") + x
lab("Numero de Variables") +
  ylab("BIC")

grid.arrange(rss_plt,adjr2_plt,cp_plt,bic_plt, ncol = 2)
```



Usualmente nos interesan valores de **BIC** y de C_p bajos. Particularmente, en el segundo caso también deseamos que $C_p \approx p$.

¿Dónde alcanza el mínimo BIC?

```
which.min(reg.summary$bic)
```

```
## [1] 6
```

¿Y el máximo R_{Adj}?

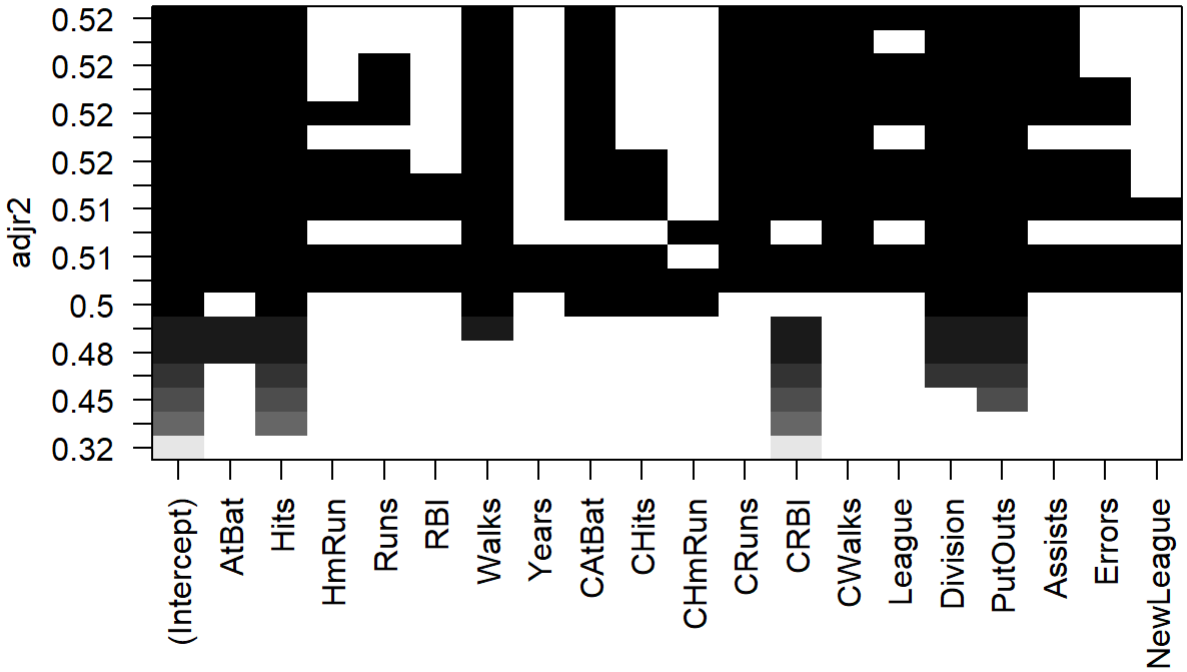
```
which.max(reg.summary$adjr2)
```

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

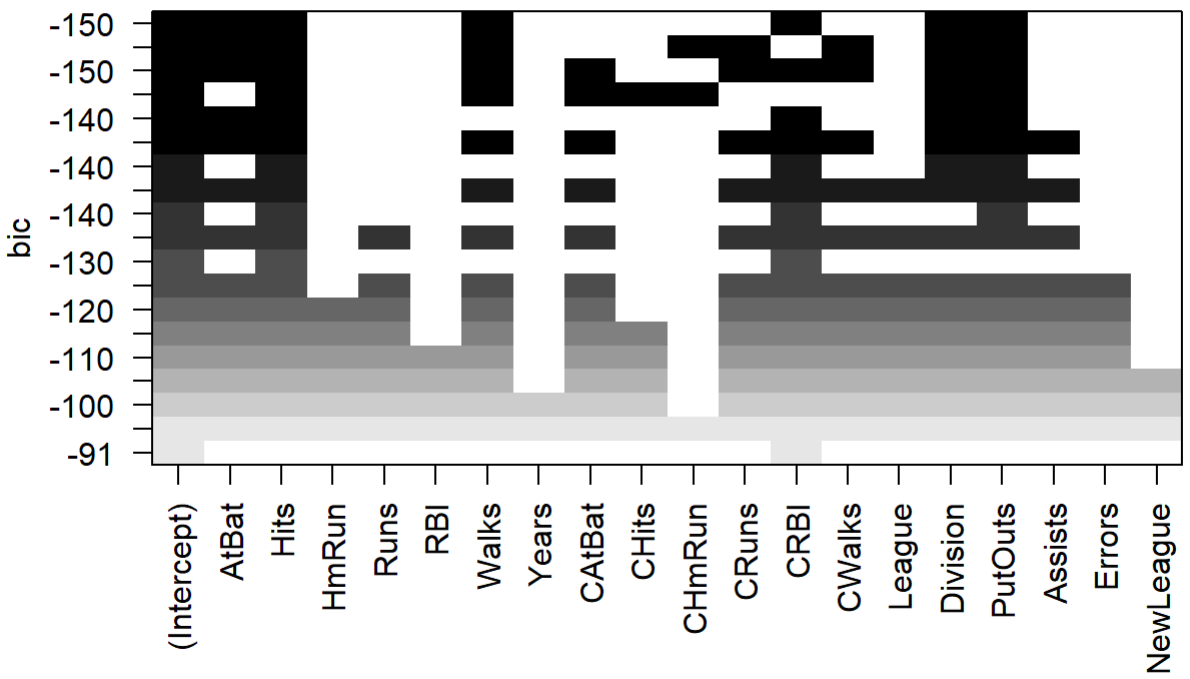
¿Cómo saber qué coeficientes se usaron?

Primera forma

```
plot( regfit.full , scale ="adjr2")
```




```
plot( regfit.full , scale ="bic")
```



Segunda forma:

```
coef(regfit.full,6)
```

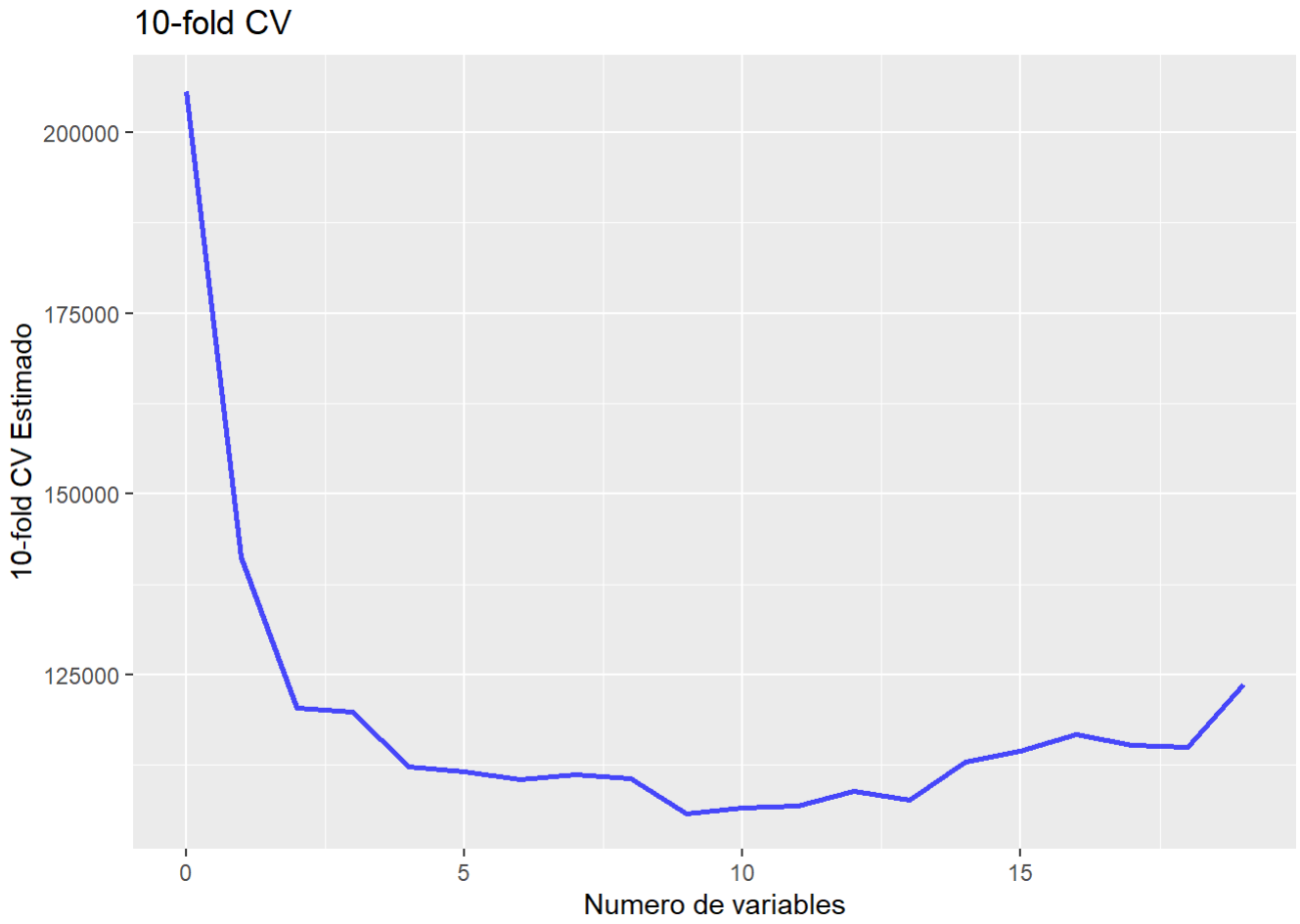
```
##  (Intercept)        AtBat         Hits        Walks         CRBI
##   91.5117981   -1.8685892    7.6043976    3.6976468    0.6430169
##     Division      PutOuts
## -122.9515338    0.2643076
```

Nota:

Los criterios anteriores son in-sample. Es decir, se basan completamente en los datos sobre los que fue entrenado el modelo. No sabemos mucho sobre cómo se comportan los modelos cuando incorporamos información nueva. Para ello podemos compararlos utilizando validación cruzada

Validación cruzada

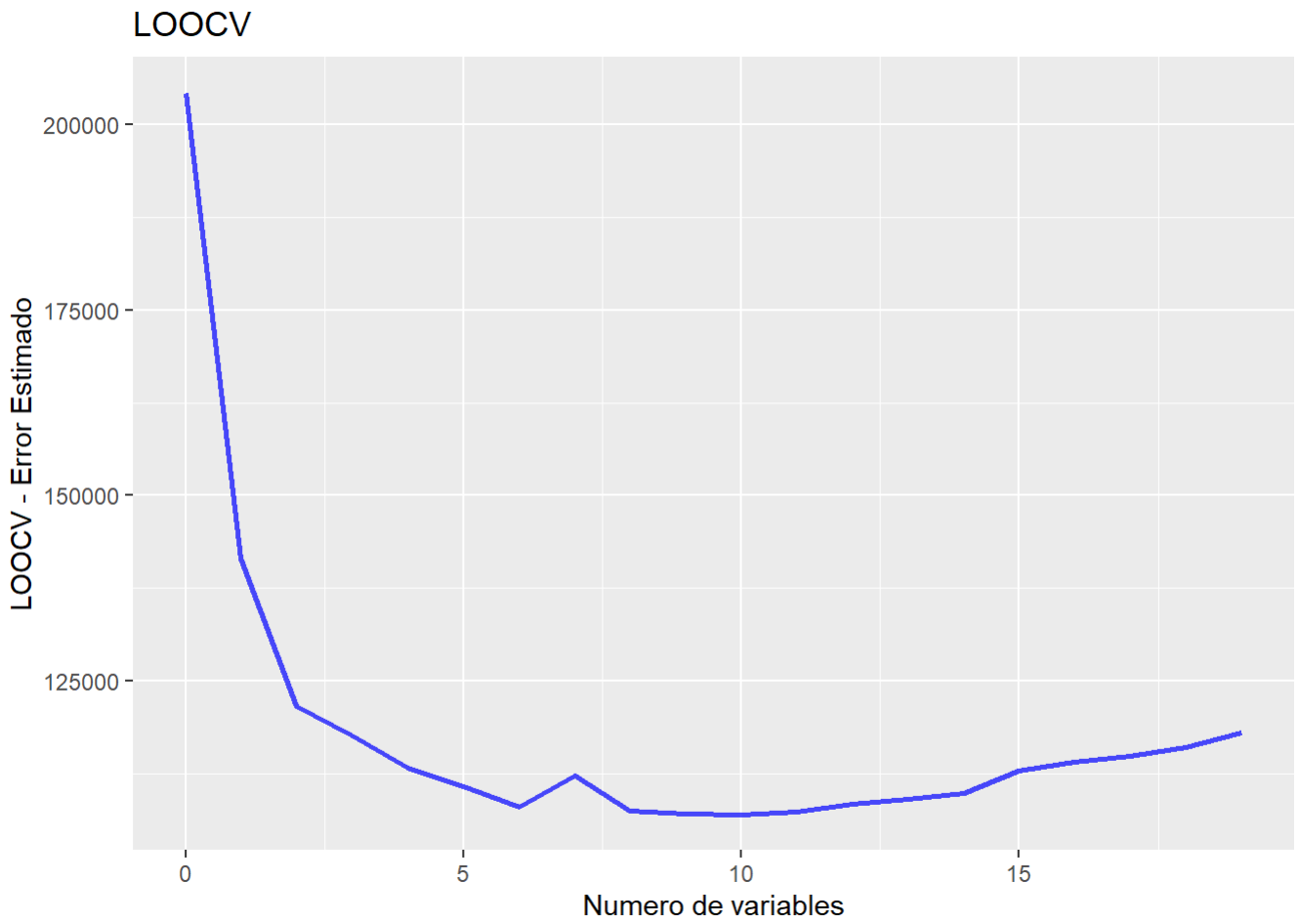
```
set.seed(2020)

cv_error <- rep(0,n)

for (i in 1:n) {
    coeffs <- coef(regfit.full,i)
    covars <- names(coeffs)[-1]

    fmla <- as.formula(paste("Salary ~ ", paste(covars, collapse= "+")))

    glm.fit = glm (fmla , data = Hitters )
    cv_error[i]= cv.glm( Hitters , glm.fit ,K =10)$delta[1]
}

glm.fit = glm (Salary ~ 1 , data = Hitters )
cv_error = c(cv.glm( Hitters , glm.fit ,K =10)$delta[1],cv_error)

ggplot(data = NULL, aes(x = 0:n,y = cv_error)) + geom_line(colour = "blue", size = 1, alpha = 0.
7)  + xlab("Numero de variables") + ylab("10-fold CV Estimado") + ggtitle("10-fold CV") + ggtitl
e("10-fold CV")
```



LOOCV

Como ya se había mencionado, el leave-one-out puede ser computacionalmente costoso, por lo que sólo dejaré el código y cargaré el resultado preguardado

```
set.seed(2019)

loocv_error <- rep(0,n)
#
# for (i in 1:n) {
#    coeffs <- coef(regfit.full,i)
#    covars <- names(coeffs)[-1]
#
#    fmla <- as.formula(paste("Salary ~ ", paste(covars, collapse= "+")))
#
#    glm.fit = glm (fmla , data = Hitters )
#    loocv_error[i]= cv.glm( Hitters , glm.fit)$delta[1]
# }
#
# glm.fit = glm (Salary ~ 1 , data = Hitters )
# loocv_error = c(cv.glm( Hitters , glm.fit)$delta[1],loocv_error)
#
# save(loocv_error,file = "loocv.RData")

load("loocv.RData")

ggplot(data = NULL, aes(x = 0:n,y = loocv_error)) +
  geom_line(colour = "blue", size = 1, alpha = 0.7)  +
  xlab("Numero de variables") + ylab("LOOCV - Error Estimado") +
  ggtitle("LOOCV")
```



En algún lugar cercano a 5, la estimación del error deja de disminuir de manera importante. No hay evidencia de que se requieran más de cinco parámetros.

Desventajas

Evidentemente, evaluar exhaustivamente 2^p modelos no es factible en términos computacionales. Cuando p = 20, hay 1048576 posibles modelos que evaluar.

Selección forward

La selección forward considera un universo mucho más pequeño que el método exhaustivo.

- El modelo más simple es el modelo naíf $\mathbb{E}[Y] = \beta_0$.
- A este modelo se le pueden agregar predictores de uno por uno, hasta que todos los predictores estén dentro del modelo.
- En cada paso, se agrega el predictor que represente la mayor mejora adicional (i.e. el que más incremente R^2)
- En la k-ésima iteración se evalúan $p - k$ modelos. En total, los modelos a evaluar suman $1 + p(p+1)/2$ modelos, un diferencia sustancial a 2^p

El algoritmo funciona como sigue:

1. Sea $\mathcal{M}_0$ el modelo naíf $\mathbb{E}[Y] = \beta_0$
2. Para $k = 0, 1, \ldots, p-1$:

    a. Consideramos los $p-k$ modelos que resultan de agregar un predictor de los que no están en $\mathcal{M}_k$
    b. Entre los $p-k$ modelos, escoger el que resulte en la menor suma de cuadrados residuales RSS
3. Entre los modelos $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$, se selecciona el mejor modelo en términos del método de evaluación de nuestra preferencia: C_p, AIC, BIC, o R^2 ajustada

```
regfit.fwd = regsubsets( Salary ~ . , data = Hitters , nvmax =19 , method = "forward")

fwd.summary = summary(regfit.fwd)
```

Con fines ilustrativos, veamos qué hace en los primeros dos pasos

- ¿Cuál es el modelo con dos variables?

```
coef(regfit.fwd,2)
```

```
## (Intercept)        Hits        CRBI
## -47.9559022   3.3008446   0.6898994
```

- Queremos llegar del modelo naíf al modelo completo, pero sólo necesitamos ver los primeros dos pasos

```
fit0 <- lm(Salary ~ 1,data = Hitters)
fit_full <- lm(Salary ~ . ,data = Hitters)

step(fit0,scope=list(lower=fit0,upper=fit_full),direction="forward", steps = 2)
```

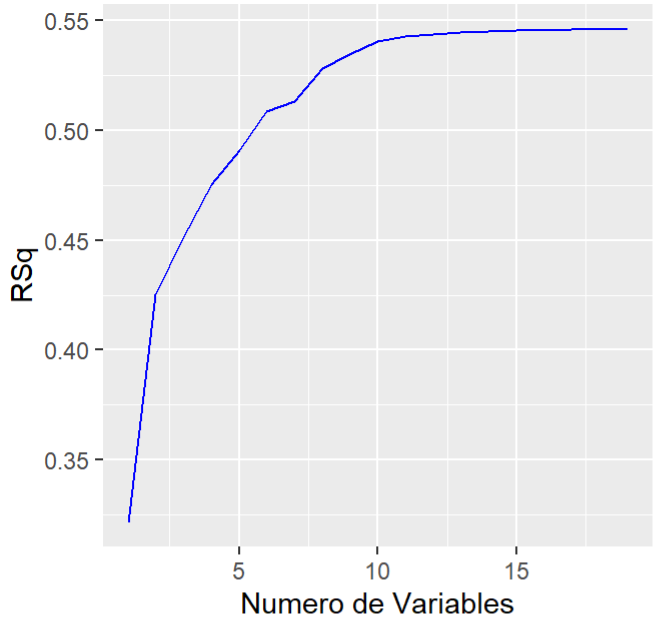
```
## Start:  AIC=3215.77
## Salary ~ 1
##
##             Df Sum of Sq      RSS    AIC
## + CRBI       1  17139434 36179679 3115.8
## + CRuns      1  16881162 36437951 3117.6
## + CHits      1  16065140 37253973 3123.5
## + CAtBat     1  14759710 38559403 3132.5
## + CHmRun     1  14692193 38626920 3133.0
## + CWalks     1  12792622 40526491 3145.6
## + RBI        1  10771083 42548030 3158.4
## + Walks      1  10504833 42814280 3160.1
## + Hits       1  10260491 43058621 3161.6
## + Runs       1   9399158 43919955 3166.8
## + Years      1   8559105 44760007 3171.7
## + AtBat      1   8309469 45009644 3173.2
## + HmRun      1   6273967 47045145 3184.8
## + PutOuts    1   4814100 48505013 3192.9
## + Division   1   1976102 51343011 3207.8
## <none>                   53319113 3215.8
## + Assists    1     34497 53284615 3217.6
## + League     1     10876 53308237 3217.7
## + Errors     1      1555 53317558 3217.8
## + NewLeague  1       428 53318684 3217.8
##
## Step:  AIC=3115.78
## Salary ~ CRBI
##
##             Df Sum of Sq      RSS    AIC
## + Hits       1   5533119 30646560 3074.1
## + Runs       1   5176532 31003147 3077.2
## + Walks      1   4199733 31979946 3085.3
## + AtBat      1   4064585 32115095 3086.4
## + RBI        1   3308272 32871407 3092.6
## + PutOuts    1   3267035 32912644 3092.9
## + Division   1   1733887 34445793 3104.9
## + Years      1   1667339 34512340 3105.4
## + HmRun      1   1271587 34908092 3108.4
## + CRuns      1    354561 35825119 3115.2
## + Assists    1    346020 35833659 3115.2
## <none>                   36179679 3115.8
## + Errors     1    194403 35985276 3116.4
## + CAtBat     1     92261 36087418 3117.1
## + CHits      1     75469 36104210 3117.2
## + CWalks     1     51974 36127705 3117.4
## + NewLeague  1     17778 36161901 3117.7
## + League     1     11825 36167855 3117.7
## + CHmRun     1       515 36179165 3117.8
##
## Step:  AIC=3074.13
## Salary ~ CRBI + Hits
```

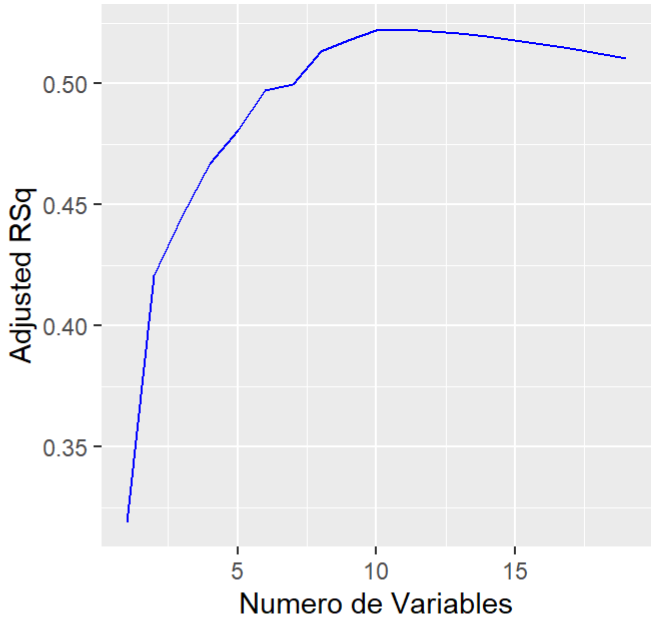
```
##
## Call:
## lm(formula = Salary ~ CRBI + Hits, data = Hitters)
##
## Coefficients:
## (Intercept)         CRBI         Hits
##    -47.9559       0.6899       3.3008
```

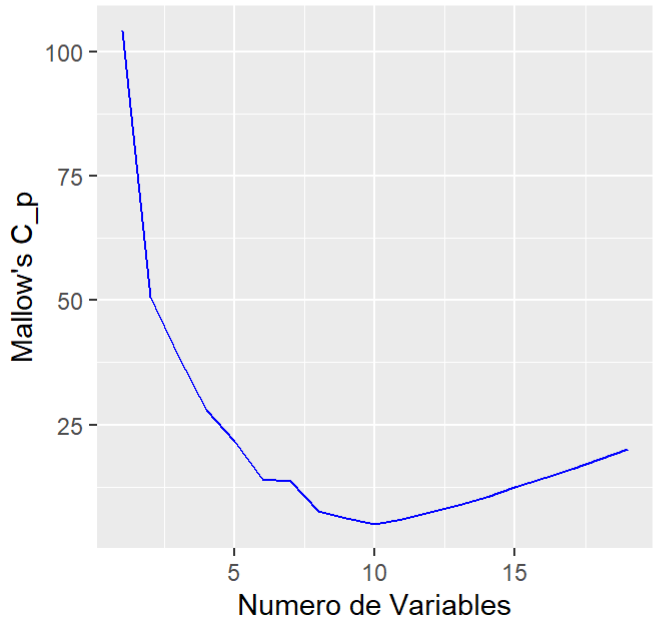
Nota: La función “step” no selecciona modelos con base en el RSS, pero la usé de todas formas para mostrar cómo funciona a grandes rasgos el forward

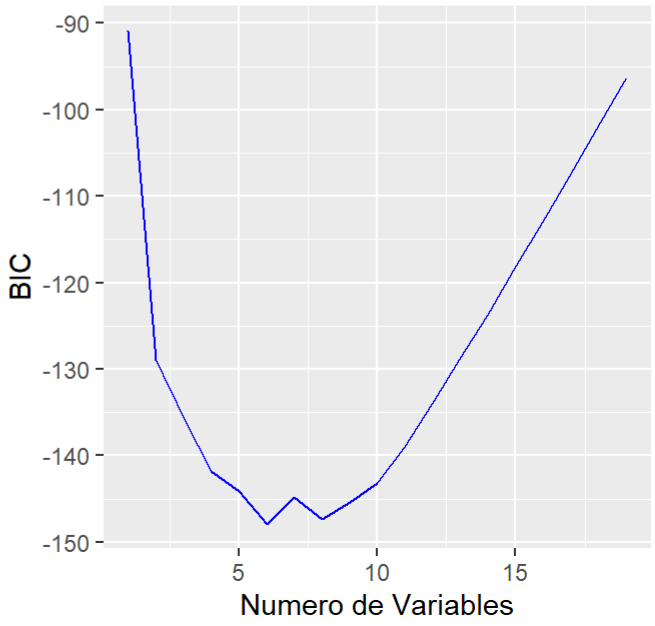
Ahora sí, a interpretar resultados

```
rss_plt <- ggplot(data = NULL, aes(x = 1:n,y = fwd.summary$rsq)) +
  geom_line(color = "blue") + xlab("Numero de Variables") +  ylab("RSq")

adjr2_plt <- ggplot(data = NULL, aes(x = 1:n,y = fwd.summary$adjr2)) + geom_line(color = "blue")
+ xlab("Numero de Variables") +
  ylab("Adjusted RSq")

cp_plt <- ggplot(data = NULL, aes(x = 1:n,y = fwd.summary$cp)) + geom_line(color = "blue") + xla
b("Numero de Variables") +
  ylab("Mallow's C_p")

bic_plt <- ggplot(data = NULL, aes(x = 1:n,y = fwd.summary$bic)) + geom_line(color = "blue") + x
lab("Numero de Variables") +
  ylab("BIC")

grid.arrange(rss_plt,adjr2_plt,cp_plt,bic_plt, ncol = 2)
```


RSq
0.55
0.50
0.45
0.40
0.35
5
10
15
Numero de Variables


Adjusted RSq
0.50
0.45
0.40
0.35
5
10
15
Numero de Variables


Mallow's C_p
100
75
50
25
5
10
15
Numero de Variables

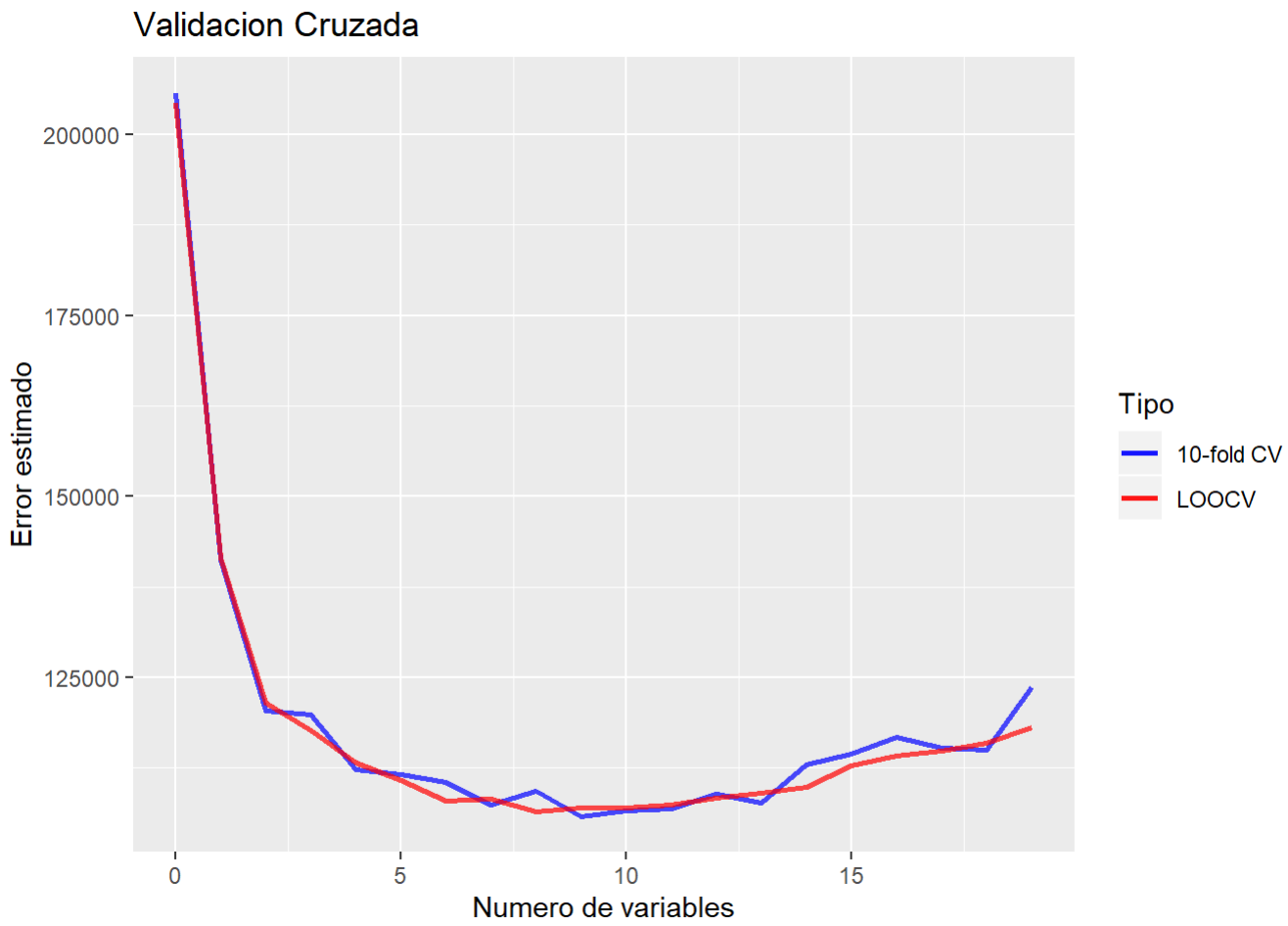

BIC
-90
-100
-110
-120
-130
-140
-150
5
10
15
Numero de Variables

Validación cruzada y LOOCV

```
set.seed(2020)

cv_error <- rep(0,n)

for (i in 1:n) {
    coeffs <- coef(regfit.fwd,i)
    covars <- names(coeffs)[-1]

    fmla <- as.formula(paste("Salary ~ ", paste(covars, collapse= "+")))

    glm.fit = glm (fmla , data = Hitters )
    cv_error[i]= cv.glm( Hitters , glm.fit ,K =10)$delta[1]
}

glm.fit = glm (Salary ~ 1 , data = Hitters )
cv_error = c(cv.glm( Hitters , glm.fit ,K =10)$delta[1],cv_error)
```

De nuevo sólo dejaré el código y cargaré el resultado preguardado

```
set.seed(2019)

loocv_error <- rep(0,n)

# for (i in 1:n) {
#    coeffs <- coef(regfit.fwd,i)
#    covars <- names(coeffs)[-1]
#
#    fmla <- as.formula(paste("Salary ~ ", paste(covars, collapse= "+")))
#
#    glm.fit = glm (fmla , data = Hitters )
#    loocv_error[i]= cv.glm( Hitters , glm.fit)$delta[1]
# }
#
# glm.fit = glm (Salary ~ 1 , data = Hitters )
# loocv_error = c(cv.glm( Hitters , glm.fit)$delta[1],loocv_error)
#
# save(loocv_error,file = "loocv_fwd.RData")

load("loocv_fwd.RData")

ggplot(data = NULL, aes(x = 0:n)) +
  ggtitle("Validacion Cruzada") +
  geom_line( size = 1, alpha = 0.7, mapping = aes(y = cv_error, color = "10-fold CV"))  +
  geom_line( size = 1, alpha = 0.7, mapping = aes(y = loocv_error, color = "LOOCV")) +
  scale_color_manual(values = c("10-fold CV" = "blue","LOOCV" = "red")) +
  labs(y = "Error estimado", x = "Numero de variables" , color = "Tipo")
```



Si elegimos usar 5 predictores, ¿cuáles serían sus coeficientes según el algoritmo forward?

```
coef(regfit.fwd,5)
```

```
##  (Intercept)        AtBat         Hits         CRBI     Division
##   97.7684116   -1.4401428    7.1753197    0.6882079 -129.7319386
##      PutOuts
##    0.2905164
```

El modelo ya ajustado sería:

```
coeffs <- coef(regfit.fwd,5)
covars <- names(coeffs)[-1]

fmla <- as.formula(paste("Salary ~ ", paste(covars, collapse= "+")))
fit5 <- lm(fmla,data = Hitters)

summary(fit5)
```

```
##
## Call:
## lm(formula = fmla, data = Hitters)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -743.56 -167.91  -39.03  132.40 2023.71
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   97.76841   66.01266   1.481  0.13982
## AtBat         -1.44014    0.51661  -2.788  0.00571 **
## Hits           7.17532    1.68324   4.263 2.84e-05 ***
## CRBI           0.68821    0.06372  10.800  < 2e-16 ***
## DivisionW   -129.73194   40.39782  -3.211  0.00149 **
## PutOuts        0.29052    0.07547   3.849  0.00015 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 325 on 257 degrees of freedom
## Multiple R-squared:  0.4908, Adjusted R-squared:  0.4809
## F-statistic: 49.54 on 5 and 257 DF,  p-value: < 2.2e-16
```

Revisando de nuevo las primeras gráficas, notamos que todavía se puede aumentar R^2 un poco más al incluir más variables. No obstante, recordemos que un R^2 alto no implica necesariamente poder predictivo, y la evidencia basada en validación cruzada no apunta a que incluir más predictores mejore notoriamente la capacidad predictiva del modelo.

Selección Backward

La selección backward considera también un universo más pequeño que el método exhaustivo.

- Consideramos primero el modelo que usa los p predictores
- A este modelo se le pueden quitar predictores de uno por uno, hasta obtener el modelo naif.
- En cada paso, se elimina el predictor que menos información adicional aporte.
- En la k-ésima iteración se evalúan k modelos. En total, los modelos a evaluar suman $1 + p(p+1)/2$ modelos

El algoritmo es el siguiente:

1. Sea $\mathcal{M}_p$ el modelo completo, i.e. que usa todos los predictores
2. Para $k = p, p-1, \ldots, 1$:
    a. Consideramos los k modelos que resultan de quitar algún un predictor de los que están en $\mathcal{M}_k$ y usar los $k-1$ restantes.
    b. Entre los k modelos, escoger el que resulte en la menor suma de cuadrados residuales RSS
3. Entre los modelos $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$, se selecciona el mejor modelo en términos del método de evaluación de nuestra preferencia: C_p, AIC, BIC, o R^2 ajustada

```
regfit.bkw = regsubsets( Salary ~ . , data = Hitters , nvmax =19 , method = "backward")

bkw.summary = summary(regfit.bkw)
```

Con fines ilustrativos, veamos qué hace en los primeros dos pasos

- Queremos llegar del modelo completo al modelo naíf, pero sólo necesitamos ver los primeros dos pasos

```
fit0 <- lm(Salary ~ 1,data = Hitters)
fit_full <- lm(Salary ~ . ,data = Hitters)

step(fit_full,scope=list(lower=fit0,upper=fit_full),direction="backward", steps = 2)
```

```
## Start:  AIC=3046.02
## Salary ~ AtBat + Hits + HmRun + Runs + RBI + Walks + Years +
##     CAtBat + CHits + CHmRun + CRuns + CRBI + CWalks + League +
##     Division + PutOuts + Assists + Errors + NewLeague
##
##             Df Sum of Sq      RSS    AIC
## - CHmRun     1      1138 24201837 3044.0
## - CHits      1      3930 24204629 3044.1
## - Years      1      7869 24208569 3044.1
## - NewLeague  1      9784 24210484 3044.1
## - RBI        1     16076 24216776 3044.2
## - HmRun      1     48572 24249272 3044.6
## - Errors     1     58324 24259023 3044.7
## - League     1     62121 24262821 3044.7
## - Runs       1     63291 24263990 3044.7
## - CRBI       1    135439 24336138 3045.5
## - CAtBat     1    159864 24360564 3045.8
## <none>                   24200700 3046.0
## - Assists    1    280263 24480963 3047.1
## - CRuns      1    374007 24574707 3048.1
## - CWalks     1    609408 24810108 3050.6
## - Division   1    834491 25035190 3052.9
## - AtBat      1    971288 25171987 3054.4
## - Hits       1    991242 25191941 3054.6
## - Walks      1   1156606 25357305 3056.3
## - PutOuts    1   1319628 25520328 3058.0
##
## Step:  AIC=3044.03
## Salary ~ AtBat + Hits + HmRun + Runs + RBI + Walks + Years +
##     CAtBat + CHits + CRuns + CRBI + CWalks + League + Division +
##     PutOuts + Assists + Errors + NewLeague
##
##             Df Sum of Sq      RSS    AIC
## - Years      1      7609 24209447 3042.1
## - NewLeague  1     10268 24212106 3042.2
## - CHits      1     14003 24215840 3042.2
## - RBI        1     14955 24216793 3042.2
## - HmRun      1     52777 24254614 3042.6
## - Errors     1     59530 24261367 3042.7
## - League     1     63407 24265244 3042.7
## - Runs       1     64860 24266698 3042.7
## - CAtBat     1    174992 24376830 3043.9
## <none>                   24201837 3044.0
## - Assists    1    285766 24487603 3045.1
## - CRuns      1    611358 24813196 3048.6
## - CWalks     1    645627 24847464 3049.0
## - Division   1    834637 25036474 3050.9
## - CRBI       1    864220 25066057 3051.3
## - AtBat      1    970861 25172699 3052.4
## - Hits       1   1025981 25227819 3052.9
## - Walks      1   1167378 25369216 3054.4
## - PutOuts    1   1325273 25527110 3056.1
```

```
##
## Step:  AIC=3042.12
## Salary ~ AtBat + Hits + HmRun + Runs + RBI + Walks + CAtBat +
##     CHits + CRuns + CRBI + CWalks + League + Division + PutOuts +
##     Assists + Errors + NewLeague
```

```
##
## Call:
## lm(formula = Salary ~ AtBat + Hits + HmRun + Runs + RBI + Walks +
##     CAtBat + CHits + CRuns + CRBI + CWalks + League + Division +
##     PutOuts + Assists + Errors + NewLeague, data = Hitters)
##
## Coefficients:
## (Intercept)        AtBat         Hits        HmRun         Runs
##    148.4333      -1.9509       7.3914       4.0828      -2.2397
##         RBI        Walks       CAtBat        CHits        CRuns
##     -0.9940       6.1971      -0.1913       0.2067       1.4250
##        CRBI       CWalks      LeagueN    DivisionW      PutOuts
##      0.7415      -0.8038      64.1928    -116.0618       0.2830
##     Assists       Errors   NewLeagueN
##      0.3773      -3.3200     -24.8892
```

- ¿Cuál es el modelo con 17 variables? (sólo para verificar que quedó lo mismo)

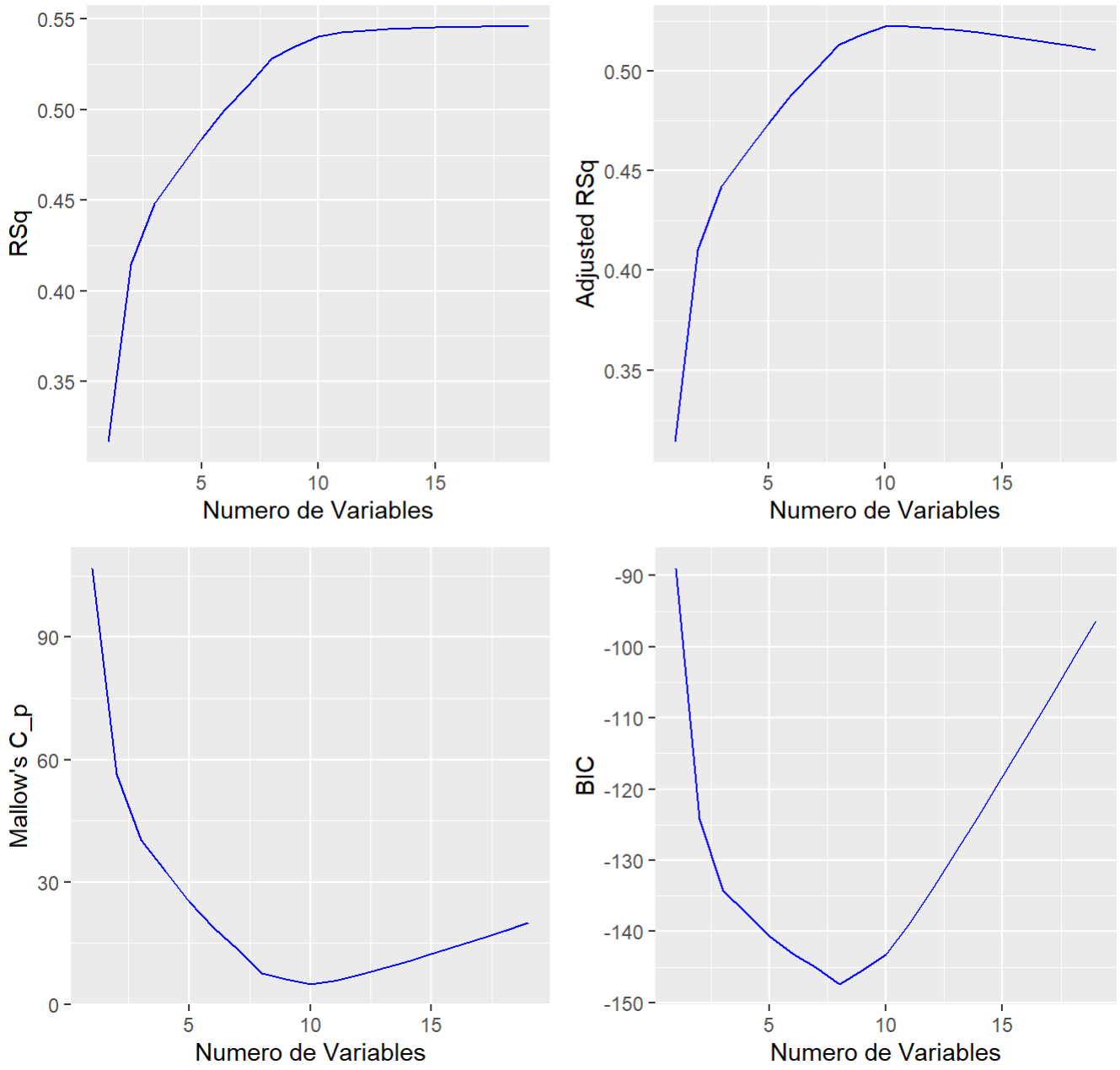
```
coef(regfit.bkw,17)
```

```
##  (Intercept)        AtBat         Hits        HmRun         Runs
##  148.4333315   -1.9509056    7.3914057    4.0827974   -2.2396702
##          RBI        Walks       CAtBat        CHits        CRuns
##   -0.9940157    6.1970575   -0.1913281    0.2067327    1.4249652
##         CRBI       CWalks       League     Division      PutOuts
##    0.7414741   -0.8037607   64.1928201 -116.0617552    0.2830325
##      Assists       Errors    NewLeague
##    0.3773191   -3.3199889  -24.8892236
```

Regresando al resultado de backward

Volvemos a ver las gráficas de los criterios in-sample

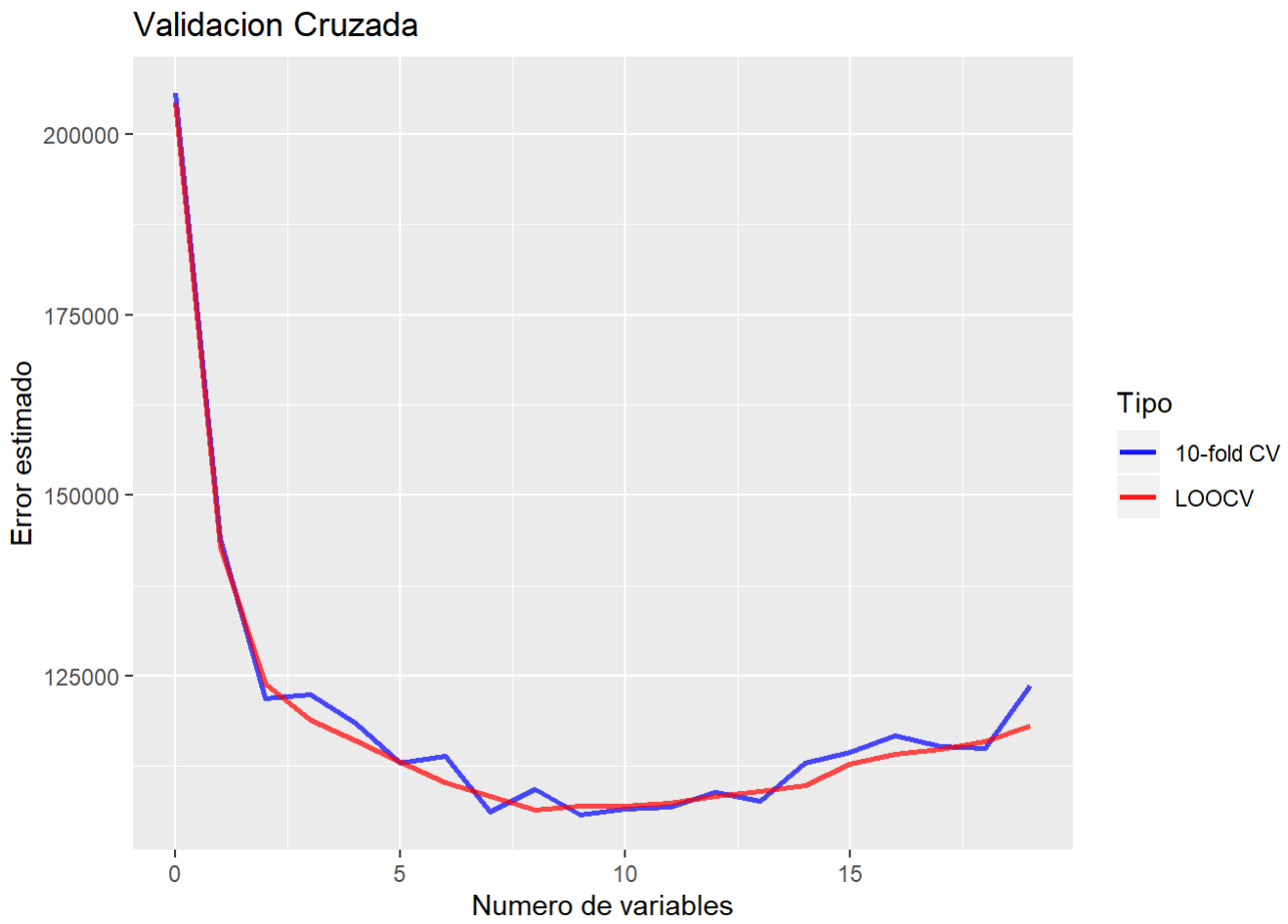
```
rss_plt <- ggplot(data = NULL, aes(x = 1:n,y = bkw.summary$rsq)) +
  geom_line(color = "blue") + xlab("Numero de Variables") +  ylab("RSq")

adjr2_plt <- ggplot(data = NULL, aes(x = 1:n,y = bkw.summary$adjr2)) + geom_line(color = "blue")
+ xlab("Numero de Variables") +
  ylab("Adjusted RSq")

cp_plt <- ggplot(data = NULL, aes(x = 1:n,y = bkw.summary$cp)) + geom_line(color = "blue") + xla
b("Numero de Variables") +
  ylab("Mallow's C_p")

bic_plt <- ggplot(data = NULL, aes(x = 1:n,y = bkw.summary$bic)) + geom_line(color = "blue") + x
lab("Numero de Variables") +
  ylab("BIC")

grid.arrange(rss_plt,adjr2_plt,cp_plt,bic_plt, ncol = 2)
```



Notemos que los estadísticos vuelven a indicar valores cercanos a 10. De nuevo, con únicamente estos gráficos sabemos poco sobre la capacidad predictiva.

Validación cruzada y LOOCV

```
set.seed(2020)

cv_error <- rep(0,n)

for (i in 1:n) {
    coeffs <- coef(regfit.bkw,i)
    covars <- names(coeffs)[-1]

    fmla <- as.formula(paste("Salary ~ ", paste(covars, collapse= "+")))

    glm.fit = glm (fmla , data = Hitters )
    cv_error[i]= cv.glm( Hitters , glm.fit ,K =10)$delta[1]
}

glm.fit = glm (Salary ~ 1 , data = Hitters )
cv_error = c(cv.glm( Hitters , glm.fit ,K =10)$delta[1],cv_error)



loocv_error <- rep(0,n)
#
# for (i in 1:n) {
#   coeffs <- coef(regfit.bkw,i)
#   covars <- names(coeffs)[-1]
#
#   fmla <- as.formula(paste("Salary ~ ", paste(covars, collapse= "+")))
#
#   glm.fit = glm (fmla , data = Hitters )
#   loocv_error[i]= cv.glm( Hitters , glm.fit)$delta[1]
# }
#
# glm.fit = glm (Salary ~ 1 , data = Hitters )
# loocv_error = c(cv.glm( Hitters , glm.fit)$delta[1],loocv_error)
#
# save(loocv_error,file = "loocv_bkw.RData")

load("loocv_bkw.RData")

ggplot(data = NULL, aes(x = 0:n)) +
  ggtitle("Validacion Cruzada") +
  geom_line( size = 1, alpha = 0.7, mapping = aes(y = cv_error, color = "10-fold CV"))  +
  geom_line( size = 1, alpha = 0.7, mapping = aes(y = loocv_error, color = "LOOCV")) +
  scale_color_manual(values = c("10-fold CV" = "blue","LOOCV" = "red")) +
  labs(y = "Error estimado", x = "Numero de variables" , color = "Tipo")
```



No está de más recordar de nuevo que un R^2 alto no implica necesariamente poder predictivo.

Si elegimos usar 5 predictores, ¿cuáles serían sus coeficientes según el algoritmo backward?

```
coef(regfit.bkw,5)
```

```
## (Intercept)       AtBat        Hits       Walks       CRuns     PutOuts
##  16.3605866  -1.9686325   7.8904780   3.6851865   0.6218929   0.3000029
```

¿Son los mismos que el forward?

```
coef(regfit.fwd,5)
```

```
##  (Intercept)        AtBat         Hits         CRBI     Division
##   97.7684116   -1.4401428    7.1753197    0.6882079 -129.7319386
##      PutOuts
##    0.2905164
```

¿Y el exhaustivo?

```
coef(regfit.full,5)
```

```
##  (Intercept)        AtBat         Hits         CRBI     Division
##   97.7684116   -1.4401428    7.1753197    0.6882079 -129.7319386
##      PutOuts
##    0.2905164
```

Nota:

- Si bien el exhaustivo y el forward arrojaron el mismo resultado, esto no siempre ocurre. En general, los resultados no tienen por qué ser los mismos entre los tres métodos.

Por ejemplo, si el modelo con una variable que mayor R^2 arroja incluye la variable x_1, tanto el forward como el exhaustivo van a incorporar x_1 en el modelo $\mathcal{M}$. Sin embargo, si el modelo con dos variables que mayor R^2 arroja incluye las variables x_2 y x_3, el algoritmo exhaustivo sí las va a tomar en cuenta $\mathcal{M}_2$, mientras que el algoritmo forward va tomar en cuenta para $\mathcal{M}_2$ a x_1 más alguna otra variable.

- Cuando el número de covariables p es muy alto (por ejemplo, $p > n$), es conveniente elegir el algoritmo forward.

Finalmente, el modelo ya ajustado sería:

```
coeffs <- coef(regfit.bkw,5)
covars <- names(coeffs)[-1]

fmla <- as.formula(paste("Salary ~ ", paste(covars, collapse= "+")))
fit5 <- lm(fmla,data = Hitters)

summary(fit5)
```

```
##
## Call:
## lm(formula = fmla, data = Hitters)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -925.9 -167.0  -37.7  122.0 2115.3
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16.36059   62.91212   0.260 0.795030
## AtBat       -1.96863    0.53686  -3.667 0.000298 ***
## Hits         7.89048    1.69113   4.666 4.95e-06 ***
## Walks        3.68519    1.24272   2.965 0.003307 **
## CRuns        0.62189    0.06496   9.574  < 2e-16 ***
## PutOuts      0.30000    0.07656   3.919 0.000114 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 327.2 on 257 degrees of freedom
## Multiple R-squared:  0.4841, Adjusted R-squared:  0.474
## F-statistic: 48.22 on 5 and 257 DF,  p-value: < 2.2e-16
```

Otros métodos (Shrinkage)

En los algoritmos anteriores, hemos tratado de encontrar un subconjunto conveniente de los parámetros para simplificar nuestro modelo lineal.

Otra técnica, en cambio, incluye incluir los p predictores “restringir” o “regularizar” las estimaciones de los coeficientes (dicho de otra forma, “encogerlas” hacia cero), para reducir su varianza.

Dichas técnicas incluyen:

- Regresión Ridge
- Regresión Lasso