Centro de Investigación en Matemáticas (CIMAT). Unidad Monterrey

Cómputo Estadístico
Tarea 4

Marcelo Alberto Sanchez Zaragoza

6 de noviembre de 2021

1. Problema 1

Se ha visto que a medida que aumenta el número de características de un modelo, el error de entrenamiento disminuirá necesariamente, pero el error de prueba no. Explorar esto con datos simulados.

a) Genera un conjunto de datos con $p = 20$ características, $n = 1000$ observaciones y un vector de respuesta cuantitativo generado de acuerdo con el modelo

$$Y = X\beta + \varepsilon$$

b) Divide tu conjunto de datos en un conjunto de entrenamiento que contenga 100 observaciones y un conjunto de prueba que contenga 900 observaciones

c) Realiza la *seleccion del mejor subconjunto* sobre el conjunto de entrenamiento y grafica el error de entrenamiento MSE asociado con el mejor modelo en cada tamaño.

d) Grafica el error de prueba MSE asociado con el mejor modelo de cada tamaño.

e) ¿ Para qué tamaño de modelo el error del prueba MSE toma su valor mínimo?

f) ¿Cómo se compara el modelo con el que se minimiza el error de prueba con el modelo verdadero utilizado para generar los datos?

Solución

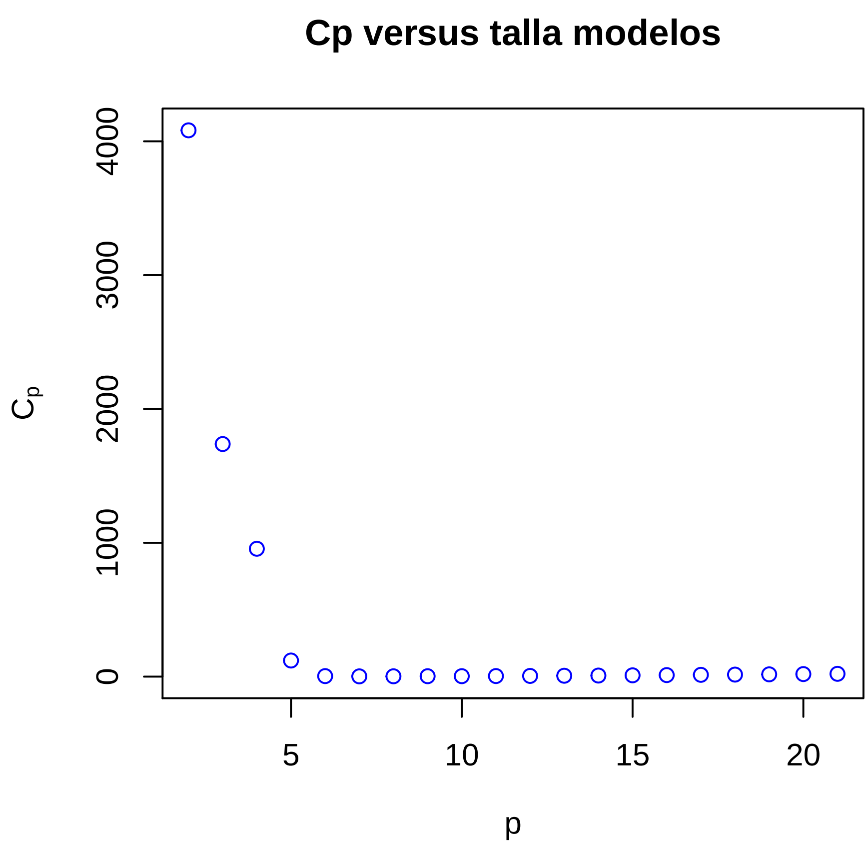
Inciso a)
Generamos nuestros datos que nos solicita el inciso, en este caso pedimos los 1000 datos de una distribución normal. Damos los valores de beta que queremos sean distintos de cero, en este caso seleccionamos los siguientes: 3,5,6,7 y 12.

```
set.seed(123457)
X <- matrix(rnorm(20000),ncol = 20)
betas <- rep(0, 20)
betas[c(3, 5, 7, 12, 6)] = 1:5
#betas[c(2)] = 5
betas

 [1] 0 0 1 0 2 5 3 0 0 0 0 4 0 0 0 0 0 0 0 0

y = X %*% betas + rnorm(1000)
#y
```

Inciso b)
Dividimos nuestros datos en un conjunto de entrenamiento con 100 datos y otro conjunto de prueba con 900 datos.

```r
set.seed(1) #fijamos la semilla para reproducir los resultados
train <- sample (1:1000, 100)# se selecciona aleatoriamente la
train

  [1] 836 679 129 930 509 471 299 270 978 187 307 597 277 874 950 494 330 775
 [19] 841 591 725  37 105 729 878 485 677 802 987 382 601 940 801 852 931 326
 [37] 984 554 422 111 404 924 532 506 556 889 343 582 121  40 684 537 375 248
 [55] 198 378  39 435 810 390 280 672 526 642  45 402  22 718 742 193 371 499
 [73] 104 965 767 492 838 616 615 843 465 525 808 977 176 345 791 110  84 871
 [91]  29 141 252 733 620 304 545 557 661 287

#mitad de obs para el
#conjunto de entrenamiento

test <- (-train)
#test



X_train <- X[train ,]
X_test <- X[test, ]
#X_test

y_train <- y[train]
y_test <- y[test]
#View(X_test)
```

Inciso c)
Finalmente en la siguientes lineas buscamos el mejor conjunto de variables para cada tamaño de variables, es decir, de todas las posibles combinaciones de nCp tomamos la que mejor nos de el valor de MSE.
Una vez que encontramos estos mejores conjuntos lo que hacemos es realizar el pronostico para encontrar el MSE de entrenamiento y prueba, estos los guardamos y finalmente los gráficamos.

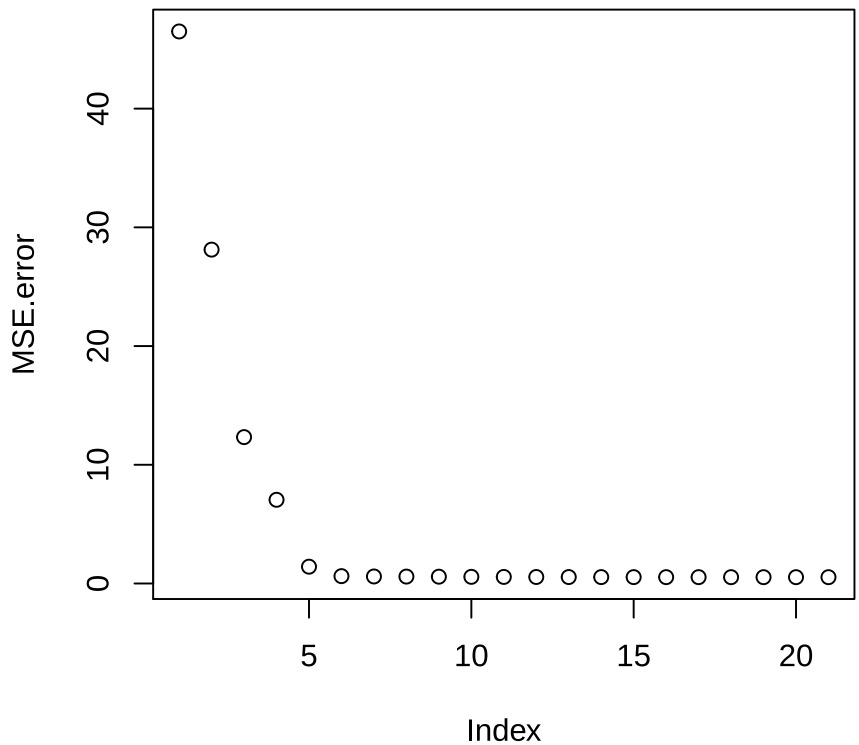



```
###################################################################
###################################################################

## por medio del comando seleccionamos los mejores modelos
## para cada valor de p=1,2,...,20
```

```r
model_subset <- regsubsets(x = datos[1:20], y = y_train, nvmax = 20)
variables <- summary(model_subset)$which[,-1]
MSE.error <- rep (0, 20 )
MSE.error_test <- rep (0, 20 )
for (i in 1:20) {
  variable_finales <- which(variables[i,])
  #print( variable_finales)
  c2 <- lm(y_train ~., datos[variable_finales] )

  ### train
  c2_p <- predict( c2, datos[ variable_finales ] )
  MSE.error[i] <- mean( (c2_p - y_train)**2 )

  ### test
  c2_p <- predict( c2, datos_p[ variable_finales ] )
  MSE.error_test[i] <- mean( (c2_p - y_test)**2 )

}
e0 <- mean((lm(y_train~1)$residuals)^2)
MSE.error <- c(e0, MSE.error)
MSE.error
```

```
 [1] 46.5018354 28.1236651 12.3315527  7.0503002  1.4155536  0.6188151
 [7]  0.5918655  0.5826938  0.5733816  0.5610504  0.5540695  0.5476703
[13]  0.5427275  0.5390767  0.5372393  0.5352930  0.5339877  0.5331196
[19]  0.5323354  0.5318448  0.5317645
```

```r
plot(MSE.error)
```



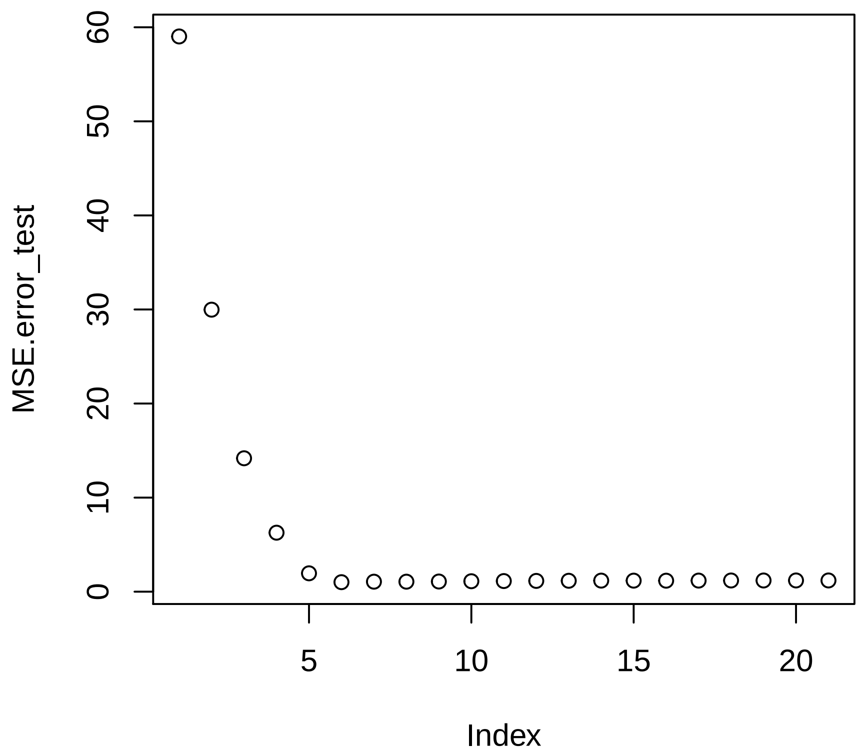
Adicionalmente a lo que solicita el ejercicio se agrego el gráfico tomando encuenta el C_p, solo con fines ilustrativos y tambien para observar si coinciden con nuestros resultados.

Inciso d)
Se muestra la gráfica de los MSE de prueba, se puede observar que se agrega el caso cuando no tomamos ninguna variable y solo el intercepto.

```r
e02 <- mean((y_test-1*lm(y_train~1)$coef)^2)
MSE.error_test <- c(e02, MSE.error_test)
MSE.error_test
```

```
 [1] 59.023269 29.978208 14.183038  6.270686  1.951733  1.009632  1.062225
 [8]  1.062116  1.078434  1.110662  1.128809  1.146103  1.160591  1.175614
[15]  1.172282  1.168780  1.178514  1.189812  1.189818  1.192720  1.193409
```

```r
plot(MSE.error_test)
```



Inciso e)

El tamaño de modelo que es el mínimo para el error de prueba MSE es 5. Se observa que su valor de error de MSE es de 1.009632.

Inciso f)

```
mejor_1 <- summary(model_subset)$which[,-1]
mejor_2 <- which( mejor_1[5,] )
mejor_2

 V3  V5  V6  V7 V12
  3   5   6   7  12

best <- lm(y_train ~., datos[mejor_2] )

best$coefficients

(Intercept)          V3          V5          V6          V7         V12
 0.03304589  0.97781217  1.99645773  4.97810280  3.12653619  3.83985209

betas

 [1] 0 0 1 0 2 5 3 0 0 0 0 4 0 0 0 0 0 0 0 0
```

Al observar los valores encontramos que son muy similares los valores de los betas y coinciden con las entradas que se pusieron al principio.

2. Problema 2

Generación de datos simulados y aplicación de los métodos de selección de subconjuntos

a) Usa una función de R para generar una variable predictora X de longitus $n = 100$, así como un vector de ruido ε de tamaños $n = 100$

b) Genera un vector de respuesta Y de longitud $n = 100$ de acuerdo al modelo

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \varepsilon$$

donde $\beta_0, \beta_1, \beta_2, \beta_3$ son constantes de tu elección

c) Utiliza la funcion $regsubserts()$ para realizar la seleccion de los mejores subconjuntos con el fin de elegir el mejor modelo que contenga los predictores $X, X^2, X^3, ..., X^{10}$, ¿ Cuál es el mejor modelo obtenido según el C_p, BIC y el R^2 ajustado?

d) Repite (c) usando la selección forward stepwise y backward stepwise ¿ Cómo se compara tu respuesta con los resultados obtenidos en (c)

Solución

Inciso a)
Generamos nuestros datos como lo menciona el inciso, en este caso pedimos datos de una distribución normal e igual para el vector de ruido.

```
set.seed(1234)
X <- matrix(rnorm(100),ncol = 1)
#X
e <- matrix(rnorm(100), ncol = 1)
#e
```

Inciso b)
Generamos los valores para la variable Y.

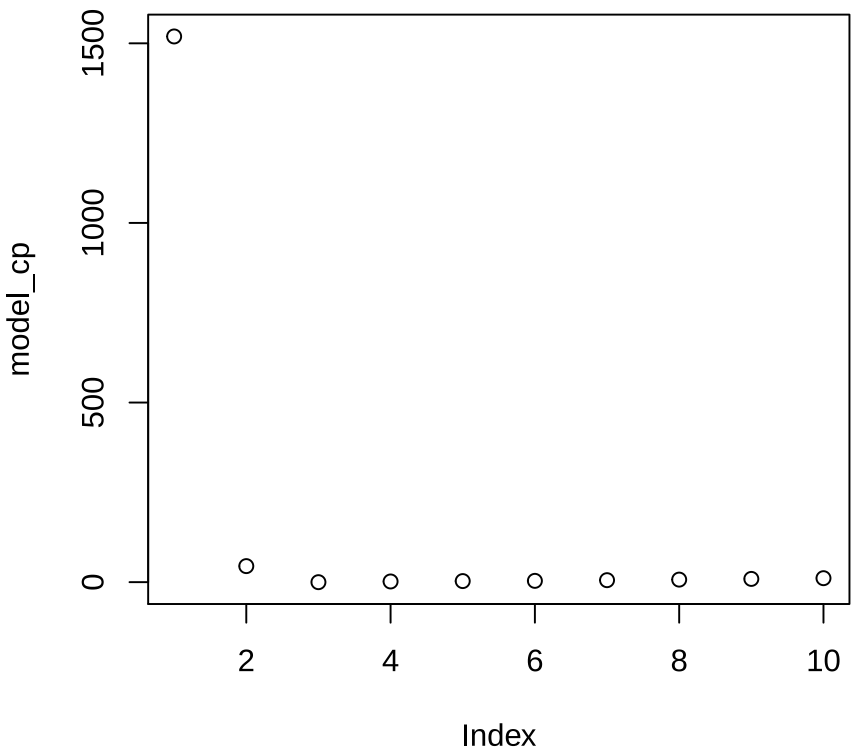
```
## Definimos el valor de las constantes
b_0 <- 2
b_1 <- 1.4
b_2 <- 10
b_3 <- 0.9

## Generamos el vector
Y <- b_0 + b_1*X + b_2*(X)**2 + b_3*(X)**3 + e
#Y
```

Inciso c)
Al realizar la función *regsubsets* y graficando observamos que el número

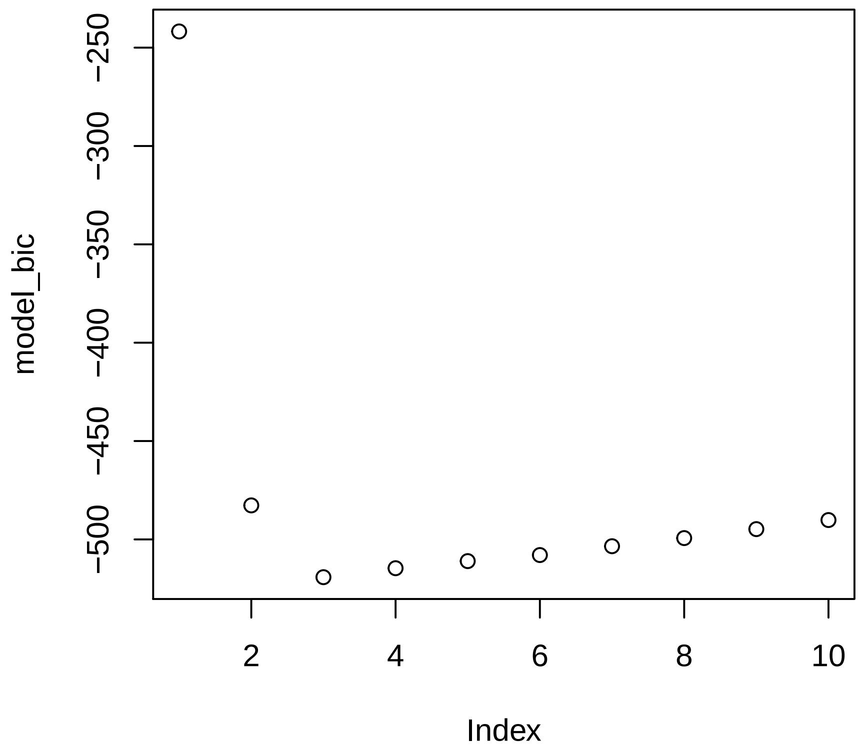
de variables que mejor nos ayuda es tomando un conjunto de tamanño 3 y estás se mandan a imprimir en pantalla(β_0=2.1324, β_1=1.3125, β_2=9.8936 y β_3=0.9323).
Adicional a lo anterior se muestran los graficos tomando en cuenta el C_p, BIC y el R^2 ajustado, en dichos graficos nos menciona que debemos tomar tambien 3 elementos.

```
X_p <- cbind(X, X**2, X**3, X**4, X**5, X**6, X**7,
             X**8, X**9, X**10)
#X_p
model_2 <- regsubsets(x = X_p, y = Y, nvmax = 10)
p <- summary(model_2)$which[,-1]
summary(model_2)$which

   (Intercept)     a    b     c     d     e     f     g     h     i     j
1         TRUE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
2         TRUE FALSE TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
3         TRUE  TRUE TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
4         TRUE  TRUE TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
5         TRUE  TRUE TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE
6         TRUE  TRUE TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE
7         TRUE  TRUE TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE
8         TRUE  TRUE TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
9         TRUE  TRUE TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
10        TRUE  TRUE TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE

### Dado que el intercepto sale en todos los conjuntos se omitio,
##   se imitio por cuestiones de espacio al momento de imprimir en
##   pantalla los resultados.

for (i in 1:10) {
  kobe <- which(p[i,])
  print(kobe)
}
```

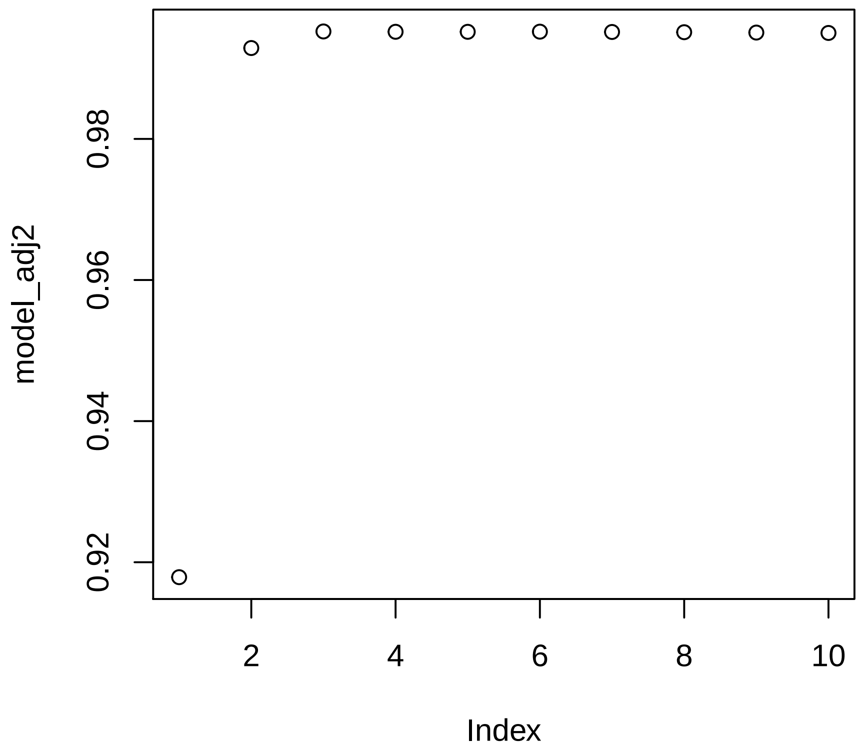
```
b
2
b c
2 3
a b c
1 2 3
a b c e
1 2 3 5
a b c d f
1 2 3 4 6
a b c d f h
1 2 3 4 6 8
a b c d e f h
1 2 3 4 5 6 8
 a  b  d  e  f  g  i  j
 1  2  4  5  6  7  9 10
 a  b  c  d  e  f  g  i  j
 1  2  3  4  5  6  7  9 10
 a  b  c  d  e  f  g  h  i  j
 1  2  3  4  5  6  7  8  9 10

coef(model_2,3)

(Intercept)           a           b           c
  2.1324697   1.3125864   9.8936268   0.9323054

model_cp <- summary(model_2)$cp
plot(model_cp)
```




```
model_bic <- summary(model_2)$bic
plot(model_bic)
```




```r
model_adj2 <- summary(model_2)$adjr2
plot(model_adj2)
```

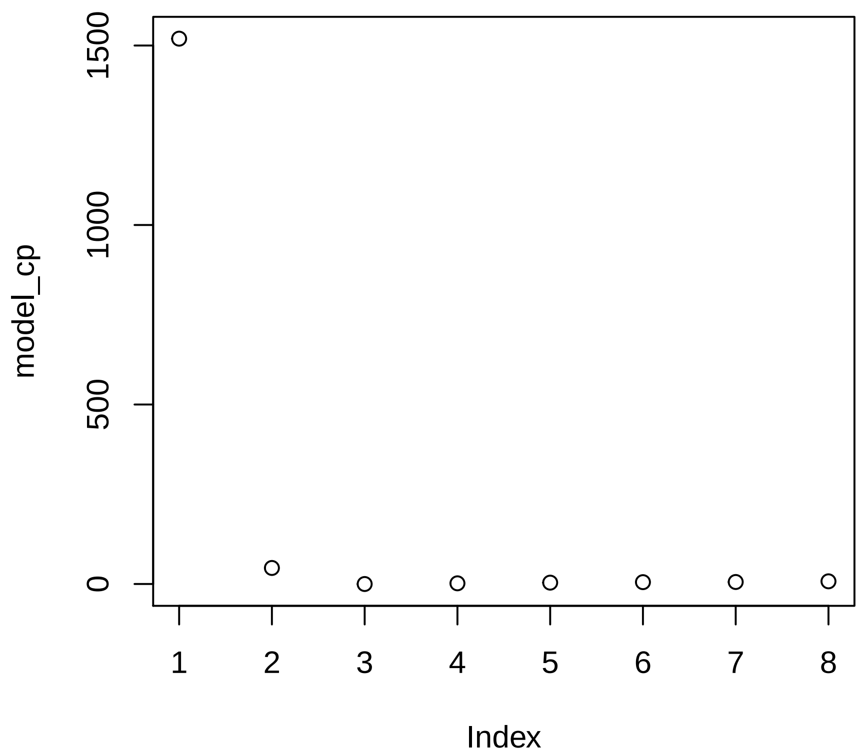


Inciso d)

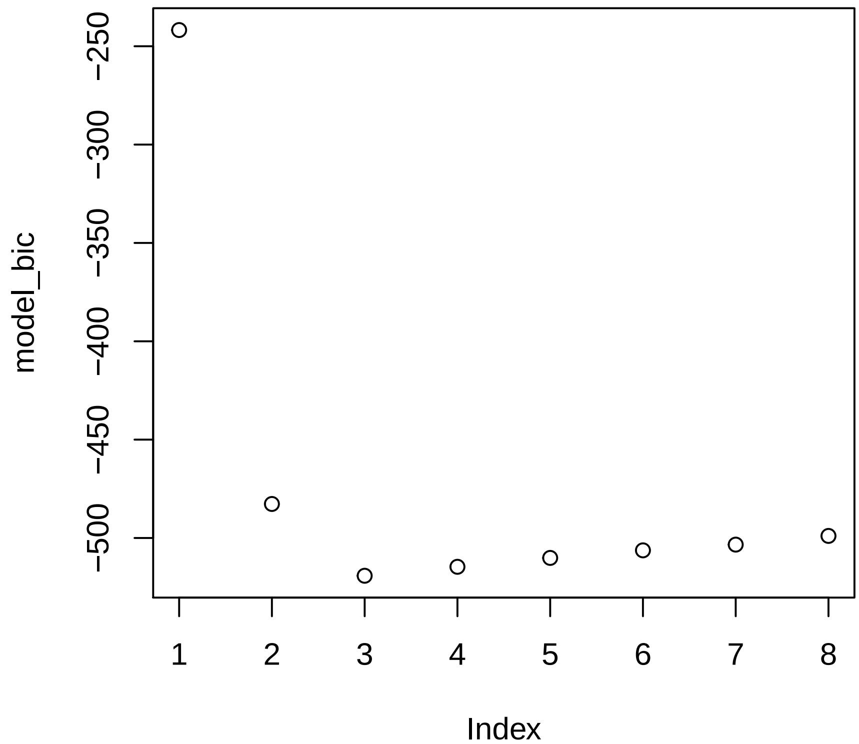
Al utilizar el método de *forward* observamos que coinciden los valores de los coeficientes con los que encontramos en el inciso c) pero con el método de *backward* cambiando estos valores.

```
##### metodo de forward
model_for <- regsubsets(x=X_p, y= Y, method = 'forward')
coef(model_for,3)
```

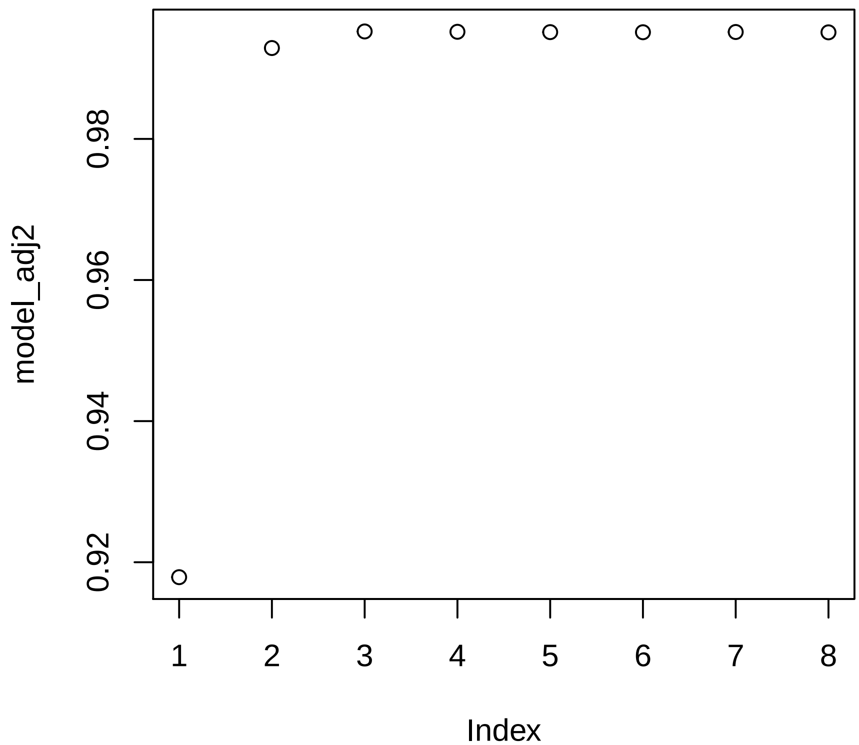
```
(Intercept)           a           b           c
  2.1324697   1.3125864   9.8936268   0.9323054

model_cp <- summary(model_for)$cp
plot(model_cp)
```



```r
model_bic <- summary(model_for)$bic
plot(model_bic)
```




```r
model_adj2 <- summary(model_for)$adjr2
plot(model_adj2)
```

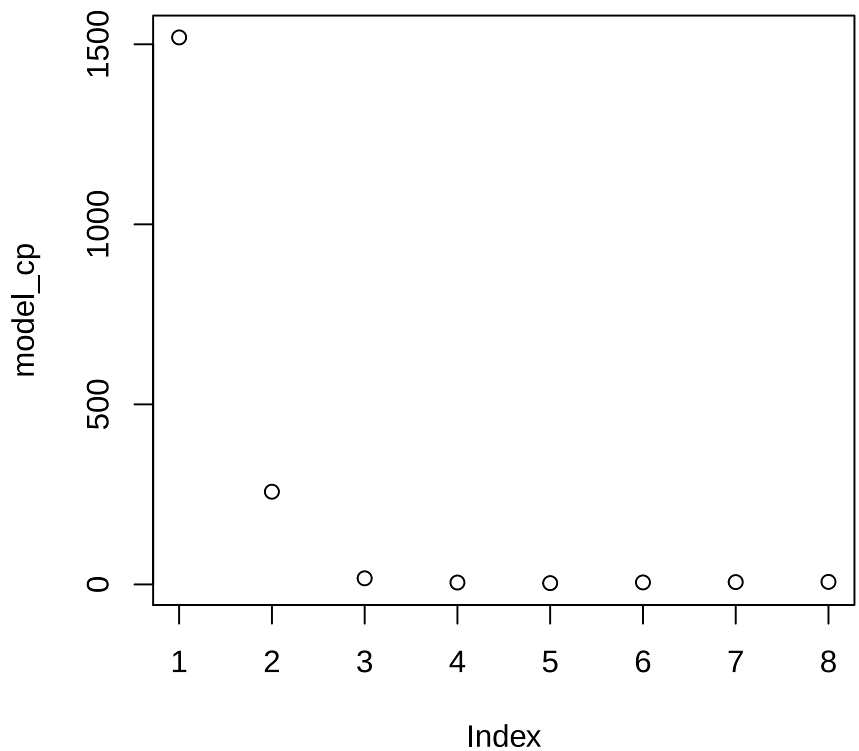



```r
###### metodo de backward
model_bac <- regsubsets(x=X_p, y= Y, method = 'backward')
coef(model_bac,3)
```

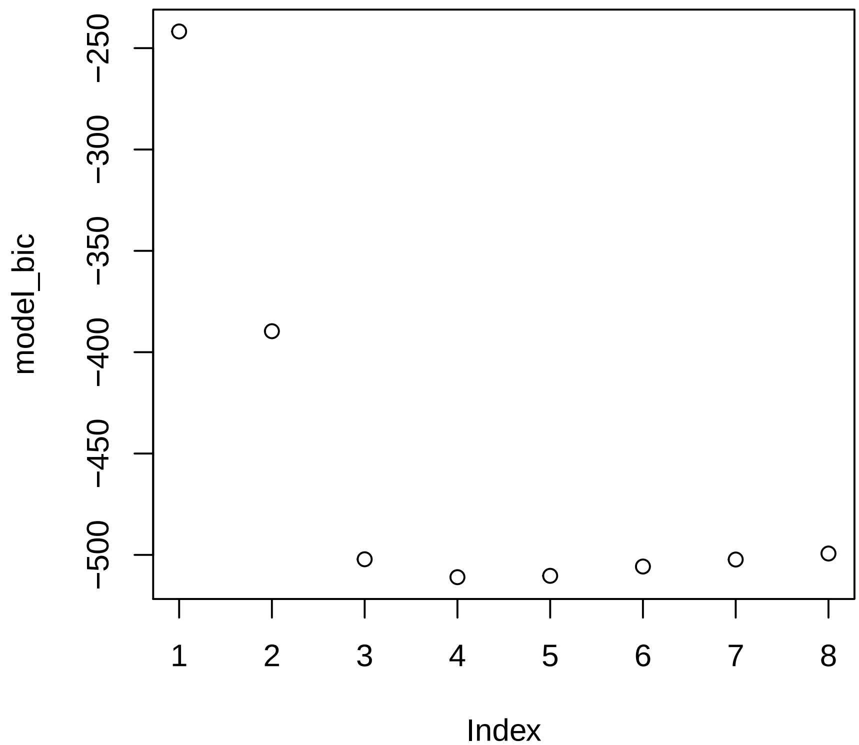
```
(Intercept)           a           b           e
  2.2802567   2.3921917   9.8341308   0.1335067
```

```r
model_cp <- summary(model_bac)$cp
```

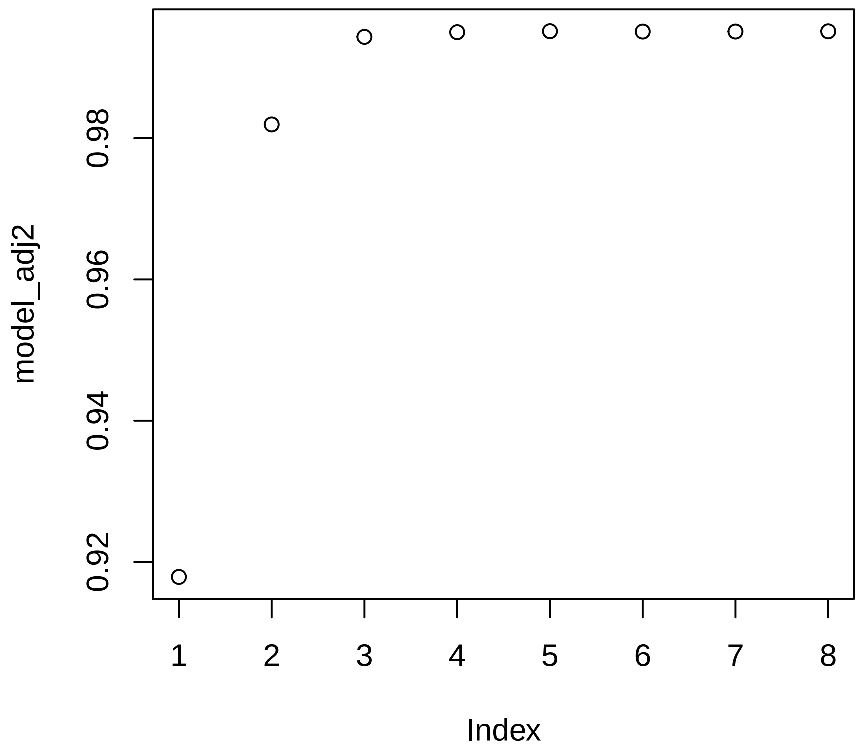
```r
plot(model_cp)
```




```r
model_bic <- summary(model_bac)$bic
plot(model_bic)
```




```
model_adj2 <- summary(model_bac)$adjr2
plot(model_adj2)
```


model_adj2
0.92
0.94
0.96
0.98
1
2
3
4
5
6
7
8
Index