Proyecto NBA2

Alejandro Carrillo Vera

17/10/2019

```r
#Libreria utilizada

library(ggplot2)
library(leaps)
library(rsample)
```

```
## Loading required package: tidyr
```

```r
library(glmnet)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loading required package: foreach

## Loaded glmnet 2.0-18
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```r
library(ISLR)
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library(glmnet)

#Cargamos base de datos

nba <- read.csv("nba.csv")
nba <- na.omit(nba)

#Calculamos los coeficientes de la regresión excluyendo las variables jugador, pais y equipo
#ya que no las considero relevantes a la hora de explicar el salario
coef(lm(Salary ~.-Player-NBA_Country-Tm, data = nba))
```

```
##     (Intercept) NBA_DraftNumber             Age               G
##    -2250339.314      -60481.433      516820.691     -154410.877
##              MP             PER             TS.           X3PAr
##        5656.643     -355059.419    -2162766.695    -3458208.806
##             FTr            ORB.            DRB.            TRB.
##     -158470.428    -1055234.282     -855005.007     2006675.768
##            AST.            STL.            BLK.            TOV.
##      -19606.224     -196551.448      110237.671        4208.313
##            USG.             OWS             DWS              WS
##      169430.644    -1271685.169    -1735775.413     1827796.178
##           WS.48            OBPM            DBPM             BPM
##     1914676.695     1878971.437     1438901.496    -1295953.999
##            VORP
##      629465.435
```

```r
#Calculamos una regresión a través del método backward
mejores_mod <- regsubsets(Salary~.-Player-NBA_Country-Tm, data = nba, nvmax = 25, method = "backward")
summary(mejores_mod)
```

```
## Subset selection object
## Call: regsubsets.formula(Salary ~ . - Player - NBA_Country - Tm, data = nba,
##     nvmax = 25, method = "backward")
## 24 Variables  (and intercept)
##                 Forced in Forced out
## NBA_DraftNumber     FALSE      FALSE
## Age                 FALSE      FALSE
## G                   FALSE      FALSE
## MP                  FALSE      FALSE
## PER                 FALSE      FALSE
## TS.                 FALSE      FALSE
## X3PAr                FALSE      FALSE
## FTr                  FALSE      FALSE
## ORB.                 FALSE      FALSE
## DRB.                 FALSE      FALSE
## TRB.                 FALSE      FALSE
## AST.                 FALSE      FALSE
## STL.                 FALSE      FALSE
## BLK.                 FALSE      FALSE
## TOV.                 FALSE      FALSE
## USG.                 FALSE      FALSE
## OWS                  FALSE      FALSE
## DWS                  FALSE      FALSE
## WS                   FALSE      FALSE
## WS.48                FALSE      FALSE
## OBPM                 FALSE      FALSE
## DBPM                 FALSE      FALSE
## BPM                  FALSE      FALSE
## VORP                 FALSE      FALSE
## 1 subsets of each size up to 24
## Selection Algorithm: backward
##           NBA_DraftNumber Age G   MP  PER TS. X3PAr FTr ORB. DRB. TRB.
## 1  ( 1 )  " "             " " " " "*" " " " " " "   " " " "  " "  " "
## 2  ( 1 )  " "             " " "*" "*" " " " " " "   " " " "  " "  " "
## 3  ( 1 )  " "             "*" "*" "*" " " " " " "   " " " "  " "  " "
## 4  ( 1 )  " "             "*" "*" "*" " " " " " "   " " " "  " "  " "
## 5  ( 1 )  "*"             "*" "*" "*" " " " " " "   " " " "  " "  " "
## 6  ( 1 )  "*"             "*" "*" "*" " " " " " "   " " " "  " "  "*"
## 7  ( 1 )  "*"             "*" "*" "*" " " " " " "   " " "*"  " "  "*"
## 8  ( 1 )  "*"             "*" "*" "*" " " " " " "   " " "*"  " "  "*"
## 9  ( 1 )  "*"             "*" "*" "*" " " " " " "   " " "*"  " "  "*"
## 10  ( 1 ) "*"             "*" "*" "*" "*" " " " "   " " "*"  " "  "*"
## 11  ( 1 ) "*"             "*" "*" "*" "*" " " "*"   " " "*"  " "  "*"
## 12  ( 1 ) "*"             "*" "*" "*" "*" " " "*"   " " "*"  " "  "*"
## 13  ( 1 ) "*"             "*" "*" "*" "*" " " "*"   " " "*"  "*"  "*"
## 14  ( 1 ) "*"             "*" "*" "*" "*" " " "*"   " " "*"  "*"  "*"
## 15  ( 1 ) "*"             "*" "*" "*" "*" " " "*"   " " "*"  "*"  "*"
## 16  ( 1 ) "*"             "*" "*" "*" "*" " " "*"   " " "*"  "*"  "*"
## 17  ( 1 ) "*"             "*" "*" "*" "*" " " "*"   " " "*"  "*"  "*"
## 18  ( 1 ) "*"             "*" "*" "*" "*" "*" "*"   " " "*"  "*"  "*"
## 19  ( 1 ) "*"             "*" "*" "*" "*" "*" "*"   " " "*"  "*"  "*"
## 20  ( 1 ) "*"             "*" "*" "*" "*" "*" "*"   " " "*"  "*"  "*"
## 21  ( 1 ) "*"             "*" "*" "*" "*" "*" "*"   " " "*"  "*"  "*"
## 22  ( 1 ) "*"             "*" "*" "*" "*" "*" "*"   " " "*"  "*"  "*"
## 23  ( 1 ) "*"             "*" "*" "*" "*" "*" "*"   "*" "*"  "*"  "*"
## 24  ( 1 ) "*"             "*" "*" "*" "*" "*" "*"   "*" "*"  "*"  "*"
##           AST. STL. BLK. TOV. USG. OWS DWS WS  WS.48 OBPM DBPM BPM VORP
## 1  ( 1 )  " "  " "  " "  " "  " "  " " " " " " " "   " "  " "  " " " "
## 2  ( 1 )  " "  " "  " "  " "  " "  " " " " " " " "   " "  " "  " " " "
## 3  ( 1 )  " "  " "  " "  " "  " "  " " " " " " " "   " "  " "  " " " "
## 4  ( 1 )  " "  " "  " "  " "  " "  " " " " "*" " "   " "  " "  " " " "
## 5  ( 1 )  " "  " "  " "  " "  " "  " " " " "*" " "   " "  " "  " " " "
## 6  ( 1 )  " "  " "  " "  " "  " "  " " " " "*" " "   " "  " "  " " " "
## 7  ( 1 )  " "  " "  " "  " "  " "  " " " " "*" " "   " "  " "  " " " "
## 8  ( 1 )  " "  " "  " "  " "  "*"  " " " " "*" " "   " "  " "  " " " "
## 9  ( 1 )  " "  " "  " "  " "  "*"  " " " " "*" " "   "*"  " "  " " " "
## 10  ( 1 ) " "  " "  " "  " "  "*"  " " " " "*" " "   "*"  " "  " " " "
## 11  ( 1 ) " "  " "  " "  " "  "*"  " " " " "*" " "   "*"  " "  " " " "
## 12  ( 1 ) " "  " "  " "  " "  "*"  " " " " "*" " "   "*"  " "  " " "*"
## 13  ( 1 ) " "  " "  " "  " "  "*"  " " " " "*" " "   "*"  " "  " " "*"
## 14  ( 1 ) " "  "*"  " "  " "  "*"  " " " " "*" " "   "*"  " "  " " "*"
## 15  ( 1 ) " "  "*"  " "  " "  "*"  " " " " "*" " "   "*"  "*"  " " "*"
## 16  ( 1 ) " "  "*"  " "  " "  "*"  " " "*" "*" " "   "*"  "*"  " " "*"
## 17  ( 1 ) "*"  "*"  " "  " "  "*"  " " "*" "*" " "   "*"  "*"  " " "*"
## 18  ( 1 ) "*"  "*"  " "  " "  "*"  " " "*" "*" " "   "*"  "*"  " " "*"
## 19  ( 1 ) "*"  "*"  "*"  " "  "*"  " " "*" "*" " "   "*"  "*"  " " "*"
## 20  ( 1 ) "*"  "*"  "*"  " "  "*"  "*" "*" "*" " "   "*"  "*"  " " "*"
## 21  ( 1 ) "*"  "*"  "*"  " "  "*"  "*" "*" "*" " "   "*"  "*"  "*" "*"
## 22  ( 1 ) "*"  "*"  "*"  " "  "*"  "*" "*" "*" "*"   "*"  "*"  "*" "*"
## 23  ( 1 ) "*"  "*"  "*"  " "  "*"  "*" "*" "*" "*"   "*"  "*"  "*" "*"
## 24  ( 1 ) "*"  "*"  "*"  "*"  "*"  "*" "*" "*" "*"   "*"  "*"  "*" "*"
```

```r
names(summary(mejores_mod))
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```r
summary(mejores_mod)$adjr2
```

```
##  [1] 0.2535726 0.3425206 0.4343876 0.4908340 0.5260436 0.5280557 0.5295117
##  [8] 0.5304986 0.5295724 0.5306824 0.5328592 0.5332432 0.5331498 0.5322922
## [15] 0.5318014 0.5313328 0.5307330 0.5300199 0.5291040 0.5281729 0.5272245
## [22] 0.5262195 0.5252191 0.5241891
```

```r
which.max(summary(mejores_mod)$adjr2)
```

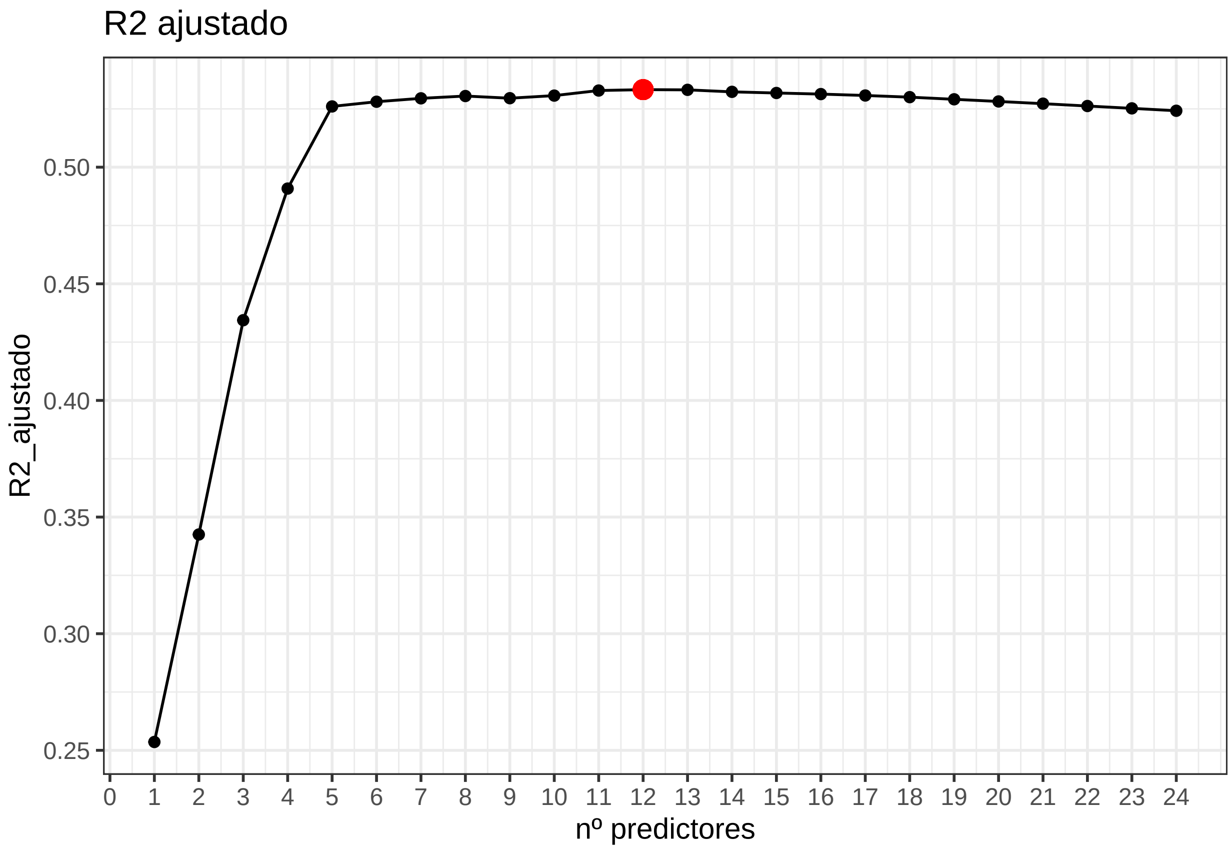
```
## [1] 12
```

```r
#El mayor valor de R2 es el 12
coef(object = mejores_mod, 12)
```

```
##     (Intercept) NBA_DraftNumber             Age               G
##    -4268728.268      -62571.804      515670.889     -150289.535
##              MP             PER           X3PAr            ORB.
##        5214.425     -290651.425    -2915862.340     -190994.539
##            TRB.            USG.              WS            OBPM
##      315165.126      122263.328      543272.358      499292.407
##            VORP
##      609504.140
```

```r
#Dibujamos el R2 ajustado y vemos que como nos había mostrado antes, el mayor valor es en el predictor
p <- ggplot(data = data.frame(Predictores = 1:24,
                              R2_ajustado = summary(mejores_mod)$adjr2),
            aes(x = Predictores, y = R2_ajustado)) +
  geom_line() +
  geom_point()
```

```
p <- p + geom_point(aes(
  x = Predictores[which.max(summary(mejores_mod)$adjr2)],
  y = R2_ajustado[which.max(summary(mejores_mod)$adjr2)]),
  colour = "red", size = 3)
p <- p +  scale_x_continuous(breaks = c(0:24)) +
  theme_bw() +
  labs(title = 'R2 ajustado',
       x =  'nº predictores')
p
```



```
#CROSS VALIDATION - VALIDATION SET

set.seed(250)
train <- sample(x = 1:483, size = 97, replace = FALSE)


mejor_mod <- regsubsets(Salary~.-Player-NBA_Country-Tm, data = nba[train,], nvmax = 25, method = "backwa
mejor_mod
```

```
## Subset selection object
## Call: regsubsets.formula(Salary ~ . - Player - NBA_Country - Tm, data = nba[train,
##     ], nvmax = 25, method = "backward")
## 24 Variables  (and intercept)
##                   Forced in Forced out
## NBA_DraftNumber     FALSE      FALSE
## Age                 FALSE      FALSE
## G                   FALSE      FALSE
## MP                  FALSE      FALSE
## PER                 FALSE      FALSE
## TS.                 FALSE      FALSE
## X3PAr               FALSE      FALSE
## FTr                 FALSE      FALSE
## ORB.                FALSE      FALSE
## DRB.                FALSE      FALSE
## TRB.                FALSE      FALSE
## AST.                FALSE      FALSE
## STL.                FALSE      FALSE
## BLK.                FALSE      FALSE
## TOV.                FALSE      FALSE
## USG.                FALSE      FALSE
## OWS                 FALSE      FALSE
## DWS                 FALSE      FALSE
## WS                  FALSE      FALSE
## WS.48               FALSE      FALSE
## OBPM                FALSE      FALSE
## DBPM                FALSE      FALSE
## BPM                 FALSE      FALSE
## VORP                FALSE      FALSE
## 1 subsets of each size up to 24
## Selection Algorithm: backward
```

```r
Error_val <- rep(NA, 24)
test_matrix <- model.matrix(Salary~.-Player-NBA_Country-Tm, data = nba[-train, ])


for (i in 1:24) {
  coeficientes <- coef(object = mejor_mod, id = i)
  predictores <- test_matrix[, names(coeficientes)]
  predicciones <- predictores %*% coeficientes
  Error_val[i] <- mean((nba$Salary[-train] - predicciones)^2)
}

which.min(Error_val)
```

```
## [1] 2
```

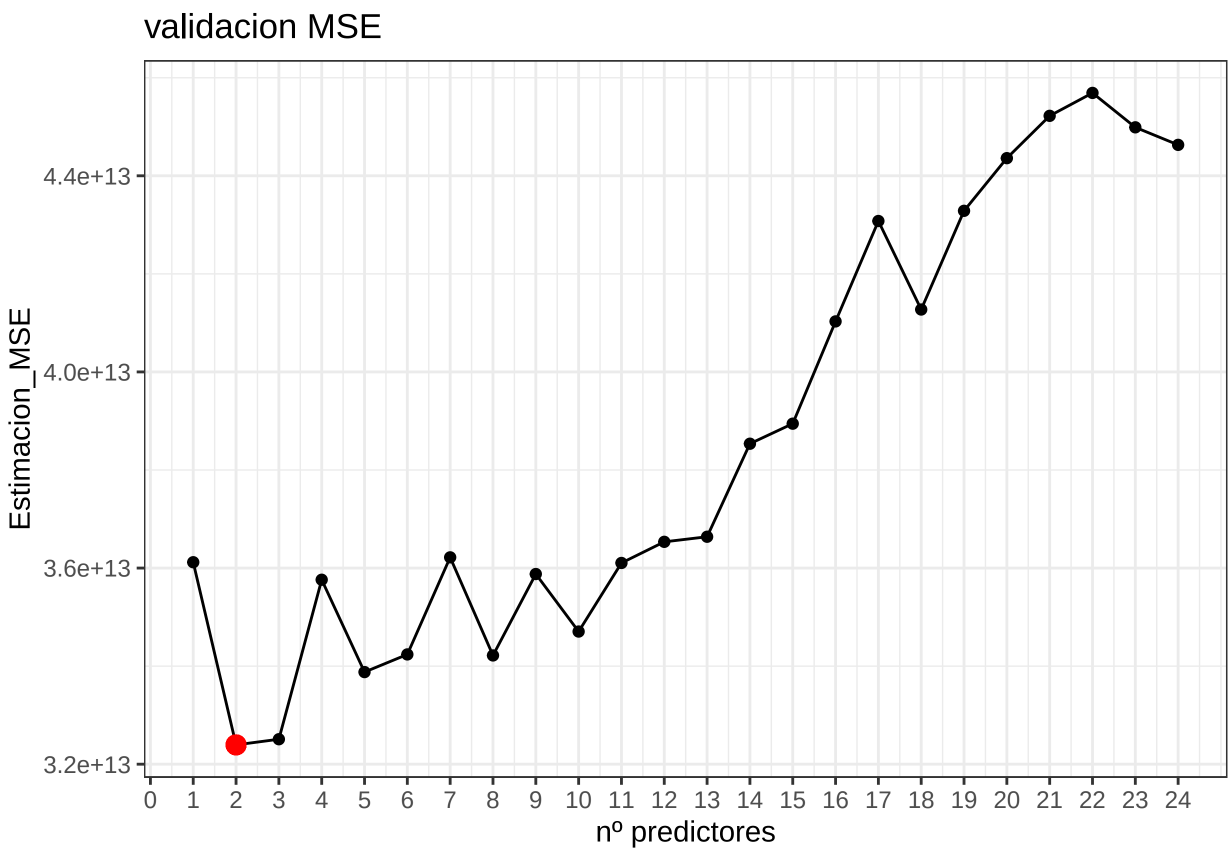
```r
sqrt(Error_val[2])
```

```
## [1] 5691545
```

```r
Error_val
```

```
##  [1] 3.611877e+13 3.239369e+13 3.250928e+13 3.576191e+13 3.387897e+13
##  [6] 3.423871e+13 3.621862e+13 3.421818e+13 3.587842e+13 3.470534e+13
## [11] 3.610407e+13 3.653499e+13 3.663878e+13 3.853589e+13 3.894491e+13
## [16] 4.103028e+13 4.307806e+13 4.127322e+13 4.328575e+13 4.435967e+13
## [21] 4.522278e+13 4.569004e+13 4.498848e+13 4.463059e+13
```

```r
#El minimo error sale en el predictor 2

q <- ggplot(data = data.frame(n_predictores = 1:24,
                              Estimacion_MSE = Error_val),
            aes(x = n_predictores, y = Estimacion_MSE)) +
  geom_line() +
  geom_point()

q <- q + geom_point(aes(x = n_predictores[which.min(Error_val)],
                        y = Error_val[which.min(Error_val)]),
                    colour = "red", size = 3)

q <- q +  scale_x_continuous(breaks = c(0:24)) +
  theme_bw() +
  labs(title = 'validacion MSE',
       x =  'nº predictores')
q
```



```r
#Dibujamos la validación por MSE dandonos dos
mejor_mod1 <- regsubsets(Salary~.-Player-NBA_Country-Tm, data = nba[train,], nvmax = 25, method = "backw

coef(object = mejor_mod1, id = 2)
```

```
## (Intercept)         Age          WS
## -11548086.5    538843.9   1622806.3
```

```r
#ELASTIC NET

set.seed(250)
nba_split <- initial_split(nba, prop = .80, strata = "Salary")
nba_train <- training(nba_split)
nba_test  <- testing(nba_split)

nba_train_x <- model.matrix(Salary ~ .,nba_train)[, -1]
nba_train_y <- nba_train$Salary


nba_test_x <- model.matrix(Salary ~ ., nba_test)[, -1]
nba_test_y <- nba_test$Salary


train_control <- trainControl(method = "cv", number = 10)

caret_mod <- train(
  x = nba_train_x,
  y = nba_train_y,
  method = "glmnet",
  preProc = c("center", "scale", "zv", "nzv"),
  trControl = train_control,
  tuneLength = 10
)


caret_mod
```

```
## glmnet
##
## 388 samples
## 579 predictors
##
## Pre-processing: centered (26), scaled (26), remove (553)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 349, 350, 350, 350, 348, 348, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda       RMSE     Rsquared   MAE
##   0.1       2021.669  5418587  0.4724612  3998066
##   0.1       4670.318  5417743  0.4725484  3997659
##   0.1      10789.042  5384445  0.4755292  3985691
##   0.1      24924.086  5337347  0.4805124  3963867
##   0.1      57577.873  5282409  0.4876809  3933218
##   0.1     133012.359  5243917  0.4940016  3898201
##   0.1     307275.808  5234010  0.4950917  3864200
##   0.1     709846.988  5241620  0.4938790  3833906
##   0.1    1639838.652  5305005  0.4858002  3876012
##   0.1    3788240.067  5486751  0.4657077  4117432
##   0.2       2021.669  5431197  0.4712892  4002629
##   0.2       4670.318  5416027  0.4726229  3997804
##   0.2      10789.042  5376472  0.4763037  3982913
##   0.2     24924.086  5324714  0.4820238  3958794
##   0.2     57577.873  5265619  0.4905363  3923520
##   0.2    133012.359  5226327  0.4973860  3882796
##   0.2    307275.808  5207514  0.5001478  3831095
##   0.2    709846.988  5231327  0.4969052  3807754
##   0.2   1639838.652  5374041  0.4758421  3943537
##   0.2   3788240.067  5601483  0.4623084  4291015
##   0.3      2021.669  5403703  0.4730361  3996964
##   0.3      4670.318  5395881  0.4740804  3993221
##   0.3     10789.042  5370828  0.4769123  3980515
##   0.3     24924.086  5311288  0.4838198  3952269
##   0.3     57577.873  5250187  0.4934343  3914684
##   0.3    133012.359  5208220  0.5006387  3863460
##   0.3    307275.808  5190035  0.5036852  3804336
##   0.3    709846.988  5252953  0.4940543  3810966
##   0.3   1639838.652  5420360  0.4718024  4021125
##   0.3   3788240.067  5778087  0.4490119  4497119
##   0.4      2021.669  5400180  0.4733234  3996160
##   0.4      4670.318  5391543  0.4744506  3991763
##   0.4     10789.042  5362344  0.4777752  3976914
##   0.4     24924.086  5301114  0.4852164  3946408
##   0.4     57577.873  5233526  0.4967111  3903638
##   0.4    133012.359  5196343  0.5027560  3846810
##   0.4    307275.808  5179556  0.5059912  3782402
##   0.4    709846.988  5295211  0.4868128  3842498
##   0.4   1639838.652  5468919  0.4694654  4106929
##   0.4   3788240.067  5970801  0.4338984  4707210
##   0.5      2021.669  5430037  0.4711208  4003791
##   0.5      4670.318  5403275  0.4736427  3994251
##   0.5     10789.042  5352890  0.4787656  3973384
##   0.5     24924.086  5289103  0.4870233  3939942
##   0.5     57577.873  5221540  0.4990119  3892843
##   0.5    133012.359  5184682  0.5048916  3829570
##   0.5    307275.808  5180843  0.5062091  3771617
##   0.5    709846.988  5348781  0.4771815  3893339
##   0.5   1639838.652  5532414  0.4648891  4201855
##   0.5   3788240.067  6201898  0.4028306  4932727
##   0.6      2021.669  5425355  0.4717121  4000264
##   0.6      4670.318  5396409  0.4743292  3991706
##   0.6     10789.042  5345721  0.4795912  3970164
##   0.6     24924.086  5277108  0.4889436  3934311
##   0.6     57577.873  5212025  0.5006464  3883693
##   0.6    133012.359  5174958  0.5067316  3815128
##   0.6    307275.808  5191228  0.5045753  3769294
##   0.6    709846.988  5374668  0.4735805  3928872
##   0.6   1639838.652  5607103  0.4585092  4302064
##   0.6   3788240.067  6430116  0.3621911  5133825
##   0.7      2021.669  5398895  0.4734649  3994988
##   0.7      4670.318  5387924  0.4750532  3989370
##   0.7     10789.042  5339481  0.4803123  3966763
##   0.7     24924.086  5267241  0.4905964  3928869
##   0.7     57577.873  5204221  0.5019325  3875261
##   0.7    133012.359  5166134  0.5083860  3801575
##   0.7    307275.808  5208070  0.5017091  3775021
##   0.7     709846.988  5391174  0.4724737  3958768
##   0.7    1639838.652  5685830  0.4521076  4399761
##   0.7    3788240.067  6584302  0.3539515  5255716
##   0.8       2021.669  5393215  0.4739988  3992790
##   0.8       4670.318  5383094  0.4755541  3987301
##   0.8      10789.042  5332888  0.4810970  3963753
##   0.8      24924.086  5257659  0.4922949  3923481
##   0.8      57577.873  5199194  0.5027467  3868696
##   0.8     133012.359  5159088  0.5097217  3789351
##   0.8     307275.808  5230829  0.4976569  3786106
##   0.8     709846.988  5412468  0.4706726  3996870
##   0.8    1639838.652  5764708  0.4460851  4492815
##   0.8    3788240.067  6741619  0.3495807  5376536
##   0.9       2021.669  5397824  0.4734584  3995546
##   0.9       4670.318  5383124  0.4756643  3986580
##   0.9      10789.042  5324655  0.4821319  3959687
##   0.9      24924.086  5246974  0.4942903  3917479
##   0.9      57577.873  5194277  0.5035322  3861897
##   0.9     133012.359  5153900  0.5108140  3778509
##   0.9     307275.808  5259485  0.4923737  3803622
##   0.9     709846.988  5438567  0.4679672  4039207
##   0.9    1639838.652  5847414  0.4385271  4584984
##   0.9    3788240.067  6912969  0.3379245  5502648
##   1.0       2021.669  5392313  0.4741428  3992438
##   1.0       4670.318  5379602  0.4760837  3985072
##   1.0      10789.042  5318690  0.4829089  3956930
##   1.0      24924.086  5234906  0.4966691  3909302
##   1.0      57577.873  5188197  0.5045786  3853314
##   1.0     133012.359  5152339  0.5111845  3770457
##   1.0     307275.808  5293566  0.4859836  3828361
##   1.0     709846.988  5467714  0.4648061  4084098
##   1.0    1639838.652  5948451  0.4241635  4685863
##   1.0    3788240.067  7087514  0.3346667  5626820
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 1 and lambda = 133012.4.
```

```r
#Elegimos alpha 1 por lo que pasaríamos a hacer un LASSO

cv_lasso <- cv.glmnet(x = nba_train_x, y = nba_train_y, alpha = 0.7)
min(cv_lasso$cvm)
```

```
## [1] 3.07027e+13
```

```r
pred <- predict(cv_lasso,s=cv_lasso$lambda.min,nba_test_x)
mean((nba_test_y-  pred)^2)
```

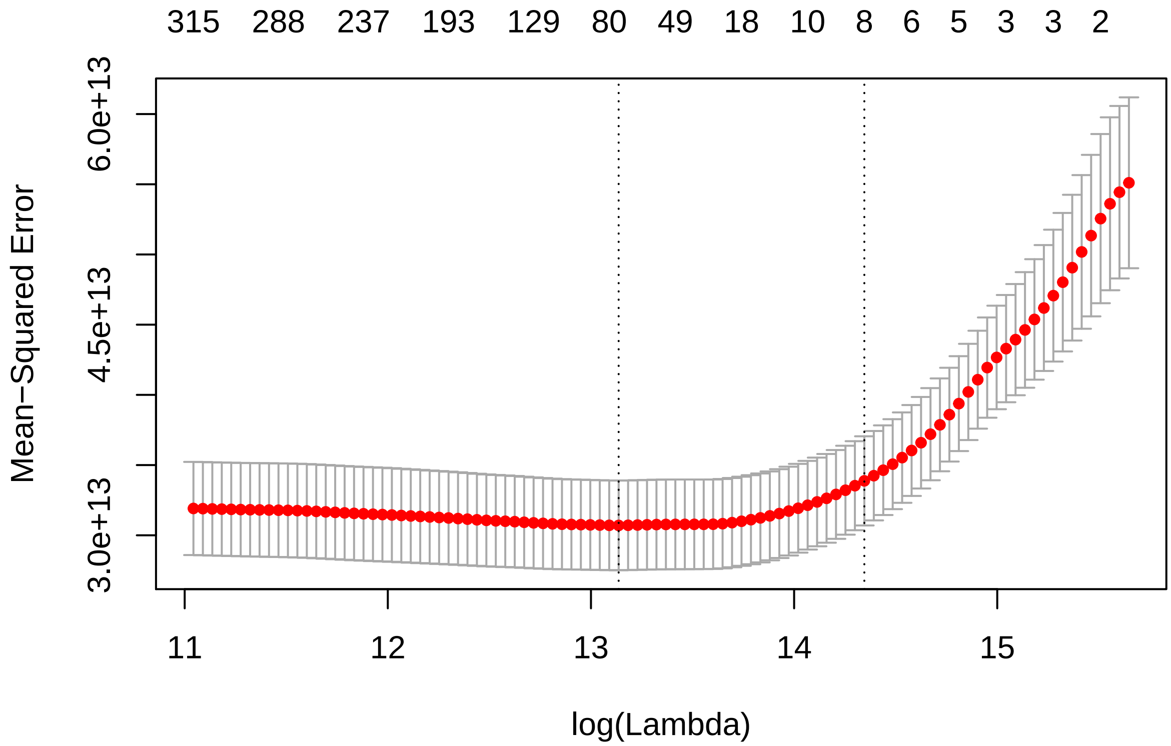
```
## [1] 2.732014e+13
```

```r
sqrt(2.732014e+13)
```

```
## [1] 5226867
```

```r
plot(cv_lasso, xvar = "lambda", label = TRUE)
```

```
## Warning in plot.window(...): "xvar" is not a graphical parameter

## Warning in plot.window(...): "label" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "xvar" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "label" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "xvar" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "label" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "xvar" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "label" is not
## a graphical parameter

## Warning in box(...): "xvar" is not a graphical parameter

## Warning in box(...): "label" is not a graphical parameter

## Warning in title(...): "xvar" is not a graphical parameter

## Warning in title(...): "label" is not a graphical parameter
```



```
#El error resultante es de un poco más de 5 millones
```