nba2_regularizacion

Alvaro Muñoz Jimenez

17/10/2019

#PREDICCIÓN

#EJERCICIO REGULARIZACIÓN

#Para realizar el trabajado he utilizado los sigiuentes paquetes de librerías:

```
library(rsample)
```

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

```
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
```

```
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
```

```
library(ggplot2)
library(nortest)
library(readr)
library(ISLR)
library(leaps)
library(caret)
```

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

```r
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
## Attaching package: 'GGally'
```

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

```r
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
library(PerformanceAnalytics)
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
##
## Attaching package: 'PerformanceAnalytics'
```

```
## The following object is masked from 'package:graphics':
##
##     legend
```

#El objetico principal del trabajo es comprobar la regularización.

#En primer lugar importo el dataset correspondiente a los datos de “nba”.

```
nba<-read.csv("C:/Users/alvar/Desktop/CUNEF/PREDICCIÓN/nba.csv")
```

#El siguiente paso es suprimir los NAs correspondientes al objeto nba.

```
nba <- unique(nba)
nba <- na.omit(nba)
```

#Técnica de regularización Elastic net. #Training y test split.

```
set.seed(123)
nba_split <- initial_split(nba, prop = .7, strata = "Salary")
nba_train <- training(nba_split)
nba_test <- testing(nba_split)
```

#Creación de matrices. Establecemos las matrices de entrenamiento y test con #lasmvariables más relevantes.

```
nba_train_x <- model.matrix(Salary ~ Age  + NBA_DraftNumber +  MP + USG. + VORP + WS +
                            AST. + TS.  , data = nba_train)[, -1]

nba_train_y <- log(nba_train$Salary)

nba_test_x <- model.matrix(Salary ~ Age  + NBA_DraftNumber + MP + USG. + VORP + WS +
                           AST. + TS. , data = nba_test)[, -1]

nba_test_y <- log(nba_test$Salary)
```

#Comprobamos la dimensión de la matriz

```
dim(nba_train_x)
```

```
## [1] 340   8
```

```
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
##
## 340 samples
##   8 predictor
##
## Pre-processing: centered (8), scaled (8)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 306, 307, 306, 305, 307, 306, ...
## Resampling results across tuning parameters:
##
```

```
##    alpha  lambda        RMSE      Rsquared   MAE
##    0.1    0.0009157003  1.041941  0.5045152  0.7791183
##    0.1    0.0021153864  1.041941  0.5045152  0.7791183
##    0.1    0.0048868171  1.041941  0.5045152  0.7791183
##    0.1    0.0112891816  1.041703  0.5047184  0.7788559
##    0.1    0.0260794744  1.040601  0.5055322  0.7775559
##    0.1    0.0602469699  1.038793  0.5069774  0.7751594
##    0.1    0.1391783180  1.036148  0.5102999  0.7710761
##    0.1    0.3215199740  1.038634  0.5140732  0.7708695
##    0.1    0.7427528597  1.065882  0.5132223  0.7890731
##    0.2    0.0009157003  1.041806  0.5045821  0.7791763
##    0.2    0.0021153864  1.041806  0.5045821  0.7791763
##    0.2    0.0048868171  1.041806  0.5045821  0.7791763
##    0.2    0.0112891816  1.041313  0.5049850  0.7787220
##    0.2    0.0260794744  1.039731  0.5061577  0.7772409
##    0.2    0.0602469699  1.036457  0.5090601  0.7740981
##    0.2    0.1391783180  1.032494  0.5143862  0.7698411
##    0.2    0.3215199740  1.039057  0.5181070  0.7732487
##    0.2    0.7427528597  1.088348  0.5115124  0.8105783
##    0.3    0.0009157003  1.041764  0.5046090  0.7792569
##    0.3    0.0021153864  1.041764  0.5046090  0.7792569
##    0.3    0.0048868171  1.041764  0.5046090  0.7792569
##    0.3    0.0112891816  1.040924  0.5052543  0.7785875
##    0.3    0.0260794744  1.038833  0.5068378  0.7768902
##    0.3    0.0602469699  1.034283  0.5110930  0.7730293
##    0.3    0.1391783180  1.031379  0.5163092  0.7699970
##    0.3    0.3215199740  1.044727  0.5181004  0.7789052
##    0.3    0.7427528597  1.118650  0.5051854  0.8404811
##    0.4    0.0009157003  1.041721  0.5046338  0.7792854
##    0.4    0.0021153864  1.041721  0.5046338  0.7792854
##    0.4    0.0048868171  1.041701  0.5046568  0.7792725
##    0.4    0.0112891816  1.040536  0.5055241  0.7784377
##    0.4    0.0260794744  1.037770  0.5077253  0.7763809
##    0.4    0.0602469699  1.032645  0.5127105  0.7723764
##    0.4    0.1391783180  1.031195  0.5174673  0.7707199
##    0.4    0.3215199740  1.052575  0.5169299  0.7847723
##    0.4    0.7427528597  1.154714  0.4927605  0.8732040
##    0.5    0.0009157003  1.041724  0.5046363  0.7793418
##    0.5    0.0021153864  1.041724  0.5046363  0.7793418
##    0.5    0.0048868171  1.041584  0.5047570  0.7792514
##    0.5    0.0112891816  1.040167  0.5057747  0.7782924
##    0.5    0.0260794744  1.036707  0.5086517  0.7758084
##    0.5    0.0602469699  1.031598  0.5138333  0.7720548
##    0.5    0.1391783180  1.031375  0.5185125  0.7713882
##    0.5    0.3215199740  1.061914  0.5153105  0.7922974
##    0.5    0.7427528597  1.198878  0.4660483  0.9171176
##    0.6    0.0009157003  1.041719  0.5046220  0.7793748
##    0.6    0.0021153864  1.041719  0.5046220  0.7793748
##    0.6    0.0048868171  1.041426  0.5048659  0.7791968
##    0.6    0.0112891816  1.039797  0.5060313  0.7781340
##    0.6    0.0260794744  1.035733  0.5095107  0.7753065
##    0.6    0.0602469699  1.030772  0.5147710  0.7717969
##    0.6    0.1391783180  1.032140  0.5192369  0.7722925
##    0.6    0.3215199740  1.073163  0.5126151  0.8028918
```

```
##   0.6    0.7427528597  1.241123  0.4414213  0.9560575
##   0.7    0.0009157003  1.041730  0.5045973  0.7794000
##   0.7    0.0021153864  1.041730  0.5045973  0.7794000
##   0.7    0.0048868171  1.041257  0.5049823  0.7791330
##   0.7    0.0112891816  1.039337  0.5063942  0.7779012
##   0.7    0.0260794744  1.034862  0.5102823  0.7749282
##   0.7    0.0602469699  1.030178  0.5155490  0.7717388
##   0.7    0.1391783180  1.034135  0.5190816  0.7738914
##   0.7    0.3215199740  1.086324  0.5085847  0.8153345
##   0.7    0.7427528597  1.280913  0.4263139  0.9878960
##   0.8    0.0009157003  1.041721  0.5046157  0.7794265
##   0.8    0.0021153864  1.041721  0.5046157  0.7794265
##   0.8    0.0048868171  1.041092  0.5050934  0.7790717
##   0.8    0.0112891816  1.038881  0.5067583  0.7776725
##   0.8    0.0260794744  1.034073  0.5109896  0.7746199
##   0.8    0.0602469699  1.029788  0.5161242  0.7718177
##   0.8    0.1391783180  1.036743  0.5185613  0.7760334
##   0.8    0.3215199740  1.101307  0.5030337  0.8295054
##   0.8    0.7427528597  1.325887  0.3902157  1.0229008
##   0.9    0.0009157003  1.041721  0.5046220  0.7794419
##   0.9    0.0021153864  1.041721  0.5046220  0.7794419
##   0.9    0.0048868171  1.040928  0.5052028  0.7790079
##   0.9    0.0112891816  1.038427  0.5071209  0.7774423
##   0.9    0.0260794744  1.033480  0.5115427  0.7744271
##   0.9    0.0602469699  1.029502  0.5166448  0.7718747
##   0.9    0.1391783180  1.039704  0.5178880  0.7784006
##   0.9    0.3215199740  1.118243  0.4953113  0.8451587
##   0.9    0.7427528597  1.362979  0.3593719  1.0513380
##   1.0    0.0009157003  1.041730  0.5046082  0.7794676
##   1.0    0.0021153864  1.041730  0.5046082  0.7794676
##   1.0    0.0048868171  1.040765  0.5053119  0.7789462
##   1.0    0.0112891816  1.037955  0.5075181  0.7771952
##   1.0    0.0260794744  1.032937  0.5120633  0.7742433
##   1.0    0.0602469699  1.029336  0.5171085  0.7718880
##   1.0    0.1391783180  1.043150  0.5168890  0.7809253
##   1.0    0.3215199740  1.137244  0.4845003  0.8624786
##   1.0    0.7427528597  1.396669  0.3561774  1.0784185
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 1 and lambda = 0.06024697.
```

#A través de este método hemos obtenido que el mejor modelo es en el que alpha = 1. #Como consecuencia emplearemos el método de regularización por Lasso.

```
cv_lasso <- cv.glmnet(nba_train_x, nba_train_y, alpha = 1)
min(cv_lasso$cvm)
```

```
## [1] 1.063743
```

#Obtenemos la media de MSE en la muestra de test de el modelo.

```
pred <- predict(cv_lasso, s = cv_lasso$lambda.min, nba_test_x)
media_error_modelo1 <- mean((nba_test_y - pred)^2)
media_error_modelo1
```

```
## [1] 1.151153
```

#Cuando reajustamos el modelo lasso podemos observar los valores de los #coeficientes pertenecientes a las variables explicativas.

#Comprobamos que en las variables edad, minutos jugados y la contribución al #equipo en las victorias el coeficiente es superior a 0.

```
predict(cv_lasso, type = "coefficients", s = cv_lasso$lambda.min)
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                               1
## (Intercept)     12.6316321848
## Age              0.0778039291
## NBA_DraftNumber -0.0213274865
## MP               0.0006533257
## USG.             .
## VORP             .
## WS               0.0604555148
## AST.             .
## TS.              .
```

#Comparación con un modelo teniendo en cuenta todas las variables #de la base de datos a excepción del nombre de los jugadores, #equipo y país de procedencia.

```
nba_train_x2 <- model.matrix(Salary ~. -Player -NBA_Country -Tm, data = nba_train)[, -1]
nba_train_y2 <- log(nba_train$Salary)
nba_test_x2 <- model.matrix(Salary~. -Player -NBA_Country -Tm, data = nba_test)[, -1]
nba_test_y2 <- log(nba_test$Salary)
dim(nba_train_x2)
```

```
## [1] 340  24
```

#Para obtener los valores de alpha y lambda utilizamos un cross-validation con K = 10.

```
train_control2 <- trainControl(method = "cv", number = 10)
caret_mod2 <- train( x = nba_train_x2, y = nba_train_y2, method = "glmnet",
                     preProc = c("center", "scale", "zv", "nzv"),
                     trControl = train_control2, tuneLength = 10 )

caret_mod2
```

```
## glmnet
##
## 340 samples
##  24 predictor
##
```

```
## Pre-processing: centered (24), scaled (24)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 305, 308, 304, 307, 305, 307, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda        RMSE      Rsquared   MAE
##   0.1    0.0003963848  1.110744  0.4458965  0.7966459
##   0.1    0.0009157003  1.110593  0.4460258  0.7965039
##   0.1    0.0021153864  1.109575  0.4465739  0.7966401
##   0.1    0.0048868171  1.109213  0.4462328  0.7993404
##   0.1    0.0112891816  1.108666  0.4454668  0.8024945
##   0.1    0.0260794744  1.103308  0.4476108  0.8039420
##   0.1    0.0602469699  1.089456  0.4562379  0.7971763
##   0.1    0.1391783180  1.080354  0.4631548  0.7956879
##   0.1    0.3215199740  1.071291  0.4744236  0.7960577
##   0.1    0.7427528597  1.072443  0.4916744  0.8046208
##   0.2    0.0003963848  1.111115  0.4456441  0.7971641
##   0.2    0.0009157003  1.110093  0.4463089  0.7961878
##   0.2    0.0021153864  1.109048  0.4468098  0.7967705
##   0.2    0.0048868171  1.108366  0.4464871  0.7996472
##   0.2    0.0112891816  1.107564  0.4459125  0.8037194
##   0.2    0.0260794744  1.096677  0.4519414  0.8016329
##   0.2    0.0602469699  1.082602  0.4621116  0.7940692
##   0.2    0.1391783180  1.073675  0.4705204  0.7937923
##   0.2    0.3215199740  1.056128  0.4943465  0.7907480
##   0.2    0.7427528597  1.088938  0.4949283  0.8190067
##   0.3    0.0003963848  1.111123  0.4455383  0.7972393
##   0.3    0.0009157003  1.109658  0.4465767  0.7961008
##   0.3    0.0021153864  1.108631  0.4469025  0.7969331
##   0.3    0.0048868171  1.107751  0.4465965  0.7998219
##   0.3    0.0112891816  1.106835  0.4460173  0.8047385
##   0.3    0.0260794744  1.091180  0.4557726  0.7992727
##   0.3    0.0602469699  1.079406  0.4656801  0.7931772
##   0.3    0.1391783180  1.062948  0.4806457  0.7895904
##   0.3    0.3215199740  1.050041  0.5072828  0.7871252
##   0.3    0.7427528597  1.116317  0.4925886  0.8426197
##   0.4    0.0003963848  1.110540  0.4460245  0.7968064
##   0.4    0.0009157003  1.109275  0.4468114  0.7959602
##   0.4    0.0021153864  1.108020  0.4472226  0.7968657
##   0.4    0.0048868171  1.107552  0.4464341  0.8003292
##   0.4    0.0112891816  1.105191  0.4467701  0.8049274
##   0.4    0.0260794744  1.086685  0.4591948  0.7966577
##   0.4    0.0602469699  1.076214  0.4687976  0.7919776
##   0.4    0.1391783180  1.052739  0.4916853  0.7847046
##   0.4    0.3215199740  1.057540  0.5062324  0.7916587
##   0.4    0.7427528597  1.152294  0.4841133  0.8744361
##   0.5    0.0003963848  1.110194  0.4462835  0.7964888
##   0.5    0.0009157003  1.108956  0.4469716  0.7958633
##   0.5    0.0021153864  1.107503  0.4475069  0.7969159
##   0.5    0.0048868171  1.107526  0.4462337  0.8010773
##   0.5    0.0112891816  1.102302  0.4484540  0.8043398
##   0.5    0.0260794744  1.083590  0.4621025  0.7945443
##   0.5    0.0602469699  1.070643  0.4738134  0.7896707
##   0.5    0.1391783180  1.043468  0.5028257  0.7796099
```

```
##   0.5    0.3215199740  1.065676  0.5061465  0.7970090
##   0.5    0.7427528597  1.196315  0.4616289  0.9160567
##   0.6    0.0003963848  1.110299  0.4462117  0.7964701
##   0.6    0.0009157003  1.109008  0.4468619  0.7959598
##   0.6    0.0021153864  1.107408  0.4474424  0.7971023
##   0.6    0.0048868171  1.107481  0.4460638  0.8017667
##   0.6    0.0112891816  1.098996  0.4505372  0.8032780
##   0.6    0.0260794744  1.081932  0.4639979  0.7943944
##   0.6    0.0602469699  1.065038  0.4786768  0.7875940
##   0.6    0.1391783180  1.036670  0.5121083  0.7762120
##   0.6    0.3215199740  1.074661  0.5061673  0.8051600
##   0.6    0.7427528597  1.238840  0.4388139  0.9542438
##   0.7    0.0003963848  1.110016  0.4463970  0.7962447
##   0.7    0.0009157003  1.108765  0.4469523  0.7958855
##   0.7    0.0021153864  1.107131  0.4474988  0.7972683
##   0.7    0.0048868171  1.107490  0.4459502  0.8025221
##   0.7    0.0112891816  1.095707  0.4527049  0.8021608
##   0.7    0.0260794744  1.081029  0.4652908  0.7943874
##   0.7    0.0602469699  1.059725  0.4834162  0.7859674
##   0.7    0.1391783180  1.034977  0.5159106  0.7754012
##   0.7    0.3215199740  1.085554  0.5051519  0.8155607
##   0.7    0.7427528597  1.278955  0.4258635  0.9865383
##   0.8    0.0003963848  1.109780  0.4465561  0.7961007
##   0.8    0.0009157003  1.108626  0.4469954  0.7958402
##   0.8    0.0021153864  1.106898  0.4475067  0.7974699
##   0.8    0.0048868171  1.107732  0.4456757  0.8036272
##   0.8    0.0112891816  1.093002  0.4546072  0.8010332
##   0.8    0.0260794744  1.080414  0.4661252  0.7944698
##   0.8    0.0602469699  1.054706  0.4880853  0.7842794
##   0.8    0.1391783180  1.037291  0.5154005  0.7771588
##   0.8    0.3215199740  1.099133  0.5021213  0.8279941
##   0.8    0.7427528597  1.323922  0.3911049  1.0213193
##   0.9    0.0003963848  1.109542  0.4466668  0.7958399
##   0.9    0.0009157003  1.108421  0.4471115  0.7957244
##   0.9    0.0021153864  1.106785  0.4474186  0.7977357
##   0.9    0.0048868171  1.108070  0.4452355  0.8047285
##   0.9    0.0112891816  1.090590  0.4563152  0.7998767
##   0.9    0.0260794744  1.079072  0.4671986  0.7939342
##   0.9    0.0602469699  1.050226  0.4925232  0.7826941
##   0.9    0.1391783180  1.040874  0.5137793  0.7798109
##   0.9    0.3215199740  1.114941  0.4970268  0.8421360
##   0.9    0.7427528597  1.362470  0.3591738  1.0516680
##   1.0    0.0003963848  1.109214  0.4468529  0.7956269
##   1.0    0.0009157003  1.108103  0.4472770  0.7955611
##   1.0    0.0021153864  1.106886  0.4471923  0.7981535
##   1.0    0.0048868171  1.108035  0.4450773  0.8054082
##   1.0    0.0112891816  1.088508  0.4579687  0.7985587
##   1.0    0.0260794744  1.075935  0.4698339  0.7924830
##   1.0    0.0602469699  1.045830  0.4969839  0.7809476
##   1.0    0.1391783180  1.044806  0.5120843  0.7828285
##   1.0    0.3215199740  1.133733  0.4873392  0.8588677
##   1.0    0.7427528597  1.396145  0.3582377  1.0788107
##
## RMSE was used to select the optimal model using the smallest value.
```

```
## The final values used for the model were alpha = 0.7 and lambda
##  = 0.1391783.
```

#En este caso obtenemos el valos para lambda=0.7

```
cv_elastic_net <- cv.glmnet(nba_train_x2, nba_train_y2, alpha = 0.7)
min(cv_elastic_net$cvm)
```

```
## [1] 1.091505
```

#Obtenemos la media de MSE para la muestra de test.

```
pred2 <- predict(cv_elastic_net, s = cv_elastic_net$lambda.min, nba_test_x2)
media_error_modelo2 <- mean((nba_test_y2 - pred2)^2)
media_error_modelo2
```

```
## [1] 1.163816
```

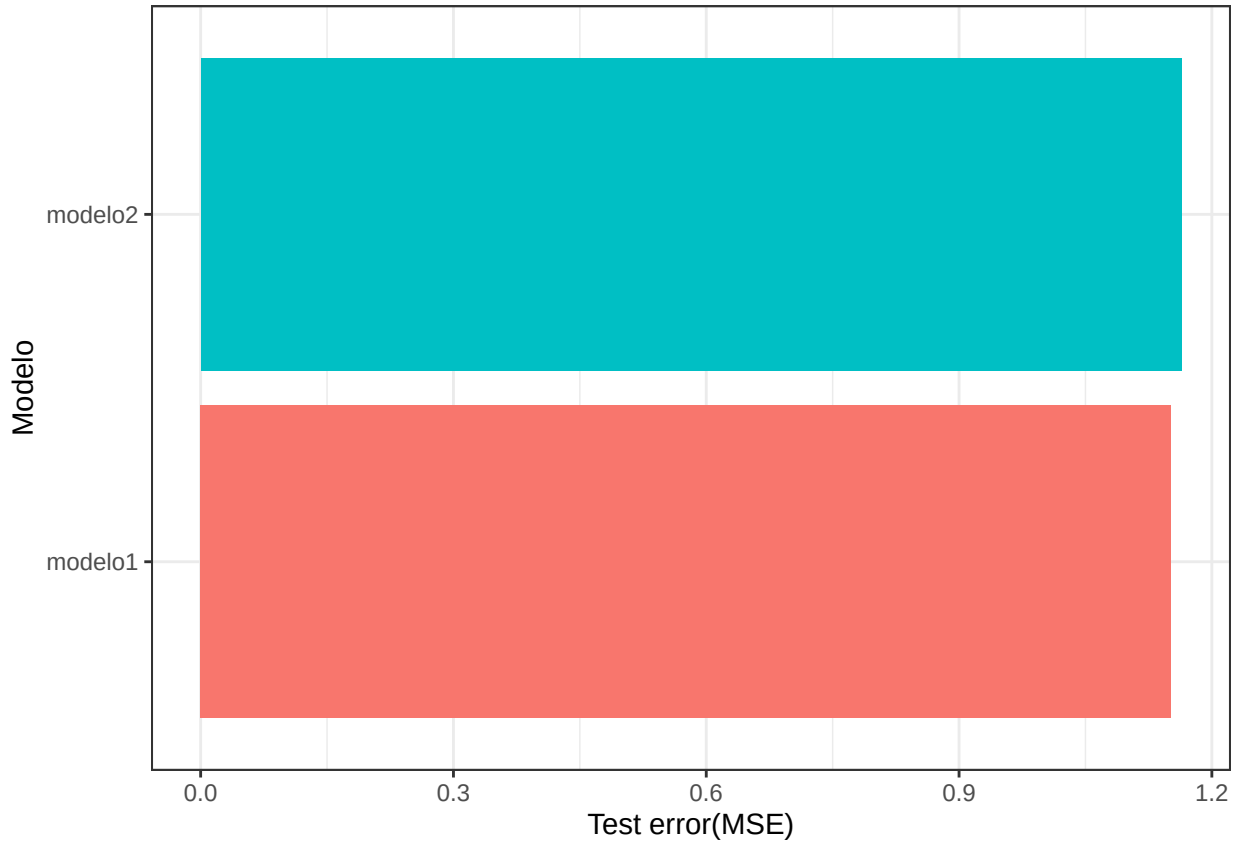
#Al reajustar el modelo observamos los valores de los coeficientes de las variables #explicativas, los cuales se mantienen con un coeficiente superior a 0 las variables de #edad, los minutos jugados, contribución a las victorias del equipo, #la contribución en defensa a las victorias del equipo y el porcentaje de rebotes #defensivos.

```
predict(cv_elastic_net, type = "coefficients", s = cv_elastic_net$lambda.min)
```

```
## 25 x 1 sparse Matrix of class "dgCMatrix"
##                               1
## (Intercept)     12.6923103967
## NBA_DraftNumber -0.0187048277
## Age              0.0679827921
## G                .
## MP               0.0006226728
## PER              .
## TS.              .
## X3PAr            .
## FTr              .
## ORB.             .
## DRB.             0.0118872397
## TRB.             .
## AST.             .
## STL.             .
## BLK.             .
## TOV.             .
## USG.             .
## OWS              .
## DWS              0.0129072904
## WS               0.0445062130
## WS.48            .
## OBPM             .
## DBPM             .
## BPM              .
## VORP             .
```

#Comparamos la media de error entre los dos modelos.

```
modelo <- c("modelo1", "modelo2")
test.MSE <- c(media_error_modelo1, media_error_modelo2)
comparacion <- data.frame(modelo, test.MSE)
ggplot(data = comparacion, aes(x = reorder(x = modelo, X = test.MSE), y = test.MSE)) +
  geom_bar(stat = "identity", aes(fill = modelo)) + labs(x = "Modelo", y = "Test error(MSE)") +
  theme_bw() + coord_flip() + theme(legend.position = "none")
```



#Concluímos afirmando que el mejor modelo obtenido es el planteado ya que #la media de MSE es inferior al segundo modelo.