Nba - Modelo Ridge, Lasso y Elastic Net

1.Introducción al trabajo

El próximo documento, se tratará de encontrar el modelo predictivo que mayor capacidad de predicción tenga acerca de los salarios de los jugadores de la NBA en función de otras 29 variables.

El dataset consta, tras eliminar los valores nulos, de 485 observaciones y 28 variables, las cuales se resumirán en las siguientes líneas del documento:

1:Player Nombre del jugador.

2:Salary Salario del jugador.

3:NBA_Country Nacionalidad del jugador.

4:NBA_DraftNumber Número del draft.

5:Age Edad del jugador.

6:Tm Abreviatura del equipo

7:G Partidos jugados

8:MP Minutos jugados

9:PER Eficiencia de jugador

10:TS% Porcentaje de tiro

11:3PAr % de triples

12:FTr % de tiros libres

13:ORB% % Rebotes Ofensivos ganados

14:DRB% % Rebotes defensivos ganados

15:TRB% % Rebotes totales

16:AST% % Asistencia

17:STL% % Robos

18:BLK% % Bloqueos

19:TOV% % Robo previo a tiro

20:USG% % de participacion en jugadas

21:OWS Acciones en ataque acertadas

22:DWS Acciones defensivas acertadas

23:WS Victorias contribuidas

24:WS/48 Ratio de contribución por partido

25:OBPM +/- rendimiento respecto al equipo (cada 100 jugadas en ataque)

26:DBPM +/- rendimiento respecto al equipo (cada 100 jugadas en defensa)

27:BPM +/- rendimiento respecto al equipo (cada 100 posesiones generales)

28:VORP Valor respecto a jugador involucrado en el cambio

2.Importación de librerías y Dataset

```
library(glmnet)
library(tidyverse)
library(fBasics)
library(car)
library(dplyr)
library(ggplot2)
library(knitr)
library(MASS)
library(corrplot)
library(PerformanceAnalytics)
library(gvlma)
library(tinytex)
library(devtools)
```

```
nba <- read_delim("nba_2.csv", ";", escape_double = FALSE,
                  trim_ws = TRUE)

attach(nba)
names(nba)
```

```
##  [1] "Player"          "Salary"          "NBA_Country"     "NBA_DraftNumber"
##  [5] "Age"             "Tm"              "G"               "MP"
##  [9] "PER"             "TS"              "PAr"             "FTr"
## [13] "ORB"             "DRB"             "TRB"             "AST"
## [17] "STL"             "BLK"             "TOV"             "USG"
## [21] "OWS"             "DWS"             "WS"              "WS_1"
## [25] "OBPM"            "DBPM"            "BPM"             "VORP"
```

```
head(nba)
```

```
## # A tibble: 6 x 28
##   Player Salary NBA_Country NBA_DraftNumber   Age Tm        G    MP   PER    TS
##   <chr>   <dbl> <chr>                <dbl> <dbl> <chr> <dbl> <dbl> <dbl> <dbl>
## 1 Zhou ~ 8.16e5 China                   43    22 HOU      16    87   0.6 0.303
## 2 Zaza ~ 3.48e6 Georgia                 42    33 GSW      66   937  16.8 0.608
## 3 Zach ~ 1.23e7 USA                     19    36 SAC      59  1508  17.3 0.529
## 4 Zach ~ 3.20e6 USA                     13    22 CHI      24   656  14.6 0.499
## 5 Zach ~ 3.06e6 USA                     10    20 POR      62   979   8.2 0.487
## 6 Yogi ~ 1.31e6 USA                     62    24 DAL      79  2238  11.5 0.543
## # ... with 18 more variables: PAr <dbl>, FTr <dbl>, ORB <dbl>, DRB <dbl>,
## #   TRB <dbl>, AST <dbl>, STL <dbl>, BLK <dbl>, TOV <dbl>, USG <dbl>,
## #   OWS <dbl>, DWS <dbl>, WS <dbl>, WS_1 <dbl>, OBPM <dbl>, DBPM <dbl>,
## #   BPM <dbl>, VORP <dbl>
```

```
summary(nba)
```

```
##     Player              Salary         NBA_Country        NBA_DraftNumber
##  Length:485         Min.   :   46080   Length:485         Min.   : 1.00
##  Class :character   1st Qu.: 1471382   Class :character   1st Qu.:11.00
##  Mode  :character   Median : 3202217   Mode  :character   Median :25.00
##                     Mean   : 6636507                      Mean   :29.45
##                     3rd Qu.:10000000                      3rd Qu.:47.00
##                     Max.   :34682550                      Max.   :62.00
##
##       Age             Tm                  G               MP
##  Min.   :19.00   Length:485         Min.   : 1.00   Min.   :   1
##  1st Qu.:23.00   Class :character   1st Qu.:29.00   1st Qu.: 381
##  Median :26.00   Mode  :character   Median :59.00   Median :1134
##  Mean   :26.26                      Mean   :50.17   Mean   :1154
##  3rd Qu.:29.00                      3rd Qu.:71.00   3rd Qu.:1819
##  Max.   :41.00                      Max.   :79.00   Max.   :2898
##
##       PER               TS              PAr              FTr
##  Min.   :-41.10   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:  9.80   1st Qu.:0.5055   1st Qu.:0.1670   1st Qu.:0.1550
##  Median : 13.20   Median :0.5450   Median :0.3460   Median :0.2310
##  Mean   : 13.26   Mean   :0.5354   Mean   :0.3374   Mean   :0.2634
##  3rd Qu.: 16.50   3rd Qu.:0.5825   3rd Qu.:0.4810   3rd Qu.:0.3195
##  Max.   :134.10   Max.   :1.5000   Max.   :1.0000   Max.   :5.3330
##                   NA's   :2        NA's   :2        NA's   :2
##       ORB              DRB              TRB              AST
##  Min.   : 0.000   Min.   : 0.00   Min.   : 0.000   Min.   : 0.00
##  1st Qu.: 1.800   1st Qu.:10.20   1st Qu.: 6.200   1st Qu.: 6.90
##  Median : 3.200   Median :14.00   Median : 8.700   Median : 9.90
##  Mean   : 4.874   Mean   :14.95   Mean   : 9.908   Mean   :12.95
##  3rd Qu.: 7.000   3rd Qu.:18.80   3rd Qu.:13.300   3rd Qu.:17.60
##  Max.   :35.900   Max.   :37.60   Max.   :26.500   Max.   :49.40
##
##       STL              BLK              TOV              USG
##  Min.   : 0.000   Min.   : 0.000   Min.   : 0.00   Min.   : 0.0
##  1st Qu.: 1.000   1st Qu.: 0.600   1st Qu.: 9.90   1st Qu.:15.0
##  Median : 1.500   Median : 1.200   Median :12.50   Median :17.9
##  Mean   : 1.529   Mean   : 1.713   Mean   :13.14   Mean   :18.9
##  3rd Qu.: 1.900   3rd Qu.: 2.200   3rd Qu.:15.75   3rd Qu.:22.2
##  Max.   :12.500   Max.   :13.400   Max.   :66.70   Max.   :45.1
##                                    NA's   :2
##       OWS              DWS               WS              WS_1
##  Min.   :-2.300   Min.   :0.000   Min.   :-1.200   Min.   :-1.06300
##  1st Qu.: 0.000   1st Qu.:0.300   1st Qu.: 0.300   1st Qu.: 0.04000
##  Median : 0.800   Median :1.000   Median : 1.800   Median : 0.08300
##  Mean   : 1.275   Mean   :1.176   Mean   : 2.455   Mean   : 0.07996
##  3rd Qu.: 2.000   3rd Qu.:1.800   3rd Qu.: 3.600   3rd Qu.: 0.12300
##  Max.   :11.400   Max.   :5.600   Max.   :15.000   Max.   : 2.71300
##
##       OBPM              DBPM              BPM              VORP
##  Min.   :-36.500   Min.   :-14.3000   Min.   :-49.20   Min.   :-1.3000
##  1st Qu.: -2.700   1st Qu.: -1.7000   1st Qu.: -3.60   1st Qu.:-0.1000
```

```
##  Median : -1.100   Median : -0.4000   Median : -1.30   Median : 0.1000
##  Mean   : -1.271   Mean   : -0.4895   Mean   : -1.76   Mean   : 0.5988
##  3rd Qu.:  0.400   3rd Qu.:  1.0000   3rd Qu.:  0.50   3rd Qu.: 0.9000
##  Max.   : 68.700   Max.   :  6.8000   Max.   : 54.40   Max.   : 8.6000
##
```

3. Limpieza del Dataset

```
#Limpieza de datos
nba <- na.omit(nba)

#Convertir la variable Tm (Teams) de cualitativa a Cuantitativa
unique(nba$Tm)
```

```
##  [1] "HOU" "GSW" "SAC" "CHI" "POR" "DAL" "BOS" "MEM" "DEN" "TOT" "LAC" "ORL"
## [13] "MIA" "IND" "LAL" "MIN" "PHO" "ATL" "CLE" "NYK" "CHO" "MIL" "SAS" "UTA"
## [25] "NOP" "WAS" "PHI" "BRK" "OKC" "DET" "TOR"
```

```
levels <- c('HOU', 'GSW', 'SAC', 'CHI', 'POR',
            'DAL', 'BOS', 'MEM', 'DEN',
            'TOT', 'LAC', 'ORL', 'MIA',
            'IND', 'LAL', 'MIN', 'PHO',
            'ATL', 'CLE', 'NYK', 'CHO',
            'MIL', 'SAS', 'UTA', 'NOP',
            'WAS', 'PHI', 'BRK', 'OKC',
            'DET', 'TOR')

nba$Team <- match(nba$Tm, levels)

#Convertir la variable NBA_Country (Países) de cualitativa a Cuantitativa
unique(nba$NBA_Country)
```

```
##  [1] "China"            "Georgia"          "USA"              "Canada"
##  [5] "Spain"            "France"           "Czech Republic"   "Russia"
##  [9] "South Sudan"      "Switzerland"      "New Zealand"      "Haiti"
## [13] "Democratic Re_"   "Tunisia"          "Brazil"           "Germany"
## [17] "Australia"        "Cameroon"         "Israel"           "Turkey"
## [21] "United Kingdo..." "Montenegro"       "Serbia"           "Argentina"
## [25] "Bosnia"           "Lithuania"        "Croatia"          "Italy"
## [29] "Poland"           "Dominican Rep..." "Finland"          "Latvia"
## [33] "Bosnia & Herz..." "Sweden"           "Ukraine"          "Austria"
## [37] "Puerto Rico"      "Senegal"          "Slovenia"         "Greece"
## [41] "Democratic Re..." "Mali"             "Bahamas"          "Egypt"
```

```
niveles <- c('China', 'Georgia', 'USA', 'Canada', 'Spain',
             'France', 'Czech Republic', 'Russia', 'South Sudan',
             'Switzerland', 'New Zealand', 'Haiti', 'Democratic Re_',
             'Tunisia', 'Brazil', 'Germany', 'Australia',
             'Cameroon', 'Israel', 'Turkey', 'United Kingdo...',
             'Montenegro', 'Serbia', 'Argentina', 'Bosnia',
```

```
            'Lithuania', 'Croatia', 'Italy', 'Poland',
            'Dominican Rep...', 'Finland', 'Latvia', 'Bosnia & Herz...',
            'Sweden', 'Ukraine', 'Austria', 'Puerto Rico',
            'Senegal', 'Slovenia', 'Greece', 'Democratic Re...',
            'Mali', 'Bahamas', 'Egypt')
nba$Pais <- match(nba$NBA_Country, niveles)
```

4. Ejemplificación Modelo Lineal con todas las variables

```
#Realizamos el modelo con todas las variables numericas y las cualitativas convertidas a cuantitativas

model <- lm(log(Salary, base = 10) ~ NBA_DraftNumber + Age + G + MP + PER + PAr + FTr + ORB + DRB + TRB
              AST + STL + BLK + TOV + USG + OWS + DWS + WS + WS_1 + OBPM + DBPM + BPM + VORP + Team + Pa
model
```

```
##
## Call:
## lm(formula = log(Salary, base = 10) ~ NBA_DraftNumber + Age +
##     G + MP + PER + PAr + FTr + ORB + DRB + TRB + AST + STL +
##     BLK + TOV + USG + OWS + DWS + WS + WS_1 + OBPM + DBPM + BPM +
##     VORP + Team + Pais, data = nba)
##
## Coefficients:
##     (Intercept)  NBA_DraftNumber              Age                G
##       5.5371834       -0.0093861        0.0438368       -0.0013098
##              MP              PER              PAr              FTr
##       0.0003702       -0.0639629       -0.3810026       -0.0756245
##             ORB              DRB              TRB              AST
##      -0.0592330       -0.0449221        0.1260209        0.0010175
##             STL              BLK              TOV              USG
##       0.0035687        0.0163819       -0.0033433        0.0219521
##             OWS              DWS               WS             WS_1
##      -0.0264411       -0.0941729        0.0362625        1.0697027
##            OBPM             DBPM              BPM             VORP
##      -0.1059997       -0.1499403        0.1729863       -0.0108669
##            Team             Pais
##       0.0037295        0.0031131
```

5. División dataset Training vs Testing

Dividimos el datase de nba en dos parte, una, que representa el 70% de los datos, para entrenar a los modelos y luego otra, que representa el 30% que servirá para poder testear la eficacia del modelo.

```
library(rsample)

set.seed(2020)

#Dividir el Dataset
```

```r
nba_split <- initial_split(nba, prop = .7)

dim(nba)
```

```
## [1] 483  30
```

```r
#Practica (70%)
nba_training_data <- training(nba_split)
dim(nba_training_data)
```

```
## [1] 339  30
```

```r
#Test (30%)
nba_testing_data <- testing(nba_split)
dim(nba_testing_data)
```

```
## [1] 144  30
```

```r
# Create training and testing feature model matrices and response vectors.
# we use model.matrix(...)[, -1] to discard the intercept
nba_training_data_a <- model.matrix(Player ~ ., nba_training_data)[, -1]
nba_training_data_b <- log(nba_training_data$Salary)

nba_testing_data_a <- model.matrix(Player ~ ., nba_testing_data)[, -1]
nba_testing_data_b <- log(nba_testing_data$Salary)
```

6. Regresiones

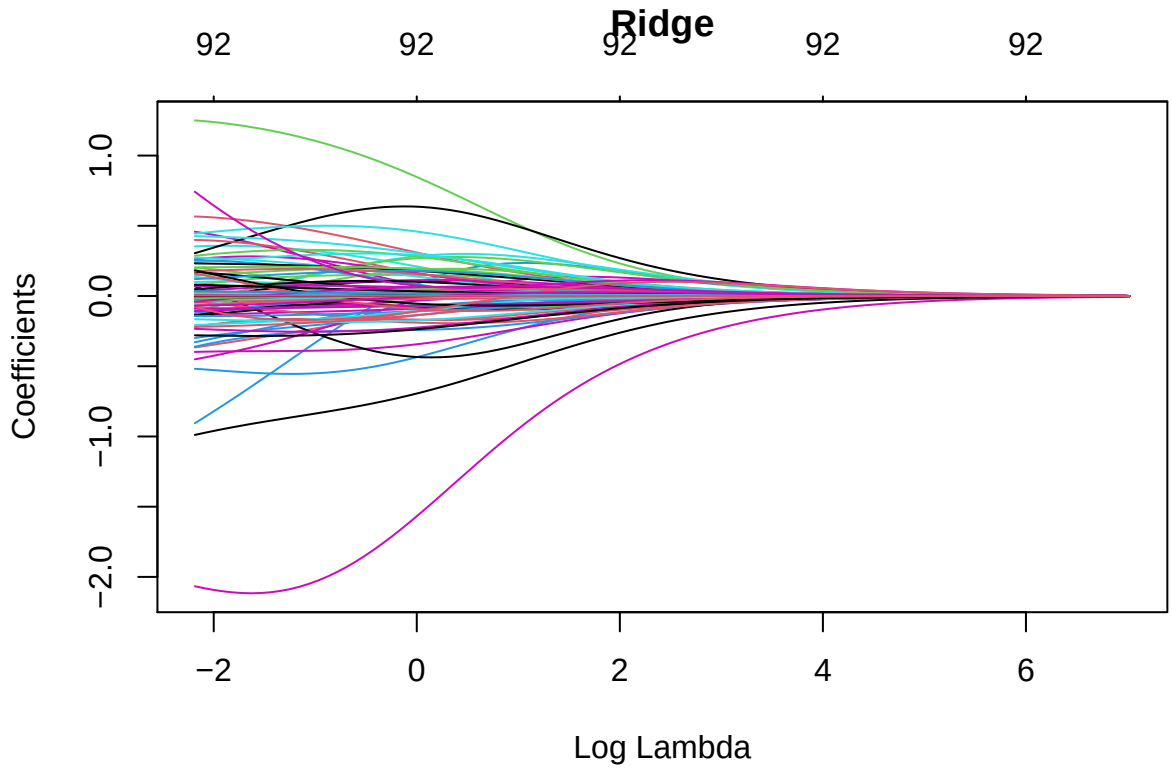
6.1. Ridge Regression

A continuación, vamos a realizar la Regresión de Ridge, esta regresión busca estimadores de coecifientes que reducen el RSS, pero penaliza aquellos coecifientes que se acerquen a cero, por eso, esta estimación tiene el efecto de reducir las estimacions del coecifiente a cero.

En este apartado, seguiremos la siguiente estructura:

- Visualización de las estimaciones de los coecifientes de las variables con la función glmnet.
- Visualización de los MSE (Mean Squared Error) para cada uno de los valores de Lambda.
- Elegir el modelo con menor valor de MSE y sumarle una desviación típica a la derecha (debido a que se favorece a los modelos parsimoniosos, es decir, más sencillos).
- Visualización del valor de LOG(Lambda) escogido junto con el valor de las estimaciones de los coecifientes.
- Visualización de la tabla con el top 25 variables que más influyen en el modelo de Ridge.

```
nba_ridge <- glmnet(
  x = nba_training_data_a,
  y = nba_training_data_b,
  alpha = 0
)

plot(nba_ridge, xvar = 'lambda', main = 'Ridge')
```




```
#Encabezado
nba_ridge$lambda %>% head()
```

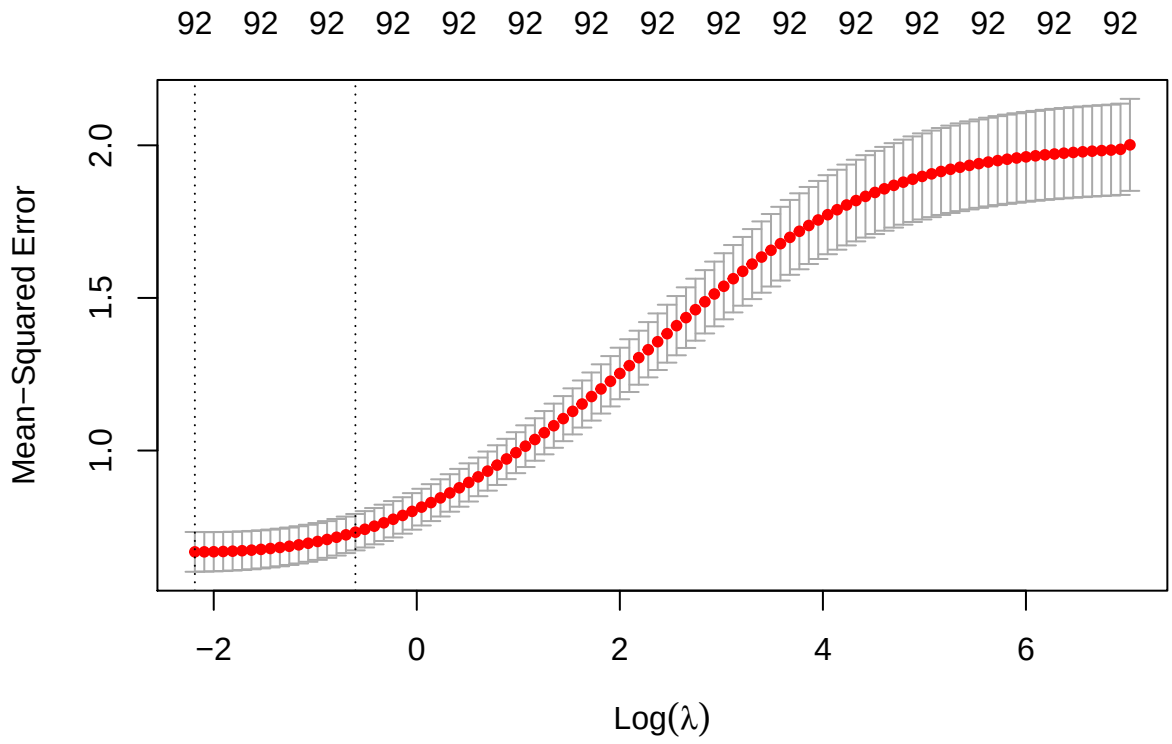
```
## [1] 1123.6650 1023.8417  932.8864  850.0113  774.4987  705.6944
```

```
names(nba_ridge)
```

```
##  [1] "a0"        "beta"      "df"        "dim"       "lambda"    "dev.ratio"
##  [7] "nulldev"   "npasses"   "jerr"      "offset"    "call"      "nobs"
```

```
#------------
#Tuning A - Ver modelo con menor error al cuadrado en funcion de su lambda
#------------

nba_ridge_cv <- cv.glmnet(
```

```
  x = nba_training_data_a,
  y = nba_training_data_b,
  alpha = 0
)

# plot results
plot(nba_ridge_cv)
```




```
#Minimo
min(nba_ridge_cv$cvm) #0.6679
```

```
## [1] 0.6679933
```

```
#Valor de lambda para el minimo
nba_ridge_cv$lambda.min  #0.112
```

```
## [1] 0.1123665
```

```
log(nba_ridge_cv$lambda.min) #-2.185
```

```
## [1] -2.185989
```

```r
#Minimo LAMBDA + 1 desv estandar
nba_ridge_cv$cvm[nba_ridge_cv$lambda == nba_ridge_cv$lambda.1se]
```

```
## [1] 0.7325181
```

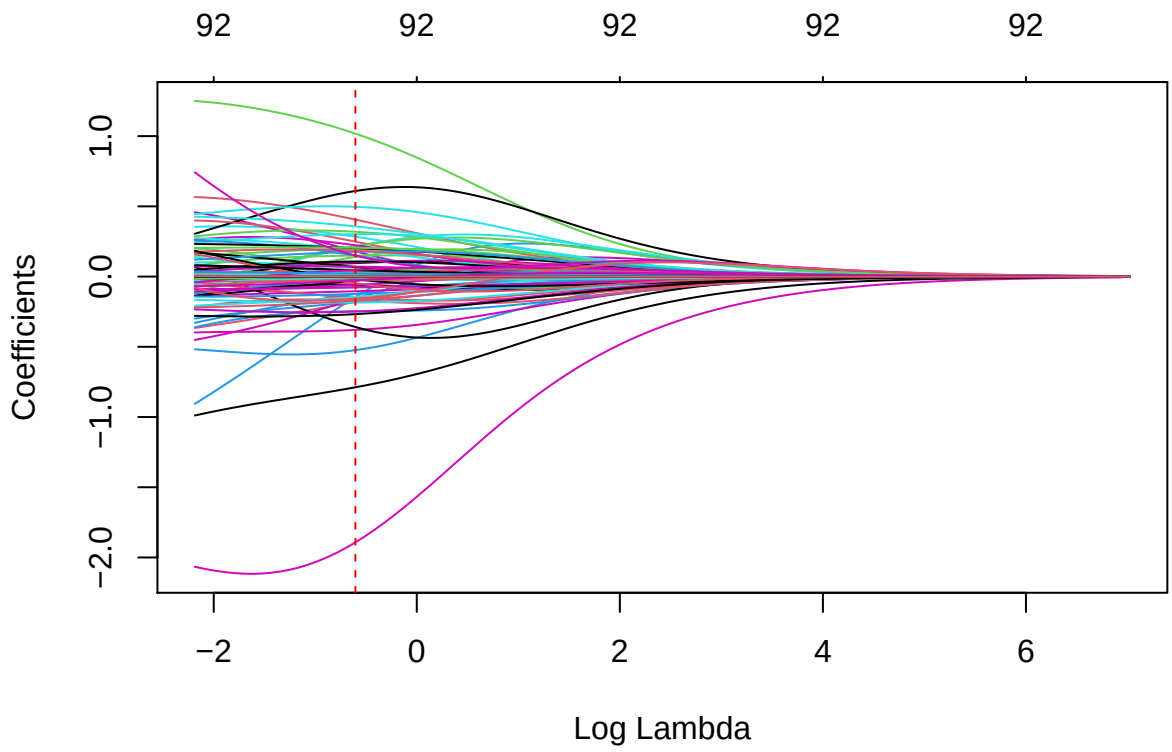
```r
nba_ridge_cv$lambda.1se
```

```
## [1] 0.5463935
```

```r
#Valor Lmabda
log(nba_ridge_cv$lambda.1se)
```

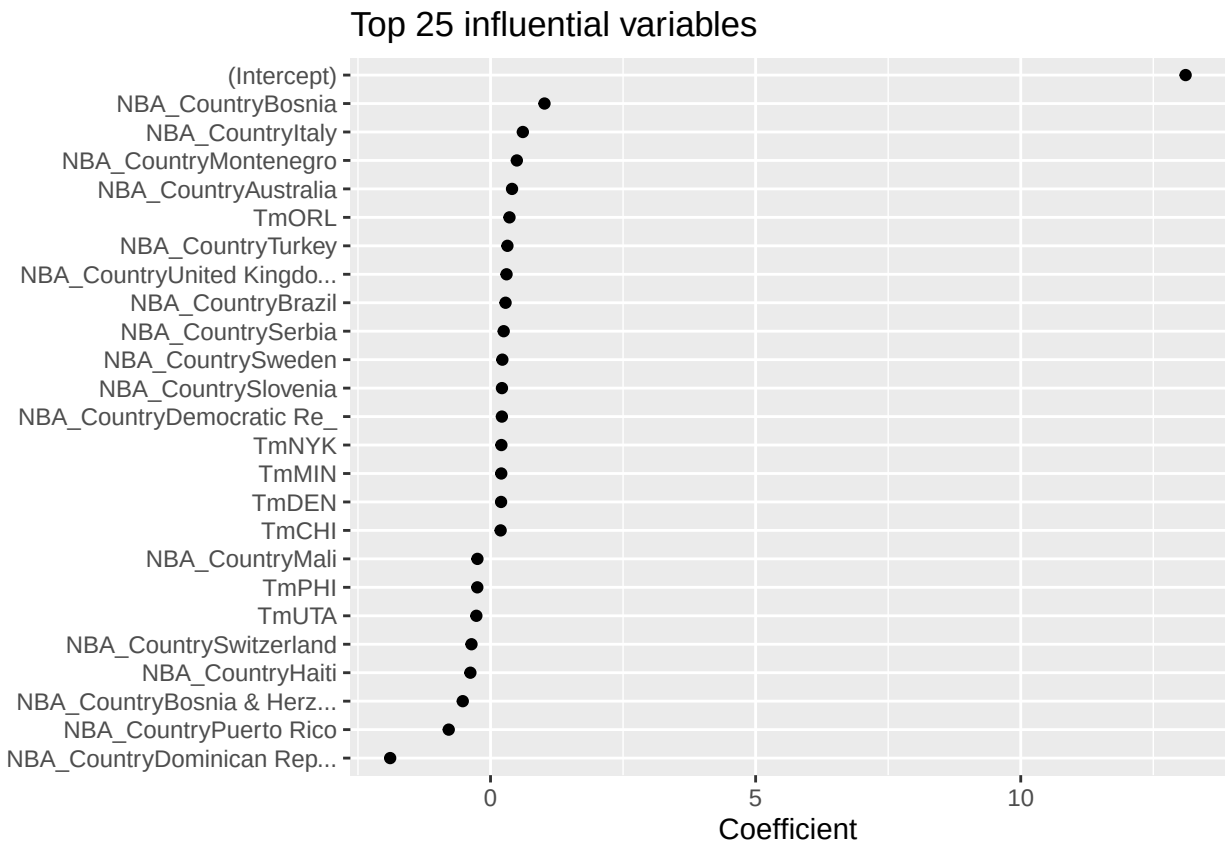
```
## [1] -0.6044158
```

```r
#Grafica del valor de los coecifientes para el valor de lambda
plot(nba_ridge, xvar = "lambda")
abline(v = log(nba_ridge_cv$lambda.1se), col = "red", lty = "dashed")
```




```r
#Tabla top 25 coecifientes con mas influencia
coef(nba_ridge_cv, s = "lambda.1se") %>%
  broom::tidy() %>%
  top_n(25, wt = abs(value)) %>%
  ggplot(aes(value, reorder(row, value))) +
```

```
  geom_point() +
  ggtitle("Top 25 influential variables") +
  xlab("Coefficient") +
  ylab(NULL)
```



Podemos observar que para este modelo, el valor de lambda que genera un menor RSE es cuando Log(Lambda) es igual a -0.51 (aunque este puede variar en función de la muestra).

También se muestra el top 25 variables que más afectan al modelo y con un mayor valor de coecifientes (sin tener en cuenta 'Intercept').

6.2 Lasso Regression

La Regresión Lasso es un método de análisis de regresión que realiza selección de variables y regularización para mejorar la exactitud e interpretabilidad del modelo estadístico producido por este.

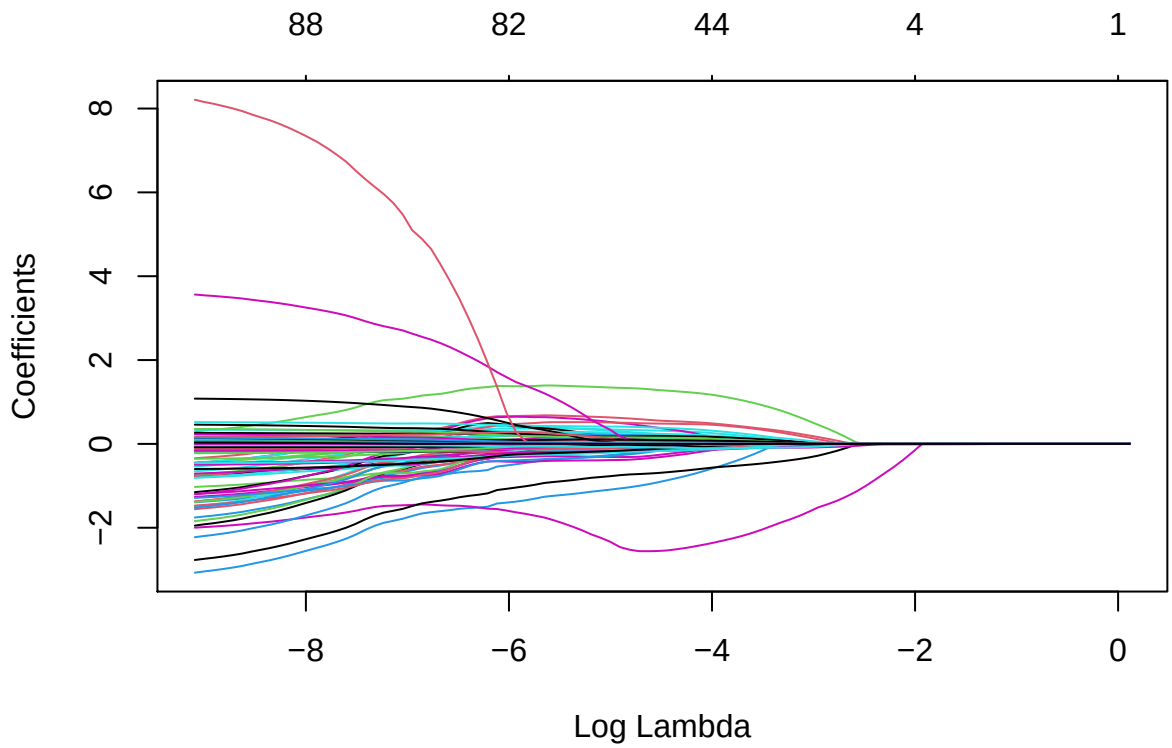
La penalización de la Regresión Lasso se da por el valor absoluto de los parámetro, lo que provoca que según Log(Lambda) se hace más estricto, es decir, aumenta su valor positivamente, se reduce el número de variables que conforman el modelo.

En este apartado, seguiremos la siguiente estructura:

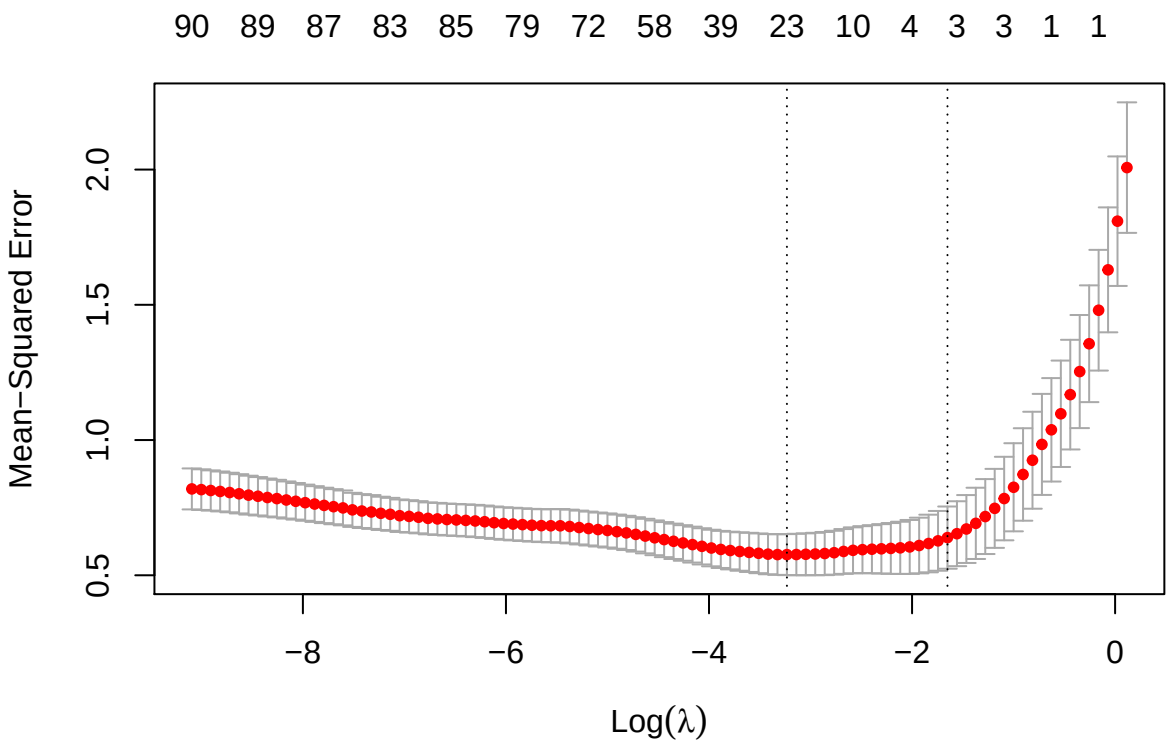
- Visualización de las estimaciones de los coecifientes de las variables con la función glmnet.
- Visualización de los MSE (Mean Squared Error) para cada uno de los valores de Lambda.

- Elegir el modelo con menor valor de MSE y sumarle una desviación típica a la derecha (debido a que se favorece a los modelos parsimoniosos, es decir, más sencillos).
- Visualización del valor de LOG(Lambda) escogido junto con el valor de las estimaciones de los coecifientes.
- Visualización de la tabla con las variables que más influyen en el modelo de Lasso

```
#Calculo coecifientes en funcion de lambda
nba_lasso <- glmnet(
  x = nba_training_data_a,
  y = nba_training_data_b,
  alpha = 1
)

plot(nba_lasso, xvar = "lambda")
```




```
#Calculo medio de residuos en funcion del logaritmo de lambda

nba_lasso_cv <- cv.glmnet(
  x = nba_training_data_a,
  y = nba_training_data_b,
  alpha = 1
)

plot(nba_lasso_cv)
```




```
#Valor de lambda para el minimo MSE
min(nba_lasso_cv$cvm)
```

```
## [1] 0.5764144
```

```
#Valor minimo de lambda
nba_lasso_cv$lambda.min
```

```
## [1] 0.03945403
```

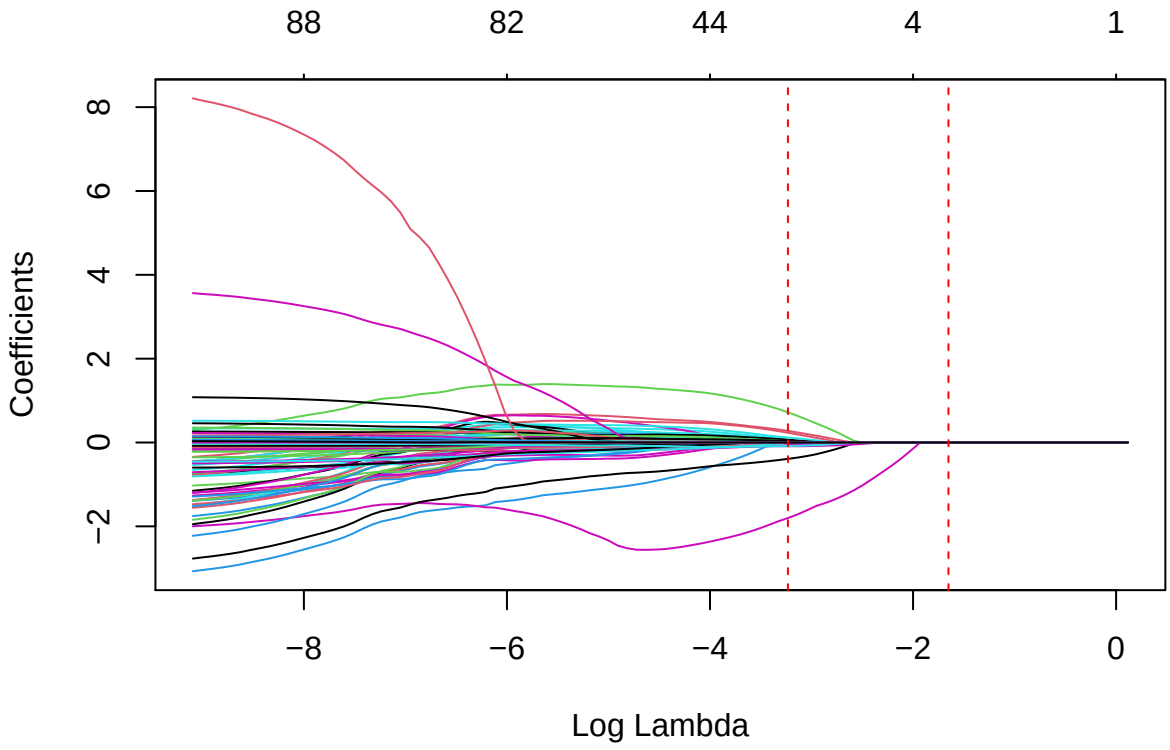
```
#Valor minimo de lambda + 1 desviacion estandar
nba_lasso_cv$cvm[nba_lasso_cv$lambda == nba_lasso_cv$lambda.1se]
```

```
## [1] 0.639276
```

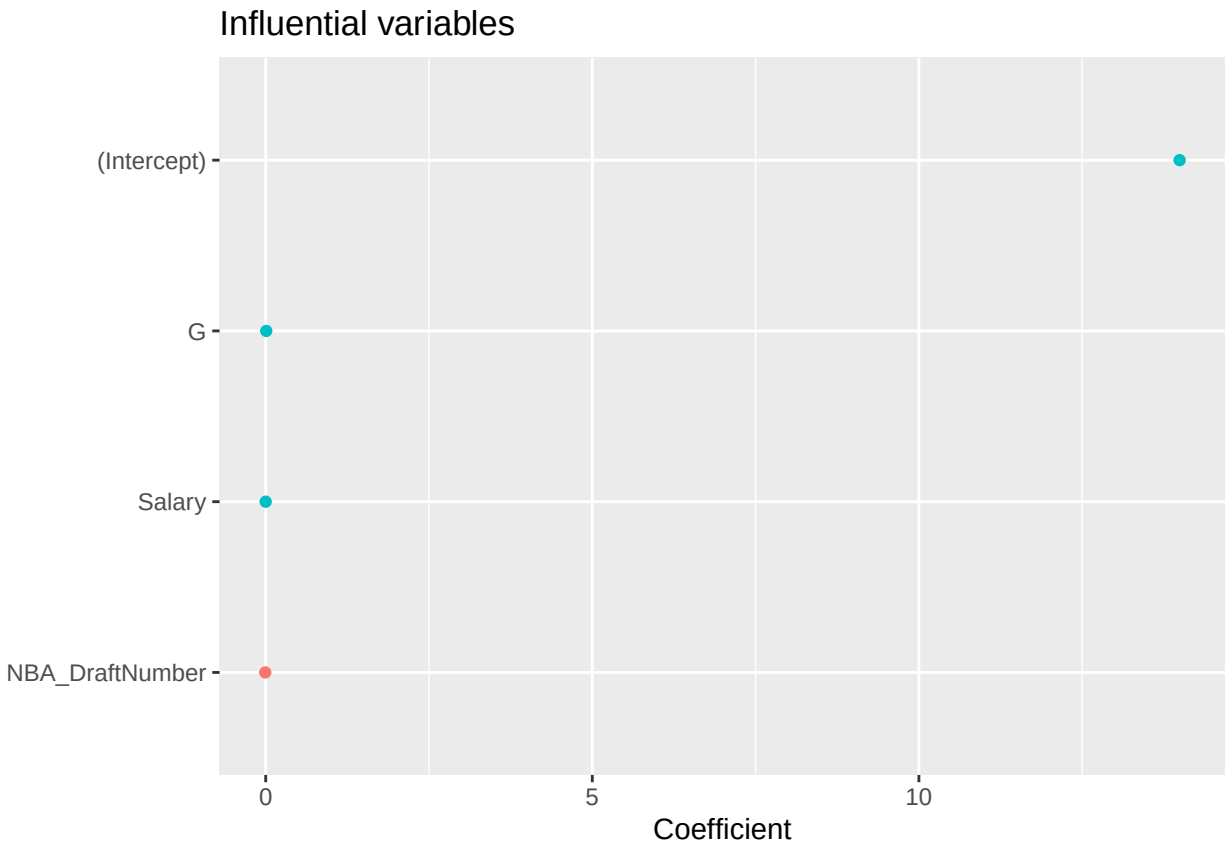
```
log(nba_lasso_cv$lambda.1se)
```

```
## [1] -1.651045
```

```
#Grafica del valor de los coecifientes en funcion de lambda

plot(nba_lasso, xvar = 'lambda')
abline(v = log(nba_lasso_cv$lambda.min), col = "red", lty = "dashed")
abline(v = log(nba_lasso_cv$lambda.1se), col = "red", lty = "dashed")
```




```
#Tabla coecifientes con mas influencia


coef(nba_lasso_cv, s = "lambda.1se") %>%
  tidy() %>%
  ggplot(aes(value, reorder(row, value), color = value > 0)) +
  geom_point(show.legend = FALSE) +
  ggtitle("Influential variables") +
  xlab("Coefficient") +
  ylab(NULL)
```



Podemos observar que para este modelo, el valor de lambda que genera un menor RSE es cuando Log(Lambda) es igual a -1.65 (aunque este puede variar en función de la muestra).

También se muestran las variables que más afectan al modelo y con un mayor valor de coecifientes (sin tener en cuenta 'Intercept').

6.3. Elastic Net Regression

La Regresión de Elastic Net no es más que la combinación lineal de ambos modelos.

La Regresión de Lasso y Ridge no son más que dos casos particulares de Elastic Net donde Alpha = 1 (Lasso) o Alpha = 0 (Ridge).

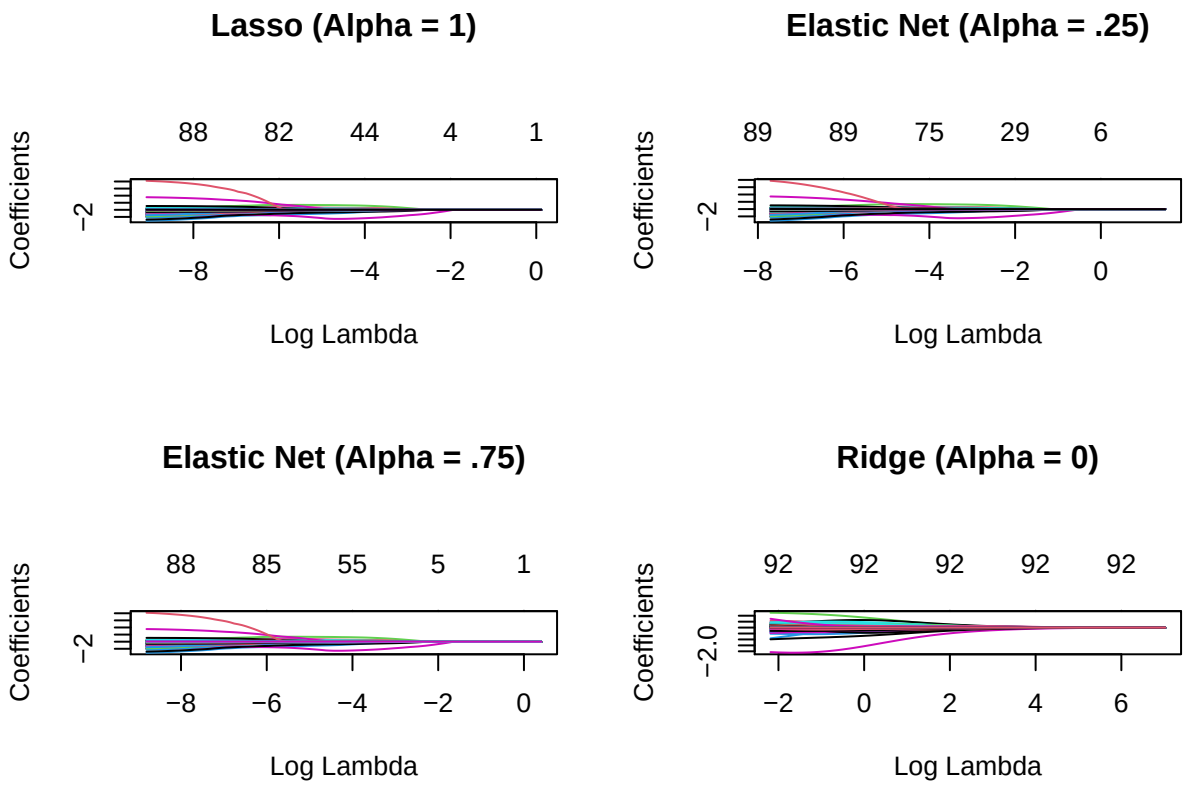
Elastic Net ajusta el valor de alpha para realizar la combiación de Lasso y Ridge.

La estructura que se seguirá en este apartado es:

- Visualización de los estimadores de los coecifientes para diferentes valores de Lambda (0, 0.25, 0.75 y 1)
- Visualización de los MSE (minimos valores de errores al cuadrado), para cada uno de los valores de alpha (de 0 a 1 en intervalos de 0.1)

```
lasso    <- glmnet(nba_training_data_a, nba_training_data_b, alpha = 1.0)
elastic1 <- glmnet(nba_training_data_a, nba_training_data_b, alpha = 0.25)
elastic2 <- glmnet(nba_training_data_a, nba_training_data_b, alpha = 0.75)
ridge    <- glmnet(nba_training_data_a, nba_training_data_b, alpha = 0.0)
```

```r
par(mfrow = c(2, 2), mar = c(6, 4, 6, 2) + 0.1)
plot(lasso, xvar = "lambda", main = "Lasso (Alpha = 1)\n\n\n")
plot(elastic1, xvar = "lambda", main = "Elastic Net (Alpha = .25)\n\n\n")
plot(elastic2, xvar = "lambda", main = "Elastic Net (Alpha = .75)\n\n\n")
plot(ridge, xvar = "lambda", main = "Ridge (Alpha = 0)\n\n\n")
```




```r
#Ajuste del modelo (tuning)

# maintain the same folds across all models
fold_id <- sample(1:10, size = length(nba_training_data_b), replace=TRUE)

# search across a range of alphas
tuning_grid <- tibble::tibble(
  alpha      = seq(0, 1, by = .1),
  mse_min    = NA,
  mse_1se    = NA,
  lambda_min = NA,
  lambda_1se = NA
)
tuning_grid
```

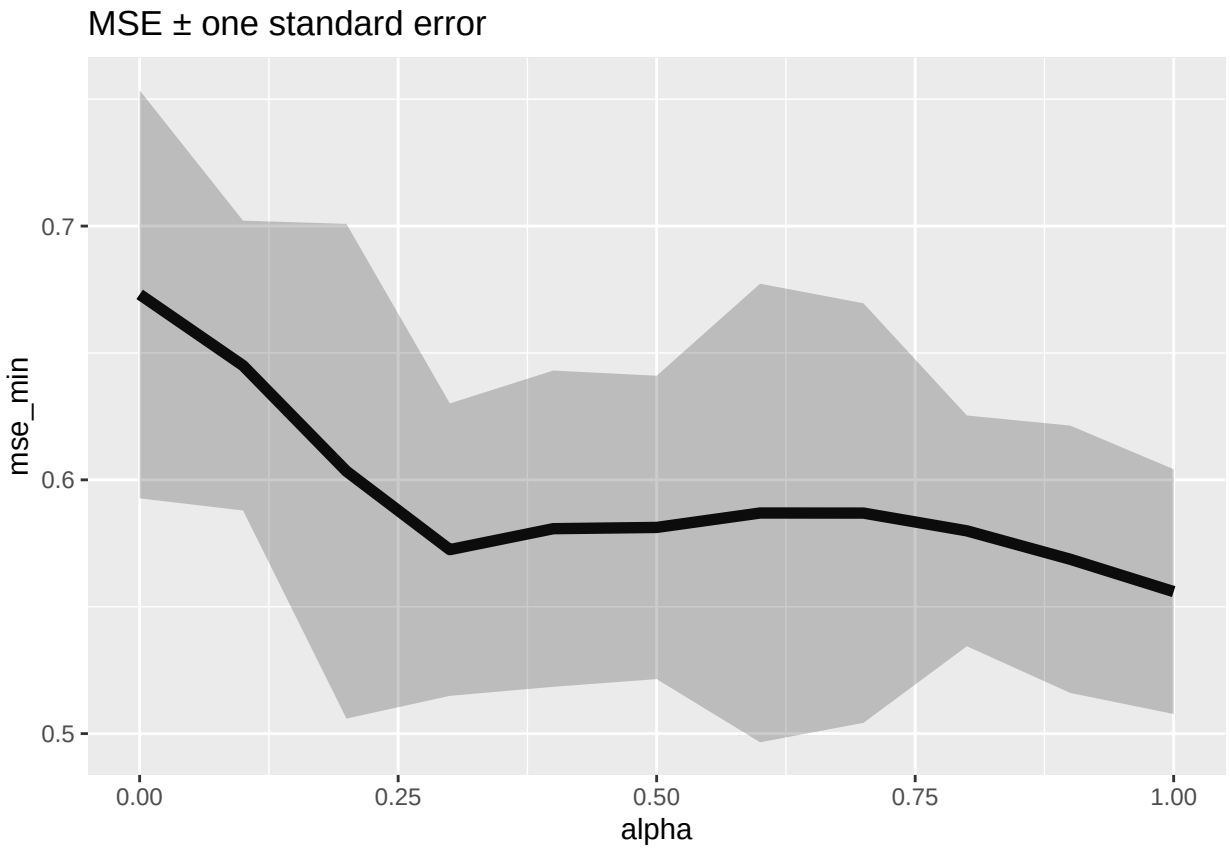
```
## # A tibble: 11 x 5
##    alpha mse_min mse_1se lambda_min lambda_1se
##    <dbl> <lgl>   <lgl>   <lgl>      <lgl>
```

```
##  1   0   NA       NA       NA          NA
##  2   0.1 NA       NA       NA          NA
##  3   0.2 NA       NA       NA          NA
##  4   0.3 NA       NA       NA          NA
##  5   0.4 NA       NA       NA          NA
##  6   0.5 NA       NA       NA          NA
##  7   0.6 NA       NA       NA          NA
##  8   0.7 NA       NA       NA          NA
##  9   0.8 NA       NA       NA          NA
## 10   0.9 NA       NA       NA          NA
## 11   1   NA       NA       NA          NA
```

```
for(i in seq_along(tuning_grid$alpha)) {

  # fit CV model for each alpha value
  fit <- cv.glmnet(nba_training_data_a, nba_training_data_b, alpha = tuning_grid$alpha[i])

  # extract MSE and lambda values
  tuning_grid$mse_min[i]    <- fit$cvm[fit$lambda == fit$lambda.min]
  tuning_grid$mse_1se[i]    <- fit$cvm[fit$lambda == fit$lambda.1se]
  tuning_grid$lambda_min[i] <- fit$lambda.min
  tuning_grid$lambda_1se[i] <- fit$lambda.1se
}

tuning_grid
```

```
## # A tibble: 11 x 5
##    alpha mse_min mse_1se lambda_min lambda_1se
##    <dbl>   <dbl>   <dbl>      <dbl>      <dbl>
##  1   0     0.673   0.753     0.112       0.600
##  2   0.1   0.645   0.702     0.226       0.689
##  3   0.2   0.603   0.701     0.103       0.602
##  4   0.3   0.572   0.630     0.0906      0.366
##  5   0.4   0.581   0.643     0.0819      0.363
##  6   0.5   0.581   0.641     0.0719      0.319
##  7   0.6   0.587   0.677     0.0658      0.351
##  8   0.7   0.587   0.670     0.0564      0.301
##  9   0.8   0.580   0.625     0.0594      0.219
## 10   0.9   0.569   0.621     0.0481      0.194
## 11   1     0.556   0.604     0.0395      0.159
```

```
tuning_grid %>%
  mutate(se = mse_1se - mse_min) %>%
  ggplot(aes(alpha, mse_min)) +
  geom_line(size = 2) +
  geom_ribbon(aes(ymax = mse_min + se, ymin = mse_min - se), alpha = .25) +
  ggtitle("MSE ± one standard error")
```



Independientemente de la muestra que se haya escogido, se puede observar que el error es más elevado para los valores de alpha entre 0 y 0.25 ó 0.30, es por eso que cualquier modelo entre alpha 0.30 hasta 1 nos puede servir para poder predecir los salarios de la NBA.

Pero debido a nuestro sesgo de querer elegir modelos parsimoniosos, es decir, más sencillos, el modelo que más se ajsuta a nuestras necesidades es el modelo Lasso (alpha = 1), cuyo mse es de 0.575 aproximadamente.

7. Predicción

En este apartado, auqnue ya hayamos observado que el modelo Lasso es que mejor predice, vamos a ponerle valor a esa predicción, para ello, utilizaremos ese 30% de los datos que no habiamos utilizado para poder ver cuál es el error medio de la predicción de los valores.

Se obtiene que el error cuadrado mínimo es de 0.556, pero el error medio de predicción no puedo calcularlo por errores que no he podido solucionar.

```
#Lasso
cv_lasso <- cv.glmnet(nba_training_data_a, nba_training_data_b, alpha = 1.0)
min(cv_lasso$cvm)
```

```
## [1] 0.5622004
```