TFM - Kaggle House Prices: Advanced Regression Techniques with caret

03 Selección de predictores con caret

Juan Carlos Santiago Culebras

2019-09-22

El objetivo de esta fase es reducir el volumen de características, de tal forma que únicamente los predictores que están relacionados con la variable respuestas se incluyan en el modelo.

Para ello utilizaremos varias de las metodología de selección de características (feature selection) que ofrece el paquete caret.

*Métodos wrapper: evalúan múltiples modelos utilizando procedimientos que agregan y / o eliminan predictores para encontrar la combinación óptima que maximice el rendimiento del modelo, son algoritmos de búsqueda donde los predictores son las entradas y el modelo a optimizar es la salida. **Eliminación de características recursivas** Algoritmos genéticos **Simulated annealing

*Métodos de filtro: Analizan la relación que tiene cada predictor con la variable respuesta, evaluando la relevancia de los predictores fuera de los modelos y seleccionando los que pasan algún criterio.

Además, cada uno de estos métodos puede utilizar distintos algoritmos (regresión lineal, naive bayes, random forest) y métodos de entrenamiento (Validación cruzada o bootstrapping).

Primeros pasos

Librerías

Realizamos la carga de las librerías necesarias

```r
if(!is.element("dplyr", installed.packages()[, 1]))
      install.packages("dplyr", repos = 'http://cran.us.r-project.org')
library(dplyr)

if(!is.element("tidyr", installed.packages()[, 1]))
      install.packages("tidyr", repos = 'http://cran.us.r-project.org')
library(tidyr)

if(!is.element("ggplot2", installed.packages()[, 1]))
      install.packages("ggplot2", repos = 'http://cran.us.r-project.org')
library(ggplot2)

if(!is.element("tibble", installed.packages()[, 1]))
      install.packages("tibble", repos = 'http://cran.us.r-project.org')
library(tibble)

if(!is.element("randomForest", installed.packages()[, 1]))
      install.packages("randomForest", repos = 'http://cran.us.r-project.org')
library(randomForest)
```

```r
if(!is.element("recipes", installed.packages()[, 1]))
      install.packages("recipes", repos = 'http://cran.us.r-project.org')
library(recipes)

if(!is.element("caret", installed.packages()[, 1]))
      install.packages("caret", repos = 'http://cran.us.r-project.org')
library(caret)

if(!is.element("gam", installed.packages()[, 1]))
      install.packages("gam", repos = 'http://cran.us.r-project.org')
library(gam)
```

Cargamos datos

Partimos de los dataset generados en la fase 2

Repetimos el proceso de selección con distintos dataset de origen (F02_01_dsDataAll y F02_03_dsDataAll_Recipe) para poder comparar soluciones.

```r
strOrigenF2 <- 'F02_01_dsDataAll'

# Segunda ejecución
#strOrigenF2 <- 'F02_03_dsDataAll_Recipe'

file <- paste('./F02_Datos/',strOrigenF2,'.RData',sep='')

load(file)


dirSalida <- paste('./F03_SelPredictores/',strOrigenF2,sep='')

if (!file.exists(dirSalida)){
     dir.create(file.path(dirSalida))
}

rm(strOrigenF2)
rm(file)
```

Lectura de modelos ya entrenados si se realiza es estudio posteriormente

```r
# load('./F03_SelPredictores/F02_01_dsDataAll/F03_1_rfe_lm.RData')
# load('./F03_SelPredictores/F02_01_dsDataAll/F03_2_rfe_rf.RData')
# load('./F03_SelPredictores/F02_01_dsDataAll/F03_3_ga_20.RData')
# load('./F03_SelPredictores/F02_01_dsDataAll/F03_4_ga_100.RData')
# load('./F03_SelPredictores/F02_01_dsDataAll/F03_5_sbf_lm.RData')
# load('./F03_SelPredictores/F02_01_dsDataAll/F03_6_sbf_rf.RData')
```

Separamos los datos

Dividimos el dataset de origen: *dsTrain - Que a su vez se divide en* ***dsTrain.training*** *dsTrain.CV* dsTest (no se utiliza en esta fase)

```
dsTrain <- dsDataAll %>%
  filter(indTrain == 1) %>%
  select(SalePrice, everything()) %>%
  select(-c(Id,indTrain))

dim(dsTrain)
```

```
## [1] 1458   93
```

```
set.seed(123)
iTrain  <- createDataPartition(y=dsTrain$SalePrice, p=0.7, list=F)

dsTrain.training <- dsTrain[iTrain, ]
dsTrain.CV       <- dsTrain[-iTrain, ]

rm(iTrain)
```

Selección de predictores mediante caret

Definimos los parámetros de control para realizar procesos

He seleccionado como método de evaluación, la validación cruzada con 5 particiones y 5 repeticiones.

```
# Parámetros para CV y Bootstrapping
particiones = 5
repeticiones = 5

# conjunto de números de predictores a calcular
# nos permiten posteriormente identificar el número de predictores optimo
subsets <- c(5, seq(10, 20, by=2), seq(25, 60, by=5)) # Origen F02_01_dsDataAll
# subsets <- c(seq(10, 170, by=10)) # Origen F02_03_dsDataAll_Recipe
```

Métodos wrapper

RFE (Recursive feature elimination) de Caret

RFE (Recursive feature elimination) de Caret ofrece multitud de posibilidades para ejecutar estas funciones, yo he implementado varias de ellas: *Regresión lineal y validación cruzada* Randon Forest y validación cruzada

Es necesario indicar el parámetro "size" que permite determinar sobre que tamaños de conjuntos de variables se desea que busque el algoritmo. En este caso y después de realizar varias pruebas he optado por buscar en conjuntos con tamaños:

- (5 10 12 14 16 18 20 25 30 35 40 45 50 55 60)

Esto me permite dibujar gráficas para ver la evolución de RMSE con los distintos subconjuntos, caret además añade un cálculo con todas las posibles variables. En algunos casos ha sido necesario refinar la búsqueda, modificando estos conjuntos.

```
subsets <- c(5, seq(10, 20, by=2), seq(25, 60, by=5)) # Origen F02_01_dsDataAll
```

Eliminación Recursiva con Regresión linal y validación cruzada

```
ctrl <- rfeControl(functions = lmFuncs
                      ,method = "repeatedcv" # Validación cruzada
                      ,number = particiones
                      ,repeats = repeticiones
                      ,verbose = FALSE)

t <- proc.time() # Inicia el cronómetro
F03_1_rfe_lm <- rfe(SalePrice ~ .
              , data = dsTrain.training
              , sizes = subsets
              , metric = "RMSE"
              , rfeControl = ctrl)
proc.time()-t    # Detiene el cronómetro
```

```
##    user  system elapsed
##    5.50    0.11    5.61
```

```
# Tiempo de ejecución
# user  system elapsed
# 5.00    0.20    5.15

# Guardo resultado del calculo
fileOuput <- paste(dirSalida,'F03_1_rfe_lm.RData',sep="/")
save(F03_1_rfe_lm, file = fileOuput)
```

Estudio de resultados

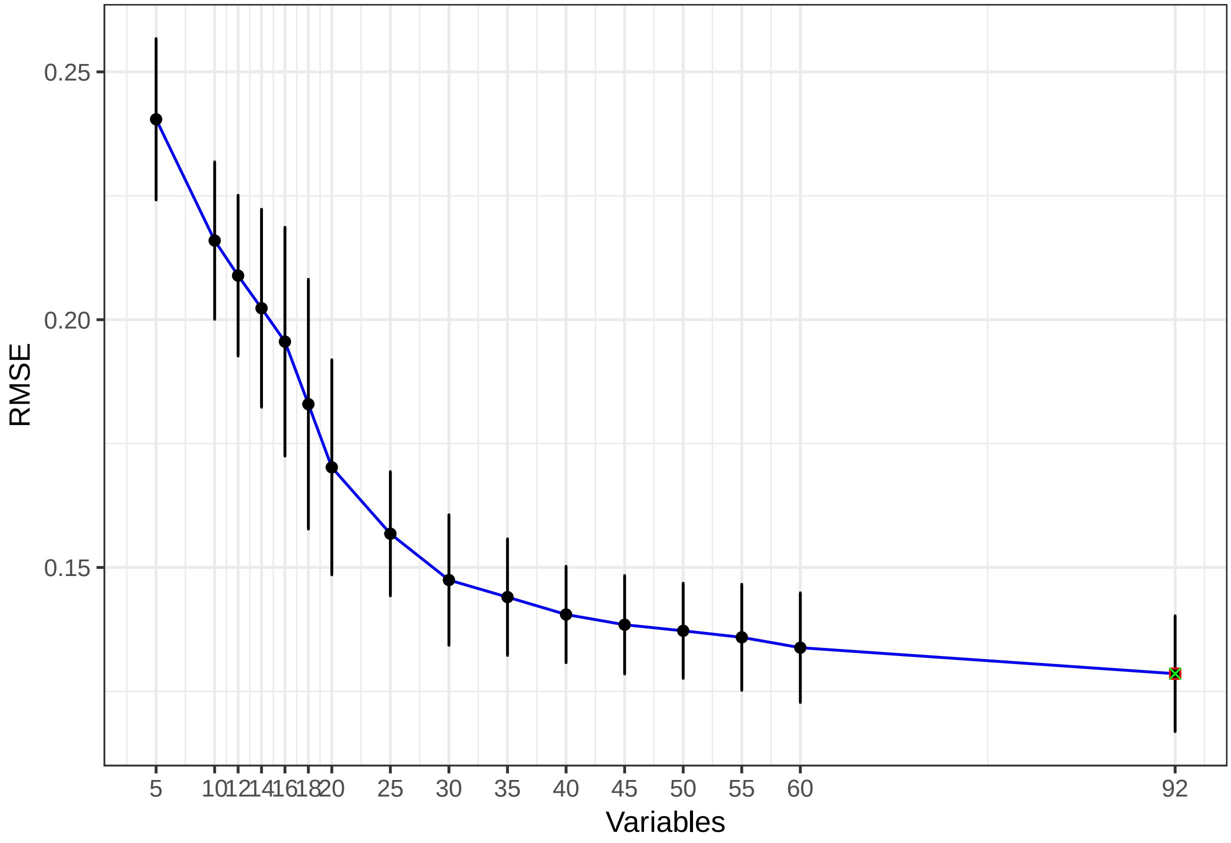
```
F03_1_rfe_lm
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (5 fold, repeated 5 times)
##
## Resampling performance over subset size:
##
##  Variables   RMSE Rsquared     MAE   RMSESD RsquaredSD    MAESD Selected
##          5 0.2404   0.6475 0.18458 0.016247    0.04573 0.012467
##         10 0.2160   0.7155 0.16320 0.015849    0.04067 0.011530
##         12 0.2089   0.7336 0.15731 0.016189    0.04351 0.012771
##         14 0.2023   0.7498 0.15244 0.019946    0.04838 0.014495
##         16 0.1956   0.7652 0.14727 0.023052    0.05289 0.016519
##         18 0.1829   0.7942 0.13786 0.025164    0.05214 0.017330
##         20 0.1702   0.8224 0.12824 0.021661    0.04330 0.014907
##         25 0.1568   0.8504 0.11678 0.012512    0.02499 0.008711
##         30 0.1475   0.8679 0.10827 0.013142    0.02551 0.008266
##         35 0.1440   0.8742 0.10509 0.011739    0.02189 0.007951
##         40 0.1405   0.8809 0.10190 0.009692    0.01677 0.006584
##         45 0.1384   0.8846 0.09916 0.009902    0.01653 0.006557
##         50 0.1372   0.8866 0.09805 0.009569    0.01604 0.006340
##         55 0.1359   0.8884 0.09714 0.010670    0.01828 0.006478
##         60 0.1338   0.8917 0.09558 0.011026    0.01890 0.006961
##         92 0.1286   0.9001 0.09070 0.011653    0.01777 0.006089        *
##
## The top 5 variables (out of 92):
##    GrLivArea, SaleType_New, MasVnrType_None, SaleCondition_Partial, X1stFlrSF
```

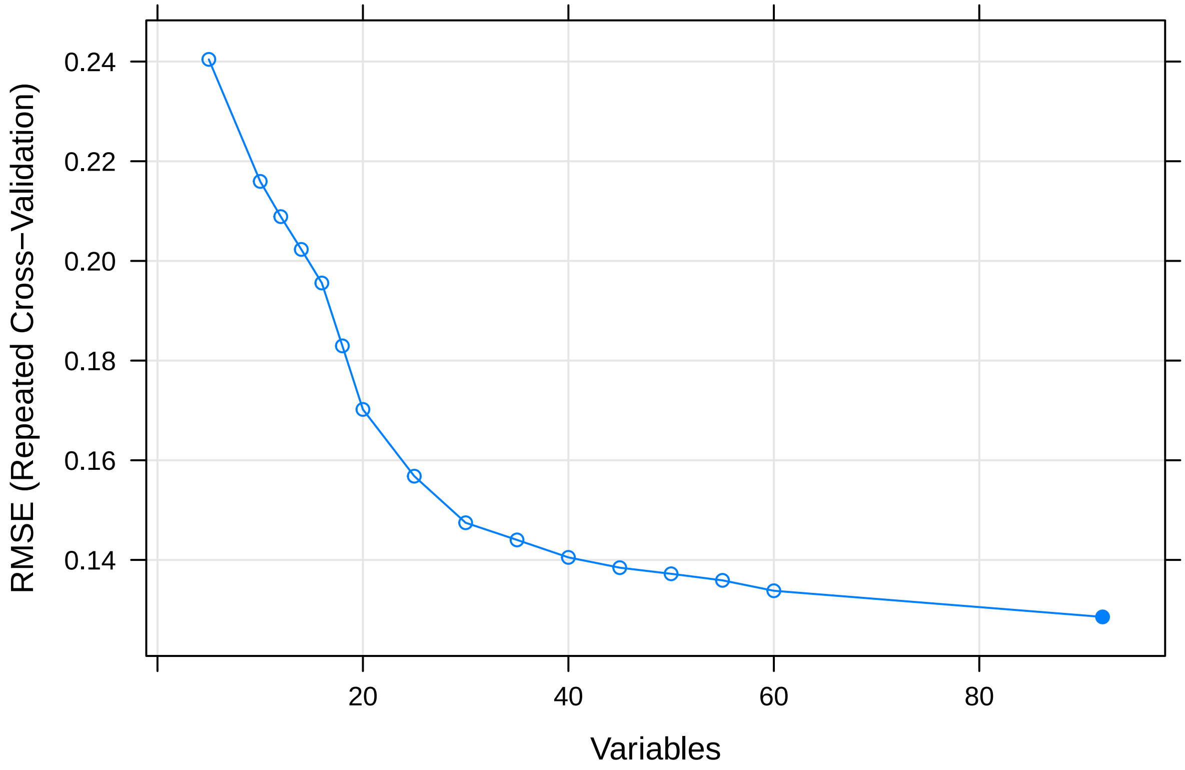
```
dsResults <- F03_1_rfe_lm$results

# Métricas promedio de cada tamaño
dsResults %>%
  group_by(Variables) %>%
  summarise(media_RMSE = mean(RMSE), media_Rsquared = mean(Rsquared)) %>%
  arrange(media_RMSE)
```

```
## # A tibble: 16 x 3
##    Variables media_RMSE media_Rsquared
##        <dbl>      <dbl>          <dbl>
##  1        92      0.129          0.900
##  2        60      0.134          0.892
##  3        55      0.136          0.888
##  4        50      0.137          0.887
##  5        45      0.138          0.885
##  6        40      0.141          0.881
##  7        35      0.144          0.874
##  8        30      0.147          0.868
##  9        25      0.157          0.850
## 10        20      0.170          0.822
## 11        18      0.183          0.794
## 12        16      0.196          0.765
## 13        14      0.202          0.750
## 14        12      0.209          0.734
## 15        10      0.216          0.716
## 16         5      0.240          0.648
```

```
mejorAbsoluto <- pickSizeBest(select(dsResults,RMSE,Variables)
                              , metric = "RMSE"
                              , maximize = FALSE)
mejorRendimiento <- pickSizeTolerance(select(dsResults,RMSE,Variables)
                                  , metric = "RMSE"
                                  , maximize = FALSE)

## Percent Loss in performance (positive)
# ToDo: example$PctLoss <- (example$RMSE - min(example$RMSE))/min(example$RMSE)*100

# Gráfica de disminución de RMSE
ggplot(data = dsResults, aes(x = Variables, y = RMSE)) +
  geom_line(color = "blue") +
  scale_x_continuous(breaks = unique(dsResults$Variables)) +
  geom_point() +
  geom_errorbar(aes(ymin = RMSE - RMSESD, ymax = RMSE + RMSESD),
                width = 0.2) +

  geom_point(data = filter(dsResults, Variables==mejorAbsoluto)
             , shape=0, cex= 1.5, color = "red") +

  geom_point(data = filter(dsResults, Variables==mejorRendimiento)
             , shape = 4, cex= 1.5, color = "green") +

  theme_bw()
```



```
plot(F03_1_rfe_lm,type = c("g", "o"))
```



```
resumenResultatos <- bind_rows(
  filter(dsResults, Variables==mejorAbsoluto) %>%
    mutate(modelo = 'F03_1_rfe_lm', tipo = 'mejorAbsoluto'),
  filter(dsResults, Variables==mejorRendimiento) %>%
    mutate(modelo = 'F03_1_rfe_lm', tipo = 'mejorRendimiento'))
```

Eliminación Recursiva con Randon Forest y validación cruzada

```
ctrl <- rfeControl(functions = rfFuncs
                      ,method = "repeatedcv" # Validación cruzada
                      ,number = particiones
                      ,repeats = repeticiones
                      ,verbose = FALSE)

t <- proc.time() # Inicia el cronómetro
F03_2_rfe_rf <- rfe(SalePrice ~ .
              , data = dsTrain.training
              , sizes = subsets
              , metric = "RMSE"
              , rfeControl = ctrl)
proc.time()-t     # Detiene el cronómetro1
```

```
##    user  system elapsed
## 1486.86    7.98 1496.89
```

```r
# Tiempo de ejecución
# user  system elapsed
# 1713.67    9.26 1734.17

# Guardo resultado del calculo
fileOuput <- paste(dirSalida,'F03_2_rfe_rf.RData',sep="/")
save(F03_2_rfe_rf, file = fileOuput)
```

Estudio de resultados

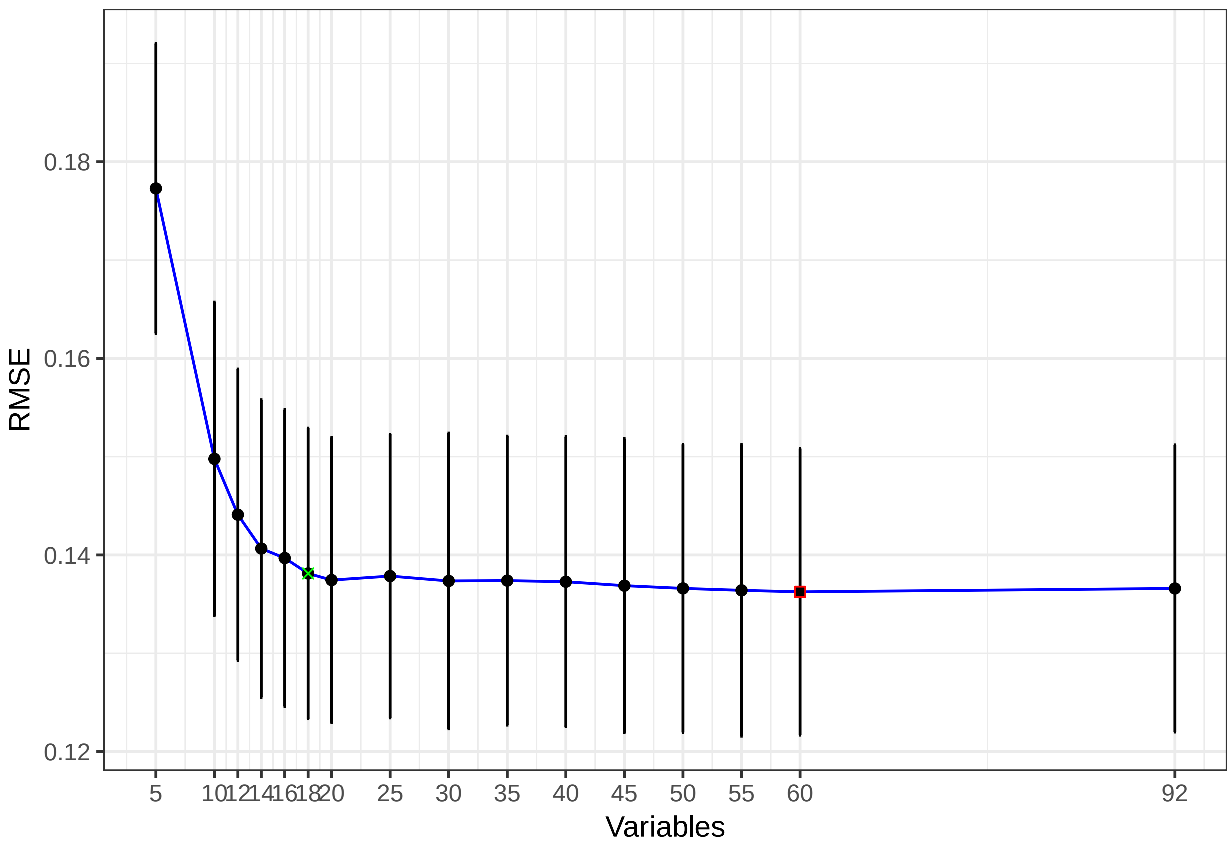
```r
F03_2_rfe_rf
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (5 fold, repeated 5 times)
##
## Resampling performance over subset size:
##
##  Variables   RMSE Rsquared     MAE  RMSESD RsquaredSD    MAESD Selected
##          5 0.1773   0.8147 0.12900 0.01475    0.02732 0.007566
##         10 0.1498   0.8670 0.10412 0.01595    0.02780 0.008373
##         12 0.1441   0.8772 0.09935 0.01482    0.02476 0.007010
##         14 0.1407   0.8841 0.09664 0.01514    0.02381 0.007493
##         16 0.1397   0.8858 0.09590 0.01510    0.02338 0.007621
##         18 0.1381   0.8886 0.09506 0.01479    0.02166 0.007244
##         20 0.1374   0.8903 0.09432 0.01451    0.02089 0.007003
##         25 0.1379   0.8902 0.09458 0.01443    0.02057 0.007113
##         30 0.1374   0.8910 0.09414 0.01505    0.02165 0.007477
##         35 0.1374   0.8911 0.09393 0.01470    0.02113 0.007060
##         40 0.1373   0.8916 0.09390 0.01476    0.02105 0.007049
##         45 0.1369   0.8922 0.09376 0.01496    0.02126 0.007298
##         50 0.1366   0.8929 0.09355 0.01465    0.02069 0.007237
##         55 0.1364   0.8932 0.09341 0.01484    0.02097 0.007166
##         60 0.1362   0.8936 0.09327 0.01458    0.02048 0.007250        *
##         92 0.1366   0.8935 0.09342 0.01461    0.02046 0.007241
##
## The top 5 variables (out of 60):
##    GrLivArea, OverallQual, TotalBsmtSF, X1stFlrSF, BsmtFinSF1
```

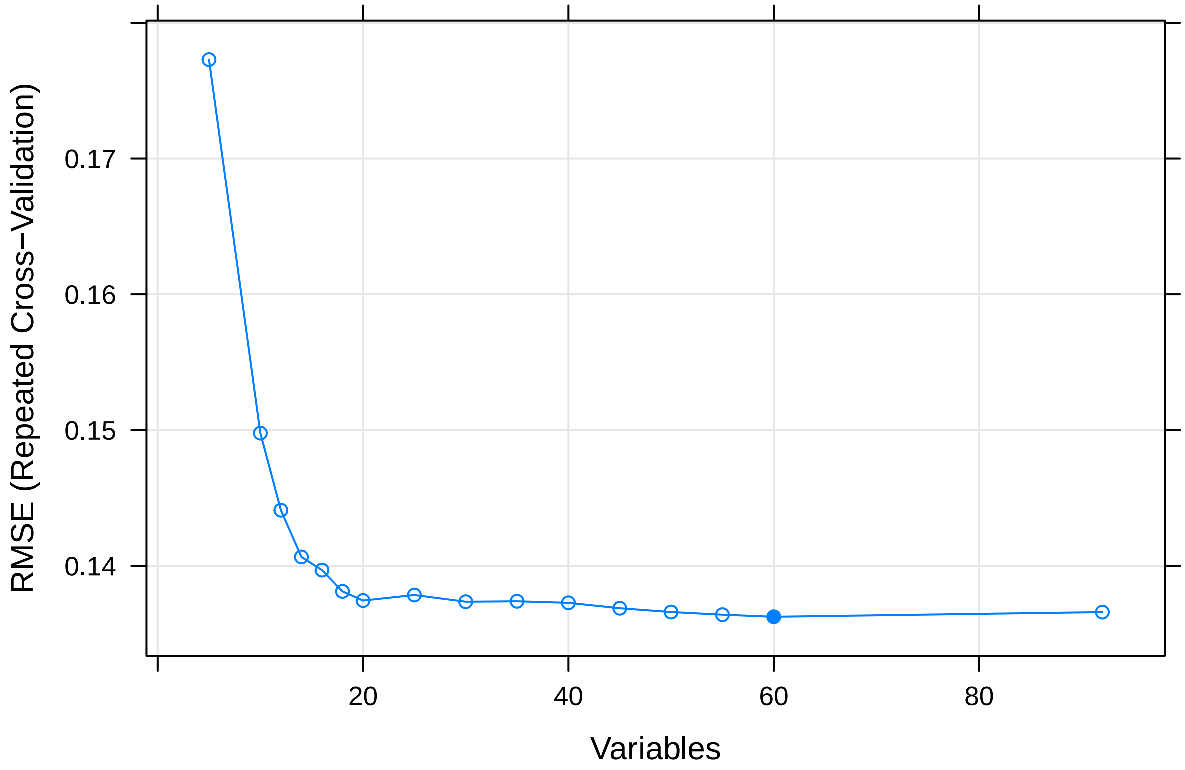
```r
dsResults <- F03_2_rfe_rf$results

# Métricas promedio de cada tamaño
dsResults %>%
  group_by(Variables) %>%
  summarise(media_RMSE = mean(RMSE), media_Rsquared = mean(Rsquared)) %>%
  arrange(media_RMSE)
```

```
## # A tibble: 16 x 3
##    Variables media_RMSE media_Rsquared
##        <dbl>      <dbl>          <dbl>
##  1        60      0.136          0.894
##  2        55      0.136          0.893
##  3        92      0.137          0.894
##  4        50      0.137          0.893
##  5        45      0.137          0.892
##  6        40      0.137          0.892
##  7        30      0.137          0.891
##  8        35      0.137          0.891
##  9        20      0.137          0.890
## 10        25      0.138          0.890
## 11        18      0.138          0.889
## 12        16      0.140          0.886
## 13        14      0.141          0.884
## 14        12      0.144          0.877
## 15        10      0.150          0.867
## 16         5      0.177          0.815
```

```r
mejorAbsoluto <- pickSizeBest(select(dsResults,RMSE,Variables)
                              , metric = "RMSE"
                              , maximize = FALSE)
mejorRendimiento <- pickSizeTolerance(select(dsResults,RMSE,Variables)
                                  , metric = "RMSE"
                                  , maximize = FALSE)

## Percent Loss in performance (positive)
# ToDo: example$PctLoss <- (example$RMSE - min(example$RMSE))/min(example$RMSE)*100

# Gráfica de disminución de RMSE
ggplot(data = dsResults, aes(x = Variables, y = RMSE)) +
  geom_line(color = "blue") +
  scale_x_continuous(breaks = unique(dsResults$Variables)) +
  geom_point() +
  geom_errorbar(aes(ymin = RMSE - RMSESD, ymax = RMSE + RMSESD),
                width = 0.2) +

  geom_point(data = filter(dsResults, Variables==mejorAbsoluto)
             , shape=0, cex= 1.5, color = "red") +

  geom_point(data = filter(dsResults, Variables==mejorRendimiento)
             , shape = 4, cex= 1.5, color = "green") +

  theme_bw()
```



```r
plot(F03_2_rfe_rf,type = c("g", "o"))
```



```
# guardo los mejores resultados para comparar
resumenResultatos <- bind_rows(
  resumenResultatos,
  filter(dsResults, Variables==mejorAbsoluto) %>%
    mutate(modelo = 'F03_2_rfe_rf', tipo = 'mejorAbsoluto'),
  filter(dsResults, Variables==mejorRendimiento) %>%
    mutate(modelo = 'F03_2_rfe_rf', tipo = 'mejorRendimiento')
  )
```

Algoritmos Genéticos

Algoritmos Genéticos con Randon Forest y validación cruzada

20 Iteraciones

```
# ctrl <- gafsControl(functions = rfGA,
#                         method = "cv",
#                         number = particiones,
#                         allowParallel = TRUE,
#                         genParallel = TRUE,
#                         verbose = FALSE)
#
#
# F03_3_ga_20 <- gafs(x = dsTrain.training[,-1]
#                , y = dsTrain.training$SalePrice
#               , iters = 20
#               , popSize = 10
#               , gafsControl = ctrl
#               )
#
# # Tiempo de ejecución
# # user  system elapsed
# #
#
# # Guardo resultado del calculo
# fileOuput <- paste(dirSalida,'F03_3_ga_20.RData',sep="/")
# save(F03_3_ga_20, file = fileOuput)
```

Estudio de resultados

```
#
# F03_3_ga_20
# F03_3_ga_20$optVariables
#
# # Métricas promedio de cada iteración
# ga.results <- F03_3_ga_20$external %>%
#   group_by(Iter) %>%
#   dplyr::summarise(media_RMSE = mean(RMSE)
#                     , media_Rsquared = mean(Rsquared)) %>%
#   arrange(media_RMSE)
#
# ga.results
#
# # Gráfica de disminución de RMSE
# ggplot(data = ga.results, aes(x = Iter, y = media_RMSE)) +
#   geom_line() +
#   scale_x_continuous(breaks  = unique(ga.results$Iter)) +
#   theme_bw()
```

100 Iteraciones

```
# ctrl <- gafsControl(functions = rfGA,
#                         method = "cv",
#                         number = particiones,
#                         allowParallel = TRUE,
#                         genParallel = TRUE,
#                         verbose = FALSE)
#
# t <- proc.time() # Inicia el cronómetro
# F03_4_ga_100 <- gafs(x = dsTrain.training[,-1]
#               , y = dsTrain.training$SalePrice
#               , iters = 100
#               , popSize = 10
#               , gafsControl = ctrl
#               )
# proc.time()-t    # Detiene el cronómetro
#
# # Tiempo de ejecución
# # user  system elapsed
# # 6543.12   24.70 6568.33
#
# # Guardo resultado del calculo
# fileOuput <- paste(dirSalida,'F03_4_ga_100.RData',sep="/")
# save(F03_4_ga_100, file = fileOuput)

load('./F03_SelPredictores/F03_4_ga_100.RData')
```

Estudio de resultados

```
F03_4_ga_100
```

```
##
## Genetic Algorithm Feature Selection
##
## 1023 samples
## 92 predictors
##
## Maximum generations: 100
## Population per generation: 10
## Crossover probability: 0.8
## Mutation probability: 0.1
## Elitism: 0
##
## Internal performance values: RMSE, Rsquared
## Subset selection driven to minimize internal RMSE
##
## External performance values: RMSE, Rsquared, MAE
## Best iteration chose by minimizing external RMSE
## External resampling method: Cross-Validated (5 fold)
##
## During resampling:
##   * the top 5 selected variables (out of a possible 92):
##     BsmtExposure (100%), BsmtHalfBath (100%), Electrical (100%), Exterior1st_WdSdng (100%), Fireplac
##   * on average, 58 variables were selected (min = 51, max = 72)
##
## In the final search using the entire training set:
##    * 46 features selected at iteration 15 including:
##      LotFrontage, LotArea, LotShape, OverallQual, OverallCond ...
##    * external performance at this iteration is
##
##         RMSE     Rsquared          MAE
##      0.13749      0.89384      0.09386
```

Mejores variables

```
F03_4_ga_100$optVariables
```

```
##  [1] "LotFrontage"          "LotArea"               "LotShape"
##  [4] "OverallQual"          "OverallCond"           "YearBuilt"
##  [7] "YearRemodAdd"         "ExterQual"             "ExterCond"
## [10] "BsmtFinType1"         "BsmtUnfSF"            "TotalBsmtSF"
## [13] "HeatingQC"            "CentralAir"           "Electrical"
## [16] "X2ndFlrSF"            "GrLivArea"            "BsmtFullBath"
## [19] "FullBath"             "KitchenQual"          "Fireplaces"
## [22] "GarageArea"           "GarageQual"           "PavedDrive"
## [25] "WoodDeckSF"           "YrSold"               "MSSubClass_120"
## [28] "MSSubClass_20"        "MSSubClass_50"        "MSSubClass_60"
## [31] "MSZoning_RL"          "MSZoning_RM"          "LotConfig_Corner"
## [34] "Neighborhood_Edwards" "Neighborhood_OldTown" "Neighborhood_Sawyer"
## [37] "Condition1_Norm"      "HouseStyle_1Story"    "Exterior1st_WdSdng"
## [40] "Exterior2nd_MetalSd"  "Exterior2nd_Plywood"  "Exterior2nd_WdSdng"
## [43] "MasVnrType_None"      "Foundation_PConc"     "GarageType_None"
## [46] "SaleType_New"
```

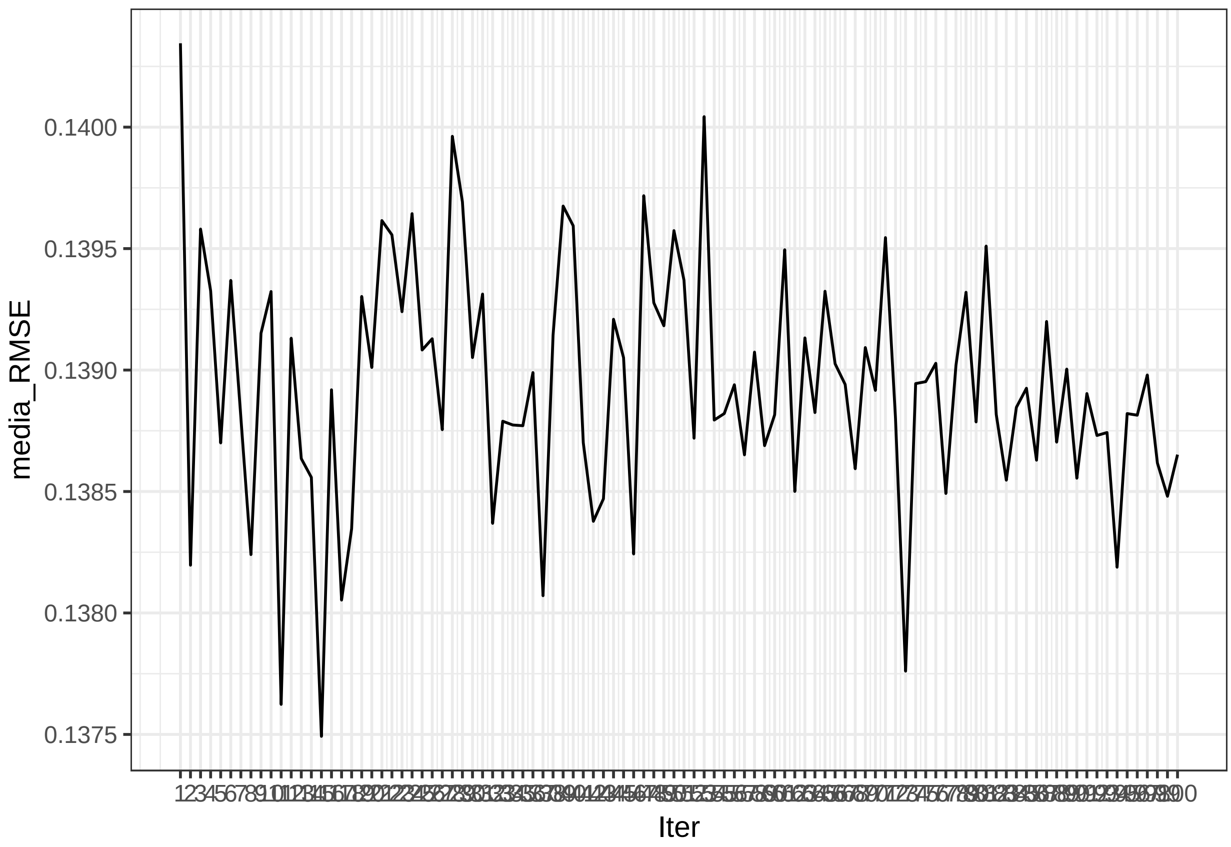
Métricas promedio de cada iteración

```
ga.results <- F03_4_ga_100$external %>%
  group_by(Iter) %>%
  dplyr::summarise(media_RMSE = mean(RMSE)
                   , media_Rsquared = mean(Rsquared)) %>%
  arrange(media_RMSE)

ga.results
```

```
## # A tibble: 100 x 3
##     Iter media_RMSE media_Rsquared
##    <int>      <dbl>          <dbl>
##  1    15      0.137          0.894
##  2    11      0.138          0.894
##  3    73      0.138          0.894
##  4    17      0.138          0.893
##  5    37      0.138          0.893
##  6    94      0.138          0.893
##  7     2      0.138          0.893
##  8     8      0.138          0.893
##  9    46      0.138          0.892
## 10    18      0.138          0.892
## # ... with 90 more rows
```

Gráfica de disminución de RMSE

```
ggplot(data = ga.results, aes(x = Iter, y = media_RMSE)) +
  geom_line() +
  scale_x_continuous(breaks  = unique(ga.results$Iter)) +
  theme_bw()
```



Métodos de filtrado

Selección por filtros Recursiva con Regresión linal y validación cruzada

```r
ctrl <- sbfControl(functions = lmSBF
                  , method = "repeatedcv"
                  , number = particiones
                  , repeats = repeticiones
                  , verbose = FALSE
                  , saveDetails = TRUE)

t <- proc.time() # Inicia el cronómetro
F03_5_sbf_lm <- sbf(SalePrice ~ .
              , data = dsTrain.training
              , sbfControl = ctrl
              )
proc.time()-t    # Detiene el cronómetro
```

```
##    user  system elapsed
##    7.28    0.03    7.31
```

```r
# Tiempo de ejecución
# user  system elapsed
#  304.03    4.96  308.92

# Guardo resultado del calculo
fileOuput <- paste(dirSalida,'F03_5_sbf_lm.RData',sep="/")
save(F03_5_sbf_lm, file = fileOuput)
```

Estudio de resultados

```
F03_5_sbf_lm
```

```
##
## Selection By Filter
##
## Outer resampling method: Cross-Validated (5 fold, repeated 5 times)
##
## Resampling performance:
##
##    RMSE Rsquared     MAE  RMSESD RsquaredSD    MAESD
##  0.1319   0.8952 0.09504 0.01023    0.01365 0.004632
##
## Using the training set, 84 variables were selected:
##    LotFrontage, LotArea, LotShape, OverallQual, YearBuilt...
##
## During resampling, the top 5 selected variables (out of a possible 85):
##    BedroomAbvGr (100%), BldgType_1Fam (100%), BsmtExposure (100%), BsmtFinSF1 (100%), BsmtFinType1 (
##
## On average, 81.2 variables were selected (min = 78, max = 84)
```

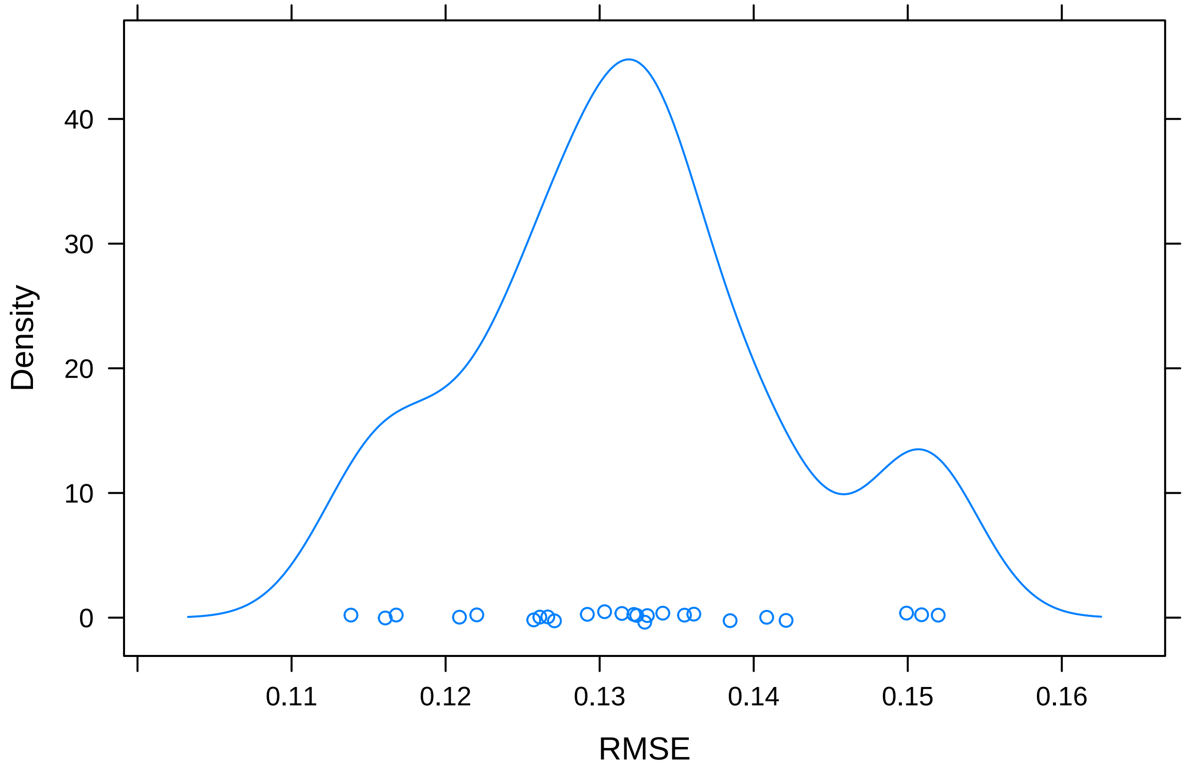
```
summary(F03_5_sbf_lm)
```

```
##              Length Class      Mode
## pred          3     -none-     list
## variables    25     -none-     list
## results       6     data.frame list
## fit          12     lm         list
## optVariables 84     -none-     character
## call          4     -none-     call
## control      14     -none-     list
## resample      4     data.frame list
## metrics       3     -none-     character
## times         3     -none-     list
## resampledCM   0     -none-     NULL
## obsLevels     0     -none-     NULL
## dots          0     -none-     list
## terms         3     terms      call
## coefnames    92     -none-     character
## xlevels       0     -none-     list
```

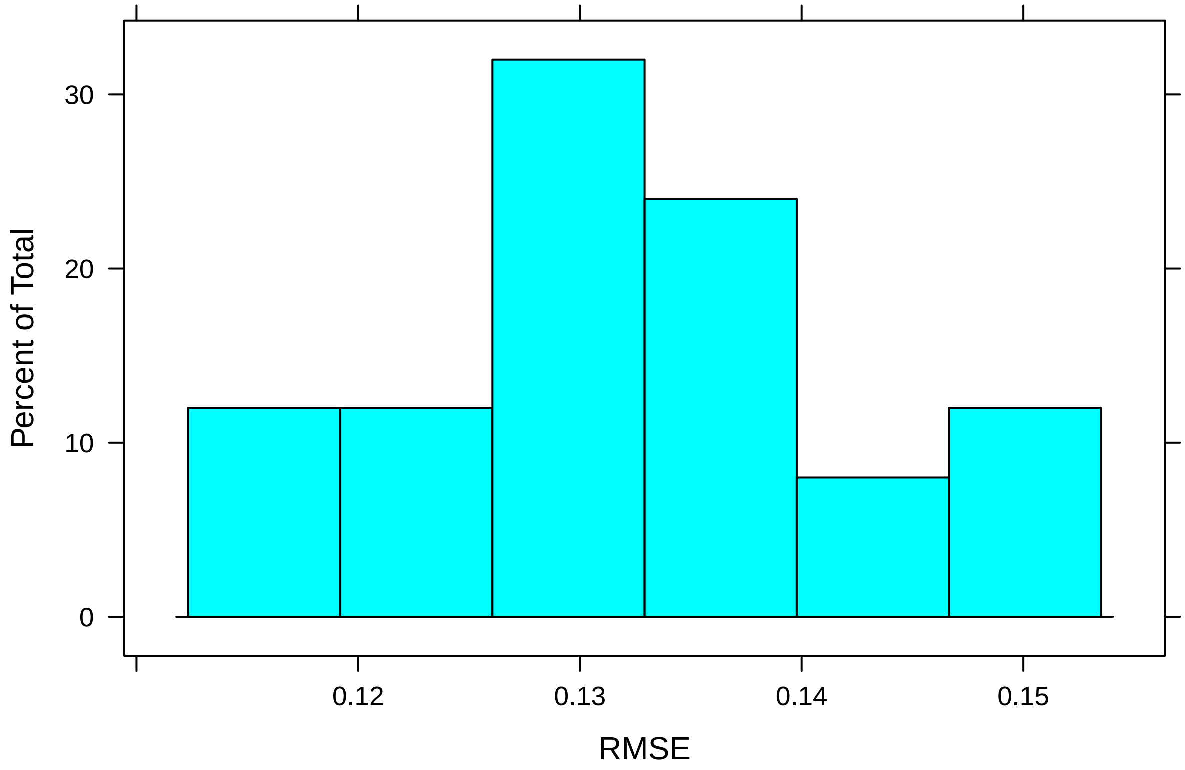
```
F03_5_sbf_lm$optVariables
```

```
##  [1] "LotFrontage"           "LotArea"
##  [3] "LotShape"             "OverallQual"
##  [5] "YearBuilt"            "YearRemodAdd"
##  [7] "MasVnrArea"           "ExterQual"
##  [9] "ExterCond"            "BsmtQual"
## [11] "BsmtExposure"         "BsmtFinType1"
## [13] "BsmtFinSF1"           "BsmtUnfSF"
## [15] "TotalBsmtSF"          "HeatingQC"
## [17] "CentralAir"           "Electrical"
## [19] "X1stFlrSF"            "X2ndFlrSF"
## [21] "GrLivArea"            "BsmtFullBath"
## [23] "FullBath"             "HalfBath"
## [25] "BedroomAbvGr"         "KitchenQual"
## [27] "TotRmsAbvGrd"         "Fireplaces"
## [29] "FireplaceQu"          "GarageFinish"
## [31] "GarageCars"           "GarageArea"
## [33] "GarageQual"           "GarageCond"
## [35] "PavedDrive"           "WoodDeckSF"
## [37] "MoSold"               "YrSold"
## [39] "MSSubClass_120"       "MSSubClass_20"
## [41] "MSSubClass_50"        "MSSubClass_60"
## [43] "MSZoning_RL"          "MSZoning_RM"
## [45] "LotConfig_CulDSac"    "LotConfig_Inside"
## [47] "Neighborhood_CollgCr" "Neighborhood_Edwards"
## [49] "Neighborhood_Gilbert" "Neighborhood_NAmes"
## [51] "Neighborhood_NridgHt" "Neighborhood_OldTown"
## [53] "Neighborhood_Sawyer"  "Neighborhood_Somerst"
## [55] "Condition1_Feedr"     "Condition1_Norm"
## [57] "BldgType_1Fam"        "HouseStyle_1.5Fin"
## [59] "HouseStyle_1Story"    "HouseStyle_2Story"
## [61] "RoofStyle_Gable"      "RoofStyle_Hip"
## [63] "Exterior1st_HdBoard"  "Exterior1st_MetalSd"
## [65] "Exterior1st_VinylSd"  "Exterior1st_WdSdng"
## [67] "Exterior2nd_MetalSd"  "Exterior2nd_VinylSd"
## [69] "Exterior2nd_WdSdng"   "MasVnrType_BrkFace"
## [71] "MasVnrType_None"      "MasVnrType_Stone"
## [73] "Foundation_BrkTil"    "Foundation_CBlock"
## [75] "Foundation_PConc"     "GarageType_Attchd"
## [77] "GarageType_BuiltIn"   "GarageType_Detchd"
## [79] "GarageType_None"      "SaleType_New"
## [81] "SaleType_WD"          "SaleCondition_Abnorml"
## [83] "SaleCondition_Normal" "SaleCondition_Partial"
```

```
densityplot(F03_5_sbf_lm)
```



```
histogram(F03_5_sbf_lm)
```



```
predictors(F03_5_sbf_lm)
```

```
##  [1] "LotFrontage"          "LotArea"
##  [3] "LotShape"             "OverallQual"
##  [5] "YearBuilt"            "YearRemodAdd"
##  [7] "MasVnrArea"           "ExterQual"
##  [9] "ExterCond"            "BsmtQual"
## [11] "BsmtExposure"         "BsmtFinType1"
## [13] "BsmtFinSF1"           "BsmtUnfSF"
## [15] "TotalBsmtSF"          "HeatingQC"
## [17] "CentralAir"           "Electrical"
## [19] "X1stFlrSF"            "X2ndFlrSF"
## [21] "GrLivArea"            "BsmtFullBath"
## [23] "FullBath"             "HalfBath"
## [25] "BedroomAbvGr"         "KitchenQual"
## [27] "TotRmsAbvGrd"         "Fireplaces"
## [29] "FireplaceQu"          "GarageFinish"
## [31] "GarageCars"           "GarageArea"
## [33] "GarageQual"           "GarageCond"
## [35] "PavedDrive"           "WoodDeckSF"
## [37] "MoSold"               "YrSold"
## [39] "MSSubClass_120"       "MSSubClass_20"
## [41] "MSSubClass_50"        "MSSubClass_60"
## [43] "MSZoning_RL"          "MSZoning_RM"
## [45] "LotConfig_CulDSac"    "LotConfig_Inside"
## [47] "Neighborhood_CollgCr"  "Neighborhood_Edwards"
## [49] "Neighborhood_Gilbert"  "Neighborhood_NAmes"
## [51] "Neighborhood_NridgHt"  "Neighborhood_OldTown"
## [53] "Neighborhood_Sawyer"   "Neighborhood_Somerst"
## [55] "Condition1_Feedr"      "Condition1_Norm"
## [57] "BldgType_1Fam"         "HouseStyle_1.5Fin"
## [59] "HouseStyle_1Story"     "HouseStyle_2Story"
## [61] "RoofStyle_Gable"       "RoofStyle_Hip"
## [63] "Exterior1st_HdBoard"   "Exterior1st_MetalSd"
## [65] "Exterior1st_VinylSd"   "Exterior1st_WdSdng"
## [67] "Exterior2nd_MetalSd"   "Exterior2nd_VinylSd"
## [69] "Exterior2nd_WdSdng"    "MasVnrType_BrkFace"
## [71] "MasVnrType_None"       "MasVnrType_Stone"
## [73] "Foundation_BrkTil"     "Foundation_CBlock"
## [75] "Foundation_PConc"      "GarageType_Attchd"
## [77] "GarageType_BuiltIn"    "GarageType_Detchd"
## [79] "GarageType_None"       "SaleType_New"
## [81] "SaleType_WD"           "SaleCondition_Abnorml"
## [83] "SaleCondition_Normal"  "SaleCondition_Partial"
```

```
# Similar to rfe, there are methods for predictors, densityplot, histogram and varImp
```

Selección por filtros Recursiva con random forest y validación cruzada

```
ctrl <- sbfControl(functions = rfSBF
                            , method = "repeatedcv"
                            , number = particiones
                            , repeats = repeticiones
                            , verbose = FALSE
                            , saveDetails = TRUE)

t <- proc.time() # Inicia el cronómetro
F03_6_sbf_rf <- sbf(SalePrice ~ .
              , data = dsTrain.training
              , sbfControl = ctrl
              )
proc.time()-t    # Detiene el cronómetro
```

```
##    user  system elapsed
##  138.61    0.62  139.42
```

```
# Tiempo de ejecución
# user  system elapsed
#  304.03    4.96  308.92

# Guardo resultado del calculo
fileOuput <- paste(dirSalida,'F03_6_sbf_rf.RData',sep="/")
save(F03_6_sbf_rf, file = fileOuput)
```

Estudio de resultados

```r
F03_6_sbf_rf
```

```
##
## Selection By Filter
##
## Outer resampling method: Cross-Validated (5 fold, repeated 5 times)
##
## Resampling performance:
##
##    RMSE Rsquared     MAE  RMSESD RsquaredSD    MAESD
##  0.1387   0.8891 0.09544 0.01302    0.02286 0.007464
##
## Using the training set, 79 variables were selected:
##    LotFrontage, LotArea, OverallQual, OverallCond, YearBuilt...
##
## During resampling, the top 5 selected variables (out of a possible 82):
##    BedroomAbvGr (100%), BldgType_1Fam (100%), BsmtExposure (100%), BsmtFinSF1 (100%), BsmtFinType1 (
##
## On average, 76.4 variables were selected (min = 73, max = 80)
```

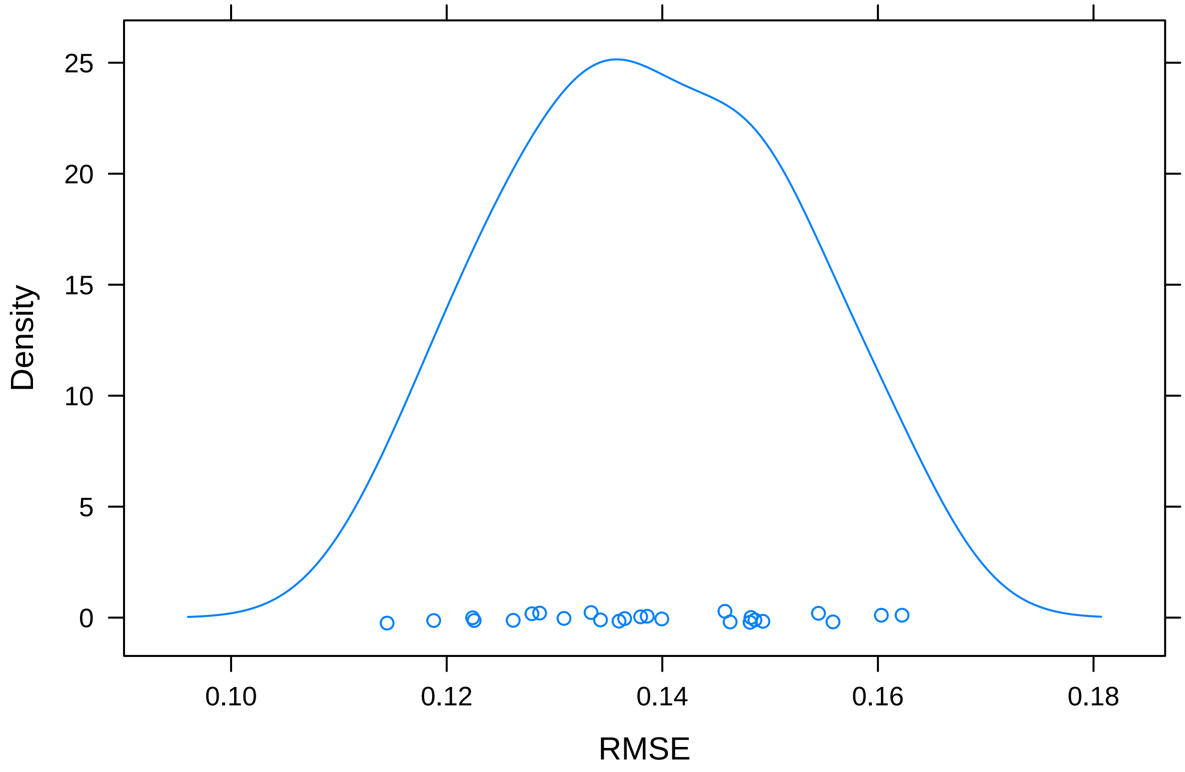
```r
summary(F03_6_sbf_rf)
```

```
##              Length Class        Mode
## pred          3     -none-       list
## variables    25     -none-       list
## results       6     data.frame   list
## fit          17     randomForest list
## optVariables 79     -none-       character
## call          4     -none-       call
## control      14     -none-       list
## resample      4     data.frame   list
## metrics       3     -none-       character
## times         3     -none-       list
## resampledCM   0     -none-       NULL
## obsLevels     0     -none-       NULL
## dots          0     -none-       list
## terms         3     terms        call
## coefnames    92     -none-       character
## xlevels       0     -none-       list
```

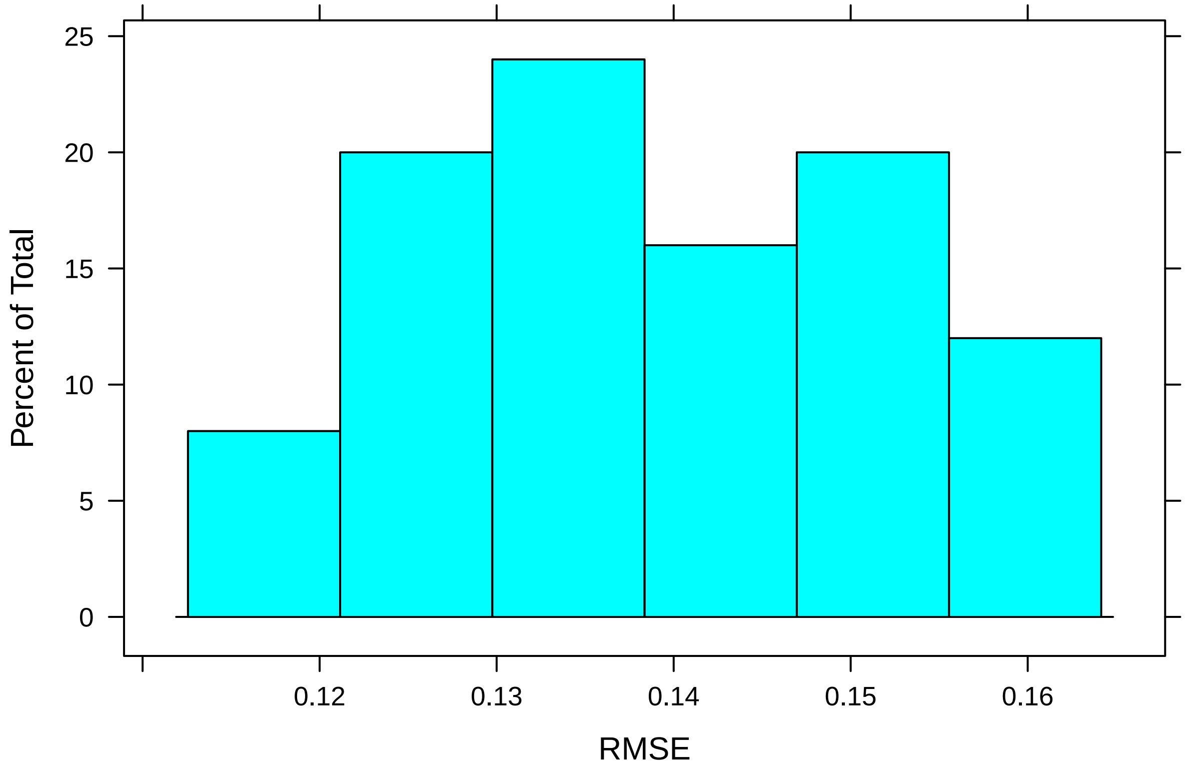
```r
F03_6_sbf_rf$optVariables
```

```
##  [1] "LotFrontage"           "LotArea"
##  [3] "OverallQual"           "OverallCond"
##  [5] "YearBuilt"             "YearRemodAdd"
##  [7] "MasVnrArea"            "ExterCond"
##  [9] "BsmtQual"              "BsmtExposure"
## [11] "BsmtFinType1"          "BsmtFinSF1"
## [13] "BsmtUnfSF"             "TotalBsmtSF"
## [15] "HeatingQC"             "CentralAir"
## [17] "Electrical"            "X1stFlrSF"
## [19] "X2ndFlrSF"             "GrLivArea"
## [21] "BsmtFullBath"          "HalfBath"
## [23] "BedroomAbvGr"          "TotRmsAbvGrd"
## [25] "Fireplaces"            "FireplaceQu"
## [27] "GarageFinish"          "GarageCars"
## [29] "GarageArea"            "GarageQual"
## [31] "GarageCond"            "PavedDrive"
## [33] "WoodDeckSF"            "MSSubClass_120"
## [35] "MSSubClass_20"         "MSSubClass_50"
## [37] "MSSubClass_60"         "MSZoning_RL"
## [39] "MSZoning_RM"           "LotConfig_CulDSac"
## [41] "LotConfig_Inside"      "Neighborhood_CollgCr"
## [43] "Neighborhood_Edwards"  "Neighborhood_Gilbert"
## [45] "Neighborhood_NAmes"    "Neighborhood_NridgHt"
## [47] "Neighborhood_OldTown"  "Neighborhood_Sawyer"
## [49] "Neighborhood_Somerst"  "Condition1_Feedr"
## [51] "Condition1_Norm"       "BldgType_1Fam"
## [53] "HouseStyle_1.5Fin"     "HouseStyle_1Story"
## [55] "HouseStyle_2Story"     "RoofStyle_Gable"
## [57] "RoofStyle_Hip"         "Exterior1st_HdBoard"
## [59] "Exterior1st_MetalSd"   "Exterior1st_VinylSd"
## [61] "Exterior1st_WdSdng"    "Exterior2nd_MetalSd"
## [63] "Exterior2nd_VinylSd"   "Exterior2nd_WdSdng"
## [65] "MasVnrType_BrkFace"    "MasVnrType_None"
## [67] "MasVnrType_Stone"      "Foundation_BrkTil"
## [69] "Foundation_CBlock"     "Foundation_PConc"
## [71] "GarageType_Attchd"     "GarageType_BuiltIn"
## [73] "GarageType_Detchd"     "GarageType_None"
## [75] "SaleType_New"          "SaleType_WD"
## [77] "SaleCondition_Abnorml" "SaleCondition_Normal"
## [79] "SaleCondition_Partial"
```

```r
densityplot(F03_6_sbf_rf)
```



```
histogram(F03_6_sbf_rf)
```



```
predictors(F03_6_sbf_rf)
```

```
##  [1] "LotFrontage"          "LotArea"
##  [3] "OverallQual"          "OverallCond"
##  [5] "YearBuilt"            "YearRemodAdd"
##  [7] "MasVnrArea"           "ExterCond"
##  [9] "BsmtQual"             "BsmtExposure"
## [11] "BsmtFinType1"         "BsmtFinSF1"
## [13] "BsmtUnfSF"            "TotalBsmtSF"
## [15] "HeatingQC"            "CentralAir"
## [17] "Electrical"           "X1stFlrSF"
## [19] "X2ndFlrSF"            "GrLivArea"
## [21] "BsmtFullBath"         "HalfBath"
## [23] "BedroomAbvGr"         "TotRmsAbvGrd"
## [25] "Fireplaces"           "FireplaceQu"
## [27] "GarageFinish"         "GarageCars"
## [29] "GarageArea"           "GarageQual"
## [31] "GarageCond"           "PavedDrive"
## [33] "WoodDeckSF"           "MSSubClass_120"
## [35] "MSSubClass_20"        "MSSubClass_50"
## [37] "MSSubClass_60"        "MSZoning_RL"
## [39] "MSZoning_RM"          "LotConfig_CulDSac"
## [41] "LotConfig_Inside"     "Neighborhood_CollgCr"
## [43] "Neighborhood_Edwards" "Neighborhood_Gilbert"
## [45] "Neighborhood_NAmes"   "Neighborhood_NridgHt"
## [47] "Neighborhood_OldTown"  "Neighborhood_Sawyer"
## [49] "Neighborhood_Somerst"  "Condition1_Feedr"
## [51] "Condition1_Norm"       "BldgType_1Fam"
## [53] "HouseStyle_1.5Fin"     "HouseStyle_1Story"
## [55] "HouseStyle_2Story"     "RoofStyle_Gable"
## [57] "RoofStyle_Hip"         "Exterior1st_HdBoard"
## [59] "Exterior1st_MetalSd"   "Exterior1st_VinylSd"
## [61] "Exterior1st_WdSdng"    "Exterior2nd_MetalSd"
## [63] "Exterior2nd_VinylSd"   "Exterior2nd_WdSdng"
## [65] "MasVnrType_BrkFace"    "MasVnrType_None"
## [67] "MasVnrType_Stone"      "Foundation_BrkTil"
## [69] "Foundation_CBlock"     "Foundation_PConc"
## [71] "GarageType_Attchd"     "GarageType_BuiltIn"
## [73] "GarageType_Detchd"     "GarageType_None"
## [75] "SaleType_New"          "SaleType_WD"
## [77] "SaleCondition_Abnorml" "SaleCondition_Normal"
## [79] "SaleCondition_Partial"
```

Comparación de selección de caracteristicas

Comparación de variables seleccionadas según origen

Selección final de variables

De los conjuntos de variables seleccionados cojo el que mejor

Selección RFE

```
select(resumenResultatos, modelo, tipo, Variables, RMSE) %>% arrange(RMSE)
```

```
##         modelo             tipo Variables      RMSE
## 1 F03_1_rfe_lm    mejorAbsoluto        92 0.1285633
## 2 F03_1_rfe_lm mejorRendimiento        92 0.1285633
## 3 F03_2_rfe_rf    mejorAbsoluto        60 0.1362473
## 4 F03_2_rfe_rf mejorRendimiento        18 0.1381177
```

```
# Origen F02_01_dsDataAll
# F03_1_rfe_lm Devuelve todas las filas por lo que no se coje
# F03_2_rfe_rf Cojemos dos conjuntos:

# mejor rendimiento (18 predictores)

# ESTE CODIGO DE SELECCIÓN SE EJECUTA MANUALMENTE DESPUES DE REALIZAR LOS MODELOS
# dsVarSel <- as.data.frame(F03_2_rfe_rf$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo)) %>%
#   top_n(-18, Orden)
#
# dsDataAllVarSel <- dsDataAll %>%
#     select(SalePrice, indTrain, Id, c(dsVarSel$Campo))
```

```
#
# # Guardo resultado del calculo
# fileOuput <- paste(dirSalida,'F03_11_dsDataSelVar_rfe_MejorRendimiento_top18.RData',sep="/")
# save(dsDataAllVarSel, file = fileOuput)
#
# # mejor absoluto (60 predictores)
#
# dsVarSel <- as.data.frame(F03_2_rfe_rf$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo)) %>%
#   top_n(-60, Orden)
#
# dsDataAllVarSel <- dsDataAll %>%
#     select(SalePrice, indTrain, Id, c(dsVarSel$Campo))
#
# # Guardo resultado del calculo
# fileOuput <- paste(dirSalida,'F03_12_dsDataSelVar_rfe_MenorRMSE_top60.RData',sep="/")
# save(dsDataAllVarSel, file = fileOuput)


# Origen F02_01_dsDataAll
# F03_1_rfe_lm Devuelve todas las filas por lo que no se coje
# F03_2_rfe_rf Cojemos solo 2 conjunto:

# F03_1_rfe_lm  mejorAbsoluto   86  0.1257209
# F03_1_rfe_lm  mejorRendimiento    86  0.1257209
# F03_2_rfe_rf  mejorAbsoluto   55  0.1365676
# F03_2_rfe_rf  mejorRendimiento    18  0.1375916

# En este caso escogemos un solo conjunto mejor rendimiento (90 predictores)

# ESTE CODIGO DE SELECCIÓN SE EJECUTA MANUALMENTE DESPUES DE REALIZAR LOS MODELOS
# dsVarSel <- as.data.frame(F03_2_rfe_rf$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo)) %>%
#   top_n(-18, Orden)
#
# dsDataAllVarSel <- dsDataAll %>%
#     select(SalePrice, indTrain, Id, c(dsVarSel$Campo))
#
# # Guardo resultado del calculo
# fileOuput <- paste(dirSalida,'F03_11_dsDataSelVar_rfe_MejorRendimiento_top18.RData',sep="/")
# save(dsDataAllVarSel, file = fileOuput)
#
# # mejor absoluto (60 predictores)
#
# dsVarSel <- as.data.frame(F03_2_rfe_rf$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo)) %>%
#   top_n(-55, Orden)
```

```
#
# dsDataAllVarSel <- dsDataAll %>%
#     select(SalePrice, indTrain, Id, c(dsVarSel$Campo))
#
# # Guardo resultado del calculo
# fileOuput <- paste(dirSalida,'F03_12_dsDataSelVar_rfe_MenorRMSE_top55.RData',sep="/")
# save(dsDataAllVarSel, file = fileOuput)
```

Selección Algoritmos Genéticos

```
# La selección mediante algoritmos geneticos solo se ejecuta con el conjunto F02_01_dsDataAll
# y se guardan las variables seleccionadas

# ESTE CODIGO DE SELECCIÓN SE EJECUTA MANUALMENTE DESPUES DE REALIZAR LOS MODELOS
# dsDataAllVarSel <- dsDataAll %>%
#     select(SalePrice, indTrain, Id, c(F03_4_ga_100$optVariables))
#
# # Guardo resultado del calculo
# fileOuput <- paste(dirSalida,'F03_13_dsDataSelVar_ga_100_46.RData',sep="/")
# save(dsDataAllVarSel, file = fileOuput)
```

Selección SBF

```
F03_5_sbf_lm
```

```
##
## Selection By Filter
##
## Outer resampling method: Cross-Validated (5 fold, repeated 5 times)
##
## Resampling performance:
##
##    RMSE Rsquared     MAE  RMSESD RsquaredSD    MAESD
##  0.1319   0.8952 0.09504 0.01023    0.01365 0.004632
##
## Using the training set, 84 variables were selected:
##    LotFrontage, LotArea, LotShape, OverallQual, YearBuilt...
##
## During resampling, the top 5 selected variables (out of a possible 85):
##    BedroomAbvGr (100%), BldgType_1Fam (100%), BsmtExposure (100%), BsmtFinSF1 (100%), BsmtFinType1 (
##
## On average, 81.2 variables were selected (min = 78, max = 84)
```

```
F03_6_sbf_rf
```

```
##
## Selection By Filter
##
## Outer resampling method: Cross-Validated (5 fold, repeated 5 times)
##
## Resampling performance:
##
```

```
##    RMSE Rsquared     MAE  RMSESD RsquaredSD    MAESD
##  0.1387   0.8891 0.09544 0.01302    0.02286 0.007464
##
## Using the training set, 79 variables were selected:
##    LotFrontage, LotArea, OverallQual, OverallCond, YearBuilt...
##
## During resampling, the top 5 selected variables (out of a possible 82):
##    BedroomAbvGr (100%), BldgType_1Fam (100%), BsmtExposure (100%), BsmtFinSF1 (100%), BsmtFinType1 (
##
## On average, 76.4 variables were selected (min = 73, max = 80)
```

```
# Origen F02_01_dsDataAll
# F03_5_sbf_lm  variables 84 RMSE 0.1319
# F03_6_sbf_rf  variables 79 RMSE 0.1389
# No seleccionamos conjunto

# Origen F02_03_dsDataAll_Recipe
# F03_5_sbf_lm  variables 80 RMSE 0.1316
# F03_6_sbf_rf  variables 76 RMSE 0.1382
# No seleccionamos conjunto
```

```
# Mezcla

# Se realiza a mano

# dsVarSel001 <- as.data.frame(F03_1_rfe_lm$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo))
#
# dsVarSel002 <- as.data.frame(F03_2_rfe_rf$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo))
#
# dsVarSel004 <- as.data.frame(F03_4_ga_100$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo))
#
# dsVarSel005 <- as.data.frame(F03_5_sbf_lm$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo))
#
# dsVarSel006 <- as.data.frame(F03_6_sbf_rf$optVariables) %>%
#   rename(Campo = 1) %>%
#   rownames_to_column("Orden") %>%
#   mutate(Orden = as.numeric(Orden), Campo = as.character(Campo))
#
# dsVarSelMix30 <- bind_rows(dsVarSel001,dsVarSel002,dsVarSel005,dsVarSel005) %>%
#   group_by(Campo) %>%
#   dplyr::summarise(Orden = mean(Orden)) %>%
#   arrange(Orden) %>%
#   top_n(-30, Orden)
#
# dsDataAllVarSel <- dsDataAll %>%
#     select(SalePrice, indTrain, Id, c(dsVarSelMix30$Campo))
#
# # Guardo resultado del calculo
# fileOuput <- paste(dirSalida,'F03_14_dsDataSelVar_mezcla_31.RData',sep="/")
# save(dsDataAllVarSel, file = fileOuput)
```

Guardamos tambien el dataset completo

```
dsDataAllVarSel <- dsDataAll

# Guardo resultado del calculo
fileOuput <- paste(dirSalida,'F03_15_dsDataSelVar_Completo.RData',sep="/")
save(dsDataAllVarSel, file = fileOuput)
```