Predicción de Abandono

Isidoro Garcia

2021

```
library(tidyverse)
library(data.table)
library(broom)
library(knitr)
library(lubridate)
library(RCT)
library(gamlr)
library(ranger)
library(tree)
library(parallel)
library(tidymodels)
library(xgboost)
```

Contexto

Cell2Cell es una compañía de teléfonos celulares que intenta mitigar el abandono de sus usuarios. Te contratan para 1) Encontrar un modelo que prediga el abandono con acierto y para usar los insights de este modelo para proponer una estrategia de manejo de abandono.

Las preguntas que contestaremos son:

1. Se puede predecir el abandono con los datos que nos compartieron?
2. Cuáles son las variables que explican en mayor medida el abandono?
3. Qué incentivos da Cell2Cell a sus usarios para prevenir el abandono?
4. Cuál es el valor de una estrategia de prevención de abandono focalizada y cómo difiere entre los segmentos de los usuarios? Qué usuarios deberían de recibir incentivos de prevención? Qué montos de incentivos

Nota: Voy a evaluar las tareas con base en la respuesta a cada pregunta. Como hay algunas preguntas que no tienen una respuesta clara, al final ponderaré de acuerdo al poder predictivo de su modelo vs las respuestas sugeridas.

Datos

Los dotos los pueden encontrar en `Cell2Cell.Rdata`. En el archivo `Cell2Cell-Database-Documentation.xlsx` pueden encontrar documentación de la base de datos.

Cargemos los datos

```
load('Bases input/Cell2Cell.Rdata')
```

1. Qué variables tienen missing values? Toma alguna decisión con los missing values. Justifica tu respuesta

```
missings<-map_dbl(cell2cell %>% select_all(),
                  ~100*sum(is.na(.)/nrow(cell2cell)))

(missings<-missings[missings>0])
```

```
    revenue         mou    recchrge    directas     overage        roam
0.304024097 0.304024097 0.304024097 0.304024097 0.304024097 0.304024097
    changem     changer      phones      models     eqpdays        age1
0.706574521 0.706574521 0.001407519 0.001407519 0.001407519 1.750953594
       age2
1.750953594
```

```
# Revenue, mou, rcchrge, directas, overage, roam, changem changer, phones, models w/zero
cell2cell <-
   cell2cell %>%
   mutate(across(names(missings[1:10]), ~if_else(is.na(.), 0, as.double(.))))

# eqpdays, age1, age2, quitar
cell2cell<-
   cell2cell %>%
   filter(!is.na(eqpdays), !is.na(age1), !is.na(age2))
```

2. Tabula la distribución de la variable `churn`. Muestra la frecuencia absoluta y relativa. Crees que se debe hacer oversampling/undersamping?

```
kable(cell2cell %>%
         group_by(churn) %>%
         summarise(n = n()) %>%
         mutate(share = 100*n/sum(n)), digits = 2)
```

churn	n	share
0	49534	70.96
1	20268	29.04

3. (2 pts) Divide tu base en entrenamiento y validación (80/20). Además, considera hacer oversampling (SMOTE) o undersampling. (Tip: Recuerda que el objetivo final es tener muestra ~balanceada en el traning set. En el validation la distribución debe ser la original)

La distribución esta 70% vs 30%. Si queremos construir un training set = 80%. Dentro del training, hay que hacer el undersampling tal que se balanceen las clases.

```
set_validation <-
   treatment_assign(data = cell2cell,
                    share_control = 0.8,
                    n_t = 1, strata_varlist = 'customer',
                    seed = 1908, key = 'customer')
```

```
Warning: Unknown or uninitialised column: `treat`.
```

```
Warning in if (missfits == "NA") {: the condition has length > 1 and only the
first element will be used
```

```
Warning in if (missfits == "global") {: the condition has length > 1 and only
the first element will be used
```

```
Warning: Unknown or uninitialised column: `treat`.
```

```
set_validation<-set_validation$data
cell2cell<-
   left_join(cell2cell, set_validation %>%
                ungroup %>%
                select(-c(strata, missfit)))

# Divido entre training y validation
cell2cell_V<-
   cell2cell %>%
   filter(treat == 1) %>%
   select(-treat)

rm(set_validation)

cell2cell_T<-
   cell2cell %>%
   filter(treat==0) %>%
   select(-treat)

# Undersampling
cell2cell_1_T<-
   cell2cell_T %>%
   filter(churn==1)

undersampling<-
   treatment_assign(data = cell2cell_T %>% filter(churn==0),
                    share_control = 0.41,
                    n_t = 1,
                    strata_varlist = 'customer', seed = 19987, key = 'customer')
```

```
Warning: Unknown or uninitialised column: `treat`.
```

```
Warning in if (missfits == "NA") {: the condition has length > 1 and only the
first element will be used
```

```
Warning in if (missfits == "global") {: the condition has length > 1 and only
the first element will be used
```

```
Warning: Unknown or uninitialised column: `treat`.
```

```
undersampling<-undersampling$data
```

```
cell2cell_T<-left_join(cell2cell_T,
                       undersampling %>%
                          ungroup() %>%
                          select(-strata, -missfit))

cell2cell_T<-bind_rows(cell2cell_T %>% filter(treat ==0), cell2cell_1_T)
table(cell2cell_T$churn)
```

```
    0     1
16272 16153
```

```
rm(cell2cell_1_T, undersampling)
```

Model estimation

Pondremos a competir 3 modelos:

1. Cross-Validated LASSO-logit
2. Prune Trees
3. Random Forest

4.GBM

4 (2 pts). Estima un cross validated LASSO. Muestra el la gráfica de CV Binomial Deviance vs Complejidad

```
# Matriz de covariates
cell2cell_T$treat<-NULL

X<-
   cell2cell_T %>%
   ungroup() %>%
   select(-customer, -churn)

X<-sparse.model.matrix(~.+0, data = X)
dim(X)
```

```
[1] 32425    66
```

```
churn<-cell2cell_T$churn

# CV LASSO
detectCores()
```

```
[1] 12
```

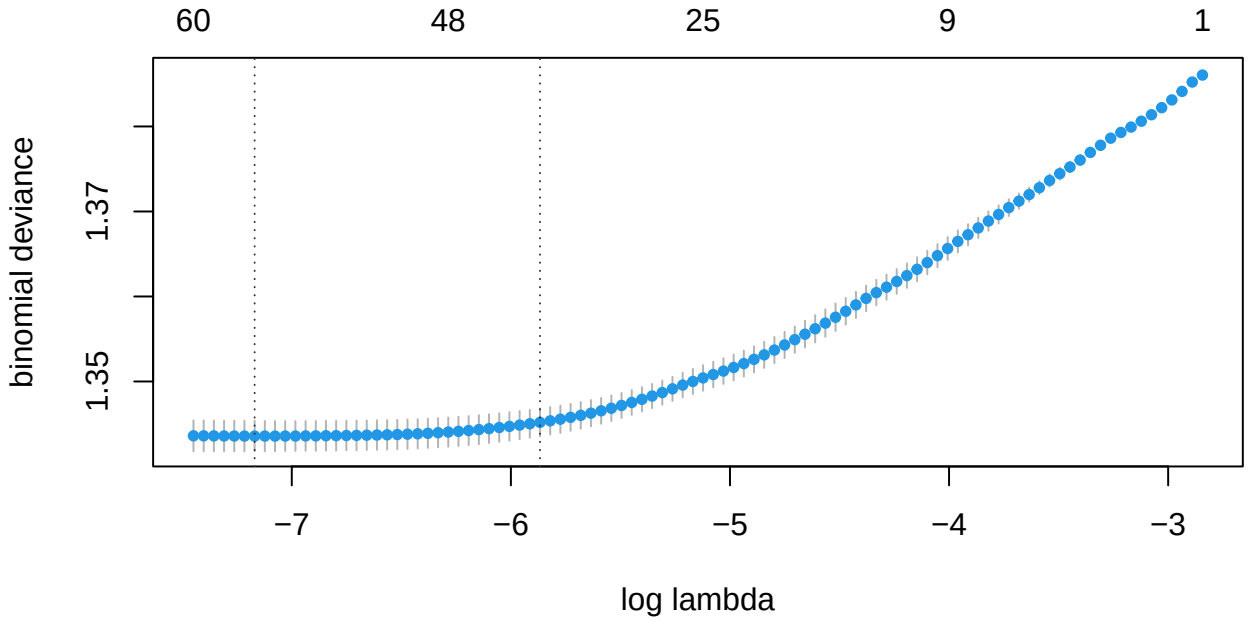
```
cl<-makeCluster(12)

inicio<-Sys.time()
lasso<-cv.gamlr(x = X, y = churn, verb = T, cl = cl, family = 'binomial')
```

```
fold done.
```

```
(tiempo<-Sys.time() - inicio)
```

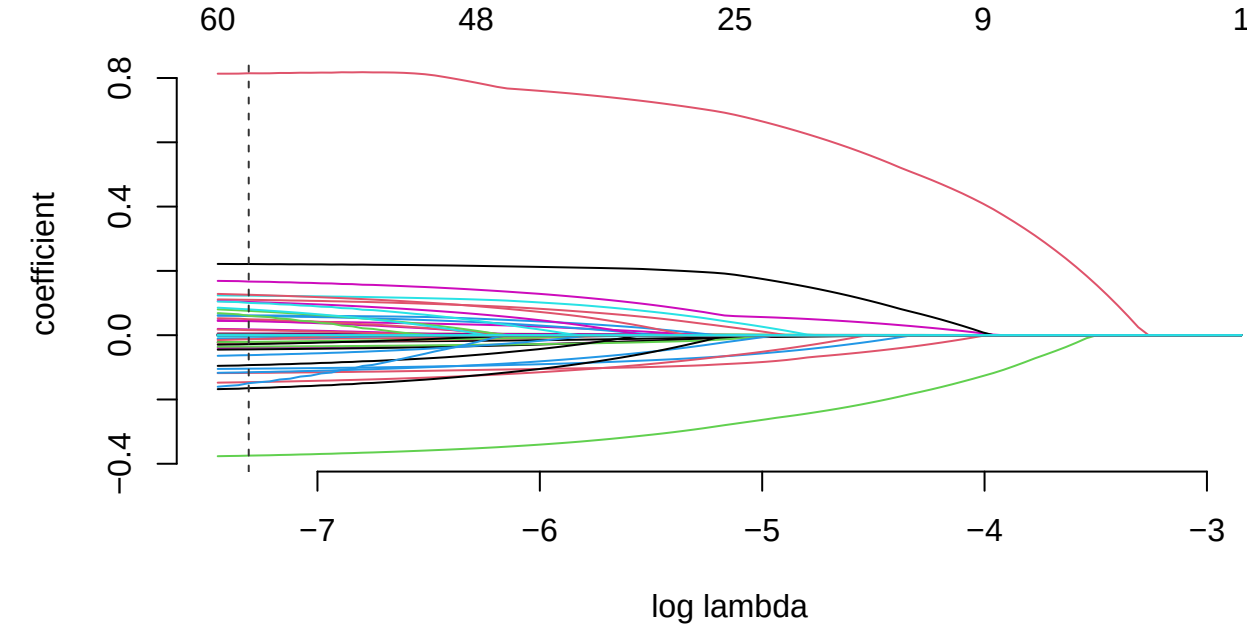
```
Time difference of 11.49308 secs
```

```r
stopCluster(cl)

save(lasso, file = 'Modelos/cv_lasso.Rdata')

# Plot del CV binomial dev
plot(lasso)
```



5. Grafica el Lasso de los coeficientes vs la complejidad del modelo.

```r
plot(lasso$gamlr)
```



6 (2 pts). Cuál es la λ resultante? Genera una tabla con los coeficientes que selecciona el CV LASSO. Cuántas variables deja iguales a cero? Cuales son las 3 variables más importantes para predecir el abandono? Da una explicación intuitiva a la última pregunta

Las variables más relevantes son:

- `retcall` [El usuario ya recibió una llamada del equipo de retención]: Esto puede indicar que el usuario está próximo a terminar contrato o que las llamadas del equipo de retención en verdad tienen un efecto contraproducente en términos de la supervivencia del usuario.
- `creditaa` [El usuario tiene la calificación crediticia más alta]: Esta disminuye la probabilidad de abandono. Muy útil en programas de pos-pago.
- `refurb` [El auricular del teléfono es reparado]: Esto podría indicar una experiencia negativa de uso de estos auriculares.

```
lasso$lambda.min
```

```
[1] 0.0007696674
```

```
coeficientes<-as.matrix(coef(lasso, select = 'min'))

coeficientes<-tibble(variable = rownames(coeficientes),
                     coeficiente = coeficientes[,1])

coeficientes<-coeficientes %>% filter(coeficiente !=0)

kable(
   coeficientes<-
   coeficientes %>%
   arrange(desc(abs(coeficiente))))
```

variable	coeficiente
retcall	0.8156834
creditaa	-0.3726389
refurb	0.2208763
uniqsubs	0.1647732
actvsubs	-0.1615133
refer	-0.1443873
retaccpt	-0.1378846
occcler	0.1226888
children	0.1220908
mailres	-0.1165030
credita	-0.1133504
prizmrur	0.1086174
webcap	-0.1034437
mcycle	0.1001941
creditcd	0.0959783
setprcm	-0.0905156
occhmkr	0.0740675
occstud	0.0702888
phones	0.0610163
occret	-0.0597822
rv	0.0581167
intercept	0.0554109
retcalls	0.0516047
marryyes	0.0463030

variable	coeficiente
mailord	-0.0433610
prizmtwn	0.0427891
marryun	0.0412705
threeway	-0.0382060
prizmub	-0.0350963
travel	-0.0258979
callfwdv	-0.0221297
months	-0.0211867
newcelly	-0.0199876
pcown	0.0162476
models	0.0138686
income	-0.0111488
mailflag	-0.0094103
newcelln	-0.0089338
roam	0.0049909
dropvce	0.0049693
age1	-0.0041687
custcare	-0.0036080
changer	0.0023073
recchrge	-0.0021778
dropblk	0.0018898
eqpdays	0.0014490
directas	-0.0012966
overage	0.0011625
truck	0.0009712
age2	-0.0009244
incalls	-0.0008989
revenue	0.0004786
changem	-0.0004677
peakvce	-0.0003127
mou	-0.0002350
setprc	0.0002069
unansvce	0.0001318
outcalls	-0.0000402
blckvce	0.0000014

7. Genera un data frame (usando el validation set) que tenga: `customer`, `churn` y las predicciones del LASSO.

```
X<-
   cell2cell_V %>%
   select(-customer, -churn)

X<-sparse.model.matrix(~.+0, data = X)

table(cell2cell_V$churn)
```

```
   0    1
9846 4115
```

```
predicciones<-
   cell2cell_V %>%
   select(customer, churn) %>%
   mutate(lasso = drop(predict(lasso, newdata = X, select = 'min', type = 'response')))
```

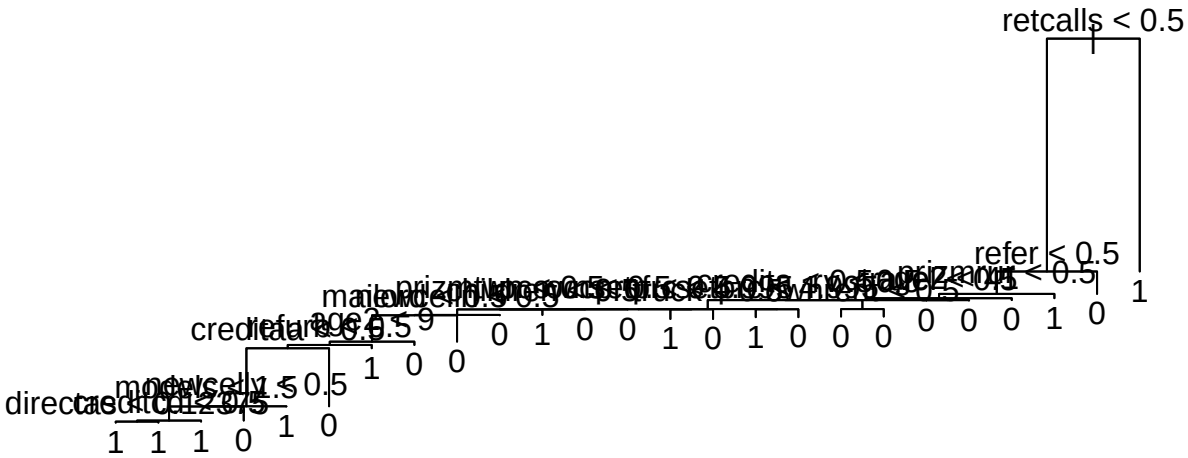
8. Estima ahora tree. Usa `mindev = 0.05, mincut = 1000` Cuántos nodos terminales salen? Muestra el summary del árbol

```
# Primero el árbol
cell2cell_T<-
   cell2cell_T %>%
   mutate(churn = factor(churn, levels = c("0","1")))


arbol<-tree(churn~., split = 'gini',
            cell2cell_T %>% ungroup() %>% select(-customer),
            mindev = 0.05, mincut = 1000)

summary(arbol)
```

```
Classification tree:
tree(formula = churn ~ ., data = cell2cell_T %>% ungroup() %>%
    select(-customer), split = "gini", mindev = 0.05, mincut = 1000)
Variables actually used in tree construction:
 [1] "retcalls" "refer"    "prizmrur" "travel"   "rv"       "credita"
 [7] "truck"    "occprof"  "pcown"    "children" "prizmtwn" "newcelln"
[13] "mailord"  "age2"     "refurb"   "creditaa" "newcelly" "models"
[19] "creditcd" "directas" "setprc"   "ownrent"
Number of terminal nodes:  25
Residual mean deviance:  1.377 = 44630 / 32400
Misclassification error rate: 0.4661 = 15114 / 32425
```

9. Grafica el árbol resultante

```
plot(arbol); text(arbol, pretty = 0)
```

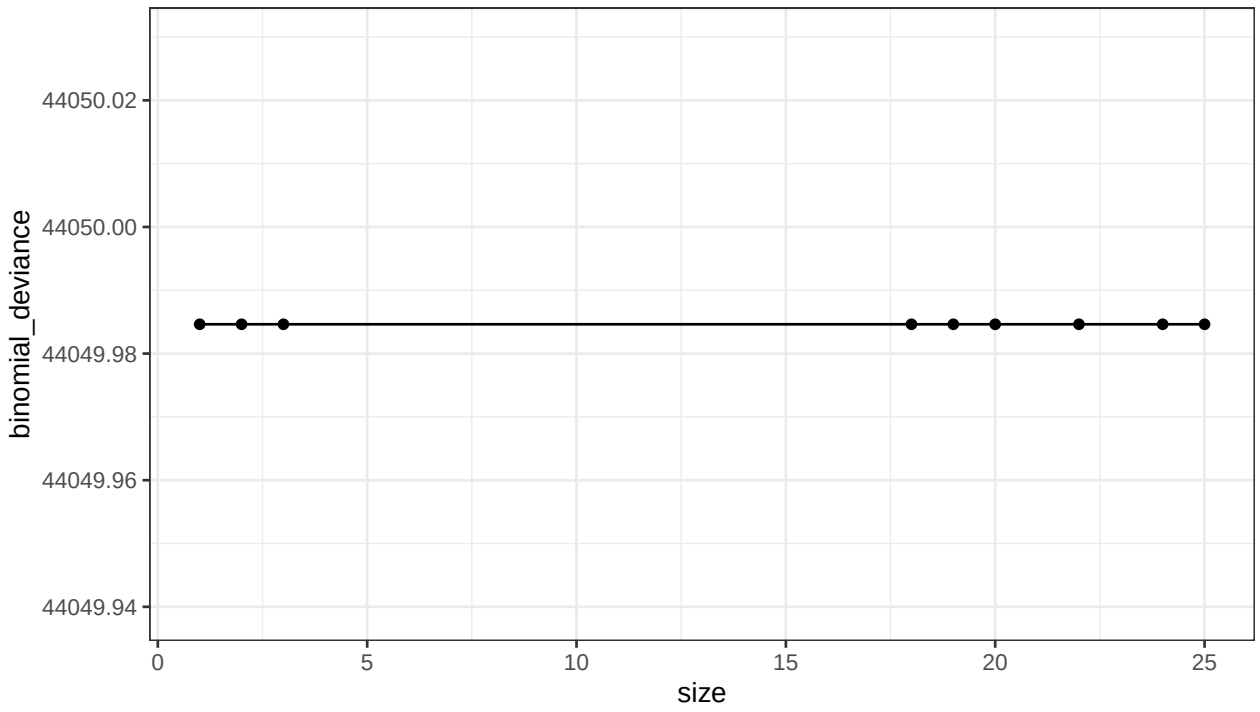


10. Poda el árbol usando CV. Muestra el resultado. Grafica Tree Size vs Binomial Deviance. Cuál es el mejor tamaño del árbol? Mejora el Error?

```
(cv_arbol<-cv.tree(arbol, K = 5))
```

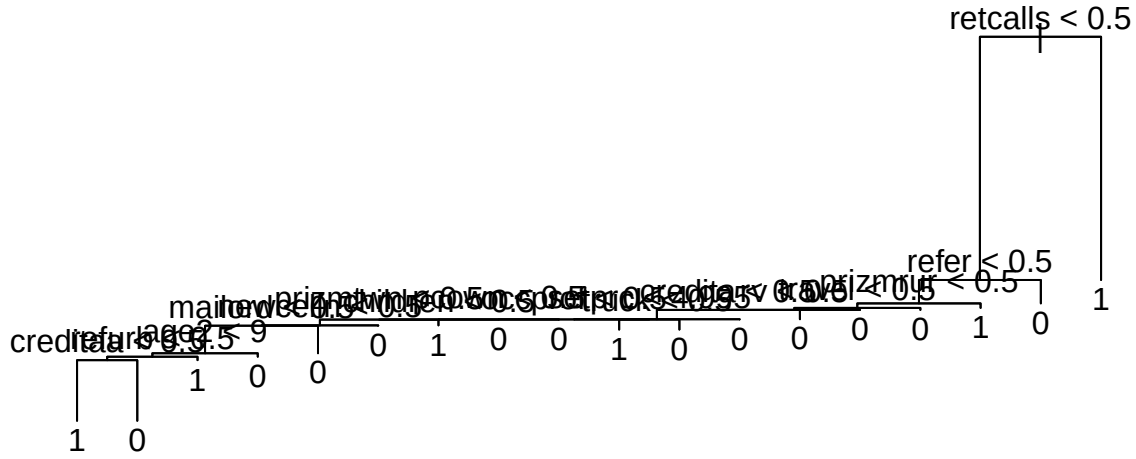
```
$size
[1] 25 24 22 20 19 18  3  2  1

$dev
[1] 44049.98 44049.98 44049.98 44049.98 44049.98 44049.98 44049.98 44049.98
[9] 44049.98

$k
[1]        -Inf   0.2889196   0.8183143   5.6150824   5.8942294   6.2601607
[7]   6.4101217  17.6499275 183.3326909

$method
[1] "deviance"

attr(,"class")
[1] "prune"         "tree.sequence"
```

```
arbol_graf<-
   tibble(size = cv_arbol$size, binomial_deviance = cv_arbol$dev)


ggplot(arbol_graf, aes(size, binomial_deviance))+geom_point()+geom_path()+
   theme_bw()
```



11. Gráfica el árbol final. (Tip: Checa `prune.tree`)

```
arbol_prune<-prune.tree(arbol, best = 4)

plot(arbol_prune); text(arbol_prune, pretty = 0)
```




```
save(arbol_prune, file = 'Modelos/arbol.Rdata')
```

12. Genera las predicciones del árbol pruned. Guardalas en la base de predicciones. Guarda el score y la prediccion categorica en la misma data frame donde guardaste las predicciones del LASSO

```
X<-
   cell2cell_V %>%
   select(-customer, -churn)

predicciones<-
   predicciones %>%
   mutate(arbol_class = drop(predict(arbol_prune, newdata = X, type = 'class')),
          arbol_prob = drop(predict(arbol_prune, newdata = X, type = 'vector'))[,2])
```

13 (4pts). Corre un Random Forest ahora. Cuál es la B para la que ya no ganamos mucho más en poder predictivo?

- Corre para `num.trees=100,200,300, 500, 700, 800`
- En cada caso, guarda únicamente el `prediction.error`
- Recuerda fijar el parámetro `probability-T` para que el RF genere predicciones categoricas y sus probabilidades

```
X<-
   cell2cell_T %>%
   ungroup() %>%
   select(-customer, -churn)

X<-sparse.model.matrix(~.+0, data = X)
dim(X)
```

```
[1] 32425    66
```

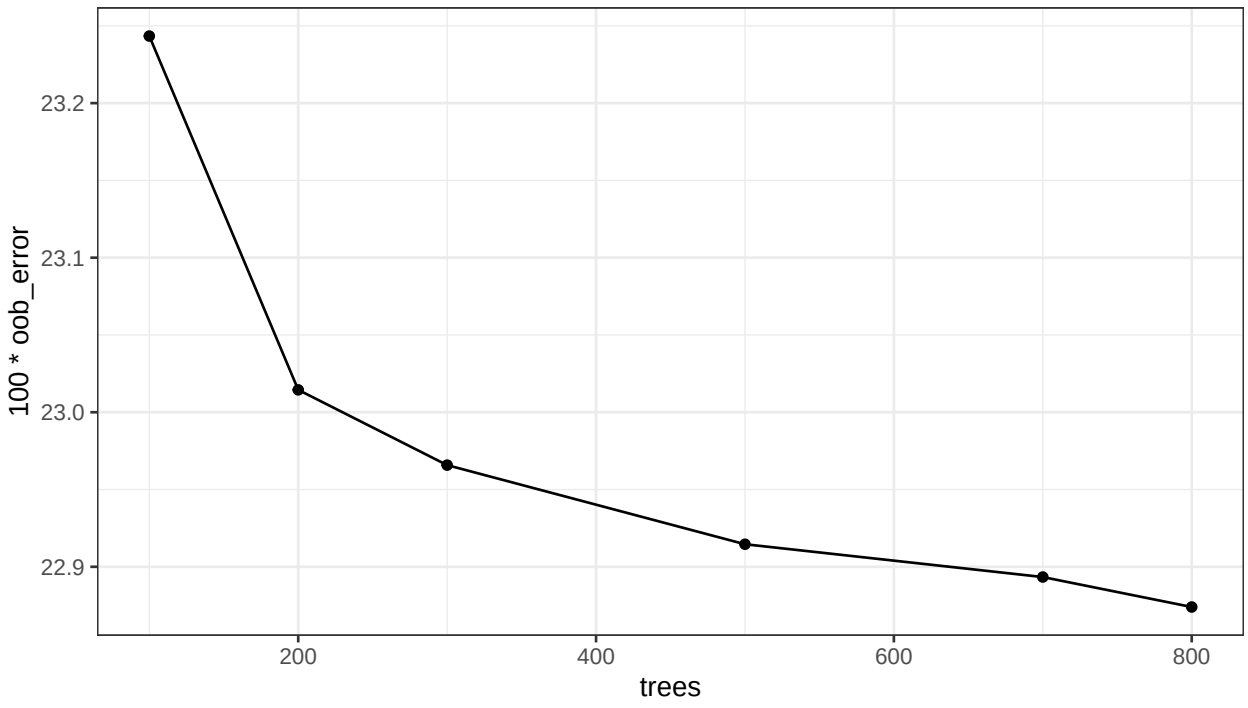
```
churn<-cell2cell_T$churn

# Grid de Random Forests
detectCores()
```

```
[1] 12
```

```
cl<-makeCluster(12)

inicio<-Sys.time()
random_forest<-map(c(100,200,300,500, 700, 800),
                   function(z)
                      ranger(x = X,
                             y = churn,
                             importance = 'impurity',
                             probability = T,
                             num.trees = z))

(tiempo<-Sys.time() - inicio)
```

```
Time difference of 1.423926 mins
```

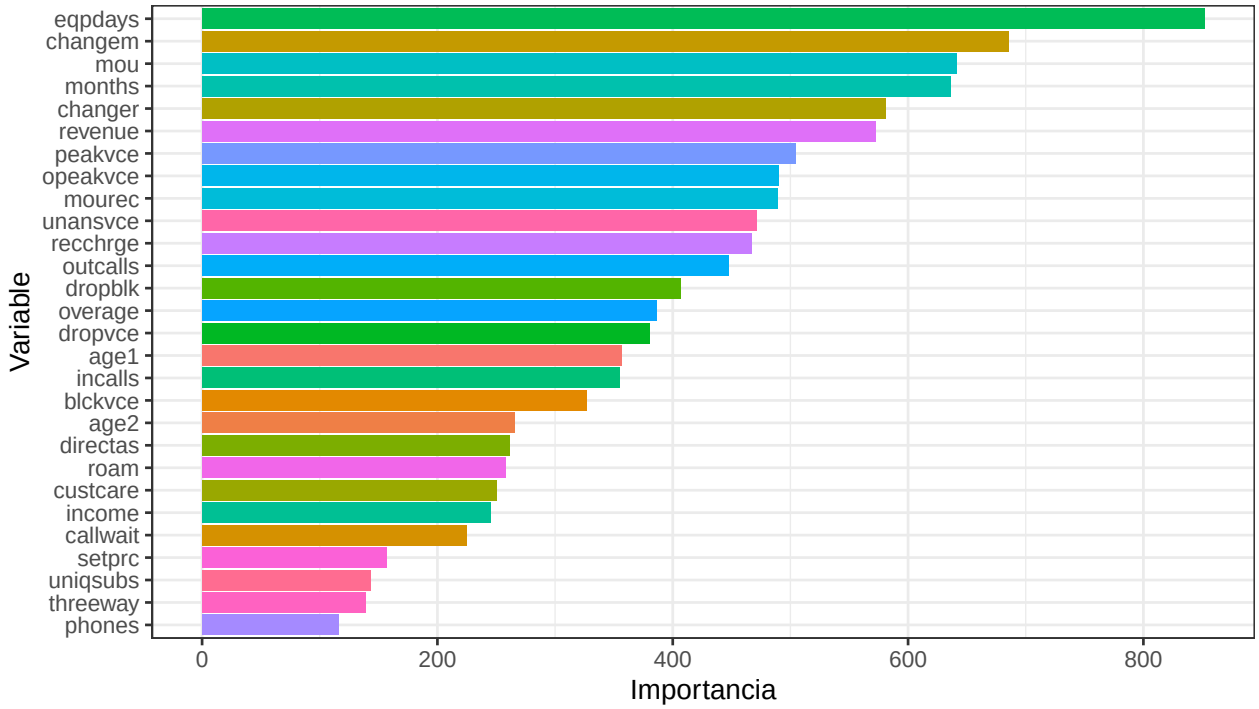
```
stopCluster(cl)

error_prediccion<-tibble(trees = c(100,200,300,500, 700, 800),
                         oob_error = map_dbl(random_forest, ~.$prediction.error))


ggplot(error_prediccion, aes(trees, 100*oob_error))+
   geom_point()+geom_path()+theme_bw()
```



```
random_forest<-random_forest[[6]]

save(random_forest, file = 'Modelos/random_forest.Rdata')
```

14. Escoge un random forest para hacer las predicciones. Grafica la importancia de las variables. Interpreta

```
variable_importance<-random_forest$variable.importance

variable_importance<-tibble(variable = names(variable_importance),
                            importance = variable_importance)

ggplot(variable_importance %>% filter(importance> 100),
       aes(fct_reorder(variable, importance), importance, fill = variable))+
   geom_col()+coord_flip()+theme_bw()+theme(legend.position = 'none')+
   labs(x = 'Variable', y = 'Importancia')
```



15. Genera las predicciones OOS para el random forest. Guardalas en la misma data.frame que los otros modelos

```
# predicciones continuas
X<-
   cell2cell_V %>%
   select(-customer, -churn)

X<-sparse.model.matrix(~.+0, data = X)
```

```
predicciones<-
   predicciones %>%
   mutate(random_forest_prob = predict(random_forest, data = X)$predictions[,2])


rm(a, arbol, arbol_graf, arbol2, cl)
```

```
Warning in rm(a, arbol, arbol_graf, arbol2, cl): object 'a' not found
```

```
Warning in rm(a, arbol, arbol_graf, arbol2, cl): object 'arbol2' not found
```

```
save(predicciones, file = 'Bases output/predicciones.Rdata')
```

16. Estima un GBM.

- Encuenta el número de boosting rounds ideal B
- Encuentra la profundidad de los árboles d
- Estima un grid de modelos para calibrar los dos hiperparametros

```
library(tidymodels)
library(xgboost)


X<-
   cell2cell_T %>%
   select(-customer, -churn)

X<-sparse.model.matrix(~.+0, data = X)

churn<-as.numeric(cell2cell_T$churn)-1


# Creating the grid that covers must of the hyper space
xgb_grid<-grid_latin_hypercube(
                tree_depth(),
                loss_reduction(),
                sample_size = sample_prop(),
                learn_rate(),
                mtry(c(1,ncol(X))),
                size = 15)



# XGBOOST matrix
training<-xgb.DMatrix(data = X, label = churn)


# Validation
X<-
   cell2cell_V %>%
   select(-customer, -churn)

X<-sparse.model.matrix(~.+0, data = X)
```

```
churn<-cell2cell_V$churn

validation<-xgb.DMatrix(data = X, label = churn)

# Watchlist
watchlist<-list(training = training, validation = validation)
# names(watchlist$validation)<- names(watchlist$training)

# transforming xgb_grid to have proper names
xgb_grid<-xgb_grid %>%
                rename(max_depth = tree_depth,
                       gamma = loss_reduction,
                       subsample = sample_size,
                       eta = learn_rate,
                       colsample_bytree = mtry) %>%
                mutate(eval_metric = 'auc',
                       colsample_bytree = colsample_bytree/66)


# Modelo
listas<-map(1:15, ~as.list(xgb_grid[.,]))

a<-Sys.time()
xgb<-map(listas,
             function(x) {
               xgb.train(params = x,
                         data = training,
                         nrounds = 300,
                         objective = "binary:logistic",
                         early_stopping_rounds = 5,
                         verbose = 1,
                         watchlist = watchlist) })
```

```
[1] training-auc:0.647920   validation-auc:0.638442
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.655253   validation-auc:0.643946
[3] training-auc:0.657465   validation-auc:0.646606
[4] training-auc:0.658609   validation-auc:0.647244
[5] training-auc:0.658525   validation-auc:0.647490
[6] training-auc:0.659565   validation-auc:0.647864
[7] training-auc:0.661414   validation-auc:0.649330
[8] training-auc:0.661174   validation-auc:0.648968
[9] training-auc:0.661262   validation-auc:0.649123
[10]    training-auc:0.661270   validation-auc:0.649535
[11]    training-auc:0.661469   validation-auc:0.650153
[12]    training-auc:0.661386   validation-auc:0.650080
[13]    training-auc:0.661811   validation-auc:0.650208
[14]    training-auc:0.662852   validation-auc:0.650594
[15]    training-auc:0.662729   validation-auc:0.651047
[16]    training-auc:0.662641   validation-auc:0.651204
[17]    training-auc:0.662802   validation-auc:0.651015
[18]    training-auc:0.663154   validation-auc:0.651188
```

```
[19]    training-auc:0.663069   validation-auc:0.651122
[20]    training-auc:0.663149   validation-auc:0.651502
[21]    training-auc:0.663263   validation-auc:0.651613
[22]    training-auc:0.663768   validation-auc:0.652130
[23]    training-auc:0.663836   validation-auc:0.652415
[24]    training-auc:0.663840   validation-auc:0.652376
[25]    training-auc:0.663643   validation-auc:0.652292
[26]    training-auc:0.663836   validation-auc:0.652258
[27]    training-auc:0.663979   validation-auc:0.652459
[28]    training-auc:0.664070   validation-auc:0.652422
[29]    training-auc:0.663840   validation-auc:0.652308
[30]    training-auc:0.663807   validation-auc:0.652356
[31]    training-auc:0.663781   validation-auc:0.652489
[32]    training-auc:0.663771   validation-auc:0.652291
[33]    training-auc:0.663985   validation-auc:0.652656
[34]    training-auc:0.663969   validation-auc:0.652700
[35]    training-auc:0.664020   validation-auc:0.652723
[36]    training-auc:0.663987   validation-auc:0.652704
[37]    training-auc:0.664079   validation-auc:0.653017
[38]    training-auc:0.664021   validation-auc:0.653062
[39]    training-auc:0.664016   validation-auc:0.653084
[40]    training-auc:0.664064   validation-auc:0.653089
[41]    training-auc:0.664134   validation-auc:0.653071
[42]    training-auc:0.664210   validation-auc:0.653074
[43]    training-auc:0.664368   validation-auc:0.653172
[44]    training-auc:0.664401   validation-auc:0.653275
[45]    training-auc:0.664708   validation-auc:0.653372
[46]    training-auc:0.664720   validation-auc:0.653368
[47]    training-auc:0.664666   validation-auc:0.653394
[48]    training-auc:0.664707   validation-auc:0.653433
[49]    training-auc:0.664770   validation-auc:0.653481
[50]    training-auc:0.664743   validation-auc:0.653389
[51]    training-auc:0.664765   validation-auc:0.653390
[52]    training-auc:0.664742   validation-auc:0.653485
[53]    training-auc:0.664889   validation-auc:0.653568
[54]    training-auc:0.664816   validation-auc:0.653567
[55]    training-auc:0.664720   validation-auc:0.653495
[56]    training-auc:0.664863   validation-auc:0.653574
[57]    training-auc:0.664857   validation-auc:0.653628
[58]    training-auc:0.664827   validation-auc:0.653604
[59]    training-auc:0.664953   validation-auc:0.653751
[60]    training-auc:0.664934   validation-auc:0.653604
[61]    training-auc:0.664982   validation-auc:0.653627
[62]    training-auc:0.664951   validation-auc:0.653560
[63]    training-auc:0.664955   validation-auc:0.653586
[64]    training-auc:0.665043   validation-auc:0.653683
Stopping. Best iteration:
[59]    training-auc:0.664953   validation-auc:0.653751

[1] training-auc:0.648573   validation-auc:0.626451
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.658551   validation-auc:0.638188
```

```
[3] training-auc:0.673811   validation-auc:0.648961
[4] training-auc:0.680175   validation-auc:0.651717
[5] training-auc:0.683983   validation-auc:0.653528
[6] training-auc:0.686979   validation-auc:0.655199
[7] training-auc:0.688592   validation-auc:0.655133
[8] training-auc:0.690215   validation-auc:0.659423
[9] training-auc:0.691342   validation-auc:0.659476
[10]    training-auc:0.693692   validation-auc:0.660160
[11]    training-auc:0.693783   validation-auc:0.660417
[12]    training-auc:0.695530   validation-auc:0.662007
[13]    training-auc:0.695325   validation-auc:0.662112
[14]    training-auc:0.697257   validation-auc:0.661996
[15]    training-auc:0.697271   validation-auc:0.661977
[16]    training-auc:0.697071   validation-auc:0.662162
[17]    training-auc:0.696962   validation-auc:0.662839
[18]    training-auc:0.696560   validation-auc:0.663050
[19]    training-auc:0.698394   validation-auc:0.664661
[20]    training-auc:0.698551   validation-auc:0.664938
[21]    training-auc:0.697970   validation-auc:0.665033
[22]    training-auc:0.697595   validation-auc:0.665110
[23]    training-auc:0.698105   validation-auc:0.665915
[24]    training-auc:0.697995   validation-auc:0.665842
[25]    training-auc:0.697583   validation-auc:0.666012
[26]    training-auc:0.697163   validation-auc:0.665882
[27]    training-auc:0.696865   validation-auc:0.665384
[28]    training-auc:0.697029   validation-auc:0.665428
[29]    training-auc:0.696834   validation-auc:0.664922
[30]    training-auc:0.696657   validation-auc:0.665057
Stopping. Best iteration:
[25]    training-auc:0.697583   validation-auc:0.666012

[1] training-auc:0.629092   validation-auc:0.603514
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.645093   validation-auc:0.605765
[3] training-auc:0.657370   validation-auc:0.610515
[4] training-auc:0.666984   validation-auc:0.619186
[5] training-auc:0.677609   validation-auc:0.626551
[6] training-auc:0.682118   validation-auc:0.633533
[7] training-auc:0.681807   validation-auc:0.629833
[8] training-auc:0.683806   validation-auc:0.628657
[9] training-auc:0.687251   validation-auc:0.628631
[10]    training-auc:0.691523   validation-auc:0.630434
[11]    training-auc:0.691946   validation-auc:0.632968
Stopping. Best iteration:
[6] training-auc:0.682118   validation-auc:0.633533

[1] training-auc:0.500000   validation-auc:0.500000
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.500000   validation-auc:0.500000
[3] training-auc:0.500000   validation-auc:0.500000
```

```
[4] training-auc:0.500000   validation-auc:0.500000
[5] training-auc:0.500000   validation-auc:0.500000
[6] training-auc:0.500000   validation-auc:0.500000
Stopping. Best iteration:
[1] training-auc:0.500000   validation-auc:0.500000

[1] training-auc:0.500000   validation-auc:0.500000
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.500000   validation-auc:0.500000
[3] training-auc:0.500000   validation-auc:0.500000
[4] training-auc:0.500000   validation-auc:0.500000
[5] training-auc:0.500000   validation-auc:0.500000
[6] training-auc:0.500000   validation-auc:0.500000
Stopping. Best iteration:
[1] training-auc:0.500000   validation-auc:0.500000

[1] training-auc:0.663945   validation-auc:0.623688
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.684106   validation-auc:0.628627
[3] training-auc:0.706486   validation-auc:0.636827
[4] training-auc:0.710539   validation-auc:0.642265
[5] training-auc:0.720622   validation-auc:0.648821
[6] training-auc:0.719707   validation-auc:0.648780
[7] training-auc:0.725983   validation-auc:0.648488
[8] training-auc:0.732200   validation-auc:0.648082
[9] training-auc:0.737685   validation-auc:0.651266
[10]    training-auc:0.740657   validation-auc:0.653223
[11]    training-auc:0.743273   validation-auc:0.651995
[12]    training-auc:0.745803   validation-auc:0.656492
[13]    training-auc:0.747093   validation-auc:0.657539
[14]    training-auc:0.748437   validation-auc:0.657553
[15]    training-auc:0.748987   validation-auc:0.657411
[16]    training-auc:0.749284   validation-auc:0.657144
[17]    training-auc:0.750866   validation-auc:0.656295
[18]    training-auc:0.751530   validation-auc:0.655156
[19]    training-auc:0.751486   validation-auc:0.656547
Stopping. Best iteration:
[14]    training-auc:0.748437   validation-auc:0.657553

[1] training-auc:0.614566   validation-auc:0.608029
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.634041   validation-auc:0.628428
[3] training-auc:0.638050   validation-auc:0.631636
[4] training-auc:0.647608   validation-auc:0.639205
[5] training-auc:0.655020   validation-auc:0.644668
[6] training-auc:0.654699   validation-auc:0.644758
[7] training-auc:0.658061   validation-auc:0.646269
[8] training-auc:0.658935   validation-auc:0.647123
```

```
[9] training-auc:0.659997   validation-auc:0.648829
[10]    training-auc:0.660484   validation-auc:0.650840
[11]    training-auc:0.660658   validation-auc:0.651289
[12]    training-auc:0.662519   validation-auc:0.652941
[13]    training-auc:0.663265   validation-auc:0.653585
[14]    training-auc:0.663574   validation-auc:0.653462
[15]    training-auc:0.663964   validation-auc:0.653454
[16]    training-auc:0.665201   validation-auc:0.654178
[17]    training-auc:0.665830   validation-auc:0.655301
[18]    training-auc:0.667497   validation-auc:0.656995
[19]    training-auc:0.667794   validation-auc:0.656945
[20]    training-auc:0.668790   validation-auc:0.657901
[21]    training-auc:0.669108   validation-auc:0.658255
[22]    training-auc:0.669187   validation-auc:0.658380
[23]    training-auc:0.669596   validation-auc:0.658363
[24]    training-auc:0.669648   validation-auc:0.658333
[25]    training-auc:0.669956   validation-auc:0.658288
[26]    training-auc:0.670361   validation-auc:0.658914
[27]    training-auc:0.670364   validation-auc:0.658673
[28]    training-auc:0.671016   validation-auc:0.659023
[29]    training-auc:0.671208   validation-auc:0.658989
[30]    training-auc:0.671198   validation-auc:0.659562
[31]    training-auc:0.671241   validation-auc:0.659865
[32]    training-auc:0.671295   validation-auc:0.659683
[33]    training-auc:0.671398   validation-auc:0.659953
[34]    training-auc:0.672051   validation-auc:0.660451
[35]    training-auc:0.672464   validation-auc:0.660640
[36]    training-auc:0.672239   validation-auc:0.660781
[37]    training-auc:0.672234   validation-auc:0.661186
[38]    training-auc:0.672709   validation-auc:0.661432
[39]    training-auc:0.673040   validation-auc:0.661737
[40]    training-auc:0.673820   validation-auc:0.662073
[41]    training-auc:0.674090   validation-auc:0.662178
[42]    training-auc:0.674080   validation-auc:0.662351
[43]    training-auc:0.673888   validation-auc:0.662315
[44]    training-auc:0.673764   validation-auc:0.662189
[45]    training-auc:0.673681   validation-auc:0.662261
[46]    training-auc:0.674046   validation-auc:0.662658
[47]    training-auc:0.674255   validation-auc:0.662912
[48]    training-auc:0.674389   validation-auc:0.663028
[49]    training-auc:0.674862   validation-auc:0.663665
[50]    training-auc:0.675665   validation-auc:0.664101
[51]    training-auc:0.675414   validation-auc:0.664126
[52]    training-auc:0.675493   validation-auc:0.664072
[53]    training-auc:0.675966   validation-auc:0.664574
[54]    training-auc:0.676288   validation-auc:0.664660
[55]    training-auc:0.676500   validation-auc:0.664765
[56]    training-auc:0.676546   validation-auc:0.664658
[57]    training-auc:0.676660   validation-auc:0.664882
[58]    training-auc:0.676850   validation-auc:0.664964
[59]    training-auc:0.677068   validation-auc:0.664974
[60]    training-auc:0.677428   validation-auc:0.665183
[61]    training-auc:0.677749   validation-auc:0.665345
[62]    training-auc:0.677700   validation-auc:0.665397
```

```
[63]    training-auc:0.678041   validation-auc:0.665452
[64]    training-auc:0.678134   validation-auc:0.665755
[65]    training-auc:0.678334   validation-auc:0.665813
[66]    training-auc:0.678567   validation-auc:0.665990
[67]    training-auc:0.678699   validation-auc:0.665920
[68]    training-auc:0.678699   validation-auc:0.665920
[69]    training-auc:0.678748   validation-auc:0.665996
[70]    training-auc:0.679208   validation-auc:0.666306
[71]    training-auc:0.679443   validation-auc:0.666380
[72]    training-auc:0.679442   validation-auc:0.666380
[73]    training-auc:0.679529   validation-auc:0.666374
[74]    training-auc:0.679529   validation-auc:0.666374
[75]    training-auc:0.679529   validation-auc:0.666374
[76]    training-auc:0.679589   validation-auc:0.666518
[77]    training-auc:0.679684   validation-auc:0.666589
[78]    training-auc:0.679684   validation-auc:0.666589
[79]    training-auc:0.679753   validation-auc:0.666688
[80]    training-auc:0.680191   validation-auc:0.666915
[81]    training-auc:0.680306   validation-auc:0.666901
[82]    training-auc:0.680497   validation-auc:0.667159
[83]    training-auc:0.680497   validation-auc:0.667159
[84]    training-auc:0.680648   validation-auc:0.667155
[85]    training-auc:0.680862   validation-auc:0.667208
[86]    training-auc:0.680862   validation-auc:0.667208
[87]    training-auc:0.681315   validation-auc:0.667459
[88]    training-auc:0.681316   validation-auc:0.667459
[89]    training-auc:0.681809   validation-auc:0.667621
[90]    training-auc:0.681982   validation-auc:0.667927
[91]    training-auc:0.682085   validation-auc:0.668047
[92]    training-auc:0.682385   validation-auc:0.668321
[93]    training-auc:0.682385   validation-auc:0.668321
[94]    training-auc:0.682605   validation-auc:0.668288
[95]    training-auc:0.682605   validation-auc:0.668288
[96]    training-auc:0.682664   validation-auc:0.668295
[97]    training-auc:0.683014   validation-auc:0.668611
[98]    training-auc:0.683014   validation-auc:0.668611
[99]    training-auc:0.683308   validation-auc:0.668705
[100]   training-auc:0.683529   validation-auc:0.669074
[101]   training-auc:0.683668   validation-auc:0.669007
[102]   training-auc:0.683903   validation-auc:0.669291
[103]   training-auc:0.684203   validation-auc:0.669513
[104]   training-auc:0.684294   validation-auc:0.669410
[105]   training-auc:0.684294   validation-auc:0.669410
[106]   training-auc:0.684294   validation-auc:0.669410
[107]   training-auc:0.684294   validation-auc:0.669410
[108]   training-auc:0.684247   validation-auc:0.669402
Stopping. Best iteration:
[103]   training-auc:0.684203   validation-auc:0.669513

[1] training-auc:0.500000   validation-auc:0.500000
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.503100   validation-auc:0.502011
```

```
[3] training-auc:0.532068   validation-auc:0.516896
[4] training-auc:0.547334   validation-auc:0.520366
[5] training-auc:0.560379   validation-auc:0.526087
[6] training-auc:0.576267   validation-auc:0.529007
[7] training-auc:0.590902   validation-auc:0.536152
[8] training-auc:0.609945   validation-auc:0.546864
[9] training-auc:0.618699   validation-auc:0.552446
[10]    training-auc:0.632552   validation-auc:0.553571
[11]    training-auc:0.655997   validation-auc:0.568075
[12]    training-auc:0.663971   validation-auc:0.569939
[13]    training-auc:0.681185   validation-auc:0.584985
[14]    training-auc:0.694639   validation-auc:0.595802
[15]    training-auc:0.706450   validation-auc:0.604924
[16]    training-auc:0.716179   validation-auc:0.612662
[17]    training-auc:0.725371   validation-auc:0.616712
[18]    training-auc:0.730459   validation-auc:0.620873
[19]    training-auc:0.736406   validation-auc:0.625043
[20]    training-auc:0.740149   validation-auc:0.623717
[21]    training-auc:0.742613   validation-auc:0.626264
[22]    training-auc:0.746250   validation-auc:0.627191
[23]    training-auc:0.748279   validation-auc:0.630074
[24]    training-auc:0.748519   validation-auc:0.629617
[25]    training-auc:0.752711   validation-auc:0.633663
[26]    training-auc:0.755323   validation-auc:0.636267
[27]    training-auc:0.755298   validation-auc:0.636627
[28]    training-auc:0.758794   validation-auc:0.638982
[29]    training-auc:0.761441   validation-auc:0.641330
[30]    training-auc:0.761851   validation-auc:0.643343
[31]    training-auc:0.764211   validation-auc:0.643524
[32]    training-auc:0.766918   validation-auc:0.644518
[33]    training-auc:0.767048   validation-auc:0.645344
[34]    training-auc:0.768362   validation-auc:0.646397
[35]    training-auc:0.770991   validation-auc:0.647493
[36]    training-auc:0.770740   validation-auc:0.647263
[37]    training-auc:0.771850   validation-auc:0.645355
[38]    training-auc:0.772314   validation-auc:0.646153
[39]    training-auc:0.772749   validation-auc:0.645808
[40]    training-auc:0.775939   validation-auc:0.648417
[41]    training-auc:0.776478   validation-auc:0.648451
[42]    training-auc:0.778214   validation-auc:0.649713
[43]    training-auc:0.778406   validation-auc:0.649726
[44]    training-auc:0.779017   validation-auc:0.649891
[45]    training-auc:0.777539   validation-auc:0.649808
[46]    training-auc:0.778601   validation-auc:0.650148
[47]    training-auc:0.779790   validation-auc:0.649535
[48]    training-auc:0.781595   validation-auc:0.650610
[49]    training-auc:0.782084   validation-auc:0.649615
[50]    training-auc:0.782346   validation-auc:0.649302
[51]    training-auc:0.781317   validation-auc:0.649404
[52]    training-auc:0.781628   validation-auc:0.650248
[53]    training-auc:0.783109   validation-auc:0.649046
Stopping. Best iteration:
[48]    training-auc:0.781595   validation-auc:0.650610
```

```
[1] training-auc:0.671824   validation-auc:0.578347
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.721618   validation-auc:0.596732
[3] training-auc:0.738992   validation-auc:0.612206
[4] training-auc:0.754698   validation-auc:0.618252
[5] training-auc:0.765397   validation-auc:0.625754
[6] training-auc:0.776044   validation-auc:0.629548
[7] training-auc:0.784051   validation-auc:0.630608
[8] training-auc:0.793233   validation-auc:0.631904
[9] training-auc:0.799366   validation-auc:0.632434
[10]    training-auc:0.803522   validation-auc:0.635421
[11]    training-auc:0.811694   validation-auc:0.640965
[12]    training-auc:0.813147   validation-auc:0.642789
[13]    training-auc:0.819084   validation-auc:0.646209
[14]    training-auc:0.823247   validation-auc:0.648552
[15]    training-auc:0.826908   validation-auc:0.650113
[16]    training-auc:0.830234   validation-auc:0.652743
[17]    training-auc:0.832979   validation-auc:0.653623
[18]    training-auc:0.833169   validation-auc:0.655173
[19]    training-auc:0.839412   validation-auc:0.656432
[20]    training-auc:0.839454   validation-auc:0.657433
[21]    training-auc:0.843898   validation-auc:0.658163
[22]    training-auc:0.844791   validation-auc:0.658067
[23]    training-auc:0.845992   validation-auc:0.657969
[24]    training-auc:0.846515   validation-auc:0.658228
[25]    training-auc:0.849269   validation-auc:0.658195
[26]    training-auc:0.849974   validation-auc:0.658440
[27]    training-auc:0.849406   validation-auc:0.659193
[28]    training-auc:0.851525   validation-auc:0.659285
[29]    training-auc:0.852148   validation-auc:0.659212
[30]    training-auc:0.851778   validation-auc:0.659243
[31]    training-auc:0.853762   validation-auc:0.660593
[32]    training-auc:0.853819   validation-auc:0.660710
[33]    training-auc:0.853484   validation-auc:0.660573
[34]    training-auc:0.853578   validation-auc:0.660637
[35]    training-auc:0.855520   validation-auc:0.660169
[36]    training-auc:0.855312   validation-auc:0.660038
[37]    training-auc:0.856179   validation-auc:0.661258
[38]    training-auc:0.855076   validation-auc:0.661391
[39]    training-auc:0.855091   validation-auc:0.661585
[40]    training-auc:0.855020   validation-auc:0.661655
[41]    training-auc:0.856638   validation-auc:0.661856
[42]    training-auc:0.857477   validation-auc:0.661449
[43]    training-auc:0.857326   validation-auc:0.661366
[44]    training-auc:0.857875   validation-auc:0.661812
[45]    training-auc:0.857420   validation-auc:0.662242
[46]    training-auc:0.857682   validation-auc:0.662037
[47]    training-auc:0.858287   validation-auc:0.662430
[48]    training-auc:0.858250   validation-auc:0.662654
[49]    training-auc:0.857834   validation-auc:0.662644
[50]    training-auc:0.857512   validation-auc:0.662692
[51]    training-auc:0.856965   validation-auc:0.662229
```

```
[52]    training-auc:0.856732   validation-auc:0.662404
[53]    training-auc:0.857129   validation-auc:0.662731
[54]    training-auc:0.856986   validation-auc:0.663115
[55]    training-auc:0.858363   validation-auc:0.663905
[56]    training-auc:0.858477   validation-auc:0.664221
[57]    training-auc:0.858276   validation-auc:0.663917
[58]    training-auc:0.858463   validation-auc:0.664230
[59]    training-auc:0.857999   validation-auc:0.664402
[60]    training-auc:0.858186   validation-auc:0.664644
[61]    training-auc:0.859537   validation-auc:0.664912
[62]    training-auc:0.860054   validation-auc:0.665084
[63]    training-auc:0.860158   validation-auc:0.665304
[64]    training-auc:0.859811   validation-auc:0.664926
[65]    training-auc:0.859695   validation-auc:0.664707
[66]    training-auc:0.859564   validation-auc:0.664583
[67]    training-auc:0.860285   validation-auc:0.664894
[68]    training-auc:0.860196   validation-auc:0.664855
Stopping. Best iteration:
[63]    training-auc:0.860158   validation-auc:0.665304

[1] training-auc:0.547003   validation-auc:0.540599
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.617976   validation-auc:0.613925
[3] training-auc:0.623669   validation-auc:0.617871
[4] training-auc:0.626762   validation-auc:0.619665
[5] training-auc:0.639286   validation-auc:0.632621
[6] training-auc:0.641047   validation-auc:0.635096
[7] training-auc:0.640405   validation-auc:0.633811
[8] training-auc:0.640897   validation-auc:0.632769
[9] training-auc:0.639814   validation-auc:0.634788
[10]    training-auc:0.639821   validation-auc:0.634170
[11]    training-auc:0.640547   validation-auc:0.633455
Stopping. Best iteration:
[6] training-auc:0.641047   validation-auc:0.635096

[1] training-auc:0.585406   validation-auc:0.573614
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.597084   validation-auc:0.587823
[3] training-auc:0.602515   validation-auc:0.592376
[4] training-auc:0.605622   validation-auc:0.593560
[5] training-auc:0.602263   validation-auc:0.588977
[6] training-auc:0.600952   validation-auc:0.588109
[7] training-auc:0.601757   validation-auc:0.588495
[8] training-auc:0.623748   validation-auc:0.609997
[9] training-auc:0.625773   validation-auc:0.611136
[10]    training-auc:0.625019   validation-auc:0.607579
[11]    training-auc:0.624568   validation-auc:0.607400
[12]    training-auc:0.622750   validation-auc:0.606632
[13]    training-auc:0.629835   validation-auc:0.614774
[14]    training-auc:0.630680   validation-auc:0.616947
```

```
[15]    training-auc:0.628477   validation-auc:0.615225
[16]    training-auc:0.628835   validation-auc:0.615976
[17]    training-auc:0.626721   validation-auc:0.613869
[18]    training-auc:0.633982   validation-auc:0.621505
[19]    training-auc:0.633225   validation-auc:0.620730
[20]    training-auc:0.633390   validation-auc:0.620630
[21]    training-auc:0.632704   validation-auc:0.619253
[22]    training-auc:0.636093   validation-auc:0.621830
[23]    training-auc:0.636262   validation-auc:0.621777
[24]    training-auc:0.635337   validation-auc:0.619971
[25]    training-auc:0.639376   validation-auc:0.624753
[26]    training-auc:0.637931   validation-auc:0.623343
[27]    training-auc:0.640452   validation-auc:0.626098
[28]    training-auc:0.639933   validation-auc:0.625260
[29]    training-auc:0.640607   validation-auc:0.626167
[30]    training-auc:0.641784   validation-auc:0.627031
[31]    training-auc:0.642006   validation-auc:0.627501
[32]    training-auc:0.643331   validation-auc:0.629144
[33]    training-auc:0.642559   validation-auc:0.628225
[34]    training-auc:0.643204   validation-auc:0.629540
[35]    training-auc:0.642423   validation-auc:0.628244
[36]    training-auc:0.642294   validation-auc:0.628030
[37]    training-auc:0.642639   validation-auc:0.628405
[38]    training-auc:0.642987   validation-auc:0.629336
[39]    training-auc:0.642655   validation-auc:0.628579
Stopping. Best iteration:
[34]    training-auc:0.643204   validation-auc:0.629540

[1] training-auc:0.500000   validation-auc:0.500000
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.500000   validation-auc:0.500000
[3] training-auc:0.500000   validation-auc:0.500000
[4] training-auc:0.500000   validation-auc:0.500000
[5] training-auc:0.500000   validation-auc:0.500000
[6] training-auc:0.500000   validation-auc:0.500000
Stopping. Best iteration:
[1] training-auc:0.500000   validation-auc:0.500000

[1] training-auc:0.500000   validation-auc:0.500000
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.500000   validation-auc:0.500000
[3] training-auc:0.505887   validation-auc:0.504918
[4] training-auc:0.532905   validation-auc:0.516533
[5] training-auc:0.561320   validation-auc:0.525615
[6] training-auc:0.584028   validation-auc:0.537137
[7] training-auc:0.613069   validation-auc:0.545386
[8] training-auc:0.644133   validation-auc:0.556239
[9] training-auc:0.671772   validation-auc:0.571911
[10]    training-auc:0.697755   validation-auc:0.584741
[11]    training-auc:0.716735   validation-auc:0.594233
```

```
[12]    training-auc:0.731382   validation-auc:0.603349
[13]    training-auc:0.742693   validation-auc:0.610279
[14]    training-auc:0.754901   validation-auc:0.616778
[15]    training-auc:0.765559   validation-auc:0.621645
[16]    training-auc:0.772621   validation-auc:0.625780
[17]    training-auc:0.779824   validation-auc:0.628205
[18]    training-auc:0.786830   validation-auc:0.633695
[19]    training-auc:0.794678   validation-auc:0.635470
[20]    training-auc:0.797672   validation-auc:0.636909
[21]    training-auc:0.801385   validation-auc:0.639978
[22]    training-auc:0.805242   validation-auc:0.642608
[23]    training-auc:0.809638   validation-auc:0.643680
[24]    training-auc:0.812309   validation-auc:0.643840
[25]    training-auc:0.815227   validation-auc:0.646268
[26]    training-auc:0.818476   validation-auc:0.645846
[27]    training-auc:0.820612   validation-auc:0.647049
[28]    training-auc:0.821932   validation-auc:0.649685
[29]    training-auc:0.824255   validation-auc:0.649469
[30]    training-auc:0.826379   validation-auc:0.649871
[31]    training-auc:0.827747   validation-auc:0.652003
[32]    training-auc:0.829904   validation-auc:0.650957
[33]    training-auc:0.830664   validation-auc:0.653805
[34]    training-auc:0.832300   validation-auc:0.652645
[35]    training-auc:0.834222   validation-auc:0.656071
[36]    training-auc:0.835559   validation-auc:0.656241
[37]    training-auc:0.837454   validation-auc:0.657901
[38]    training-auc:0.839575   validation-auc:0.658169
[39]    training-auc:0.840138   validation-auc:0.657762
[40]    training-auc:0.840486   validation-auc:0.656847
[41]    training-auc:0.841740   validation-auc:0.658313
[42]    training-auc:0.843041   validation-auc:0.658804
[43]    training-auc:0.844536   validation-auc:0.658791
[44]    training-auc:0.844752   validation-auc:0.659862
[45]    training-auc:0.845801   validation-auc:0.660843
[46]    training-auc:0.846381   validation-auc:0.659555
[47]    training-auc:0.847726   validation-auc:0.659646
[48]    training-auc:0.848617   validation-auc:0.660147
[49]    training-auc:0.849370   validation-auc:0.661999
[50]    training-auc:0.848635   validation-auc:0.661885
[51]    training-auc:0.848699   validation-auc:0.662804
[52]    training-auc:0.849414   validation-auc:0.664328
[53]    training-auc:0.849850   validation-auc:0.666040
[54]    training-auc:0.850318   validation-auc:0.666407
[55]    training-auc:0.850340   validation-auc:0.667227
[56]    training-auc:0.851390   validation-auc:0.666734
[57]    training-auc:0.851721   validation-auc:0.668139
[58]    training-auc:0.851843   validation-auc:0.666802
[59]    training-auc:0.852775   validation-auc:0.667668
[60]    training-auc:0.853172   validation-auc:0.667989
[61]    training-auc:0.853466   validation-auc:0.667483
[62]    training-auc:0.853285   validation-auc:0.667054
Stopping. Best iteration:
[57]    training-auc:0.851721   validation-auc:0.668139
```

```
[1] training-auc:0.650557   validation-auc:0.574586
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.715705   validation-auc:0.604836
[3] training-auc:0.743575   validation-auc:0.617012
[4] training-auc:0.767385   validation-auc:0.627971
[5] training-auc:0.778021   validation-auc:0.639153
[6] training-auc:0.788293   validation-auc:0.639248
[7] training-auc:0.797726   validation-auc:0.645065
[8] training-auc:0.803255   validation-auc:0.646696
[9] training-auc:0.806047   validation-auc:0.650041
[10]    training-auc:0.811731   validation-auc:0.654138
[11]    training-auc:0.815468   validation-auc:0.654327
[12]    training-auc:0.817084   validation-auc:0.657988
[13]    training-auc:0.816929   validation-auc:0.657789
[14]    training-auc:0.816994   validation-auc:0.659188
[15]    training-auc:0.817936   validation-auc:0.660417
[16]    training-auc:0.819505   validation-auc:0.661701
[17]    training-auc:0.823850   validation-auc:0.660650
[18]    training-auc:0.826763   validation-auc:0.661821
[19]    training-auc:0.828928   validation-auc:0.661897
[20]    training-auc:0.829287   validation-auc:0.663855
[21]    training-auc:0.831157   validation-auc:0.664204
[22]    training-auc:0.832326   validation-auc:0.664082
[23]    training-auc:0.833083   validation-auc:0.664511
[24]    training-auc:0.833610   validation-auc:0.665054
[25]    training-auc:0.834233   validation-auc:0.665765
[26]    training-auc:0.834135   validation-auc:0.665856
[27]    training-auc:0.834332   validation-auc:0.666064
[28]    training-auc:0.835369   validation-auc:0.666393
[29]    training-auc:0.836663   validation-auc:0.666567
[30]    training-auc:0.837001   validation-auc:0.666625
[31]    training-auc:0.838512   validation-auc:0.667472
[32]    training-auc:0.838879   validation-auc:0.667684
[33]    training-auc:0.839682   validation-auc:0.668037
[34]    training-auc:0.839276   validation-auc:0.668107
[35]    training-auc:0.840069   validation-auc:0.668416
[36]    training-auc:0.840515   validation-auc:0.668781
[37]    training-auc:0.840807   validation-auc:0.668989
[38]    training-auc:0.840999   validation-auc:0.669797
[39]    training-auc:0.840963   validation-auc:0.669565
[40]    training-auc:0.841488   validation-auc:0.670211
[41]    training-auc:0.841918   validation-auc:0.670455
[42]    training-auc:0.841428   validation-auc:0.670724
[43]    training-auc:0.841461   validation-auc:0.670831
[44]    training-auc:0.841228   validation-auc:0.670727
[45]    training-auc:0.841436   validation-auc:0.670781
[46]    training-auc:0.841154   validation-auc:0.671008
[47]    training-auc:0.841282   validation-auc:0.670595
[48]    training-auc:0.841348   validation-auc:0.671076
[49]    training-auc:0.841635   validation-auc:0.670968
[50]    training-auc:0.842650   validation-auc:0.671142
[51]    training-auc:0.842758   validation-auc:0.671181
```

```
[52]    training-auc:0.842624   validation-auc:0.670992
[53]    training-auc:0.843204   validation-auc:0.671195
[54]    training-auc:0.843658   validation-auc:0.671335
[55]    training-auc:0.843617   validation-auc:0.671415
[56]    training-auc:0.843446   validation-auc:0.671658
[57]    training-auc:0.843458   validation-auc:0.672030
[58]    training-auc:0.843359   validation-auc:0.671796
[59]    training-auc:0.843125   validation-auc:0.672021
[60]    training-auc:0.843650   validation-auc:0.672454
[61]    training-auc:0.843900   validation-auc:0.672362
[62]    training-auc:0.844091   validation-auc:0.672525
[63]    training-auc:0.843884   validation-auc:0.672318
[64]    training-auc:0.843995   validation-auc:0.672385
[65]    training-auc:0.843899   validation-auc:0.672775
[66]    training-auc:0.843980   validation-auc:0.672770
[67]    training-auc:0.844146   validation-auc:0.672561
[68]    training-auc:0.844169   validation-auc:0.672739
[69]    training-auc:0.844322   validation-auc:0.672704
[70]    training-auc:0.844621   validation-auc:0.673138
[71]    training-auc:0.844682   validation-auc:0.673236
[72]    training-auc:0.844807   validation-auc:0.673297
[73]    training-auc:0.844944   validation-auc:0.673582
[74]    training-auc:0.845171   validation-auc:0.673613
[75]    training-auc:0.844974   validation-auc:0.673522
[76]    training-auc:0.844989   validation-auc:0.673888
[77]    training-auc:0.845176   validation-auc:0.673840
[78]    training-auc:0.845776   validation-auc:0.673942
[79]    training-auc:0.845659   validation-auc:0.673831
[80]    training-auc:0.845358   validation-auc:0.673903
[81]    training-auc:0.845467   validation-auc:0.673891
[82]    training-auc:0.845308   validation-auc:0.673869
[83]    training-auc:0.845213   validation-auc:0.673968
[84]    training-auc:0.845338   validation-auc:0.674235
[85]    training-auc:0.845499   validation-auc:0.674265
[86]    training-auc:0.845421   validation-auc:0.674021
[87]    training-auc:0.845186   validation-auc:0.674195
[88]    training-auc:0.845142   validation-auc:0.673978
[89]    training-auc:0.845190   validation-auc:0.673859
[90]    training-auc:0.845124   validation-auc:0.674085
Stopping. Best iteration:
[85]    training-auc:0.845499   validation-auc:0.674265

[1] training-auc:0.573126   validation-auc:0.576189
Multiple eval metrics are present. Will use validation_auc for early stopping.
Will train until validation_auc hasn't improved in 5 rounds.

[2] training-auc:0.599735   validation-auc:0.600864
[3] training-auc:0.608227   validation-auc:0.609918
[4] training-auc:0.606751   validation-auc:0.607534
[5] training-auc:0.608457   validation-auc:0.610620
[6] training-auc:0.608774   validation-auc:0.610362
[7] training-auc:0.607766   validation-auc:0.608712
[8] training-auc:0.613957   validation-auc:0.615430
[9] training-auc:0.617968   validation-auc:0.618522
```

```
[10]    training-auc:0.619291   validation-auc:0.620146
[11]    training-auc:0.615331   validation-auc:0.616558
[12]    training-auc:0.616702   validation-auc:0.617512
[13]    training-auc:0.610980   validation-auc:0.610975
[14]    training-auc:0.617599   validation-auc:0.617873
[15]    training-auc:0.616563   validation-auc:0.616662
Stopping. Best iteration:
[10]    training-auc:0.619291   validation-auc:0.620146
```

```
Sys.time() -a
```

```
Time difference of 56.67389 secs
```

```
save(xgb, xgb_grid, training, watchlist, file = "Modelos/xgb.Rdata")
```

```
# Winning combination
params_xgb<-map_dfr(xgb, ~.$params)
params_xgb<-bind_cols(params_xgb, auc = map_dbl(xgb, function(x) x$best_score))
params_xgb$model_number<-seq(1:15)

# tabla
kable(params_xgb)
```

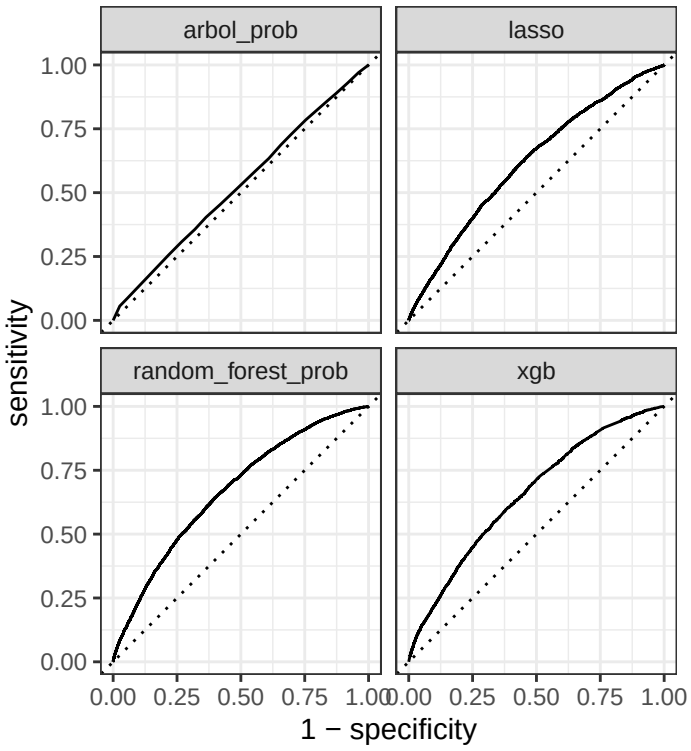
max_depth	gamma	subsample	eta	colsample_bytree	eval_metric	objective	validate_parameters	auc	model_number
5	0.0000000	0.8571860	0.0000778	0.8939394	auc	binary:logistic	TRUE	0.653751	1
6	0.0529435	0.7554947	0.0000020	0.6060606	auc	binary:logistic	TRUE	0.666012	2
7	0.0001262	0.6849654	0.0132164	0.1666667	auc	binary:logistic	TRUE	0.633533	3
9	0.0000022	0.6134857	0.0000000	0.6969697	auc	binary:logistic	TRUE	0.500000	4
15	0.2004157	0.8069170	0.0000000	0.0757576	auc	binary:logistic	TRUE	0.500000	5
8	0.0002120	0.9351226	0.0002485	0.3484848	auc	binary:logistic	TRUE	0.657553	6
5	13.8234861	0.3274263	0.0687369	0.4393939	auc	binary:logistic	TRUE	0.669513	7
13	4.9568172	0.3843921	0.0000000	0.2575758	auc	binary:logistic	TRUE	0.650610	8
12	0.0000155	0.4180194	0.0000195	0.4848485	auc	binary:logistic	TRUE	0.665304	9
3	0.0000000	0.5250437	0.0037858	0.3030303	auc	binary:logistic	TRUE	0.635096	10
3	0.0180537	0.1168934	0.0008345	0.1363636	auc	binary:logistic	TRUE	0.629540	11
10	0.0000000	0.1883817	0.0000000	0.6515152	auc	binary:logistic	TRUE	0.500000	12
14	0.0000004	0.2629531	0.0000000	0.9545455	auc	binary:logistic	TRUE	0.668139	13
11	0.0000000	0.4650565	0.0000002	0.7424242	auc	binary:logistic	TRUE	0.674265	14
2	0.0010333	0.9902304	0.0000005	0.8333333	auc	binary:logistic	TRUE	0.620146	15

```
# Quedandome con el ganador
a<-pull(params_xgb %>% arrange(desc(auc)) %>% dplyr::slice(1) %>% select(model_number))

xgb<-xgb[[1]]

# Predicting
# Validation
X<-
   cell2cell_V %>%
   select(-customer, -churn)

X<-sparse.model.matrix(~.+0, data = X)
```

```
predicciones<-
  predicciones %>%
  mutate(xgb = predict(xgb, newdata = X))

save(predicciones, file = 'Bases output/predicciones.Rdata')
```

17 (4 pts). Genera graficas de las curvas ROC para los 4 modelos. Cual parece ser mejor?

```
# Factor truth
predicciones<-
   predicciones %>%
   mutate(churn = factor(churn, levels = c('1', '0')))

# ROC para cada una
predicciones1<-
   predicciones %>%
   pivot_longer(cols = c("lasso", "arbol_prob", "random_forest_prob", "xgb"),
                names_to = 'modelo',
                values_to = 'pred')

bases<-predicciones1 %>% split(.$modelo)

rocs<-map2_dfr(.x =  bases,
               .y = names(bases),
               function(x,y) roc_curve(data = x, churn, pred) %>%
                  mutate(modelo = y))


ggplot(rocs, aes(1-specificity, y = sensitivity))+
   geom_path()+
   geom_abline(lty =3)+coord_equal()+theme_bw()+
   facet_wrap(~modelo)
```



18. Genera una tabla con el AUC ROC. Cuál es el mejor modelo ?

El random forest marginalmente sobre el XGB

```
auc_rocs<-map2_dfr(.x =  bases,
                .y = names(bases),
                function(x,y) roc_auc(data = x, churn, pred) %>%
                   mutate(modelo = y))

kable(auc_rocs)
```

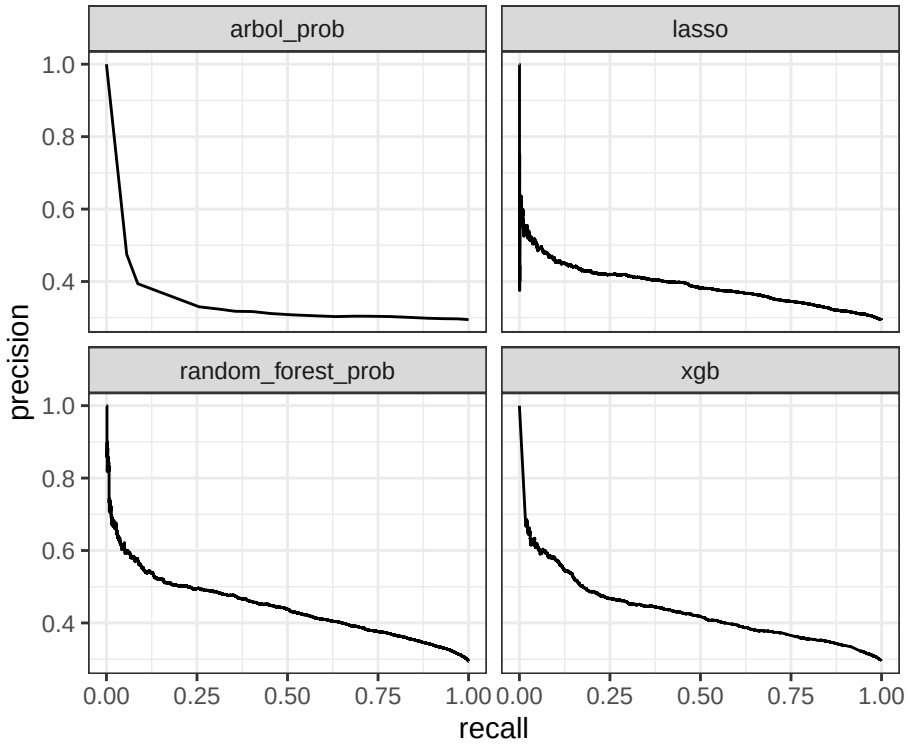
.metric	.estimator	.estimate	modelo
roc_auc	binary	0.5286201	arbol_prob
roc_auc	binary	0.6174667	lasso
roc_auc	binary	0.6689970	random_forest_prob
roc_auc	binary	0.6537525	xgb

19 (2 pts). Genera las gráficas de las curvas precission recall para los 4 modelos. Cuál parece ser mejor ahora?

Ahora el XGB parece ser mejor que el random forest

```
pr_curves<-map2_dfr(.x =  bases,
                .y = names(bases),
                function(x,y) pr_curve(data = x, churn, pred) %>%
                   mutate(modelo = y))

save(rocs, pr_curves, file = 'Tablas/rocs_pr.Rdata')

ggplot(pr_curves, aes(recall, y = precision))+
```

```
   geom_path()+
   coord_equal()+theme_bw()+
   facet_wrap(~modelo)
```




```
# PR AUC
auc_pr<-map2_dfr(.x =  bases,
                 .y = names(bases),
                 function(x,y) pr_auc(data = x, churn, pred) %>%
                    mutate(modelo = y))

kable(auc_pr)
```

.metric	.estimator	.estimate	modelo
pr_auc	binary	0.3445663	arbol_prob
pr_auc	binary	0.3890416	lasso
pr_auc	binary	0.4445194	random_forest_prob
pr_auc	binary	0.4338018	xgb

20 (2pts). Escoge un punto de corte para generar predicciones categoricas para el LASSO basado en la Curva ROC. Genera las matrices de confusión para cada modelo. Compáralas. Qué tipo de error es mas pernicioso?

```
# Cortes usando ROC
cortes<-
   rocs %>%
   group_by(modelo) %>%
   filter(sensitivity >= 0.80) %>%
   arrange(desc(specificity)) %>%
   dplyr::slice(1)
```

```r
cortes_pr<-
  pr_curves %>%
  group_by(modelo) %>%
  filter(recall>=0.8) %>%
  arrange(desc(precision)) %>%
  dplyr::slice(1) %>%
  select(modelo, everything())


predicciones<-
   predicciones %>%
   mutate(lasso_cat = if_else(lasso>=cortes_pr$.threshold[cortes_pr$modelo=='lasso'], 1, 0),
          rf_cat = if_else(random_forest_prob>=cortes_pr$.threshold[cortes_pr$modelo=='random_forest_pr
          xgb_cat = if_else(xgb>=cortes_pr$.threshold[cortes_pr$modelo=='xgb'], 1, 0))


# Matrices
kable(prop.table(table(predicciones$lasso_cat, as.numeric(as.character(predicciones$churn))), margin = 1
```

Table 6: Matriz de Confusión LASSO

	0	1
0	0.34	0.2
1	0.66	0.8

```r
kable(prop.table(table(predicciones$arbol_class, as.numeric(as.character(predicciones$churn))), margin =
```

Table 7: Matriz de Confusión Prune Tree

	0	1
0	0.64	0.6
1	0.36	0.4

```r
kable(prop.table(table(predicciones$rf_cat, as.numeric(as.character(predicciones$churn))), margin = 2),
```

Table 8: Matriz de Confusión RF

	0	1
0	0.42	0.2
1	0.58	0.8

```r
kable(prop.table(table(predicciones$xgb_cat, as.numeric(as.character(predicciones$churn))), margin = 2)
```

Table 9: Matriz de Confusión XGB

	0	1
0	0.39	0.2
1	0.61	0.8

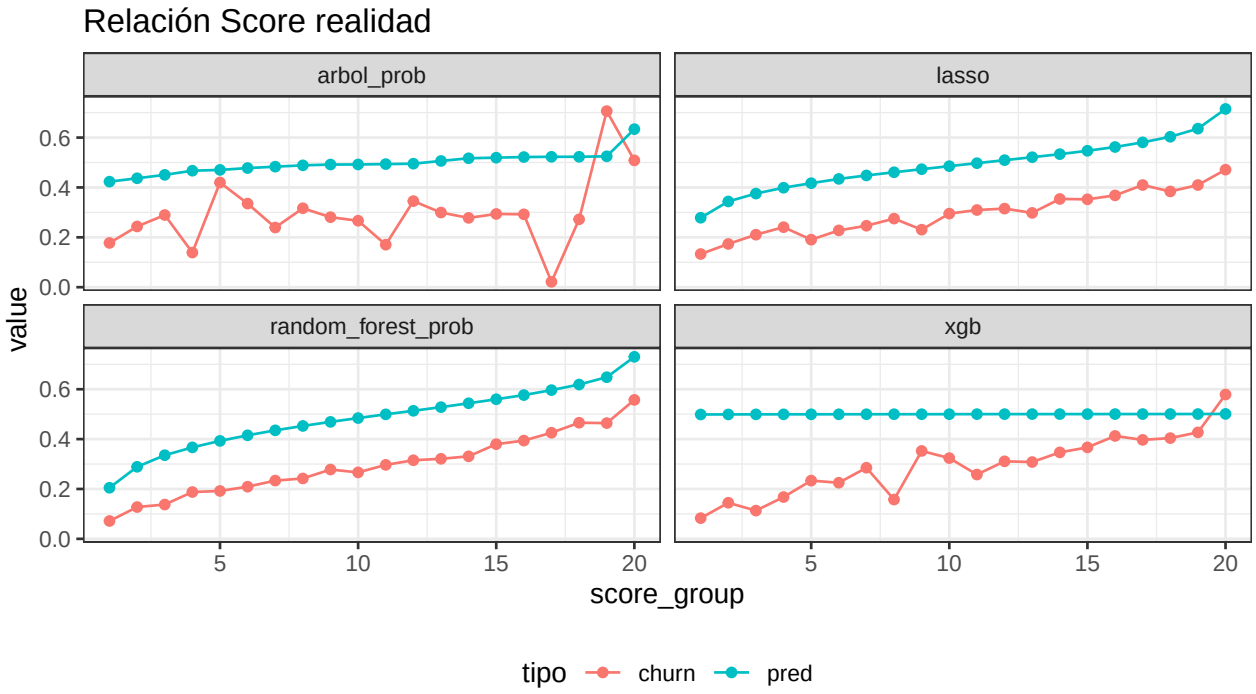
21 (4pts). Construye una lift table. Esto es, para 20 grupos del score predecido de manera decreciente, genera 1) El promedio de las predicciones, 2) el promedio del churn rate observado. Estima el lift:

$$lift_k = \frac{churn\ \ rate_k}{churn\ \ rate\ \ promedio}$$

Donde churn rate promedio es el promedio global. Muesta la tabla. Que te dice el lift intuitivamente

Intuitivamente, el lift me dice si el nivel de churn observado es más alto en scores más altos o no. Esperarías ver una relación monotónica entre el lift y el score promedio

```
# ROC para cada una
predicciones1<-
   predicciones %>%
   pivot_longer(cols = c("lasso", "arbol_prob", "random_forest_prob", "xgb"),
                names_to = 'modelo',
                values_to = 'pred')



predicciones1<-
   predicciones1 %>%
   group_by(modelo) %>%
   mutate(score_group = ntile(x = pred, 20)) %>%
   group_by(modelo, score_group) %>%
   summarise(pred  = mean(pred),
             churn = mean(as.numeric(as.character(churn))))


base_grafica<-
   predicciones1 %>%
   pivot_longer(cols = c(pred, churn), names_to = 'tipo', values_to = 'value')

ggplot(base_grafica, aes(score_group, value, fill = tipo, color = tipo))+
   geom_point()+geom_line()+
   facet_wrap(~modelo)+
   theme_bw()+
   labs(title = 'Relación Score realidad')+
   theme(legend.position = 'bottom')
```



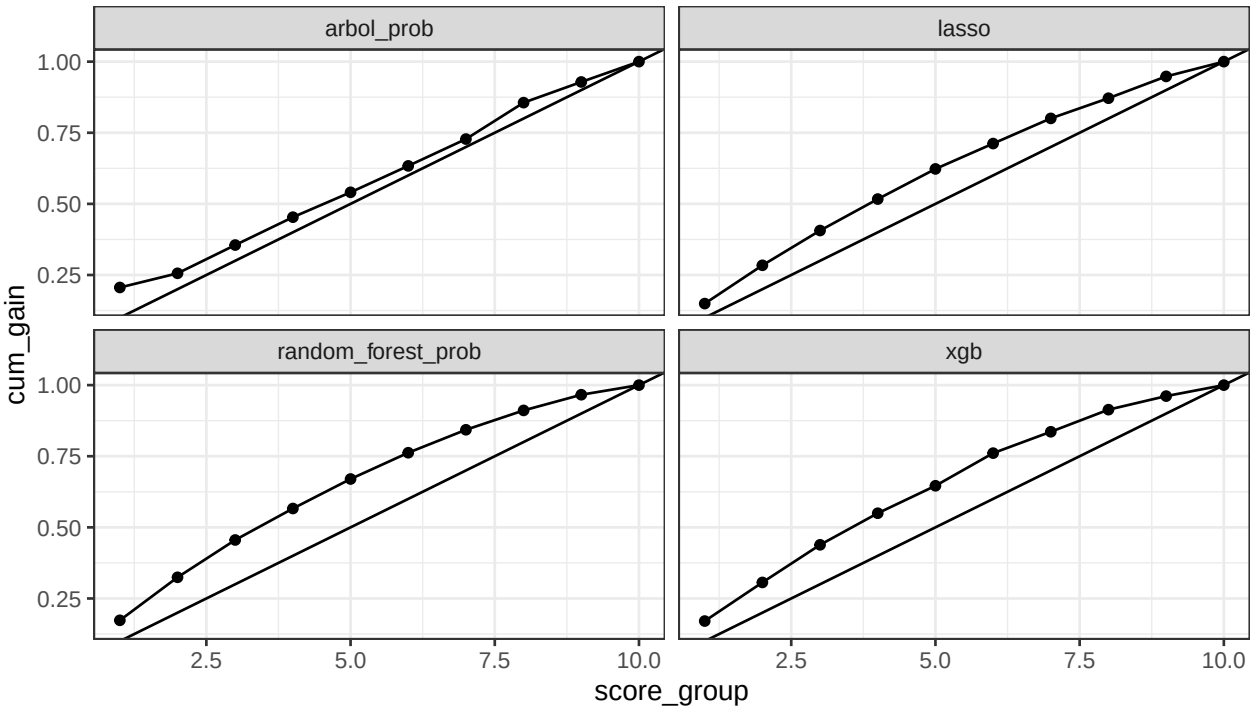
22 (4pts). Construye un Gain table. Esto es: Cuántas personas detectarías con el modelo del total de churned users al buscar al x% de la población? Como se compara esto para una selección aleatoria?

Ejemplo: Si tengo 10% de churn y tengo 10 grupos:

- Si focalizo al 10% aleatoriamente, voy a obtener el 10% de la población churned.
- Si focalizo al 20% aleatoriamente, voy a obtener el 20% de la población churned.
- Si focalizo al 10% usando al 10% de la población con score predecido mas alto, que porcentaje obtengo?
- Como se compara tu modelo vs el modelo aleatorio

```
# Gain
predicciones1<-
   predicciones %>%
   pivot_longer(cols = c("lasso", "arbol_prob", "random_forest_prob", "xgb"),
                names_to = 'modelo',
                values_to = 'pred')


gain<-
   predicciones1 %>%
   group_by(modelo) %>%
   mutate(score_group = ntile(x = pred, 10),
          total_churn = sum(as.numeric(as.character(churn)))) %>%
  ungroup() %>%
    group_by(modelo, score_group) %>%
   summarise(gain  = sum(as.numeric(as.character(churn)))) %>%
  group_by(modelo) %>%
  mutate(total_churn = sum(gain),
         score_group = abs(score_group-10)+1,
         gain = gain/total_churn) %>%
```

```
  group_by(modelo) %>%
  arrange(score_group) %>%
  mutate(cum_gain = cumsum(gain))



# Graficas
ggplot(gain, aes(score_group, cum_gain))+geom_point()+geom_line()+
  geom_abline(intercept = 0 ,slope = 1/10)+facet_wrap(~modelo)+
  theme_bw()
```



23 (2pts). Calcula el AUC Gain del mejor modelo. Interpreta

```
library(pracma)

bases <- gain %>% split(.$modelo)

map(bases, ~trapz(.$score_group, .$cum_gain)/10)
```

```
$arbol_prob
[1] 0.5353949

$lasso
[1] 0.5736817

$random_forest_prob
[1] 0.6085541

$xgb
[1] 0.5998177
```

24. Concluye. Que estrategia harías con este modelo? Cómo generarías valor a partir de el?

Haría una estrategia focalizada de prevencion de churn con el modelo usando el Random Forest. Se que estaré atacando al 80% de los casos (recall) y que la gente que ataque tendré una precisión de ~34%. Al ser una estrategia que no genera dolor, no hay problema con tener baja precisión. Así mismo, puedo ver que con sólo tratar al 30% de la base, puedo capturar al ~50% de la base de churn users!!!