***Building\_Telecommunication\_Churn\_Model\_Using\_R***

.libPaths("libs")

library(data.table)

library(h2o)

library(bit64)

library(pROC)

library(logging)

source("find\_best\_model.R")

logging::basicConfig()

loginfo("--> Loading data...")

all\_data <- fread("data/edw\_cdr.csv")

all\_data <- all\_data[, !c("month", "year"), with = FALSE]

all\_data <- all\_data[complete.cases(all\_data)]

all\_data <- all\_data[!duplicated(all\_data)]

loginfo("--> loaded %s rows", nrow(all\_data))

set.seed(1234)

loginfo("--> Cleaning&transforming data...")

all\_data[, ind := factor(sample(0:1, size = .N, replace = TRUE, prob = c(0.3, 0.7)),

levels = 0:1,

labels = c("Test", "Train"))]

all\_data[, churn := factor(ifelse(churn == 1, "churn", "nochurn"))]

loginfo("--> done")

h2o\_local <- h2o.init(nthreads = 4,

max\_mem\_size = "6g")

h2o.removeAll()

h2o\_train <- as.h2o(x = all\_data[ind == "Train"],

destination\_frame = "churn\_train")

h2o\_test <- as.h2o(x = all\_data[ind == "Test"],

destination\_frame = "churn\_test")

loginfo("--> Datasets imported into H2O cluster")

predictors <- setdiff(colnames(all\_data),

c("churn",

"customerid"))

churn\_var <- "churn"

gbm\_model <- h2o.grid(algorithm = "gbm",

grid\_id = "gbm\_grid",

training\_frame = h2o\_train,

x = predictors,

y = churn\_var,

nfolds = 5,

balance\_classes = TRUE,

distribution = "bernoulli",

hyper\_params = list(

ntrees = c(50,

100,

500),

max\_depth = c(4,

8,

16,

32)))

loginfo("--> Grid search done")

best\_model <- find\_best\_model(gbm\_model)

best\_model <- find\_best\_model(algorithm = "gbm",

grid\_id = "gbm\_grid",

training\_frame = h2o\_train,

x = predictors,

y = churn\_var,

nfolds = 5,

balance\_classes = TRUE,

distribution = "bernoulli",

hyper\_params = list(

ntrees = c(50,

100#,

#500

),

max\_depth = c(4#,

#8,

#16,

#32

)))

h2o.saveModel(best\_model$model, path = "export", force = TRUE)

loginfo("--> Best model exported into export folder")

loginfo("--> Best model with AUC=%s", h2o.auc(best\_model$model, xval = TRUE))

loginfo("--> Threshold for min per class accuracy metric = %s", best\_model$threshold)

test\_preds <- cbind(all\_data[ind == "Test"],

as.data.table(h2o.predict(object = best\_model$model,

newdata = h2o\_test))[, .(churn\_pred = ifelse(churn > 1 - best\_model$threshold,

"churn",

"nochurn"),

churn\_prob = churn)])

loginfo("--> Scoring test datasets done")

loginfo("Test accuracy: %.3f", test\_preds[, mean(churn == churn\_pred)])

loginfo("Test precision: %.3f", test\_preds[, sum(churn == "churn" & churn\_pred == "churn")/(sum(churn == "churn" & churn\_pred == "churn") +

sum(churn == "nochurn" & churn\_pred == "churn"))])

loginfo("Test recall: %.3f", test\_preds[, sum(churn == "churn" & churn\_pred == "churn")/(sum(churn == "churn" & churn\_pred == "churn") +

sum(churn == "churn" & churn\_pred == "nochurn"))])

tree\_roc <- pROC::roc(test\_preds[, churn], test\_preds[, churn\_prob])

loginfo("Test AUC: %.3f", pROC::auc(tree\_roc))

***Find the best model.R***

find\_best\_model <- function(h2o\_grid) {

find\_best\_model <- function(...) {

h2o\_grid <- h2o.grid(...)

best\_model <- NULL

for (model\_id in h2o\_grid@model\_ids) {

model <- h2o.getModel(model\_id)

model\_auc <- h2o.auc(model, xval = TRUE)

print(sprintf("Model %s got %s AUC", model@model\_id, model\_auc))

if (is.null(best\_model)) {

best\_model <- list(

model = model,

threshold = h2o.find\_threshold\_by\_max\_metric(h2o.performance(model, xval = TRUE),

"min\_per\_class\_accuracy"),

thresholds = h2o.performance(model, xval = TRUE))

} else if (h2o.auc(best\_model$model, xval = TRUE) < model\_auc) {

best\_model <- list(

model = model,

threshold = h2o.find\_threshold\_by\_max\_metric(h2o.performance(model, xval = TRUE),

"min\_per\_class\_accuracy"),

thresholds = h2o.performance(model, xval = TRUE))

}

}

return(best\_model)

}