bimbo_train.R

gquai

2020-01-16

```
# DATA SCIENCE ACADEMY
# Big Data Analytics com R e Microsoft Azure Machine Learning
#
# Model to accurately predict inventory demand based on data sales histories
#
# Gabriel Quaiotti
# Dez 2019
#
# In this competition, you will forecast the demand of a product for a given week, at a
# particular store.
# The dataset you are given consists of 9 weeks of sales transactions in Mexico. Every week,
# there are delivery
# trucks that deliver products to the vendors. Each transaction consists of sales and returns.
# Returns are the products that are unsold and expired. The demand for a product in a certain
# week is defined as the sales this week subtracted by the return next week.
#
#
# The train and test dataset are split based on time, as well as the public and private
# leaderboard dataset split.
#
#
# Things to note:
#
# There may be products in the test set that don't exist in the train set. This is the expected
# behavior of inventory data,
# since there are new products being sold all the time. Your model should be able to accommodate
# this.
#
# The adjusted demand (Demanda_uni_equil) is always >= 0 since demand should be either 0 or a
# positive value. The reason that Venta_uni_hoy - Dev_uni_proxima
# sometimes has negative values is that the returns records sometimes carry over a few weeks.

# File descriptions
# train.csv - the training set
# test.csv - the test set
# sample_submission.csv - a sample submission file in the correct format
# cliente_tabla.csv - client names (can be joined with train/test on Cliente_ID)
# producto_tabla.csv - product names (can be joined with train/test on Producto_ID)
# town_state.csv - town and state (can be joined with train/test on Agencia_ID)

# Data fields
# Semana - Week number (From Thursday to Wednesday)
# Agencia_ID - Sales Depot ID
# Canal_ID - Sales Channel ID
# Ruta_SAK - Route ID (Several routes = Sales Depot)
# Cliente_ID - Client ID
# NombreCliente - Client name
```

```r
# Producto_ID - Product ID
# NombreProducto - Product Name
# Venta_uni_hoy - Sales unit this week (integer)
# Venta_hoy - Sales this week (unit: pesos)
# Dev_uni_proxima - Returns unit next week (integer)
# Dev_proxima - Returns next week (unit: pesos)
# Demanda_uni_equil - Adjusted Demand (integer) (This is the target you will predict)

setwd('D:/Github/DSA_BIMBO_INVENTORY')

library(data.table)
library(caTools)
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```r
library(corrplot)
```

```
## corrplot 0.84 loaded
```

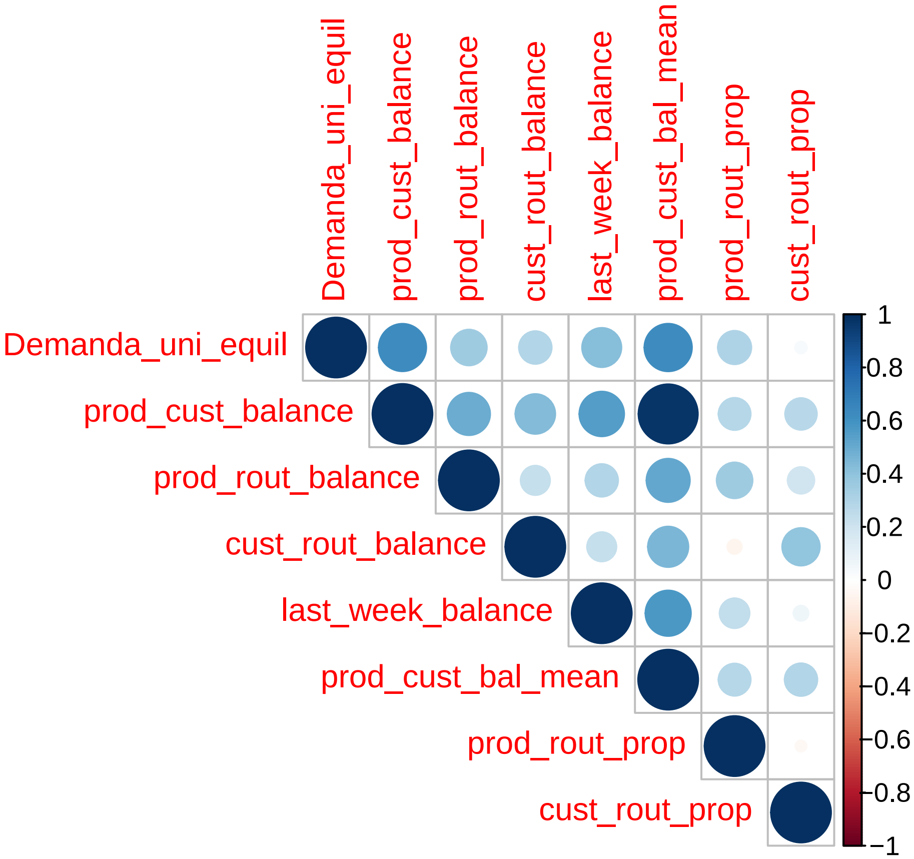
```r
library(lares)
```

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

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.fracdiff    fracdiff
##   residuals.fracdiff fracdiff
```

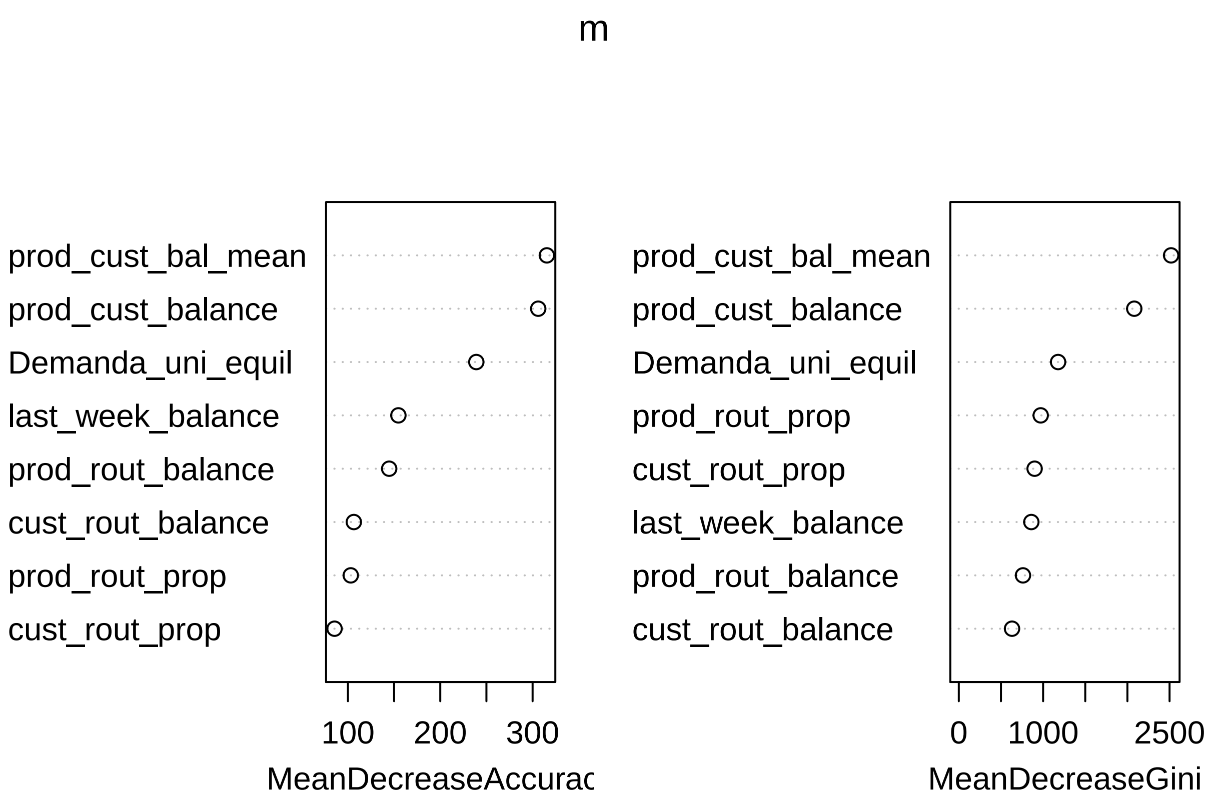
```r
library(MASS)
library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
v_c_file_train <- "dataset/train_split.csv"
v_c_file_test <- "dataset/test_split.csv"

####################################
# TRAIN DATASET
####################################
train <- data.table::fread(file = v_c_file_train)

# Correlation
s <- sample(nrow(train), 10000)

c <- cor(train[s , list(Demanda_uni_equil,
                        prod_cust_balance,
                        prod_rout_balance,
                        cust_rout_balance,
                        last_week_balance,
                        prod_cust_bal_mean,
                        prod_rout_prop,
                        cust_rout_prop)])

corrplot::corrplot(c, type = "upper")
```



```
remove(c)

m <- randomForest(x = train[s , list(Demanda_uni_equil,
                                     prod_cust_balance,
                                     prod_rout_balance,
                                     cust_rout_balance,
                                     last_week_balance,
                                     prod_cust_bal_mean,
                                     prod_rout_prop,
                                     cust_rout_prop)],
                  formula = Demanda_uni_equil ~ .,
                  importance = TRUE,
                  keep.forest = FALSE)

varImpPlot(m)
```



```
remove(m)

# The training step was made with 1.000.000 rows
# train <- train[sample(nrow(train), 1000000)]
# To Knit the doc will use only 10.000 rows sample
train <- train[sample(nrow(train), 10000)]


glm_model <- glm(formula = Demanda_uni_equil ~ .,
                 data = train[, list(Demanda_uni_equil,
                                     prod_cust_balance,
                                     prod_rout_balance,
                                     cust_rout_balance,
                                     last_week_balance,
                                     prod_cust_bal_mean,
                                     prod_rout_prop,
                                     cust_rout_prop)])

# Do not save on knit doc
# saveRDS(glm_model, file = 'model/glm_model.rds')
remove(glm_model)
gc()
```

```
##            used (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  2788538  149    4165202  222.5   4165202  222.5
## Vcells 44423977  339  662194844 5052.2 790369729 6030.1
```

```
lqs_model <- MASS::lqs(formula = Demanda_uni_equil ~ .,
                       data = train[, list(Demanda_uni_equil,
                                           prod_cust_balance,
                                           prod_rout_balance,
                                           cust_rout_balance,
                                           last_week_balance,
                                           prod_cust_bal_mean,
                                           prod_rout_prop,
                                           cust_rout_prop)])

# Do not save on knit doc
# saveRDS(lqs_model, file = 'model/lqs_model.rds')
remove(lqs_model)
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  2791644 149.1    4165202  222.5   4165202  222.5
## Vcells 44421602 339.0  529755876 4041.8 790369729 6030.1
```

```
rlm_model <- MASS::rlm(formula = Demanda_uni_equil ~ .,
                       data = train[,list(Demanda_uni_equil,
                                          prod_cust_balance,
                                          prod_rout_balance,
                                          cust_rout_balance,
                                          last_week_balance,
                                          prod_cust_bal_mean,
                                          prod_rout_prop,
                                          cust_rout_prop)])

# Do not save on knit doc
# saveRDS(rlm_model, file = 'model/rlm_model.rds')
remove(rlm_model)
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  2796349 149.4    4165202  222.5   4165202  222.5
## Vcells 44431962 339.0  423804701 3233.4 790369729 6030.1
```

```
caret_model <- caret::train(x = train[,list(prod_cust_balance,
                                            prod_rout_balance,
                                            cust_rout_balance,
                                            last_week_balance,
                                            prod_cust_bal_mean,
                                            prod_rout_prop,
                                            cust_rout_prop)],
                            y = train[,Demanda_uni_equil],
                            method = "lm")

# Do not save on knit doc
# saveRDS(caret_model, file = 'model/caret_model.rds')
remove(caret_model)
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  2842537 151.9    4165202  222.5   4165202  222.5
## Vcells 44532639 339.8  271235009 2069.4 790369729 6030.1
```

```
rm(list=ls())
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  2842530 151.9    4165202  222.5   4165202  222.5
## Vcells 44430600 339.0  216988008 1655.5 790369729 6030.1
```