RF

maycd

Contents

Import packages

Import data 285K obs. of 11 variables

```
default <- read.csv("train.csv", stringsAsFactors = TRUE)[c("bad_good", "GENDER", "LOAN_FLAG", "OS_PRCP_
str(default)
```

```
## 'data.frame':    285285 obs. of  11 variables:
##  $ bad_good               : int  0 0 1 0 0 0 0 0 0 0 ...
##  $ GENDER                 : Factor w/ 3 levels "1","2","X": 2 1 2 1 1 2 1 2 2 1 ...
##  $ LOAN_FLAG              : Factor w/ 2 levels "N","Y": 2 1 2 1 1 1 1 1 1 1 ...
##  $ OS_PRCP_SUM_THREE      : num  197750 0 466667 0 0 ...
##  $ OS_PRCP_SUM_SIX        : num  98875 0 233333 0 0 ...
##  $ G_OS_PRCP_SUM          : num  204750 0 0 0 0 ...
##  $ L6_CUST_DEBT_AVG_AMT   : num  222434 0 0 0 0 ...
##  $ CUST_DEBT_AMT          : num  220403 0 0 0 0 ...
##  $ L3_CUST_DEBT_AVG_AMT   : num  224465 0 0 0 0 ...
##  $ DEP_SA_OPEN_TENURE_DAYS: int  4940 3373 4781 0 1569 3362 4212 4781 812 4857 ...
##  $ DEP_SA_AVG_TENURE_DAYS : int  2196 1077 1777 0 1493 3286 1691 4689 520 4689 ...
```

Change the data type

```
default$bad_good <- factor(default$bad_good)
```

```
summary(default)
```

```
##  bad_good   GENDER     LOAN_FLAG  OS_PRCP_SUM_THREE  OS_PRCP_SUM_SIX
##  0:280401   1:147624   N:271634   Min.   :       0   Min.   :       0
##  1:  4884   2:136548   Y: 13651   1st Qu.:       0   1st Qu.:       0
##             X:  1113              Median :       0   Median :       0
##                                   Mean   :   14222   Mean   :    7111
##                                   3rd Qu.:       0   3rd Qu.:       0
##                                   Max.   :26666667   Max.   :13333333
##  G_OS_PRCP_SUM      L6_CUST_DEBT_AVG_AMT CUST_DEBT_AMT
##  Min.   :       0   Min.   :       0     Min.   :       0
##  1st Qu.:       0   1st Qu.:       0     1st Qu.:       0
##  Median :       0   Median :       0     Median :       0
##  Mean   :   14442   Mean   :   85885     Mean   :   93183
##  3rd Qu.:       0   3rd Qu.:       0     3rd Qu.:       0
##  Max.   :26000000   Max.   :40903235     Max.   :40816100
##  L3_CUST_DEBT_AVG_AMT DEP_SA_OPEN_TENURE_DAYS DEP_SA_AVG_TENURE_DAYS
##  Min.   :       0     Min.   :   0            Min.   :   0
##  1st Qu.:       0     1st Qu.: 390            1st Qu.: 299
##  Median :       0     Median : 887            Median : 744
##  Mean   :   79214     Mean   :1213            Mean   :1024
##  3rd Qu.:       0     3rd Qu.:1838            3rd Qu.:1536
##  Max.   :40990370     Max.   :6410            Max.   :6334
```

Stratified sampling split into train and test

```
set.seed(123)
split_strat <- rsample::initial_split(default, prop = 0.6, strata = 'bad_good')
default_train <- rsample::training(split_strat)
default_test <- rsample::testing(split_strat)
```

Data preprocessing

```
blueprint <- recipe(bad_good ~ ., data = default_train) %>%
  step_normalize(all_numeric(), -all_outcomes()) %>%
  step_center(all_numeric(), -all_outcomes()) %>%
  step_scale(all_numeric(), -all_outcomes()) %>%
  step_dummy(all_nominal(), -all_outcomes(), one_hot = TRUE)

prepare <- prep(blueprint, training = default_train)
prepare
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
```

```
##  predictor         10
##
## Training data contained 171171 data points and no missing data.
##
## Operations:
##
## Centering and scaling for OS_PRCP_SUM_THREE, OS_PRCP_SUM_SIX, G_OS_PRCP_S... [trained]
## Centering for OS_PRCP_SUM_THREE, OS_PRCP_SUM_SIX, G_OS_PRCP_S... [trained]
## Scaling for OS_PRCP_SUM_THREE, OS_PRCP_SUM_SIX, G_OS_PRCP_S... [trained]
## Dummy variables from GENDER, LOAN_FLAG [trained]
```

```
default_train <- bake(prepare, new_data = default_train)
summary(default_train)
```

```
##  OS_PRCP_SUM_THREE   OS_PRCP_SUM_SIX     G_OS_PRCP_SUM
##  Min.   : -0.07636   Min.   : -0.07636   Min.   : -0.07717
##  1st Qu.: -0.07636   1st Qu.: -0.07636   1st Qu.: -0.07717
##  Median : -0.07636   Median : -0.07636   Median : -0.07717
##  Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.00000
##  3rd Qu.: -0.07636   3rd Qu.: -0.07636   3rd Qu.: -0.07717
##  Max.   :137.91131   Max.   :137.91131   Max.   :138.84812
##  L6_CUST_DEBT_AVG_AMT CUST_DEBT_AMT     L3_CUST_DEBT_AVG_AMT
##  Min.   :-0.1639      Min.   :-0.1624   Min.   :-0.1637
##  1st Qu.:-0.1639      1st Qu.:-0.1624   1st Qu.:-0.1637
##  Median :-0.1639      Median :-0.1624   Median :-0.1637
##  Mean   : 0.0000      Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.:-0.1639      3rd Qu.:-0.1624   3rd Qu.:-0.1637
##  Max.   :76.8114      Max.   :69.8849   Max.   :83.5475
##  DEP_SA_OPEN_TENURE_DAYS DEP_SA_AVG_TENURE_DAYS bad_good     GENDER_X1
##  Min.   :-1.1807         Min.   :-1.1197        0:168280   Min.   :0.0000
##  1st Qu.:-0.8007         1st Qu.:-0.7924        1:  2891   1st Qu.:0.0000
##  Median :-0.3173         Median :-0.3075                   Median :1.0000
##  Mean   : 0.0000         Mean   : 0.0000                   Mean   :0.5172
##  3rd Qu.: 0.6095         3rd Qu.: 0.5584                   3rd Qu.:1.0000
##  Max.   : 4.7991         Max.   : 5.5137                   Max.   :1.0000
##    GENDER_X2        GENDER_X         LOAN_FLAG_N      LOAN_FLAG_Y
##  Min.   :0.000   Min.   :0.000000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.000   1st Qu.:0.000000   1st Qu.:1.0000   1st Qu.:0.00000
##  Median :0.000   Median :0.000000   Median :1.0000   Median :0.00000
##  Mean   :0.479   Mean   :0.003809   Mean   :0.9524   Mean   :0.04763
##  3rd Qu.:1.000   3rd Qu.:0.000000   3rd Qu.:1.0000   3rd Qu.:0.00000
##  Max.   :1.000   Max.   :1.000000   Max.   :1.0000   Max.   :1.00000
```

```
default_test <- bake(prepare, new_data = default_test)
```

```
write.csv(default_train, file = "default_train.csv", row.names = FALSE)
write.csv(default_test, file = "default_test.csv", row.names = FALSE)
rm(list = ls())
```

Read train data

```
tic()
default_train <- read.csv("default_train.csv", stringsAsFactors = TRUE)
dim(default_train)  # dataset: default_train, response: bad_good
```

```
## [1] 171171     14
```

```
toc()
```

```
## 0.97 sec elapsed
```

RF

Model before tuning

```
# number of features
n_features <- length(setdiff(names(default_train), "bad_good"))
```

```
tic()
# train a default random forest model
default_rf1 <- ranger(
  bad_good ~ .,
  data = default_train,
  mtry = floor(n_features / 3),
  respect.unordered.factors = "order",
  importance = 'impurity',
  seed = 123
)

# get OOB RMSE
(default_pred.err <- default_rf1$prediction.error)
```

```
## [1] 7.761837e-06
```

```
toc()
```

```
## 6.82 sec elapsed
```

```
default_rf1
```

```
## Ranger result
##
## Call:
##  ranger(bad_good ~ ., data = default_train, mtry = floor(n_features/3),      respect.unordered.factor
##
## Type:                             Regression
## Number of trees:                  500
## Sample size:                      171171
## Number of independent variables:  13
## Mtry:                             4
## Target node size:                 5
## Variable importance mode:         impurity
## Splitrule:                        variance
## OOB prediction error (MSE):       7.761837e-06
## R squared (OOB):                  0.9995325
```

Hyperparameter tuning

```
# create hyperparameter grid
hyper_grid <- expand.grid(
  mtry = floor(n_features * c(.33, .4, .5)),
```

```
  min.node.size = c(1, 3, 10),
  replace = c(TRUE, FALSE),
  sample.fraction = c(.5, .63, .8),
  pred.err = NA
)

tic()
# execute full cartesian grid search
for(i in seq_len(nrow(hyper_grid))) {
  # fit model for ith hyperparameter combination
  fit <- ranger(
    formula = bad_good ~ .,
    data = default_train,
    num.trees = n_features * 10,
    mtry = hyper_grid$mtry[i],
    min.node.size = hyper_grid$min.node.size[i],
    replace = hyper_grid$replace[i],
    sample.fraction = hyper_grid$sample.fraction[i],
    verbose = FALSE,
    seed = 123,
    respect.unordered.factors = 'order',
    )
  #export OOB error
  hyper_grid$pred.err[i] <- fit$prediction.error
}
toc()
```

```
## 77.74 sec elapsed
```

```
# assess top 10 models
hyper_grid %>%
  arrange(pred.err) %>%
  mutate(perc_gain = (default_pred.err - pred.err) / default_pred.err * 100) %>%
  head(10)
```

```
##    mtry min.node.size replace sample.fraction     pred.err perc_gain
## 1     6             3   FALSE            0.80 4.109852e-06  47.05052
## 2     6             1   FALSE            0.80 4.114016e-06  46.99688
## 3     6            10   FALSE            0.80 4.139908e-06  46.66329
## 4     6             1   FALSE            0.63 4.269546e-06  44.99310
## 5     6             3   FALSE            0.63 4.275577e-06  44.91540
## 6     6            10   FALSE            0.63 4.338024e-06  44.11086
## 7     6            10    TRUE            0.80 4.877209e-06  37.16424
## 8     6             1    TRUE            0.80 4.881204e-06  37.11278
## 9     6             3    TRUE            0.80 4.881204e-06  37.11278
## 10    5             1    TRUE            0.80 5.960278e-06  23.21047
```

```
best <- which.min(hyper_grid$pred.err)
```

Model after tuning

```
tic()
default_rf <- ranger(
    formula = bad_good ~ .,
```

```
    data = default_train,
    num.trees = n_features * 10,
    mtry = hyper_grid$mtry[best],
    importance = "impurity",
    min.node.size = hyper_grid$min.node.size[best],
    replace = hyper_grid$replace[best],
    sample.fraction = hyper_grid$sample.fraction[best],
    verbose = FALSE,
    seed = 123,
    respect.unordered.factors = 'order'
  )
default_rf
```

```
## Ranger result
##
## Call:
##  ranger(formula = bad_good ~ ., data = default_train, num.trees = n_features *      10, mtry = hyper
##
## Type:                             Regression
## Number of trees:                  130
## Sample size:                      171171
## Number of independent variables:  13
## Mtry:                             6
## Target node size:                 3
## Variable importance mode:         impurity
## Splitrule:                        variance
## OOB prediction error (MSE):       4.109852e-06
## R squared (OOB):                  0.9997525
```

```
# get OOB RMSE
(default_rf_pred.err <- default_rf$prediction.error)
```

```
## [1] 4.109852e-06
```

```
toc()
```

```
## 1.75 sec elapsed
```

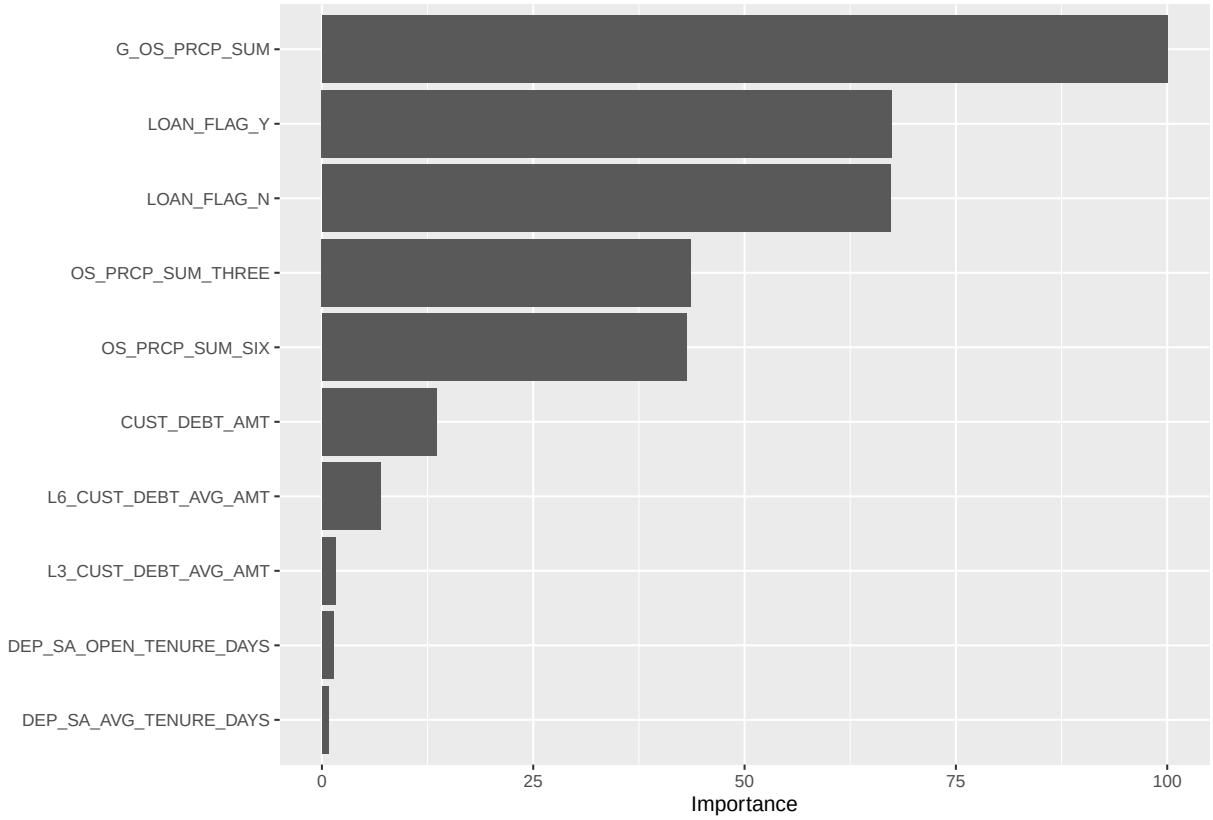
```
save(default_rf, file = "default_rf.rda")
```

Variable importance

```
vi_scores <- vi(default_rf)
head(vi_scores, 5)
```

```
## # A tibble: 5 x 2
##   Variable          Importance
##   <chr>                  <dbl>
## 1 G_OS_PRCP_SUM           658.
## 2 LOAN_FLAG_Y             443.
## 3 LOAN_FLAG_N             442.
## 4 OS_PRCP_SUM_THREE       287.
## 5 OS_PRCP_SUM_SIX         284.
```

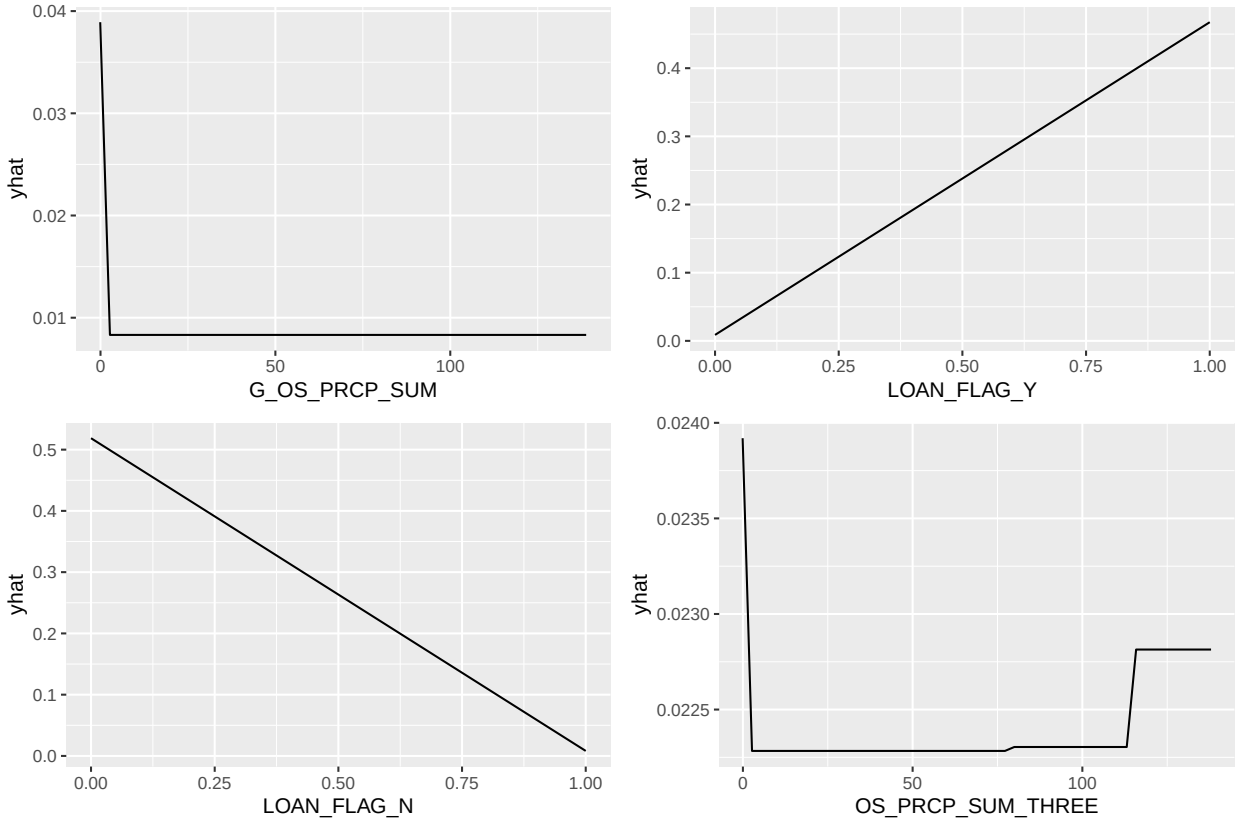
```
vip(default_rf, num_features = 10, scale = TRUE)
```



PDP plots

```
tic()
p1 <- partial(default_rf, pred.var = vi_scores[[1, 1]]) %>% autoplot()
p2 <- partial(default_rf, pred.var = vi_scores[[2, 1]]) %>% autoplot()
p3 <- partial(default_rf, pred.var = vi_scores[[3, 1]]) %>% autoplot()
p4 <- partial(default_rf, pred.var = vi_scores[[4, 1]]) %>% autoplot()
grid.arrange(p1, p2, p3, p4, ncol = 2)
```

```
## Warning: Use of `object[[1L]]` is discouraged. Use `.data[[1L]]` instead.

## Warning: Use of `object[["yhat"]]` is discouraged. Use `.data[["yhat"]]`
## instead.

## Warning: Use of `object[[1L]]` is discouraged. Use `.data[[1L]]` instead.

## Warning: Use of `object[["yhat"]]` is discouraged. Use `.data[["yhat"]]`
## instead.

## Warning: Use of `object[[1L]]` is discouraged. Use `.data[[1L]]` instead.

## Warning: Use of `object[["yhat"]]` is discouraged. Use `.data[["yhat"]]`
## instead.

## Warning: Use of `object[[1L]]` is discouraged. Use `.data[[1L]]` instead.

## Warning: Use of `object[["yhat"]]` is discouraged. Use `.data[["yhat"]]`
## instead.
```




```
toc()
```

```
## 49.07 sec elapsed
```