pjt_tree

Yen-Hsun Lin

2023-04-27

This document will explore tree methods on the training data, conduct CV and select best tuning variables, and finally calculate AUC for the best model

0. getting data

```
suppressMessages(library(data.table))
suppressMessages(library(ggplot2))
suppressMessages(library(rpart))
suppressMessages(library(rpart.plot))
suppressMessages(library(caret))
suppressMessages(library(randomForest))
suppressMessages(library(DescTools))
suppressMessages(library(gbm))
suppressMessages(library(plyr))


trn <- fread("loan_balanced_trn.csv", stringsAsFactors = TRUE)
tst <- fread("loan_balanced_tst.csv", stringsAsFactors = TRUE)

trn$TARGET <- factor(trn$TARGET)
tst$TARGET <- factor(tst$TARGET)


## doing this because the random forest predict method has some bug. It is
## sensitive to levels of factor
for (name in names(trn)[sapply(trn,class) == "factor"]){
  levels(tst[[name]]) <- levels(trn[[name]])
}


selected_fml <- TARGET ~ DAYS_EMPLOYED + OCCUPATION_TYPE + REGION_RATING_CLIENT_W_CITY +
    DAYS_LAST_PHONE_CHANGE + NAME_CONTRACT_TYPE + AMT_GOODS_PRICE +
    AMT_CREDIT + DAYS_BIRTH + NAME_EDUCATION_TYPE + FLAG_OWN_CAR +
    NAME_INCOME_TYPE + DAYS_ID_PUBLISH + CODE_GENDER + DAYS_REGISTRATION +
    FLAG_WORK_PHONE + AMT_REQ_CREDIT_BUREAU_YEAR + FLAG_OWN_REALTY +
    DEF_30_CNT_SOCIAL_CIRCLE + AMT_ANNUITY + AMT_REQ_CREDIT_BUREAU_HOUR


ctrl <- trainControl(method = "CV", number = 10)
```

```r
calc_accu <- function(actual, pred){
  mean(actual == pred)
}
classifier <- function(prob, cutoff, pos = "1", neg = "0"){
  ifelse(prob > cutoff, pos, neg)
}
```

1. Single tree

Use 10-fold CV (the train chunk is not evaluate in RMD for times sake)

```r
set.seed(432)
sing_tree_train <- train(selected_fml, data = trn, trControl = ctrl,
                         method = "rpart",
                         tuneGrid = data.frame(cp = seq(0.001,0.1,by = 0.001)))


plot(sing_tree_train)
sing_tree_train$bestTune

rpart.plot(sing_tree_train$finalModel)

sing_tree_train$bestTune
```

```r
## best cp selected is 0.002

stree <- rpart(selected_fml, data = trn, cp = 0.002)




## testing accuracy
calc_accu(predict(stree, tst, type = "class"), tst$TARGET)
```

```
## [1] 0.607
```

2. Random forest

This chunk is for demonstration. It is not eval in RMD

```r
set.seed(432)
rf_train <- train(selected_fml, data = trn, trControl = ctrl,
                         method = "rf",
                         tuneGrid = data.frame(mtry = 4:15) )
rf_train$results
plot(rf_train)
importance(RF4)
```

```r
## the cv suggest selecting 12 predictors at each split gives the lowest CV
## error
```

```r
set.seed(432)

best_RF <- randomForest(selected_fml, data = trn, trControl = ctrl,
        mtry = 12)


calc_accu(predict(best_RF, tst, type = "class"), tst$TARGET)
```

```
## [1] 0.6195
```

```r
importance(best_RF)
```

```
##                              MeanDecreaseGini
## DAYS_EMPLOYED                      591.506352
## OCCUPATION_TYPE                    407.797290
## REGION_RATING_CLIENT_W_CITY        109.506581
## DAYS_LAST_PHONE_CHANGE             475.614151
## NAME_CONTRACT_TYPE                  33.364122
## AMT_GOODS_PRICE                    323.426222
## AMT_CREDIT                         377.507610
## DAYS_BIRTH                         518.584319
## NAME_EDUCATION_TYPE                 92.025656
## FLAG_OWN_CAR                        48.573066
## NAME_INCOME_TYPE                    80.580310
## DAYS_ID_PUBLISH                    507.558332
## CODE_GENDER                         42.628148
## DAYS_REGISTRATION                  513.463771
## FLAG_WORK_PHONE                     62.231486
## AMT_REQ_CREDIT_BUREAU_YEAR         220.483741
## FLAG_OWN_REALTY                     45.293137
## DEF_30_CNT_SOCIAL_CIRCLE            73.209721
## AMT_ANNUITY                        471.834275
## AMT_REQ_CREDIT_BUREAU_HOUR           4.196389
```

3. Boosting

I use parallel running to reduce running time.

```r
library(parallel)

train_gbm <- function(d){
  train(selected_fml, data = trn, trControl = ctrl,
                         method = "gbm",
                         tuneGrid = expand.grid(
                           n.trees = c(500),
                           interaction.depth = d,
                           shrinkage = seq(0.01,0.5,by = 0.01),
                           n.minobsinnode = 10
                         ), verbose = FALSE)
}
```

```r
clus <- makeCluster(10, type = "PSOCK")
clusterEvalQ(clus, library(gbm))
clusterEvalQ(clus, library(caret))
clusterEvalQ(clus, library(plyr))
clusterEvalQ(clus, set.seed(432))
clusterExport(clus, varlist = c("trn", "ctrl", "selected_fml", "train_gbm"))

boosting_trains <- parLapply(cl = clus, 1:10, train_gbm)

stopCluster(clus)
```

```r
get_best_tune <- function(train_obj){
  best_idx <- as.numeric(rownames(train_obj$bestTune))
  print(train_obj$results[best_idx,])
}


get_best_tune(boosting_trains[[1]])
get_best_tune(boosting_trains[[2]])
get_best_tune(boosting_trains[[3]])
get_best_tune(boosting_trains[[4]])
get_best_tune(boosting_trains[[5]])
get_best_tune(boosting_trains[[6]])
get_best_tune(boosting_trains[[7]])
get_best_tune(boosting_trains[[8]])
get_best_tune(boosting_trains[[9]])
get_best_tune(boosting_trains[[10]])


test <- boosting_trains[[2]]$finalModel
## The best model is shrinkage = 0.05 with interaction.depth = 2
```

```r
selected_fml1 <- as.character(TARGET) ~ DAYS_EMPLOYED + OCCUPATION_TYPE + REGION_RATING_CLIENT_W_CITY +
    DAYS_LAST_PHONE_CHANGE + NAME_CONTRACT_TYPE + AMT_GOODS_PRICE +
    AMT_CREDIT + DAYS_BIRTH + NAME_EDUCATION_TYPE + FLAG_OWN_CAR +
    NAME_INCOME_TYPE + DAYS_ID_PUBLISH + CODE_GENDER + DAYS_REGISTRATION +
    FLAG_WORK_PHONE + AMT_REQ_CREDIT_BUREAU_YEAR + FLAG_OWN_REALTY +
    DEF_30_CNT_SOCIAL_CIRCLE + AMT_ANNUITY + AMT_REQ_CREDIT_BUREAU_HOUR

best_boost <- gbm(selected_fml1, data = trn, n.trees = 500,
                  interaction.depth = 2, shrinkage = 0.05,
                  n.minobsinnode = 10,
                  )
```

```
## Distribution not specified, assuming bernoulli ...
```

```r
pred_boost <- classifier(predict(best_boost, tst, type = "response"), 0.5)
```

```
## Using 500 trees...
```

```r
calc_accu(pred_boost, tst$TARGET)
```

```
## [1] 0.6376
```