Credit score approval prediction

Jan Alexander Jensen

2020/4/16

Handling nominal variable with dummy variable

Use library 'fastDummies' to handle the nominal variable $OCCUPATION_TYPE.

```r
library(fastDummies)
library(janitor)

dummy_var <- function(data){

  # Since the library 'fastDummies' tansforms all factor variable into dummy variables,
  # we will convert our target "y" (factor) into a character variable
  # to avoid it being transformed to dummy variable.

  data$y <- as.numeric(as.character(data$y))

  # transform all factor variables to dummy variables,
  # and removes the original variables that were used to generate the dummy variables.
  data_dummy <- fastDummies::dummy_cols(data, remove_selected_columns=TRUE)

  # column name convention fix (mlr3 name convention - space to underscore)
  data_dummy <- clean_names(data_dummy)


  data_dummy <- as.data.frame(sapply(data_dummy, as.numeric))
  data_dummy$y <- as.factor(data_dummy$y)

  dummy_var <- data_dummy
}

# --------------------------------------------
# ----------------------- handle missing data
# ----------------------- OCCUPATION_TYPE
# -------------------------------------------


dl_dummy_data <- read.csv2("../../credit_card_prediction/dl_na_data.csv", header = TRUE)
dl_dummy_data <- dummy_var(dl_dummy_data)

mf_dummy_data <- read.csv2("../../credit_card_prediction/mf_na_data.csv", header = TRUE)
mf_dummy_data <- dummy_var(mf_dummy_data)

mice_dummy_data <- read.csv2("../../credit_card_prediction/mice_na_data.csv", header = TRUE)
mice_dummy_data <- dummy_var(mice_dummy_data)

# row, column, NA
cat("dl_dummy_data\t", dim(dl_dummy_data), any(is.na(dl_dummy_data)), "\n")
cat("mf_dummy_data\t", dim(mf_dummy_data), any(is.na(mf_dummy_data)), "\n")
cat("mice_dummy_data\t", dim(mice_dummy_data), any(is.na(mice_dummy_data)), "\n")
```

```r
rm(dl_dummy_data, mf_dummy_data, mice_dummy_data)

# store data with dummy variable
# write_csv2(dl_dummy_data,"credit_card_prediction/dummy_data/dl_dummy_data.csv")
# write_csv2(mf_dummy_data,"credit_card_prediction/dummy_data/mf_dummy_data.csv")
# write_csv2(mice_dummy_data,"credit_card_prediction/dummy_data/mice_dummy_data.csv")
```

```
## dl_dummy_data     25134 55 FALSE
## mf_dummy_data     36457 55 FALSE
## mice_dummy_data   36457 55 FALSE
```

Load all data for training (one-hot, dummy, IV)

```r
library(mlr3)

# function to load data into task and define target
dataToTask <- function(path, id, sep=';', header=TRUE){
  dt <- read.csv2(path, sep = sep, header = header)
  dt <- as.data.frame(sapply(dt, as.numeric))
  dt$y <- as.factor(dt$y)
  dataToTask <- TaskClassif$new(id = id, backend = dt, target = "y")
}

dl_dummy_task <-
  dataToTask("../../credit_card_prediction/dummy_data/dl_dummy_data.csv", "dl_dummy")
dl_oh_task <-
  dataToTask("../../credit_card_prediction/oh_data/dl_oh_data.csv", "dl_oh", sep=',')
dl_iv_task <-
  dataToTask("../../credit_card_prediction/iv_data/dl_iv_data.csv", "dl_iv")

mf_dummy_task <-
  dataToTask("../../credit_card_prediction/dummy_data/mf_dummy_data.csv", "mf_dummy")
mf_oh_task <-
  dataToTask("../../credit_card_prediction/oh_data/mf_oh_data.csv", "mf_oh", sep=',')
mf_iv_task <-
  dataToTask("../../credit_card_prediction/iv_data/mf_iv_data.csv", "mf_iv")

mice_dummy_task <-
  dataToTask("../../credit_card_prediction/dummy_data/mice_dummy_data.csv", "mice_dummy")
mice_oh_task <-
  dataToTask("../../credit_card_prediction/oh_data/mice_oh_data.csv", "mice_oh", sep=',')
mice_iv_task <-
  dataToTask("../../credit_card_prediction/iv_data/mice_iv_data.csv", "mice_iv")

# combine all tasks into one list
dl <- list(dummy=dl_dummy_task, oh=dl_oh_task, iv=dl_iv_task)
mf <- list(dummy=mf_dummy_task, oh=mf_oh_task, iv=mf_iv_task)
mice <- list(dummy=mice_dummy_task, oh=mice_oh_task, iv=mice_iv_task)


# tasks[["<type>"]][["<code>"]], tasks$<type>$<code>
# ex. tasks[["dl"]][["dummy"]], tasks$dl$dummy
tasks <- list(dl=dl, mf=mf, mice=mice)
```

```r
# remove unused variables (save memory)
rm(dl, mf, mice)
rm(dl_dummy_task, mf_dummy_task, mice_dummy_task)
rm(dl_oh_task, mf_oh_task, mice_oh_task)
rm(dl_iv_task, mf_iv_task, mice_iv_task)

# print task ids and data size
for(t in tasks){
  for(c in t){
    cat(c$id, dim(c$data()), "\n")
  }
}
```

```
## dl_dummy 25134 47
## dl_oh 25134 55
## dl_iv 25134 33
## mf_dummy 36457 47
## mf_oh 36457 55
## mf_iv 36457 33
## mice_dummy 36457 47
## mice_oh 36457 55
## mice_iv 36457 33
```

```r
rm(t, c)
```

KNN

```r
#warnings()
#

library(mlr3)
library(mlr3learners)
library(mlr3viz)
library(ggplot2)
library(gridExtra)

# turn info log off, only show warnings
lgr::get_logger("mlr3")$set_threshold("warn")

# train one model with fixed seed
train_model <- function(task, learner, resampling){
  set.seed(2020)
  print(task$id)
  start_time <- Sys.time()
  model <- resample(task, learner, resampling, store_models = TRUE)
  end_time <- Sys.time()
  duration <- (end_time-start_time)[[1]]
  print(model)
  print(paste0("training time (5 fold CV): ", duration))
  train_model <- model
}

# train all task with one learner
```

```r
train_all <- function(tasks, learner, resampling){

  models <- list()
  miss_name <- c('dl', 'mf', 'mice')
  code_name <- c('dummy', 'oh', 'iv')

  for(missing in miss_name){
    for(coding in code_name){
      name <- paste0(missing, "_", coding)
      task <- tasks[[missing]][[coding]]
      models[[name]] <- train_model(task, learner, resampling)
    }
  }
  train_all <- models
}

# evaluate multiple models with AUC
evaluate_models <- function(models){
  for(m in models){
    name <- m$task$id
    auc <- m$aggregate(msr("classif.auc"))[[1]]
    max_auc <- max(m$score(msr("classif.auc"))[,9])
    print(sprintf("%10s: %.4f (max: %.4f)", name, auc, max_auc))
    #cat(paste0(name, ": ", auc, "\t(max: ", max_auc, ")\n"))
  }
}

multiplot_roc <- function(models){
  plots <- list()
  k <- 1
  for(m in models){
    name <- m$task$id
    plots[[k]] <- autoplot(m, type = "roc") + xlab("") + ylab("") + ggtitle(name)
    k <- k+1
  }
  do.call("grid.arrange", plots)
  # grid.arrange(plots[[1]], plots[[2]], plots[[3]],
  #              plots[[4]], plots[[5]], plots[[6]],
  #              plots[[7]], plots[[8]], plots[[9]])
}

resampling = rsmp("cv", folds = 5)
learner <- lrn("classif.kknn", id = "knn", predict_type = "prob", k=15, distance=2, scale=FALSE)
models <- train_all(tasks, learner, resampling)
```

```
## [1] "dl_dummy"
## <ResampleResult> of 5 iterations
## * Task: dl_dummy
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 7.27843403816223"
## [1] "dl_oh"
## <ResampleResult> of 5 iterations
```

```
## * Task: dl_oh
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 6.14534783363342"
## [1] "dl_iv"
## <ResampleResult> of 5 iterations
## * Task: dl_iv
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 5.05030107498169"
## [1] "mf_dummy"
## <ResampleResult> of 5 iterations
## * Task: mf_dummy
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 13.8096261024475"
## [1] "mf_oh"
## <ResampleResult> of 5 iterations
## * Task: mf_oh
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 11.7247531414032"
## [1] "mf_iv"
## <ResampleResult> of 5 iterations
## * Task: mf_iv
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 12.2844760417938"
## [1] "mice_dummy"
## <ResampleResult> of 5 iterations
## * Task: mice_dummy
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 13.4485490322113"
## [1] "mice_oh"
## <ResampleResult> of 5 iterations
## * Task: mice_oh
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 11.7488088607788"
## [1] "mice_iv"
## <ResampleResult> of 5 iterations
## * Task: mice_iv
## * Learner: knn
## * Warnings: 0 in 0 iterations
## * Errors: 0 in 0 iterations
## [1] "training time (5 fold CV): 12.7940769195557"
```

```r
evaluate_models(models)
```

```
## [1] "  dl_dummy: 0.7445 (max: 0.7839)"
## [1] "     dl_oh: 0.7568 (max: 0.7838)"
## [1] "     dl_iv: 0.7445 (max: 0.7841)"
## [1] "  mf_dummy: 0.7520 (max: 0.7884)"
## [1] "     mf_oh: 0.7506 (max: 0.7744)"
## [1] "     mf_iv: 0.7520 (max: 0.7885)"
## [1] "mice_dummy: 0.7519 (max: 0.7884)"
## [1] "   mice_oh: 0.7506 (max: 0.7745)"
## [1] "   mice_iv: 0.7520 (max: 0.7886)"
```

```r
multiplot_roc(models)
```