



Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного автономного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и управление»

КАФЕДРА ИУК4 «Программная инженерия»

Лабораторная работа №4

Логические методы классификации многомерных объектов пересекающихся классов

ДИСЦИПЛИНА: «Методы машинного обучения»

Выполнил: студент гр. ИУК5-72Б

(Подпись)

Ли Р. В.

(Ф.И.О.)

Проверил:

(Подпись)

(Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:

- Оценка:

Вариант 14

Разработать логический классификатор с использованием алгоритмов «Кора», «ID3», «CART», «Бэггинг», «Бустинг», генетического алгоритма для классификации товаров супермаркета по категориям «Скидки - Нет» 0-4 %, «Скидка-мини» 5-25 %, «Выгодная Скидка» 26-40%, «Супер Скидка» 50-70% за пять дней.

Ko д	K п	K в	K с	K ч	K п	Kс б	K вс	K з	Kв н	K л	K о	Kт	
02 0	1							0.7 4	0.5	1	1	0.4 5	
02 5	0.7 7											1	
03 0													
03 5													
04 0													
04 5													
05 0													
00 1												0.1 5	
00 3	1		1				1			1		0.2 3	
01 1	0.5 8												

1). Для алгоритма Коры: выбрать частоту встречаемости конъюнкций MinNum= 7

2). Для генетического алгоритма выбрать: генную бинарную комбинацию **1010111**

3). Для алгоритма CART в функции gpart выбрать параметр method = "class"

4). Для алгоритма Bagging в функции RandomForest выбрать параметр N.trees= 624, в функции train выбрать параметр method = "bagFDA"

5). Для алгоритма Boosting в функции gbm выбрать параметр N.trees = 340, параметр distribution = "bernouilly", параметр bag.Fraction = 0.441

6). Результаты визуализировать и сравнить.

task.R

```
required_pkgs <- c("rpart", "randomForest", "gbm", "caret", "Rweka")
missing_pkgs <- required_pkgs[!(required_pkgs %in% installed.packages()[, "Package"])]
if(length(missing_pkgs)>0){
  stop(paste0("Пакеты отсутствуют: ", paste(missing_pkgs, collapse=", "),
             ". Установите их, например: install.packages(c('",
             paste(missing_pkgs, collapse='','')), ')))")
}
library(rpart)
library(randomForest)
library(gbm)
library(caret)
library(Rweka)

set.seed(123)

raw <- data.frame(
  Код = c("020", "025", "030", "035", "040", "045", "050", "001", "003", "011"),
  Кп = c(1, 0.77, NA, NA, NA, NA, NA, 1, 0.58),
  Кв = c(NA, NA, NA, NA, NA, NA, NA, NA, NA),
  Кс = c(NA, NA, NA, NA, NA, NA, NA, 1, NA),
  Кч = c(NA, NA, NA, NA, NA, NA, NA, NA, NA),
  Кп2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA),
  Ксб = c(NA, NA, NA, NA, NA, NA, NA, NA, NA),
  Квс = c(0.7, NA, NA, NA, NA, NA, NA, NA, NA),
  Кз = c(0.54, NA, NA, NA, NA, NA, 0.15, NA, NA),
  Квн = c(1, 1, NA, NA, NA, NA, NA, 1, NA),
  Кл = c(1, NA, NA, NA, NA, NA, NA, NA, NA),
  Ко = c(0.45, 1, NA, NA, NA, NA, NA, 0.23, NA),
  stringsAsFactors = FALSE
)
extra <- data.frame(
  Код = "014",
  Кп = NA, Кв = NA, Кс = NA, Кч = NA, Кп2 = NA, Ксб = NA,
  Квс = NA, Кз = NA, Квн = NA, Кл = NA, Ко = NA, stringsAsFactors =
  FALSE
)

raw[is.na(raw)] <- 0

raw$Эталонная_цена <- NA
raw$Наименование <- NA
raw$Эталонная_цена[raw$Код=="014"] <- 44
```

```

raw$Наименование[raw$Код=="014"] <- "Лимонад Аквалайн"

num_cols <- setdiff(names(raw), c("Код","Наименование"))
raw[num_cols] <- lapply(raw[num_cols], as.numeric)

days <- 1:5
data5 <- do.call(rbind, lapply(days, function(d){
  df <- raw
  df$Day <- d
  df$Kbc <- df$Kbc * (1 + 0.05*(d-1))
  df$Kz <- df$Kz * (1 + 0.03*(d-1))
  df$Ko <- df$Ko * (1 + 0.02*(d-1))
  df
})))
rownames(data5) <- NULL

data5$score_raw <- with(data5, 10*Kп + 8*Kbc + 6*Kз + 5*Kо + 3*Kл)
minr <- min(data5$score_raw)
maxr <- max(data5$score_raw)
if(maxr - minr == 0) data5$DiscountPct <- 0 else {
  data5$DiscountPct <- round( (data5$score_raw - minr) / (maxr - minr) * 70,
  1)
}
data5$Category <- cut(data5$DiscountPct,
                      breaks = c(-Inf,4,25,40,Inf),
                      labels = c("Скидки - Нет","Скидка-мини","Выгодная
Скидка","Супер Скидка"),
                      right = TRUE)
data5$Category <- as.factor(data5$Category)

cat("Данные после генерации (первые строки):\n")
print(head(data5))

model_df <- data5[, !(names(data5) %in%
c("Код","Наименование","score_raw","DiscountPct"))]
model_df$Код <- data5$Код

predictors <-
c("Кп","Кв","Кс","Кч","Кп2","Ксб","Квс","Кз","Квн","Кл","Ко","Day")
predictors <- predictors[predictors %in% names(model_df)]
train_data <- model_df[, predictors]
train_data$Category <- model_df$Category

set.seed(42)
trainIndex <- createDataPartition(train_data$Category, p = .75, list = FALSE)
train <- train_data[trainIndex, ]
test <- train_data[-trainIndex, ]

jrip_control <- Weka_control(N = 7) # попытка передать MinNum = 7
cat("Обучаем JRip (\\"Kopa\\") с MinNum=7 ... \n")
jrip_form <- as.formula("Category ~ .")
jrip_model <- tryCatch({
  JRip(jrip_form, data = train, control = jrip_control)
}, error = function(e){
  warning("JRip обучение не сработало: ", e$message)
  NULL
})

cat("Обучаем CART (rpart) с method='class' ... \n")

```

```

cart_model <- rpart(Category ~ ., data = train, method = "class", control =
rpart.control(cp = 0.01))

cat("Обучаем RandomForest (bagging style) с ntree = 624 ...\\n")
rf_model <- randomForest(Category ~ ., data = train, ntree = 624)

cat("Обучаем bagFDA через caret::train (для сравнения) ...\\n")
ctrl <- trainControl(method = "cv", number = 5)
bagfda_model <- tryCatch({
  train(Category ~ ., data = train, method = "bagFDA", trControl = ctrl)
}, error = function(e){
  warning("caret::bagFDA не сработал: ", e$message)
  NULL
})

cat("Обучаем gbm (boosting) с n.trees = 340, distribution='multinomial' (так
как 4 класса) ...\\n")
gbm_model <- tryCatch({
  gbm.fit <- gbm(formula = as.formula("Category ~ ."),
                 distribution = "multinomial",
                 data = train,
                 n.trees = 340,
                 interaction.depth = 3,
                 n.minobsinnode = 10,
                 shrinkage = 0.05,
                 bag.fraction = 0.441,
                 verbose = FALSE)
  gbm.fit
}, error = function(e){
  warning("gbm обучение не сработало: ", e$message)
  NULL
})

cat("Генетический подход: применяем заданную маску 1010111 для отбора
признаков и обучаем CART...\\n")
gene_mask_str <- "1010111"
feat_for_mask <- predictors[predictors != "Day"] # исключим Day из маски,
оставим только основные
k <- min(nchar(gene_mask_str), length(feat_for_mask))
mask_vec <- as.integer(strsplit(substr(gene_mask_str,1,k),"")[[1]])
selected_feats <- feat_for_mask[which(mask_vec == 1)]
if(length(selected_feats) == 0){
  warning("Маска не выбрала ни одного признака – добавляю Day в качестве
признака.")
  selected_feats <- "Day"
}
form_ga <- as.formula(paste("Category ~", paste(c(selected_feats, "Day"),
collapse = " + ")))
ga_cart_model <- rpart(form_ga, data = train, method = "class", control =
rpart.control(cp = 0.01))

predict_and_eval <- function(model, model_name, test_df){
  cat("Оцениваю модель:", model_name, "\\n")

  if(is.null(model)) return(list(name=model_name, ok=FALSE))

  if(!"Category" %in% colnames(test_df)) stop("В test_df нет колонки
'Category'")
}

```

```

preds <- NA

if(inherits(model, "rpart")){
  preds <- predict(model, test_df, type = "class")
} else if(inherits(model, "JRip") || inherits(model, "weka_classifier")){
  preds <- predict(model, test_df)
  preds <- factor(preds, levels = levels(test_df$Category))
} else if(inherits(model, "randomForest")){
  preds <- predict(model, test_df)
} else if(inherits(model, "train")){ # caret
  preds <- predict(model, test_df)
} else if(inherits(model, "gbm")){
  pb <- predict(model, newdata = test_df, n.trees = model$n.trees, type =
"response")
  preds_idx <- apply(pb, 1, which.max)
  preds <- factor(levels(train$Category)[preds_idx], levels =
levels(test_df$Category))
} else {
  preds <- tryCatch({
    predict(model, test_df)
  }, error = function(e) NULL)
}

if(is.null(preds)) return(list(name=model_name, ok=FALSE))

cm <- confusionMatrix(preds, test_df$Category)

list(name=model_name, ok=TRUE, preds=preds, cm=cm)
}

res_jrip <- predict_and_eval(jrip_model, "JRip (Kopa, MinNum=7)", test)
res_cart <- predict_and_eval(cart_model, "CART (rpart)", test)
res_rf <- predict_and_eval(rf_model, "RandomForest (bagging style
ntree=624)", test)
res_bagfda <- if(!is.null(bagfda_model)) predict_and_eval(bagfda_model,
"bagFDA (caret)", test) else NULL
res_gbm <- predict_and_eval(gbm_model, "GBM (boosting n.trees=340)", test)
res_ga <- predict_and_eval(ga_cart_model, "GA-mask (1010111) + CART", test)

results_list <- list(res_jrip, res_cart, res_rf, res_bagfda, res_gbm, res_ga)
successful <- Filter(function(x) !is.null(x) && x$ok, results_list)

accs <- sapply(successful, function(x) round(x$cm$overall["Accuracy"],4))
names(accs) <- sapply(successful, function(x) x$name)
cat("\nТочности моделей (Accuracy):\n")
print(accs)

barplot(accs, main = "Сравнение точности моделей", ylab = "Accuracy", ylim =
c(0,1))
grid()

for(res in successful){
  cat("\n--- Матрица ошибок для:", res$name, "---\n")
  print(res$cm$table)
  tmp <- as.matrix(res$cm$table)
  op <- par(no.readonly = TRUE)
  par(mar = c(4,4,3,2))
}

```

```

image(1:nrow(tmp), 1:ncol(tmp), t(apply(tmp, 2, rev)), axes = FALSE, main =
paste("Confusion:", res$name))
axis(1, at = 1:nrow(tmp), labels = rownames(tmp))
axis(2, at = 1:ncol(tmp), labels = rev(colnames(tmp)))
par(op)
}

if(!is.null(rf_model)){
  cat("\nВажность переменных (randomForest):\n")
  print(importance(rf_model))
  varImpPlot(rf_model, main = "Variable Importance (RandomForest)")
}
if(!is.null(cart_model)){
  cat("\nCART: таблица переменных (variable importance):\n")
  print(cart_model$variable.importance)
  plot(cart_model); text(cart_model, use.n=TRUE)
}
if(!is.null(gbm_model)){
  cat("\nGBM: relative influence:\n")
  print(summary(gbm_model, n.trees = 340, plotit = FALSE))
  summary(gbm_model, n.trees = 340)
}

best_model_name <- names(which.max(accs))
cat("\nЛучшая модель по Accuracy:", best_model_name, "\n")
best_res <- successful[[which.max(accs)]]

final_out <- data.frame(
  Код = data5$Код[-trainIndex], # коды тестовых строк
  Day = test$Day,
  True = test$Category,
  Predicted = if(!is.null(best_res$preds)) as.character(best_res$preds) else
NA,
  stringsAsFactors = FALSE
)

write.csv(final_out, file = "final_predictions_by_best_model.csv", row.names
= FALSE)
cat("\nСохранён файл final_predictions_by_best_model.csv в рабочей
директории.\n")

```

