Лабораторная работа:

«DATA MINING CUP 2019»

Сунцов Евгений Валерьевич

Группа: ИСУП-19-1м

2020

**DATA MINING CUP 2019**

**КУБОК ПО ИНТЕЛЛЕКТУАЛЬНОМУ АНАЛИЗУ ДАННЫХ 2019**

Выявление случаев мошенничества при самопроверке в розничной торговле.

Растет число станций самоконтроля. Это включает стационарные самопроверки, когда клиенты доставляют свою корзину для покупок на станцию сканирования и оплачивают свою продукцию. Во-вторых, существуют полустационарные самопроверки, когда клиенты сканируют свою продукцию напрямую и платят только за прилавок. Клиенты либо используют свой смартфон для сканирования, либо магазин предоставляет мобильные сканеры. Вы, вероятно, уже столкнулись с этим.

Этот автоматизированный процесс помогает избежать длинных линий и ускоряет процесс оплаты для отдельных клиентов. Но как розничные торговцы могут предотвратить злоупотребление доверием, которое они оказывают клиентам? Как они могут решать, какие покупки нужно проверять, чтобы выявить мошенников, не раздражая невинных клиентов?

Это тема Кубка DATA MINING в этом году.

**Сценарий**

Известный розничный торговец продуктами питания внедрил систему самообслуживания, которая позволяет клиентам сканировать свои товары с помощью портативного мобильного сканера во время совершения покупок.

Этот вид платежей создает для розничных торговцев риск того, что определенное число клиентов воспользуется этой свободой для совершения мошенничества, не сканируя все предметы в своей тележке.

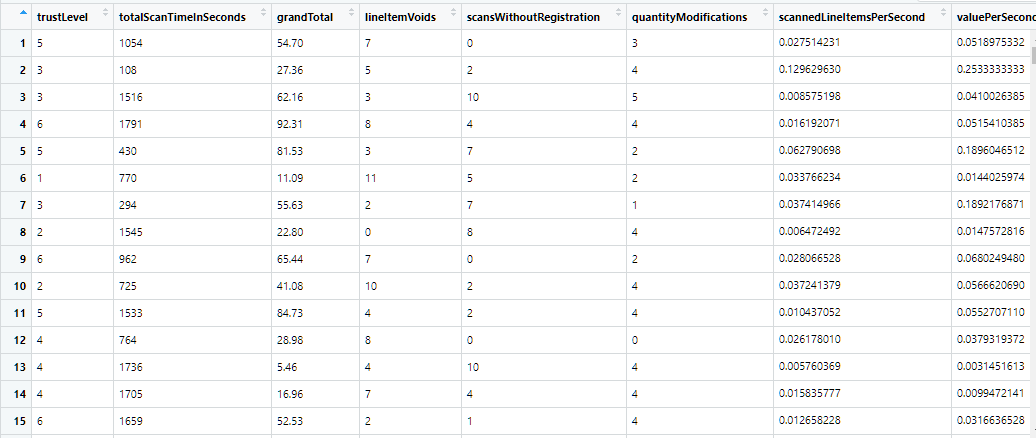
Эмпирические исследования, проведенные поставщиками, показали, что расхождения обнаруживаются примерно в 5% всех самостоятельных операций. В исследовании не проводится различия между фактическими мошенническими намерениями клиента, непреднамеренными ошибками или техническими проблемами с сканерами. Чтобы свести к минимуму потери, компания розничной торговли пищевыми продуктами надеется выявить случаи мошенничества с помощью целевых последующих проверок. В данном случае задача заключается в том, чтобы как можно меньше сократить число проверок, с тем чтобы избежать ненужных дополнительных расходов, а также не допустить оттока невинных клиентов из-за ложных обвинений. Однако в то же время цель заключается в том, чтобы выявить как можно больше мошеннических снимков.

Целью участвующих групп является создание модели для классификации сканирования как мошеннического или непроизвольного. В классификации не учитывается, было ли мошенничество совершено преднамеренно или непреднамеренно. Для создания этой модели команды получают информацию о сканировании и их классификации в обучающем наборе.

**Данные**

Реальные анонимные данные в виде структурированных текстовых файлов (csv) предоставляются для выполнения задачи.

Эти файлы содержат отдельные наборы данных.



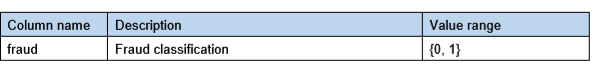
**Задача**

Использование исторических данных для создания математической модели, позволяющей достоверно классифицировать сканирование как мошенническое или не мошенническое.

Файлы "train.csv" и "test.csv" имеют одинаковую структуру, но отличаются друг от друга тем, что в последнем нет классификационной колонки.

Оба файла соответствуют свойствам, перечисленным в разделе Данные.

Для передачи данных о решении следует использовать один файл, содержащий следующую информацию:



Здесь нет ключевого атрибута. Количество и порядок прогнозируемых классификаций должны соответствовать числу и порядку соответствующих наборов данных или сканирования в файле " test.csv ".

Возможными значениями столбца "мошенничество" являются целые значения 0 или 1.

Возможный отрывок из файла решения может выглядеть так:

fraud

1

0

0

…

**Ход решения**

# Установка необходимых пакетов

library(caretEnsemble)

# Импорт файла с тренировочными данными и тестовыми

trainData <- read.csv("C:\\Users\\Evgenii\\Documents\\SMIA\\Lab1\\DMC 2019\\dmc2019-master\\train.csv",sep="|",stringsAsFactors = FALSE)

testData <- read.csv("C:\\Users\\Evgenii\\Documents\\SMIA\\Lab1\\DMC 2019\\dmc2019-master\\test.csv", sep="|",stringsAsFactors = FALSE)

set.seed(777)

# Конвертирование столбца fraud в формат фактора

trainData$fraud <- as.factor(trainData$fraud)

#levels(trainData$fraud) <- c("0","1")

levels(trainData$fraud) <- c("trusty","fraud")

### Анасмбль моделей

## Алгоритмы суммирования - запуск нескольких алгоритмов за один вызов

# Число повторений 3 и

#деление набора тренировочных данных случайным образом на 10 частей,

#а затем использование каждой из 10 частей в качестве набора тестовых

#данных для модели, подготовленной на других 9

trainControl <- trainControl(method="repeatedcv",

number=10,

repeats=3,

savePredictions=TRUE,

classProbs=TRUE)

# Модели: Рандомный лес, Бустинг, Метод опорных векторов

algorithmList <- c('rf','xgbDART', 'svmRadial')

set.seed(700)

models <- caretList(fraud~., data=trainData, trControl=trainControl, methodList=algorithmList)

results <- resamples(models)

summary(results)

save(models, file="models\_final.RData")

# Объединение прогнозов моделей

# для формирования окончательного прогноза

set.seed(777)

stackControl <- trainControl(method="repeatedcv",

number=10,

repeats=3,

savePredictions=TRUE,

classProbs=TRUE)

# Ансамбль предсказаний моделей,

# чтобы сформировать новый комбинированный прогноз на основе glm (Обобщенной линейной модели)

stack.glm <- caretStack(models, method="glm", metric="Accuracy", trControl=stackControl)

save(stack.glm,file="combined\_predictions.RData")

print(stack.glm)

# Прогноз testData

stack\_predicteds <- predict(stack.glm, newdata=testData)

head(stack\_predicteds)

write.csv(stack\_predicteds,file="predictions.csv")

save.image("script.RData")

levels(stack\_predicteds) <- c("0","1")

testData$fraud <- stack\_predicteds

# Сохранение результатов в testData\_predictions.csv

write.table(testData, file="testData\_predictions.csv",row.names=FALSE,sep="|")