KKBOX'S CHURN PREDICTION CHALLENGE

Submitted in partial fulfillment of the

requirements of the course

CS 636: Data Analytics with R Programming

By

SHUBHAM GULIA (sg952)

RAVI VED (rsv9)

ISHA PANESAR (ip62)

LAKHAN CHHABRIYA (lsc25)

Under the Guidance of:

Prof. Zhi Wei

MASTER OF SCIENCE IN INFORMATION SYSTEMS

New Jersey Institute of Technology

Contents

Project Description:

KKBOX is Asia's leading music streaming service, holding the world's most comprehensive Asia-Pop music library with over 30 million tracks. They offer a generous, unlimited version of their service to millions of people, supported by advertising and paid subscriptions. This delicate model is dependent on accurately predicting churn of their paid users.

In this competition we are tasked to build an algorithm that predicts whether a user will churn after their subscription expires. Currently, the company uses survival analysis techniques to determine the residual membership life time for each subscriber. We tried to predict user activity based on the data provided. This will help KKBOX to anticipate new insights to why user's leave so they can be proactive in keeping users.

Data:

The train and the test data are selected from users whose membership expire within a certain month. The train data consists of users whose subscription expires within the month of February 2017, and the test data is with users whose subscription expires within the month of March 2017. This means we are looking at user churn or renewal roughly in the month of March 2017 for train set, and the user churn or renewal roughly in the month of April 2017. Train and test sets are split by transaction date, as well as the public and private leaderboard data.

Files used:

1. train.csv
2. transaction.csv
3. transaction_v2.csv
4. Sample_submission_zero.csv

Data fields:

1. msno
2. Is_churn
3. Transaction_data
4. Plan_list_price
5. Payment_method_id
6. Payment_plan_days
7. Actual_amount_paid
8. Is_auto_renew
9. Membership_expire_date
10. Is_cancel

Source code:

```
install.packages("data.table")
library(data.table)
library(dplyr)
library(glmnet)
library(readr)

## Load data in csv files
trainfile=fread("train.csv", sep = ",", header= TRUE)
transactionfile=fread("transactions.csv", sep = ",", header= TRUE)
## merge trainfile and transactionfile by "msno"
Data = merge(trainfile, transactionfile, by = "msno", all = FALSE)

## Train the MODEL
MyTrainModel = glm(is_churn ~ transaction_date+plan_list_price +
                   payment_method_id+ payment_plan_days+actual_amount_paid+
                   is_auto_renew + membership_expire_date +
                   is_cancel, family=binomial(link='logit'), data = Data)

## Working on kaggle submission dataset
kaggleData = fread("sample_submission_v2.csv", sep = ",", header= TRUE)
transaction2 = fread("transactions_v2.csv", sep = ",", header= TRUE)
transaction2 = rbind(transaction2,transaction)
DataKG = merge(kaggleData, transaction2, by = "msno", all = FALSE)
#Prediction
kgPred = predict(MyTrainModel, type='response', newdata = DataKG)
dataOutput = data.frame(msno=DataKG$msno,is_churn=kgPred)
dataOutput = dataOutput %>% select(msno,is_churn) %>% group_by(msno) %>% summarise(is_churn =
mean(is_churn))
write.csv(dataOutput, "submission_Preacher.csv",row.names = FALSE)
```

Data preprocessing:

The dataset provided had a lot of unnecessary or redundant features. So, the first step was filtering the data and extracting the required or relevant features. The next step was generating additional features so as to obtain a good bit but also making sure that we don't over fit. After this the data had to be transformed into a format as required by the glm package so as to build the model.

Data Modelling:

We started off modelling the data using random forest technique but the model we obtained was not a good fit and made poor predictions. So, we decided to go for another model. We thought of using random forest, but to building a model with good prediction performance requires the value of the ntree parameter to be high that leads to very high execution time. Also, random forest might lead to overfitting so we opted for extreme gradient boosting. Since boosting tries to reduce error by mainly reducing bias and also to some extent the variance, the model proved to be a good fit for the data. The main advantage of using glm was the execution speed and model performance.

Contribution Statement:

The completion of project was a team work with equal contribution from all the group members to put the entire code together. Every member individually worked on the part of the code that one knows better and helped each other with the understanding of the program. Ravi and Isha worked towards data cleaning, getting rid of unnecessary features. Shubham and Lakhan worked on identifying, extracting and generating the features relevant and getting the data into a format required for glm and implemented the technique.

Acknowledgement:

We would like to thank Prof. Dr. Zhi Wei for providing us with the opportunity of working on this project which proved to be very knowledgeable and helpful. Data Analytics with R programming being a new subject of study for us, working on this project proved to be very interesting. We would also like to thank TA Mr. Fei Tan for helping us overcoming various milestones in this project and clearing our doubts during the work on this project. We would finally thank all the classmates who replied to our queries on moodle which have proved to be helpful.

Kaggle results: