**SFU Beedie Business Analytics Hackathon 2019**

**RStudio Cheatsheet**

**Introduction**

To be competitive in the hackathon, each team needs to develop several models that TOGETHER meet three requirements: (1) capturing the largest number of target events (e.g., lost customers) using only 40% of the sample, (2) interpretability, and (3) taking only a limited amount of time. Note the word TOGETHER: it’d be ideal to have ONE model that can achieve all three but we can have several models that vary in predictive performances and interpretability, and a comparison of these models would allow us to identify the minimum viable product (MVP) within a tight time frame.

While the hackathon is not like most analytics projects where 80% of your time in data wrangling, knowing some data manipulation would come in handy.

The below R scripts are based on the hackathon case in 2017

**RStudio**

* Installing packages

install.packages("tidyverse")

install.packages("car")

install.packages("rpart")

install.packages("rpart.plot")

install.packages("nnet")

install.packages("randomForest")

install.packages("effects")

library("tidyverse")

library("car")

library("rpart")

library("rpart.plot")

library("nnet")

library("randomForest")

library("effects")

* Sourcing codes developed for BUS445 by Robert Krider, Dan Pulter and Ryan Tavakol

source("C:\\Users\\Rachel\\Downloads\\BCA\_functions\_source\_file20191031.R")

**Data Wrangling**

* Reading csv files

library(readr)

Retention2017 <- read\_csv("BADM Hackathon/Retention2017.csv")

view(Retention2017)

* Transforming date variables

Retention2017$lost <- as.factor(Retention2017$lost)

* Convert variables

Retention2017$created <- as.Date(Retention2017$created,format = "%d/%m/%Y")

* Counting missing values

variable.summary(Retention2017)

* Handle missing values – missing values can be treated differently depending on the situation. Consider why the following variables in various ways.

Retention2017 <- Retention2017[!is.na(Retention2017$firstorder),]

eclickrate - eclickrate can be NA if the customer never received emails or received emails that never had links.

Retention2017 <- mutate(Retention2017, NA\_eclickrate = ifelse(is.na(eclickrate) == 1,1,0))

Retention2017$eclickrate[Retention2017$NA\_eclickrate == 1] <- 0

Refill and doorstep - NA values occur in the same record. The customers could have never been offered these options

Retention2017 <- mutate(Retention2017, NA\_refill\_doorstep = ifelse(is.na(refill) == 1,1,0))

Retention2017$refill[Retention2017$NA\_refill\_doorstep == 1] <- "N"

Retention2017$doorstep[Retention2017$NA\_refill\_doorstep == 1] <- "N"

* Create new variables

Retention2017$cust\_period = as.numeric(Retention2017$lastorder - Retention2017$firstorder)

**Model Building**

* Running a logistic regression

Model1.RPart <- rpart(formula = lost~

eopenrate

+eclickrate

+avgorder

+ordfreq

+paperless

+refill

+doorstep

+favday

+city

+cust\_period,

data = filter(Retention2017, Sample == "Estimation"),

cp = 0.01,

model = TRUE)

plotcp(Model1.RPart)

printcp(Model1.RPart)

rpart.plot(Model1.RPart,

type = 0,

fallen.leaves = TRUE,

uniform = TRUE,

yes.text = "TRUE",

no.text = "FALSE",

cex = .8)

* Running a stepwise regression

Model3.StepReg <- step(Model2.LogReg, direction = "both")

summary(Model3.StepReg)

* Running a classification tree

Model2.LogReg <- glm(formula = lost ~

eopenrate

+eclickrate

+avgorder

+ordfreq

+paperless

+refill

+doorstep

+favday

+city

+cust\_period,

data = filter(Retention2017, Sample == "Estimation"),

family = binomial(logit))

summary(Model2.LogReg)

* Running a neural network

Model4.NeuNet <- Nnet(formula = lost ~

eopenrate

+eclickrate

+avgorder

+ordfreq

+paperless

+refill

+doorstep

+favday

+city

+cust\_period,

data = filter(Retention2017, Sample == "Estimation"),

decay = 0.10, # decay parameter

size = 2)

Model4.NeuNet$value

summary(Model4.NeuNet)

* Running a random forest

Model5.RanFor <- randomForest(formula = lost ~

eopenrate

+eclickrate

+avgorder

+ordfreq

+paperless

+refill

+doorstep

+favday

+city

+cust\_period,

data = filter(Retention2017, Sample == "Estimation"),

importance = TRUE,

ntree = 500, mtry = 4)

Model5.RanFor

importance(Model5.RanFor,type = 2)

varImpPlot(Model5.RanFor,type = 2, main = "Importance Plot")

**Model Assessment & Interpretation**

* Build a cumulative captured lift chart

lift.chart(modelList = c("Model1.RPart", "Model2.LogReg", "Model3.StepReg", "Model4.NeuNet","Model5.RanFor"),

data = filter(Retention2017, Sample == "Estimation"),

targLevel = "Y",

trueResp = 4749/(30729-7685),

type = "cumulative", sub = "Estimation")

* Build an incremental captured lift chart

lift.chart(modelList = c("Model4.NeuNet"),

data = as.data.frame(filter(Retention2017, Sample == "Validation")),

targLevel = "Y",

trueResp = 4749/(30729-7685),

type = "incremental", sub = "Validation")

* Effect Plot for logistic regression

plot(allEffects(Model2.LogReg),type = "response")

* Partial Dependence Plot for Random Forest

partialPlot(Model5.RanFor,

pred.data = as.data.frame(filter(Retention2017, Sample == "Validation")),

x.var = eopenrate,

sub = "Validation Set",

which.class = "Y")

**Scoring the prediction**

* Generate predicted probability

Retention2017$lost.Model1.RPart <- rawProbScore(model = "Model1.RPart",

data = Retention2017,

targLevel = "Y")

Submission.Model1.RPart <- Retention2017[Retention2017$Sample == "Holdout",c("custid","lost.Model1.RPart")]

names(Submission.Model1.RPart) <- c("custid", "score")

write.csv(Submission.Model1.RPart,"Submission.Model1.RPart.csv")

**Final Words**

Finding a model that has good predictive performance is only the starting point in this hackathon. Make sure you invest time on understanding the relations among the target variable and various predictors or even other metrics