############R scripts to Exploring the relationships between Retention Rate and Recency, Frequency, and Monetary

##########################################



where,

\* ${m}$ is a time period, e.g. the first month (${m}$=1), the second month (${m}$=2)

\* ${n} is the total number of periods the driver will stay before he/she finally churns

\* Prob(Active)m is the retention rate/possibility in month m

\* Margin is the profit (cash flow) the driver will contribute in the month m

\* WACC is the discount rate

Here we assume that WACC is constant in the formula

#Data Set

## potentially more indicator variables can be added to the LTV model

################################################################################

**setwd("/Users/qian/Dropbox")**

**read.csv("driver.csv")->df**

**as.Date(df$Date)->df$Date**

**str(df)**

'data.frame': 195594 obs. of 3 variables:

$ ID : Factor w/ 937 levels "002be0ffdc997bd5c50703158b7c2491",..: 1 1 1 1 1 1 1 1 1 1 ...

$ DATE : Date, format: "2016-03-29" "2016-03-29" ...

$ AMOUNT: num 5.68 6.68 6.79 9.12 9.79 ...

#################################################################################

getDataFrame <- function(df,startDate,endDate,tIDColName="ID",tDateColName="Date",tAmountColName="Amount"){

#order the dataframe by date descendingly

df <- df[order(df[,tDateColName],decreasing = TRUE),]

#remove the record before the start data and after the end Date

df <- df[df[,tDateColName]>= startDate,]

df <- df[df[,tDateColName]<= endDate,]

#remove the rows with the duplicated IDs, and assign the df to a new df.

newdf <- df[!duplicated(df[,tIDColName]),]

# caculate the Recency(days) to the endDate, the smaller days value means more recent

Recency<-as.numeric(difftime(endDate,newdf[,tDateColName],units="days"))

# add the Days column to the newdf data frame

newdf <-cbind(newdf,Recency)

#order the dataframe by ID to fit the return order of table() and tapply()

newdf <- newdf[order(newdf[,tIDColName]),]

# caculate the frequency

fre <- as.data.frame(table(df[,tIDColName]))

Frequency <- fre[,2]

newdf <- cbind(newdf,Frequency)

#caculate the Money per deal

m <- as.data.frame(tapply(df[,tAmountColName],df[,tIDColName],sum))

Monetary <- m[,1]/Frequency

newdf <- cbind(newdf,Monetary)

return(newdf)

} # end of function getDataFrame

require(plyr)

getPercentages <- function(df,colNames){

Var<-c(colNames,"Retain")

df<-df[,names(df) %in% Var,drop=F]

a <- ddply(df,Var,summarize,Number=length(Retain))

b <- ddply(a,

.(),

.fun=function(x){

transform(x, Percentage=with(x,round(ave(Number,a[,names(a) %in% Var,drop=F],FUN=sum)/ave(Number,a[,names(a) %in% colNames,drop=F],FUN=sum),2)))

})

b<-b[b$Retain==1,-1]

return(b)

}

getCLV<-function(r,f,m,n,cost,periods,dr,pModel){

df<-data.frame(period=c(0),r=c(r),f=c(f),n=c(n),value=c(0))

for(i in 1:periods){

backstep<-df[df$period==i-1,]

nrow<-nrow(backstep)

for(j in 1:nrow){

r<-backstep[j,]$r

f<-backstep[j,]$f

n<-backstep[j,]$n

p<-predict(pModel,data.frame(Recency=r,Frequency=f),type='response')[1]

Retains<-n\*p

df<-rbind(df,c(i,0,f+1,Retains,Retains\*(m-cost) / (1+dr)^i))

df<-rbind(df,c(i,r+1,f,n-Retains,(n-Retains)\*(-cost) / (1+dr)^i ))

}

}

return(sum(df$value))

}

# set the "forecast" transaction time scope

# set the "history" transaction time scope

**startDate\_forcast <- as.Date("20160528","%Y%m%d")**

**endDate\_forcast <- as.Date("20160627","%Y%m%d")**

**startDate\_history <- as.Date("20160328","%Y%m%d")**

**endDate\_history <- as.Date("20160527","%Y%m%d")**

#get the rolled up R,F,M data frames

**history <- getDataFrame(df,startDate\_history,endDate\_history)**

**forecast <- getDataFrame(df,startDate\_forcast,endDate\_forcast)**

#discrete the Monetary by $10 interval

breaks<-seq(0,round(max(history$monetary)+9),by=10)

history$monetary<-as.numeric(cut(history$monetary,breaks,labels=FALSE))

#add “retention” column to the RFM data frame

**Retain<-rep(0,nrow(history))**

**history<-cbind(history,Retain)**

# find out the those who retained in the forcast period

**history[history$id %in% forecast$id, ]$Retain<-1**

**train<-history**

**head(train)**

id date amount recency frequency monetary

1 002be0ffdc997bd5c50703158b7c2491 2016-03-29 5.680405 4 5.893617021 15516.89

2 002be0ffdc997bd5c50703158b7c2491 2016-03-29 6.684834 4 5.893617021 15516.89

3 002be0ffdc997bd5c50703158b7c2491 2016-03-29 6.790817 4 5.893617021 15516.89

4 002be0ffdc997bd5c50703158b7c2491 2016-03-29 9.118740 4 5.893617021 15516.89

5 002be0ffdc997bd5c50703158b7c2491 2016-03-29 9.791545 4 5.893617021 15516.89

6 002be0ffdc997bd5c50703158b7c2491 2016-03-29 11.254875 4 5.893617021 15516.89

DATE retain

1 2016-03-29 1

2 2016-03-29 1

3 2016-03-29 1

4 2016-03-29 1

5 2016-03-29 1

6 2016-03-29 1

# get "retain" percentages based on the variable Recency

**colNames<-c("Recency")**

**p<-getPercentages(train,colNames)**

**# get the retain ~ Recency model**

**r.glm=glm(Percentage~Recency,family=quasibinomial(link='logit'),data=p)**

**p\_r<-p**

# get "retain" percentages based on the variable Frequency

**colNames<-c("frequency")**

**p<-getPercentages(train,colNames)**

# get the retain ~ Frequency model

**f.glm=glm(Percentage~Frequency,family=quasibinomial(link='logit'),data=p)**

**p\_f<-p**



# get "retain" percentages based on the variable Monetary

**colNames<-c("monetary")**

**p<-getPercentages(train,colNames)**

# get the retain ~ Monetary model

**m.glm=glm(Percentage~monetary,family=quasibinomial(link='logit'),data=p)**

**p\_m<-p**

model<-glm(Retain~Recency+Frequency,family=quasibinomial(link='logit'),data=train)

pred<-predict(model,data.frame(Recency=c(0),Frequency=c(1)),type='response')

-----------------hive scripts for data preparation -----------------

set hive.exec.parallel=true;

set hive.exec.compress=true;

set hive.exec.compress.intermediate=true;

set mapred.compress.map.output=true;

set mapreduce.tasktracker.reduce.tasks.maximum=true;

set mapreduce.tasktracker.map.tasks.maximum=true;

set hive.optimize.index.filter=true;

set hive.support.concurrency=false;

set hive.groupby.orderby.position.alias = true;

-----------------------------------------------------

create table driver\_profile as

select a.\*

,b.rides

,b.avg\_ride\_distance

,b.avg\_ride\_duration

,b.prime\_ratio

from

(

select driver\_id, datediff(to\_date('2016-05-15'),driver\_onboard\_date) as tenure from driver\_ids

)a

inner join

(

select driver\_id

,count(distinct ride\_id) as rides --total rides

,avg(coalesce(ride\_distance,0)) as avg\_ride\_distance

,avg(coalesce(ride\_duration,0)) as avg\_ride\_duration

,sum(ride\_prime\_time)/sum(ride\_duration) as prime\_ratio

,sum(2+0.22\*(ride\_duration-ride\_prime\_time)/60+1.15\*ride\_distance\*0.000621371

+0.22\*1.5\*ride\_prime\_time/60) as total\_cash

from ride\_ids

group by 1

)b

on a.driver\_id = b.driver\_id

;

-----------driver cash flow----------------------------------

drop table cash\_flow;

create table cash\_flow as

select a.\*

,2+0.22\*(ride\_duration-ride\_prime\_time)/60+1.15\*ride\_distance\*0.000621371

+0.22\*1.5\*ride\_prime\_time/60+1.75

as total\_cash

,b.timestamp as dropped\_off\_ts

,to\_date(b.timestamp) as dropped\_off\_dt

from ride\_ids a left join ride\_timestamps b on a.ride\_id = b.ride\_id

--group by 1,2,3,4,5,6,7

;

drop table df;

create table df as

select a.driver\_id as id

,b.dropped\_off\_dt as date

,b.total\_cash as amount

,datediff('2016-06-27',last\_ride\_dt) as recency, a.frequency,c.monetary--RFM

from

(

select driver\_id,sum(rides)/sum(tenure) as frequency

from

driver\_profile

group by 1

)a

inner join

(

select driver\_id,dropped\_off\_dt,total\_cash

from cash\_flow

group by 1,2,3

)b on a.driver\_id = b.driver\_id

inner join

(

select driver\_id,max(dropped\_off\_dt) as last\_ride\_dt,sum(total\_cash) as monetary

from cash\_flow

group by 1

)c--recency

on a.driver\_id = c.driver\_id

group by 1,2,3,4,5,6

;

select \* from df

--where recency is not null and frequency is not null