Group Assignment 2

Team D

Hitika Hirani
Anna Rao Kataru
Ramana Sriwidya
Pravin Kumar Gunasekaran
Shreyas Bhargava

Introduction

Our Objective with this Project is to find if the customers of ISMS Durables will be retained during 2004 based on the customer purchase data that is available from 1998-2002. First we start our analysis by skimming through the database, analyzing the numerical variables , categorical variables and shortlisting which can be helpful in predicting whether the particular customer will be retained for the future periods or not.

PreProcessing : We find that the dates in our dataset need to be formatted into a singular standard format to be used in our prediction analysis and also we need to create a variable to extract the year out of the dates column.

Feature Engineering: Aggregating data for each household level to predict retention at household level. We are then creating new variables such as number of purchases, amount purchased, age, income, children, gender, male child, female child, etc. These variables can be used to see if they have any impact on the prediction of the retention at household level.

Then we need to address how we are going to resolve the problem of missing variables in the dataset. Since missing variables will lead to error in prediction of the retention value, we need to either remove the missing values or reassign missing values for each household for each column in the dataset. We chose to take mean values for numeric variables and Unknown values for categorical variables.

We are also creating new variables in the dataset which will help in measuring the Recency , Frequency and Monetary for the customer level . Then we split the dataset into training data and split data. In this case training data is 1998-2002 observations and test data is 2003-2004 observations.

Logistic regression

We have to predict whether the customers will be retained for each household based on the independent variables available in the dataset. Since it is either 0 or 1, True or False, it will be ideal to use a logistic regression model to predict the output.

We are using independent variables such as RFMscore, income, gender, last purchase years, total amount spent and total number of different brands bought by consumers. First we run the regression with a logistic model with all the variables. Then we use step aic function , which allows the model to select best variables among the selected variables depending on the AIC value. We then chose the model which has the least AIC value. So according to this regression model, the best independent variables for our model are RFMscore, income, last purchase year, number of categories and number of brands.

Then based on the variables we finalized we are validating the model using a validation dataset and predicting the retention values and creating a confusion matrix to confirm the performance of the model.

	Reference	
Prediction	0	1
0	2689	681
1	253	272

Accuracy : 0.7602
Kappa : 0.2351
Sensitivity : 0.28541
Specificity : 0.91400

The model has accuracy of 76.02% specificity of 91.4%. High sensitivity rate here defines that we are accurately predicting True Negative when compared to False Negatives. This is a very important measurement because in our prediction it is critical to predict which are customers who are not retained. We now have to see if our model is good enough for overall performance, which can be determined by measuring the AUC. For our validation dataset we are getting an AUC of 0.6844424 , which is good for this model since it is more than 50%.

Now we have to see if our model is good enough to predict accurately for the test dataset for year 2003. When we fit this model for the test data set we get the following confusion matrix.

	Reference	
Prediction	0	1
0	2696	710
1	203	285

Accuracy : 0.7655
Kappa : 0.2599
Sensitivity : 0.28643
Specificity : 0.92998

Once again our accuracy (76.55%), specificity(92.998%) and AUC 0.70474 are good indicators that our model is good in predicting the retention for our test data set for the year 2003.

Now we finally have to see if our model is good to predict the retention rate for each customer household for the year 2004.

	Reference	
Prediction	0	1
0	2887	519
1	279	209

Accuracy : 0.7951
Kappa : 0.2279
Sensitivity : 0.28709
Specificity : 0.91188

Once again our accuracy (79.51%), specificity(91.188%) and AUC 0.6694548 are good indicators that our model is good in predicting the retention for our test data set for the year 2004.

Random Forest

In a random forest we randomly select a predefined number of features as candidates. The latter will result in a larger variance between the trees which would otherwise contain the same features. As defined by the goal of the project the training and testing data were taken. The data as mentioned above was split in the ratio of **80:20** in test and train. This was done to measure accuracy.The number of decision trees in the forest is **150** and the number of features used as potential candidates for each split is 6. We have used a confusion matrix to evaluate the performance of our model. Values on the diagonal correspond to true positives and true negatives (correct predictions) whereas the others correspond to false positives and false negatives. The final prediction is made by taking the **majority** of the predictions made by each individual decision tree in the forest. The **accuracy** of the model is **65.55%**.The sensitivity and specificity is **0.6268** and **0.6648** respectively. The kappa is **0.2395.**

Further on we also did undersampling. Tried to reduce the observations from the majority class so that the final dataset is more balanced. We utilized the undersampling methods to our dataset and reran all the models to see if there's any improvement and validate our hypothesis on the influence of data imbalance. This method gave us **better results**.

```
Confusion Matrix and Statistics

          Reference
Prediction FALSE TRUE
     FALSE  1950  359
     TRUE    983  603

               Accuracy : 0.6555
                 95% CI : (0.6403, 0.6704)
    No Information Rate : 0.753
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2395

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.6268
            Specificity : 0.6648
         Pos Pred Value : 0.3802
         Neg Pred Value : 0.8445
             Prevalence : 0.2470
         Detection Rate : 0.1548
   Detection Prevalence : 0.4072
      Balanced Accuracy : 0.6458

       'Positive' Class : TRUE

>
```

The above values are for undersampling.We were able to increase our sensitivity by **0.4** and our accuracy by **0.1**, after undersampling.

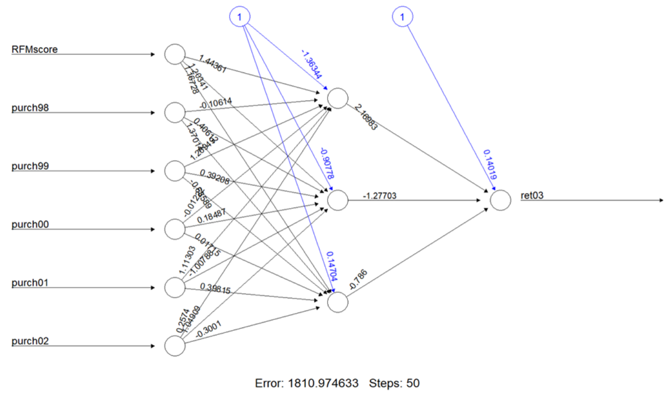
Neural Network Model

We used the neuralnet package in R to develop the neural network model for predicting the retention of customers for the electronic store. The data is split into 80% training and 20% testing and the training data is further split into 75% training and 25% validation samples. The model is trained on the training data and then the parameters are fine tuned using the validation data using cross validation. Once the model error is low enough, the testing data is used to predict the retention rate of the customers.

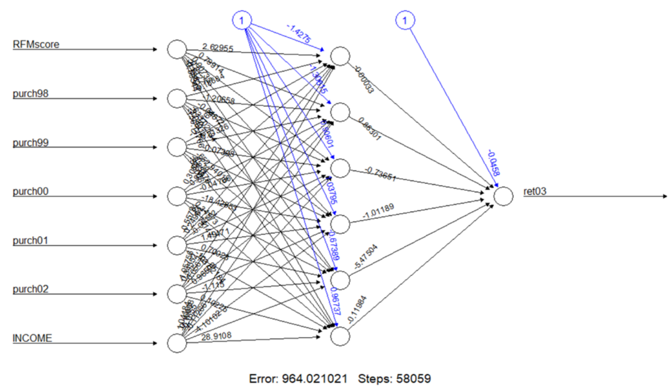
We are selecting independent variables such as rfmscore, purchase amount, income and gender etc. We can also have more categorical variables in the model to increase the complexity and accuracy of the model. But we have not incorporated it since it requires more feature engineering.

We tried with a different number of nodes in the hidden layer.

Hidden = 3



Hidden = 6



Hidden layer with 6 nodes gave a lower error at 964 than hidden layer with 3 nodes which gave 1810.

Unlike other regression models, where performance of the model can be measured by RMSE, MAE or AUC, we can measure the model performance only through the error value in the output of the neural network.

From the above two figures we can see that the error value is reduced when we increase the number of nodes in the hidden layer. To improve the model better we can add more variables by doing feature engineering and also by using a different activation function.

```
install.packages(c("BBmisc","purrr","C50","psych","neuralnet","dplyr","pdp","gmodels","randomForest","pacman","MASS","ROCR"))

# Load packagses, making sure they're installed first

library(BBmisc)# nice normalize function
library(purrr)
```

```
library(C50)
library(psych)
library(neuralnet)
library(dplyr)
library(pdp)
library(gmodels)
library(randomForest)
library(caret)
library(lubridate)
library(pacman)
library(MASS)
library(ROCR)
library(lift)
library(stringr)
library(nnet)
library(mltools)
library(data.table)

Transact.df<-read.csv("C:/Users/admin/Downloads/ISMSDataset.csv",stringsAsFactors=FALSE)

dim(Transact.df)
str(Transact.df)
summary(Transact.df)
View(describe(Transact.df))
View(Transact.df)
str(Transact.df$TRANSACTION_LOCATION)
summary(Transact.df$TRANSACTION_LOCATION)
summary(Transact.df$QUANTITY)

# Fix dates - Create year variable

Transact.df$TRANSACTION_DATE_2<-ymd(dmy_hms(Transact.df$TRANSACTION_DATE))
Transact.df$TRANSACTION_DATE_YR<-year(Transact.df$TRANSACTION_DATE_2)
Transact.df$TRANSACTION_DATE_month <-month(Transact.df$TRANSACTION_DATE_2)


hh_transact <- subset(Transact.df,TRANSACTION_DATE_YR<=2004) %>%
group_by(HOUSEHOLD_ID) %>%
  summarise(firstpurch = min(as.Date(TRANSACTION_DATE_2)),
        lastpurch02 =
max(as.Date(TRANSACTION_DATE_2[TRANSACTION_DATE_YR<=2002])), #most recent
purchase as of 12-31-2002
        purch98 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==1998]),
        purch99 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==1999]),
```

```
        purch00 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2000]),
        purch01 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2001]),
        purch02 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2002]),
        purch03 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2003]),
        purch04 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2004]),
        doll98 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==1998]),
        doll99 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==1999]),
        doll00 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2000]),
        doll01 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2001]),
        doll02 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2002]),
        AGE_H_HEAD = first(AGE_H_HEAD),
        CHILDERN_PRESENCE = first(CHILDERN_PRESENCE),
        INCOME = first(INCOME),
        GENDER_H_HEAD = first(GENDER_H_HEAD),
        GENDER_INDIVIDUAL = first(GENDER_INDIVIDUAL),
        MALE_CHID_AGE_0_2 = first(MALE_CHID_AGE_0_2),
        MALE_CHID_AGE_3_5 = first(MALE_CHID_AGE_3_5),
        MALE_CHID_AGE_6_10 = first(MALE_CHID_AGE_6_10),
        MALE_CHID_AGE_11_15 = first(MALE_CHID_AGE_11_15),
        MALE_CHID_AGE_16_17 = first(MALE_CHID_AGE_16_17),
        FEMALE_CHID_AGE_0_2 = first(FEMALE_CHID_AGE_0_2),
        FEMALE_CHID_AGE_3_5 = first(FEMALE_CHID_AGE_3_5),
        FEMALE_CHID_AGE_6_10 = first(FEMALE_CHID_AGE_6_10),
        FEMALE_CHID_AGE_11_15 = first(FEMALE_CHID_AGE_11_15),
        FEMALE_CHID_AGE_16_17 = first(FEMALE_CHID_AGE_16_17),
        UNKNOWN_CHID_AGE_0_2 = first(UNKNOWN_CHID_AGE_0_2),
        UNKNOWN_CHID_AGE_3_5 = first(UNKNOWN_CHID_AGE_3_5),
        UNKNOWN_CHID_AGE_6_10 = first(UNKNOWN_CHID_AGE_6_10),
        UNKNOWN_CHID_AGE_11_15 = first(UNKNOWN_CHID_AGE_11_15),
        UNKNOWN_CHID_AGE_16_17 = first(UNKNOWN_CHID_AGE_16_17),
        #TRANSACTION_DATE_month = first(TRANSACTION_DATE_month),
        #return = first(return),
        numcat = n_distinct(CATEGORY_DESCRIPTION[TRANSACTION_DATE_YR>=1998 &
TRANSACTION_DATE_YR<=2002]),
        numbrand =
n_distinct(TRANSACTION_TYPE_DESCRIPTION[TRANSACTION_DATE_YR>=1998 &
TRANSACTION_DATE_YR<=2002])
  )
View(hh_transact)

#return dummy

#Transact.df$return = ifelse((Transact.df$QUANTITY >=0),0,1)
```

```
#deal with nas

describe(hh_transact)
hh_transact$AGE_H_HEAD <- ifelse(is.na(hh_transact$AGE_H_HEAD),
mean(hh_transact$AGE_H_HEAD,na.rm=TRUE),hh_transact$AGE_H_HEAD)
hh_transact$INCOME <- ifelse(is.na(hh_transact$INCOME),
mean(hh_transact$INCOME,na.rm=TRUE),hh_transact$INCOME)

# New category for variables

hh_transact$CHILDERN_PRESENCE <-
as.factor(ifelse(hh_transact$CHILDERN_PRESENCE=="N" |
hh_transact$CHILDERN_PRESENCE=="Y",hh_transact$CHILDERN_PRESENCE,
"UNKNOWN"))
hh_transact$GENDER_H_HEAD <- as.factor(hh_transact$GENDER_H_HEAD)
hh_transact$GENDER_INDIVIDUAL <-
as.factor(ifelse(hh_transact$GENDER_INDIVIDUAL=="F" |
hh_transact$GENDER_INDIVIDUAL=="M" ,hh_transact$GENDER_INDIVIDUAL,
"UNKNOWN"))


# We want to get rid of those households that had their first purchase after 2002

hh_transact <- hh_transact[year(hh_transact$firstpurch)<=2002,]
```

 # RFM

```
# RFM

hh_transact$numpurch <- hh_transact$purch98 + hh_transact$purch99 + hh_transact$purch00
+ hh_transact$purch01 + hh_transact$purch02
hh_transact$dollpurch <- hh_transact$doll98 + hh_transact$doll99 + hh_transact$doll00 +
hh_transact$doll01 + hh_transact$doll02
#recency
hh_transact$recent <- as.numeric(as.Date("2002-12-31")-as.Date(hh_transact$lastpurch02))
#frequency
hh_transact$frequent <- hh_transact$numpurch/(as.numeric(as.Date("2002-12-31")-
hh_transact$firstpurch))*30
#monetary
```

```
hh_transact$monetary <- hh_transact$dollpurch/(as.numeric(as.Date("2002-12-31")-
hh_transact$firstpurch))*30

#quintile
hh_transact <- as.data.frame(hh_transact[order(hh_transact$recent),])
hh_transact$rix <- 1:nrow(hh_transact) # recency index
hh_transact$R_Q<-ceiling(hh_transact$rix/4000)

hh_transact <- as.data.frame(hh_transact[order(-hh_transact$frequent),])
hh_transact$fix <- 1:nrow(hh_transact) # frequency index
hh_transact$F_Q<-ceiling(hh_transact$fix/4000)

hh_transact <- as.data.frame(hh_transact[order(-hh_transact$monetary),])
hh_transact$mix <- 1:nrow(hh_transact) # monetary index
hh_transact$M_Q<-ceiling(hh_transact$mix/4000)

#RFM importance
z1 = aggregate(hh_transact$ret03, by=list(group = hh_transact$R_Q), mean)
z2 = aggregate(hh_transact$ret03, by=list(group = hh_transact$F_Q), mean)
z3 = aggregate(hh_transact$ret03, by=list(group = hh_transact$M_Q), mean)
par(mfrow=c(3,1))
barplot(z1$x, names.arg = z1$group,main = "R")
barplot(z2$x, names.arg = z2$group,main = "F")
barplot(z3$x, names.arg = z3$group,main = "M")

#RFM score
hh_transact$RFMscore <- hh_transact$R_Q*100+hh_transact$F_Q*10+hh_transact$M_Q



# The two basic "retention" measures: at least on purchase in 2003 ...

hh_transact$ret03 <- (hh_transact$purch03>0)

# ... at least one purchase in 2004

hh_transact$ret04 <- (hh_transact$purch04>0)



#try CLV

library(BTYDPlus)
customer_rdf <- BTYDplus::elog2cbs(
```

```
  data,
  unit = 'days',
  T.cal = max(data$date),
  T.tot = max(data$date)
)
customer_rdf$sales_avg = customer_rdf$sales / (customer_rdf$x + 1)

bgnbd_rdf <- customer_rdf
bgnbd_rdf$T.star <- 365

#spliting


set.seed(13343)
train_ind <- createDataPartition(y = hh_transact$ret03,p=.8,list = FALSE)
training <- hh_transact[train_ind,]
test <- hh_transact[-train_ind,]

# To check balance

prop.table(table(hh_transact$ret03))
prop.table(table(training$ret03))
prop.table(table(test$ret03))

# Divide training sample to create validation sample
set.seed(34331)
val_ind <- createDataPartition(y = training$ret03,p=.25,list = FALSE)
val <- training[val_ind,]
training <- training[-val_ind,]

prop.table(table(val$ret03))


#cross validation

#set.seed(125)

# defining training control
# as cross-validation and
# value of K equal to 10
#train_control <- trainControl(method = "cv",
                              #number = 10)

# training the model by assigning sales column
```

```
# as target variable and rest other column
# as independent variable
#model <- train(sales ~., data = marketing,
              #method = "lm",
              #trControl = train_control)



###############################
##  RANDOM FOREST ANALYSIS  ##
###############################

nondates<-c(4:8,11:37,40:44,46,48) # selects the columns that aren't dates
View(nondates)
rf_model <- randomForest(training[,c(4:8,11:37,40:44,46,48,49)],as.factor(training$ret03),ntree =
150,na.action = na.roughfix)
show(rf_model)
rf_class <- predict(rf_model,val[,c(4:8,11:37,40:44,46,48,49)],type="response")
rfconfmat <- confusionMatrix(rf_class, positive = "TRUE", as.factor(val$ret03))
rfconfmat

#################
# Undersampling #
#################

# Here we remove negative ("0")  observations to reduce their concentration and
#  again make them 50/50.

pos_corpus <- training[training$ret03==1,]
neg_corpus <- training[training$ret03==0,]

#create vector of neg_need2drop integers to drop the negs

neg_need2drop <- nrow(neg_corpus)-nrow(pos_corpus)

set.seed(98771)
for (i in 1:neg_need2drop){
  ix_drop <- floor(runif(1,min=1,max=(nrow(neg_corpus)+1)))
  neg_corpus <- neg_corpus[-ix_drop,]
}

us_training <- rbind(neg_corpus,pos_corpus)
```

```
rf_model <-
randomForest(us_training[,c(4:8,11:37,40:44,46,48,49)],as.factor(us_training$ret03),ntree =
150,na.action = na.roughfix)
show(rf_model)
rf_class <- predict(rf_model,val[,c(4:8,11:37,40:44,46,48,49)],type="response")
rfconfmat <- confusionMatrix(rf_class, positive = "TRUE", as.factor(val$ret03))
rfconfmat
```

After adding the variable:

```
install.packages(c("BBmisc","purrr","C50","psych","neuralnet","dplyr","pdp","gmodels","randomF
orest","pacman","MASS","ROCR"))

# Load packagses, making sure they're installed first

library(BBmisc)# nice normalize function
library(purrr)
library(C50)
library(psych)
library(neuralnet)
library(dplyr)
library(pdp)
library(gmodels)
library(randomForest)
library(caret)
library(lubridate)
library(pacman)
library(MASS)
library(ROCR)
library(lift)
library(stringr)
library(nnet)
library(mltools)
library(data.table)

Transact.df<-read.csv("C:/Users/admin/Downloads/ISMSDataset.csv",stringsAsFactors=FALSE)

dim(Transact.df)
str(Transact.df)
summary(Transact.df)
View(describe(Transact.df))
```

```
View(Transact.df)
str(Transact.df$TRANSACTION_LOCATION)
summary(Transact.df$TRANSACTION_LOCATION)
summary(Transact.df$QUANTITY)

# Fix dates - Create year variable

Transact.df$TRANSACTION_DATE_2<-ymd(dmy_hms(Transact.df$TRANSACTION_DATE))
Transact.df$TRANSACTION_DATE_YR<-year(Transact.df$TRANSACTION_DATE_2)
Transact.df$TRANSACTION_DATE_month <-month(Transact.df$TRANSACTION_DATE_2)


hh_transact <- subset(Transact.df,TRANSACTION_DATE_YR<=2004) %>%
group_by(HOUSEHOLD_ID) %>%
  summarise(firstpurch = min(as.Date(TRANSACTION_DATE_2)),
        lastpurch02 =
max(as.Date(TRANSACTION_DATE_2[TRANSACTION_DATE_YR<=2002])), #most recent
purchase as of 12-31-2002
        purch98 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==1998]),
        purch99 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==1999]),
        purch00 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2000]),
        purch01 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2001]),
        purch02 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2002]),
        purch03 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2003]),
        purch04 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2004]),
        doll98 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==1998]),
        doll99 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==1999]),
        doll00 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2000]),
        doll01 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2001]),
        doll02 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2002]),
        AGE_H_HEAD = first(AGE_H_HEAD),
        CHILDERN_PRESENCE = first(CHILDERN_PRESENCE),
        INCOME = first(INCOME),
        GENDER_H_HEAD = first(GENDER_H_HEAD),
        GENDER_INDIVIDUAL = first(GENDER_INDIVIDUAL),
        MALE_CHID_AGE_0_2 = first(MALE_CHID_AGE_0_2),
        MALE_CHID_AGE_3_5 = first(MALE_CHID_AGE_3_5),
        MALE_CHID_AGE_6_10 = first(MALE_CHID_AGE_6_10),
        MALE_CHID_AGE_11_15 = first(MALE_CHID_AGE_11_15),
        MALE_CHID_AGE_16_17 = first(MALE_CHID_AGE_16_17),
        FEMALE_CHID_AGE_0_2 = first(FEMALE_CHID_AGE_0_2),
        FEMALE_CHID_AGE_3_5 = first(FEMALE_CHID_AGE_3_5),
        FEMALE_CHID_AGE_6_10 = first(FEMALE_CHID_AGE_6_10),
        FEMALE_CHID_AGE_11_15 = first(FEMALE_CHID_AGE_11_15),
```

```
          FEMALE_CHID_AGE_16_17 = first(FEMALE_CHID_AGE_16_17),
          UNKNOWN_CHID_AGE_0_2 = first(UNKNOWN_CHID_AGE_0_2),
          UNKNOWN_CHID_AGE_3_5 = first(UNKNOWN_CHID_AGE_3_5),
          UNKNOWN_CHID_AGE_6_10 = first(UNKNOWN_CHID_AGE_6_10),
          UNKNOWN_CHID_AGE_11_15 = first(UNKNOWN_CHID_AGE_11_15),
          UNKNOWN_CHID_AGE_16_17 = first(UNKNOWN_CHID_AGE_16_17),
          return = first(return),
          numcat = n_distinct(CATEGORY_DESCRIPTION[TRANSACTION_DATE_YR>=1998 &
TRANSACTION_DATE_YR<=2002]),
          numbrand =
n_distinct(TRANSACTION_TYPE_DESCRIPTION[TRANSACTION_DATE_YR>=1998 &
TRANSACTION_DATE_YR<=2002])
  )
View(hh_transact)

#return dummy

Transact.df$return = as.factor(ifelse((Transact.df$QUANTITY >=0),0,1))


#deal with nas

describe(hh_transact)
hh_transact$AGE_H_HEAD <- ifelse(is.na(hh_transact$AGE_H_HEAD),
mean(hh_transact$AGE_H_HEAD,na.rm=TRUE),hh_transact$AGE_H_HEAD)
hh_transact$INCOME <- ifelse(is.na(hh_transact$INCOME),
mean(hh_transact$INCOME,na.rm=TRUE),hh_transact$INCOME)

# New category for variables

hh_transact$CHILDERN_PRESENCE <-
as.factor(ifelse(hh_transact$CHILDERN_PRESENCE=="N" |
hh_transact$CHILDERN_PRESENCE=="Y",hh_transact$CHILDERN_PRESENCE,
"UNKNOWN"))
hh_transact$GENDER_H_HEAD <- as.factor(hh_transact$GENDER_H_HEAD)
hh_transact$GENDER_INDIVIDUAL <-
as.factor(ifelse(hh_transact$GENDER_INDIVIDUAL=="F" |
hh_transact$GENDER_INDIVIDUAL=="M" ,hh_transact$GENDER_INDIVIDUAL,
"UNKNOWN"))


# We want to get rid of those households that had their first purchase after 2002

hh_transact <- hh_transact[year(hh_transact$firstpurch)<=2002,]
```

```
# RFM

hh_transact$numpurch <- hh_transact$purch98 + hh_transact$purch99 + hh_transact$purch00
+ hh_transact$purch01 + hh_transact$purch02
hh_transact$dollpurch <- hh_transact$doll98 + hh_transact$doll99 + hh_transact$doll00 +
hh_transact$doll01 + hh_transact$doll02
#recency
hh_transact$recent <- as.numeric(as.Date("2002-12-31")-as.Date(hh_transact$lastpurch02))
#frequency
hh_transact$frequent <- hh_transact$numpurch/(as.numeric(as.Date("2002-12-31")-
hh_transact$firstpurch))*30
#monetary
hh_transact$monetary <- hh_transact$dollpurch/(as.numeric(as.Date("2002-12-31")-
hh_transact$firstpurch))*30

#quintile
hh_transact <- as.data.frame(hh_transact[order(hh_transact$recent),])
hh_transact$rix <- 1:nrow(hh_transact) # recency index
hh_transact$R_Q<-ceiling(hh_transact$rix/4000)

hh_transact <- as.data.frame(hh_transact[order(-hh_transact$frequent),])
hh_transact$fix <- 1:nrow(hh_transact) # frequency index
hh_transact$F_Q<-ceiling(hh_transact$fix/4000)

hh_transact <- as.data.frame(hh_transact[order(-hh_transact$monetary),])
hh_transact$mix <- 1:nrow(hh_transact) # monetary index
hh_transact$M_Q<-ceiling(hh_transact$mix/4000)

#RFM importance
z1 = aggregate(hh_transact$ret03, by=list(group = hh_transact$R_Q), mean)
z2 = aggregate(hh_transact$ret03, by=list(group = hh_transact$F_Q), mean)
z3 = aggregate(hh_transact$ret03, by=list(group = hh_transact$M_Q), mean)
par(mfrow=c(3,1))
barplot(z1$x, names.arg = z1$group,main = "R")
barplot(z2$x, names.arg = z2$group,main = "F")
barplot(z3$x, names.arg = z3$group,main = "M")

#RFM score
hh_transact$RFMscore <- hh_transact$R_Q*100+hh_transact$F_Q*10+hh_transact$M_Q



# The two basic "retention" measures: at least on purchase in 2003 ...
```

```
hh_transact$ret03 <- (hh_transact$purch03>0)

# ... at least one purchase in 2004

hh_transact$ret04 <- (hh_transact$purch04>0)



#try CLV

library(BTYDPlus)
customer_rdf <- BTYDplus::elog2cbs(
  data,
  unit = 'days',
  T.cal = max(data$date),
  T.tot = max(data$date)
)
customer_rdf$sales_avg = customer_rdf$sales / (customer_rdf$x + 1)

bgnbd_rdf <- customer_rdf
bgnbd_rdf$T.star <- 365

#spliting


set.seed(13343)
train_ind <- createDataPartition(y = hh_transact$ret03,p=.8,list = FALSE)
training <- hh_transact[train_ind,]
test <- hh_transact[-train_ind,]

# To check balance

prop.table(table(hh_transact$ret03))
prop.table(table(training$ret03))
prop.table(table(test$ret03))

# Divide training sample to create validation sample
set.seed(34331)
val_ind <- createDataPartition(y = training$ret03,p=.25,list = FALSE)
val <- training[val_ind,]
training <- training[-val_ind,]

prop.table(table(val$ret03))
```

```
#cross validation

#set.seed(125)

# defining training control
# as cross-validation and
# value of K equal to 10
#train_control <- trainControl(method = "cv",
                        #number = 10)

# training the model by assigning sales column
# as target variable and rest other column
# as independent variable
#model <- train(sales ~., data = marketing,
            #method = "lm",
            #trControl = train_control)

################################
##  RANDOM FOREST ANALYSIS  ##
################################

nondates<-c(4:8,11:37,40:44,46,48) # selects the columns that aren't dates
View(nondates)
rf_model <- randomForest(training[,c(4:8,11:38,41:43,50)],as.factor(training$ret03),ntree =
150,na.action = na.roughfix)
show(rf_model)
rf_class <- predict(rf_model,val[,c(4:8,11:38,41:43,50)],type="response")
rfconfmat <- confusionMatrix(rf_class, positive = "TRUE", as.factor(val$ret03))
rfconfmat

#################
# Undersampling #
#################

# Here we remove negative ("0")  observations to reduce their concentration and
#  again make them 50/50.

pos_corpus <- training[training$ret03==1,]
neg_corpus <- training[training$ret03==0,]

#create vector of neg_need2drop integers to drop the negs
```

```
neg_need2drop <- nrow(neg_corpus)-nrow(pos_corpus)

set.seed(98771)
for (i in 1:neg_need2drop){
  ix_drop <- floor(runif(1,min=1,max=(nrow(neg_corpus)+1)))
  neg_corpus <- neg_corpus[-ix_drop,]
}

us_training <- rbind(neg_corpus,pos_corpus)

rf_model <- randomForest(us_training[,c(4:8,11:38,41:43,50)],as.factor(us_training$ret03),ntree
= 150,na.action = na.roughfix)
show(rf_model)
rf_class <- predict(rf_model,val[,c(4:8,11:38,41:43,50)],type="response")
rfconfmat <- confusionMatrix(rf_class, positive = "TRUE", as.factor(val$ret03))
rfconfmat


# For use with Retention/Churn Exercise

install.packages(c("BBmisc","purrr","C50","psych","neuralnet","dplyr","pdp","gmodels","randomF
orest","pacman","MASS","ROCR"))

# Load packagses, making sure they're installed first

library(BBmisc)# nice normalize function
library(purrr)
library(C50)
library(psych)
library(neuralnet)
library(dplyr)
library(pdp)
library(gmodels)
library(randomForest)
library(caret)
library(lubridate)
library(pacman)
library(MASS)
library(ROCR)
library(lift)
library(stringr)
library(nnet)
library(mltools)
library(data.table)
```

```
#Read in data file ** put in your path **

Transact.df<-read.csv("D:/4th Sem/Group Asst_2/ISMSDataset1.csv",stringsAsFactors=FALSE)

# Fix dates - Create year variable

Transact.df$TRANSACTION_DATE_2<-ymd(dmy_hms(Transact.df$TRANSACTION_DATE))
Transact.df$TRANSACTION_DATE_YR<-year(Transact.df$TRANSACTION_DATE_2)

#Aggregate to household level, create variables you want
# NB: These are just example. Lots of opportunity for feature
# engineering here.

# NOTE: This will throw a warning, but that's OK ...

hh_transact <- subset(Transact.df,TRANSACTION_DATE_YR<=2004) %>%
group_by(HOUSEHOLD_ID) %>%
  summarise(firstpurch = min(as.Date(TRANSACTION_DATE_2)),
          lastpurch02 =
max(as.Date(TRANSACTION_DATE_2[TRANSACTION_DATE_YR<=2002])), #most recent
purchase as of 12-31-2002
          purch98 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==1998]),
          purch99 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==1999]),
          purch00 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2000]),
          purch01 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2001]),
          purch02 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2002]),
          purch03 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2003]),
          purch04 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2004]),
          doll98 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==1998]),
          doll99 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==1999]),
          doll00 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2000]),
          doll01 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2001]),
          doll02 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2002]),
          AGE_H_HEAD = first(AGE_H_HEAD),
          CHILDERN_PRESENCE = first(CHILDERN_PRESENCE),
          INCOME = first(INCOME),
          GENDER_H_HEAD = first(GENDER_H_HEAD),
          GENDER_INDIVIDUAL = first(GENDER_INDIVIDUAL),
          MALE_CHID_AGE_0_2 = first(MALE_CHID_AGE_0_2),
          MALE_CHID_AGE_3_5 = first(MALE_CHID_AGE_3_5),
          MALE_CHID_AGE_6_10 = first(MALE_CHID_AGE_6_10),
          MALE_CHID_AGE_11_15 = first(MALE_CHID_AGE_11_15),
          MALE_CHID_AGE_16_17 = first(MALE_CHID_AGE_16_17),
          FEMALE_CHID_AGE_0_2 = first(FEMALE_CHID_AGE_0_2),
```

```
        FEMALE_CHID_AGE_3_5 = first(FEMALE_CHID_AGE_3_5),
        FEMALE_CHID_AGE_6_10 = first(FEMALE_CHID_AGE_6_10),
        FEMALE_CHID_AGE_11_15 = first(FEMALE_CHID_AGE_11_15),
        FEMALE_CHID_AGE_16_17 = first(FEMALE_CHID_AGE_16_17),
        UNKNOWN_CHID_AGE_0_2 = first(UNKNOWN_CHID_AGE_0_2),
        UNKNOWN_CHID_AGE_3_5 = first(UNKNOWN_CHID_AGE_3_5),
        UNKNOWN_CHID_AGE_6_10 = first(UNKNOWN_CHID_AGE_6_10),
        UNKNOWN_CHID_AGE_11_15 = first(UNKNOWN_CHID_AGE_11_15),
        UNKNOWN_CHID_AGE_16_17 = first(UNKNOWN_CHID_AGE_16_17),
        numcat = n_distinct(CATEGORY_DESCRIPTION[TRANSACTION_DATE_YR>=1998 &
TRANSACTION_DATE_YR<=2002]),
        numbrand =
n_distinct(TRANSACTION_TYPE_DESCRIPTION[TRANSACTION_DATE_YR>=1998 &
TRANSACTION_DATE_YR<=2002])
  )

# You will still need to deal with missings
#  For numerics, maybe just use means; for factors maybe "UNKNOWN"

# Below is one approach. There are many others.
# means for income and age

hh_transact$AGE_H_HEAD <- ifelse(is.na(hh_transact$AGE_H_HEAD),
mean(hh_transact$AGE_H_HEAD,na.rm=TRUE),hh_transact$AGE_H_HEAD)
hh_transact$INCOME <- ifelse(is.na(hh_transact$INCOME),
mean(hh_transact$INCOME,na.rm=TRUE),hh_transact$INCOME)

# New category for variables

hh_transact$CHILDERN_PRESENCE <-
as.factor(ifelse(hh_transact$CHILDERN_PRESENCE=="N" |
hh_transact$CHILDERN_PRESENCE=="Y",hh_transact$CHILDERN_PRESENCE,
"UNKNOWN"))
hh_transact$GENDER_H_HEAD <- as.factor(hh_transact$GENDER_H_HEAD)
hh_transact$GENDER_INDIVIDUAL <-
as.factor(ifelse(hh_transact$GENDER_INDIVIDUAL=="F" |
hh_transact$GENDER_INDIVIDUAL=="M" ,hh_transact$GENDER_INDIVIDUAL,
"UNKNOWN"))

# We want to get rid of those households that had their first purchase after 2002

hh_transact <- hh_transact[year(hh_transact$firstpurch)<=2002,]
```

```
# These will be useful in RFM calc

hh_transact$numpurch <- hh_transact$purch98 + hh_transact$purch99 + hh_transact$purch00
+ hh_transact$purch01 + hh_transact$purch02
hh_transact$dollpurch <- hh_transact$doll98 + hh_transact$doll99 + hh_transact$doll00 +
hh_transact$doll01 + hh_transact$doll02
hh_transact$recent <- as.numeric(as.Date("2002-12-31")-as.Date(hh_transact$lastpurch02))

# The two basic "retention" measures: at least one purchase in 2003 ...

hh_transact$ret03 <- (hh_transact$purch03>0)

# ... at least one purchase in 2004

hh_transact$ret04 <- (hh_transact$purch04>0)

hh_transact$ret03 <- ifelse(hh_transact$ret03=="TRUE",1,0)
hh_transact$ret04 <- ifelse(hh_transact$ret04=="TRUE",1,0)

##############################################################################
#
# RFM measures as of end of 2002

# Recency Measure is most recent purchase
# Frequency is purchase rate since acquired (purchases per )
# Monetary is dollar purchase rate

# Average purchase frequency from first purchase until end of 2002:

hh_transact$frequent <- hh_transact$numpurch/(as.numeric(as.Date("2002-12-31")-
hh_transact$firstpurch))*30

# Average purchase amount from first purchase until end of 2002:

hh_transact$monetary <- hh_transact$dollpurch/(as.numeric(as.Date("2002-12-31")-
hh_transact$firstpurch))*30

# Now put into "quintiles" - there are 19474 hh's, so lets do 4000 - 4000 - 4000 - 4000 - 3474

hh_transact <- as.data.frame(hh_transact[order(hh_transact$recent),])
hh_transact$rix <- 1:nrow(hh_transact) # recency index
hh_transact$R_Q<-ceiling(hh_transact$rix/4000)

hh_transact <- as.data.frame(hh_transact[order(-hh_transact$frequent),])
```

```
hh_transact$fix <- 1:nrow(hh_transact) # frequency index
hh_transact$F_Q<-ceiling(hh_transact$fix/4000)

hh_transact <- as.data.frame(hh_transact[order(-hh_transact$monetary),])
hh_transact$mix <- 1:nrow(hh_transact) # monetary index
hh_transact$M_Q<-ceiling(hh_transact$mix/4000)

# Create one RFM score

hh_transact$RFMscore <- hh_transact$R_Q*100+hh_transact$F_Q*10+hh_transact$M_Q

# createDataPartition is better than the Bernoulli based approach because it ensures the target
classes
# are balanced in each sample

#spliting

set.seed(13343)
train_ind <- createDataPartition(y = hh_transact$ret03,p=.8,list = FALSE)
training <- hh_transact[train_ind,]
test <- hh_transact[-train_ind,]

# To check balance

prop.table(table(hh_transact$ret03))
prop.table(table(training$ret03))
prop.table(table(test$ret03))

# Divide training sample to create validation sample
set.seed(34331)
val_ind <- createDataPartition(y = training$ret03,p=.25,list = FALSE)
val <- training[val_ind,]
training <- training[-val_ind,]

prop.table(table(val$ret03))


####################################
##  LOGISTIC REGRESSION ANAYSIS ##
####################################

hh_transact$ret03 <- as.factor(hh_transact$ret03)
hh_transact$ret04 <- as.factor(hh_transact$ret04)
hh_transact$AGE_H_HEAD <- as.factor(hh_transact$AGE_H_HEAD)
```

```
hh_transact$GENDER_H_HEAD <- as.factor(hh_transact$GENDER_H_HEAD)

#sapply(lapply(training, unique), length)
model_logit <- glm(ret03 ~ RFMscore + INCOME + GENDER_H_HEAD + lastpurch02 +
purch98 + purch99 + purch00 + purch01 + purch02 + doll98 + doll99 + doll00 + doll01 + doll02 +
numcat + numbrand,
                data = training , family = "binomial"(link = "logit"))
summary(model_logit)

# Obviously too many variables here so we'll do "step-wise" modeling that adds/subtracts
variables
# in an effort to find the "best" model according to some criterion. Here, it's the AIC with respect
# to the training sample. AIC is a measure of fit (like R-squared) with a penalty term to reduce
# complexity

summary(model_logit_step <- stepAIC(model_logit, direction = "both",trace = 1))

# The following is the "best" model

model_logit_step <- glm(ret03 ~ RFMscore + INCOME + lastpurch02 + purch00 +
                    numcat + numbrand,
                   data = training, family = "binomial"(link = "logit"))
summary(model_logit_step)

# Let's now see how well it predicts the validation sample
# (you may receive a warning --> don't worry)

logit_probs <- predict(model_logit_step,newdata = val, type = "response")
logit_class <- rep("1",nrow(val))
logit_class[logit_probs <.40] <- "0" # This is the Pr[Retained] in the data

# Confusion Matrix - Note that we need to define what a "positive" outcome is

logit_class <- as.factor(logit_class)
confusionMatrix(logit_class, positive = "1", as.factor(val$ret03))

# Other Assessment Tools

# ROC (Receiver Operating Characteristics) Curve - Shows the tradeoff between sensitivity and
specificity
# Looking for a steep slope into upper left

val_class <- val$ret03
```

```
logistic_ROC_prediction <- ROCR::prediction(logit_probs,val_class)
logit_ROC <- performance(logistic_ROC_prediction,"tpr","fpr") # true positive rate, false positive
rate
plot(logit_ROC)

# AUC (Area under the curve)
# AUC of a perfect classifier is 100%, a random guess is 50%)

AUC.tmp <- performance(logistic_ROC_prediction,"auc")
logit_AUC <- as.numeric(AUC.tmp@y.values)
logit_AUC

# Plot the "lift"

plotLift(logit_probs,val_class, cumulative = TRUE, n.buckets = 10)


## Testing on remaining data in ret03

logit_probs <- predict(model_logit_step,newdata = test, type = "response")
logit_class <- rep("1",nrow(test))
logit_class[logit_probs <.40] <- "0" # This is the Pr[Retained] in the data

# Confusion Matrix - Note that we need to define what a "positive" outcome is

logit_class <- as.factor(logit_class)
confusionMatrix(logit_class, positive = "1", as.factor(test$ret03))

# Other Assessment Tools

# ROC (Receiver Operating Characteristics) Curve - Shows the tradeoff between sensitivity and
specificity
# Looking for a steep slope into upper left

val_class <- test$ret03

logistic_ROC_prediction <- ROCR::prediction(logit_probs,val_class)
logit_ROC <- performance(logistic_ROC_prediction,"tpr","fpr") # true positive rate, false positive
rate
plot(logit_ROC)

# AUC (Area under the curve)
# AUC of a perfect classifier is 100%, a random guess is 50%)
```

```
AUC.tmp <- performance(logistic_ROC_prediction,"auc")
logit_AUC <- as.numeric(AUC.tmp@y.values)
logit_AUC

# Plot the "lift"

plotLift(logit_probs,val_class, cumulative = TRUE, n.buckets = 10)

## Testing on testing data ret04

logit_probs <- predict(model_logit_step,newdata =test , type = "response")
logit_class <- rep("1",nrow(test))
logit_class[logit_probs <.40] <- "0" # This is the Pr[Retained] in the data

#glm_probs = data.frame(probs = predict(model_logit_step, type="response"))
# Confusion Matrix - Note that we need to define what a "positive" outcome is

logit_class <- as.factor(logit_class)
confusionMatrix(logit_class, positive = "1", as.factor(test$ret04))

# Other Assessment Tools

# ROC (Receiver Operating Characteristics) Curve - Shows the tradeoff between sensitivity and
specificity
# Looking for a steep slope into upper left

val_class <- test$ret04

logistic_ROC_prediction <- ROCR::prediction(logit_probs,val_class)
logit_ROC <- performance(logistic_ROC_prediction,"tpr","fpr") # true positive rate, false positive
rate
plot(logit_ROC)

# AUC (Area under the curve)
# AUC of a perfect classifier is 100%, a random guess is 50%)

AUC.tmp <- performance(logistic_ROC_prediction,"auc")
logit_AUC <- as.numeric(AUC.tmp@y.values)
logit_AUC

# Plot the "lift"

plotLift(logit_probs,val_class, cumulative = TRUE, n.buckets = 10)
```

```
################################
##  NEURAL NETWORKS  ##
################################

library(BBmisc)
library(purrr)
library(C50)
library(psych)
library(neuralnet)
library(dplyr)
library(pdp)
library(gmodels)
library(randomForest)
library(lubridate)
library(caret)

#Read in data file ** put in your path **

Transact.df = read.csv("C:/Users/sriwi/OneDrive/Documents/MBA/Smith School of
Business/Semester 4/BUMK 747 CRM Analytics/Retention Churn Exercise/ISMSDataset1.csv")

# Fix dates - Create year variable

Transact.df$TRANSACTION_DATE_2<-ymd(dmy_hms(Transact.df$TRANSACTION_DATE))
Transact.df$TRANSACTION_DATE_YR<-year(Transact.df$TRANSACTION_DATE_2)

#Aggregate to household level, create variables you want
# NB: These are just example. Lots of opportunity for feature
# engineering here.

# NOTE: This will throw a warning, but that's OK ...

hh_transact <- subset(Transact.df,TRANSACTION_DATE_YR<=2004) %>%
group_by(HOUSEHOLD_ID) %>%
  summarise(firstpurch = min(as.Date(TRANSACTION_DATE_2)),
            lastpurch02 =
max(as.Date(TRANSACTION_DATE_2[TRANSACTION_DATE_YR<=2002])), #most recent
purchase as of 12-31-2002
            purch98 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==1998]),
            purch99 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==1999]),
            purch00 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2000]),
            purch01 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2001]),
            purch02 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2002]),
            purch03 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2003]),
```

```
      purch04 = length(TRANSACTION_NBR[TRANSACTION_DATE_YR==2004]),
      doll98 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==1998]),
      doll99 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==1999]),
      doll00 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2000]),
      doll01 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2001]),
      doll02 = sum(EXTENDED_PRICE[TRANSACTION_DATE_YR==2002]),
      AGE_H_HEAD = first(AGE_H_HEAD),
      CHILDERN_PRESENCE = first(CHILDERN_PRESENCE),
      INCOME = first(INCOME),
      GENDER_H_HEAD = first(GENDER_H_HEAD),
      GENDER_INDIVIDUAL = first(GENDER_INDIVIDUAL),
      MALE_CHID_AGE_0_2 = first(MALE_CHID_AGE_0_2),
      MALE_CHID_AGE_3_5 = first(MALE_CHID_AGE_3_5),
      MALE_CHID_AGE_6_10 = first(MALE_CHID_AGE_6_10),
      MALE_CHID_AGE_11_15 = first(MALE_CHID_AGE_11_15),
      MALE_CHID_AGE_16_17 = first(MALE_CHID_AGE_16_17),
      FEMALE_CHID_AGE_0_2 = first(FEMALE_CHID_AGE_0_2),
      FEMALE_CHID_AGE_3_5 = first(FEMALE_CHID_AGE_3_5),
      FEMALE_CHID_AGE_6_10 = first(FEMALE_CHID_AGE_6_10),
      FEMALE_CHID_AGE_11_15 = first(FEMALE_CHID_AGE_11_15),
      FEMALE_CHID_AGE_16_17 = first(FEMALE_CHID_AGE_16_17),
      UNKNOWN_CHID_AGE_0_2 = first(UNKNOWN_CHID_AGE_0_2),
      UNKNOWN_CHID_AGE_3_5 = first(UNKNOWN_CHID_AGE_3_5),
      UNKNOWN_CHID_AGE_6_10 = first(UNKNOWN_CHID_AGE_6_10),
      UNKNOWN_CHID_AGE_11_15 = first(UNKNOWN_CHID_AGE_11_15),
      UNKNOWN_CHID_AGE_16_17 = first(UNKNOWN_CHID_AGE_16_17),
      numcat = n_distinct(CATEGORY_DESCRIPTION[TRANSACTION_DATE_YR>=1998 &
TRANSACTION_DATE_YR<=2002]),
      numbrand =
n_distinct(TRANSACTION_TYPE_DESCRIPTION[TRANSACTION_DATE_YR>=1998 &
TRANSACTION_DATE_YR<=2002])
  )

# You will still need to deal with missings
#  For numerics, maybe just use means; for factors maybe "UNKNOWN"

# Below is one approach. There are many others.
# means for income and age

hh_transact$AGE_H_HEAD <- ifelse(is.na(hh_transact$AGE_H_HEAD),
mean(hh_transact$AGE_H_HEAD,na.rm=TRUE),hh_transact$AGE_H_HEAD)
hh_transact$INCOME <- ifelse(is.na(hh_transact$INCOME),
mean(hh_transact$INCOME,na.rm=TRUE),hh_transact$INCOME)
```

```
# New category for variables

hh_transact$CHILDERN_PRESENCE <-
as.factor(ifelse(hh_transact$CHILDERN_PRESENCE=="N" |
hh_transact$CHILDERN_PRESENCE=="Y",hh_transact$CHILDERN_PRESENCE,
"UNKNOWN"))
hh_transact$GENDER_H_HEAD <- as.factor(hh_transact$GENDER_H_HEAD)
hh_transact$GENDER_INDIVIDUAL <-
as.factor(ifelse(hh_transact$GENDER_INDIVIDUAL=="F" |
hh_transact$GENDER_INDIVIDUAL=="M" ,hh_transact$GENDER_INDIVIDUAL,
"UNKNOWN"))

# We want to get rid of those households that had their first purchase after 2002

hh_transact <- hh_transact[year(hh_transact$firstpurch)<=2002,]


# These will be useful in RFM calc

hh_transact$numpurch <- hh_transact$purch98 + hh_transact$purch99 + hh_transact$purch00
+ hh_transact$purch01 + hh_transact$purch02
hh_transact$dollpurch <- hh_transact$doll98 + hh_transact$doll99 + hh_transact$doll00 +
hh_transact$doll01 + hh_transact$doll02
hh_transact$recent <- as.numeric(as.Date("2002-12-31")-as.Date(hh_transact$lastpurch02))

# The two basic "retention" measures: at least on purchase in 2003 ...

hh_transact$ret03 <- (hh_transact$purch03>0)

# ... at least one purchase in 2004

hh_transact$ret04 <- (hh_transact$purch04>0)

##############################################################################
#
# RFM measures as of end of 2002

# Recency Measure is most recent purchase
# Frequency is purchase rate since acquired (purchases per )
# Monetary is dollar purchase rate

# Average purchase frequency from first purchase until end of 2002:
```

```
hh_transact$frequent <- hh_transact$numpurch/(as.numeric(as.Date("2002-12-31")-
hh_transact$firstpurch))*30

# Average purchase amount from first purchase until end of 2002:

hh_transact$monetary <- hh_transact$dollpurch/(as.numeric(as.Date("2002-12-31")-
hh_transact$firstpurch))*30

# Now put into "quintiles" - there are 19474 hh's, so lets do 4000 - 4000 - 4000 - 4000 - 3474

hh_transact <- as.data.frame(hh_transact[order(hh_transact$recent),])
hh_transact$rix <- 1:nrow(hh_transact) # recency index
hh_transact$R_Q<-ceiling(hh_transact$rix/4000)

hh_transact <- as.data.frame(hh_transact[order(-hh_transact$frequent),])
hh_transact$fix <- 1:nrow(hh_transact) # frequency index
hh_transact$F_Q<-ceiling(hh_transact$fix/4000)

hh_transact <- as.data.frame(hh_transact[order(-hh_transact$monetary),])
hh_transact$mix <- 1:nrow(hh_transact) # monetary index
hh_transact$M_Q<-ceiling(hh_transact$mix/4000)

# Create one RFM score

hh_transact$RFMscore <- hh_transact$R_Q*100+hh_transact$F_Q*10+hh_transact$M_Q

#Neural Network

#spliting


set.seed(13343)
train_ind <- createDataPartition(y = hh_transact$ret03,p=.8,list = FALSE)
training <- hh_transact[train_ind,]
test <- hh_transact[-train_ind,]

# To check balance

prop.table(table(hh_transact$ret03))
prop.table(table(training$ret03))
prop.table(table(test$ret03))

# Divide training sample to create validation sample
set.seed(34331)
```

```
val_ind <- createDataPartition(y = training$ret03,p=.25,list = FALSE)
val <- training[val_ind,]
training <- training[-val_ind,]

prop.table(table(val$ret03))


nnet = neuralnet(ret03~RFMscore+purch98+purch99+purch00+purch01+purch02+INCOME,
               training, hidden = 6, threshold = 0.01, linear.output = TRUE)
plot(nnet)

pred = predict(nnet,val)

nnet2 =
neuralnet(ret03~RFMscore+numcat+numbrand+numpurch+purch98+purch99+purch00+purch0
1+purch02+INCOME,
                training, hidden = 6, threshold = 0.01, linear.output = TRUE)
plot(nnet2)

pred2 = predict(nnet2,val)

#2004

nnet = neuralnet(ret04 ~ RFMscore+purch98+purch99+purch00+purch01+purch02+INCOME,
               training, hidden = 6, threshold = 0.01, linear.output = TRUE)
plot(nnet)

pred = predict(nnet,val)

nnet2 = neuralnet(ret04 ~
RFMscore+numcat+numbrand+numpurch+purch98+purch99+purch00+purch01+purch02+INC
OME,
                training, hidden = 6, threshold = 0.01, linear.output = TRUE)
plot(nnet2)

pred2 = predict(nnet2,val)
```