MSiA 401: Mini Project - Group 6

Zhitao Chen Alejandra Lelo de Larrea Ibarra Sharika Mahadevan
Wencheng Zhang Weiyan Zhou

Contents

1 Business situation

A German book company would like to have a machine learning model to predict how much a customer will spend if they are sent an offer based on previous purchase history.

2 Goal

Build a machine learning model to help the company select between 20% and 40% of the customers to receive an offer so that they maximize the amount spent by customer.

3 Scope

To achieve the goal, we will build several machine learning models to predict the percentage of respondents and the average amount of spending. We take two different approaches: i) estimate a logistic regression model("one step model" here after); and ii) estimate a two step model (estimate response with logistic model and then estimate conditional demand based on those that were predicted as potential buyers).

For the one step model (and first part of the two step model), we estimate a logistic regression with regularization using Ridge and Lasso, and we also try the step-wise model. As for the conditional demand, we estimate a linear regression with regularization using again Ridge, Lasso and Step-wise.

For every possible model we estimate different variants. First of all, our data has a class imbalance problem for the response variable, thus we estimate models using the unbalanced train set and a train set that has balanced classes using SMOTE. As for the regressors, we estimate models using the original regressors and also using a set of regressors that have been transformed (most of them taking logarithms). To evaluate our models, we compute a gains table for each of them and compare the different models using the LIFT against randomly selecting customers and the AUC as reference criteria.

4 Data

```
# Load libraries
library(PerformanceAnalytics)
library(tidyverse)
library(ggpubr)
library(DMwR)
library(glmnet)
library(caret)
library(pROC)
library(kableExtra)
```

On a certain date, the book company sent an offer to some of its customers and then observed the responses to this offer. The data in the *bookall.csv* file has one row for each of the 16,781 customers and includes 68 variables. Among the variables, we have the following predictors:

1. *target*: the amount spent by the customer in response to the offer (dependent variable).
2. *tof*: time since the first order
3. *recency*: time since most recent order look if its right skewed use log.
4. *fitem*: item frequency (number of previous items)
5. *ford*: order frequency (number of previous orders)
6. *m*: monetary, total amount spent on previous orders
7. *fk*: frequency of *items* from category k in the past with $K = 1, 3, 5, 6, 7, 8, 9, 10, 12, 14, 17, 19, 20, 21, 22, 23, 26, 27, 30, 31, 35, 36, 37, 38, 39, 40, 41, 44, 50$ and 99.
8. *mk*: monetary value spend on items from category k in the past (k as previously defined).

```
# Read data
book <- read.csv("./bookall.csv") %>%
  tibble() %>%
  select(-id) %>%
  rename(recency = r)
```

4.1 Missing values

```
which(is.na(book), arr.ind = TRUE)
```

```
##      row col
```

Our data set has no missing values.

4.2 Target variable

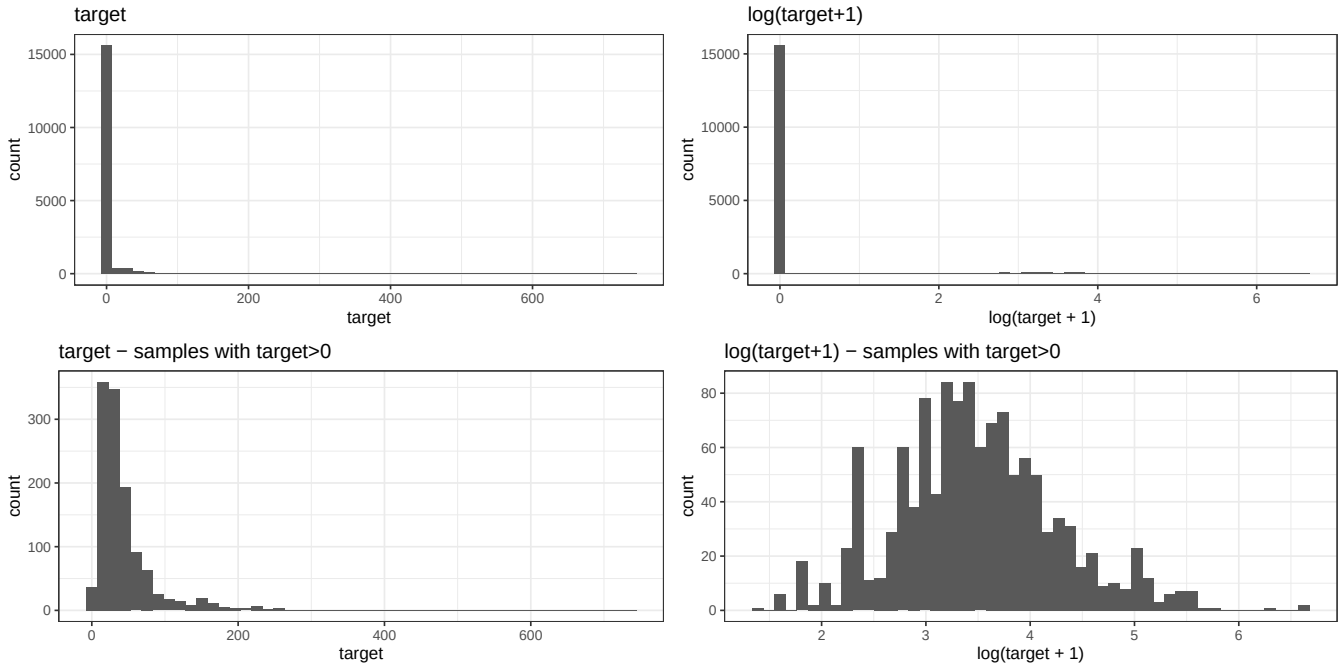
```
q1 <- ggplot(book,aes(x=target)) + theme_bw() +
  ggtitle("target")+
  geom_histogram(bins = 50)

q2 <- ggplot(book,aes(x=log(target+1))) + theme_bw() +
  ggtitle("log(target+1)") +
  geom_histogram(bins = 50)

q3 <- ggplot(filter(book, target !=0), aes(x=target)) + theme_bw() +
  ggtitle("target - samples with target>0") +
  geom_histogram(bins = 50)

q4 <- ggplot(filter(book, target !=0),aes(x=log(target+1))) + theme_bw() +
  ggtitle("log(target+1) - samples with target>0") +
  geom_histogram(bins = 50)

ggarrange(q1, q2, q3, q4, nrow = 2, ncol = 2)
```



Above are histograms for target, log(taget + 1), target with target samples greater than 0, and log(target+1) with target samples greater than 0. We can see from the histogram that target is right skewed which means that most of the customers do not spend money at all. So we use log transformation to reduce right skewedness and improve the

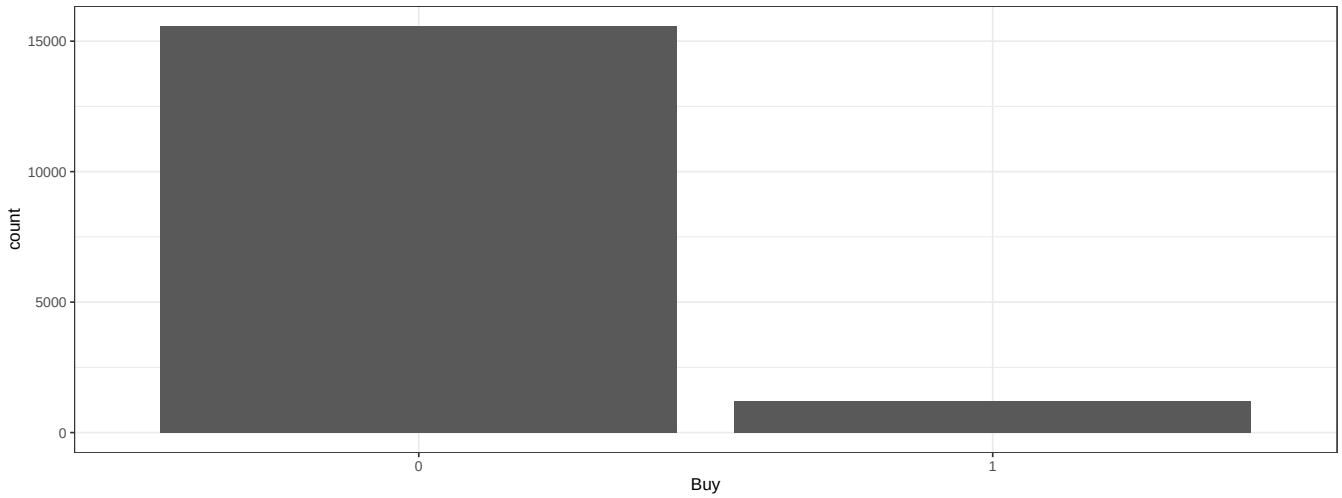
distribution to achieve a symmetry. We also pick the variable where target > 0, and we find that log(target+1) with target sample great than 0 has a normal distribution. Therefore, we will use the new variable 'log(target+1) with target sample greater than 0' in our model.

4.3 Target as binary variable

- Transform target to binary as $buy = 1$ if $target > 0$ and $buy = 0$ if $target = 0$.
- Then we encounter the Class imbalance problem where the number of $buy = 1$ far exceeds $buy = 0$
- Since the problem may affect estimations, we take action to address the class imbalance problem in the following steps.

```
# Create variable buy for dummy target
book <- book %>%
  mutate(buy = ifelse(target > 0, 1, 0))

# Histogram for binary target
ggplot(book, aes(x = as.factor(buy))) + theme_bw() +
  geom_histogram(stat = "count") +
  xlab("Buy")
```



By plotting the histogram for the binary variable 'buy', we can find that the amount of people who buy (1) and not buy (0) is highly imbalanced. This will to a great extent affect the result of our model. In this case, we may get a high accuracy just by predicting the majority class (not buy), but fail to capture the minority class (buy), which we focus on in this project. We will fix it by using SMOTe algorithm to address the class imbalance problem.

4.4 Feature engineering

In addition to the existing feature engineering, we also created several new variables: - itemPerOrder: item frequency/ order frequency - avOrderAmount: spent on previous orders/ order frequency - avItemAmount: spent on previous orders/ item frequency - purchaseRate: order frequency/ time since the first order - avOAk: average order amount in category k - n_cat: number of different categories bought in the past

```
# Number of different categories bought in the past
ncat = select(book, f1:f99) %>% apply(., MARGIN = 1, FUN = function(x){length(which(x>0))})

# Create new variables
book <- book %>%
  mutate(logTarget = log(target+1),
         itemPerOrder = fitem/ford,
         avOrderAmount = m/ford,
         avItemAmount = m/fitem,
         purchaseRate = ford/tof,
         avOA1 = ifelse(f1==0, 0, m1/f1),
         avOA3 = ifelse(f3==0, 0, m3/f3),
         avOA5 = ifelse(f5==0, 0, m5/f5),
```

```
         avOA6 = ifelse(f6==0, 0, m6/f6),
         avOA7 = ifelse(f7==0, 0, m7/f7),
         avOA8 = ifelse(f8==0, 0, m8/f8),
         avOA9 = ifelse(f9==0, 0, m9/f9),
         avOA10 = ifelse(f10==0, 0, m10/f10),
         avOA12= ifelse(f12==0, 0, m12/f12),
         avOA14 = ifelse(f14==0, 0, m14/f14),
         avOA17 = ifelse(f17==0, 0, m17/f17),
         avOA19 = ifelse(f19==0, 0, m19/f19),
         avOA20 = ifelse(f20==0, 0, m20/f20),
         avOA21 = ifelse(f21==0, 0, m21/f21),
         avOA22 = ifelse(f22==0, 0, m22/f22),
         avOA23 = ifelse(f23==0, 0, m23/f23),
         avOA26 = ifelse(f26==0, 0, m26/f26),
         avOA27 = ifelse(f27==0, 0, m27/f27),
         avOA30 = ifelse(f30==0, 0, m30/f30),
         avOA31 = ifelse(f31==0, 0, m31/f31),
         avOA35 = ifelse(f35==0, 0, m35/f35),
         avOA36 = ifelse(f36==0, 0, m36/f36),
         avOA37 = ifelse(f37==0, 0, m37/f37),
         avOA38 = ifelse(f38==0, 0, m38/f38),
         avOA39 = ifelse(f39==0, 0, m39/f39),
         avOA40 = ifelse(f40==0, 0, m40/f40),
         avOA41 = ifelse(f41==0, 0, m41/f41),
         avOA44 = ifelse(f44==0, 0, m44/f44),
         avOA50 = ifelse(f50==0, 0, m50/f50),
         avOA99 = ifelse(f99==0, 0, m99/f99)) %>%
  cbind(ncat) %>%
  as_tibble()
```

4.5 EDA Predictors

The correlation plots in the Appendix, show the scatter plots and correlation between the predictors and the target variable. We can observe the following:

- The class imbalance can be detected when plotting against target
- Most variables are skewed (e.g. recency), the observations are clustered or show heteroscedasticity (e.g. fitem and ford)
- fitem and ford (0.83), fitem and m (0.89), ford and m (0.78), avOrderAmount and avItemAmount (0.71) have comparatively high correlation.

4.6 Transformations

- After doing EDA of each regression (not included here) we decided to transform almost all the predictors by taking logarithms. The only variables that were not logged are time on file (tof) and number of categories bought (n_cat).

```
# Variables fk, mk, and avOAk to transform with log(x+1)
varsToLog <- c(grep("^[f|m][0-9]{1,2}$", x = names(book), value = TRUE),
               grep("^avOA[0-9]{1,2}$", x = names(book), value = TRUE))

# Transform variables
bookTransf <- book %>%
  mutate(logRecency = log(recency),
         logFitem = log(fitem),
         logFord = log(ford),
         logM = log(m+1),
         logItemPerOrder = log(itemPerOrder),
```

Table 1: Class balance in train set for "buy"

buy	Freq
0	92.58
1	7.42

```
         logAvOrderAmount = log(avOrderAmount+1),
         logAvItemAmount = log(avItemAmount+1),
         logPurchaseRate = log(purchaseRate)) %>%
  mutate(across(.cols = varsToLog,
                .funs = function(x){log(x+1)},
                .names = "log{.col}"))
```

4.7 Addressing class imbalance

We have a class imbalance problem: 92.58% of the observations correspond to “no buy” and only 7.42% to “buy”. To address this problem, we use the SMOTE algorithm to do oversampling of the “buy” class. SMOTE is a machine learning technique that solves problems that occur when using an imbalanced data. It is an algorithm that performs data augmentation by creating synthetic data points based on the original data points using nearest neighbors. The advantage of SMOTE is that we are not generating duplicates, but rather creating synthetic data points that are slightly different from the original data points. After using SMOTE we have a train set that is almost evenly balanced (49% “no buy”, 51% “buy”).

```
# Check original balance for target as binary in train set
originalBalance <- table(bookTransf %>% filter(train == 1) %>% select(buy))

knitr::kable(prop.table(originalBalance)*100,
             caption = "Class balance in train set for \"buy\"", digits = 2)
```

```
# --- Fix class imbalance with SMOTE ---
# set seed
set.seed(190322)
# Select percentage for oversampling the minority class
# NOTE: perc.over = (#No-#Yes)/#Yes*100
pover = round((max(originalBalance)-min(originalBalance))/min(originalBalance)*100) %>%
  as.numeric()

# New data with SMOTE from train
auxdata <- bookTransf %>%
  filter(train == 1) %>%
  mutate(buy = factor(buy, levels = c(1,0))) %>%
  as.data.frame()

newBook = SMOTE(buy~., auxdata,
                perc.over = pover,
                perc.under = 0)

# Bind original and new dataset
bookBalanced <- rbind(cbind(bookTransf,overSampled = 0),
                      cbind(newBook,overSampled = 1)) %>%
  as_tibble() %>%
  mutate(buy = as.numeric(buy))

# Check balance in merged df for train
newBalance <- table(bookBalanced %>% filter(train == 1) %>% select(buy))
knitr::kable(prop.table(newBalance), col.names = gsub("_", " ", names(originalBalance)),
             caption = "Class balance in train set after SMOTE for \"buy\"", digits = 2)
```

Table 2: Class balance in train set after SMOTE for "buy"

0	1
0	0.49
1	0.51

We check original balance for buy in train set, and we have freq of 92.58 of “not buy” compared to 7.42 of “buy”, which indicates that the original data set is highly imbalanced. Therefore, we use SMOTE to address the imbalance problem. After applying SMOTE technique, we check the balance in train set again. We get class balance of 0.49 for “buy” and 0.51 for “not buy”. In this way, we balanced the data set.

5 Modelling

5.1 Split data

We split the data in train and test according to “train” flag in the data provided. We created two versions of train set: with unbalanced classes and with balanced classes. Note thatthe test set is the original, i.e. no SMOTE was applied to the test tes.

```
# --- Train with unbalanced classes ---
# NOTE: Includes all variables transformed and untransformed.
# Do NOT include target, logTarget and buy in the same model.
trainUnbalanced <- bookBalanced %>%
  filter(train == 1) %>%
  filter(overSampled == 0 ) %>%
  select(-c(train,overSampled)) %>%
  select(target, logTarget, buy, everything())

# -- Train with balanced classes ---
# NOTE: Includes all variables transformed and untransformed.
# Do NOT include target, logTarget and buy in the same model.
trainBalanced <- bookBalanced %>%
  filter(train == 1) %>%
  select(-c(train,overSampled))

# --- Test ---
# NOTE: Includes all variables transformed and untransformed.
# Do NOT include target, logTarget and buy in the same model.
test <- bookBalanced%>%
  filter(train == 0) %>%
  select(-c(train,overSampled))
```

We also have 4 possible sets of variables: original (complete) with complete set of variables, original (selected/main) with selected variables, transformed (complete) with complete sets of variables, transformed(selected) with selected variables.

```
originalVars <- c("target","buy","tof","recency","fitem","ford","m",
                  "itemPerOrder","avOrderAmount","avItemAmount",
                  "purchaseRate","ncat",
                  "f1","f3","f5","f6","f7","f8","f9","f10","f12",
                  "f14","f17","f19","f20","f21","f22","f23","f26",
                  "f27","f30","f31","f35","f36","f37","f38","f39",
                  "f40","f41","f44","f50","f99",
                  "m1","m3","m5","m6","m7","m8","m9","m10","m12",
                  "m14","m17","m19","m20","m21","m22","m23","m26",
                  "m27","m30","m31","m35","m36","m37","m38","m39",
                  "m40","m41","m44","m50","m99",
                  "avOA1","avOA3","avOA5","avOA6","avOA7","avOA8","avOA9",
                  "avOA10","avOA12","avOA14","avOA17","avOA19","avOA20",
```

```
                  "avOA21","avOA22","avOA23","avOA26","avOA27","avOA30",
                  "avOA31","avOA35","avOA36","avOA37","avOA38","avOA39",
                  "avOA40","avOA41","avOA44","avOA50","avOA99")

mainOriginalVars <- c("target","buy","tof","recency","fitem","ford","m",
                      "itemPerOrder","avOrderAmount","avItemAmount",
                      "purchaseRate","ncat")

transformedVars <- c("logTarget","buy","tof","logRecency","logFitem","logFord",
                     "logM","logItemPerOrder","logAvOrderAmount","logAvItemAmount",
                     "logPurchaseRate","ncat",
                     "logf1","logf3","logf5","logf6","logf7","logf8","logf9",
                     "logf10","logf12","logf14","logf17","logf19","logf20",
                     "logf21","logf22","logf23","logf26","logf27","logf30",
                     "logf31","logf35","logf36","logf37","logf38","logf39",
                     "logf40","logf41","logf44","logf50","logf99",
                     "logm1","logm3","logm5","logm6","logm7","logm8","logm9",
                     "logm10","logm12","logm14","logm17","logm19","logm20",
                     "logm21","logm22","logm23","logm26","logm27","logm30",
                     "logm31","logm35","logm36","logm37","logm38","logm39",
                     "logm40","logm41","logm44","logm50","logm99",
                     "logavOA1","logavOA3","logavOA5","logavOA6","logavOA7",
                     "logavOA8","logavOA9","logavOA10","logavOA12","logavOA14",
                     "logavOA17","logavOA19","logavOA20","logavOA21","logavOA22",
                     "logavOA23","logavOA26","logavOA27","logavOA30","logavOA31",
                     "logavOA35","logavOA36","logavOA37","logavOA38","logavOA39",
                     "logavOA40","logavOA41","logavOA44","logavOA50","logavOA99")

mainTransformedVars <- c("logTarget","buy","tof","logRecency","logFitem","logFord",
                         "logM","logItemPerOrder","logAvOrderAmount","logAvItemAmount",
                         "logPurchaseRate","ncat")
```

5.2 Gains function

To evaluate our models, we take the gains function provided in the course packet to calculate the response and amounts on the different quantiles of the test set. In marketing world, people cares about money instead of model accuracy. We need to tell them how much they are saving from deploying the model we built. Therefore, we use gains tables for the second quantile (between 20 and 40 percent of the customers) to compare the model performance from the monetary aspect.

```
# Function to compute gains table
# Taken from course packet.
gains = function(yhat, respond, amt, ngrp=5){
  # Build dataframe with amount, response var (binary) and corresponding
  # quantiles (based on yhat)
  ans = data.frame(amt = amt,
                   respond = respond,
                   Quantile = cut(yhat,
                                  breaks=quantile(yhat, probs=seq(0,1, 1/ngrp)),
                                  labels=paste("Q",ngrp:1, sep=""),
                                  include.lowest = T)
  )

  # Group by quantile and compute counts, amounts and averages, cumulative counts,
  # amounts and averages
  ans <- ans %>%
    group_by(Quantile) %>%
    summarise(n = n(),
              Nrespond = sum(respond),
```

```
              amt = sum(amt),
              RespRate = Nrespond/n,
              AvgAmt = amt/n) %>%
    arrange(desc(Quantile)) %>%
    mutate(CumN = cumsum(n),
           CumResp = cumsum(Nrespond),
           CumAmt = cumsum(amt),
           CumRespRate = CumResp/CumN,
           CumAvgAmt = CumAmt/CumN)

  # calculate lifts
  ans <- ans %>%
    mutate(liftResp = CumRespRate/CumRespRate[nrow(ans)],
           liftAmt = CumAvgAmt/CumAvgAmt[nrow(ans)])

  return(ans)
}
```

5.3 One-step model

For the first approach, we estimate a logistic regression model to classify customers based on their tendency how will respond to the offer. To this end, and given the number of variables in the data set, we use regularization with Ridge and Lasso, and also a step-wise model selection. For each case, we try 4 different models varying the type of train set —unbalanced(U) or balanced (B)— and the type of variables —original (O) or transformed (T)—. We use the following name convention for our models: $Mi\_(trainType)(regMethod)(variableType)$ where $trainType = \{U, B\}$, $regMethod = \{R, L\}$ and $variableType = \{O, T\}$.

```
# Function to compute oneStep models with regularization (Ridge or Lasso)
oneStepModel <- function(train, test, test_buy, test_amt, alpha, varType = "transformed", k =5){

  # --- TRAINING ---
  # Model matrix to use glmnet: Balanced data with original variables
  if(varType == "transformed"){
    X_train = model.matrix(as.factor(buy) ~ .-1,
                           data = select(train,-logTarget))
  }else{
    X_train = model.matrix(as.factor(buy) ~ .-1,
                           data = select(train,-target))
  }

  # Estimate model: balanced data, ridge logistic regression,
  mod_cv <- cv.glmnet(X_train, as.factor(train$buy),
                      family = "binomial", type.measure = "deviance",
                      alpha = alpha, nfolds = k)

  # Pick lambda 1se (more conservative)
  mod = glmnet(X_train, as.factor(train$buy),
               family = "binomial",
               alpha = alpha, lambda = mod_cv$lambda.1se, nfolds = k)


  # --- TESTING ---

  # Model matrix for test
  if(varType == "transformed"){
    X_test = model.matrix(as.factor(buy) ~ .-1, data = select(test,-logTarget))
  }else{
    X_test = model.matrix(as.factor(buy) ~ .-1, data = select(test,-target))
  }
```

```
  # Predicted probs for test
  phat <- predict(mod, newx = X_test,
                  s = "lambda.1se", type = "response") %>%
    as.vector()

  # Gains table
  gains_tbl <- gains(phat, test_buy, test_amt)

  # Function output
  output = list(mod = mod,
                phat = phat,
                gains_tbl = gains_tbl)

  return(output)
}
```

5.3.1 Ridge Logistic Regression

For Ridge regression we use a selection of the predictors (*mainOriginalVars* listed above) to estimate the logistic regression.[1]

```
# M1: Unbalanced, Ridge, original variables.
m1_URO <- oneStepModel(train = select(trainUnbalanced, mainOriginalVars),
                       test = select(test, mainOriginalVars),
                       alpha = 0,
                       test_buy = test$buy,
                       test_amt = test$target,
                       varType = "original",
                       k = 5)
```

Table 3: Gains Table Model 1: unbalanced train set, Ridge logistic regression and main original variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	359	18608.51	0.16	8.29	2246	359	18608.51	0.16	8.29	2.23	2.55
Q2	2246	217	8189.58	0.10	3.65	4492	576	26798.09	0.13	5.97	1.79	1.83
Q3	2246	116	5212.00	0.05	2.32	6738	692	32010.09	0.10	4.75	1.43	1.46
Q4	2246	62	2438.55	0.03	1.09	8984	754	34448.64	0.08	3.83	1.17	1.18
Q5	2246	51	2103.67	0.02	0.94	11230	805	36552.31	0.07	3.25	1.00	1.00

From the table above, we can see that for the original variables with unbalanced training data set, the lift Response in the second quartile is 1.79 and the lift amount is 1.83, which means that compared to quantile 5 (no model or base case), the second quartile has 1.79 times average cumulative response rate and 1.83 average cumulative amount of revenue.

```
# M2: Unbalanced, Ridge, Transformed variables
m2_URT <- oneStepModel(train = select(trainUnbalanced, mainTransformedVars),
                       test = select(test, mainTransformedVars),
                       alpha = 0,
                       test_buy = test$buy,
                       test_amt = test$target,
```

[1]We also estimate the four combination of train set and type of variables for the entire set of predictors, but the performance measured by the AUC was lower than for the selected set of predictors. We only report the results for the selected set.

```
                        varType = "transformed",
                        k = 5)
```

Table 4: Gains Table Model 2: unbalanced train set, Ridge logistic regression and main transformed variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	401	19484.39	0.18	8.68	2246	401	19484.39	0.18	8.68	2.49	2.67
Q2	2246	182	7355.73	0.08	3.28	4492	583	26840.12	0.13	5.98	1.81	1.84
Q3	2246	110	5348.95	0.05	2.38	6738	693	32189.07	0.10	4.78	1.43	1.47
Q4	2246	62	2477.77	0.03	1.10	8984	755	34666.84	0.08	3.86	1.17	1.19
Q5	2246	50	1885.47	0.02	0.84	11230	805	36552.31	0.07	3.25	1.00	1.00

From the table above, we can see that for the original variables with unbalanced training data set, the lift Response in the second quartile is 1.81 and the lift amount is 1.84, which means that compared to quantile 5 (no model or base case), the second quartile has 1.81 times average cumulative response rate and 1.84 average cumulative amount of revenue.

```
# M3: balanced, ridge, original variables
m3_BRO <- oneStepModel(train = select(trainBalanced, mainOriginalVars),
                        test = select(test, mainOriginalVars),
                        alpha = 0,
                        test_buy = test$buy,
                        test_amt = test$target,
                        varType = "original",
                        k = 5)
```

Table 5: Gains Table Model 3: balanced train set, Ridge logistic regression and main original variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	345	18485.92	0.15	8.23	2246	345	18485.92	0.15	8.23	2.14	2.53
Q2	2246	228	7970.81	0.10	3.55	4492	573	26456.73	0.13	5.89	1.78	1.81
Q3	2246	125	5663.26	0.06	2.52	6738	698	32119.99	0.10	4.77	1.45	1.46
Q4	2246	58	1908.91	0.03	0.85	8984	756	34028.90	0.08	3.79	1.17	1.16
Q5	2246	49	2523.41	0.02	1.12	11230	805	36552.31	0.07	3.25	1.00	1.00

From the table above, we can see that for the original variables with unbalanced training data set, the lift Response in the second quartile is 1.78 and the lift amount is 1.81, which means that compared to quantile 5 (no model or base case), the second quartile has 1.78 times average cumulative response rate and 1.81 average cumulative amount of revenue.

```
# Balanced, Ridge, Transformed variables
m4_BRT <- oneStepModel(train = select(trainBalanced, mainTransformedVars),
                        test = select(test, mainTransformedVars),
                        alpha = 0,
                        test_buy = test$buy,
                        test_amt = test$target,
                        varType = "transformed",
                        k = 5)
```

Table 6: Gains Table Model 4: balanced train set, Ridge logistic regression and main transformed variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	373	17730.36	0.17	7.89	2246	373	17730.36	0.17	7.89	2.32	2.43
Q2	2246	176	6789.71	0.08	3.02	4492	549	24520.07	0.12	5.46	1.70	1.68
Q3	2246	126	5275.88	0.06	2.35	6738	675	29795.95	0.10	4.42	1.40	1.36
Q4	2246	75	4215.08	0.03	1.88	8984	750	34011.03	0.08	3.79	1.16	1.16
Q5	2246	55	2541.28	0.02	1.13	11230	805	36552.31	0.07	3.25	1.00	1.00

From the table above, we can see that for the original variables with unbalanced training data set, the lift Response in the second quartile is 1.70 and the lift amount is 1.68, which means that compared to quantile 5 (no model or base case), the second quartile has 1.70 times average cumulative response rate and 1.68 average cumulative amount of revenue.

5.3.2 Ridge Logistic Models Comparison

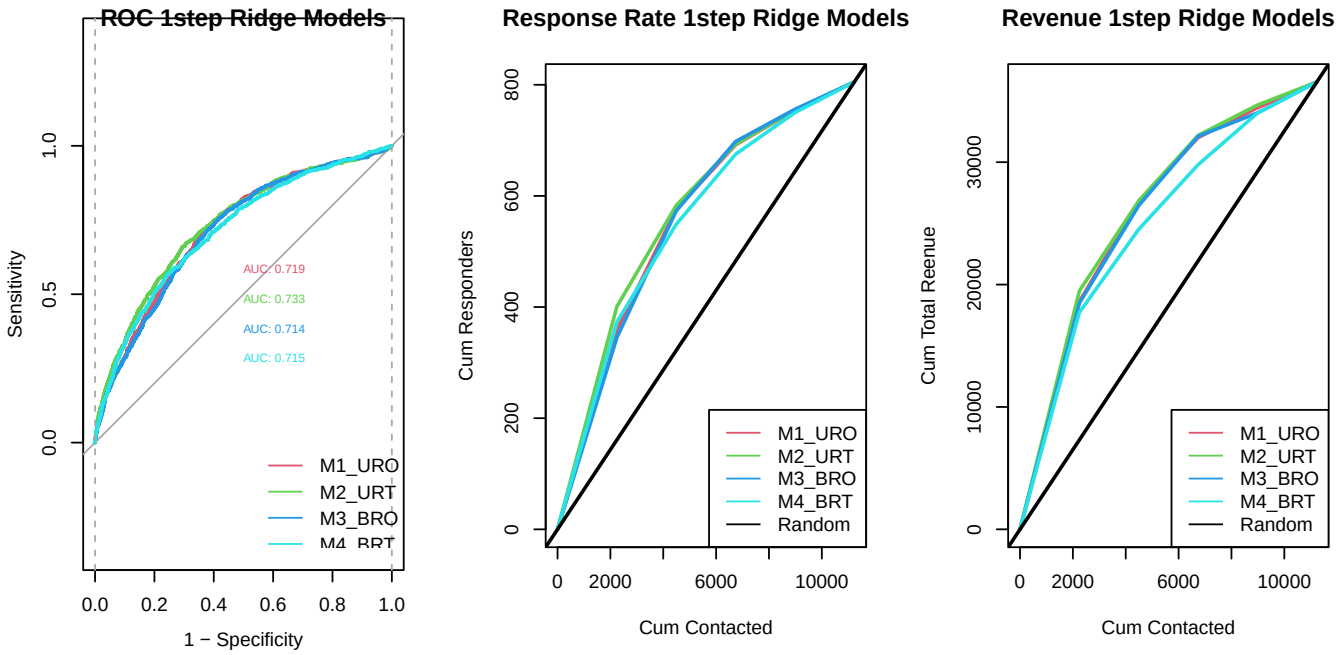
We compare the 4 different models under Ridge logistic regression using the gains table for second quantiles (20% or 40% or customers) and the AUC criteria.

From table 7, we find that M2_URT has the best performance among 4 models because it has the highest value of lift response & lift Amount for the Q1 as well as Q2.

In addition to that, from the ROC curve, we can see that M2_URT also has the largest AUC (0.733). Therefore, based on the result of gains table as well as the ROC curve, the best model is M2_URT, i.e. unbalanced train sample with transformed variable.

Table 7: Summary for Ridge Models.

Model	Lift Resp Q1	Lift Amt Q1	Lift Resp Q2	Lift Amt Q2
M1_URO	2.23	2.55	1.79	1.83
M2_URT	2.49	2.67	1.81	1.84
M3_BRO	2.14	2.53	1.78	1.81
M4_BRT	2.32	2.43	1.70	1.68



The best model is M2_URT i.e. unbalanced train sample and transformed variable. The estimated model is:

$$
\begin{aligned}
log\left(\frac{\pi_{buy}}{1-\pi_{buy}}\right) &= -3.3 - 0.02tof - 0.14log(recency) + 0.09log(fitem) + 0.12log(ford) + 0.07log(m) \\
&= +0.02log(itemPerOrder) - 0.02LOG(avOrderAmount) - 0.06log(avItemAmount) \\
&= +0.19log(purchaseRate) + 0.02ncat
\end{aligned}
$$

5.3.3 Lasso Logistic Regression

For Lasso logistic regression we use the complete set of the predictors (*originalVars* or *tranformedVars* as corresponds) to estimate the logistic regression and let the model select the most important variables.

```
# M5: Unbalanced, Lasso original variables
m5_ULO <- oneStepModel(train = select(trainUnbalanced, originalVars),
                       test = select(test, originalVars),
                       alpha = 1,
                       test_buy = test$buy,
                       test_amt = test$target,
                       varType = "original",
                       k = 5)
```

Table 8: Gains Table Model 5: unbalanced train set, Lasso logistic regression and original variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	355	16813.43	0.16	7.49	2246	355	16813.43	0.16	7.49	2.20	2.30
Q2	2246	229	9965.59	0.10	4.44	4492	584	26779.02	0.13	5.96	1.81	1.83
Q3	2246	117	5766.53	0.05	2.57	6738	701	32545.55	0.10	4.83	1.45	1.48
Q4	2246	52	1837.75	0.02	0.82	8984	753	34383.30	0.08	3.83	1.17	1.18
Q5	2246	52	2169.01	0.02	0.97	11230	805	36552.31	0.07	3.25	1.00	1.00

From table 8, we can see that for Lasso Logistic Regression using the Original variables with unbalanced training data set, the lift Response in the second quartile is 1.81 and the lift amount is 1.83, which means that compared to

quantile 5 (no model or base case), the second quartile has 1.81 times average cumulative response rate and 1.83 average cumulative amount of revenue.

```
# M6: Unbalanced, Lasso, transformed variables
m6_ULT <- oneStepModel(train = select(trainUnbalanced, transformedVars),
                       test = select(test, transformedVars),
                       alpha = 1,
                       test_buy = test$buy,
                       test_amt = test$target,
                       varType = "transformed",
                       k = 5)
```

Table 9: Gains Table Model 6: unbalanced train set, Lasso logistic regression and transformed variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	403	19143.90	0.18	8.52	2246	403	19143.90	0.18	8.52	2.50	2.62
Q2	2246	182	6863.10	0.08	3.06	4492	585	26007.00	0.13	5.79	1.82	1.78
Q3	2246	106	5279.93	0.05	2.35	6738	691	31286.93	0.10	4.64	1.43	1.43
Q4	2246	58	3018.70	0.03	1.34	8984	749	34305.63	0.08	3.82	1.16	1.17
Q5	2246	56	2246.68	0.02	1.00	11230	805	36552.31	0.07	3.25	1.00	1.00

From table 9, we can see that for Lasso Logistic Regression using the Original variables with unbalanced training data set, the lift Response in the second quartile is 1.82 and the lift amount is 1.78, which means that compared to quantile 5 (no model or base case), the second quartile has 1.82 times average cumulative response rate and 1.78 average cumulative amount of revenue.

```
# M7: Balanced, Lasso, original variables
m7_BLO <- oneStepModel(train = select(trainBalanced, originalVars),
                       test = select(test, originalVars),
                       alpha = 1,
                       test_buy = test$buy,
                       test_amt = test$target,
                       varType = "original",
                       k = 5)
```

Table 10: Gains Table Model 7: balanced train set, Lasso logistic regression and original variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	314	15262.14	0.14	6.80	2246	314	15262.14	0.14	6.80	1.95	2.09
Q2	2246	148	4998.73	0.07	2.23	4492	462	20260.87	0.10	4.51	1.43	1.39
Q3	2246	144	5850.96	0.06	2.61	6738	606	26111.83	0.09	3.88	1.25	1.19
Q4	2246	104	4202.04	0.05	1.87	8984	710	30313.87	0.08	3.37	1.10	1.04
Q5	2246	95	6238.44	0.04	2.78	11230	805	36552.31	0.07	3.25	1.00	1.00

From table 10, we can see that for Lasso Logistic Regression using the Original variables with unbalanced training data set, the lift Response in the second quartile is 1.43 and the lift amount is 1.39, which means that compared to quantile 5 (no model or base case), the second quartile has 1.43 times average cumulative response rate and 1.39 average cumulative amount of revenue.

```
# M8: balanced, Lasso, transformed variables
m8_BLT <- oneStepModel(train = select(trainBalanced, transformedVars),
                       test = select(test, transformedVars),
                       alpha = 1,
                       test_buy = test$buy,
                       test_amt = test$target,
                       varType = "transformed",
                       k = 5)
```

Table 11: Gains Table Model 8: balanced train set, Lasso logistic regression and transformed variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	293	13693.33	0.13	6.10	2246	293	13693.33	0.13	6.10	1.82	1.87
Q2	2246	163	6376.28	0.07	2.84	4492	456	20069.61	0.10	4.47	1.42	1.37
Q3	2246	150	5583.40	0.07	2.49	6738	606	25653.01	0.09	3.81	1.25	1.17
Q4	2246	98	4802.45	0.04	2.14	8984	704	30455.46	0.08	3.39	1.09	1.04
Q5	2246	101	6096.85	0.04	2.71	11230	805	36552.31	0.07	3.25	1.00	1.00

From table 11, we can see that for Lasso Logistic Regression using the Original variables with unbalanced training data set, the lift Response in the second quartile is 1.42 and the lift amount is 1.37, which means that compared to quantile 5 (no model or base case), the second quartile has 1.42 times average cumulative response rate and 1.37 average cumulative amount of revenue.

5.3.4 Lasso Logistic Models Comparison

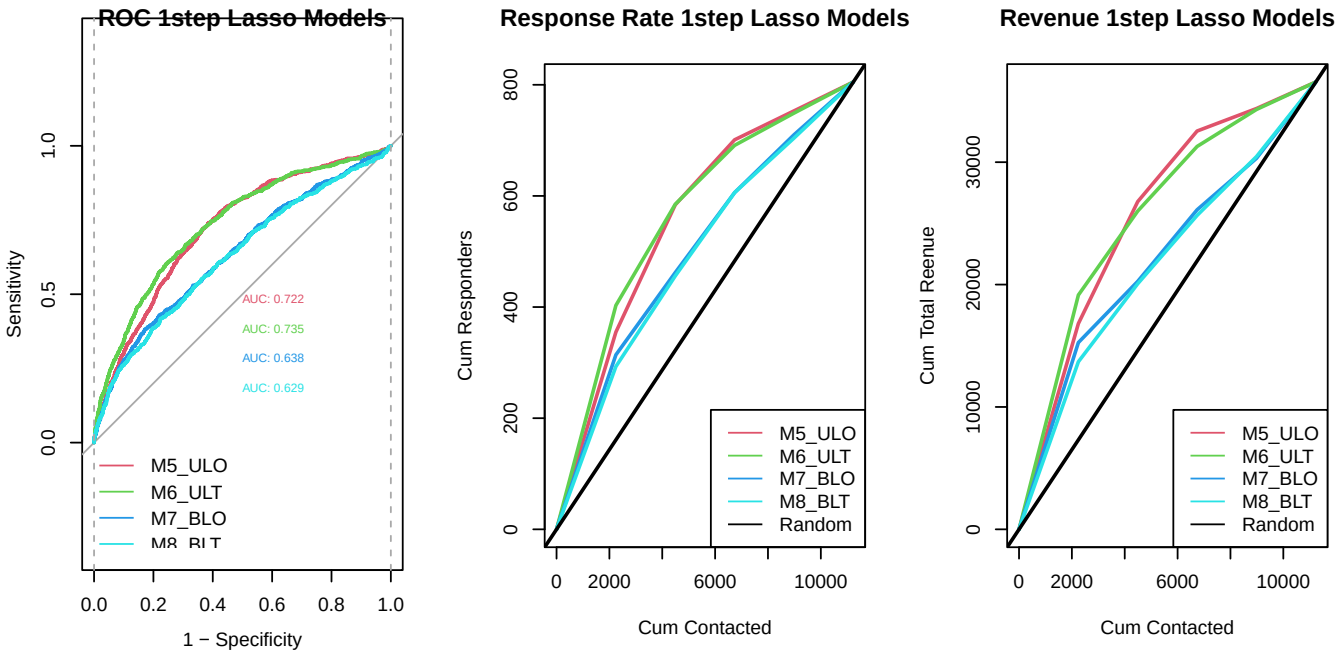
We compare the 4 different models under Lasso logistic regression using the gains table for the first and second quartiles (20% or 40% or customers) and the AUC criteria.

From table 12, we find that M6_ULT has the best performance among 4 models because it has the highest value of lift response & lift Amount for the Q1 as well as Q2.

In addition to that, from the ROC curve, we can see that M6_ULT also has the largest AUC compared with other models. Therefore, based on the result of gains table as well as the ROC curve, the best model is M6_ULT, i.e. unbalanced train sample with transformed variable.

Table 12: Summary for Lasso Models.

Model	Lift Resp Q1	Lift Amt Q1	Lift Resp Q2	Lift Amt Q2
M5_ULO	2.20	2.30	1.81	1.83
M6_ULT	2.50	2.62	1.82	1.78
M7_BLO	1.95	2.09	1.43	1.39
M8_BLT	1.82	1.87	1.42	1.37



The best model is M6_ULT i.e. unbalanced train sample and transformed variable (although M5_ULO has slightly less lift response and higher lift amount for the second quantile). The estimated model is:

$$log\left(\frac{\pi_{buy}}{1-\pi_{buy}}\right) = -2.98 - 0.13log(recency) + 0.04log(fitem) + 0.11log(ford) + 0.21log(purchaseRate)$$

5.3.5 Stepwise Logistic Regression

For Stepwise logistic regression we use the complete set of the predictors (originalVars or tranformedVars as corresponds) without f_k's, m_k's and $avOAk$'s (refer to footnote) to estimate the logistic regression and let the model select the most important variables.[2]

```
# Function to compute oneStep models with stepwise model selection
# With highly correlated predictors, stepwise selection will almost
# certainly lead to highly varying choices of predictors from fold to fold.
oneStepwiseModel <- function(train, test, test_buy, test_amt, varType = "transformed"){

  # Model matrix to use step: Balanced data with original variables
  if(varType == "transformed"){
    X_train = train %>% select(-logTarget)
  }else{
    X_train = train %>% select(-target)
  }

  # Estimate model: balanced data, logistic regression with all variables
  mod_logit <- glm(as.factor(train$buy)~., family = binomial,
                   data = X_train )

  # Perform stepwise function
  mod = step(mod_logit, trace = FALSE)

  # Model matrix for test
  if(varType == "transformed"){
```

[2]We also estimate the four combinations of train set and type of variables for the entire set of predictors, but stepwise only relies on AIC for model selection, hence even if multicollinearity exists, it might still include those variables. Initially runs including f_k's, m_k's and $avOAk$'s resulted in models that were not intuitive. Hence, based on results from Ridge and Lasso, Additionally, given that the f_k's, m_k's and $avOAk$'s were found to be not significant with Ridge and Lasso, we just picked the main set of predictors as departure point for the stepwise backward selection model.

```
    X_test = test %>% select(-logTarget, buy)
  }else{
    X_test = test %>% select(-target, buy)
  }

  # Predicted probs for test
  phat <- predict(mod, X_test, type = "response")

  # Gains table
  gains_tbl <- gains(phat, test_buy, test_amt)

  output = list(mod = mod,
                phat = phat,
                gains_tbl = gains_tbl)

  return(output)
}
```

```
# M1: Unbalanced, Step, original variables.
m9_USO <- oneStepwiseModel(train = select(trainUnbalanced, mainOriginalVars),
                           test = select(test, mainOriginalVars),
                           test_buy = test$buy,
                           test_amt = test$target,
                           varType = "original")
```

Table 13: Gains Table Model 9: unbalanced train set, Stepwise logistic regression and original variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	388	18542.46	0.17	8.26	2246	388	18542.46	0.17	8.26	2.41	2.54
Q2	2246	198	7852.57	0.09	3.50	4492	586	26395.03	0.13	5.88	1.82	1.81
Q3	2246	108	6189.28	0.05	2.76	6738	694	32584.31	0.10	4.84	1.44	1.49
Q4	2246	60	1809.99	0.03	0.81	8984	754	34394.30	0.08	3.83	1.17	1.18
Q5	2246	51	2158.01	0.02	0.96	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M1: Unbalanced, Step, original variables.
m10_UST <- oneStepwiseModel(train = select(trainUnbalanced, mainTransformedVars),
                            test = select(test, mainTransformedVars),
                            test_buy = test$buy,
                            test_amt = test$target,
                            varType = "transformed")
```

Table 14: Gains Table Model 10: unbalanced train set, Stepwise logistic regression and transformed variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	414	19492.72	0.18	8.68	2246	414	19492.72	0.18	8.68	2.57	2.67
Q2	2246	174	6622.69	0.08	2.95	4492	588	26115.41	0.13	5.81	1.83	1.79
Q3	2246	111	5399.12	0.05	2.40	6738	699	31514.53	0.10	4.68	1.45	1.44
Q4	2246	49	2821.10	0.02	1.26	8984	748	34335.63	0.08	3.82	1.16	1.17
Q5	2246	57	2216.68	0.03	0.99	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M1: Unbalanced, Step, original variables.
m11_BSO <- oneStepwiseModel(train = select(trainBalanced, mainOriginalVars),
                            test = select(test, mainOriginalVars),
                            test_buy = test$buy,
                            test_amt = test$target,
                            varType = "original")
```

Table 15: Gains Table Model 11: balanced train set, Stepwise logistic regression and original variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	344	17912.17	0.15	7.98	2246	344	17912.17	0.15	7.98	2.14	2.45
Q2	2246	177	5291.46	0.08	2.36	4492	521	23203.63	0.12	5.17	1.62	1.59
Q3	2246	144	5613.03	0.06	2.50	6738	665	28816.66	0.10	4.28	1.38	1.31
Q4	2246	83	3796.21	0.04	1.69	8984	748	32612.87	0.08	3.63	1.16	1.12
Q5	2246	57	3939.44	0.03	1.75	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M1: Unbalanced, Step, original variables.
m12_BST <- oneStepwiseModel(train = select(trainBalanced, mainTransformedVars),
                            test = select(test, mainTransformedVars),
                            test_buy = test$buy,
                            test_amt = test$target,
                            varType = "transformed")
```

Table 16: Gains Table Model 12: balanced train set, Stepwise logistic regression and transfromed variables.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	338	18252.83	0.15	8.13	2246	338	18252.83	0.15	8.13	2.10	2.50
Q2	2246	193	7233.71	0.09	3.22	4492	531	25486.54	0.12	5.67	1.65	1.74
Q3	2246	133	5213.23	0.06	2.32	6738	664	30699.77	0.10	4.56	1.37	1.40
Q4	2246	90	3725.78	0.04	1.66	8984	754	34425.55	0.08	3.83	1.17	1.18
Q5	2246	51	2126.76	0.02	0.95	11230	805	36552.31	0.07	3.25	1.00	1.00

5.3.6 Stepwise Models Comparison

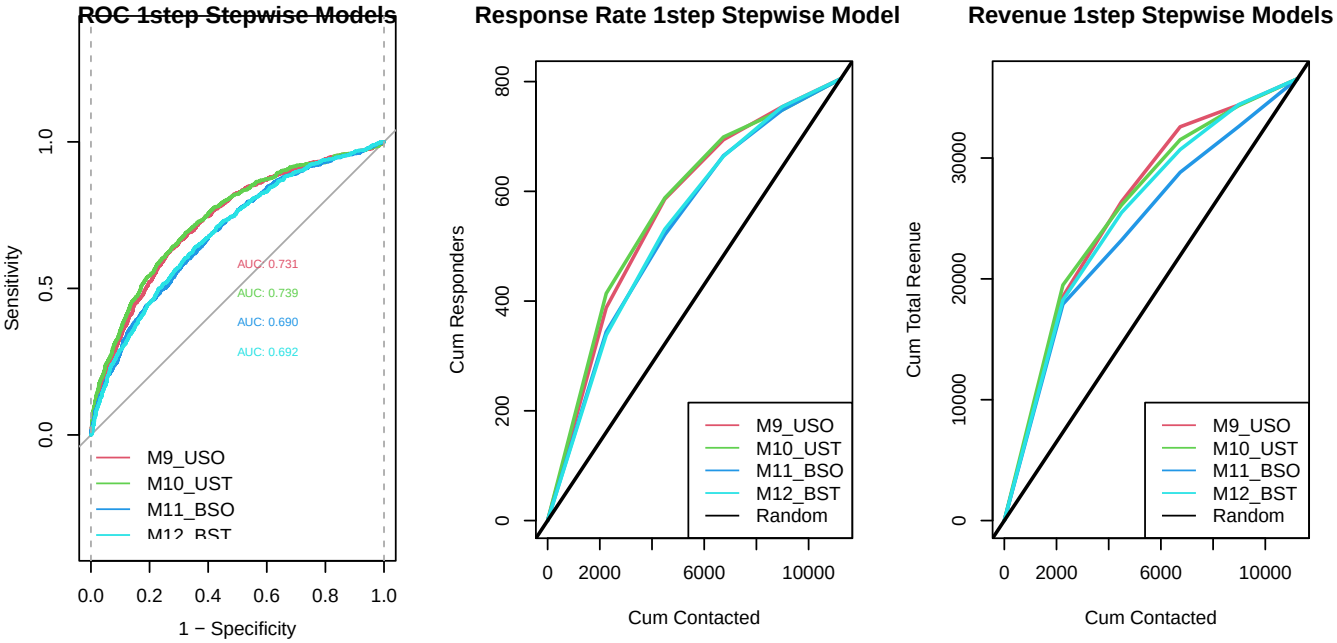
We compare the 4 different models under Stepwise Model using the gains table for the first and second quartiles (20% or 40% or customers) and the AUC criteria.

From table 17, we find that M10_UST has the best performance among 4 models because it has the highest value of lift response & lift Amount for the Q1 as well as Q2.

In addition to that, from the ROC curve, we can see that M10_UST also has the largest AUC compared with other models. Therefore, based on the result of gains table as well as the ROC curve, the best model is M10_UST, i.e. unbalanced train sample with transformed variable.

Table 17: Accuracy Metrics for Stepwise Models.

Model	Lift Resp Q1	Lift Amt Q1	Lift Resp Q2	Lift Amt Q2
M9_USO	2.41	2.54	1.82	1.81
M10_UST	2.57	2.67	1.83	1.79
M11_BSO	2.14	2.45	1.62	1.59
M12_BST	2.10	2.50	1.65	1.74



Best model m10_UST, i.e. unbalanced train sample and transformed variable. The estimated model is:

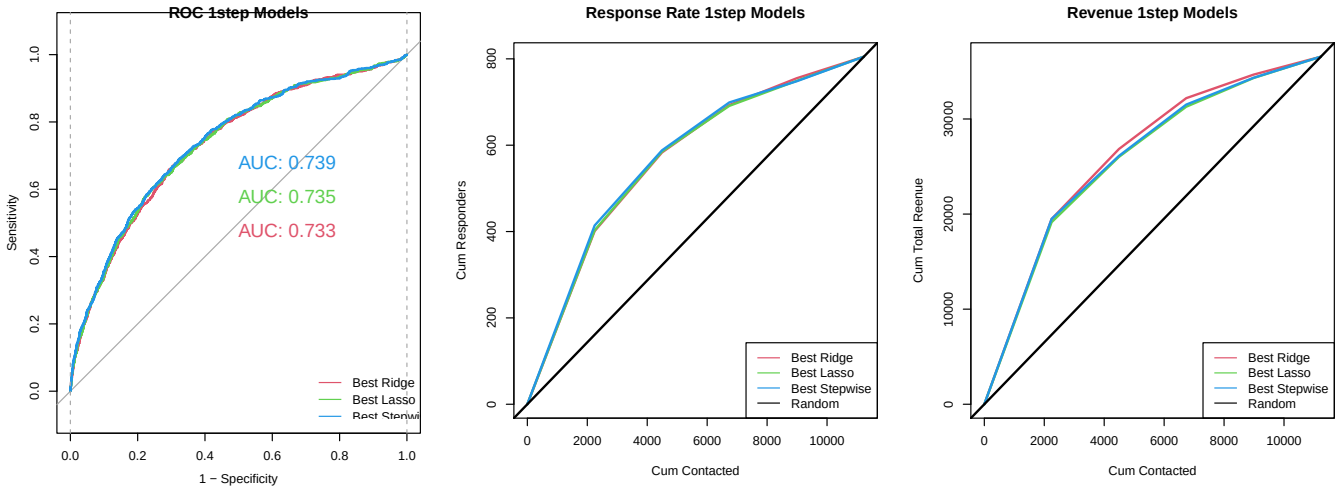
$$log\left(\frac{\pi_{buy}}{1-\pi_{buy}}\right) = -3.26 - 0.1tof - 0.24log(recency) + 0.59log(m) - 0.42log(AvOrderAmount) \\ -0.33log(AvItemAmount) + 0.26log(purchaseRate)$$

5.3.7 One step: Best Model Selection

```
# Best model for each method
bestRidge <- m2_URT
bestLasso <- m6_ULT
bestStepwise <- m10_UST
```

Table 18: Summary for Best Models

Model	Lift Resp Q1	Lift Amt Q1	Lift Resp Q2	Lift Amt Q2
Best Model Ridge	2.49	2.67	1.81	1.84
Best Model Lasso	2.50	2.62	1.82	1.78
Best Model Stepwise	2.57	2.67	1.83	1.79



Interpretation for chosen model

From the table above we can see that there is no clear model that dominates the others. Give that we want to focus on 20%-40% of the customers, based on the second quantile, we have found that the highest lift response is obtained with the stepwise model, however the highest lift amount for this quartile is obtained with the Ridge model. Nervertheless, there are caveats on model selection solely based on the best results.

In case of Stepwise Backward selection, at each step of iteration, the model will choose to drop one variable that will lead to the lowest AIC for that subset of variables. By exploring only one for each subset, in most cases, it ends up with a model that is not the best in terms of predictions. In this particular case of the model built, the terms avOrderAmt and avItemAmt can lead to overfitting as they depend on variables already represented in the equation.

In case of Ridge Logistic Regression also, there is definitely overfitting due to multiple variables selected being correlated. Hence, for continuity, it is recommended to use Lasso which is interpretable.

Therefore, we chose model M6_ULT, Lasso regression for unbalanced data and with transformed variables, as the model recommended for assessing a customer's tendency to buy. The model is:

$$log\left(\frac{\pi_{buy}}{1-\pi_{buy}}\right) = -3.04 - 0.14log(recency) + 0.06log(fitem) + 0.12log(ford) + 0.22log(purchaseRate)$$

This model is very intuitive because

1) Soon after a purchase, consumers would not tend to buy immediately. Since this is a company that supplies books, consumers who have just make a purchase would have likely bought sufficient books to last them for a period of time. Therefore, their tendency to buy again immediately after a recent purchase decreases. Hence the odds of buying decreases with recency.

2) The variable 'fitem' can be considered as an indicator of how spendthrift the consumer is. The more the frequency of items in their order history, the more likely the chances they will buy more as a response to the offer. Hence, the positive coefficient in the model concurred with our expectation.

3) Similarly, the variable 'ford' can also be considered as an indicator of how spendthrift the consumer is. The more frequen the orders, the more likely the chances the customer buys more. Hence, the positive coefficient in the model concurred with our expectation.

4) Finally, the 'purchaseRate' also follows a positive relationship with odds of buying. Although multicollinearity could arise with forder, it is more important to focus on the predictions. Hence, it is recommended to keep the term.

5.4 Two-step model

The big difference between a two-step model and a one-step model is that in the former after estimating a logistic regression model for the customer response, we estimate the conditional demand using a linear regression model.

It should be noted that the response model and the conditional demand model are not required to have the same set of predictor variables.[3] The estimation of the responses to the offer, first step, will be based on the best result from the one-step model, i.e. model M6_ULT, which was estimated for the unbalanced train set, with transformed variables and the Lasso logistic regression. The estimated probabilities of response will be taken from this result and used as an input in the second step.

As for the conditional demand (second step), we are going to try linear regression with regularization to predict the amount spent by a customer in response to the offer. Again we will use Ridge, Lasso regression and Stepwise as our estimation models and we will try using the balanced and unbalanced train set as well as the original and transformed variables. The models will be evaluated again with a gains table to account for the percentage of responders and the expected amount spent.

```
# Function to compute two-step models with regularization (Ridge or Lasso)
twoStepModel <- function(train, test,test_buy, test_amt, alpha, mod_logistic,
                         varType = "transformed", k = 5){

  # --- TRAINING ---
  # Model matrix for train
  if(varType == "transformed"){
    X_train = model.matrix(logTarget ~.-1, data = select(train, -buy))
  }else{
    X_train = model.matrix(target ~.-1, data = select(train, -buy))
  }


  # Use glmnet with cross validation to estimate the models with regularization
  if(varType == "transformed"){
    mod_cv <- cv.glmnet(X_train, train$logTarget,
                        alpha = alpha, nfolds = k)
  }else{
    mod_cv <- cv.glmnet(X_train, train$target,
                        alpha = alpha, nfolds = k)
  }


  # Pick lambda 1se
  if(varType == "transformed"){
    mod = glmnet(X_train, train$logTarget,
                 alpha = alpha, lambda = mod_cv$lambda.1se, nfolds = k)
  }else{
    mod = glmnet(X_train, train$target,
                 alpha = alpha, lambda = mod_cv$lambda.1se, nfolds = k)
  }


  # --- TESTING ---
```

[3] In the class example, we use the same predictors between response model and conditional demand model. However, this is not a requirement

```
  # Model matrix for test
  if(varType == "transformed"){
    X_test = model.matrix(logTarget ~.-1, data = select(test, -buy))
  }else{
    X_test = model.matrix(target ~.-1, data = select(test,-buy))
  }

  X_test <- as.tibble(X_test) %>%
    select(mod$beta %>% rownames()) %>%
    as.matrix()

  # Predicted target for test
  expected_amt = predict(mod, s = mod_cv$lambda.1se, newx = X_test)%>% as.vector()

  # MSE for test
  if(varType == "transformed"){
    MSE_test <- mean((exp(test$logTarget)-exp(expected_amt))^2)
  }else{
    MSE_test <- mean((test$target-expected_amt)^2)
  }

  # Get yhat using response probabilities from logistic model
  yhat = mod_logistic$phat*expected_amt

  #Gains table
  gains_tbl <- gains(yhat, test_buy, test_amt)

  output = list(mod = mod,
                yhat = yhat,
                MSE_test = MSE_test,
                gains_tbl = gains_tbl
  )

  return(output)
}
```

5.4.1 Ridge regression

```
# M1: Unbalanced, Ridge, original variables.
s2_m1_URO <- twoStepModel(train = select(trainUnbalanced, mainOriginalVars),
                          test = select(test, mainOriginalVars),
                          alpha = 0,
                          test_buy = test$buy,
                          test_amt = test$target,
                          mod_logistic = m6_ULT,
                          varType = "original",
                          k = 5)
```

Table 19: Gains Table 2-Step Model 1: unbalanced train set, Ridge linear regression and main original variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	405	19393.37	0.18	8.63	2246	405	19393.37	0.18	8.63	2.52	2.65
Q2	2246	181	7178.95	0.08	3.20	4492	586	26572.32	0.13	5.92	1.82	1.82
Q3	2246	104	4692.71	0.05	2.09	6738	690	31265.03	0.10	4.64	1.43	1.43
Q4	2246	59	3040.60	0.03	1.35	8984	749	34305.63	0.08	3.82	1.16	1.17
Q5	2246	56	2246.68	0.02	1.00	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M2: Unbalanced, Ridge, transformed variables.
s2_m2_URT <- twoStepModel(train = select(trainUnbalanced, mainTransformedVars),
                          test = select(test, mainTransformedVars),
                          alpha = 0,
                          test_buy = test$buy,
                          test_amt = test$target,
                          mod_logistic = m6_ULT,
                          varType = "transformed",
                          k = 5)
```

Table 20: Gains Table 2-Step Model 2: unbalanced train set, Ridge linear regression and main transformed variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	395	19284.13	0.18	8.59	2246	395	19284.13	0.18	8.59	2.45	2.64
Q2	2246	191	7512.41	0.09	3.34	4492	586	26796.54	0.13	5.97	1.82	1.83
Q3	2246	106	5322.62	0.05	2.37	6738	692	32119.16	0.10	4.77	1.43	1.46
Q4	2246	64	2608.43	0.03	1.16	8984	756	34727.59	0.08	3.87	1.17	1.19
Q5	2246	49	1824.72	0.02	0.81	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M3: balanced, Ridge, original variables.
s2_m3_BRO <- twoStepModel(train = select(trainBalanced, mainOriginalVars),
                          test = select(test, mainOriginalVars),
                          alpha = 0,
                          test_buy = test$buy,
                          test_amt = test$target,
                          mod_logistic = m6_ULT,
                          varType = "original",
                          k = 5)
```

Table 21: Gains Table 2-Step Model 3: balanced train set, Ridge linear regression and main original variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	358	20279.45	0.16	9.03	2246	358	20279.45	0.16	9.03	2.22	2.77
Q2	2246	214	8100.17	0.10	3.61	4492	572	28379.62	0.13	6.32	1.78	1.94
Q3	2246	113	3783.42	0.05	1.68	6738	685	32163.04	0.10	4.77	1.42	1.47
Q4	2246	70	2503.91	0.03	1.11	8984	755	34666.95	0.08	3.86	1.17	1.19
Q5	2246	50	1885.36	0.02	0.84	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M2: Unbalanced, Ridge, transformed variables.
s2_m4_BRT <- twoStepModel(train = select(trainBalanced, mainTransformedVars),
                          test = select(test, mainTransformedVars),
                          alpha = 0,
                          test_buy = test$buy,
                          test_amt = test$target,
                          mod_logistic = m6_ULT,
                          varType = "transformed",
                          k = 5)
```

Table 22: Gains Table 2-Step Model 4: balanced train set, Ridge linear regression and main transformed variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	395	19058.07	0.18	8.49	2246	395	19058.07	0.18	8.49	2.45	2.61
Q2	2246	186	7504.02	0.08	3.34	4492	581	26562.09	0.13	5.91	1.80	1.82
Q3	2246	107	5208.30	0.05	2.32	6738	688	31770.39	0.10	4.72	1.42	1.45
Q4	2246	66	2673.90	0.03	1.19	8984	754	34444.29	0.08	3.83	1.17	1.18
Q5	2246	51	2108.02	0.02	0.94	11230	805	36552.31	0.07	3.25	1.00	1.00

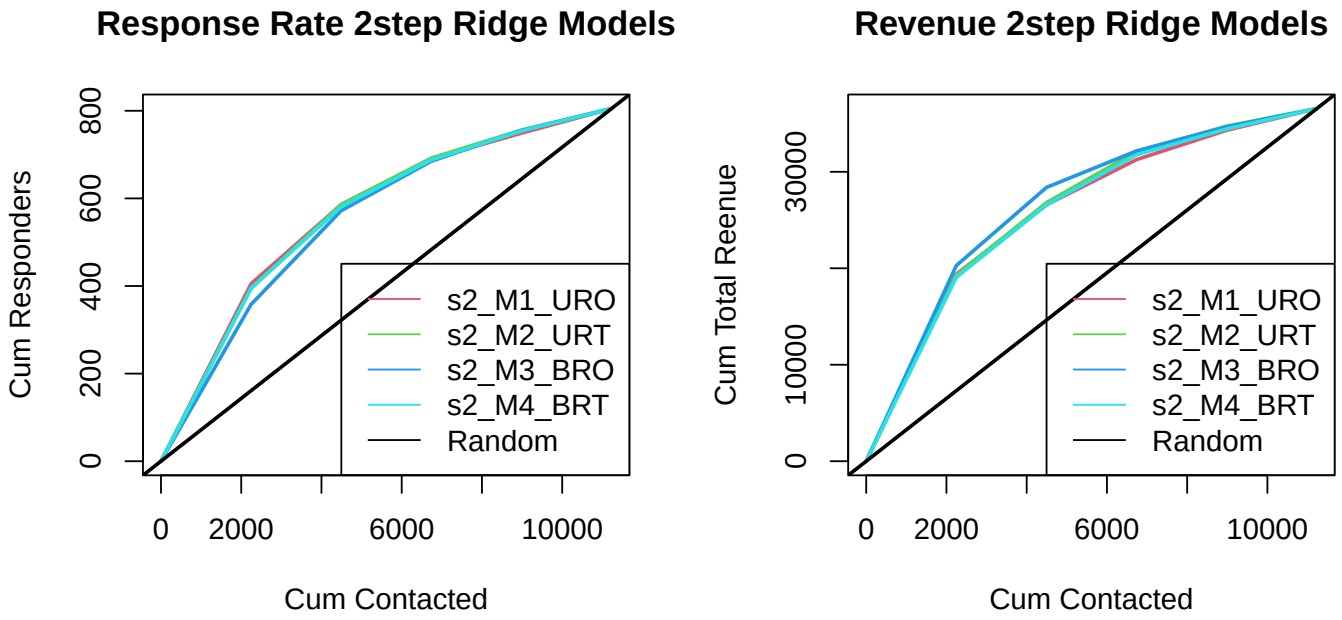
5.4.2 Linear Ridge Models Comparison

We compare the 4 different models under Linear Ridge Models using the gains table for the first and second quartiles (20% or 40% or customers) and the MSE criteria.

From table 23, we find that S2_M1_URO has the best performance among 4 models because it has a comparatively low MSE (375.05) at the same time having the highest value of lift response & lift Amount for the Q1 (2.52 Lift Response & 2.65 Lift Amount).

Table 23: Metrics for Conditional Demand with Ridge Models.

Model	MSE	Lift Resp Q1	Lift Amt Q1	Lift Resp Q2	Lift Amt Q2
S2_M1_URO	375.05	2.52	2.65	1.82	1.82
S2_M2_URT	383.53	2.45	2.64	1.82	1.83
S2_M3_BRO	683.20	2.22	2.77	1.78	1.94
S2_M4_BRT	374.20	2.45	2.61	1.80	1.82



The best model is s2_M2_URT i.e. unbalanced train sample and original variable. The estimated model is:

$$\begin{aligned} log(buy) &= 0.17 + -0.0002tof - 0.008log(recency) + 0.008log(fitem) + 0.009log(ford) + 0.006log(m) \\ &= + + 0.006log(itemPerOrder) + 0.002log(avOrderAmount) + 0.001log(avItemAmoun) \\ &= +0.012log(purchaseRate) + 0.0017ncat \end{aligned}$$

5.4.3 Lasso Regression

```
# M1: Unbalanced, Ridge, original variables.
s2_m5_ULO <- twoStepModel(train = select(trainUnbalanced, mainOriginalVars),
                          test = select(test, mainOriginalVars),
                          alpha = 1,
                          test_buy = test$buy,
                          test_amt = test$target,
                          mod_logistic = m6_ULT,
                          varType = "original",
                          k = 5)
```

Table 24: Gains Table 2-Step Model 5: unbalanced train set, Lasso linear regression and main original variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	403	19143.90	0.18	8.52	2246	403	19143.90	0.18	8.52	2.50	2.62
Q2	2246	182	6863.10	0.08	3.06	4492	585	26007.00	0.13	5.79	1.82	1.78
Q3	2246	106	5279.93	0.05	2.35	6738	691	31286.93	0.10	4.64	1.43	1.43
Q4	2246	58	3018.70	0.03	1.34	8984	749	34305.63	0.08	3.82	1.16	1.17
Q5	2246	56	2246.68	0.02	1.00	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M2: Unbalanced, Ridge, transformed variables.
s2_m6_ULT <- twoStepModel(train = select(trainUnbalanced, mainTransformedVars),
                          test = select(test, mainTransformedVars),
                          alpha = 1,
                          test_buy = test$buy,
                          test_amt = test$target,
                          mod_logistic = m6_ULT,
                          varType = "transformed",
                          k = 5)
```

Table 25: Gains Table 2-Step Model 6: unbalanced train set, Lasso linear regression and main transformed variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	396	18744.56	0.18	8.35	2246	396	18744.56	0.18	8.35	2.46	2.56
Q2	2246	185	7155.51	0.08	3.19	4492	581	25900.07	0.13	5.77	1.80	1.77
Q3	2246	111	5440.49	0.05	2.42	6738	692	31340.56	0.10	4.65	1.43	1.43
Q4	2246	57	3008.02	0.03	1.34	8984	749	34348.58	0.08	3.82	1.16	1.17
Q5	2246	56	2203.73	0.02	0.98	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M3: balanced, Ridge, original variables.
s2_m7_BLO <- twoStepModel(train = select(trainBalanced, mainOriginalVars),
                          test = select(test, mainOriginalVars),
                          alpha = 1,
                          test_buy = test$buy,
                          test_amt = test$target,
                          mod_logistic = m6_ULT,
                          varType = "original",
                          k = 5)
```

Table 26: Gains Table 2-Step Model 7: balanced train set, Lasso linear regression and main original variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	359	19578.93	0.16	8.72	2246	359	19578.93	0.16	8.72	2.23	2.68
Q2	2246	207	8315.32	0.09	3.70	4492	566	27894.25	0.13	6.21	1.76	1.91
Q3	2246	124	4371.84	0.06	1.95	6738	690	32266.09	0.10	4.79	1.43	1.47
Q4	2246	65	2448.51	0.03	1.09	8984	755	34714.60	0.08	3.86	1.17	1.19
Q5	2246	50	1837.71	0.02	0.82	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M2: Unbalanced, Ridge, transformed variables.
s2_m8_BLT <- twoStepModel(train = select(trainBalanced, mainTransformedVars),
                          test = select(test, mainTransformedVars),
                          alpha = 1,
                          test_buy = test$buy,
                          test_amt = test$target,
                          mod_logistic = m6_ULT,
                          varType = "transformed",
                          k = 5)
```

Table 27: Gains Table 2-Step Model 8: balanced train set, Lasso linear regression and main transformed variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	386	18388.84	0.17	8.19	2246	386	18388.84	0.17	8.19	2.40	2.52
Q2	2246	192	8186.36	0.09	3.64	4492	578	26575.20	0.13	5.92	1.80	1.82
Q3	2246	108	4615.62	0.05	2.06	6738	686	31190.82	0.10	4.63	1.42	1.42
Q4	2246	69	3204.42	0.03	1.43	8984	755	34395.24	0.08	3.83	1.17	1.18
Q5	2246	50	2157.07	0.02	0.96	11230	805	36552.31	0.07	3.25	1.00	1.00

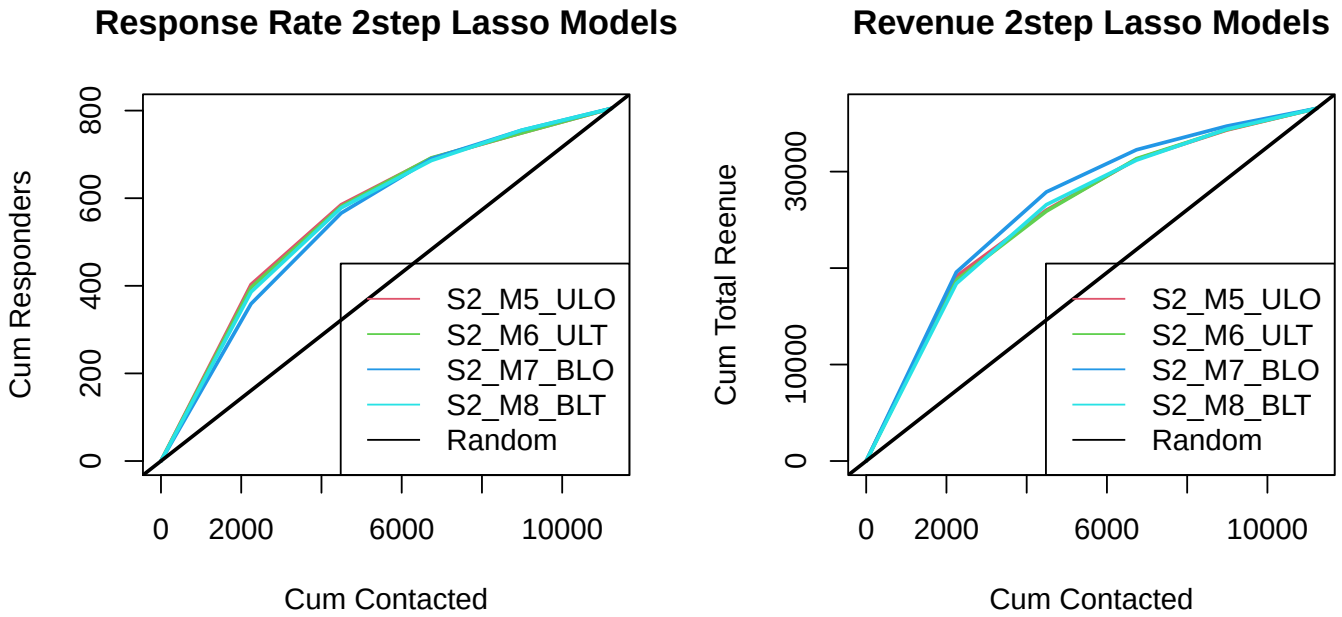
5.4.4 Linear Lasso Models Comparison

We compare the 4 different models under Lasso linear regression using the gains table for the first and second quartiles (20% or 40% or customers) and the MSE criteria.

From table 28, we find that S2_M5_ULO has the best performance among 4 models because it has a lowest MSE (375.67) at the same time having comparatively high value of lift response & lift Amount for the Q1 (2.50 Lift Response & 2.62 Lift Amount) and Q2 (1.82 Lift Response & 1.78 Lift Amount).

Table 28: Metrics for Conditional Demand with Lasso Models.

Model	MSE	Lift Resp Q1	Lift Amt Q1	Lift Resp Q2	Lift Amt Q2
S2_M5_ULO	375.67	2.50	2.62	1.82	1.78
S2_M6_ULT	384.20	2.46	2.56	1.80	1.77
S2_M7_BLO	677.34	2.23	2.68	1.76	1.91
S2_M8_BLT	400.75	2.40	2.52	1.80	1.82



Even though model S2_M5_ULO has higher lift response for the second quartile, it only has one predictor with a very small coefficient. Thus, to be consisten with the previous results, we select the secod best model S2_M6_ULT as the best Lasso model. the model is given by

Best model S2_M6_ULT

$$log(target) = 0.16 - 0.03log(recency) + 0.03log(fitem) + 0.04log(purchaseRate)$$

5.4.5 Stepwise

```
# Function to compute two-step models with regularization (Ridge or Lasso)
twoStepwiseModel <- function(train, test, mod_logistic, test_buy, test_amt,
                             varType = "transformed"){

  # --- TRAINING ---
  # Defining train set
  X_train = train %>% select(-buy)

  # Use linear regression to build model with all variables
  if(varType == "transformed"){
    model <- lm(X_train$logTarget~., data = X_train)
  }else{
    model <- lm(X_train$target~., data = X_train)
  }


  # Begin Stepwise function
```

```
  mod = step(model, trace = FALSE)


  # --- TESTING ---

  # Model matrix for test
  X_test = test %>% select(-buy)

  # Predictions from model
  expected_amt = predict(mod, X_test)

  # MSE for test
  if(varType == "transformed"){
    MSE_test <- mean((exp(test$logTarget)-exp(expected_amt))^2)
  }else{
    MSE_test <- mean((test$target-expected_amt)^2)
  }

  # Get yhat using response probabilities from logistic model
  yhat = mod_logistic$phat*expected_amt

  #Gains table
  gains_tbl <- gains(yhat, test_buy, test_amt)

  output = list(mod = mod,
                yhat = yhat,
                MSE_test = MSE_test,
                gains_tbl = gains_tbl
  )

  return(output)
}
```

```
# M2: Unbalanced, stepwise, original variables.
s2_m9_USO <- twoStepwiseModel(train = select(trainBalanced, mainOriginalVars),
                              test = select(test, mainOriginalVars),
                              mod_logistic = m6_ULT,
                              test_buy = test$buy,
                              test_amt = test$target,
                              varType = "original")
```

Table 29: Gains Table 2-Step Model 9: unbalanced train set, stepwise linear regression and main original variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	347	19697.64	0.15	8.77	2246	347	19697.64	0.15	8.77	2.16	2.69
Q2	2246	210	8140.52	0.09	3.62	4492	557	27838.16	0.12	6.20	1.73	1.90
Q3	2246	125	3831.03	0.06	1.71	6738	682	31669.19	0.10	4.70	1.41	1.44
Q4	2246	67	2418.11	0.03	1.08	8984	749	34087.30	0.08	3.79	1.16	1.17
Q5	2246	56	2465.01	0.02	1.10	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M10: Unbalanced, Stepwise, transformed variables.
s2_m10_UST <- twoStepwiseModel(train = select(trainUnbalanced, mainTransformedVars),
```

```
                                test = select(test, mainTransformedVars),
                                mod_logistic = m6_ULT,
                                test_buy = test$buy,
                                test_amt = test$target,
                                varType = "transformed")
```

Table 30: Gains Table 2-Step Model 10: Unbalanced train set, Stepwise linear regression and main transformed variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	403	19461.01	0.18	8.66	2246	403	19461.01	0.18	8.66	2.50	2.66
Q2	2246	185	7490.86	0.08	3.34	4492	588	26951.87	0.13	6.00	1.83	1.84
Q3	2246	112	5521.14	0.05	2.46	6738	700	32473.01	0.10	4.82	1.45	1.48
Q4	2246	52	1979.57	0.02	0.88	8984	752	34452.58	0.08	3.83	1.17	1.18
Q5	2246	53	2099.73	0.02	0.93	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M11: Balanced, Stepwise, Original variables.
s2_m11_BSO <- twoStepwiseModel(train = select(trainBalanced, mainOriginalVars),
                                test = select(test, mainOriginalVars),
                                mod_logistic = m6_ULT,
                                test_buy = test$buy,
                                test_amt = test$target,
                                varType = "original")
```

Table 31: Gains Table 2-Step Model 11: balanced train set, Stepwise linear regression and main transformed variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	347	19697.64	0.15	8.77	2246	347	19697.64	0.15	8.77	2.16	2.69
Q2	2246	210	8140.52	0.09	3.62	4492	557	27838.16	0.12	6.20	1.73	1.90
Q3	2246	125	3831.03	0.06	1.71	6738	682	31669.19	0.10	4.70	1.41	1.44
Q4	2246	67	2418.11	0.03	1.08	8984	749	34087.30	0.08	3.79	1.16	1.17
Q5	2246	56	2465.01	0.02	1.10	11230	805	36552.31	0.07	3.25	1.00	1.00

```
# M9: Balanced, Stepwise, transformed variables.
s2_m12_BST <- twoStepwiseModel(train = select(trainBalanced, mainTransformedVars),
                                test = select(test, mainTransformedVars),
                                mod_logistic = m6_ULT,
                                test_buy = test$buy,
                                test_amt = test$target,
                                varType = "transformed")
```

Table 32: Gains Table 2-Step Model 12: balanced train set, Stepwise linear regression and main transformed variables for expected demand.

Quantile	n	Nrespond	amt	RespRate	AvgAmt	CumN	CumResp	CumAmt	CumRespRate	CumAvgAmt	liftResp	liftAmt
Q1	2246	384	18593.00	0.17	8.28	2246	384	18593.00	0.17	8.28	2.39	2.54
Q2	2246	191	8013.51	0.09	3.57	4492	575	26606.51	0.13	5.92	1.79	1.82
Q3	2246	110	4961.78	0.05	2.21	6738	685	31568.29	0.10	4.69	1.42	1.44
Q4	2246	73	2983.00	0.03	1.33	8984	758	34551.29	0.08	3.85	1.18	1.18
Q5	2246	47	2001.02	0.02	0.89	11230	805	36552.31	0.07	3.25	1.00	1.00

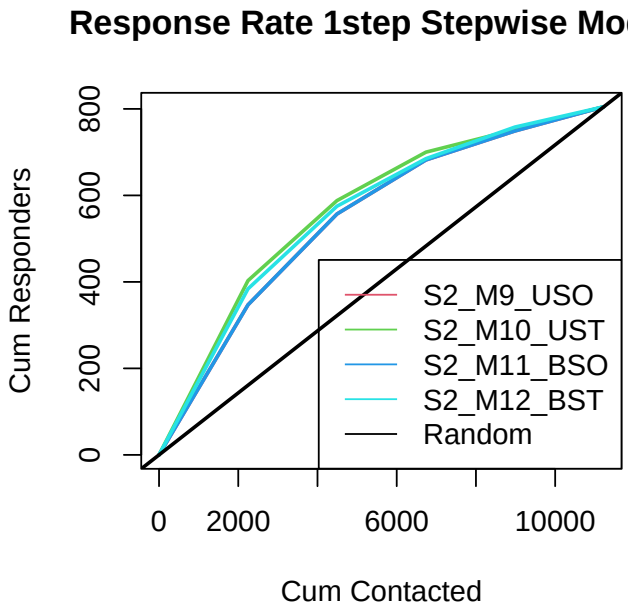
5.4.6 Linear Stepwise Models Comparison

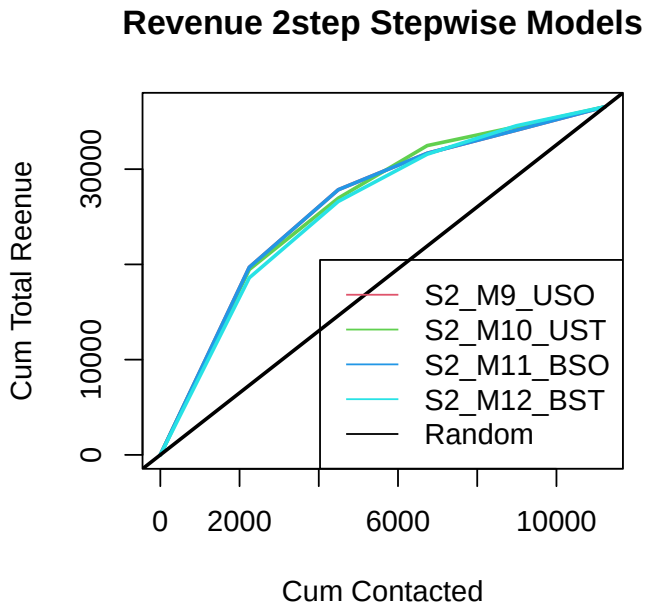
We compare the 4 different models under Ridge linear regression using the gains table for the first and second quartiles (20% or 40% or customers) and the MSE criteria.

From table 33, we find that S2_M10_UST has the best performance among 4 models because it has a lowest MSE (382.13) at the same time having the highest value of lift response & lift Amount for the Q1 (2.50 Lift Response & 2.66 Lift Amount) and Q2(1.83 Lift Response & 1.84 Lift Amount).

Table 33: Metrics for Conditional Demand with Lasso Models.

Model	MSE	Lift Resp Q1	Lift Amt Q1	Lift Resp Q2	Lift Amt Q2
S2_M9_USO	775.43	2.16	2.69	1.73	1.90
S2_M10_UST	382.13	2.50	2.66	1.83	1.84
S2_M11_BSO	775.43	2.16	2.69	1.73	1.90
S2_M12_BST	408.46	2.39	2.54	1.79	1.82





best model is s2_M10_UST

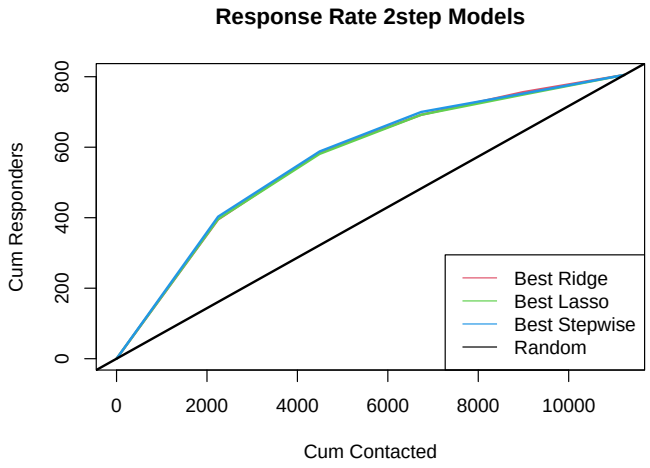
$log(target) = 0.09 - 0.03tof - 0.08log(recency) + 0.06log(fitem) - 1.23log(ford) + 1.34log(m) - 1.35log(avOrderAmount)$

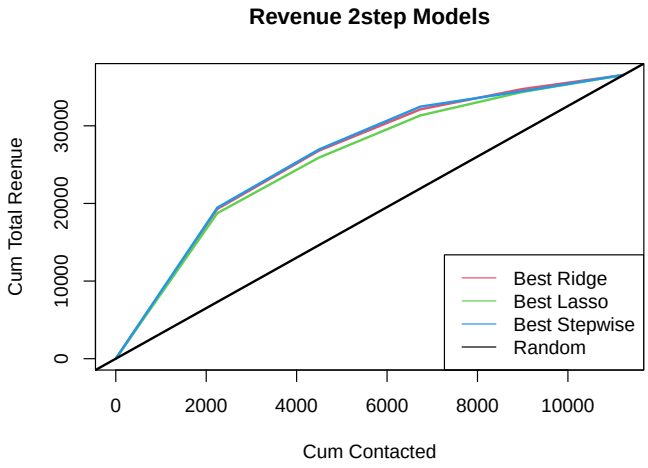
5.4.7 2 Step Model: Best Model Selection

```
# Best model for each method
bestRidge2s <- s2_m2_URT
bestLasso2s <- s2_m6_ULT
bestStepwise2s <- s2_m10_UST
```

Table 34: Metrics Summary for Best 2 step Models.

Model	Lift Amt Q1	Lift Resp Q2	Lift Amt Q2
Best Model Ridge	2.64	1.82	1.83
Best Model Lasso	2.56	1.80	1.77
Best Model Stepwise	2.66	1.83	1.84





5.5 Interpretation of the best model

The model recommended for assessing a customer's tendency to buy is given by:

$$log(target) = 0.16 - 0.03log(recency) + 0.03log(fitem) + 0.04log(purchaseRate)$$

This is intuitive because

1) Similarly to probability of odds of a consumer buying again, a more recent purchase delays the purchase of the next. Hence, it is negatively related to the target amount.

2) The variables 'fitem' and 'purchaseRate' are an indicator of how willing a consumer is to purchase. The more they have purchased previously, the more likely they will continue similar trend (assuming there are no unforeseen events that adversely affect the buying patterns of consumers).

This model also provides the best liftAmt and liftResp in Q1 and Q2.

6 Selected Model

As previously described, for both the one step model and the 2step model, we consider that Lasso regression with the unbalanced set and transformed variables works best.

For the one-step model, the selected model is:

$$log\left(\frac{\pi_{buy}}{1-\pi_{buy}}\right) = -3.04 - 0.14log(recency) + 0.06log(fitem) + 0.12log(ford) + 0.22log(purchaseRate)$$

which gives an increase of 182% responders and of 178% of amount spend compared to random selection of customers (refer to section 5.3.7 for the description of the model). With this model, if 40% of the customers are contacted we would expect a response of 585 represent around $26,007 on sales.

If we select a two-step model, first using the Lasso logistic regression above to estimate the response rate and then estimating the conditional demand we find that also the model with Lasso regression, unbalanced train set and transformed variables is the most reliable model (refer to section 5.4.7 for the description of the model). Then the model for estimating the conditional demand is:

$$log(target) = 0.16 - 0.03log(recency) + 0.03log(fitem) + 0.04log(purchaseRate)$$

This model increases the number of responders in 180% and the amoount spent in 177% compared to random selection of customers. With this model, if 40% of the customers are contacted we would expect a response of 581 representing around $25,900 on sales. It is worth noting that, since we do not have a measure equivalent to confidence intervals, the improvement of using a two-step model is not perceivable.

7 Conclusions

We found that transforming the dependent variable is crucial for this problem. From the histograms presented in section 4.2, we can observe that target is right skewed which means that most of the customers do not spend money at all and the distribution for those who do buy books is skewed. Log transforming this variable helps to reduce the skewness and achieve a symmetric distribution.

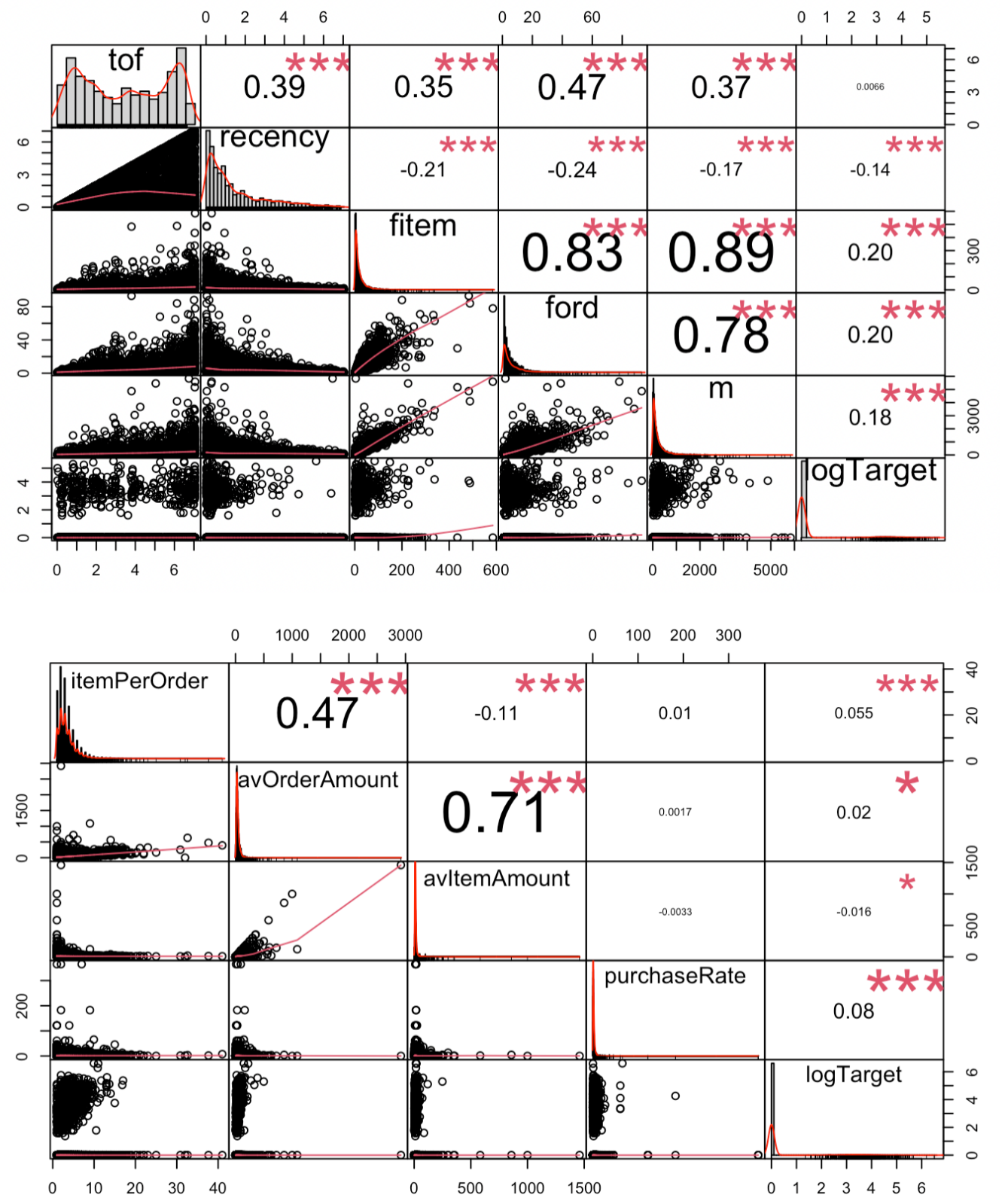
As for transformations on the predictors, for the 1 step models, we find that the model performance using transformed variables is better than using the original variables. Therefore, transformation on x for 1 Step Model helped. For 2-step model, we find that using original variables or transformed variables does not have a significant impact on the lift response or lift amount.

Additionally, trying to overcome the unbalanced classes did not make a difference when estimating our models. In most of the cases, the unbalanced train set outperforms the balanced train set.

Shrinkage and selection methods are important. Our train set ended up with around 106 predictors, thus using variable selection (shrinkage) methods was important. The variables included in the model using Lasso, Ridge and Stepwise where different; however, we find that recency, fitem, purchase rate, and ford were consistently selected. Thus, we consider this to be the main expalantory variables for this problem.

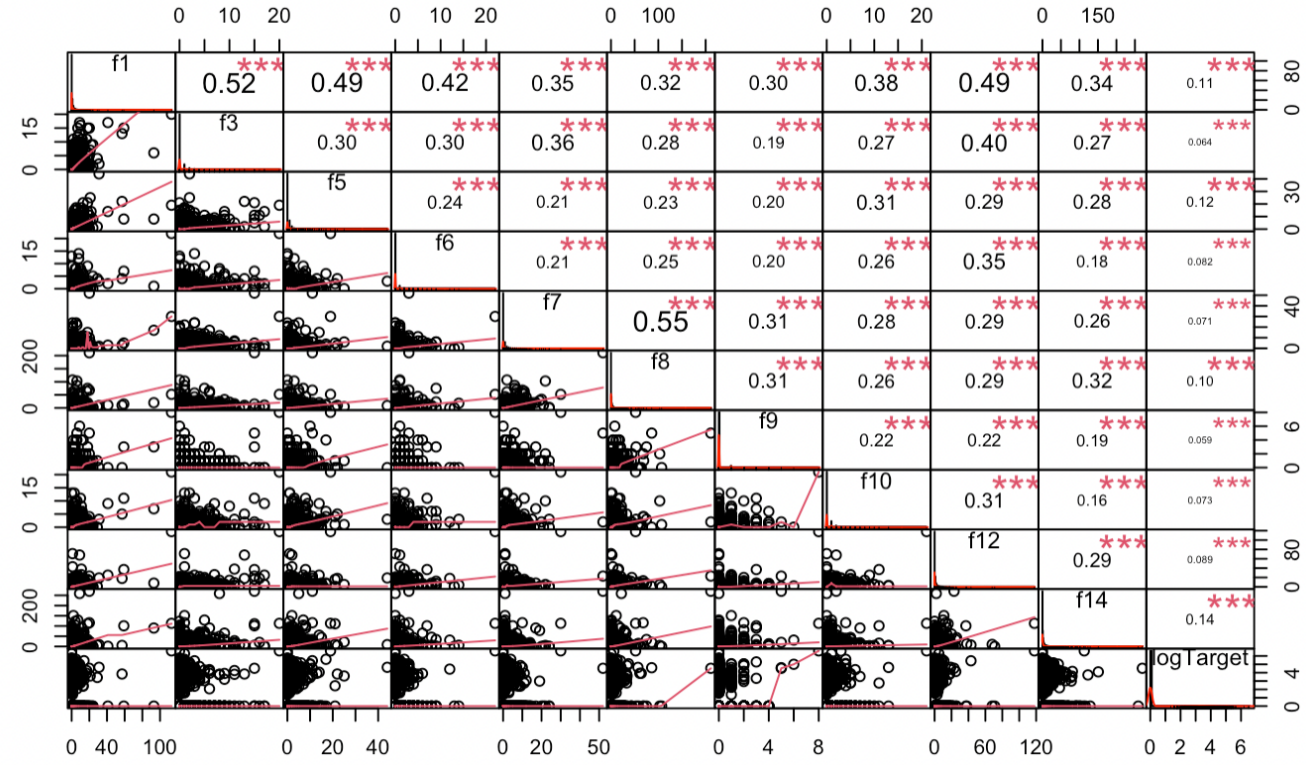
8 Appendix

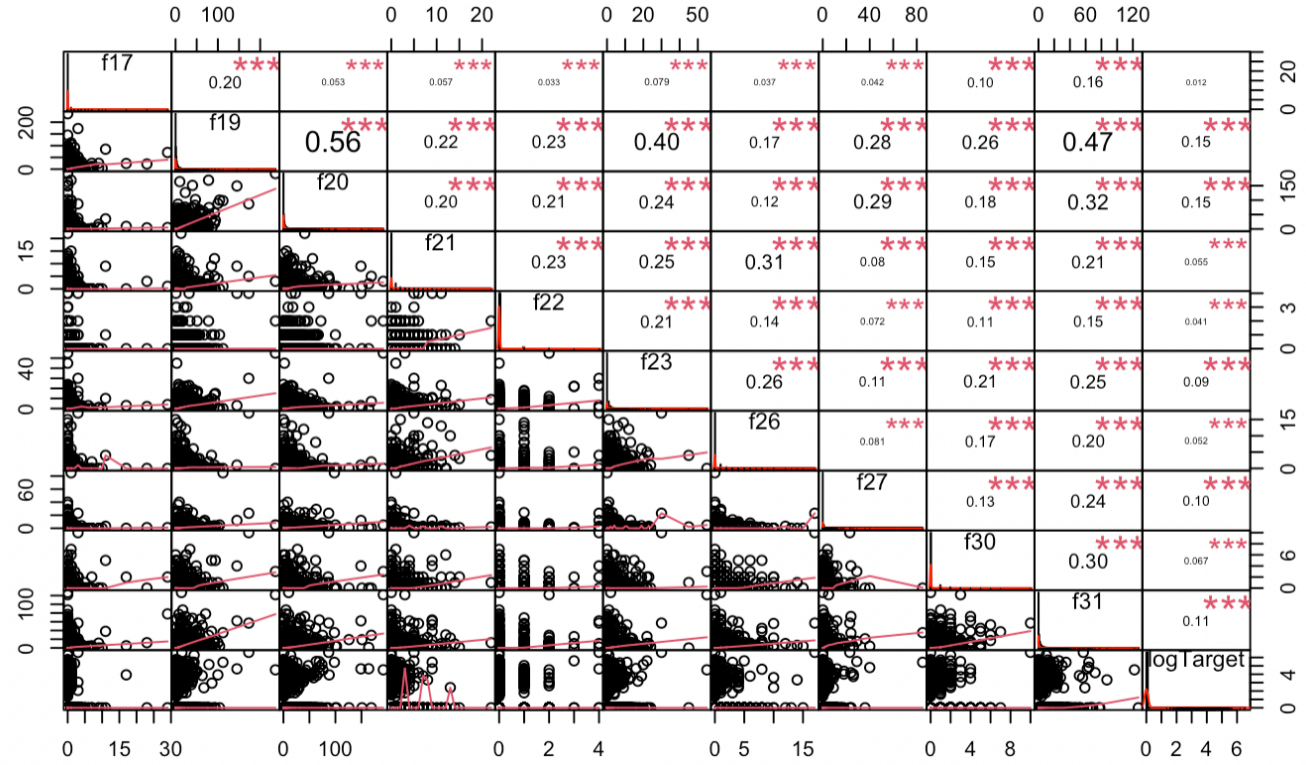
8.1 Correlation plots.

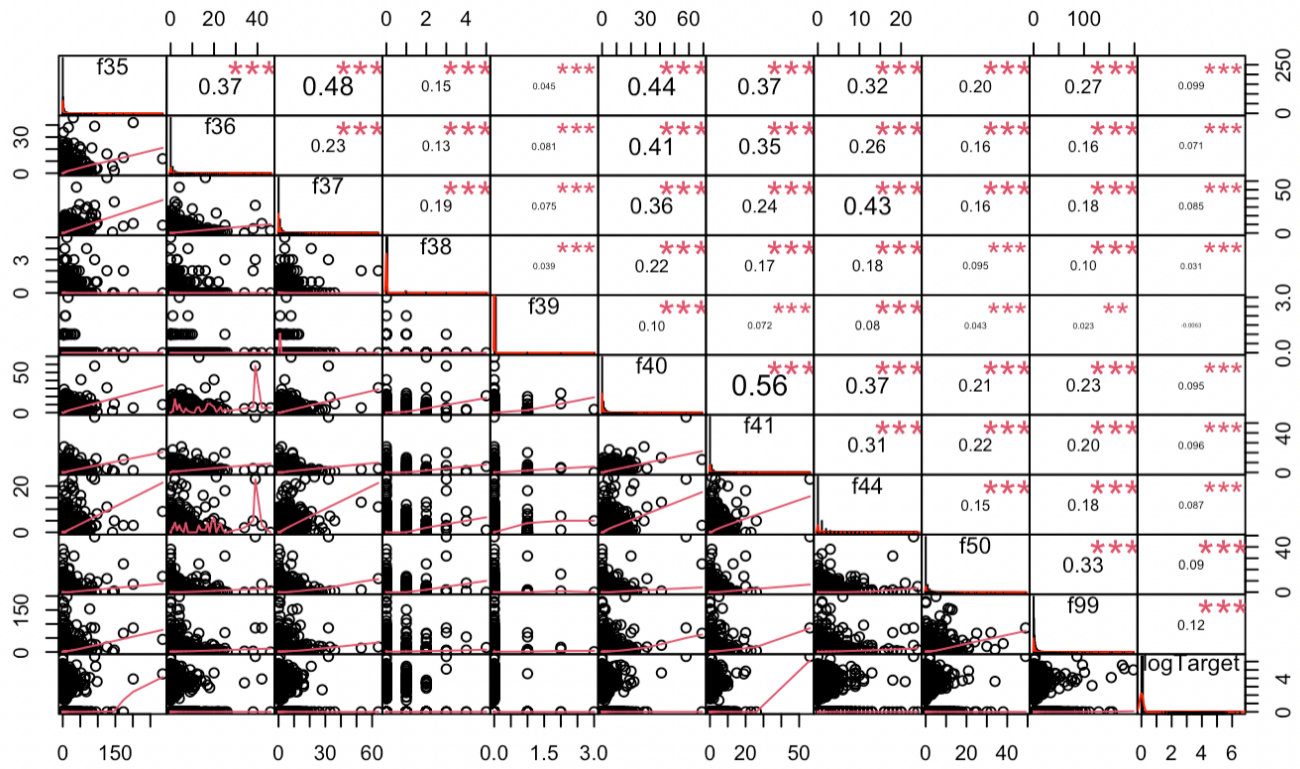



```
# Correlation plot
df2 <- book %>%
  select(starts_with("f"), logTarget) %>%
  select(-c(fitem, ford))
```

```
chart.Correlation(select(df2, f1:f14,logTarget), histogram=TRUE, pch=19)
chart.Correlation(select(df2, f17:f31,logTarget), histogram=TRUE, pch=19)
chart.Correlation(select(df2, f35:f99,logTarget), histogram=TRUE, pch=19)
```

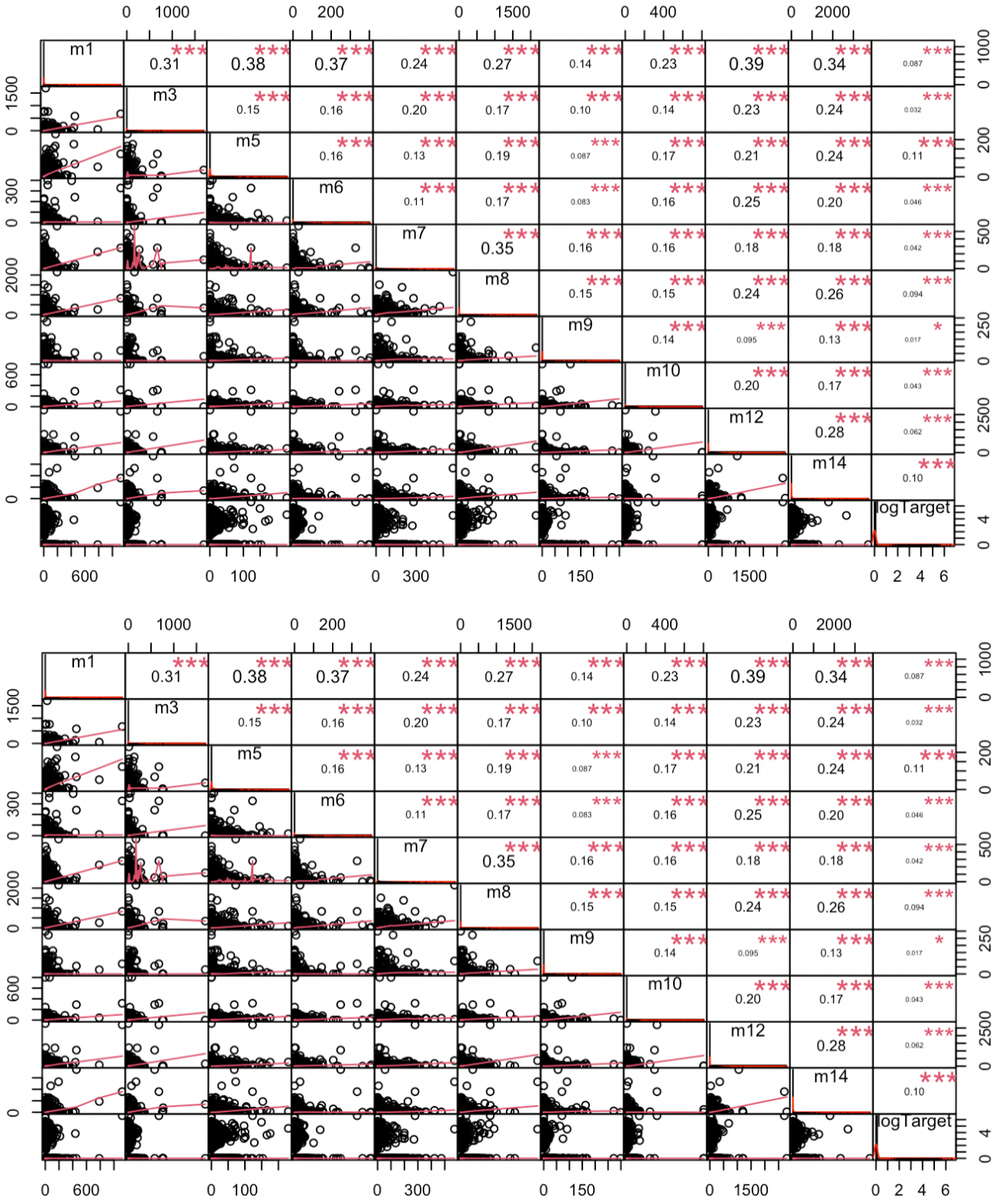


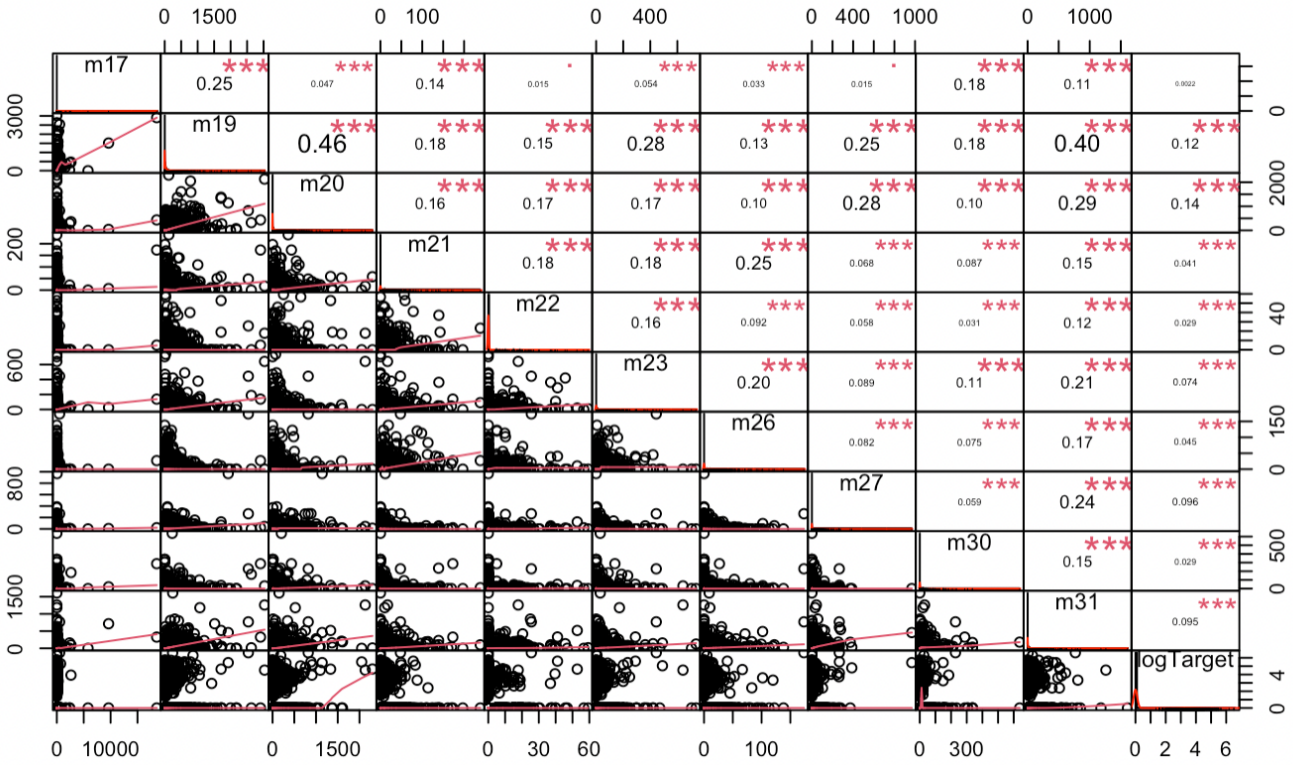


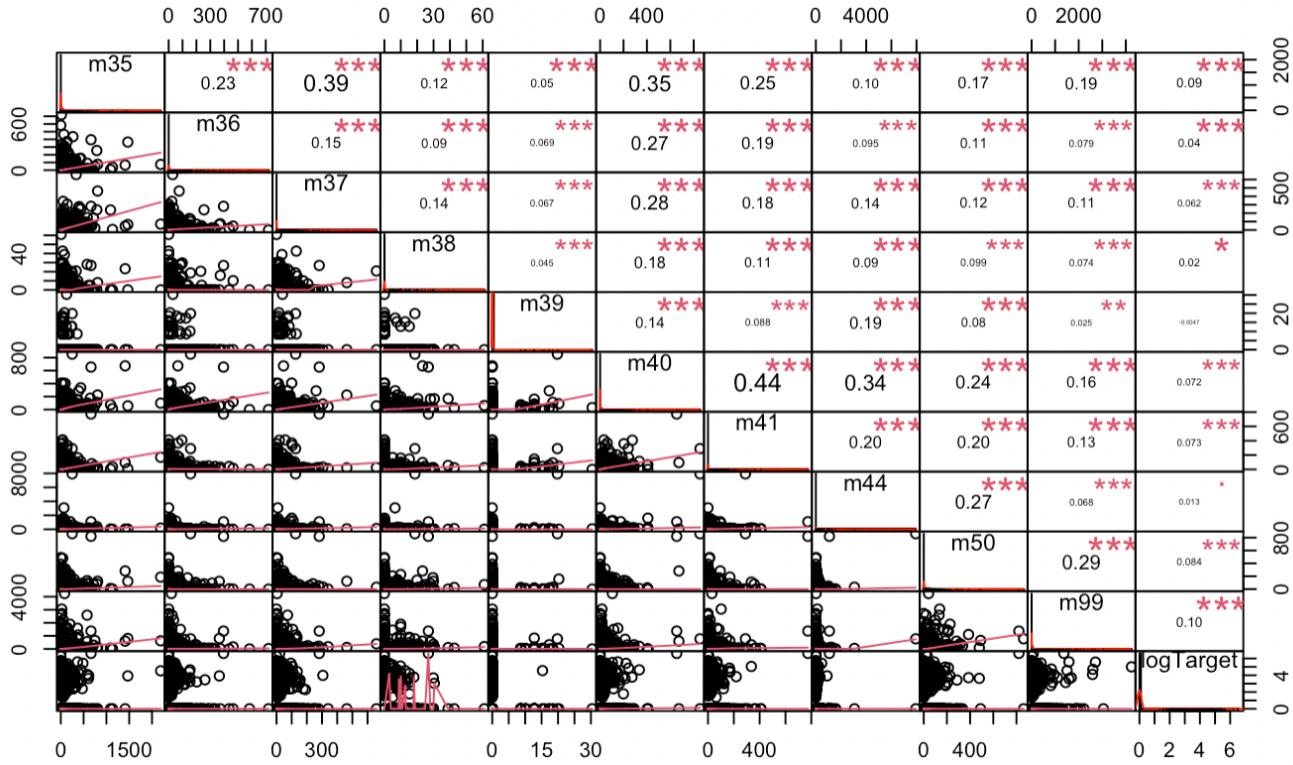



```
# Correlation plot
df3 <- book %>%
  mutate(logTarget = log(target+1)) %>%
  select(starts_with("m"), logTarget) %>%
  select(-c(m))

chart.Correlation(select(df3, m1:m14,logTarget), histogram=TRUE, pch=19)
chart.Correlation(select(df3, m17:m31,logTarget), histogram=TRUE, pch=19)
chart.Correlation(select(df3, m35:m99,logTarget), histogram=TRUE, pch=19)
```


m1
m3
m5
m6
m7
m8
m9
m10
m12
m14
logTarget


m17
m19
m20
m21
m22
m23
m26
m27
m30
m31
logTarget


m35
m36
m37
m38
m39
m40
m41
m44
m50
m99
logTarget