Customer Acquisition & Scoring

## 0.1 First Steps

# Clear All Variables & Clear Screen  
rm(list=ls())  
cat("\014")

# Read in the Data  
data.train = read.csv("Data\_Estimation\_R.csv")  
data.test = read.csv("Data\_Holdout\_R.csv")  
  
# Explore the data  
str(data.train)

## 'data.frame': 200 obs. of 8 variables:  
## $ id : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ gender: int 1 0 0 1 0 1 1 1 0 0 ...  
## $ hl1 : int 302 221 202 148 43 183 163 474 446 113 ...  
## $ hl2 : int 0 0 9 0 0 0 0 9 0 0 ...  
## $ hl3 : int 0 10 45 15 15 0 0 40 20 0 ...  
## $ hl5 : int 0 12 0 0 0 12 0 0 12 0 ...  
## $ hl6 : int 0 26 13 0 0 0 0 0 26 15 ...  
## $ y : int 1 0 0 0 0 0 1 1 1 0 ...

summary(data.train)

## id gender hl1 hl2   
## Min. : 1.00 Min. :0.000 Min. : 16.0 Min. : 0.00   
## 1st Qu.: 50.75 1st Qu.:0.000 1st Qu.:120.8 1st Qu.: 0.00   
## Median :100.50 Median :0.000 Median :191.5 Median : 0.00   
## Mean :100.50 Mean :0.325 Mean :205.9 Mean : 4.08   
## 3rd Qu.:150.25 3rd Qu.:1.000 3rd Qu.:278.0 3rd Qu.: 9.00   
## Max. :200.00 Max. :1.000 Max. :476.0 Max. :57.00   
## hl3 hl5 hl6 y   
## Min. : 0.00 Min. : 0.00 Min. : 0.00 Min. :0.00   
## 1st Qu.: 0.00 1st Qu.: 0.00 1st Qu.: 0.00 1st Qu.:0.00   
## Median :10.00 Median : 0.00 Median : 0.00 Median :0.00   
## Mean :10.65 Mean : 2.88 Mean : 6.24 Mean :0.36   
## 3rd Qu.:15.00 3rd Qu.: 0.00 3rd Qu.:13.00 3rd Qu.:1.00   
## Max. :60.00 Max. :39.00 Max. :69.00 Max. :1.00

str(data.test)

## 'data.frame': 300 obs. of 8 variables:  
## $ id : int 201 202 203 204 205 206 207 208 209 210 ...  
## $ gender: int 0 1 1 1 0 0 1 0 0 0 ...  
## $ hl1 : int 158 187 313 310 37 78 427 30 286 249 ...  
## $ hl2 : int 0 0 0 0 9 0 9 0 0 0 ...  
## $ hl3 : int 0 0 25 0 0 0 10 0 15 10 ...  
## $ hl5 : int 0 0 0 0 0 0 0 0 0 24 ...  
## $ hl6 : int 13 0 0 0 0 0 26 13 0 26 ...  
## $ y : int 1 0 0 1 1 1 1 1 0 0 ...

summary(data.test)

## id gender hl1 hl2 hl3   
## Min. :201.0 Min. :0.00 Min. : 17.0 Min. : 0.00 Min. : 0.00   
## 1st Qu.:275.8 1st Qu.:0.00 1st Qu.:135.0 1st Qu.: 0.00 1st Qu.: 0.00   
## Median :350.5 Median :0.00 Median :219.0 Median : 0.00 Median :10.00   
## Mean :350.5 Mean :0.28 Mean :216.8 Mean : 4.18 Mean :13.28   
## 3rd Qu.:425.2 3rd Qu.:1.00 3rd Qu.:291.5 3rd Qu.: 9.00 3rd Qu.:20.00   
## Max. :500.0 Max. :1.00 Max. :474.0 Max. :33.00 Max. :70.00   
## hl5 hl6 y   
## Min. : 0.00 Min. : 0.000 Min. :0.0000   
## 1st Qu.: 0.00 1st Qu.: 0.000 1st Qu.:0.0000   
## Median : 0.00 Median : 0.000 Median :0.0000   
## Mean : 3.74 Mean : 6.253 Mean :0.3333   
## 3rd Qu.: 0.00 3rd Qu.:13.000 3rd Qu.:1.0000   
## Max. :39.00 Max. :56.000 Max. :1.0000

## 1. Predict y (i.e., the decision to join the club) as a function of the available scoring variables x (gender and all hl…) using a LOGIT model. Include an intercept term to account for a base response rate. Keep all coefficients (i.e., do not eliminate coefficients which seems to be statistically insignificant).

# Run the Binary Logit Model on the training set (includes an INTERCEPT)  
glm.model <- glm(y ~ gender + hl1 + hl2 + hl3 + hl5 + hl6, family=binomial(link='logit'), data=data.train)  
  
# Display Results  
summary(glm.model)

##   
## Call:  
## glm(formula = y ~ gender + hl1 + hl2 + hl3 + hl5 + hl6, family = binomial(link = "logit"),   
## data = data.train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6854 -0.9444 -0.6260 1.1212 2.1831   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.928404 0.361689 -2.567 0.01026 \*   
## gender -0.016632 0.344941 -0.048 0.96154   
## hl1 0.005733 0.001840 3.115 0.00184 \*\*   
## hl2 -0.045830 0.026570 -1.725 0.08455 .   
## hl3 -0.068239 0.017004 -4.013 5.99e-05 \*\*\*  
## hl5 0.004349 0.026228 0.166 0.86830   
## hl6 -0.004919 0.017404 -0.283 0.77746   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 261.37 on 199 degrees of freedom  
## Residual deviance: 234.26 on 193 degrees of freedom  
## AIC: 248.26  
##   
## Number of Fisher Scoring iterations: 4

# Evaluate the accuracy of the model   
data.train$p <- round(predict(glm.model, data.train, type = c("response")), digits = 0)  
  
print(paste("The accuracy rate of the model w.r.t the training set is ", sum(data.train$p==data.train$y)/200\*100, "%"))

## [1] "The accuracy rate of the model w.r.t the training set is 70.5 %"

## 2. Based on your logit model, score all individuals on in the Testing sample (you can do this manually, e.g., in Excel, or adapt the R code from class). This means calculate, for all prospects in the Testing sample, the predicted response rate. Using your model, compute (for each individual):

### (a) Predicted Response Rate

# Predicting response for the 300 TESTING IDs based on the Model Estimates  
prediction.test <- data.frame(ID = data.test$id,   
 BinaryLogitProbability = predict(glm.model, data.test, type = c("response")),  
 BinaryLogitPredict = round(predict(glm.model, data.test, type = c("response")), digits = 0),  
 BinaryLogitActual = data.test$y)

### (b) Lift

# Add Lift to the Forecast. Recall lift is simply the predicted response rate divided by the average response rate of the Training sample  
prediction.test$lift = prediction.test$BinaryLogitProbability/mean(data.train$y)  
  
# print out the first 10 prospects  
head(prediction.test,10)

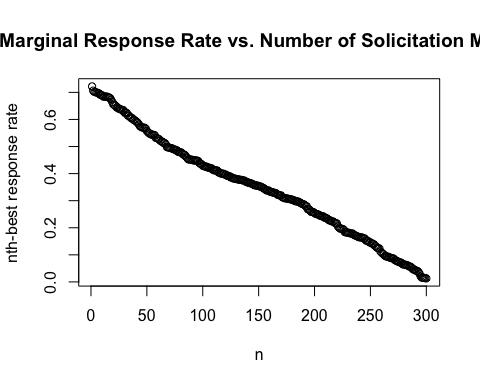
## ID BinaryLogitProbability BinaryLogitPredict BinaryLogitActual lift  
## 1 201 0.4783915 0 1 1.3288654  
## 2 202 0.5317304 1 0 1.4770288  
## 3 203 0.2980727 0 0 0.8279797  
## 4 204 0.6968386 1 1 1.9356628  
## 5 205 0.2443930 0 1 0.6788694  
## 6 206 0.3819674 0 1 1.0610205  
## 7 207 0.5696267 1 1 1.5822965  
## 8 208 0.3056892 0 1 0.8491368  
## 9 209 0.4225614 0 0 1.1737816  
## 10 210 0.4485039 0 0 1.2458442

## 3. Sort the holdout-list in decreasing order of lift.

# sort the table  
prediction.test.sort <- prediction.test[order(prediction.test$lift, decreasing=TRUE),]  
  
# reset row index  
row.names(prediction.test.sort) <- NULL  
prediction.test.sort$n <- row.names(prediction.test.sort)

## 4. Plot Marginal Response Rate vs. Number of Prospects Targeted

# Now we can make a plot of the response rate by number of prospects targeted  
plot(prediction.test.sort$BinaryLogitProbability, main="Marginal Response Rate vs. Number of Solicitation Made",  
 xlab="n", ylab="nth-best response rate")



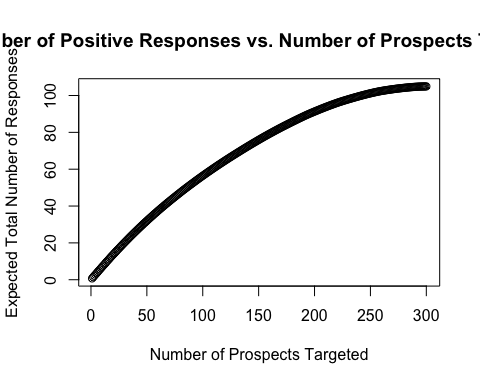
## 5. We know that average CLV is $30 and the solicitation cost is $12. Based on the Marginal Cost Rule determine who the CD club should send invitations to.

# These two values are given  
solicitation\_cost <- 12  
mean\_CLV <- 30  
  
# calculate the minimum response rate that the firm should target based on Marginal Cost Rule  
min\_response\_rate <- solicitation\_cost / mean\_CLV  
# get the prospects to send invitations to based on MCR (Marginal Cost Rule)  
MCR.target <- prediction.test.sort[prediction.test.sort$BinaryLogitProbability>min\_response\_rate,]  
  
  
print(paste("According to the Marginal Cost Rule, the Cut-Off Response is at ", min\_response\_rate, ". Therefore, the CD club should send invitation to the", nrow(MCR.target) ,"prospects with highest predicted response rate."))

## [1] "According to the Marginal Cost Rule, the Cut-Off Response is at 0.4 . Therefore, the CD club should send invitation to the 116 prospects with highest predicted response rate."

## 6. Compute the Cumulative Sum (aka running sum) for the Predicted Response Rates in decreasing order. Plot the curve for Number of Positive Responses vs. Number of Prospects Targeted.

# Add a column and calculate the running sum use the 'cumsum' function  
prediction.test.sort$cum\_sum\_p <- cumsum(prediction.test.sort$BinaryLogitProbability)  
  
# plot the curve for running sum of the predicted response rate  
plot(x=as.integer(rownames(prediction.test.sort)), y=prediction.test.sort$cum\_sum\_p, xlab = "Number of Prospects Targeted", ylab = "Expected Total Number of Responses", main = "Number of Positive Responses vs. Number of Prospects Targeted")



## 7. The CD club has only 40 items of the collector’s edition of “Pink Floyd’s The Wall”. Based on the Limited Supply Rule, which prospects (and how many) on the Testing list should the CD club send an invitation to?

# set the item limit  
k = 40  
  
# Find the target prospects based on LSR (Limited Supply Rule)  
LSR.target <- prediction.test.sort[prediction.test.sort$cum\_sum\_p < k,]  
  
# Inspect the result  
str(LSR.target)

## 'data.frame': 64 obs. of 7 variables:  
## $ ID : int 331 392 220 360 301 485 204 491 498 332 ...  
## $ BinaryLogitProbability: num 0.722 0.706 0.702 0.702 0.699 ...  
## $ BinaryLogitPredict : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ BinaryLogitActual : int 0 1 1 1 0 0 1 0 1 1 ...  
## $ lift : num 2.01 1.96 1.95 1.95 1.94 ...  
## $ n : chr "1" "2" "3" "4" ...  
## $ cum\_sum\_p : num 0.722 1.428 2.13 2.832 3.531 ...

summary(LSR.target)

## ID BinaryLogitProbability BinaryLogitPredict BinaryLogitActual  
## Min. :202.0 Min. :0.5174 Min. :1 Min. :0.0000   
## 1st Qu.:267.0 1st Qu.:0.5685 1st Qu.:1 1st Qu.:0.0000   
## Median :331.5 Median :0.6190 Median :1 Median :0.0000   
## Mean :342.0 Mean :0.6171 Mean :1 Mean :0.4688   
## 3rd Qu.:415.5 3rd Qu.:0.6799 3rd Qu.:1 3rd Qu.:1.0000   
## Max. :500.0 Max. :0.7220 Max. :1 Max. :1.0000   
## lift n cum\_sum\_p   
## Min. :1.437 Length:64 Min. : 0.722   
## 1st Qu.:1.579 Class :character 1st Qu.:11.609   
## Median :1.719 Mode :character Median :21.729   
## Mean :1.714 Mean :21.157   
## 3rd Qu.:1.889 3rd Qu.:31.014   
## Max. :2.006 Max. :39.494

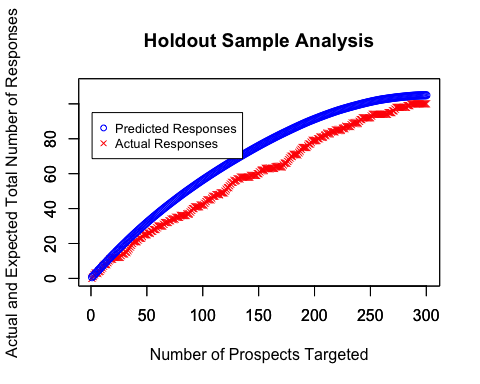
print("According to the Limited Supply Rule, the CD club should send invitation to prospects when the running sum is less than 40, that is, the 64 prospects with the highest predicted response rate.")

## [1] "According to the Limited Supply Rule, the CD club should send invitation to prospects when the running sum is less than 40, that is, the 64 prospects with the highest predicted response rate."

## 8. Compute the Cumulative Sum (aka running sum) for the Actual Response Rate (recall this is either 0 or 1) in decreasing order of Predicted Response Rate. Plot the curve for curve for number of Actual Positive Responses vs. Number of Prospects Targeted. Superimpose on this the curve obtained in step 6 above.

Using the chart, comment on the differences between the Actual Response Rates and the Predicted Response Rates for the prospects in the Testing Sample. What is the impact on your results in step 7?

# Calculate the running sum of Actual Response Rate  
prediction.test.sort$cum\_sum\_a <- cumsum(prediction.test.sort$BinaryLogitActual)  
  
# plot the curve for running sum of the actual response rate  
# plot the actual response curve  
plot(prediction.test.sort$cum\_sum\_a, ylim = c(0,110), col="red", pch = 4, xlab='', ylab='')  
# allow superimposition  
par(new = TRUE)  
# plot the predicted response curve and add the labels  
plot(prediction.test.sort$cum\_sum\_p, ylim=c(0,110), col = "blue", xlab="Number of Prospects Targeted", ylab="Actual and Expected Total Number of Responses", main="Holdout Sample Analysis")  
# add a legend  
legend(1, 95, legend=c("Predicted Responses", "Actual Responses"),  
 col=c("blue", "red"), pch=c(1,4), cex=0.8)



# Comment  
print("The model tends to over-predict. Therefore the CD shop will need to send invitation to more than 64 prospects in order to sell all of the 40 items. According to the actual response, the CD club should send out around 90 invitations.")

## [1] "The model tends to over-predict. Therefore the CD shop will need to send invitation to more than 64 prospects in order to sell all of the 40 items. According to the actual response, the CD club should send out around 90 invitations."

## Bonus: 1. Confusion Matrix

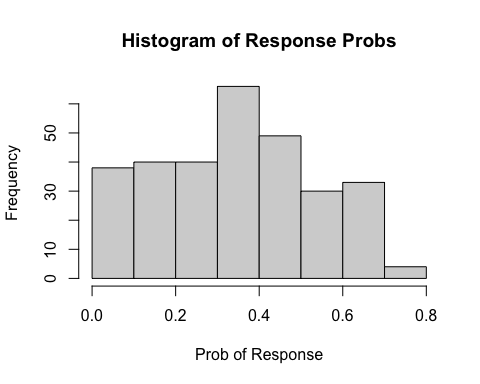
# Prediciton of Number of Buyers  
sum(prediction.test["BinaryLogitPredict"])

## [1] 67

sum(prediction.test["BinaryLogitProbability"])

## [1] 104.9432

# Histogram of Props  
hist(prediction.test$BinaryLogitProbability, main = paste("Histogram of Response Probs"), xlab = "Prob of Response")



# Confusion Matrix  
#install.packages("gmodels")  
library(gmodels)  
CrossTable(data.test$y, prediction.test$BinaryLogitPredict,prop.r=TRUE, prop.c=FALSE, prop.t=FALSE,  
 prop.chisq=FALSE, dnn = c("Real Response", "Predicted Response"))

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 300   
##   
##   
## | Predicted Response   
## Real Response | 0 | 1 | Row Total |   
## --------------|-----------|-----------|-----------|  
## 0 | 165 | 35 | 200 |   
## | 0.825 | 0.175 | 0.667 |   
## --------------|-----------|-----------|-----------|  
## 1 | 68 | 32 | 100 |   
## | 0.680 | 0.320 | 0.333 |   
## --------------|-----------|-----------|-----------|  
## Column Total | 233 | 67 | 300 |   
## --------------|-----------|-----------|-----------|  
##   
##

# Exporting the Predictions to Excel  
# You can open a csv file in xl  
write.csv(prediction.test, file = "Prediction\_Testing.csv")