# BBB - Logistic Regression

* Name: Vidya Chockalingam
* Gitlab id: rsm-vichocka
* Gitlab username: vichocka

## Setup

## make all required libraries available  
suppressPackageStartupMessages(library(radiant))

## Warning: package 'ggplot2' was built under R version 3.3.2

## Warning: package 'tidyr' was built under R version 3.3.2

library(dplyr)  
#install.packages("caret")  
library(caret)

## Warning: package 'caret' was built under R version 3.3.2

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:radiant.model':  
##   
## sensitivity

## finding the path to the class dropbox folder  
fp <- file.path(find\_dropbox(),"MGTA455-2017")  
  
## loading the data  
## DO NOT change the next line!  
## Data must be loaded from Dropbox/MGTA455-2017/data  
loadr(file.path(fp, "data/bbb.rda"))

## Question answers

### Part 1

Building a logistic regression model with buyer as the dependent variable and gender, last, total etc. as the independent variables.

#Part 1  
#1  
bbb$buyer\_int <- ifelse(bbb$buyer == "yes",1 ,0)   
response\_rate <- sum(bbb$buyer\_int)/nrow(bbb)  
model <- glm(buyer\_int ~ gender + last + total + child + youth + cook + do\_it +reference + art + geog,family=binomial(link='logit'),data=bbb)  
bbb$purch\_prob <- predict(model, newdata = bbb, type = "response")

All the p values are <0.05 and hence with a 95% confidence interval we can reject the null hypothesis of having the coefficients of the independent variables to be 0.

#2  
summary(model)

##   
## Call:  
## glm(formula = buyer\_int ~ gender + last + total + child + youth +   
## cook + do\_it + reference + art + geog, family = binomial(link = "logit"),   
## data = bbb)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.4031 -0.4129 -0.2807 -0.1839 3.2650   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.3608301 0.0492961 -47.891 < 0.0000000000000002 \*\*\*  
## genderM 0.7607204 0.0357608 21.272 < 0.0000000000000002 \*\*\*  
## last -0.0947124 0.0027924 -33.918 < 0.0000000000000002 \*\*\*  
## total 0.0011160 0.0001982 5.630 0.000000018 \*\*\*  
## child -0.1862162 0.0172824 -10.775 < 0.0000000000000002 \*\*\*  
## youth -0.1129745 0.0261087 -4.327 0.000015110 \*\*\*  
## cook -0.2703210 0.0171283 -15.782 < 0.0000000000000002 \*\*\*  
## do\_it -0.5391648 0.0269657 -19.994 < 0.0000000000000002 \*\*\*  
## reference 0.2346876 0.0265583 8.837 < 0.0000000000000002 \*\*\*  
## art 1.1555840 0.0221439 52.185 < 0.0000000000000002 \*\*\*  
## geog 0.5742763 0.0186311 30.824 < 0.0000000000000002 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 30355 on 49999 degrees of freedom  
## Residual deviance: 24122 on 49989 degrees of freedom  
## AIC: 24144  
##   
## Number of Fisher Scoring iterations: 6

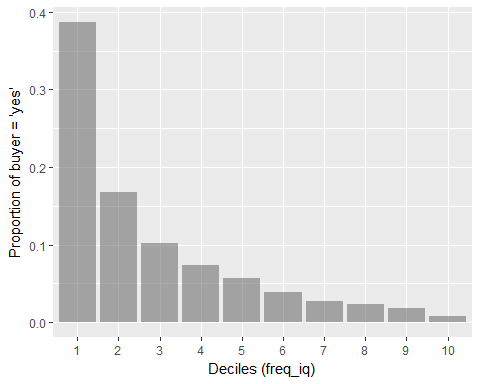
### Part 2

Splitting the data into deciles based on the predicted scores of the logit model.

# Part 2  
#1  
bbb$prob\_dec <- xtile(bbb$purch\_prob, 10, rev=TRUE)  
by\_decile <- group\_by(bbb,prob\_dec)  
avg\_score <- summarize(by\_decile,avg\_score=mean(purch\_prob))

Plotting the response rate across deciles - making sure that the highest response rate is in the first decile.

#2  
visualize(dataset = "bbb", xvar = "prob\_dec", yvar = "buyer\_int", type = "bar", custom = TRUE) + ylab("Proportion of buyer = 'yes'") + xlab("Deciles (freq\_iq)") + theme(legend.position = "none")



Storing the response rate across deciles in a table.

#3  
no.of.customers <- summarize(by\_decile,customer\_count=n())  
no.of.customers.bought <- summarize(by\_decile,customer\_count\_yes=sum(buyer\_int))  
response.rate <- summarize(by\_decile,response\_rate=sum(buyer\_int)/n())

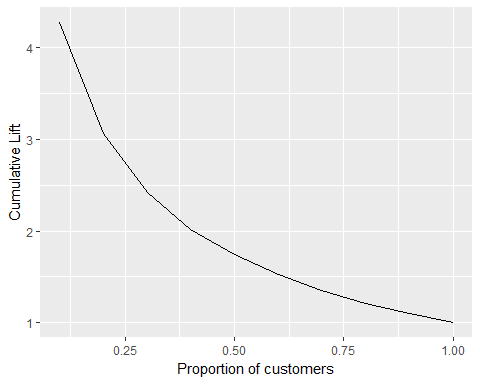
Creating another glm model with child as the only independent variable.

#4  
model1 <- glm(buyer\_int ~ child ,family=binomial(link='logit'),data=bbb)  
summary(model1)

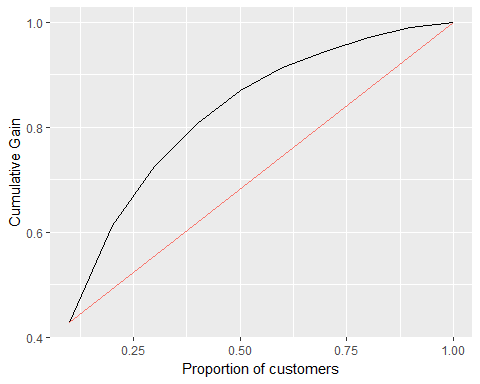
##   
## Call:  
## glm(formula = buyer\_int ~ child, family = binomial(link = "logit"),   
## data = bbb)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.5578 -0.4371 -0.4219 -0.4219 2.2197   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.37445 0.01988 -119.412 < 0.0000000000000002 \*\*\*  
## child 0.07406 0.01321 5.606 0.0000000207 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 30355 on 49999 degrees of freedom  
## Residual deviance: 30325 on 49998 degrees of freedom  
## AIC: 30329  
##   
## Number of Fisher Scoring iterations: 5

Creating Lift and Gain charts for the logit model.

# Part 3  
#1  
lift\_table <- cbind(no.of.customers,no.of.customers.bought,response.rate)  
lift\_table <- lift\_table[,c(-3,-5)]  
for(i in 1:nrow(lift\_table))  
{  
 if(i==1)  
 {  
 lift\_table$cum\_cust\_count[i] <- lift\_table[i,c("customer\_count")]  
 lift\_table$cum\_buyer\_count[i] <- lift\_table[i,c("customer\_count\_yes")]  
 }  
 else  
 {  
 lift\_table$cum\_cust\_count[i] <- lift\_table[i,c("customer\_count")] + lift\_table$cum\_cust\_count[i-1]  
 lift\_table$cum\_buyer\_count[i] <- lift\_table[i,c("customer\_count\_yes")] + lift\_table$cum\_buyer\_count[i-1]  
 }  
}  
lift\_table$cum\_prop <- lift\_table$cum\_cust\_count/nrow(bbb)  
lift\_table$lift <- lift\_table$response\_rate/response\_rate  
lift\_table$cum\_resp\_rate <- lift\_table$cum\_buyer\_count/lift\_table$cum\_cust\_count  
lift\_table$cum\_lift <- lift\_table$cum\_resp\_rate/response\_rate  
#2  
visualize(dataset = "lift\_table", xvar = "cum\_prop", yvar = "cum\_lift", type = "line", custom = TRUE) + ylab("Cumulative Lift") + xlab("Proportion of customers") + theme(legend.position = "none")

 Gain Chart

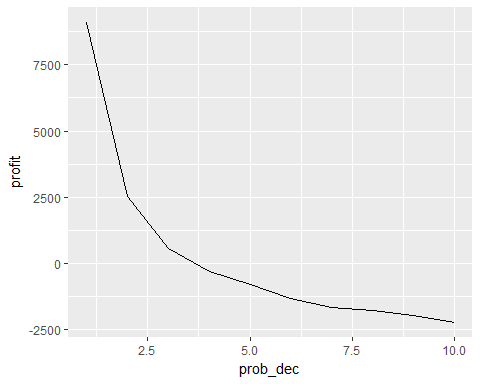
#3  
gain\_table <- lift\_table  
gain\_table$buyer\_prop <- gain\_table$customer\_count\_yes/sum(gain\_table$customer\_count\_yes)  
for(i in 1:nrow(gain\_table))  
{  
 if(i==1)  
 {  
 gain\_table$cum\_buyer\_prop[i] <- gain\_table[i,c("buyer\_prop")]  
 }  
 else  
 {  
 gain\_table$cum\_buyer\_prop[i] <- gain\_table[i,c("buyer\_prop")] + gain\_table$cum\_buyer\_prop[i-1]  
 }  
}  
#4  
visualize(dataset = "gain\_table", xvar = "cum\_prop", yvar = "cum\_buyer\_prop", type = "line", custom = TRUE) + geom\_segment(aes(x = min(cum\_prop), y = min(cum\_buyer\_prop), xend = max(cum\_prop), yend = max(cum\_buyer\_prop), colour = "segment"), data = gain\_table) + ylab("Cumulative Gain") + xlab("Proportion of customers") + theme(legend.position = "none")



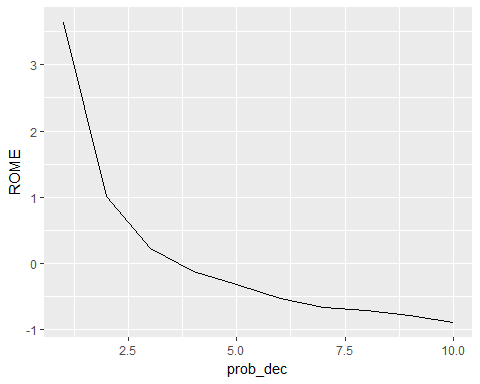
### Part 4

Creating the Profit and ROME curve using the predictions by the logit model.

#Part 4  
mail <- 0.50  
price <- 9  
shipping <- 3  
selling\_price <- 18  
#1  
lift\_table$profit <- (selling\_price-price-shipping)\*lift\_table$customer\_count\_yes - mail\*lift\_table$customer\_count  
lift\_table$ROME <- lift\_table$profit/(mail\*lift\_table$customer\_count)  
#2  
ggplot(lift\_table,aes(x=prob\_dec,y=profit,group=1)) + geom\_line()



#3  
ggplot(lift\_table,aes(x=prob\_dec,y=ROME,group=1)) + geom\_line()



Creating the confusion matrix for the above model.

#4  
bbb$prediction <- ifelse(bbb$purch\_prob>=0.5, 1, 0)  
confusion\_table <- table(bbb$prediction,bbb$buyer\_int)  
print(confusion\_table)

##   
## 0 1  
## 0 45126 3838  
## 1 352 684

confusion\_matrix <- confusionMatrix(confusion\_table)  
confusion\_matrix$overall['Accuracy']

## Accuracy   
## 0.9162

### Part 5

Calculating the expected number of buyers, reponse rate of the responders, profit and ROME if we target only the customers who have a purchase probability > break even rate.

#Part 5  
#1  
break\_even\_rate <- mail/(selling\_price - shipping - price)  
#2  
bbb$mailto\_logit <- ifelse(bbb$purch\_prob>break\_even\_rate,1,0)  
#3  
no.of.customers.to.target <- mean(bbb$mailto\_logit)  
no.of.customers <- 500000 \* no.of.customers.to.target  
print(paste("Targeted Customers ",no.of.customers))

## [1] "Targeted Customers 155600"

bbb\_responders <- bbb[bbb$mailto\_logit == 1,]  
response\_rate\_responders <- sum(bbb\_responders$buyer\_int)/nrow(bbb\_responders)  
print(paste("Response Rate of the Responders ",response\_rate\_responders))

## [1] "Response Rate of the Responders 0.213560411311054"

expected.buyers <- round(no.of.customers \* response\_rate\_responders)  
print(paste("Expected number of buyers ",expected.buyers))

## [1] "Expected number of buyers 33230"

#4  
profit <- (selling\_price-shipping-price)\*expected.buyers - mail\*no.of.customers  
print(paste("Profit if we target only the customers with purchase probability > break even rate is $",profit))

## [1] "Profit if we target only the customers with purchase probability > break even rate is $ 121580"

ROME\_targeted <- profit/(mail\*no.of.customers)  
ROME\_targeted

## [1] 1.562725

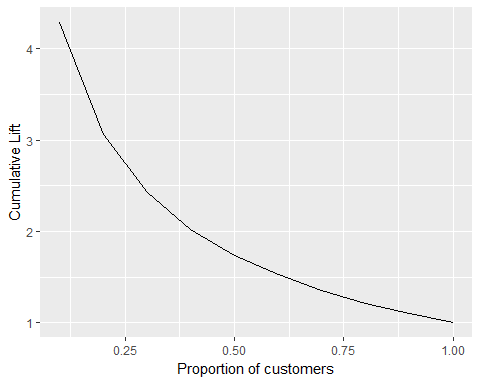
## Logistic Regression Accounting for Standard Error

If we predict logistic regression accounting for standard error in the prediction of the parameters, then we find that there is not much variation in the ROME, accuracy and profit.

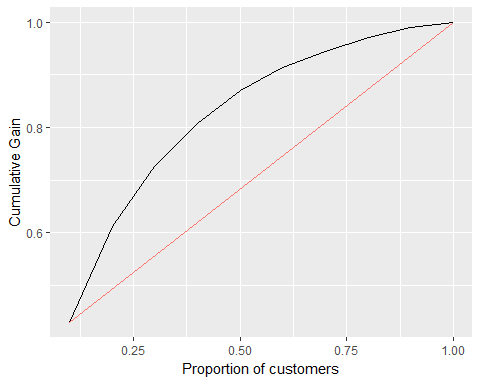
#5 Logistic with Error  
loadr(file.path(fp, "data/bbb.rda"))  
bbb\_ler <- bbb  
bbb\_ler$buyer\_int <- ifelse(bbb\_ler$buyer == "yes",1 ,0)   
bbb\_ler$purch\_prob <- predict(model, newdata = bbb\_ler, type = "response",se = TRUE)$fit - predict(model, newdata = bbb\_ler, type = "response",se = TRUE)$se.fit  
bbb\_ler$prob\_dec <- xtile(bbb\_ler$purch\_prob, 10, rev=TRUE)  
  
by\_decile <- group\_by(bbb\_ler,prob\_dec)  
no.of.customers <- summarize(by\_decile,customer\_count=n())  
no.of.customers.bought <- summarize(by\_decile,customer\_count\_yes=sum(buyer\_int))  
response.rate <- summarize(by\_decile,response\_rate=sum(buyer\_int)/n())

Lift and Gains Table for Logistic regression with standard error.

# Part 3  
#1  
lift\_table <- cbind(no.of.customers,no.of.customers.bought,response.rate)  
lift\_table <- lift\_table[,c(-3,-5)]  
for(i in 1:nrow(lift\_table))  
{  
 if(i==1)  
 {  
 lift\_table$cum\_cust\_count[i] <- lift\_table[i,c("customer\_count")]  
 lift\_table$cum\_buyer\_count[i] <- lift\_table[i,c("customer\_count\_yes")]  
 }  
 else  
 {  
 lift\_table$cum\_cust\_count[i] <- lift\_table[i,c("customer\_count")] + lift\_table$cum\_cust\_count[i-1]  
 lift\_table$cum\_buyer\_count[i] <- lift\_table[i,c("customer\_count\_yes")] + lift\_table$cum\_buyer\_count[i-1]  
 }  
}  
lift\_table$cum\_prop <- lift\_table$cum\_cust\_count/nrow(bbb)  
lift\_table$lift <- lift\_table$response\_rate/response\_rate  
lift\_table$cum\_resp\_rate <- lift\_table$cum\_buyer\_count/lift\_table$cum\_cust\_count  
lift\_table$cum\_lift <- lift\_table$cum\_resp\_rate/response\_rate  
#2  
visualize(dataset = "lift\_table", xvar = "cum\_prop", yvar = "cum\_lift", type = "line", custom = TRUE) + ylab("Cumulative Lift") + xlab("Proportion of customers") + theme(legend.position = "none")

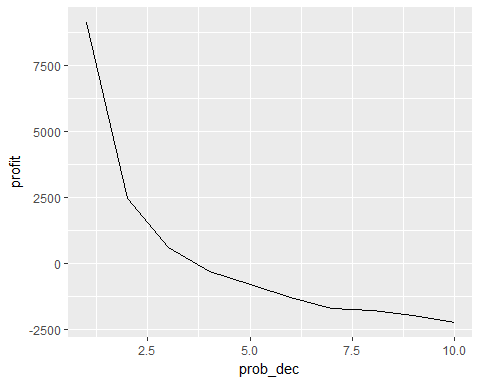


#3  
gain\_table <- lift\_table  
gain\_table$buyer\_prop <- gain\_table$customer\_count\_yes/sum(gain\_table$customer\_count\_yes)  
for(i in 1:nrow(gain\_table))  
{  
 if(i==1)  
 {  
 gain\_table$cum\_buyer\_prop[i] <- gain\_table[i,c("buyer\_prop")]  
 }  
 else  
 {  
 gain\_table$cum\_buyer\_prop[i] <- gain\_table[i,c("buyer\_prop")] + gain\_table$cum\_buyer\_prop[i-1]  
 }  
}  
#4  
visualize(dataset = "gain\_table", xvar = "cum\_prop", yvar = "cum\_buyer\_prop", type = "line", custom = TRUE) + geom\_segment(aes(x = min(cum\_prop), y = min(cum\_buyer\_prop), xend = max(cum\_prop), yend = max(cum\_buyer\_prop), colour = "segment"), data = gain\_table) + ylab("Cumulative Gain") + xlab("Proportion of customers") + theme(legend.position = "none")

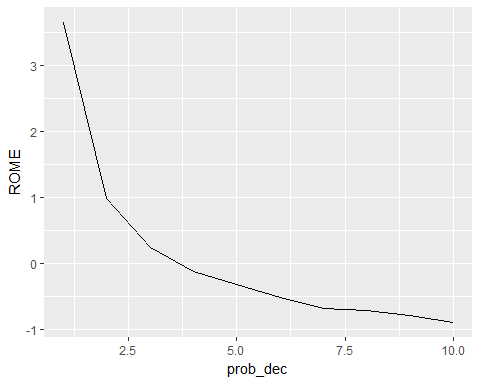


Profit and ROME curve for Logistic Regression with Standard Error

#Part 4  
#1  
lift\_table$profit <- (selling\_price-price-shipping)\*lift\_table$customer\_count\_yes - mail\*lift\_table$customer\_count  
lift\_table$ROME <- lift\_table$profit/(mail\*lift\_table$customer\_count)  
#2  
ggplot(lift\_table,aes(x=prob\_dec,y=profit,group=1)) + geom\_line()



#3  
ggplot(lift\_table,aes(x=prob\_dec,y=ROME,group=1)) + geom\_line()



Accuracy for this model.

#4  
bbb\_ler$prediction <- ifelse(bbb\_ler$purch\_prob>=0.5, 1, 0)  
confusion\_table <- table(bbb\_ler$prediction,bbb\_ler$buyer\_int)  
print(confusion\_table)

##   
## 0 1  
## 0 45167 3881  
## 1 311 641

confusion\_matrix <- confusionMatrix(confusion\_table)  
confusion\_matrix$overall['Accuracy']

## Accuracy   
## 0.91616

#2  
bbb\_ler$mailto\_logit <- ifelse(bbb\_ler$purch\_prob>break\_even\_rate,1,0)  
#3  
no.of.customers.to.target <- mean(bbb\_ler$mailto\_logit)  
no.of.customers <- 500000 \* no.of.customers.to.target  
print(paste("Number of customers to target", no.of.customers))

## [1] "Number of customers to target 148700"

bbb\_responders <- bbb\_ler[bbb\_ler$mailto\_logit == 1,]  
response\_rate\_responders <- sum(bbb\_responders$buyer\_int)/nrow(bbb\_responders)  
print(paste("Response rate expected",response\_rate\_responders))

## [1] "Response rate expected 0.220107599193006"

expected.buyers <- round(no.of.customers \* response\_rate\_responders)  
print(paste("Expected Number of Buyers",expected.buyers))

## [1] "Expected Number of Buyers 32730"

#4  
profit <- (selling\_price-shipping-price)\*expected.buyers - mail\*no.of.customers  
ROME\_targeted <- profit/(mail\*no.of.customers)  
print(paste("Profit if we target only the customers with purchase probability > break even rate is $",profit))

## [1] "Profit if we target only the customers with purchase probability > break even rate is $ 122030"

print(paste("ROME for these targeted customers",ROME\_targeted))

## [1] "ROME for these targeted customers 1.64129119031607"

## Sequential RFM

Creating quintiles based on the recency(last), frequency(purch), monetary(total) values of the dataset.

# Sequential RFM  
bbb\_sq <- bbb  
bbb\_sq$buyer\_int <- ifelse(bbb$buyer == "yes",1 ,0)   
response\_rate <- sum(bbb\_sq$buyer\_int)/nrow(bbb\_sq)  
bbb\_sq$rec\_iq <- xtile(bbb\_sq$last, 5)  
bbb\_sq <- group\_by(bbb\_sq, rec\_iq) %>% mutate(freq\_sq = xtile(purch, 5, rev = TRUE)) %>% ungroup  
  
## create new variable(s)  
bbb\_sq <- group\_by(bbb\_sq, rec\_iq, freq\_sq) %>% mutate(mon\_sq = xtile(total, 5, rev = TRUE)) %>% ungroup  
bbb\_sq$rfm\_sq <- paste0(bbb\_sq$rec\_iq,bbb\_sq$freq\_sq,bbb\_sq$mon\_sq)

Finding the profit and ROME of the customers in the cells which has a response rate > break even rate.

bbb\_sq <- group\_by(bbb\_sq, rfm\_sq) %>% mutate(mailto\_sq = mean(buyer == "yes") > break\_even\_rate) %>% ungroup  
perc\_mail = mean(bbb\_sq$mailto\_sq)  
nr\_mail = 500000 \* perc\_mail  
rep\_rate = filter(bbb\_sq, mailto\_sq == TRUE) %>% summarize(mean(buyer == "yes")) %>% unlist  
nr\_resp = nr\_mail \* rep\_rate  
mail\_cost = mail \* nr\_mail  
profit\_sq = (selling\_price-shipping-price) \* nr\_resp - mail\_cost  
print(paste("Profit if we target only the customers with purchase probability > break even rate is $",profit\_sq))

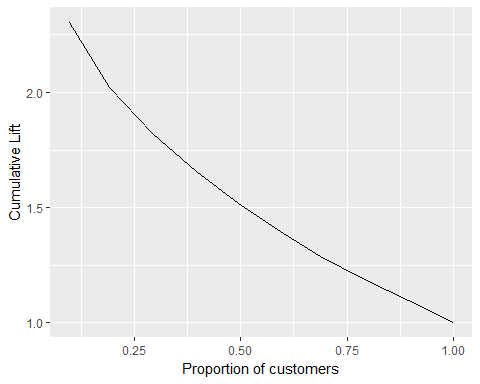
## [1] "Profit if we target only the customers with purchase probability > break even rate is $ 80100"

ROME\_sq = profit\_sq / mail\_cost  
print(paste("ROME ",ROME\_sq))

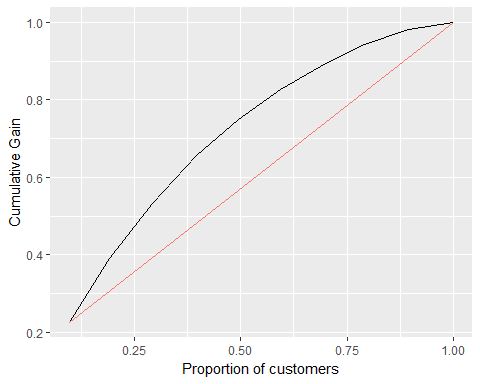
## [1] "ROME 0.675949367088608"

Creating Lift and Gain Charts for Sequential RFM based on the response rate per rfm cell.

bbb\_sq <- group\_by(bbb\_sq, rfm\_sq) %>% mutate(response\_rate = mean(buyer == "yes")) %>% ungroup %>% mutate(prob\_dec = xtile(response\_rate,10, rev=TRUE)) %>% ungroup  
by\_decile <- group\_by(bbb\_sq,prob\_dec)  
no.of.customers <- summarize(by\_decile,customer\_count=n())  
no.of.customers.bought <- summarize(by\_decile,customer\_count\_yes=sum(buyer\_int))  
response.rate <- summarize(by\_decile,response\_rate=sum(buyer\_int)/n())  
lift\_table <- cbind(no.of.customers,no.of.customers.bought,response.rate)  
lift\_table <- lift\_table[,c(-3,-5)]  
for(i in 1:nrow(lift\_table))  
{  
 if(i==1)  
 {  
 lift\_table$cum\_cust\_count[i] <- lift\_table[i,c("customer\_count")]  
 lift\_table$cum\_buyer\_count[i] <- lift\_table[i,c("customer\_count\_yes")]  
 }  
 else  
 {  
 lift\_table$cum\_cust\_count[i] <- lift\_table[i,c("customer\_count")] + lift\_table$cum\_cust\_count[i-1]  
 lift\_table$cum\_buyer\_count[i] <- lift\_table[i,c("customer\_count\_yes")] + lift\_table$cum\_buyer\_count[i-1]  
 }  
}  
lift\_table$cum\_prop <- lift\_table$cum\_cust\_count/nrow(bbb)  
lift\_table$lift <- lift\_table$response\_rate/response\_rate  
lift\_table$cum\_resp\_rate <- lift\_table$cum\_buyer\_count/lift\_table$cum\_cust\_count  
lift\_table$cum\_lift <- lift\_table$cum\_resp\_rate/response\_rate  
#2  
visualize(dataset = "lift\_table", xvar = "cum\_prop", yvar = "cum\_lift", type = "line", custom = TRUE) + ylab("Cumulative Lift") + xlab("Proportion of customers") + theme(legend.position = "none")

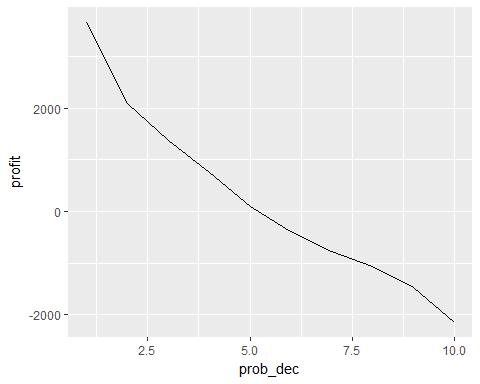


#3  
gain\_table <- lift\_table  
gain\_table$buyer\_prop <- gain\_table$customer\_count\_yes/sum(gain\_table$customer\_count\_yes)  
for(i in 1:nrow(gain\_table))  
{  
 if(i==1)  
 {  
 gain\_table$cum\_buyer\_prop[i] <- gain\_table[i,c("buyer\_prop")]  
 }  
 else  
 {  
 gain\_table$cum\_buyer\_prop[i] <- gain\_table[i,c("buyer\_prop")] + gain\_table$cum\_buyer\_prop[i-1]  
 }  
}  
visualize(dataset = "gain\_table", xvar = "cum\_prop", yvar = "cum\_buyer\_prop", type = "line", custom = TRUE) +geom\_segment(aes(x = min(cum\_prop), y = min(cum\_buyer\_prop), xend = max(cum\_prop), yend = max(cum\_buyer\_prop), colour = "segment"), data = gain\_table) + ylab("Cumulative Gain") + xlab("Proportion of customers") + theme(legend.position = "none")

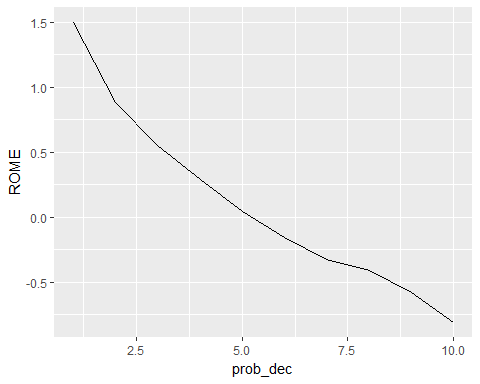


Creating Profit and ROME curves for sequential RFM.

lift\_table$profit <- (selling\_price-price-shipping)\*lift\_table$customer\_count\_yes - mail\*lift\_table$customer\_count  
lift\_table$ROME <- lift\_table$profit/(mail\*lift\_table$customer\_count)  
#2  
ggplot(lift\_table,aes(x=prob\_dec,y=profit,group=1)) + geom\_line()



#3  
ggplot(lift\_table,aes(x=prob\_dec,y=ROME,group=1)) + geom\_line()



Confusion Matrix and Accuracy for sequential RFM is calculated by taking the prediction scores for all the customers in a RFM cell with response rate > break even rate as 1. Then we compare it with the actual response rate of the customers in those cells.

#Confusion matrix  
bbb\_sq$buyer\_int <- ifelse(bbb\_sq$buyer == "yes",1 ,0)   
bbb\_sq$mailto\_sq <- ifelse(bbb\_sq$mailto\_sq == 1, 1, 0)  
confusion\_table <- table(bbb\_sq$mailto\_sq,bbb\_sq$buyer\_int)  
print(confusion\_table)

##   
## 0 1  
## 0 25088 1212  
## 1 20390 3310

confusion\_matrix <- confusionMatrix(confusion\_table)  
confusion\_matrix$overall['Accuracy']

## Accuracy   
## 0.56796

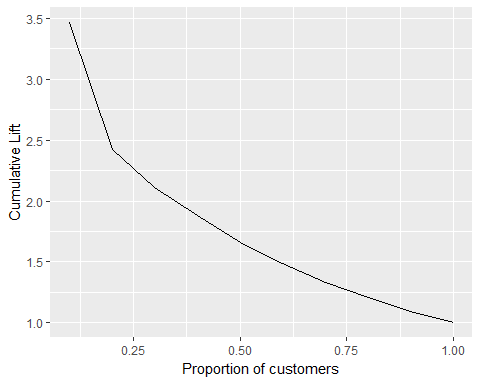
## Naive Bayes

Calculating predictions based on Naive Bayes Model.

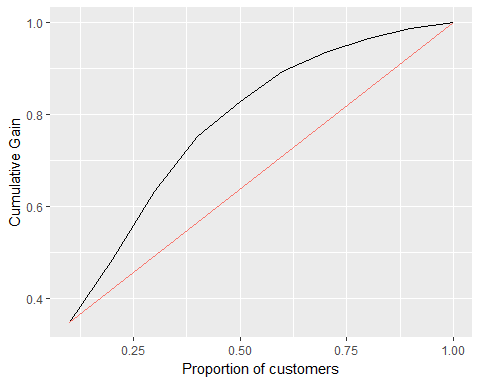
## Naive Bayes  
bbb\_nb <- bbb  
bbb\_nb$buyer\_int <- ifelse(bbb\_nb$buyer == "yes",1 ,0)   
response\_rate <- sum(bbb\_nb$buyer\_int)/nrow(bbb\_nb)  
levels(bbb\_nb$buyer) <- c(0,1)  
nb\_model <- nb(dataset = "bbb\_nb",rvar = "buyer", evar = c("gender","last","total","child","youth","cook","do\_it", "reference", "art", "geog"))  
bbb\_nb$purch\_prob <- predict(nb\_model, pred\_data = "bbb\_nb")[,"0"]

Plotting Lift and Gain Charts for Naive Bayes

# Part 2  
#1  
bbb\_nb$prob\_dec <- xtile(bbb\_nb$purch\_prob, 10, rev=TRUE)  
by\_decile <- group\_by(bbb\_nb,prob\_dec)  
no.of.customers <- summarize(by\_decile,customer\_count=n())  
no.of.customers.bought <- summarize(by\_decile,customer\_count\_yes=sum(buyer\_int))  
response.rate <- summarize(by\_decile,response\_rate=sum(buyer\_int)/n())  
lift\_table <- cbind(no.of.customers,no.of.customers.bought,response.rate)  
lift\_table <- lift\_table[,c(-3,-5)]  
for(i in 1:nrow(lift\_table))  
{  
 if(i==1)  
 {  
 lift\_table$cum\_cust\_count[i] <- lift\_table[i,c("customer\_count")]  
 lift\_table$cum\_buyer\_count[i] <- lift\_table[i,c("customer\_count\_yes")]  
 }  
 else  
 {  
 lift\_table$cum\_cust\_count[i] <- lift\_table[i,c("customer\_count")] + lift\_table$cum\_cust\_count[i-1]  
 lift\_table$cum\_buyer\_count[i] <- lift\_table[i,c("customer\_count\_yes")] + lift\_table$cum\_buyer\_count[i-1]  
 }  
}  
lift\_table$cum\_prop <- lift\_table$cum\_cust\_count/nrow(bbb)  
lift\_table$lift <- lift\_table$response\_rate/response\_rate  
lift\_table$cum\_resp\_rate <- lift\_table$cum\_buyer\_count/lift\_table$cum\_cust\_count  
lift\_table$cum\_lift <- lift\_table$cum\_resp\_rate/response\_rate  
#2  
visualize(dataset = "lift\_table", xvar = "cum\_prop", yvar = "cum\_lift", type = "line", custom = TRUE) + ylab("Cumulative Lift") + xlab("Proportion of customers") + theme(legend.position = "none")

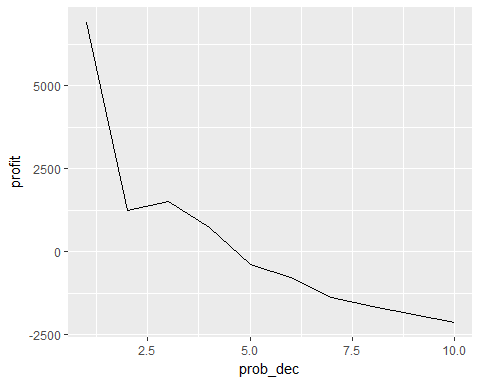


#3  
gain\_table <- lift\_table  
gain\_table$buyer\_prop <- gain\_table$customer\_count\_yes/sum(gain\_table$customer\_count\_yes)  
for(i in 1:nrow(gain\_table))  
{  
 if(i==1)  
 {  
 gain\_table$cum\_buyer\_prop[i] <- gain\_table[i,c("buyer\_prop")]  
 }  
 else  
 {  
 gain\_table$cum\_buyer\_prop[i] <- gain\_table[i,c("buyer\_prop")] + gain\_table$cum\_buyer\_prop[i-1]  
 }  
}  
#4  
visualize(dataset = "gain\_table", xvar = "cum\_prop", yvar = "cum\_buyer\_prop", type = "line", custom = TRUE) + geom\_segment(aes(x = min(cum\_prop), y = min(cum\_buyer\_prop), xend = max(cum\_prop), yend = max(cum\_buyer\_prop), colour = "segment"), data = gain\_table) + ylab("Cumulative Gain") + xlab("Proportion of customers") + theme(legend.position = "none")

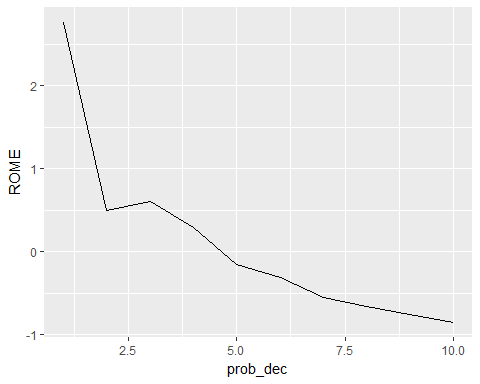


Plotting ROME and Profit curves across deciles in Naive Bayes.

#1  
lift\_table$profit <- (selling\_price-price-shipping)\*lift\_table$customer\_count\_yes - mail\*lift\_table$customer\_count  
lift\_table$ROME <- lift\_table$profit/(mail\*lift\_table$customer\_count)  
#2  
ggplot(lift\_table,aes(x=prob\_dec,y=profit,group=1)) + geom\_line()



#3  
ggplot(lift\_table,aes(x=prob\_dec,y=ROME,group=1)) + geom\_line()



Creating the confusion matrix and accuracy using Naive Bayes Model.

#4  
bbb\_nb$prediction <- ifelse(bbb\_nb$purch\_prob>=0.5, 1, 0)  
confusion\_table <- table(bbb\_nb$prediction,bbb\_nb$buyer\_int)  
print(confusion\_table)

##   
## 0 1  
## 0 42230 2991  
## 1 3248 1531

confusion\_matrix <- confusionMatrix(confusion\_table)  
confusion\_matrix$overall['Accuracy']

## Accuracy   
## 0.87522

Calculating the profit and ROME using Naive Bayes.

bbb\_nb$mailto\_logit <- ifelse(bbb\_nb$purch\_prob>break\_even\_rate,1,0)  
#3  
no.of.customers.to.target <- mean(bbb\_nb$mailto\_logit)  
no.of.customers <- 500000 \* no.of.customers.to.target  
bbb\_responders <- bbb\_nb[bbb\_nb$mailto\_logit == 1,]  
response\_rate\_responders <- sum(bbb\_responders$buyer\_int)/nrow(bbb\_responders)  
expected.buyers <- round(no.of.customers \* response\_rate\_responders)  
  
#4  
profit <- (selling\_price-shipping-price)\*expected.buyers - mail\*no.of.customers  
ROME\_targeted <- profit/(mail\*no.of.customers)  
print(paste("Profit is $", profit))

## [1] "Profit is $ 87310"

ROME\_targeted

## [1] 1.465671

print(paste("ROME", ROME\_targeted))

## [1] "ROME 1.46567063958368"