Stat 412

Yolanda Jin

24/11/2021

Contents

0. Cleaning Up Data and Creating Data Groups

a) Remove NA Data

- Split out data with NA in any predictors
- split non-NA group to minority (default) vs majority (non-default) group
- Additional TODO: can check out characteristics of NA group so we can replace the NA values

```
# Read Credit Scoring Data Training Set
#cs_train <- cs_training
cs_train = read.csv("cs-training.csv")
train <- cs_train
raw_data <- cs_train    #150k rows

# Remove NA cases otherwise cannot predict
cs_train_omit <- na.omit(raw_data)  # 150k -> 120,269 obs
Predictor_Variables <- subset.data.frame(cs_train_omit, select = c(RevolvingUtilizationOfUnsecuredLines
))

summary(cs_train)
```

```
##        X          SeriousDlqin2yrs  RevolvingUtilizationOfUnsecuredLines
##  Min.   :     1   Min.   :0.00000   Min.   :    0.00
##  1st Qu.: 37501   1st Qu.:0.00000   1st Qu.:    0.03
##  Median : 75001   Median :0.00000   Median :    0.15
##  Mean   : 75001   Mean   :0.06684   Mean   :    6.05
##  3rd Qu.:112500   3rd Qu.:0.00000   3rd Qu.:    0.56
##  Max.   :150000   Max.   :1.00000   Max.   :50708.00
##
##       age        NumberOfTime30.59DaysPastDueNotWorse   DebtRatio
##  Min.   :  0.0   Min.   : 0.000                       Min.   :     0.0
##  1st Qu.: 41.0   1st Qu.: 0.000                       1st Qu.:     0.2
##  Median : 52.0   Median : 0.000                       Median :     0.4
##  Mean   : 52.3   Mean   : 0.421                       Mean   :   353.0
##  3rd Qu.: 63.0   3rd Qu.: 0.000                       3rd Qu.:     0.9
##  Max.   :109.0   Max.   :98.000                       Max.   :329664.0
##
##  MonthlyIncome     NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
##  Min.   :      0   Min.   : 0.000                  Min.   : 0.000
##  1st Qu.:   3400   1st Qu.: 5.000                  1st Qu.: 0.000
##  Median :   5400   Median : 8.000                  Median : 0.000
##  Mean   :   6670   Mean   : 8.453                  Mean   : 0.266
##  3rd Qu.:   8249   3rd Qu.:11.000                  3rd Qu.: 0.000
```

```
##  Max.   :3008750   Max.   :58.000                  Max.   :98.000
##  NA's   :29731
##  NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
##  Min.   : 0.000               Min.   : 0.0000
##  1st Qu.: 0.000               1st Qu.: 0.0000
##  Median : 1.000               Median : 0.0000
##  Mean   : 1.018               Mean   : 0.2404
##  3rd Qu.: 2.000               3rd Qu.: 0.0000
##  Max.   :54.000               Max.   :98.0000
##
##  NumberOfDependents
##  Min.   : 0.000
##  1st Qu.: 0.000
##  Median : 0.000
##  Mean   : 0.757
##  3rd Qu.: 1.000
##  Max.   :20.000
##  NA's   :3924
```

b) Split the data to Testing/Training & Minority, Majority Groups

```
# Sample 60% of data for training Purpose
set.seed(1)
n = nrow(cs_train_omit)
na.idx = raw_data$X[-cs_train_omit$X]  # indexes of data with NA values that we removed
n.idx = sample(n, n*0.6) # Indexes for test train split

cs_train = cs_train_omit[n.idx,]  # Training data 72,161 obs
cs_test = cs_train_omit[-n.idx,]  # Testing data  48,108 obs.
cs_NA = raw_data[na.idx,]  # data with NA value  29,731 obs


## 1. Separate minority data vs majority data vs NA data total 72,161 obs
cs_train_min <- cs_train[cs_train$SeriousDlqin2yrs==1,]  # (omit) 10,026 -> (training) 5,009 obs
cs_train_maj <- cs_train[cs_train$SeriousDlqin2yrs==0,]  # (omit) 139,974 -> (training) 67,152 obs

str(cs_train)
```

```
## 'data.frame':    72161 obs. of  12 variables:
##  $ X                                   : int  30484 74216 54077 86784 14388 31442 40844 145293 17360
##  $ SeriousDlqin2yrs                    : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ RevolvingUtilizationOfUnsecuredLines: num  0.09474 0.05277 0.73665 0.02366 0.00582 ...
##  $ age                                 : int  47 68 36 36 44 40 59 65 50 49 ...
##  $ NumberOfTime30.59DaysPastDueNotWorse: int  0 0 0 0 0 1 0 0 0 0 ...
##  $ DebtRatio                           : num  0.3684 0.2276 0.0945 0.2171 0.1853 ...
##  $ MonthlyIncome                       : int  6250 11884 2000 5600 5100 2946 3950 10500 18381 10796
##  $ NumberOfOpenCreditLinesAndLoans     : int  8 13 6 9 24 17 7 23 10 7 ...
##  $ NumberOfTimes90DaysLate             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NumberRealEstateLoansOrLines        : int  1 2 0 1 1 3 0 2 1 2 ...
##  $ NumberOfTime60.89DaysPastDueNotWorse: int  0 0 0 0 0 0 0 0 1 0 ...
##  $ NumberOfDependents                  : int  3 1 1 0 2 0 0 0 1 0 ...
##  - attr(*, "na.action")= 'omit' Named int [1:29731] 7 9 17 33 42 53 59 63 72 87 ...
```

```
##   ..- attr(*, "names")= chr [1:29731] "7" "9" "17" "33" ...
```

```r
str(cs_train_min)
```

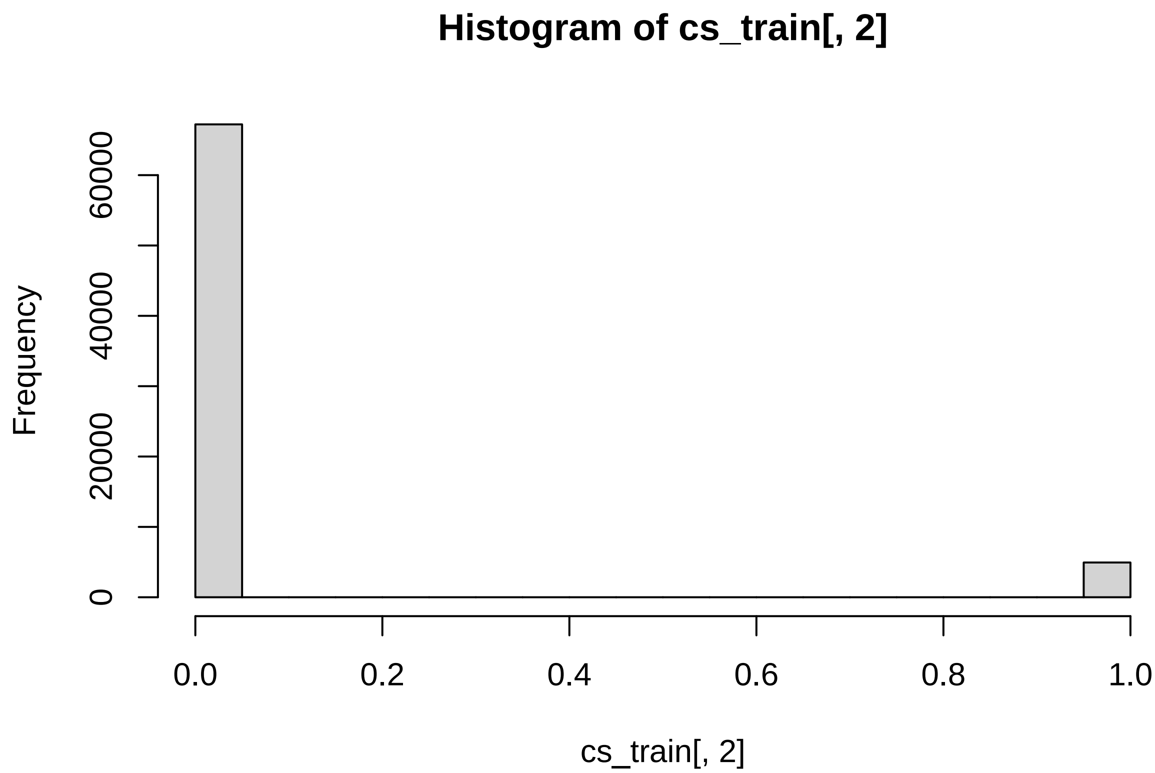
```
## 'data.frame':    4941 obs. of  12 variables:
##  $ X                                   : int  17360 90565 146254 102792 49072 142766 73164 75761 7788
##  $ SeriousDlqin2yrs                    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ RevolvingUtilizationOfUnsecuredLines: num  0.667 0.942 1 1.026 0.659 ...
##  $ age                                 : int  50 59 28 30 43 37 46 47 37 42 ...
##  $ NumberOfTime30.59DaysPastDueNotWorse: int  0 1 98 1 0 0 0 0 98 0 ...
##  $ DebtRatio                           : num  0.103 0.1653 0 0.2348 0.0897 ...
##  $ MonthlyIncome                       : int  18381 8008 1664 2763 3900 16666 4200 11000 6000 6450 .
##  $ NumberOfOpenCreditLinesAndLoans     : int  10 6 0 9 4 22 7 14 0 12 ...
##  $ NumberOfTimes90DaysLate             : int  0 5 98 0 2 0 0 0 98 0 ...
##  $ NumberRealEstateLoansOrLines        : int  1 0 0 0 0 3 1 1 0 2 ...
##  $ NumberOfTime60.89DaysPastDueNotWorse: int  1 0 98 2 1 0 1 0 98 0 ...
##  $ NumberOfDependents                  : int  1 0 0 3 2 0 1 0 1 0 ...
##  - attr(*, "na.action")= 'omit' Named int [1:29731] 7 9 17 33 42 53 59 63 72 87 ...
##   ..- attr(*, "names")= chr [1:29731] "7" "9" "17" "33" ...
```

```r
str(cs_train_maj)
```

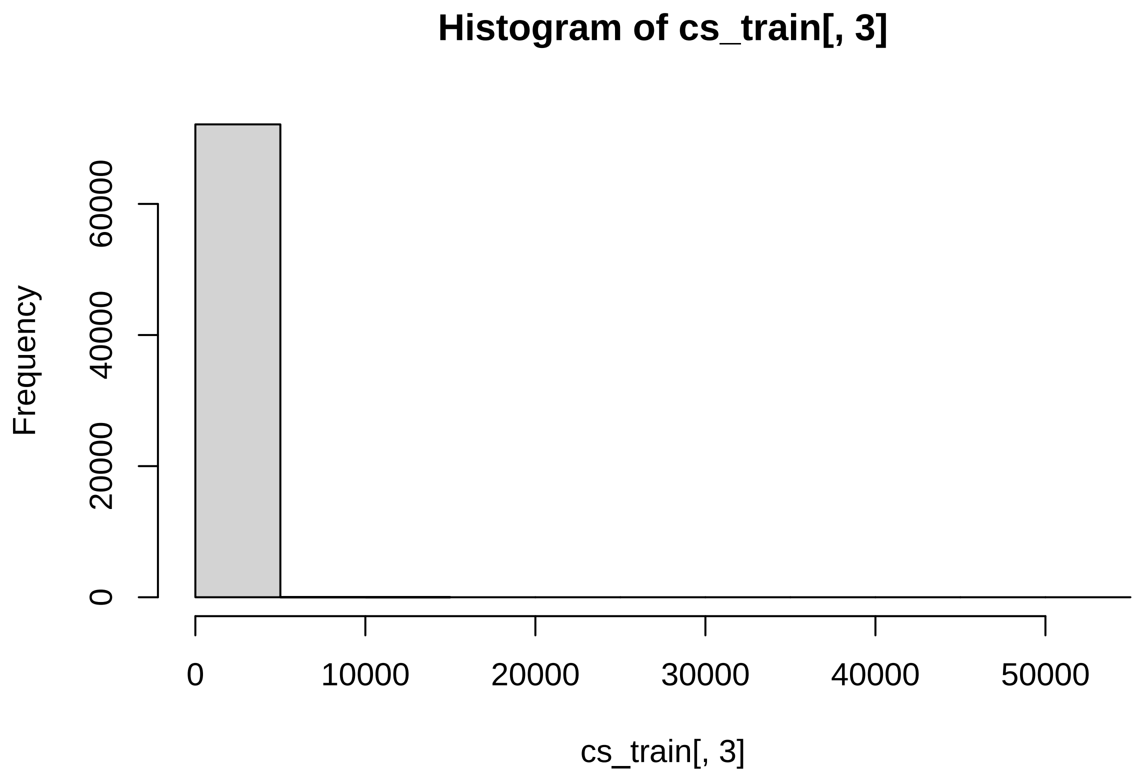
```
## 'data.frame':    67220 obs. of  12 variables:
##  $ X                                   : int  30484 74216 54077 86784 14388 31442 40844 145293 10260
##  $ SeriousDlqin2yrs                    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ RevolvingUtilizationOfUnsecuredLines: num  0.09474 0.05277 0.73665 0.02366 0.00582 ...
##  $ age                                 : int  47 68 36 36 44 40 59 65 49 53 ...
##  $ NumberOfTime30.59DaysPastDueNotWorse: int  0 0 0 0 0 1 0 0 0 0 ...
##  $ DebtRatio                           : num  0.3684 0.2276 0.0945 0.2171 0.1853 ...
##  $ MonthlyIncome                       : int  6250 11884 2000 5600 5100 2946 3950 10500 10796 3741 .
##  $ NumberOfOpenCreditLinesAndLoans     : int  8 13 6 9 24 17 7 23 7 7 ...
##  $ NumberOfTimes90DaysLate             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NumberRealEstateLoansOrLines        : int  1 2 0 1 1 3 0 2 2 2 ...
##  $ NumberOfTime60.89DaysPastDueNotWorse: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NumberOfDependents                  : int  3 1 1 0 2 0 0 0 0 1 ...
##  - attr(*, "na.action")= 'omit' Named int [1:29731] 7 9 17 33 42 53 59 63 72 87 ...
##   ..- attr(*, "names")= chr [1:29731] "7" "9" "17" "33" ...
```

```r
# Split x and y variables
train.x = cs_train[,-which(names(cs_train) == "SeriousDlqin2yrs")]  # 72,161 obs of 10 var
train.x = cs_train[,-c(which(names(cs_train) == "SeriousDlqin2yrs"), which(names(cs_train) == "X"), whi
train.y = cs_train$SeriousDlqin2yrs
test.x = cs_test[,-which(names(cs_test) == "SeriousDlqin2yrs")]   # 48,108 obs
test.x = cs_test[,-c(which(names(cs_test) == "SeriousDlqin2yrs"), which(names(cs_test) == "X"))]
test.y = cs_test$SeriousDlqin2yrs
```

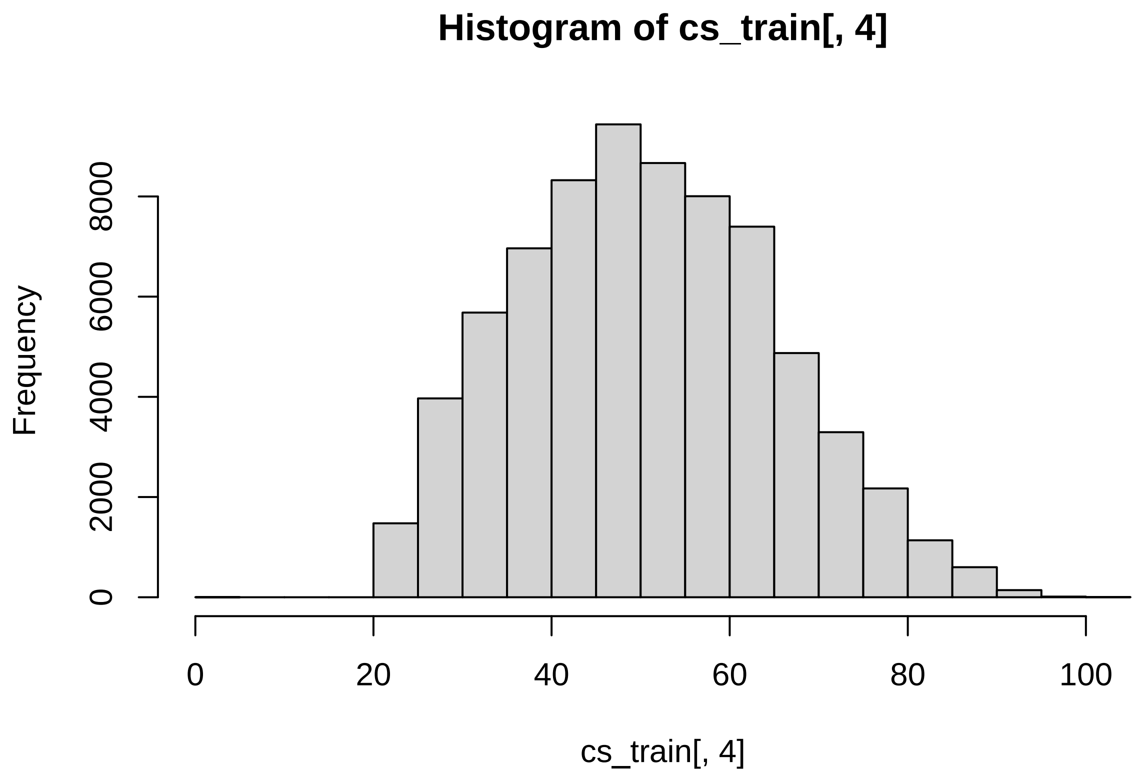
```r
hist(cs_train[,2]) #Response Variable
```



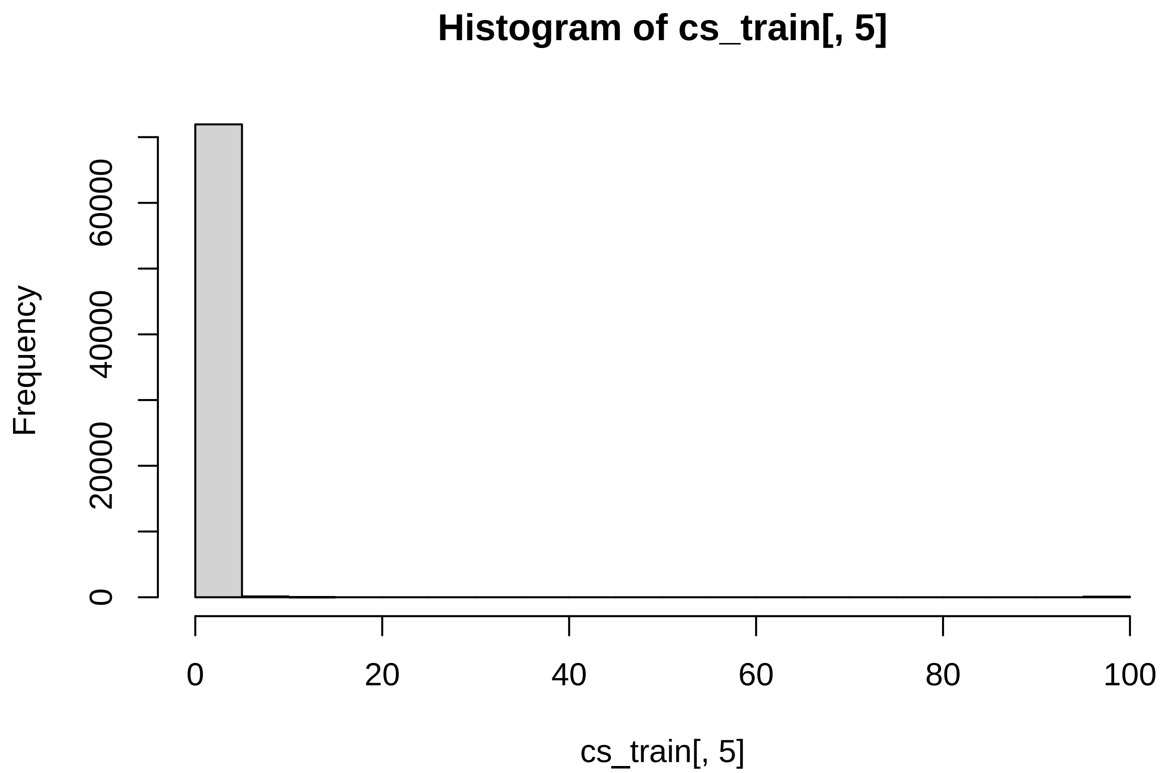

```
hist(cs_train[,3]) #RevolvingUtilizationOfUnsecuredLines
```



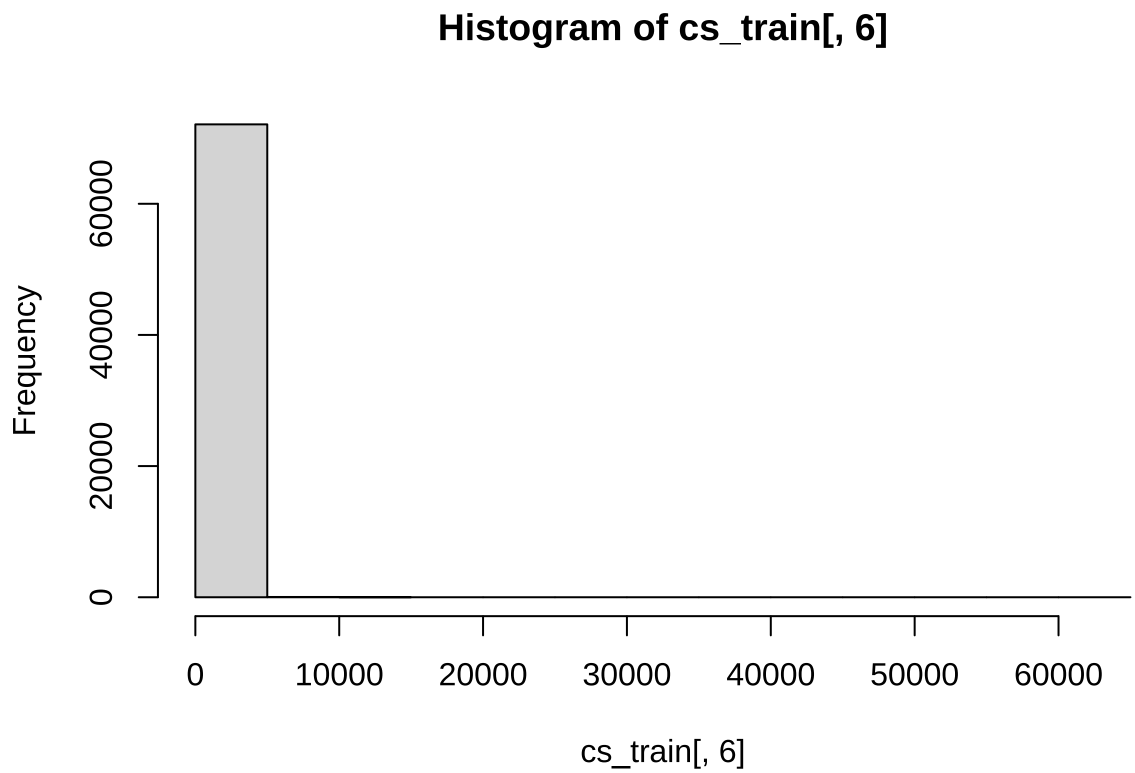

```
hist(cs_train[,4]) #age
```



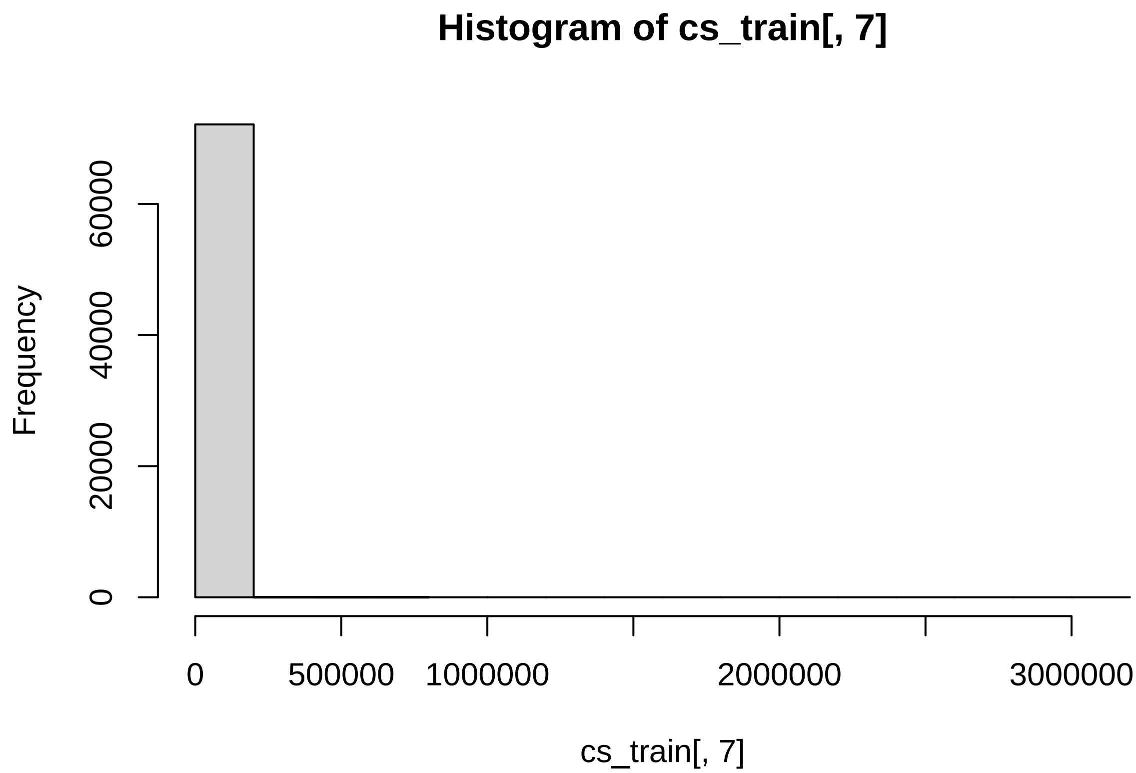

```
hist(cs_train[,5]) #NumberOfTime30.59DaysPastDueNotWorse
```



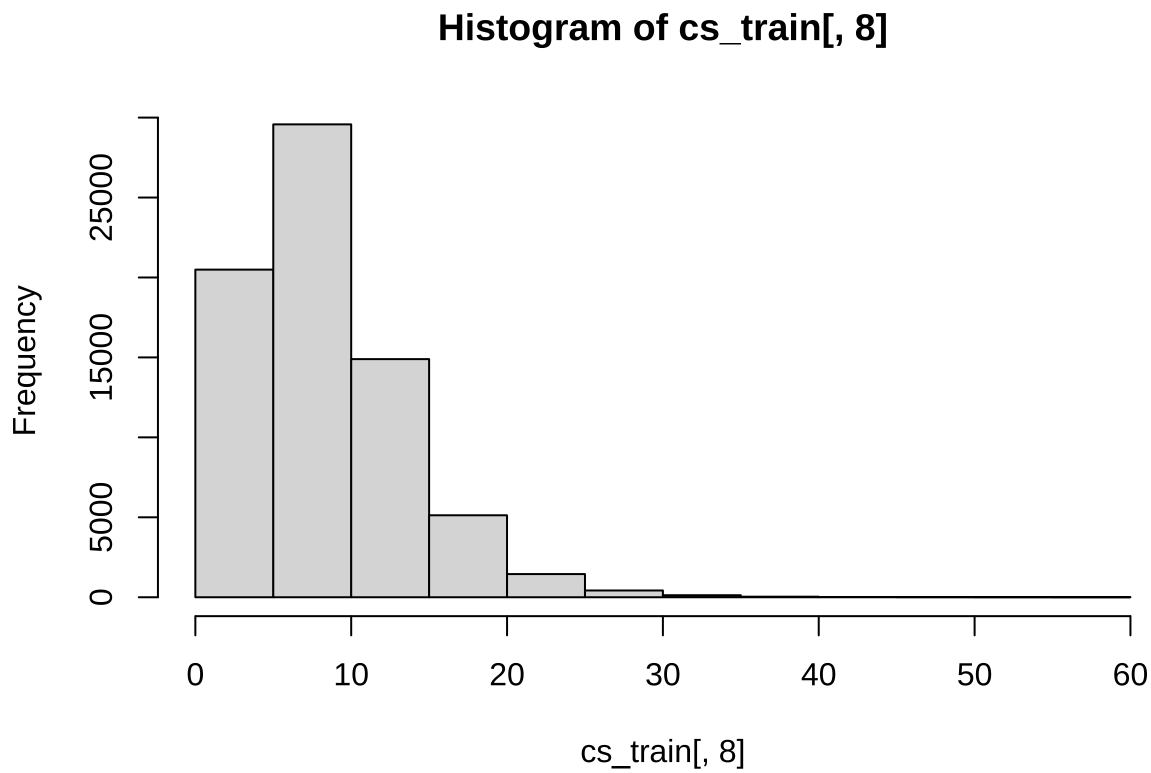

```r
hist(cs_train[,6]) #DebtRatio
```



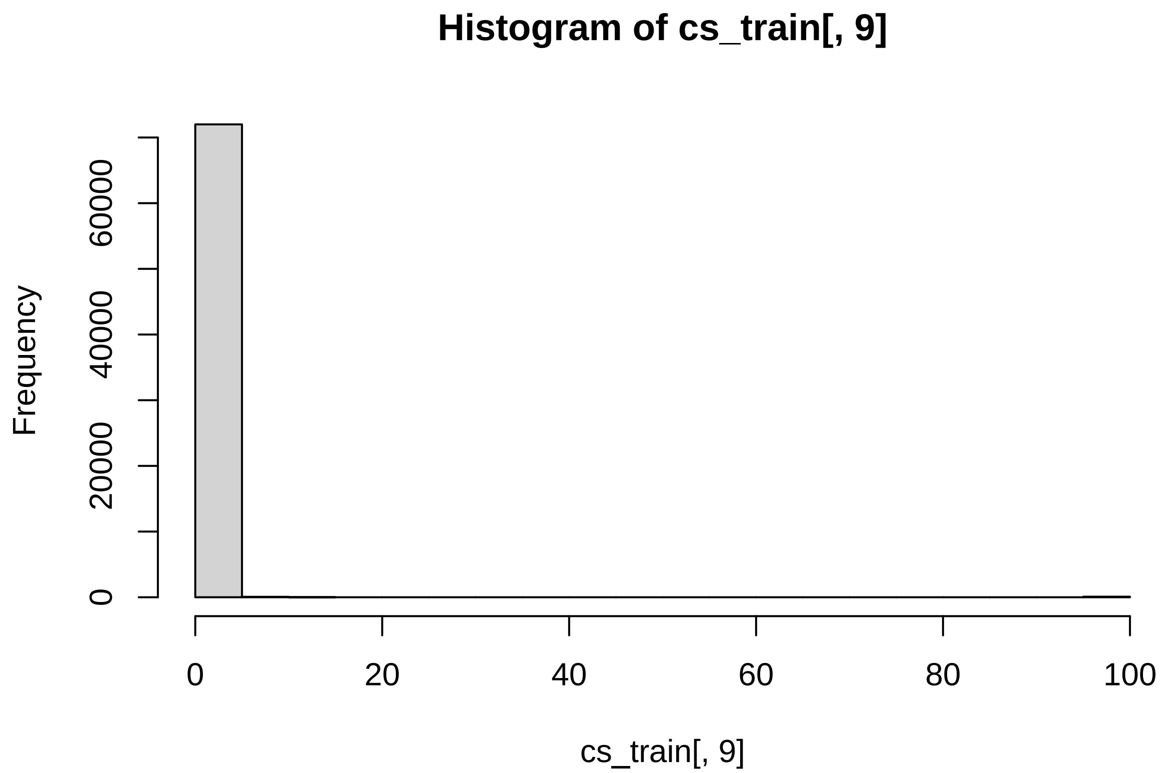

```
hist(cs_train[,7]) #MonthlyIncome
```



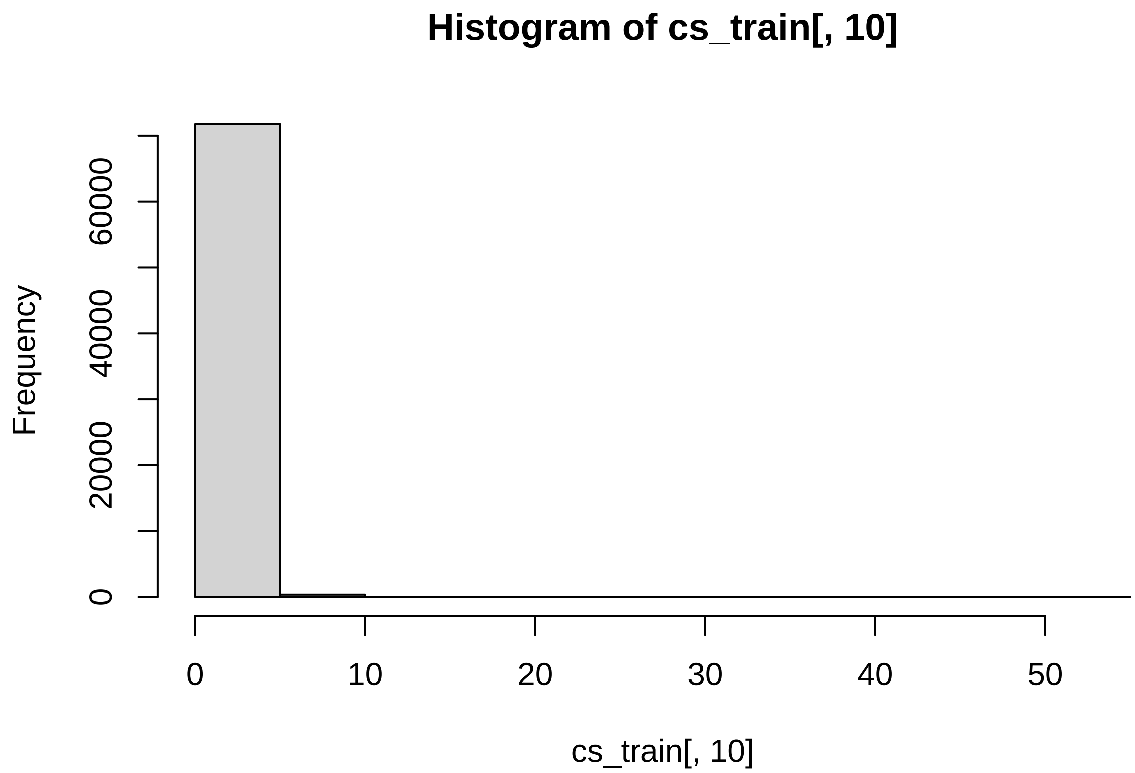

```
hist(cs_train[,8]) #NumberOfOpenCreditLinesAndLoans
```



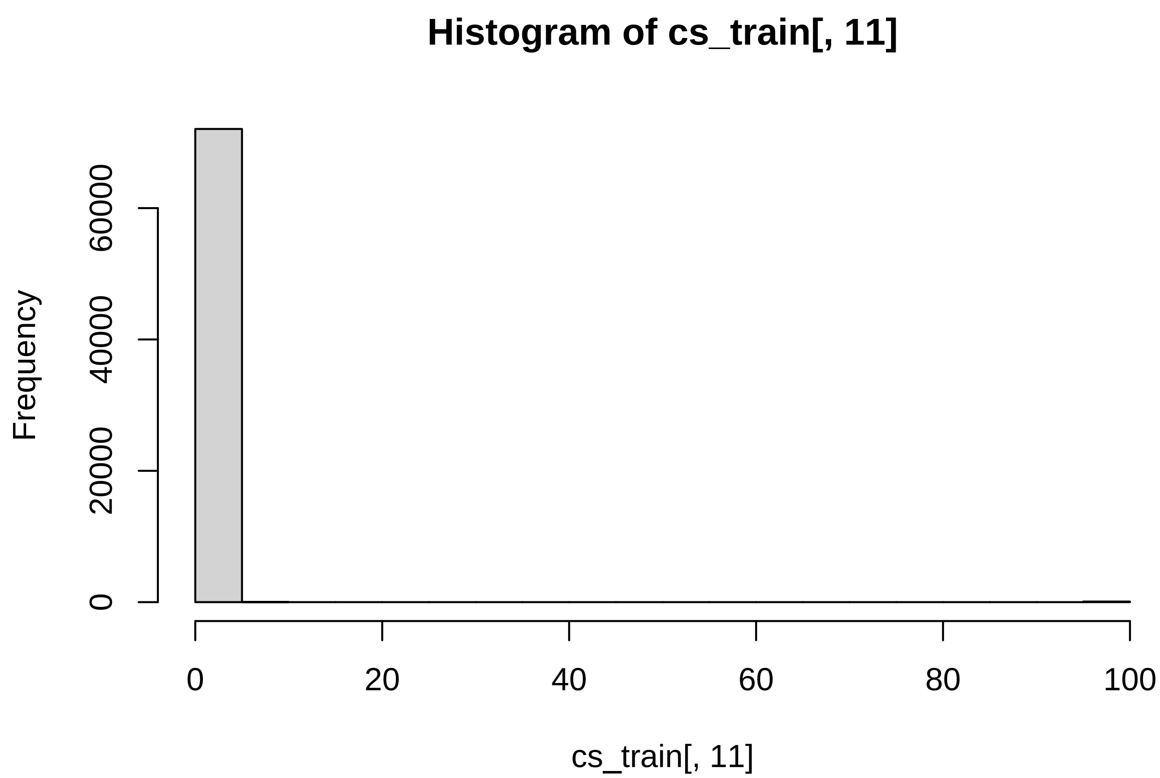

```
hist(cs_train[,9]) #NumberOfTimes90DaysLate
```



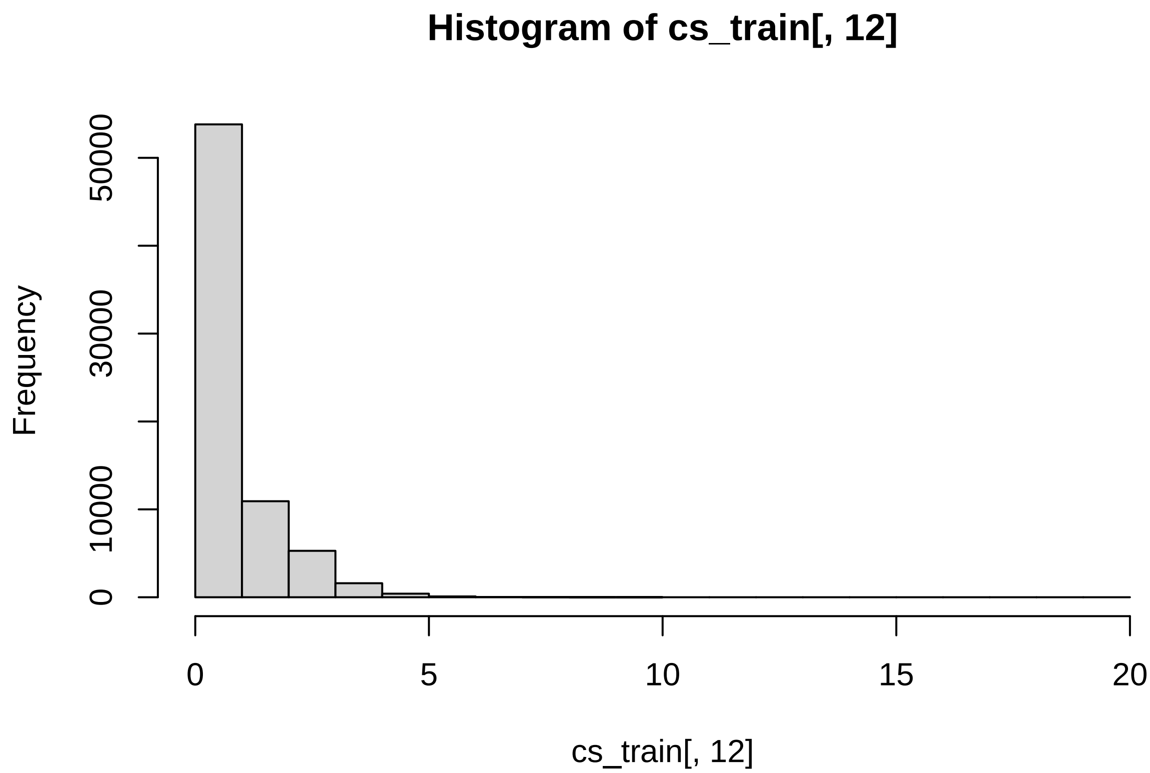

```
hist(cs_train[,10])#NumberRealEstateLoansOrLines
```




```r
hist(cs_train[,11])#NumberOfTime60.89DaysPastDueNotWorse
```




```
hist(cs_train[,12])#NumberOfDependents`    ``
```



Creating Indicator Variables for Extreme Data

```
Jimmys_Bad_Variables <- cs_train
Jimmys_Bad_Variables$HighRevolving <- 0
Jimmys_Bad_Variables$HighRevolving[quantile(Jimmys_Bad_Variables[,3],0.95)] <- 1

Jimmys_Bad_Variables$Many_LessThan2MonthsLate <- 0
Jimmys_Bad_Variables$Many_LessThan2MonthsLate[quantile(Jimmys_Bad_Variables[,5],0.95)] <- 1

Jimmys_Bad_Variables$HighDebtRatio <- 0
Jimmys_Bad_Variables$HighDebtRatio[quantile(Jimmys_Bad_Variables[,6],0.95)] <- 1

Jimmys_Bad_Variables$Rich <- 0
Jimmys_Bad_Variables$Rich[quantile(Jimmys_Bad_Variables[,6],0.95)] <- 1

Jimmys_Bad_Variables$Many_CurrentLoans <- 0
Jimmys_Bad_Variables$Many_CurrentLoans[quantile(Jimmys_Bad_Variables[,8],0.95)] <- 1

Jimmys_Bad_Variables$Many_AtLeastThreeMonthsLate <- 0
Jimmys_Bad_Variables$Many_AtLeastThreeMonthsLate[quantile(Jimmys_Bad_Variables[,9],0.95)] <- 1

Jimmys_Bad_Variables$Many_HouseLoans <- 0
Jimmys_Bad_Variables$Many_HouseLoans[quantile(Jimmys_Bad_Variables[,10],0.95)] <- 1
```

```r
Jimmys_Bad_Variables$Many_TwoToThreeMonthsLate <- 0
Jimmys_Bad_Variables$Many_TwoToThreeMonthsLate[quantile(Jimmys_Bad_Variables[,11],0.95)] <- 1

Jimmys_Bad_Variables$NA_Dependents_are_Mean <- Jimmys_Bad_Variables[,12] #Need to Change NA values to M
Jimmys_Bad_Variables$NA_Dependents_are_Mean[is.na(Jimmys_Bad_Variables[,12])] <- mean(Jimmys_Bad_Variab
Jimmys_Bad_Variables$Many_Dependents <- 0
Jimmys_Bad_Variables$Many_Dependents[quantile(Jimmys_Bad_Variables$NA_Dependents_are_Mean,0.95)] <- 1

train1 <-Jimmys_Bad_Variables[Jimmys_Bad_Variables[,3]< quantile(Jimmys_Bad_Variables[,3],0.95),]
```

```r
head(Jimmys_Bad_Variables)
```

```
##           X SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines age
## 30484 30484                0                           0.09474044  47
## 74216 74216                0                           0.05277307  68
## 54077 54077                0                           0.73665267  36
## 86784 86784                0                           0.02365700  36
## 14388 14388                0                           0.00581900  44
## 31442 31442                0                           0.11523783  40
##       NumberOfTime30.59DaysPastDueNotWorse  DebtRatio MonthlyIncome
## 30484                                    0 0.36842105          6250
## 74216                                    0 0.22759781         11884
## 54077                                    0 0.09445277          2000
## 86784                                    0 0.21710409          5600
## 14388                                    0 0.18525779          5100
## 31442                                    1 2.49168646          2946
##       NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 30484                               8                       0
## 74216                              13                       0
## 54077                               6                       0
## 86784                               9                       0
## 14388                              24                       0
## 31442                              17                       0
##       NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 30484                            1                                    0
## 74216                            2                                    0
## 54077                            0                                    0
## 86784                            1                                    0
## 14388                            1                                    0
## 31442                            3                                    0
##       NumberOfDependents HighRevolving Many_LessThan2MonthsLate HighDebtRatio
## 30484                  3             0                        0             1
## 74216                  1             0                        1             0
## 54077                  1             0                        0             0
## 86784                  0             0                        0             0
## 14388                  2             0                        0             0
## 31442                  0             0                        0             0
##       Rich Many_CurrentLoans Many_AtLeastThreeMonthsLate Many_HouseLoans
## 30484    1                 0                           1               0
## 74216    0                 0                           0               0
## 54077    0                 0                           0               1
## 86784    0                 0                           0               0
## 14388    0                 0                           0               0
```

```
## 31442    0                  0                             0                0
##        Many_TwoToThreeMonthsLate NA_Dependents_are_Mean Many_Dependents
## 30484                         1                      3               0
## 74216                         0                      1               0
## 54077                         0                      1               1
## 86784                         0                      0               0
## 14388                         0                      2               0
## 31442                         0                      0               0
```

```
Just_Bad_Variables <- subset.data.frame(Jimmys_Bad_Variables, select = c(X,SeriousDlqin2yrs, age, HighRe
```

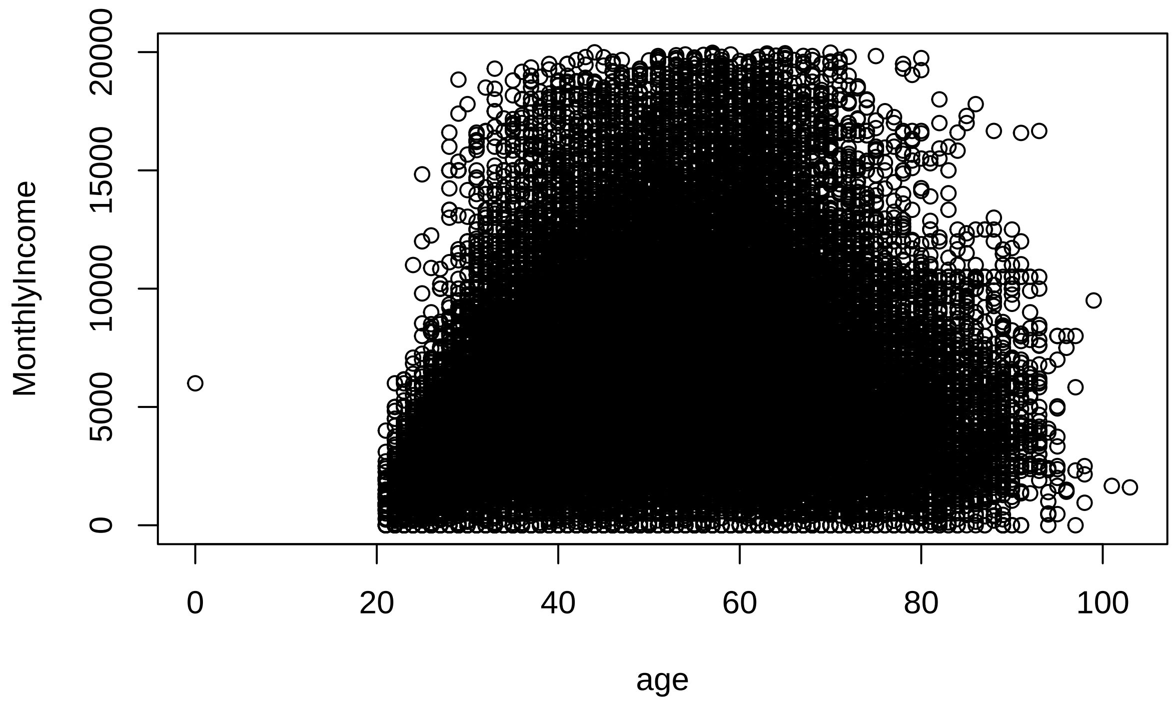
1. Question/Goal

- Comparing our model performance against paper's model
- Most important factors

2. EDA (Still need to Clean up for better graphs)

- monthly income = 0
- 30k monthly income = NA
- Dependent = NA 4k
- age 0 remove - only 1 point
- age very old
- group by age group etc
- 13 - 101 or older (maybe cut at 100)
- 80 or more

```
#plot(SeriousDlqin2yrs ~ age, data = cs_train_maj)
cs_train_maj_g1 <- cs_train_maj[cs_train_maj$MonthlyIncome<500000,] # less than 500k monthly income
cs_train_maj_g1 <- cs_train_maj[cs_train_maj$MonthlyIncome<20000,] # less than 20k monthly income
plot(MonthlyIncome ~ age, data = cs_train_maj_g1)
```




```
train <- cs_train
##Shelly EDA Code
tapply(cs_train$RevolvingUtilizationOfUnsecuredLines, train$SeriousDlqin2yrs, median) #use this; data v
```

```
##         0         1
## 0.1544904 0.8066074
```

```
##Next Steps
# ***need final data set which excludes outliers for which we will create final model from
# do box plots for age group and income; fix axes so that box is readable -Shelly
# histogram of groupings like income by age groups, different colors -Yolanda
# sampling of data --> randomly select instead of adding additional data for response variable -Shelly
# ClassDiscovery package and DataExplorer --> provide initial graphs and analyses of data; also initial



 #Remove outliers; maybe only keep anything greater than 10
 tapply(train$RevolvingUtilizationOfUnsecuredLines, train$SeriousDlqin2yrs, mean) #d
```

```
##        0        1
## 6.566204 3.184407
```

```
 tapply(train$age, train$SeriousDlqin2yrs, median) #median age of 45 for people defaulting
```

```
##  0  1
## 51 46
```

```r
#tapply(train$`NumberOfTime30-59DaysPastDueNotWorse`, train$SeriousDlqin2yrs, mean) #2.4 times for tho
tapply(train$NumberOfTimes90DaysLate, train$SeriousDlqin2yrs, mean) #2.1 times for those with delinque
```

```
##        0        1
## 0.111931 1.523173
```

```r
tapply(train$DebtRatio, train$SeriousDlqin2yrs, median) #0.43 for delinquency incidence vs 0.36 for no
```

```
##         0         1
## 0.2924788 0.3600771
```

```r
#tapply(train$NumIncome, train$SeriousDlqin2yrs, median) #delinquency monthly income of $3.8K vs $4.4K
tapply(train$NumberOfTimes90DaysLate, train$SeriousDlqin2yrs, mean)
```

```
##        0        1
## 0.111931 1.523173
```

```r
#summary(train$NumIncome)
```

3. Balancing Data (Creating New Datasets)

- Use clustering and pick one sub-group from majority
  - can try Age/Income
  - try Income/debt ratio
- Use bagging algorithm to create more minority data
- Use NA indicator 0 or 1 (maybe use mean/median)

b) Resampling Majority Data (new_train.final 1 & 2)

- Method 1: use Full omit Data to sample majority data (new_train.final1) (8357*2)
- Method 2: use Training Data to sample majority data (new_train.final2) (4941*2)

```r
###SHELLY -  Balance Response Variable
set.seed(2) # for reproducibility


## Method 1: Balance Full omit Data (new_train.final1)
new_train <- filter(cs_train_omit, cs_train_omit$SeriousDlqin2yrs == 0) #112k rows

#Check Response Variable Balance in cs_train_omit data
table(cs_train_omit$SeriousDlqin2yrs) #8,357 1's
```

```
##
##      0      1
## 111912   8357
```

```
n_add <- sum(cs_train_omit$SeriousDlqin2yrs ==1)  # Number of Default
n_add
```

```
## [1] 8357
```

```
new_train2 <- new_train[sample(1:nrow(new_train)),] #112k rows without replacement
new_train3 <- new_train2[1:n_add,]

#######**Merge Data Together (use new_train5 variable)
new_train4 <- filter(cs_train_omit, cs_train_omit$SeriousDlqin2yrs == 1)
#nrow(new_train4)
new_train.final1 <- rbind(new_train3, new_train4)
nrow(new_train.final1) #16,714 rows as there should be (8357*2)
```

```
## [1] 16714
```

```
table(new_train.final1$SeriousDlqin2yrs) #correct; equal # of 0's and 1's
```

```
##
##    0    1
## 8357 8357
```

```
## Method 2: Balance Training Data (new_train.final2)

  #Check Response Variable Balance in cs_train_omit data
  table(cs_train$SeriousDlqin2yrs) #0: 67220  1: 4941
```

```
##
##     0     1
## 67220  4941
```

```
  n_add <- nrow(cs_train_min) # Number of Default

 # Randomly Sample data from Non-Default group (sample = min group obs#)
 new_train <- cs_train_maj[sample(1:nrow(cs_train_maj),n_add, replace=FALSE),] #67220 rows without repl

 #######**Merge Data Together (use new_train.final2 variable)
 new_train.final2 <- rbind(new_train, cs_train_min)
 nrow(new_train.final2) #10k rows as there should be (4941*2)
```

```
## [1] 9882
```

```
table(new_train.final2$SeriousDlqin2yrs) #correct; equal # of 0's and 1's
```

```
##
##    0    1
## 4941 4941
```

c) Using Clustering to select from Majority Group

Select one cluster from majority group and combine with minority group data to form new Balanced Training Data

```
training_dataset <- data.frame(train.x, train.y)
kmeans_model_train  <- kmeans(training_dataset, centers = 5) # centers because paper had 5 clusters
kmeans_model_train$centers #what each cluster's average values are for each variable. The most obvious
```

```
##   RevolvingUtilizationOfUnsecuredLines      age
## 1                           0.25563963 51.85714
## 2                           0.30094326 53.69880
## 3                           0.00732813 52.00000
## 4                          14.17070502 53.63628
## 5                           4.13445042 50.60142
##   NumberOfTime30.59DaysPastDueNotWorse    DebtRatio MonthlyIncome
## 1                            0.2857143  0.003406329    580671.714
## 2                            0.2228916  0.076526033     91885.494
## 3                            0.0000000  0.001470045   3008750.000
## 4                            0.2515422  0.296182799     12837.435
## 5                            0.4164810 35.479464445      4525.353
##   NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 1                        9.142857              0.00000000
## 2                       11.415663              0.06626506
## 3                       10.000000              0.00000000
## 4                       10.787612              0.05615007
## 5                        8.178820              0.25217019
##   NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 1                    1.0000000                           0.28571429
## 2                    2.1927711                           0.01807229
## 3                    1.0000000                           0.00000000
## 4                    1.6671912                           0.05470225
## 5                    0.8784692                           0.22136860
##   NumberOfDependents    train.y
## 1          1.2857143 0.00000000
## 2          1.1566265 0.06024096
## 3          3.0000000 0.00000000
## 4          1.1644215 0.04418985
## 5          0.7642823 0.07538190
```

```
#Cluster 3 and 4 are different than Cluster 1 and 2 in terms of the Response Variable.
kmeans_model_train$size
```

```
## [1]     7   166     1 15886 56101
```

##Sample from Majority Data Proportional to Minority Data

```
set.seed(500)
Majority <- cs_train_maj[,-which(names(cs_train_maj) == "X")]
kmeans_model_train_majority <- kmeans(Majority, centers = 5)
kmeans_model_train_majority$centers
```

```
##   SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines      age
## 1                0                           0.25563963 51.85714
## 2                0                           0.29232591 54.07692
## 3                0                           4.17487638 51.02706
## 4                0                           0.00732813 52.00000
## 5                0                          14.81383169 53.81218
##   NumberOfTime30.59DaysPastDueNotWorse    DebtRatio MonthlyIncome
## 1                            0.2857143  0.003406329    580671.714
## 2                            0.1923077  0.075301645     92001.218
## 3                            0.2853509 35.684460036      4552.355
## 4                            0.0000000  0.001470045   3008750.000
## 5                            0.2149120  0.289472207     12835.539
##   NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 1                        9.142857              0.00000000
## 2                       11.429487              0.04487179
## 3                        8.217530              0.13319329
## 4                       10.000000              0.00000000
## 5                       10.728789              0.03994990
##   NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 1                    1.0000000                           0.28571429
## 2                    2.0641026                           0.01282051
## 3                    0.8795845                           0.12134061
## 4                    1.0000000                           0.00000000
## 5                    1.6433516                           0.04146615
##   NumberOfDependents
## 1          1.2857143
## 2          1.1666667
## 3          0.7441363
## 4          3.0000000
## 5          1.1581515
```

```
kmeans_model_train_majority$size
```

```
## [1]     7   156 51887     1 15169
```

```
cluster_majority <-kmeans_model_train_majority$cluster
Majority_with_cluster <- cbind(Majority, cluster = kmeans_model_train_majority$cluster)
head(Majority_with_cluster)
```

```
##       SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines age
## 30484                0                           0.09474044  47
## 74216                0                           0.05277307  68
## 54077                0                           0.73665267  36
## 86784                0                           0.02365700  36
## 14388                0                           0.00581900  44
## 31442                0                           0.11523783  40
##       NumberOfTime30.59DaysPastDueNotWorse  DebtRatio MonthlyIncome
## 30484                                    0 0.36842105          6250
## 74216                                    0 0.22759781         11884
## 54077                                    0 0.09445277          2000
## 86784                                    0 0.21710409          5600
## 14388                                    0 0.18525779          5100
## 31442                                    1 2.49168646          2946
```

```
##       NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 30484                               8                       0
## 74216                              13                       0
## 54077                               6                       0
## 86784                               9                       0
## 14388                              24                       0
## 31442                              17                       0
##       NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 30484                            1                                    0
## 74216                            2                                    0
## 54077                            0                                    0
## 86784                            1                                    0
## 14388                            1                                    0
## 31442                            3                                    0
##       NumberOfDependents cluster
## 30484                  3       3
## 74216                  1       5
## 54077                  1       3
## 86784                  0       3
## 14388                  2       3
## 31442                  0       3
```

```r
Cluster1 <- subset.data.frame(Majority_with_cluster,cluster == 1)
Cluster2 <- subset.data.frame(Majority_with_cluster,cluster == 2)
Cluster3 <- subset.data.frame(Majority_with_cluster,cluster == 3)
Cluster4 <- subset.data.frame(Majority_with_cluster,cluster == 4)
Cluster5 <- subset.data.frame(Majority_with_cluster, cluster == 5)
Special_Small_Clusters <- sum(nrow(Cluster1),nrow(Cluster2),nrow(Cluster4))
Sample3 <- Cluster3[sample(nrow(Cluster3), nrow(Cluster3)/(nrow(Majority) - Special_Small_Clusters) * n
head(Sample3)
```

```
##       SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines age
## 35580                0                           0.20281849  46
## 46953                0                           0.01629959  64
## 52453                0                           0.00000000  47
## 24027                0                           0.02444725  81
## 1521                 0                           0.35189283  45
## 61774                0                           0.02084403  52
##       NumberOfTime30.59DaysPastDueNotWorse    DebtRatio MonthlyIncome
## 35580                                    0    0.4228743          7973
## 46953                                    0    0.6090246          3700
## 52453                                    0    0.2183726          5050
## 24027                                    0 1187.0000000             0
## 1521                                     0    0.5638525          3280
## 61774                                    0    0.4754493          6231
##       NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 35580                              10                       0
## 46953                              10                       0
## 52453                               5                       0
## 24027                              12                       0
## 1521                               18                       0
## 61774                              15                       0
##       NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 35580                            2                                    1
```

```
## 46953                          1                                  0
## 52453                          1                                  0
## 24027                          1                                  0
## 1521                           3                                  0
## 61774                          3                                  0
##       NumberOfDependents cluster
## 35580                  2       3
## 46953                  0       3
## 52453                  1       3
## 24027                  1       3
## 1521                   4       3
## 61774                  3       3
```

```
nrow(Sample3)
```

```
## [1] 3823
```

```
Sample5 <- Cluster5[sample(nrow(Cluster5), nrow(Cluster5)/(nrow(Majority) - Special_Small_Clusters) * n
nrow(Sample5)
```

```
## [1] 1117
```

```
balanced_dataset.1 <- rbind(cs_train_min[-which(names(cs_train_min) == "X")],Cluster1[-which(names(Clus
head(balanced_dataset.1)
```

```
##        SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines age
## 17360                 1                           0.66672086  50
## 90565                 1                           0.94196675  59
## 146254                1                           0.99999990  28
## 102792                1                           1.02589064  30
## 49072                 1                           0.65871090  43
## 142766                1                           0.08758682  37
##        NumberOfTime30.59DaysPastDueNotWorse  DebtRatio MonthlyIncome
## 17360                                     0 0.10303558         18381
## 90565                                     1 0.16531402          8008
## 146254                                   98 0.00000000          1664
## 102792                                    1 0.23480463          2763
## 49072                                     0 0.08972058          3900
## 142766                                    0 0.37049259         16666
##        NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 17360                               10                       0
## 90565                                6                       5
## 146254                               0                      98
## 102792                               9                       0
## 49072                                4                       2
## 142766                              22                       0
##        NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 17360                             1                                    1
## 90565                             0                                    0
## 146254                            0                                   98
## 102792                            0                                    2
## 49072                             0                                    1
```

```
## 142766                            3                                      0
##        NumberOfDependents
## 17360                   1
## 90565                   0
## 146254                  0
## 102792                  3
## 49072                   2
## 142766                  0
```

```r
nrow(balanced_dataset.1)
```

```
## [1] 10045
```

```r
balanced_x <- data.matrix(balanced_dataset.1[,-1])
balanced_y <- data.matrix(balanced_dataset.1[,1])
```

```r
Minority <- cs_train_min[,-which(names(cs_train_min) == "X")]
kmeans_model_train_minority <- kmeans(Minority, centers = 5)
kmeans_model_train_minority$centers
```

```
##   SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines      age
## 1                1                            3.0460536 44.18669
## 2                1                            3.9472824 48.66770
## 3                1                            0.3757828 62.00000
## 4                1                            0.5243205 45.52381
## 5                1                            0.5747007 50.53819
##   NumberOfTime30.59DaysPastDueNotWorse   DebtRatio MonthlyIncome
## 1                            2.2734194 45.78957515      3199.691
## 2                            1.3074534  0.44067173      7530.187
## 3                            0.0000000  0.02271191    250000.000
## 4                            0.8571429  0.16647645     55664.190
## 5                            1.0798611  0.50577270     15951.382
##   NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 1                         6.88911               2.0648792
## 2                        10.10745               0.7211180
## 3                        15.00000               0.0000000
## 4                        14.61905               0.5238095
## 5                        13.15625               0.4027778
##   NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 1                    0.6765972                            1.7457795
## 2                    1.4540373                            0.6136646
## 3                    2.0000000                            0.0000000
## 4                    4.6666667                            0.0952381
## 5                    2.6319444                            0.3472222
##   NumberOfDependents
## 1          0.9483615
## 2          1.2012422
## 3          0.0000000
## 4          1.0476190
## 5          1.3368056
```

```r
kmeans_model_train_minority$size
```

```
## [1] 3021 1610    1   21  288
```

```r
testing_data <- data.frame(test.x,test.y)
kmeans_model_test  <- kmeans(testing_data, centers = 5) #testing to see if test data has similar cluste
kmeans_model_test$centers #Clusters are grouped as (1,2), (3,5) and (4) for Response Variables.
```

```
##   RevolvingUtilizationOfUnsecuredLines      age
## 1                            3.3340523 50.35998
## 2                            0.2950418 54.23077
## 3                            0.1501357 60.16667
## 4                            7.2909033 52.99487
## 5                           27.6093312 54.01161
##   NumberOfTime30.59DaysPastDueNotWorse    DebtRatio MonthlyIncome
## 1                            0.4511763 38.266839034      3910.045
## 2                            0.3974359  0.077906671     96138.115
## 3                            0.3333333  0.002840504   1103530.500
## 4                            0.2605648  0.309173812      9940.086
## 5                            0.2585139  0.234410533     25034.900
##   NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 1                        7.916175              0.29827306
## 2                       10.666667              0.08974359
## 3                       11.666667              0.00000000
## 4                       10.209088              0.06744563
## 5                       11.632353              0.03715170
##   NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 1                    0.8020238                           0.26264883
## 2                    2.3333333                           0.08974359
## 3                    1.3333333                           0.00000000
## 4                    1.4717300                           0.06452451
## 5                    2.0890093                           0.05263158
##   NumberOfDependents     test.y
## 1           0.718358 0.08213362
## 2           1.243590 0.05128205
## 3           0.500000 0.00000000
## 4           1.074132 0.04985394
## 5           1.330495 0.05495356
```

4. Applying Different Methods (with and without Balanced Data)

- Logistic regression (compare the different link functions)
- look maybe merge Lasso with logistic regression
- RF

```r
AIC_unbalanced <- 1:4
AIC_balanced <- 1:4
names(AIC_unbalanced)[1] <- "PCA"
names(AIC_unbalanced)[2] <- "Regular GLM"
names(AIC_unbalanced)[3] <- "Ridge"
```

```r
names(AIC_unbalanced)[4] <- "Lasso"

names(AIC_balanced)[1] <- "PCA"
names(AIC_balanced)[2] <- "Regular GLM"
names(AIC_balanced)[3] <- "Ridge"
names(AIC_balanced)[4] <- "Lasso"
```

Method 1: PCA

1-a) PCA with Unbalanced Data

PCA using all 22 variables

```r
pca_including_dummy_variables <- prcomp(na.omit(Jimmys_Bad_Variables[,-c(which(names(Jimmys_Bad_Variable
```

```r
summary(pca_including_dummy_variables) #First PC has like 99.66% of variance
```

```
## Importance of components:
##                              PC1      PC2      PC3   PC4   PC5   PC6   PC7
## Standard deviation     1.420e+04 426.3224 2.96e+02 14.49 5.925 5.059 1.586
## Proportion of Variance 9.987e-01   0.0009 4.30e-04  0.00 0.000 0.000 0.000
## Cumulative Proportion  9.987e-01   0.9996 1.00e+00  1.00 1.000 1.000 1.000
##                          PC8   PC9  PC10     PC11     PC12     PC13     PC14
## Standard deviation     1.027 0.565 0.346 0.007445 0.005265 0.003723 0.003722
## Proportion of Variance 0.000 0.000 0.000 0.000000 0.000000 0.000000 0.000000
## Cumulative Proportion  1.000 1.000 1.000 1.000000 1.000000 1.000000 1.000000
##                            PC15      PC16      PC17      PC18      PC19
## Standard deviation     3.59e-15 1.287e-17 9.127e-18 3.089e-20 3.354e-23
## Proportion of Variance 0.00e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00
## Cumulative Proportion  1.00e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00
##                             PC20
## Standard deviation     2.404e-37
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

```r
pca_including_dummy_variables$rotation[,1] #I think it really likes Monthly Income as it's close to 1.
```

```
## RevolvingUtilizationOfUnsecuredLines                                  age
##                         1.663543e-04                         3.754805e-05
## NumberOfTime30.59DaysPastDueNotWorse                            DebtRatio
##                        -2.494353e-06                        -9.026497e-04
##                        MonthlyIncome      NumberOfOpenCreditLinesAndLoans
##                         9.999996e-01                         3.387604e-05
##              NumberOfTimes90DaysLate         NumberRealEstateLoansOrLines
##                        -2.982336e-06                         1.014049e-05
## NumberOfTime60.89DaysPastDueNotWorse                   NumberOfDependents
##                        -2.547678e-06                         5.400872e-06
##                        HighRevolving            Many_LessThan2MonthsLate
##                         0.000000e+00                         3.594719e-10
##                        HighDebtRatio                                 Rich
```

```
##                    -2.774522e-11                        -2.774522e-11
##                 Many_CurrentLoans          Many_AtLeastThreeMonthsLate
##                     3.130335e-09                        -2.774522e-11
##                   Many_HouseLoans            Many_TwoToThreeMonthsLate
##                    -3.198419e-10                        -2.774522e-11
##              NA_Dependents_are_Mean                      Many_Dependents
##                     5.400872e-06                        -3.198419e-10
```

```
glm_pca_including_dummy_variables <- glm(train.y ~ pca_including_dummy_variables$x[,1], family = "binomi
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(glm_pca_including_dummy_variables)
```

```
##
## Call:
## glm(formula = train.y ~ pca_including_dummy_variables$x[, 1],
##     family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.4302  -0.3975  -0.3784  -0.3507   5.2302
##
## Coefficients:
##                                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)                          -2.638e+00  1.522e-02 -173.29   <2e-16 ***
## pca_including_dummy_variables$x[, 1] -4.536e-05  3.798e-06  -11.94   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 36033  on 72160  degrees of freedom
## Residual deviance: 35857  on 72159  degrees of freedom
## AIC: 35861
##
## Number of Fisher Scoring iterations: 6
```

PCA using original 10 variables

```
pca_original_data <- prcomp(train.x)
```

```
summary(pca_original_data) #PC1 has 99.66% variance explained
```

```
## Importance of components:
##                             PC1      PC2      PC3   PC4   PC5   PC6   PC7
## Standard deviation     1.420e+04 426.3224 2.96e+02 14.49 5.925 5.058 1.144
## Proportion of Variance 9.987e-01   0.0009 4.30e-04  0.00 0.000 0.000 0.000
## Cumulative Proportion  9.987e-01   0.9996 1.00e+00  1.00 1.000 1.000 1.000
##                            PC8   PC9  PC10
```

```
## Standard deviation     1.007 0.565 0.346
## Proportion of Variance 0.000 0.000 0.000
## Cumulative Proportion  1.000 1.000 1.000
```

```
pca_original_data$rotation[,1] #It also really likes MonthlyIncome and not anything else
```

```
## RevolvingUtilizationOfUnsecuredLines                                  age
##                         1.663543e-04                         3.754805e-05
## NumberOfTime30.59DaysPastDueNotWorse                            DebtRatio
##                        -2.494353e-06                        -9.026497e-04
##                        MonthlyIncome      NumberOfOpenCreditLinesAndLoans
##                         9.999996e-01                         3.387604e-05
##              NumberOfTimes90DaysLate         NumberRealEstateLoansOrLines
##                        -2.982336e-06                         1.014049e-05
## NumberOfTime60.89DaysPastDueNotWorse                   NumberOfDependents
##                        -2.547678e-06                         5.400872e-06
```

```
glm_pca_original_data <- glm(train.y ~ pca_original_data$x[,1], family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(glm_pca_original_data)
```

```
##
## Call:
## glm(formula = train.y ~ pca_original_data$x[, 1], family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.4302  -0.3975  -0.3784  -0.3507   5.2302
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -2.638e+00  1.522e-02 -173.29   <2e-16 ***
## pca_original_data$x[, 1] -4.536e-05  3.798e-06  -11.94   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 36033  on 72160  degrees of freedom
## Residual deviance: 35857  on 72159  degrees of freedom
## AIC: 35861
##
## Number of Fisher Scoring iterations: 6
```

```
AIC_unbalanced[1] <- glm_pca_original_data$aic
```

PCA Using Dummy variables

```
pca_dummy <- prcomp(Just_Bad_Variables[,-c(which(names(Just_Bad_Variables) == "SeriousDlqin2yrs"), whic
```

```
summary(pca_dummy) #PC 1 is 100%?
```

```
## Importance of components:
##                          PC1      PC2      PC3      PC4      PC5       PC6
## Standard deviation     14.46 0.007445 0.005265 0.003723 0.003723 7.281e-16
## Proportion of Variance  1.00 0.000000 0.000000 0.000000 0.000000 0.000e+00
## Cumulative Proportion   1.00 1.000000 1.000000 1.000000 1.000000 1.000e+00
##                              PC7       PC8       PC9      PC10
## Standard deviation     6.661e-16 1.404e-19 8.404e-23 1.584e-35
## Proportion of Variance 0.000e+00 0.000e+00 0.000e+00 0.000e+00
## Cumulative Proportion  1.000e+00 1.000e+00 1.000e+00 1.000e+00
```

```
pca_dummy$rotation[,1] #Age is significant
```

```
##                         age               HighRevolving
##                1.000000e+00                0.000000e+00
##    Many_LessThan2MonthsLate                HighDebtRatio
##                1.108186e-06               -2.834080e-07
##                        Rich            Many_CurrentLoans
##               -2.834080e-07                5.117887e-07
## Many_AtLeastThreeMonthsLate              Many_HouseLoans
##               -2.834080e-07               -1.012338e-06
##   Many_TwoToThreeMonthsLate              Many_Dependents
##               -2.834080e-07               -1.012338e-06
```

```
glm_pca_dummy <- glm(train.y ~ pca_dummy$x[,1], family = "binomial")
summary(glm_pca_dummy)
```

```
##
## Call:
## glm(formula = train.y ~ pca_dummy$x[, 1], family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7208  -0.4135  -0.3553  -0.3049   2.8707
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -2.681737   0.015742 -170.36   <2e-16 ***
## pca_dummy$x[, 1] -0.028603   0.001092  -26.19   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 36033  on 72160  degrees of freedom
## Residual deviance: 35308  on 72159  degrees of freedom
## AIC: 35312
##
## Number of Fisher Scoring iterations: 5
```

1 b) PCA with Balanced Data

```
pca_balanced <- prcomp(balanced_dataset.1[,-which(names(balanced_dataset.1) == "SeriousDlqin2yrs")])
summary(pca_balanced) #99.98% Variance is explained by PC1
```

```
## Importance of components:
##                              PC1       PC2       PC3   PC4   PC5   PC6   PC7
## Standard deviation     3.625e+04 520.00821 435.91150 13.82 11.52 5.317 1.248
## Proportion of Variance 9.997e-01   0.00021   0.00014  0.00  0.00 0.000 0.000
## Cumulative Proportion  9.997e-01   0.99986   1.00000  1.00  1.00 1.000 1.000
##                          PC8   PC9  PC10
## Standard deviation     1.159 0.903 0.627
## Proportion of Variance 0.000 0.000 0.000
## Cumulative Proportion  1.000 1.000 1.000
```

```
pca_balanced$rotation[,1]
```

```
## RevolvingUtilizationOfUnsecuredLines                                  age
##                         1.070390e-04                         1.358236e-05
## NumberOfTime30.59DaysPastDueNotWorse                            DebtRatio
##                        -2.634321e-06                        -1.573957e-04
##                        MonthlyIncome      NumberOfOpenCreditLinesAndLoans
##                         1.000000e+00                         9.098638e-06
##              NumberOfTimes90DaysLate         NumberRealEstateLoansOrLines
##                        -2.675902e-06                         2.790161e-06
## NumberOfTime60.89DaysPastDueNotWorse                   NumberOfDependents
##                        -2.291266e-06                         1.294679e-06
```

```
glm_pca_balanced <- glm(balanced_dataset.1$SeriousDlqin2yrs ~ pca_balanced$x[,1], family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(glm_pca_balanced)
```

```
##
## Call:
## glm(formula = balanced_dataset.1$SeriousDlqin2yrs ~ pca_balanced$x[,
##     1], family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2896  -1.1847  -0.3043   1.1506   4.6412
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -9.622e-02  2.158e-02  -4.459 8.22e-06 ***
## pca_balanced$x[, 1] -4.412e-05  3.746e-06 -11.779  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 13923  on 10044  degrees of freedom
## Residual deviance: 13655  on 10043  degrees of freedom
## AIC: 13659
##
## Number of Fisher Scoring iterations: 6
```

```
AIC_balanced[1] <- glm_pca_balanced$aic
```

Method 2: Using GLM Original Predictors vs Extreme Binned Predictors + MonthlyIncome

We compared GLM original Predictors with extreme binned predictors using unbalanced data and see better performance in original numeric predictors.

We now try to improve the model using balanced data.

2 a) GLM Original Predictors with Unbalanced Data

```
## Orignial Predictors

model <- glm(Jimmys_Bad_Variables$SeriousDlqin2yrs ~ ., family = "binomial", data = subset(Jimmys_Bad_Va
```

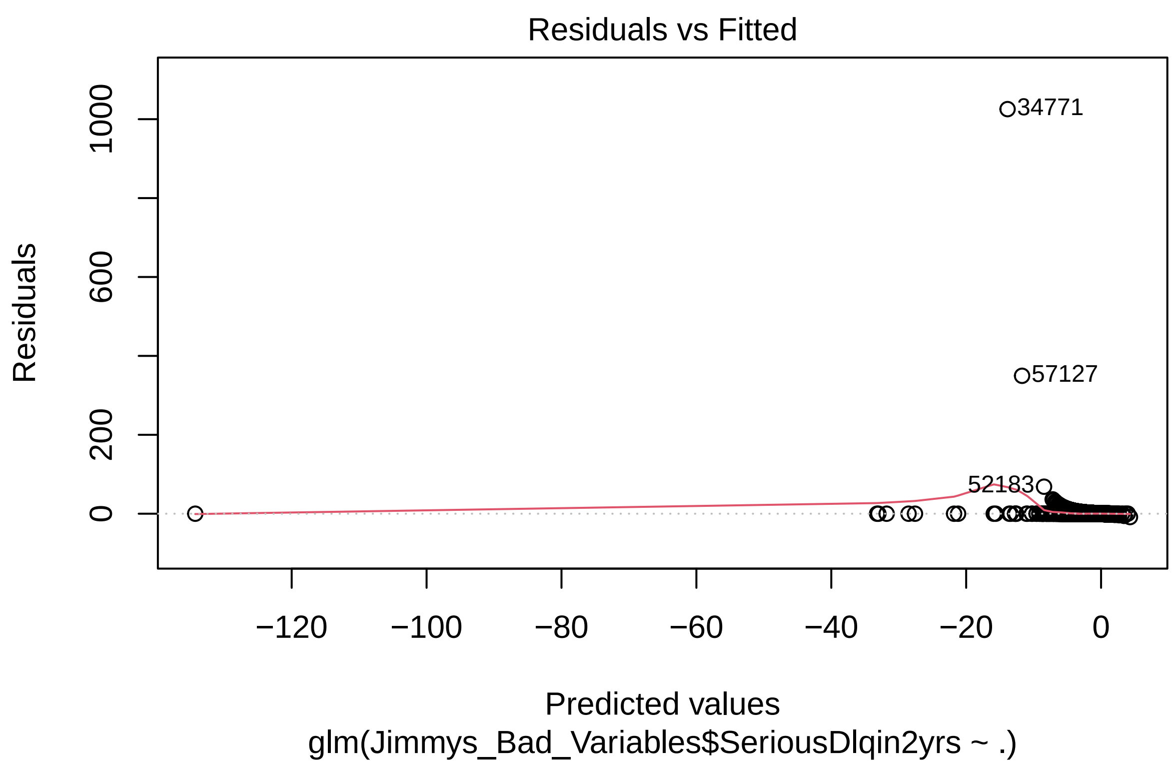
```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
#To Change family link use family = quasi(variance = "mu^3", link = "log") change quasi
summary(model)
```

```
##
## Call:
## glm(formula = Jimmys_Bad_Variables$SeriousDlqin2yrs ~ ., family = "binomial",
##     data = subset(Jimmys_Bad_Variables, select = RevolvingUtilizationOfUnsecuredLines:NumberOfDepende
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9318  -0.3956  -0.3264  -0.2654   5.2661
##
## Coefficients:
##                                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)                          -1.562e+00  6.016e-02 -25.957  < 2e-16 ***
## RevolvingUtilizationOfUnsecuredLines -8.778e-05  1.123e-04  -0.782    0.434
## age                                  -2.411e-02  1.203e-03 -20.045  < 2e-16 ***
## NumberOfTime30.59DaysPastDueNotWorse  4.991e-01  1.537e-02  32.464  < 2e-16 ***
## DebtRatio                            -7.277e-05  5.259e-05  -1.384    0.166
## MonthlyIncome                        -4.383e-05  4.296e-06 -10.202  < 2e-16 ***
## NumberOfOpenCreditLinesAndLoans      -1.905e-03  3.508e-03  -0.543    0.587
## NumberOfTimes90DaysLate               4.205e-01  2.195e-02  19.154  < 2e-16 ***
## NumberRealEstateLoansOrLines          8.801e-02  1.403e-02   6.272 3.57e-10 ***
## NumberOfTime60.89DaysPastDueNotWorse -8.845e-01  2.524e-02 -35.043  < 2e-16 ***
## NumberOfDependents                    1.061e-01  1.250e-02   8.493  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 36033  on 72160  degrees of freedom
## Residual deviance: 33374  on 72150  degrees of freedom
## AIC: 33396
##
## Number of Fisher Scoring iterations: 6
```

```
plot(model, which = 1) #Outliers skew residuals plot
```




```
## Recode the model Orignial Predictors
model <- glm(cs_train$SeriousDlqin2yrs ~ ., family = "binomial", data = train.x)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
AIC_unbalanced[2] <- model$aic
```

```
# Run AIC on the model
model_AIC <- stepAIC(model)
```

```
## Start:  AIC=33396.05
```

```
## cs_train$SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines +
##     age + NumberOfTime30.59DaysPastDueNotWorse + DebtRatio +
##     MonthlyIncome + NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
##     NumberRealEstateLoansOrLines + NumberOfTime60.89DaysPastDueNotWorse +
##     NumberOfDependents

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                                        Df Deviance   AIC
## - NumberOfOpenCreditLinesAndLoans       1    33374 33394
## - RevolvingUtilizationOfUnsecuredLines  1    33375 33395
## <none>                                       33374 33396
## - DebtRatio                             1    33376 33396
## - NumberRealEstateLoansOrLines          1    33412 33432
## - NumberOfDependents                    1    33444 33464
## - MonthlyIncome                         1    33504 33524
## - NumberOfTimes90DaysLate               1    33738 33758
## - age                                   1    33796 33816
## - NumberOfTime30.59DaysPastDueNotWorse  1    34316 34336
## - NumberOfTime60.89DaysPastDueNotWorse  1    34539 34559

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=33394.35
## cs_train$SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines +
##     age + NumberOfTime30.59DaysPastDueNotWorse + DebtRatio +
##     MonthlyIncome + NumberOfTimes90DaysLate + NumberRealEstateLoansOrLines +
##     NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                                        Df Deviance   AIC
## - RevolvingUtilizationOfUnsecuredLines  1    33375 33393
## <none>                                       33374 33394
## - DebtRatio                             1    33377 33395
## - NumberRealEstateLoansOrLines          1    33415 33433
## - NumberOfDependents                    1    33444 33462
## - MonthlyIncome                         1    33508 33526
## - NumberOfTimes90DaysLate               1    33744 33762
## - age                                   1    33816 33834
## - NumberOfTime30.59DaysPastDueNotWorse  1    34323 34341
## - NumberOfTime60.89DaysPastDueNotWorse  1    34540 34558

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=33393.21
## cs_train$SeriousDlqin2yrs ~ age + NumberOfTime30.59DaysPastDueNotWorse +
##     DebtRatio + MonthlyIncome + NumberOfTimes90DaysLate + NumberRealEstateLoansOrLines +
##     NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                                        Df Deviance   AIC
## <none>                                       33375 33393
## - DebtRatio                             1    33378 33394
## - NumberRealEstateLoansOrLines          1    33416 33432
## - NumberOfDependents                    1    33445 33461
## - MonthlyIncome                         1    33509 33525
## - NumberOfTimes90DaysLate               1    33745 33761
## - age                                   1    33816 33832
## - NumberOfTime30.59DaysPastDueNotWorse  1    34324 34340
## - NumberOfTime60.89DaysPastDueNotWorse  1    34541 34557
```

```r
model_AIC
```

```
##
## Call:  glm(formula = cs_train$SeriousDlqin2yrs ~ age + NumberOfTime30.59DaysPastDueNotWorse +
##     DebtRatio + MonthlyIncome + NumberOfTimes90DaysLate + NumberRealEstateLoansOrLines +
##     NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents,
##     family = "binomial", data = train.x)
##
## Coefficients:
##                          (Intercept)                                   age
##                           -1.567e+00                            -2.422e-02
## NumberOfTime30.59DaysPastDueNotWorse                             DebtRatio
##                            4.981e-01                            -7.371e-05
##                        MonthlyIncome               NumberOfTimes90DaysLate
##                           -4.418e-05                             4.219e-01
##         NumberRealEstateLoansOrLines  NumberOfTime60.89DaysPastDueNotWorse
##                            8.517e-02                            -8.849e-01
##                   NumberOfDependents
##                            1.061e-01
##
## Degrees of Freedom: 72160 Total (i.e. Null);  72152 Residual
## Null Deviance:       36030
## Residual Deviance: 33380     AIC: 33390
```

2 b) GLM Extreme Binned Predictors with Unbalanced Data

```r
model2 <- glm(Jimmys_Bad_Variables$SeriousDlqin2yrs ~ MonthlyIncome + Jimmys_Bad_Variables[,13] + Jimmys
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
model2
```

```
##
## Call:  glm(formula = Jimmys_Bad_Variables$SeriousDlqin2yrs ~ MonthlyIncome +
##     Jimmys_Bad_Variables[, 13] + Jimmys_Bad_Variables[, 14] +
##     Jimmys_Bad_Variables[, 15] + Jimmys_Bad_Variables[, 16] +
##     Jimmys_Bad_Variables[, 17] + Jimmys_Bad_Variables[, 18] +
##     Jimmys_Bad_Variables[, 19] + Jimmys_Bad_Variables[, 20] +
##     Jimmys_Bad_Variables[, 22], family = "binomial", data = Jimmys_Bad_Variables)
##
## Coefficients:
##                (Intercept)               MonthlyIncome
##                 -2.336e+00                  -4.536e-05
## Jimmys_Bad_Variables[, 13]  Jimmys_Bad_Variables[, 14]
##                         NA                  -7.691e+00
## Jimmys_Bad_Variables[, 15]  Jimmys_Bad_Variables[, 16]
##                 -7.946e+00                          NA
## Jimmys_Bad_Variables[, 17]  Jimmys_Bad_Variables[, 18]
##                 -5.862e+00                          NA
## Jimmys_Bad_Variables[, 19]  Jimmys_Bad_Variables[, 20]
```

```
##                  -8.139e+00                          NA
## Jimmys_Bad_Variables[, 22]
##                          NA
##
## Degrees of Freedom: 72160 Total (i.e. Null);  72155 Residual
## Null Deviance:       36030
## Residual Deviance: 35860     AIC: 35870
```

```
model2$rank # equals 7 which is how many variables are not NA
```

```
## [1] 6
```

```
#Certain Dummy Variables are a  Singular Matrix Meaning that some of our variables can be constructed u
#Not Sure what to do, but I'll remove these variables in the meantime
```

2 c) GLM Original Predictors Using Balanced Data

```
model_balanced <- glm(balanced_dataset.1$SeriousDlqin2yrs ~ ., data = balanced_dataset.1)
summary(model_balanced)
```

```
##
## Call:
## glm(formula = balanced_dataset.1$SeriousDlqin2yrs ~ ., data = balanced_dataset.1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2672  -0.4510  -0.2253   0.4675   1.9283
##
## Coefficients:
##                                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)                           7.414e-01  1.935e-02  38.322  < 2e-16 ***
## RevolvingUtilizationOfUnsecuredLines -6.955e-06  9.127e-06  -0.762    0.446
## age                                  -6.129e-03  3.626e-04 -16.902  < 2e-16 ***
## NumberOfTime30.59DaysPastDueNotWorse  8.424e-02  4.481e-03  18.800  < 2e-16 ***
## DebtRatio                             6.081e-06  1.090e-05   0.558    0.577
## MonthlyIncome                        -7.979e-07  1.315e-07  -6.068 1.34e-09 ***
## NumberOfOpenCreditLinesAndLoans      -1.151e-03  1.010e-03  -1.140    0.254
## NumberOfTimes90DaysLate               4.436e-02  5.254e-03   8.443  < 2e-16 ***
## NumberRealEstateLoansOrLines          6.004e-03  3.859e-03   1.556    0.120
## NumberOfTime60.89DaysPastDueNotWorse -1.217e-01  6.108e-03 -19.917  < 2e-16 ***
## NumberOfDependents                    1.844e-02  4.020e-03   4.588 4.53e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.2261911)
##
##     Null deviance: 2510.6  on 10044  degrees of freedom
## Residual deviance: 2269.6  on 10034  degrees of freedom
## AIC: 13589
##
## Number of Fisher Scoring iterations: 2
```

```r
AIC_balanced[2] <- model_balanced$aic
```

```r
stepAIC(model_balanced) #remove Debt Ratio and NumberOfOpenCreditLinesAndLoans
```

```
## Start:  AIC=13588.83
## balanced_dataset.1$SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines +
##     age + NumberOfTime30.59DaysPastDueNotWorse + DebtRatio +
##     MonthlyIncome + NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
##     NumberRealEstateLoansOrLines + NumberOfTime60.89DaysPastDueNotWorse +
##     NumberOfDependents
##
##                                        Df Deviance   AIC
## - DebtRatio                             1   2269.7 13587
## - RevolvingUtilizationOfUnsecuredLines  1   2269.7 13587
## - NumberOfOpenCreditLinesAndLoans       1   2269.9 13588
## <none>                                      2269.6 13589
## - NumberRealEstateLoansOrLines          1   2270.2 13589
## - NumberOfDependents                    1   2274.4 13608
## - MonthlyIncome                         1   2277.9 13624
## - NumberOfTimes90DaysLate               1   2285.7 13658
## - age                                   1   2334.2 13869
## - NumberOfTime30.59DaysPastDueNotWorse  1   2349.6 13935
## - NumberOfTime60.89DaysPastDueNotWorse  1   2359.3 13976
##
## Step:  AIC=13587.14
## balanced_dataset.1$SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines +
##     age + NumberOfTime30.59DaysPastDueNotWorse + MonthlyIncome +
##     NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
##     NumberRealEstateLoansOrLines + NumberOfTime60.89DaysPastDueNotWorse +
##     NumberOfDependents
##
##                                        Df Deviance   AIC
## - RevolvingUtilizationOfUnsecuredLines  1   2269.8 13586
## - NumberOfOpenCreditLinesAndLoans       1   2270.0 13586
## <none>                                      2269.7 13587
## - NumberRealEstateLoansOrLines          1   2270.2 13588
## - NumberOfDependents                    1   2274.4 13606
## - MonthlyIncome                         1   2278.0 13622
## - NumberOfTimes90DaysLate               1   2285.8 13656
## - age                                   1   2334.2 13867
## - NumberOfTime30.59DaysPastDueNotWorse  1   2349.6 13933
## - NumberOfTime60.89DaysPastDueNotWorse  1   2359.4 13975
##
## Step:  AIC=13585.72
## balanced_dataset.1$SeriousDlqin2yrs ~ age + NumberOfTime30.59DaysPastDueNotWorse +
##     MonthlyIncome + NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
##     NumberRealEstateLoansOrLines + NumberOfTime60.89DaysPastDueNotWorse +
##     NumberOfDependents
##
##                                        Df Deviance   AIC
## - NumberOfOpenCreditLinesAndLoans       1   2270.1 13585
## <none>                                      2269.8 13586
## - NumberRealEstateLoansOrLines          1   2270.4 13586
```

```
## - NumberOfDependents                     1   2274.6 13605
## - MonthlyIncome                          1   2278.2 13621
## - NumberOfTimes90DaysLate                1   2285.9 13655
## - age                                    1   2334.4 13865
## - NumberOfTime30.59DaysPastDueNotWorse  1   2349.8 13932
## - NumberOfTime60.89DaysPastDueNotWorse  1   2359.6 13974
##
## Step:  AIC=13585.02
## balanced_dataset.1$SeriousDlqin2yrs ~ age + NumberOfTime30.59DaysPastDueNotWorse +
##     MonthlyIncome + NumberOfTimes90DaysLate + NumberRealEstateLoansOrLines +
##     NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents
##
##                                        Df Deviance   AIC
## - NumberRealEstateLoansOrLines          1   2270.4 13584
## <none>                                      2270.1 13585
## - NumberOfDependents                    1   2274.9 13604
## - MonthlyIncome                         1   2278.6 13620
## - NumberOfTimes90DaysLate               1   2287.1 13658
## - age                                   1   2338.7 13882
## - NumberOfTime30.59DaysPastDueNotWorse  1   2350.0 13930
## - NumberOfTime60.89DaysPastDueNotWorse  1   2360.1 13974
##
## Step:  AIC=13584.44
## balanced_dataset.1$SeriousDlqin2yrs ~ age + NumberOfTime30.59DaysPastDueNotWorse +
##     MonthlyIncome + NumberOfTimes90DaysLate + NumberOfTime60.89DaysPastDueNotWorse +
##     NumberOfDependents
##
##                                        Df Deviance   AIC
## <none>                                      2270.4 13584
## - NumberOfDependents                    1   2275.4 13604
## - MonthlyIncome                         1   2278.7 13619
## - NumberOfTimes90DaysLate               1   2287.1 13656
## - age                                   1   2338.9 13881
## - NumberOfTime30.59DaysPastDueNotWorse  1   2351.7 13936
## - NumberOfTime60.89DaysPastDueNotWorse  1   2360.3 13972

##
## Call:  glm(formula = balanced_dataset.1$SeriousDlqin2yrs ~ age + NumberOfTime30.59DaysPastDueNotWors
##     MonthlyIncome + NumberOfTimes90DaysLate + NumberOfTime60.89DaysPastDueNotWorse +
##     NumberOfDependents, data = balanced_dataset.1)
##
## Coefficients:
##                          (Intercept)                                   age
##                            7.387e-01                            -6.150e-03
## NumberOfTime30.59DaysPastDueNotWorse                         MonthlyIncome
##                            8.414e-02                            -7.935e-07
##              NumberOfTimes90DaysLate  NumberOfTime60.89DaysPastDueNotWorse
##                            4.456e-02                            -1.217e-01
##                   NumberOfDependents
##                            1.883e-02
##
## Degrees of Freedom: 10044 Total (i.e. Null);  10038 Residual
## Null Deviance:       2511
## Residual Deviance: 2270  AIC: 13580
```

```r
model_balanced_final <- glm(formula = balanced_dataset.1$SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecu
    age + NumberOfTime30.59DaysPastDueNotWorse + MonthlyIncome +
    NumberOfTimes90DaysLate + NumberRealEstateLoansOrLines +
    NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents,
    data = balanced_dataset.1)
summary(model_balanced_final)
```

```
##
## Call:
## glm(formula = balanced_dataset.1$SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines +
##     age + NumberOfTime30.59DaysPastDueNotWorse + MonthlyIncome +
##     NumberOfTimes90DaysLate + NumberRealEstateLoansOrLines +
##     NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents,
##     data = balanced_dataset.1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2685  -0.4509  -0.2253   0.4661   1.9416
##
## Coefficients:
##                                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)                           7.374e-01  1.904e-02  38.737  < 2e-16 ***
## RevolvingUtilizationOfUnsecuredLines -6.892e-06  9.126e-06  -0.755    0.450
## age                                  -6.203e-03  3.561e-04 -17.417  < 2e-16 ***
## NumberOfTime30.59DaysPastDueNotWorse  8.367e-02  4.453e-03  18.789  < 2e-16 ***
## MonthlyIncome                        -8.029e-07  1.314e-07  -6.109 1.04e-09 ***
## NumberOfTimes90DaysLate               4.517e-02  5.210e-03   8.669  < 2e-16 ***
## NumberRealEstateLoansOrLines          4.168e-03  3.483e-03   1.197    0.231
## NumberOfTime60.89DaysPastDueNotWorse -1.218e-01  6.107e-03 -19.947  < 2e-16 ***
## NumberOfDependents                    1.842e-02  4.019e-03   4.582 4.66e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.2261825)
##
##     Null deviance: 2510.6  on 10044  degrees of freedom
## Residual deviance: 2270.0  on 10036  degrees of freedom
## AIC: 13586
##
## Number of Fisher Scoring iterations: 2
```

```r
model3 <- glm(Jimmys_Bad_Variables$SeriousDlqin2yrs ~ MonthlyIncome + Jimmys_Bad_Variables[,15] + Jimmys
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
model3
```

```
##
## Call:  glm(formula = Jimmys_Bad_Variables$SeriousDlqin2yrs ~ MonthlyIncome +
##     Jimmys_Bad_Variables[, 15] + Jimmys_Bad_Variables[, 16] +
##     Jimmys_Bad_Variables[, 17] + Jimmys_Bad_Variables[, 18] +
##     Jimmys_Bad_Variables[, 19], family = "binomial", data = Jimmys_Bad_Variables)
```

```
##
## Coefficients:
##                (Intercept)               MonthlyIncome
##                 -2.336e+00                  -4.536e-05
## Jimmys_Bad_Variables[, 15]  Jimmys_Bad_Variables[, 16]
##                 -7.946e+00                          NA
## Jimmys_Bad_Variables[, 17]  Jimmys_Bad_Variables[, 18]
##                 -5.862e+00                          NA
## Jimmys_Bad_Variables[, 19]
##                 -8.139e+00
##
## Degrees of Freedom: 72160 Total (i.e. Null);  72156 Residual
## Null Deviance:       36030
## Residual Deviance: 35860     AIC: 35870
```

```
model4 <- glm(Jimmys_Bad_Variables$SeriousDlqin2yrs ~ MonthlyIncome + Jimmys_Bad_Variables[,15] + Jimmys
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
model4
```

```
##
## Call:  glm(formula = Jimmys_Bad_Variables$SeriousDlqin2yrs ~ MonthlyIncome +
##     Jimmys_Bad_Variables[, 15] + Jimmys_Bad_Variables[, 16] +
##     Jimmys_Bad_Variables[, 17] + Jimmys_Bad_Variables[, 18] +
##     Jimmys_Bad_Variables[, 19], family = binomial(link = "probit"),
##     data = Jimmys_Bad_Variables)
##
## Coefficients:
##                (Intercept)               MonthlyIncome
##                 -1.385e+00                  -1.658e-05
## Jimmys_Bad_Variables[, 15]  Jimmys_Bad_Variables[, 16]
##                 -2.677e+00                          NA
## Jimmys_Bad_Variables[, 17]  Jimmys_Bad_Variables[, 18]
##                 -1.915e+00                          NA
## Jimmys_Bad_Variables[, 19]
##                 -2.748e+00
##
## Degrees of Freedom: 72160 Total (i.e. Null);  72156 Residual
## Null Deviance:       36030
## Residual Deviance: 35890     AIC: 35900
```

```
model5 <- glm(Jimmys_Bad_Variables$SeriousDlqin2yrs ~ MonthlyIncome + Jimmys_Bad_Variables[,15] + Jimmys
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
model5
```

```
##
## Call:  glm(formula = Jimmys_Bad_Variables$SeriousDlqin2yrs ~ MonthlyIncome +
##     Jimmys_Bad_Variables[, 15] + Jimmys_Bad_Variables[, 16] +
```

```
##     Jimmys_Bad_Variables[, 17] + Jimmys_Bad_Variables[, 18] +
##     Jimmys_Bad_Variables[, 19], family = binomial(link = "cloglog"),
##     data = Jimmys_Bad_Variables)
##
## Coefficients:
##               (Intercept)               MonthlyIncome
##                -2.3775977                  -0.0000445
## Jimmys_Bad_Variables[, 15]  Jimmys_Bad_Variables[, 16]
##                -7.8205053                          NA
## Jimmys_Bad_Variables[, 17]  Jimmys_Bad_Variables[, 18]
##                -5.7758409                          NA
## Jimmys_Bad_Variables[, 19]
##                -8.0096201
##
## Degrees of Freedom: 72160 Total (i.e. Null);  72156 Residual
## Null Deviance:       36030
## Residual Deviance: 35850     AIC: 35860
```

```
anova(model,model2,model3,model4,model5) #model 2 is the one with NA values
```

```
## Warning in anova.glmlist(c(list(object), dotargs), dispersion = dispersion, :
## c("models with response 'c(\"Jimmys_Bad_Variables$SeriousDlqin2yrs\",
## \"Jimmys_Bad_Variables$SeriousDlqin2yrs\", ' removed because response differs
## from model 1", "models with response '\"Jimmys_Bad_Variables$SeriousDlqin2yrs\",
## \"Jimmys_Bad_Variables$SeriousDlqin2yrs\"' removed because response differs from
## model 1", "models with response ')' removed because response differs from model
## 1")

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: cs_train$SeriousDlqin2yrs
##
## Terms added sequentially (first to last)
##
##
##                                      Df Deviance Resid. Df Resid. Dev
## NULL                                                 72160      36033
## RevolvingUtilizationOfUnsecuredLines  1     0.98     72159      36032
## age                                   1   725.52     72158      35306
## NumberOfTime30.59DaysPastDueNotWorse  1   338.11     72157      34968
## DebtRatio                             1     0.06     72156      34968
```

```
## MonthlyIncome                          1    75.45     72155      34892
## NumberOfOpenCreditLinesAndLoans        1    14.61     72154      34878
## NumberOfTimes90DaysLate                1   205.44     72153      34672
## NumberRealEstateLoansOrLines           1    39.50     72152      34633
## NumberOfTime60.89DaysPastDueNotWorse   1  1189.20     72151      33444
## NumberOfDependents                     1    69.70     72150      33374
```

```
#model 1 is the best but we can check model 5 using our dummy variables
```

Method 3: Ridge

3 a) Ridge with Unbalanced Data

```
x <- data.frame(training_dataset)
x <- na.omit(x) #Remember Monthly Income contains NA values
x_for_ridge <- data.matrix(x[,-1]) #Everything but Response Variable
ridge_model5 <- cv.glmnet(x_for_ridge,x[,1], alpha = 0, standardize = TRUE, nfolds = length(x))
ridge_model5
```

```
##
## Call:  cv.glmnet(x = x_for_ridge, y = x[, 1], nfolds = length(x), alpha = 0,      standardize = TRUE)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min  275.9    25   87606 35789      10
## 1se 2573.2     1   87615 35791      10
```

```
Log.Likelihood <- ridge_model5$glmnet.fit$nulldev - ridge_model5$glmnet.fit$nulldev * (1- ridge_model5$g
k <- ridge_model5$glmnet.fit$df[which(ridge_model5$glmnet.fit$lambda == ridge_model5$lambda.min)]
n <- ridge_model5$glmnet.fit$nobs
AICc <- -Log.Likelihood + 2 * k + 2 * k * (k+1)/(n-k-1)
AIC_unbalanced[3] <- AICc
```

3 b) Ridge with Balanced Data

```
x_for_ridge_balanced <- data.matrix(balanced_dataset.1[,-1])
ridge_model_balanced <- cv.glmnet(x_for_ridge_balanced,balanced_dataset.1[,1], alpha = 0, standardize =
ridge_model_balanced
```

```
##
## Call:  cv.glmnet(x = x_for_ridge_balanced, y = balanced_dataset.1[,      1], nfolds = length(balance
##
## Measure: Mean-Squared Error
##
##      Lambda Index Measure       SE Nonzero
## min 0.00953   100  0.2339 0.003373      10
## 1se 0.03193    87  0.2371 0.003105      10
```

```
Log.Likelihood <- ridge_model_balanced$glmnet.fit$nulldev - ridge_model_balanced$glmnet.fit$nulldev * (
k <- ridge_model_balanced$glmnet.fit$df[which(ridge_model_balanced$glmnet.fit$lambda == ridge_model_bala
n <- ridge_model_balanced$glmnet.fit$nobs
AICc <- -Log.Likelihood + 2 * k + 2 * k * (k+1)/(n-k-1)
AIC_balanced[3] <- AICc
```

Method 4: Lasso

4 a) Lasso with Unbalanced Data

```
lasso_model5 <- cv.glmnet(x_for_ridge,x[,1], alpha = 1, standardize = TRUE, nfolds = length(x))
lasso_model5
```

```
##
## Call:  cv.glmnet(x = x_for_ridge, y = x[, 1], nfolds = length(x), alpha = 1,      standardize = TRUE
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min 0.3323    23   87592 33077       6
## 1se 2.5732     1   87615 33082       0
```

```
Log.Likelihood <- lasso_model5$glmnet.fit$nulldev - lasso_model5$glmnet.fit$nulldev * (1- lasso_model5$g
k <- lasso_model5$glmnet.fit$df[which(lasso_model5$glmnet.fit$lambda == lasso_model5$lambda.min)]
n <- lasso_model5$glmnet.fit$nobs
AICc <- -Log.Likelihood + 2 * k + 2 * k * (k+1)/(n-k-1)
AICc
```

```
## [1] -2142095
```

```
AIC_unbalanced[4] <- AICc
```

4 b) Lasso with Balanced Data

```
lasso_model_balanced <- cv.glmnet(x_for_ridge_balanced,balanced_dataset.1[,1], alpha = 1, standardize =
lasso_model_balanced
```

```
##
## Call:  cv.glmnet(x = x_for_ridge_balanced, y = balanced_dataset.1[,      1], nfolds = length(balance
##
## Measure: Mean-Squared Error
##
##       Lambda Index Measure       SE Nonzero
## min 0.000170    69   0.2294 0.003206      10
## 1se 0.003345    37   0.2324 0.002600       8
```

```
Log.Likelihood <- lasso_model_balanced$glmnet.fit$nulldev - lasso_model_balanced$glmnet.fit$nulldev * (
k <- lasso_model_balanced$glmnet.fit$df[which(lasso_model_balanced$glmnet.fit$lambda == lasso_model_bala
n <- lasso_model_balanced$glmnet.fit$nobs
AICc <- -2 * Log.Likelihood + 2 * k + 2 * k * (k+1)/(n-k-1)
AICc
```

```
## [1] -461.6335
```

```
AIC_balanced[4] <- AICc
```

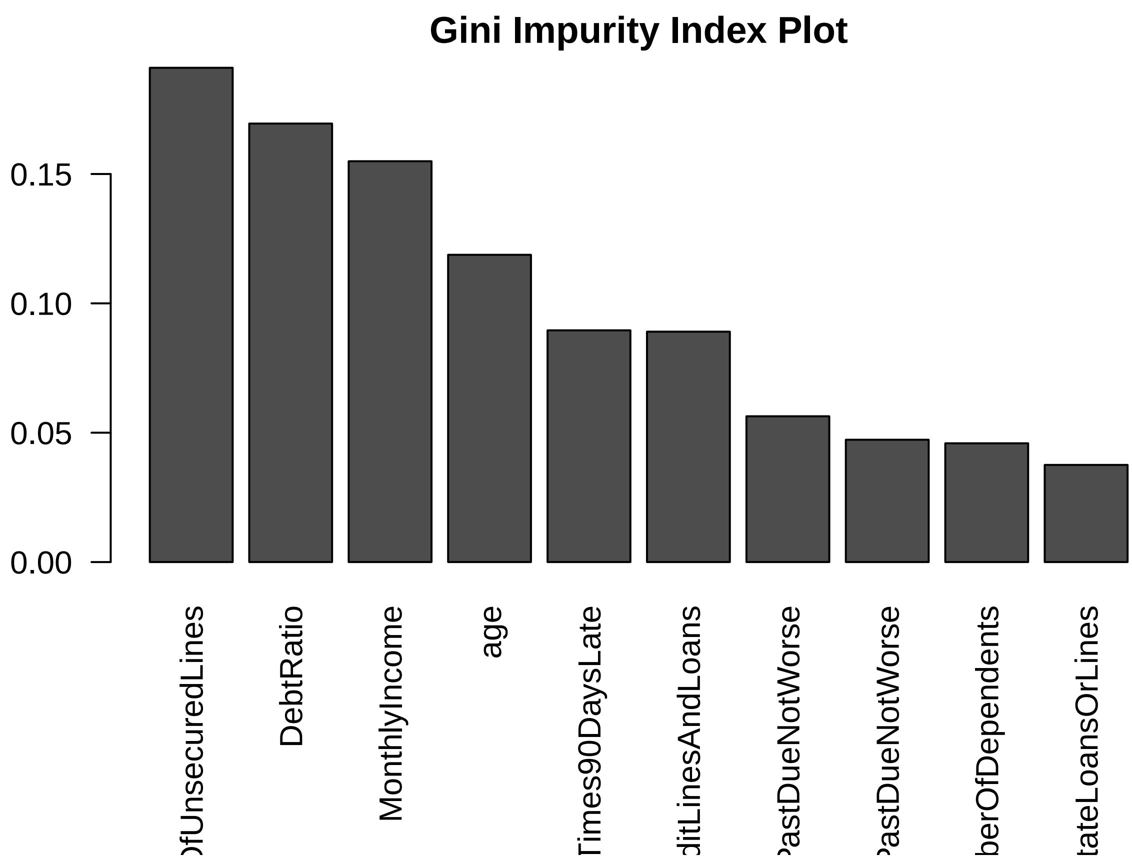
Method 4: Random Forest

4 a) Random Forest with Unbalanced Data

```
# Fits Random Forest
model.rf = randomForest(y = as.factor(train.y), x=train.x, xtest=test.x, ytest=as.factor(test.y), mtry =

model.rf # Output shows confusion matrix for both train and test
```

```
##
## Call:
##  randomForest(x = train.x, y = as.factor(train.y), xtest = test.x,      ytest = as.factor(test.y), mt
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 6.61%
## Confusion matrix:
##       0   1 class.error
## 0 66601 619 0.009208569
## 1  4150 791 0.839910949
##                 Test set error rate: 6.7%
## Confusion matrix:
##       0   1 class.error
## 0 44304 388 0.008681643
## 1  2837 579 0.830503513
```

```
# Random Forest Output
var.imp = data.frame(importance(model.rf, type=2))
var.imp$Variables = row.names(var.imp)
varimp = var.imp[order(var.imp$MeanDecreaseGini,decreasing = T),]
par(mar = c(7.5,3,2,2))
giniplot = barplot(t(varimp[-2]/sum(varimp[-2])),las=2,cex.names=1, main="Gini Impurity Index Plot")
```

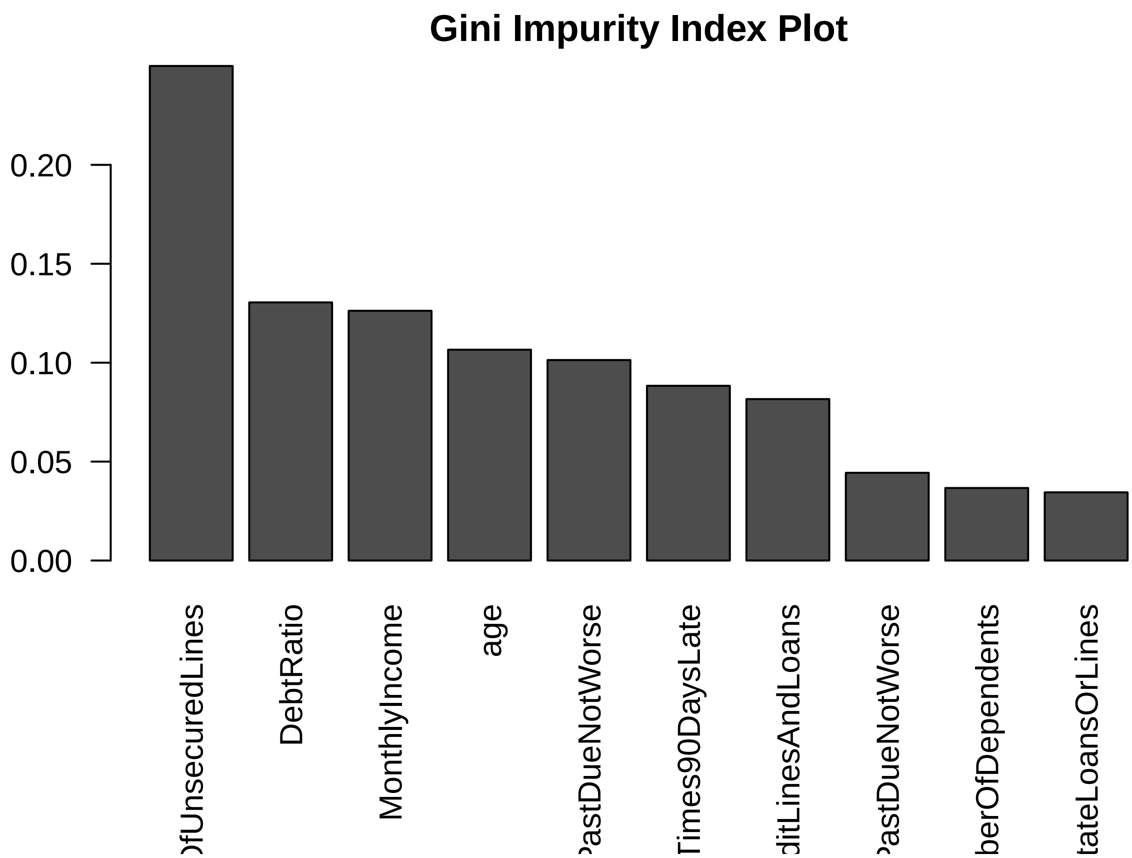


4 b) Random Forest with Balanced Data

```
model_balanced.rf <- randomForest(y = as.factor(balanced_dataset.1$SeriousDlqin2yrs), x=balanced_x, xtes
, na.action = na.omit)

model_balanced.rf # Output shows confusion matrix for both train and test
```

```
##
## Call:
##  randomForest(x = balanced_x, y = as.factor(balanced_dataset.1$SeriousDlqin2yrs),      xtest = test.
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 22.95%
## Confusion matrix:
##      0    1 class.error
## 0 4062 1042   0.2041536
## 1 1263 3678   0.2556163
##                 Test set error rate: 21.46%
## Confusion matrix:
##       0    1 class.error
## 0 35223 9469   0.2118724
## 1   854 2562   0.2500000
```

```r
# Random Forest Output
var_balanced.imp = data.frame(importance(model_balanced.rf, type=2))
var_balanced.imp$Variables = row.names(var_balanced.imp)
var_balancedimp = var_balanced.imp[order(var_balanced.imp$MeanDecreaseGini,decreasing = T),]
par(mar = c(7.5,3,2,2))
giniplot = barplot(t(var_balancedimp[-2]/sum(var_balancedimp[-2])),las=2,cex.names=1, main="Gini Impuri
```




```r
Tree <- (getTree(model_balanced.rf, k = 1, labelVar =  TRUE))
head(Tree)
```

```
##   left daughter right daughter                          split var split point
## 1             2              3 RevolvingUtilizationOfUnsecuredLines   0.4778472
## 2             4              5              NumberOfTimes90DaysLate   0.5000000
## 3             6              7 RevolvingUtilizationOfUnsecuredLines   0.9028845
## 4             8              9                            DebtRatio   0.3846390
## 5            10             11      NumberOfOpenCreditLinesAndLoans  23.5000000
## 6            12             13 NumberOfTime30.59DaysPastDueNotWorse   0.5000000
##   status prediction
## 1      1       <NA>
## 2      1       <NA>
## 3      1       <NA>
## 4      1       <NA>
## 5      1       <NA>
## 6      1       <NA>
```

5. Evaluating Model/Comparing results

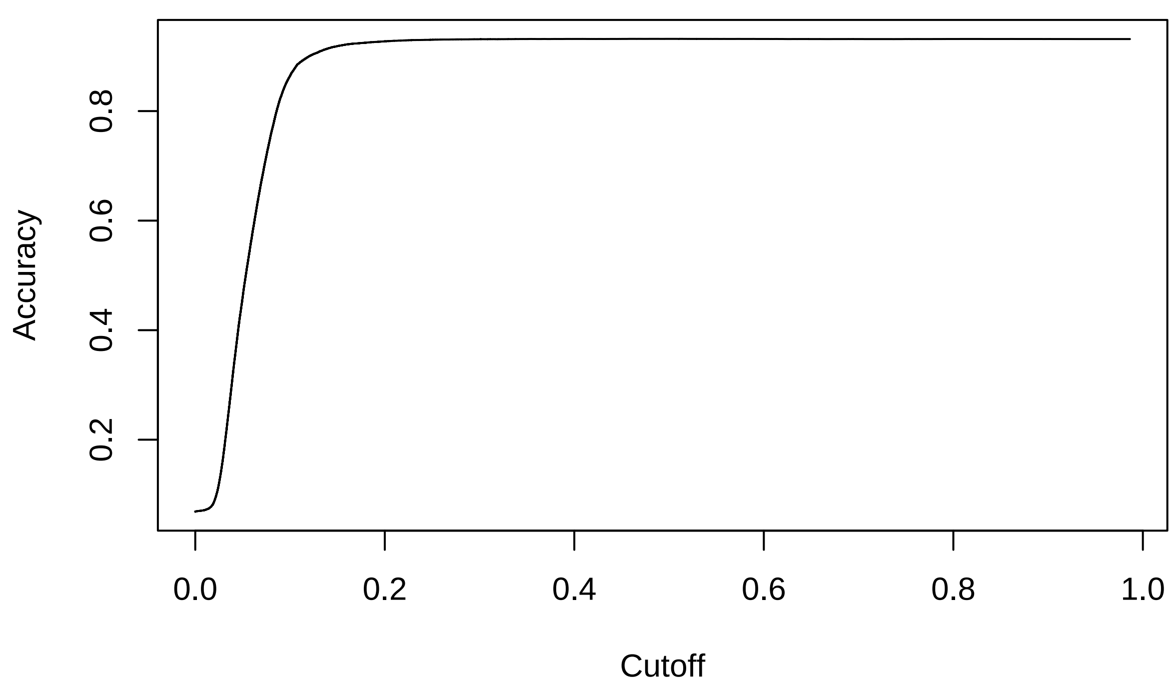
- mmp plots to see if model fits the data, as well as the pearson Chi-square test
- checking for outliers and influential points
- Some measure of pesudo R-square and accuracy of the model
- Use confusion matrix, ROC/AUC curve, AIC to evaluate the different models

5-1 Evaluation on Final Model using Training Data

```r
## Set model as final model
model.final <- model

## Evaluation on Final model using Training Data
train_preds = predict(model.final, newdata=train.x, type="response")
head(train_preds[is.na(train_preds)])
```

```
## named numeric(0)
```

```r
#train.x[c(7,9),]
#train_preds[is.na(train_preds)]

pred_compare = prediction(train_preds, train.y)
plot(performance(pred_compare, "acc"))
```



```r
table(train.y, train_preds>0.2) # accurary on train
```

```
##
## train.y FALSE  TRUE
##       0 66148  1072
##       1  4161   780
```

5-3 Goodness of Fit using Hosmer-Lemeshow Test

a)

The p-value is 0, meaning that we want to reject the null hypothesis that the model is adequate.

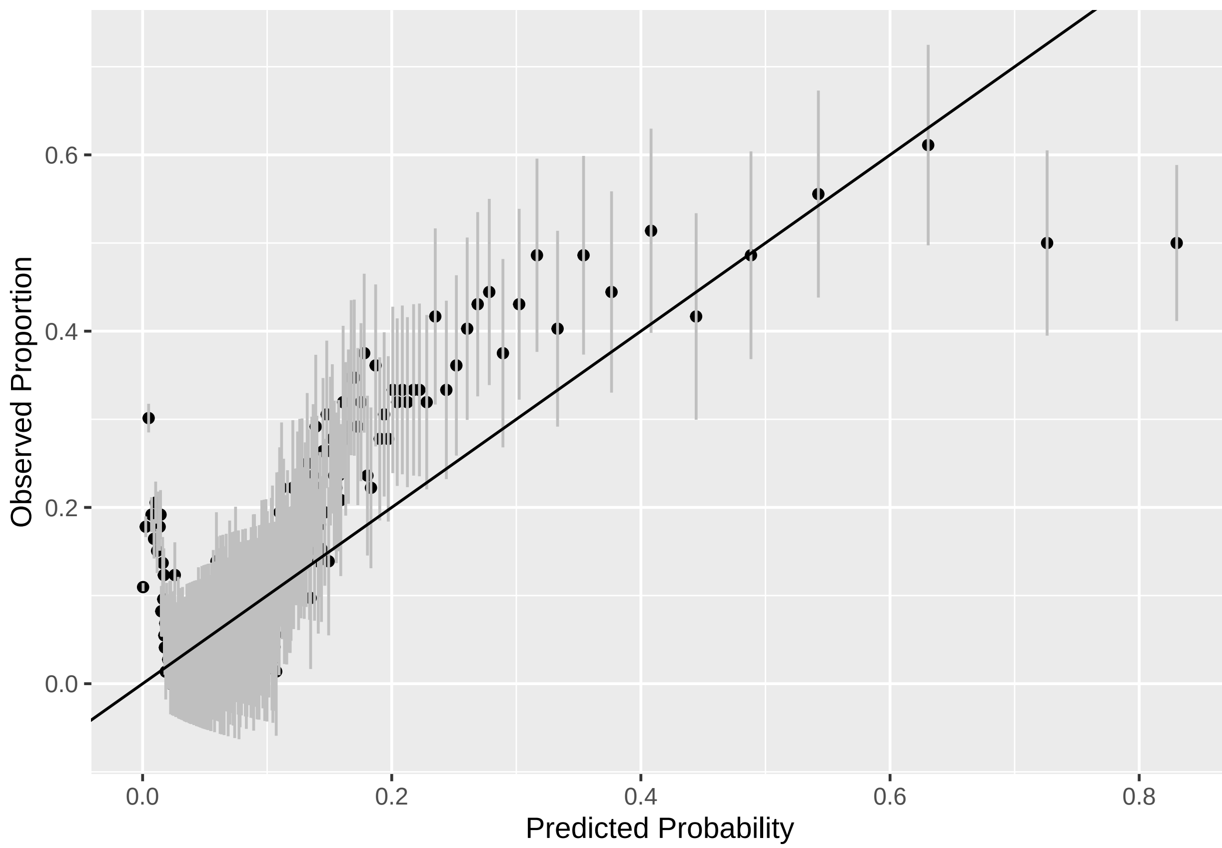
```r
# Goodness of Fit using Hosmer-Lemeshow Test
linpred=predict(model.final)

cs_train_m <- mutate(cs_train, predprob=predict(model.final, type="response"))  # cal p^_i
gdf <- group_by(cs_train_m, ntile(linpred, 1000)) # group up the data by eta_x into 100 groups
hldf <- summarise(gdf, y=sum(SeriousDlqin2yrs==1), ppred=mean(predprob), count=n())
head(hldf)
```

```
## # A tibble: 6 x 4
##   `ntile(linpred, 1000)`     y    ppred count
##                    <int> <int>    <dbl> <int>
## 1                      1     8 0.000387    73
## 2                      2    13 0.00251     73
## 3                      3    22 0.00485     73
## 4                      4    14 0.00717     73
## 5                      5    12 0.00920     73
## 6                      6    15 0.0105      73
```

```r
# We adjust the size of the bins until there's only one group with less than 5
hldf[hldf$count<5,]
```

```
## # A tibble: 0 x 4
## # ... with 4 variables: ntile(linpred, 1000) <int>, y <int>, ppred <dbl>,
## #   count <int>
```

```r
# Observed Proportion Confidence Interval vs Predicted Probability
hldf <- mutate(hldf, se.fit=sqrt(ppred*(1-ppred)/count))

ggplot(hldf,aes(x=ppred,y=y/count,ymin=y/count-2*se.fit, ymax=y/count+2*se.fit))+
  geom_point()+geom_linerange(color=grey(0.75))+
  geom_abline(intercept = 0,slope = 1)+
  xlab("Predicted Probability")+
  ylab("Observed Proportion")
```




```
# Hosmer-Lemeshow statistics
hlstat <- with(hldf, sum((y-count*ppred) ^2/(count * ppred * (1-ppred))))
c(hlstat, nrow(hldf))
```

```
## [1] 7509.678 1000.000
```

```
# The p-value is given by:
1-pchisq(hlstat, nrow(hldf)-2)
```

```
## [1] 0
```

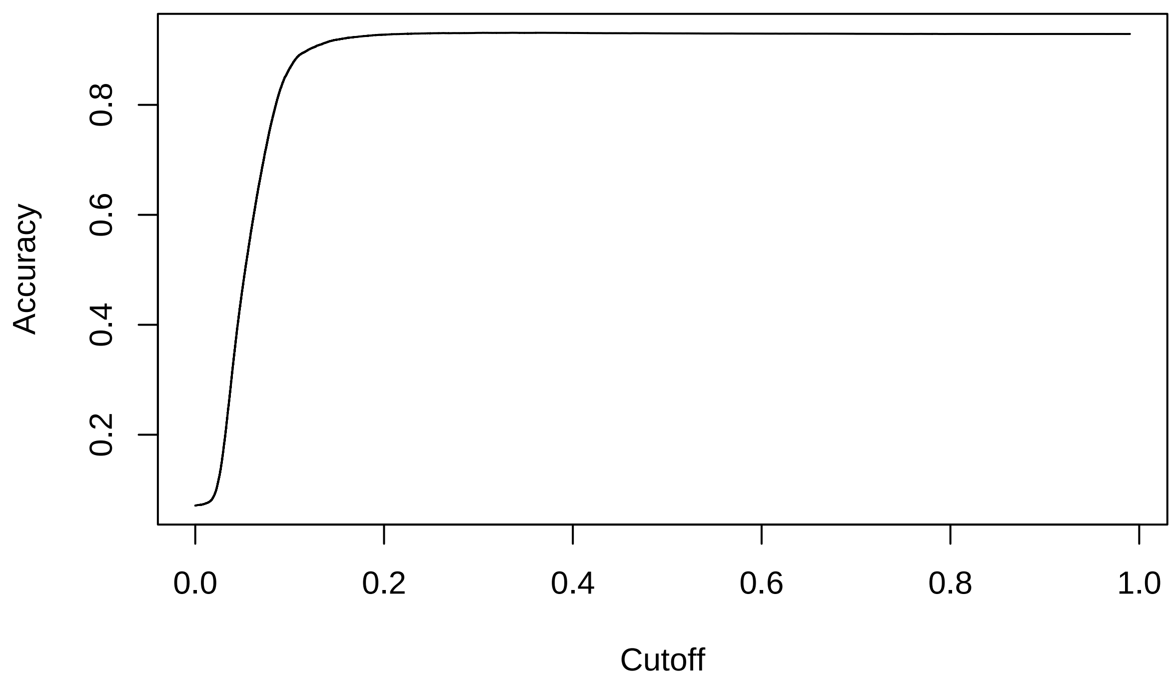
AUC

5-3 Model Performance with Test Data

```
#Final Model(w/ interaction term ReugularMedicine * PhysicallyActive)
result_m2 = predict(model.final, newdata=test.x, type="response")

head(test.y)
```

```
## [1] 0 0 0 0 0 0
```

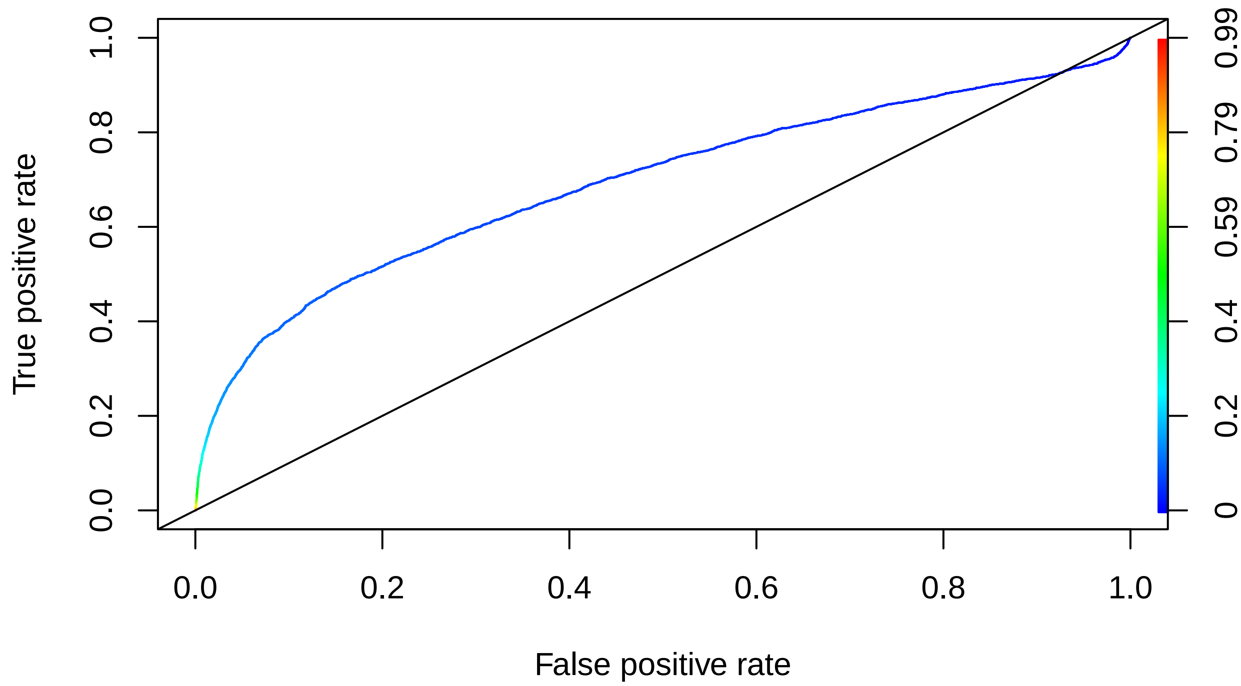
```
pred_m2 = prediction(result_m2, test.y)


plot(performance(pred_m2, "acc")) #It seems like 0.52 cutoff has the highest accuracy
```




```
table(test.y, result_m2>0.2)
```

```
##
## test.y FALSE  TRUE
##      0 44063   629
##      1  2853   563
```

```
#Accuracy :
#Sensitivity :
#Specificity :
#The Specificity and accuracy improved a bit compared to the previous model without interaction term, s
plot(performance(pred_m2,"tpr","fpr"), colorize=T)
abline(0,1)
```




```r
#Now we calculate the area under the curve (AUC) and accuracy of the model given above (glmModel2)
auc_ROCR2 <- performance(pred_m2, measure = "auc")
auc_ROCR2@y.values[[1]]
```

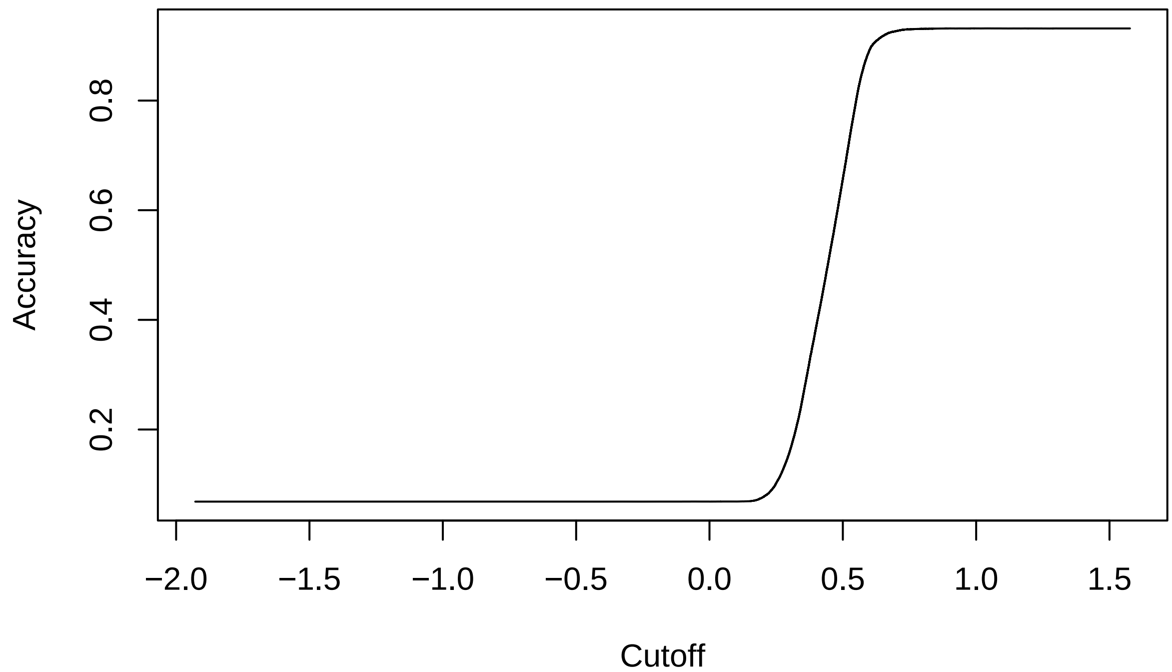
```
## [1] 0.6931242
```

GLM Model with Balanced Dataset

```r
## Set model as final model
model.final <- model_balanced

## Evaluation on Final model using Training Data
train_preds = predict(model.final, newdata=train.x, type="response")
head(train_preds[is.na(train_preds)])
```

```
## named numeric(0)
```

```r
#train.x[c(7,9),]
#train_preds[is.na(train_preds)]

pred_compare = prediction(train_preds, train.y)
plot(performance(pred_compare, "acc"))
```




```r
table(train.y, train_preds>0.66) # accurary on train
```

```
##
## train.y FALSE  TRUE
##       0 65474  1746
##       1  3990   951
```

5-3 Goodness of Fit using Hosmer-Lemeshow Test

a)

The p-value is 0, meaning that we want to reject the null hypothesis that the model is adequate.

```r
# Goodness of Fit using Hosmer-Lemeshow Test
linpred=predict(model.final)

cs_train_m <- mutate(balanced_dataset.1, predprob=predict(model.final, type="response"))  # cal p^_i
gdf <- group_by(cs_train_m, ntile(linpred, 1000)) # group up the data by eta_x into 100 groups
hldf <- summarise(gdf, y=sum(SeriousDlqin2yrs==1), ppred=mean(predprob), count=n())
head(hldf)
```

```
## # A tibble: 6 x 4
##   `ntile(linpred, 1000)`     y   ppred count
##                    <int> <int>   <dbl> <int>
```

```
## 1                      1     5 -0.356     11
## 2                      2     9 -0.0201    11
## 3                      3     9  0.0429    11
## 4                      4     9  0.0854    11
## 5                      5     9  0.115     11
## 6                      6     6  0.138     11
```

```
# We adjust the size of the bins until there's only one group with less than 5
hldf[hldf$count<5,]
```

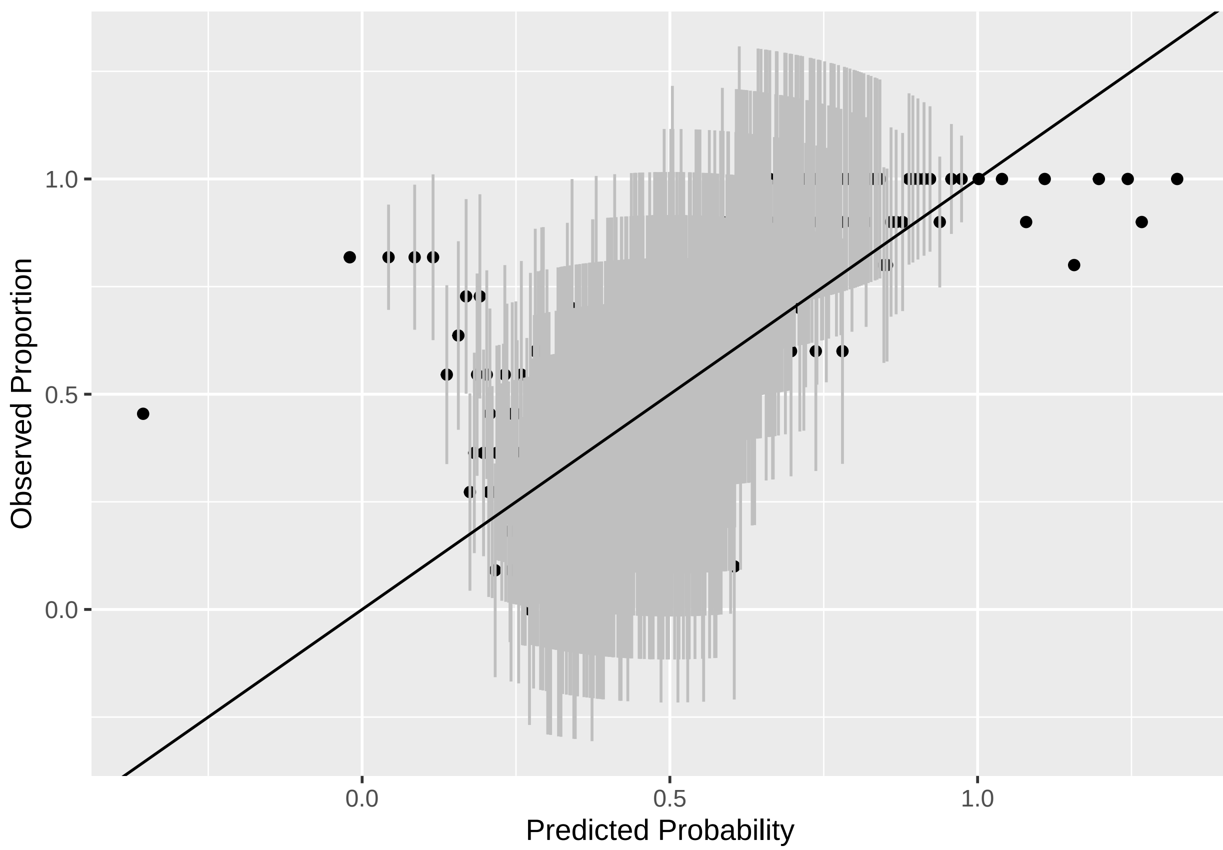
```
## # A tibble: 0 x 4
## # ... with 4 variables: ntile(linpred, 1000) <int>, y <int>, ppred <dbl>,
## #   count <int>
```

```
# Observed Proportion Confidence Interval vs Predicted Probability
hldf <- mutate(hldf, se.fit=sqrt(ppred*(1-ppred)/count))
```

```
## Warning in sqrt(ppred * (1 - ppred)/count): NaNs produced
```

```
ggplot(hldf,aes(x=ppred,y=y/count,ymin=y/count-2*se.fit, ymax=y/count+2*se.fit))+
  geom_point()+geom_linerange(color=grey(0.75))+
  geom_abline(intercept = 0,slope = 1)+
  xlab("Predicted Probability")+
  ylab("Observed Proportion")
```

```
## Warning: Removed 11 rows containing missing values (geom_segment).
```



```r
# Hosmer-Lemeshow statistics
hlstat <- with(hldf, sum((y-count*ppred) ^2/(count * ppred * (1-ppred))))
c(hlstat, nrow(hldf))
```

```
## [1] 1074.56 1000.00
```

```r
# The p-value is given by:
1-pchisq(hlstat, nrow(hldf)-2)
```

```
## [1] 0.04584838
```

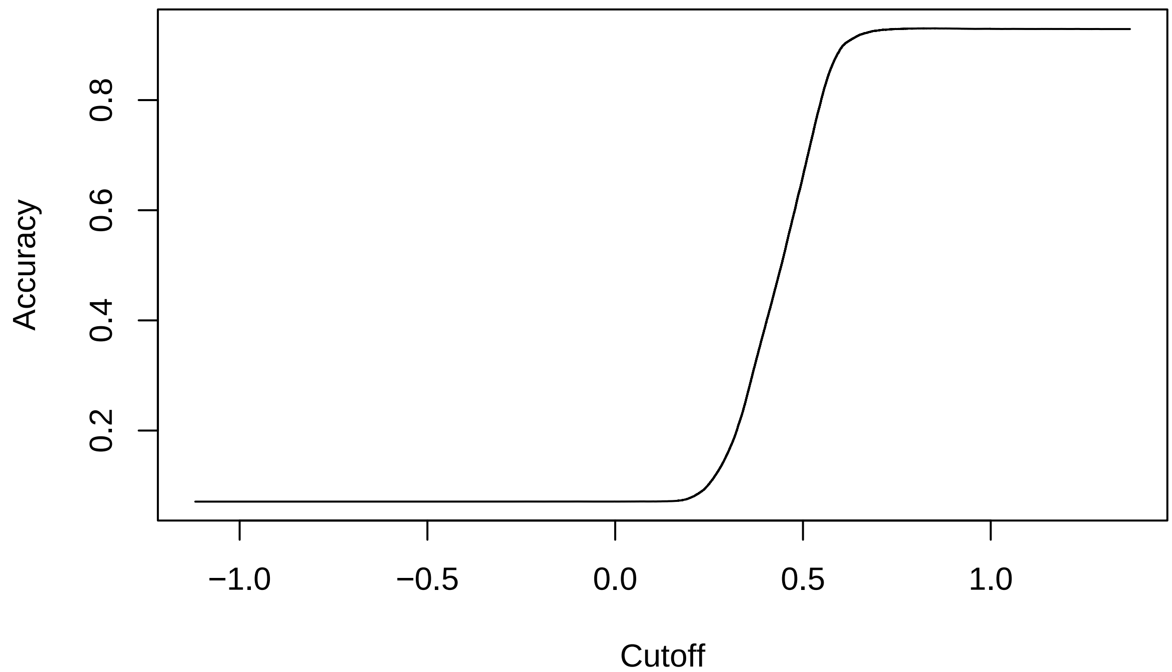
AUC

5-3 Model Performance with Test Data

```r
#Final Model(w/ interaction term ReugularMedicine * PhysicallyActive)
result_m2 = predict(model.final, newdata=test.x, type="response")

head(test.y)
```

```
## [1] 0 0 0 0 0 0
```

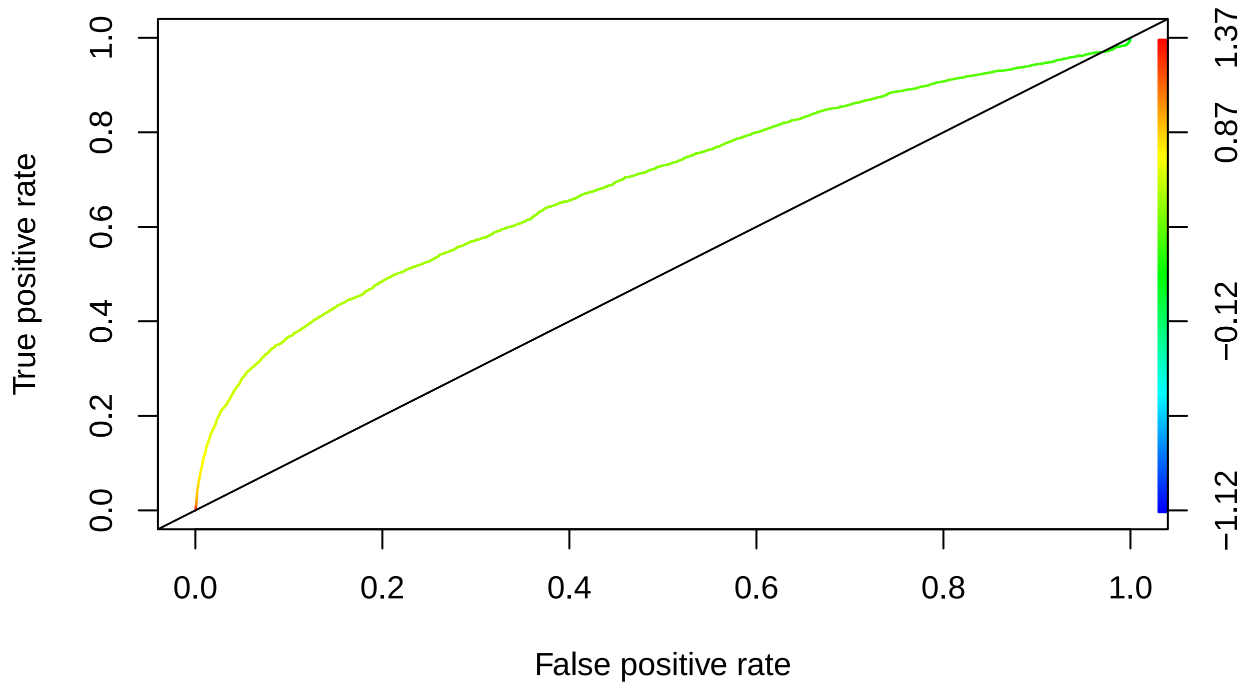
```r
pred_m2 = prediction(result_m2, test.y)


plot(performance(pred_m2, "acc")) #It seems like 0.52 cutoff has the highest accuracy
```




```
table(test.y, result_m2>0.2)
```

```
##
## test.y FALSE  TRUE
##      0   398 44294
##      1    58  3358
```

```
#Accuracy :
#Sensitivity :
#Specificity :
#The Specificity and accuracy improved a bit compared to the previous model without interaction term, s
plot(performance(pred_m2,"tpr","fpr"), colorize=T)
abline(0,1)
```




```
#Now we calculate the area under the curve (AUC) and accuracy of the model given above (glmModel2)
auc_ROCR2 <- performance(pred_m2, measure = "auc")
auc_ROCR2@y.values[[1]]
```

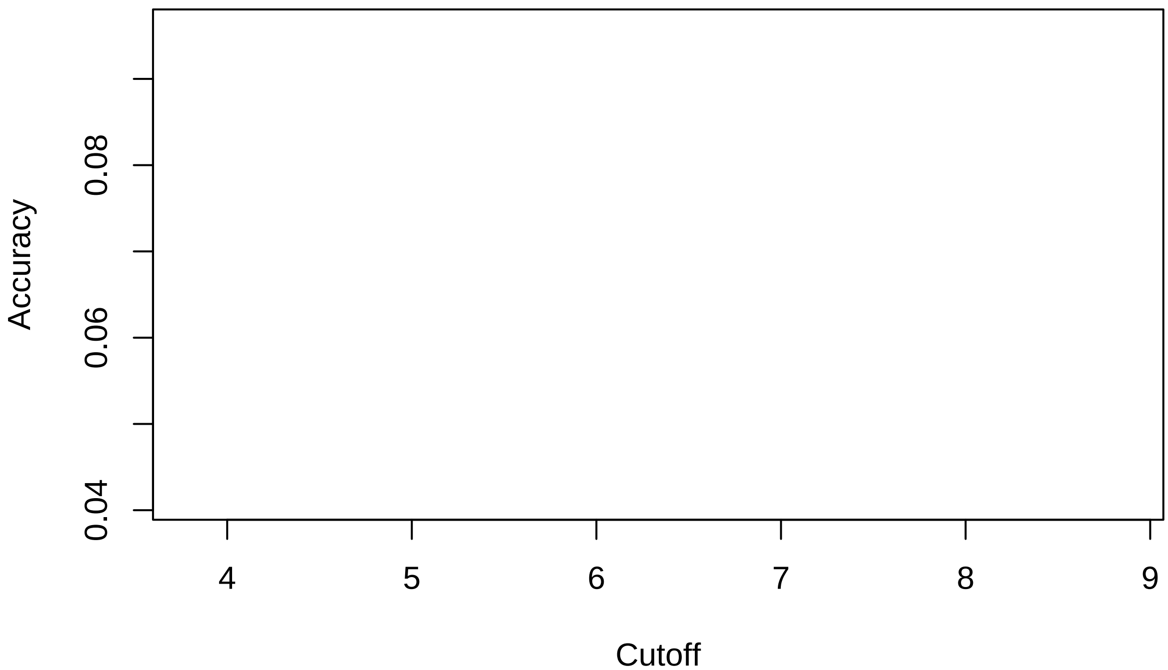
```
## [1] 0.6890155
```

###Ridge Model Unbiased Dataset

```
## Set model as final model
model.final <- ridge_model5

## Evaluation on Final model using Training Data
train_preds = predict(model.final, newx=as.matrix(train.x), type="response")
head(train_preds[is.na(train_preds)])
```

```
## numeric(0)
```

```
#train.x[c(7,9),]
#train_preds[is.na(train_preds)]

pred_compare = prediction(train_preds, train.y)
plot(performance(pred_compare, "acc"))
```




```
table(train.y, train_preds>0.2) # accurary on train
```

```
##
## train.y  TRUE
##       0 67220
##       1  4941
```

5-3 Goodness of Fit using Hosmer-Lemeshow Test

a)

The p-value is 0, meaning that we want to reject the null hypothesis that the model is adequate.

```
# Goodness of Fit using Hosmer-Lemeshow Test
linpred=predict(model.final, newx = as.matrix(train.x), type = "response")

cs_train_m <- mutate(cs_train, predprob=predict(model.final, newx = as.matrix(train.x), type="response")
gdf <- group_by(cs_train_m, ntile(linpred, 1000)) # group up the data by eta_x into 100 groups
hldf <- summarise(gdf, y=sum(SeriousDlqin2yrs==1), ppred=mean(predprob), count=n())
head(hldf)
```

```
## # A tibble: 6 x 4
##   `ntile(linpred, 1000)`     y ppred count
##                    <int> <int> <dbl> <int>
```

```
## 1                       1     6  6.33    73
## 2                       2     6  6.33    73
## 3                       3     4  6.33    73
## 4                       4     3  6.33    73
## 5                       5     8  6.33    73
## 6                       6     3  6.33    73
```

```
# We adjust the size of the bins until there's only one group with less than 5
hldf[hldf$count<5,]
```

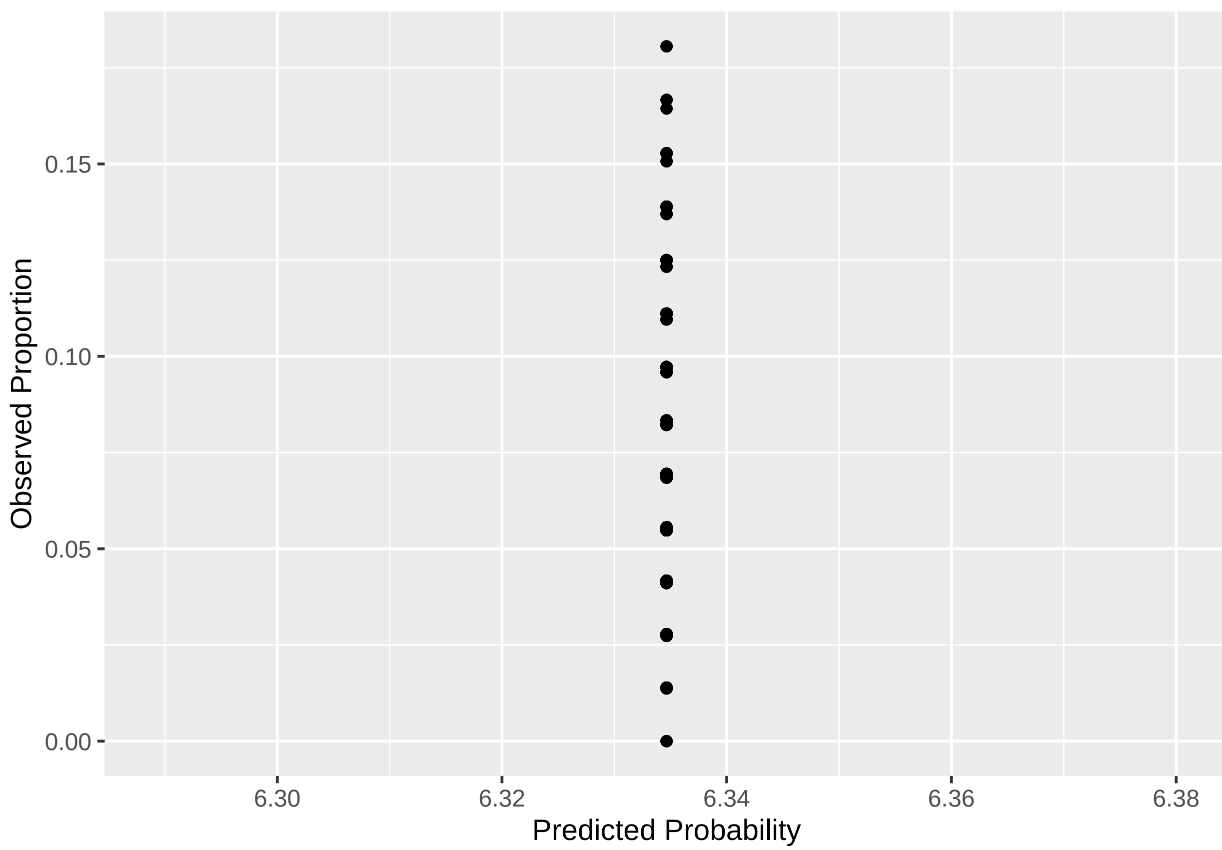
```
## # A tibble: 0 x 4
## # ... with 4 variables: ntile(linpred, 1000) <int>, y <int>, ppred <dbl>,
## #   count <int>
```

```
# Observed Proportion Confidence Interval vs Predicted Probability
hldf <- mutate(hldf, se.fit=sqrt(ppred*(1-ppred)/count))
```

```
## Warning in sqrt(ppred * (1 - ppred)/count): NaNs produced
```

```
ggplot(hldf,aes(x=ppred,y=y/count,ymin=y/count-2*se.fit, ymax=y/count+2*se.fit))+
  geom_point()+geom_linerange(color=grey(0.75))+
  geom_abline(intercept = 0,slope = 1)+
  xlab("Predicted Probability")+
  ylab("Observed Proportion")
```

```
## Warning: Removed 1000 rows containing missing values (geom_segment).
```



```r
# Hosmer-Lemeshow statistics
hlstat <- with(hldf, sum((y-count*ppred) ^2/(count * ppred * (1-ppred))))
c(hlstat, nrow(hldf))
```

```
## [1] -83847.4   1000.0
```

```r
# The p-value is given by:
1-pchisq(hlstat, nrow(hldf)-2)
```

```
## [1] 1
```

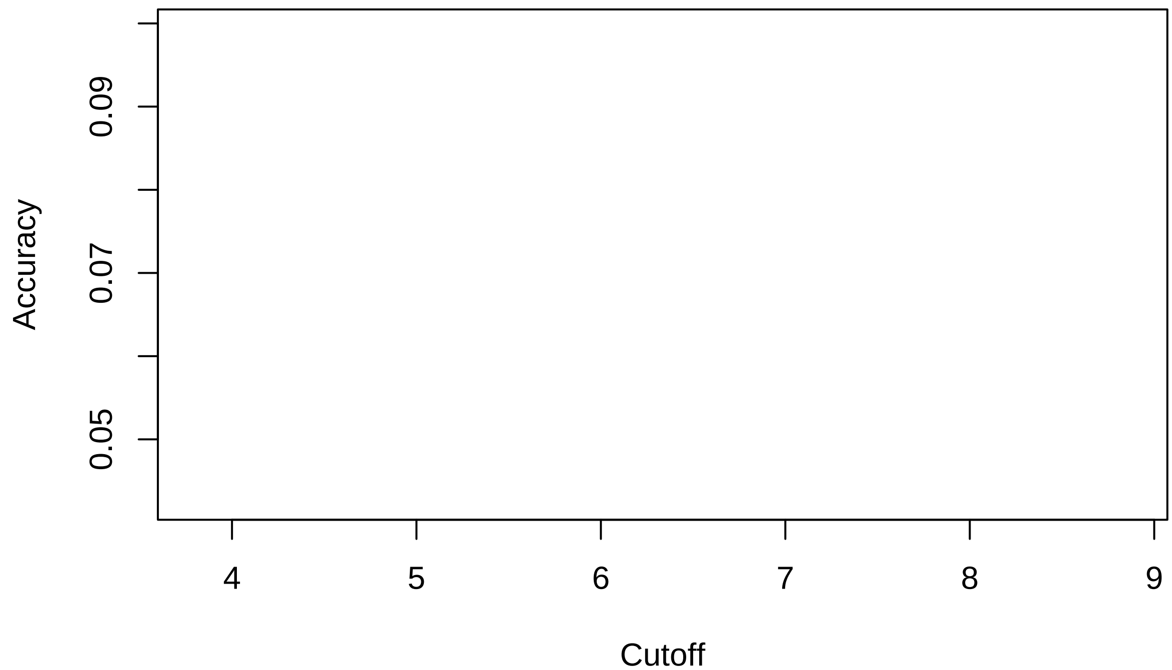
AUC

5-3 Model Performance with Test Data

```r
#Final Model(w/ interaction term ReugularMedicine * PhysicallyActive)
result_m2 = predict(model.final, newx= as.matrix(test.x), type="response")

head(test.y)
```

```
## [1] 0 0 0 0 0 0
```

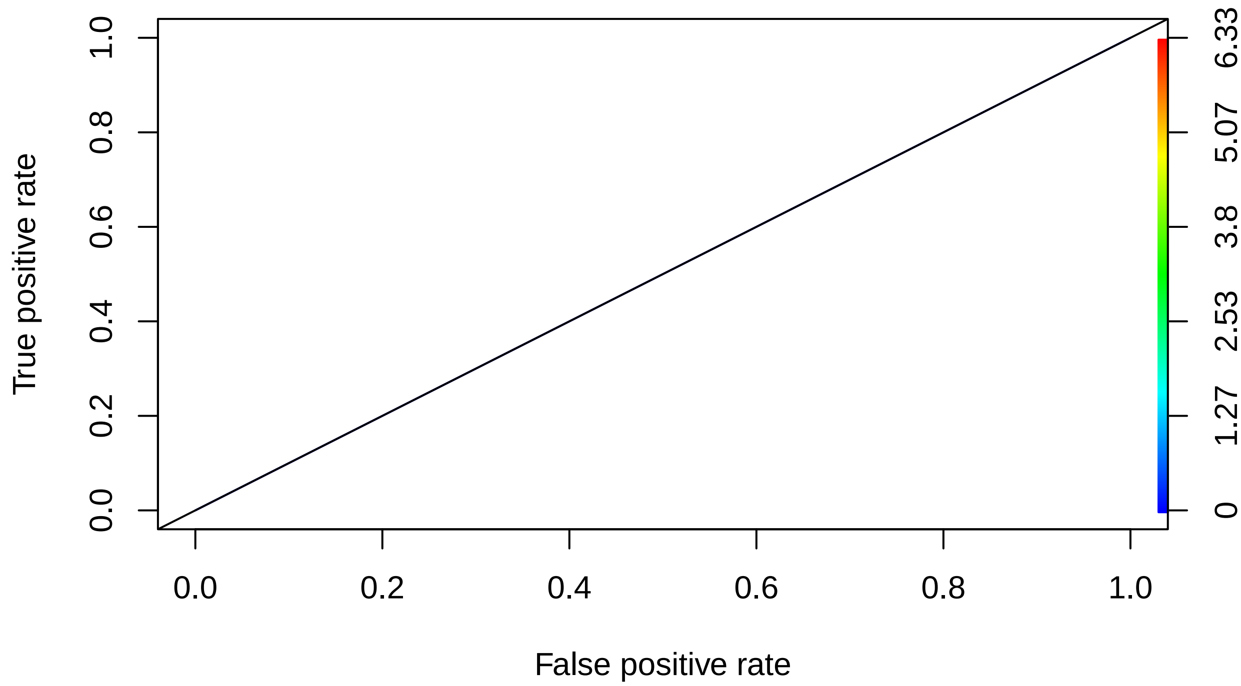
```r
pred_m2 = prediction(result_m2, test.y)


plot(performance(pred_m2, "acc")) #It seems like 0.52 cutoff has the highest accuracy
```




```
table(test.y, result_m2>0.2)
```

```
##
## test.y  TRUE
##      0 44692
##      1  3416
```

```
#Accuracy :
#Sensitivity :
#Specificity :
#The Specificity and accuracy improved a bit compared to the previous model without interaction term, s
plot(performance(pred_m2,"tpr","fpr"), colorize=T)
abline(0,1)
```




```r
#Now we calculate the area under the curve (AUC) and accuracy of the model given above (glmModel2)
auc_ROCR2 <- performance(pred_m2, measure = "auc")
auc_ROCR2@y.values[[1]]
```

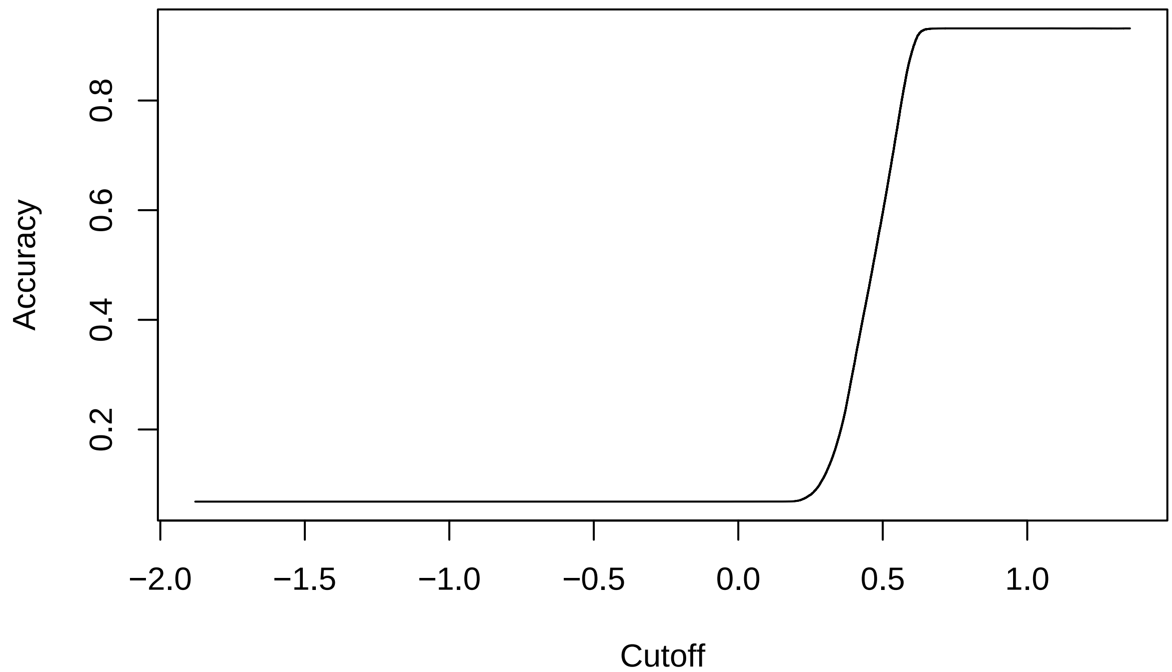
```
## [1] 0.5
```

Ridge Model with Balanced Dataset

```r
## Set model as final model
model.final <- ridge_model_balanced

## Evaluation on Final model using Training Data
train_preds = predict(model.final, newx=as.matrix(train.x), type="response")
head(train_preds[is.na(train_preds)])
```

```
## numeric(0)
```

```r
#train.x[c(7,9),]
#train_preds[is.na(train_preds)]

pred_compare = prediction(train_preds, train.y)
plot(performance(pred_compare, "acc"))
```




```r
table(train.y, train_preds>0.66) # accurary on train
```

```
##
## train.y FALSE  TRUE
##       0 67048   172
##       1  4835   106
```

5-3 Goodness of Fit using Hosmer-Lemeshow Test

a)

The p-value is 0, meaning that we want to reject the null hypothesis that the model is adequate.

```r
# Goodness of Fit using Hosmer-Lemeshow Test
linpred=predict(model.final, newx = as.matrix(train.x), type = "response")

cs_train_m <- mutate(cs_train, predprob=predict(model.final, newx = as.matrix(train.x), type="response"
gdf <- group_by(cs_train_m, ntile(linpred, 1000)) # group up the data by eta_x into 100 groups
hldf <- summarise(gdf, y=sum(SeriousDlqin2yrs==1), ppred=mean(predprob), count=n())
head(hldf)
```

```
## # A tibble: 6 x 4
##   ‘ntile(linpred, 1000)‘     y ppred count
##                      <int> <int> <dbl> <int>
```

```
## 1                      1     3 0.136    73
## 2                      2     1 0.203    73
## 3                      3     1 0.212    73
## 4                      4     1 0.218    73
## 5                      5     0 0.223    73
## 6                      6     3 0.227    73
```

```r
# We adjust the size of the bins until there's only one group with less than 5
hldf[hldf$count<5,]
```

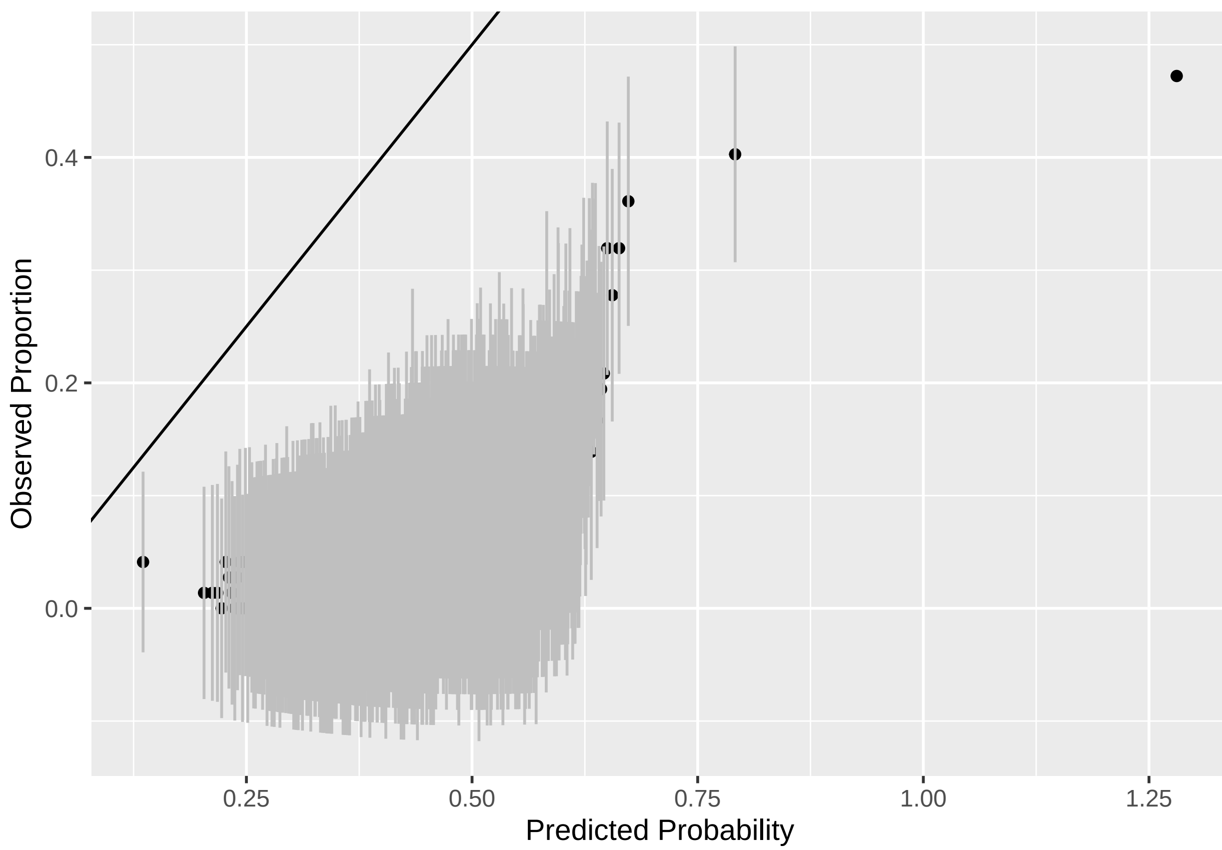
```
## # A tibble: 0 x 4
## # ... with 4 variables: ntile(linpred, 1000) <int>, y <int>, ppred <dbl>,
## #   count <int>
```

```r
# Observed Proportion Confidence Interval vs Predicted Probability
hldf <- mutate(hldf, se.fit=sqrt(ppred*(1-ppred)/count))
```

```
## Warning in sqrt(ppred * (1 - ppred)/count): NaNs produced
```

```r
ggplot(hldf,aes(x=ppred,y=y/count,ymin=y/count-2*se.fit, ymax=y/count+2*se.fit))+
  geom_point()+geom_linerange(color=grey(0.75))+
  geom_abline(intercept = 0,slope = 1)+
  xlab("Predicted Probability")+
  ylab("Observed Proportion")
```

```
## Warning: Removed 1 rows containing missing values (geom_segment).
```



```r
# Hosmer-Lemeshow statistics
hlstat <- with(hldf, sum((y-count*ppred) ^2/(count * ppred * (1-ppred))))
c(hlstat, nrow(hldf))
```

```
## [1] 48963.4  1000.0
```

```r
# The p-value is given by:
1-pchisq(hlstat, nrow(hldf)-2)
```

```
## [1] 0
```

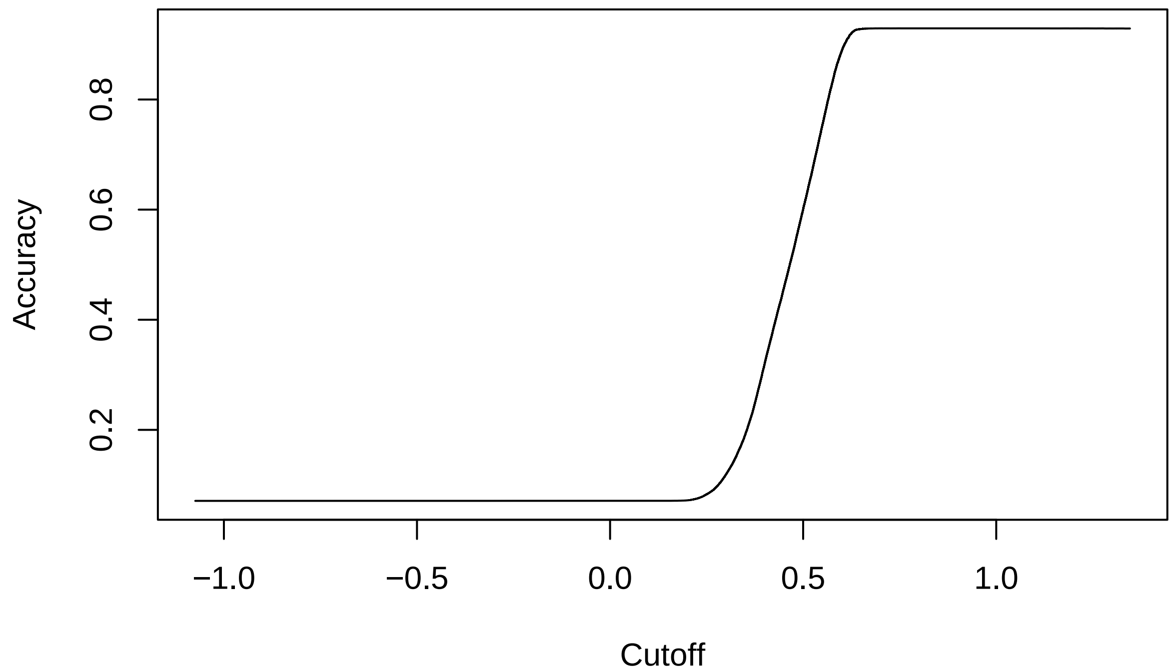
AUC

5-3 Model Performance with Test Data

```r
#Final Model(w/ interaction term ReugularMedicine * PhysicallyActive)
result_m2 = predict(model.final, newx= as.matrix(test.x), type="response")

head(test.y)
```

```
## [1] 0 0 0 0 0 0
```

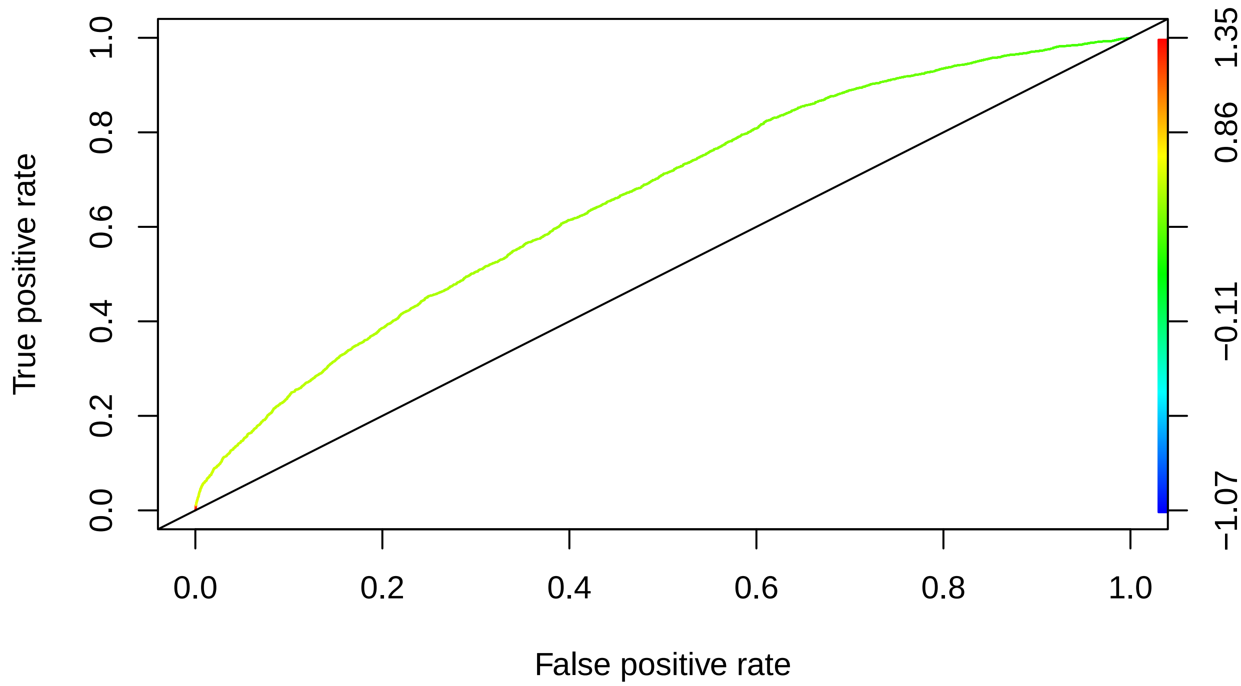
```r
pred_m2 = prediction(result_m2, test.y)


plot(performance(pred_m2, "acc")) #It seems like 0.52 cutoff has the highest accuracy
```




```
table(test.y, result_m2>0.2)
```

```
##
## test.y FALSE  TRUE
##      0    47 44645
##      1     2  3414
```

```
#Accuracy :
#Sensitivity :
#Specificity :
#The Specificity and accuracy improved a bit compared to the previous model without interaction term, s
plot(performance(pred_m2,"tpr","fpr"), colorize=T)
abline(0,1)
```




```
#Now we calculate the area under the curve (AUC) and accuracy of the model given above (glmModel2)
auc_ROCR2 <- performance(pred_m2, measure = "auc")
auc_ROCR2@y.values[[1]]
```

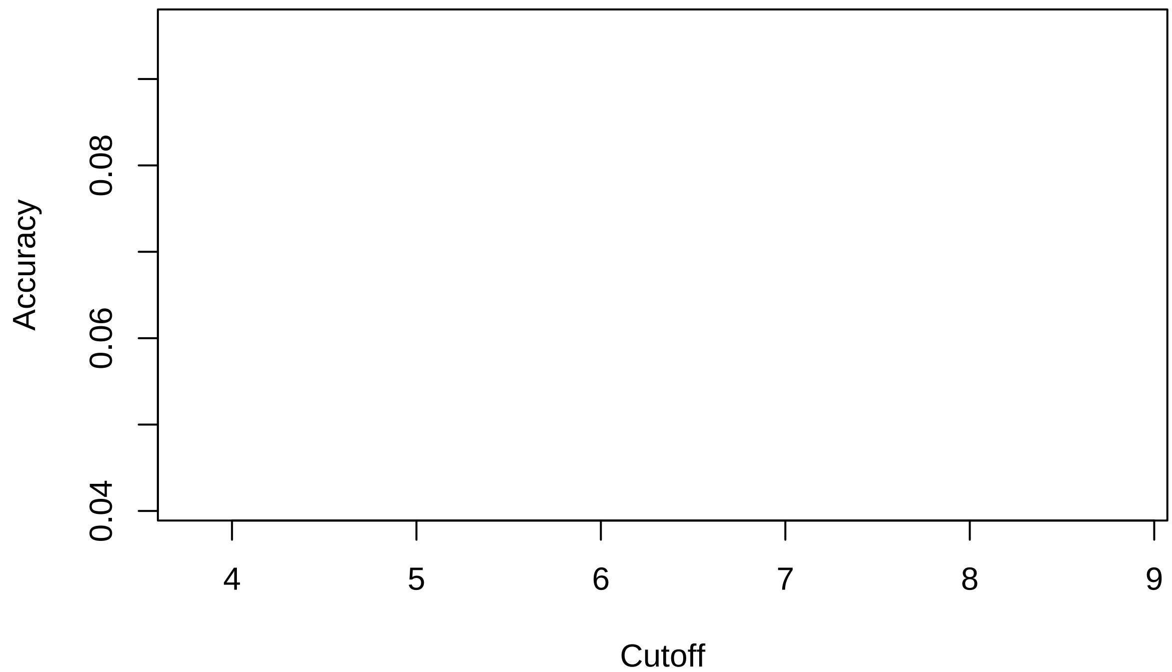
```
## [1] 0.6593516
```

###Lasso Model Unbiased Dataset

```
## Set model as final model
model.final <- lasso_model5

## Evaluation on Final model using Training Data
train_preds = predict(model.final, newx=as.matrix(train.x), type="response")
head(train_preds[is.na(train_preds)])
```

```
## numeric(0)
```

```
#train.x[c(7,9),]
#train_preds[is.na(train_preds)]

pred_compare = prediction(train_preds, train.y)
plot(performance(pred_compare, "acc"))
```




```
table(train.y, train_preds>0.2) # accurary on train
```

```
##
## train.y  TRUE
##       0 67220
##       1  4941
```

5-3 Goodness of Fit using Hosmer-Lemeshow Test

a)

The p-value is 0, meaning that we want to reject the null hypothesis that the model is adequate.

```
# Goodness of Fit using Hosmer-Lemeshow Test
linpred=predict(model.final, newx = as.matrix(train.x), type = "response")

cs_train_m <- mutate(cs_train, predprob=predict(model.final, newx = as.matrix(train.x), type="response")
gdf <- group_by(cs_train_m, ntile(linpred, 1000)) # group up the data by eta_x into 100 groups
hldf <- summarise(gdf, y=sum(SeriousDlqin2yrs==1), ppred=mean(predprob), count=n())
head(hldf)
```

```
## # A tibble: 6 x 4
##   `ntile(linpred, 1000)`     y ppred count
##                    <int> <int> <dbl> <int>
```

```
## 1                      1     6  6.33    73
## 2                      2     6  6.33    73
## 3                      3     4  6.33    73
## 4                      4     3  6.33    73
## 5                      5     8  6.33    73
## 6                      6     3  6.33    73
```

```r
# We adjust the size of the bins until there's only one group with less than 5
hldf[hldf$count<5,]
```

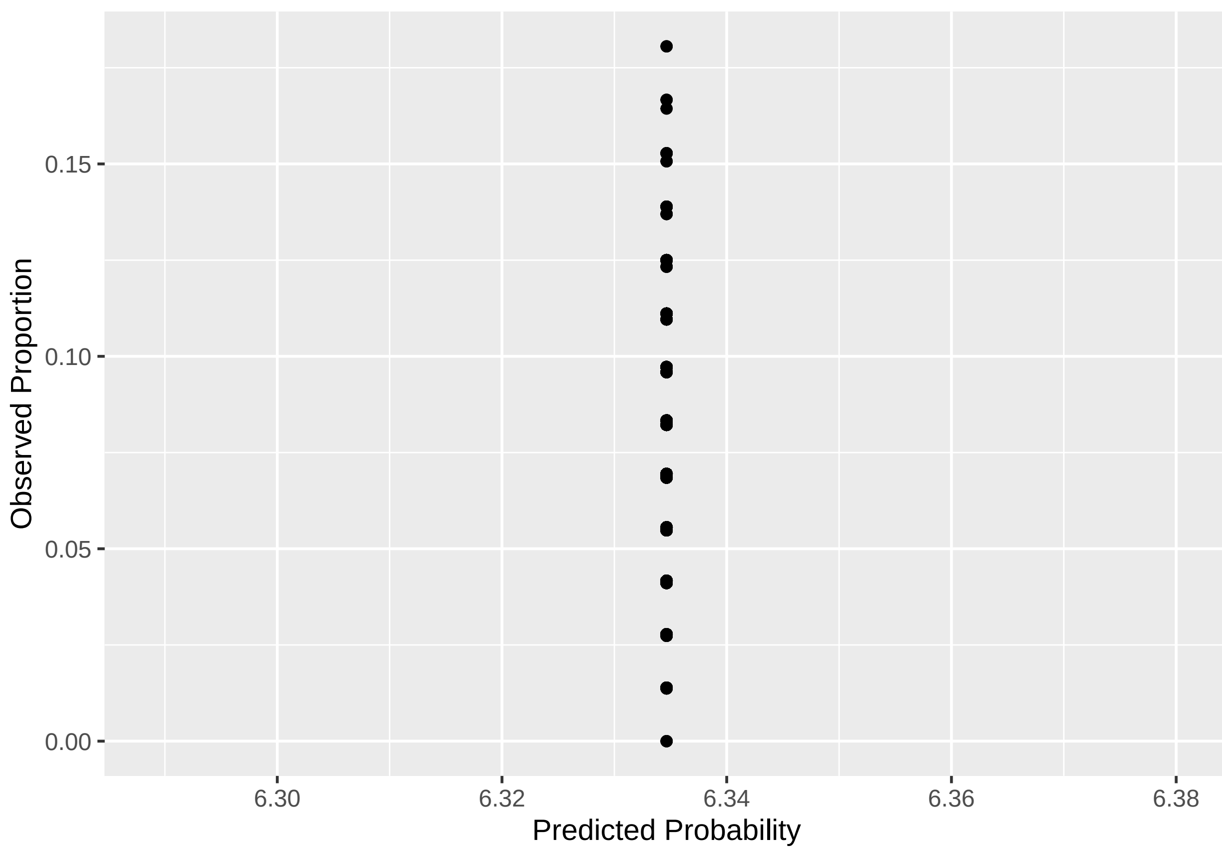
```
## # A tibble: 0 x 4
## # ... with 4 variables: ntile(linpred, 1000) <int>, y <int>, ppred <dbl>,
## #   count <int>
```

```r
# Observed Proportion Confidence Interval vs Predicted Probability
hldf <- mutate(hldf, se.fit=sqrt(ppred*(1-ppred)/count))
```

```
## Warning in sqrt(ppred * (1 - ppred)/count): NaNs produced
```

```r
ggplot(hldf,aes(x=ppred,y=y/count,ymin=y/count-2*se.fit, ymax=y/count+2*se.fit))+
  geom_point()+geom_linerange(color=grey(0.75))+
  geom_abline(intercept = 0,slope = 1)+
  xlab("Predicted Probability")+
  ylab("Observed Proportion")
```

```
## Warning: Removed 1000 rows containing missing values (geom_segment).
```



```r
# Hosmer-Lemeshow statistics
hlstat <- with(hldf, sum((y-count*ppred) ^2/(count * ppred * (1-ppred))))
c(hlstat, nrow(hldf))
```

```
## [1] -83847.4   1000.0
```

```r
# The p-value is given by:
1-pchisq(hlstat, nrow(hldf)-2)
```

```
## [1] 1
```

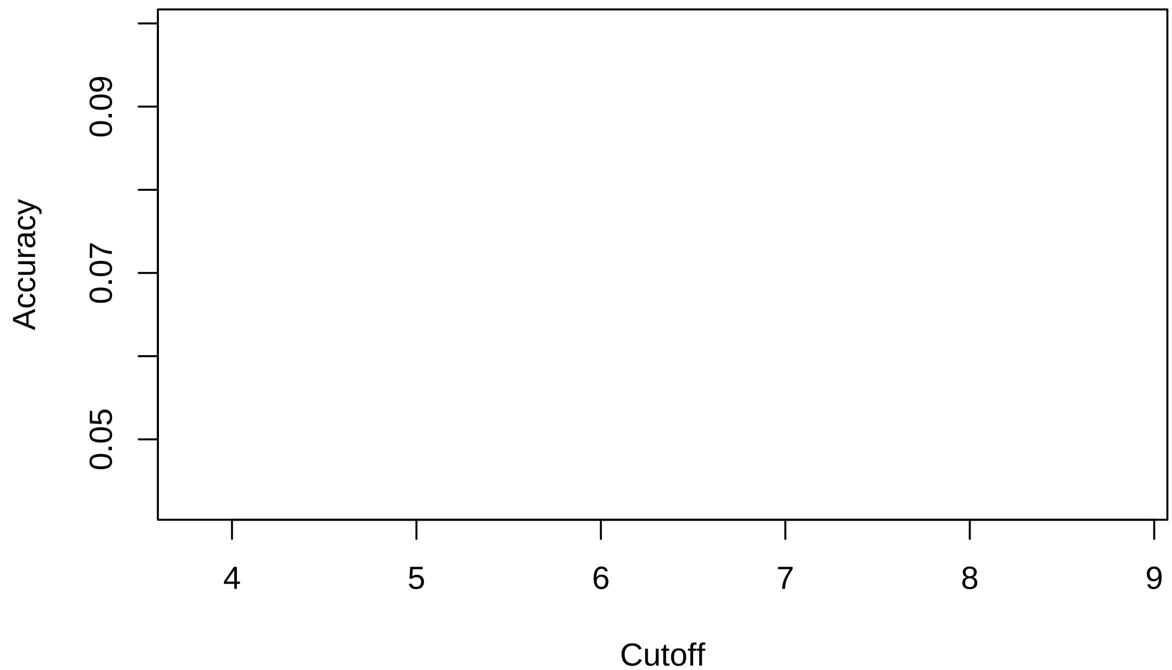
AUC

5-3 Model Performance with Test Data

```r
#Final Model(w/ interaction term ReugularMedicine * PhysicallyActive)
result_m2 = predict(model.final, newx= as.matrix(test.x), type="response")

head(test.y)
```

```
## [1] 0 0 0 0 0 0
```

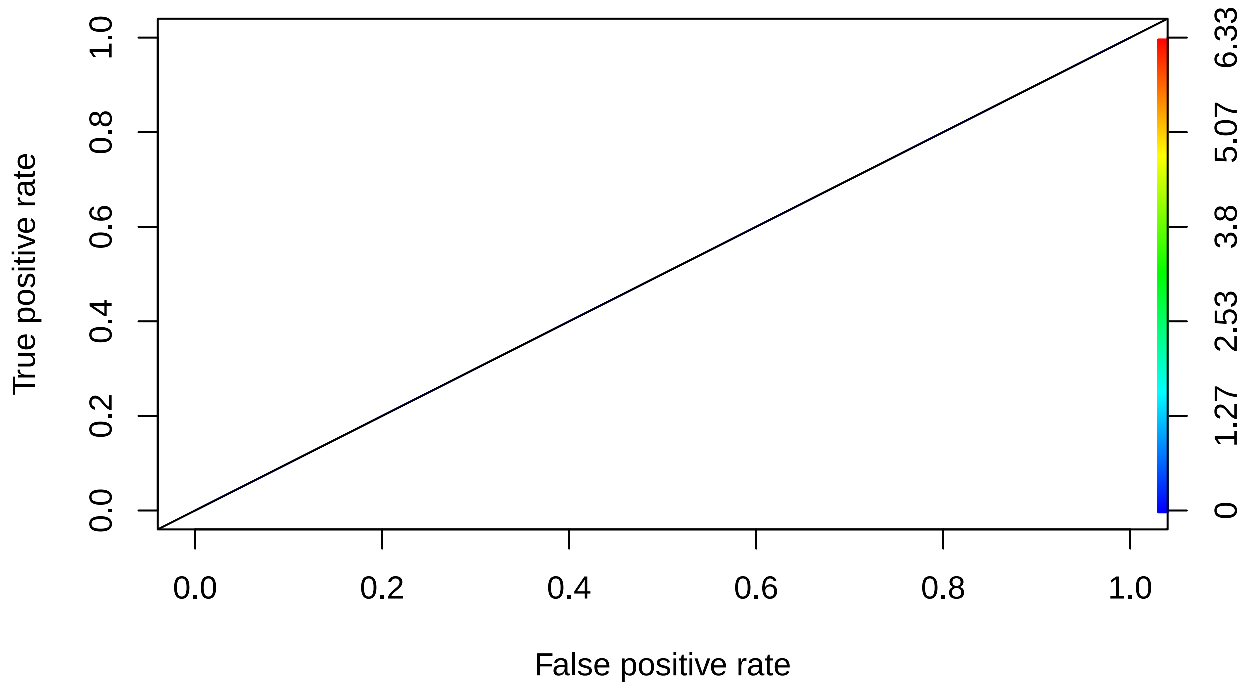
```r
pred_m2 = prediction(result_m2, test.y)


plot(performance(pred_m2, "acc")) #It seems like 0.52 cutoff has the highest accuracy
```




```
table(test.y, result_m2>0.2)
```

```
##
## test.y  TRUE
##      0 44692
##      1  3416
```

```
#Accuracy :
#Sensitivity :
#Specificity :
#The Specificity and accuracy improved a bit compared to the previous model without interaction term, s
plot(performance(pred_m2,"tpr","fpr"), colorize=T)
abline(0,1)
```




```r
#Now we calculate the area under the curve (AUC) and accuracy of the model given above (glmModel2)
auc_ROCR2 <- performance(pred_m2, measure = "auc")
auc_ROCR2@y.values[[1]]
```

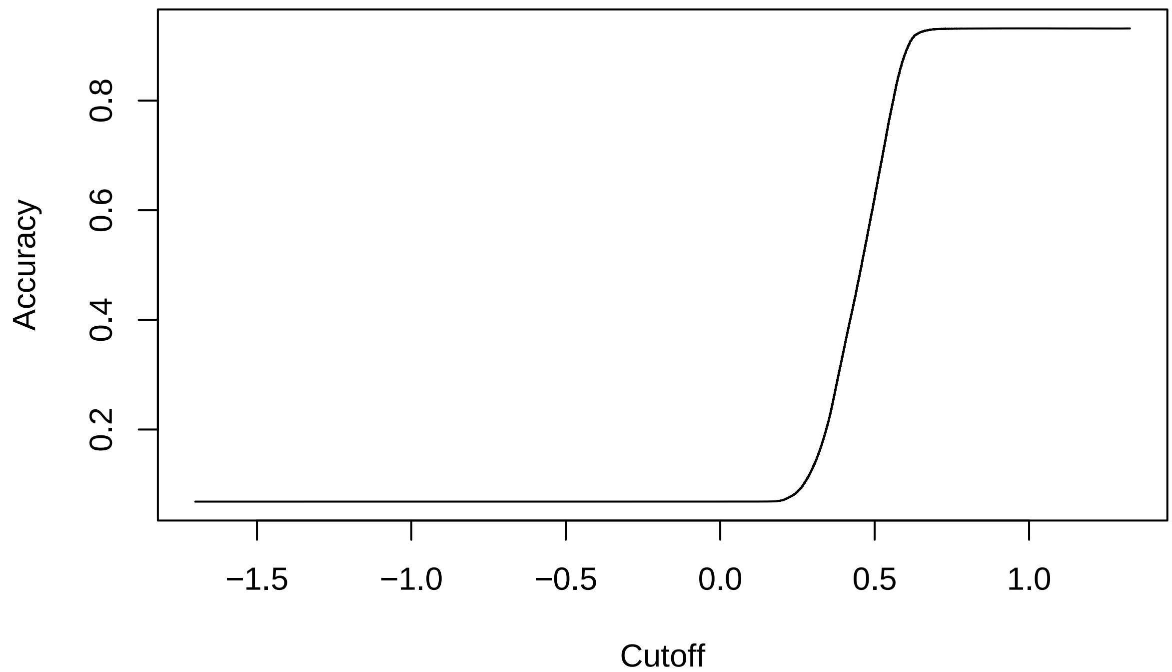
```
## [1] 0.5
```

###Lasso Model Balanced Dataset

```r
## Set model as final model
model.final <- lasso_model_balanced

## Evaluation on Final model using Training Data
train_preds = predict(model.final, newx=as.matrix(train.x), type="response")
head(train_preds[is.na(train_preds)])
```

```
## numeric(0)
```

```r
#train.x[c(7,9),]
#train_preds[is.na(train_preds)]

pred_compare = prediction(train_preds, train.y)
plot(performance(pred_compare, "acc"))
```




```
table(train.y, train_preds>0.6) # accurary on train
```

```
##
## train.y FALSE  TRUE
##       0 62873  4347
##       1  3808  1133
```

5-3 Goodness of Fit using Hosmer-Lemeshow Test

a)

The p-value is 0, meaning that we want to reject the null hypothesis that the model is adequate.

```
# Goodness of Fit using Hosmer-Lemeshow Test
linpred=predict(model.final, newx = as.matrix(train.x), type = "response")

cs_train_m <- mutate(cs_train, predprob=predict(model.final, newx = as.matrix(train.x), type="response")
gdf <- group_by(cs_train_m, ntile(linpred, 1000)) # group up the data by eta_x into 100 groups
hldf <- summarise(gdf, y=sum(SeriousDlqin2yrs==1), ppred=mean(predprob), count=n())
head(hldf)
```

```
## # A tibble: 6 x 4
##   `ntile(linpred, 1000)`     y ppred count
##                      <int> <int> <dbl> <int>
```

```
## 1                      1     6 0.124    73
## 2                      2     5 0.188    73
## 3                      3     3 0.198    73
## 4                      4     0 0.205    73
## 5                      5     1 0.209    73
## 6                      6     4 0.213    73
```

```
# We adjust the size of the bins until there's only one group with less than 5
hldf[hldf$count<5,]
```

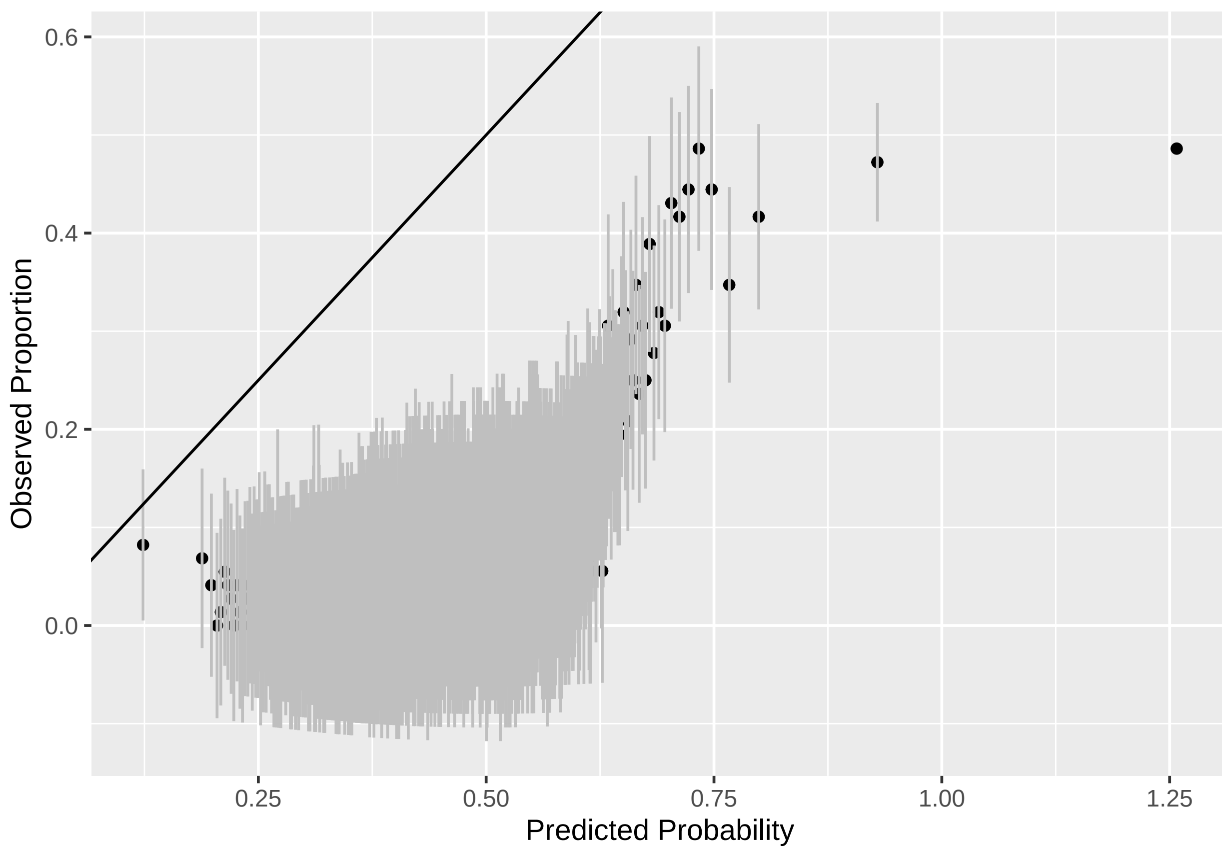
```
## # A tibble: 0 x 4
## # ... with 4 variables: ntile(linpred, 1000) <int>, y <int>, ppred <dbl>,
## #   count <int>
```

```
# Observed Proportion Confidence Interval vs Predicted Probability
hldf <- mutate(hldf, se.fit=sqrt(ppred*(1-ppred)/count))
```

```
## Warning in sqrt(ppred * (1 - ppred)/count): NaNs produced
```

```
ggplot(hldf,aes(x=ppred,y=y/count,ymin=y/count-2*se.fit, ymax=y/count+2*se.fit))+
  geom_point()+geom_linerange(color=grey(0.75))+
  geom_abline(intercept = 0,slope = 1)+
  xlab("Predicted Probability")+
  ylab("Observed Proportion")
```

```
## Warning: Removed 1 rows containing missing values (geom_segment).
```



```r
# Hosmer-Lemeshow statistics
hlstat <- with(hldf, sum((y-count*ppred) ^2/(count * ppred * (1-ppred))))
c(hlstat, nrow(hldf))
```

```
## [1] 48107.41  1000.00
```

```r
# The p-value is given by:
1-pchisq(hlstat, nrow(hldf)-2)
```

```
## [1] 0
```

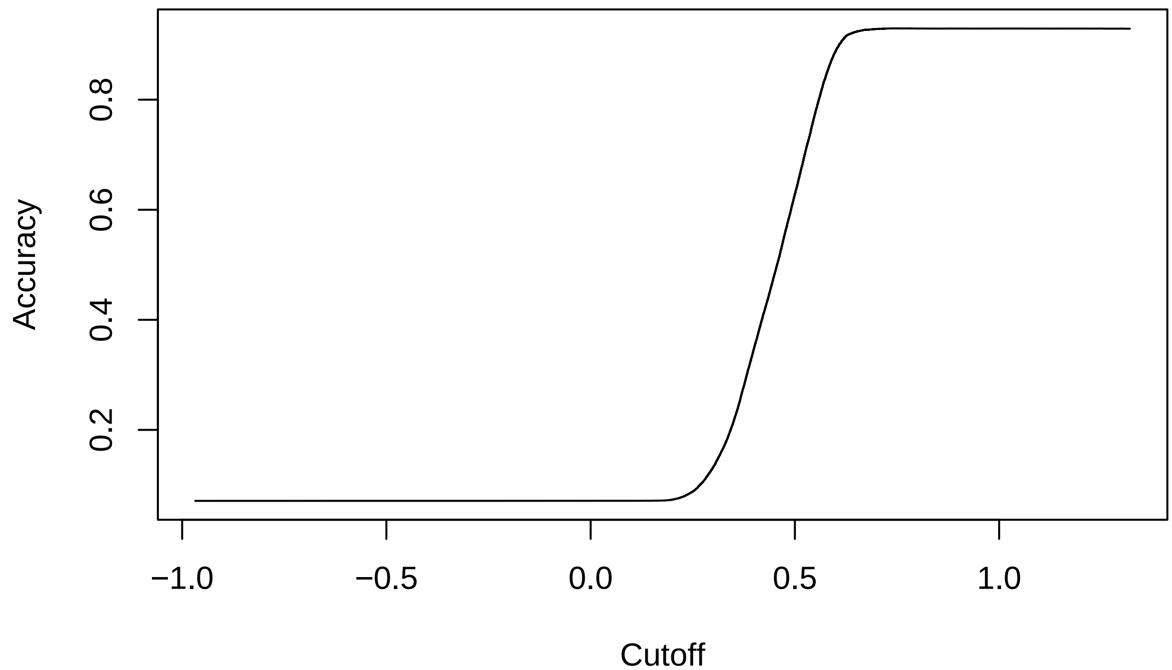
AUC

5-3 Model Performance with Test Data

```r
#Final Model(w/ interaction term ReugularMedicine * PhysicallyActive)
result_m2 = predict(model.final, newx= as.matrix(test.x), type="response")

head(test.y)
```

```
## [1] 0 0 0 0 0 0
```

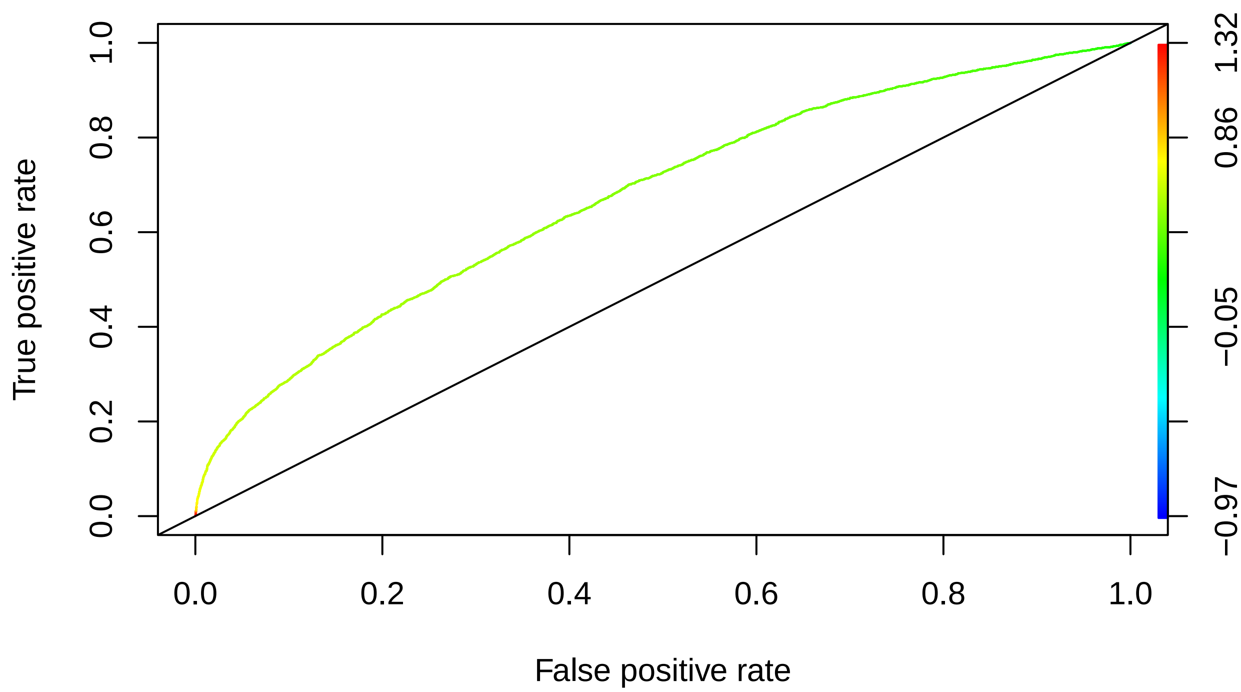
```r
pred_m2 = prediction(result_m2, test.y)


plot(performance(pred_m2, "acc")) #It seems like 0.52 cutoff has the highest accuracy
```




```r
table(test.y, result_m2>0.2)
```

```
##
## test.y FALSE  TRUE
##      0   117 44575
##      1     2  3414
```

```r
#Accuracy :
#Sensitivity :
#Specificity :
#The Specificity and accuracy improved a bit compared to the previous model without interaction term, s
plot(performance(pred_m2,"tpr","fpr"), colorize=T)
abline(0,1)
```




```
#Now we calculate the area under the curve (AUC) and accuracy of the model given above (glmModel2)
auc_ROCR2 <- performance(pred_m2, measure = "auc")
auc_ROCR2@y.values[[1]]
```

```
## [1] 0.6752937
```

```
AIC_models <- data.frame(AIC_unbalanced,AIC_balanced)
AIC_models
```

```
##             AIC_unbalanced AIC_balanced
## PCA               35860.93   13658.5098
## Regular GLM       33396.05   13588.8303
## Ridge           -1505532.95    -175.3686
## Lasso           -2142094.89    -461.6335
```

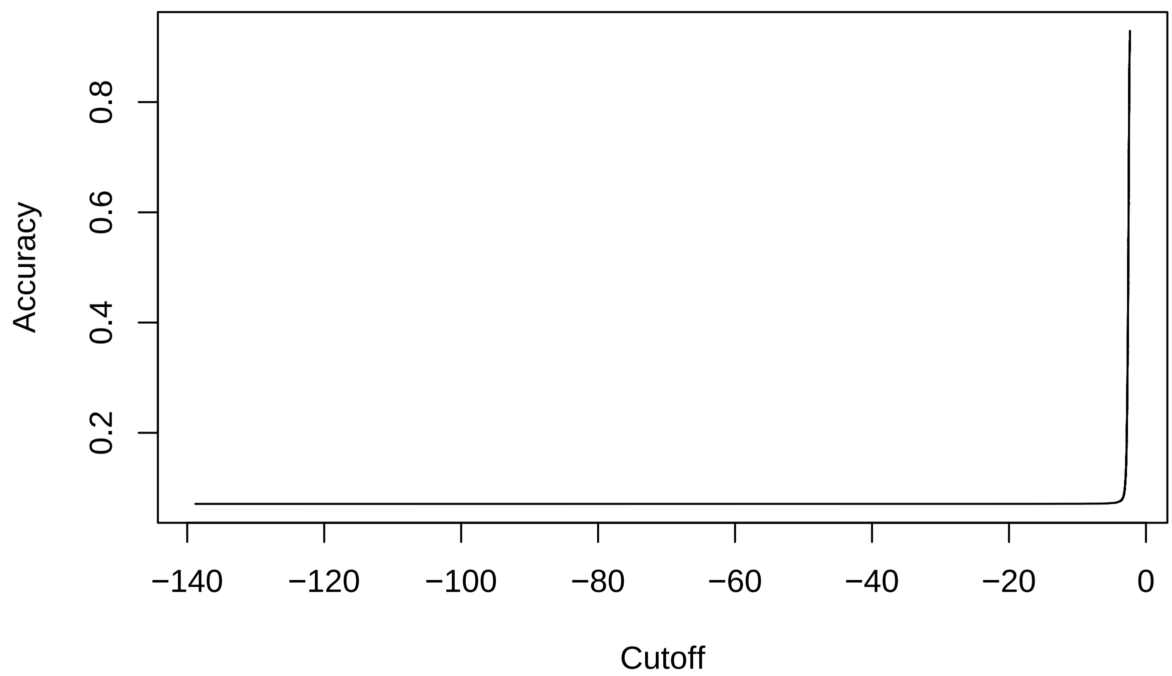
```
#predict(kmeans_model_train, newdata=test.x, type="response")
prediction_result <- predict.glm(glm_pca_original_data, newdata = testing_data)
```

```
## Warning: 'newdata' had 48108 rows but variables found have 72161 rows
```

```
prediction_result <- prediction_result[1:length(test.y)]
head(test.y)
```

```
## [1] 0 0 0 0 0 0
```

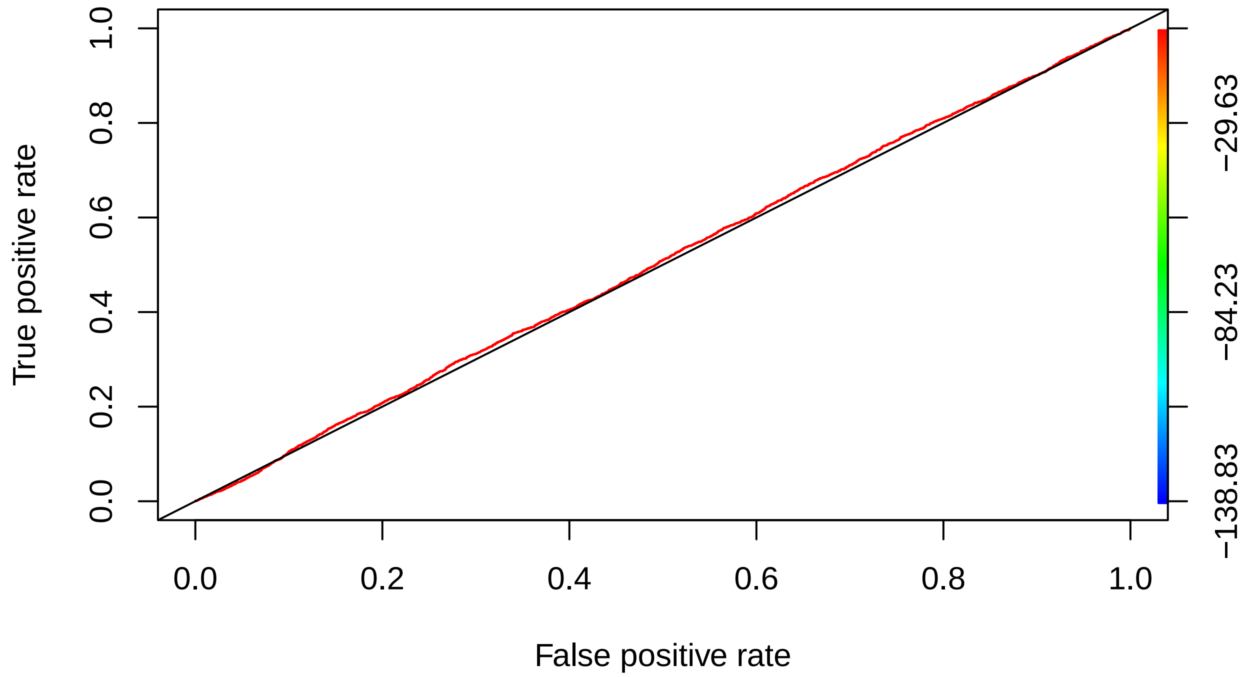
```
prediction_pca = prediction(prediction_result, test.y)
plot(performance(prediction_pca, "acc"))
```




```
table(test.y, result_m2>0.2)
```

```
##
## test.y FALSE  TRUE
##      0   117 44575
##      1     2  3414
```

```
plot(performance(prediction_pca,"tpr","fpr"), colorize=T)
abline(0,1)
```




```
pca_auc_ROCR2 <- performance(prediction_pca, measure = "auc")
pca_auc_ROCR2@y.values[[1]]
```

```
## [1] 0.5073262
```

Maybe don't need session

(0) Ensemble Learning Used by Paper

- Lasso Ensemble Algorithm
- Aggregating base learner: Weighted base=learner

Not Sure if we use this!!

```
# Check how the model fits the data
# From model.final, we can calculate the difference in the two deviances from the summary(model): 987.2
#Number of regressors in the model: 813-802=11
pchisq(473.75,12)
```

```
## [1] 1
```

```r
#The area below 473.75 is one which means the area above it is almost zero. This means that our model h
print(paste("Pearson's X^2 =",round(sum(residuals(model.final,type="pearson")^2),3)))
```

```
## [1] "Pearson's X^2 = 0"
```

```r
qchisq(0.95,802)
```

```
## [1] 868.9936
```

```r
#781.61<868.99, so we fail to reject the null hypothesis and conclude that the logistic model fits the
```

```r
sum(cs_train[,2])
```

```
## [1] 4941
```

```r
sum(cs_test[,2])
```

```
## [1] 3416
```

Cluster Attempt

```r
# Cluster Grouping Majority Data (Try to cluster data by age/monthly income)
set.seed(1) # for reproducibility

head(cs_train_maj)
```

```
##           X SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines age
## 30484 30484                0                           0.09474044  47
## 74216 74216                0                           0.05277307  68
## 54077 54077                0                           0.73665267  36
## 86784 86784                0                           0.02365700  36
## 14388 14388                0                           0.00581900  44
## 31442 31442                0                           0.11523783  40
##       NumberOfTime30.59DaysPastDueNotWorse  DebtRatio MonthlyIncome
## 30484                                    0 0.36842105          6250
## 74216                                    0 0.22759781         11884
## 54077                                    0 0.09445277          2000
## 86784                                    0 0.21710409          5600
## 14388                                    0 0.18525779          5100
## 31442                                    1 2.49168646          2946
##       NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 30484                               8                       0
## 74216                              13                       0
## 54077                               6                       0
## 86784                               9                       0
## 14388                              24                       0
## 31442                              17                       0
##       NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
```

```
## 30484                           1                                     0
## 74216                           2                                     0
## 54077                           0                                     0
## 86784                           1                                     0
## 14388                           1                                     0
## 31442                           3                                     0
##       NumberOfDependents
## 30484                  3
## 74216                  1
## 54077                  1
## 86784                  0
## 14388                  2
## 31442                  0
```

```
# Test with smaller group based on monthly income range
cs_train_maj_g1 <- cs_train_maj[cs_train_maj$NA_Indicator==0,] # Filter out NA monthly income first
cs_train_maj_g1 <- cs_train_maj_g1[cs_train_maj_g1$MonthlyIncome<20000,]  #38035 obs
head(cs_train_maj_g1)
```

```
##  [1] X                                    SeriousDlqin2yrs
##  [3] RevolvingUtilizationOfUnsecuredLines age
##  [5] NumberOfTime30.59DaysPastDueNotWorse DebtRatio
##  [7] MonthlyIncome                        NumberOfOpenCreditLinesAndLoans
##  [9] NumberOfTimes90DaysLate              NumberRealEstateLoansOrLines
## [11] NumberOfTime60.89DaysPastDueNotWorse NumberOfDependents
## <0 rows> (or 0-length row.names)
```

```
# Create a dat with the two predictors of interest
dat <- cs_train_maj_g1[,c(4,7)]  # Age and MonthlyIncome
head(dat)
```

```
## [1] age           MonthlyIncome
## <0 rows> (or 0-length row.names)
```

```
n_maj <- nrow(dat) # get number of rows

# Initial assignments to three groups that will need to update
assignments <- factor(sample(c(1,2,3), n_maj, replace = TRUE))
#plot(dat, col=assignments, xlim = c(0,110), asp=1)
#plot(dat, col=assignments)
```

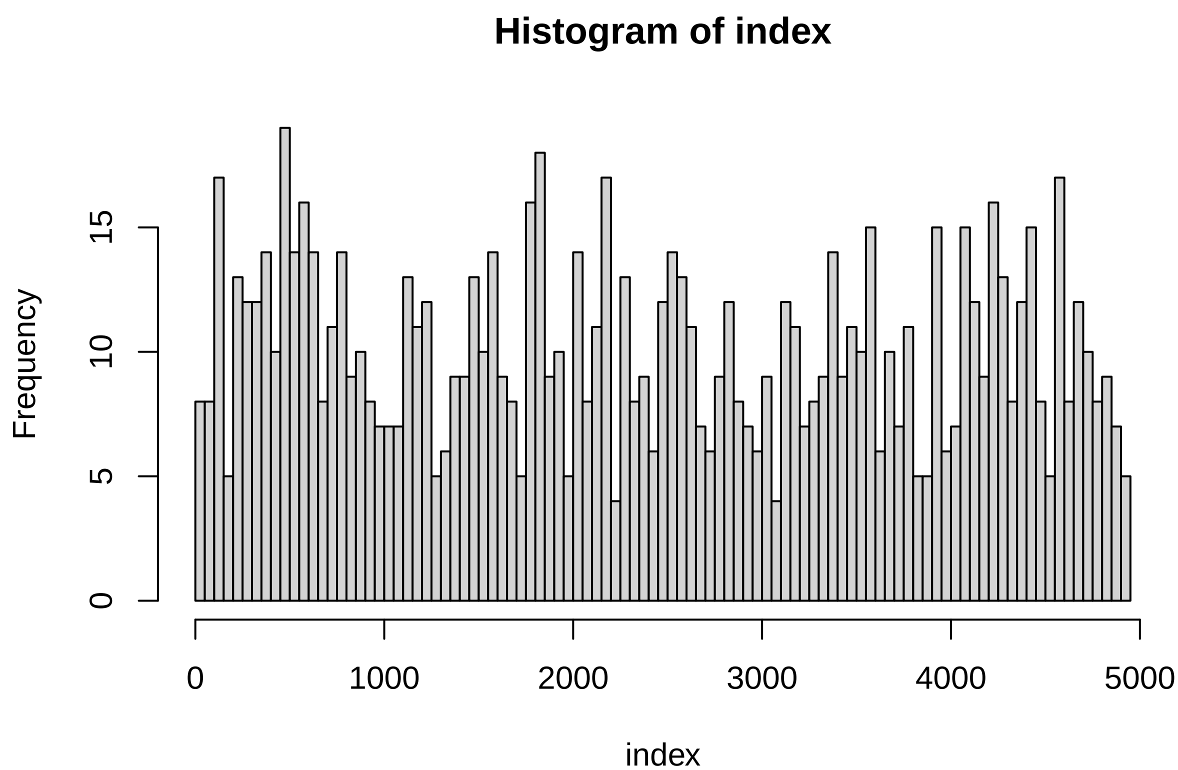
a) REMOVE SECTION - NOT APPLICABLE —Boostrapping Minority Data (cs_train_min_add) (1000 obs)

- Currently created 1000 additional data, can add more

```
# Boostrapping Minority Data
set.seed(1) # for reproducibility
# set number of minority data to reproduce
n_add <- 1000

n_min <- nrow(cs_train_min)
n_min
```

```
## [1] 4941
```

```r
n_maj <- nrow(cs_train_maj)
index <- sample(n_min, n_add, replace = TRUE)
index_maj <- sample(n_maj, n_add, replace = TRUE)
#plot(density(index),  main="")  # show density curve of the index we randomized
hist(index, breaks = 100)
```




```r
min(index)
```

```
## [1] 15
```

```r
max(index)
```

```
## [1] 4939
```

```r
length(index)
```

```
## [1] 1000
```

```r
# We add the additional data for future analysis
cs_train_min_add <- cs_train_min[index,]
head(cs_train_min_add)
```

```
##              X SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines age
## 88154   88154                1                           0.03545848  33
## 64256   64256                1                           0.98335183  37
## 105760 105760                1                           1.46506986  37
## 137307 137307                1                           0.11727646  34
## 20062   20062                1                           0.67475736  53
## 47668   47668                1                           0.99989018  41
##        NumberOfTime30.59DaysPastDueNotWorse  DebtRatio MonthlyIncome
## 88154                                     3 0.29635182          2000
## 64256                                     2 0.32223968          7000
## 105760                                    0 0.21990439          2300
## 137307                                    0 0.08461026          1500
## 20062                                     0 2.16232772          3264
## 47668                                     0 0.07576094          4500
##        NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
## 88154                               10                       0
## 64256                                5                       0
## 105760                               5                       3
## 137307                               9                       0
## 20062                               21                       0
## 47668                                4                       0
##        NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 88154                             0                                    1
## 64256                             3                                    0
## 105760                            0                                    2
## 137307                            0                                    0
## 20062                             4                                    0
## 47668                             0                                    0
##        NumberOfDependents
## 88154                   0
## 64256                   1
## 105760                  2
## 137307                  4
## 20062                   0
## 47668                   0
```

```
cs_train_maj_add <- cs_train_maj[index_maj,]
head(cs_train_maj_add)
```

```
##              X SeriousDlqin2yrs RevolvingUtilizationOfUnsecuredLines age
## 39095   39095                0                            0.4191770  40
## 53178   53178                0                            0.1770347  54
## 134752 134752                0                            1.1369727  60
## 7328     7328                0                            0.1300842  48
## 101807 101807                0                            0.0317618  28
## 110064 110064                0                            0.0000000  83
##        NumberOfTime30.59DaysPastDueNotWorse   DebtRatio MonthlyIncome
## 39095                                     0   0.8230508          1474
## 53178                                     0   0.4121380          8666
## 134752                                    0   1.0512030          6483
## 7328                                      0   0.3726312         10500
## 101807                                    1 262.5000000             1
## 110064                                    0   0.0000000          5200
##        NumberOfOpenCreditLinesAndLoans NumberOfTimes90DaysLate
```

```
## 39095                              6                   1
## 53178                             11                   0
## 134752                            16                   0
## 7328                              10                   0
## 101807                            12                   0
## 110064                             1                   0
##        NumberRealEstateLoansOrLines NumberOfTime60.89DaysPastDueNotWorse
## 39095                             1                                    0
## 53178                             1                                    0
## 134752                            2                                    0
## 7328                              1                                    0
## 101807                            0                                    0
## 110064                            0                                    0
##        NumberOfDependents
## 39095                   2
## 53178                   2
## 134752                  0
## 7328                    1
## 101807                  2
## 110064                  0
```

```
nrow(cs_train_maj)
```

```
## [1] 67220
```

```
nrow(cs_train_maj_add)
```

```
## [1] 1000
```

5-2 Marginal Model Plots

Need to Update!!

```
# residual plots
#residualPlots(model.final)

# Marginial Model Plots
#library(car)
#mmp(model.final)
```

Maybe don't need session (END)