Ni Ma Nov. 17

All the code and code outputs are written in the r script and more explanations and all plots are shown in this doc file.

I first read data and split this 70% sample data into training (50%) and validation sets (20%). There is no missing data in the training set.

1. Exploratory data analysis on the training set
   Summary statistics for all the predictors are shown in the r file for both categorical and continuous x's.

   From the summary statistics for X3 education, I found that it can take values of 0, 4, 5 and 6 in addition to 1 to 4. To understand these values, we summarize the relationship between Y and X3 as below. I assume that 0, 4, 5 and 6 denote missing or unknown values and I group them as one category. The proportion of defaults is obviously much smaller for X3 being missing or unknown, so I keep them as one category. Later in the data preprocessing part, I regard X3 as categorical variable and change its value to 0 when it equals 4, 5 and 6. I don't regard it as a continuous value since there is no order relationship between 0 and 1, 2 and 3.

```
summary(factor(data_train$X3))
# 0    1    2    3    4    5    6
# 6 5257 7029 2475   60  144   28

xtabs(~ X3 + Y, data_train)
#         Y
#X3     0    1
# 0     6    0
# 1  4244 1013
# 2  5385 1644
# 3  1849  626
# 4    57    3
# 5   134   10
# 6    23    5
```

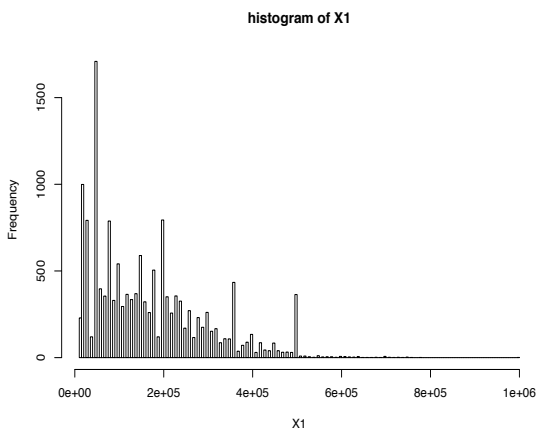
   Also for X4 Marital status, it can take the value of 0 in addition to 1 to 3. The table for values of Y and X4 is as below. I assume that 0 denotes missing or unknown values. The proportion of defaults is obviously much smaller for X4 being missing or unknown, so I keep them as one category. X4 is regarded as a categorical variable.

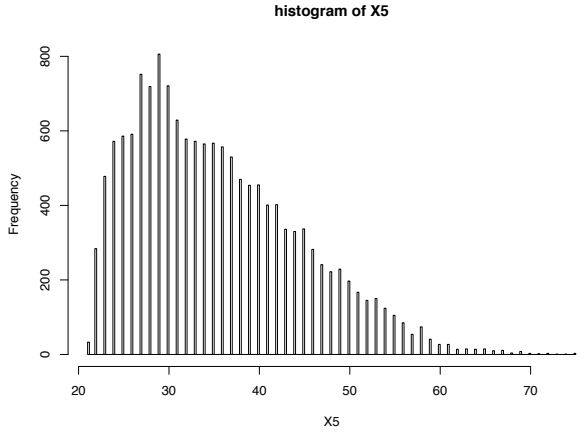
```
summary(factor(data_train$X4))
#  0    1    2    3
# 30 6784 8013  172

xtabs(~ X4 + Y, data_train)
#          Y
#X4     0    1
# 0    28    2
# 1  5226 1558
# 2  6318 1695
# 3   126   46
```

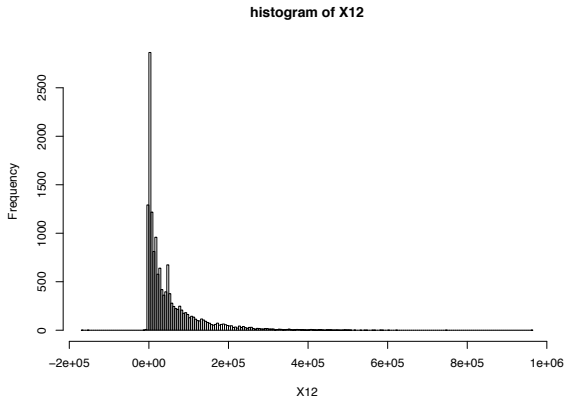
Below are summary statistics for X6 to X11, history of repayment status. They can take values of -2 and 0 in addition to -1, 1 to 8. As will be shown in the correlation matrix later, they are all positively correlated with Y. By inspecting the data, I found that -2 means that the bill statement is 0 for that month and 0 means that a proportion of the bill statement is repaid for that month (possibly larger than a minimum pay proportion) but not in full.

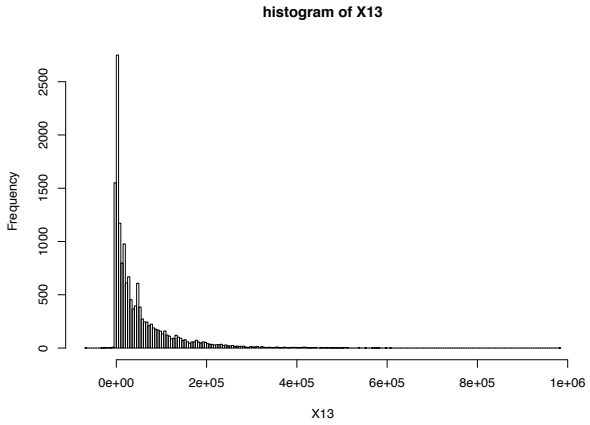
```
#-2   -1    0    1    2    3    4    5    6    7    8
#1364 2777 7434 1856 1329  171   39   12    4    5    8

#-2   -1    0    1    2    3    4    5    6    7
#1864 2993 7916   16 1965  169   48   12    8    8

#-2   -1    0    1    2    3    4    5    6    7    8
#2024 2918 7914    3 1943  115   43   13    9   15    2

#-2   -1    0    1    2    3    4    5    6    7    8
#2142 2808 8283    1 1583   90   38   19    3   31    1

#-2   -1    0    2    3    4    5    6    7
#2229 2778 8462 1353   93   41    9    3   31

#-2   -1    0    2    3    4    5    6    7    8
#2401 2931 8099 1409   91   24    9   10   24    1
```

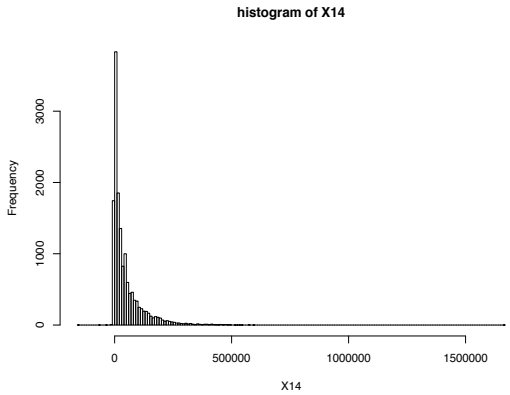
All the histograms for continuous predictors are presented below.

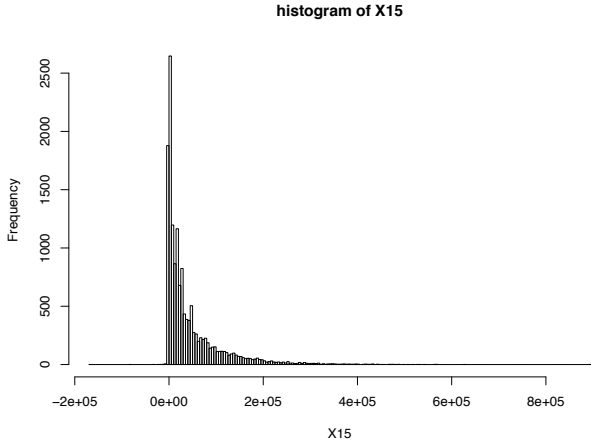


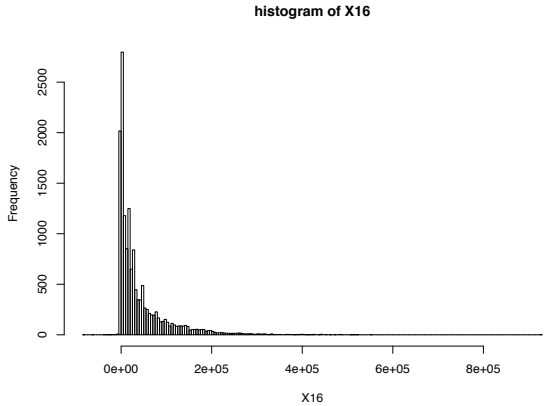


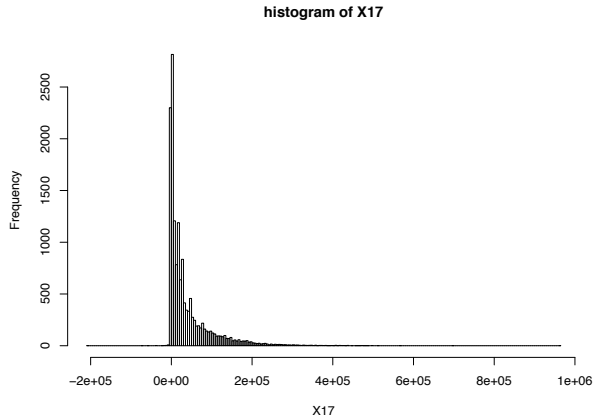

histogram of X12
Frequency
0
500
1000
1500
2000
2500
−2e+05
0e+00
2e+05
4e+05
6e+05
8e+05
1e+06
X12

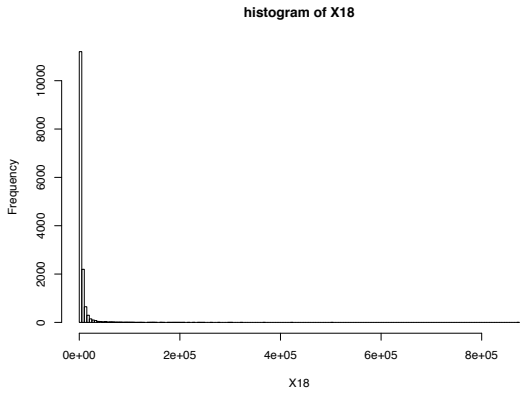

histogram of X13
Frequency
0
500
1000
1500
2000
2500
0e+00
2e+05
4e+05
6e+05
8e+05
1e+06
X13

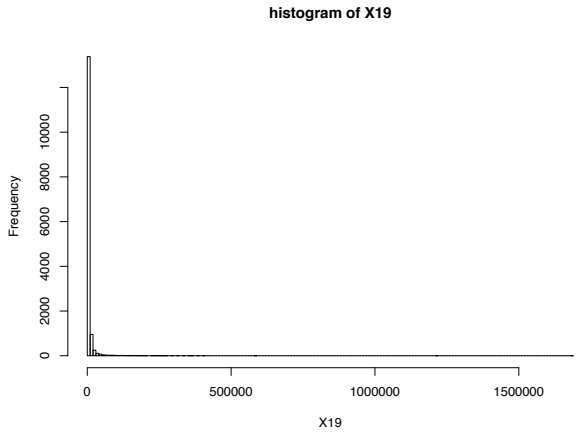

histogram of X14
Frequency
0
1000
2000
3000
0
500000
1000000
1500000
X14

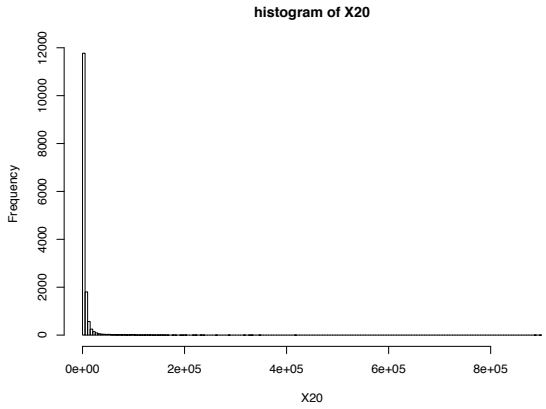

histogram of X15
Frequency
0
500
1000
1500
2000
2500
−2e+05
0e+00
2e+05
4e+05
6e+05
8e+05
X15

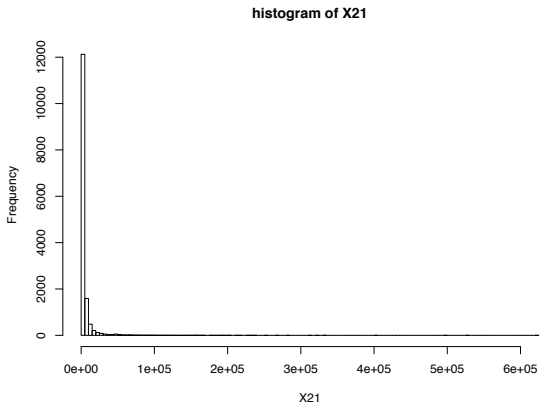

histogram of X16
Frequency
0
500
1000
1500
2000
2500
0e+00
2e+05
4e+05
6e+05
8e+05
X16

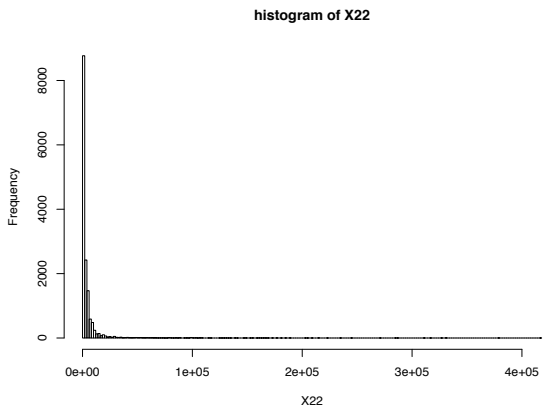

histogram of X17
Frequency
0
500
1000
1500
2000
2500
−2e+05
0e+00
2e+05
4e+05
6e+05
8e+05
1e+06
X17

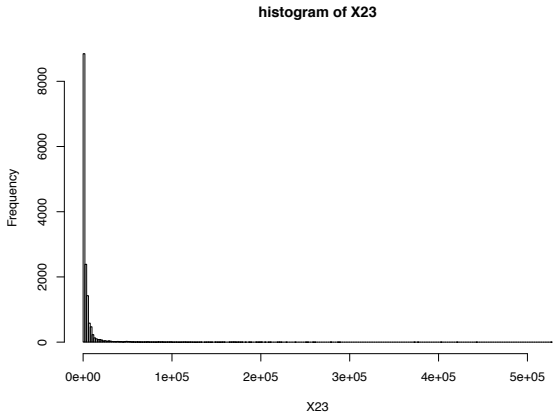












From the big correlation matrix in the r file, I see that correlations among the block X6 to X11, X12 to X17 and X18 to X23 are relatively high. The correlation matrices for these three sets are written below. The variables in the first block have correlations with Y have are perfectly ranked by time and are relatively large. The variables in the second block have

very small correlations with Y. The different signs of their correlations with Y indicate that they may not be significant in predicting Y. The variables in the last block are obviously very noisy as can be seen from correlations among them and their correlations with Y. For this noisy time series data, the moving average may be a better predictor which I will calculate later.

```
cor(data_train[,c('Y','X6','X7','X8','X9','X10','X11')])
#             Y        X6        X7        X8        X9       X10       X11
#Y   1.0000000 0.3280910 0.2755344 0.2443075 0.2307628 0.2136000 0.2023709
#X6  0.3280910 1.0000000 0.6706811 0.5748977 0.5434272 0.5180565 0.4838274
#X7  0.2755344 0.6706811 1.0000000 0.7669204 0.6607969 0.6236317 0.5783183
#X8  0.2443075 0.5748977 0.7669204 1.0000000 0.7769829 0.6868064 0.6328247
#X9  0.2307628 0.5434272 0.6607969 0.7769829 1.0000000 0.8220590 0.7196916
#X10 0.2136000 0.5180565 0.6236317 0.6868064 0.8220590 1.0000000 0.8173997
#X11 0.2023709 0.4838274 0.5783183 0.6328247 0.7196916 0.8173997 1.0000000

cor(data_train[,c('Y','X12','X13','X14','X15','X16','X17')])
#              Y         X12          X13          X14        X15           X16         X17
#Y    1.0000000000 -0.01701772 -0.009363582 -0.005332058 -0.0024254 -0.0007582323
0.002728852
#X12 -0.0170177249  1.00000000  0.949839447  0.885533313  0.8561558  0.8275950987
0.799465706
#X13 -0.0093635823  0.94983945  1.000000000  0.923605988  0.8887192  0.8587992853
0.828929845
#X14 -0.0053320583  0.88553331  0.923605988  1.000000000  0.9190318  0.8784396639
0.850456752
#X15 -0.0024254004  0.85615577  0.888719237  0.919031832  1.0000000  0.9377399501
0.898903888
#X16 -0.0007582323  0.82759510  0.858799285  0.878439664  0.9377400  1.0000000000
0.947533597
#X17  0.0027288521  0.79946571  0.828929845  0.850456752  0.8989039  0.9475335966
1.000000000

cor(data_train[,c('Y','X18','X19','X20','X21','X22','X23')])
#              Y         X18          X19          X20          X21          X22          X23
#Y    1.00000000 -0.07067932 -0.05515099 -0.05700203 -0.06332489 -0.05445561 -0.05811667
#X18 -0.07067932  1.00000000  0.30567455  0.29985196  0.23910641  0.16791872  0.20476780
#X19 -0.05515099  0.30567455  1.00000000  0.29767786  0.22110988  0.22568469  0.15004699
#X20 -0.05700203  0.29985196  0.29767786  1.00000000  0.26759833  0.16355969  0.17797484
#X21 -0.06332489  0.23910641  0.22110988  0.26759833  1.00000000  0.16535623  0.15600330
#X22 -0.05445561  0.16791872  0.22568469  0.16355969  0.16535623  1.00000000  0.15861595
#X23 -0.05811667  0.20476780  0.15004699  0.17797484  0.15600330  0.15861595  1.00000000
```

2. Data manipulation and preprocessing

   I conduct the following data manipulation and preprocessing.

   I change the X3 value to 0 when it equals 4, 5 or 6. I regard X2, X3 and X4 as categorical variables and use dummy variables to denote them.

   I create the following variables aiming to find features with a higher predictive power. avgDelay is the mean of X6 to X11 to indicate average repayment status, avgState is the mean of X12 to X17 to indicate average statement, avgPay is the mean of X18 to X23 to indicate average payment, propState is the ratio of avgState and X1 (the granted credit), propPay is the ratio of avgPay and X1. I calculate the expenditure incurred for each month exp1 to exp5 using: bill statement for that month + payment for the previous month – bill

statement for the previous month. avgExp is the average of expenditures and propExp is the ratio of avgExp and X1. Furthermore, I create a score to indicate the debt severity of each person, which score increases as the amount of debt for each month increases and as the time length of the debt increases. debtScore is calculated as the sum product of repayment status X6 to X10 and expenditures exp1 to exp5. propDebt is the ratio of debtScore and X1.

The correlation matrix of newly created variables is as below. I do some manual selection by reading these correlations here. Variable avgDelay has a relatively large correlation with Y, but the recent prepayment status seems better in predicting Y, thus I select the most recent X6 and X7 to the logistic regression model instead of using the average avgDelay. X18 to X23 are relatively noisy with quite small correlations with Y, but their average avgPay has a large magnitude of correlation with Y than any of the original variables, so I select avgPay to the model. I don't use propPay since it's not as good as avgPay in predicting Y. Similarly I select propState which has a much larger correlation with Y than any of the original variables X12 to X17. The very small correlation of avgState with Y seems not significant because the sign changes. Similarly I select propDebt which has a large correlation with Y. avgExp seems also quite correlated with Y, but it has a very

```
#                   Y    avgDelay      avgPay     propPay    avgState propState
#Y          1.00000000  0.29394815 -0.10056130 -0.03696564 -0.00614026 0.1305836
#avgDelay   0.29394815  1.00000000 -0.07058968  0.04748093  0.28023697 0.5637011
#avgPay    -0.10056130 -0.07058968  1.00000000  0.59726067  0.33458939 0.0301163
#propPay   -0.03696564  0.04748093  0.59726067  1.00000000  0.10600595 0.2715948
#avgState  -0.00614026  0.28023697  0.33458939  0.10600595  1.00000000 0.5514328
#propState  0.13058365  0.56370106  0.03011630  0.27159480  0.55143283 1.0000000
#avgExp    -0.10093139 -0.03488089  0.77697446  0.42744390  0.46717566 0.1307249
#propExp   -0.05386684  0.05427435  0.34987270  0.63713829  0.17240550 0.3689423
#debtScore  0.12212114  0.31559615 -0.55815052 -0.31678784  0.09833689 0.1893561
#propDebt   0.16804023  0.40872407 -0.32901351 -0.31541926  0.08554334 0.2494177

#               avgExp     propExp   debtScore   propDebt
#Y          -0.10093139 -0.05386684  0.12212114 0.16804023
#avgDelay   -0.03488089  0.05427435  0.31559615 0.40872407
#avgPay      0.77697446  0.34987270 -0.55815052 -0.32901351
#propPay     0.42744390  0.63713829 -0.31678784 -0.31541926
#avgState    0.46717566  0.17240550  0.09833689 0.08554334
#propState   0.13072486  0.36894233  0.18935610 0.24941775
#avgExp      1.00000000  0.57769902 -0.44178501 -0.24691192
#propExp     0.57769902  1.00000000 -0.17639012 -0.13378348
#debtScore  -0.44178501 -0.17639012  1.00000000 0.70215382
#propDebt   -0.24691192 -0.13378348  0.70215382 1.00000000
```

I standardize all continuous variables by deducting the mean and dividing by the standard deviation. I save the means and standard deviations of each predictors in the training set for processing the validation set or testing set when testing. They are standardized because I want to add regularization to the logistic regression later.

3. Logistic regression

   We first train a logistic regression model without regularization on the training set. Model logreg1 is the logistic regression model with our manually selected predictors. I do forward

variable selection using the step function to achieve at the model with the smallest AIC score. The optimal model includes variables X6, avgPay, X7, X3, X1, X4, X2, propState and propDebt. The estimated coefficients for all variables (either continuous or categorical) are all significant and make sense except for the variable propState. propState has a positive correlation with Y while its estimated bet is negative. This happens due to its high correlations with other predictors and I decide to drop this variable propDebt.

```
#Call:
#glm(formula = Y ~ X6 + avgPay + X7 + X3 + X1 + X4 + X2 + propState +
#propDebt, family = binomial, data = data_train)
#
#Deviance Residuals:
#Min       1Q   Median       3Q      Max
#-3.3338  -0.6973  -0.5461  -0.2694   2.9813
#
#Coefficients:
#Estimate Std. Error z value Pr(>|z|)
#(Intercept) -3.88419    0.80120  -4.848 1.25e-06 ***
#X6           0.64237    0.02829  22.703  < 2e-16 ***
#avgPay      -0.38229    0.05294  -7.221 5.15e-13 ***
#X7           0.21979    0.02762   7.957 1.77e-15 ***
#X31          1.16705    0.25641   4.551 5.33e-06 ***
#X32          1.04339    0.25526   4.088 4.36e-05 ***
#X33          1.07765    0.25851   4.169 3.06e-05 ***
#X1          -0.17232    0.02964  -5.813 6.12e-09 ***
#X41          1.53514    0.75958   2.021 0.043277 *
#X42          1.31617    0.75979   1.732 0.083222 .
#X43          1.50764    0.78100   1.930 0.053558 .
#X22         -0.15731    0.04317  -3.644 0.000269 ***
#propState   -0.06373    0.02609  -2.442 0.014587 *
#propDebt     0.05724    0.02789   2.052 0.040158 *
#---
#Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
#
#(Dispersion parameter for binomial family taken to be 1)
#
#Null deviance: 15809  on 14998  degrees of freedom
#Residual deviance: 13833  on 14985  degrees of freedom
#AIC: 13861
#
#Number of Fisher Scoring iterations: 5
```

By looking at the distribution of Y variable in the training set, I found that the dataset is highly unbalanced, so I train the cutoff value that decides the prediction results. I choose the objective to be the geometric mean of the true negative proportion and true positive proportion, because I assume we care these two proportions equally. If one proportion is more valued than the other one, I can maximize a weighted mean of these two proportions to address that. The optimal cutoff is 0.23 and the optimal geometric mean is 0.6793.

```
# 0         1
# 0.7799187 0.2200813
```

I next fit a logistic regression model with L2 regularization using the glmnet package. The best regularization parameter lambda is trained using cross-validation. I use the same set of

features in the above model for this model. The optimal cutoff is still 0.23 and the optimal geometric mean is 0.6809.

4. Random forest
   I applied random forest classifier to predict Y. I construct 200 randomized decision trees. If time allows, I should use a much larger number of trees to prevent overfitting. I train each tree with bootstrapped sample from training set. At each node I randomly choose a subset of features from all the features. The size of the subset $k = \sqrt{total\ number\ of\ features}$. To further eliminate over-fitting, I set the minimum samples size in each node to be 10. At last, I want to tune the cutoff value in the random forest model with 10-fold cross-validation. Since r is very slow in running random forest functions, I decide to use 0.25 for the cutoff value which is from the optimal cutoff value trained in the logistic regression model previously. The training performance has a geometric mean of 0.9817, which is very high.

5. Validation
   I use validation set to select the better model of the three. The function in the r script validPerf applies both models to the validation set and the performances measure by the geometric mean are 0.6787 for logistic regression without regularization, 0.6746 for logistic regression with regularization and 0.6978 for random forest respectively. I select the first model that has a slightly better performance than the second model and has a closet training and validation performance. Random forest has a validation performance that is much worse than the training performance, which indicates overfitting.

6. Testing
   The function testPerf in the r script applies the first logistic regression model to the testing data and returns the confusion table.