Final Assignment - BankAuto

Manish Grewal

28/11/2020

library(readxl)  
library(car)  
library(caret)  
library(ROSE)  
library(pROC)  
library(rpart)  
library(rpart.plot)  
library(randomForest)

## Load dataset and check for missing values

setwd("C:/Users/manish.grewal/emdp/R/Final")  
xlsx <- read\_excel("IMB469-XLS-ENG.xlsx")  
  
# Removing not recommended columns  
xlsx <- xlsx[, -c(1:3,17,18,20)]  
  
#Check for missing values  
paste0("Is any NA?: ", any(is.na(xlsx)))

## [1] "Is any NA?: FALSE"

#Check for null values  
paste0("Is any null?: ", any(is.null(xlsx)))

## [1] "Is any null?: FALSE"

The columns that are not recommended for model building are removed except for the column AGE which may be used in building alternative model.

There are no missing or null values in the truncated data set.

Missing values are present in the “start date” column but this column has been omitted from the truncated dataset.

# Question 1. Build and interpret a logistic regression model to identify good and bad customers – a good customer is one who has never defaulted, any customer with a single default is a bad customer (15 Marks)

### Pre-process the data

All columns are loaded as numeric variables from the excel dataset. The first five variables take continuous values. The rest of the columns take values 0 and 1 with interpretation as provided in the case study. Convert the variables that take 0 and 1 values into factors.

xlsx[, 7:14] <- lapply(xlsx[, 7:14],   
 function(x) {recode(x, "0=0; 1=1", as.factor = TRUE)})

# Remove AGE column (using only the recommended columns for model-building)  
data <- xlsx[, -c(1)]

Below is the structure of the data after pre-processing:

str(data)

## tibble [28,906 x 13] (S3: tbl\_df/tbl/data.frame)  
## $ NOOFDEPE : num [1:28906] 2 2 2 0 2 2 2 0 0 0 ...  
## $ MTHINCTH : num [1:28906] 4.5 5.59 8.8 5 12 ...  
## $ SALDATFR : num [1:28906] 1 1 1 1 1 1 1 1 0.97 1 ...  
## $ TENORYR : num [1:28906] 1.5 2 1 1 1 2 1 2 1.5 2 ...  
## $ DWNPMFR : num [1:28906] 0.27 0.25 0.51 0.66 0.17 0.18 0.37 0.42 0.27 0.47 ...  
## $ PROFBUS : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ QUALHSC : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 1 ...  
## $ QUAL\_PG : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ SEXCODE : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...  
## $ FULLPDC : Factor w/ 2 levels "0","1": 2 2 2 2 2 1 1 2 2 2 ...  
## $ FRICODE : Factor w/ 2 levels "0","1": 1 2 2 2 2 1 1 1 1 1 ...  
## $ WASHCODE : Factor w/ 2 levels "0","1": 1 1 2 2 1 1 1 1 1 1 ...  
## $ DefaulterFlag: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...

### Split the data into training and validation set in 80:20 ratio

set.seed(123)  
myindex <- createDataPartition(data$DefaulterFlag, p = 0.8, list = FALSE)  
traindata <- train <- data[c(myindex),]  
testdata <- test <- data[-c(myindex),]

### Base model with recommended columns

set.seed(123)  
reg1 <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg1)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.6874 -1.0547 0.5789 0.7799 1.7711   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.111921 0.117000 9.504 < 2e-16 \*\*\*  
## NOOFDEPE 0.026514 0.010056 2.637 0.008374 \*\*   
## MTHINCTH -0.006510 0.003371 -1.931 0.053499 .   
## SALDATFR -0.482396 0.038695 -12.467 < 2e-16 \*\*\*  
## TENORYR 0.709133 0.042249 16.785 < 2e-16 \*\*\*  
## DWNPMFR -1.087016 0.121745 -8.929 < 2e-16 \*\*\*  
## PROFBUS1 0.262845 0.045363 5.794 6.86e-09 \*\*\*  
## QUALHSC1 0.127784 0.038062 3.357 0.000787 \*\*\*  
## QUAL\_PG1 -0.363409 0.075041 -4.843 1.28e-06 \*\*\*  
## SEXCODE1 0.217094 0.056640 3.833 0.000127 \*\*\*  
## FULLPDC1 -1.289056 0.032927 -39.149 < 2e-16 \*\*\*  
## FRICODE1 -0.244298 0.036192 -6.750 1.48e-11 \*\*\*  
## WASHCODE1 -0.130400 0.042963 -3.035 0.002404 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 27775 on 23124 degrees of freedom  
## Residual deviance: 24834 on 23112 degrees of freedom  
## AIC: 24860  
##   
## Number of Fisher Scoring iterations: 4

### Base model interpretation

Based on the coefficients of the dependent variables and their significance levels, we can deduce the following:

* **NOOFDEPE**: As the number of dependents increase, the chance of turning into a defaulter increases and this is significant at 0.1% level.
* **MTHINCTH**: As the monthly income in thousands increases the chance of turning into a defaulter decreases and this is significant at 5% level.
* **SALDATFR**: The later in the month a person receives salary decreases the chance of becoming a defaulter and this is highly significant.
* **TENORYR**: As the tenor in years increases the chance of turning into a defaulter increases and this is highly significant.
* **DWNPMFR**: As the fraction of loan in down-payment increases the chance of turning into a defaulter decreases and this is highly significant.
* **PROFBUS1**: A person with business is more likely to become a defaulter compared to a professional and this is highly significant.
* **QUALHSC**: A HSC person is more likely to become a defaulter compared to a graduate and this is highly significant.
* **QUAL\_PG**: A post-graduate is less likely to become a defaulter compared to a graduate and this is highly significant.
* **SEXCODE**: A male is more likely to become a defaulter compared to a female and this is highly significant
* **FULLPDC**: A person who has given post-dated checks in full is less likely to become a defaulter compared to a person who has not and and this is highly significant.
* **FRICODE**: A person who owns a refrigerator is less likely to become a defaulter compared to a person who does not and this is highly significant.
* **WASHCODE**: A person who owns a washing machine is less likely to become a defaulter compared to a person who does not and this is highly significant.

### Base model validation with test data

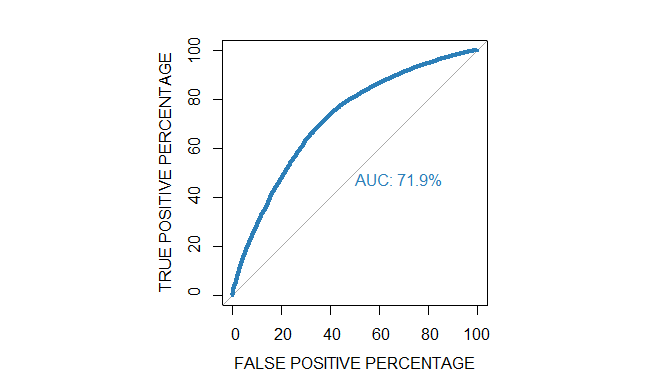
reg1\_class <- predict(reg1, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg1\_class > 0.5, 1, 0))  
cm1 <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm1

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 432 347  
## 1 1234 3768  
##   
## Accuracy : 0.7265   
## 95% CI : (0.7148, 0.738)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 0.006862   
##   
## Kappa : 0.2079   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.9157   
## Specificity : 0.2593   
## Pos Pred Value : 0.7533   
## Neg Pred Value : 0.5546   
## Prevalence : 0.7118   
## Detection Rate : 0.6518   
## Detection Prevalence : 0.8652   
## Balanced Accuracy : 0.5875   
##   
## 'Positive' Class : 1   
##

The accuracy of the model at 72.6517903% is only marginally better than the no information rate of 71.1814565%. The sensitivity is good at 91.5674362%, however the specificity is low at 25.9303721%. This means the model is able to identify defaulter reasonably well, but it is bad at predicting the good customers.

### Base model - ROC

par (pty = "s")  
auc1 <- roc(train$DefaulterFlag, reg1$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



### Check for data imbalance in training data

table(traindata$DefaulterFlag)

##   
## 0 1   
## 6665 16460

Number of defaulter is significantly higher in the training data.

### Resample training data to mitigate the imbalance

over <- ovun.sample(DefaulterFlag ~ ., data = train, method = "over", seed = 123,  
 N = table(train$DefaulterFlag)[2] \* 2)$data  
under <- ovun.sample(DefaulterFlag ~ ., data = train, method = "under", seed = 123,  
 N = table(train$DefaulterFlag)[1] \* 2)$data  
both <- ovun.sample(DefaulterFlag ~ ., data = train, method = "both", seed = 123,  
 p = 0.5 )$data  
  
prop.table(table(over$DefaulterFlag))

##   
## 1 0   
## 0.5 0.5

prop.table(table(under$DefaulterFlag))

##   
## 1 0   
## 0.5 0.5

prop.table(table(both$DefaulterFlag))

##   
## 1 0   
## 0.505427 0.494573

### Base Model with Under sampling

train <- under  
  
set.seed(123)  
reg1u <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg1u)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.17536 -0.99346 0.03293 0.99353 2.30387   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.209260 0.141379 -1.480 0.13884   
## NOOFDEPE -0.031972 0.012166 -2.628 0.00859 \*\*   
## MTHINCTH 0.008498 0.004170 2.038 0.04156 \*   
## SALDATFR 0.483430 0.047014 10.283 < 2e-16 \*\*\*  
## TENORYR -0.670670 0.049459 -13.560 < 2e-16 \*\*\*  
## DWNPMFR 1.141689 0.147046 7.764 8.22e-15 \*\*\*  
## PROFBUS1 -0.268162 0.054774 -4.896 9.79e-07 \*\*\*  
## QUALHSC1 -0.139443 0.045143 -3.089 0.00201 \*\*   
## QUAL\_PG1 0.464177 0.096605 4.805 1.55e-06 \*\*\*  
## SEXCODE1 -0.276978 0.070459 -3.931 8.46e-05 \*\*\*  
## FULLPDC1 1.235051 0.039504 31.264 < 2e-16 \*\*\*  
## FRICODE1 0.243087 0.043568 5.579 2.41e-08 \*\*\*  
## WASHCODE1 0.148342 0.052982 2.800 0.00511 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 18479 on 13329 degrees of freedom  
## Residual deviance: 16514 on 13317 degrees of freedom  
## AIC: 16540  
##   
## Number of Fisher Scoring iterations: 4

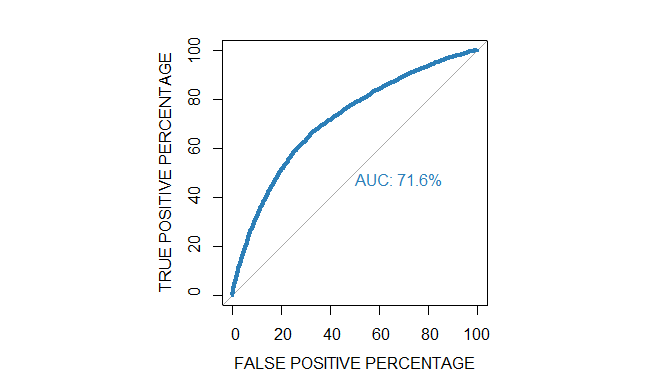
### Base Model with Under sampling - validation

reg1u\_class <- predict(reg1u, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg1u\_class > 0.5, 1, 0))  
cm1u <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm1u

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 623 2821  
## 1 1043 1294  
##   
## Accuracy : 0.3316   
## 95% CI : (0.3195, 0.3439)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.2365   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.3145   
## Specificity : 0.3739   
## Pos Pred Value : 0.5537   
## Neg Pred Value : 0.1809   
## Prevalence : 0.7118   
## Detection Rate : 0.2238   
## Detection Prevalence : 0.4043   
## Balanced Accuracy : 0.3442   
##   
## 'Positive' Class : 1   
##

### Base Model with Under sampling - ROC

par (pty = "s")  
auc1u <- roc(train$DefaulterFlag, reg1u$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



### Base model with Over sampling

train <- over  
  
set.seed(123)  
reg1o <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg1o)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.16883 -0.98485 0.03864 0.97869 2.32993   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.270521 0.089833 -3.011 0.00260 \*\*   
## NOOFDEPE -0.032805 0.007704 -4.258 2.06e-05 \*\*\*  
## MTHINCTH 0.008176 0.002632 3.107 0.00189 \*\*   
## SALDATFR 0.485721 0.030018 16.181 < 2e-16 \*\*\*  
## TENORYR -0.687014 0.031798 -21.605 < 2e-16 \*\*\*  
## DWNPMFR 1.150559 0.093898 12.253 < 2e-16 \*\*\*  
## PROFBUS1 -0.290296 0.035311 -8.221 < 2e-16 \*\*\*  
## QUALHSC1 -0.110014 0.028884 -3.809 0.00014 \*\*\*  
## QUAL\_PG1 0.423929 0.060038 7.061 1.65e-12 \*\*\*  
## SEXCODE1 -0.215116 0.044539 -4.830 1.37e-06 \*\*\*  
## FULLPDC1 1.284895 0.025294 50.798 < 2e-16 \*\*\*  
## FRICODE1 0.254709 0.027823 9.155 < 2e-16 \*\*\*  
## WASHCODE1 0.140868 0.033865 4.160 3.19e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 45637 on 32919 degrees of freedom  
## Residual deviance: 40538 on 32907 degrees of freedom  
## AIC: 40564  
##   
## Number of Fisher Scoring iterations: 4

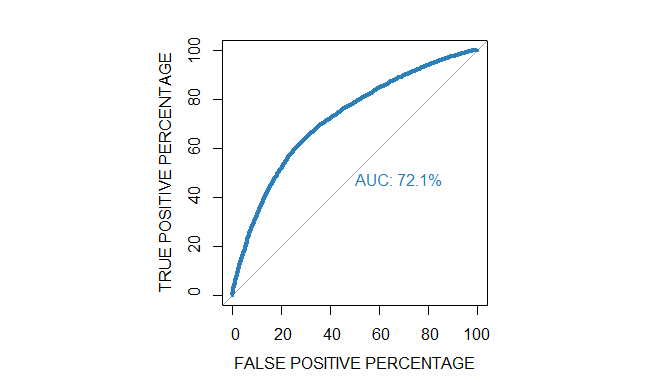
### Base Model with Over sampling - validation

reg1o\_class <- predict(reg1o, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg1o\_class > 0.5, 1, 0))  
cm1o <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm1o

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 623 2828  
## 1 1043 1287  
##   
## Accuracy : 0.3304   
## 95% CI : (0.3183, 0.3427)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.2376   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.3128   
## Specificity : 0.3739   
## Pos Pred Value : 0.5524   
## Neg Pred Value : 0.1805   
## Prevalence : 0.7118   
## Detection Rate : 0.2226   
## Detection Prevalence : 0.4030   
## Balanced Accuracy : 0.3434   
##   
## 'Positive' Class : 1   
##

### Base Model with Over sampling - ROC

par (pty = "s")  
auc1o <- roc(train$DefaulterFlag, reg1o$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



### Base model with both Over and Under sampling

train <- both  
  
set.seed(123)  
reg1b <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg1b)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1808 -0.9777 -0.5652 0.9913 2.3004   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.442473 0.107367 -4.121 3.77e-05 \*\*\*  
## NOOFDEPE -0.031169 0.009298 -3.352 0.000801 \*\*\*  
## MTHINCTH 0.012549 0.003216 3.902 9.52e-05 \*\*\*  
## SALDATFR 0.511403 0.035913 14.240 < 2e-16 \*\*\*  
## TENORYR -0.636083 0.037361 -17.025 < 2e-16 \*\*\*  
## DWNPMFR 1.349917 0.111429 12.115 < 2e-16 \*\*\*  
## PROFBUS1 -0.172222 0.041723 -4.128 3.66e-05 \*\*\*  
## QUALHSC1 -0.160197 0.034568 -4.634 3.58e-06 \*\*\*  
## QUAL\_PG1 0.447674 0.072895 6.141 8.18e-10 \*\*\*  
## SEXCODE1 -0.246379 0.053461 -4.609 4.05e-06 \*\*\*  
## FULLPDC1 1.280778 0.030085 42.572 < 2e-16 \*\*\*  
## FRICODE1 0.228165 0.033457 6.820 9.12e-12 \*\*\*  
## WASHCODE1 0.130663 0.040376 3.236 0.001211 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 32055 on 23124 degrees of freedom  
## Residual deviance: 28459 on 23112 degrees of freedom  
## AIC: 28485  
##   
## Number of Fisher Scoring iterations: 4

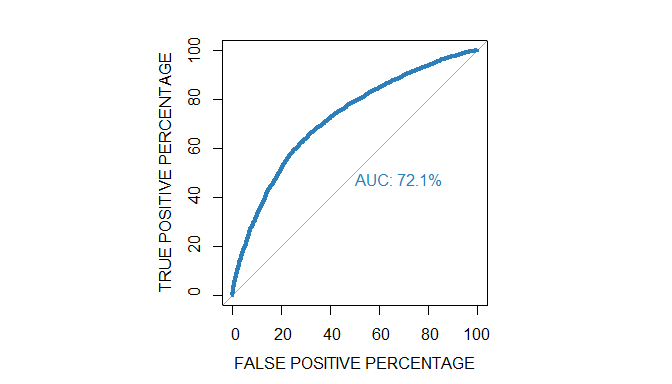
### Base model with both Over and Under sampling - validation

reg1b\_class <- predict(reg1b, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg1b\_class > 0.5, 1, 0))  
cm1b <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm1b

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 626 2828  
## 1 1040 1287  
##   
## Accuracy : 0.3309   
## 95% CI : (0.3188, 0.3432)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.2361   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.3128   
## Specificity : 0.3758   
## Pos Pred Value : 0.5531   
## Neg Pred Value : 0.1812   
## Prevalence : 0.7118   
## Detection Rate : 0.2226   
## Detection Prevalence : 0.4025   
## Balanced Accuracy : 0.3443   
##   
## 'Positive' Class : 1   
##

### Base model with both Over and Under sampling - ROC

par (pty = "s")  
auc1b <- roc(train$DefaulterFlag, reg1b$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



## Summary and comparison of models - Question 1

summ1 <- data.frame("Model" = c("Q1. Logistic - Base", "Q1. Logistic - Base - Under", "Q1. Logistic - Base - Over", "Q1. Logistic - Base - Both"),  
 "Accuracy" = c(cm1$overall[1], cm1u$overall[1],   
 cm1o$overall[1], cm1b$overall[1]),  
 "Sensitivity" = c(cm1$byClass[1], cm1u$byClass[1],   
 cm1o$byClass[1], cm1b$byClass[1]),  
 "Specificity" = c(cm1$byClass[2], cm1u$byClass[2],   
 cm1o$byClass[2], cm1b$byClass[2]),  
 "AUC" = c(auc1$auc, auc1u$auc, auc1o$auc, auc1b$auc))  
  
knitr::kable(summ1)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy | Sensitivity | Specificity | AUC |
| Q1. Logistic - Base | 0.7265179 | 0.9156744 | 0.2593037 | 71.89709 |
| Q1. Logistic - Base - Under | 0.3316035 | 0.3144593 | 0.3739496 | 71.60433 |
| Q1. Logistic - Base - Over | 0.3303927 | 0.3127582 | 0.3739496 | 72.08501 |
| Q1. Logistic - Base - Both | 0.3309116 | 0.3127582 | 0.3757503 | 72.12659 |

## Conclusion - Question 1

All three models with re-sampled data have poor accuracy of 33% with sensitivity and specificity at around 31% and 37% respectively. The accuracy of re-sampled models is much lower than the base model

The Area under Receiver Operating Characteristics curve (AUC) is also lower than the base model for under sampled data. So, we reject the model with under sampling compared to the base model.

The AUC for the “over” and both" models are marginally higher than the base model. However, based on accuracy, we reject the resampled models.

The “base” model has the best characteristics out of the models tried so far.

# Question 2. Build alternative logistic regression models & check the predictive accuracy of the models and construct the ROC and estimate the AUC (10 Marks)

## Alternative model 1 - with AGE

To build alternative model, we include AGE column from the original data set. As per the Exhibit I in the case study, relation of AGE between defaulters and non defaulters has been analyzed. Following are the columns included in the alternative model:

data <- xlsx  
names(data)

## [1] "AGE" "NOOFDEPE" "MTHINCTH" "SALDATFR"   
## [5] "TENORYR" "DWNPMFR" "PROFBUS" "QUALHSC"   
## [9] "QUAL\_PG" "SEXCODE" "FULLPDC" "FRICODE"   
## [13] "WASHCODE" "DefaulterFlag"

### Split the data into training and validation set in 80:20 ratio

set.seed(123)  
myindex <- createDataPartition(data$DefaulterFlag, p = 0.8, list = FALSE)  
traindata <- train <- data[c(myindex),]  
testdata <- test <- data[-c(myindex),]

### Alternative model 1

set.seed(123)  
reg2 <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg2)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.6644 -1.0473 0.5729 0.7819 1.8267   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.556424 0.127955 12.164 < 2e-16 \*\*\*  
## AGE -0.014066 0.001605 -8.765 < 2e-16 \*\*\*  
## NOOFDEPE 0.037130 0.010177 3.648 0.000264 \*\*\*  
## MTHINCTH -0.003829 0.003398 -1.127 0.259778   
## SALDATFR -0.455758 0.038881 -11.722 < 2e-16 \*\*\*  
## TENORYR 0.697958 0.042385 16.467 < 2e-16 \*\*\*  
## DWNPMFR -1.130965 0.122110 -9.262 < 2e-16 \*\*\*  
## PROFBUS1 0.245155 0.045474 5.391 7.00e-08 \*\*\*  
## QUALHSC1 0.133552 0.038143 3.501 0.000463 \*\*\*  
## QUAL\_PG1 -0.372103 0.075165 -4.951 7.40e-07 \*\*\*  
## SEXCODE1 0.259937 0.056922 4.567 4.96e-06 \*\*\*  
## FULLPDC1 -1.299345 0.033025 -39.344 < 2e-16 \*\*\*  
## FRICODE1 -0.235447 0.036273 -6.491 8.53e-11 \*\*\*  
## WASHCODE1 -0.135602 0.043029 -3.151 0.001625 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 27775 on 23124 degrees of freedom  
## Residual deviance: 24757 on 23111 degrees of freedom  
## AIC: 24785  
##   
## Number of Fisher Scoring iterations: 4

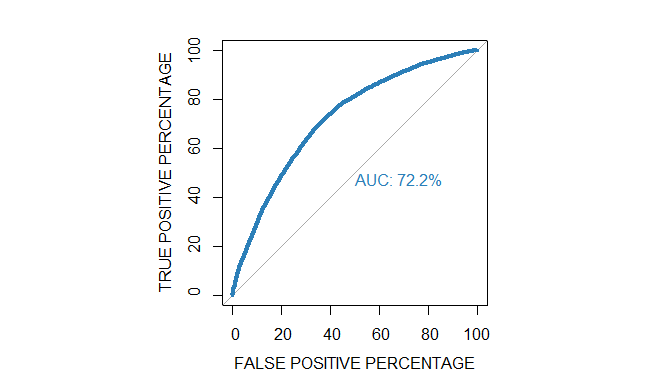
### Alternative model 1 - validation

reg2\_class <- predict(reg2, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg2\_class > 0.5, 1, 0))  
cm2 <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm2

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 463 347  
## 1 1203 3768  
##   
## Accuracy : 0.7319   
## 95% CI : (0.7203, 0.7433)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 0.0003679   
##   
## Kappa : 0.2285   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.9157   
## Specificity : 0.2779   
## Pos Pred Value : 0.7580   
## Neg Pred Value : 0.5716   
## Prevalence : 0.7118   
## Detection Rate : 0.6518   
## Detection Prevalence : 0.8599   
## Balanced Accuracy : 0.5968   
##   
## 'Positive' Class : 1   
##

### Alternative model 1 - ROC

par (pty = "s")  
auc2 <- roc(train$DefaulterFlag, reg2$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



## Alternative model 2 - with AGE, without MTHINCTH

As the variable MTHINCTH is not significant in the model, we build a simpler model without including this variable.

### Split the data into training and validation set in 80:20 ratio

train <- subset(train, select = -c(MTHINCTH))  
test <- subset(test, select = -c(MTHINCTH))

### Alternative model 2

set.seed(123)  
reg2a <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg2a)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.6654 -1.0460 0.5729 0.7824 1.8177   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.514831 0.122510 12.365 < 2e-16 \*\*\*  
## AGE -0.014230 0.001598 -8.904 < 2e-16 \*\*\*  
## NOOFDEPE 0.037148 0.010175 3.651 0.000261 \*\*\*  
## SALDATFR -0.448969 0.038401 -11.691 < 2e-16 \*\*\*  
## TENORYR 0.703304 0.042141 16.689 < 2e-16 \*\*\*  
## DWNPMFR -1.115310 0.121324 -9.193 < 2e-16 \*\*\*  
## PROFBUS1 0.241258 0.045343 5.321 1.03e-07 \*\*\*  
## QUALHSC1 0.137594 0.037973 3.623 0.000291 \*\*\*  
## QUAL\_PG1 -0.376761 0.075049 -5.020 5.16e-07 \*\*\*  
## SEXCODE1 0.261706 0.056897 4.600 4.23e-06 \*\*\*  
## FULLPDC1 -1.303388 0.032838 -39.691 < 2e-16 \*\*\*  
## FRICODE1 -0.241055 0.035930 -6.709 1.96e-11 \*\*\*  
## WASHCODE1 -0.136614 0.043022 -3.175 0.001496 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 27775 on 23124 degrees of freedom  
## Residual deviance: 24758 on 23112 degrees of freedom  
## AIC: 24784  
##   
## Number of Fisher Scoring iterations: 4

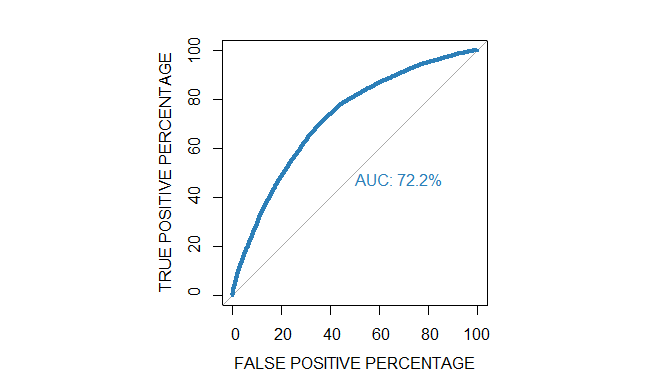
### Alternative model 2 - validation

reg2a\_class <- predict(reg2a, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg2a\_class > 0.5, 1, 0))  
cm2a <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm2a

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 457 343  
## 1 1209 3772  
##   
## Accuracy : 0.7315   
## 95% CI : (0.7199, 0.7429)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 0.000455   
##   
## Kappa : 0.2259   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.9166   
## Specificity : 0.2743   
## Pos Pred Value : 0.7573   
## Neg Pred Value : 0.5713   
## Prevalence : 0.7118   
## Detection Rate : 0.6525   
## Detection Prevalence : 0.8616   
## Balanced Accuracy : 0.5955   
##   
## 'Positive' Class : 1   
##

### Alternative model 2 - ROC

par (pty = "s")  
auc2a <- roc(train$DefaulterFlag, reg2a$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



### Check for data imbalance in training data

prop.table(table(traindata$DefaulterFlag))

##   
## 0 1   
## 0.2882162 0.7117838

Number of defaulter is significantly higher in the training data.

### Resample training data to mitigate the imbalance

over <- ovun.sample(DefaulterFlag ~ ., data = train, method = "over", seed = 123,  
 N = table(train$DefaulterFlag)[2] \* 2)$data  
under <- ovun.sample(DefaulterFlag ~ ., data = train, method = "under", seed = 123,  
 N = table(train$DefaulterFlag)[1] \* 2)$data  
both <- ovun.sample(DefaulterFlag ~ ., data = train, method = "both", seed = 123,  
 p = 0.5 )$data  
  
paste("Over sampling")

## [1] "Over sampling"

table(over$DefaulterFlag)

##   
## 1 0   
## 16460 16460

paste("Over sampling")

## [1] "Over sampling"

table(under$DefaulterFlag)

##   
## 1 0   
## 6665 6665

paste("Both sampling")

## [1] "Both sampling"

table(both$DefaulterFlag)

##   
## 1 0   
## 11688 11437

### Alternative model 2 with Under sampling

train <- under  
  
set.seed(123)  
reg2u <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg2u)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.20813 -0.99429 0.01221 0.99450 2.28006   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.54838 0.14696 -3.732 0.000190 \*\*\*  
## AGE 0.01306 0.00191 6.839 7.99e-12 \*\*\*  
## NOOFDEPE -0.04219 0.01228 -3.435 0.000592 \*\*\*  
## SALDATFR 0.44689 0.04665 9.581 < 2e-16 \*\*\*  
## TENORYR -0.66602 0.04931 -13.507 < 2e-16 \*\*\*  
## DWNPMFR 1.16196 0.14642 7.936 2.10e-15 \*\*\*  
## PROFBUS1 -0.25095 0.05473 -4.585 4.53e-06 \*\*\*  
## QUALHSC1 -0.14777 0.04507 -3.279 0.001043 \*\*   
## QUAL\_PG1 0.48460 0.09656 5.019 5.21e-07 \*\*\*  
## SEXCODE1 -0.32240 0.07073 -4.558 5.15e-06 \*\*\*  
## FULLPDC1 1.25034 0.03945 31.697 < 2e-16 \*\*\*  
## FRICODE1 0.24474 0.04328 5.655 1.56e-08 \*\*\*  
## WASHCODE1 0.15082 0.05303 2.844 0.004456 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 18479 on 13329 degrees of freedom  
## Residual deviance: 16471 on 13317 degrees of freedom  
## AIC: 16497  
##   
## Number of Fisher Scoring iterations: 4

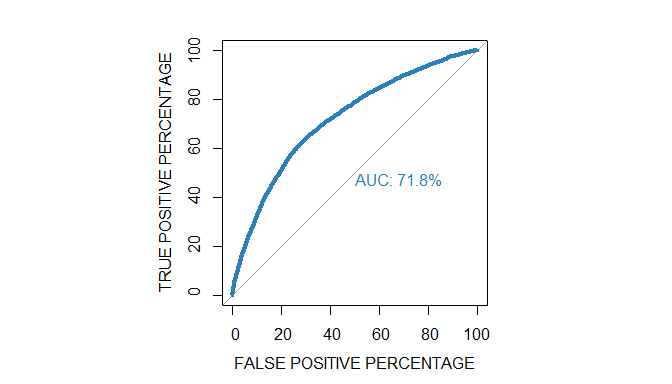
### Alternative model 2 with Under sampling - validation

reg2u\_class <- predict(reg2u, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg2u\_class > 0.5, 1, 0))  
cm2u <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm2u

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 615 2813  
## 1 1051 1302  
##   
## Accuracy : 0.3316   
## 95% CI : (0.3195, 0.3439)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.2392   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.3164   
## Specificity : 0.3691   
## Pos Pred Value : 0.5533   
## Neg Pred Value : 0.1794   
## Prevalence : 0.7118   
## Detection Rate : 0.2252   
## Detection Prevalence : 0.4070   
## Balanced Accuracy : 0.3428   
##   
## 'Positive' Class : 1   
##

### Alternative model 2 with Under sampling - ROC

par (pty = "s")  
auc2u <- roc(train$DefaulterFlag, reg2u$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



#auc2u

### Alternative model 2 with Over sampling

train <- over  
  
set.seed(123)  
reg2o <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg2o)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.20533 -0.98466 0.02327 0.98230 2.30517   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.648092 0.093939 -6.899 5.24e-12 \*\*\*  
## AGE 0.014047 0.001227 11.451 < 2e-16 \*\*\*  
## NOOFDEPE -0.043200 0.007776 -5.555 2.77e-08 \*\*\*  
## SALDATFR 0.446852 0.029793 14.999 < 2e-16 \*\*\*  
## TENORYR -0.682120 0.031736 -21.494 < 2e-16 \*\*\*  
## DWNPMFR 1.172726 0.093585 12.531 < 2e-16 \*\*\*  
## PROFBUS1 -0.272597 0.035272 -7.728 1.09e-14 \*\*\*  
## QUALHSC1 -0.120383 0.028846 -4.173 3.00e-05 \*\*\*  
## QUAL\_PG1 0.441226 0.060013 7.352 1.95e-13 \*\*\*  
## SEXCODE1 -0.258912 0.044727 -5.789 7.09e-09 \*\*\*  
## FULLPDC1 1.300556 0.025252 51.503 < 2e-16 \*\*\*  
## FRICODE1 0.255515 0.027612 9.254 < 2e-16 \*\*\*  
## WASHCODE1 0.143270 0.033917 4.224 2.40e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 45637 on 32919 degrees of freedom  
## Residual deviance: 40416 on 32907 degrees of freedom  
## AIC: 40442  
##   
## Number of Fisher Scoring iterations: 4

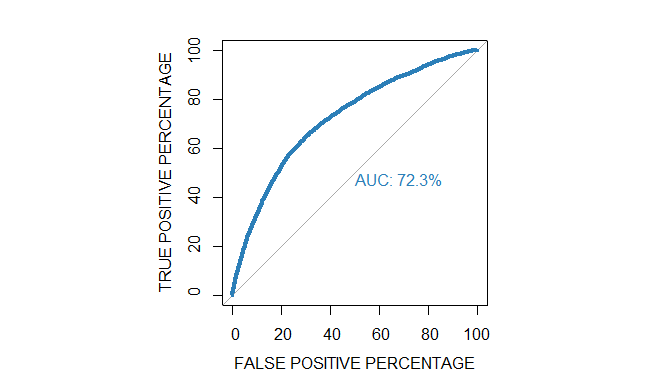
### Alternative model 2 with Over sampling - validation

reg2o\_class <- predict(reg2o, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg2o\_class > 0.5, 1, 0))  
cm2o <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm2o

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 615 2810  
## 1 1051 1305  
##   
## Accuracy : 0.3321   
## 95% CI : (0.32, 0.3444)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.2387   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.3171   
## Specificity : 0.3691   
## Pos Pred Value : 0.5539   
## Neg Pred Value : 0.1796   
## Prevalence : 0.7118   
## Detection Rate : 0.2257   
## Detection Prevalence : 0.4075   
## Balanced Accuracy : 0.3431   
##   
## 'Positive' Class : 1   
##

### Alternative model 2 with Over sampling - ROC

par (pty = "s")  
auc2o <- roc(train$DefaulterFlag, reg2o$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



auc2o

##   
## Call:  
## roc.default(response = train$DefaulterFlag, predictor = reg2o$fitted.values, percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE", col = "#2c7fb8", lwd = 4, print.auc = TRUE)  
##   
## Data: reg2o$fitted.values in 16460 controls (train$DefaulterFlag 1) < 16460 cases (train$DefaulterFlag 0).  
## Area under the curve: 72.33%

### Alternative model 2 with both Over and Under sampling

train <- both  
  
set.seed(123)  
reg2b <- glm(DefaulterFlag ~ ., family = binomial, data = train)  
summary(reg2b)

##   
## Call:  
## glm(formula = DefaulterFlag ~ ., family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1999 -0.9781 -0.5586 0.9875 2.2801   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.691779 0.111737 -6.191 5.97e-10 \*\*\*  
## AGE 0.011890 0.001460 8.145 3.81e-16 \*\*\*  
## NOOFDEPE -0.039633 0.009376 -4.227 2.37e-05 \*\*\*  
## SALDATFR 0.469824 0.035645 13.181 < 2e-16 \*\*\*  
## TENORYR -0.639096 0.037240 -17.162 < 2e-16 \*\*\*  
## DWNPMFR 1.344314 0.110964 12.115 < 2e-16 \*\*\*  
## PROFBUS1 -0.150677 0.041652 -3.618 0.000297 \*\*\*  
## QUALHSC1 -0.175507 0.034509 -5.086 3.66e-07 \*\*\*  
## QUAL\_PG1 0.473328 0.072735 6.508 7.64e-11 \*\*\*  
## SEXCODE1 -0.288194 0.053647 -5.372 7.79e-08 \*\*\*  
## FULLPDC1 1.297774 0.030015 43.238 < 2e-16 \*\*\*  
## FRICODE1 0.236888 0.033192 7.137 9.55e-13 \*\*\*  
## WASHCODE1 0.137619 0.040408 3.406 0.000660 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 32055 on 23124 degrees of freedom  
## Residual deviance: 28408 on 23112 degrees of freedom  
## AIC: 28434  
##   
## Number of Fisher Scoring iterations: 4

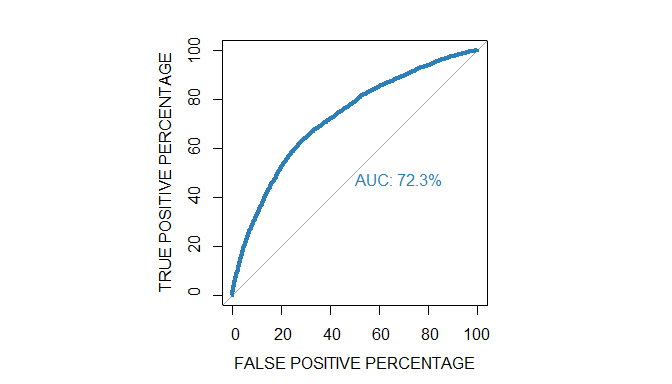
### Alternative model 2 with both Over and Under sampling - validation

reg2b\_class <- predict(reg2b, newdata = test, type = "response")  
pred <- as.factor(ifelse(reg2b\_class > 0.5, 1, 0))  
cm2b <- confusionMatrix(pred, test$DefaulterFlag, positive = "1")  
cm2b

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 628 2825  
## 1 1038 1290  
##   
## Accuracy : 0.3318   
## 95% CI : (0.3196, 0.3441)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.2347   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.3135   
## Specificity : 0.3770   
## Pos Pred Value : 0.5541   
## Neg Pred Value : 0.1819   
## Prevalence : 0.7118   
## Detection Rate : 0.2231   
## Detection Prevalence : 0.4027   
## Balanced Accuracy : 0.3452   
##   
## 'Positive' Class : 1   
##

### Alternative model 2 with both Over and Under sampling - ROC

par(pty = "s")  
auc2b <- roc(train$DefaulterFlag, reg2b$fitted.values,   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE",   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE)



auc2b

##   
## Call:  
## roc.default(response = train$DefaulterFlag, predictor = reg2b$fitted.values, percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "FALSE POSITIVE PERCENTAGE", ylab = "TRUE POSITIVE PERCENTAGE", col = "#2c7fb8", lwd = 4, print.auc = TRUE)  
##   
## Data: reg2b$fitted.values in 11688 controls (train$DefaulterFlag 1) < 11437 cases (train$DefaulterFlag 0).  
## Area under the curve: 72.29%

## Summary and comparion of alternative models - Question 2

summ2 <- data.frame("Model" = c("Q2. Alternate 1", "Q2. Alternate 2", "Q2. Alternate 2 - Under", "Q2. Alternate 2 - Over", "Q2. Alternate 2 - Both"),  
 "Accuracy" = c(cm2$overall[1], cm2a$overall[1], cm2u$overall[1],   
 cm2o$overall[1], cm2b$overall[1]),  
 "Sensitivity" = c(cm2$byClass[1], cm2a$byClass[1], cm2u$byClass[1],   
 cm2o$byClass[1], cm2b$byClass[1]),  
 "Specificity" = c(cm2$byClass[2], cm2a$byClass[2], cm2u$byClass[2],   
 cm2o$byClass[2], cm2b$byClass[2]),  
 "AUC" = c(auc2$auc, auc2a$auc, auc2u$auc, auc2o$auc, auc2b$auc))  
  
knitr::kable(summ2)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy | Sensitivity | Specificity | AUC |
| Q2. Alternate 1 | 0.7318803 | 0.9156744 | 0.2779112 | 72.17796 |
| Q2. Alternate 2 | 0.7315343 | 0.9166464 | 0.2743097 | 72.17146 |
| Q2. Alternate 2 - Under | 0.3316035 | 0.3164034 | 0.3691477 | 71.81490 |
| Q2. Alternate 2 - Over | 0.3321225 | 0.3171324 | 0.3691477 | 72.33437 |
| Q2. Alternate 2 - Both | 0.3317765 | 0.3134872 | 0.3769508 | 72.28653 |

## Conclusion - Question 2

All three models with re-sampled data have poor accuracy of 33% with sensitivity and specificity at around 31% and 37% respectively. The accuracy of resampled models is much lower than the base model

The Area under Receiver Operating Characteristics curve (AUC) is also lower than the base model for under sampled data. So, we reject the model with under sampling compared to the base model.

The AUC for the “over” and both" models are marginally higher than the base model. However, based on accuracy, we reject the resampled models.

The “base” model has the best characteristics out of the models tried so far.

# Question 3. Apply alternative machine learning algorithms (e.g., Decision Tree & Random Forest) and examine whether the use of these algorithms is suitable to add better predictive power over the logistic regression constructed to predict the creditworthiness of the customers (20 Marks [10+10]).

data[,1] <- (data[,1] - min(data[,1])) / (max(data[,1]) - min(data[,1]))  
data[,2] <- (data[,2] - min(data[,2])) / (max(data[,2]) - min(data[,2]))  
data[,3] <- (data[,3] - min(data[,3])) / (max(data[,3]) - min(data[,3]))  
data[,4] <- (data[,4] - min(data[,4])) / (max(data[,4]) - min(data[,4]))  
data[,5] <- (data[,5] - min(data[,5])) / (max(data[,5]) - min(data[,5]))  
data[,6] <- (data[,6] - min(data[,6])) / (max(data[,6]) - min(data[,6]))  
  
set.seed(123)  
myindex <- createDataPartition(data$DefaulterFlag, p = 0.8, list = FALSE)  
train <- train <- data[c(myindex),]  
test <- test <- data[-c(myindex),]

### Resample training data to mitigate the imbalance

over <- ovun.sample(DefaulterFlag ~ ., data = train, method = "over", seed = 123,  
 N = table(train$DefaulterFlag)[2] \* 2)$data  
under <- ovun.sample(DefaulterFlag ~ ., data = train, method = "under", seed = 123,  
 N = table(train$DefaulterFlag)[1] \* 2)$data  
both <- ovun.sample(DefaulterFlag ~ ., data = train, method = "both", seed = 123,  
 p = 0.5 )$data  
  
prop.table(table(over$DefaulterFlag))

##   
## 1 0   
## 0.5 0.5

prop.table(table(under$DefaulterFlag))

##   
## 1 0   
## 0.5 0.5

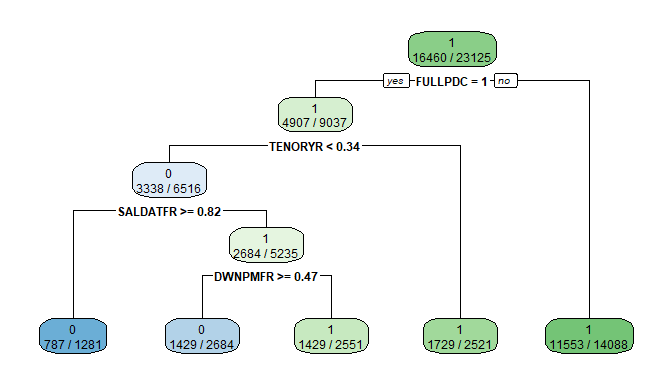
prop.table(table(both$DefaulterFlag))

##   
## 1 0   
## 0.505427 0.494573

## Decision Tree

### Decision Tree model

regdt <- rpart(DefaulterFlag ~ ., data = train)  
#summary(regdt)  
rpart.plot(regdt, extra = 2)



### Decision Tree model - validation

regdt\_class <- predict(regdt, newdata = test, type = "class")  
regdt\_class1 <- predict(regdt, newdata = test, type = "prob")  
#summary(regdt\_class)  
  
cmdt <- confusionMatrix(regdt\_class, test$DefaulterFlag, positive = "1")  
cmdt

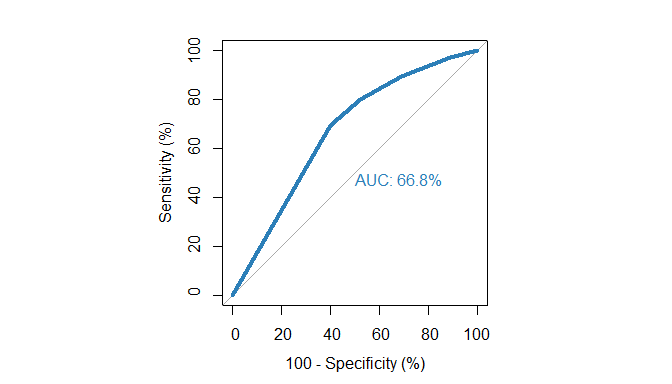
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 518 434  
## 1 1148 3681  
##   
## Accuracy : 0.7263   
## 95% CI : (0.7147, 0.7378)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 0.007445   
##   
## Kappa : 0.2355   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8945   
## Specificity : 0.3109   
## Pos Pred Value : 0.7623   
## Neg Pred Value : 0.5441   
## Prevalence : 0.7118   
## Detection Rate : 0.6367   
## Detection Prevalence : 0.8353   
## Balanced Accuracy : 0.6027   
##   
## 'Positive' Class : 1   
##

varImp(regdt)

## Overall  
## AGE 41.29440  
## DWNPMFR 296.57362  
## FRICODE 146.22971  
## FULLPDC 845.27871  
## NOOFDEPE 11.79276  
## PROFBUS 37.35086  
## SALDATFR 44.78622  
## TENORYR 389.15764  
## WASHCODE 93.52362  
## MTHINCTH 0.00000  
## QUALHSC 0.00000  
## QUAL\_PG 0.00000  
## SEXCODE 0.00000

### Decision Tree model - ROC

par(pty="s")  
aucdt <- roc(test$DefaulterFlag, regdt\_class1[,2],   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE #, print.auc.y = 80, add = TRUE  
 )



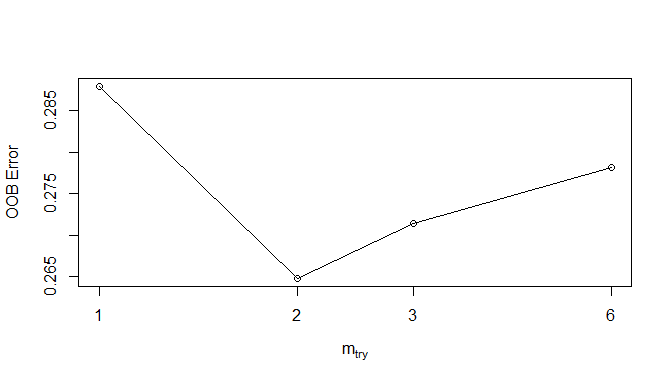
## Random Forest

set.seed(123)  
myindex <- createDataPartition(data$DefaulterFlag, p = 0.8, list = FALSE)  
train <- train <- data[c(myindex),]  
test <- test <- data[-c(myindex),]

### Random Forest model - tuning

train <- as.data.frame(train)  
tuneRF(train[,-14], train[,14],   
 stepFactor = 2, plot = TRUE, ntreeTry = 300, improve = .01)

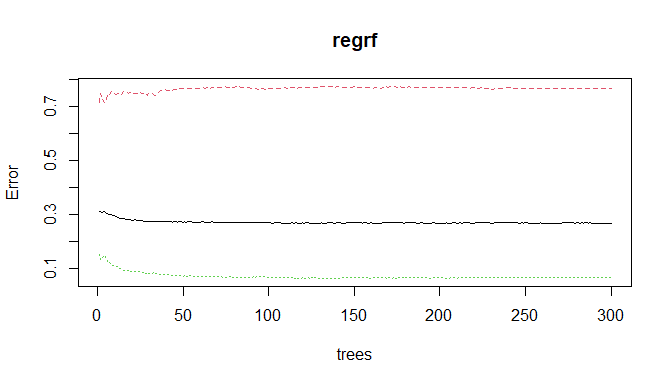
## mtry = 3 OOB error = 27.15%   
## Searching left ...  
## mtry = 2 OOB error = 26.48%   
## 0.02453011 0.01   
## mtry = 1 OOB error = 28.8%   
## -0.0873612 0.01   
## Searching right ...  
## mtry = 6 OOB error = 27.82%   
## -0.05062051 0.01



## mtry OOBError  
## 1.OOB 1 0.2879568  
## 2.OOB 2 0.2648216  
## 3.OOB 3 0.2714811  
## 6.OOB 6 0.2782270

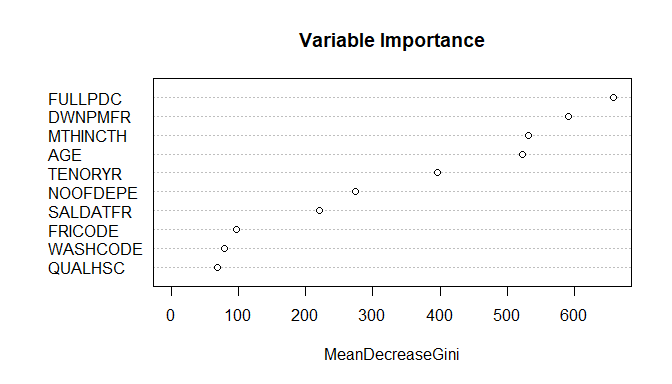
### RandomForest model

set.seed(123)  
regrf <- randomForest(DefaulterFlag ~ ., data = train, mtry = 2, ntree = 300)  
plot(regrf)



### RandomForest model - variable importance

varImpPlot(regrf, sort = TRUE, n.var = 10, main="Variable Importance")



regrf\_class <- predict(regrf, newdata = test, type = "class")  
regrf\_class1 <- predict(regrf, newdata = test, type = "prob")  
summary(regrf\_class)

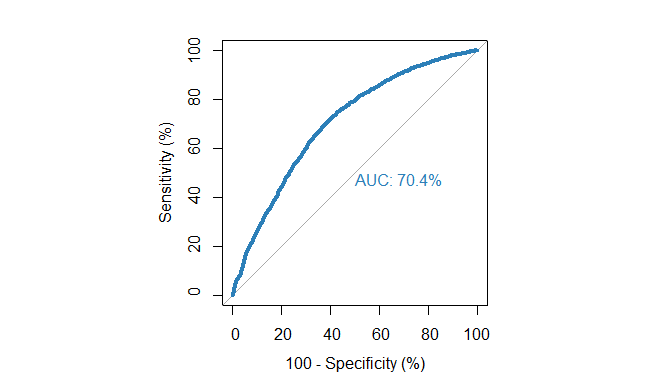
## 0 1   
## 633 5148

cmrf <- confusionMatrix(regrf\_class, test$DefaulterFlag, positive = "1")  
cmrf

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 378 255  
## 1 1288 3860  
##   
## Accuracy : 0.7331   
## 95% CI : (0.7215, 0.7445)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 0.0001703   
##   
## Kappa : 0.2022   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.9380   
## Specificity : 0.2269   
## Pos Pred Value : 0.7498   
## Neg Pred Value : 0.5972   
## Prevalence : 0.7118   
## Detection Rate : 0.6677   
## Detection Prevalence : 0.8905   
## Balanced Accuracy : 0.5825   
##   
## 'Positive' Class : 1   
##

### RandomForest model - ROC

par(pty="s")  
aucrf <- roc(test$DefaulterFlag, regrf\_class1[,2],   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE #, print.auc.y = 80, add = TRUE  
 )



### Resample training data to mitigate the imbalance

over <- ovun.sample(DefaulterFlag ~ ., data = train, method = "over", seed = 123,  
 N = table(train$DefaulterFlag)[2] \* 2)$data  
under <- ovun.sample(DefaulterFlag ~ ., data = train, method = "under", seed = 123,  
 N = table(train$DefaulterFlag)[1] \* 2)$data  
both <- ovun.sample(DefaulterFlag ~ ., data = train, method = "both", seed = 123,  
 p = 0.5 )$data  
  
prop.table(table(over$DefaulterFlag))

##   
## 1 0   
## 0.5 0.5

prop.table(table(under$DefaulterFlag))

##   
## 1 0   
## 0.5 0.5

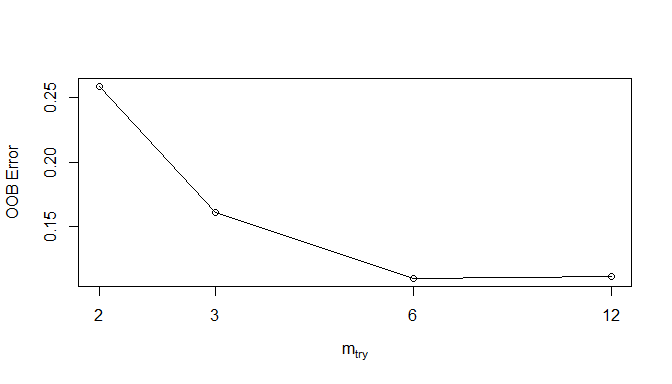
prop.table(table(both$DefaulterFlag))

##   
## 1 0   
## 0.505427 0.494573

### RandomForest with over sampled data - tuning

train <- over  
tuneRF(train[,-14], train[,14],   
 stepFactor = 2, plot = TRUE, ntreeTry = 300, improve = .01)

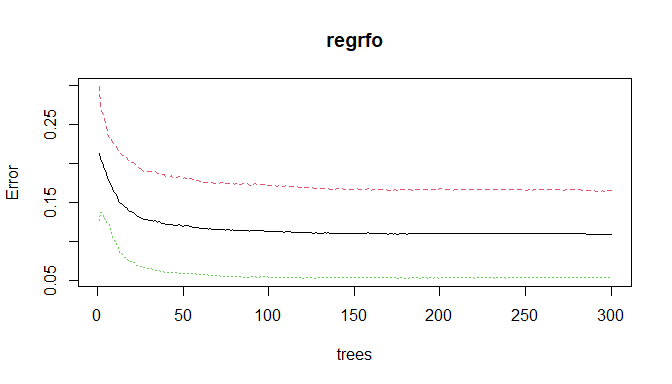
## mtry = 3 OOB error = 16.08%   
## Searching left ...  
## mtry = 2 OOB error = 25.84%   
## -0.6071024 0.01   
## Searching right ...  
## mtry = 6 OOB error = 11%   
## 0.3158292 0.01   
## mtry = 12 OOB error = 11.18%   
## -0.01656543 0.01



## mtry OOBError  
## 2.OOB 2 0.2584447  
## 3.OOB 3 0.1608141  
## 6.OOB 6 0.1100243  
## 12.OOB 12 0.1118469

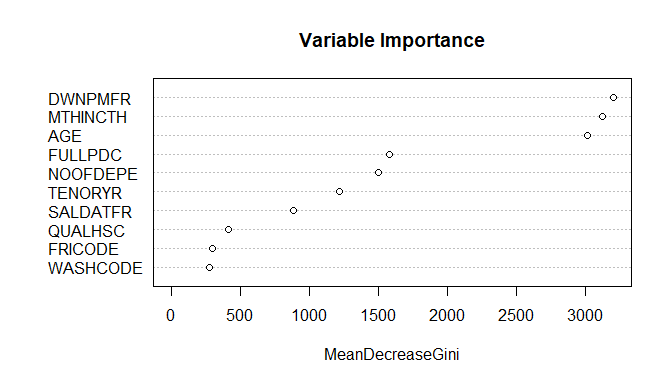
### RandomForest with over sampled data

set.seed(123)  
regrfo <- randomForest(DefaulterFlag ~ ., data = train, mtry = 6, ntree = 300)  
plot(regrfo)



### RandomForest with over sampled data - variable importance

varImpPlot(regrfo, sort = TRUE, n.var = 10, main="Variable Importance")



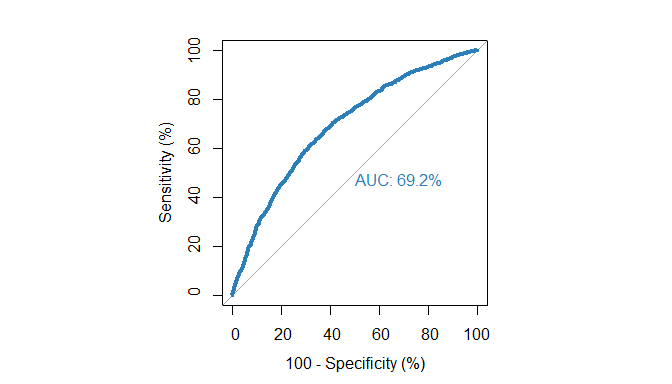
### RandomForest with over sampled data - validation

regrfo\_class <- predict(regrfo, newdata = test, type = "class")  
regrfo\_class1 <- predict(regrfo, newdata = test, type = "prob")  
  
cmrfo <- confusionMatrix(regrfo\_class, test$DefaulterFlag, positive = "1")  
cmrfo

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 667 680  
## 1 999 3435  
##   
## Accuracy : 0.7096   
## 95% CI : (0.6977, 0.7212)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 0.6531   
##   
## Kappa : 0.2493   
##   
## Mcnemar's Test P-Value : 8.445e-15   
##   
## Sensitivity : 0.8348   
## Specificity : 0.4004   
## Pos Pred Value : 0.7747   
## Neg Pred Value : 0.4952   
## Prevalence : 0.7118   
## Detection Rate : 0.5942   
## Detection Prevalence : 0.7670   
## Balanced Accuracy : 0.6176   
##   
## 'Positive' Class : 1   
##

### RandomForest with over sampled data - ROC

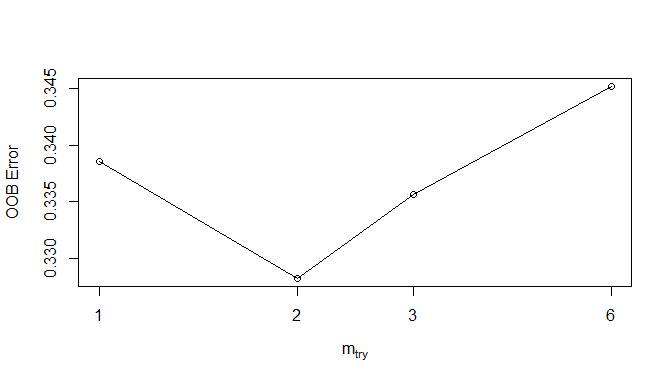
par(pty="s")  
aucrfo <- roc(test$DefaulterFlag, regrfo\_class1[,2],   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE #, print.auc.y = 80, add = TRUE  
 )



# RandomForest with under sampled data - tuning

train <- under  
set.seed(123)  
tuneRF(train[,-14], train[,14],   
 stepFactor = 2, plot = TRUE, ntreeTry = 300, improve = .01)

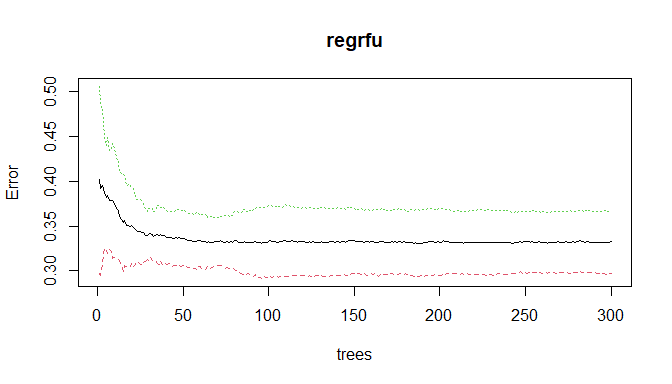
## mtry = 3 OOB error = 33.56%   
## Searching left ...  
## mtry = 2 OOB error = 32.83%   
## 0.02190434 0.01   
## mtry = 1 OOB error = 33.86%   
## -0.03130713 0.01   
## Searching right ...  
## mtry = 6 OOB error = 34.52%   
## -0.05141682 0.01



## mtry OOBError  
## 1.OOB 1 0.3385596  
## 2.OOB 2 0.3282821  
## 3.OOB 3 0.3356339  
## 6.OOB 6 0.3451613

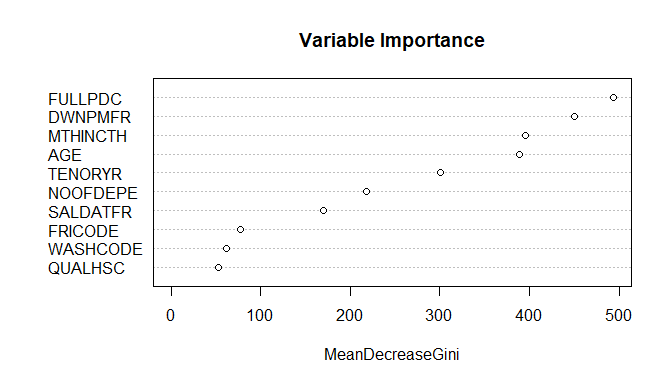
# RandomForest with under sampled data

set.seed(123)  
regrfu <- randomForest(DefaulterFlag ~ ., data = train, mtry = 2, ntree = 300)  
plot(regrfu)



### RandomForest with under sampled data - variable importance

varImpPlot(regrfu, sort = TRUE, n.var = 10, main="Variable Importance")



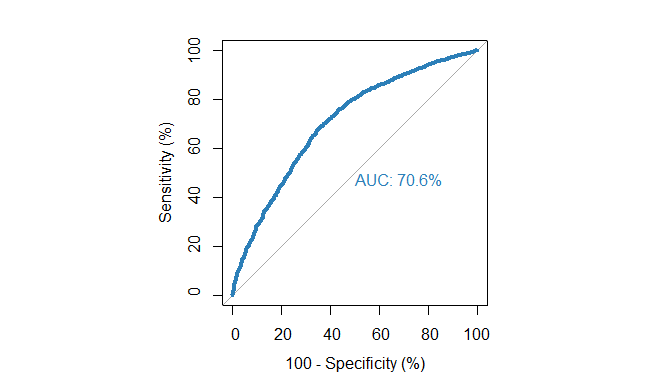
# RandomForest with under sampled data - validation

regrfu\_class <- predict(regrfu, newdata = test, type = "class")  
regrfu\_class1 <- predict(regrfu, newdata = test, type = "prob")  
  
cmrfu <- confusionMatrix(regrfu\_class, test$DefaulterFlag, positive = "1")  
cmrfu

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1023 1192  
## 1 643 2923  
##   
## Accuracy : 0.6826   
## 95% CI : (0.6704, 0.6946)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.2954   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.7103   
## Specificity : 0.6140   
## Pos Pred Value : 0.8197   
## Neg Pred Value : 0.4619   
## Prevalence : 0.7118   
## Detection Rate : 0.5056   
## Detection Prevalence : 0.6168   
## Balanced Accuracy : 0.6622   
##   
## 'Positive' Class : 1   
##

### RandomForest with under sampled data - ROC

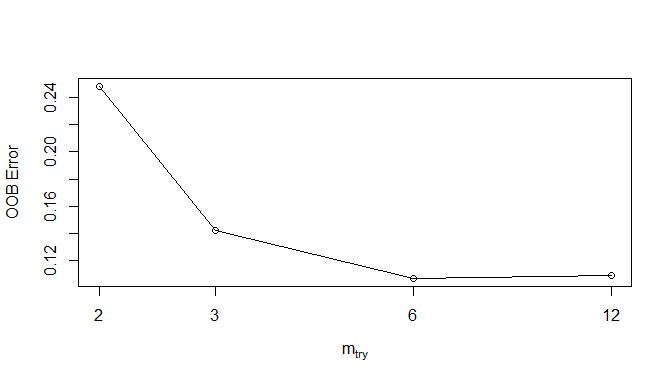
par(pty="s")  
aucrfu <- roc(test$DefaulterFlag, regrfu\_class1[,2],   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE #, print.auc.y = 80, add = TRUE  
 )



### RandomForest with both over and under sampling - tuning

train <- both  
  
set.seed(123)  
tuneRF(train[,-14], train[,14],   
 stepFactor = 2, plot = TRUE, ntreeTry = 300, improve = .01)

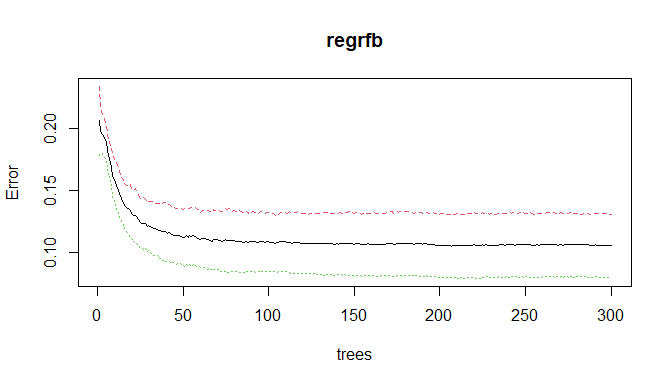
## mtry = 3 OOB error = 14.26%   
## Searching left ...  
## mtry = 2 OOB error = 24.8%   
## -0.7392359 0.01   
## Searching right ...  
## mtry = 6 OOB error = 10.72%   
## 0.2486355 0.01   
## mtry = 12 OOB error = 10.93%   
## -0.02017756 0.01



## mtry OOBError  
## 2.OOB 2 0.2480432  
## 3.OOB 3 0.1426162  
## 6.OOB 6 0.1071568  
## 12.OOB 12 0.1093189

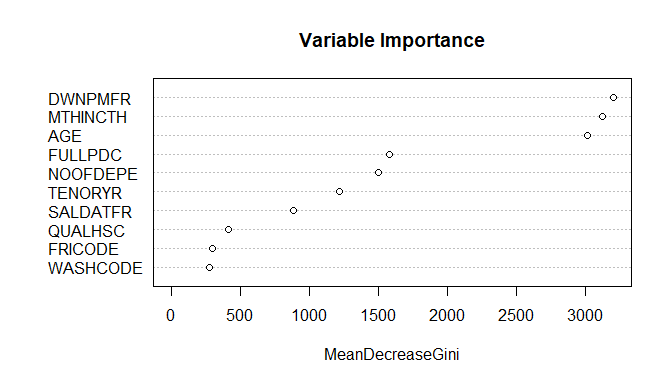
### RandomForest with both over and under sampling

set.seed(123)  
regrfb <- randomForest(DefaulterFlag ~ ., data = train, mtry = 6, ntree = 300)  
plot(regrfb)



### RandomForest with both over and under sampling - variable importance

varImpPlot(regrfo, sort = TRUE, n.var = 10, main="Variable Importance")



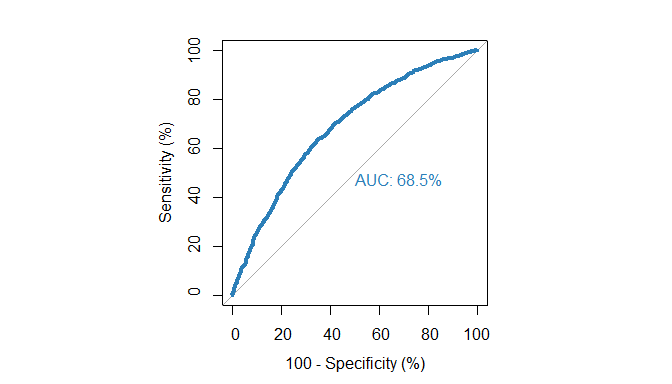
### RandomForest with both over and under sampling - validation

regrfb\_class <- predict(regrfb, newdata = test, type = "class")  
regrfb\_class1 <- predict(regrfb, newdata = test, type = "prob")  
cmrfb <- confusionMatrix(regrfb\_class, test$DefaulterFlag, positive = "1")  
cmrfb

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 893 1087  
## 1 773 3028  
##   
## Accuracy : 0.6783   
## 95% CI : (0.666, 0.6903)  
## No Information Rate : 0.7118   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.2574   
##   
## Mcnemar's Test P-Value : 3.943e-13   
##   
## Sensitivity : 0.7358   
## Specificity : 0.5360   
## Pos Pred Value : 0.7966   
## Neg Pred Value : 0.4510   
## Prevalence : 0.7118   
## Detection Rate : 0.5238   
## Detection Prevalence : 0.6575   
## Balanced Accuracy : 0.6359   
##   
## 'Positive' Class : 1   
##

### RandomForest with both over and under sampling - ROC

par(pty="s")  
aucrfb <- roc(test$DefaulterFlag, regrfb\_class1[,2],   
 plot = TRUE, legacy.axes = TRUE, percent = TRUE,   
 col = "#2c7fb8", lwd = 4, print.auc = TRUE #, print.auc.y = 80, add = TRUE  
 )



## Summary and comparion of alternative models - Question 3

summ3 <- data.frame("Model" = c("Q3. Decision Tree", "Q3. Random Forest", "Q3. Random Forest - Under", "Q3. Random Forest - Over", "Q3. Random Forest - Both"),  
 "Accuracy" =   
 c(cmdt$overall[1], cmrf$overall[1], cmrfu$overall[1], cmrfo$overall[1], cmrfb$overall[1]),  
 "Sensitivity" =   
 c(cmdt$byClass[1], cmrf$byClass[1], cmrfu$byClass[1],  
 cmrfo$byClass[1], cmrfb$byClass[1]),  
 "Specificity" =   
 c(cmdt$byClass[2], cmrf$byClass[2], cmrfu$byClass[2],  
 cmrfo$byClass[2], cmrfb$byClass[2]),  
 "AUC" = c(aucdt$auc, aucrf$auc, aucrfu$auc,   
 aucrfo$auc, aucrfb$auc))  
  
knitr::kable(summ3)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy | Sensitivity | Specificity | AUC |
| Q3. Decision Tree | 0.7263449 | 0.8945322 | 0.3109244 | 66.77268 |
| Q3. Random Forest | 0.7330912 | 0.9380316 | 0.2268908 | 70.40859 |
| Q3. Random Forest - Under | 0.6825809 | 0.7103281 | 0.6140456 | 70.62702 |
| Q3. Random Forest - Over | 0.7095658 | 0.8347509 | 0.4003601 | 69.24208 |
| Q3. Random Forest - Both | 0.6782564 | 0.7358445 | 0.5360144 | 68.51260 |

## Conclusion - Question 3

The Random Forest model with under sampled data has the largest AUC. However, the Random Forest model without any resampling has the highest accuracy and sensitivty and only marginally smaller AUC.

The Decision Tree model also has good accuracy, however its AUC is lower than the other models.

The Decision Tree model with re-sampling can be further explored

Of the models developed in this question, the Random Forest model without re-sampling is recommended.

# Question 4. Based on the models constructed, suggest the necessary recommendations from your side as an analyst, so that the company can better assess the creditworthiness of its future/potential customers (5 Marks)

### Summary of all models

summ <- rbind(summ1, summ2, summ3)  
knitr::kable(summ, digits = 3, caption = "Summary of models created in Questions 1, 2 and 3")

Summary of models created in Questions 1, 2 and 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy | Sensitivity | Specificity | AUC |
| Q1. Logistic - Base | 0.727 | 0.916 | 0.259 | 71.897 |
| Q1. Logistic - Base - Under | 0.332 | 0.314 | 0.374 | 71.604 |
| Q1. Logistic - Base - Over | 0.330 | 0.313 | 0.374 | 72.085 |
| Q1. Logistic - Base - Both | 0.331 | 0.313 | 0.376 | 72.127 |
| Q2. Alternate 1 | 0.732 | 0.916 | 0.278 | 72.178 |
| Q2. Alternate 2 | 0.732 | 0.917 | 0.274 | 72.171 |
| Q2. Alternate 2 - Under | 0.332 | 0.316 | 0.369 | 71.815 |
| Q2. Alternate 2 - Over | 0.332 | 0.317 | 0.369 | 72.334 |
| Q2. Alternate 2 - Both | 0.332 | 0.313 | 0.377 | 72.287 |
| Q3. Decision Tree | 0.726 | 0.895 | 0.311 | 66.773 |
| Q3. Random Forest | 0.733 | 0.938 | 0.227 | 70.409 |
| Q3. Random Forest - Under | 0.683 | 0.710 | 0.614 | 70.627 |
| Q3. Random Forest - Over | 0.710 | 0.835 | 0.400 | 69.242 |
| Q3. Random Forest - Both | 0.678 | 0.736 | 0.536 | 68.513 |

### Summary of all models - sorted by AUC descending

summ <- summ[with(summ, order(-AUC, -Accuracy, -Sensitivity, -Specificity)),]  
knitr::kable(summ, digits = 3, caption = "Summary of models created in Questions 1, 2 and 3, sorted by AUC")

Summary of models created in Questions 1, 2 and 3, sorted by AUC

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Model | Accuracy | Sensitivity | Specificity | AUC |
| 8 | Q2. Alternate 2 - Over | 0.332 | 0.317 | 0.369 | 72.334 |
| 9 | Q2. Alternate 2 - Both | 0.332 | 0.313 | 0.377 | 72.287 |
| 5 | Q2. Alternate 1 | 0.732 | 0.916 | 0.278 | 72.178 |
| 6 | Q2. Alternate 2 | 0.732 | 0.917 | 0.274 | 72.171 |
| 4 | Q1. Logistic - Base - Both | 0.331 | 0.313 | 0.376 | 72.127 |
| 3 | Q1. Logistic - Base - Over | 0.330 | 0.313 | 0.374 | 72.085 |
| 1 | Q1. Logistic - Base | 0.727 | 0.916 | 0.259 | 71.897 |
| 7 | Q2. Alternate 2 - Under | 0.332 | 0.316 | 0.369 | 71.815 |
| 2 | Q1. Logistic - Base - Under | 0.332 | 0.314 | 0.374 | 71.604 |
| 12 | Q3. Random Forest - Under | 0.683 | 0.710 | 0.614 | 70.627 |
| 11 | Q3. Random Forest | 0.733 | 0.938 | 0.227 | 70.409 |
| 13 | Q3. Random Forest - Over | 0.710 | 0.835 | 0.400 | 69.242 |
| 14 | Q3. Random Forest - Both | 0.678 | 0.736 | 0.536 | 68.513 |
| 10 | Q3. Decision Tree | 0.726 | 0.895 | 0.311 | 66.773 |

### Recommendations

* Based on AUC, the Alternative 1 or Alternative 2 Logistic Regression models developed in Question 2 are recommended.
* The Alternative 1 Logistic regression model uses AGE as a variable in addition to all the recommended variables.
* The Alternative 2 Logistic regression model uses AGE as an independent variable and uses all the recommended independent variables except MTHINCTH (monthly income in thousands).
* The alternative 2 Logistic regression model has similar performance characteristics as the alternative 1 model but is more parsimonious as it uses one less independent variable.
* Although the alternative model 2 with over and both sampling have marginally bigger AUC, their accuracy and sensitivity is far below the models without re-sampling.
* A neural network based model should be tried to see if it gives better model parameters.
* Random forest model has the highest accuracy and sensitivity, however it is computationally expensive and has less AUC.