Stat_Markdown

2023-06-19

Splitting into Training and Test Sets

To be able to build and evaluate a model that predicts if a Customer is likely to churn, we divide the dataset so that 75% of the data will be the *Training set* and the remaining 25% the *Test set*. These two new datasets will be used respectively to train our models and to evaluate the performances of the models. We will then concentrate on the properties of the training set, while the test set will remain "unknown" until the evaluation of the models.

```
set.seed(0237)

sample <- sample.split(log_cleaned_bank_data_withoutNA_quan[,2:20]$Attrition_Flag,
                       SplitRatio = 0.75)
train <- subset(log_cleaned_bank_data_withoutNA_quan[,2:20],sample == TRUE)
test <- subset(log_cleaned_bank_data_withoutNA_quan[,2:20],sample == FALSE)
```

We also check that the proportion of the response variable is approximately the same in the two new datasets.

```
#Proportion of Attrited and Existing Customer in Training set
prop.table(table(train$Attrition_Flag))
```

```
##
##        0        1
## 0.843169 0.156831
```

```
#Proportion of Attrited and Existing Customer in Test set
prop.table(table(test$Attrition_Flag))
```

```
##
##         0         1
## 0.8434466 0.1565534
```

It can be seen that the dataset is unbalanced. We decide then to create a balanced training set. By using this dataset the models will be able to represent better the Attrited Customers. To define this new Training set we utilized a mix of Oversampling and Undersampling to obtain a dataset with the same length of the original Training dataset.

```
# ovun.sample is a function form the "Rose" library
train_bal <- ovun.sample(Attrition_Flag~.,data = train, method = "both", p = 0.5,
                         N =4948)$data
```

We can see that this new Training set is indeed balanced

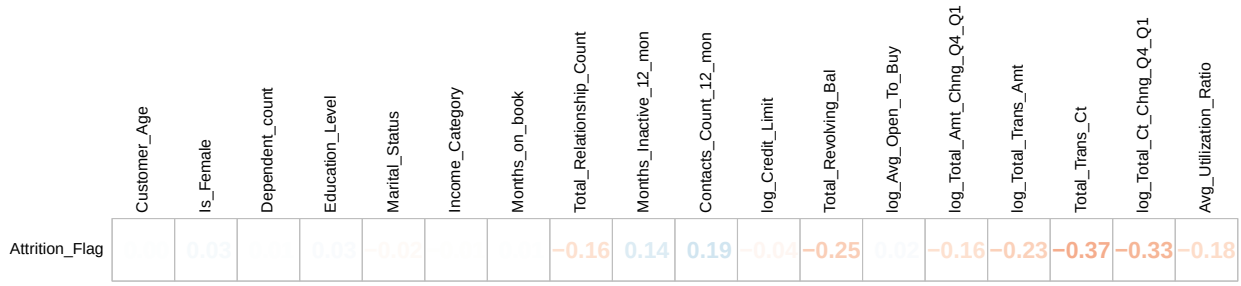
```
#Proportion of Attrited and Existing Customer
prop.table(table(train_bal$Attrition_Flag))
```

```
##
##         0         1
## 0.4931285 0.5068715
```

Correlations

We will now focus on the unbalanced training set and we will investigate the correlation between variables. We will then try to understand which features are connected to attrited customer. First of all we compute the correlations between the response variable and the predictor variables using the *corrplot* function, from the homonym library.

```
# Changing Attrition_Flag from factor to numeric
train$Attrition_Flag <- as.numeric(train$Attrition_Flag)

#Correlation matrix
cor_mat <- cor(train)

#Correlation with Attrition_Flag
corrplot(cor_mat[1,,drop=FALSE],method = "number",number.cex = 1.3,
         cl.pos = "n",tl.col = "black" ,tl.cex=0.9,diag = FALSE)
```

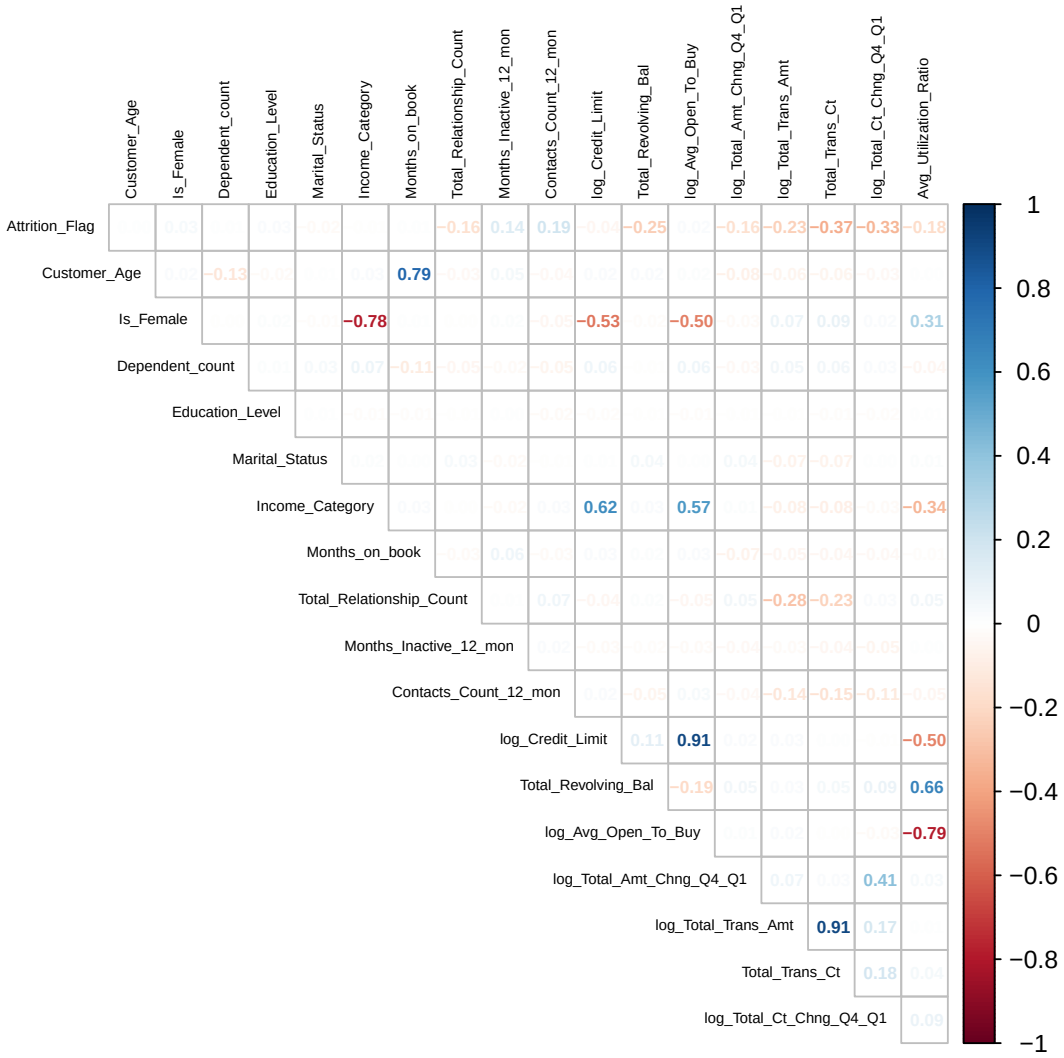


We report in the following table the highest values observed

Variable	Correlation with Attrition_Flag
Total_Trans_Ct	-0.37
log_Total_Ct_Chng_Q4_Q1	-0.33
Total_Revolving_Bal	-0.25
log_Total_trans_Amt	-0.23
Contacts_Count_12_mon	0.19
Avg_Utilization_ratio	-0.18
log_Total_Trans_Amt	-0.16
Total_Relantioship_Count	-0.16

Focusing now on the other correlations, we create the following figure

```
#Correlations
corrplot(cor_mat[1:19,1:19],method = "number",type = "upper",number.cex = 0.6,
         tl.pos = "td",tl.cex=0.5, tl.col = "black" ,diag = FALSE)
```



We then show the highest correlations in the following table

Variable 1	Variable 2	Correlation
Customer_Age	Months_on_book	0.79
Is_Female	Income_Category	-0.78
Is_Female	log_Credit_Limit	-0.53
Is_Female	log_Avg_Open_To_Buy	-0.50
Income_Category	log_Credit_Limit	0.62
Income_Category	log_Avg_Open_To_Buy	0.57
log_Credit_Limit	log_Avg_Open_To_Buy	0.91
log_Credit_Limit	Avg_Utilization_Ratio	-0.50
Total_Revolving_Bal	Avg_Utilization_Ratio	0.66
log_Avg_Open_To_Buy	Avg_Utilization_Ratio	-0.79

Variable 1	Variable 2	Correlation
log_Total_Amt_Chng_Q4_Q1	log_Total_Ct_Chng_Q4_Q1	0.41
log_Total_Trans_Amt	Total_Trans_Ct	0.91

We can notice that there are really high value of correlations. This might led to a case of collinearity, as we will see and discuss in the models' definition.

Partial Correlation

In the last figure, we analyzed the correlations between the variables in such a way that takes into account the relationships between the two variables involved in the correlation and the other variables present in the training set. To investigate the relationship between only the two variables we use the *correlation* function from the *correlation* library to calculate the partial correlations.

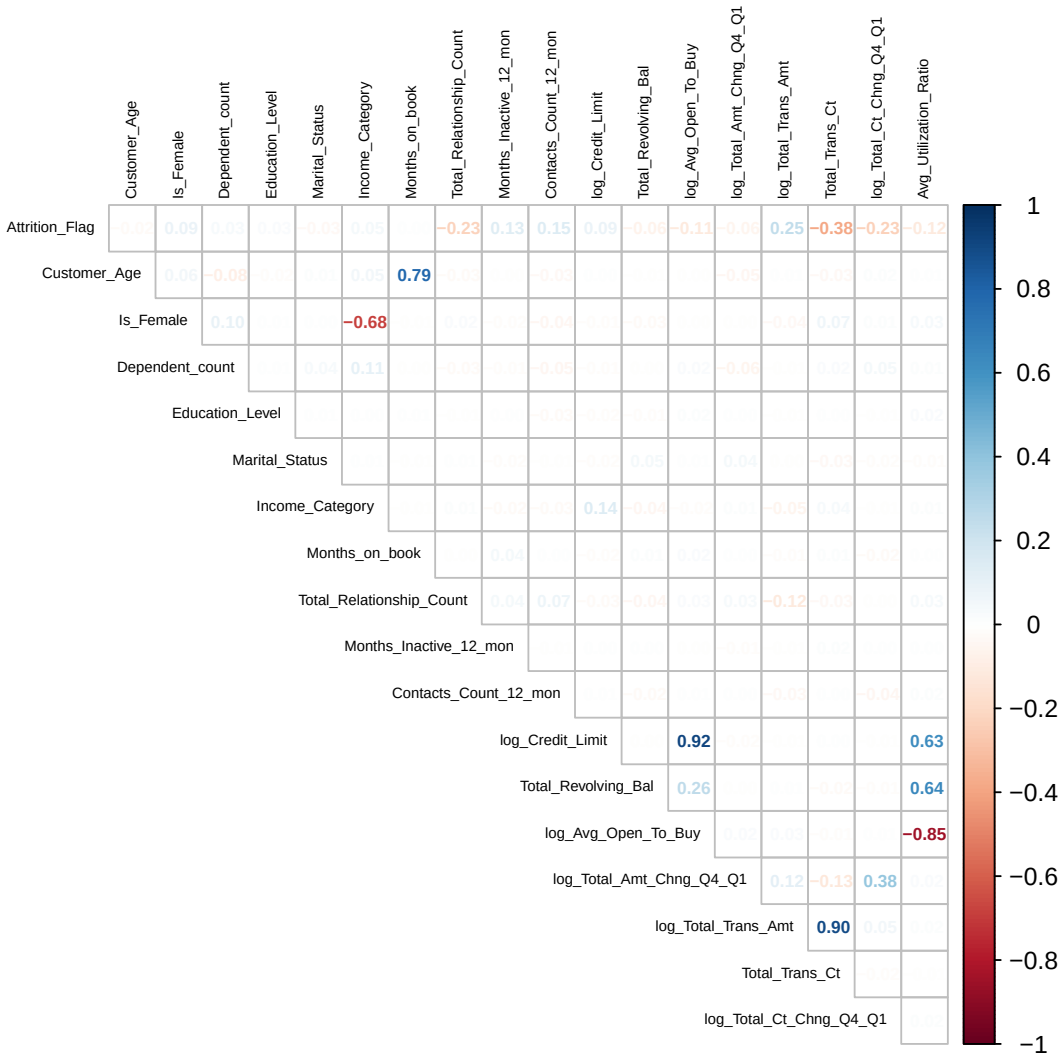
```
#Partial correlation (We will not show the output, since it takes too much space)
correlation(train,partial = TRUE)
```

For the response variable the most significant partial correlations are the following

Variable	Correlation with Attrition_Flag
Total_Trans_Ct	-0.38
log_Total_Trans_Amt	0.25
log_Total_Ct_Chng_Q4_Q1	-0.23
Total_Relantioship_Count	-0.23
Contacts_Count_12_mon	0.15
Months_Inactive_12_mon	0.13
Avg_Utilization_Ratio	-0.12
log_Avg_Open_To_Buy	-0.11

We also point out that many variables that had significant correlations have lower partial correlations values. Now we visualize the full results by using the function *corrplot* from the hononym library.

```
#Partial Correlation matrix
part_cor_mat <- pcor(train)$estimate
corrplot(part_cor_mat, method = "number",type = "upper",number.cex = 0.6,
         tl.pos = "td",tl.cex=0.5, tl.col = "black" ,diag = FALSE)
```

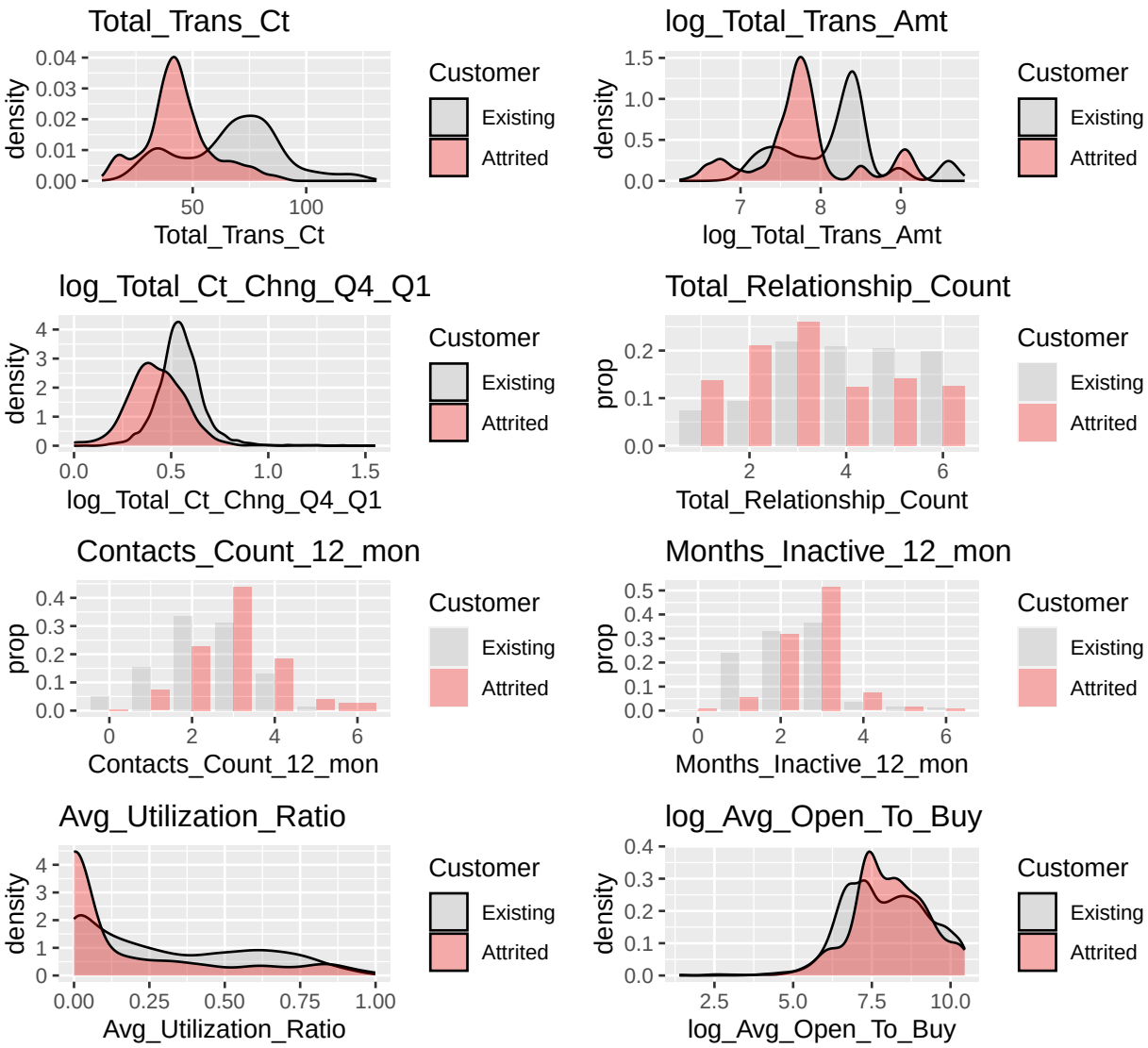


As done previously, we will show the highest partial correlations in a table:

Variable 1	Variable 2	Correlation
Customer_Age	Months_on_book	0.79
Is_Female	Income_Category	-0.68
log_Credit_Limit	log_Avg_Open_To_Buy	0.92
log_Credit_Limit	Avg_Utilization_Ratio	0.63
Total_Revolving_Bal	log_Avg_Open_To_Buy	0.26
Total_Revolving_Bal	Avg_Utilization_Ratio	0.64
log_Avg_Open_To_Buy	Avg_Utilization_Ratio	-0.85
log_Total_Amt_Chng_Q4_Q1	log_Total_Ct_Chng_Q4_Q1	0.38
log_Total_Trans_Amt	Total_Trans_Ct	0.90

Next, we will investigate the relationship between the response variable and the other variables. To do so we separate the most correlated variables into the two classes given by *Attrition_Flag* and we plot the results.

```r
Customer <- as.factor(train$Attrition_Flag)
a <- ggplot(train, aes(x = Total_Trans_Ct, fill = Customer)) +
  geom_density(alpha=0.3) + ggtitle("Total_Trans_Ct") +
  scale_fill_manual(values = c("darkgrey","red"),labels = c("Existing","Attrited"))

b <- ggplot(train, aes(x = log_Total_Trans_Amt, fill = Customer)) +
  geom_density(alpha=0.3) + ggtitle("log_Total_Trans_Amt") +
  scale_fill_manual(values = c("darkgrey","red"),labels = c("Existing","Attrited"))

c <- ggplot(train, aes(x = log_Total_Ct_Chng_Q4_Q1, fill = Customer)) +
  geom_density(alpha=0.3) + ggtitle("log_Total_Ct_Chng_Q4_Q1")+
  scale_fill_manual(values = c("darkgrey","red"),labels = c("Existing","Attrited"))

d <- ggplot(train, aes(x = Total_Relationship_Count, fill = Customer)) +
  geom_bar(aes(y = after_stat(prop) ),alpha=0.3, position = "dodge") +
  ggtitle("Total_Relationship_Count") +
  scale_fill_manual(values = c("darkgrey","red"),labels = c("Existing","Attrited"))

e <- ggplot(train, aes(x = Contacts_Count_12_mon, fill = Customer)) +
  geom_bar(aes(y = after_stat(prop) ),alpha=0.3, position = "dodge") +
  ggtitle("Contacts_Count_12_mon") +
  scale_fill_manual(values = c("darkgrey","red"),labels = c("Existing","Attrited"))

f <- ggplot(train, aes(x = Months_Inactive_12_mon, fill = Customer)) +
  geom_bar(aes(y = after_stat(prop) ),alpha=0.3, position = "dodge") +
  ggtitle("Months_Inactive_12_mon") +
  scale_fill_manual(values = c("darkgrey","red"),labels = c("Existing","Attrited"))

g <- ggplot(train, aes(x = Avg_Utilization_Ratio, fill = Customer)) +
  geom_density(alpha=0.3) + ggtitle("Avg_Utilization_Ratio") +
  scale_fill_manual(values = c("darkgrey","red"),labels = c("Existing","Attrited"))

h <- ggplot(train, aes(x = log_Avg_Open_To_Buy, fill = Customer)) +
  geom_density(alpha=0.3) + ggtitle("log_Avg_Open_To_Buy") +
  scale_fill_manual(values = c("darkgrey","red"),labels = c("Existing","Attrited"))


ggarrange(a,b,c,d,e,f,g,h,nrow=4,ncol=2)
```



From these plots we can notice that a customer is more likely to churn if:

- In the last year he didn't make many transactions or the total amount of the transactions is low.
- In the 1st quarter he made less transactions in comparison to the last quarter.
- He holds a small amount of the bank's products.
- In the last year he contacted many times the bank.
- In the last year he has been inactive for many months.
- He didn't use much his card.

For *log_Avg_Open_To_Buy* we notice that there is a difference between the classes of Attriting and Existing Customer but it's not enough to infer a clear relationship between Attriting customers and the variable analyzed.

Model Definition

We are interested in finding the customers who are likely to churn, without penalizing too much the other customers. To do so we are going to define different models and compare how well they can identify Attriting customers. We are then searching for models with high values of *Recall*, which tells us how many Attriting customers there are, and *Precision*, which tells us how many of our predictions did churn. Because of these choices, we select the *F1 score* as our principal performance's test since it is the Harmonic mean of *Precision* and *Recall*.

To define the models we will use three different techniques: *Stepwise selection*, *Discriminant analysis* and *Regularized Regression* and for each of these we will compare the models obtained from the unbalanced and the balanced training set.

Simple logistic regression

As the first model we try the binomial generalized linear model, also known as *logistic regression*. It is a modelling technique used to solve binary classification problems by estimating the probability of an event happening based on the value of the predictor variables. In our analysis we will are trying to guess the value of the binary response variable *Attrition_Flag*, in other words we are estimating the probability of a customer's attrition. We will initially define the model containing all the predictor variables and, after addressing the problem of *Collinearity* we choose a subset of predictor variables that gives us a better model through *stepwise selection*.

Unbalanced dataset

We will start by defining the complete model and assess its performance.

```
glm_1 <- glm(data = train,Attrition_Flag~ .,family = "binomial")
summary(glm_1)
```

```
##
## Call:
## glm(formula = Attrition_Flag ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6836  -0.3420  -0.1498  -0.0530   3.5101
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -1.622e+01  1.710e+00  -9.486  < 2e-16 ***
## Customer_Age             -9.797e-03  1.179e-02  -0.831 0.405839
## Is_Female                 9.477e-01  1.866e-01   5.079 3.79e-07 ***
## Dependent_count           7.712e-02  4.477e-02   1.723 0.084953 .
## Education_Level           6.899e-02  4.049e-02   1.704 0.088448 .
## Marital_Status           -2.060e-01  9.836e-02  -2.094 0.036265 *
## Income_Category           2.060e-01  7.303e-02   2.821 0.004794 **
## Months_on_book           -5.879e-03  1.172e-02  -0.502 0.615829
## Total_Relationship_Count -5.269e-01  4.074e-02 -12.932  < 2e-16 ***
## Months_Inactive_12_mon    4.977e-01  5.815e-02   8.559  < 2e-16 ***
## Contacts_Count_12_mon     4.735e-01  5.376e-02   8.808  < 2e-16 ***
## log_Credit_Limit          1.312e+00  2.804e-01   4.679 2.89e-06 ***
## Total_Revolving_Bal      -4.389e-04  1.263e-04  -3.476 0.000509 ***
## log_Avg_Open_To_Buy      -1.598e+00  2.773e-01  -5.764 8.22e-09 ***
```

```
## log_Total_Amt_Chng_Q4_Q1 -1.807e+00  5.126e-01  -3.526 0.000422 ***
## log_Total_Trans_Amt       3.700e+00  2.309e-01  16.024  < 2e-16 ***
## Total_Trans_Ct           -1.649e-01  7.688e-03 -21.455  < 2e-16 ***
## log_Total_Ct_Chng_Q4_Q1  -5.038e+00  4.755e-01 -10.596  < 2e-16 ***
## Avg_Utilization_Ratio    -4.603e+00  7.748e-01  -5.941 2.83e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4298.6  on 4947  degrees of freedom
## Residual deviance: 2131.4  on 4929  degrees of freedom
## AIC: 2169.4
##
## Number of Fisher Scoring iterations: 7
```

We take $1/n$, $n = 1, ..., 10$ as different thresholds for testing the *F1 score* and we take the one that maximize it. Here we show the *F1 score* and other significant accuracy's tests obtained for that value of the thresold. As you can expect from an unbalanced dataset we have that the specificity is close to 1 while the recall is much lower.

```
#Threshold
Threshold <- 0.4
pred_glm_i <- predict(glm_1,test,type="response")
pred_i <- ifelse(pred_glm_i >= Threshold , 1,0)

#Confusion matrix

c_mat_i <- table(test$Attrition_Flag,pred_i)
```

	Predicted Values		
Real Values	0	1	Total
0	1331	59	1390
1	82	176	258
Total	1413	235	1648

```
#Accuracy

mean(pred_i==test$Attrition_Flag)*100
```

```
## [1] 91.44417
```

```
#True Negative Rate / Specificity

Spec_i <- c_mat_i[1,1]/sum(c_mat_i[1,])
Spec_i
```

```
## [1] 0.957554
```

```
#Precision / Positive Predicted Value

Prec_i <- c_mat_i[2,2]/sum(c_mat_i[,2])
Prec_i
```

```
## [1] 0.7489362
```

```
#Recall / True Positive Rate / Sensitivity

Rec_i <- c_mat_i[2,2]/sum(c_mat_i[2,])
Rec_i
```

```
## [1] 0.6821705
```

```
#F1 Score

F1_i <- 2 * (Prec_i * Rec_i)/(Prec_i + Rec_i)
F1_i
```

```
## [1] 0.7139959
```

Before starting with the stepwise selection we check for collinearity by checking the *Variance Inflation Function* of the predictor variables.

```
vif(glm_1)
```

```
##              Customer_Age                Is_Female          Dependent_count
##                  2.990913                 2.749986                 1.058424
##           Education_Level           Marital_Status          Income_Category
##                  1.005775                 1.039265                 3.281569
##            Months_on_book Total_Relationship_Count   Months_Inactive_12_mon
##                  2.989025                 1.147507                 1.043349
##     Contacts_Count_12_mon         log_Credit_Limit      Total_Revolving_Bal
##                  1.040355                20.355535                 3.734824
##       log_Avg_Open_To_Buy log_Total_Amt_Chng_Q4_Q1      log_Total_Trans_Amt
##                 38.455876                 1.236129                 6.477327
##            Total_Trans_Ct  log_Total_Ct_Chng_Q4_Q1    Avg_Utilization_Ratio
##                  6.582049                 1.256063                14.777916
```

We can notice that there are a lot of values much higher than 1. That means that there is a problem of collinearity which happens because there are variables highly correlated. Collinearity could be a problem in the model definition because it makes it difficult to determine the effects of the highly correlated variables on the response variable. To mitigate this effect we remove the indipendent variable *log_Avg_Open_To_Buy*. We also remove *Months_on_book* since, from the summary, it seems that it is not significant and it would be removed with the stepwise selection.

```
glm_2 <- update(glm_1, . ~ . - log_Avg_Open_To_Buy - Months_on_book)

vif(glm_2)
```

```
##              Customer_Age                Is_Female          Dependent_count
##                  1.056917                 2.746313                 1.057680
##           Education_Level           Marital_Status          Income_Category
##                  1.004594                 1.039585                 3.249190
## Total_Relationship_Count   Months_Inactive_12_mon    Contacts_Count_12_mon
##                  1.148948                 1.034207                 1.040338
##         log_Credit_Limit      Total_Revolving_Bal log_Total_Amt_Chng_Q4_Q1
##                  2.602631                 3.376016                 1.236455
##      log_Total_Trans_Amt           Total_Trans_Ct  log_Total_Ct_Chng_Q4_Q1
##                  6.485363                 6.576721                 1.253018
##    Avg_Utilization_Ratio
##                  3.943486
```

We notice that the values of the *VIF* are smaller and they are at least smaller than 10. We decide to keep the other variables even if there are two values bigger than 5, because they are statistically significant in face of that collinearity. In fact removing them leads to a much worse performance of the model.

Stepwise selection

Stepwise selection consists in a refinement of the model by iteratively removing (Backward selection) or adding (forward selection) variables based on their significance to the model's performance. By using this method we aim to find the most important variables and interactions, so that the model can fit accurately the data without being too complex. We will show the final choice of variables and interactions based on the p-value and the *Akaike information criterion* (AIC), which is an estimator of the quality of the model on the training set. Moreover AIC penalize the models with a big number of estimated parameters, reducing the possibility of overfitting.

```
glm_3 <- update(glm_2, . ~ . + log_Total_Amt_Chng_Q4_Q1*Customer_Age
                + log_Total_Amt_Chng_Q4_Q1*Dependent_count
                + log_Total_Amt_Chng_Q4_Q1*log_Total_Trans_Amt
                + log_Total_Amt_Chng_Q4_Q1*Total_Trans_Ct)

glm_4 <- update(glm_3, . ~ . + Total_Revolving_Bal*log_Credit_Limit
                + Total_Revolving_Bal*Avg_Utilization_Ratio)

glm_5 <- update(glm_4, . ~ . + log_Credit_Limit*Avg_Utilization_Ratio)

glm_6 <- update(glm_5, . ~ . + log_Total_Ct_Chng_Q4_Q1*Is_Female)

glm_7 <- update(glm_6, . ~ . - Education_Level)

glm_8 <- update(glm_7, . ~ .+ Customer_Age*Marital_Status)
```

```
summary(glm_8)
```

```
##
## Call:
## glm(formula = Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count +
##     Marital_Status + Income_Category + Total_Relationship_Count +
##     Months_Inactive_12_mon + Contacts_Count_12_mon + log_Credit_Limit +
##     Total_Revolving_Bal + log_Total_Amt_Chng_Q4_Q1 + log_Total_Trans_Amt +
##     Total_Trans_Ct + log_Total_Ct_Chng_Q4_Q1 + Avg_Utilization_Ratio +
##     Customer_Age:log_Total_Amt_Chng_Q4_Q1 + Dependent_count:log_Total_Amt_Chng_Q4_Q1 +
##     log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt + log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct +
```

```
##     log_Credit_Limit:Total_Revolving_Bal + Total_Revolving_Bal:Avg_Utilization_Ratio +
##     log_Credit_Limit:Avg_Utilization_Ratio + Is_Female:log_Total_Ct_Chng_Q4_Q1 +
##     Customer_Age:Marital_Status, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8089  -0.2957  -0.1139  -0.0356   3.3026
##
## Coefficients:
##                                             Estimate Std. Error z value
## (Intercept)                                2.256e+01  7.570e+00   2.980
## Customer_Age                              -9.990e-02  3.719e-02  -2.686
## Is_Female                                  2.859e+00  5.006e-01   5.711
## Dependent_count                            4.706e-01  2.217e-01   2.123
## Marital_Status                             1.519e+00  6.044e-01   2.514
## Income_Category                            2.674e-01  7.744e-02   3.453
## Total_Relationship_Count                  -5.165e-01  4.305e-02 -11.997
## Months_Inactive_12_mon                     5.284e-01  6.216e-02   8.501
## Contacts_Count_12_mon                      4.945e-01  5.701e-02   8.674
## log_Credit_Limit                          -2.431e-01  1.060e-01  -2.294
## Total_Revolving_Bal                        4.002e-02  7.016e-03   5.704
## log_Total_Amt_Chng_Q4_Q1                  -7.860e+01  1.330e+01  -5.911
## log_Total_Trans_Amt                       -1.264e+00  1.076e+00  -1.174
## Total_Trans_Ct                            -1.194e-01  3.442e-02  -3.469
## log_Total_Ct_Chng_Q4_Q1                   -3.894e+00  6.150e-01  -6.332
## Avg_Utilization_Ratio                      1.621e+02  2.426e+01   6.680
## Customer_Age:log_Total_Amt_Chng_Q4_Q1      2.656e-01  5.853e-02   4.538
## Dependent_count:log_Total_Amt_Chng_Q4_Q1  -7.942e-01  4.006e-01  -1.983
## log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt  9.041e+00  1.906e+00   4.743
## log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct   -8.906e-02  6.050e-02  -1.472
## log_Credit_Limit:Total_Revolving_Bal      -3.718e-03  6.507e-04  -5.714
## Total_Revolving_Bal:Avg_Utilization_Ratio   5.499e-03  4.575e-04  12.018
## log_Credit_Limit:Avg_Utilization_Ratio    -2.593e+01  3.779e+00  -6.862
## Is_Female:log_Total_Ct_Chng_Q4_Q1         -3.737e+00  9.509e-01  -3.929
## Customer_Age:Marital_Status               -3.695e-02  1.294e-02  -2.856
##                                           Pr(>|z|)
## (Intercept)                               0.002883 **
## Customer_Age                              0.007228 **
## Is_Female                                 1.12e-08 ***
## Dependent_count                           0.033790 *
## Marital_Status                            0.011947 *
## Income_Category                           0.000554 ***
## Total_Relationship_Count                   < 2e-16 ***
## Months_Inactive_12_mon                    < 2e-16 ***
## Contacts_Count_12_mon                     < 2e-16 ***
## log_Credit_Limit                          0.021815 *
## Total_Revolving_Bal                       1.17e-08 ***
## log_Total_Amt_Chng_Q4_Q1                  3.40e-09 ***
## log_Total_Trans_Amt                       0.240407
## Total_Trans_Ct                            0.000522 ***
## log_Total_Ct_Chng_Q4_Q1                   2.42e-10 ***
## Avg_Utilization_Ratio                     2.40e-11 ***
## Customer_Age:log_Total_Amt_Chng_Q4_Q1     5.68e-06 ***
## Dependent_count:log_Total_Amt_Chng_Q4_Q1  0.047421 *
```

```
## log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt 2.10e-06 ***
## log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct      0.141017
## log_Credit_Limit:Total_Revolving_Bal         1.10e-08 ***
## Total_Revolving_Bal:Avg_Utilization_Ratio     < 2e-16 ***
## log_Credit_Limit:Avg_Utilization_Ratio       6.79e-12 ***
## Is_Female:log_Total_Ct_Chng_Q4_Q1            8.52e-05 ***
## Customer_Age:Marital_Status                  0.004293 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4298.6  on 4947  degrees of freedom
## Residual deviance: 1899.0  on 4923  degrees of freedom
## AIC: 1949
##
## Number of Fisher Scoring iterations: 7
```

We decided to not remove *log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct* since removing it actually increases the *AIC*. Instead, we decided to keep *log_Total_Trans_Amt* since it's been used in a interaction.

We now check the AIC values. We can notice that it gradually gets smaller except in the step where we removed *log_Avg_Open_To_Buy* to reduce the collinearity.

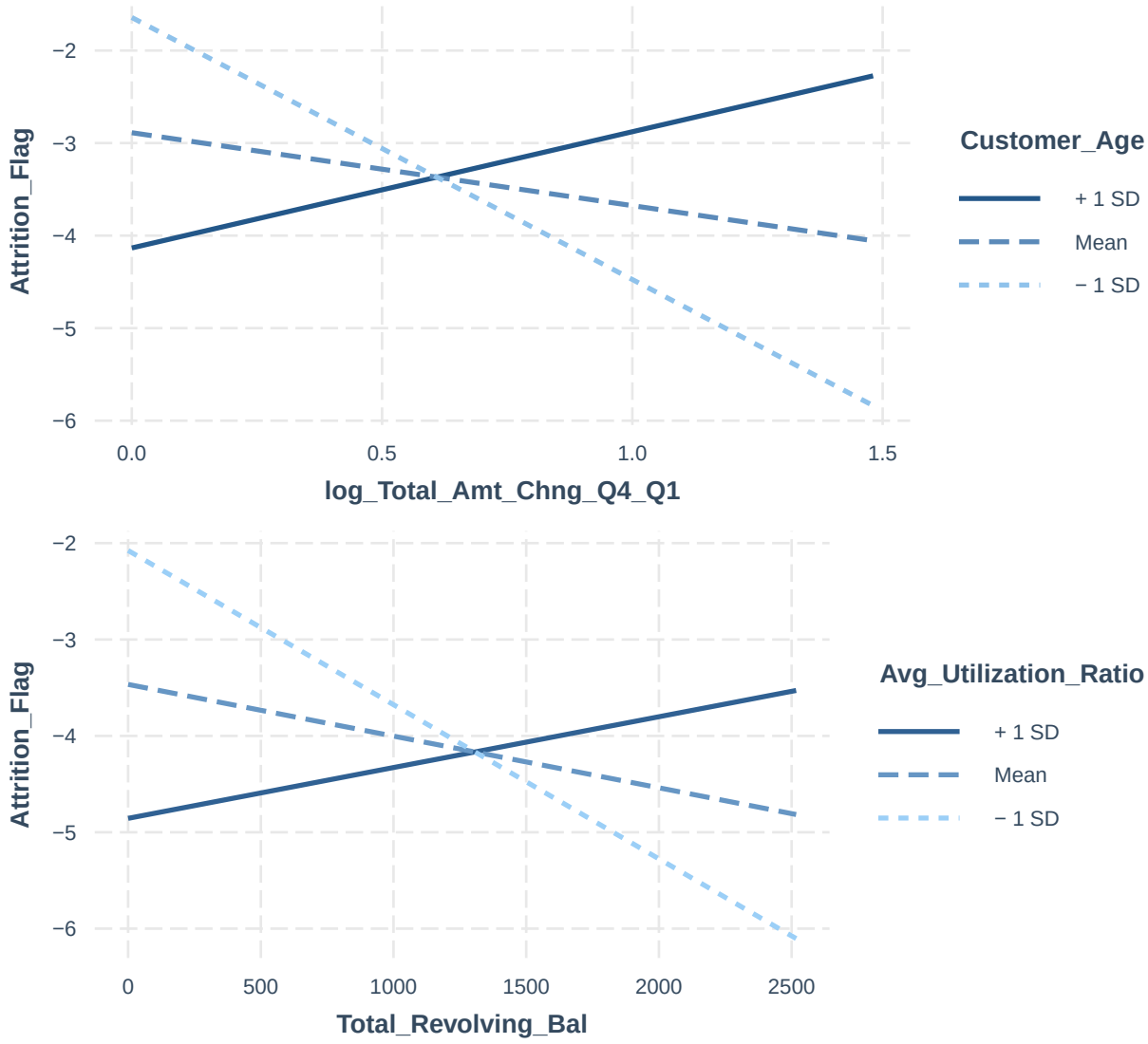
```
AIC(glm_1,glm_2,glm_3,glm_4,glm_5,glm_6,glm_7,glm_8)
```

```
##       df      AIC
## glm_1 19 2169.352
## glm_2 17 2200.458
## glm_3 21 2135.898
## glm_4 23 2016.077
## glm_5 24 1969.282
## glm_6 25 1955.137
## glm_7 24 1955.125
## glm_8 25 1948.950
```

We can see from the following figures the effect that one variable has on the response variable based on different values of the other variable of the interaction. We plotted the results on the models before applying the logit function, so that it's easier to verify the interactions by seeing the slopes of the lines. We decided to show only two of these plots since they are all pretty similar and we didn't want to use too much space.

```
i1 <- interact_plot(glm_3,pred = log_Total_Amt_Chng_Q4_Q1,modx = Customer_Age,
                    outcome.scale = "link")
i2 <- interact_plot(glm_3,pred = log_Total_Amt_Chng_Q4_Q1,modx = Dependent_count,
                    outcome.scale = "link")
i3 <- interact_plot(glm_3,pred = log_Total_Amt_Chng_Q4_Q1,modx = log_Total_Trans_Amt,
                    outcome.scale = "link")
i4 <- interact_plot(glm_3,pred = log_Total_Amt_Chng_Q4_Q1,modx = Total_Trans_Ct,
                    outcome.scale = "link")
i5 <- interact_plot(glm_4,pred = Total_Revolving_Bal,modx = log_Credit_Limit,
                    outcome.scale = "link")
i6 <- interact_plot(glm_4,pred = Total_Revolving_Bal,modx = Avg_Utilization_Ratio,
                    outcome.scale = "link")
```

```
i7 <- interact_plot(glm_5,pred = log_Credit_Limit,modx = Avg_Utilization_Ratio,
                    outcome.scale = "link")
i8 <- interact_plot(glm_6,pred = log_Total_Ct_Chng_Q4_Q1,modx = Is_Female,
                    outcome.scale = "link")
i9 <- interact_plot(glm_8,pred = Customer_Age,modx = Marital_Status ,
                    outcome.scale = "link")

ggarrange(i1,i6,nrow=2,ncol=1)
```



As before, we check the best *F1 score* and the other significant values. We can notice that *glm_8* performs better than *glm_1* for every accuracy's test that we considered.

```
#Threshold
Threshold <- 0.4
pred_glm_f <- predict(glm_8,test,type="response")
pred_f <- ifelse(pred_glm_f >= Threshold , 1,0)
```

```r
#Confusion matrix
c_mat_f <- table(test$Attrition_Flag,pred_f)
```

	Predicted Values		
Real Values	0	1	Total
0	1333	57	1390
1	72	186	258
Total	1405	243	1648

```r
#Accuracy

mean(pred_f==test$Attrition_Flag)*100
```

```
## [1] 92.17233
```

```r
#True Negative Rate / Specificity

Spec_f <- c_mat_f[1,1]/sum(c_mat_f[1,])
Spec_f
```

```
## [1] 0.9589928
```

```r
#Precision / Positive Predicted Value

Prec_f <- c_mat_f[2,2]/sum(c_mat_f[,2])
Prec_f
```

```
## [1] 0.7654321
```

```r
#Recall / True Positive Rate / Sensitivity

Rec_f <- c_mat_f[2,2]/sum(c_mat_f[2,])
Rec_f
```

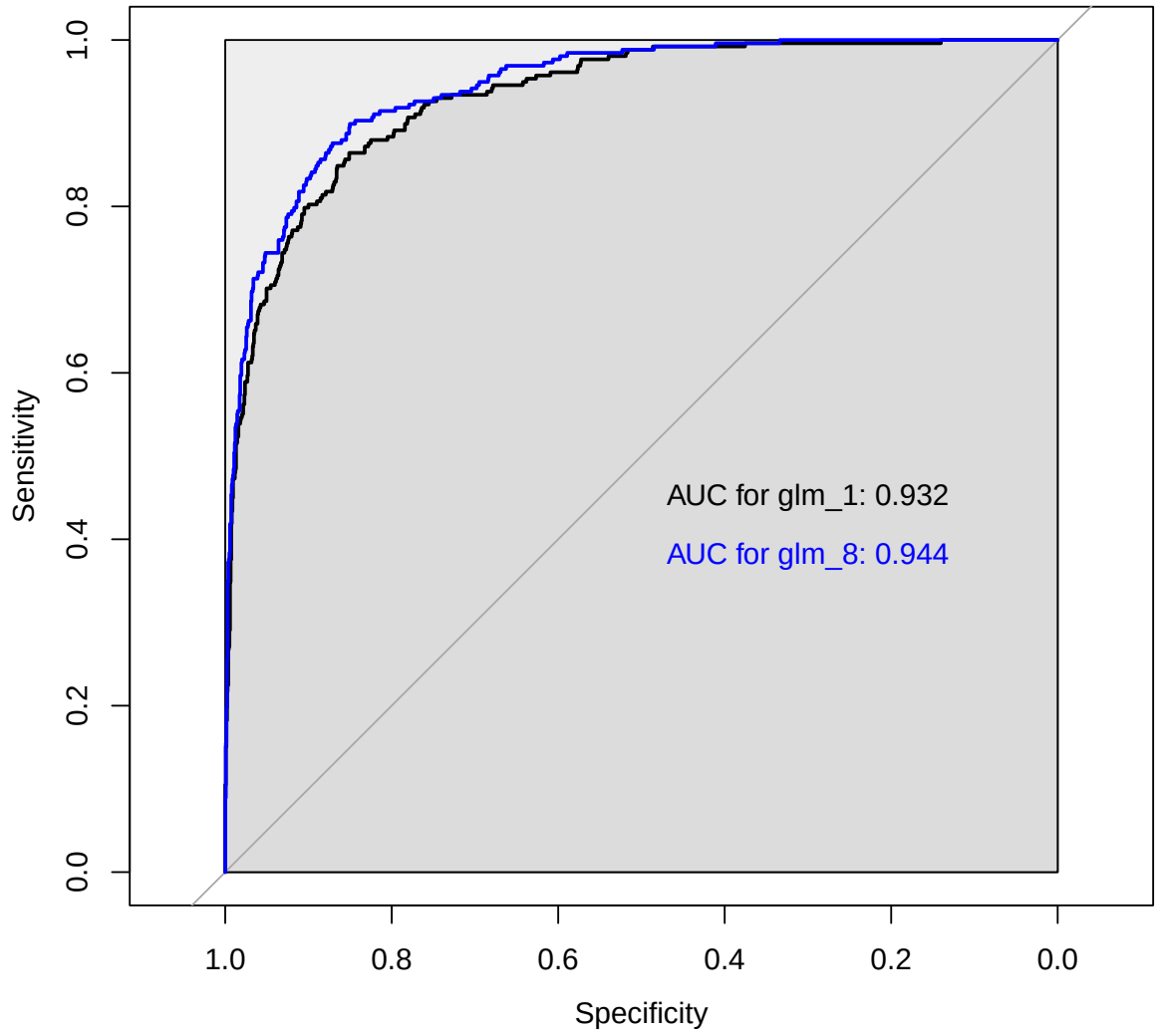
```
## [1] 0.7209302
```

```r
#F1 Score

F1_f <- 2 * (Prec_f * Rec_f)/(Prec_f + Rec_f)
F1_f
```

```
## [1] 0.742515
```

We will now compute the ROC curve and the corrisponding AUC values of the initial and final models. The ROC curve is obtained by plotting the *Sensitivity* against the *Specificity* for various thresholds. The ROC curve and the corrispondent *Area under the curve* (AUC) are used to evaluate the performance of the models. In fact a ROC curve which is close to the top-left corner, or equivalently a AUC value close to 1, will indicate that the model predicts accurately the positive and the negative istances. We can see that the final model has a higher AUC than the inital one. This, in conjunction with lower AIC and higher F1 score tells us that *glm_8* is a better choice then *glm_1*

```r
#ROC curves
roc_i <- roc(test$Attrition_Flag ~ pred_glm_i)
roc_f <- roc(test$Attrition_Flag ~ pred_glm_f)


plot(roc_i, col = "black",print.auc = FALSE, auc.polygon = TRUE,
     max.auc.polygon = TRUE, lwd=2)
plot(roc_f,add = TRUE,col = "blue", print.auc = FALSE, lwd=2)
text(0.3, 0.45, paste("AUC for glm_1:", round(roc_i$auc, 3)), col = "black")
text(0.3, 0.38, paste("AUC for glm_8:", round(roc_f$auc, 3)), col = "blue")
```



Balanced dataset

We now define a logistic model trained on a balanced dataset. We repeat the same procedure we used for the unbalanced dataset. We initially analyze the complete model and we check its performance

```r
glm_1_bal <- glm(data = train_bal,Attrition_Flag~ .,family = "binomial")
summary(glm_1_bal)
```

```
##
## Call:
## glm(formula = Attrition_Flag ~ ., family = "binomial", data = train_bal)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2500  -0.4374   0.0476   0.4535   2.8582
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -1.540e+01  1.320e+00 -11.671  < 2e-16 ***
## Customer_Age              -1.317e-02  9.006e-03  -1.462  0.14362
## Is_Female                  8.479e-01  1.484e-01   5.712 1.12e-08 ***
## Dependent_count            5.842e-02  3.533e-02   1.654  0.09822 .
## Education_Level            5.588e-02  3.172e-02   1.762  0.07807 .
## Marital_Status            -2.364e-01  7.521e-02  -3.144  0.00167 **
## Income_Category            1.011e-01  5.709e-02   1.770  0.07668 .
## Months_on_book            -1.160e-02  9.173e-03  -1.265  0.20590
## Total_Relationship_Count -4.486e-01  3.053e-02 -14.693  < 2e-16 ***
## Months_Inactive_12_mon     5.257e-01  4.817e-02  10.911  < 2e-16 ***
## Contacts_Count_12_mon      4.052e-01  4.294e-02   9.435  < 2e-16 ***
## log_Credit_Limit           1.197e+00  2.691e-01   4.447 8.70e-06 ***
## Total_Revolving_Bal       -2.994e-04  9.276e-05  -3.228  0.00125 **
## log_Avg_Open_To_Buy       -1.429e+00  2.712e-01  -5.270 1.37e-07 ***
## log_Total_Amt_Chng_Q4_Q1 -2.211e+00  4.245e-01  -5.209 1.90e-07 ***
## log_Total_Trans_Amt        3.863e+00  1.777e-01  21.741  < 2e-16 ***
## Total_Trans_Ct            -1.753e-01  6.243e-03 -28.089  < 2e-16 ***
## log_Total_Ct_Chng_Q4_Q1  -4.200e+00  3.618e-01 -11.609  < 2e-16 ***
## Avg_Utilization_Ratio     -4.492e+00  6.870e-01  -6.539 6.21e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6858.4  on 4947  degrees of freedom
## Residual deviance: 3271.9  on 4929  degrees of freedom
## AIC: 3309.9
##
## Number of Fisher Scoring iterations: 6
```

By comparing the *F1 score* we found that the best threshold is 0.8. Considering equal thresholds, it has a better *Recall* than the unbalanced one since it can represent better the Attriting customers. However it also has much worse Precision and this leads to an overall worse performance, as you can see from the *F1 score*.

```r
#Threshold
Threshold <- 0.8
pred_glm_i_bal <- predict(glm_1_bal,test,type="response")
pred_i_bal <- ifelse(pred_glm_i_bal >= Threshold , 1,0)

#Confusion matrix
```

```
c_mat_i_bal <- table(test$Attrition_Flag,pred_i_bal)
```

Predicted Values			
Real Values	0	1	Total
0	1331	59	1390
1	88	170	258
Total	1419	229	1648

```
#Accuracy

mean(pred_i_bal==test$Attrition_Flag)*100
```

```
## [1] 91.0801
```

```
#True Negative Rate / Specificity

Spec_i_bal <- c_mat_i_bal[1,1]/sum(c_mat_i_bal[1,])
Spec_i_bal
```

```
## [1] 0.957554
```

```
#Precision / Positive Predicted Value

Prec_i_bal <- c_mat_i_bal[2,2]/sum(c_mat_i_bal[,2])
Prec_i_bal
```

```
## [1] 0.7423581
```

```
#Recall / True Positive Rate / Sensitivity

Rec_i_bal <- c_mat_i_bal[2,2]/sum(c_mat_i_bal[2,])
Rec_i_bal
```

```
## [1] 0.6589147
```

```
#F1 Score

F1_i_bal <- 2 * (Prec_i_bal * Rec_i_bal)/(Prec_i_bal + Rec_i_bal)
F1_i_bal
```

```
## [1] 0.698152
```

We check for collinearity and, as for the unbalanced case, we decide to remove the variable *log_Avg_Open_To_Buy* and *Months_on_book*.

```
vif(glm_1_bal)
```

```
##              Customer_Age                Is_Female           Dependent_count
##                  2.733832                 2.820076                 1.093106
##           Education_Level           Marital_Status           Income_Category
##                  1.017500                 1.035145                 3.226073
##            Months_on_book Total_Relationship_Count    Months_Inactive_12_mon
##                  2.757685                 1.143847                 1.066254
##     Contacts_Count_12_mon          log_Credit_Limit       Total_Revolving_Bal
##                  1.050294                29.896922                 3.688831
##       log_Avg_Open_To_Buy log_Total_Amt_Chng_Q4_Q1       log_Total_Trans_Amt
##                 58.073721                 1.266048                 6.626061
##            Total_Trans_Ct  log_Total_Ct_Chng_Q4_Q1     Avg_Utilization_Ratio
##                  6.836700                 1.214635                19.393708
```

```
glm_2_bal <- update(glm_1_bal, . ~ . - log_Avg_Open_To_Buy - Months_on_book)

vif(glm_2_bal)
```

```
##              Customer_Age                Is_Female           Dependent_count
##                  1.093209                 2.810224                 1.088551
##           Education_Level           Marital_Status           Income_Category
##                  1.015237                 1.029246                 3.202816
## Total_Relationship_Count    Months_Inactive_12_mon     Contacts_Count_12_mon
##                  1.141156                 1.045824                 1.047354
##          log_Credit_Limit       Total_Revolving_Bal log_Total_Amt_Chng_Q4_Q1
##                  2.590618                 3.286344                 1.265429
##       log_Total_Trans_Amt            Total_Trans_Ct  log_Total_Ct_Chng_Q4_Q1
##                  6.616615                 6.811076                 1.214975
##     Avg_Utilization_Ratio
##                  3.930422
```

Stepwise selection

We performed stepwise selection by comparing AIC values and p-values. We report the principal steps. In the final model we decided to keep *Education_level* even if not highly significant. We made this choice because removing it increase the value of the AIC.

```
glm_3_bal <- update(glm_2_bal, . ~ . + log_Total_Amt_Chng_Q4_Q1*Customer_Age +
                      log_Total_Amt_Chng_Q4_Q1*Dependent_count +
                      log_Total_Amt_Chng_Q4_Q1*log_Total_Trans_Amt)
glm_4_bal <- update(glm_3_bal, . ~ . + Total_Revolving_Bal*log_Credit_Limit +
                      Total_Revolving_Bal*Avg_Utilization_Ratio)
glm_5_bal <- update(glm_4_bal, . ~ . + log_Credit_Limit*Avg_Utilization_Ratio)
glm_6_bal <- update(glm_5_bal, . ~ . + log_Total_Ct_Chng_Q4_Q1*Is_Female)
glm_7_bal <- update(glm_6_bal, . ~ . + Customer_Age*Marital_Status)
```

```
summary(glm_7_bal)
```

```
##
## Call:
```

```
## glm(formula = Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count +
##     Education_Level + Marital_Status + Income_Category + Total_Relationship_Count +
##     Months_Inactive_12_mon + Contacts_Count_12_mon + log_Credit_Limit +
##     Total_Revolving_Bal + log_Total_Amt_Chng_Q4_Q1 + log_Total_Trans_Amt +
##     Total_Trans_Ct + log_Total_Ct_Chng_Q4_Q1 + Avg_Utilization_Ratio +
##     Customer_Age:log_Total_Amt_Chng_Q4_Q1 + Dependent_count:log_Total_Amt_Chng_Q4_Q1 +
##     log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt + log_Credit_Limit:Total_Revolving_Bal +
##     Total_Revolving_Bal:Avg_Utilization_Ratio + log_Credit_Limit:Avg_Utilization_Ratio +
##     Is_Female:log_Total_Ct_Chng_Q4_Q1 + Customer_Age:Marital_Status,
##     family = "binomial", data = train_bal)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2472  -0.3429   0.0162   0.3825   2.6608
##
## Coefficients:
##                                              Estimate Std. Error z value
## (Intercept)                                 2.572e+01  4.602e+00   5.589
## Customer_Age                               -9.638e-02  3.148e-02  -3.061
## Is_Female                                   2.767e+00  4.176e-01   6.625
## Dependent_count                             6.970e-01  1.993e-01   3.497
## Education_Level                             6.608e-02  3.443e-02   1.919
## Marital_Status                              1.442e+00  4.648e-01   3.102
## Income_Category                             1.714e-01  6.319e-02   2.712
## Total_Relationship_Count                   -4.489e-01  3.323e-02 -13.511
## Months_Inactive_12_mon                      5.821e-01  5.200e-02  11.194
## Contacts_Count_12_mon                       4.130e-01  4.613e-02   8.952
## log_Credit_Limit                           -2.190e-01  8.824e-02  -2.482
## Total_Revolving_Bal                         3.959e-02  5.238e-03   7.558
## log_Total_Amt_Chng_Q4_Q1                   -7.911e+01  7.631e+00 -10.367
## log_Total_Trans_Amt                        -1.302e+00  5.258e-01  -2.477
## Total_Trans_Ct                             -1.732e-01  6.577e-03 -26.333
## log_Total_Ct_Chng_Q4_Q1                    -2.894e+00  4.655e-01  -6.218
## Avg_Utilization_Ratio                       1.659e+02  1.838e+01   9.028
## Customer_Age:log_Total_Amt_Chng_Q4_Q1       2.407e-01  4.999e-02   4.816
## Dependent_count:log_Total_Amt_Chng_Q4_Q1   -1.169e+00  3.496e-01  -3.344
## log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt  8.799e+00  8.489e-01  10.365
## log_Credit_Limit:Total_Revolving_Bal        -3.657e-03  4.854e-04  -7.534
## Total_Revolving_Bal:Avg_Utilization_Ratio    5.513e-03  3.566e-04  15.460
## log_Credit_Limit:Avg_Utilization_Ratio      -2.645e+01  2.855e+00  -9.265
## Is_Female:log_Total_Ct_Chng_Q4_Q1           -3.465e+00  7.553e-01  -4.587
## Customer_Age:Marital_Status                 -3.582e-02  9.984e-03  -3.587
##                                             Pr(>|z|)
## (Intercept)                                 2.28e-08 ***
## Customer_Age                                0.002203 **
## Is_Female                                   3.48e-11 ***
## Dependent_count                             0.000471 ***
## Education_Level                             0.054931 .
## Marital_Status                              0.001922 **
## Income_Category                             0.006696 **
## Total_Relationship_Count                     < 2e-16 ***
## Months_Inactive_12_mon                       < 2e-16 ***
## Contacts_Count_12_mon                        < 2e-16 ***
## log_Credit_Limit                            0.013073 *
```

```
## Total_Revolving_Bal                            4.09e-14 ***
## log_Total_Amt_Chng_Q4_Q1                        < 2e-16 ***
## log_Total_Trans_Amt                            0.013266 *
## Total_Trans_Ct                                  < 2e-16 ***
## log_Total_Ct_Chng_Q4_Q1                        5.03e-10 ***
## Avg_Utilization_Ratio                           < 2e-16 ***
## Customer_Age:log_Total_Amt_Chng_Q4_Q1          1.47e-06 ***
## Dependent_count:log_Total_Amt_Chng_Q4_Q1       0.000826 ***
## log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt   < 2e-16 ***
## log_Credit_Limit:Total_Revolving_Bal           4.91e-14 ***
## Total_Revolving_Bal:Avg_Utilization_Ratio       < 2e-16 ***
## log_Credit_Limit:Avg_Utilization_Ratio          < 2e-16 ***
## Is_Female:log_Total_Ct_Chng_Q4_Q1              4.49e-06 ***
## Customer_Age:Marital_Status                    0.000334 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6858.4  on 4947  degrees of freedom
## Residual deviance: 2858.1  on 4923  degrees of freedom
## AIC: 2908.1
##
## Number of Fisher Scoring iterations: 6
```

By checking the AIC we notice that the AIC gets smaller except for the first step done to reduce the effect of collinearity.

```
AIC(glm_1_bal,glm_2_bal,glm_3_bal,glm_4_bal,glm_5_bal,glm_6_bal,glm_7_bal)
```

```
##           df      AIC
## glm_1_bal 19 3309.868
## glm_2_bal 17 3342.759
## glm_3_bal 20 3219.639
## glm_4_bal 22 3027.721
## glm_5_bal 23 2939.042
## glm_6_bal 24 2919.088
## glm_7_bal 25 2908.103
```

We analyze now the performance of the final model. The best threshold obtained by comparing the f1 score has changed from the one used in the inital model. Because of this change we obtain lower accuracy, specificity and precision. However we also have a much higher Recall. Like it's usually the case, picking the balanced data helps the model to recognize the minority class, at the cost of a higher *type I error*. So you can pick the model obtained from the balanced data if you prioritize finding customers who are likely to churn, or you can pick the unbalanced one if you don't want to penalize the existing customer who didn't plan to churn.

```
#Threshold
Threshold <- 0.7
pred_glm_f_bal <- predict(glm_7_bal,test,type="response")
pred_f_bal <- ifelse(pred_glm_f_bal >= Threshold , 1,0)

#Confusion matrix
```

```
c_mat_f_bal <- table(test$Attrition_Flag,pred_f_bal)
```

Predicted Values			
Real Values	0	1	Total
0	1294	96	1390
1	56	202	258
Total	1350	298	1648

```
#Accuracy

mean(pred_f_bal==test$Attrition_Flag)*100
```

```
## [1] 90.7767
```

```
#True Negative Rate / Specificity

Spec_f_bal <- c_mat_f_bal[1,1]/sum(c_mat_f_bal[1,])
Spec_f_bal
```

```
## [1] 0.9309353
```

```
#Precision / Positive Predicted Value

Prec_f_bal <- c_mat_f_bal[2,2]/sum(c_mat_f_bal[,2])
Prec_f_bal
```

```
## [1] 0.6778523
```

```
#Recall / True Positive Rate / Sensitivity

Rec_f_bal <- c_mat_f_bal[2,2]/sum(c_mat_f_bal[2,])
Rec_f_bal
```

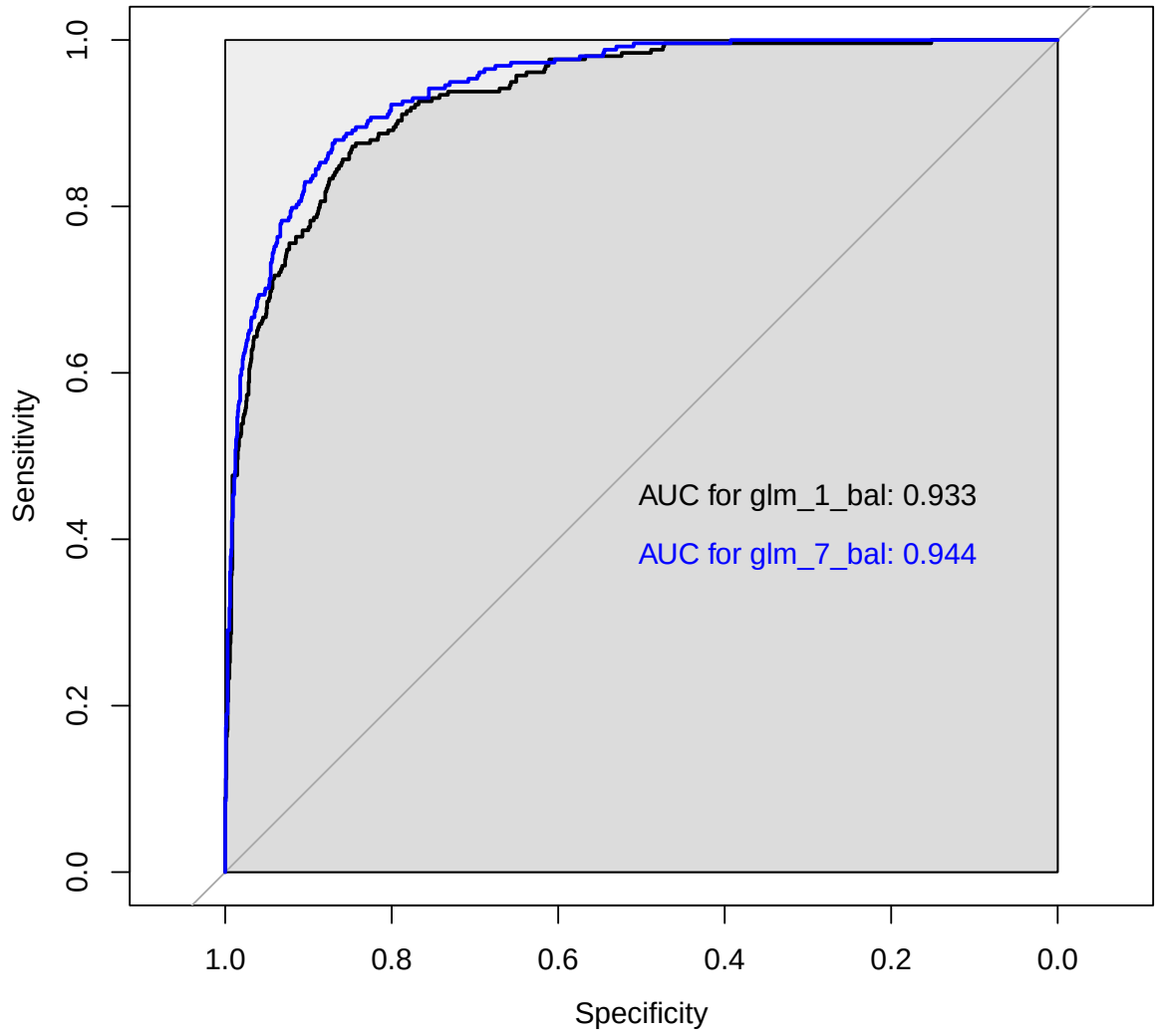
```
## [1] 0.7829457
```

```
#F1 Score

F1_f_bal <- 2 * (Prec_f_bal * Rec_f_bal)/(Prec_f_bal + Rec_f_bal)
F1_f_bal
```

```
## [1] 0.7266187
```

We now plot the ROC curve and the corresponding AUC values. In this case, the *glm_7_bal* seems to perform better than *glm_1_bal*, in terms of AUC values and F1 score. However, between the unbalanced final model and the balanced one we prefer the model obtained with the unbalanced dataset since it has a higher F1 score, while having the same AUC values. Instead, if we aimed to have a better *Sensitivity*, we might had taken into consideration the balanced model.

```r
#ROC curves
roc_i_bal <- roc(test$Attrition_Flag ~ pred_glm_i_bal)
roc_f_bal <- roc(test$Attrition_Flag ~ pred_glm_f_bal)


plot(roc_i_bal, col = "black",print.auc = FALSE, auc.polygon = TRUE,
     max.auc.polygon = TRUE, lwd=2)
plot(roc_f_bal,add = TRUE,col = "blue", print.auc = FALSE, lwd=2)
text(0.3, 0.45, paste("AUC for glm_1_bal:", round(roc_i_bal$auc, 3)), col = "black")
text(0.3, 0.38, paste("AUC for glm_7_bal:", round(roc_f_bal$auc, 3)), col = "blue")
```



Discriminant analysis

Discriminant analysis is a multivariate technique based on Bayes' rule, used to separate two or more groups of observations based on a set of predictor variables. It works by estimating the contribution that each

predictor has in separating the groups. To do so it aims to minimize the variance within the classes and maximizing the variance between classes. we will consider the binary case, where it has to separate only two groups. As with regression, discriminant analysis can be linear, aiming to find a linear decision boundary for each class but it also can be polynomial. We will analyze the models obtained in the linear (LDA) and quadratic (QDA) case.

First of all we want to point out that discriminant analysis assumes that the variables are distributed normally on the two groups. In our case we can see with the *Shapiro-Wilk normality test* that the normality assumption is not satisfied. We still decide to apply the models to our data and compare theirs performances with the other models. We report only some of the *Shapiro-Wilk tests* to not take too much space.

```
# We can see that the continuous predictor variables don't follow a normal
# distribution on both classes
apply(train[train$Attrition_Flag == 1,][17:18],2,shapiro.test )
```

```
## $Total_Trans_Ct
##
##  Shapiro-Wilk normality test
##
## data:  newX[, i]
## W = 0.96628, p-value = 2.162e-12
##
##
## $log_Total_Ct_Chng_Q4_Q1
##
##  Shapiro-Wilk normality test
##
## data:  newX[, i]
## W = 0.97888, p-value = 3.747e-09
```

```
apply(train[train$Attrition_Flag == 0,][17:18],2,shapiro.test )
```

```
## $Total_Trans_Ct
##
##  Shapiro-Wilk normality test
##
## data:  newX[, i]
## W = 0.97923, p-value < 2.2e-16
##
##
## $log_Total_Ct_Chng_Q4_Q1
##
##  Shapiro-Wilk normality test
##
## data:  newX[, i]
## W = 0.93604, p-value < 2.2e-16
```

Linear discriminant analysis

We start with the Linear discriminant analysis (LDA) which works by projecting the data onto a lower-dimensional space that maximizes the separation between the classes. It does this by finding the contributions that each predictive variable has in the classification of the response variable. This will results in

discovering the best linear decision boundary. Other than the normality assumption, LDA also assumes that the covariance matrices are equal in all the classes.

Unbalanced dataset

We will consider the set of variables and interactions chosen in the final model of *Generalized Linear models*, since it performed better than the complete one and it mitigated the problem of collinearity.

```
lda_u <- lda(Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count
             + Marital_Status + Income_Category + Total_Relationship_Count
             + Months_Inactive_12_mon + Contacts_Count_12_mon + log_Credit_Limit
             + Total_Revolving_Bal + log_Total_Amt_Chng_Q4_Q1
             + log_Total_Trans_Amt + Total_Trans_Ct + log_Total_Ct_Chng_Q4_Q1
             + Avg_Utilization_Ratio + Customer_Age:log_Total_Amt_Chng_Q4_Q1
             + Dependent_count:log_Total_Amt_Chng_Q4_Q1
             + log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt
             + log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct
             + log_Credit_Limit:Total_Revolving_Bal
             + Total_Revolving_Bal:Avg_Utilization_Ratio
             + log_Credit_Limit:Avg_Utilization_Ratio
             + Is_Female:log_Total_Ct_Chng_Q4_Q1 + Customer_Age:Marital_Status ,
             data = train, family = "binomial")
lda_u
```

```
## Call:
## lda(Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count +
##     Marital_Status + Income_Category + Total_Relationship_Count +
##     Months_Inactive_12_mon + Contacts_Count_12_mon + log_Credit_Limit +
##     Total_Revolving_Bal + log_Total_Amt_Chng_Q4_Q1 + log_Total_Trans_Amt +
##     Total_Trans_Ct + log_Total_Ct_Chng_Q4_Q1 + Avg_Utilization_Ratio +
##     Customer_Age:log_Total_Amt_Chng_Q4_Q1 + Dependent_count:log_Total_Amt_Chng_Q4_Q1 +
##     log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt + log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct +
##     log_Credit_Limit:Total_Revolving_Bal + Total_Revolving_Bal:Avg_Utilization_Ratio +
##     log_Credit_Limit:Avg_Utilization_Ratio + Is_Female:log_Total_Ct_Chng_Q4_Q1 +
##     Customer_Age:Marital_Status, data = train, family = "binomial")
##
## Prior probabilities of groups:
##        0        1
## 0.843169 0.156831
##
## Group means:
##   Customer_Age Is_Female Dependent_count Marital_Status Income_Category
## 0     46.31807 0.4805849        2.333413       1.668744        2.311841
## 1     46.32474 0.5270619        2.364691       1.628866        2.291237
##   Total_Relationship_Count Months_Inactive_12_mon Contacts_Count_12_mon
## 0                 3.979147               2.287152              2.366970
## 1                 3.295103               2.682990              2.960052
##   log_Credit_Limit Total_Revolving_Bal log_Total_Amt_Chng_Q4_Q1
## 0         8.488633           1255.4830                0.5668119
## 1         8.382879            698.8531                0.5142421
##   log_Total_Trans_Amt Total_Trans_Ct log_Total_Ct_Chng_Q4_Q1
## 0            8.183691       67.17186               0.5476496
## 1            7.772999       43.68299               0.4255542
##   Avg_Utilization_Ratio Customer_Age:log_Total_Amt_Chng_Q4_Q1
## 0             0.3211337                              26.15920
```

```
## 1             0.1803750                                23.86579
##   Dependent_count:log_Total_Amt_Chng_Q4_Q1
## 0                                1.319584
## 1                                1.199463
##   log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt
## 0                                    4.636002
## 1                                    4.026824
##   log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct log_Credit_Limit:Total_Revolving_Bal
## 0                                37.86805                                10724.776
## 1                                23.02526                                 5949.472
##   Total_Revolving_Bal:Avg_Utilization_Ratio
## 0                                  532.4306
## 1                                  335.0325
##   log_Credit_Limit:Avg_Utilization_Ratio Is_Female:log_Total_Ct_Chng_Q4_Q1
## 0                              2.588709                           0.2666409
## 1                              1.453220                           0.2192857
##   Customer_Age:Marital_Status
## 0                    77.34636
## 1                    75.47423
##
## Coefficients of linear discriminants:
##                                                      LD1
## Customer_Age                                -5.024756e-02
## Is_Female                                    1.517160e+00
## Dependent_count                              2.444113e-01
## Marital_Status                               5.208342e-01
## Income_Category                              1.004978e-01
## Total_Relationship_Count                    -2.381279e-01
## Months_Inactive_12_mon                       2.049182e-01
## Contacts_Count_12_mon                        2.015361e-01
## log_Credit_Limit                            -1.689417e-01
## Total_Revolving_Bal                          9.966297e-03
## log_Total_Amt_Chng_Q4_Q1                    -2.548910e+01
## log_Total_Trans_Amt                          2.948880e-02
## Total_Trans_Ct                              -7.041043e-02
## log_Total_Ct_Chng_Q4_Q1                     -2.020555e+00
## Avg_Utilization_Ratio                        4.434047e+01
## Customer_Age:log_Total_Amt_Chng_Q4_Q1        1.138823e-01
## Dependent_count:log_Total_Amt_Chng_Q4_Q1    -3.877739e-01
## log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt  2.516525e+00
## log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct      9.864627e-03
## log_Credit_Limit:Total_Revolving_Bal        -9.398113e-04
## Total_Revolving_Bal:Avg_Utilization_Ratio     2.141061e-03
## log_Credit_Limit:Avg_Utilization_Ratio      -7.210270e+00
## Is_Female:log_Total_Ct_Chng_Q4_Q1           -2.051566e+00
## Customer_Age:Marital_Status                 -1.313956e-02
```

We start by checking what is the best threshold in our model and the values of the F1 score and other significant performance's measures. The models seems to perform slightly worse than the final GLM in terms of F1 score and Recall, but it seems to predict better the negative response, since it has higher Specificity and Precision.

```r
#Threshold
Threshold <- 0.4
lda_u_predict <- predict(lda_u,test,type = "response")
lda_predict_u <- lda_u_predict$posterior
pred_lda_u <- ifelse(lda_predict_u[,2] >= Threshold , 1,0)

#Confusion matrix
c_mat_lda_u <- table(test$Attrition_Flag,pred_lda_u)
```

	Predicted Values		
Real Values	0	1	Total
0	1338	52	1390
1	78	180	258
Total	1416	232	1648

```r
#Accuracy

mean(pred_lda_u==test$Attrition_Flag)*100
```

```
## [1] 92.11165
```

```r
#True Negative Rate / Specificity

Spec_lda_u <- c_mat_lda_u[1,1]/sum(c_mat_lda_u[1,])
Spec_lda_u
```

```
## [1] 0.9625899
```

```r
#Precision / Positive Predicted Value

Prec_lda_u <- c_mat_lda_u[2,2]/sum(c_mat_lda_u[,2])
Prec_lda_u
```

```
## [1] 0.7758621
```

```r
#Recall / True Positive Rate / Sensitivity

Rec_lda_u <- c_mat_lda_u[2,2]/sum(c_mat_lda_u[2,])
Rec_lda_u
```

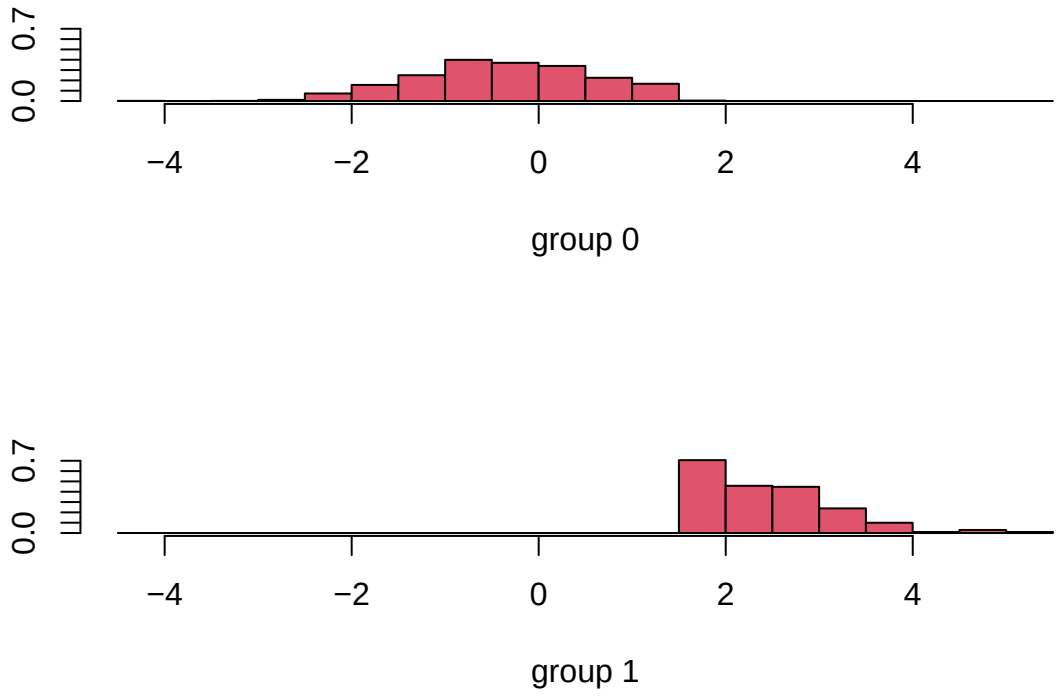
```
## [1] 0.6976744
```

```r
#F1 Score

F1_lda_u <- 2 * (Prec_lda_u * Rec_lda_u)/(Prec_lda_u + Rec_lda_u)
F1_lda_u
```

```
## [1] 0.7346939
```

Finally we shows how the model separates the two categories. We can notice from the graph below that it manages to separate accurately the two classes, with only an overlap where there are mostly elements belonging to the *Attrition* class.

```
# x indicates the linear combinations of the variables obtained by the model
# class indicates the two classes Existing and Attriting Customers.

ldahist(lda_u_predict$x[,1], g = lda_u_predict$class , col = 2)
```



Balanced dataset

We are now repeating the same steps done in the Unbalanced case. We consider the final set of variables used on the GLM.

```
lda_b <- lda(Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count
             + Education_Level + Marital_Status + Income_Category
             + Total_Relationship_Count + Months_Inactive_12_mon
             + Contacts_Count_12_mon + log_Credit_Limit + Total_Revolving_Bal
             + log_Total_Amt_Chng_Q4_Q1 + log_Total_Trans_Amt + Total_Trans_Ct
             + log_Total_Ct_Chng_Q4_Q1  + Avg_Utilization_Ratio
             + Customer_Age:log_Total_Amt_Chng_Q4_Q1
             + Dependent_count:log_Total_Amt_Chng_Q4_Q1
             + log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt
             + log_Credit_Limit:Total_Revolving_Bal
             + Total_Revolving_Bal:Avg_Utilization_Ratio
             + log_Credit_Limit:Avg_Utilization_Ratio
```

```
            + Is_Female:log_Total_Ct_Chng_Q4_Q1 + Customer_Age:Marital_Status ,
            data = train_bal, family = "binomial")
lda_b
```

```
## Call:
## lda(Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count +
##     Education_Level + Marital_Status + Income_Category + Total_Relationship_Count +
##     Months_Inactive_12_mon + Contacts_Count_12_mon + log_Credit_Limit +
##     Total_Revolving_Bal + log_Total_Amt_Chng_Q4_Q1 + log_Total_Trans_Amt +
##     Total_Trans_Ct + log_Total_Ct_Chng_Q4_Q1 + Avg_Utilization_Ratio +
##     Customer_Age:log_Total_Amt_Chng_Q4_Q1 + Dependent_count:log_Total_Amt_Chng_Q4_Q1 +
##     log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt + log_Credit_Limit:Total_Revolving_Bal +
##     Total_Revolving_Bal:Avg_Utilization_Ratio + log_Credit_Limit:Avg_Utilization_Ratio +
##     Is_Female:log_Total_Ct_Chng_Q4_Q1 + Customer_Age:Marital_Status,
##     data = train_bal, family = "binomial")
##
## Prior probabilities of groups:
##         0         1
## 0.4931285 0.5068715
##
## Group means:
##   Customer_Age Is_Female Dependent_count Education_Level Marital_Status
## 0     46.36189 0.4819672        2.320082        3.044672       1.665574
## 1     46.11164 0.5482456        2.319777        3.159888       1.615630
##   Income_Category Total_Relationship_Count Months_Inactive_12_mon
## 0        2.302869                 4.013115               2.311066
## 1        2.227273                 3.393142               2.681818
##   Contacts_Count_12_mon log_Credit_Limit Total_Revolving_Bal
## 0              2.377459         8.485016           1258.1189
## 1              2.930622         8.353818            713.8373
##   log_Total_Amt_Chng_Q4_Q1 log_Total_Trans_Amt Total_Trans_Ct
## 0                0.5655445            8.178498       67.08115
## 1                0.5124986            7.770140       43.67624
##   log_Total_Ct_Chng_Q4_Q1 Avg_Utilization_Ratio
## 0               0.5459014             0.3191320
## 1               0.4242582             0.1891061
##   Customer_Age:log_Total_Amt_Chng_Q4_Q1
## 0                              26.11212
## 1                              23.68924
##   Dependent_count:log_Total_Amt_Chng_Q4_Q1
## 0                                 1.311156
## 1                                 1.170198
##   log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt
## 0                                     4.623741
## 1                                     4.013659
##   log_Credit_Limit:Total_Revolving_Bal
## 0                            10762.028
## 1                             6057.782
##   Total_Revolving_Bal:Avg_Utilization_Ratio
## 0                                  532.3245
## 1                                  346.6438
##   log_Credit_Limit:Avg_Utilization_Ratio Is_Female:log_Total_Ct_Chng_Q4_Q1
## 0                               2.574800                          0.2650848
```

```
## 1                                1.518731                          0.2290144
##   Customer_Age:Marital_Status
## 0                    77.27131
## 1                    74.40670
##
## Coefficients of linear discriminants:
##                                                        LD1
## Customer_Age                                   -0.019241552
## Is_Female                                       0.687411340
## Dependent_count                                 0.228626198
## Education_Level                                 0.017642121
## Marital_Status                                  0.454461150
## Income_Category                                 0.075785675
## Total_Relationship_Count                       -0.184189230
## Months_Inactive_12_mon                          0.246639198
## Contacts_Count_12_mon                           0.167774038
## log_Credit_Limit                               -0.100043956
## Total_Revolving_Bal                             0.017779385
## log_Total_Amt_Chng_Q4_Q1                      -19.321505732
## log_Total_Trans_Amt                             0.482042311
## Total_Trans_Ct                                 -0.077659917
## log_Total_Ct_Chng_Q4_Q1                        -1.654777362
## Avg_Utilization_Ratio                          72.036190337
## Customer_Age:log_Total_Amt_Chng_Q4_Q1           0.055907085
## Dependent_count:log_Total_Amt_Chng_Q4_Q1       -0.384107687
## log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt    2.166725279
## log_Credit_Limit:Total_Revolving_Bal           -0.001652166
## Total_Revolving_Bal:Avg_Utilization_Ratio       0.002377059
## log_Credit_Limit:Avg_Utilization_Ratio        -11.520430023
## Is_Female:log_Total_Ct_Chng_Q4_Q1              -0.602196834
## Customer_Age:Marital_Status                    -0.012440655
```

We now search for the threshold resulting in the best F1 score and we obtain that it is 0.8. We have better results than the balanced GLM for everything other than Recall. Its performance is also comparable to the unbalanced LDA with better Recall at the price of a worse Precision.

```
#Threshold
Threshold <- 0.8
lda_b_predict <- predict(lda_b,test,type = "response")
lda_predict_b <- lda_b_predict$posterior
pred_lda_b <- ifelse(lda_predict_b[,2] >= Threshold , 1,0)

#Confusion matrix
c_mat_lda_b <- table(test$Attrition_Flag,pred_lda_b)
```

	Predicted Values		
Real Values	0	1	Total
0	1333	57	1390
1	76	182	258
Total	1409	239	1648

```r
#Accuracy

mean(pred_lda_b==test$Attrition_Flag)*100
```

```
## [1] 91.92961
```

```r
#True Negative Rate / Specificity

Spec_lda_b <- c_mat_lda_b[1,1]/sum(c_mat_lda_b[1,])
Spec_lda_b
```

```
## [1] 0.9589928
```

```r
#Precision / Positive Predicted Value

Prec_lda_b <- c_mat_lda_b[2,2]/sum(c_mat_lda_b[,2])
Prec_lda_b
```

```
## [1] 0.7615063
```

```r
#Recall / True Positive Rate / Sensitivity

Rec_lda_b <- c_mat_lda_b[2,2]/sum(c_mat_lda_b[2,])
Rec_lda_b
```

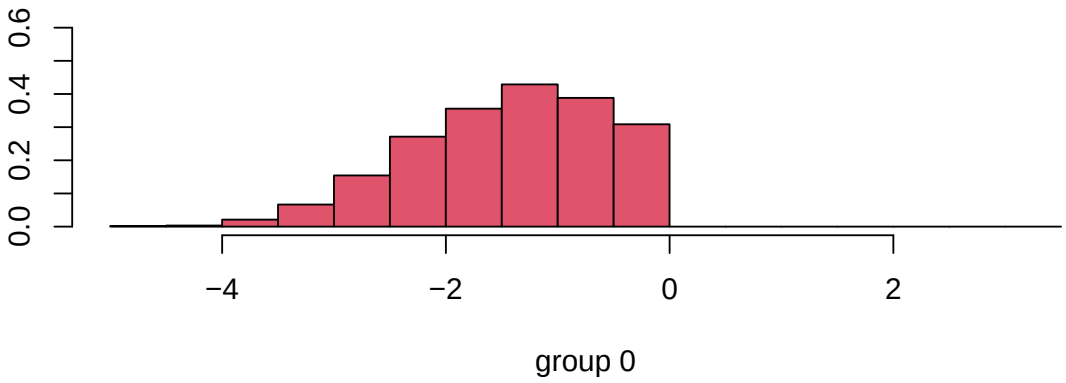
```
## [1] 0.7054264
```

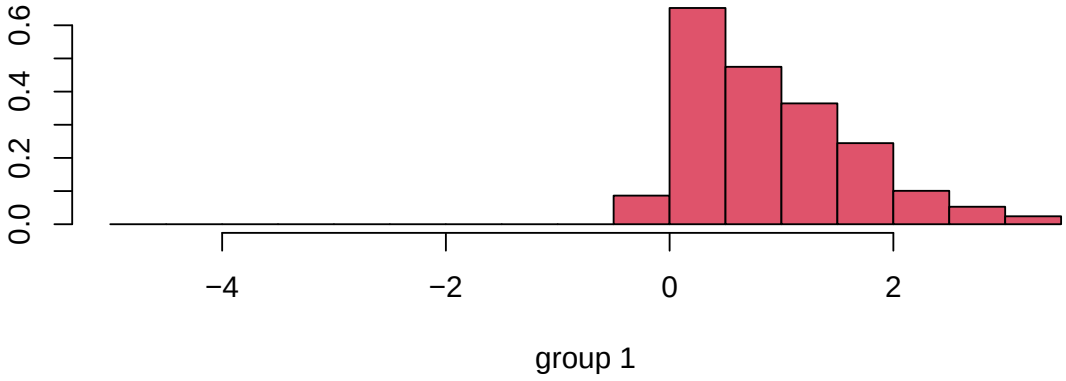
```r
#F1 Score

F1_lda_b <- 2 * (Prec_lda_b * Rec_lda_b)/(Prec_lda_b + Rec_lda_b)
F1_lda_b
```

```
## [1] 0.7323944
```

We can notice from the plots below how the model separate nicely the two classes but, in this case, the overlaps is more prominent than in the unbalanced model.

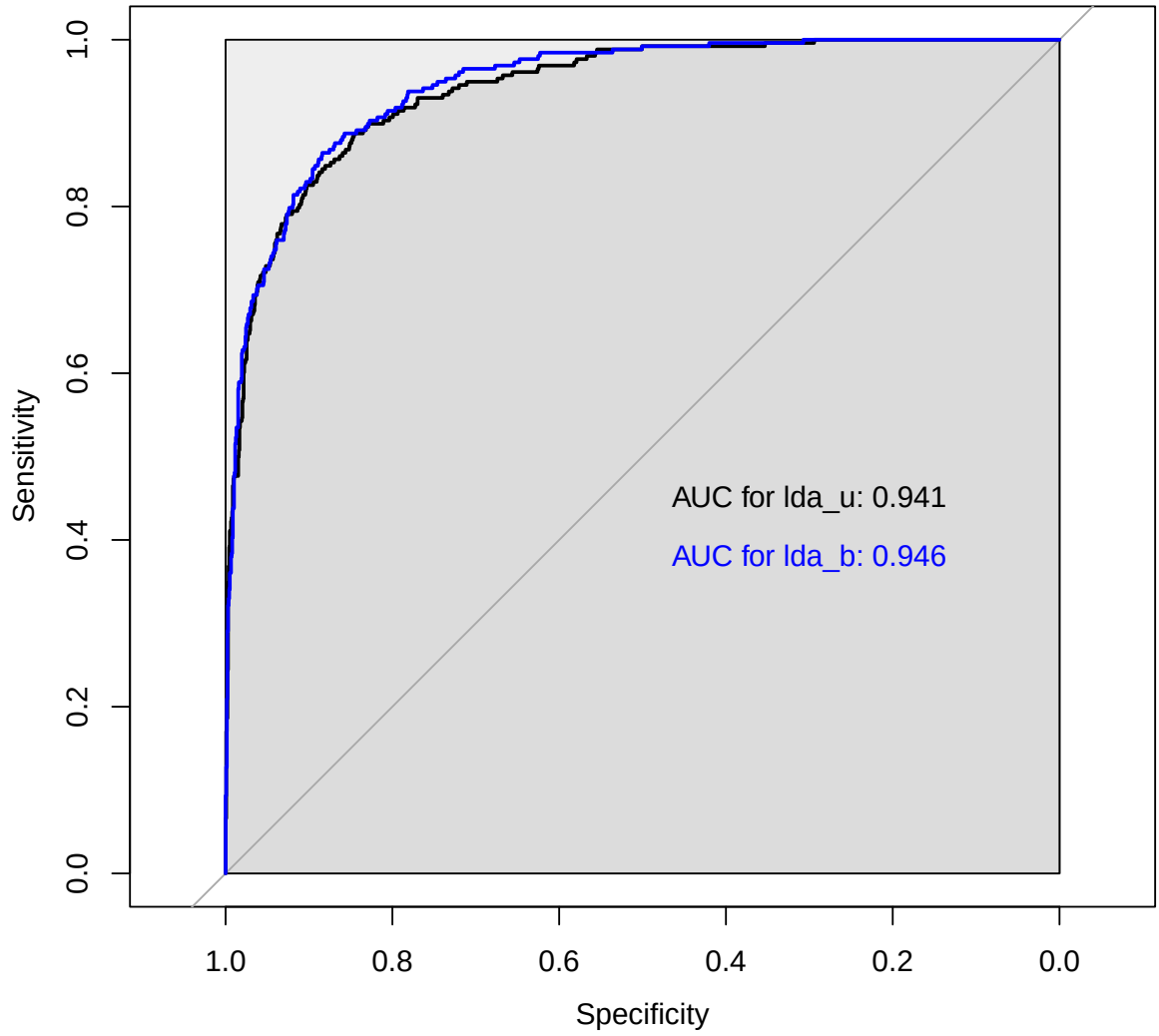
```r
# x indicates the linear combinations of the variables obtained by the model
# class indicates the two classes Existing and Attriting Customers.

ldahist(lda_b_predict$x[,1], g = lda_b_predict$class , col = 2)
```





We now shows the ROC curves and the AUC values corresponding to the LDA models trained on the unbalanced and balanced datasets. We can notice that the AUC value for the Balanced LDA is even higher than the best model obtained by GLM, even if the F1 score is lower.

```
#ROC curves
roc_lda_u <- roc(test$Attrition_Flag ~ lda_predict_u[,2])
roc_lda_b <- roc(test$Attrition_Flag ~ lda_predict_b[,2])

plot(roc_lda_u, col = "black",print.auc = FALSE, auc.polygon = TRUE,
     max.auc.polygon = TRUE, lwd=2)
plot(roc_lda_b,add = TRUE,col = "blue", print.auc = FALSE, lwd=2)
text(0.3, 0.45, paste("AUC for lda_u:", round(roc_lda_u$auc, 3)), col = "black")
text(0.3, 0.38, paste("AUC for lda_b:", round(roc_lda_b$auc, 3)), col = "blue")
```



Quadratic discriminant analysis

Quadratic discriminant analysis is computationally more expensive than LDA but it is able to capture more complex relationship between predictors and the response variable by using quadratic decision boundaries for each class. It also doesn't assume that the covariance matrices are equal in all the classes.

Unbalanced dataset

Like in LDA, we consider the set of variables and interactions chosen in *glm_8*, beacuse of its performance and the problem of collinearity.

```
qda_u <- qda(Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count
             + Marital_Status + Income_Category + Total_Relationship_Count
             + Months_Inactive_12_mon + Contacts_Count_12_mon + log_Credit_Limit
             + Total_Revolving_Bal + log_Total_Amt_Chng_Q4_Q1
             + log_Total_Trans_Amt + Total_Trans_Ct + log_Total_Ct_Chng_Q4_Q1
             + Avg_Utilization_Ratio + Customer_Age:log_Total_Amt_Chng_Q4_Q1
             + Dependent_count:log_Total_Amt_Chng_Q4_Q1
```

```
            + log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt
            + log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct
            + log_Credit_Limit:Total_Revolving_Bal
            + Total_Revolving_Bal:Avg_Utilization_Ratio
            + log_Credit_Limit:Avg_Utilization_Ratio
            + Is_Female:log_Total_Ct_Chng_Q4_Q1 + Customer_Age:Marital_Status ,
            data = train, family = "binomial")
qda_u
```

```
## Call:
## qda(Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count +
##     Marital_Status + Income_Category + Total_Relationship_Count +
##     Months_Inactive_12_mon + Contacts_Count_12_mon + log_Credit_Limit +
##     Total_Revolving_Bal + log_Total_Amt_Chng_Q4_Q1 + log_Total_Trans_Amt +
##     Total_Trans_Ct + log_Total_Ct_Chng_Q4_Q1 + Avg_Utilization_Ratio +
##     Customer_Age:log_Total_Amt_Chng_Q4_Q1 + Dependent_count:log_Total_Amt_Chng_Q4_Q1 +
##     log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt + log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct +
##     log_Credit_Limit:Total_Revolving_Bal + Total_Revolving_Bal:Avg_Utilization_Ratio +
##     log_Credit_Limit:Avg_Utilization_Ratio + Is_Female:log_Total_Ct_Chng_Q4_Q1 +
##     Customer_Age:Marital_Status, data = train, family = "binomial")
##
## Prior probabilities of groups:
##        0        1
## 0.843169 0.156831
##
## Group means:
##   Customer_Age Is_Female Dependent_count Marital_Status Income_Category
## 0     46.31807 0.4805849        2.333413       1.668744        2.311841
## 1     46.32474 0.5270619        2.364691       1.628866        2.291237
##   Total_Relationship_Count Months_Inactive_12_mon Contacts_Count_12_mon
## 0                 3.979147               2.287152              2.366970
## 1                 3.295103               2.682990              2.960052
##   log_Credit_Limit Total_Revolving_Bal log_Total_Amt_Chng_Q4_Q1
## 0         8.488633           1255.4830                0.5668119
## 1         8.382879            698.8531                0.5142421
##   log_Total_Trans_Amt Total_Trans_Ct log_Total_Ct_Chng_Q4_Q1
## 0            8.183691       67.17186               0.5476496
## 1            7.772999       43.68299               0.4255542
##   Avg_Utilization_Ratio Customer_Age:log_Total_Amt_Chng_Q4_Q1
## 0             0.3211337                              26.15920
## 1             0.1803750                              23.86579
##   Dependent_count:log_Total_Amt_Chng_Q4_Q1
## 0                                1.319584
## 1                                1.199463
##   log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt
## 0                                     4.636002
## 1                                     4.026824
##   log_Total_Amt_Chng_Q4_Q1:Total_Trans_Ct log_Credit_Limit:Total_Revolving_Bal
## 0                                37.86805                              10724.776
## 1                                23.02526                               5949.472
##   Total_Revolving_Bal:Avg_Utilization_Ratio
## 0                                  532.4306
## 1                                  335.0325
```

```
##   log_Credit_Limit:Avg_Utilization_Ratio Is_Female:log_Total_Ct_Chng_Q4_Q1
## 0                              2.588709                          0.2666409
## 1                              1.453220                          0.2192857
##   Customer_Age:Marital_Status
## 0                    77.34636
## 1                    75.47423
```

We have that 0.4 is the threshold resulting in the best F1 score and we use it to calculate other significant measures for the model's performance. In terms of F1 score it is the better performing model analyzed until now. It has much higher Sensitivity in comparison to the other models but lower Specificity and accuracy.

```
#Threshold
Threshold <- 0.4
qda_u_predict <- predict(qda_u,test,type = "response")
qda_predict_u <- qda_u_predict$posterior
pred_qda_u <- ifelse(qda_predict_u[,2] >= Threshold , 1,0)

#Confusion matrix
c_mat_qda_u <- table(test$Attrition_Flag,pred_qda_u)
```

	Predicted Values		
Real Values	0	1	Total
0	1304	86	1390
1	50	208	258
Total	1354	294	1648

```
#Accuracy

mean(pred_qda_u==test$Attrition_Flag)*100
```

```
## [1] 91.74757
```

```
#True Negative Rate / Specificity

Spec_qda_u <- c_mat_qda_u[1,1]/sum(c_mat_qda_u[1,])
Spec_qda_u
```

```
## [1] 0.9381295
```

```
#Precision / Positive Predicted Value

Prec_qda_u <- c_mat_qda_u[2,2]/sum(c_mat_qda_u[,2])
Prec_qda_u
```

```
## [1] 0.707483
```

```
#Recall / True Positive Rate / Sensitivity

Rec_qda_u <- c_mat_qda_u[2,2]/sum(c_mat_qda_u[2,])
Rec_qda_u
```

```
## [1] 0.8062016
```

```
#F1 Score

F1_qda_u <- 2 * (Prec_qda_u * Rec_qda_u)/(Prec_qda_u + Rec_qda_u)
F1_qda_u
```

```
## [1] 0.7536232
```

Balanced dataset

We consider the final set of variables used on the balanced GLM and we define the corresponding qda models.

```
qda_b <- qda(Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count
             + Education_Level + Marital_Status + Income_Category
             + Total_Relationship_Count + Months_Inactive_12_mon
             + Contacts_Count_12_mon + log_Credit_Limit + Total_Revolving_Bal
             + log_Total_Amt_Chng_Q4_Q1 + log_Total_Trans_Amt + Total_Trans_Ct
             + log_Total_Ct_Chng_Q4_Q1  + Avg_Utilization_Ratio
             + Customer_Age:log_Total_Amt_Chng_Q4_Q1
             + Dependent_count:log_Total_Amt_Chng_Q4_Q1
             + log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt
             + log_Credit_Limit:Total_Revolving_Bal
             + Total_Revolving_Bal:Avg_Utilization_Ratio
             + log_Credit_Limit:Avg_Utilization_Ratio
             + Is_Female:log_Total_Ct_Chng_Q4_Q1 + Customer_Age:Marital_Status ,
             data = train_bal, family = "binomial")
qda_b
```

```
## Call:
## qda(Attrition_Flag ~ Customer_Age + Is_Female + Dependent_count +
##     Education_Level + Marital_Status + Income_Category + Total_Relationship_Count +
##     Months_Inactive_12_mon + Contacts_Count_12_mon + log_Credit_Limit +
##     Total_Revolving_Bal + log_Total_Amt_Chng_Q4_Q1 + log_Total_Trans_Amt +
##     Total_Trans_Ct + log_Total_Ct_Chng_Q4_Q1 + Avg_Utilization_Ratio +
##     Customer_Age:log_Total_Amt_Chng_Q4_Q1 + Dependent_count:log_Total_Amt_Chng_Q4_Q1 +
##     log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt + log_Credit_Limit:Total_Revolving_Bal +
##     Total_Revolving_Bal:Avg_Utilization_Ratio + log_Credit_Limit:Avg_Utilization_Ratio +
##     Is_Female:log_Total_Ct_Chng_Q4_Q1 + Customer_Age:Marital_Status,
##     data = train_bal, family = "binomial")
##
## Prior probabilities of groups:
##         0         1
## 0.4931285 0.5068715
##
## Group means:
##   Customer_Age Is_Female Dependent_count Education_Level Marital_Status
## 0     46.36189 0.4819672        2.320082        3.044672       1.665574
## 1     46.11164 0.5482456        2.319777        3.159888       1.615630
##   Income_Category Total_Relationship_Count Months_Inactive_12_mon
## 0        2.302869                 4.013115               2.311066
## 1        2.227273                 3.393142               2.681818
##   Contacts_Count_12_mon log_Credit_Limit Total_Revolving_Bal
```

```
## 0              2.377459         8.485016           1258.1189
## 1              2.930622         8.353818            713.8373
##   log_Total_Amt_Chng_Q4_Q1 log_Total_Trans_Amt Total_Trans_Ct
## 0                0.5655445            8.178498       67.08115
## 1                0.5124986            7.770140       43.67624
##   log_Total_Ct_Chng_Q4_Q1 Avg_Utilization_Ratio
## 0               0.5459014             0.3191320
## 1               0.4242582             0.1891061
##   Customer_Age:log_Total_Amt_Chng_Q4_Q1
## 0                              26.11212
## 1                              23.68924
##   Dependent_count:log_Total_Amt_Chng_Q4_Q1
## 0                                 1.311156
## 1                                 1.170198
##   log_Total_Amt_Chng_Q4_Q1:log_Total_Trans_Amt
## 0                                     4.623741
## 1                                     4.013659
##   log_Credit_Limit:Total_Revolving_Bal
## 0                            10762.028
## 1                             6057.782
##   Total_Revolving_Bal:Avg_Utilization_Ratio
## 0                                  532.3245
## 1                                  346.6438
##   log_Credit_Limit:Avg_Utilization_Ratio Is_Female:log_Total_Ct_Chng_Q4_Q1
## 0                              2.574800                          0.2650848
## 1                              1.518731                          0.2290144
##   Customer_Age:Marital_Status
## 0                    77.27131
## 1                    74.40670
```

The best threshold based on the F1 score is 0.8. The performance is comparable to the Unbalanced QDA with a slightly better Specificity at the cost of a slightly worse Sensitivity which results in a lower F1 score.

```
#Threshold
Threshold <- 0.8
qda_b_predict <- predict(qda_b,test,type = "response")
qda_predict_b <- qda_b_predict$posterior
pred_qda_b <- ifelse(qda_predict_b[,2] >= Threshold , 1,0)

#Confusion matrix
c_mat_qda_b <- table(test$Attrition_Flag,pred_qda_b)
```

	Predicted Values		
Real Values	0	1	Total
0	1309	81	1390
1	57	201	258
Total	1366	282	1648

```
#Accuracy

mean(pred_qda_b==test$Attrition_Flag)*100
```

```
## [1] 91.62621
```

```
#True Negative Rate / Specificity

Spec_qda_b <- c_mat_qda_b[1,1]/sum(c_mat_qda_b[1,])
Spec_qda_b
```

```
## [1] 0.9417266
```

```
#Precision / Positive Predicted Value

Prec_qda_b <- c_mat_qda_b[2,2]/sum(c_mat_qda_b[,2])
Prec_qda_b
```

```
## [1] 0.712766
```

```
#Recall / True Positive Rate / Sensitivity

Rec_qda_b <- c_mat_qda_b[2,2]/sum(c_mat_qda_b[2,])
Rec_qda_b
```

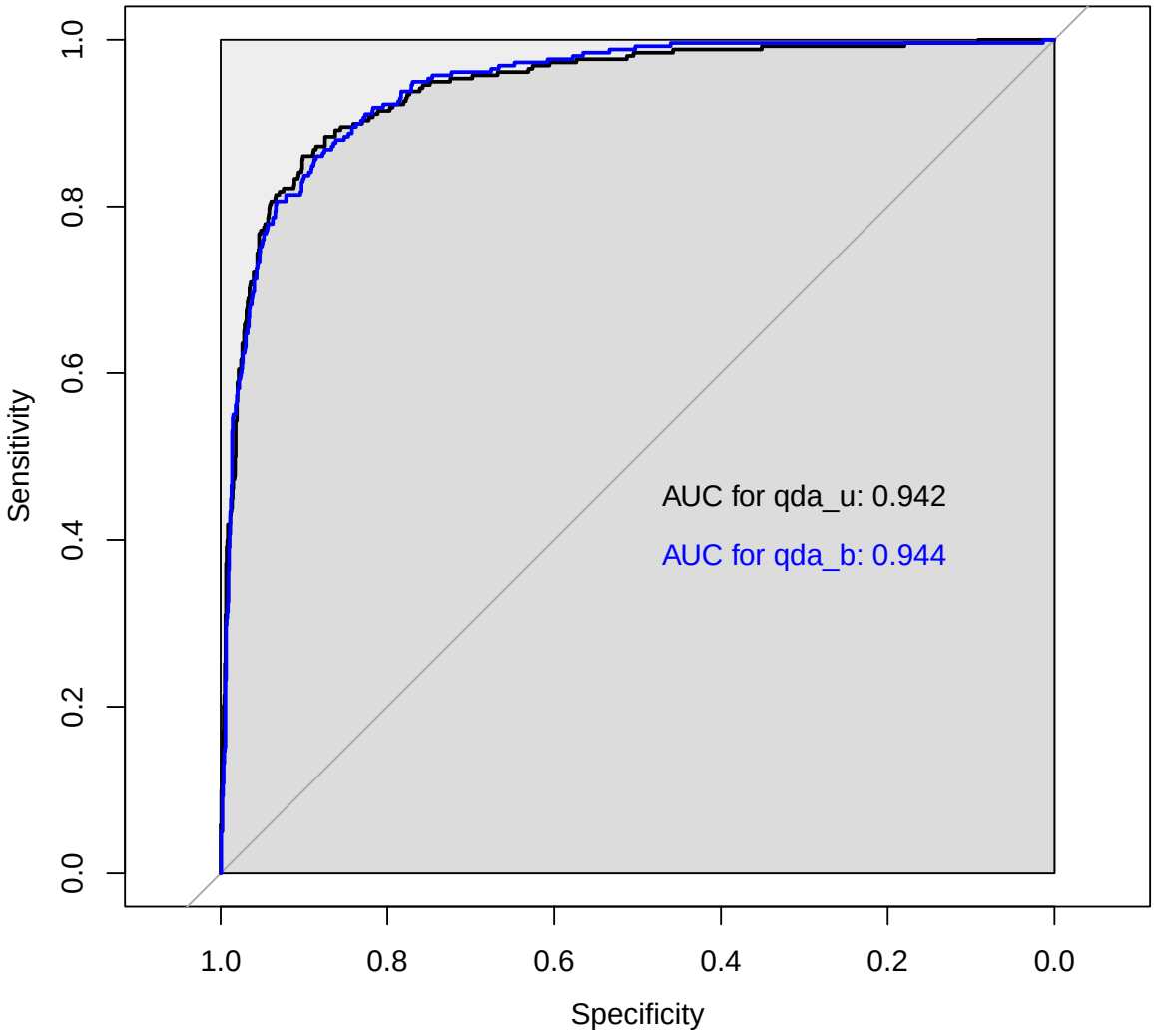
```
## [1] 0.7790698
```

```
#F1 Score

F1_qda_b <- 2 * (Prec_qda_b * Rec_qda_b)/(Prec_qda_b + Rec_qda_b)
F1_qda_b
```

```
## [1] 0.7444444
```

We plot the ROC curves and corresponding AUC values of the two QDA models. We can notice that the two ROC curve are pretty similar and there is not a curve which is strictly above the other. The AUC values are also pretty high but still lower than the LDA model built on the balanced dataset.

```
#ROC curves
roc_qda_u <- roc(test$Attrition_Flag ~ qda_predict_u[,2])
roc_qda_b <- roc(test$Attrition_Flag ~ qda_predict_b[,2])

plot(roc_qda_u, col = "black",print.auc = FALSE, auc.polygon = TRUE,
     max.auc.polygon = TRUE, lwd=2)
plot(roc_qda_b,add = TRUE,col = "blue", print.auc = FALSE, lwd=2)
text(0.3, 0.45, paste("AUC for qda_u:", round(roc_qda_u$auc, 3)), col = "black")
text(0.3, 0.38, paste("AUC for qda_b:", round(roc_qda_b$auc, 3)), col = "blue")
```



Shrinkage methods

We now analyze the models obtained by using *Ridge* and *Lasso*, two famous shrinkage methods. A shrinkage method, also called Regularization, is used to train models that generalize better on unseen data, in our case in the test dataset. It do so by shrinking the coefficients estimates towards 0, preventing the model from overfitting. The two shrinking methods that we are considering, *Ridge regression* and *Lasso regression*, works by addind a penalization term to the ordinary least square (OLS) function, which is the objective function of linear regression. In this way they reduce the impact of collinearity and prevent overfitting. Since collinearity is a problem in our models, we explore the performances of these models.

To begin with, we define a new training and test set obtained by adding to the original datasets the interactions term as variables and we will later transform them into matrices. This is done since we will use the function *cv.glmnet*, from the *glmnet* library. In fact this function takes a matrix as an input and to include the interactions we will then need to create new variables.

```r
#Creation of train and test set with interactions
train_int <- data.frame(train)

train_int$log_Total_Amt_Chng_Q4_Q1_Customer_Age <-
  train$log_Total_Amt_Chng_Q4_Q1 * train$Customer_Age
train_int$log_Total_Amt_Chng_Q4_Q1_Dependent_count <-
    train$log_Total_Amt_Chng_Q4_Q1 * train$Dependent_count
train_int$log_Total_Amt_Chng_Q4_Q1_log_Total_Trans_Amt <-
    train$log_Total_Amt_Chng_Q4_Q1*train$log_Total_Trans_Amt
train_int$log_Total_Amt_Chng_Q4_Q1_Total_Trans_Ct <-
    train$log_Total_Amt_Chng_Q4_Q1 * train$Total_Trans_Ct
train_int$log_Credit_Limit_Total_Revolving_Bal <-
    train$log_Credit_Limit * train$Total_Revolving_Bal
train_int$Total_Revolving_Bal_Avg_Utilization_Ratio <-
    train$Total_Revolving_Bal * train$Avg_Utilization_Ratio
train_int$log_Credit_Limit_Avg_Utilization_Ratio <-
    train$log_Credit_Limit * train$Avg_Utilization_Ratio
train_int$Is_Female_log_Total_Ct_Chng_Q4_Q1 <-
    train$Is_Female * train$log_Total_Ct_Chng_Q4_Q1
train_int$Customer_Age_Marital_Status <-
    train$Customer_Age * train$Marital_Status

test_int <- data.frame(test)

test_int$log_Total_Amt_Chng_Q4_Q1_Customer_Age <-
    test$log_Total_Amt_Chng_Q4_Q1 * test$Customer_Age
test_int$log_Total_Amt_Chng_Q4_Q1_Dependent_count <-
    test$log_Total_Amt_Chng_Q4_Q1 * test$Dependent_count
test_int$log_Total_Amt_Chng_Q4_Q1_log_Total_Trans_Amt <-
    test$log_Total_Amt_Chng_Q4_Q1 * test$log_Total_Trans_Amt
test_int$log_Total_Amt_Chng_Q4_Q1_Total_Trans_Ct <-
    test$log_Total_Amt_Chng_Q4_Q1 * test$Total_Trans_Ct
test_int$log_Credit_Limit_Total_Revolving_Bal <-
    test$log_Credit_Limit * test$Total_Revolving_Bal
test_int$Total_Revolving_Bal_Avg_Utilization_Ratio <-
    test$Total_Revolving_Bal * test$Avg_Utilization_Ratio
test_int$log_Credit_Limit_Avg_Utilization_Ratio <-
    test$log_Credit_Limit * test$Avg_Utilization_Ratio
test_int$Is_Female_log_Total_Ct_Chng_Q4_Q1 <-
    test$Is_Female * test$log_Total_Ct_Chng_Q4_Q1
test_int$Customer_Age_Marital_Status <-
    test$Customer_Age * test$Marital_Status
```

We do the same for the balanced set

```r
#Creation of train_bal and test set with interactions
train_bal_int <- data.frame(train_bal)

train_bal_int$log_Total_Amt_Chng_Q4_Q1_Customer_Age <-
    train_bal$log_Total_Amt_Chng_Q4_Q1 * train_bal$Customer_Age
train_bal_int$log_Total_Amt_Chng_Q4_Q1_Dependent_count <-
    train_bal$log_Total_Amt_Chng_Q4_Q1 * train_bal$Dependent_count
train_bal_int$log_Total_Amt_Chng_Q4_Q1_log_Total_Trans_Amt <-
```

```
    train_bal$log_Total_Amt_Chng_Q4_Q1 * train_bal$log_Total_Trans_Amt
train_bal_int$log_Credit_Limit_Total_Revolving_Bal <-
    train_bal$log_Credit_Limit * train_bal$Total_Revolving_Bal
train_bal_int$Total_Revolving_Bal_Avg_Utilization_Ratio <-
    train_bal$Total_Revolving_Bal * train_bal$Avg_Utilization_Ratio
train_bal_int$log_Credit_Limit_Avg_Utilization_Ratio <-
    train_bal$log_Credit_Limit * train_bal$Avg_Utilization_Ratio
train_bal_int$Is_Female_log_Total_Trans_Ct_Q4_Q1 <-
    train_bal$Is_Female * train_bal$log_Total_Ct_Chng_Q4_Q1
train_bal_int$Customer_Age_Marital_Status <-
    train_bal$Customer_Age * train_bal$Marital_Status

test_bal_int <- data.frame(test)

test_bal_int$log_Total_Amt_Chng_Q4_Q1_Customer_Age <-
    test$log_Total_Amt_Chng_Q4_Q1 * test$Customer_Age
test_bal_int$log_Total_Amt_Chng_Q4_Q1_Dependent_count <-
    test$log_Total_Amt_Chng_Q4_Q1 * test$Dependent_count
test_bal_int$log_Total_Amt_Chng_Q4_Q1_log_Total_Trans_Amt <-
    test$log_Total_Amt_Chng_Q4_Q1 * test$log_Total_Trans_Amt
test_bal_int$log_Credit_Limit_Total_Revolving_Bal <-
    test$log_Credit_Limit * test$Total_Revolving_Bal
test_bal_int$Total_Revolving_Bal_Avg_Utilization_Ratio <-
    test$Total_Revolving_Bal * test$Avg_Utilization_Ratio
test_bal_int$log_Credit_Limit_Avg_Utilization_Ratio <-
    test$log_Credit_Limit * test$Avg_Utilization_Ratio
test_bal_int$Is_Female_log_Total_Trans_Ct_Q4_Q1 <-
    test$Is_Female * test$log_Total_Ct_Chng_Q4_Q1
test_bal_int$Customer_Age_Marital_Status <-
    test$Customer_Age * test$Marital_Status
```

Ridge regression

Ridge regression adds a penalty term to the objective function that is proportional to the L2 norm of the vector of the coefficients, shrinking them towards 0. However, since it doesn't actually set them to zero, we will consider the set of variables which gave the best performing GLM.

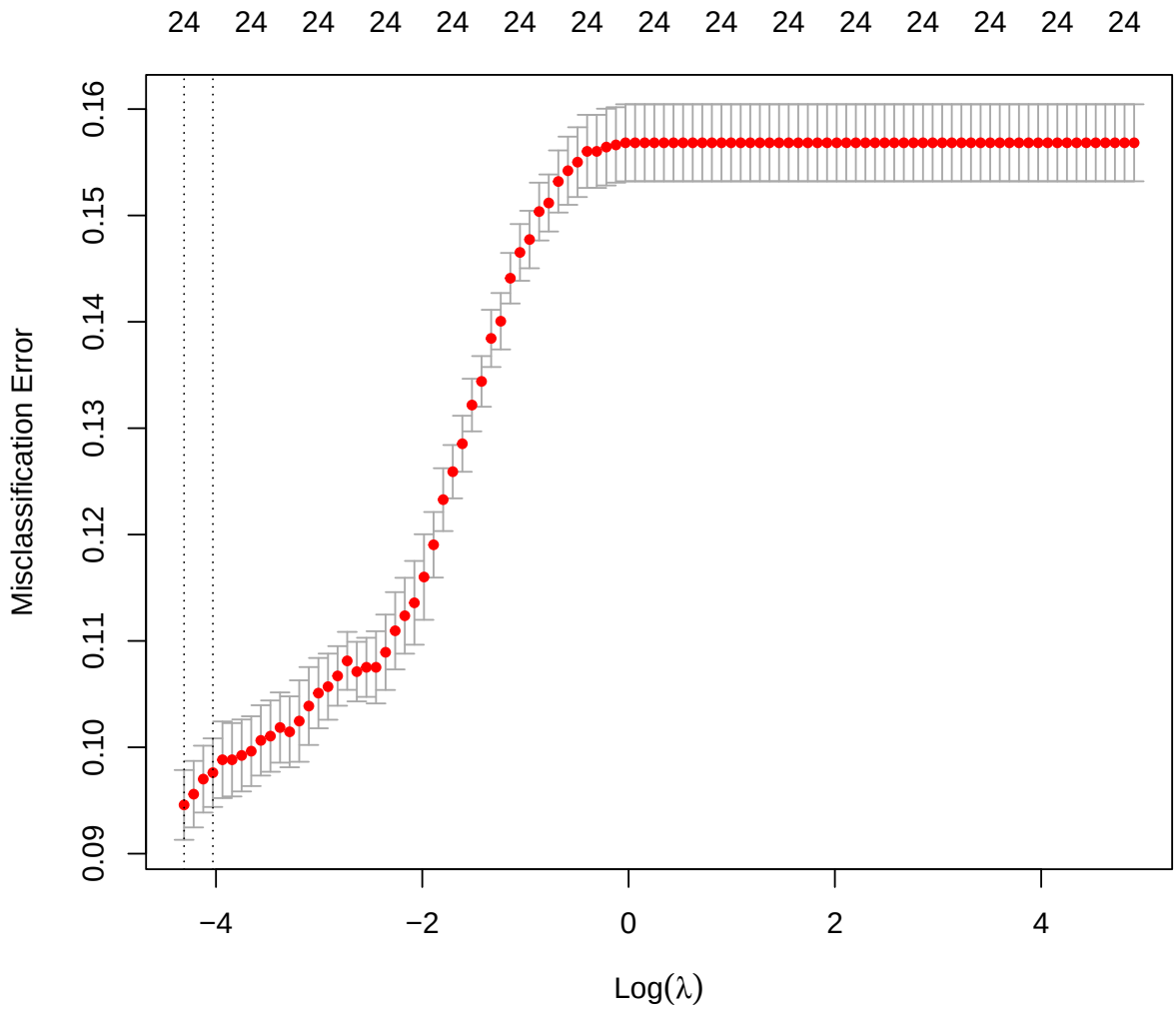
Unbalanced dataset

We create the matrix that we will use for the model

```
# Matrix without Attrition_Flag, Education_Level , Months_on_book and log_Avg_Open_To_Buy
train_mat <- data.matrix(train_int[,-c(1,5,8,14)])
test_mat <- data.matrix(test_int[,-c(1,5,8,14)])
```

We apply Ridge logistic regression to our matrix and we extract the optimal plot the misclassification error, which depends on the values of $\log(\lambda)$, where λ is the parameter that controls the amount of penalization. We can notice that the increase of the penalization leads to an increase of the misclassification error based on the minimum misclassification error. Finally we extract the value of λ that minimizes the misclassification error.

```
ridge_u <- cv.glmnet(train_mat,train$Attrition_Flag, alpha = 0,
                     family = "binomial", type.measure = "class")
plot(ridge_u)
```




```
lambda_ridge_u <- ridge_u$lambda.min
lambda_ridge_u
```

```
## [1] 0.01344236
```

The best threshold for the F1 score is 0.3. It seems that the model has a worse performance in comparison to the models previously analyzed.

```
#Threshold
Threshold <- 0.3
ridge_predict_u <- predict(ridge_u,test_mat,type = "response", s = lambda_ridge_u)
pred_ridge_u <- ifelse(ridge_predict_u >= Threshold , 1,0)

#Confusion matrix
c_mat_ridge_u <- table(test$Attrition_Flag,pred_ridge_u)
```

Predicted Values			
Real Values	0	1	Total
0	1300	90	1390
1	79	179	258
Total	1379	269	1648

```
#Accuracy

mean(pred_ridge_u==test$Attrition_Flag)*100
```

```
## [1] 89.74515
```

```
#True Negative Rate / Specificity

Spec_ridge_u <- c_mat_ridge_u[1,1]/sum(c_mat_ridge_u[1,])
Spec_ridge_u
```

```
## [1] 0.9352518
```

```
#Precision / Positive Predicted Value

Prec_ridge_u <- c_mat_ridge_u[2,2]/sum(c_mat_ridge_u[,2])
Prec_ridge_u
```

```
## [1] 0.6654275
```

```
#Recall / True Positive Rate / Sensitivity

Rec_ridge_u <- c_mat_ridge_u[2,2]/sum(c_mat_ridge_u[2,])
Rec_ridge_u
```

```
## [1] 0.6937984
```

```
#F1 Score

F1_ridge_u <- 2 * (Prec_ridge_u * Rec_ridge_u)/(Prec_ridge_u + Rec_ridge_u)
F1_ridge_u
```

```
## [1] 0.6793169
```

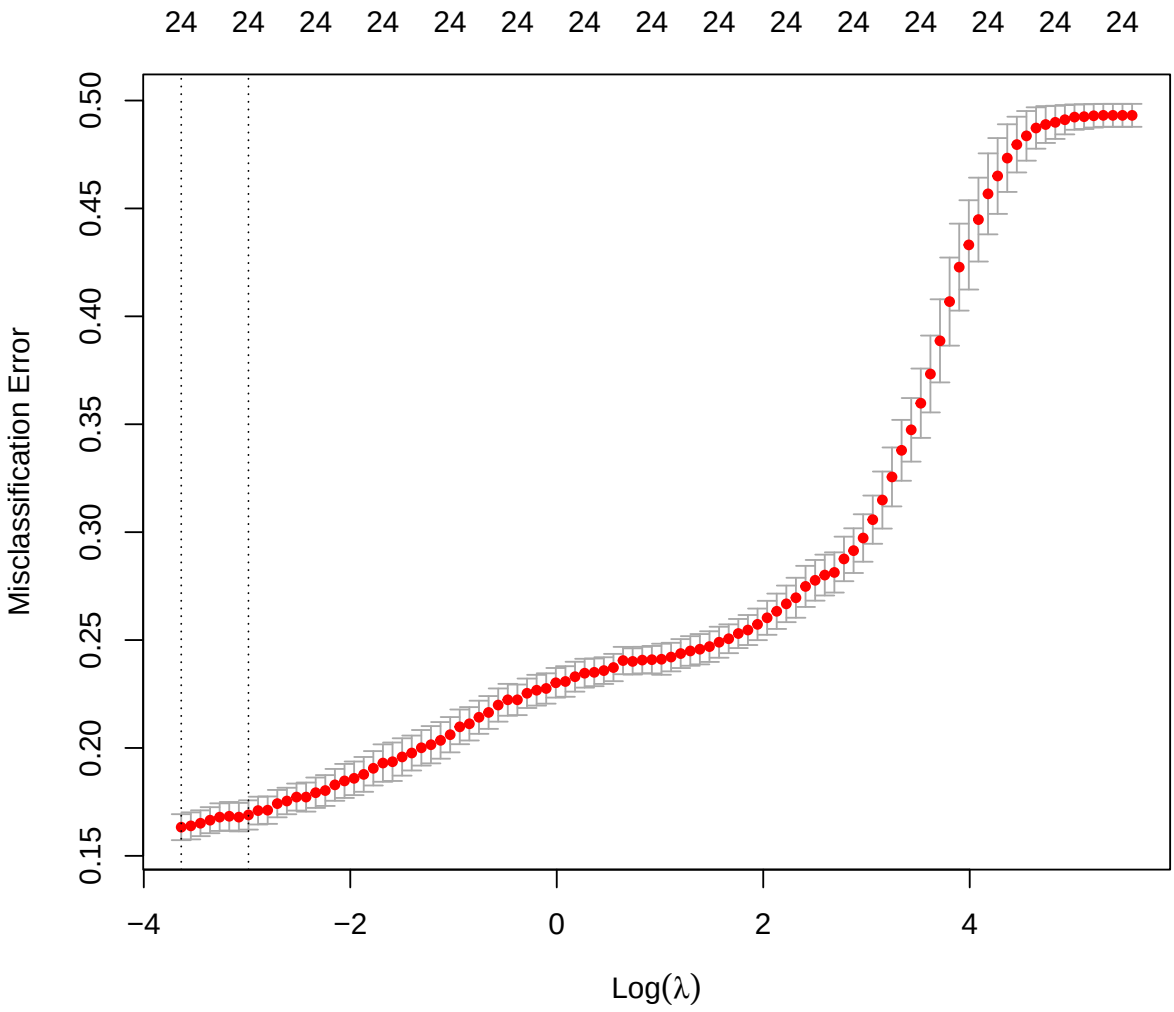
Balanced dataset

We create the matrix that we will use for the model

```
# Matrix without Attrition_Flag, Months_on_book and log_Avg_Open_To_Buy
train_mat_b <- data.matrix(train_bal_int[,-c(1,8,14)])
test_mat_b <- data.matrix(test_bal_int[,-c(1,8,14)])
```

We apply the same step done with the balanced step. We can notice that, like in the previous case the increase of the penalization leads to an increase of the misclassification error. Moreover for big values of λ the misclassification error gets close to 0.5.

```
ridge_b <- cv.glmnet(train_mat_b,train_bal$Attrition_Flag, alpha = 0,
                     family = "binomial", type.measure = "class")
plot(ridge_b)
```



Finally we extract the optimal lambda based on the minimum misclassification error.

```
lambda_ridge_b <- ridge_b$lambda.min
lambda_ridge_b
```

```
## [1] 0.0263209
```

The best threshold for the F1 score is 0.6. While it has a better Recall that the unbalanced model, it seems that its performance is much worse in comparison to the other models.

```r
#Threshold
Threshold <- 0.6
ridge_predict_b <- predict(ridge_b,test_mat_b,type = "response", s = lambda_ridge_b)
pred_ridge_b <- ifelse(ridge_predict_b >= Threshold , 1,0)

#Confusion matrix
c_mat_ridge_b <- table(test$Attrition_Flag,pred_ridge_b)
```

	Predicted Values		
Real Values	0	1	Total
0	1262	128	1390
1	66	192	258
Total	1328	320	1648

```r
#Accuracy

mean(pred_ridge_b==test$Attrition_Flag)*100
```

```
## [1] 88.22816
```

```r
#True Negative Rate / Specificity

Spec_ridge_b <- c_mat_ridge_b[1,1]/sum(c_mat_ridge_b[1,])
Spec_ridge_b
```

```
## [1] 0.9079137
```

```r
#Precision / Positive Predicted Value

Prec_ridge_b <- c_mat_ridge_b[2,2]/sum(c_mat_ridge_b[,2])
Prec_ridge_b
```

```
## [1] 0.6
```

```r
#Recall / True Positive Rate / Sensitivity

Rec_ridge_b <- c_mat_ridge_b[2,2]/sum(c_mat_ridge_b[2,])
Rec_ridge_b
```

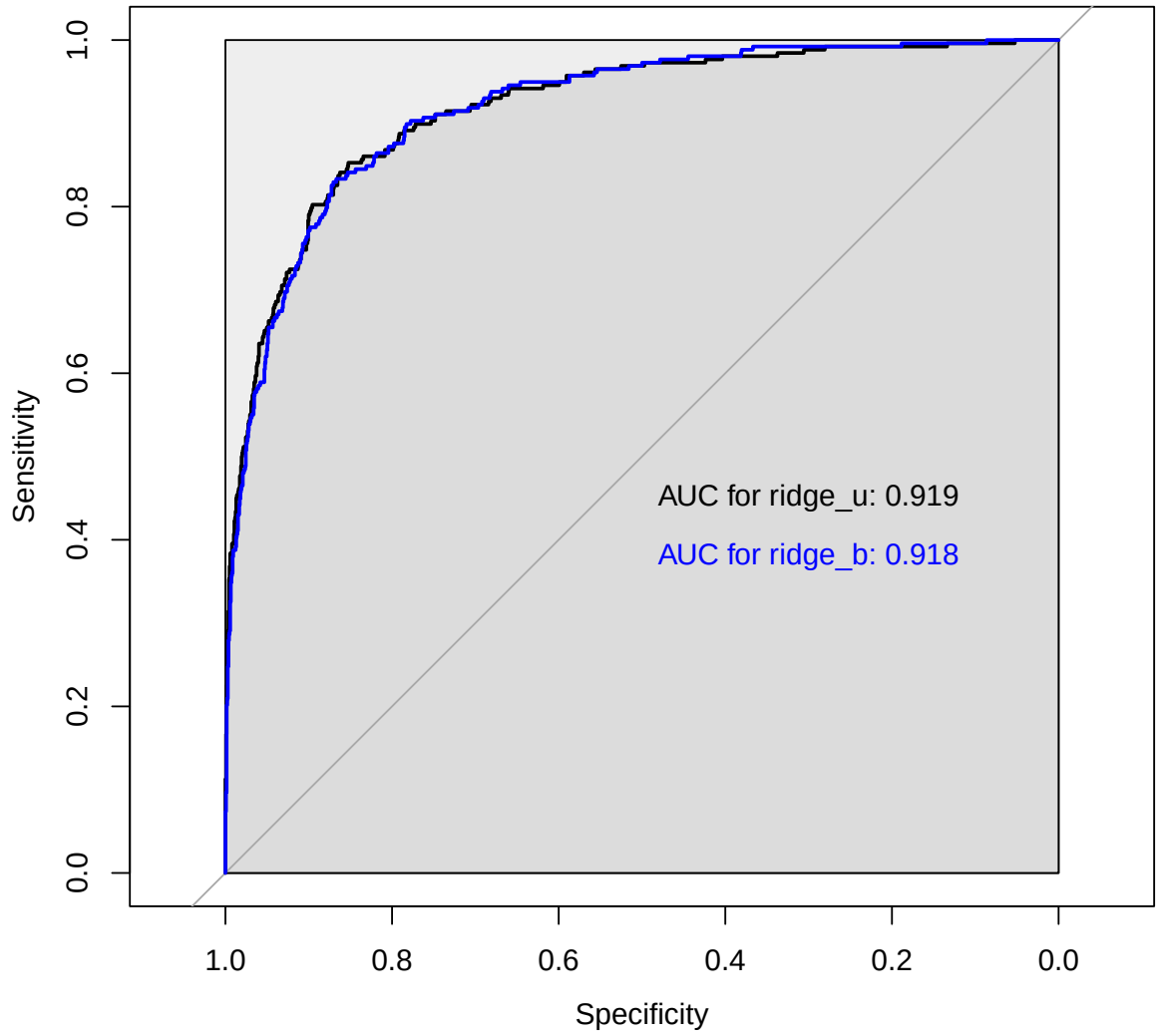
```
## [1] 0.744186
```

```r
#F1 Score

F1_ridge_b <- 2 * (Prec_ridge_b * Rec_ridge_b)/(Prec_ridge_b + Rec_ridge_b)
F1_ridge_b
```

```
## [1] 0.6643599
```

We plot the ROC curves and corresponding AUC values of the two models. Like the F1 score, we have the AUC values are much lower compared to the models obtained with the other techniques. It seems that adding the L2 norm as a penalization term leads to a worse model in terms of performance.

```
#ROC curves
roc_ridge_u <- roc(test$Attrition_Flag ~ as.numeric(ridge_predict_u))
roc_ridge_b <- roc(test$Attrition_Flag ~ as.numeric(ridge_predict_b))

plot(roc_ridge_u, col = "black",print.auc = FALSE, auc.polygon = TRUE,
     max.auc.polygon = TRUE, lwd=2)
plot(roc_ridge_b,add = TRUE,col = "blue", print.auc = FALSE, lwd=2)
text(0.3, 0.45, paste("AUC for ridge_u:", round(roc_ridge_u$auc, 3)), col = "black")
text(0.3, 0.38, paste("AUC for ridge_b:", round(roc_ridge_b$auc, 3)), col = "blue")
```



Lasso regression

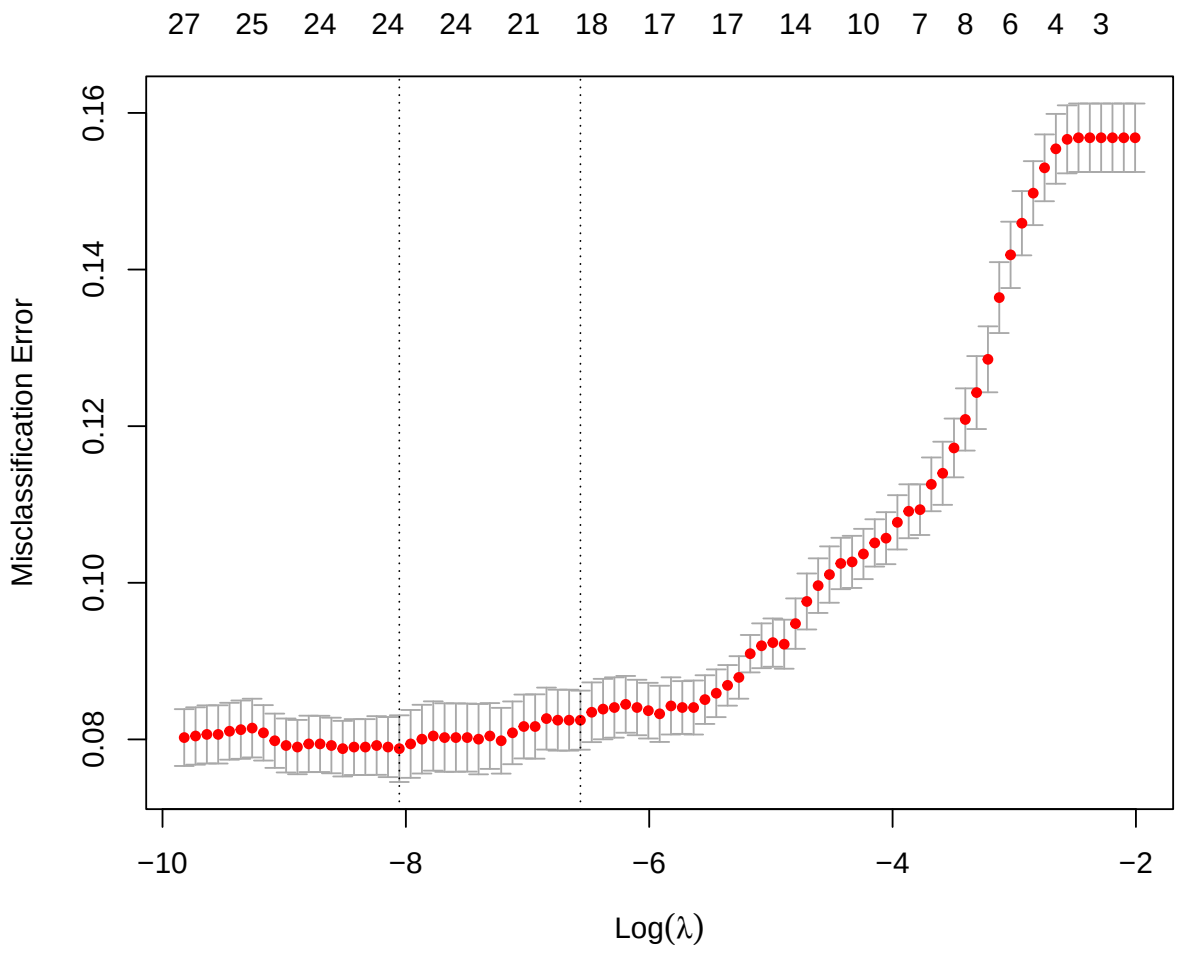
Lasso regression adds the L1 norm of the vector of the coefficients as a penalty term to the objective function. By doing so it shrinks the coefficients towards 0 and can set some coefficient exactly equal to 0. Thanks to this property, Lasso regression can identify and exclude the predictors which do not contribute to the predictions of the model. Because of this property we consider the set of all the predictor variables in addition to the interaction used in the GLM.

Unbalanced dataset

```
# Matrix without Attrition_Flag
train_mat <- data.matrix(train_int[,-c(1)])
test_mat <- data.matrix(test_int[,-c(1)])
```

Like for the Ridge models, we plot the misclassification error in function of the values of $\log(\lambda)$.

```
lasso_u <- cv.glmnet(train_mat,train$Attrition_Flag, alpha = 1,
                     family = "binomial", type.measure = "class")
plot(lasso_u)
```



We pick λ so that it minimizes the misclassification error.

```
lambda_lasso_u <- lasso_u$lambda.min
lambda_lasso_u
```

```
## [1] 0.0003178433
```

The best threshold for the F1 score is 0.4. Unlike with Ridge, the model's performance seems comparable to the models obtained with stepwise selection and discriminant analyis.

```
#Threshold
Threshold <- 0.4
lasso_predict_u <- predict(lasso_u,test_mat,type = "response", s = lambda_lasso_u)
pred_lasso_u <- ifelse(lasso_predict_u >= Threshold , 1,0)

#Confusion matrix
c_mat_lasso_u <- table(test$Attrition_Flag,pred_lasso_u)
```

	Predicted Values		
Real Values	0	1	Total
0	1330	60	1390
1	76	182	258
Total	1406	242	1648

```
#Accuracy

mean(pred_lasso_u==test$Attrition_Flag)*100
```

```
## [1] 91.92961
```

```
#True Negative Rate / Specificity

Spec_lasso_u <- c_mat_lasso_u[1,1]/sum(c_mat_lasso_u[1,])
Spec_lasso_u
```

```
## [1] 0.9597122
```

```
#Precision / Positive Predicted Value

Prec_lasso_u <- c_mat_lasso_u[2,2]/sum(c_mat_lasso_u[,2])
Prec_lasso_u
```

```
## [1] 0.7637131
```

```
#Recall / True Positive Rate / Sensitivity

Rec_lasso_u <- c_mat_lasso_u[2,2]/sum(c_mat_lasso_u[2,])
Rec_lasso_u
```

```
## [1] 0.7015504
```

```
#F1 Score

F1_lasso_u <- 2 * (Prec_lasso_u * Rec_lasso_u)/(Prec_lasso_u + Rec_lasso_u)
F1_lasso_u
```

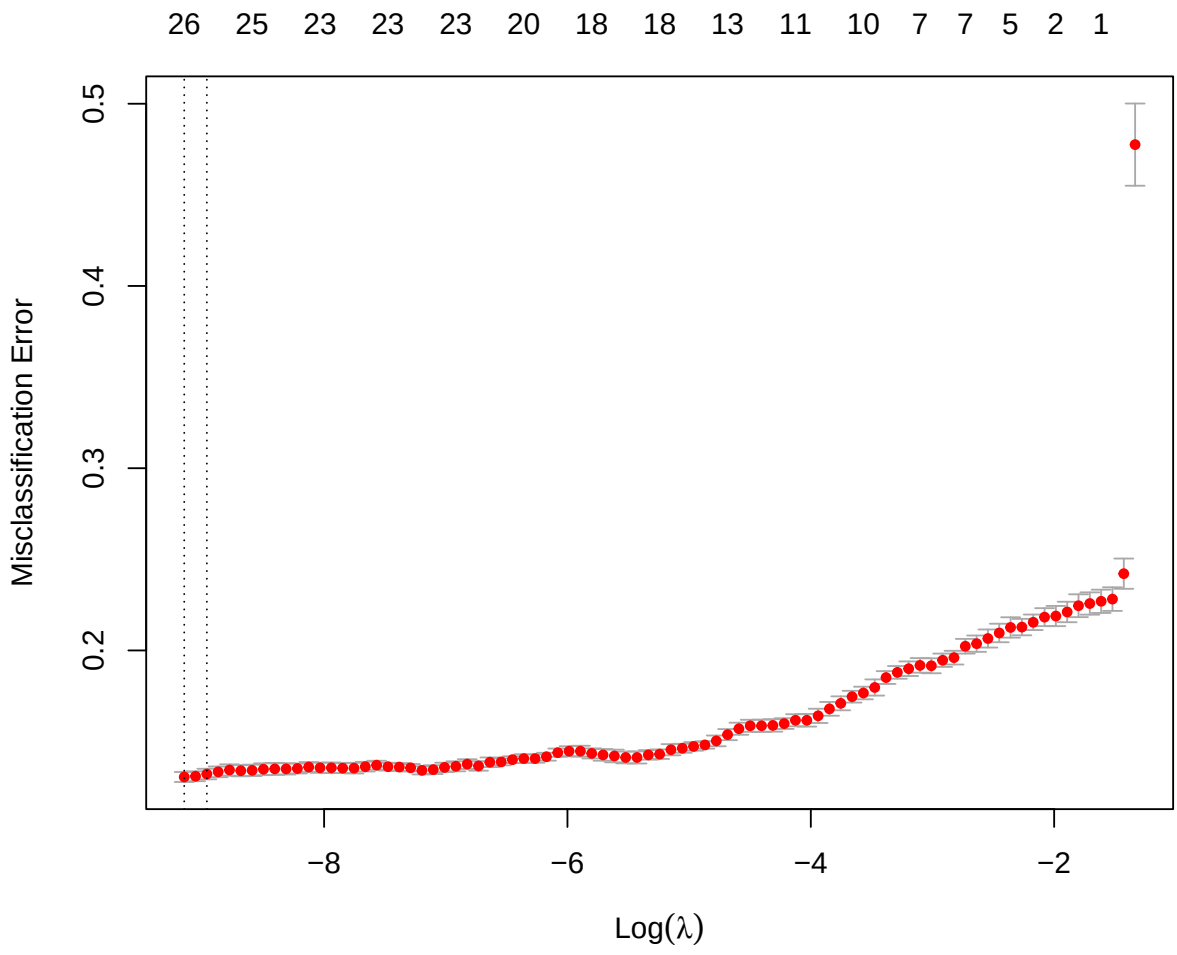
```
## [1] 0.7313131
```

Balanced dataset

```
# Matrix without Attrition_Flag
train_mat_b <- data.matrix(train_bal_int[,-c(1)])
test_mat_b <- data.matrix(test_bal_int[,-c(1)])
```

We shows the values of the misclassification error depending on $\log(\lambda)$.

```
lasso_b <- cv.glmnet(train_mat_b,train_bal$Attrition_Flag, alpha = 1,
                     family = "binomial", type.measure = "class")
plot(lasso_b)
```



We report the optimal lambda based on the minimum misclassification error that we will use to evaluate the model performance.

```
lambda_lasso_b <- lasso_b$lambda.min
lambda_lasso_b
```

```
## [1] 0.0001062579
```

After choosing the best threshold we compute some useful test for the model's performance. We have that the F1 score is better than the ones obtained with Ridge but lower than the other models.

```
#Threshold
Threshold <- 0.8
lasso_predict_b <- predict(lasso_b,test_mat_b,type = "response", s = lambda_lasso_b)
pred_lasso_b <- ifelse(lasso_predict_b >= Threshold , 1,0)

#Confusion matrix

c_mat_lasso_b <- table(test$Attrition_Flag,pred_lasso_b)
```

	Predicted Values		
Real Values	0	1	Total
0	1330	60	1390
1	86	172	258
Total	1416	232	1648

```
#Accuracy

mean(pred_lasso_b==test$Attrition_Flag)*100
```

```
## [1] 91.44417
```

```
#True Negative Rate / Specificity

Spec_lasso_b <- c_mat_lasso_b[1,1]/sum(c_mat_lasso_b[1,])
Spec_lasso_b
```

```
## [1] 0.9568345
```

```
#Precision / Positive Predicted Value

Prec_lasso_b <- c_mat_lasso_b[2,2]/sum(c_mat_lasso_b[,2])
Prec_lasso_b
```

```
## [1] 0.7468354
```

```
#Recall / True Positive Rate / Sensitivity

Rec_lasso_b <- c_mat_lasso_b[2,2]/sum(c_mat_lasso_b[2,])
Rec_lasso_b
```

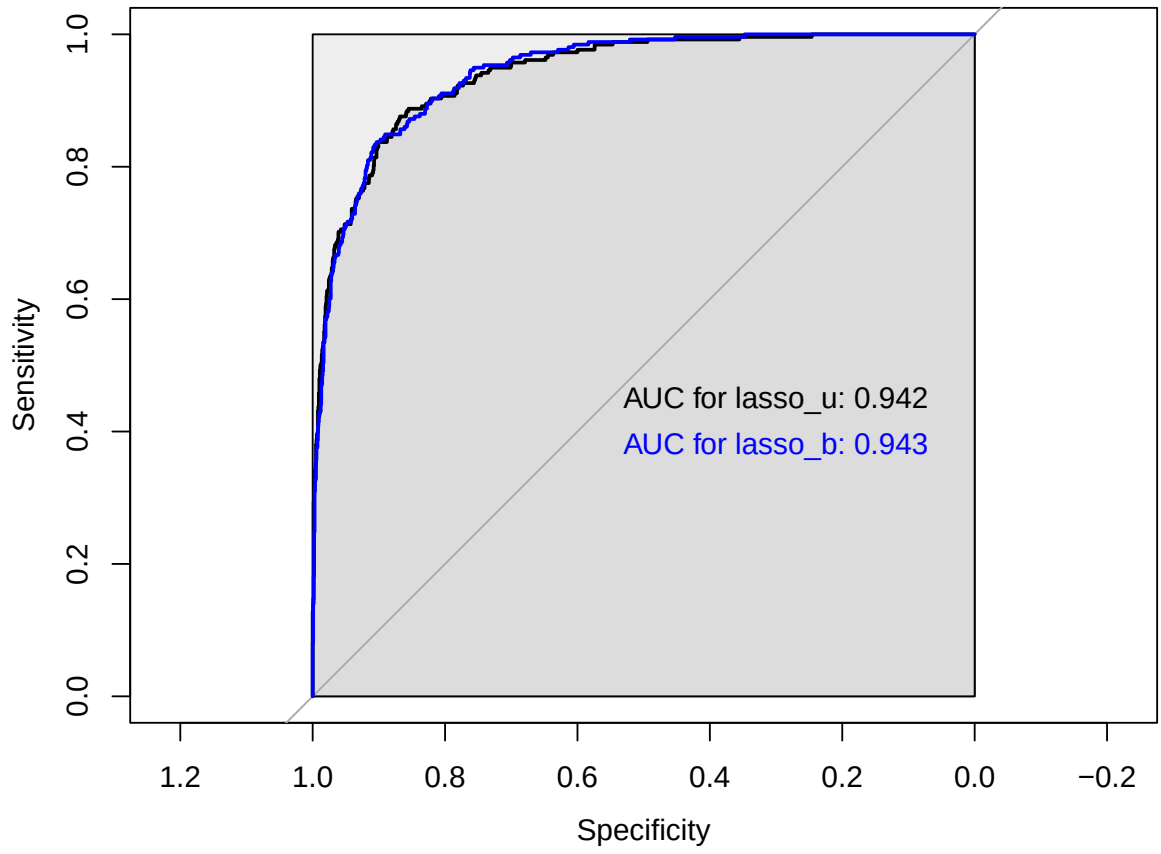
```
## [1] 0.6860465
```

```
#F1 Score

F1_lasso_b <- 2 * (Prec_lasso_b * Rec_lasso_b)/(Prec_lasso_b + Rec_lasso_b)
F1_lasso_b
```

```
## [1] 0.7151515
```

We plot the ROC curves and corresponding AUC values of the two models. We notice that the ROC curves and the AUC values are nearly equal, however the balanced model had a higher F1 score and is then a better choice for our problem.

```
#ROC curves
roc_lasso_u <- roc(test$Attrition_Flag ~ as.numeric(lasso_predict_u))
roc_lasso_b <- roc(test$Attrition_Flag ~ as.numeric(lasso_predict_b))

plot(roc_lasso_u, col = "black",print.auc = FALSE, auc.polygon = TRUE,
     max.auc.polygon = TRUE, lwd=2)
plot(roc_lasso_b,add = TRUE,col = "blue", print.auc = FALSE, lwd=2)
text(0.3, 0.45, paste("AUC for lasso_u:", round(roc_lasso_u$auc, 3)), col = "black")
text(0.3, 0.38, paste("AUC for lasso_b:", round(roc_lasso_b$auc, 3)), col = "blue")
```



Models comparison and conclusions

We are now going to summarize the results obtained with the different models. We report in two different tables the best F1 score, the corresponding Recall and the the AUC value associated with the models fitted in the unbalanced and balanced datasets.

Unbalanced dataset

We denote with *GLM_i* the GLM with all the independent variables considered and with *GLM_f* the GLM with the addiction of the interaction and the use of stepwise selection. We report the Recall of each value since, in case of models with similar performance, we prioritize identifying the Attriting Customer over penalizing Customers who weren't going to churn.

Model	*F1*	*Recall*	*AUC*
GLM_i	0.714	0.682	0.932
GLM_f	0.742	0.721	0.944
LDA	0.735	0.698	0.941
QDA	0.754	0.806	0.942
RIDGE	0.679	0.694	0.919
LASSO	0.731	0.701	0.942

We will consider as the best model the one obtained with QDA, since it has the highest F1 score and Recall, and the second highest AUC values. We also notice how Lasso regression leads to a model with the same AUC values as the QDA but lower F1 score. Moreover we can assert that Ridge regression is not a good choice for this dataset since its performance it's even worse than the complete GLM.

Balanced dataset

We apply the same notation used in the Unbalanced models.

Model	*F1*	*Recall*	*AUC*
GLM_i	0.698	0.659	0.933
GLM_f	0.727	0.783	0.944
LDA	0.732	0.705	0.946
QDA	0.744	0.779	0.944
RIDGE	0.664	0.744	0.918
LASSO	0.715	0.686	0.943

With the LDA, we obtained the model with the highest AUC value among all the models we analayzed. However its F1 score its considerably lower in comparison to the model built with QDA . For this reason, in addition to the high Recall and AUC values, we pick QDA as the best model fitted on the balanced dataset. We also note that, as in the unbalanced models, Ridge as the worst performance among the models analyzed.

Conclusions

Like we said at the beginning of the model's definition, we aim to discover customers who are going to churn without bothering too many customer who did not plan to leave the bank. To do so we have chosen the F1 score as the most significant measure of the model's performance. By comparing the two previous tables, we can see that the unbalanced models have a better F1 score compared to the balanced ones. So, even if the highest AUC values was obtained by fitting a model on the balanced dataset, we will pick the model built using QDA and fitted on the unbalanced training set.