**Project: Detecting credit defaulters using logistic regression in R**

### **Introduction**

The aim of this project is to analyze the credit card of clients in Taiwan in order to determine whether the client will default or not in the next month. This data is obtained from the UC Irvine dataset collection:

<https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients>

To make this prediction, Logistic regression analysis is performed. After trying to improve the performance of the logistic regression model, analysis using Random Forests is also performed to predict defaulters.

**Data set:**

Our credit dataset consists of 30,000 observations and 23 factors along with 1 factor for classification as potential defaulter for the next month or not.

This research employed a binary variable, default payment (Yes = 1, No = 0), as the response variable. This study reviewed the literature and used the following 23 variables as explanatory variables:

* X1: Amount of the given credit (NT dollar): it includes both the individual consumer credit and his/her family (supplementary) credit.
* X2: Gender (1 = male; 2 = female).
* X3: Education (1 = graduate school; 2 = university; 3 = high school; 4 = others).
* X4: Marital status (1 = married; 2 = single; 3 = others).
* X5: Age (year).

X6 - X11: History of past payment. The measurement scale for the repayment status is: -1 = pay duly; 1 = payment delay for one month; 2 = payment delay for two months; . . .; 8 = payment delay for eight months; 9 = payment delay for nine months and above.   
 Past monthly payment records (from April to September, 2005) as follows:

* X6 = the repayment status in September, 2005;
* X7 = the repayment status in August, 2005;
* X11 = the repayment status in April, 2005.

X12-X17: Amount of bill statement (NT dollar):

* X12 = amount of bill statement in September, 2005;
* X13 = amount of bill statement in August, 2005;

…

* X17 = amount of bill statement in April, 2005

X18-X23: Amount of previous payment (NT dollar):

* X18 = amount paid in September, 2005;
* X19 = amount paid in August, 2005;

…

* X23 = amount paid in April, 2005.

**Algorithm:**

A logistic regression model, models a **binary dependent variable**

Y=1or Yes

or

Y=0 or No

where

P(Y=1|X) is modelled in terms of the predictors X.

Since our dependent variable ‘default’ is a binary variable, we can use logistic regression to predict it.

As we are using more than one predictors or independent variables, our algorithm is actually **Multiple Logistic Regression**.

Although both Categorical or Numeric features can be used as predictors, our data consists of only numeric features.

### **Step 1 - collecting data**

This data is obtained from the UC Irvine dataset collection:

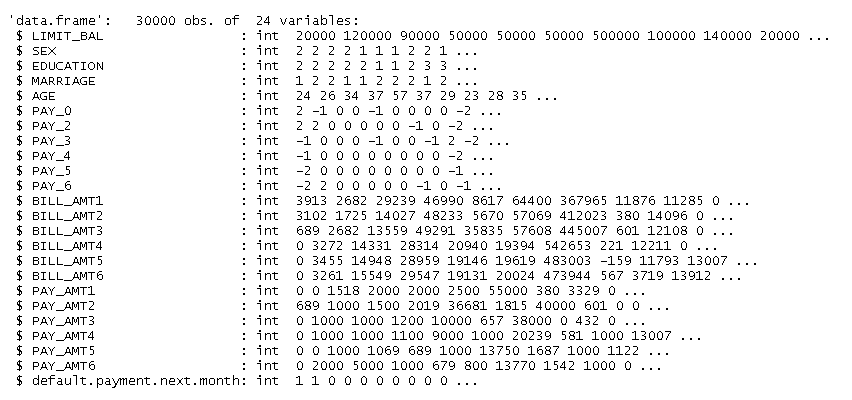
<https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients>

The data is in an excel file. The first row consists of pneumonic variables X1, X2 … X24, followed by a second row containing the actual variable names like LIMIT\_BAL, SEX… For ease of processing the data, I removed the first row and saved the data in csv format.

This csv file is read and saved into a data frame.

### **Step 2 - exploring and preparing the data**

Our credit dataset consists of 30000 observations and 23 numeric factors along with 1 factor for classification type.



* The data previously contained an additional column of ID variables. To avoid overfitting of the model, this column was removed from our dataset.
* The data consists of 6636 customers (22.12%) who are defaulters and 23364 customers who are not defaulters

prop.table(table(credit$default.payment.next.month))

##   
## 0 1   
## 0.7788 0.2212

* We split the data into training and testing datasets. The training dataset consists of 80% of the total samples and is used to train the logistic regression model

The remaining 20% data is used as test data and is used to evaluate the model to determine its accuracy

* After splitting the data, we confirm that the proportion of defaulters in the training and test datasets is the same and is also representative of the original dataset.

prop.table(table(credit\_train$default.payment.next.month))

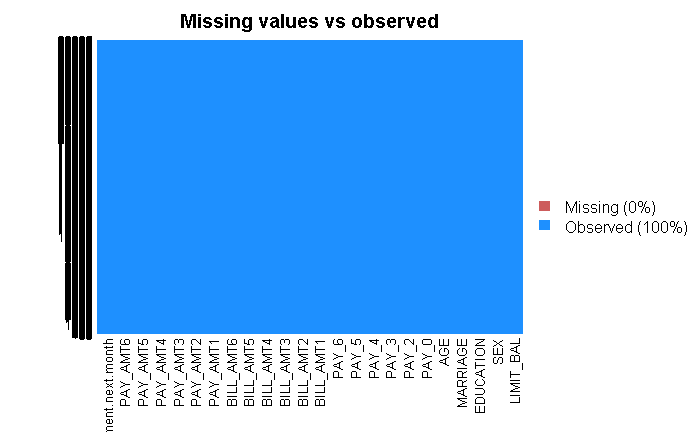
##   
## 0 1   
## 0.77625 0.22375

prop.table(table(credit\_test$default.payment.next.month))

##   
## 0 1   
## 0.789 0.211

* Using the Amelia package, we check whether dataset contains missing values.

Our dataset does not contain any missing values in any of the variables/ features.



* Next, we try to find out the number of original values in each feature, for example, there are 2 categories of Sex, 7 educational levels

#see number of unique values per feature  
sapply(credit, function(x) length(unique(x)))

## LIMIT\_BAL SEX   
## 81 2   
## EDUCATION MARRIAGE   
## 7 4   
## AGE PAY\_0   
## 56 11   
## PAY\_2 PAY\_3   
## 11 11   
## PAY\_4 PAY\_5   
## 11 10   
## PAY\_6 BILL\_AMT1   
## 10 22723   
## BILL\_AMT2 BILL\_AMT3   
## 22346 22026   
## BILL\_AMT4 BILL\_AMT5   
## 21548 21010   
## BILL\_AMT6 PAY\_AMT1   
## 20604 7943   
## PAY\_AMT2 PAY\_AMT3   
## 7899 7518   
## PAY\_AMT4 PAY\_AMT5   
## 6937 6897   
## PAY\_AMT6 default.payment.next.month   
## 6939 2

### **Step 3 - training a model on the data**

To train the logistic regression model, the glm() (Generalized linear models) function was used.

glm( ) function is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution.

The form of the **glm( )** function is:

**glm(***formula***, family=***familytype***(link=***linkfunction***), data=)**

The arguments used for our code are:

model <- glm(default.payment.next.month ~.,family=binomial(link='logit'),data=credit\_train)  
model

##   
## Call: glm(formula = default.payment.next.month ~ ., family = binomial(link = "logit"),   
## data = credit\_train)

### **Step 4 - evaluating model/algorithm performance**

To evaluate the performance of the algorithm, we summarize the model parameters.

Of the 23 features used to predict the dependent variable, not all of them have a strong association with the dependent variable. The last column in the following table represents ‘p-values’. The lower the p-value, the stronger the association between that feature and the independent variable. The p-values are also marked by their respective significant levels as indicated in the index at the bottom of the table.

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -6.039e-01 1.331e-01 -4.536 5.73e-06 \*\*\*

LIMIT\_BAL -6.322e-07 1.751e-07 -3.611 0.000305 \*\*\*

SEX -1.018e-01 3.454e-02 -2.948 0.003200 \*\*

EDUCATION -1.008e-01 2.365e-02 -4.260 2.05e-05 \*\*\*

MARRIAGE -1.678e-01 3.525e-02 -4.759 1.95e-06 \*\*\*

AGE 5.434e-03 1.974e-03 2.752 0.005917 \*\*

PAY\_0 5.693e-01 1.956e-02 29.103 < 2e-16 \*\*\*

PAY\_2 8.061e-02 2.235e-02 3.606 0.000311 \*\*\*

PAY\_3 7.451e-02 2.505e-02 2.974 0.002936 \*\*

PAY\_4 -1.238e-02 2.823e-02 -0.438 0.661054

PAY\_5 7.286e-02 2.969e-02 2.454 0.014130 \*

PAY\_6 4.358e-03 2.426e-02 0.180 0.857439

BILL\_AMT1 -6.937e-06 1.330e-06 -5.215 1.84e-07 \*\*\*

BILL\_AMT2 3.438e-06 1.707e-06 2.015 0.043934 \*

BILL\_AMT3 9.539e-07 1.514e-06 0.630 0.528613

BILL\_AMT4 7.023e-07 1.570e-06 0.447 0.654622

BILL\_AMT5 1.165e-06 1.717e-06 0.678 0.497466

BILL\_AMT6 -3.544e-07 1.305e-06 -0.272 0.785883

PAY\_AMT1 -1.516e-05 2.637e-06 -5.749 8.96e-09 \*\*\*

PAY\_AMT2 -7.874e-06 2.174e-06 -3.622 0.000292 \*\*\*

PAY\_AMT3 -4.891e-06 2.087e-06 -2.343 0.019124 \*

PAY\_AMT4 -3.277e-06 1.998e-06 -1.640 0.101091

PAY\_AMT5 -1.726e-06 1.890e-06 -0.913 0.361060

PAY\_AMT6 -3.307e-06 1.522e-06 -2.172 0.029839 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

From this, we see that the most significant factors (\*\*\*) that affect the dependent variable ‘default.payment.next.month’ are

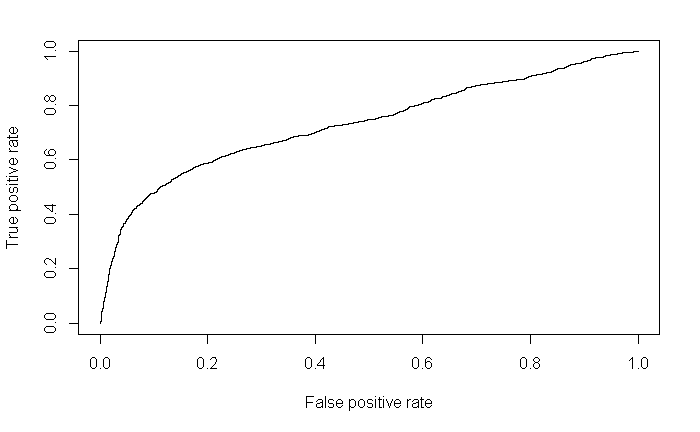
1. LIMIT\_BAL- Amount of the given credit (NT dollar): it includes both the individual consumer credit and his/her family (supplementary) credit.
2. EDUCATION-(1 = graduate school; 2 = university; 3 = high school; 4 = others).
3. MARRIAGE -(1 = married; 2 = single; 3 = others).
4. PAY\_0 - the repayment status in September, 2005;
5. PAY\_2 - the repayment status in August, 2005;
6. BILL\_AMT1- amount of bill statement (NT dollar) in August, 2005
7. PAY\_AMT1 - Amount of previous payment (NT dollar) paid in August, 2005;
8. PAY\_AMT2 - Amount of previous payment (NT dollar) paid in July, 2005

We first check the accuracy of its predictions on the training data with which the model was formed.

#On testing the model on training data, we get 99.45% accuracy

After this, we run the model on the test dataset.

#On using the test data, we get an accuracy of 82.75%



From the ROC Curve above, we get a value of 0.7323 via the AUC function

### **Step 5 - improving model/algorithm performance**

The earlier model made use of all 23 features to form its predictions. Not all of them were significantly related to the dependent variable. To improve our model, we drop the insignificant predictors by specifying an alpha value of 0.10.

The new model will now use only 15 features:

1. LIMIT\_BAL
2. SEX
3. EDUCATION
4. MARRIAGE
5. AGE
6. PAY\_0
7. PAY\_2
8. PAY\_3
9. PAY\_5
10. BILL\_AMT1
11. BILL\_AMT2
12. PAY\_AMT1
13. PAY\_AMT2
14. PAY\_AMT3
15. PAY\_AMT6

The modified code for model formation is:

model2 <- glm(default.payment.next.month ~ LIMIT\_BAL +SEX+ EDUCATION+ MARRIAGE + AGE + PAY\_0 +PAY\_2+ PAY\_3 + PAY\_5 + BILL\_AMT1 +BILL\_AMT2 + PAY\_AMT1 + PAY\_AMT2+ PAY\_AMT3+ PAY\_AMT6 ,family=binomial(link='logit'),data=credit\_train)

As seen in the table below, all features used in model2 have significant p-values:-

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -6.064e-01 1.330e-01 -4.559 5.14e-06 \*\*\*

LIMIT\_BAL -6.519e-07 1.721e-07 -3.788 0.000152 \*\*\*

SEX -9.878e-02 3.450e-02 -2.863 0.004195 \*\*

EDUCATION -1.029e-01 2.363e-02 -4.357 1.32e-05 \*\*\*

MARRIAGE -1.700e-01 3.522e-02 -4.827 1.39e-06 \*\*\*

AGE 5.465e-03 1.974e-03 2.769 0.005617 \*\*

PAY\_0 5.723e-01 1.952e-02 29.322 < 2e-16 \*\*\*

PAY\_2 7.816e-02 2.236e-02 3.496 0.000473 \*\*\*

PAY\_3 7.033e-02 2.260e-02 3.113 0.001854 \*\*

PAY\_5 8.034e-02 1.950e-02 4.121 3.78e-05 \*\*\*

BILL\_AMT1 -7.113e-06 1.304e-06 -5.453 4.96e-08 \*\*\*

BILL\_AMT2 5.536e-06 1.355e-06 4.084 4.43e-05 \*\*\*

PAY\_AMT1 -1.633e-05 2.638e-06 -6.191 5.99e-10 \*\*\*

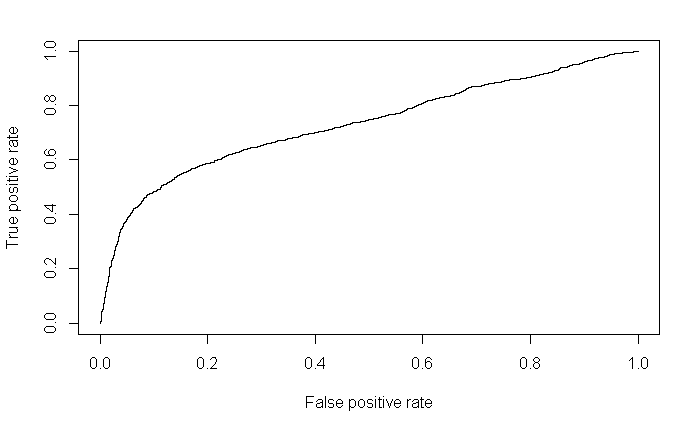
PAY\_AMT2 -6.712e-06 1.918e-06 -3.500 0.000465 \*\*\*

PAY\_AMT3 -4.278e-06 1.832e-06 -2.336 0.019495 \*

PAY\_AMT6 -3.540e-06 1.502e-06 -2.357 0.018423 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1



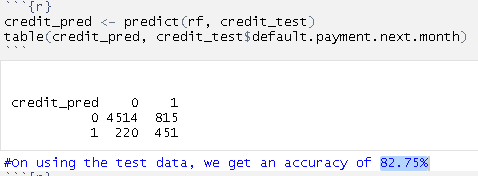
After improving our model, our new accuracy is 81.97%. From the ROC Curve we have a value of 0.7317 via the AUC function.

This is a slight improvement than the earlier model.

### **Using Random Forest**

I also tried using the Random Forest Algorithm on this data to perform the predictions. Using default settings, the accuracy rate on the test data was 82.75%

However, we note 815 clients who were not actual defaulters were classified as defaulters by the model. This can prove costly for the company as they will end up losing potential valuable clients.



Due to system performance issues, I was unable to run the code for auto-tuning the Random Forest using Caret.

The notebook is present in the zip folder and code is added to the appendix section for reference.

### **Conclusion:**

The most significant factors that determine whether a client will default payment next month are-

1. **LIMIT\_BAL**- Amount of the given credit (NT dollar): it includes both the individual consumer credit and his/her family (supplementary) credit.
2. **EDUCATION**-(1 = graduate school; 2 = university; 3 = high school; 4 = others).
3. **MARRIAGE** -(1 = married; 2 = single; 3 = others).
4. **PAY\_0** - the repayment status in September, 2005;
5. **PAY\_2** - the repayment status in August, 2005;
6. **BILL\_AMT1**- amount of bill statement (NT dollar) in August, 2005
7. **PAY\_AMT1** - Amount of previous payment (NT dollar) paid in August, 2005;
8. **PAY\_AMT2** - Amount of previous payment (NT dollar) paid in July, 2005

|  |  |  |
| --- | --- | --- |
| **Algorithm** | **Accuracy (%)** | **AUC (Area Under the Curve)** |
| Multiple logistic regression using all 23 features | 81.9166666666667 | 0.7323 |
| Multiple logistic regression using 15 significant features with alpha >=0.01 | 81.9666666666667 | 0.7317 |
| Random Forest with default settings and all features | 82.75 |  |

The best accuracy amongst the three algorithms was given by Random Forests.

### **Appendix:**

Detecting credit defaulters using logistic regression

# Logistic Regression analysis of the credit data

#https://archive.ics.uci.edu/ml/machine-learning-databases/00350/default%20of%20credit%20card%20clients.xls  
#Step 1: Collect data  
credit <- read.csv("default\_of\_credit\_card\_clients\_1.csv")  
  
#In the csv file used, we discard column names X1, X2... to more meaningful names given in row2- LIMIT\_BAL, SEX… present in the second row  
#remove the ID variable to avoid overfitting and remove factor names additionally present in row  
credit<-credit[,-1]

#Step 2: exploring the data  
str(credit)  
table(credit$default.payment.next.month)  
prop.table(table(credit$default.payment.next.month))

# Our credit dataset consists of 30000 observations and 23 factors along with 1 factor for classification type.

# 6636 customers (22.12%) are defaulters and 23364 customers (77.88%) are not defaulters

#preparing the data:  
  
#preparing test and train data 80%-20%  
credit\_train <- credit[1:24000,]  
credit\_test <- credit[24001:30000,]  
  
credit\_train\_labels <- credit[1:24000,]$default.payment.next.month  
credit\_test\_labels <- credit[24001:30000,]$default.payment.next.month  
  
prop.table(table(credit\_train$default.payment.next.month))  
prop.table(table(credit\_test$default.payment.next.month))

# The proportion od defaulters in training and test datasets is almost the same

# We now check our data for any missing values

library(Amelia)  
missmap(credit, main = "Missing values vs observed")  
sapply(credit,function(x) sum(is.na(x)))

# From the plot and the table, we see that there are no missing values in our data

#see number of unique values per feature  
sapply(credit, function(x) length(unique(x)))

#Step 3: Training the Model  
model <- glm(default.payment.next.month ~.,family=binomial(link='logit'),data=credit\_train)  
model

summary(model)

anova(model, test="Chisq")

#Step 4: Evaluate the model  
fitted.results <- predict(model,newdata=credit\_test,type='response')  
fitted.results <- ifelse(fitted.results > 0.5,1,0)  
  
misClasificError <- mean(fitted.results != credit\_test$default.payment.next.month)  
print(paste('Accuracy',1-misClasificError))

# The accuracy of our model is 81.92% (Misclassification rate is 18.08%)

library(ROCR)  
p <- predict(model, newdata=credit\_test, type="response")  
pr <- prediction(p, credit\_test$default.payment.next.month)  
prf <- performance(pr, measure = "tpr", x.measure = "fpr")  
plot(prf)  
auc <- performance(pr, measure = "auc")  
auc <- auc@y.values[[1]]  
auc

# From the ROC Curve we have a value of 0.7323 via the AUC function

# Step 5: Improving the Model

# To improve our model, we drop the insignificant predictors by specifying an alpha value of 0.10 The new model will now use the features:

model2 <- glm(default.payment.next.month ~ LIMIT\_BAL +SEX+ EDUCATION+ MARRIAGE + AGE + PAY\_0 +PAY\_2+ PAY\_3 + PAY\_5 + BILL\_AMT1 +BILL\_AMT2 + PAY\_AMT1 + PAY\_AMT2+ PAY\_AMT3+ PAY\_AMT6 ,family=binomial(link='logit'),data=credit\_train)  
model2

summary(model2)

fitted.results2 <- predict(model2,newdata=credit\_test,type='response')  
fitted.results2 <- ifelse(fitted.results2 > 0.5,1,0)  
misClasificError2 <- mean(fitted.results2 != credit\_test$default.payment.next.month)  
print(paste('Accuracy',1-misClasificError2))  
  
p2 <- predict(model2, newdata=credit\_test, type="response")  
pr2 <- prediction(p2, credit\_test$default.payment.next.month)  
prf2 <- performance(pr2, measure = "tpr", x.measure = "fpr")  
plot(prf2)  
auc1 <- performance(pr2, measure = "auc")  
auc1 <- auc1@y.values[[1]]  
auc1

# The improved model now gives an Accuracy of 81.97%. From the ROC Curve we have a value of 0.7317 via the AUC function.

This is a slight improvement.

**Part2: Using Random Forest**

Detecting credit defaulters using random forest

# Random Forest analysis of the credit data

#https://archive.ics.uci.edu/ml/machine-learning-databases/00350/default%20of%20credit%20card%20clients.xls  
#Step 1: Collect data  
credit <- read.csv("default\_of\_credit\_card\_clients\_1.csv")

# In the csv file used, we discard column names X1, X2… to more meaningful names given in row2- LIMIT\_BAL, SEX… present in the second row

# remove the ID variable to avoid overfitting and remove factor names additionally present in row

credit<-credit[,-1] ```

#Step 2: exploring the data  
str(credit)  
table(credit$default.payment.next.month)  
prop.table(table(credit$default.payment.next.month))

#preparing the data:  
  
#preparing test and train data 80%-20%  
credit\_train <- credit[1:24000,]  
credit\_test <- credit[24001:30000,]  
  
credit\_train\_labels <- credit[1:24000,]$default.payment.next.month  
credit\_test\_labels <- credit[24001:30000,]$default.payment.next.month

#Step 3: Training the model  
library(randomForest)  
credit\_rf <-credit  
credit\_rf$default.payment.next.month <- as.factor(credit\_rf$default.payment.next.month)  
#preparing test and train data 80%-20%  
credit\_train <- credit\_rf[1:24000,]  
credit\_test <- credit\_rf[24001:30000,]  
  
credit\_train\_labels <- credit\_rf[1:24000,]$default.payment.next.month  
credit\_test\_labels <- credit\_rf[24001:30000,]$default.payment.next.month  
  
## Using the random forest library we create a forest of trees based on the training data  
set.seed(123)  
rf <- randomForest(default.payment.next.month ~ ., data = credit\_train , mtry = 5)

#Step 4: Evaluating model performance  
pred\_train <- predict(rf,credit\_train)  
table(pred\_train, credit\_train$default.payment.next.month)

# On testing the model on training data, we get 99.45% accuracy

credit\_pred <- predict(rf, credit\_test)  
table(credit\_pred, credit\_test$default.payment.next.month)

# On using the test data, we get an accuracy of 82.75%

#Step 5: Improving the model  
library(caret)  
ctrl <- trainControl(method = "repeatedcv",  
 number = 4, repeats = 4)  
# auto-tune a random forest  
grid\_rf <- expand.grid(.mtry = c(2, 4, 8))  
  
set.seed(123)  
m\_rf <- train(default.payment.next.month ~ ., data = credit\_train, method = "rf",  
 metric = "Kappa", trControl = ctrl,  
 tuneGrid = grid\_rf)  
pred\_train <- predict(rf,credit\_train)  
table(pred\_train, credit\_train$default.payment.next.month)

credit\_pred <- predict(rf, credit\_test)  
table(credit\_pred, credit\_test$default.payment.next.month)