ASDS-6303 Final Project

Default of Credit Card Clients

Phuong Trinh, Kyra Stolarski

2025-12-10

Load Packages

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(caret)
```

```
## Loading required package: lattice
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(rpart)
library(rpart.plot)
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```r
library(randomForest)
```

```
## randomForest 4.7-1.2

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:dplyr':
##
##     combine
```

Load Dataset and set seed for reproducibility

```r
set.seed(7)

credit <- read.csv("C:/Users/stolarskikm/Downloads/default+of+credit+card+clients/default of credit card
                   header = FALSE,
                   stringsAsFactors = FALSE)

head(credit)
```

```
##   V1        V2  V3        V4       V5  V6    V7    V8    V9   V10   V11   V12
## 1           X1  X2        X3       X4  X5    X6    X7    X8    X9   X10   X11
## 2 ID LIMIT_BAL SEX EDUCATION MARRIAGE AGE PAY_0 PAY_2 PAY_3 PAY_4 PAY_5 PAY_6
## 3  1     20000   2         2        1  24     2     2    -1    -1    -2    -2
## 4  2    120000   2         2        2  26    -1     2     0     0     0     2
## 5  3     90000   2         2        2  34     0     0     0     0     0     0
## 6  4     50000   2         2        1  37     0     0     0     0     0     0
##         V13       V14       V15       V16       V17       V18      V19      V20
## 1       X12       X13       X14       X15       X16       X17      X18      X19
## 2 BILL_AMT1 BILL_AMT2 BILL_AMT3 BILL_AMT4 BILL_AMT5 BILL_AMT6 PAY_AMT1 PAY_AMT2
## 3      3913      3102       689         0         0         0        0      689
## 4      2682      1725      2682      3272      3455      3261        0     1000
## 5     29239     14027     13559     14331     14948     15549     1518     1500
## 6     46990     48233     49291     28314     28959     29547     2000     2019
##        V21      V22      V23      V24                        V25
## 1      X20      X21      X22      X23                          Y
## 2 PAY_AMT3 PAY_AMT4 PAY_AMT5 PAY_AMT6 default payment next month
## 3        0        0        0        0                          1
## 4     1000     1000        0     2000                          1
## 5     1000     1000     1000     5000                          0
## 6     1200     1100     1069     1000                          0
```

Data Cleaning and Preprocessing

```r
# Fix the header row
# Set first row as column names
colnames(credit) <- credit[2, ]

# Remove the first 2 rows
credit <- credit[-c(1,2), ]
head(credit)
```

```
##   ID LIMIT_BAL SEX EDUCATION MARRIAGE AGE PAY_0 PAY_2 PAY_3 PAY_4 PAY_5 PAY_6
## 3  1     20000   2         2        1  24     2     2    -1    -1    -2    -2
## 4  2    120000   2         2        2  26    -1     2     0     0     0     2
## 5  3     90000   2         2        2  34     0     0     0     0     0     0
## 6  4     50000   2         2        1  37     0     0     0     0     0     0
## 7  5     50000   1         2        1  57    -1     0    -1     0     0     0
## 8  6     50000   1         1        2  37     0     0     0     0     0     0
##   BILL_AMT1 BILL_AMT2 BILL_AMT3 BILL_AMT4 BILL_AMT5 BILL_AMT6 PAY_AMT1 PAY_AMT2
## 3      3913      3102       689         0         0         0        0      689
## 4      2682      1725      2682      3272      3455      3261        0     1000
## 5     29239     14027     13559     14331     14948     15549     1518     1500
## 6     46990     48233     49291     28314     28959     29547     2000     2019
## 7      8617      5670     35835     20940     19146     19131     2000    36681
## 8     64400     57069     57608     19394     19619     20024     2500     1815
##   PAY_AMT3 PAY_AMT4 PAY_AMT5 PAY_AMT6 default payment next month
## 3        0        0        0        0                          1
## 4     1000     1000        0     2000                          1
## 5     1000     1000     1000     5000                          0
## 6     1200     1100     1069     1000                          0
## 7    10000     9000      689      679                          0
## 8      657     1000     1000      800                          0
```

```r
# Convert all character columns to numeric
credit <- credit %>% mutate(across(where(is.character), readr::parse_number))

# Sanity check
str(credit)
```

```
## 'data.frame':    30000 obs. of  25 variables:
##  $ ID                        : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ LIMIT_BAL                 : num  20000 120000 90000 50000 50000 50000 500000 100000 140000 20000
##  $ SEX                       : num  2 2 2 2 1 1 1 2 2 1 ...
##  $ EDUCATION                 : num  2 2 2 2 2 1 1 2 3 3 ...
##  $ MARRIAGE                  : num  1 2 2 1 1 2 2 2 1 2 ...
##  $ AGE                       : num  24 26 34 37 57 37 29 23 28 35 ...
##  $ PAY_0                     : num  2 -1 0 0 -1 0 0 0 0 -2 ...
##  $ PAY_2                     : num  2 2 0 0 0 0 0 -1 0 -2 ...
##  $ PAY_3                     : num  -1 0 0 0 -1 0 0 -1 2 -2 ...
##  $ PAY_4                     : num  -1 0 0 0 0 0 0 0 0 -2 ...
##  $ PAY_5                     : num  -2 0 0 0 0 0 0 0 0 -1 ...
##  $ PAY_6                     : num  -2 2 0 0 0 0 0 -1 0 -1 ...
##  $ BILL_AMT1                 : num  3913 2682 29239 46990 8617 ...
```

```
##  $ BILL_AMT2                 : num  3102 1725 14027 48233 5670 ...
##  $ BILL_AMT3                 : num  689 2682 13559 49291 35835 ...
##  $ BILL_AMT4                 : num  0 3272 14331 28314 20940 ...
##  $ BILL_AMT5                 : num  0 3455 14948 28959 19146 ...
##  $ BILL_AMT6                 : num  0 3261 15549 29547 19131 ...
##  $ PAY_AMT1                  : num  0 0 1518 2000 2000 ...
##  $ PAY_AMT2                  : num  689 1000 1500 2019 36681 ...
##  $ PAY_AMT3                  : num  0 1000 1000 1200 10000 657 38000 0 432 0 ...
##  $ PAY_AMT4                  : num  0 1000 1000 1100 9000 ...
##  $ PAY_AMT5                  : num  0 0 1000 1069 689 ...
##  $ PAY_AMT6                  : num  0 2000 5000 1000 679 ...
##  $ default payment next month: num  1 1 0 0 0 0 0 0 0 0 ...
```

```r
head(credit)
```

```
##   ID LIMIT_BAL SEX EDUCATION MARRIAGE AGE PAY_0 PAY_2 PAY_3 PAY_4 PAY_5 PAY_6
## 3  1     20000   2         2        1  24     2     2    -1    -1    -2    -2
## 4  2    120000   2         2        2  26    -1     2     0     0     0     2
## 5  3     90000   2         2        2  34     0     0     0     0     0     0
## 6  4     50000   2         2        1  37     0     0     0     0     0     0
## 7  5     50000   1         2        1  57    -1     0    -1     0     0     0
## 8  6     50000   1         1        2  37     0     0     0     0     0     0
##   BILL_AMT1 BILL_AMT2 BILL_AMT3 BILL_AMT4 BILL_AMT5 BILL_AMT6 PAY_AMT1 PAY_AMT2
## 3      3913      3102       689         0         0         0        0      689
## 4      2682      1725      2682      3272      3455      3261        0     1000
## 5     29239     14027     13559     14331     14948     15549     1518     1500
## 6     46990     48233     49291     28314     28959     29547     2000     2019
## 7      8617      5670     35835     20940     19146     19131     2000    36681
## 8     64400     57069     57608     19394     19619     20024     2500     1815
##   PAY_AMT3 PAY_AMT4 PAY_AMT5 PAY_AMT6 default payment next month
## 3        0        0        0        0                          1
## 4     1000     1000        0     2000                          1
## 5     1000     1000     1000     5000                          0
## 6     1200     1100     1069     1000                          0
## 7    10000     9000      689      679                          0
## 8      657     1000     1000      800                          0
```

```r
# Removing ID Column
credit <- select(credit, -ID)

# Cleaning column names
names(credit) <- tolower(names(credit))
names(credit)[names(credit) == 'default payment next month'] <- 'default_next_month'
names(credit)[names(credit) == 'pay_0'] <- 'pay_1'
head(credit)
```

```
##   limit_bal sex education marriage age pay_1 pay_2 pay_3 pay_4 pay_5 pay_6
## 3     20000   2         2        1  24     2     2    -1    -1    -2    -2
## 4    120000   2         2        2  26    -1     2     0     0     0     2
## 5     90000   2         2        2  34     0     0     0     0     0     0
## 6     50000   2         2        1  37     0     0     0     0     0     0
## 7     50000   1         2        1  57    -1     0    -1     0     0     0
## 8     50000   1         1        2  37     0     0     0     0     0     0
```

```
##   bill_amt1 bill_amt2 bill_amt3 bill_amt4 bill_amt5 bill_amt6 pay_amt1 pay_amt2
## 3      3913      3102       689         0         0         0        0      689
## 4      2682      1725      2682      3272      3455      3261        0     1000
## 5     29239     14027     13559     14331     14948     15549     1518     1500
## 6     46990     48233     49291     28314     28959     29547     2000     2019
## 7      8617      5670     35835     20940     19146     19131     2000    36681
## 8     64400     57069     57608     19394     19619     20024     2500     1815
##   pay_amt3 pay_amt4 pay_amt5 pay_amt6 default_next_month
## 3        0        0        0        0                  1
## 4     1000     1000        0     2000                  1
## 5     1000     1000     1000     5000                  0
## 6     1200     1100     1069     1000                  0
## 7    10000     9000      689      679                  0
## 8      657     1000     1000      800                  0
```

```r
# Re-coding categorical attributes and target

# SEX: 1=Male, 2=Female
credit <- credit %>%
  mutate(sex = factor(sex, levels = c(1,2), labels = c("Male","Female")))

# EDUCATION: fix odd codes to "Others"
credit <- credit %>%
  mutate(education = dplyr::recode(education,
                                   `0`="Others", `1`="GradSchool", `2`="University",
                                   `3`="HighSchool", `4`="Others", `5`="Others", `6`="Others"),
         education = factor(education))

# MARRIAGE: 1=Married, 2=Single, others -> "Others"
credit <- credit %>%
  mutate(marriage = dplyr::recode(marriage,
                                  `0`="Others", `1`="Married", `2`="Single", `3`="Others"),
         marriage = factor(marriage))

# PAY_0..PAY_6 as ordered factors (payment status codes)
pay_cols <- c("pay_1","pay_2","pay_3","pay_4","pay_5","pay_6")
credit <- credit %>%
  mutate(across(all_of(pay_cols), ~factor(., ordered = TRUE)))

# Target Y to factor (0=No Default, 1=Default)
credit <- credit %>%
  mutate(default_next_month =
           factor(default_next_month, levels = c(0,1),
                  labels = c("NoDefault","Default"))) %>%
  rename(y = default_next_month)


str(credit[, c("sex","education","marriage",pay_cols,"y")])
```

```
## 'data.frame':    30000 obs. of  10 variables:
##  $ sex      : Factor w/ 2 levels "Male","Female": 2 2 2 2 1 1 1 2 2 1 ...
##  $ education: Factor w/ 4 levels "GradSchool","HighSchool",..: 4 4 4 4 4 1 1 4 2 2 ...
##  $ marriage : Factor w/ 3 levels "Married","Others",..: 1 3 3 1 1 3 3 3 1 3 ...
##  $ pay_1    : Ord.factor w/ 11 levels "-2"<"-1"<"0"<..: 5 2 3 3 2 3 3 3 3 1 ...
```

```
##  $ pay_2    : Ord.factor w/ 11 levels "-2"<"-1"<"0"<..: 5 5 3 3 3 3 3 2 3 1 ...
##  $ pay_3    : Ord.factor w/ 11 levels "-2"<"-1"<"0"<..: 2 3 3 3 2 3 3 2 5 1 ...
##  $ pay_4    : Ord.factor w/ 11 levels "-2"<"-1"<"0"<..: 2 3 3 3 3 3 3 3 3 1 ...
##  $ pay_5    : Ord.factor w/ 10 levels "-2"<"-1"<"0"<..: 1 3 3 3 3 3 3 3 3 2 ...
##  $ pay_6    : Ord.factor w/ 10 levels "-2"<"-1"<"0"<..: 1 4 3 3 3 3 3 2 3 2 ...
##  $ y        : Factor w/ 2 levels "NoDefault","Default": 2 2 1 1 1 1 1 1 1 1 ...
```

```
# Missing values per column
colSums(is.na(credit))
```

```
## limit_bal       sex education  marriage       age     pay_1     pay_2     pay_3
##         0         0         0         0         0         0         0         0
##     pay_4     pay_5     pay_6 bill_amt1 bill_amt2 bill_amt3 bill_amt4 bill_amt5
##         0         0         0         0         0         0         0         0
## bill_amt6  pay_amt1  pay_amt2  pay_amt3  pay_amt4  pay_amt5  pay_amt6         y
##         0         0         0         0         0         0         0         0
```

Descriptive Statistics

```
summary(credit)
```

```
##    limit_bal          sex             education        marriage
##  Min.   :  10000   Male  :11888   GradSchool:10585   Married:13659
##  1st Qu.:  50000   Female:18112   HighSchool: 4917   Others :  377
##  Median : 140000                  Others    :  468   Single :15964
##  Mean   : 167484                  University:14030
##  3rd Qu.: 240000
##  Max.   :1000000
##
##       age            pay_1           pay_2           pay_3
##  Min.   :21.00   0      :14737   0      :15730   0      :15764
##  1st Qu.:28.00   -1     : 5686   -1     : 6050   -1     : 5938
##  Median :34.00   1      : 3688   2      : 3927   -2     : 4085
##  Mean   :35.49   -2     : 2759   -2     : 3782   2      : 3819
##  3rd Qu.:41.00   2      : 2667   3      :  326   3      :  240
##  Max.   :79.00   3      :  322   4      :   99   4      :   76
##                  (Other):  141   (Other):   86   (Other):   78
##      pay_4           pay_5           pay_6         bill_amt1
##  0      :16455   0      :16947   0      :16286   Min.   :-165580
##  -1     : 5687   -1     : 5539   -1     : 5740   1st Qu.:   3559
##  -2     : 4348   -2     : 4546   -2     : 4895   Median :  22382
##  2      : 3159   2      : 2626   2      : 2766   Mean   :  51223
##  3      :  180   3      :  178   3      :  184   3rd Qu.:  67091
##  4      :   69   4      :   84   4      :   49   Max.   : 964511
##  (Other):  102   (Other):   80   (Other):   80
##    bill_amt2        bill_amt3         bill_amt4         bill_amt5
##  Min.   :-69777   Min.   :-157264   Min.   :-170000   Min.   :-81334
##  1st Qu.:  2985   1st Qu.:   2666   1st Qu.:   2327   1st Qu.:  1763
##  Median : 21200   Median :  20089   Median :  19052   Median : 18105
##  Mean   : 49179   Mean   :  47013   Mean   :  43263   Mean   : 40311
##  3rd Qu.: 64006   3rd Qu.:  60165   3rd Qu.:  54506   3rd Qu.: 50191
```

```
##  Max.   :983931   Max.   :1664089   Max.   : 891586   Max.   :927171
##
##    bill_amt6          pay_amt1          pay_amt2          pay_amt3
##  Min.   :-339603   Min.   :     0   Min.   :      0   Min.   :     0
##  1st Qu.:   1256   1st Qu.:  1000   1st Qu.:    833   1st Qu.:   390
##  Median :  17071   Median :  2100   Median :   2009   Median :  1800
##  Mean   :  38872   Mean   :  5664   Mean   :   5921   Mean   :  5226
##  3rd Qu.:  49198   3rd Qu.:  5006   3rd Qu.:   5000   3rd Qu.:  4505
##  Max.   : 961664   Max.   :873552   Max.   :1684259   Max.   :896040
##
##     pay_amt4          pay_amt5           pay_amt6                y
##  Min.   :     0   Min.   :     0.0   Min.   :     0.0   NoDefault:23364
##  1st Qu.:   296   1st Qu.:   252.5   1st Qu.:   117.8   Default  : 6636
##  Median :  1500   Median :  1500.0   Median :  1500.0
##  Mean   :  4826   Mean   :  4799.4   Mean   :  5215.5
##  3rd Qu.:  4013   3rd Qu.:  4031.5   3rd Qu.:  4000.0
##  Max.   :621000   Max.   :426529.0   Max.   :528666.0
##
```

The descriptive analysis shows several important patterns in the credit card default dataset. The average credit limit (LIMIT_BAL) is about 167,000, but the distribution is highly skewed, with some clients having extremely high limits. Most customers fall within the 30–41 age range, with a median age of 34. The majority of clients are female, university educated, and single or married, reflecting the demographic structure of the dataset.

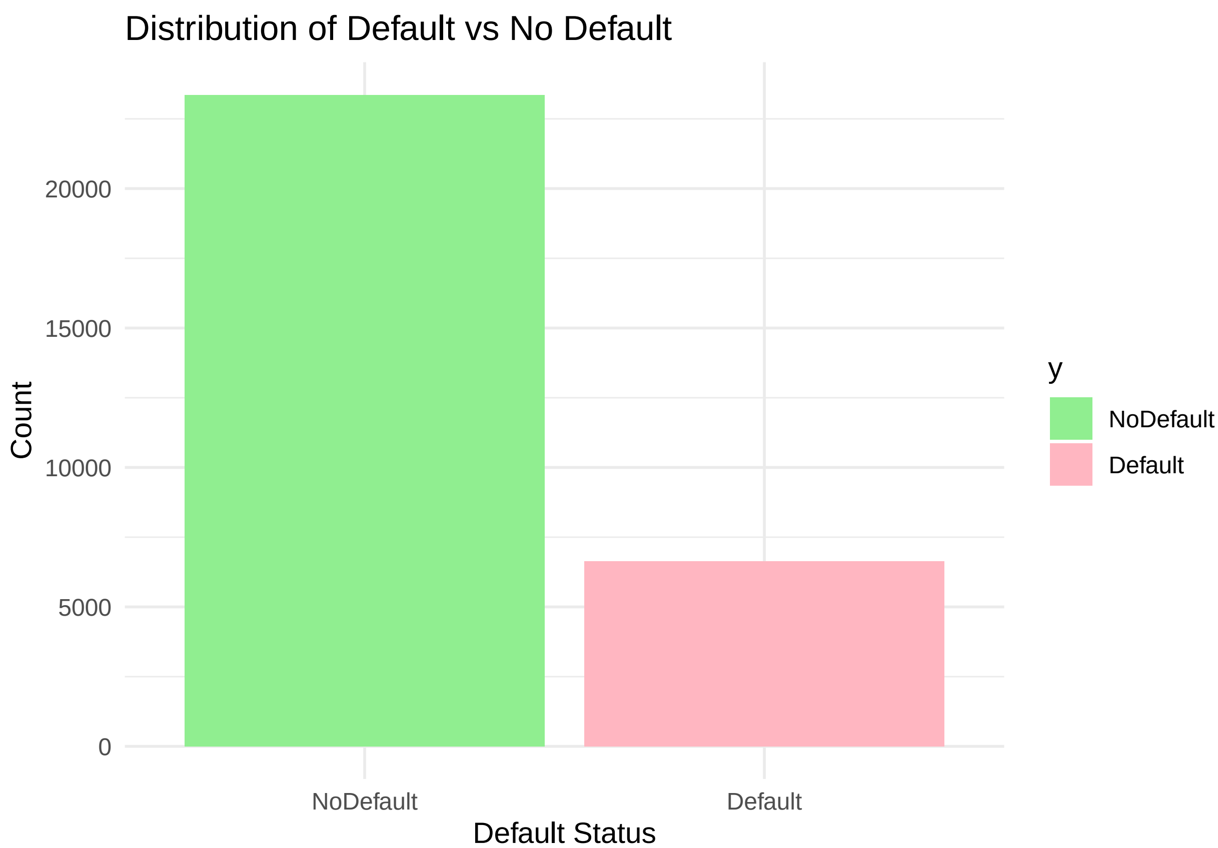
Payment history variables (PAY_0 to PAY_6) indicate that most customers are either on time or slightly delayed in previous months, though the presence of negative values (e.g., –1, –2) shows early payments or no usage. Bill amounts and payment amounts have strong right-skewness, with a few very large outliers across BILL_AMT and PAY_AMT variables.

Finally, the target variable shows that 23,335 clients (78%) are non-defaulters, while 6,630 clients (22%) defaulted, indicating moderate class imbalance. This imbalance is important to consider during modeling, especially for evaluation metrics and potential resampling techniques.

Exploratory Data Analysis

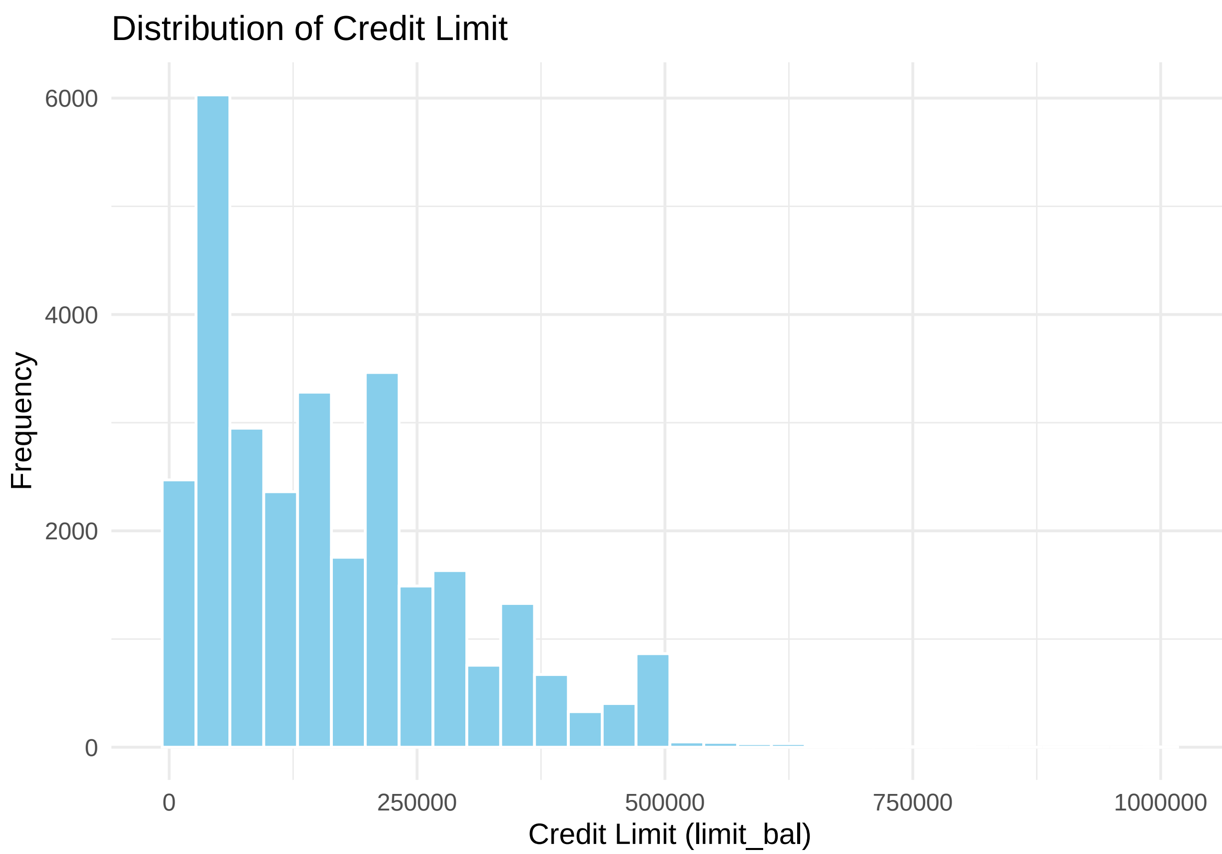
Default Distribution

```r
library(ggplot2)

ggplot(credit, aes(x = y, fill = y)) +
  geom_bar() +
  scale_fill_manual(values = c("lightgreen", "lightpink")) +
  labs(title = "Distribution of Default vs No Default",
       x = "Default Status",
       y = "Count") +
  theme_minimal()
```



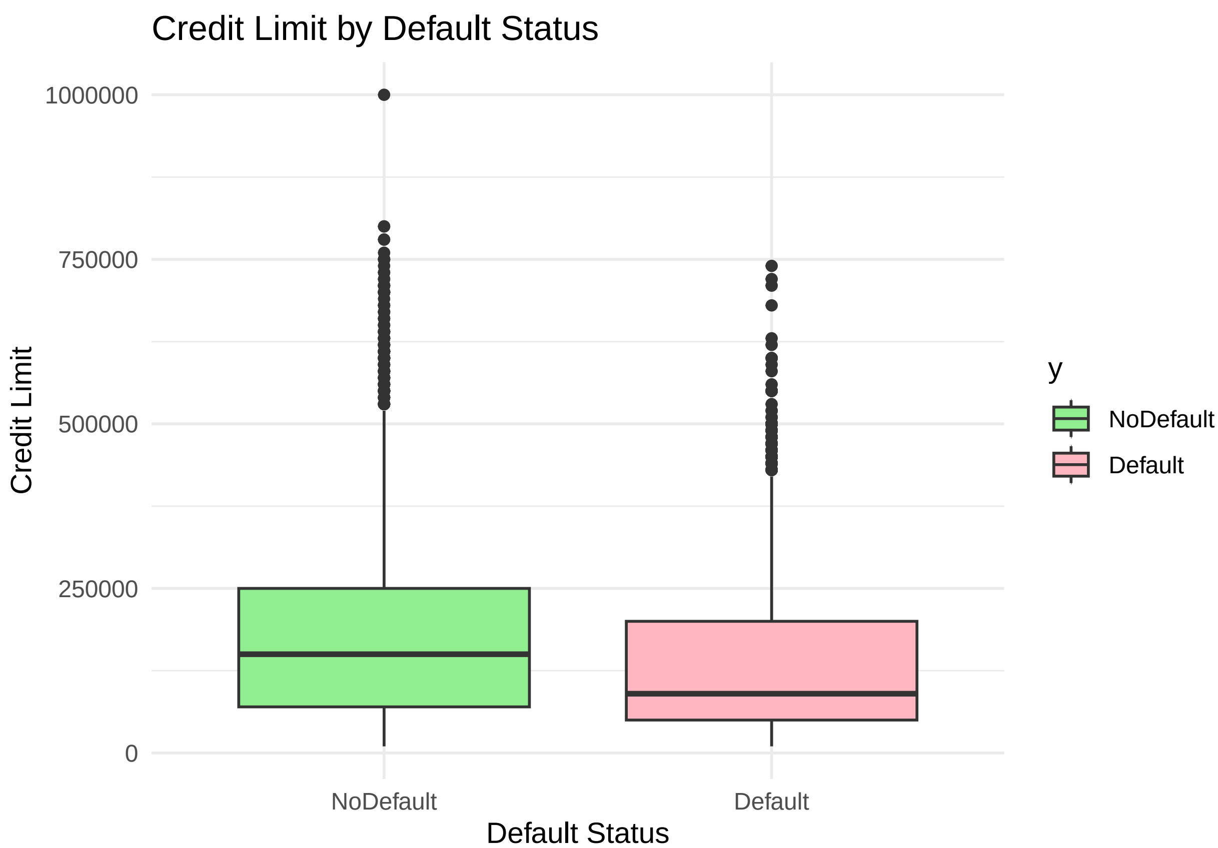
Histogram of Credit Limit

```
ggplot(credit, aes(x = limit_bal)) +
  geom_histogram(bins = 30, fill = "skyblue", color = "white") +
  labs(title = "Distribution of Credit Limit",
       x = "Credit Limit (limit_bal)",
       y = "Frequency") +
  theme_minimal()
```



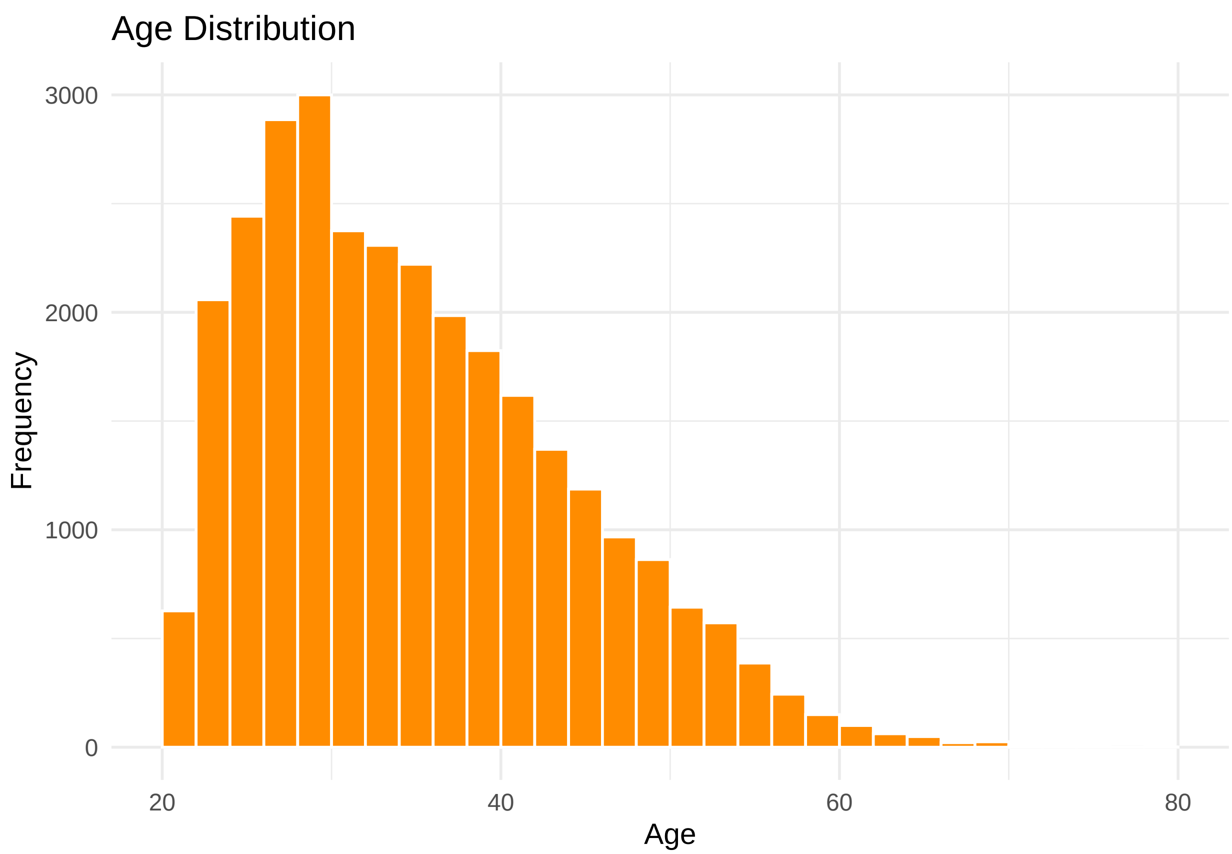
Credit Limit by Default Status

```
ggplot(credit, aes(x = y, y = limit_bal, fill = y)) +
  geom_boxplot() +
  scale_fill_manual(values = c("lightgreen", "lightpink")) +
  labs(title = "Credit Limit by Default Status",
       x = "Default Status",
       y = "Credit Limit") +
  theme_minimal()
```



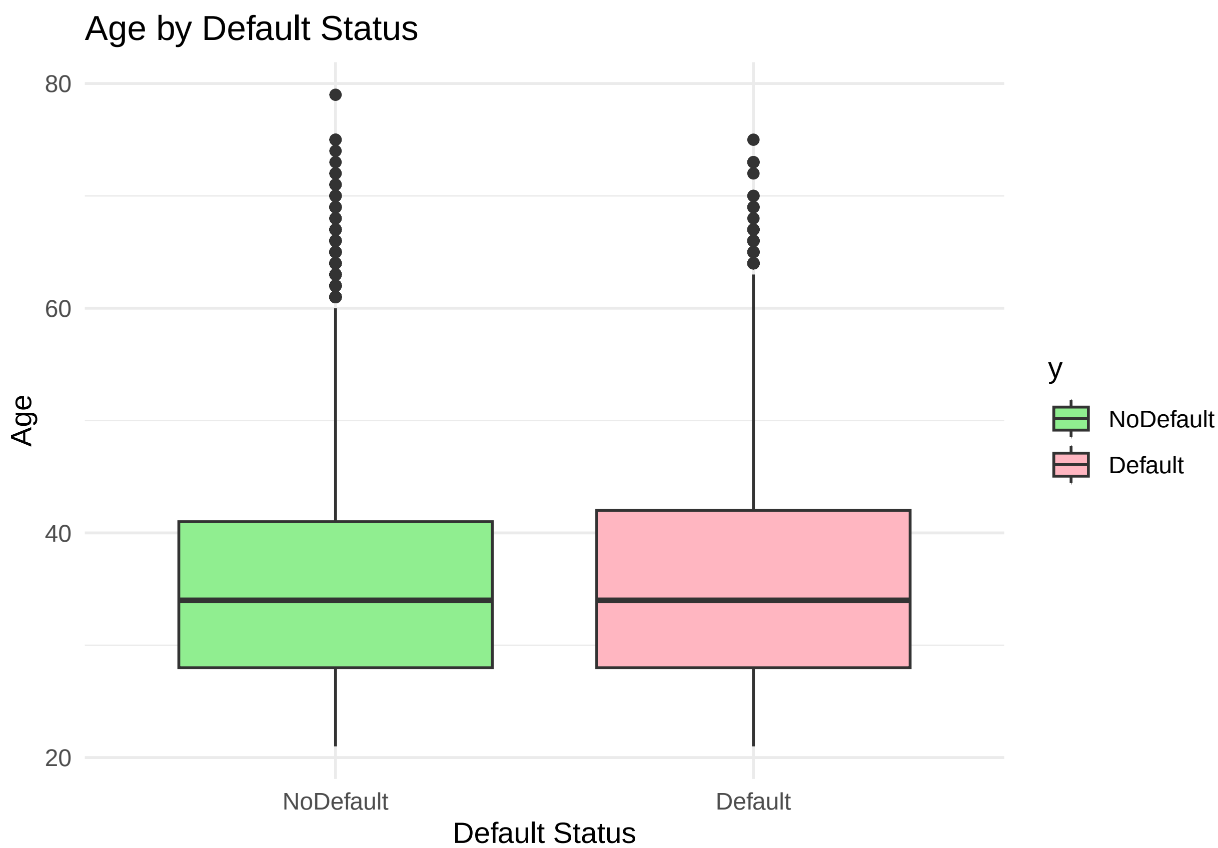
Histogram of age

```r
ggplot(credit, aes(x = age)) +
  geom_histogram(bins = 30, fill = "darkorange", color = "white") +
  labs(title = "Age Distribution",
       x = "Age",
       y = "Frequency") +
  theme_minimal()
```



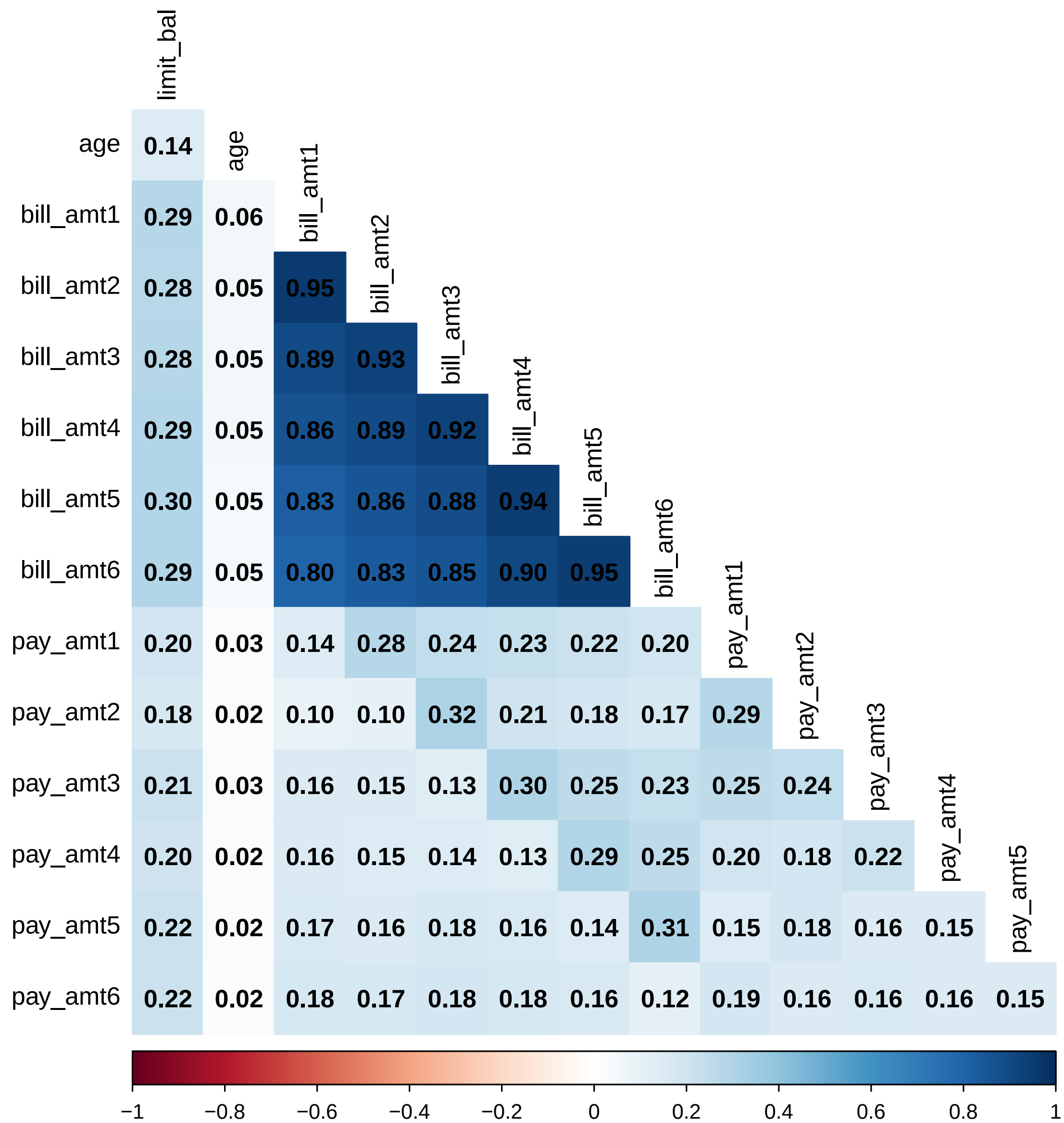
Age by Default

```
ggplot(credit, aes(x = y, y = age, fill = y)) +
  geom_boxplot() +
  scale_fill_manual(values = c("lightgreen", "lightpink"))+
  labs(title = "Age by Default Status",
       x = "Default Status",
       y = "Age") +
  theme_minimal()
```



Correlation Heatmmap

```
num_vars <- dplyr::select(credit, dplyr::where(is.numeric))
corr_matrix <- stats::cor(num_vars, use = "pairwise.complete.obs")

corrplot::corrplot(
  corr_matrix,
  method = "color",
  type   = "lower",
  addCoef.col = "black",
  tl.col = "black",
  diag   = FALSE
)
```



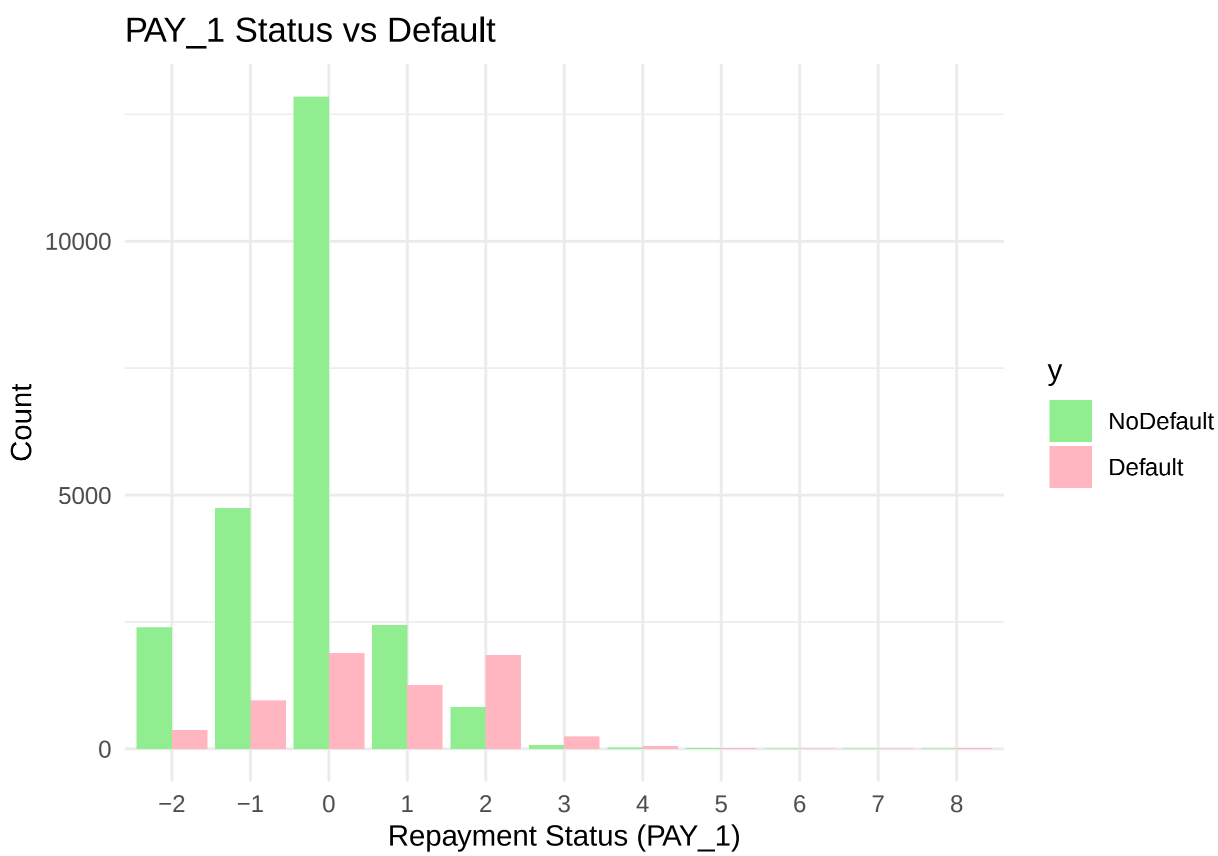

```r
# Find highly correlated variables & remove them
threshold <- 0.70
high_corr_vars <- caret::findCorrelation(
  corr_matrix,
  cutoff = threshold,
  names  = TRUE
)

print(high_corr_vars)
```

```
## [1] "bill_amt4" "bill_amt5" "bill_amt3" "bill_amt6" "bill_amt2"
```

```r
credit <- credit[, !(names(credit) %in% high_corr_vars)]
```

Payment History vs Default

```r
ggplot(credit, aes(x = pay_1, fill = y)) +
  geom_bar(position = "dodge") +
  labs(title = "PAY_1 Status vs Default",
       x = "Repayment Status (PAY_1)",
       y = "Count") +
  scale_fill_manual(values = c("lightgreen", "lightpink")) +
  theme_minimal()
```



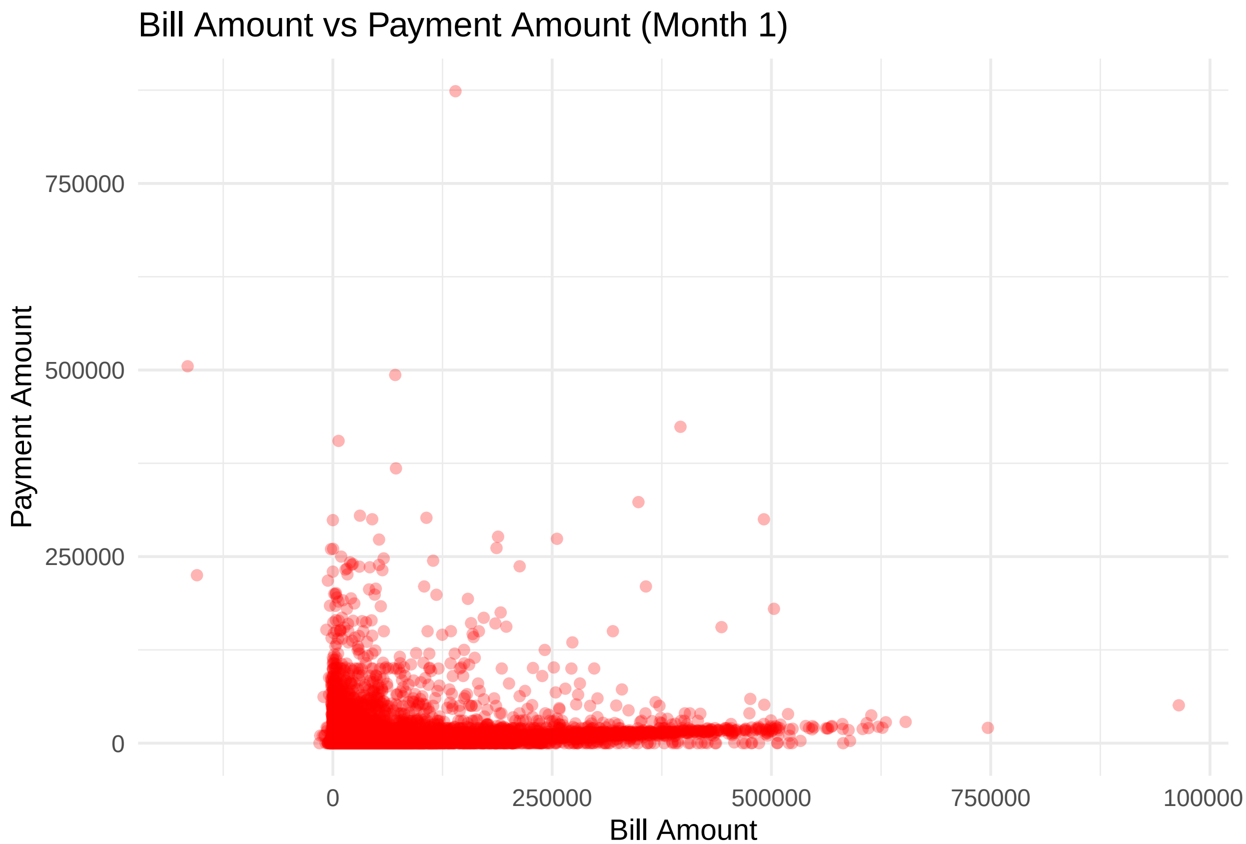
PAY_0 = 0 or -1 → low default PAY_0   1 → strong default risk

This visually justifies why Decision Trees split on PAY_0 early

Payment Amount vs Bill Amount (Scatter)

```r
ggplot(credit, aes(x = bill_amt1, y = pay_amt1)) +
  geom_point(alpha = 0.3, color = "red") +
  labs(title = "Bill Amount vs Payment Amount (Month 1)",
       x = "Bill Amount",
```

```r
       y = "Payment Amount") +
  theme_minimal()
```



###Train-Test Split###

```r
# Train-Test Split
train_index <- createDataPartition(credit$y, p = 0.7, list = FALSE)

train_set <- credit[train_index, ]
test_set  <- credit[-train_index, ]

table(train_set$y)
```

```
##
## NoDefault   Default
##     16355      4646
```

```r
# Oversampling Training Set
train_set_bal <- upSample(x = train_set[, -which(names(train_set) == "y")],
                          y = train_set$y,
                          yname = "y")

table(train_set_bal$y)
```

```
##
```

```
## NoDefault   Default
##     16355     16355
```

Models

```r
# Function to plot confusion matrix
plot_confusion_matrix <- function(cfm, title, color_low, color_high) {
  cm_table <- as.data.frame(cfm$table)
  ggplot(cm_table, aes(x = Reference, y = Prediction)) +
    geom_tile(aes(fill = Freq), colour = "white") +
    geom_text(aes(label = Freq), size = 5) +
    scale_fill_gradient(low = color_low, high = color_high) +
    scale_y_discrete(limits = rev(levels(cm_table$Reference))) +
    ggtitle(title) +
    theme_minimal() +
    theme(
      plot.title = element_text(face = "bold", size = 14),
      axis.title = element_text(size = 12),
      axis.text = element_text(size = 11)
    )
}
```

Logistic Regression - Imbalanced Training

```r
log_reg <- glm(y ~., family=binomial(link = 'logit'), data=train_set)
```

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

```r
summary(log_reg)
```

```
##
## Call:
## glm(formula = y ~ ., family = binomial(link = "logit"), data = train_set)
##
## Coefficients: (1 not defined because of singularities)
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          3.369e+00  8.426e+01   0.040 0.968102
## limit_bal           -2.020e-06  2.095e-07  -9.644  < 2e-16 ***
## sexFemale           -1.416e-01  3.858e-02  -3.670 0.000242 ***
## educationHighSchool -6.460e-02  6.017e-02  -1.074 0.282981
## educationOthers     -8.521e-01  2.119e-01  -4.020 5.81e-05 ***
## educationUniversity  3.855e-04  4.466e-02   0.009 0.993112
## marriageOthers      -1.111e-01  1.660e-01  -0.669 0.503388
## marriageSingle      -1.408e-01  4.360e-02  -3.229 0.001241 **
## age                  4.149e-03  2.369e-03   1.751 0.079861 .
## pay_1.L              4.237e+01  1.361e+03   0.031 0.975154
## pay_1.Q              4.022e+01  1.352e+03   0.030 0.976275
## pay_1.C              3.342e+01  1.064e+03   0.031 0.974932
## pay_1^4              2.283e+01  6.736e+02   0.034 0.972962
```

```
## pay_1^5              1.111e+01  3.297e+02   0.034 0.973129
## pay_1^6              3.037e+00  1.076e+02   0.028 0.977480
## pay_1^7              7.979e-01  2.215e+01   0.036 0.971270
## pay_1^8             -9.240e-01  2.911e+01  -0.032 0.974682
## pay_1^9             -5.177e-01  1.926e+01  -0.027 0.978552
## pay_1^10             1.623e-01  6.893e+00   0.024 0.981219
## pay_2.L             -3.075e+01  1.121e+03  -0.027 0.978126
## pay_2.Q             -1.037e+01  5.846e+02  -0.018 0.985841
## pay_2.C              1.881e+01  4.927e+02   0.038 0.969546
## pay_2^4              4.100e+01  1.065e+03   0.039 0.969278
## pay_2^5              4.658e+01  1.207e+03   0.039 0.969207
## pay_2^6              3.823e+01  9.473e+02   0.040 0.967811
## pay_2^7              2.354e+01  5.419e+02   0.043 0.965346
## pay_2^8              1.062e+01  2.161e+02   0.049 0.960807
## pay_2^9              2.534e+00  4.890e+01   0.052 0.958675
## pay_2^10                    NA         NA      NA       NA
## pay_3.L              1.787e+01  3.162e+02   0.057 0.954932
## pay_3.Q              1.386e+01  3.294e+02   0.042 0.966436
## pay_3.C              2.097e+00  3.475e+02   0.006 0.995185
## pay_3^4              4.597e+00  2.378e+02   0.019 0.984576
## pay_3^5              8.727e+00  2.260e+02   0.039 0.969201
## pay_3^6              3.233e+00  2.145e+02   0.015 0.987974
## pay_3^7              8.184e+00  1.364e+02   0.060 0.952147
## pay_3^8              1.195e+01  2.521e+02   0.047 0.962177
## pay_3^9             -1.897e+00  2.711e+02  -0.007 0.994416
## pay_3^10             4.749e+00  1.522e+02   0.031 0.975103
## pay_4.L             -5.403e+01  5.227e+02  -0.103 0.917676
## pay_4.Q             -4.068e+01  4.512e+02  -0.090 0.928166
## pay_4.C             -1.066e+01  3.660e+02  -0.029 0.976759
## pay_4^4             -3.074e-01  2.397e+02  -0.001 0.998977
## pay_4^5             -6.367e-01  2.510e+02  -0.003 0.997976
## pay_4^6              3.819e+00  2.469e+02   0.015 0.987660
## pay_4^7             -5.172e+00  1.773e+02  -0.029 0.976725
## pay_4^8             -1.212e+01  2.673e+02  -0.045 0.963832
## pay_4^9              2.157e+00  2.762e+02   0.008 0.993768
## pay_4^10            -5.278e+00  1.535e+02  -0.034 0.972572
## pay_5.L              5.371e+01  8.507e+02   0.063 0.949657
## pay_5.Q              3.571e+01  7.437e+02   0.048 0.961705
## pay_5.C              9.128e+00  5.413e+02   0.017 0.986544
## pay_5^4             -8.506e+00  3.862e+02  -0.022 0.982426
## pay_5^5             -1.105e+01  2.575e+02  -0.043 0.965780
## pay_5^6             -4.037e+00  1.714e+02  -0.024 0.981207
## pay_5^7              2.206e+00  1.462e+02   0.015 0.987965
## pay_5^8              3.520e+00  1.071e+02   0.033 0.973790
## pay_5^9              2.040e+00  4.850e+01   0.042 0.966446
## pay_6.L             -7.630e+00  3.990e+02  -0.019 0.984743
## pay_6.Q             -5.594e+00  4.023e+02  -0.014 0.988905
## pay_6.C             -1.034e+00  3.332e+02  -0.003 0.997523
## pay_6^4              3.069e+00  2.437e+02   0.013 0.989951
## pay_6^5              4.073e+00  1.655e+02   0.025 0.980365
## pay_6^6              3.801e+00  1.049e+02   0.036 0.971092
## pay_6^7              3.259e+00  5.857e+01   0.056 0.955622
## pay_6^8              1.613e+00  2.620e+01   0.062 0.950927
## pay_6^9              3.559e-01  7.977e+00   0.045 0.964410
```

```
## bill_amt1            2.331e-06  3.691e-07   6.315 2.70e-10 ***
## pay_amt1            -1.322e-05  2.935e-06  -4.504 6.66e-06 ***
## pay_amt2            -9.339e-06  2.507e-06  -3.725 0.000195 ***
## pay_amt3            -3.316e-06  1.918e-06  -1.729 0.083892 .
## pay_amt4            -1.742e-06  1.856e-06  -0.938 0.347990
## pay_amt5            -2.605e-06  1.725e-06  -1.510 0.131053
## pay_amt6            -2.534e-06  1.588e-06  -1.595 0.110624
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22196  on 21000  degrees of freedom
## Residual deviance: 18253  on 20928  degrees of freedom
## AIC: 18399
##
## Number of Fisher Scoring iterations: 12
```

```r
# Training performance
log_pred_train <- predict(log_reg, newdata = train_set, type = "response")
predicted_train <- factor(ifelse(log_pred_train > 0.5, 1, 0), levels = c(0,1), labels = c("NoDefault","I

con_mat <- confusionMatrix(predicted_train, train_set$y, mode = 'everything', positive = 'Default')
print(con_mat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault     15598    2994
##   Default         757    1652
##
##                Accuracy : 0.8214
##                  95% CI : (0.8161, 0.8265)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3737
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.35557
##             Specificity : 0.95371
##          Pos Pred Value : 0.68576
##          Neg Pred Value : 0.83896
##               Precision : 0.68576
##                  Recall : 0.35557
##                      F1 : 0.46832
##              Prevalence : 0.22123
##          Detection Rate : 0.07866
##    Detection Prevalence : 0.11471
##       Balanced Accuracy : 0.65464
##
##        'Positive' Class : Default
```

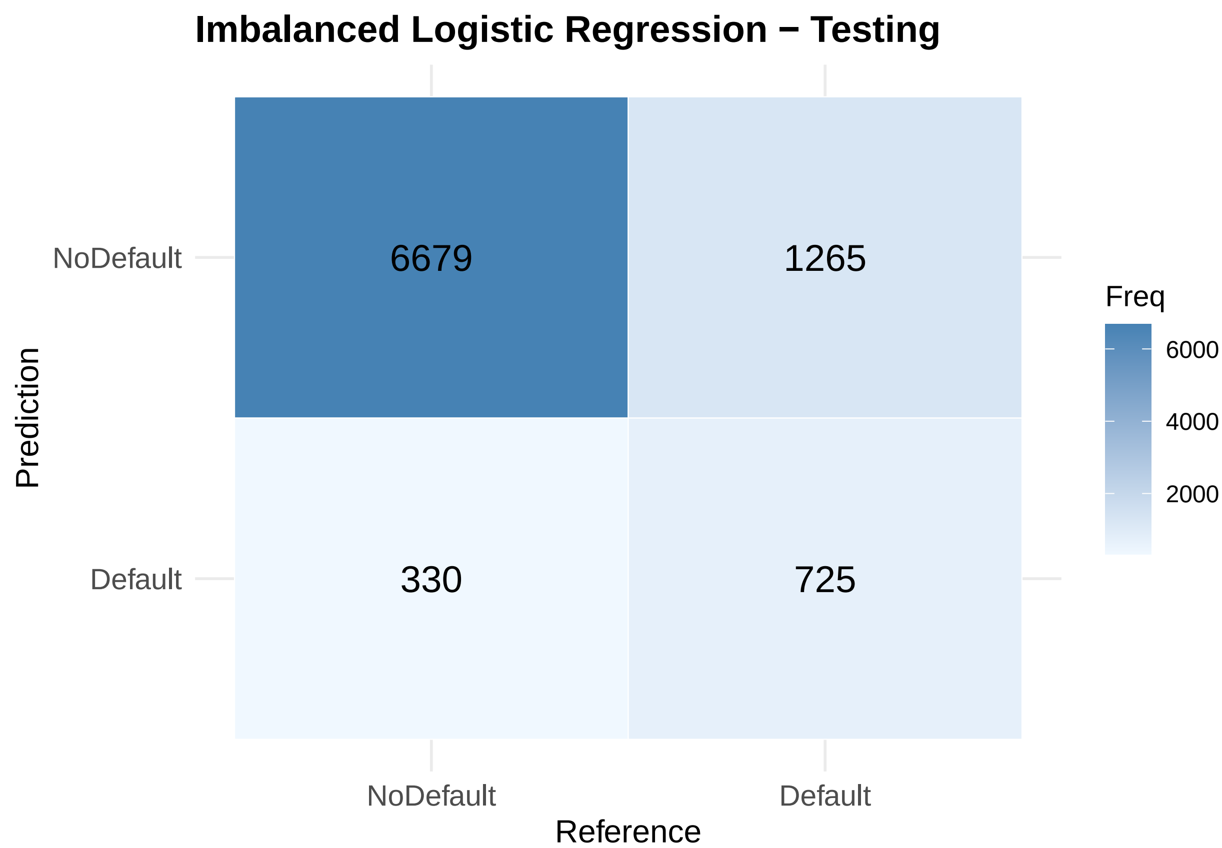
```
##
```

```r
# Test prediction
log_pred_test <- predict(log_reg, newdata = test_set, type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
predicted_test <- factor(ifelse(log_pred_test > 0.5, 1, 0), levels = c(0,1), labels = c("NoDefault","De

con_mat <- confusionMatrix(predicted_test, test_set$y, mode = 'everything', positive = 'Default')
print(con_mat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault      6679    1265
##   Default         330     725
##
##                Accuracy : 0.8228
##                  95% CI : (0.8147, 0.8306)
##     No Information Rate : 0.7789
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3814
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.36432
##             Specificity : 0.95292
##          Pos Pred Value : 0.68720
##          Neg Pred Value : 0.84076
##               Precision : 0.68720
##                  Recall : 0.36432
##                      F1 : 0.47619
##              Prevalence : 0.22114
##          Detection Rate : 0.08056
##    Detection Prevalence : 0.11724
##       Balanced Accuracy : 0.65862
##
##        'Positive' Class : Default
##
```




```
## Setting levels: control = NoDefault, case = Default

## Setting direction: controls < cases
```

Logistic Regression - Balanced Training

```
log_reg_bal <- glm(y ~., family=binomial(link = 'logit'), data=train_set_bal)
```

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

```
summary(log_reg_bal)
```

```
##
## Call:
## glm(formula = y ~ ., family = binomial(link = "logit"), data = train_set_bal)
##
## Coefficients: (1 not defined because of singularities)
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          4.552e+00  1.028e+02   0.044 0.964675
## limit_bal           -2.017e-06  1.338e-07 -15.080  < 2e-16 ***
## sexFemale           -1.159e-01  2.587e-02  -4.482 7.40e-06 ***
## educationHighSchool -9.697e-03  4.011e-02  -0.242 0.808951
```

```
## educationOthers     -6.234e-01  1.235e-01  -5.049 4.45e-07 ***
## educationUniversity  3.954e-02  2.969e-02   1.332 0.182923
## marriageOthers      -9.179e-02  1.107e-01  -0.829 0.406902
## marriageSingle      -1.502e-01  2.905e-02  -5.169 2.35e-07 ***
## age                  2.947e-03  1.588e-03   1.856 0.063507 .
## pay_1.L              5.021e+01  2.231e+03   0.023 0.982041
## pay_1.Q              4.843e+01  2.217e+03   0.022 0.982572
## pay_1.C              4.030e+01  1.743e+03   0.023 0.981550
## pay_1^4              2.752e+01  1.103e+03   0.025 0.980088
## pay_1^5              1.371e+01  5.385e+02   0.025 0.979686
## pay_1^6              4.312e+00  1.737e+02   0.025 0.980194
## pay_1^7              1.210e+00  3.260e+01   0.037 0.970391
## pay_1^8             -8.603e-01  4.729e+01  -0.018 0.985488
## pay_1^9             -6.908e-01  3.152e+01  -0.022 0.982514
## pay_1^10             1.376e-01  1.130e+01   0.012 0.990280
## pay_2.L             -3.708e+01  1.832e+03  -0.020 0.983856
## pay_2.Q             -1.352e+01  9.423e+02  -0.014 0.988557
## pay_2.C              2.108e+01  7.942e+02   0.027 0.978828
## pay_2^4              4.753e+01  1.744e+03   0.027 0.978255
## pay_2^5              5.418e+01  1.979e+03   0.027 0.978160
## pay_2^6              4.443e+01  1.554e+03   0.029 0.977197
## pay_2^7              2.730e+01  8.894e+02   0.031 0.975508
## pay_2^8              1.229e+01  3.547e+02   0.035 0.972358
## pay_2^9              3.231e+00  8.024e+01   0.040 0.967874
## pay_2^10                    NA         NA      NA       NA
## pay_3.L              1.913e+01  2.905e+02   0.066 0.947484
## pay_3.Q              1.485e+01  2.876e+02   0.052 0.958824
## pay_3.C              1.245e+00  3.914e+02   0.003 0.997461
## pay_3^4              4.654e+00  2.158e+02   0.022 0.982790
## pay_3^5              9.688e+00  3.022e+02   0.032 0.974426
## pay_3^6              2.974e+00  3.240e+02   0.009 0.992675
## pay_3^7              8.714e+00  1.920e+02   0.045 0.963807
## pay_3^8              1.327e+01  4.033e+02   0.033 0.973761
## pay_3^9             -2.998e+00  4.416e+02  -0.007 0.994585
## pay_3^10             5.505e+00  2.490e+02   0.022 0.982360
## pay_4.L             -5.867e+01  6.160e+02  -0.095 0.924113
## pay_4.Q             -4.371e+01  5.706e+02  -0.077 0.938935
## pay_4.C             -1.092e+01  5.437e+02  -0.020 0.983977
## pay_4^4              1.440e-01  3.613e+02   0.000 0.999682
## pay_4^5             -6.530e-01  3.725e+02  -0.002 0.998601
## pay_4^6              3.759e+00  3.575e+02   0.011 0.991611
## pay_4^7             -6.086e+00  2.236e+02  -0.027 0.978284
## pay_4^8             -1.345e+01  4.131e+02  -0.033 0.974022
## pay_4^9              2.269e+00  4.450e+02   0.005 0.995931
## pay_4^10            -5.550e+00  2.499e+02  -0.022 0.982281
## pay_5.L              5.690e+01  8.548e+02   0.067 0.946927
## pay_5.Q              3.742e+01  7.884e+02   0.047 0.962143
## pay_5.C              9.326e+00  6.172e+02   0.015 0.987945
## pay_5^4             -8.866e+00  4.536e+02  -0.020 0.984405
## pay_5^5             -1.125e+01  2.966e+02  -0.038 0.969743
## pay_5^6             -3.928e+00  1.801e+02  -0.022 0.982605
## pay_5^7              2.486e+00  1.363e+02   0.018 0.985452
## pay_5^8              3.530e+00  9.723e+01   0.036 0.971037
## pay_5^9              1.852e+00  4.396e+01   0.042 0.966402
```

```
## pay_6.L             -7.334e+00  4.274e+02  -0.017 0.986309
## pay_6.Q             -5.281e+00  4.392e+02  -0.012 0.990405
## pay_6.C             -9.789e-01  3.715e+02  -0.003 0.997898
## pay_6^4              2.876e+00  2.736e+02   0.011 0.991614
## pay_6^5              3.950e+00  1.807e+02   0.022 0.982555
## pay_6^6              3.666e+00  1.077e+02   0.034 0.972831
## pay_6^7              3.142e+00  5.618e+01   0.056 0.955408
## pay_6^8              1.580e+00  2.383e+01   0.066 0.947133
## pay_6^9              3.934e-01  6.997e+00   0.056 0.955164
## bill_amt1            2.103e-06  2.374e-07   8.858  < 2e-16 ***
## pay_amt1            -9.801e-06  1.486e-06  -6.594 4.28e-11 ***
## pay_amt2            -7.240e-06  1.309e-06  -5.531 3.18e-08 ***
## pay_amt3            -3.713e-06  1.036e-06  -3.584 0.000338 ***
## pay_amt4            -2.332e-06  1.140e-06  -2.045 0.040817 *
## pay_amt5            -2.896e-06  1.088e-06  -2.662 0.007777 **
## pay_amt6            -2.670e-06  9.920e-07  -2.691 0.007116 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 45346  on 32709  degrees of freedom
## Residual deviance: 37291  on 32637  degrees of freedom
## AIC: 37437
##
## Number of Fisher Scoring iterations: 13
```

```
# Balanced Training performance on original training set
log_pred_train_bal <- predict(log_reg_bal, newdata = train_set, type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```
predicted_train_bal <- factor(ifelse(log_pred_train_bal > 0.5, 1, 0), levels = c(0,1), labels = c("NoDe

con_mat <- confusionMatrix(predicted_train_bal, train_set$y, mode = 'everything', positive = 'Default')
print(con_mat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault     13708    1993
##   Default        2647    2653
##
##                Accuracy : 0.7791
##                  95% CI : (0.7734, 0.7847)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 0.4642
##
##                   Kappa : 0.3896
##
##  Mcnemar's Test P-Value : <2e-16
```

```
##
##             Sensitivity : 0.5710
##             Specificity : 0.8382
##          Pos Pred Value : 0.5006
##          Neg Pred Value : 0.8731
##               Precision : 0.5006
##                  Recall : 0.5710
##                      F1 : 0.5335
##              Prevalence : 0.2212
##          Detection Rate : 0.1263
##    Detection Prevalence : 0.2524
##       Balanced Accuracy : 0.7046
##
##        'Positive' Class : Default
##
```

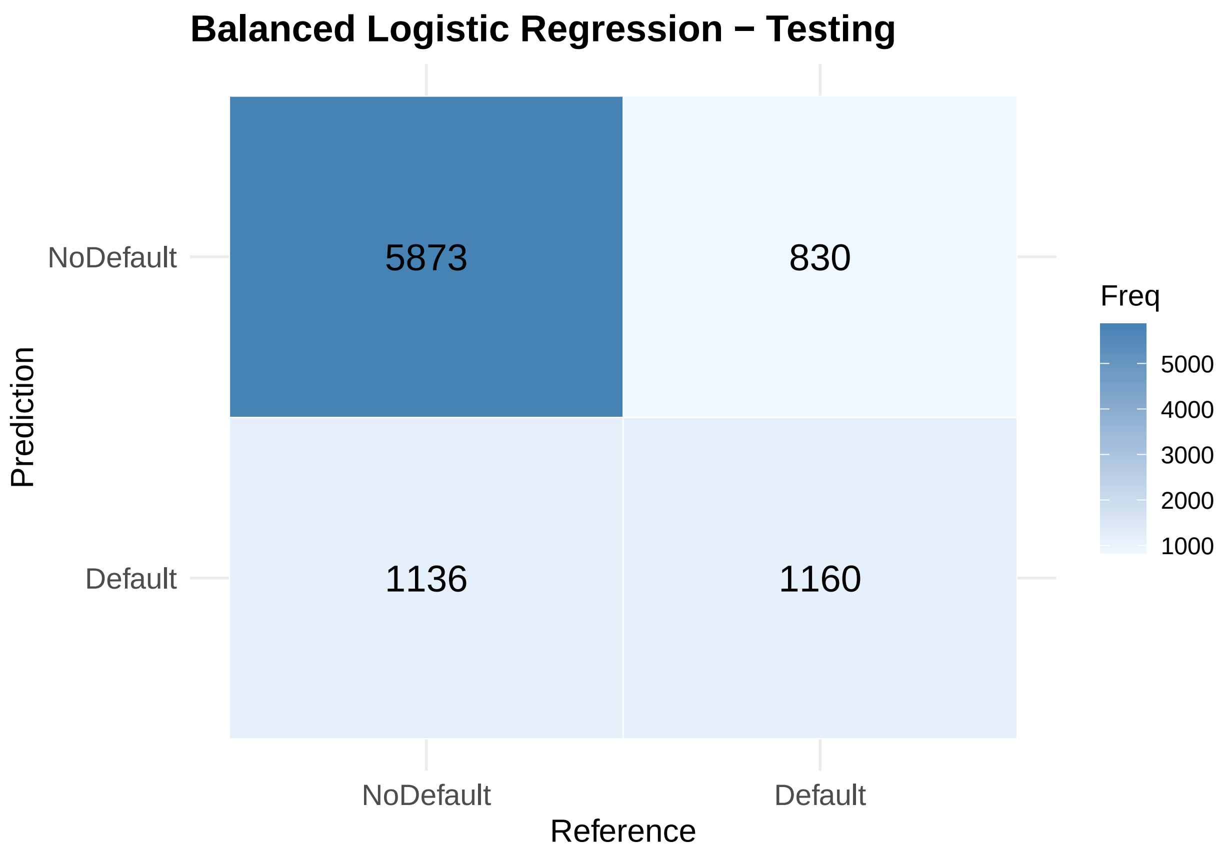
```
# Balanced testing performance
log_pred_test_bal <- predict(log_reg_bal, newdata = test_set, type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```
predicted_test_bal <- factor(ifelse(log_pred_test_bal > 0.5, 1, 0), levels = c(0,1), labels = c("NoDefa

con_mat <- confusionMatrix(predicted_test_bal, test_set$y, mode = 'everything', positive = 'Default')
print(con_mat)
```

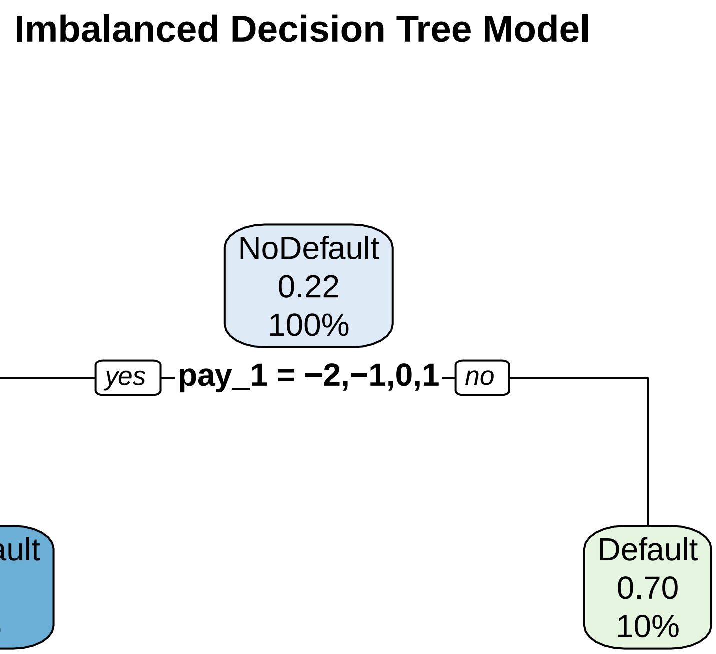
```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault      5873     830
##   Default        1136    1160
##
##                Accuracy : 0.7815
##                  95% CI : (0.7728, 0.79)
##     No Information Rate : 0.7789
##     P-Value [Acc > NIR] : 0.2758
##
##                   Kappa : 0.3989
##
##  Mcnemar's Test P-Value : 6.039e-12
##
##             Sensitivity : 0.5829
##             Specificity : 0.8379
##          Pos Pred Value : 0.5052
##          Neg Pred Value : 0.8762
##               Precision : 0.5052
##                  Recall : 0.5829
##                      F1 : 0.5413
##              Prevalence : 0.2211
##          Detection Rate : 0.1289
##    Detection Prevalence : 0.2551
```

```
##        Balanced Accuracy : 0.7104
##
##         'Positive' Class : Default
##
```



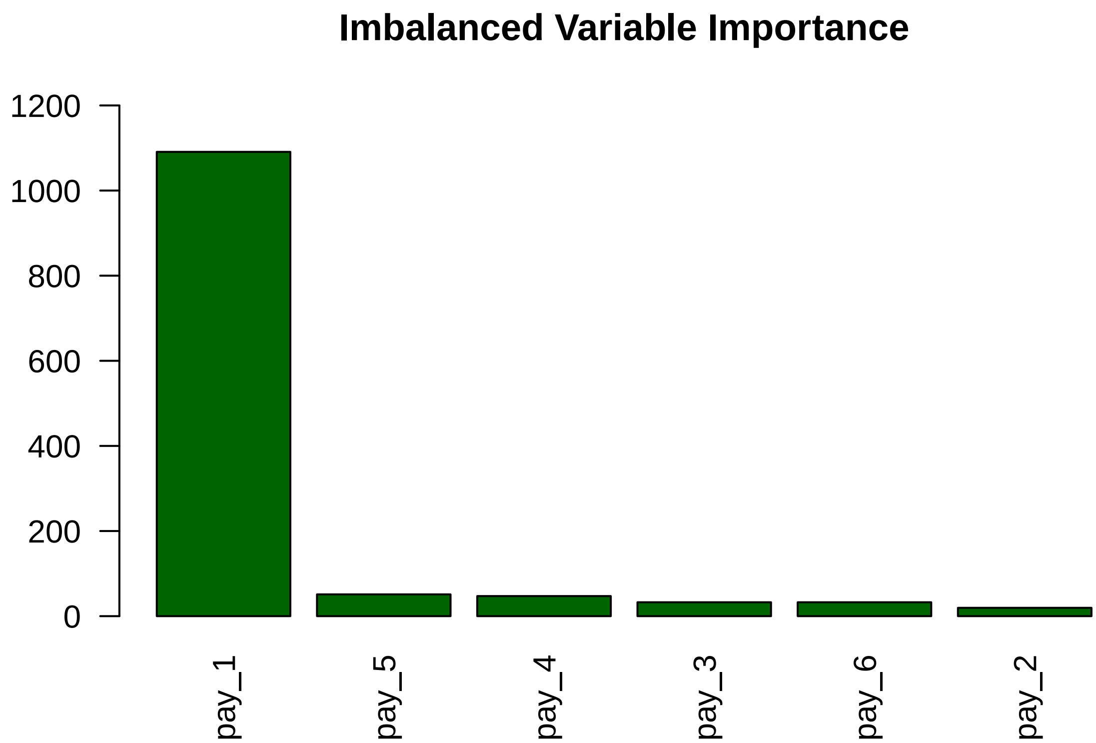

```
## Setting levels: control = NoDefault, case = Default

## Setting direction: controls < cases
```

CART Decision tree - Imbalanced Training Set




```r
print(dt_model)
```

```
## n= 21001
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 21001 4646 NoDefault (0.7787724 0.2212276)
##   2) pay_1=-2,-1,0,1 18827 3134 NoDefault (0.8335369 0.1664631) *
##   3) pay_1=2,3,4,5,6,7,8 2174  662 Default (0.3045078 0.6954922) *
```



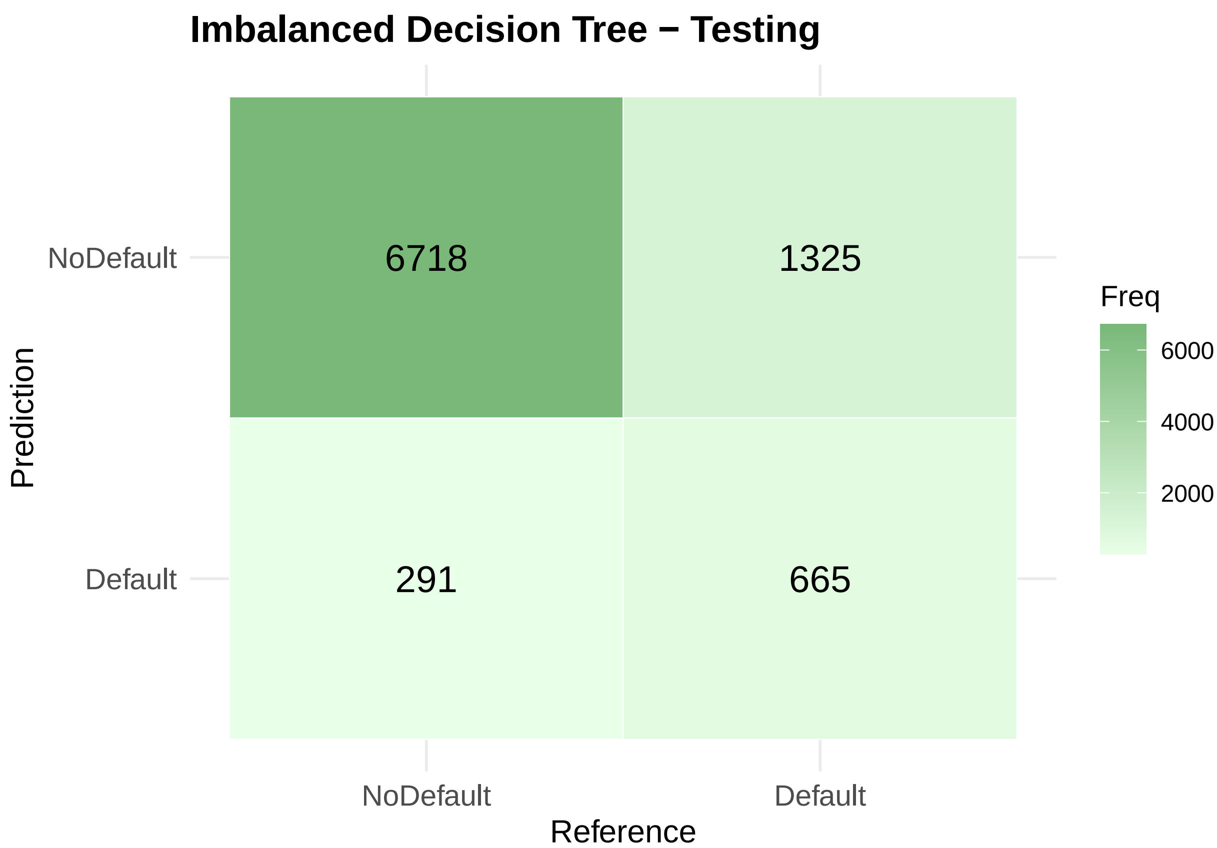

```r
# Training performance
dt_pred_train <- predict(dt_model, train_set, type = "class")
con_mat <- confusionMatrix(dt_pred_train, train_set$y, mode = 'everything', positive = "Default")
print(con_mat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault     15693    3134
##   Default         662    1512
##
##                Accuracy : 0.8192
##                  95% CI : (0.814, 0.8244)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.352
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.3254
##             Specificity : 0.9595
##          Pos Pred Value : 0.6955
##          Neg Pred Value : 0.8335
```

```
##               Precision : 0.6955
##                  Recall : 0.3254
##                      F1 : 0.4434
##              Prevalence : 0.2212
##          Detection Rate : 0.0720
##    Detection Prevalence : 0.1035
##       Balanced Accuracy : 0.6425
##
##        'Positive' Class : Default
##
```

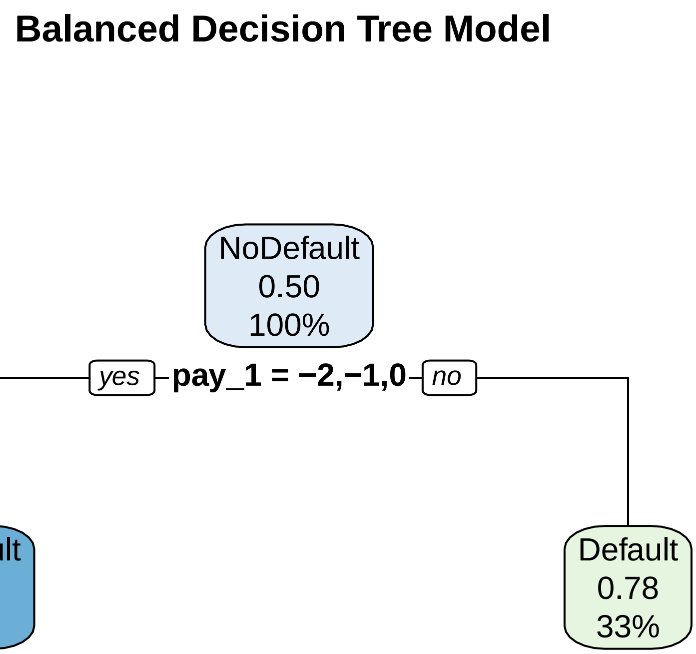
```r
# Testing performance
dt_pred_test <- predict(dt_model, test_set, type = "class")
con_mat <- confusionMatrix(dt_pred_test, test_set$y, mode = 'everything', positive = "Default")
print(con_mat)
```

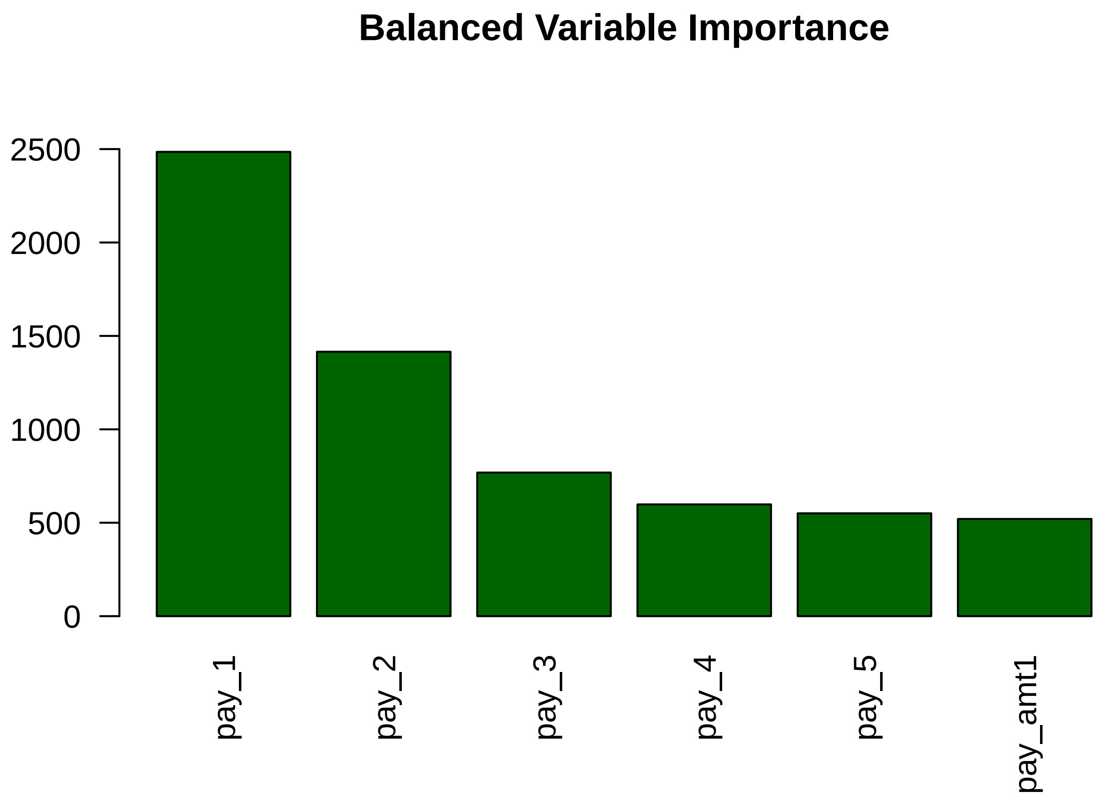
```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault      6718    1325
##   Default         291     665
##
##                Accuracy : 0.8204
##                  95% CI : (0.8123, 0.8283)
##     No Information Rate : 0.7789
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3595
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.3342
##             Specificity : 0.9585
##          Pos Pred Value : 0.6956
##          Neg Pred Value : 0.8353
##               Precision : 0.6956
##                  Recall : 0.3342
##                      F1 : 0.4515
##              Prevalence : 0.2211
##          Detection Rate : 0.0739
##    Detection Prevalence : 0.1062
##       Balanced Accuracy : 0.6463
##
##        'Positive' Class : Default
##
```




```
## Setting levels: control = NoDefault, case = Default

## Setting direction: controls < cases
```

CART Decision tree - Balanced Training Set





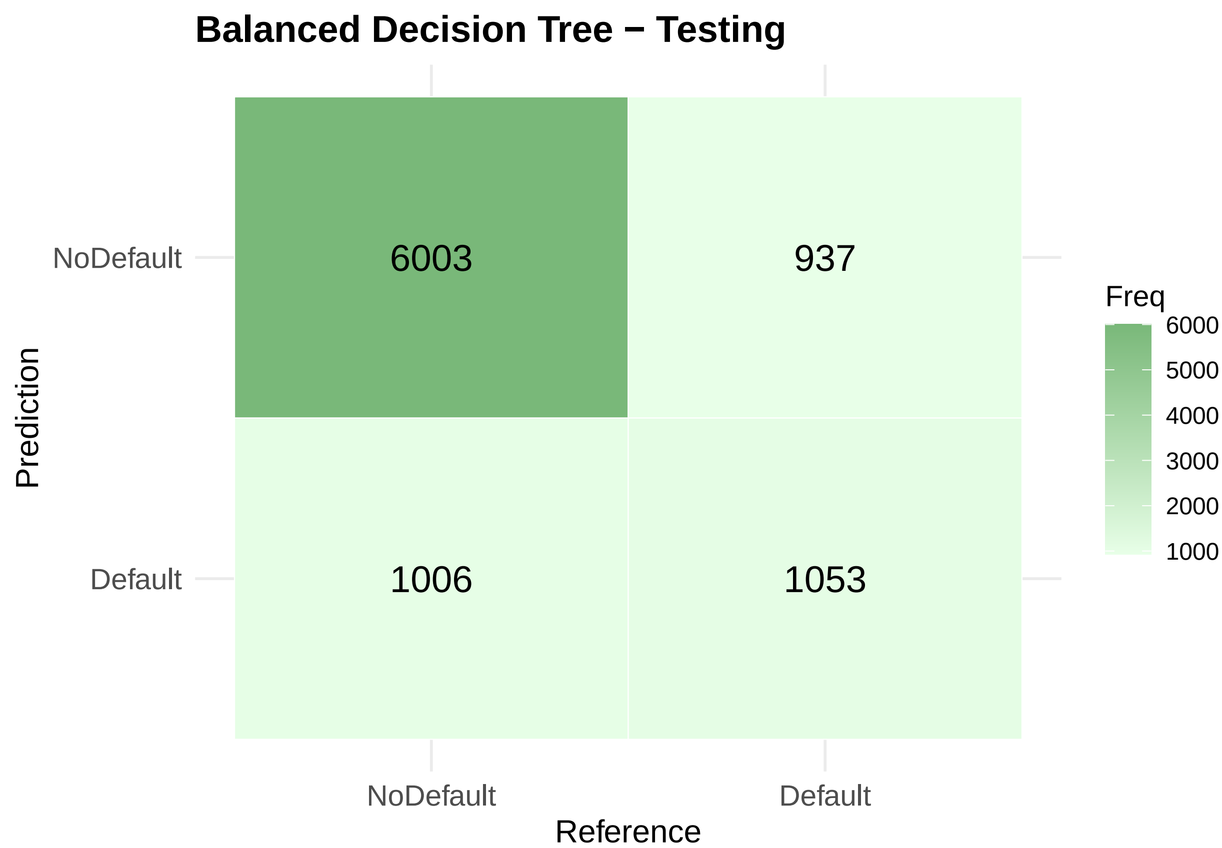

```r
# Balanced Training performance on original training set
dt_pred_train_bal <- predict(dt_model_bal, train_set, type = "class")
con_mat <- confusionMatrix(dt_pred_train_bal, train_set$y, mode = 'everything', positive = "Default")
print(con_mat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault     13972    2270
##   Default        2383    2376
##
##                Accuracy : 0.7784
##                  95% CI : (0.7728, 0.784)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 0.5502
##
##                   Kappa : 0.3625
##
##  Mcnemar's Test P-Value : 0.1006
##
##             Sensitivity : 0.5114
##             Specificity : 0.8543
##          Pos Pred Value : 0.4993
##          Neg Pred Value : 0.8602
```

```
##                Precision : 0.4993
##                   Recall : 0.5114
##                       F1 : 0.5053
##               Prevalence : 0.2212
##           Detection Rate : 0.1131
##     Detection Prevalence : 0.2266
##        Balanced Accuracy : 0.6829
##
##         'Positive' Class : Default
##
```

```
# Testing performance
dt_pred_test <- predict(dt_model_bal, test_set, type = "class")
con_mat <- confusionMatrix(dt_pred_test, test_set$y, mode = 'everything', positive = "Default")
print(con_mat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault      6003     937
##   Default        1006    1053
##
##                Accuracy : 0.7841
##                  95% CI : (0.7754, 0.7926)
##     No Information Rate : 0.7789
##     P-Value [Acc > NIR] : 0.1186
##
##                   Kappa : 0.3809
##
##  Mcnemar's Test P-Value : 0.1229
##
##             Sensitivity : 0.5291
##             Specificity : 0.8565
##          Pos Pred Value : 0.5114
##          Neg Pred Value : 0.8650
##               Precision : 0.5114
##                  Recall : 0.5291
##                      F1 : 0.5201
##              Prevalence : 0.2211
##          Detection Rate : 0.1170
##    Detection Prevalence : 0.2288
##       Balanced Accuracy : 0.6928
##
##        'Positive' Class : Default
##
```



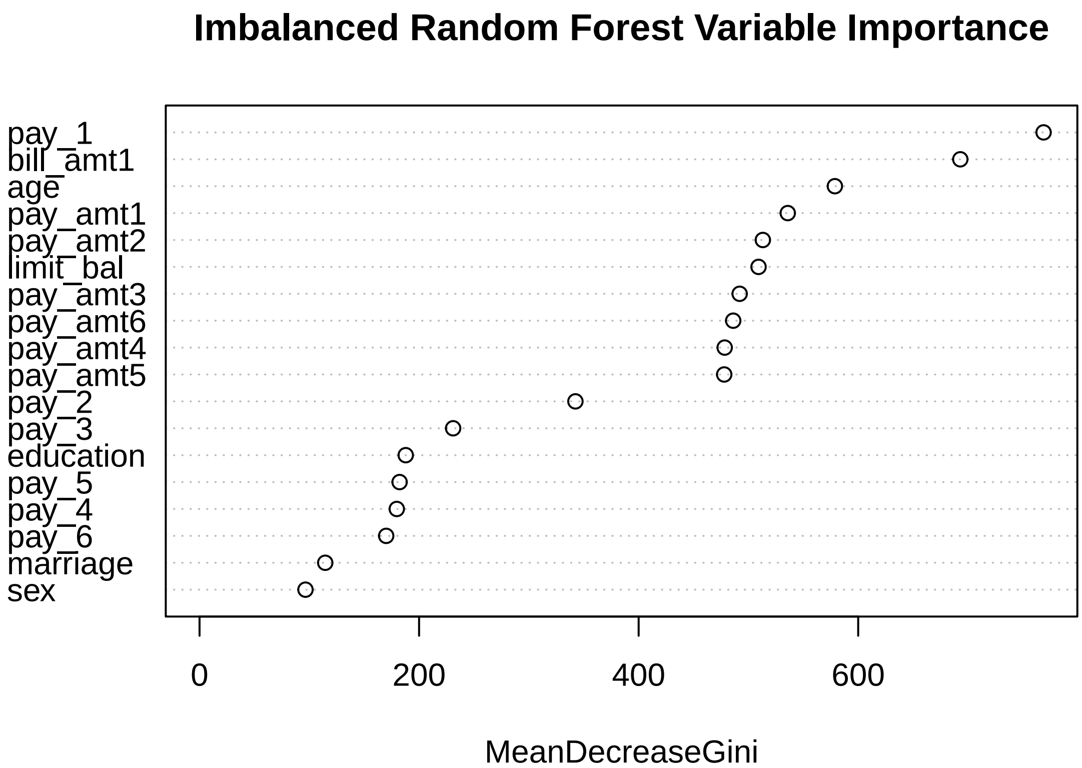

```
## Setting levels: control = NoDefault, case = Default

## Setting direction: controls < cases
```

Random forest - Imbalanced Training Set

```
##
## Call:
##  randomForest(formula = y ~ ., data = train_set)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 18.04%
## Confusion matrix:
##           NoDefault Default class.error
## NoDefault     15492     863  0.05276674
## Default        2926    1720  0.62978907
```

```
oob_accuracy <- 1 - (rf_model$err.rate[nrow(rf_model$err.rate), "OOB"])
  print(paste("OOB Training Accuracy:", oob_accuracy))
```

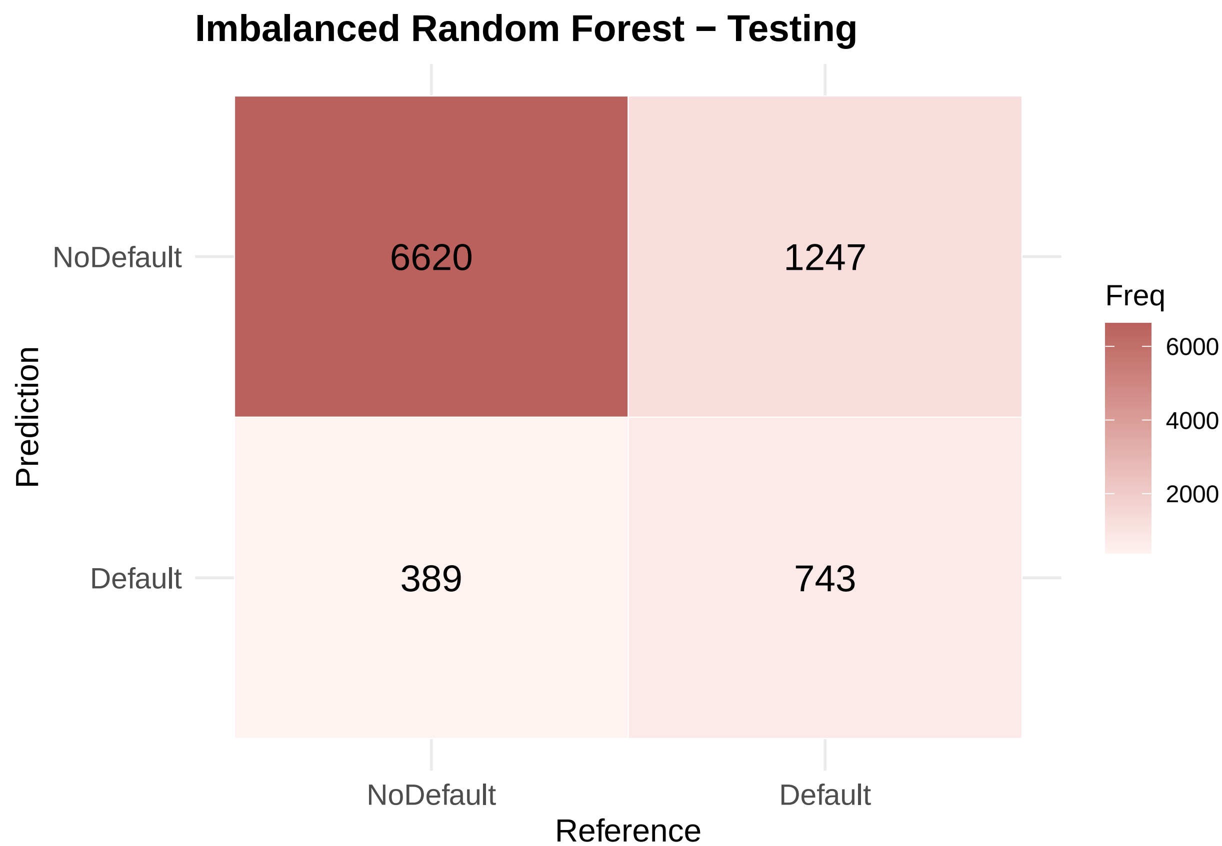
```
## [1] "OOB Training Accuracy: 0.819580019999048"
```




```r
# Testing performance
rf_pred_test <- predict(rf_model, test_set)
con_mat <- confusionMatrix(rf_pred_test, test_set$y, mode = 'everything', positive = "Default")
print(con_mat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault      6620    1247
##   Default         389     743
##
##                Accuracy : 0.8182
##                  95% CI : (0.8101, 0.8261)
##     No Information Rate : 0.7789
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3759
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.37337
##             Specificity : 0.94450
##          Pos Pred Value : 0.65636
##          Neg Pred Value : 0.84149
```

```
##               Precision : 0.65636
##                  Recall : 0.37337
##                      F1 : 0.47598
##              Prevalence : 0.22114
##          Detection Rate : 0.08256
##    Detection Prevalence : 0.12579
##       Balanced Accuracy : 0.65893
##
##        'Positive' Class : Default
##
```




```
## Setting levels: control = NoDefault, case = Default

## Setting direction: controls < cases
```

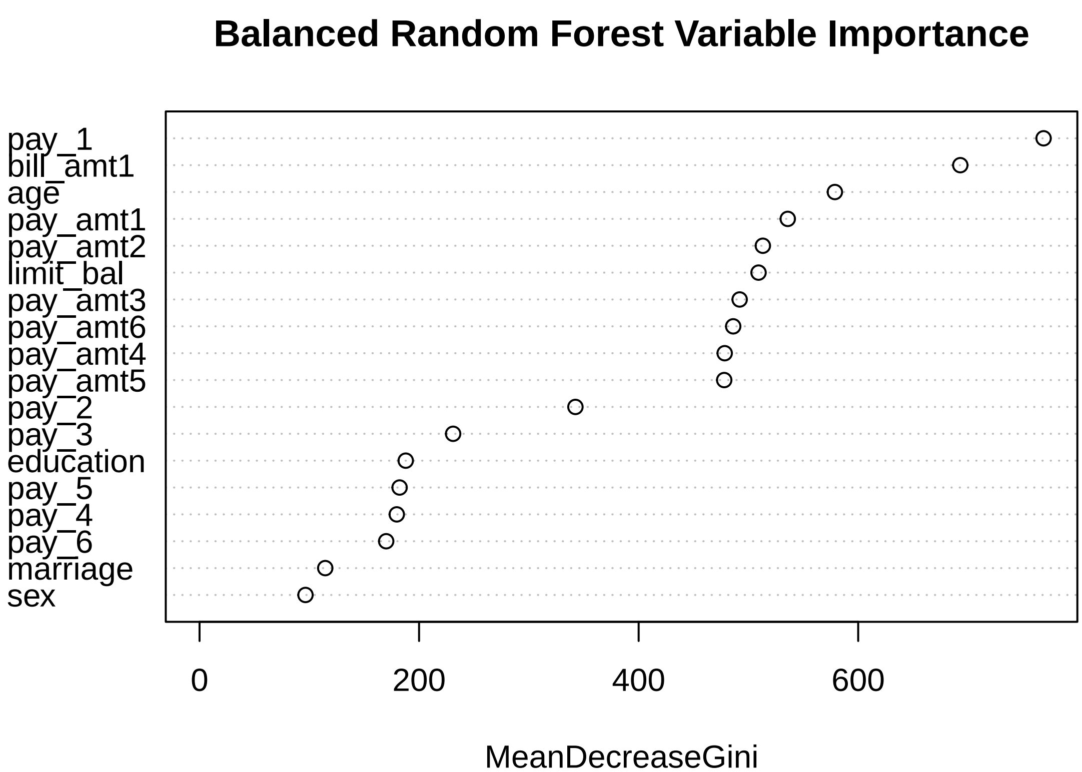
Random forest - Balanced Training Set

```
##
## Call:
##  randomForest(formula = y ~ ., data = train_set_bal)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
```

```
##         OOB estimate of  error rate: 4.99%
## Confusion matrix:
##           NoDefault Default class.error
## NoDefault     14915    1440  0.08804647
## Default         193   16162  0.01180067
```

```
oob_accuracy <- 1 - (rf_model_bal$err.rate[nrow(rf_model_bal$err.rate), "OOB"])
  print(paste("OOB Training Accuracy:", oob_accuracy))
```

```
## [1] "OOB Training Accuracy: 0.950076429226536"
```




```
# Balanced Training performance on original training set
rf_pred_train_bal <- predict(rf_model_bal, train_set)
con_mat <- confusionMatrix(rf_pred_train_bal, train_set$y, mode = 'everything', positive = "Default")
print(con_mat)
```

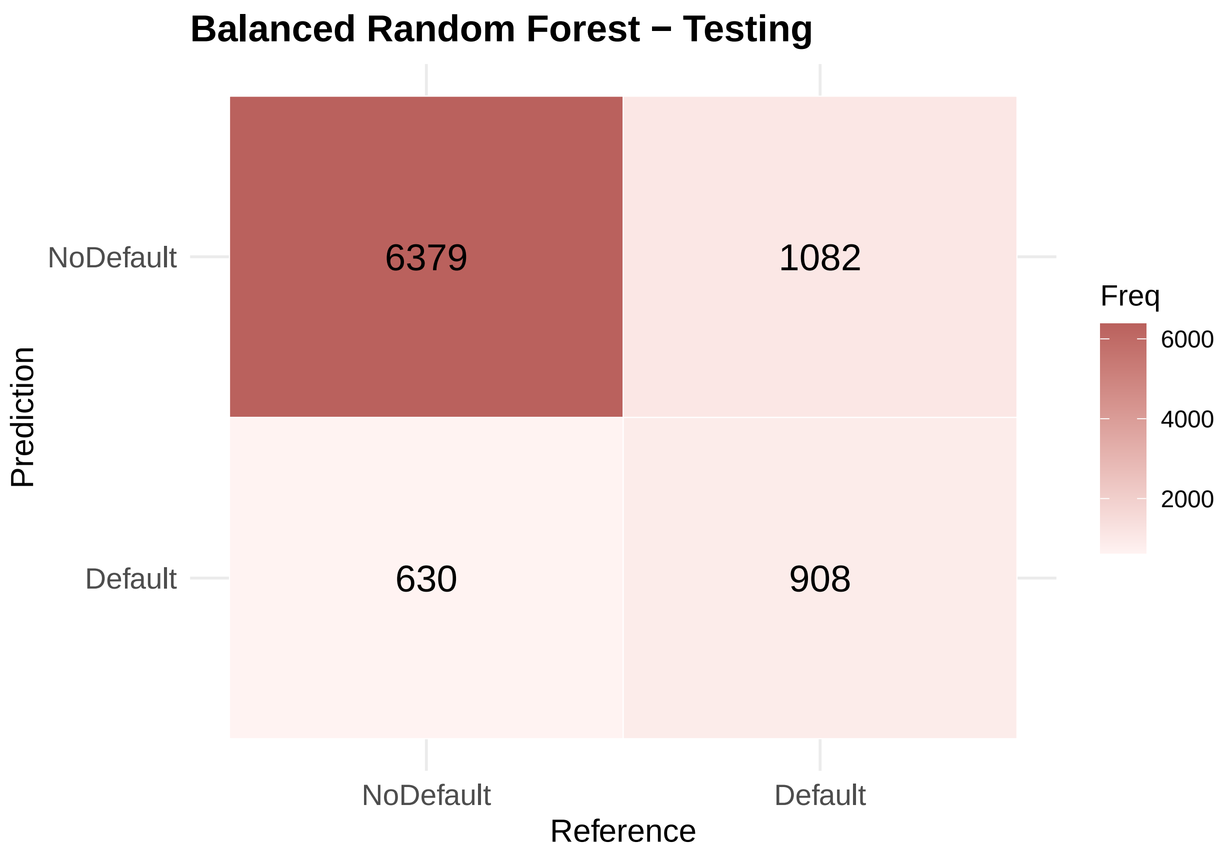
```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault     16195       0
##   Default         160    4646
##
##                Accuracy : 0.9924
```

```
##                  95% CI : (0.9911, 0.9935)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9782
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.9902
##          Pos Pred Value : 0.9667
##          Neg Pred Value : 1.0000
##               Precision : 0.9667
##                  Recall : 1.0000
##                      F1 : 0.9831
##              Prevalence : 0.2212
##          Detection Rate : 0.2212
##    Detection Prevalence : 0.2288
##       Balanced Accuracy : 0.9951
##
##        'Positive' Class : Default
##
```

```r
# Testing performance
rf_pred_test_bal <- predict(rf_model_bal, test_set)
con_mat <- confusionMatrix(rf_pred_test_bal, test_set$y, mode = 'everything', positive = "Default")
print(con_mat)
```

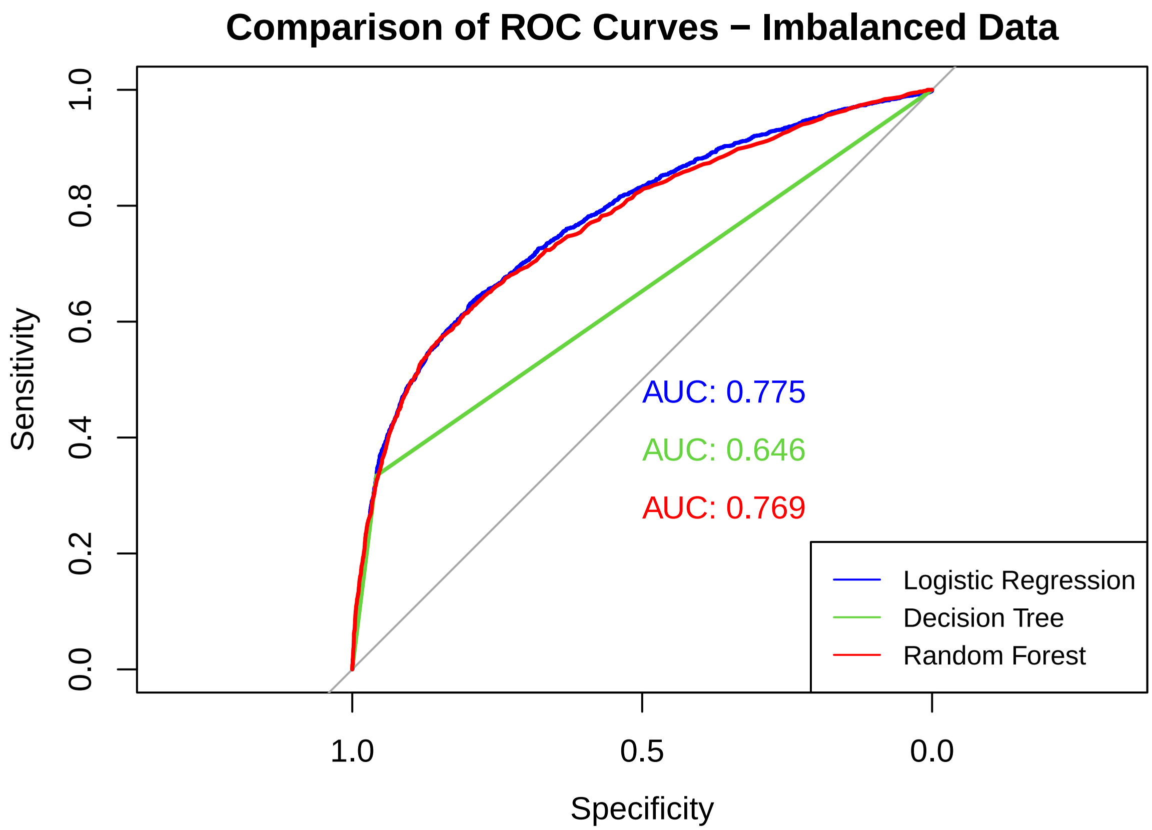
```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  NoDefault Default
##   NoDefault      6379    1082
##   Default         630     908
##
##                Accuracy : 0.8098
##                  95% CI : (0.8015, 0.8178)
##     No Information Rate : 0.7789
##     P-Value [Acc > NIR] : 3.688e-13
##
##                   Kappa : 0.3988
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4563
##             Specificity : 0.9101
##          Pos Pred Value : 0.5904
##          Neg Pred Value : 0.8550
##               Precision : 0.5904
##                  Recall : 0.4563
##                      F1 : 0.5147
##              Prevalence : 0.2211
##          Detection Rate : 0.1009
##    Detection Prevalence : 0.1709
```

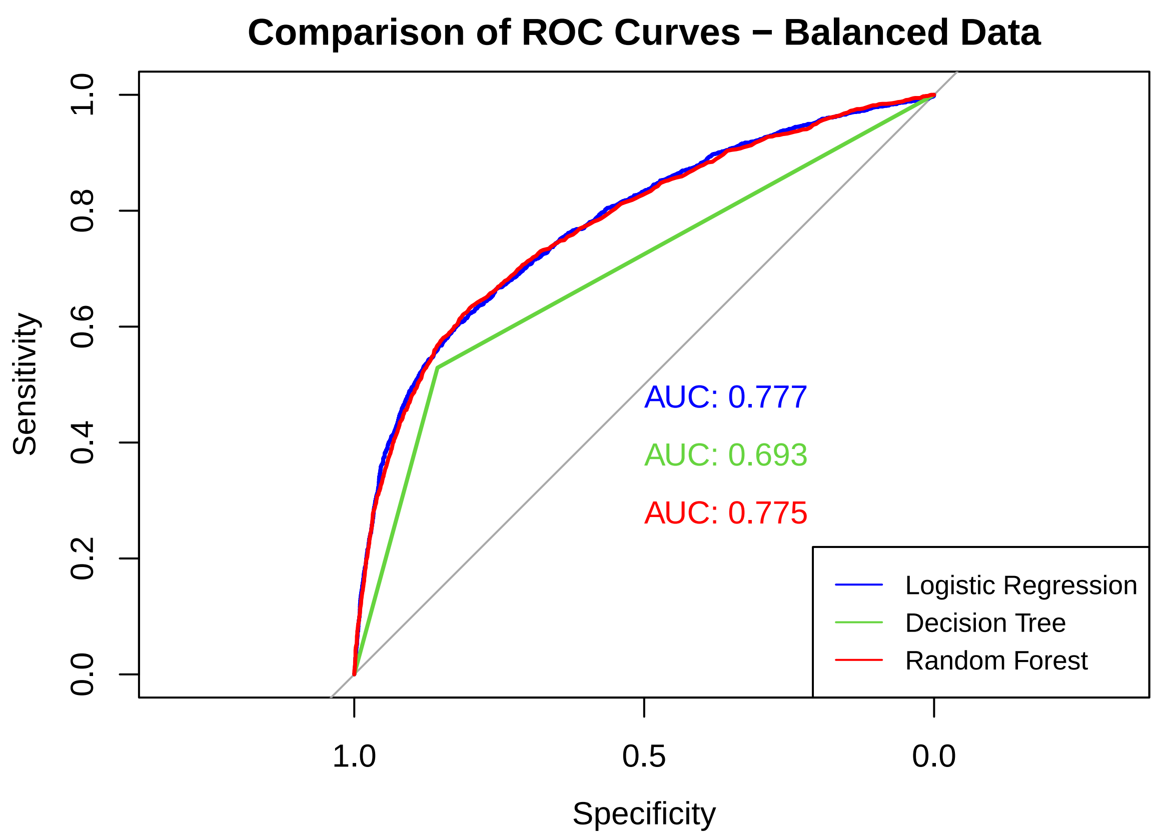
```
##          Balanced Accuracy : 0.6832
##
##           'Positive' Class : Default
##
```




```
## Setting levels: control = NoDefault, case = Default
```

```
## Setting direction: controls < cases
```


Comparison of ROC Curves − Imbalanced Data
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
Specificity
1.0
0.5
0.0
AUC: 0.775
AUC: 0.646
AUC: 0.769
Logistic Regression
Decision Tree
Random Forest


Comparison of ROC Curves − Balanced Data
Sensitivity
1.0
0.8
0.6
0.4
0.2
0.0
1.0
0.5
0.0
Specificity
AUC: 0.777
AUC: 0.693
AUC: 0.775
Logistic Regression
Decision Tree
Random Forest