ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

****

TIỂU LUẬN CUỐI KỲ MÔN

NHẬP MÔN PHÂN TÍCH DỮ LIỆU

ĐỀ TÀI: DỰ ĐOÁN KHOẢN VAY

Mã lớp học phần : MAT1060 04

Nhóm thực hiện : Nhóm 6

Giảng viên hướng dẫn : Tạ Văn Chiến

Hà Nội, tháng 1 năm 2022

Mục lục

I, Lời mở đầu

Trước tiên, chúng em xin chân thành cảm ơn thầy Tạ Văn Chiến đã tận tình giảng dạy và hỗ trợ chúng em trong suốt quá trình học tập. Trong quá trình tìm hiểu cũng như làm báo cáo không thể tránh khỏi những sai sót không đáng có. Chúng em rất mong thầy có thể bỏ qua và góp ý để bản báo cáo của chúng em xuất sắc hơn nữa.

Một lần nữa chúng em xin chân thành cảm ơn!

NHÓM 6

Thành viên :

Lưu Văn Việt - K65 Toán-Tin

Phan Minh Vũ - K65 Toán-Tin

Đặng Công Vinh - K65 Toán-Tin

Phạm Văn Tuấn - K65 Toán-Tin

Khúc Ngọc Tùng - K65 Toán-Tin

Bùi Ngọc Vũ - K65 Toán học

Trần Long Vũ - K65 Toán học

Đào Duy Tùng - K65 CLCMT&KHTT

Nguyễn Việt Tùng - K65 CLCMT&KHTT

Nguyễn Tiến Việt - K65 CLCMT&KHTT

Nguyễn Thành Vinh - K65 CLCMT&KHTT

Giảng viên:

Tạ Văn Chiến

II, Giới thiệu đề tài

1. Giới thiệu về công ty

Công ty Dream Housing Finance giải quyết tất cả các khoản vay mua nhà. Họ có mặt ở khắp các đô thị, bán đô thị và vùng nông thôn. Đầu tiên khách hàng đăng ký vay vốn mua nhà, sau đó công ty xác nhận xem khách hàng có đủ điều kiện vay vốn hay không.

2. Giới thiệu về đề tài

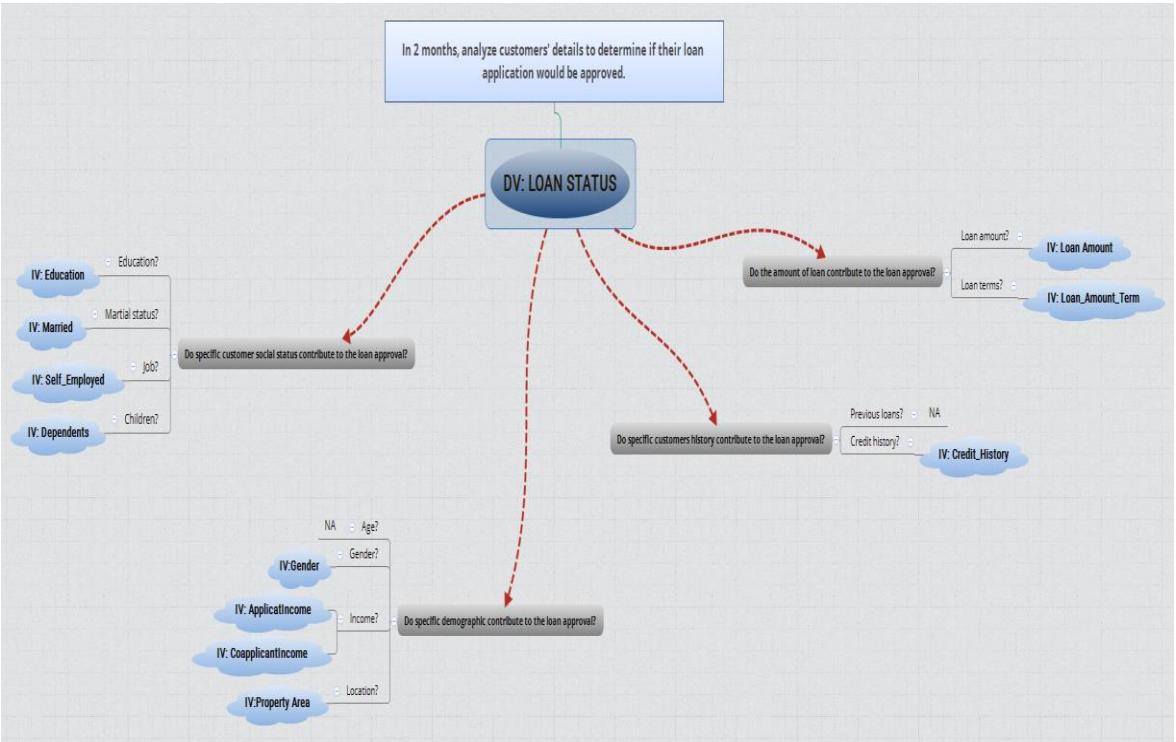
Công ty muốn tự động hóa quy trình đánh giá điều kiện vay vốn dựa trên thông tin khách hàng khi đăng ký trực tuyến. Những thông tin này là Giới tính, Tình trạng hôn nhân, Học vấn, Số người phụ thuộc, Thu nhập, Số tiền vay, Lịch sử tín dụng và những thông tin khác. Để tự động hóa quy trình này, họ đã đưa ra bài toán xác định phân khúc khách hàng - những người đủ điều kiện cho khoản vay để họ có thể nhắm cụ thể vào những khách hàng này. Dưới đây là một phần dữ liệu họ đã cung cấp.

#	**Biến số**	**Mô tả**
1.	Loan_ID	ID khoản vay
2.	Gender	Nam/Nữ (Male/ Female)
3.	Married	Tình trạng hôn nhân (Y/N)
4.	Dependents	Số người phụ thuộc
5.	Education	Tình trạng học vấn (Graduate/ Under Graduate)
6.	Self_Employed	Tự kinh doanh (Y/N)
7.	ApplicantIncome	Thu nhập của người đăng ký
8.	CoapplicantIncome	Thu nhập của người cùng đứng đơn
9.	LoanAmount	Khoản vay (Nghìn đô)
10.	Loan_Amount_Term	Kỳ hạn khoản vay (Tháng)
11.	Credit_History	Lịch sử tín dụng
12.	Property_Area	Urban/ Semi Urban/ Rural Đô thị/Bán đô thị/Nông thôn
13.	Loan_Status	Khoản vay được phê duyệt (Y/N)

III, Lập cấu trúc kế hoạch phân tích

Điều đầu tiên chúng ta cần làm trước khi phân tích dữ liệu là hiểu vấn đề và tạo ra mục tiêu SMART (Những nguyên tắc để định hình và thực hiện mục tiêu trong tương lai). Bước tiếp theo là xác định các biến độc lập và các biến phụ thuộc. Sơ đồ tư duy dưới đây mô tả quá trình chúng ta xây dựng cấu trúc kế hoạch cho dự án.

- S-Specific: Tính cụ thể dễ hiệu
- M-Measurable: Có thể đo lường được
- A-Atainable: Tính khả thi
- R-Realistic: Tính thực tế
- T-Time bound: Thiết lập thời gian



IV, Kiểm tra dữ liệu

Bước tiếp theo là xem xét dữ liệu chúng ta làm việc. Trên thực tế, hầu hết mọi dữ liệu chúng ta có được, kể cả từ chính phủ, đều có thể có sai sót, và việc xác

định những lỗi này trước khi phân tích dữ liệu rất quan trọng. Thông thường chúng ta phải trả lời các câu hỏi sau:

- Chúng ta có tìm thấy điều gì đó sai trong dữ liệu hay không?
- Có những biến không rõ ràng trong tập dữ liệu không?
- Những dữ liệu này nên được sửa đổi hay loại bỏ?

Hãy bắt đầu bằng việc đọc dữ liệu bằng hàm **read.csv()** và hiển thị phần đầu của tập dữ liệu:

1. Đọc dữ liệu và xem xét khái quát dữ liệu

Dữ liệu được chúng em lấy từ "https://github.com/DataBoosting/Loan-Prediction-with-R/blob/master/train.csv".

Báo cho R biết nơi chứa dữ liệu và đọc dữ liệu:

> setwd("C:/Users/USER/Desktop/Coaching/Projects/Loan Prediction")

> tr <- read.csv('train.csv', header = TRUE)

Hiển thị phần đầu của tập dữ liệu:

> head(tr)

```
##    Loan_ID Gender Married Dependents  Education   Self_Employed
## 1 LP001002   Male      No          0   Graduate           No
## 2 LP001003   Male     Yes          1   Graduate           No
## 3 LP001005   Male     Yes          0   Graduate          Yes
## 4 LP001006   Male     Yes          0 Not Graduate       No
## 5 LP001008   Male      No          0   Graduate           No
## 6 LP001011   Male     Yes          2   Graduate          Yes
##   ApplicantIncome CoapplicantIncome LoanAmount Loan_Amount_Term
## 1            5849                0         NA               360
## 2            4583             1508        128               360
```

```
## 3         3000                0                  66              360
## 4         2583             2358                  120             360
## 5         6000                0                 141              360
## 6         5417             4196                  267             360
##   Credit_History  Property_Area  Loan_Status
## 1              1          Urban            Y
## 2              1          Rural            N
## 3              1          Urban            Y
## 4              1          Urban            Y
## 5              1          Urban            Y
## 6              1          Urban            Y
```

Hàng đầu tiên của tập dữ liệu mô tả tiêu đề mỗi cột. Những tiêu đề này đã được mô tả trong bảng bên trên. Bây giờ, chúng ta sẽ sử dụng hàm **summary()** để có cái nhìn tổng quan về dữ liệu:

> summary(tr)

```
##     Loan_ID        Gender      Married     Dependents             Education
##  LP001002:  1          : 13          :  3          : 15       Graduate    :480
##  LP001003:  1   Female:112      No  :213          0 :345      Not Graduate:134
##  LP001005:  1   Male   :489      Yes :398          1 :102
##  LP001006:  1                                    2 :101
##  LP001008:  1                                    3+: 51
##  LP001011:  1
##  (Other) :608
##   Self_Employed  ApplicantIncome  CoapplicantIncome      LoanAmount
```

```
##       : 32                Min.    :  150        Min.     :    0      Min.     :  9.0
##  No :500                  1st Qu.  : 2878        1st Qu.  :    0      1st Qu.  :100.0
##  Yes: 82                  Median : 3812        Median : 1188      Median :128.0
##                           Mean    : 5403        Mean    : 1621      Mean    :146.4
##                           3rd Qu. : 5795        3rd Qu. : 2297      3rd Qu.:168.0
##                           Max.    :81000        Max.    :41667      Max.    :700.0
##                                                                      NA's    :22
##  Loan_Amount_Term    Credit_History        Property_Area      Loan_Status
##  Min.    : 12              Min.    :0.0000      Rural        :179      N:192
##  1st Qu. :360              1st Qu. :1.0000      Semiurban:233      Y:422
##  Median :360               Median :1.0000      Urban        :202
##  Mean    :342              Mean    :0.8422
##  3rd Qu. :360              3rd Qu. :1.0000
##  Max.    :480              Max.    :1.0000
##  NA's    :14                  NA's    :50
```

Dưới đây là một vài kết luận chúng ta có được từ việc xem xét kết quả:

- Có ký hiệu (+) ở 51 dữ liệu của biến Dependents.
- Trung bình của Credit_History là 0.8422 trong khi biến này nhận giá trị là 1 cho các khách hàng có lịch sử tín dụng và 0 nếu ngược lại.
- Có những trường trống trong Gender, Married, Dependents và Self_Employed.
- Có những dữ liệu không xác định ở LoanAmount, Loan_Amount_term và Credit_History.

2. Sửa nhanh một số biến

> setwd("C:/Users/USER/Desktop/Coaching/Projects/Loan Prediction")

```
> tr <- read.csv(file="train.csv", na.strings=c("", "NA"), header=TRUE)
> library(plyr)
> tr$Dependents <- revalue(tr$Dependents, c("3+"="3"))
```

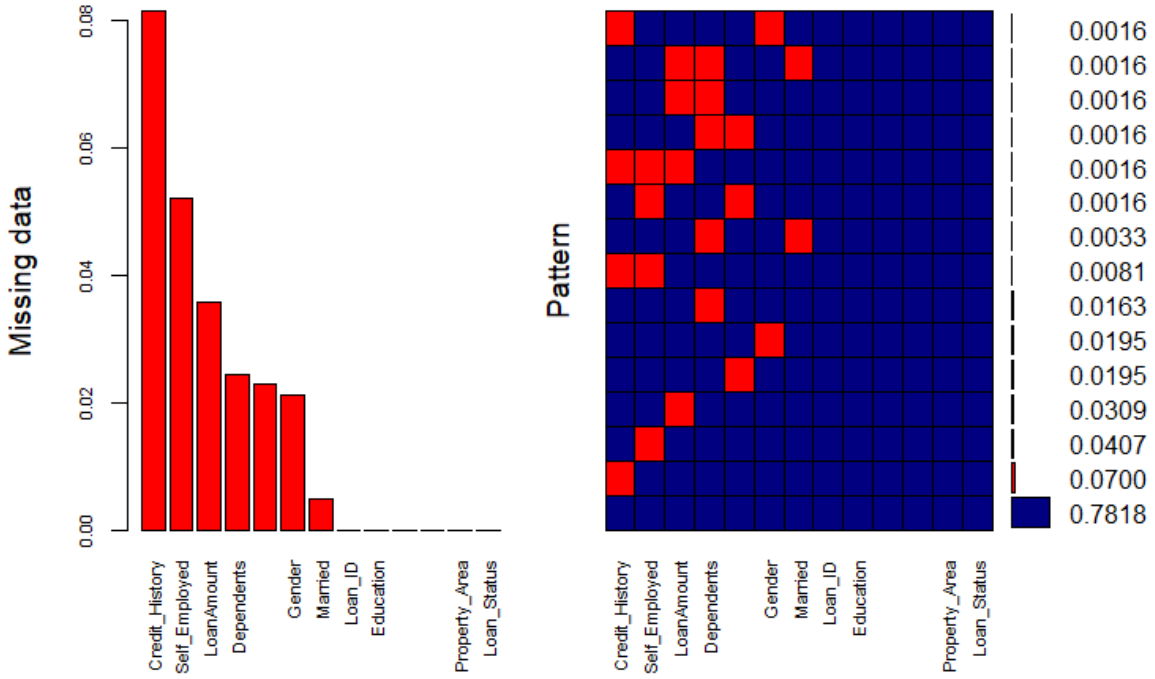
Sử dụng hàm **sapply()** để hiển thị số dữ liệu bị trống ở các biến:

```
> sapply(tr, function(x) sum(is.na(x)))
##       Loan_ID        Gender       Married    Dependents
##             0            13             3            15
##     Education  Self_Employed ApplicantIncome CoapplicantIncome
##             0            32             0             0
## LoanAmount  Loan_Amount_Term Credit_History  Property_Area
##            22            14            50             0
##  Loan_Status
##             0
```

Sử dụng hàm **aggr()** để vẽ biểu đồ thể hiện dữ liệu trống ở các biến:

```
> library(mice)
> library(VIM)
> mice_plot <- aggr(tr, col=c('navyblue','red'),
                    numbers=TRUE, sortVars=TRUE,
                    labels=names(tr), cex.axis=.7,
                    gap=3, ylab=c("Missing data","Pattern"))
##  Variables sorted by number of missings:
##              Variable       Count
##       Credit_History 0.081433225
```

```
##          Self_Employed   0.052117264
##             LoanAmount   0.035830619
##             Dependents   0.024429967
##   Loan_Amount_Term 0.022801303
##                 Gender  0.021172638
##                Married  0.004885993
##                Loan_ID  0.000000000
##              Education  0.000000000
##        ApplicantIncome  0.000000000
##    CoapplicantIncome  0.000000000
##          Property_Area  0.000000000
##             Loan_Status  0.000000000
```



Từ 2 biểu đồ, ta thấy dữ liệu trống xuất hiện ở 7 biến. 8 biến không chứa dữ liệu trống.

Dựa vào biểu đồ bên phải, ta đọc được tỉ lệ dữ liệu trống ở các biến. Biến Credit_History chứa nhiều dữ liệu trống nhất với tỉ lệ 0.081433225. Biến Married chứa ít dữ liệu trống nhất với tỉ lệ 0.004885993.

Dựa vào biểu đồ bên trái, ta đọc được phân bố của dữ liệu trống giữa các biến. Số các hàng không có dữ liệu trống chiếm tỉ lệ 0.7818. Số các hàng mà chỉ có biến Credit_History có dữ liệu trống chiếm tỉ lệ là 0.07, trong khi đó số các hàng mà cả biến Credit_History và Self_Employed cùng có dữ liệu trống là 0.008. Nhận xét tương tự với các biến còn lại.

3. Xem xét phân bố của một số biến

Biểu đồ histogram và biểu đồ boxplot cho hai biến LoanAmount và ApplicantIncome:

```
> par(mfrow=c(2,2))
> hist(tr$LoanAmount,
    main="Histogram for LoanAmount",
    xlab="Loan Amount",
    border="blue",
    col="maroon",
    las=1,
    breaks=20, prob = TRUE)
> boxplot(tr$LoanAmount, col='maroon',xlab = 'LoanAmount', main = 'Box Plot for Loan Amount')
> hist(tr$ApplicantIncome,
    main="Histogram for Applicant Income",
    xlab="Income",
    border="blue",
```

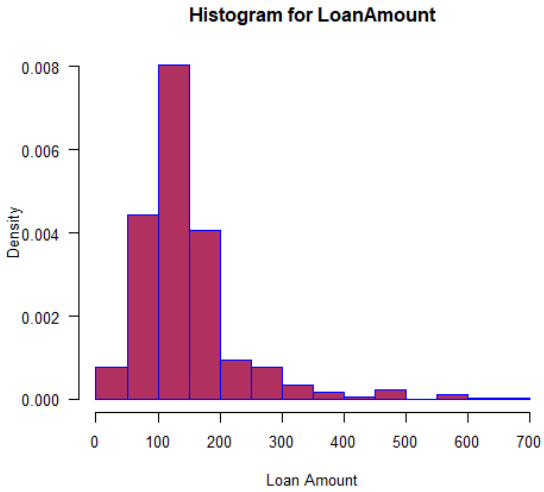
col="maroon",

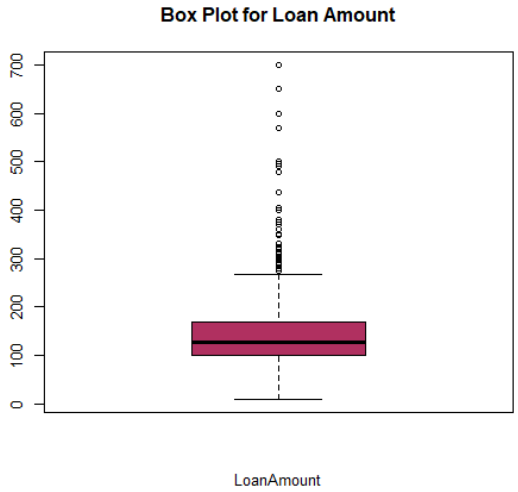
las=1,

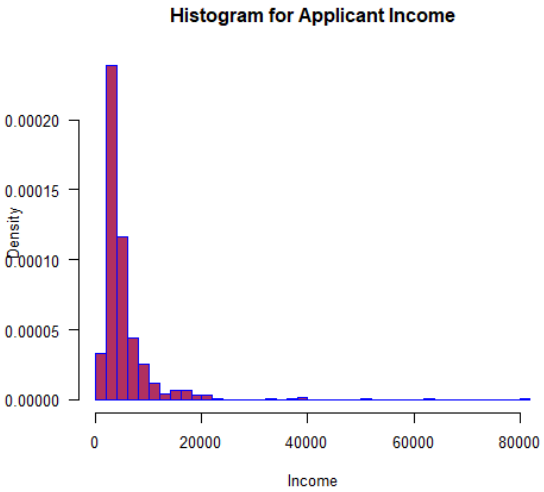
breaks=50, prob = TRUE)

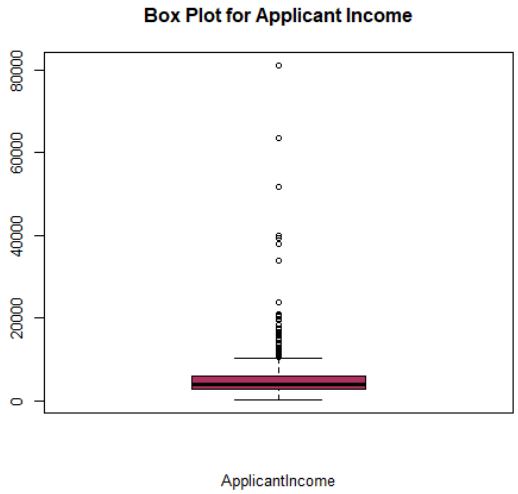
> boxplot(tr$ApplicantIncome, col='maroon',xlab = 'ApplicantIncome', main = 'Box Plot for Applicant Income')

> dev.off()





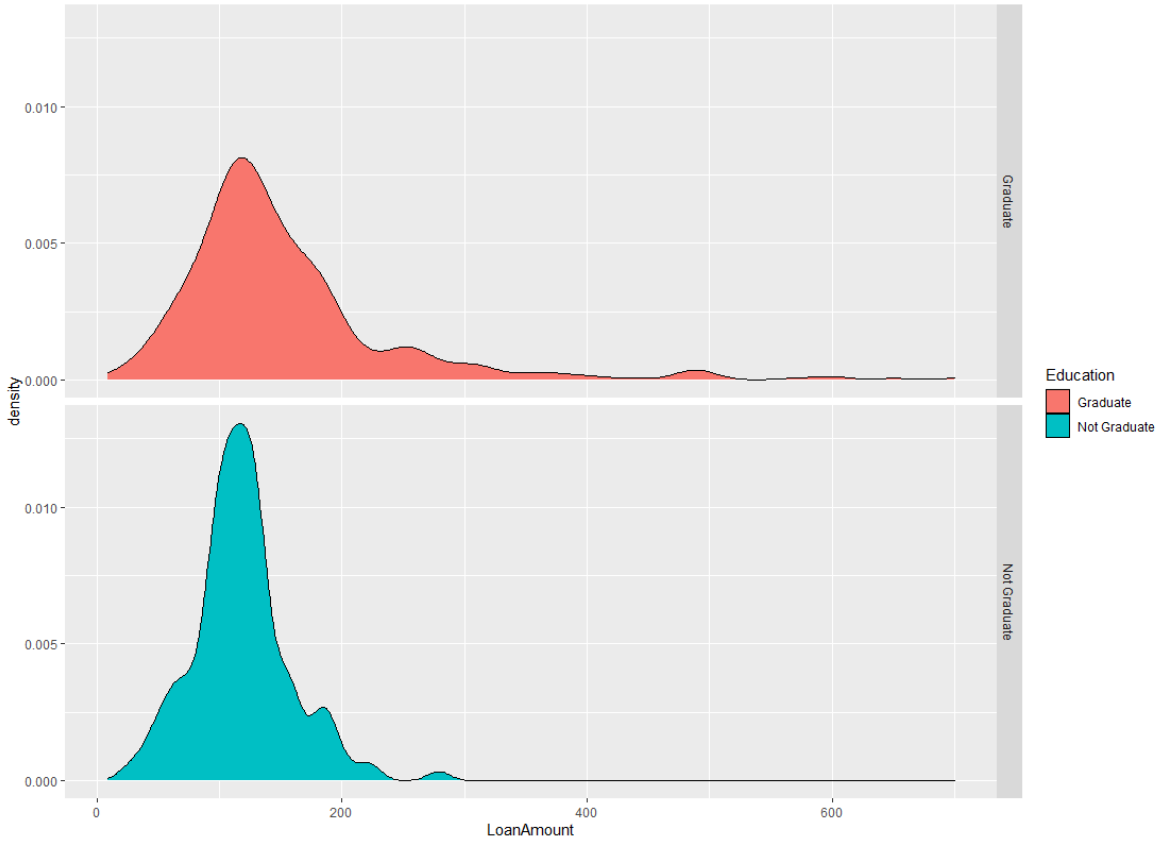




Chúng ta thấy rằng có giá trị đột xuất trong cả hai biến. Hãy kiểm tra xem phân bố khoản vay của người nộp đơn có chịu ảnh hưởng bởi trình độ học vấn của họ hay không :

> library(ggplot2)

```
> data(tr, package="lattice")

> ggplot(data=tr, aes(x=LoanAmount, fill=Education)) +

  geom_density() +

  facet_grid(Education~.)
```

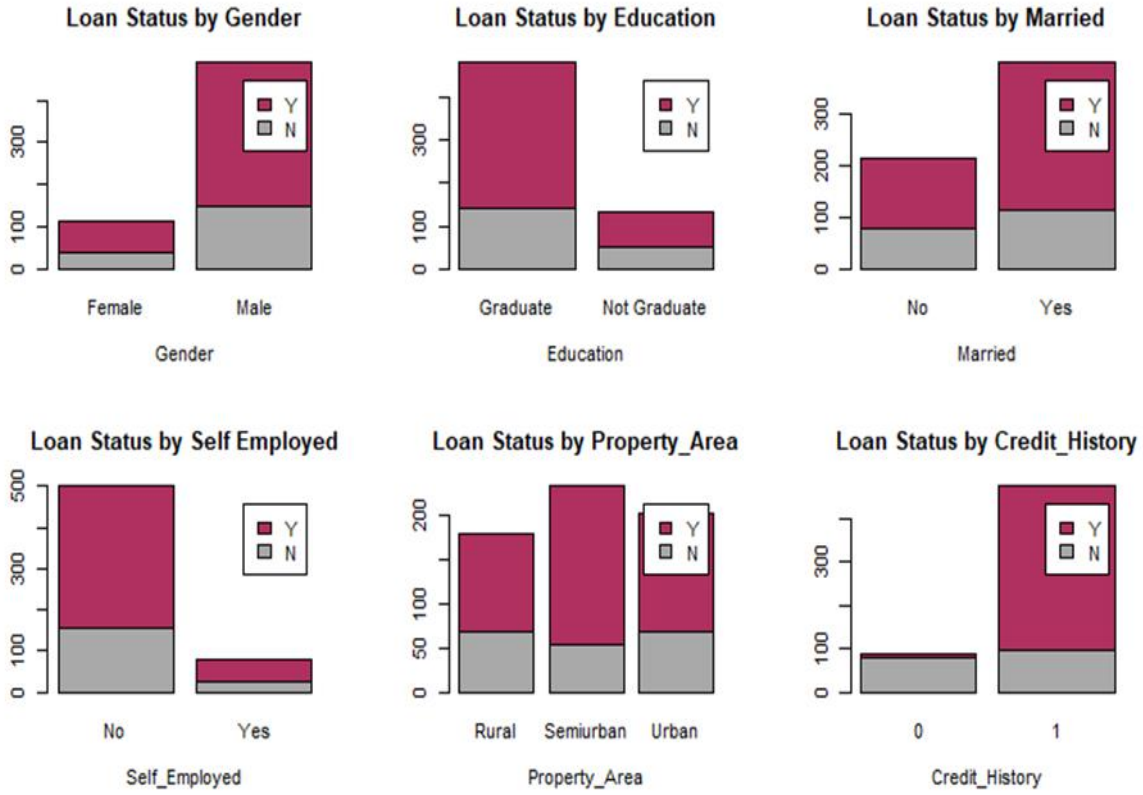


Chúng ta nhận thấy rằng những người đã tốt nghiệp có nhiều giá trị đột xuất hơn và phân bố khoản vay của họ cũng rộng hơn.

4. Biểu đồ cho các biến phân loại

```
> par(mfrow=c(2,3))

> counts <- table(tr$Loan_Status, tr$Gender)

> barplot(counts, main="Loan Status by Gender",

      xlab="Gender", col=c("darkgrey","maroon"),

      legend = rownames(counts))

> counts2 <- table(tr$Loan_Status, tr$Education)
```

```
> barplot(counts2, main="Loan Status by Education",
      xlab="Education", col=c("darkgrey","maroon"),
      legend = rownames(counts2))
> counts3 <- table(tr$Loan_Status, tr$Married)
> barplot(counts3, main="Loan Status by Married",
      xlab="Married", col=c("darkgrey","maroon"),
      legend = rownames(counts3))
> counts4 <- table(tr$Loan_Status, tr$Self_Employed)
> barplot(counts4, main="Loan Status by Self Employed",
      xlab="Self_Employed", col=c("darkgrey","maroon"),
      legend = rownames(counts4))
> counts5 <- table(tr$Loan_Status, tr$Property_Area)
> barplot(counts5, main="Loan Status by Property_Area",
      xlab="Property_Area", col=c("darkgrey","maroon"),
      legend = rownames(counts5))
> counts6 <- table(tr$Loan_Status, tr$Credit_History)
> barplot(counts6, main="Loan Status by Credit_History",
      xlab="Credit_History", col=c("darkgrey","maroon"),
      legend = rownames(counts5))
> dev.off()
```



Nếu chúng ta nhìn vào biểu đồ Gender, chúng ta nhận thấy rằng nam giới có nhiều hồ sơ hơn một nửa số đơn đăng ký được chấp nhận. Có ít nữ giới hơn nhưng vẫn nhiều hơn một nửa số đơn đăng ký được chấp nhận. Nhận xét tương tự với các biểu đồ khác để đánh giá được mối quan hệ giữa các biến phân loại và những hồ sơ được chấp thuận.

V, Làm sạch dữ liệu

1. Gán dữ liệu trống bằng hàm mice().

Chúng ta xác định được rất nhiều lỗi trong tập dữ liệu, chúng ta phải sửa chúng trước khi tiếp tục phân tích. Các vấn đề của chúng ta là:

- Có dữ liệu trống trong một số biến. Dựa vài tầm quan trọng của biến, chúng ta sẽ quyết định phương pháp để sử dụng.

- Dựa vào phân bố của dữ liệu, chúng ta nhận thấy Thu nhập của khách hàng và Khoản vay có các giá trị đột xuất.

Việc khắc phục những dữ liệu đột xuất đòi hỏi phải khéo léo. Rất khó để biết liệu chúng có phải do lỗi đo lường, lỗi khi ghi lại, hay các giá trị đột xuất là thực sự bất thường hay không. Nếu chúng ta quyết định loại bỏ giá trị đó, chúng ta phải dẫn chứng lý do cho việc này.

Ở tập dữ liệu này, chúng ta sẽ thừa nhận rằng các giá trị bị thiếu là có hệ thống bởi vì dữ liệu trống xuất hiện trong các biến nào đó một cách ngẫu nhiên. Ngoài ra, chúng ta nhận thấy rằng dữ liệu trống có ở cả dữ liệu định tính và dữ liệu định lượng, vì vậy, chúng ta sẽ sử dụng gói mice trong R.

Gói này giúp ta gán những giá trị trống bằng giá trị dữ liệu hợp lý. Những giá trị này được suy ra từ một phân bố được thiết kế cho mỗi điểm dữ liệu bị thiếu. Ở biểu đồ dữ liệu bị thiếu ở trên, chúng ta thấy rằng có 0.78 dữ liệu không thiếu bất kỳ thông tin nào, 0.07 là những giá trị Credit_History bị thiếu, và phần còn lại cho biết những mẫu bị thiếu khác.

Hàm **mice()** sẽ thực hiện quá trình gán dữ liệu :

> imputed_Data <- mice(tr, m=2, maxit = 2, method = 'cart', seed = 500)

iter imp variable

1 1 Gender Married Dependents Self_Employed LoanAmount Loan_Amount_Term Credit_History

1 2 Gender Married Dependents Self_Employed LoanAmount Loan_Amount_Term Credit_History

2 1 Gender Married Dependents Self_Employed LoanAmount Loan_Amount_Term Credit_History

2 2 Gender Married Dependents Self_Employed LoanAmount Loan_Amount_Term Credit_History

Đối số **'m'** trong hàm cho biết chúng ta muốn thực hiện quá trình gán dữ liệu bao nhiêu lần, để đơn giản, ta sẽ chọn m=2. Đối số **'method'** cho biết phương pháp gán mà chúng ta sử dụng. Ta chọn **"CART"** là viết tắt **"Classification and**

Regression trees" - "Cây phân loại và hồi quy". Phương pháp này hoạt động với tất cả kiểu biến, đó là lý do ta chọn nó. Cuối cùng ta kết hợp dữ liệu đã được gán vào tập dữ liệu ban đầu bằng hàm **complete()** :

> tr <- complete(imputed_Data,2)

Ở đây ta chọn lần gán dữ liệu thứ hai. Kiểm tra dữ liệu bị thiếu lại lần nữa, chúng ta thấy rằng không còn dữ liệu trống sau khi gán.

> sapply(tr, function(x) sum(is.na(x)))

```
##          Loan_ID           Gender          Married       Dependents
##                0                0                0                0
##        Education    Self_Employed ApplicantIncome CoapplicantIncome
##                0                0                0                0
##       LoanAmount Loan_Amount_Term   Credit_History    Property_Area
##                0                0                0                0
##      Loan_Status
##                0
```

2. Xử lý dữ liệu đột xuất bằng hàm log().

Bây giờ chúng ta sẽ xử lý dữ liệu đột xuất. Nhìn vào biến LoanAmount, chúng ta dự đoán rằng các giá trị đột xuất xảy ra với một số khách hàng, vì một số lý do, muốn đăng ký để có khoản vay lớn hơn. Chúng ta sẽ thực hiện chuyển đổi logarit để chuẩn hóa dữ liệu.

> tr$LogLoanAmount <- log(tr$LoanAmount)

> par(mfrow=c(1,2))

> hist(tr$LogLoanAmount,

main="Histogram for Loan Amount",

xlab="Loan Amount",

border="blue",

col="maroon",

las=1,

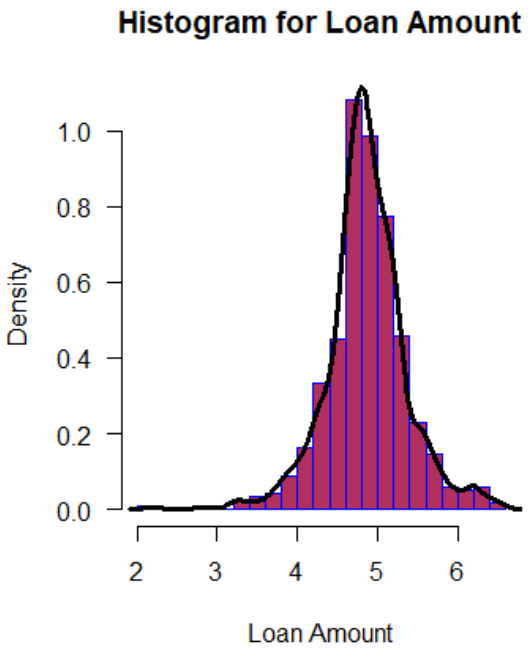
breaks=20, prob = TRUE)

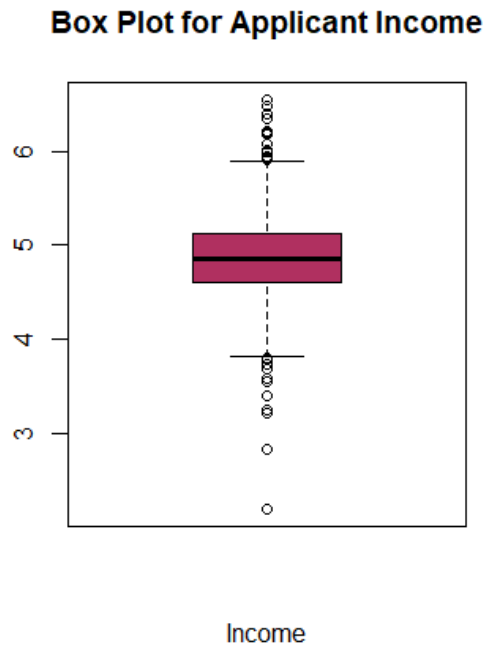
> lines(density(tr$LogLoanAmount), col='black', lwd=3)

> boxplot(tr$LogLoanAmount, col='maroon',xlab = 'Income', main = 'Box Plot for Applicant Income')

> dev.off()

Giờ phân bố đã gần với phân bố chuẩn và ảnh hưởng bởi các giá trị đột xuất đã giảm xuống đáng kể.



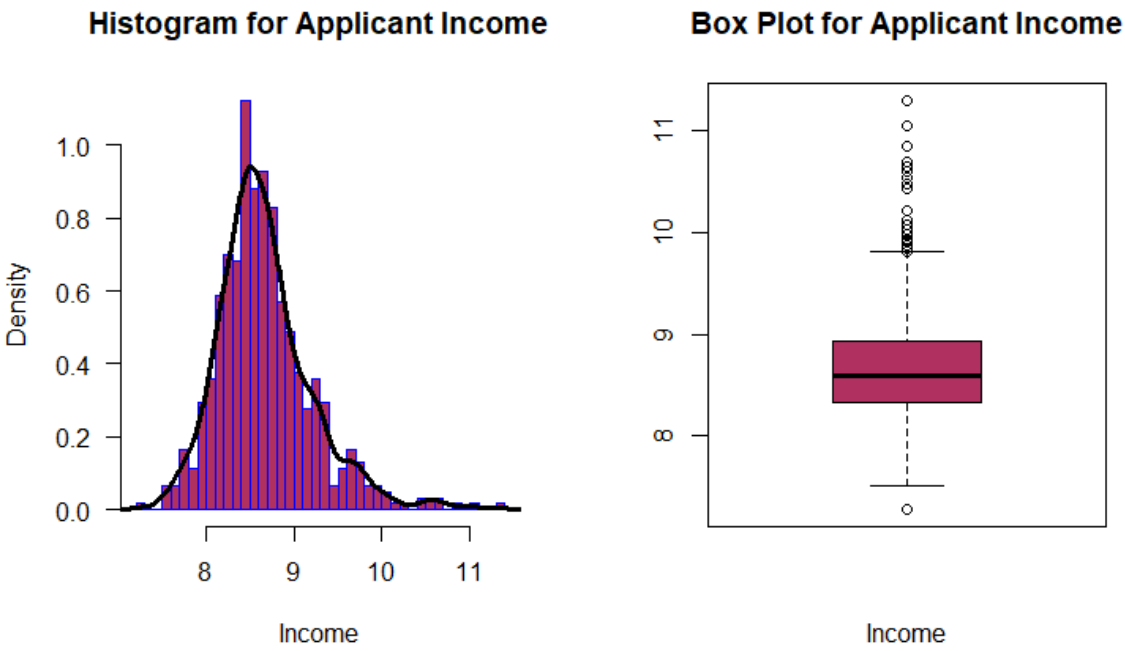


Đến với ApplicantIncome, sẽ là một ý tưởng hay khi kết hợp ApplicantIncome và CoapplicantIncome thành tổng thu nhập và thực hiện việc chuyển đổi logarit cho biến kết hợp này.

> tr$Income <- tr$ApplicantIncome + tr$CoapplicantIncome

> tr$ApplicantIncome <- NULL

```
> tr$CoapplicantIncome <- NULL


> tr$LogIncome <- log(tr$Income)
> par(mfrow=c(1,2))
> hist(tr$LogIncome,
    main="Histogram for Applicant Income",
    xlab="Income",
    border="blue",
    col="maroon",
    las=1,
    breaks=50, prob = TRUE)
> lines(density(tr$LogIncome), col='black', lwd=3)
> boxplot(tr$LogIncome, col='maroon',xlab = 'Income', main = 'Box Plot for
Applicant Income')
> dev.off()
```



Chúng ta thấy rằng phân bố đã gần với phân bố chuẩn.

VI, Xây dựng mô hình dự đoán

Bây giờ chúng ta sẽ đến với bước quan trọng tiếp theo trong phân tích dữ liệu, đó là tách dữ liệu thành **training set** và **test set**.

Training set là một tập dữ liệu mà chúng ta huấn luyện các mô hình và **test set** là tập dữ liệu được sử dụng để tính độ chính xác hoặc sai số của mô hình dự đoán đã được huấn luyện. Thông thường training set và testing set được tách ra ngay từ dữ liệu quan sát được cung cấp. Đối với mô hình này, ta sẽ lấy 70% dữ liệu quan sát được cung cấp để huấn luyện và 30% dữ liệu quan sát còn lại không liên quan đến 70% trước để đánh giá.

```
> set.seed(42)
> sample <- sample.int(n = nrow(tr), size = floor(.70*nrow(tr)), replace = F)
> trainnew <- tr[sample, ]
> testnew  <- tr[-sample, ]
```

Khái niệm Overfitting: Overfitting là hiện tượng mô hình tìm được quá khớp với dữ liệu training. Việc quá khớp này có thể dẫn đến việc dự đoán nhầm nhiễu, và

chất lượng mô hình không còn tốt trên dữ liệu test nữa. Overfitting xảy ra khi mô hình quá phức tạp để mô phỏng training data. Điều này đặc biệt xảy ra khi lượng dữ liệu training quá nhỏ trong khi độ phức tạp của mô hình quá cao

VII, Hồi quy Logistic

1. Khái niệm

Khái niệm: Phân tích hồi qui logistic là một kỹ thuật thống kê để xem xét mối liên hệ giữa biến độc lập (biến liên tục hay không liên tục) với biến phụ thuộc là biến nhị phân (Có/Không, Xảy ra/Không xảy ra).

Mô hình hồi quy Logistic: Giả sử ta cần biết mối liên hệ giữa biến độc lập x và biến phụ thuộc y. Khi đó, vấn đề mà chúng ta cần biết có thể phát biểu bằng ngôn ngữ mô hình như sau:

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \alpha + \beta x$$

$$\Rightarrow p = \frac{\exp(\alpha + \beta x)}{1+\exp(\alpha + \beta x)}$$

Trong đó:

- p là xác suất xảy ra y
- α và β là hai thông số tuyến tính cần phải ước tính từ dữ liệu nghiên cứu.

2. Xây dựng mô hình hồi quy Logistic bằng R

2.1. Xây dựng mô hình hồi quy với 1 biến

Chúng ta sẽ sử dụng biến Credit_History trong mô hình hồi quy logistic đầu tiên.

>logistic1 <- glm (Loan_Status ~ Credit_History, data = trainnew, family = binomial)

>summary(logistic1)

Call:

glm(formula = Loan_Status ~ Credit_History, family = binomial,

```
##    data = trainnew)

## Deviance Residuals:

##    Min       1Q   Median       3Q      Max

## -1.7334  -0.4912   0.7097   0.7097   2.0856


## Coefficients:

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

## (Intercept)     -2.0541     0.4750  -4.324 1.53e-05 ***

## Credit_History   3.3047     0.4954   6.671 2.55e-11 ***

## ---

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


## (Dispersion parameter for binomial family taken to be 1)


## Null deviance: 415.78  on 335  degrees of freedom

## Residual deviance: 340.78  on 334  degrees of freedom

## AIC: 344.78

## Number of Fisher Scoring iterations: 4
```

Trong đầu ra, dòng đầu tiên chúng ta thấy là “call”, đây là R nhắc nhở chúng ta về mô hình mà chúng ta đã chạy. Phần thứ nhất của kết quả là Deviance, Devicance phản ánh độ khác biệt giữa mô hình và dữ liệu. Phần kế tiếp cung cấp hệ số α (intercept) và β (Credit_History) và sai số chuẩn. Qua kết quả này, ta có α= -2,0541 và β = 3.3047. Phần cuối của kết quả cung cấp Deviance cho hai mô hình: Mô hình không có biến độc lập và mô hình có biến độc lập. Ngoài ra, R còn cung cấp giá trị của AIC được tính từ Deviance và bậc tự do.

Chúng ta sẽ tạo một bảng đánh giá kết quả để kiểm tra độ chính xác của mô hình:

> my_prediction_tr1 <- predict(logistic1, newdata = trainnew, type = "response")

> table(trainnew$Loan_Status, my_prediction_tr1 > 0.5)

FALSE TRUE

0 39 65

1 5 227

Độ chính xác của mô hình trên tập huấn luyện là 79,16%.

Thực hiện tương tự với tập kiểm nghiệm, ta được độ chính xác của mô hình là 83,24%.

2.2 Xây dựng mô hình hồi quy Logistic với nhiều biến

> logistic2 <- glm (Loan_Status ~ Credit_History+Education+Self_Employed+Property_Area+LogLoanAmount + LogIncome,data = trainnew, family = binomial)

> summary(logistic2)

> my_prediction_tr2 <- predict(logistic2, newdata = trainnew, type = "response")

> table(trainnew$Loan_Status, my_prediction_tr2 > 0.5)

> logistic_test2 <- glm (Loan_Status ~ Credit_History+Education+Self_Employed+Property_Area+LogLoanAmount + LogIncome,data = testnew, family = binomial)

> summary(logistic_test2)

> my_prediction_te2 <- predict(logistic_test2, newdata = testnew, type = "response")

> table(testnew$Loan_Status, my_prediction_te2 > 0.5)

Call:

```
## glm(formula = Loan_Status ~ Credit_History + Education + Self_Employed +
##     Property_Area + LogLoanAmount + LogIncome, family = binomial,
##     data = trainnew)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3203  -0.4114   0.5504   0.6895   2.2706
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -2.5647     2.2809  -1.124   0.2608
## Credit_History   3.6209     0.4579   7.907 2.64e-15 ***
## Education2      -0.7175     0.3073  -2.335   0.0195 *
## Self_Employed2   0.1554     0.3772   0.412   0.6804
## Property_Area2   0.8430     0.3126   2.697   0.0070 **
## Property_Area3   0.4496     0.3088   1.456   0.1454
## LogLoanAmount   -0.7932     0.3980  -1.993   0.0462 *
## LogIncome        0.4506     0.3503   1.286   0.1983
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 527.96  on 428  degrees of freedom
## Residual deviance: 393.03  on 421  degrees of freedom
```

```
## AIC: 409.03
##
## Number of Fisher Scoring iterations: 5
## Warning: contrasts dropped from factor Education
## Warning: contrasts dropped from factor Self_Employed

## Warning: contrasts dropped from factor Property_Area
##
##     FALSE TRUE
##   N    56   75
##   Y     6  292
##
## Call:
## glm(formula = Loan_Status ~ Credit_History + Education + Self_Employed +
##     Property_Area + LogLoanAmount + LogIncome, family = binomial,
##     data = testnew)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3315  -0.1622   0.4021   0.7359   2.9630
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -4.80883    3.28784  -1.463   0.1436
## Credit_History  5.38627    1.10039   4.895 9.84e-07 ***
```

```
## Education2      0.56115    0.53224   1.054   0.2917
## Self_Employed2 -0.35871    0.60503  -0.593   0.5533
## Property_Area2  1.48356    0.59642   2.487   0.0129 *
## Property_Area3 -0.07527    0.48996  -0.154   0.8779
## LogLoanAmount   0.45786    0.42743   1.071   0.2841
## LogIncome      -0.21141    0.44259  -0.478   0.6329
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 234.58  on 184  degrees of freedom
## Residual deviance: 146.35  on 177  degrees of freedom
## AIC: 162.35
##
## Number of Fisher Scoring iterations: 6
## Warning: contrasts dropped from factor Education
## Warning: contrasts dropped from factor Self_Employed
## Warning: contrasts dropped from factor Property_Area
##
##     FALSE TRUE
##   N    32   29
##   Y     1  123
```

Độ chính xác đối với dữ liệu huấn luyện là 81,11% và độ chính xác đối với dữ liệu kiểm nghiệm là 83,78%.

Chúng ta thấy rằng việc thêm các biến đã cải thiện độ chính xác của mô hình.

VIII, Cây quyết định

1, Khái niệm

Cây quyết định tạo ra một tập hợp các phân tách nhị phân trên các biến dự báo để tạo ra một cây có thể được sử dụng để phân loại các quan sát mới thành một trong hai nhóm.

Thuật toán của mô hình này như sau:

- Chọn biến dự đoán có thể chia dữ liệu thành hai nhóm tốt nhất;
- Tách dữ liệu thành hai nhóm này;
- Lặp lại các bước này cho đến khi một nhóm con chứa ít hơn số lượng quan sát tối thiểu
- Để phân loại một trường hợp, hãy chạy nó xuống cây đến một nút đầu cuối và gán cho nó giá trị kết quả mô hình được chỉ định trong bước trước.

2, Xây dựng mô hình cây quyết định bằng R

>**library(rpart)**

>dtree <- rpart(Loan_Status ~ Credit_History+Education+Self_Employed+Property_Area+LogLoanAmount + LogIncome,method="class", data=trainnew,parms=list(split="information"))

>dtree$cptable

Câu lệnh rpart: rpart là viết tắt của phân vùng đệ quy và sử dụng thuật toán phân loại và cây hồi quy, có thể dùng để "trồng" và "cắt tỉa" cây quyết định. Ngoài thư viện rpart, có rất nhiều thư viện cây quyết định khác như C50, Party, Tree, và mapTree

+ Biến sử dụng để dự đoán: Credit_History + Education + Self_Employed + Property_Area + LogLoanAmount + LogIncome
+ Biến phụ thuộc: Loan_Status
+ Do biến phụ thuộc là số nên ta sử dụng **method="class"**
+ Dữ liệu được sử dụng dựa trên tập huấn luyện (data=trainnew)

+ Tham số tùy chọn cho chức năng phân tách là “information” (parms=list(split="information"))

+ dtree$cptable : lấy bảng xác xuất có diều kiện của của biến **dtree**

```
##          CP nsplit rel error    xerror       xstd
## 1 0.38167939      0 1.0000000 1.0000000 0.07281885
## 2 0.02290076      1 0.6183206 0.6183206 0.06187742
## 3 0.01908397      2 0.5954198 0.7099237 0.06514951
## 4 0.01145038      5 0.5343511 0.7099237 0.06514951
## 5 0.01000000      7 0.5114504 0.7328244 0.06589581
```

+ xerror : lỗi có xác nhận chéo ít nhất

+ nsplit: số lần tách nhánh

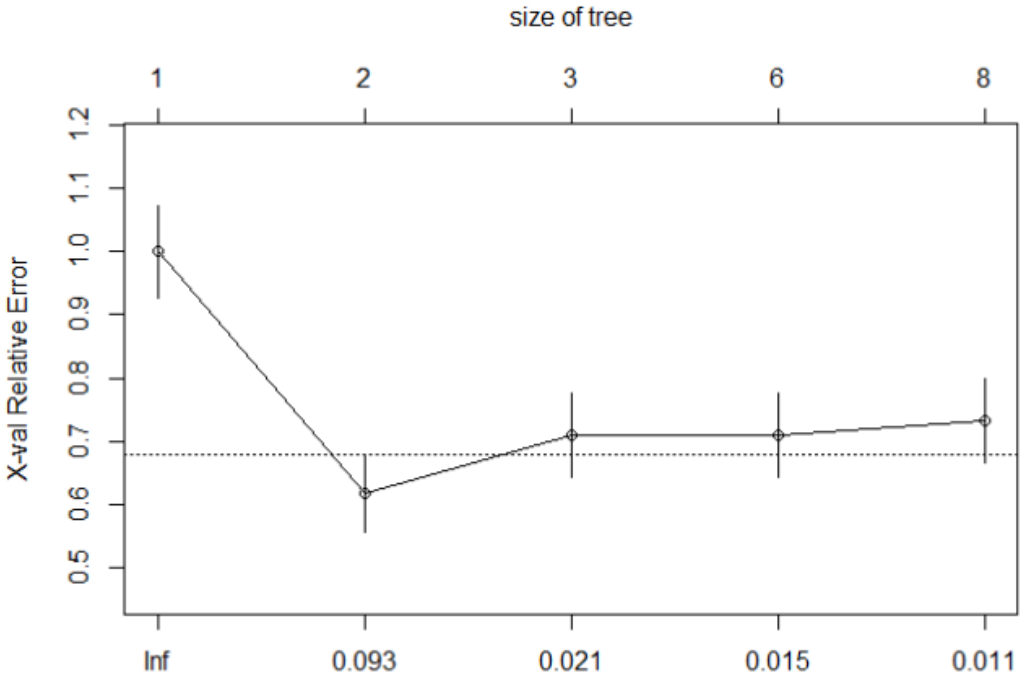
+ cp : tham số phức tạp

+ xstd : chứa lỗi chuẩn của lỗi xác thực chéo

+ rel xerror : chứa tỷ lệ lỗi cho một cây có kích thước nhất định trong mẫu huấn luyện

> **plotcp(dtree)**

+ hàm **plotcp(dtree)** : vẽ biểu đồ lỗi được xác thực chéo so với tham số độ phức tạp (cp).



> dtree.pruned <- prune(dtree, cp=.02290076)

> library(rpart.plot)

> prp(dtree.pruned, type = 2, extra = 104,

fallen.leaves = TRUE, main="Decision Tree")

+ Hàm **prune()** : "cắt tỉa" cây dựa trên tham số độ phức tạp "cp = . 02290076"

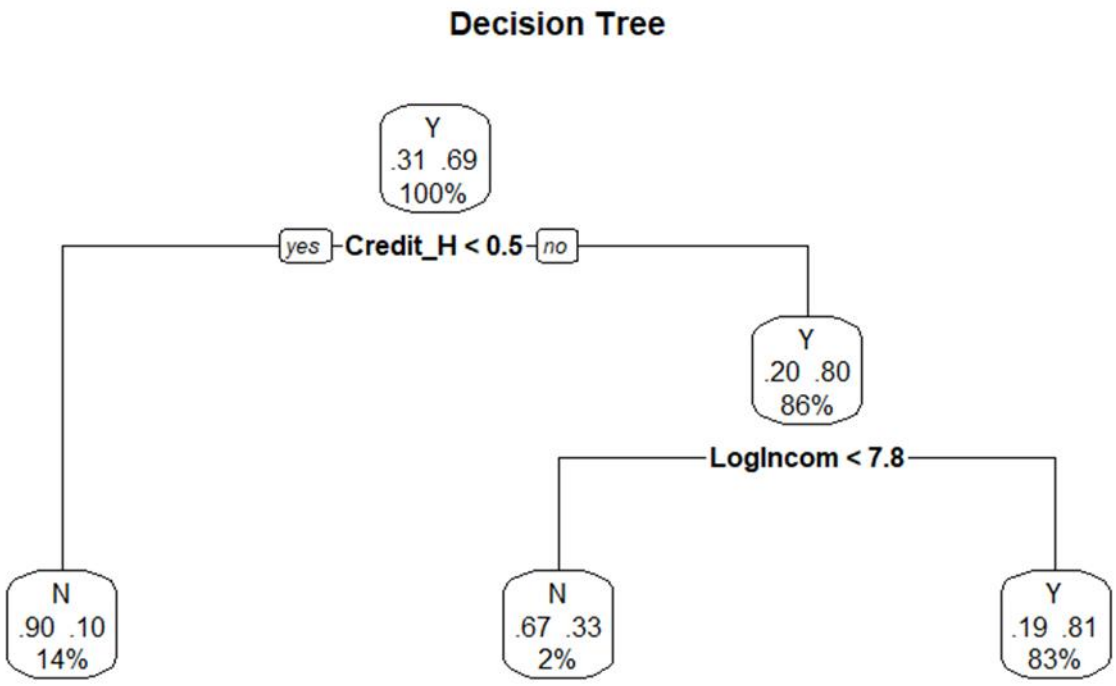
+ library(rpart.plot) : lấy câu lệnh rpart.plot ra khỏi thư viện

+ Hàm PRP : vẽ mô hình của cây

+ "type = 2" : vẽ nhánh

+ "extra = 104" : mô hình lớp với phản hồi nhị phân có nhiều hơn hai cấp độ

+ “fallen.leaves = TRUE” : đặt các “nút lá” ở cuối biểu đồ



> dtree_test.pred <- predict(dtree_test.pruned, testnew, type="class")

> dtree_test.perf <- table(testnew$Loan_Status, dtree_test.pred,

> dnn=c("Actual", "Predicted"))

> dtree_test.perf

+ hàm predict : dự đoán thử nhiệm bằng cách sử dụng cây quyết định

+ dtree_test.perf <- table(testnew$Loan_Status, dtree_test.pred,

dnn=c("Actual", "Predicted”)) : Tạo bảng thống kê độ chính xác của mô hình

Predicted

Actual N Y

N 62 69

Y 9 289

Như vậy độ chính xác của mô hình trên tập huấn luyện là 81,81%. Thực hiện tương tự các câu lệnh trên tập huấn luyện, ta được độ chính xác của mô hình là 85,4%. Độ chính xác của mô hình cây quyết định tốt hơn mô hình hồi quy Logistic.

IX, Rừng ngẫu nhiên

1. Khái niệm

Rừng ngẫu nhiên là một thuật toán học có giám sát. Như tên gọi của nó, Rừng ngẫu nhiên sử dụng các cây (tree) để làm nền tảng.

Rừng ngẫu nhiên là một tập hợp của các Decision Tree, mà mỗi cây được chọn theo một thuật toán dựa vào ngẫu nhiên.. Cách tiếp cận này phát triển nhiều mô hình dự đoán và kết quả được tổng hợp để cải thiện việc phân loại. Thuật toán như sau:

- Trồng nhiều cây quyết định bằng cách lấy mẫu.
- Trong các cây tồn tại 1 số m << M biến đầu và sao cho tại mỗi node (nốt) ta random chọn 1 biến từ m biến đầu vào đó. Và mỗi node sẽ tách ra m nhánh dẫn tới các nhánh tiếp theo, biết rằng m = const trong rừng ngẫu nhiên.
- Các cây được tạo ra có độ lớn là lớn nhất , không bị cắt tỉa (no pruning).
- Các node đầu cuối được gán cho một lớp dựa trên chế độ của các trường hợp trong nút đó.
- Phân loại các trường hợp mới bằng cách gửi các biến dữ liệu của trường hợp mới tới các cây, mỗi cây sẽ trả lại kết quả là class tương ứng. Rừng ngẫu nhiên sẽ phân loại trường hợp mới đó theo class mà được nhiều cây trả về nhất.

2. Xây dựng mô hình rừng ngẫu nhiên bằng R

Rừng ngẫu nhiên được trồng bằng cách sử dụng hàm randomForest () trong Gói randomForest trong R. Số lượng cây mặc định là 500, số lượng biến mặc định được lấy mẫu tại mỗi nút là sqrt (M) và kích thước nút tối thiểu là 1.

```
> library(randomForest)
> set.seed(42)
```

> fit.forest <- randomForest(Loan_Status ~ Credit_History+Education+Self_Employed+Property_Area+LogLoanAmount + LogIncome, data=trainnew,

na.action=na.roughfix,

importance=TRUE)

> fit.forest

Câu lệnh được sử dụng:

set.seed(): dùng để tái tạo lại một output. Tức là ta khi lấy mẫu nhiều lần thì đầu ra là giống nhau.

Hàm **randomForest()** để trồng các cây quyết định.

+ Mặc định của nó 1 lần 500 cây (ở trong bài này là trồng 500 cây từ 429 mẫu)

+ Số biến mặc định để lấy mẫu ở mỗi node là sqrt(M) với M là tổng số biến đầu vào

+ min node size (min kích cỡ của lá) = 1

=> Formula: công thức mô tả mô hình điều chỉnh.

+ **Data** : một khung dữ liệu tuy chọn chứa các biến trong mô hình. Theo mặc định sẽ chứa các biến từ trong môi trường mà RF được gọi.

+ **na.action=na.roughfix** => Sử dụng trung vị hoặc mốt để thay thế cho dữ liệu bi thiếu. Sau khi phát triển thành một khu rừng và tính toán các vùng lân cận thì ta lại lặp lại quá trình và xây dựng một khu rừng mới từ các giá trị mới điền này.

+ **importance=TRUE** => Đánh giá tầm quan trọng của các yếu tố dự báo.

##

Call:

randomForest(formula = Loan_Status ~ Credit_History + Education + Self_Employed + Property_Area + LogLoanAmount + LogIncome, data = trainnew, importance = TRUE, na.action = na.roughfix)

```
##               Type of random forest: classification
##                     Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 20.05%
## Confusion matrix:
##       N     Y     class.error
## N  59    72    0.54961832
## Y  14  284    0.04697987
```

=> Độ chính xác của training sample 59+284/429 = 0.799533 < độ chính xác của của cây quyết định ở phần trên

```
> importance(fit.forest, type=2)
```

=> Đây là hàm trích xuất cho các thước đo tầm quan trọng được tính bởi rừng ngẫu nhiên.

=> type = 2 là độ giảm tập chất trung bình của nút do phân tách trên biến đó.

+ OOB estimate of error rate: 20.05%

=> Xác suất xảy ra lỗi OOB trong rừng ngẫu nhiên là 20.05%. Tỉ lệ này càng thấp thì dự đoán của chúng ta càng chính xác.

Out-Of-Bag (OOB): khi ta lấy mẫu dữ liệu từ dữ liệu gốc bằng kỹ thuật Bootstrapping hay còn gọi là random sampling with replacement. Tức khi mình sample được 1 dữ liệu thì mình không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kĩ thuật này thì tập n dữ liệu mới của mình có thể có những dữ liệu bị trùng nhau và bộ dữ liệu mới có thể không chứa toàn bộ dữ liệu gốc, nhưng dữ liệu bị thừa ra như thế đc cho là OOB.

=> Tỉ lệ lỗi Out-Of-Bag: Lỗi OOB xảy ra khi ta thử một dữ liệu OOB vào rừng ngẫu nhiên thì cho ra kết quả phân loại sai so với thực tế.

```
##                   MeanDecreaseGini
## Credit_History          47.035374
## Education               4.796234
## Self_Employed           2.634523
## Property_Area           6.832392
## LogLoanAmount           28.572245
## LogIncome               34.357392
```

=> Biến quan trọng nhất là Credit_History và ít quan trọng nhất là Self_Employed

```
> forest.pred <- predict(fit.forest, testnew)
> forest.perf <- table(testnew$Loan_Status, forest.pred,
                       dnn=c("Actual", "Predicted"))
> forest.perf
```

=> Dự đoán dữ liệu thử nghiệm bằng cách sử dụng rừng ngẫu nhiên và tạo bảng thông kê xem độ chính xác của mô hình

```
##          Predicted
## Actual     N   Y
##        N   32  29
##        Y    4 120
```

Đây là độ chính xác của mô hình:

- Dữ liệu huấn luyện: 79,95%
- Dữ liệu thử nghiệm: 82,16%

Chúng ta sẽ chạy cùng một mô hình nhưng lần này chúng ta sẽ chọn ba mô hình có tầm quan trọng cao nhất:

> set.seed(42)

> fit.forest2 <- randomForest(Loan_Status ~
Credit_History+LogLoanAmount+
LogIncome, data=trainnew,importance=TRUE)

> fit.forest2

##

Call:

randomForest(formula = Loan_Status ~ Credit_History + LogLoanAmount + LogIncome, data = trainnew, importance = TRUE)

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 1

##

OOB estimate of error rate: 19.11%

Confusion matrix:

N Y class.error

N 58 73 0.55725191

Y 9 289 0.03020134

> forest.pred2 <- predict(fit.forest2, testnew)

> forest.perf_test <- table(testnew$Loan_Status, forest.pred2,
dnn=c("Actual", "Predicted"))

> forest.perf_test

Predicted

Actual N Y

N 31 30

Y 1 123

Ở đây, chúng ta nhận thấy những cải tiến nhỏ trên cả hai mẫu trong đó độ chính xác của mẫu huấn luyện là 80,88% và độ chính xác của mẫu thử là 83,24%. Độ chính xác cho cây quyết định vẫn tốt hơn.

X, Lựa chọn mô hình

Mặc dù rừng ngẫu nhiên có độ chính xác thấp hơn một chút so với mô hình cây quyết định, rừng ngẫu nhiên có xu hướng rất chính xác so với các phương pháp phân loại khác. Cây quyết định có thể bị hiện tượng quá khớp, trong khi rừng ngẫu nhiên ngăn chặn việc này bằng cách tạo các tập con ngẫu nhiên của các biến và xây dựng các cây nhỏ hơn bằng cách sử dụng các tập con và sau đó nó kết hợp các cây con để thu được mô hình sau cùng. Do vậy, chúng ta sẽ lựa chọn mô hình rừng ngẫu nhiên.

Bây giờ chúng ta hãy tạo một bảng dữ liệu với hai cột: Loan_ID và Loan_Status chứa các dự đoán của chúng ta:

> my_solution <- data.frame(Loan_ID = testnew$Loan_ID, Loan_Status = forest.pred2)

Ghi kết quả dự đoán vào file csv **my_solution.csv.**

> write.csv(my_solution, file = "my_solution.csv", row.names = FALSE)

Như vậy hiện tại chúng ta có dự đoán cho 185 khách hàng đăng ký vay với độ chính xác là 83,24%. Chúng ta có thể áp dụng phương pháp này cho bất kỳ tập dữ liệu mới nào có cùng các biến số để đưa ra dự đoán về khả năng đủ điều kiện nhận khoản vay của họ.

XI, Danh mục tài liệu tham khảo

[1] Nguyễn Văn Tuấn, Garvan Institute of Medical Research, Sydney, Australia: Phân tích số liệu và biểu đồ bằng R.

[2] https://github.com/DataBoosting/Loan-Prediction-with-R/blob/master/Loan.md

XII, Phụ lục

Một số của bài toán cơ bản trên tập dữ liệu

1, Đọc dữ liệu và loại bỏ dữ liệu trống

> data = read.csv("C:\\Users\\USER\\Desktop\\Coaching\\Projects\\Loan Prediction\\train.csv", na.strings=c("", "NA"), header=TRUE)

> data = na.omit(data)

Xem bảng dữ liệu:

>View(data)

2, Kiểm tra thông tin về dữ liệu

2.1. Có bao nhiêu dòng và cột số liệu:

> dim(data)

[1] 480 13

2.2. Tên của các biến:

>names(data)

[1] "Loan_ID" "Gender" "Married" "Dependents" "Education"

[6] "Self_Employed" "ApplicantIncome" "CoapplicantIncome" "LoanAmount" "Loan_Amount_Term"

[11] "Credit_History" "Property_Area" "Loan_Status"

2.3, Trong biến Gender có bao nhiêu nam và bao nhiêu nữ:

>table(data$Gender)

Female Male

86 394

3. Tách và chiết xuất dữ liệu

3.1. Tách rời dữ liệu:

Tách ra một bộ dữ liệu có Gender là Male và Female và đặt tên cho chúng tương ứng là Male và Female

> Male = subset(data, data$Gender == “Male”)

> Female = subset(data, data$Gender == “Female”)

3.2. Chiết xuất số liệu

Sinh ngẫu nhiên 70% giá trị từ 1 đến số dòng của dữ liệu sau đó chiết 1 bộ dữ liệu gồm các dòng đó và bộ dữ liệu thứ 2 là các dòng còn lại

sample <- sample.int(n = nrow(data), size = floor(.70*nrow(data)), replace = F)

data1 = data[sample,]

data2 = data[-sample,]

4. Biến đổi số liệu

Biến đổi dữ liệu dạng Y/N trong biến **LoanStatus** thành dữ liệu dạng 1/0

binary = tr$Loan_Status

binary = replace(binary, tr$Loan_Status=="Y", 1)

binary = replace(binary, tr$Loan_Status=="N", 0)

tr$Loan_Status = binary

5. Các bài toán với **if**, **for** và **function**

5.1. So sánh ApplicantIncome trung bình của bộ dữ liệu Male và Female và in ra thông báo tương ứng

> mean1 = mean(Male$Income)

> mean2 = mean(Female$Income)

```
> if(mean1>mean2) {
+       print(“Thu nhap trung binh cua khach hang nam nhieu hon “)
+ } else {
+       print(“Thu nhap trung binh cua khach hang nu nhieu hon”)
+ }
[1] "Thu nhap trung binh cua khach hang nam nhieu hon"
```

5.2. Đếm xem có bao nhiêu khách hàng nam có số người phụ thuộc là 2 sử dụng **for**

```
> count = 0
> for( i in 1:nrow(Male)) {
+       if(Male$Dependents[i]==2) {
+                 count = count+1;
+       }
+ }
> count
[1] 81
```

5.3. Viết một hàm nhận đối là Male hoặc Female, đưa ra ApplicantIncome và LoanAmount cao nhất tương ứng của bộ dữ liệu đó.

```
> func = function (x="") {
+   subdata = subset(data, data$Gender == x)
+   print(max(subdata$ApplicantIncome))
+   print(max(subdata$LoanAmount))
+ }
> func("Male")
[1] 81000
[1] 600
```

6. Vẽ và nhận xét một số biểu đồ cho dữ liệu

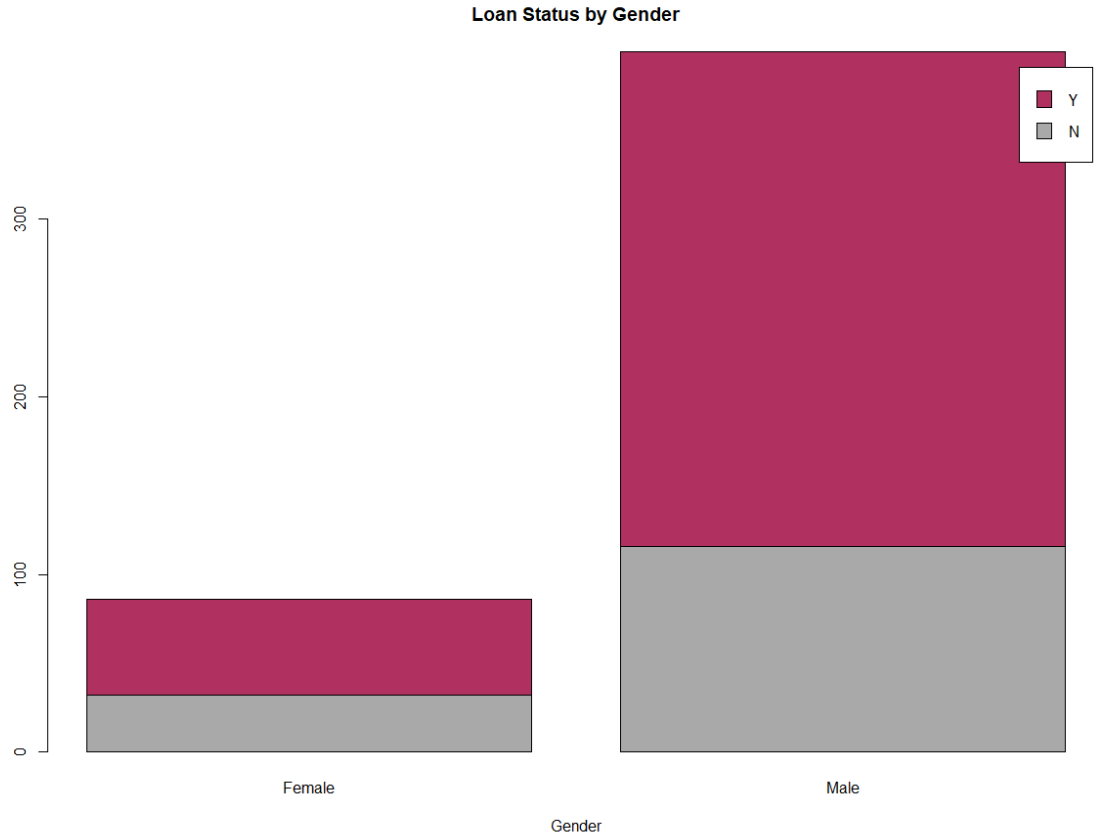
6.1. Vẽ biểu đồ cột giữa 2 biến Gender và LoanStatus

> counts <- table(data$Loan_Status, data$Gender)

> barplot(counts, main="Loan Status by Gender",

+ xlab="Gender", col=c("darkgrey","maroon"),

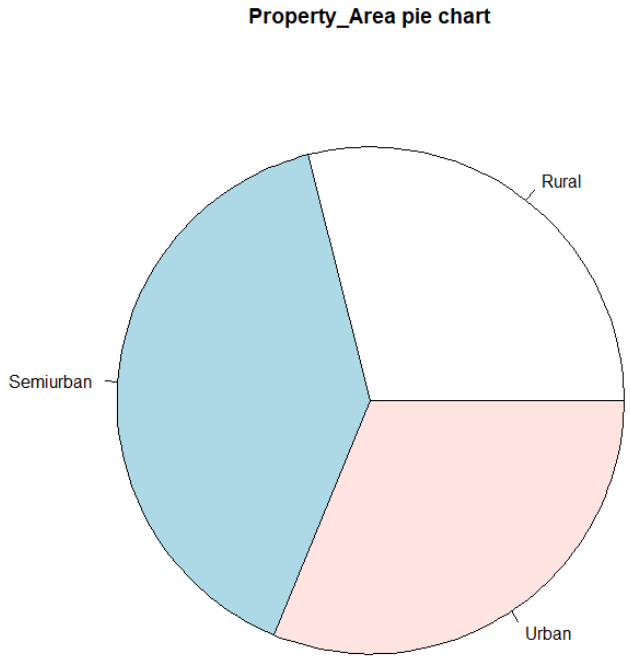
+ legend = rownames(counts))



Nhận xét: Số đơn đăng ký của khách hàng nam nhiều hơn khách hàng nữ, cả nam và nữ đều có trên 50% số đơn được chấp thuận

6.2. Vẽ biểu đồ tròn cho Property_Area

> pie(table(data$Property_Area))

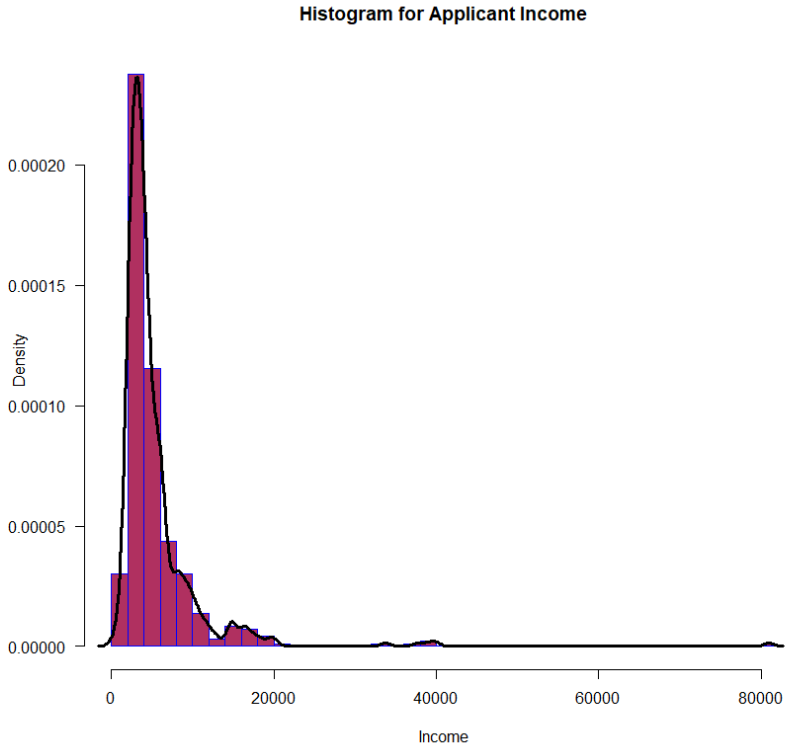


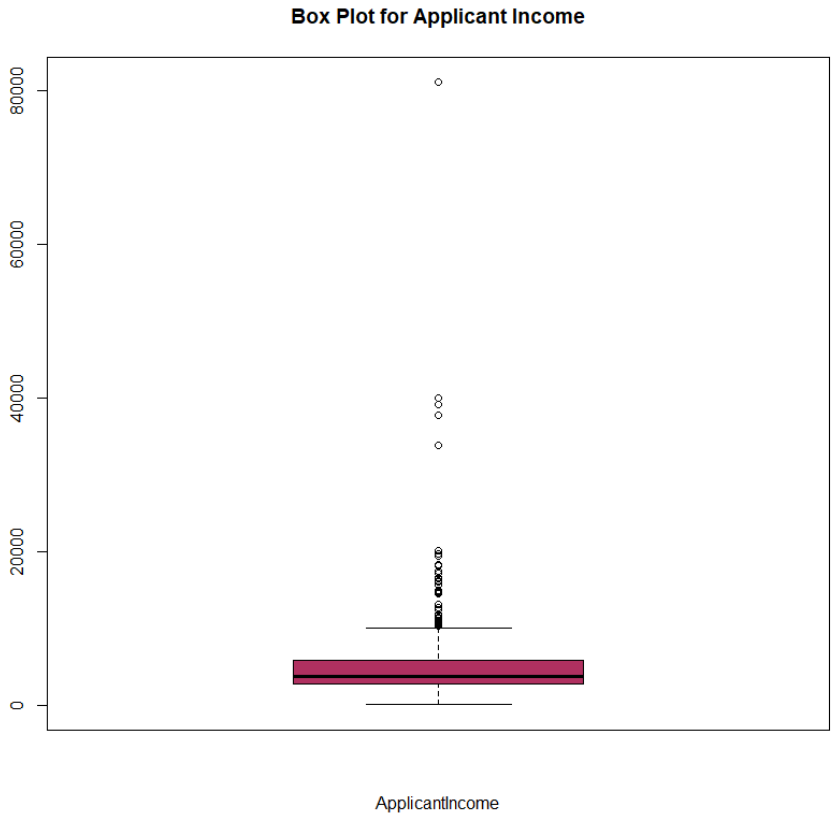
Nhận xét: Trong số các đơn đăng ký, số đơn của khách hàng đến từ bán thành thị chiếm nhiều nhất và chiếm khoảng 1/3 tổng số đơn. Số đơn của khách hàng đến từ thành thị và nông thôn gần bằng nhau và chiếm 1/6 số đơn đăng ký.

6.3. Biểu đồ cho biến ApplicantIncome

```
> hist(tr$ApplicantIncome,
+     main="Histogram for Applicant Income",
+     xlab="Income",
+     border="blue",
+     col="maroon",
+     las=1,
+     breaks=50, prob = TRUE)
>lines(density(tr$ApplicantIncome), col='black', lwd=3)
```

>boxplot(tr$ApplicantIncome, col='maroon',xlab = 'ApplicantIncome', main = 'Box Plot for Applicant Income')





Nhận xét: Thu nhập của khách hàng tập trung trong khoảng từ 0 đến 20000. Có một vài giá trị đột xuất trong thu nhập của khách hàng khiến phân bố bị lệch phải.

7. Tóm tắt và mô tả dữ liệu định lượng: Thực hiện với biến **ApplicantIncome**

> Income = data$ApplicantIncome

7.1. Các mức độ điển hình

Trung bình: Trung bình cộng của các giá trị dữ liệu, chịu ảnh hưởng bởi giá trị đột xuất

> mean(Income)

[1] 5364.231

Trung vị: Trong dãy số đã được sắp xếp, trung vị là số chính giữa, không chịu ảnh hưởng của giá trị đột xuất

> median(Income)

[1] 3859

Midrange: Bình quân của giá trị nhỏ nhất và giá trị lớn nhất, chịu ảnh hưởng của giá trị đột xuất.

> (max(Income)+min(Income))/2

[1] 40575

7.2. Tứ phân vị

Tứ phân vị thứ nhất: Số Q1 là tứ phân vị thứ nhất cho biết có 25% giá trị dữ liệu nhỏ hơn Q1.

> Q1=quantile(Income,0.25)

> Q1

25%

2898.75

Tứ phân vị thứ ba: Số Q3 là tứ phân vị thứ ba cho biết có 75% giá trị dữ liệu nhỏ hơn Q3.

> Q3=quantile(Income,0.75)

> Q3

75%

5852.5

Tứ phân vị: Là chênh lệch giữa Q3 và Q, cho biết độ biến thiên của 50% số đơn vị ở giữa, không chịu ảnh hưởng bởi giá trị đột xuất

> IQR = Q3-Q1

Midhinge = (Q1+Q3)/2

7.3. Độ biến thiên

Khoảng biến thiên: Là sự chênh lệch giữa giá trị lớn nhất và giá trị nhỏ nhất, không phụ thuộc vào sự phân bố của dữ liệu.

>Range = max(Income)-min(Income)

Phương sai: Cho biết độ biến thiên xung quanh giá trị trung bình.

> DX = (sum((Income-mean(Income))^2))/(length(Income)-1)

Độ lệch tiêu chuẩn: Cho biết độ biến thiên xung quanh giá trị trung bình

> sigma = sqrt(DX)

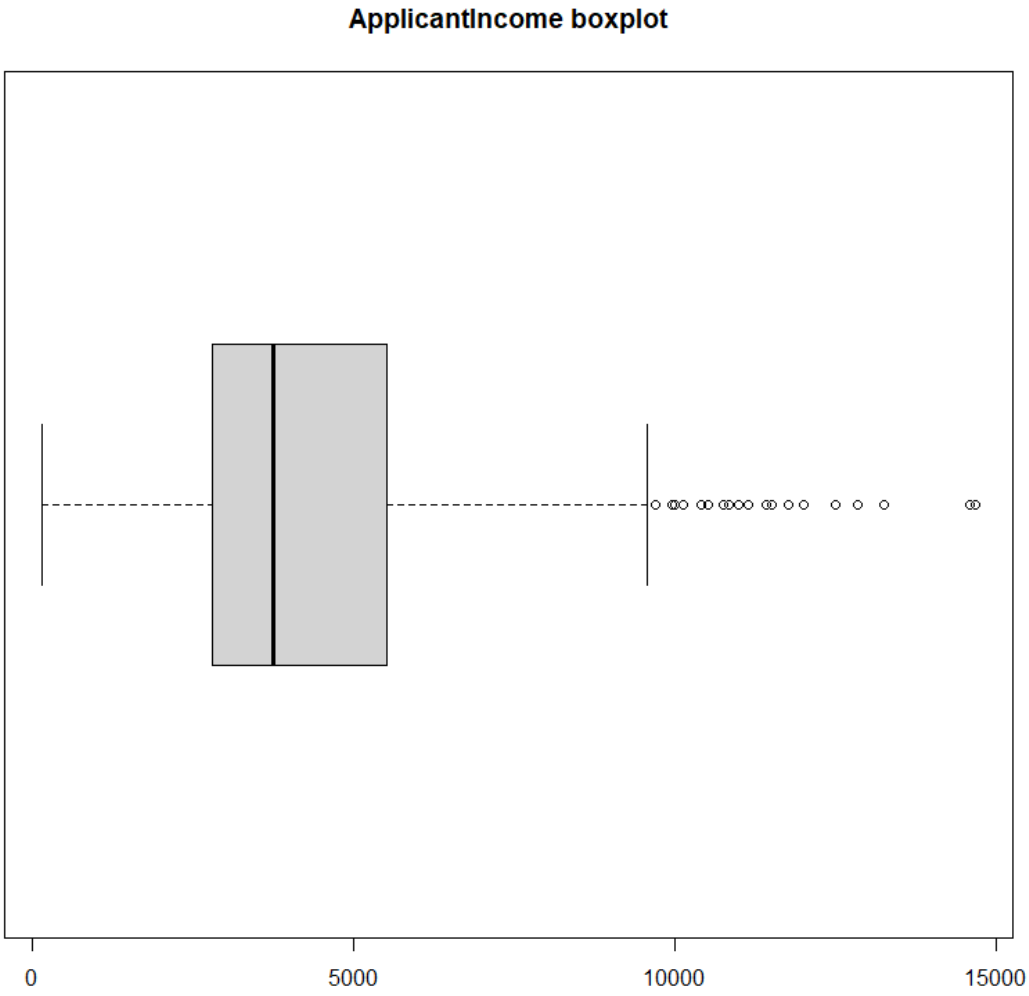
Hệ số biến thiên: Cho biết độ biến thiên tương đối xung quanh giá trị trung bình

> CV = (sigma/mean(Income))*100%

8. Quy tắc 3Q loại bỏ dữ liệu đột xuất và vẽ đồ thị hộp Boxplot

> data1 =subset(data,Q1-3*IQR<=Income & Income<=Q3+3*IQR)

> boxplot(data1$ApplicantIncome, main="ApplicantIncome boxplot",horizontal=TRUE)



Nhận xét: Giá trị nhỏ nhất của thu nhập khách hàng là 150$ và lớn nhất là gần 15000$. Có 75% khách hàng có thu nhập thấp hơn 5512$, có 50% khách hàng có thu nhập thấp hơn 3750$, 25% khách hàng có thu nhập thấp hơn 2808$.