Universitat Oberta de Catalunya

Estudios de Informática, Multimedia y Telecomunicaciones

Minería de datos: PEC3 - Clasificación con árboles de decisión

Autor: Gloria Manresa

Mayo 2023


```
# Importar librerías
library(ggplot2)
library(grid)
library(gridExtra)
library(arules)
library(dplyr)
library(DescTools)
library(C50)
library(randomForest)
library(iml)
```

1 Carga de los datos e introducción

El conjunto de datos que se utilizará proviene de la página web de “UCI Machine Learning Repository”. Recopilado originalmente por el Profesor Dr. Hans Hofmann.

Este conjunto de datos informa si un solicitante de crédito es “bueno” o “malo” en función de varias características. Contiene información sobre 1.000 solicitantes de crédito y un total de 20 atributos.

La variable objetivo indica si el solicitante de crédito es considerado un buen riesgo crediticio (etiqueta “1”) o un mal riesgo crediticio (etiqueta “2”).

```
data<-read.csv("./credit.csv",header=T,sep=",")
attach(data)
```

2 Análisis inicial

Empezaremos realizando un análisis exploratorio de los datos para conocer más sobre el conjunto de datos con el que trabajaremos a continuación.

2.1 Exploración de la base de datos

En primer lugar exploramos la dimensión del conjunto de datos, es decir el número de registros y el número de atributos. También exploraremos el tipo de atributo y descripción de cada uno de ellos.

Observamos a continuación que tenemos 1000 registros y 21 variables.

```
dim(data)
```

```
## [1] 1000   21
```

A continuación, con la función str() conocemos el tipo de variable y observamos las primeros registros de cada una de ellas.

```
str(data)
```

```
## 'data.frame':    1000 obs. of  21 variables:
##  $ checking_balance    : chr  "< 0 DM" "1 - 200 DM" "unknown" "< 0 DM" ...
##  $ months_loan_duration: int  6 48 12 42 24 36 24 36 12 30 ...
##  $ credit_history      : chr  "critical" "repaid" "critical" "repaid" ...
##  $ purpose             : chr  "radio/tv" "radio/tv" "education" "furniture" ...
##  $ amount              : int  1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...
##  $ savings_balance     : chr  "unknown" "< 100 DM" "< 100 DM" "< 100 DM" ...
##  $ employment_length   : chr  "> 7 yrs" "1 - 4 yrs" "4 - 7 yrs" "4 - 7 yrs" ...
##  $ installment_rate    : int  4 2 2 2 3 2 3 2 2 4 ...
##  $ personal_status     : chr  "single male" "female" "single male" "single male" ...
##  $ other_debtors       : chr  "none" "none" "none" "guarantor" ...
##  $ residence_history   : int  4 2 3 4 4 4 4 2 4 2 ...
##  $ property            : chr  "real estate" "real estate" "real estate" "building soc
iety savings" ...
##  $ age                 : int  67 22 49 45 53 35 53 35 61 28 ...
##  $ installment_plan    : chr  "none" "none" "none" "none" ...
##  $ housing             : chr  "own" "own" "own" "for free" ...
##  $ existing_credits    : int  2 1 1 1 2 1 1 1 1 2 ...
##  $ default             : int  1 2 1 1 2 1 1 1 1 2 ...
##  $ dependents          : int  1 1 2 2 2 2 1 1 1 1 ...
##  $ telephone           : chr  "yes" "none" "none" "none" ...
##  $ foreign_worker      : chr  "yes" "yes" "yes" "yes" ...
##  $ job                 : chr  "skilled employee" "skilled employee" "unskilled reside
nt" "skilled employee" ...
```

Una pequeña descripción de cada uno de los atributos:

- checking_balance: Estado de la cuenta corriente existente
- months_loan_duration: Duración del préstamo (en meses)
- credit_history: Historial de crédito
- purpose: Propósito del crédito
- amount: Cantidad del crédito
- savings_balance: Cuenta de ahorros
- employment_length: Duración del empleo actual
- installment_rate: Tasa de cuota en porcentaje del ingreso disponible
- personal_status: Estado civil y sexo
- other_debtors: Otros deudores / aval
- residence_history: Duración residencia actual
- property: Propiedad
- age: Edad
- installment_plan: Otros planes de cuotas
- housing: Vivienda (alquilada, propia, gratis)
- existing_credits: Número de créditos existentes en este banco
- dependents: Número de personas a su cargo
- telephone: Presencia de teléfono (sí o no)
- foreign_worker: Trabajador extranjero (sí o no)
- job: Trabajo (desempleado, no cualificado, cualificado, autónomo)

La variable objetivo es *"default"*. Este conjunto de datos clasifica a las personas descritas por un conjunto de atributos como buenos o malos riesgos crediticios. La variable "default" toma el valor 1 para "bueno" o 2 para "malo". Es decir, si una persona presenta "default" = 1 es probable que le acepten el crédito por presentar menos

riesgo.

Vemos que las variables categóricas están definidas como “character”, así que las transformamos a tipo factor.

```
data$credit_history <- factor(data$credit_history)
data$checking_balance <- factor(data$checking_balance)
data$purpose <- factor(data$purpose)
data$savings_balance <- factor(data$savings_balance,
                               levels = c("< 100 DM","101 - 500 DM","501 - 1000 DM", ">
1000 DM","unknown"),
                               labels = c("<100","101-500","501-1000",">1000","unknow
n"))
data$employment_length <- factor(data$employment_length,
                               levels = c("unemployed","0 - 1 yrs","1 - 4 yrs","4 - 7 yr
s", "> 7 yrs"),
                               labels = c("unemployed","0-1yrs","1-4yrs","4-7yrs", ">7yr
s"))
data$personal_status <- factor(data$personal_status)
data$other_debtors <- factor(data$other_debtors,
                               levels = c("co-applicant","none","guarantor"))
data$property <- factor(data$property)
data$installment_plan <- factor(data$installment_plan)
data$housing <- factor(data$housing)
data$telephone <- factor(data$telephone)
data$foreign_worker <- factor(data$foreign_worker)
data$job <- factor(data$job)
```

También hay algunas variables definidas como “integer” que o bien son categóricas o bien se pueden tratar como tal por tener pocos valores únicos.

```
data$installment_rate <- factor(data$installment_rate)
data$residence_history <- factor(data$residence_history)
data$existing_credits <- factor(data$existing_credits)
data$default <- as.factor(ifelse(data$default > 1, "Malo", "Bueno"))
data$dependents <- factor(data$dependents)
```

Todavía tenemos tres variables continuas tipo intenger. A continuación vamos a discretizar dichas variables.

2.2 Discretizar variables continuas

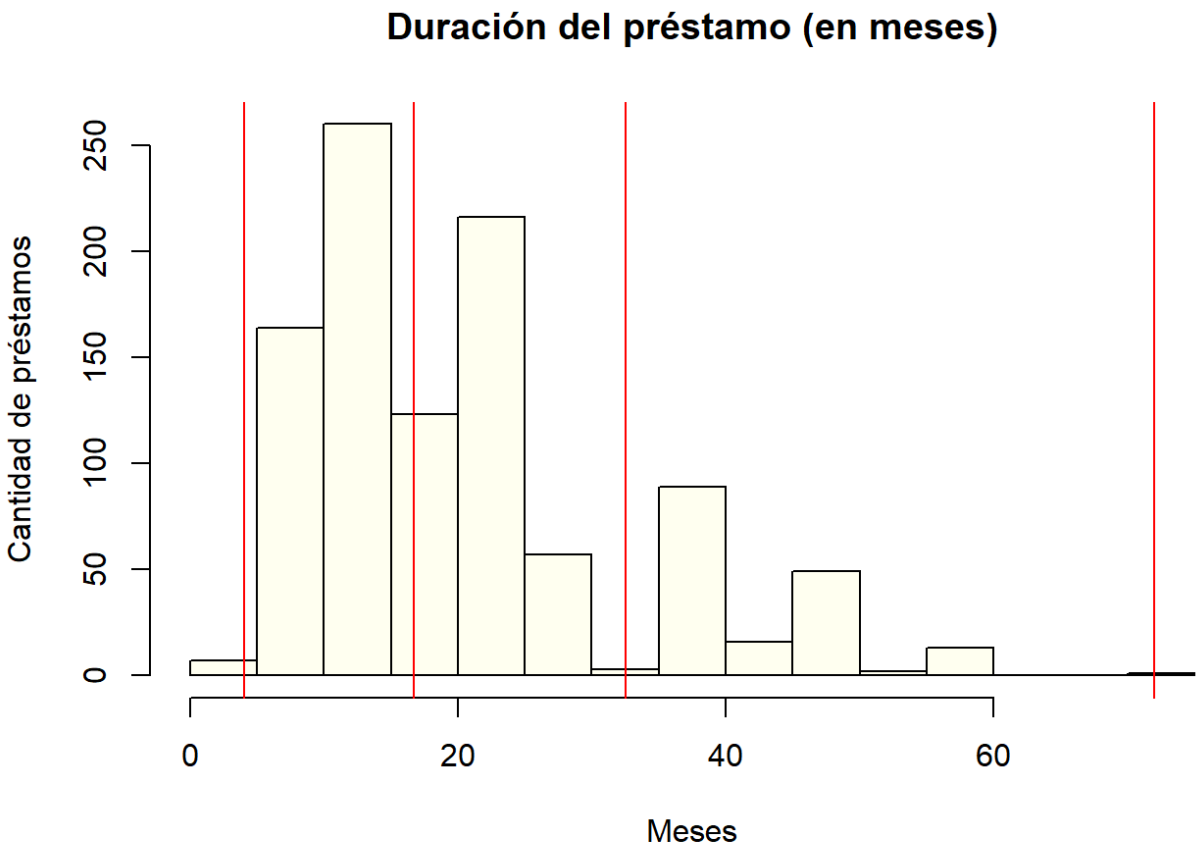
A continuación vamos a discretizar la variable “months_loan_duration” para obtener intervalos. Dejamos que el algoritmo elija el conjunto de particiones.

```
set.seed(111)
table(discretize(data$months_loan_duration, "cluster" ))
```

```
##
##    [4,16.7) [16.7,32.5)   [32.5,72]
##         433         394         173
```

Observamos que el algoritmo ha decidido 3 clústeres. A continuación los visualizamos.

```
set.seed(111)
hist(data$months_loan_duration, main="Duración del préstamo (en meses)",xlab="Meses", yl
ab="Cantidad de préstamos",col = "ivory")
abline(v=discretize(data$months_loan_duration, method="cluster", onlycuts=TRUE),col="re
d")
```



Guardamos los intervalos en una nueva variable.

```
set.seed(111)
data$months_loan_duration_disc <- discretize(data$months_loan_duration, "cluster" )
```

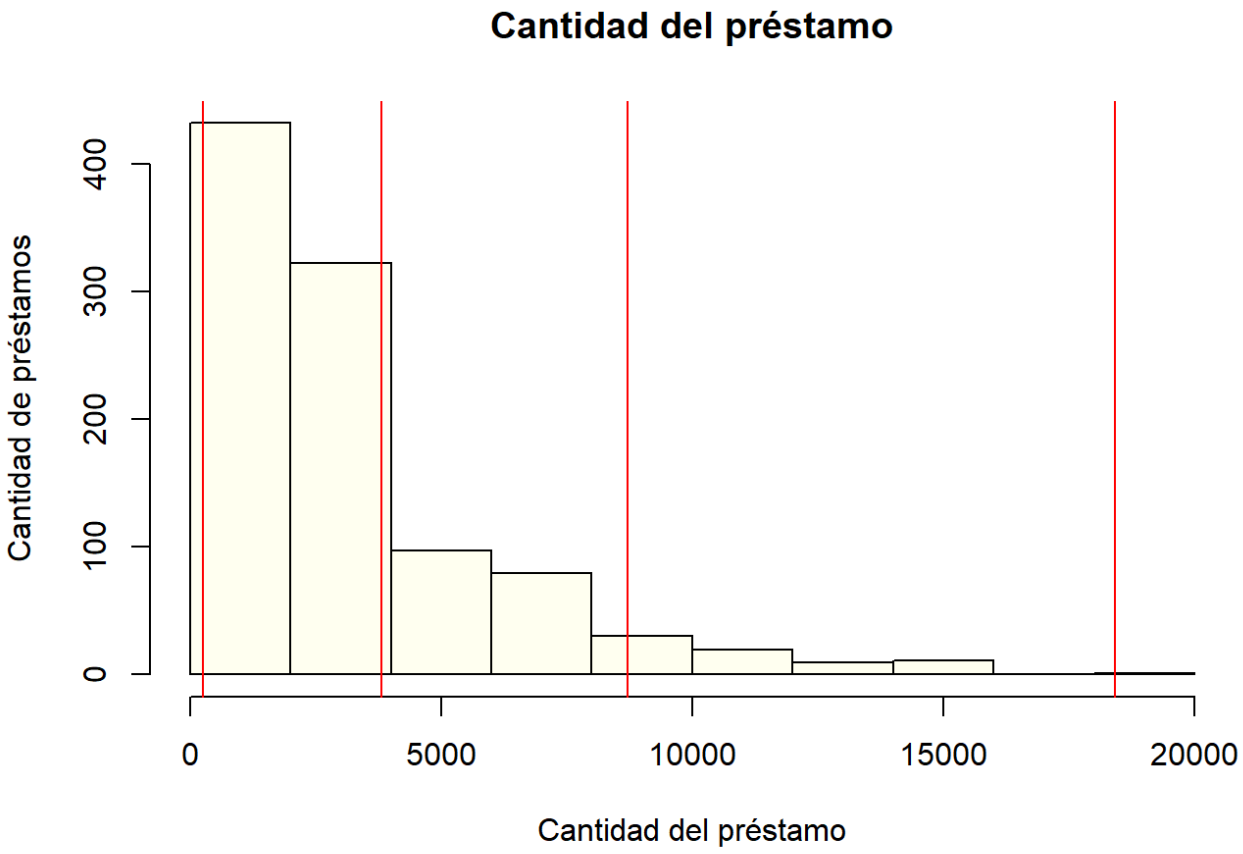
También vamos a discretizar la variable “amount” para obtener intervalos. Dejamos que el algoritmo elija el conjunto de particiones.

```
set.seed(111)
table(discretize(data$amount, "cluster" ))
```

```
##
##      [250,3.82e+03) [3.82e+03,8.72e+03) [8.72e+03,1.84e+04]
##                 728                  216                   56
```

Observamos que el algoritmo ha decidido 3 clústeres. A continuación los visualizamos.

```
set.seed(111)
hist(data$amount, main="Cantidad del préstamo",xlab="Cantidad del préstamo", ylab="Canti
dad de préstamos",col = "ivory")
abline(v=discretize(data$amount, method="cluster", onlycuts=TRUE),col="red")
```



Guardamos los intervalos en una nueva variable.

```
set.seed(111)
data$amount_disc <- discretize(data$amount, "cluster" )
```

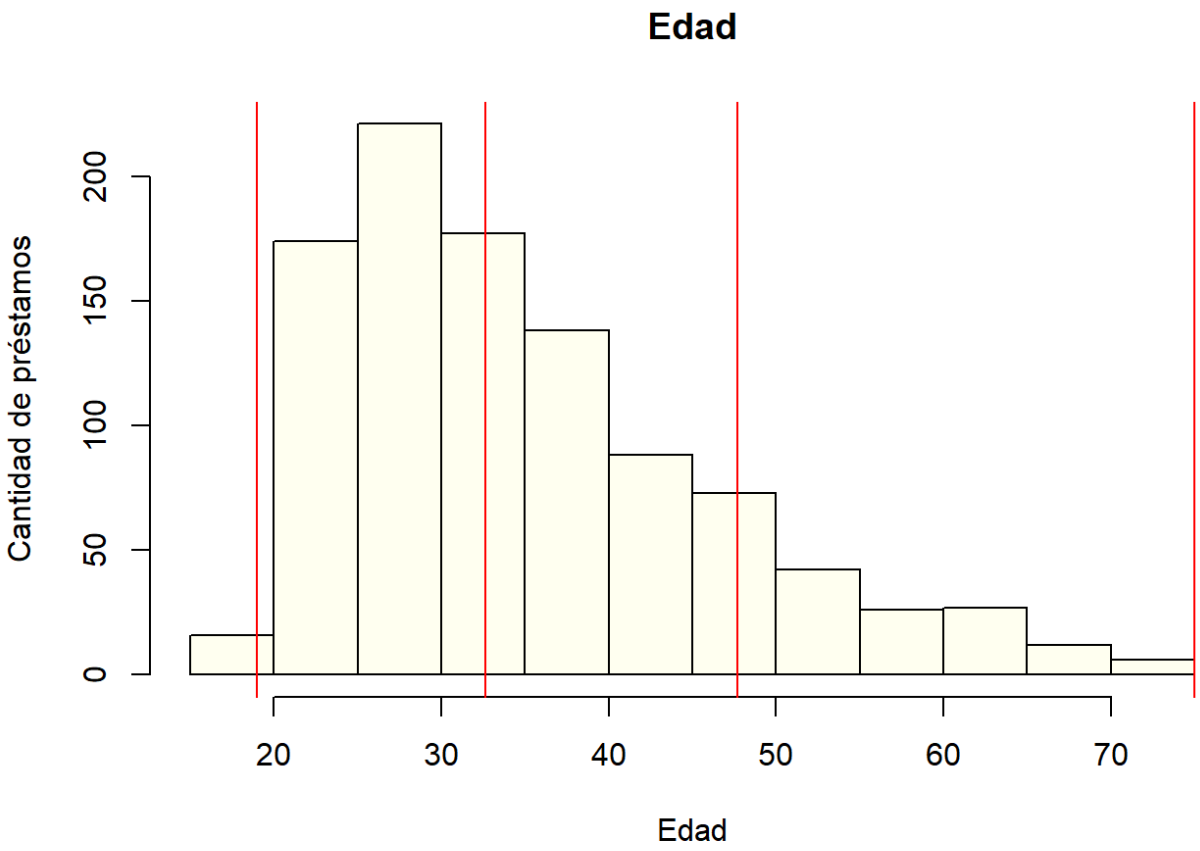
Por último, discretizamos la variable “age” para obtener intervalos. Dejamos que el algoritmo elija el conjunto de particiones.

```
set.seed(111)
table(discretize(data$age, "cluster" ))
```

```
##
##   [19,32.6) [32.6,47.7)   [47.7,75]
##         483         366         151
```

Observamos que el algoritmo ha decidido 3 clústeres. A continuación los visualizamos.

```
set.seed(111)
hist(data$age, main="Edad",xlab="Edad", ylab="Cantidad de préstamos",col = "ivory")
abline(v=discretize(data$age, method="cluster", onlycuts=TRUE),col="red")
```



Guardamos los intervalos en una nueva variable.

```
set.seed(111)
data$age_disc <- discretize(data$age, "cluster" )
```

2.3 Valores nulos

Continuamos explorando los datos, a continuación vamos a buscar cuantos valores nulos tenemos por columna.

```
colSums(is.na(data))
```

```
##          checking_balance      months_loan_duration            credit_history
##                         0                         0                         0
##                   purpose                    amount           savings_balance
##                         0                         0                         0
##         employment_length          installment_rate           personal_status
##                         0                         0                         0
##             other_debtors         residence_history                  property
##                         0                         0                         0
##                       age          installment_plan                   housing
##                         0                         0                         0
##          existing_credits                   default               dependents
##                         0                         0                         0
##                 telephone            foreign_worker                       job
##                         0                         0                         0
## months_loan_duration_disc               amount_disc                  age_disc
##                         0                         0                         0
```

Observamos que el conjunto de datos no presenta ningún valor nulo. Pero hemos visto que hay algunos valores que toman el valor de “unknown” vamos a ver cuantos tenemos.

```
sapply(data, function(x) sum(x == "unknown"))
```

```
##          checking_balance      months_loan_duration            credit_history
##                       394                         0                         0
##                   purpose                    amount           savings_balance
##                         0                         0                       183
##         employment_length          installment_rate           personal_status
##                         0                         0                         0
##             other_debtors         residence_history                  property
##                         0                         0                         0
##                       age          installment_plan                   housing
##                         0                         0                         0
##          existing_credits                   default               dependents
##                         0                         0                         0
##                 telephone            foreign_worker                       job
##                         0                         0                         0
## months_loan_duration_disc               amount_disc                  age_disc
##                         0                         0                         0
```

Observamos que tenemos valores “unknown” tanto en checking_balance como en savings_balance. De momento, los dejamos pero lo tendremos en cuenta en el próximo estudio.

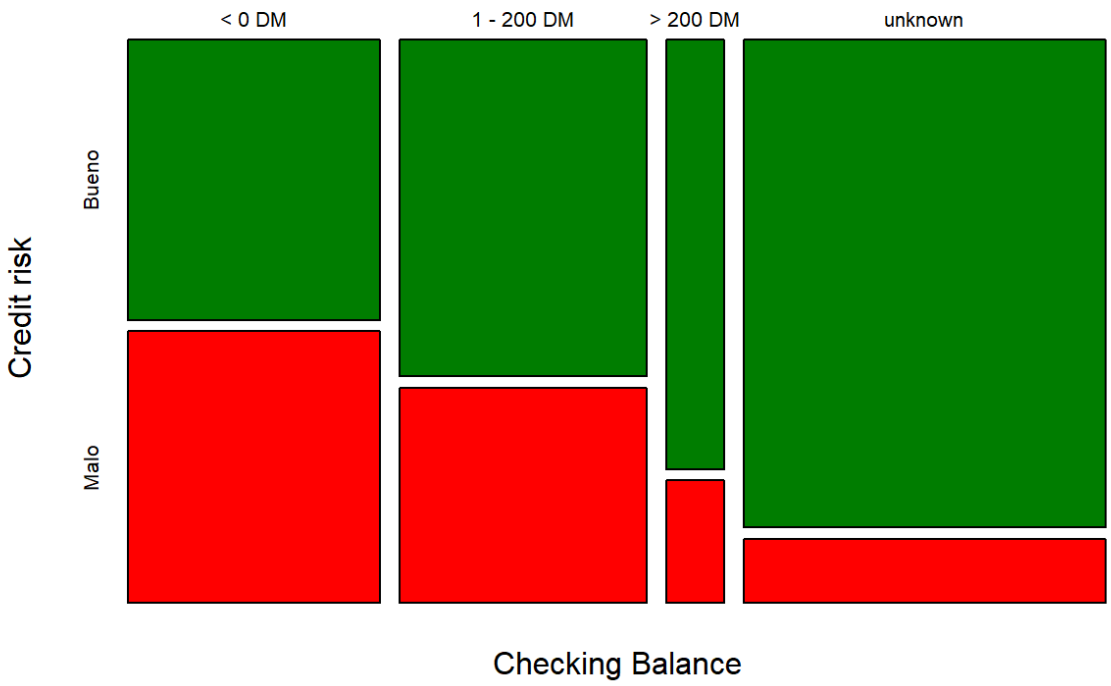
2.4 Visualización

A continuación vamos a visualizar las diferentes variables en función de la variable objetivo para conocer más acerca del conjunto de datos.

Checking balance

En el siguiente gráfico observamos como el riesgo disminuye cuanto mayor es la cantidad en la cuenta corriente. Una de las categorias es “desconocido” por lo que habrá que tenerlo en cuenta en la realización del árbol de decisión.

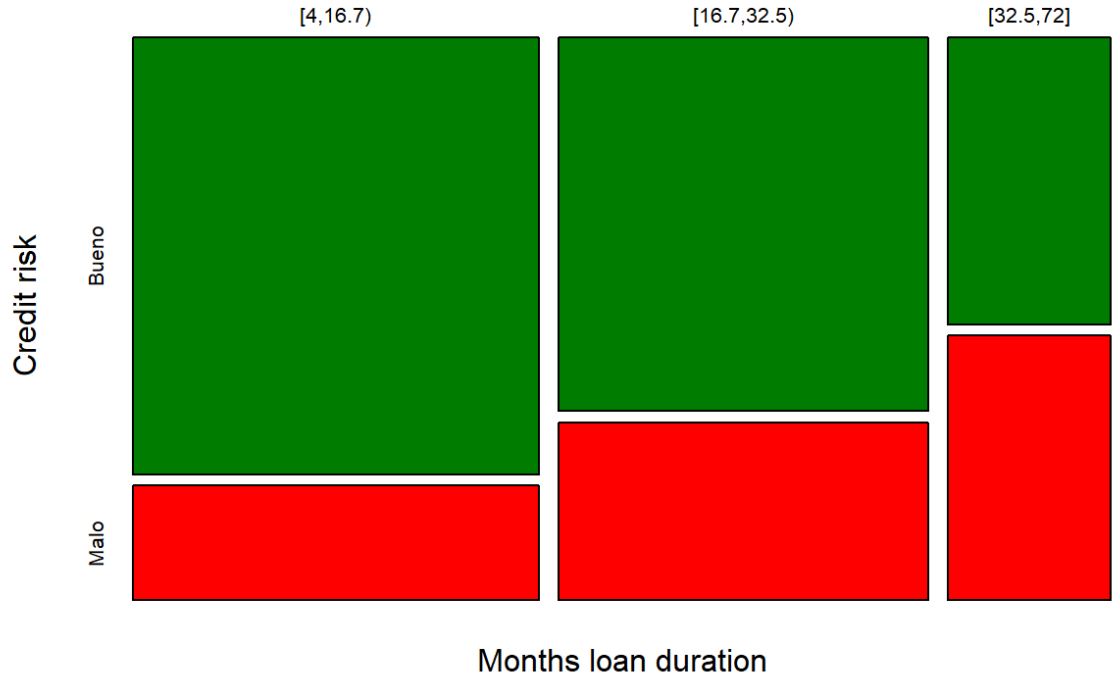
```
plot(table(checking_balance, data$default)[c("< 0 DM","1 - 200 DM","> 200 DM","unknow
n"),],
     col = c("#008000","red"),
     main = "",
     xlab = "Checking Balance",
     ylab = "Credit risk")
```



Months loan duration

En el siguiente gráfico observamos como el riesgo aumenta en función de los meses del prestamo. Es decir, cuantos más meses abarca el préstamo mayor es el riesgo de impago.

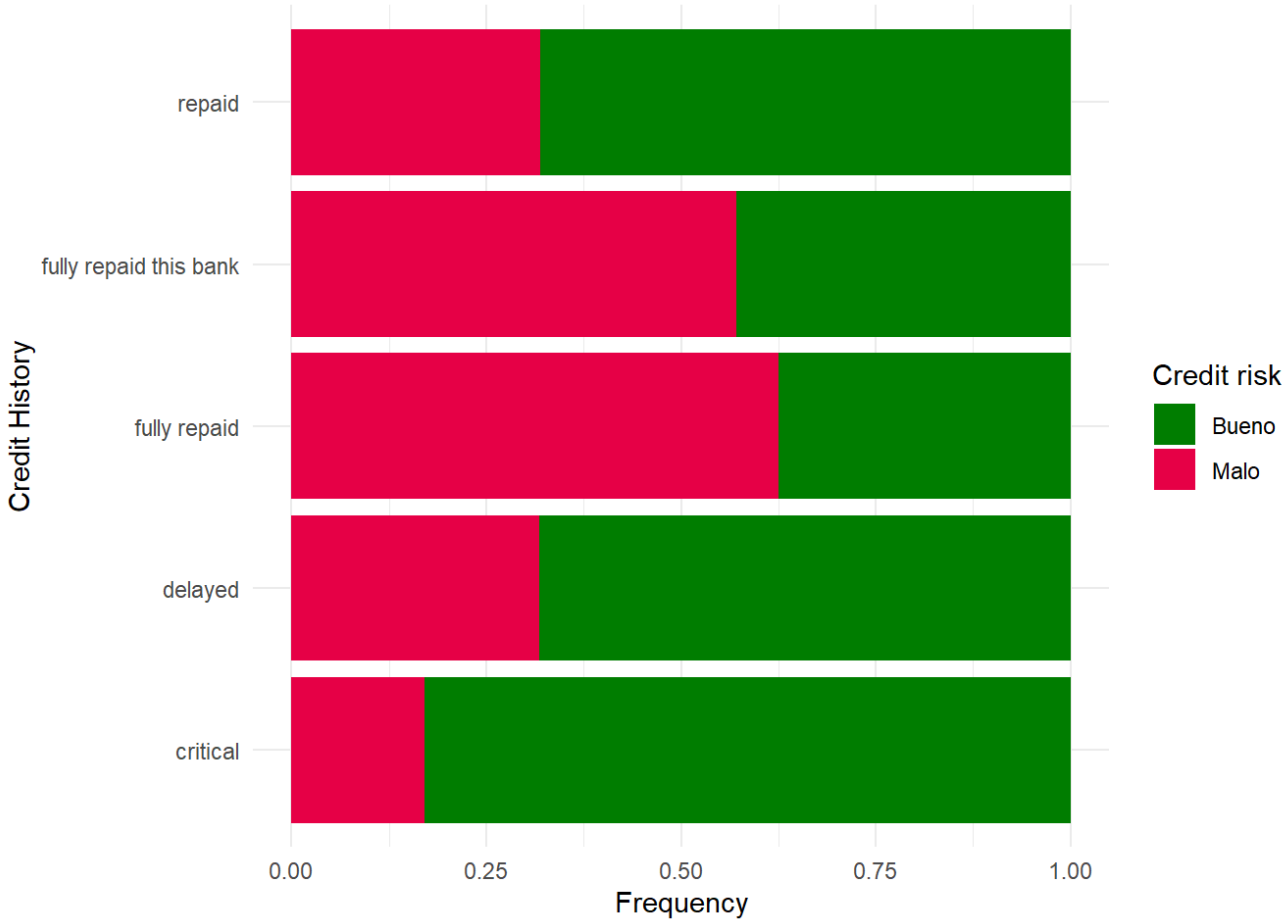
```
plot(table(data$months_loan_duration_disc,data$default), col = c("#008000","red"),
     main = "",
     xlab = "Months loan duration",
     ylab = "Credit risk")
```



Credit history

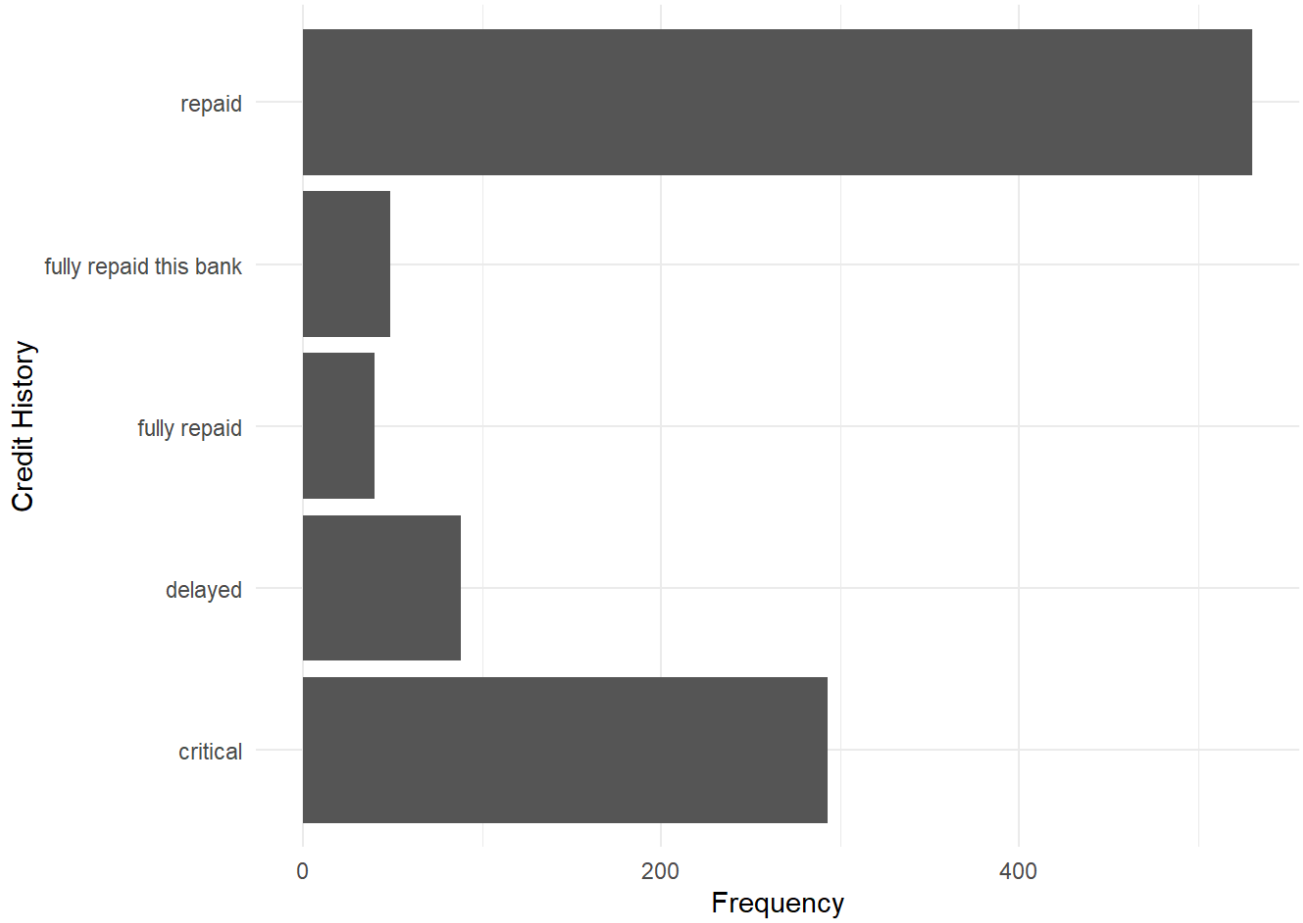
La información del siguiente gráfica puede parecer contradictoria ya que proporcionalmente indica más riesgo en los registros “fully repaid” que los valores “critical” y “delayed”.

```
df <- data %>%
  count(credit_history, default) %>%
  group_by(credit_history) %>%
  mutate(freq = n / sum(n)) %>%
  ungroup()

ggplot(data = df, aes(x = credit_history, fill = as.factor(default), y = freq, )) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(x = "Credit History", y = "Frequency") +
  scale_fill_manual(values = c("#008000", "#e60047")) +
  coord_flip()+
  guides(fill = guide_legend(title = "Credit risk"))
```



A continuación observamos que hay muy pocos registros “fully repaid” por lo que parece que proporcionalmente tienen bastantes registros de riesgo.

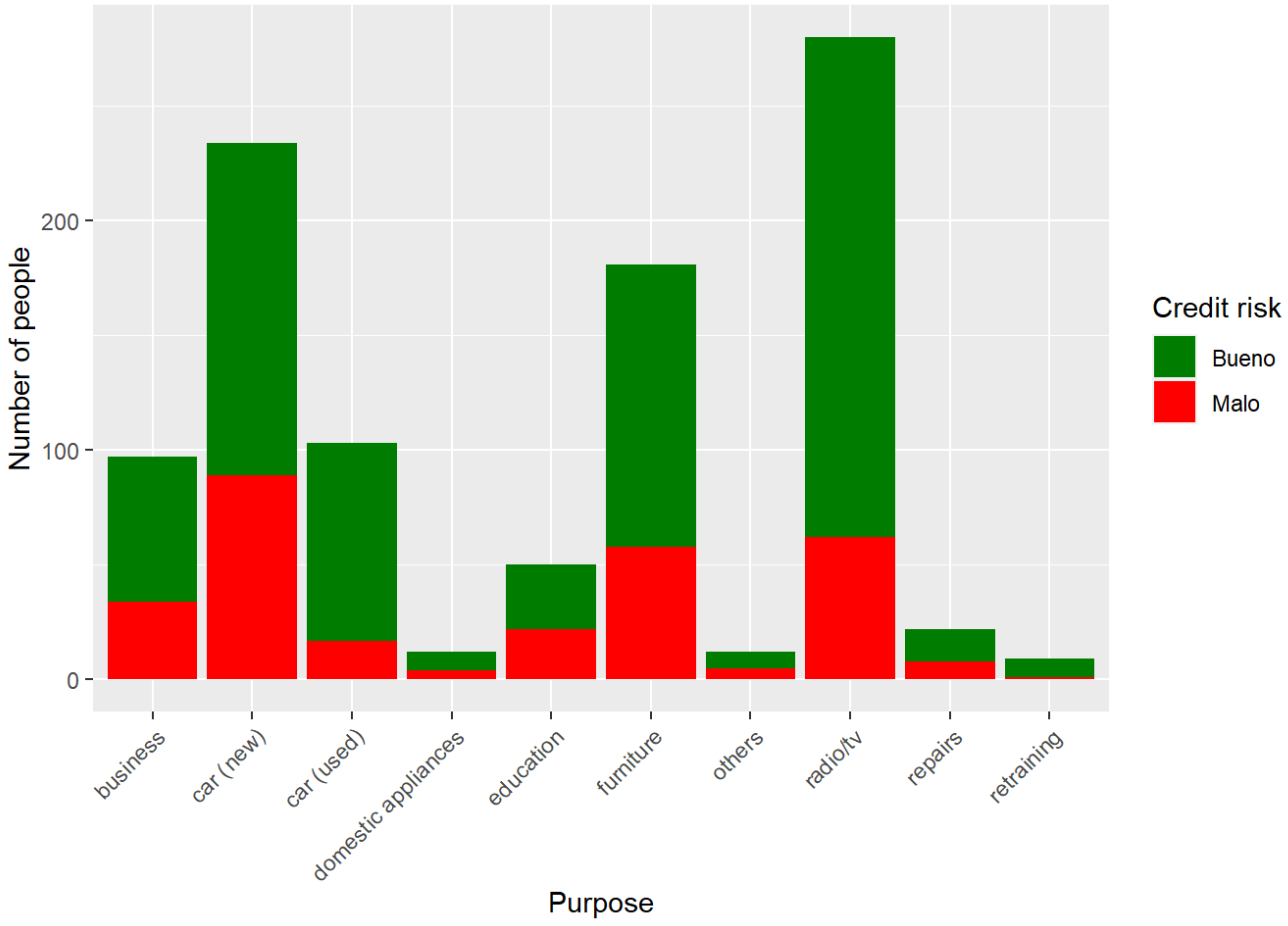
```
ggplot(data, aes(x = credit_history)) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Credit History", y = "Frequency") +coord_flip()
```



Purpose

El propósito del crédito observamos que es variado y que en todos ellos existen registros con riesgo y sin.

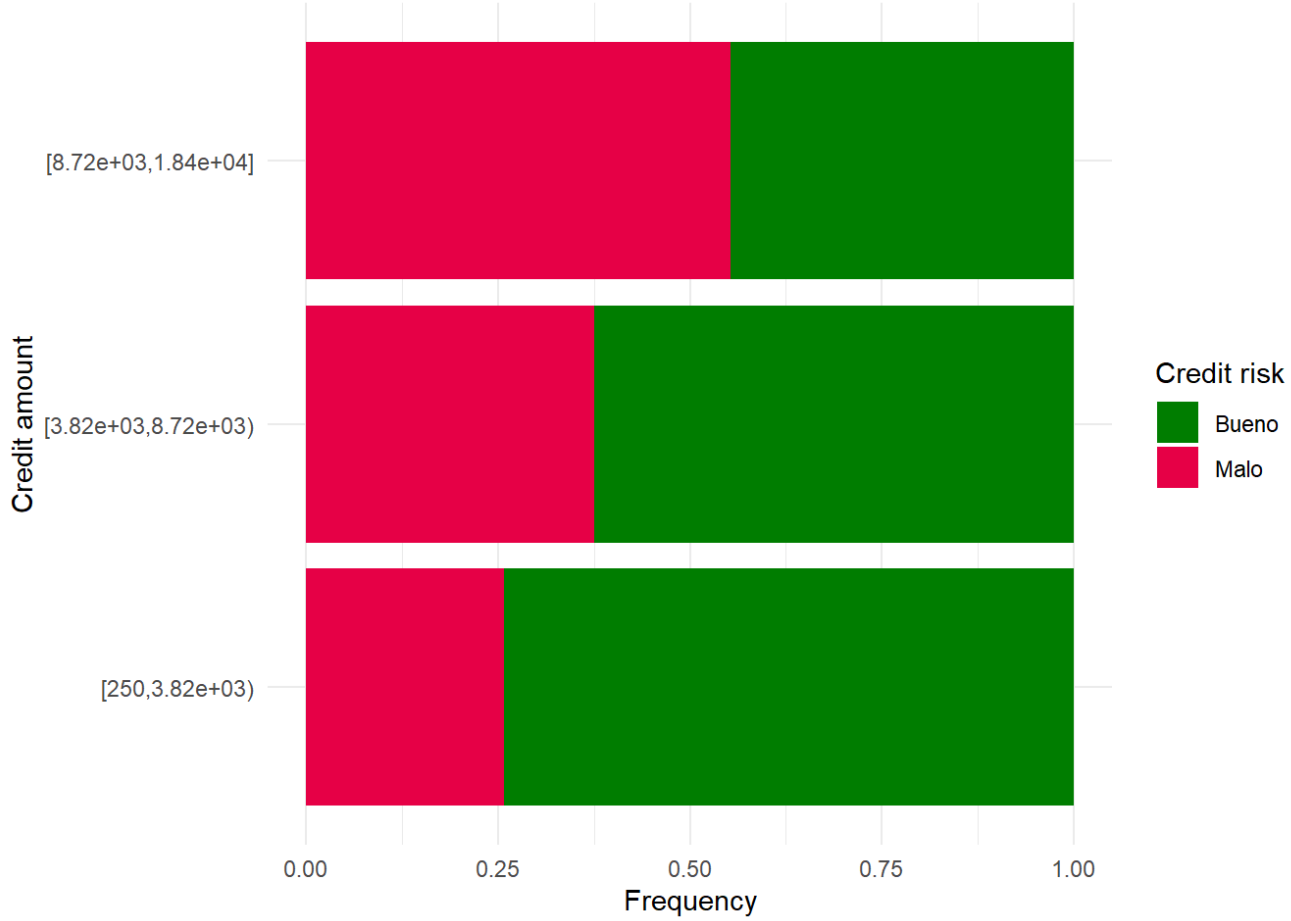
```
ggplot(data,aes(purpose))+geom_bar(aes(fill = default)) +labs(x="Purpose", y="Number of people")+ guides(fill=guide_legend(title="Credit risk"))+ scale_fill_manual(values=c("#008000","red"))+  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Amount

Observamos a continuación que según aumenta el crédito también aumenta el riesgo.

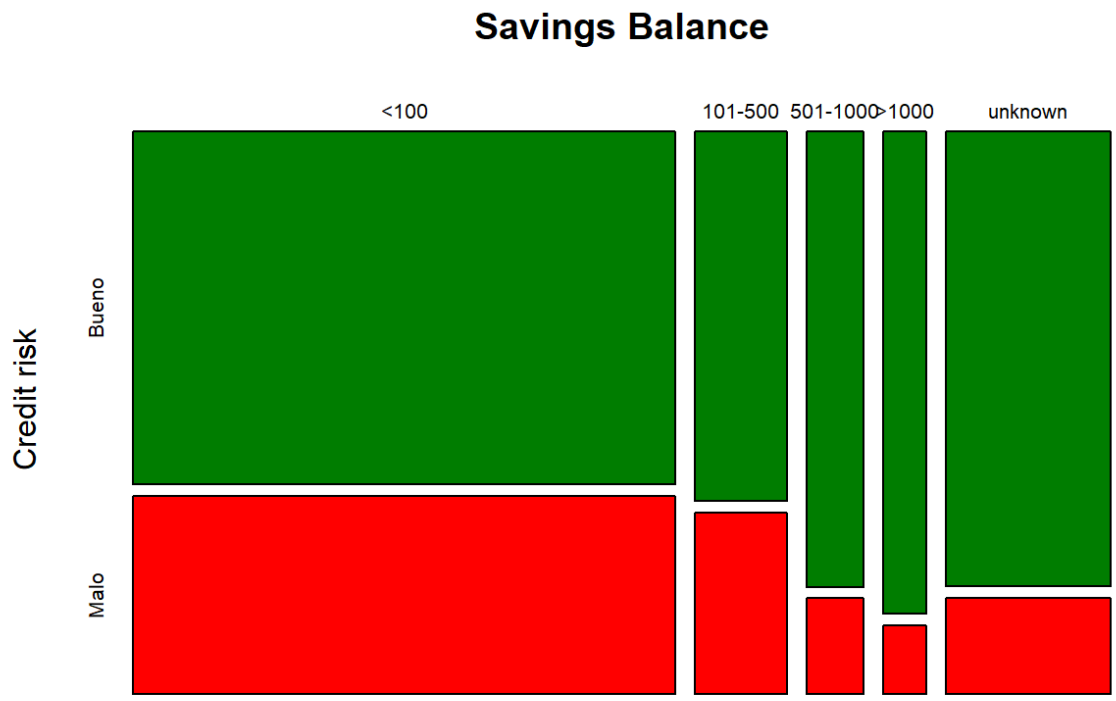
```
df <- data %>%
  count(amount_disc, default) %>%
  group_by(amount_disc) %>%
  mutate(freq = n / sum(n)) %>%
  ungroup()

ggplot(data = df, aes(x = amount_disc, fill = as.factor(default), y = freq, )) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(x = "Credit amount", y = "Frequency") +
  scale_fill_manual(values = c("#008000", "#e60047")) +
  coord_flip()+
  guides(fill = guide_legend(title = "Credit risk"))
```



Savings balance

El riesgo disminuye a medida que aumentan los ahorros de dicha persona.

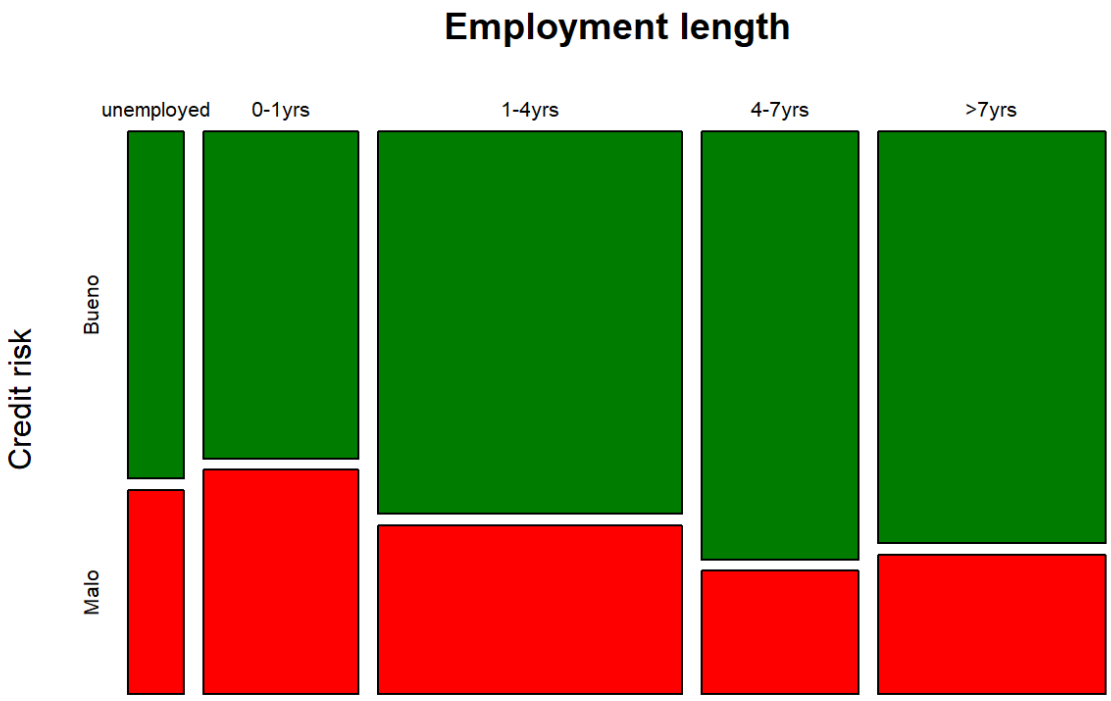
```
plot(table(data$savings_balance,data$default), col = c("#008000","red"),main="Savings Ba
lance", ylab = "Credit risk")
```



Employment length

El riesgo disminuye en función de los años de empleo, hasta llegar a los 7 años, donde parece que aumenta el riesgo ligeramente.

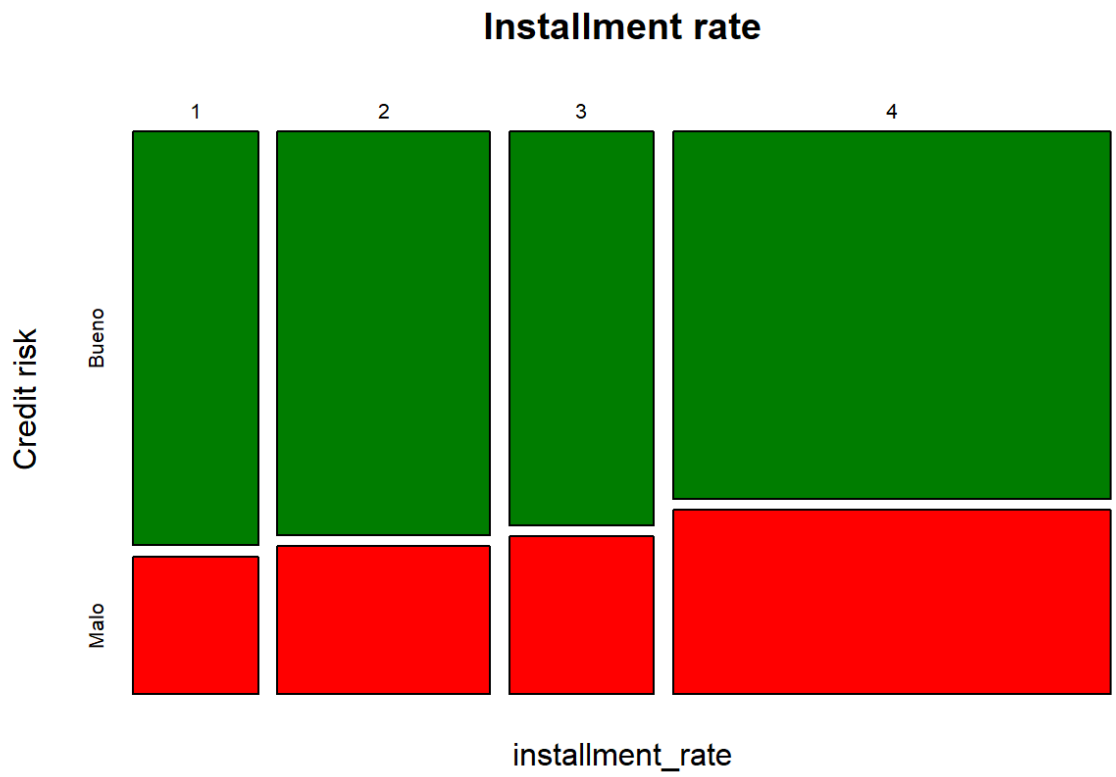
```
plot(table(data$employment_length,data$default), col = c("#008000","red"), main = "Emplo
yment length", ylab ="Credit risk")
```



Installment rate

El riesgo aumenta muy ligeramente en función de la tasa de cuota en porcentaje del ingreso disponible.

```
plot(table(installment_rate,data$default), col = c("#008000","red"), main = "Installment
rate", ylab = "Credit risk")
```



Personal status

El estado civil parece tener poca influencia en el riesgo crediticio.

```
plot(table(personal_status,data$default), col = c("#008000","red"), main = "Personal sta
tus", ylab = "Credit risk", xlab = "")
```

Other debtors

Observamos que la gran mayoría de créditos no presentan ni “co-applicant” ni “guarantor”. También se observa que los créditos con “guarantor” persentan menos riesgo que el resto.

```
plot(table(data$other_debtors,data$default), col = c("#008000","red"), main = "Other deb
tors", ylab = "Credit risk", xlab = "")
```

Residence history

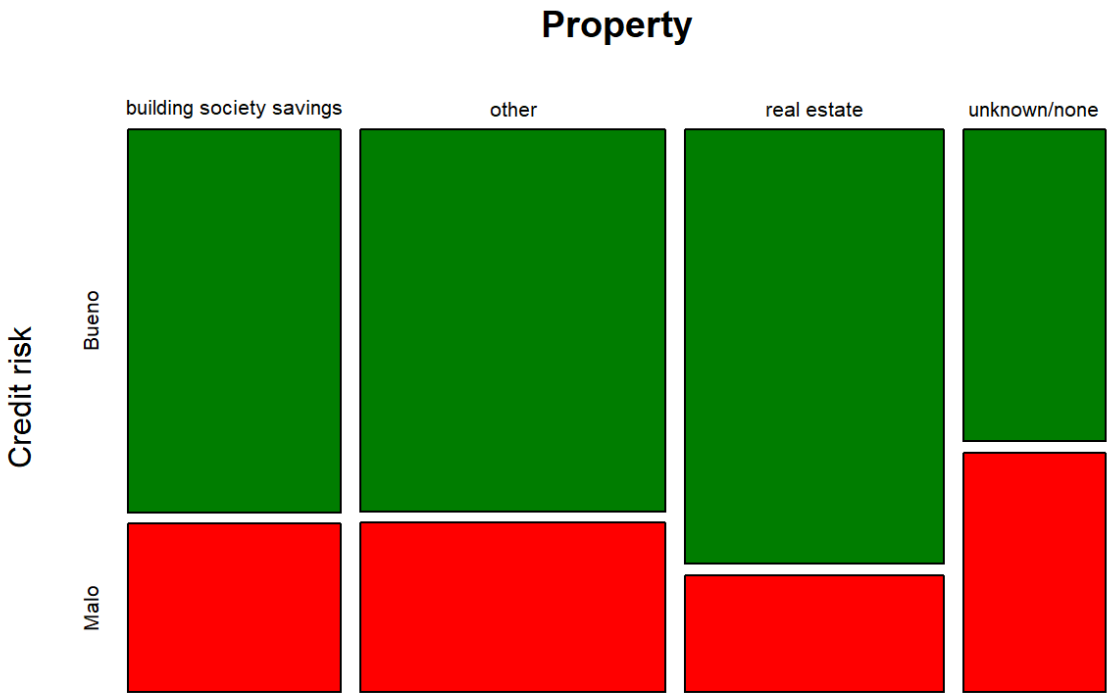
El tiempo que lleva en la residencia actual no parece tener relación con el riesgo.

```
plot(table(residence_history,data$default), col = c("#008000","red"), main = "Residence
history", ylab = "Credit risk", xlab = "")
```

Property

El riesgo disminuye en los propietarios de “real estate”.

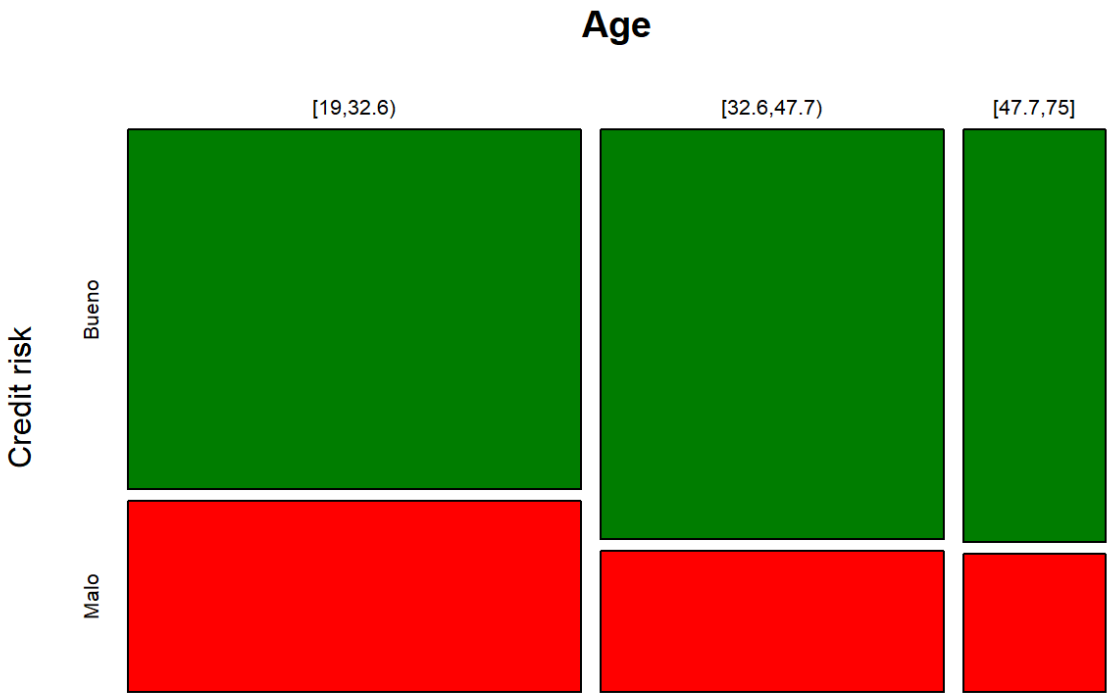
```
plot(table(property,data$default), col = c("#008000","red"), main = "Property", ylab =
"Credit risk", xlab = "")
```



Age

A continuación observamos como la edad se dividide en dos grandes grupos en relación al riesgo crediticio. Menores de 32 años existe más riesgo, y mayores de 32 años, menos riesgo.

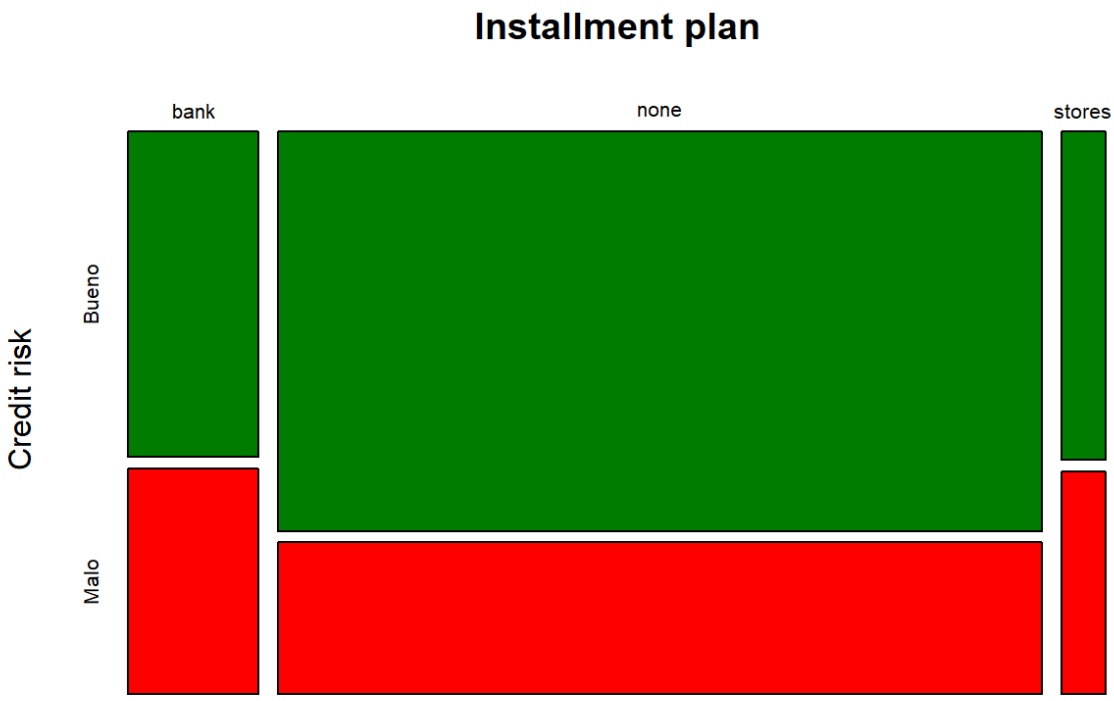
```
plot(table(data$age_disc,data$default), col = c("#008000","red"), main = "Age", ylab = "Credit risk", xlab = "")
```



Installment plan

El gráfico anterior muestra que las personas que no presentan ningún plan de pago a plazos, presentan menos riesgo que los que si que tienen.

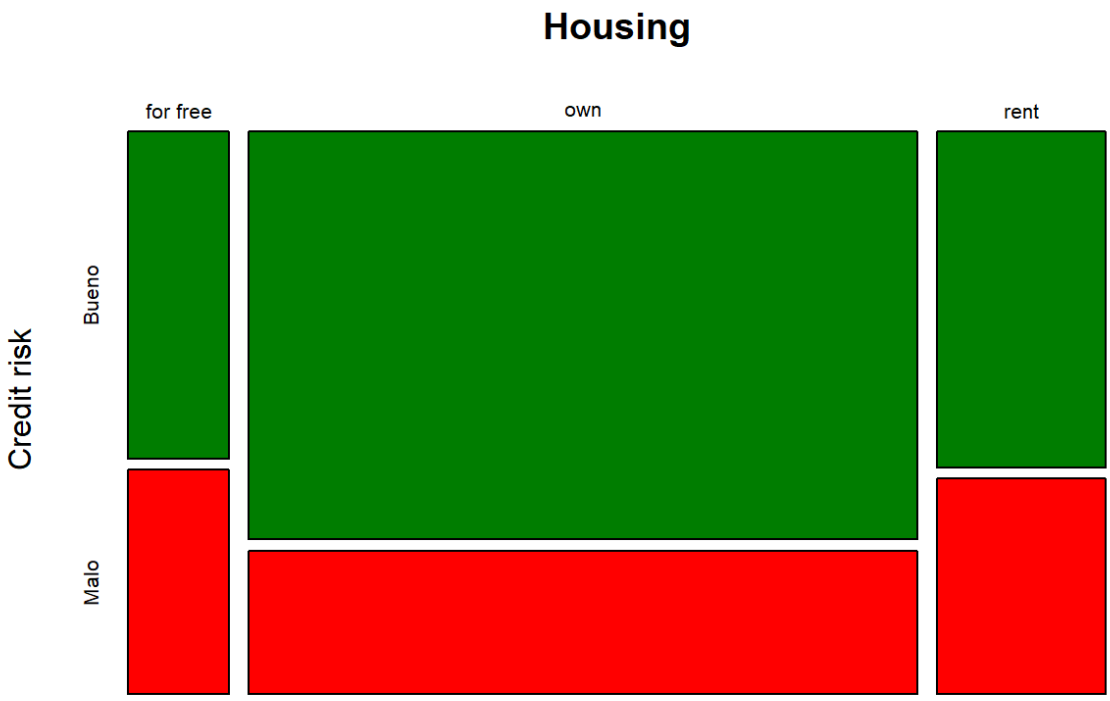
```
plot(table(installment_plan,data$default), col = c("#008000","red"), main = "Installment plan", ylab = "Credit risk", xlab = "")
```



Housing

Los propietarios de su propia vivienda presentan algo menos de riesgo que las personas que viven de alquiler o que las personas que viven gratuitamente en su vivienda.

```
plot(table(housing,data$default), col = c("#008000","red"), main = "Housing", ylab = "Cr
edit risk", xlab = "")
```

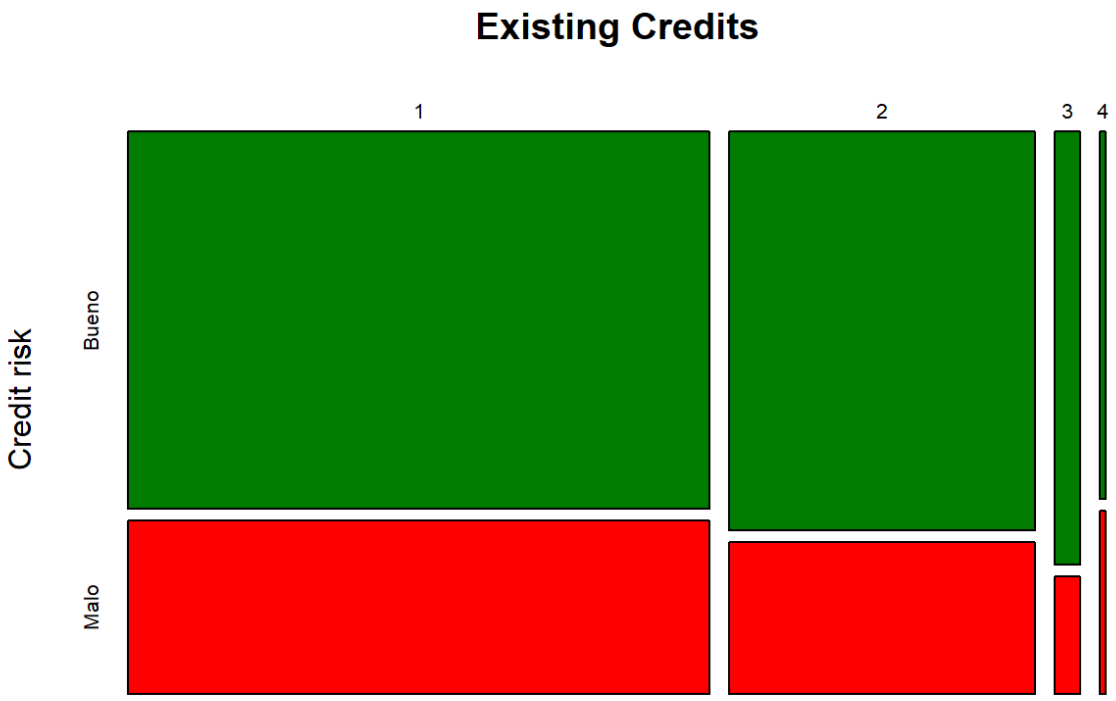


Existing Credits

En la gráfica anterior observamos que las personas que tienen 2 y 3 créditos presentan menos riesgo que las que tienen solo 1 y también que las que tienen 4.

Observamos que el número de muestras de 4 créditos es notablemente inferior al resto de muestras por lo que este conjunto podría estar sesgado.

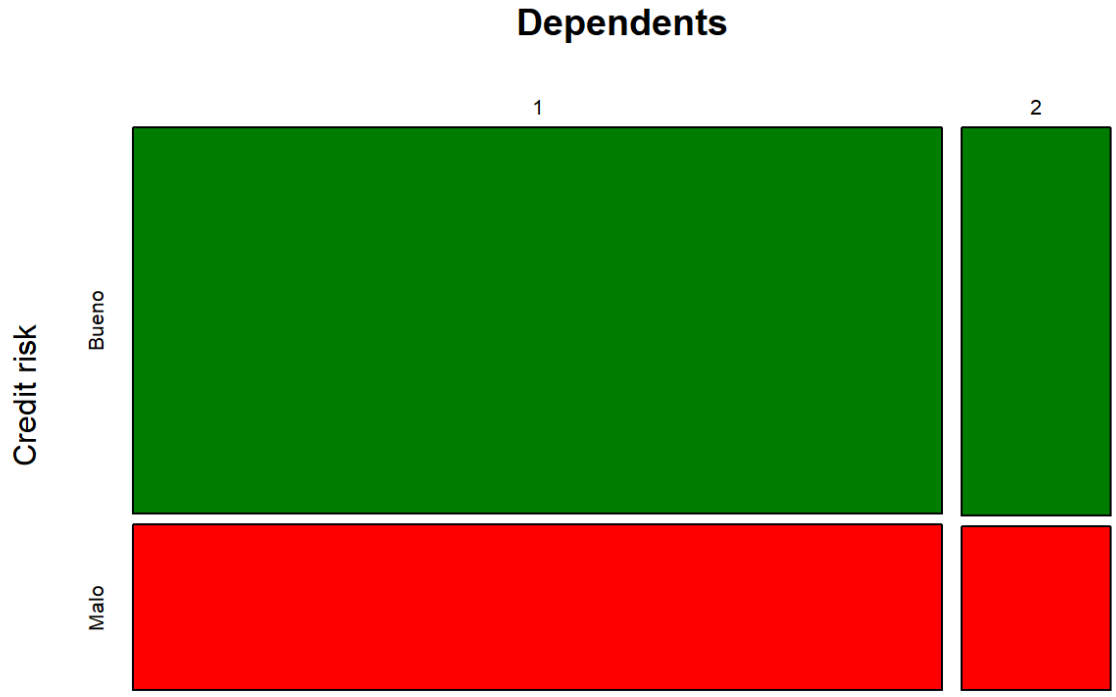
```
plot(table(existing_credits,data$default), col = c("#008000","red"), main = "Existing Cr
edits", ylab = "Credit risk", xlab = "")
```



Dependents

No se observan diferencias de riesgo en lo que se refiere al número de personas a cargo por parte de la persona solicitante del crédito.

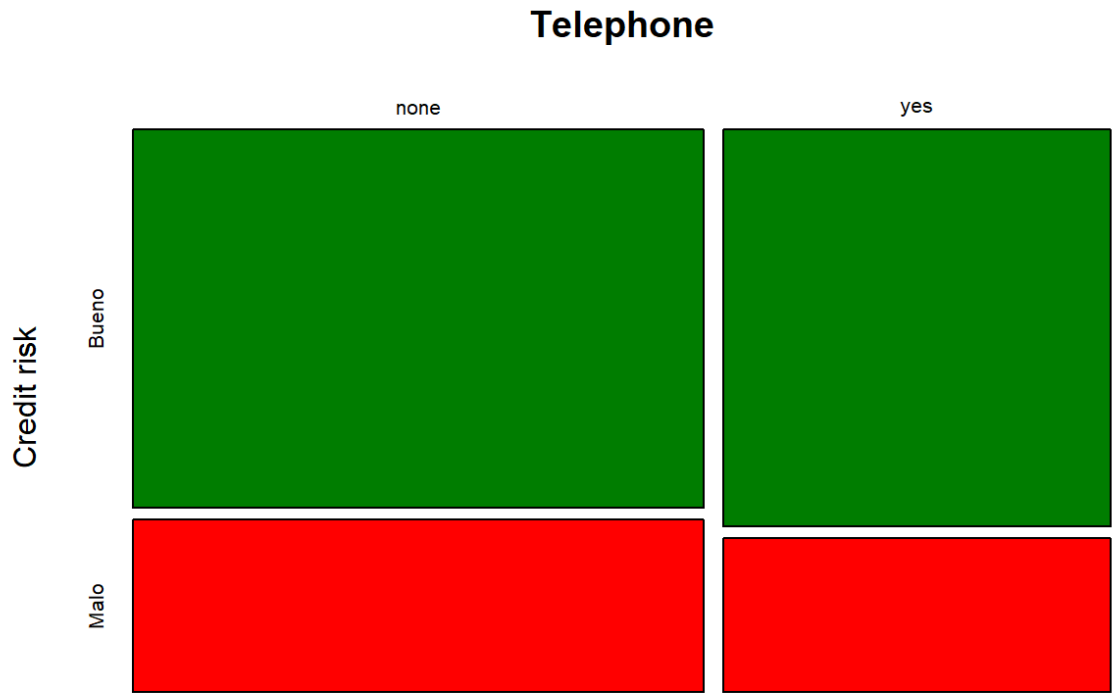
```
plot(table(dependents,data$default), col = c("#008000","red"), main = "Dependents", ylab
= "Credit risk", xlab = "")
```



Telephone

No se observan diferencias de riesgo en lo que se refiere a la posesión de un teléfono.

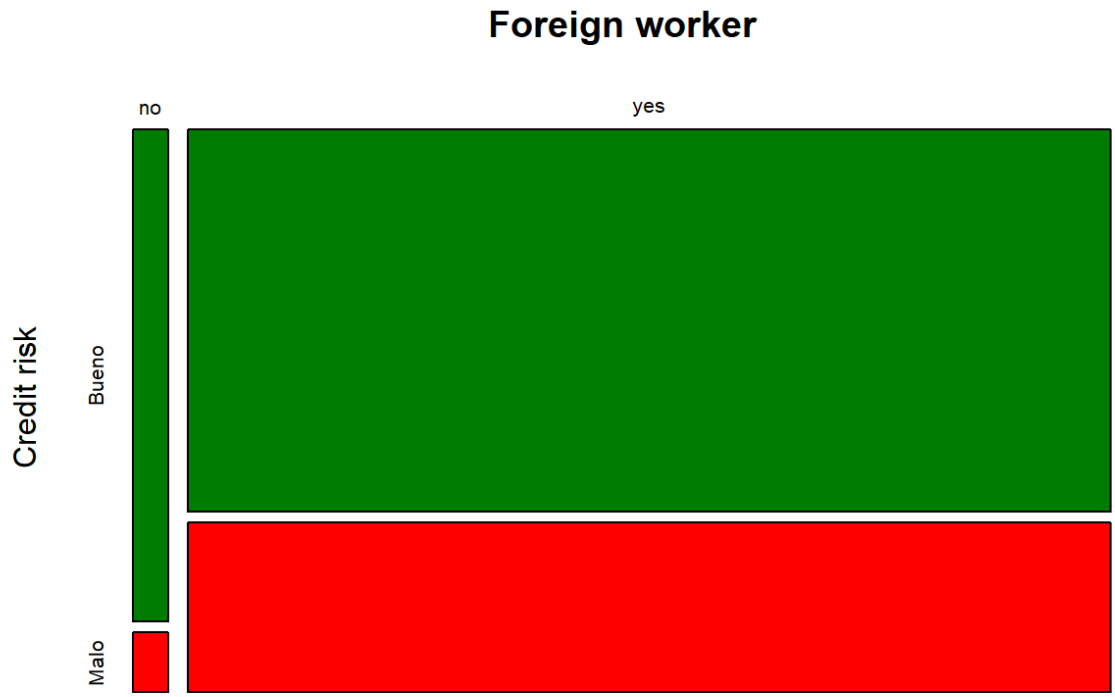
```
plot(table(telephone,data$default), col = c("#008000","red"), main = "Telephone", ylab =
"Credit risk", xlab = "")
```



Foreign worker

Observamos que el trabajador extranjero presenta bastante más riesgo que el que no lo es. También comentar que la muestra de trabajador extranjero es mucho más grande que la otra.

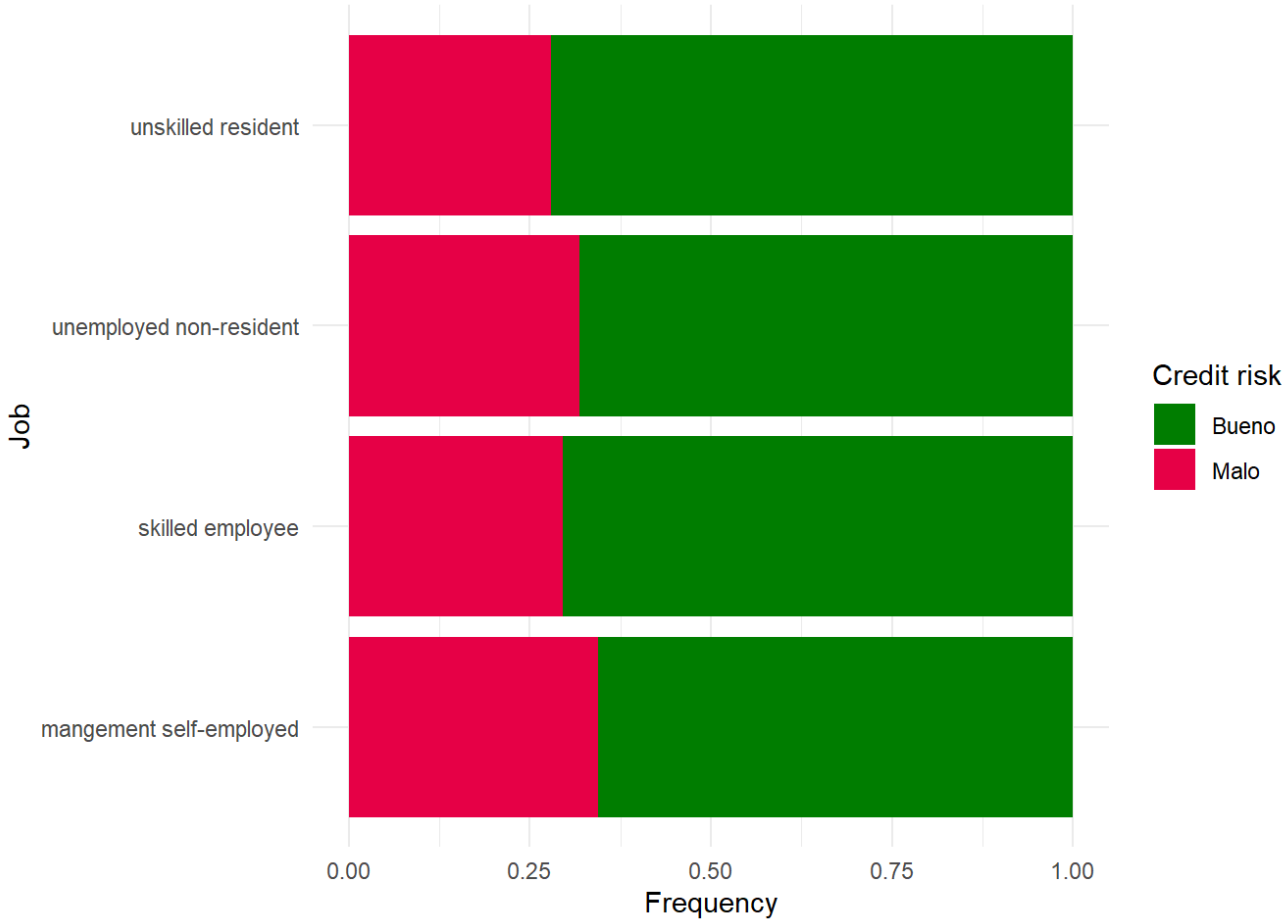
```
plot(table(foreign_worker,data$default), col = c("#008000","red"), main = "Foreign worke
r", ylab = "Credit risk", xlab = "")
```



Job

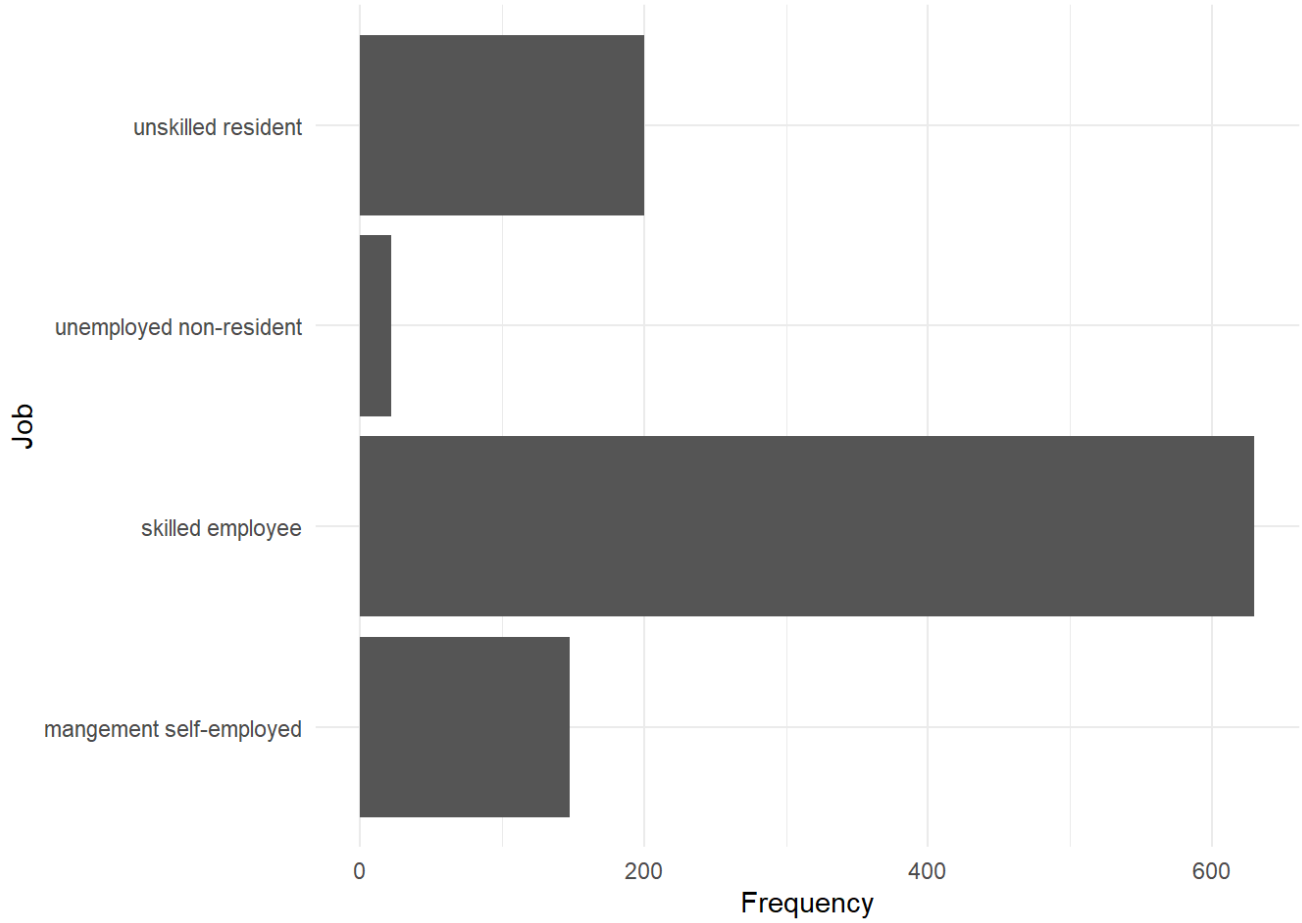
Con respecto al trabajo no se observan diferencias significativas. Podríamos decir que el trabajador autónomo es el que presenta más riesgo, seguido de las personas desempleadas

```
df <- data %>%
  count(job, default) %>%
  group_by(job) %>%
  mutate(freq = n / sum(n)) %>%
  ungroup()

ggplot(data = df, aes(x = job, fill = as.factor(default), y = freq, )) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(x = "Job", y = "Frequency") +
  scale_fill_manual(values = c("#008000", "#e60047")) +
  coord_flip()+
  guides(fill = guide_legend(title = "Credit risk"))
```



Observamos que la muestra de desempleados es muy pequeña en relación al resto.

```
ggplot(data, aes(x = job)) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Job", y = "Frequency") +coord_flip()
```



2.5 Análisis de correlaciones

A continuación vamos a estudiar la relación que existen entre las variables categórigas respecto a la variable objetivo “default”. Puesto que vamos a estudiar la asociación para tablas de contingencia de 2x2 utilizaremos el coeficiente de correlación phi.

```
Phi(table(checking_balance,default))
```

```
## [1] 0.3517399
```

```
Phi(table(data$months_loan_duration_disc,default))
```

```
## [1] 0.2122897
```

```
Phi(table(credit_history,default))
```

```
## [1] 0.2483775
```

```
Phi(table(purpose,default))
```

```
## [1] 0.1826375
```

```
Phi(table(data$amount_disc,default))
```

```
## [1] 0.1702262
```

```
Phi(table(savings_balance,default))
```

```
## [1] 0.1899972
```

```
Phi(table(employment_length,default))
```

```
## [1] 0.1355296
```

```
Phi(table(installment_rate,default))
```

```
## [1] 0.07400535
```

```
Phi(table(personal_status,default))
```

```
## [1] 0.09800619
```

```
Phi(table(other_debtors,default))
```

```
## [1] 0.08151912
```

```
Phi(table(residence_history,default))
```

```
## [1] 0.02737328
```

```
Phi(table(property,default))
```

```
## [1] 0.1540115
```

```
Phi(table(data$age_disc,default))
```

```
## [1] 0.1009425
```

```
Phi(table(installment_plan,default))
```

```
## [1] 0.1133101
```

```
Phi(table(housing,default))
```

```
## [1] 0.1349068
```

```
Phi(table(existing_credits,default))
```

```
## [1] 0.05168364
```

```
Phi(table(dependents,default))
```

```
## [1] 0.003014853
```

```
Phi(table(telephone,default))
```

```
## [1] 0.03646619
```

```
Phi(table(foreign_worker,default))
```

```
## [1] 0.0820795
```

```
Phi(table(job,default))
```

```
## [1] 0.04341838
```

Teniendo en cuenta que la interpretación de este coeficiente podría ser:

- entre 0.1 y 0.3: asociación estadística es baja
- entre 0.3 y 0.5: asociación estadística media
- superior a 0.5: asociación estadística alta

Para la creación de arboles de decisión tendremos en cuenta estos parámetros para no utilizar las variables que presentan una asociación estadística demasiado baja con la variable objetivo.

3 Creación de modelos

3.1 Primer árbol de decisión

Para el primer árbol de decisión, no tendremos en cuenta la variable “checking_balance” ya que tenía un porcentaje muy alto de valores “unknown”. Consideramos que se podría tratar como un valor NULO por lo que probaremos a realizar un árbol sin contar esta variable.

Las variables escogidas para este árbol serán las variables que tenían un coeficiente mayor a 0.15. Por otro lado eliminaremos los registros en los que “savings_balance” es “unknown”.

```
list1 = c("months_loan_duration","credit_history","purpose","amount","savings_balanc
e","employment_length","property","age","installment_plan","housing","default")
data1 <- data[list1]
data1 <- subset(data1, savings_balance != "unknown")
```

A continuación separamos la variable objetivo (y) de las variables predictoras.

```
set.seed(666)
y <- data1[,11]
X <- data1[,1:10]
```

También dividimos el conjunto de datos para entrenar el modelo y para hacer el test. La proporción que utilizaremos será 2/3 para el conjunto de entrenamiento y 1/3 para el conjunto de prueba.

```
split_prop <- 3
indexes = sample(1:nrow(data1), size=floor(((split_prop-1)/split_prop)*nrow(data1)))
trainX<-X[indexes,]
trainy<-y[indexes]
testX<-X[-indexes,]
testy<-y[-indexes]
```

Realizamos un breve análisis de datos para asegurarnos que los subconjuntos presentan valores mezclados, no sesgados.

```
summary(trainX)
```

```
##  months_loan_duration               credit_history       purpose
##  Min.   : 4.00        critical             :164    radio/tv  :160
##  1st Qu.:12.00        delayed              : 43    car (new) :119
##  Median :18.00        fully repaid         : 25    furniture :101
##  Mean   :20.82        fully repaid this bank: 28    business  : 59
##  3rd Qu.:24.00        repaid               :284    car (used): 45
##  Max.   :72.00                                     education : 35
##                                                    (Other)   : 25
##      amount      savings_balance  employment_length
##  Min.   :  250   <100    :391    unemployed: 35
##  1st Qu.: 1381   101-500 : 79    0-1yrs    : 93
##  Median : 2288   501-1000: 40    1-4yrs    :200
##  Mean   : 3162   >1000   : 34    4-7yrs    : 95
##  3rd Qu.: 3944   unknown :  0    >7yrs     :121
##  Max.   :15672
##
##                      property        age        installment_plan     housing
##  building society savings:122   Min.   :20.00   bank  : 76       for free: 55
##  other                    :181   1st Qu.:26.75   none  :439       own     :390
##  real estate              :158   Median :32.00   stores: 29       rent    : 99
##  unknown/none             : 83   Mean   :34.75
##                                  3rd Qu.:40.00
##                                  Max.   :75.00
##
```

```
summary(trainy)
```

```
## Bueno  Malo
##   365   179
```

```
summary(testX)
```

```
##  months_loan_duration                credit_history                purpose
##  Min.   : 4.00        critical              : 73    car (new)          :73
##  1st Qu.:12.00        delayed               : 29    radio/tv           :68
##  Median :18.00        fully repaid          : 11    furniture          :59
##  Mean   :19.85        fully repaid this bank: 13    car (used)         :26
##  3rd Qu.:24.00        repaid                :147    business           :24
##  Max.   :60.00                                      domestic appliances: 6
##                                                     (Other)            :17
##      amount      savings_balance  employment_length
##  Min.   :  339   <100    :212    unemployed:15
##  1st Qu.: 1299   101-500 : 24    0-1yrs    :56
##  Median : 2122   501-1000: 23    1-4yrs    :87
##  Mean   : 3064   >1000   : 14    4-7yrs    :47
##  3rd Qu.: 3650   unknown :  0    >7yrs     :68
##  Max.   :18424
##
##                      property       age        installment_plan     housing
##  building society savings:58   Min.   :19.00   bank  : 37       for free: 30
##  other                    :95   1st Qu.:26.00   none  :225       own     :193
##  real estate              :81   Median :34.00   stores: 11       rent    : 50
##  unknown/none             :39   Mean   :35.88
##                                 3rd Qu.:43.00
##                                 Max.   :74.00
##
```

```
summary(testy)
```

```
## Bueno  Malo
##   184    89
```

Una vez comprobado que no existen diferencias graves que puedan sesgar los resultados continuamos con el modelo.

A continuación creamos el árbol de decisión con los datos de entrenamiento.

```
trainy <-  as.factor(trainy)
model1 <- C50::C5.0(trainX, trainy,rules=TRUE )
summary(model1)
```

```
##
## Call:
## C5.0.default(x = trainX, y = trainy, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Tue May 30 15:06:40 2023
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 544 cases (11 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (437/105, lift 1.1)
##  months_loan_duration <= 42
##  credit_history in {critical, delayed, repaid}
##  amount <= 7814
##  ->  class Bueno  [0.759]
##
## Rule 2: (37/8, lift 2.3)
##  months_loan_duration > 42
##  ->  class Malo  [0.769]
##
## Rule 3: (25/7, lift 2.1)
##  months_loan_duration <= 42
##  amount > 7814
##  ->  class Malo  [0.704]
##
## Rule 4: (53/20, lift 1.9)
##  credit_history in {fully repaid, fully repaid this bank}
##  ->  class Malo  [0.618]
##
## Default class: Bueno
##
##
## Evaluation on training data (544 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##       4  138(25.4%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##     332    33    (a): class Bueno
##     105    74    (b): class Malo
##
##
##  Attribute usage:
```

```
##
##   91.73% months_loan_duration
##   90.07% credit_history
##   84.93% amount
##
##
## Time: 0.0 secs
```

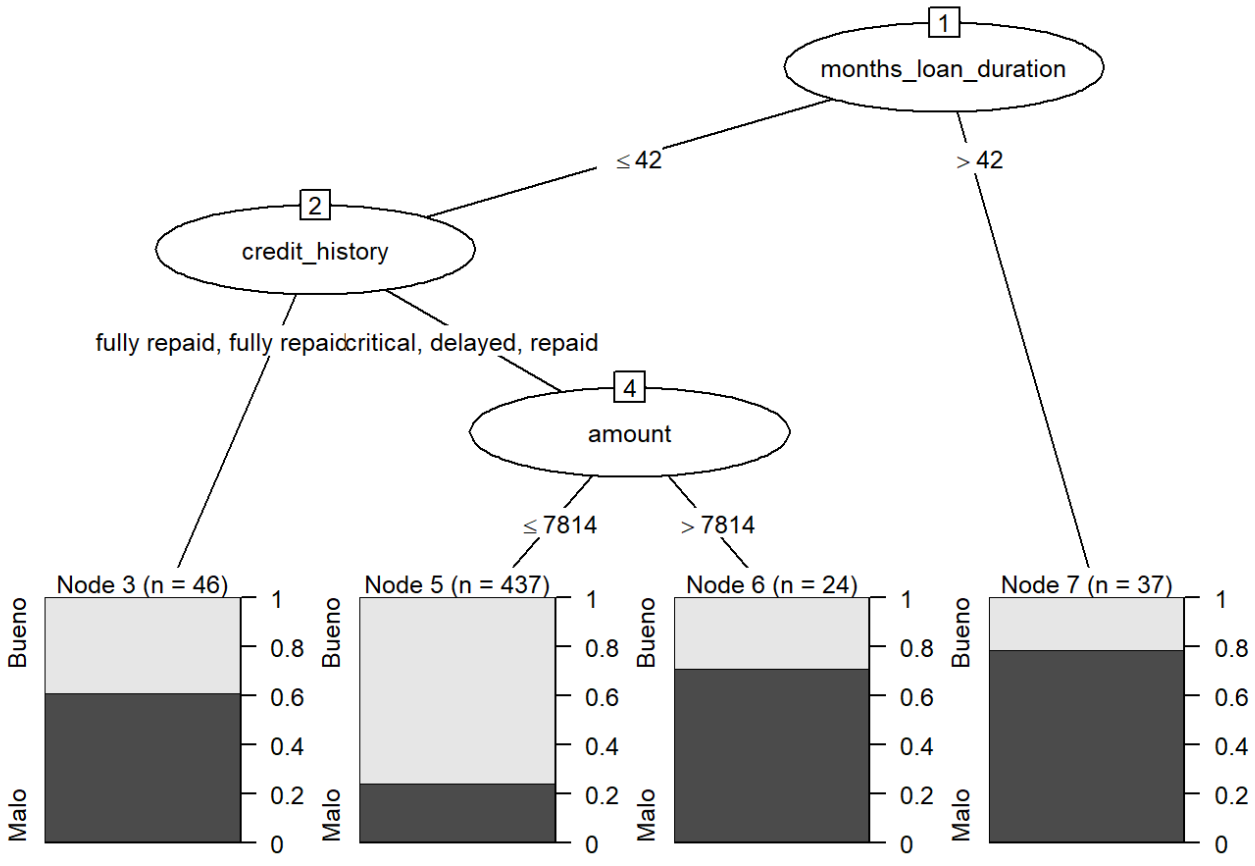
Observamos que el porcentaje de error en este caso es de 25.4%. Es decir, de 544 registros, 138 se clasifican incorrectamente.

Las principales reglas de decisión son las siguientes:

- Regla 1: Si la duración del préstamo es menor o igual a 42 meses, el historial crediticio es crítico, demorado o pagado, y el monto del préstamo es menor o igual a 7814, entonces la clase asignada es “Bueno” con una confianza del 75.9%.
- Regla 2: Si la duración del préstamo es mayor a 42 meses, entonces la clase asignada es “Malo” (existe riesgo) con una confianza del 76.9%.
- Regla 3: Si la duración del préstamo es menor o igual a 42 meses y el monto del préstamo es mayor a 7814, entonces la clase asignada es “Malo” (existe riesgo) con una confianza del 70.4%.
- Regla 4: Si el historial crediticio es completamente pagado o completamente pagado en este banco, entonces la clase asignada es “Malo” (existe riesgo) con una confianza del 61.8%.

El árbol obtenido de este primer modelo es el siguiente:

```
model1 <- C50::C5.0(trainX, trainy)
plot(model1,gp = gpar(fontsize = 9.5))
```



Observamos que aparentemente, la división de “credit_history” no parece tener demasiado sentido. Recordamos que, en el apartado de visualización, había etiquetas con muy pocos registros lo que puede llevar a sesgar los resultados.

Con este modelo, vamos a tratar de predecir la variable objetivo con el subconjunto de datos que habíamos reservado para ello.

```
predicted_model1 <- predict( model1, testX, type="class" )
precision1 <- sum(predicted_model1 == testy) / length(predicted_model1)
print(sprintf("La precisión del árbol es: %.4f ",precision1))
```

```
## [1] "La precisión del árbol es: 0.7143 "
```

La matriz de confusión es la siguiente,

```
mat_conf<-table(testy,Predicted=predicted_model1)
mat_conf
```

```
##        Predicted
## testy   Bueno Malo
##   Bueno   168   16
##   Malo     62   27
```

En la matriz de confusión observamos que 16 falsos positivos y 62 falsos negativos.

El error de tipo 2 (falsos negativos) es el que deberíamos evitar en este caso, es decir, existe riesgo pero el modelo ha predicho que no lo hay. En este caso este error es más alto que el tipo 1.

```
sensibilidad1 <- (168)/(168+62)
print(sprintf("La sensibilidad del modelo es: %.4f",sensibilidad1))
```

```
## [1] "La sensibilidad del modelo es: 0.7304"
```

```
F1 <- 2*((precision1*sensibilidad1)/(precision1+sensibilidad1))
print(sprintf("El F-measure del modelo 1 es: %.4f %%",F1))
```

```
## [1] "El F-measure del modelo 1 es: 0.7223 %"
```

3.1.1 Variación del primer árbol de decisión

En este nuevo árbol vamos a continuar con la perspectiva del árbol anterior pero sin tener en cuenta la variable “credit_history” por lo mencionado anteriormente que una de las categorías presentan muy pocos registros y una de las reglas parece no tener sentido. Lo comprobaremos observando como funciona el modelo sin esta variable.

```
list1.1 = c("months_loan_duration","purpose","amount","savings_balance","employment_leng
th","property","age","installment_plan","housing","default")
data1.1 <- data[list1.1]
data1.1 <- subset(data1.1, savings_balance != "unknown")
```

A continuación separamos la variable objetivo (y) de las variables predictoras.

```
set.seed(666)
y1.1 <- data1.1[,10]
X1.1 <- data1.1[,1:09]
```

También dividimos el conjunto de datos para entrenar el modelo y para hacer el test. La proporción que utilizaremos será 2/3 para el conjunto de entrenamiento y 1/3 para el conjunto de prueba.

```
split_prop <- 3
indexes1.1 = sample(1:nrow(data1.1), size=floor(((split_prop-1)/split_prop)*nrow(data1.
1)))
trainX1.1<-X1.1[indexes1.1,]
trainy1.1<-y1.1[indexes1.1]
testX1.1<-X1.1[-indexes1.1,]
testy1.1<-y1.1[-indexes1.1]
```

Realizamos un breve análisis de datos para asegurarnos que los subconjuntos presentan valores mezclados, no sesgados.

```
summary(trainX1.1)
```

```
##  months_loan_duration       purpose        amount      savings_balance
##  Min.   : 4.00        radio/tv  :160   Min.   :  250   <100    :391
##  1st Qu.:12.00        car (new) :119   1st Qu.: 1381   101-500 : 79
##  Median :18.00        furniture :101   Median : 2288   501-1000: 40
##  Mean   :20.82        business  : 59   Mean   : 3162   >1000   : 34
##  3rd Qu.:24.00        car (used): 45   3rd Qu.: 3944   unknown :  0
##  Max.   :72.00        education : 35   Max.   :15672
##                       (Other)   : 25
##   employment_length                     property        age
##  unemployed: 35     building society savings:122   Min.   :20.00
##  0-1yrs    : 93     other                  :181   1st Qu.:26.75
##  1-4yrs    :200     real estate            :158   Median :32.00
##  4-7yrs    : 95     unknown/none           : 83   Mean   :34.75
##  >7yrs     :121                                   3rd Qu.:40.00
##                                                   Max.   :75.00
##
##  installment_plan     housing
##  bank  : 76       for free: 55
##  none  :439       own     :390
##  stores: 29       rent    : 99
##
##
##
##
```

```
summary(trainy1.1)
```

```
## Bueno  Malo
##   365   179
```

```
summary(testX1.1)
```

```
##  months_loan_duration                purpose       amount      savings_balance
##  Min.   : 4.00        car (new)          :73   Min.   :  339   <100    :212
##  1st Qu.:12.00        radio/tv           :68   1st Qu.: 1299   101-500 : 24
##  Median :18.00        furniture          :59   Median : 2122   501-1000: 23
##  Mean   :19.85        car (used)         :26   Mean   : 3064   >1000   : 14
##  3rd Qu.:24.00        business           :24   3rd Qu.: 3650   unknown :  0
##  Max.   :60.00        domestic appliances: 6   Max.   :18424
##                       (Other)            :17
##   employment_length                    property       age
##  unemployed:15      building society savings:58   Min.   :19.00
##  0-1yrs    :56      other                   :95   1st Qu.:26.00
##  1-4yrs    :87      real estate             :81   Median :34.00
##  4-7yrs    :47      unknown/none            :39   Mean   :35.88
##  >7yrs     :68                                    3rd Qu.:43.00
##                                                   Max.   :74.00
##
##  installment_plan     housing
##  bank  : 37       for free: 30
##  none  :225       own     :193
##  stores: 11       rent    : 50
##
##
##
##
```

```
summary(testy1.1)
```

```
## Bueno  Malo
##   184    89
```

Una vez comprobado que no existen diferencias graves que puedan sesgar los resultados continuamos con el modelo.

A continuación creamos el árbol de decisión con los datos de entrenamiento.

```
trainy1.1 <-  as.factor(trainy)
model1.1 <- C50::C5.0(trainX1.1, trainy1.1,rules=TRUE )
summary(model1.1)
```

```
##
## Call:
## C5.0.default(x = trainX1.1, y = trainy1.1, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Tue May 30 15:06:40 2023
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 544 cases (10 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (23, lift 1.4)
##  months_loan_duration <= 42
##  purpose = car (used)
##  amount <= 8613
##  savings_balance = <100
##  ->  class Bueno  [0.960]
##
## Rule 2: (7, lift 1.3)
##  purpose = furniture
##  installment_plan = stores
##  ->  class Bueno  [0.889]
##
## Rule 3: (103/20, lift 1.2)
##  months_loan_duration <= 42
##  savings_balance in {101-500, >1000}
##  ->  class Bueno  [0.800]
##
## Rule 4: (499/149, lift 1.0)
##  amount <= 6967
##  ->  class Bueno  [0.701]
##
## Rule 5: (16/2, lift 2.5)
##  purpose = car (new)
##  savings_balance = <100
##  installment_plan in {bank, stores}
##  ->  class Malo  [0.833]
##
## Rule 6: (13/2, lift 2.4)
##  months_loan_duration <= 42
##  amount > 8613
##  savings_balance = <100
##  ->  class Malo  [0.800]
##
## Rule 7: (8/1, lift 2.4)
##  months_loan_duration > 16
##  purpose = furniture
##  savings_balance = <100
##  property = other
```

```
##  installment_plan = none
##  ->  class Malo  [0.800]
##
## Rule 8: (7/1, lift 2.4)
##  months_loan_duration <= 42
##  purpose = education
##  savings_balance = <100
##  employment_length in {unemployed, 1-4yrs}
##  ->  class Malo  [0.778]
##
## Rule 9: (37/8, lift 2.3)
##  months_loan_duration > 42
##  ->  class Malo  [0.769]
##
## Rule 10: (2, lift 2.3)
##  months_loan_duration <= 42
##  savings_balance = 501-1000
##  employment_length = 0-1yrs
##  ->  class Malo  [0.750]
##
## Rule 11: (9/2, lift 2.2)
##  months_loan_duration <= 42
##  purpose = radio/tv
##  age <= 22
##  ->  class Malo  [0.727]
##
## Rule 12: (12/3, lift 2.2)
##  months_loan_duration > 11
##  purpose = car (new)
##  savings_balance = <100
##  housing = rent
##  ->  class Malo  [0.714]
##
## Rule 13: (12/3, lift 2.2)
##  months_loan_duration > 11
##  purpose = car (new)
##  amount > 2150
##  savings_balance = <100
##  installment_plan = none
##  housing = own
##  ->  class Malo  [0.714]
##
## Rule 14: (33/10, lift 2.1)
##  months_loan_duration > 16
##  savings_balance = <100
##  housing = rent
##  ->  class Malo  [0.686]
##
## Rule 15: (16/5, lift 2.0)
##  purpose = furniture
##  amount > 4113
##  ->  class Malo  [0.667]
##
```

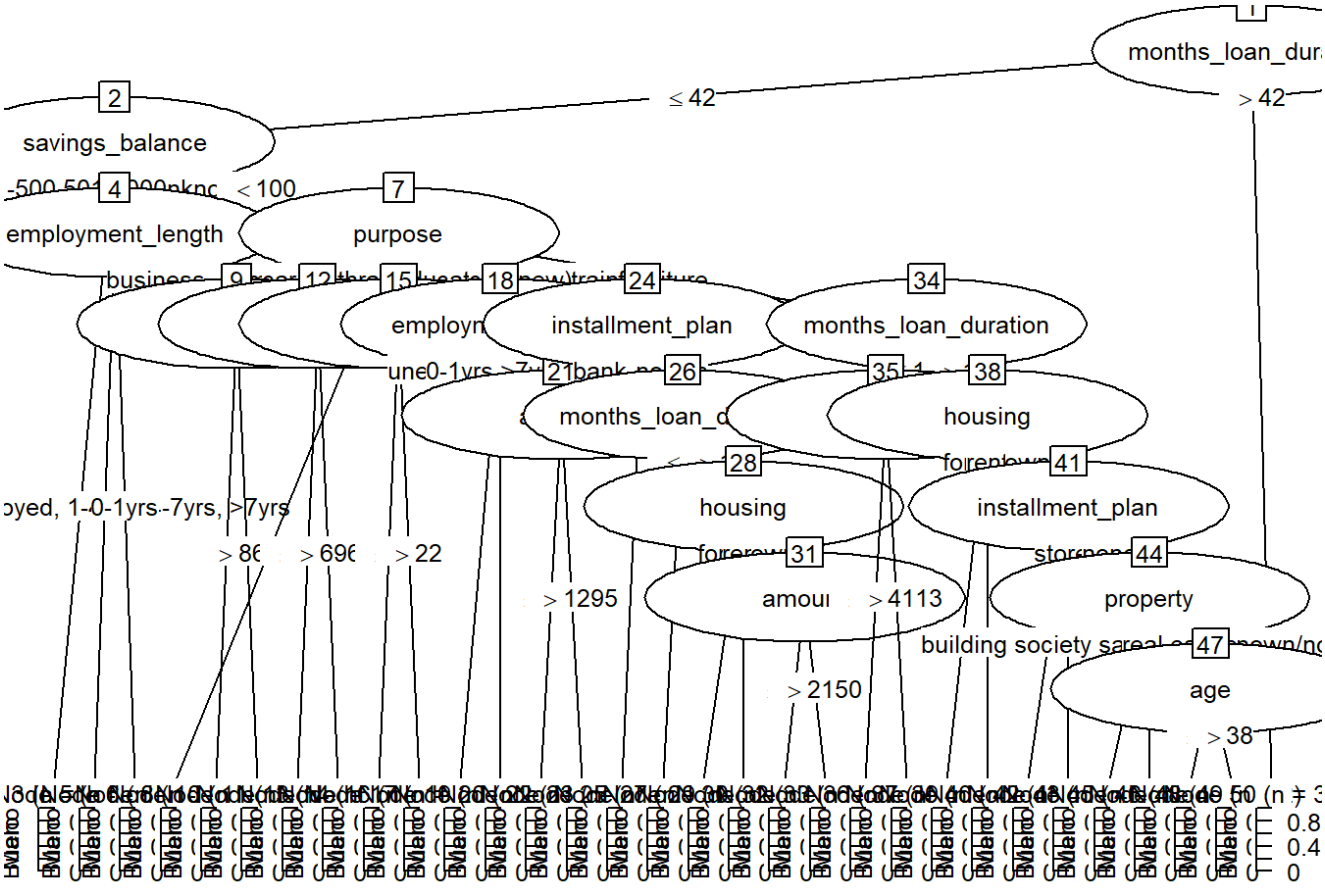
```
## Rule 16: (35/17, lift 1.6)
##  purpose = education
##  ->  class Malo  [0.514]
##
## Rule 17: (51/25, lift 1.5)
##  months_loan_duration > 16
##  installment_plan = bank
##  ->  class Malo  [0.509]
##
## Default class: Bueno
##
##
## Evaluation on training data (544 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##      17  110(20.2%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##     342    23    (a): class Bueno
##      87    92    (b): class Malo
##
##
##  Attribute usage:
##
##   95.77% amount
##   47.98% months_loan_duration
##   39.15% savings_balance
##   24.45% purpose
##   15.99% installment_plan
##    9.38% housing
##    1.65% employment_length
##    1.65% age
##    1.47% property
##
##
## Time: 0.0 secs
```

Observamos que el porcentaje de error en este caso es de 20.2%. Ha mejorado respecto al árbol anterior.

Observamos que el árbol es bastante más grande y menos práctico e intuitivo.

El árbol obtenido de este primer modelo es el siguiente:

```
model1.1 <- C50::C5.0(trainX1.1, trainy1.1)
plot(model1.1,gp = gpar(fontsize = 9.5))
```



Con este modelo, vamos a tratar de predecir la variable objetivo con el subconjunto de datos que habíamos reservado para ello.

```
predicted_model1.1 <- predict( model1.1, testX1.1, type="class" )
precision1.1 <- sum(predicted_model1.1 == testy1.1) / length(predicted_model1.1)
print(sprintf("La precisión del árbol es: %.4f",precision1.1))
```

```
## [1] "La precisión del árbol es: 0.6923"
```

Observamos que la precisión ha disminuido ligeramente.

Vamos a ver si hemos mejorado en la tasa de “Falsos negativos”. La matriz de confusión es la siguiente,

```
mat_conf1.1<-table(testy1.1,Predicted=predicted_model1.1)
mat_conf1.1
```

```
##          Predicted
## testy1.1 Bueno Malo
##    Bueno   155   29
##    Malo     55   34
```

```
sensibilidad1.1 <- (155)/(155+55)
print(sprintf("La sensibilidad del modelo es: %.4f",sensibilidad1.1))
```

```
## [1] "La sensibilidad del modelo es: 0.7381"
```

La sensibilidad ha mejorado muy ligeramente.

```
F1.1 <- 2*((precision1.1*sensibilidad1.1)/(precision1.1+sensibilidad1.1))
print(sprintf("El F-measure del modelo 1.1 es: %.4f",F1.1))
```

```
## [1] "El F-measure del modelo 1.1 es: 0.7145"
```

El F-measure no ha mejorado con respecto al anterior árbol. La mejora de la sensibilidad es demasiado pequeña para la complejidad de este árbol con respecto al enterior.

Vamos a continuar explorando para obtener un árbol con menos falsos negativos y mejor F-measure.

3.2 Segundo árbol de decisión

En el siguiente árbol tendremos en cuenta la variable “checking_balance” pero borraremos los registros que toman valores “unknown”. Tomaremos las variables que tenían un coeficiente mayor a 0.15. También se eliminaran los registros en los que “savings_balance” es “unknown”.

Se ha decidido no utilizar “credit_history” en este árbol para evitar el sesgo que podría darse por las pocas muestras encontradas en algunas de sus etiquetas.

```
list2 = c("checking_balance","months_loan_duration","purpose","amount","savings_balanc
e","employment_length","property","age","installment_plan","housing","default")
data2 <- data[list2]
data2 <- subset(data2, savings_balance != "unknown")
data2 <- subset(data2, checking_balance != "unknown")
```

De nuevo, separamos la variable objetivo (y) de las variables predictoras.

```
set.seed(666)
y2 <- data2[,11]
X2 <- data2[,1:10]
```

Y dividimos el conjunto de datos para entrenar el modelo y para hacer el test.

```
split_prop <- 3
indexes2 = sample(1:nrow(data2), size=floor(((split_prop-1)/split_prop)*nrow(data2)))
trainX2<-X2[indexes2,]
trainy2<-y2[indexes2]
testX2<-X2[-indexes2,]
testy2<-y2[-indexes2]
```

Realizamos un breve análisis de datos para asegurarnos que los subconjuntos presentan valores mezclados, no sesgados.

```
summary(trainX2)
```

```
##    checking_balance months_loan_duration       purpose       amount
##  < 0 DM    :163     Min.   : 6.00        radio/tv  :86   Min.   :  338
##  > 200 DM  : 36     1st Qu.:12.00        car (new) :80   1st Qu.: 1306
##  1 - 200 DM:149     Median :18.00        furniture :72   Median : 2363
##  unknown   :  0     Mean   :21.81        business  :43   Mean   : 3243
##                     3rd Qu.:27.75        car (used):31   3rd Qu.: 4043
##                     Max.   :72.00        education :17   Max.   :15945
##                                          (Other)   :19
##  savings_balance  employment_length                    property
##  <100    :271    unemployed: 25     building society savings: 76
##  101-500 : 44    0-1yrs    : 80     other                    :114
##  501-1000: 16    1-4yrs    :115     real estate              : 94
##  >1000   : 17    4-7yrs    : 53     unknown/none             : 64
##  unknown :  0    >7yrs     : 75
##
##
##       age        installment_plan     housing
##  Min.   :19.00   bank  : 56       for free: 48
##  1st Qu.:26.00   none  :272       own     :230
##  Median :31.00   stores: 20       rent    : 70
##  Mean   :34.87
##  3rd Qu.:41.00
##  Max.   :75.00
##
```

```
summary(trainy2)
```

```
## Bueno  Malo
##   195   153
```

```
summary(testX2)
```

```
##    checking_balance months_loan_duration       purpose       amount
##  < 0 DM    :82      Min.   : 6.0         car (new) :52   Min.   :  276
##  > 200 DM  :17      1st Qu.:12.0         radio/tv  :46   1st Qu.: 1296
##  1 - 200 DM:75      Median :18.0         furniture :35   Median : 2257
##  unknown   : 0      Mean   :20.4         business  :10   Mean   : 3350
##                     3rd Qu.:24.0         car (used):10   3rd Qu.: 3956
##                     Max.   :60.0         education : 8   Max.   :18424
##                                          (Other)   :13
##  savings_balance  employment_length                    property
##  <100    :141    unemployed:13      building society savings:48
##  101-500 : 20    0-1yrs    :34      other                    :48
##  501-1000:  7    1-4yrs    :59      real estate              :52
##  >1000   :  6    4-7yrs    :29      unknown/none             :26
##  unknown :  0    >7yrs     :39
##
##
##       age        installment_plan     housing
##  Min.   :20.00   bank  : 21       for free: 17
##  1st Qu.:26.00   none  :146       own     :119
##  Median :33.00   stores:  7       rent    : 38
##  Mean   :34.93
##  3rd Qu.:40.75
##  Max.   :74.00
##
```

```
summary(testy2)
```

```
## Bueno  Malo
##    96    78
```

Una vez comprobado que no existen diferencias graves que puedan sesgar los resultados continuamos con el modelo.

A continuación creamos el árbol de decisión con los datos de entrenamiento.

```
trainy2 <-  as.factor(trainy2)
model2 <- C50::C5.0(trainX2, trainy2,rules=TRUE )
summary(model2)
```

```
##
## Call:
## C5.0.default(x = trainX2, y = trainy2, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Tue May 30 15:06:41 2023
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 348 cases (11 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (4, lift 1.5)
##  checking_balance = > 200 DM
##  months_loan_duration > 30
##  ->  class Bueno  [0.833]
##
## Rule 2: (278/103, lift 1.1)
##  months_loan_duration <= 30
##  ->  class Bueno  [0.629]
##
## Rule 3: (10, lift 2.1)
##  months_loan_duration > 21
##  months_loan_duration <= 30
##  purpose = car (new)
##  savings_balance = <100
##  ->  class Malo  [0.917]
##
## Rule 4: (8, lift 2.0)
##  purpose = car (new)
##  employment_length in {0-1yrs, >7yrs}
##  housing = rent
##  ->  class Malo  [0.900]
##
## Rule 5: (8/1, lift 1.8)
##  purpose = radio/tv
##  savings_balance = 101-500
##  ->  class Malo  [0.800]
##
## Rule 6: (12/2, lift 1.8)
##  months_loan_duration <= 30
##  purpose = car (new)
##  property = building society savings
##  ->  class Malo  [0.786]
##
## Rule 7: (63/13, lift 1.8)
##  checking_balance in {< 0 DM, 1 - 200 DM}
##  months_loan_duration > 30
##  employment_length in {0-1yrs, 1-4yrs, 4-7yrs, >7yrs}
##  ->  class Malo  [0.785]
```

```
##
## Rule 8: (9/2, lift 1.7)
##  purpose = furniture
##  installment_plan = bank
##  ->  class Malo  [0.727]
##
## Rule 9: (12/3, lift 1.6)
##  savings_balance = 101-500
##  property = unknown/none
##  ->  class Malo  [0.714]
##
## Rule 10: (17/5, lift 1.6)
##  purpose = education
##  ->  class Malo  [0.684]
##
## Rule 11: (24/8, lift 1.5)
##  checking_balance in {< 0 DM, > 200 DM}
##  months_loan_duration > 16
##  property = real estate
##  ->  class Malo  [0.654]
##
## Default class: Bueno
##
##
## Evaluation on training data (348 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##      11   79(22.7%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##     164    31    (a): class Bueno
##      48   105    (b): class Malo
##
##
##  Attribute usage:
##
##   99.14% months_loan_duration
##   25.57% checking_balance
##   20.40% employment_length
##   17.24% purpose
##   13.79% property
##    7.76% savings_balance
##    2.59% installment_plan
##    2.30% housing
##
##
## Time: 0.0 secs
```

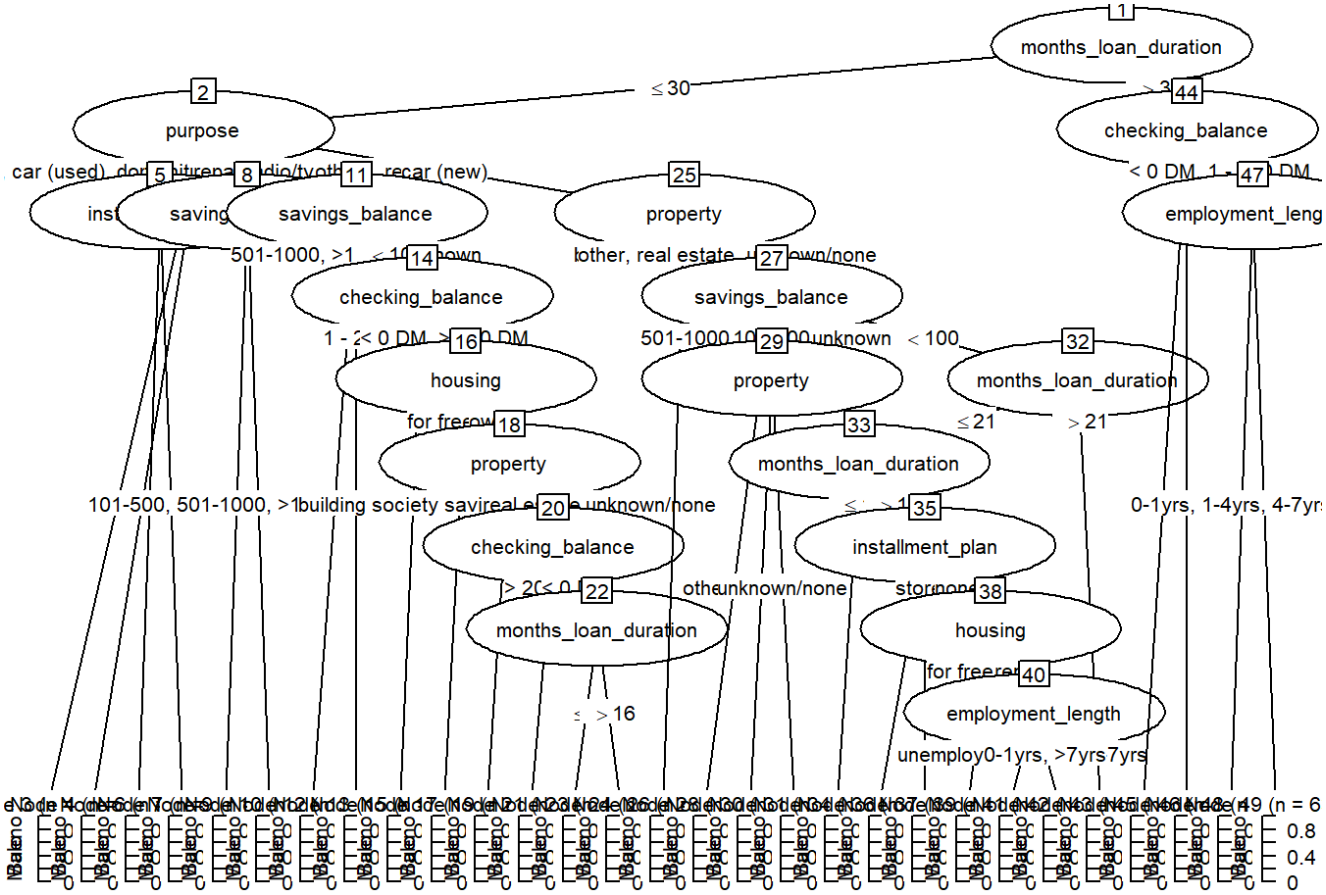
Observamos que el porcentaje de error en este caso es de 22.7%. Es decir, de 348 registros, 79 se clasifican incorrectamente. En este sentido se ha observado una ligera mejora respecto al anterior. También hay que tener en cuenta que este árbol es bastante más grande que el anterior, existen 11 reglas que pueden ser demasiadas para entender intuitivamente.

A continuación se explican las cuatro reglas principales de este árbol:

- Regla 1: Si el saldo de la cuenta corriente es mayor a 200 DM y la duración del préstamo es mayor a 30 meses, entonces se asigna la clase “Bueno” con una confianza del 83.3%.
- Regla 2: Si la duración del préstamo es menor o igual a 30 meses, entonces se asigna la clase “Bueno” con una confianza del 62.9%. Esta regla indica que, en general, para préstamos con una duración más corta, es más probable que la clase sea “Bueno”.
- Regla 3: Si la duración del préstamo está entre 22 y 30 meses, el propósito del préstamo es un automóvil nuevo, el saldo de ahorros es inferior a 100 y se cumple la condición adicional de empleo, entonces se asigna la clase “Malo” con una confianza del 91.7%.
- Regla 4: Si el propósito del préstamo es un automóvil nuevo, el tiempo de empleo está en el rango de 0 a 1 año o es mayor a 7 años, y la vivienda es de alquiler, entonces se asigna la clase “Malo” con una confianza del 90%. Esta regla muestra que para los préstamos destinados a automóviles nuevos con ciertos periodos de empleo y viviendas de alquiler, es más probable que la clase sea “Malo”.

A continuación se observa el árbol obetenido, que según comentado, es desmasiado grande y resulta poco práctico.

```
model2 <- C50::C5.0(trainX2, trainy2)
plot(model2,gp = gpar(fontsize = 8))
```



A continuación se observa como la precisión del modelo ha empeorado considerablemente con respecto al anterior.

```
predicted_model2 <- predict( model2, testX2, type="class" )
precision2 <- sum(predicted_model2 == testy2) / length(predicted_model2)
print(sprintf("La precisión del árbol es: %.4f", precision2))
```

```
## [1] "La precisión del árbol es: 0.6034"
```

```
mat_conf2<-table(testy2,Predicted=predicted_model2)
mat_conf2
```

```
##        Predicted
## testy2  Bueno Malo
##   Bueno    65   31
##   Malo     38   40
```

```
sensibilidad2 <- (65)/(65+38)
print(sprintf("La sensibilidad del modelo es: %.4f",sensibilidad2))
```

```
## [1] "La sensibilidad del modelo es: 0.6311"
```

La sensibilidad también ha empeorado por lo que el F-score, calculado a continuación, también empeorará con respecto a los modelos anteriores.

```
F2 <- 2*((precision2*sensibilidad2)/(precision2+sensibilidad2))
print(sprintf("El F-measure del modelo 2 es: %.4f",F2))
```

```
## [1] "El F-measure del modelo 2 es: 0.6169"
```

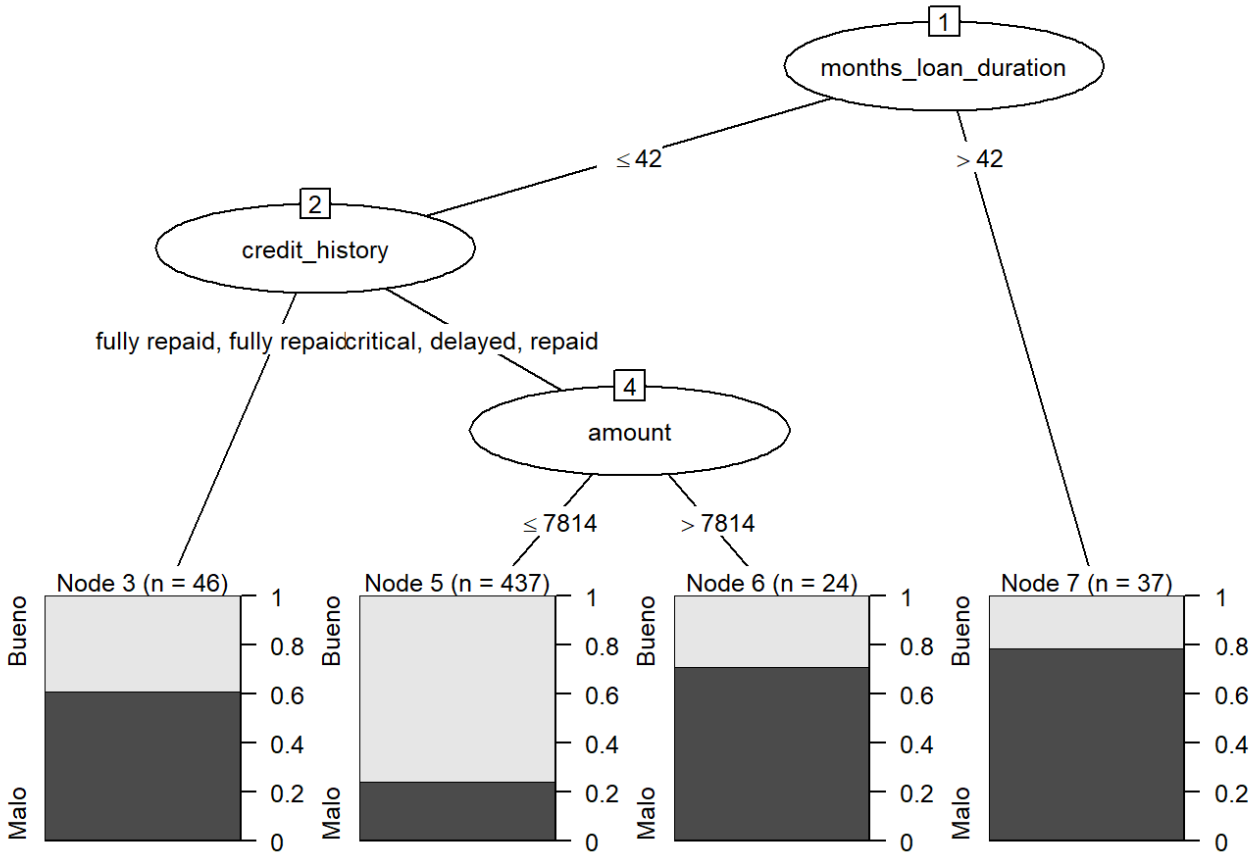
Este modelo presenta una tasa de falsos negativos demasiado alta que haría perder demasiado dinero al prestamista.

3.3 Tercer árbol de decisión con variación del paquete C5.0

A continuación, se exploran las variaciones del paquete C5.0 para generar árboles de decisión. Además, se introduce el “adaptative boosting”, una técnica que utiliza múltiples clasificadores con sus respectivos árboles de decisión y reglas. Los clasificadores emiten votos y se suman para determinar la clase final asignada a un caso. El objetivo es comparar los resultados obtenidos y analizar cómo se modifica el árbol y su capacidad predictiva en el conjunto de prueba.

Puesto que por el momento el primer modelo ha sido el árbol de decisión con F-measure mayor los subconjuntos de datos que se utilizarán a continuación serán los mismos que se han utilizado en el primer árbol.

```
modelo3 <- C50::C5.0(trainX, trainy, trials = 10)
plot(modelo3,gp = gpar(fontsize = 9.5))
```




```
predicted_model3 <- predict( modelo3, testX, type="class" )
precision3 <- sum(predicted_model3 == testy) / length(predicted_model3)
print(sprintf("La precisión del árbol es: %.4f ",precision3))
```

```
## [1] "La precisión del árbol es: 0.7326 "
```

Observamos que la precisión ha mejorado con respecto al árbol 1 realizado anteriormente.

```
mat_conf3<-table(testy,Predicted=predicted_model3)
mat_conf3
```

```
##        Predicted
## testy   Bueno Malo
##   Bueno   169   15
##   Malo     58   31
```

```
sensibilidad3 <- (169)/(169+58)
print(sprintf("La sensibilidad del modelo es: %.4f",sensibilidad3))
```

```
## [1] "La sensibilidad del modelo es: 0.7445"
```

La sensibilidad también ha mejorado, encontrando en este caso menos falsos negativos.

```
F3 <- 2*((precision3*sensibilidad3)/(precision3+sensibilidad3))
print(sprintf("El F-measure del modelo 3 es: %.4f",F3))
```

```
## [1] "El F-measure del modelo 3 es: 0.7385"
```

Por último observamos que el F-measure también ha mejorado en este caso, aumentando de 0.7223 a 0.7385.

Vamos a estudiar la importancia de las variables en este modelo.

```
importancia_usage <- C50::C5imp(modelo3, metric = "usage")
importancia_splits <- C50::C5imp(modelo3, metric = "splits")
importancia_usage
```

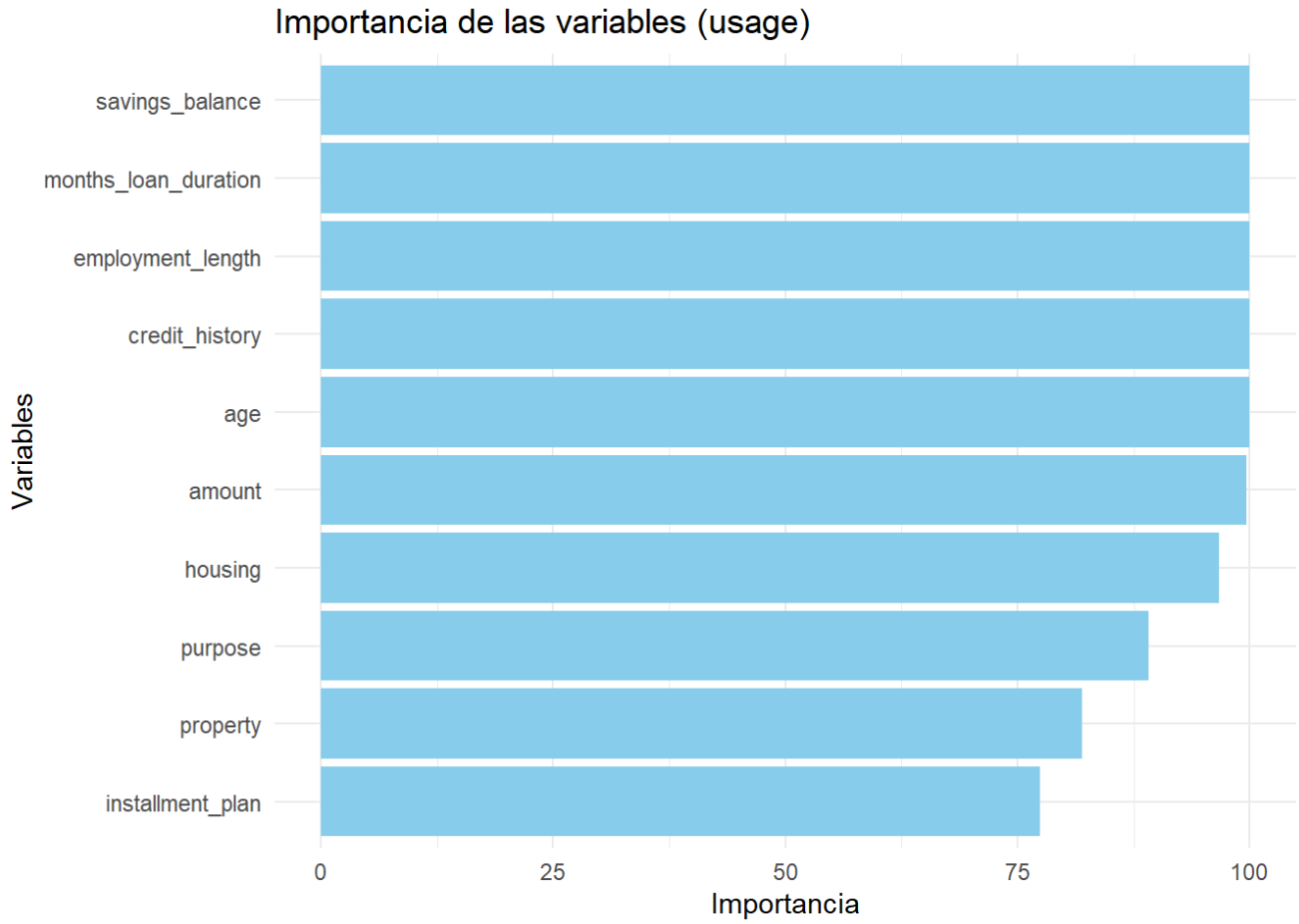
```
##                      Overall
## months_loan_duration  100.00
## credit_history        100.00
## savings_balance       100.00
## employment_length     100.00
## age                   100.00
## amount                 99.63
## housing                96.69
## purpose                89.15
## property               81.99
## installment_plan       77.39
```

```
importancia_splits
```

```
##                       Overall
## months_loan_duration 27.777778
## savings_balance      19.444444
## amount               11.111111
## credit_history       11.111111
## age                   5.555556
## housing               5.555556
## installment_plan      5.555556
## property              5.555556
## purpose               5.555556
## employment_length     2.777778
```

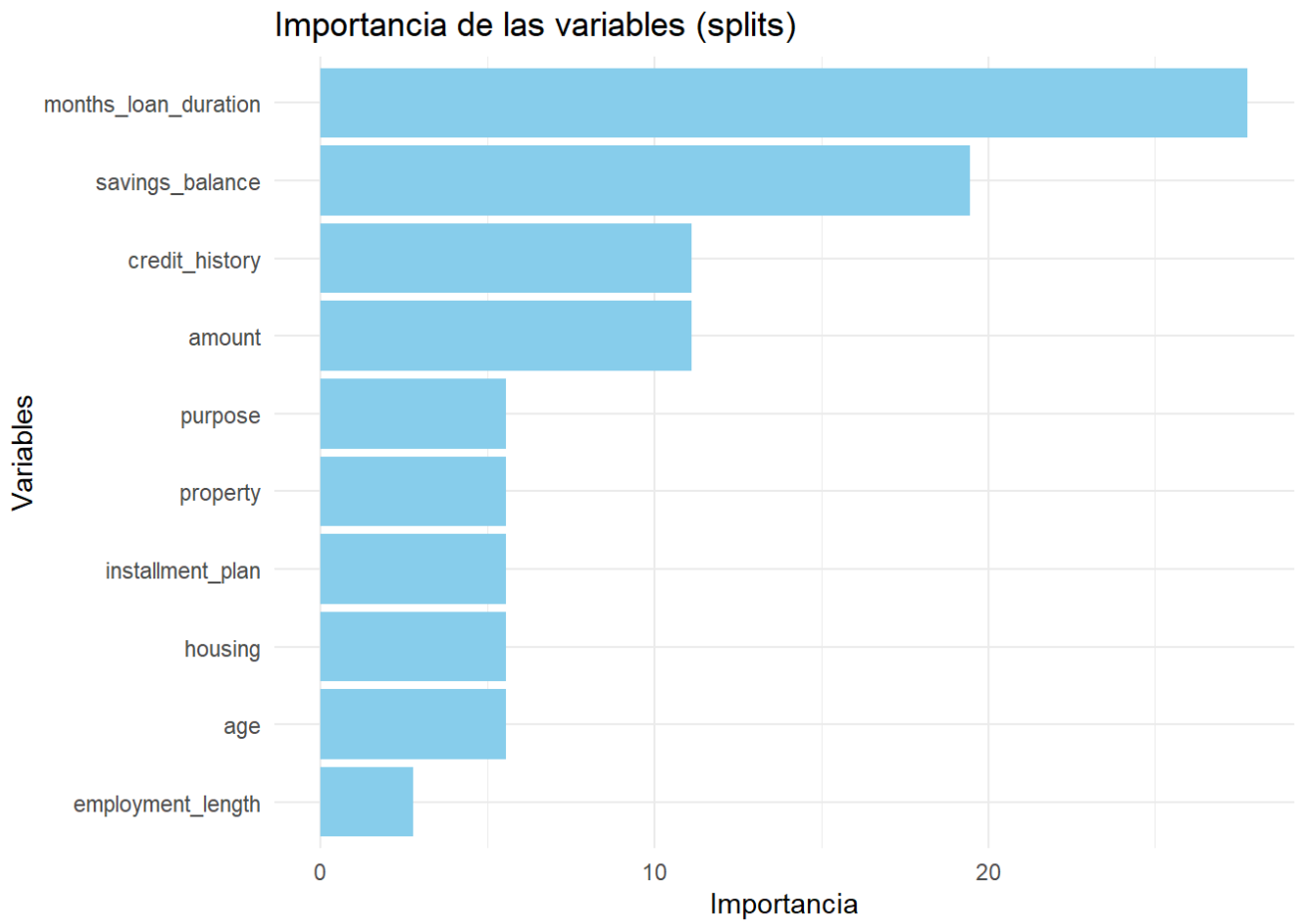
De una manera más visual observamos a continuación la importancia de las variables en función de la frecuencia con la que se utilizan en los árboles generados.

```
# USAGE- Convertir los valores de importancia en un dataframe
usage_df <- data.frame(Variables = rownames(importancia_usage),
                               Importancia = importancia_usage[,1])

# Crear el gráfico de barras con ggplot
ggplot(usage_df, aes(x = Importancia, y = reorder(Variables, Importancia))) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(title = "Importancia de las variables (usage)",
       x = "Importancia", y = "Variables") +
  theme_minimal()
```



Y en el siguiente gráfico observamos la importancia de las variables en función del número de divisiones que se realizan utilizando cada variable en los árboles generados.

```
# SPLITS - Convertir los valores de importancia en un dataframe
splits_df <- data.frame(Variables = rownames(importancia_splits),
                               Importancia = importancia_splits[,1])

# Crear el gráfico de barras con ggplot
ggplot(splits_df, aes(x = Importancia, y = reorder(Variables, Importancia))) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(title = "Importancia de las variables (splits)",
       x = "Importancia", y = "Variables") +
  theme_minimal()
```



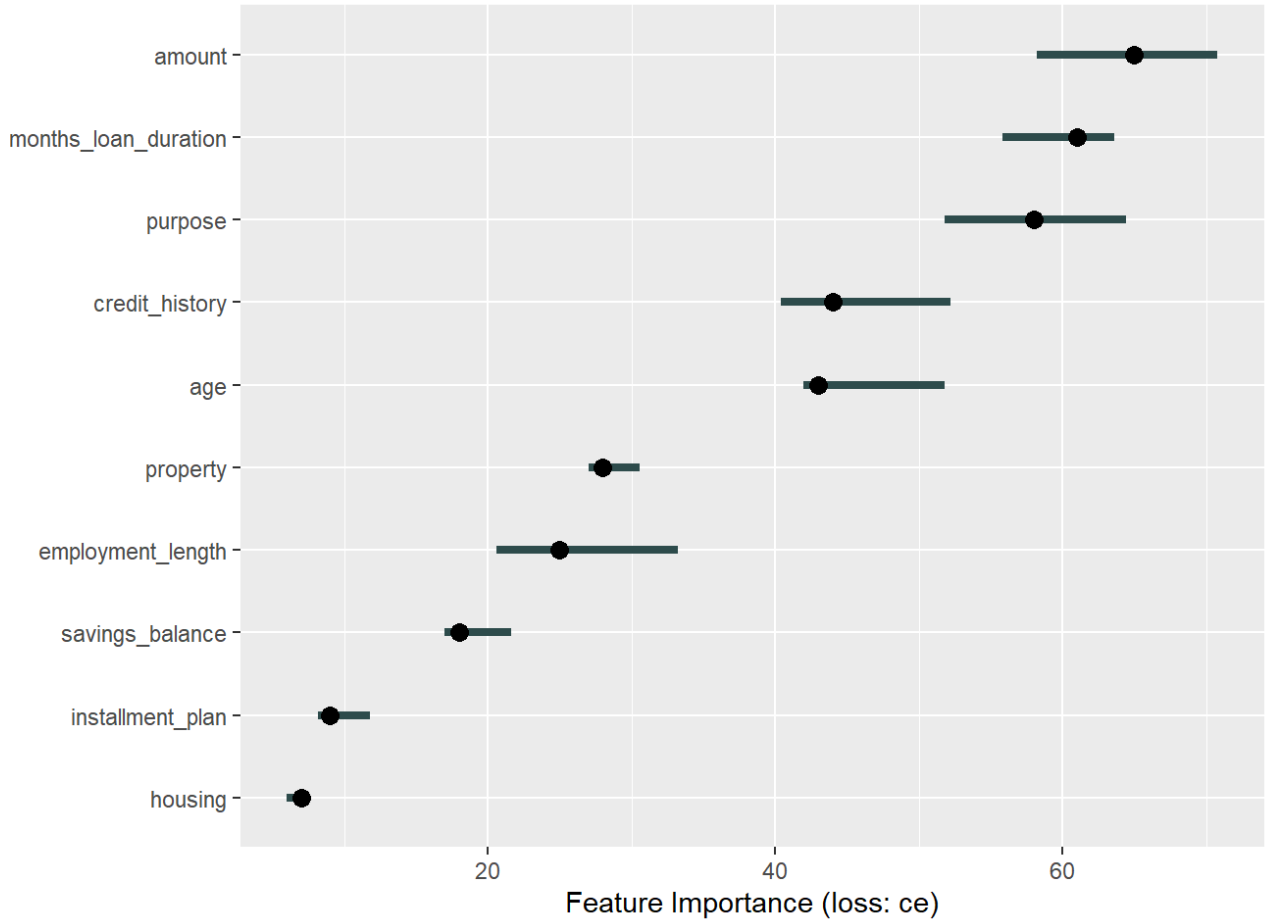
3.4 Cuarto árbol de decisión con Random forest

Vamos a crear un conjunto de árboles de decisión con Random Forest y observaremos si podemos mejorar el modelo anterior.

```
train.data <- as.data.frame(cbind(trainX,trainy))
colnames(train.data)[11] <- "DEFAULT"
rf <-  randomForest(DEFAULT ~ ., data = train.data, ntree = 50)
```

A continuación podemos visualizar la importancia de las variables. Este gráfico muestra la importancia relativa de cada variable en el modelo generado por randomForest.

```
X4 <- train.data[which(names(train.data) != "DEFAULT")]
predictor <- Predictor$new(rf, data = X4, y = train.data$DEFAULT)
imp <- FeatureImp$new(predictor, loss = "ce")
plot(imp)
```




```
predicted_model4 <- predict( rf, testX, type="class" )
precision4 <- sum(predicted_model4 == testy) / length(predicted_model4)
print(sprintf("La precisión del árbol es: %.4f",precision4))
```

```
## [1] "La precisión del árbol es: 0.6960"
```

```
mat_conf4<-table(testy, predicted_model4)
mat_conf4
```

```
##        predicted_model4
## testy   Bueno Malo
##   Bueno   160   24
##   Malo     59   30
```

```
sensibilidad4 <- (160)/(160+59)
print(sprintf("La sensibilidad del modelo es: %.4f",sensibilidad4))
```

```
## [1] "La sensibilidad del modelo es: 0.7306"
```

```
F4 <- 2*((precision4*sensibilidad4)/(precision4+sensibilidad4))
print(sprintf("El F-measure del modelo de Random Forest es: %.4f",F4))
```

```
## [1] "El F-measure del modelo de Random Forest es: 0.7129"
```

Observamos que este modelo no consigue mejorar el modelo creado anteriormente, presenta menor precisión, sensibilidad y por lo tanto F-measure.

4 Conclusiones

En primer lugar realizamos una tabla con los árboles creados y sus métricas.

```
# Crear los datos para el dataframe
nombre_modelo <- c("Primer árbol", "Variación primer árbol", "Segundo árbol", "Tercer ár
bol (Variación del paquete C5.0)", "Random forest")
precision <- c(precision1, precision1.1, precision2, precision3, precision4)
sensibilidad <- c(sensibilidad1, sensibilidad1.1, sensibilidad2,sensibilidad3,sensibilid
ad4)
fscore <- c(F1, F1.1, F2, F3, F4)

# Crear el dataframe
dataframe <- data.frame(
  "Precisión" = precision,
  "Sensibilidad" = sensibilidad,
  "F-score" = fscore
)

# Establecer los nombres de las filas
rownames(dataframe) <- nombre_modelo

# Mostrar el dataframe
print(dataframe)
```

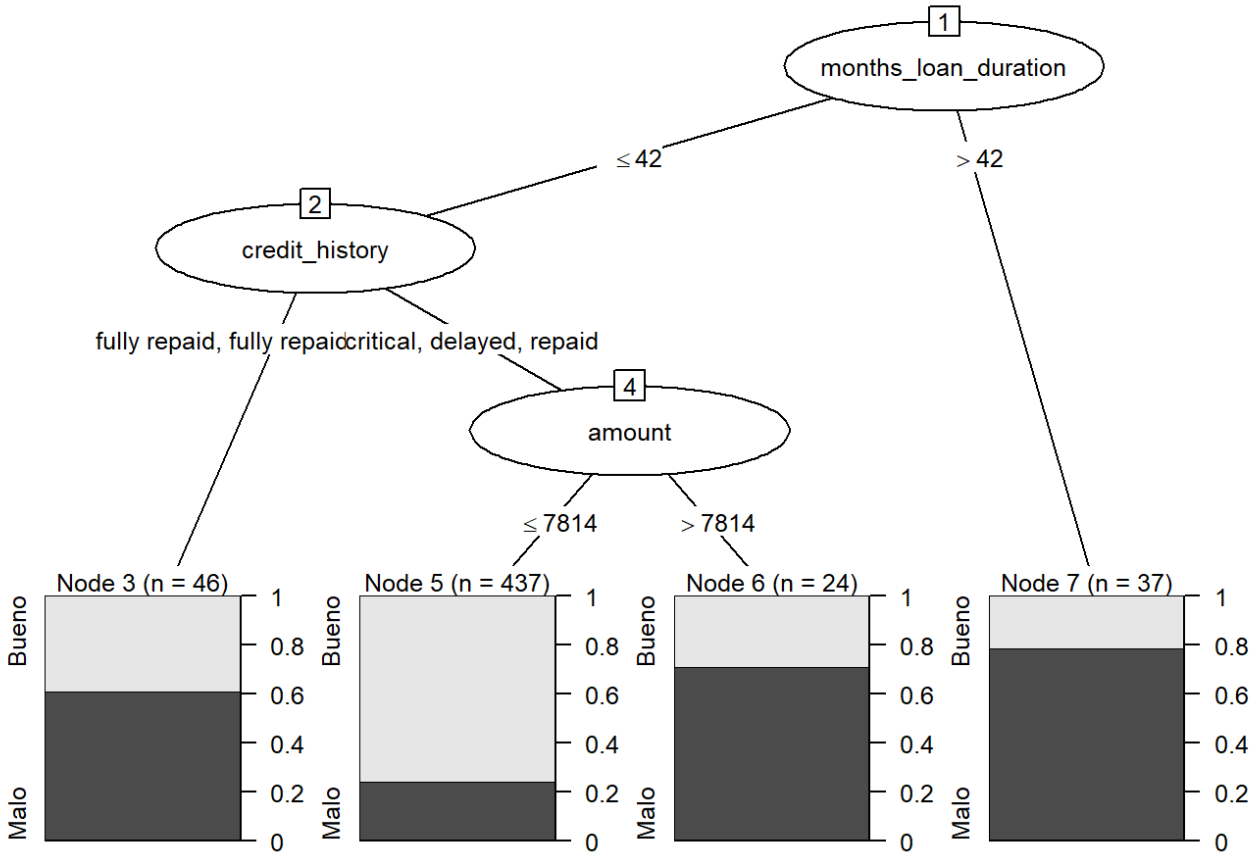
```
##                                       Precisión Sensibilidad   F.score
## Primer árbol                          0.7142857    0.7304348 0.7222700
## Variación primer árbol                0.6923077    0.7380952 0.7144686
## Segundo árbol                         0.6034483    0.6310680 0.6169492
## Tercer árbol (Variación del paquete C5.0) 0.7326007    0.7444934 0.7384992
## Random forest                         0.6959707    0.7305936 0.7128620
```

El modelo que ha dado mejores resultados es el tercer árbol de decisión que se ha realizado utilizando el algoritmo C5.0 implementado en el paquete C50 en R e indicando trials = 10, lo que significa que se generarán 10 árboles y se seleccionará el mejor.

Este modelo ha obtenido el F-measure más alto (0.738), obteniendo también los valores de precisión (0.732) y sensibilidad (0.744) más altos que el resto de modelos generados.

Cabe mencionar que el árbol generado es sencillo por lo que es práctico e intuitivo.

```
plot(modelo3,gp = gpar(fontsize = 9.5))
```



A pesar de haber obtenido las mejores métricas con este modelo se cree conveniente intentar ampliar el conjunto de datos para que las etiquetas de la variable “credit_history” tengan un número similar de registros cada una.