Ejercicio práctico desarrollo modelo de credit scoring

Daniel Fuentes

Mayo 2022

## R Markdown

EJERCICIO 1: EXTRACCIÓN Y VALIDACIÓN DE LOS DATOS

El primer paso en cualquier tipo de modelo de scoring o de rating es conocer bien los datos de los que dispones. Conocer bien el dataset sobre el que se trabaja puede determinar que un modelo sea útil y con buen poder predictivo o no.

En este caso tenemos una base de datos de préstamos obtenido en el siguiente enlace:

<https://www.lendingclub.com/info/download-data.action>

Se trata de una Fintech que concede préstamos a través de dinero de inversores. Cada solicitud es evaluada y catalogada en diferentes categorías de riesgo. Los inversores recibirán un mayor o menor interés en función de como de riesgosa sea la cartera de préstamos en los que invierta.

Además proveen de un excel con el significado de cada variable.

El dataset era de un tamaño considerable (600Mb) y ha sido reducido para que podáis trabajar más fácilmente y rápido a modo de ejemplo.

En la mayor parte de cursos de scoring podréis ver como se centran en lo que es el propio algoritmo: regresión logística, árboles de decisión, random forest…, lanzar este tipo de modelos es fácil pero no lo más importante, sobre todo desde el punto de vista práctico, donde un analista de crédito o usuario del modelo debe comprender el modelo y estar conforme o cómodo con él. Vamos a preparar y cargar todas las librerías necesarias para poder ejecutar todo el código que os mostramos a continuación:

# if(!is.element("plyr", installed.packages()[, 1])) install.packages("plyr")  
# if(!is.element("lattice", installed.packages()[, 1])) install.packages("lattice")  
# if(!is.element("plotly", installed.packages()[, 1])) install.packages("plotly")  
# if(!is.element("ggplot2", installed.packages()[, 1])) install.packages("ggplot2")  
# if(!is.element("Hmisc", installed.packages()[, 1])) install.packages("Hmisc")  
# if(!is.element("sqldf", installed.packages()[, 1])) install.packages("sqldf")  
# if(!is.element("zoo", installed.packages()[, 1])) install.packages("zoo")  
# if(!is.element("caTools", installed.packages()[, 1])) install.packages("caTools")  
# if(!is.element("RcmdrMisc", installed.packages()[, 1])) install.packages("RcmdrMisc")  
# if(!is.element("smbinning", installed.packages()[, 1])) install.packages("smbinning")  
# if(!is.element("DBI", installed.packages()[, 1])) install.packages("DBI")  
# if(!is.element("vcd", installed.packages()[, 1])) install.packages("vcd")  
# if(!is.element("ROCR", installed.packages()[, 1])) install.packages("ROCR")  
# if(!is.element("corrplot", installed.packages()[, 1])) install.packages("corrplot")  
# if(!is.element("partykit", installed.packages()[, 1])) install.packages("partykit")  
# if(!is.element("rpart", installed.packages()[, 1])) install.packages("rpart")  
# if(!is.element("stringr", installed.packages()[, 1])) install.packages("stringr")  
# if(!is.element("lubridate", installed.packages()[, 1])) install.packages("lubridate")  
# if(!is.element("plyr", installed.packages()[, 1])) install.packages("plyr")  
# if(!is.element("dplyr", installed.packages()[, 1])) install.packages("dplyr")  
# if(!is.element("leaps", installed.packages()[, 1])) install.packages("leaps")  
# if(!is.element("randomForest", installed.packages()[, 1])) install.packages("randomForest")  
# if(!is.element("fBasics", installed.packages()[, 1])) install.packages("fBasics")  
# if(!is.element("tabplot", installed.packages()[, 1])) install.packages("tabplot")  
# if(!is.element("dataMaid", installed.packages()[, 1])) install.packages("dataMaid")  
# if(!is.element("smbinning", installed.packages()[, 1])) install.packages("smbinning")  
# if(!is.element("RDCOMClient", installed.packages()[, 1])) install.packages('RDCOMClient', repos = 'http://www.stats.ox.ac.uk/pub/RWin/')  
# if(!is.element("ggcorrplot", installed.packages()[, 1])) install.packages("ggcorrplot")  
# if(!is.element("Amelia", installed.packages()[, 1])) install.packages("Amelia")  
# if(!is.element("ggplot", installed.packages()[, 1])) install.packages("ggplot")  
# if(!is.element("randomForest", installed.packages()[, 1])) install.packages("randomForest")  
# if(!is.element("ada", installed.packages()[, 1])) install.packages("ada")  
# if(!is.element("patchwork", installed.packages()[, 1])) install.packages("patchwork")  
# if(!is.element("DataExplorer", installed.packages()[, 1])) install.packages("DataExplorer ")  
  
  
  
library(smbinning)  
library(corrplot)  
library(Hmisc)  
library(sqldf)  
library(zoo)  
library(caTools)  
library(RcmdrMisc)  
library(ROCR)  
library(vcd)  
library(ggplot2)  
library(plotly)  
library(lattice)  
library(plyr)  
library(partykit)  
library(rpart)  
library(stringr)  
library(lubridate)  
library(plyr)  
library(dplyr)  
library(leaps)  
library(randomForest)  
library(fBasics)  
library(dataMaid)  
library(DescTools)  
library(ggcorrplot)  
library(Amelia)  
library(DataExplorer)  
library(ggplot2)  
library(lubridate)

Vamos a cargar la base de datos que hemos preparado y analizar todas las variables que tenemos disponibles:

# load("Database\_LoansAFI.RData")  
  
library(readxl)  
Loans <- read\_excel("C:/Users/Javier/Documents/masterAFI/20. Masterclass/06. Credit Scoring/Loans.xlsx")  
colnames(Loans)

## [1] "id" "loan\_amnt" "term"   
## [4] "int\_rate" "installment" "grade"   
## [7] "emp\_title" "emp\_length" "home\_ownership"   
## [10] "annual\_inc" "verification\_status" "issue\_d"   
## [13] "loan\_status" "desc" "purpose"   
## [16] "addr\_state" "dti" "delinq\_2yrs"   
## [19] "earliest\_cr\_line" "Fico\_score" "inq\_last\_6mths"   
## [22] "mths\_since\_last\_delinq" "mths\_since\_last\_record" "open\_acc"   
## [25] "revol\_bal" "revol\_util"

dim(Loans)

## [1] 39786 26

Como puede observarse tenemos 26 variables, hemos filtrado el dataset global también para que vieráis cuáles son las variables más comunes en un modelo de scoring. Para conocer que es cada una de ellas podáis consultar el excel diccionario que también os paso. De las que tenemos en este dataset, estas son las definiciones:

**id:** A unique LC assigned ID for the loan listing.  
**loan\_amnt:** The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value.  
**term:** The number of payments on the loan. Values are in months and can be either 36 or 60.  
**int\_rate:** Interest Rate on the loan  
**installment:** The monthly payment owed by the borrower if the loan originates. **grade:** LC assigned loan grade **emp\_title:** The job title supplied by the Borrower when applying for the loan.\* **emp\_length:** Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years. **home\_ownership:** The home ownership status provided by the borrower during registration or obtained from the credit report.Our values are: RENT, OWN, MORTGAGE, OTHER **annual\_inc:** The self-reported annual income provided by the borrower during registration. **verification\_status:** Indicates if income was verified by LC, not verified, or if the income source was verified **issue\_d:** The month which the loan was funded **loan\_status:** Current status of the loan **desc:** Loan description provided by the borrower **purpose:** A category provided by the borrower for the loan request. **addr\_state:** The state provided by the borrower in the loan application **dti:** A ratio calculated using the borrower’s total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower’s self-reported monthly income. **delinq\_2yrs:** The number of 30+ days past-due incidences of delinquency in the borrower’s credit file for the past 2 years **earliest\_cr\_line:** The month the borrower’s earliest reported credit line was opened \_\_**Fico\_score:**\_\_the borrower’s FICO score at loan origination belongs to. \_\_**inq\_last\_6mths:**\_\_The number of inquiries in past 6 months (excluding auto and mortgage inquiries) **mths\_since\_last\_delinq:** The number of months since the borrower’s last delinquency. **mths\_since\_last\_record:** The number of months since the last public record. **open\_acc:** The number of open credit lines in the borrower’s credit file. **revol\_bal:** Total credit revolving balance **revol\_util:** Revolving line utilization rate, or the amount of credit the borrower is

Analicemos la estructura de las variables del modelo y vamos estudiando poco a poco cada una de ellas:

## tibble [39,786 x 26] (S3: tbl\_df/tbl/data.frame)  
## $ id : num [1:39786] 1077501 1077430 1077175 1076863 1075358 ...  
## $ loan\_amnt : num [1:39786] 5000 2500 2400 10000 3000 ...  
## $ term : chr [1:39786] "36 months" "60 months" "36 months" "36 months" ...  
## $ int\_rate : chr [1:39786] "10.65%" "15.27%" "15.96%" "13.49%" ...  
## $ installment : num [1:39786] 162.9 59.8 84.3 339.3 67.8 ...  
## $ grade : chr [1:39786] "B" "C" "C" "C" ...  
## $ emp\_title : chr [1:39786] NA "Ryder" NA "AIR RESOURCES BOARD" ...  
## $ emp\_length : chr [1:39786] "10+ years" "< 1 year" "10+ years" "10+ years" ...  
## $ home\_ownership : chr [1:39786] "RENT" "RENT" "RENT" "RENT" ...  
## $ annual\_inc : num [1:39786] 24000 30000 12252 49200 80000 ...  
## $ verification\_status : chr [1:39786] "Verified" "Source Verified" "Not Verified" "Source Verified" ...  
## $ issue\_d : chr [1:39786] "11-Dec" "11-Dec" "11-Dec" "11-Dec" ...  
## $ loan\_status : chr [1:39786] "Fully Paid" "Default" "Fully Paid" "Fully Paid" ...  
## $ desc : chr [1:39786] "Borrower added on 12/22/11 > I need to upgrade my business technologies.<br>" "Borrower added on 12/22/11 > I plan to use this money to finance the motorcycle i am looking at. I plan to have"| \_\_truncated\_\_ NA "Borrower added on 12/21/11 > to pay for property tax (borrow from friend, need to pay back) & central A/C need "| \_\_truncated\_\_ ...  
## $ purpose : chr [1:39786] "credit\_card" "car" "small\_business" "other" ...  
## $ addr\_state : chr [1:39786] "AZ" "GA" "IL" "CA" ...  
## $ dti : num [1:39786] 27.65 1 8.72 20 17.94 ...  
## $ delinq\_2yrs : num [1:39786] 0 0 0 0 0 0 0 0 0 0 ...  
## $ earliest\_cr\_line : chr [1:39786] "Jan-85" "Apr-99" "1-Nov" "Feb-96" ...  
## $ Fico\_score : num [1:39786] 735 740 735 690 695 730 690 660 675 725 ...  
## $ inq\_last\_6mths : num [1:39786] 1 5 2 1 0 3 1 2 2 0 ...  
## $ mths\_since\_last\_delinq: num [1:39786] NA NA NA 35 38 NA NA NA NA NA ...  
## $ mths\_since\_last\_record: num [1:39786] NA NA NA NA NA NA NA NA NA NA ...  
## $ open\_acc : num [1:39786] 3 3 2 10 15 9 7 4 11 2 ...  
## $ revol\_bal : num [1:39786] 13648 1687 2956 5598 27783 ...  
## $ revol\_util : chr [1:39786] "83.70%" "9.40%" "98.50%" "21%" ...

Como podéis obsevar hay variables numéricas y categóricas y precisamente su tratamiento deberá ser realizado de manera diferente. Ya se dejan ver algunas variables que tienen algunos valores missing y podemos ver cuál es nuestra variable target o la que queremos predecir en nuestros préstamos: loan\_status.

Veamos los 5 primeros registros del dataset para seguir haciéndonos una idea de la información con la que contamos:

head(Loans,n=5)

## # A tibble: 5 x 26  
## id loan\_amnt term int\_rate installment grade emp\_title emp\_length  
## <dbl> <dbl> <chr> <chr> <dbl> <chr> <chr> <chr>   
## 1 1077501 5000 36 months 10.65% 163. B <NA> 10+ years   
## 2 1077430 2500 60 months 15.27% 59.8 C Ryder < 1 year   
## 3 1077175 2400 36 months 15.96% 84.3 C <NA> 10+ years   
## 4 1076863 10000 36 months 13.49% 339. C AIR RESOURC~ 10+ years   
## 5 1075358 3000 60 months 12.69% 67.8 B University ~ 1 year   
## # ... with 18 more variables: home\_ownership <chr>, annual\_inc <dbl>,  
## # verification\_status <chr>, issue\_d <chr>, loan\_status <chr>, desc <chr>,  
## # purpose <chr>, addr\_state <chr>, dti <dbl>, delinq\_2yrs <dbl>,  
## # earliest\_cr\_line <chr>, Fico\_score <dbl>, inq\_last\_6mths <dbl>,  
## # mths\_since\_last\_delinq <dbl>, mths\_since\_last\_record <dbl>, open\_acc <dbl>,  
## # revol\_bal <dbl>, revol\_util <chr>

Lo mismo podría hacerse para ver los últimos registros del dataset

tail(Loans,n=7)

## # A tibble: 7 x 26  
## id loan\_amnt term int\_rate installment grade emp\_title emp\_length  
## <dbl> <dbl> <chr> <chr> <dbl> <chr> <chr> <chr>   
## 1 92507 5000 36 months 7.43% 155. A Rush Univ Med~ 1 year   
## 2 92402 5000 36 months 8.70% 158. B A. F. Wolfers~ 5 years   
## 3 92187 2500 36 months 8.07% 78.4 A FiSite Resear~ 4 years   
## 4 90665 8500 36 months 10.28% 275. C Squarewave So~ 3 years   
## 5 90395 5000 36 months 8.07% 157. A <NA> < 1 year   
## 6 90376 5000 36 months 7.43% 155. A <NA> < 1 year   
## 7 87023 7500 36 months 13.75% 255. E Evergreen Cen~ < 1 year   
## # ... with 18 more variables: home\_ownership <chr>, annual\_inc <dbl>,  
## # verification\_status <chr>, issue\_d <chr>, loan\_status <chr>, desc <chr>,  
## # purpose <chr>, addr\_state <chr>, dti <dbl>, delinq\_2yrs <dbl>,  
## # earliest\_cr\_line <chr>, Fico\_score <dbl>, inq\_last\_6mths <dbl>,  
## # mths\_since\_last\_delinq <dbl>, mths\_since\_last\_record <dbl>, open\_acc <dbl>,  
## # revol\_bal <dbl>, revol\_util <chr>

Se tendrán que ir analizando variable a variable como se comportan, como se encuentran en el dataset y modifarlas si procede:

#Clases de cada variable  
Class<-as.data.frame(sapply(Loans, class))  
colnames(Class)<-"variable\_class"  
Class$variable\_name<-row.names(Class)  
row.names(Class)<-NULL  
  
Class

## variable\_class variable\_name  
## 1 numeric id  
## 2 numeric loan\_amnt  
## 3 character term  
## 4 character int\_rate  
## 5 numeric installment  
## 6 character grade  
## 7 character emp\_title  
## 8 character emp\_length  
## 9 character home\_ownership  
## 10 numeric annual\_inc  
## 11 character verification\_status  
## 12 character issue\_d  
## 13 character loan\_status  
## 14 character desc  
## 15 character purpose  
## 16 character addr\_state  
## 17 numeric dti  
## 18 numeric delinq\_2yrs  
## 19 character earliest\_cr\_line  
## 20 numeric Fico\_score  
## 21 numeric inq\_last\_6mths  
## 22 numeric mths\_since\_last\_delinq  
## 23 numeric mths\_since\_last\_record  
## 24 numeric open\_acc  
## 25 numeric revol\_bal  
## 26 character revol\_util

Algunas variables aparecen como de tipo integer, para evitar problemas posteriores las vamos a convertir en formato numéricas:

Class[Class$variable\_class=="integer",]

## [1] variable\_class variable\_name   
## <0 rows> (or 0-length row.names)

for (i in 1:ncol(Loans)){  
   
 if (class(Loans[,i])=="integer"){  
   
   
 Loans[,i]<-as.numeric(Loans[,i])  
   
 }  
   
}

## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado  
  
## Warning in if (class(Loans[, i]) == "integer") {: la condición tiene longitud >  
## 1 y sólo el primer elemento será usado

Vamos a ir analizando ahora variable a variable las variables categóricas y haciendo trasnformaciones si es necesario:

Class[Class$variable\_class=="character",]

## variable\_class variable\_name  
## 3 character term  
## 4 character int\_rate  
## 6 character grade  
## 7 character emp\_title  
## 8 character emp\_length  
## 9 character home\_ownership  
## 11 character verification\_status  
## 12 character issue\_d  
## 13 character loan\_status  
## 14 character desc  
## 15 character purpose  
## 16 character addr\_state  
## 19 character earliest\_cr\_line  
## 26 character revol\_util

Empecemos con la variable plazo del préstamo:

table(Loans$term)

##   
## 36 months 60 months   
## 29096 10690

Esta variable podría convertirse a numérica pero dado que solamente toma dos valores la vamos a mantener así. Veamos la siguiente variable: int\_rate (tipo de interés del préstamo):

head(Loans$int\_rate)

## [1] "10.65%" "15.27%" "15.96%" "13.49%" "12.69%" "7.90%"

La variable es numérica pero aparece como categórica, vamos a transformarla:

Loans$int\_rate<-as.numeric(gsub("%","",Loans$int\_rate))/100  
  
summary(Loans$int\_rate)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.0542 0.0925 0.1186 0.1203 0.1459 0.2459

Variable “grade” (Categoria de riesgo que asigna LendingClub a cada préstamo):

table(Loans$grade)

##   
## A B C D E F G   
## 10085 12035 8111 5325 2858 1054 318

La categoría A corresponde a préstamos de solicitantes con mejor perfil de pago o más solventes, siendo la G la categoría de préstamos más riesgosa:

Más variables que están disponibles: Descripción del puesto que ocupa el solicitante en la empresa:

head(table(Loans$emp\_title))

##   
## $260M '06 vintage technology venture capital firm   
## 1   
## (Collaborative) Abbott Nutrition Intl   
## 1   
## (self) Castleforte Group   
## 1   
## :coderow   
## 1   
## 1-800 Contacts   
## 1   
## 1 Cochran Hyundai   
## 1

print("La variable tiene ")

## [1] "La variable tiene "

(length(table(Loans$emp\_title)))

## [1] 28660

print("valores diferentes.")

## [1] "valores diferentes."

Variable “emp\_length” o antigüedad en el empleo:

table(Loans$emp\_length)

##   
## < 1 year 1 year 10+ years 2 years 3 years 4 years 5 years 6 years   
## 4590 3247 8899 4394 4098 3444 3286 2231   
## 7 years 8 years 9 years n/a   
## 1775 1485 1259 1078

Vamos a transformar la variable en numérica. Si el valor que toma es “< 1 year” pondremos un 0 y si es mayor que 10 se convertira en 10 años.

Loans$emp\_length<-gsub("years","",Loans$emp\_length)  
Loans$emp\_length<-gsub("year","",Loans$emp\_length)  
Loans$emp\_length<-gsub("< 1","0",Loans$emp\_length)  
Loans$emp\_length<-gsub("[+]","",Loans$emp\_length)  
Loans$emp\_length<-as.numeric(Loans$emp\_length)

## Warning: NAs introducidos por coerción

summary(Loans$emp\_length)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 0.000 2.000 4.000 4.974 9.000 10.000 1078

Más variables:

“home\_ownership”: Situación de la vivienda habitual

table(Loans$home\_ownership)

##   
## MORTGAGE OTHER OWN RENT   
## 17703 101 3064 18918

“verification\_status”: Si los ingresos declarados por el solicitante han sido verificados o no

table(Loans$verification\_status)

##   
## Not Verified Source Verified Verified   
## 16926 10016 12844

“purpose”: Destino del préstamo

table(Loans$purpose)

##   
## car credit\_card debt\_consolidation educational   
## 1551 5137 18676 325   
## home\_improvement house major\_purchase medical   
## 2985 382 2188 695   
## moving other renewable\_energy small\_business   
## 583 4001 103 1831   
## vacation wedding   
## 381 948

“addr\_state”: Estado de residencia del solicitante, los datos son de solicitudes en Estados Unidos.

table(Loans$addr\_state)

##   
## AK AL AR AZ CA CO CT DC DE FL GA HI IA ID IL IN   
## 81 452 246 882 7105 792 754 214 114 2872 1399 174 5 6 1525 9   
## KS KY LA MA MD ME MI MN MO MS MT NC NE NH NJ NM   
## 272 327 436 1344 1055 3 722 616 686 19 85 789 5 172 1855 190   
## NV NY OH OK OR PA RI SC SD TN TX UT VA VT WA WI   
## 498 3817 1226 300 451 1519 199 472 64 17 2734 259 1408 54 842 460   
## WV WY   
## 177 83

Con la variable “revol\_util” o porcentaje de las líneas de crédito que tiene utilizadas, ocurre lo mismo que con la variable tipo de interés. Convertimos de igual forma la variable en numérica:

head(Loans$revol\_util)

## [1] "83.70%" "9.40%" "98.50%" "21%" "53.90%" "28.30%"

Loans$revol\_util<-as.numeric(gsub("%","",Loans$revol\_util))/100  
  
summary(Loans$revol\_util)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 0.0000 0.2540 0.4930 0.4886 0.7240 0.9990 50

La variable “issue\_d” es la fecha en la que el préstamo fue concedido, como está en formato carácter, vamos a convertirla en formato fecha:

head(Loans$issue\_d)

## [1] "11-Dec" "11-Dec" "11-Dec" "11-Dec" "11-Dec" "11-Dec"

tail(Loans$issue\_d)

## [1] "7-Jul" "7-Jul" "7-Jul" "7-Jul" "7-Jul" "7-Jun"

unique(Loans$issue\_d)

## [1] "11-Dec" "11-Nov" "11-Oct" "11-Sep" "11-Aug" "11-Jul" "11-Jun" "11-May"  
## [9] "11-Apr" "11-Mar" "11-Feb" "11-Jan" "10-Dec" "10-Nov" "10-Oct" "10-Sep"  
## [17] "10-Aug" "10-Jul" "10-Jun" "10-May" "10-Apr" "10-Mar" "10-Feb" "10-Jan"  
## [25] "9-Dec" "9-Nov" "9-Oct" "9-Sep" "9-Aug" "9-Jul" "9-Jun" "9-May"   
## [33] "9-Apr" "9-Mar" "9-Feb" "9-Jan" "8-Dec" "8-Nov" "8-Oct" "8-Sep"   
## [41] "8-Aug" "8-Jul" "8-Jun" "8-May" "8-Apr" "8-Mar" "8-Feb" "8-Jan"   
## [49] "7-Dec" "7-Nov" "7-Oct" "7-Sep" "7-Aug" "7-Jul" "7-Jun"

Loans$issue\_d<-ymd(paste(Loans$issue\_d,"-01",sep=""))  
  
unique(Loans$issue\_d)

## [1] "2011-12-01" "2011-11-01" "2011-10-01" "2011-09-01" "2011-08-01"  
## [6] "2011-07-01" "2011-06-01" "2011-05-01" "2011-04-01" "2011-03-01"  
## [11] "2011-02-01" "2011-01-01" "2010-12-01" "2010-11-01" "2010-10-01"  
## [16] "2010-09-01" "2010-08-01" "2010-07-01" "2010-06-01" "2010-05-01"  
## [21] "2010-04-01" "2010-03-01" "2010-02-01" "2010-01-01" "2009-12-01"  
## [26] "2009-11-01" "2009-10-01" "2009-09-01" "2009-08-01" "2009-07-01"  
## [31] "2009-06-01" "2009-05-01" "2009-04-01" "2009-03-01" "2009-02-01"  
## [36] "2009-01-01" "2008-12-01" "2008-11-01" "2008-10-01" "2008-09-01"  
## [41] "2008-08-01" "2008-07-01" "2008-06-01" "2008-05-01" "2008-04-01"  
## [46] "2008-03-01" "2008-02-01" "2008-01-01" "2007-12-01" "2007-11-01"  
## [51] "2007-10-01" "2007-09-01" "2007-08-01" "2007-07-01" "2007-06-01"

class(Loans$issue\_d)

## [1] "Date"

También la variable “earliest\_cr\_line” o fecha en la que el solicitante abrió su primera cuenta se encuentra en formato texto. La transformamos de igual forma a formato fecha:

head(Loans$earliest\_cr\_line)

## [1] "Jan-85" "Apr-99" "1-Nov" "Feb-96" "Jan-96" "4-Nov"

Loans$earliest\_cr\_line<-dmy(paste("01-",Loans$earliest\_cr\_line,sep=""))

## Warning: 11612 failed to parse.

head(Loans$earliest\_cr\_line)

## [1] "1985-01-01" "1999-04-01" NA "1996-02-01" "1996-01-01"  
## [6] NA

save.image("Backup1.RData")

Con esta variable, vamos a calcular la antigüedad del cliente en la entidad, para ello restaremos las fechas de concesión de solicitud del préstamo con la fecha que acabamos de obtener. La variable antigüedad como cliente estará expresada en años:

Loans$ant\_cliente<-as.numeric(round((Loans$issue\_d-Loans$earliest\_cr\_line)/365,0))  
  
summary(Loans$ant\_cliente)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## -60.00 12.00 15.00 16.31 19.00 43.00 11612

table(Loans$ant\_cliente)

##   
## -60 -59 -58 -57 -56 -55 -54 -53 -52 -51 -48 -45 -44 -43 -40 -35   
## 2 4 10 15 13 10 9 8 8 5 1 1 1 1 1 1   
## 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22   
## 42 140 505 1445 2815 3015 2712 2395 2129 1980 1660 1292 1091 984 916 784   
## 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38   
## 650 582 470 417 362 276 226 176 163 181 123 122 85 61 80 53   
## 39 40 41 42 43   
## 50 37 44 20 1

Podemos ver que hay algunos registros que aparecen con un valor negativo, parece que se debe a algún tipo de error. Vemos también que la variable parece tener muchos valores no informados. Los valores negativos serán remplazados por un valor 0:

Loans$ant\_cliente<-ifelse(Loans$ant\_cliente<0,0,Loans$ant\_cliente)  
  
summary(Loans$ant\_cliente)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 0.00 12.00 15.00 16.49 19.00 43.00 11612

Vamos a crear otra variable relacionada cn esta: una marca que nos indique si el solicitante es cliente anteriormente o no:

Loans$flag\_cliente<-ifelse(Loans$ant\_cliente>0,1,0)  
  
Loans$flag\_cliente<-ifelse(is.na(Loans$flag\_cliente),0,1)  
  
summary(Loans$flag\_cliente)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.0000 0.0000 1.0000 0.7081 1.0000 1.0000

EJERCICIO 2: DEFINICIÓN DE PERFORMANCE Y MUESTRAS

Normalmente la variable target suele ser la marca de default, o lo que es lo mismo, una variable que toma dos valores: 0 si el cliente pagó su préstamo o 1 en el caso de que fuera un cliente moroso.

Se suele mirar si un cliente presento problemas de pago en un periodo de tiempo concreto, lo más común es marcar a un cliente como default si tuvo incumplimientos en las cuotas de sus préstamos en más de 90 días (definición regulatoria) durante un periodo de un año o superior.

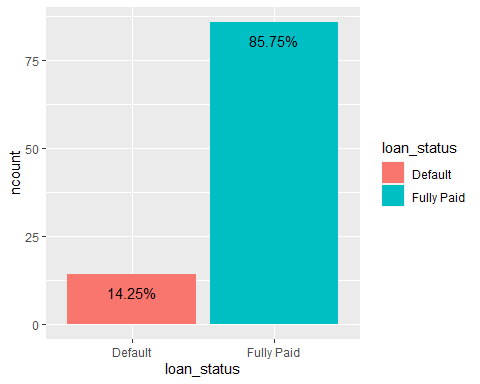
Vamos a ver qué valores toma nuestra variable target:

table(Loans$loan\_status)

##   
## Default Fully Paid   
## 5671 34115

Al parecer, de 39726 préstamos de la base de datos, 5.671 presentaron problemas de pago:

tmp = Loans %>% group\_by(loan\_status) %>% summarise(ncount = n())  
tmp$ncount = 100 \* tmp$ncount/nrow(Loans)  
tmp$ncount\_p = str\_c(round(tmp$ncount,2),"%")  
ggplot(tmp,aes(x=loan\_status,y=ncount,fill=loan\_status)) + geom\_bar(stat="identity") +  
geom\_text(aes(label=ncount\_p),vjust = 2)



Como vemos, esto supone que el 14.25% de los préstamos presentaron problemas de pago. Vamos a transformar los valores Default y FullyPaid en valores numéricos. Así si un préstamo presenta problemas será marcado como un 1, un 0 en caso contrario:

Loans$loan\_status<-ifelse(Loans$loan\_status=="Default",1,0)

Nos queda una última variable categórica: “desc”, que no hemos analizado. En esta variable se detalla en texto libre los comentarios que introduce el solicitante en la aplicación/web cuando solicitan el préstamo, aclarando el objeto del préstamo, más allá de la categoría genérica que queda recogida en la variable “purpose”:

head(Loans$desc)

## [1] "Borrower added on 12/22/11 > I need to upgrade my business technologies.<br>"   
## [2] "Borrower added on 12/22/11 > I plan to use this money to finance the motorcycle i am looking at. I plan to have it paid off as soon as possible/when i sell my old bike. I only need this money because the deal im looking at is to good to pass up.<br><br> Borrower added on 12/22/11 > I plan to use this money to finance the motorcycle i am looking at. I plan to have it paid off as soon as possible/when i sell my old bike.I only need this money because the deal im looking at is to good to pass up. I have finished college with an associates degree in business and its takingmeplaces<br>"  
## [3] NA   
## [4] "Borrower added on 12/21/11 > to pay for property tax (borrow from friend, need to pay back) & central A/C need to be replace. I'm very sorry to let my loan expired last time.<br>"   
## [5] "Borrower added on 12/21/11 > I plan on combining three large interest bills together and freeing up some extra each month to pay toward other bills. I've always been a good payor but have found myself needing to make adjustments to my budget due to a medical scare. My job is very stable, I love it.<br>"   
## [6] NA

Esta variable no será utilizada.

Respecto a las variables numéricas, podemos hacer un análisis similar observando que valores toman las variables:

#Clases de cada variable  
Class<-as.data.frame(sapply(Loans, class))  
colnames(Class)<-"variable\_class"  
Class$variable\_name<-row.names(Class)  
row.names(Class)<-NULL  
  
  
numeric\_vars<-Class[Class$variable\_class=="numeric","variable\_name"]  
numeric\_vars

## [1] "id" "loan\_amnt" "int\_rate"   
## [4] "installment" "emp\_length" "annual\_inc"   
## [7] "loan\_status" "dti" "delinq\_2yrs"   
## [10] "Fico\_score" "inq\_last\_6mths" "mths\_since\_last\_delinq"  
## [13] "mths\_since\_last\_record" "open\_acc" "revol\_bal"   
## [16] "revol\_util" "ant\_cliente" "flag\_cliente"

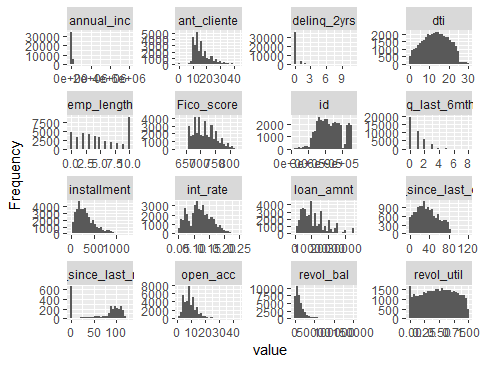
categorical\_vars<-Class[Class$variable\_class=="character","variable\_name"]

Con el paquete fBasics podemos obtener rápidamente descriptivos básicos de las variables numéricas:

library(fBasics)  
  
descriptives\_num<-as.data.frame(t(basicStats(Loans[,numeric\_vars])))  
  
descriptives\_num

## nobs NAs Minimum Maximum 1. Quartile  
## id 39786 0 54734.0000 1077501.0000 516351.7500  
## loan\_amnt 39786 0 500.0000 35000.0000 5500.0000  
## int\_rate 39786 0 0.0542 0.2459 0.0925  
## installment 39786 0 15.6900 1305.1900 167.0800  
## emp\_length 39786 1078 0.0000 10.0000 2.0000  
## annual\_inc 39786 0 4000.0000 6000000.0000 40500.0000  
## loan\_status 39786 0 0.0000 1.0000 0.0000  
## dti 39786 0 0.0000 29.9900 8.1800  
## delinq\_2yrs 39786 0 0.0000 11.0000 0.0000  
## Fico\_score 39786 0 625.0000 825.0000 685.0000  
## inq\_last\_6mths 39786 0 0.0000 8.0000 0.0000  
## mths\_since\_last\_delinq 39786 25727 0.0000 120.0000 18.0000  
## mths\_since\_last\_record 39786 36995 0.0000 129.0000 22.5000  
## open\_acc 39786 0 2.0000 44.0000 6.0000  
## revol\_bal 39786 0 0.0000 149588.0000 3704.2500  
## revol\_util 39786 50 0.0000 0.9990 0.2540  
## ant\_cliente 39786 11612 0.0000 43.0000 12.0000  
## flag\_cliente 39786 0 0.0000 1.0000 0.0000  
## 3. Quartile Mean Median Sum  
## id 837871.0000 6.833939e+05 666229.5000 2.718951e+10  
## loan\_amnt 15000.0000 1.123136e+04 10000.0000 4.468509e+08  
## int\_rate 0.1459 1.202790e-01 0.1186 4.785410e+03  
## installment 430.7800 3.247336e+02 280.6100 1.291985e+07  
## emp\_length 9.0000 4.974346e+00 4.0000 1.925470e+05  
## annual\_inc 82342.5000 6.897907e+04 59000.0000 2.744401e+09  
## loan\_status 0.0000 1.425380e-01 0.0000 5.671000e+03  
## dti 18.6000 1.331779e+01 13.4100 5.298618e+05  
## delinq\_2yrs 0.0000 1.465340e-01 0.0000 5.830000e+03  
## Fico\_score 740.0000 7.149980e+02 710.0000 2.844691e+07  
## inq\_last\_6mths 1.0000 8.690490e-01 1.0000 3.457600e+04  
## mths\_since\_last\_delinq 52.0000 3.590191e+01 34.0000 5.047450e+05  
## mths\_since\_last\_record 104.0000 6.974740e+01 90.0000 1.946650e+05  
## open\_acc 12.0000 9.294023e+00 9.0000 3.697720e+05  
## revol\_bal 17065.0000 1.339198e+04 8859.5000 5.328135e+08  
## revol\_util 0.7240 4.885810e-01 0.4930 1.941426e+04  
## ant\_cliente 19.0000 1.648626e+01 15.0000 4.644840e+05  
## flag\_cliente 1.0000 7.081390e-01 1.0000 2.817400e+04  
## SE Mean LCL Mean UCL Mean Variance  
## id 1056.214156 6.813237e+05 6.854642e+05 4.438480e+10  
## loan\_amnt 37.422955 1.115801e+04 1.130471e+04 5.571940e+07  
## int\_rate 0.000187 1.199120e-01 1.206450e-01 1.389000e-03  
## installment 1.047422 3.226807e+02 3.267866e+02 4.364891e+04  
## emp\_length 0.018110 4.938851e+00 5.009842e+00 1.269469e+01  
## annual\_inc 319.669435 6.835251e+04 6.960563e+04 4.065674e+09  
## loan\_status 0.001753 1.391020e-01 1.459730e-01 1.222240e-01  
## dti 0.033481 1.325217e+01 1.338342e+01 4.459969e+01  
## delinq\_2yrs 0.002466 1.417010e-01 1.513670e-01 2.418930e-01  
## Fico\_score 0.179685 7.146458e+02 7.153502e+02 1.284554e+03  
## inq\_last\_6mths 0.005365 8.585340e-01 8.795640e-01 1.145048e+00  
## mths\_since\_last\_delinq 0.185694 3.553793e+01 3.626590e+01 4.847877e+02  
## mths\_since\_last\_record 0.829172 6.812155e+01 7.137325e+01 1.918884e+03  
## open\_acc 0.022059 9.250787e+00 9.337259e+00 1.935997e+01  
## revol\_bal 79.686623 1.323580e+04 1.354817e+04 2.526394e+08  
## revol\_util 0.001422 4.857950e-01 4.913670e-01 8.029600e-02  
## ant\_cliente 0.035715 1.641626e+01 1.655627e+01 3.593761e+01  
## flag\_cliente 0.002279 7.036710e-01 7.126060e-01 2.066840e-01  
## Stdev Skewness Kurtosis  
## id 2.106770e+05 0.077206 -0.730150  
## loan\_amnt 7.464543e+03 1.057785 0.762701  
## int\_rate 3.727500e-02 0.292948 -0.449266  
## installment 2.089232e+02 1.126760 1.240869  
## emp\_length 3.562961e+00 0.200407 -1.365811  
## annual\_inc 6.376263e+04 30.938998 2302.814060  
## loan\_status 3.496050e-01 2.044899 2.181665  
## dti 6.678300e+00 -0.027759 -0.850982  
## delinq\_2yrs 4.918260e-01 5.019368 39.360725  
## Fico\_score 3.584068e+01 0.464756 -0.558091  
## inq\_last\_6mths 1.070069e+00 1.389730 2.558412  
## mths\_since\_last\_delinq 2.201789e+01 0.306403 -0.842904  
## mths\_since\_last\_record 4.380507e+01 -0.718537 -1.154322  
## open\_acc 4.399997e+00 1.003384 1.675974  
## revol\_bal 1.589464e+04 3.189163 14.870777  
## revol\_util 2.833660e-01 -0.034928 -1.105870  
## ant\_cliente 5.994798e+00 1.267477 1.912245  
## flag\_cliente 4.546250e-01 -0.915627 -1.161656

plot\_histogram(Loans[,numeric\_vars])



Una vez tenemos todas las variables transformadas como queremos, podemos realizar un resumen de todas ellas:

describe(Loans[,c(numeric\_vars,categorical\_vars)])

## Loans[, c(numeric\_vars, categorical\_vars)]   
##   
## 26 Variables 39786 Observations  
## --------------------------------------------------------------------------------  
## id   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 39786 1 683394 240904 372506 427812   
## .25 .50 .75 .90 .95   
## 516352 666230 837871 1007095 1039975   
##   
## lowest : 54734 55742 57245 57416 58915  
## highest: 1075358 1076863 1077175 1077430 1077501  
## --------------------------------------------------------------------------------  
## loan\_amnt   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 885 0.999 11231 8109 2400 3200   
## .25 .50 .75 .90 .95   
## 5500 10000 15000 22000 25000   
##   
## lowest : 500 700 725 750 800, highest: 34475 34525 34675 34800 35000  
## --------------------------------------------------------------------------------  
## int\_rate   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 371 1 0.1203 0.04241 0.0617 0.0714   
## .25 .50 .75 .90 .95   
## 0.0925 0.1186 0.1459 0.1689 0.1862   
##   
## lowest : 0.0542 0.0579 0.0599 0.0600 0.0603, highest: 0.2359 0.2391 0.2411 0.2440 0.2459  
## --------------------------------------------------------------------------------  
## installment   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 15405 1 324.7 226.1 71.23 99.50   
## .25 .50 .75 .90 .95   
## 167.08 280.61 430.78 623.10 763.75   
##   
## lowest : 15.69 16.08 16.25 16.31 16.47  
## highest: 1283.50 1288.10 1295.21 1302.69 1305.19  
## --------------------------------------------------------------------------------  
## emp\_length   
## n missing distinct Info Mean Gmd .05 .10   
## 38708 1078 11 0.981 4.974 4.048 0 0   
## .25 .50 .75 .90 .95   
## 2 4 9 10 10   
##   
## lowest : 0 1 2 3 4, highest: 6 7 8 9 10  
##   
## Value 0 1 2 3 4 5 6 7 8 9 10  
## Frequency 4590 3247 4394 4098 3444 3286 2231 1775 1485 1259 8899  
## Proportion 0.119 0.084 0.114 0.106 0.089 0.085 0.058 0.046 0.038 0.033 0.230  
## --------------------------------------------------------------------------------  
## annual\_inc   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 5323 1 68979 43352 24000 30000   
## .25 .50 .75 .90 .95   
## 40500 59000 82343 116000 142000   
##   
## lowest : 4000 4080 4200 4800 4888  
## highest: 1782000 1900000 2039784 3900000 6000000  
## --------------------------------------------------------------------------------  
## loan\_status   
## n missing distinct Info Sum Mean Gmd   
## 39786 0 2 0.367 5671 0.1425 0.2444   
##   
## --------------------------------------------------------------------------------  
## dti   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 2868 1 13.32 7.678 2.13 4.09   
## .25 .50 .75 .90 .95   
## 8.18 13.41 18.60 22.34 23.84   
##   
## lowest : 0.00 0.01 0.02 0.03 0.04, highest: 29.89 29.92 29.93 29.95 29.99  
## --------------------------------------------------------------------------------  
## delinq\_2yrs   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 11 0.291 0.1465 0.2681 0 0   
## .25 .50 .75 .90 .95   
## 0 0 0 1 1   
##   
## lowest : 0 1 2 3 4, highest: 6 7 8 9 11  
##   
## Value 0 1 2 3 4 5 6 7 8 9 11  
## Frequency 35466 3309 688 221 62 22 10 4 2 1 1  
## Proportion 0.891 0.083 0.017 0.006 0.002 0.001 0.000 0.000 0.000 0.000 0.000  
## --------------------------------------------------------------------------------  
## Fico\_score   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 36 0.998 715 40.71 665 670   
## .25 .50 .75 .90 .95   
## 685 710 740 765 780   
##   
## lowest : 625 630 660 665 670, highest: 805 810 815 820 825  
## --------------------------------------------------------------------------------  
## inq\_last\_6mths   
## n missing distinct Info Mean Gmd   
## 39786 0 9 0.861 0.869 1.079   
##   
## lowest : 0 1 2 3 4, highest: 4 5 6 7 8  
##   
## Value 0 1 2 3 4 5 6 7 8  
## Frequency 19337 10986 5824 3053 326 146 64 35 15  
## Proportion 0.486 0.276 0.146 0.077 0.008 0.004 0.002 0.001 0.000  
## --------------------------------------------------------------------------------  
## mths\_since\_last\_delinq   
## n missing distinct Info Mean Gmd .05 .10   
## 14059 25727 95 1 35.9 25.22 3 8   
## .25 .50 .75 .90 .95   
## 18 34 52 69 75   
##   
## lowest : 0 1 2 3 4, highest: 103 106 107 115 120  
## --------------------------------------------------------------------------------  
## mths\_since\_last\_record   
## n missing distinct Info Mean Gmd .05 .10   
## 2791 36995 111 0.986 69.75 46.92 0.0 0.0   
## .25 .50 .75 .90 .95   
## 22.5 90.0 104.0 113.0 115.0   
##   
## lowest : 0 5 6 7 11, highest: 117 118 119 120 129  
## --------------------------------------------------------------------------------  
## open\_acc   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 40 0.994 9.294 4.802 3 4   
## .25 .50 .75 .90 .95   
## 6 9 12 15 17   
##   
## lowest : 2 3 4 5 6, highest: 38 39 41 42 44  
## --------------------------------------------------------------------------------  
## revol\_bal   
## n missing distinct Info Mean Gmd .05 .10   
## 39786 0 21738 1 13392 14205 321.2 1117.0   
## .25 .50 .75 .90 .95   
## 3704.2 8859.5 17065.0 29168.0 41677.5   
##   
## lowest : 0 1 2 3 4, highest: 148804 148829 149000 149527 149588  
## --------------------------------------------------------------------------------  
## revol\_util   
## n missing distinct Info Mean Gmd .05 .10   
## 39736 50 1089 1 0.4886 0.327 0.0270 0.0854   
## .25 .50 .75 .90 .95   
## 0.2540 0.4930 0.7240 0.8790 0.9360   
##   
## lowest : 0.0000 0.0001 0.0003 0.0004 0.0005, highest: 0.9950 0.9960 0.9970 0.9980 0.9990  
## --------------------------------------------------------------------------------  
## ant\_cliente   
## n missing distinct Info Mean Gmd .05 .10   
## 28174 11612 38 0.995 16.49 6.323 10 11   
## .25 .50 .75 .90 .95   
## 12 15 19 25 29   
##   
## lowest : 0 7 8 9 10, highest: 39 40 41 42 43  
## --------------------------------------------------------------------------------  
## flag\_cliente   
## n missing distinct Info Sum Mean Gmd   
## 39786 0 2 0.62 28174 0.7081 0.4134   
##   
## --------------------------------------------------------------------------------  
## term   
## n missing distinct   
## 39786 0 2   
##   
## Value 36 months 60 months  
## Frequency 29096 10690  
## Proportion 0.731 0.269  
## --------------------------------------------------------------------------------  
## grade   
## n missing distinct   
## 39786 0 7   
##   
## lowest : A B C D E, highest: C D E F G  
##   
## Value A B C D E F G  
## Frequency 10085 12035 8111 5325 2858 1054 318  
## Proportion 0.253 0.302 0.204 0.134 0.072 0.026 0.008  
## --------------------------------------------------------------------------------  
## emp\_title   
## n missing distinct   
## 37325 2461 28660   
##   
## lowest : $260M '06 vintage technology venture capital firm (Collaborative) Abbott Nutrition Intl (self) Castleforte Group :coderow 1-800 Contacts   
## highest: Zwicker and Associates P.C. Zwiebel Center for Plastic Surger Zwingli United Church of Christ Zylog Systems Ltd Zynga   
## --------------------------------------------------------------------------------  
## home\_ownership   
## n missing distinct   
## 39786 0 4   
##   
## Value MORTGAGE OTHER OWN RENT  
## Frequency 17703 101 3064 18918  
## Proportion 0.445 0.003 0.077 0.475  
## --------------------------------------------------------------------------------  
## verification\_status   
## n missing distinct   
## 39786 0 3   
##   
## Value Not Verified Source Verified Verified  
## Frequency 16926 10016 12844  
## Proportion 0.425 0.252 0.323  
## --------------------------------------------------------------------------------  
## desc   
## n missing distinct   
## 26610 13176 26562   
##   
## lowest : - Pay off Dell Financial: $ 1300.00 - Pay off IRS for 2005: $ 1400.00 - Pay off Mac Comp : $ 1700.00 - Pay off Bill Me Later : $ 1000.00 - Pay off WAMU card : $ 1000.00 - Pay off Becu Visa : $ 1000.00 -To pay off loans and Credit Card at 17-25% interest -Daughter College tuition and books -Bring VW BUS 1970 and House Good storage in NM. Storage not covered by US Army when assign to Oversee duty. -Replace Matters for better sleeps "Hello, After graduating from college, I took an opportunity to start my own business operating a self-funded music publishing and recording company. And during the 5 years of my success in the public eye, most likely due to my lack of experience and preparedness, financially I made a few errors in judgment. Since then I have been doing everything I can to improve my financial situation: I got a very good job at the University of California, San Diego managing several projects in Medical Education (4 years currently); I???ve been pushing aggressively for promotions at work (about 1 month away from the latest approval); I am currently in the pursuit of a second job; I have been and continue to sell items of value in order to supplement any income; To save money on gas, I bike to work on average 1-2 times per week (approximately 50 miles round trip); and most importantly, I have turned music into a more realistic and financially stable hobby. Below is the detail of the debt that I am hoping to consolidate. I plan on eliminating 3 of the 4 cards. In the years that I have had this debt, I have been late on payments maybe once or twice (occasions where I overlooked the date). Also, a little less than one-third of this debt was built from the assistance of pursuing my bachelor???s degree. \* indicates the credit I intend on eliminating with this assistance Card 1 $3902.38 (%15.24) \* Card 2 $3955.99 (%17.90) \*Card 3 $13,475.70 (%24.50) \*Card 4 $4777.13 (%27.74) I appreciate any help that is out there and look forward to getting out of this hole so I can finally be able to focus on moving forward with building a family and career. " Borrower added on 07/03/10 > With the approved loan amount updated to $15,000, I plan on eliminating and decreasing the debt on cards #3 and 4<br/> "I bought a duplex in june. The only thing left to renovate is the roof and redo the sidewalk and drive way. I am secure in my job, never had any trouble with debts.""I bought a duplex in june. The only thing left to renovate is the roof and redo the sidewalk and drive way. I am secure in my job, never had any trouble with debts." Borrower added on 12/10/11 > Loan was relisted because my employer held verification of employment for me to sign but didnt contact me.<br> "My intentions are to purchase a engagement ring, hopefully in time for the holidays. I want to thank, in advance any and all who consider funding. My job is extremely stable, and I have a steady contractually binding income. My risk of layoff and or termination is extremely minuscule. I am also due for a substantial contractually guaranteed pay increase. I also have a second job with the luxury of being able to do as much or as little as I want, this does not interfere with my primary source of income due to the second job only allowing me to schedule on days that I have off. My monthly expenses will be substantially lower due to my increase in pay. My monthly rent is extremely low."   
## highest: www.music-housecalls.com--The Amadeus Home Music School will be using the loan to expand our Home Music Lesson business which currently has 250 instructors in 17 states. The expansion will be to add Ontario and other states within the United States of America. This will assist us in paying our first few months of Google charges. With Google we have generated a 5 to 1 return. Our web presence is at the top of Google, Yahoo and MSN in the sponsored links. Please take a look at our work and consider funding our expansion. Yes I have a line of credit that the interest rate just went up to 20.99 so I am looking to refi for a lower rate. Young couple just married. Own home but have two car loans and wife has school loans. Husband never had credit card debt. Was lucky enough that parents paid school college tuition. Husband is Senior Finance Officer at large denominational healthcare facility chain. Wife is an Electronic Medical Record Intermediate Analyst and BSN RN by training. Wife accumulated $15000 in credit card debt during school and throughout her twenties. Attempting to consolidate this debt into a more manageable rate and not just open up another credit card line. Young professional who needs to pay down debt acquired from college. Zensah Performance Apparel is a start-up, and we are now seeking to expand our product line as we enter the midwya point in our 3rd year! We currently sell through our website at www.zensah.com and directly to professional teams. Type ZENSAH in YOU TUBE, and you will see one of the most famous athletes in the world who sports ZENSAH! Zensah has create a proprietary moisture wicking fabric that doesn't carry odor and is not coarse. Unlike Under Armour and Nike you won't chafe with our fabric. We want to try Lending Club even though we are well capitalized. If this works out we thinks it could be a great way to allow our fans helps us grow even more! I encourage you to visit our website to learn more. Thanks for your interest. TEAM ZENSAH   
## --------------------------------------------------------------------------------  
## purpose   
## n missing distinct   
## 39786 0 14   
##   
## lowest : car credit\_card debt\_consolidation educational home\_improvement   
## highest: other renewable\_energy small\_business vacation wedding   
## --------------------------------------------------------------------------------  
## addr\_state   
## n missing distinct   
## 39786 0 50   
##   
## lowest : AK AL AR AZ CA, highest: VT WA WI WV WY  
## --------------------------------------------------------------------------------

Un mejor análisis puede ser llevado a cabo y además guardado en un documento Word:

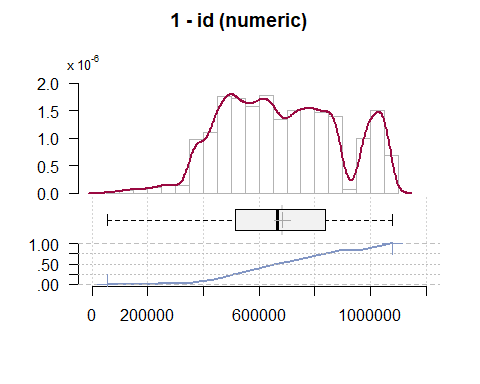
wrd<-GetNewWrd()

## Loading required namespace: RDCOMClient

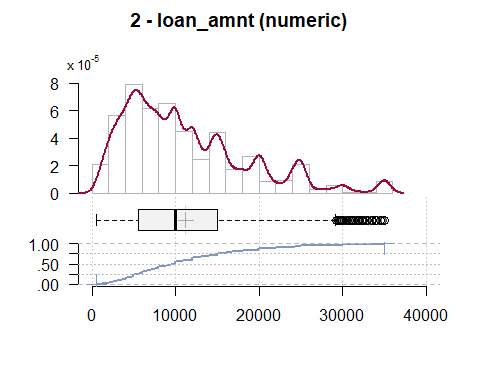
## Warning in GetCOMAppHandle("Word.Application", option = "lastWord",  
## existing = FALSE, : RDCOMClient is not available. To install it use:  
## install.packages('RDCOMClient', repos = 'http://www.stats.ox.ac.uk/pub/RWin/')

Desc(Loans[,c(numeric\_vars,categorical\_vars)],plotit = T,wrd=wrd)

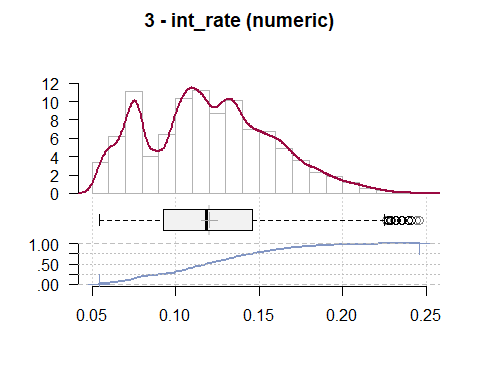
## ------------------------------------------------------------------------------   
## Describe Loans[, c(numeric\_vars, categorical\_vars)] (tbl\_df, tbl, data.frame):  
##   
## data frame: 39786 obs. of 26 variables  
## 1016 complete cases (2.6%)  
##   
## Nr ColName Class NAs Levels  
## 1 id numeric .   
## 2 loan\_amnt numeric .   
## 3 int\_rate numeric .   
## 4 installment numeric .   
## 5 emp\_length numeric 1078 (2.7%)   
## 6 annual\_inc numeric .   
## 7 loan\_status numeric .   
## 8 dti numeric .   
## 9 delinq\_2yrs numeric .   
## 10 Fico\_score numeric .   
## 11 inq\_last\_6mths numeric .   
## 12 mths\_since\_last\_delinq numeric 25727 (64.7%)   
## 13 mths\_since\_last\_record numeric 36995 (93.0%)   
## 14 open\_acc numeric .   
## 15 revol\_bal numeric .   
## 16 revol\_util numeric 50 (0.1%)   
## 17 ant\_cliente numeric 11612 (29.2%)   
## 18 flag\_cliente numeric .   
## 19 term character .   
## 20 grade character .   
## 21 emp\_title character 2461 (6.2%)   
## 22 home\_ownership character .   
## 23 verification\_status character .   
## 24 desc character 13176 (33.1%)   
## 25 purpose character .   
## 26 addr\_state character .   
##   
##   
## ------------------------------------------------------------------------------   
## 1 - id (numeric)  
##   
## length n NAs unique 0s mean'  
## 39'786 39'786 0 = n 0 683'393.95  
## 100.0% 0.0% 0.0%   
##   
## .05 .10 .25 median .75 .90  
## 372'506.25 427'812.00 516'351.75 666'229.50 837'871.00 1'007'094.50  
##   
## range sd vcoef mad IQR skew  
## 1'022'767.00 210'677.00 0.31 237'147.80 321'519.25 0.08  
##   
## meanCI  
## 681'323.74  
## 685'464.15  
##   
## .95  
## 1'039'974.75  
##   
## kurt  
## -0.73  
##   
## lowest : 54'734.0, 55'742.0, 57'245.0, 57'416.0, 58'915.0  
## highest: 1'075'358.0, 1'076'863.0, 1'077'175.0, 1'077'430.0, 1'077'501.0  
##   
## ' 95%-CI (classic)



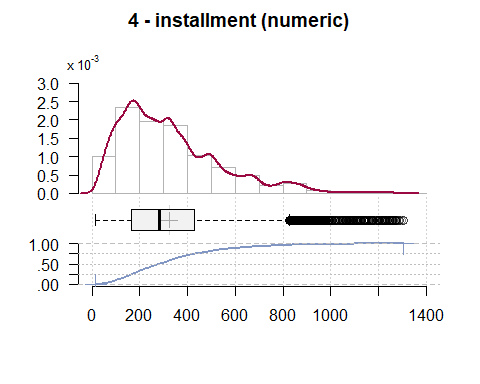
## ------------------------------------------------------------------------------   
## 2 - loan\_amnt (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 885 0 11'231.36 11'158.01  
## 100.0% 0.0% 0.0% 11'304.71  
##   
## .05 .10 .25 median .75 .90 .95  
## 2'400.00 3'200.00 5'500.00 10'000.00 15'000.00 22'000.00 25'000.00  
##   
## range sd vcoef mad IQR skew kurt  
## 34'500.00 7'464.54 0.66 7'413.00 9'500.00 1.06 0.76  
##   
## lowest : 500.0 (5), 700.0, 725.0, 750.0, 800.0  
## highest: 34'475.0 (5), 34'525.0, 34'675.0, 34'800.0 (2), 35'000.0 (685)  
##   
## heap(?): remarkable frequency (7.1%) for the mode(s) (= 10000)  
##   
## ' 95%-CI (classic)



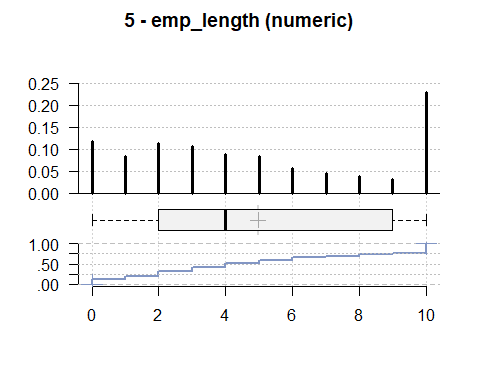
## ------------------------------------------------------------------------------   
## 3 - int\_rate (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 371 0 0.1203 0.1199  
## 100.0% 0.0% 0.0% 0.1206  
##   
## .05 .10 .25 median .75 .90 .95  
## 0.0617 0.0714 0.0925 0.1186 0.1459 0.1689 0.1862  
##   
## range sd vcoef mad IQR skew kurt  
## 0.1917 0.0373 0.3099 0.0397 0.0534 0.2929 -0.4493  
##   
## lowest : 0.0542 (573), 0.0579 (410), 0.0599 (347), 0.06 (18), 0.0603 (447)  
## highest: 0.2359 (4), 0.2391 (11), 0.2411 (3), 0.244, 0.2459  
##   
## ' 95%-CI (classic)



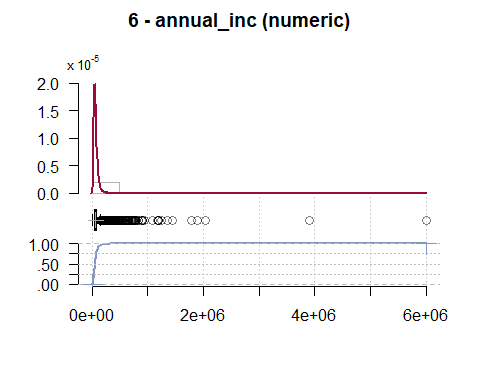
## ------------------------------------------------------------------------------   
## 4 - installment (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 15'405 0 324.7336 322.6807  
## 100.0% 0.0% 0.0% 326.7866  
##   
## .05 .10 .25 median .75 .90 .95  
## 71.2350 99.5000 167.0800 280.6100 430.7800 623.0950 763.7525  
##   
## range sd vcoef mad IQR skew kurt  
## 1'289.5000 208.9232 0.6434 182.9677 263.7000 1.1268 1.2409  
##   
## lowest : 15.69, 16.08, 16.25, 16.31, 16.47  
## highest: 1'283.5000, 1'288.1000 (2), 1'295.2100, 1'302.6900, 1'305.1900  
##   
## ' 95%-CI (classic)



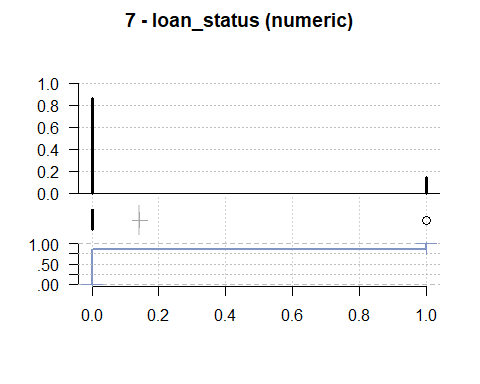
## ------------------------------------------------------------------------------   
## 5 - emp\_length (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 38'708 1'078 11 4'590 4.97 4.94  
## 97.3% 2.7% 11.5% 5.01  
##   
## .05 .10 .25 median .75 .90 .95  
## 0.00 0.00 2.00 4.00 9.00 10.00 10.00  
##   
## range sd vcoef mad IQR skew kurt  
## 10.00 3.56 0.72 4.45 7.00 0.20 -1.37  
##   
##   
## value freq perc cumfreq cumperc  
## 1 0 4'590 11.9% 4'590 11.9%  
## 2 1 3'247 8.4% 7'837 20.2%  
## 3 2 4'394 11.4% 12'231 31.6%  
## 4 3 4'098 10.6% 16'329 42.2%  
## 5 4 3'444 8.9% 19'773 51.1%  
## 6 5 3'286 8.5% 23'059 59.6%  
## 7 6 2'231 5.8% 25'290 65.3%  
## 8 7 1'775 4.6% 27'065 69.9%  
## 9 8 1'485 3.8% 28'550 73.8%  
## 10 9 1'259 3.3% 29'809 77.0%  
## 11 10 8'899 23.0% 38'708 100.0%  
##   
## ' 95%-CI (classic)



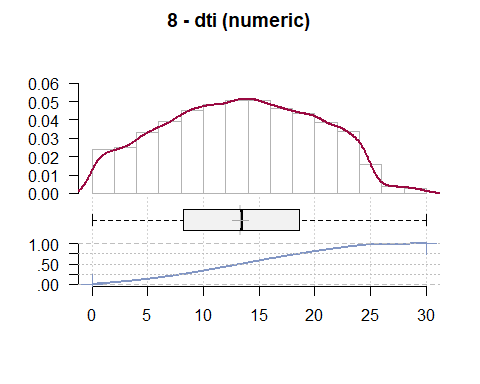
## ------------------------------------------------------------------------------   
## 6 - annual\_inc (numeric)  
##   
## length n NAs unique 0s mean'  
## 39'786 39'786 0 5'323 0 68'979.07  
## 100.0% 0.0% 0.0%   
##   
## .05 .10 .25 median .75 .90  
## 24'000.00 30'000.00 40'500.00 59'000.00 82'342.50 116'000.00  
##   
## range sd vcoef mad IQR skew  
## 5'996'000.00 63'762.63 0.92 29'652.00 41'842.50 30.94  
##   
## meanCI  
## 68'352.51  
## 69'605.63  
##   
## .95  
## 142'000.00  
##   
## kurt  
## 2'302.81  
##   
## lowest : 4'000.0, 4'080.0, 4'200.0 (2), 4'800.0 (4), 4'888.0  
## highest: 1'782'000.0, 1'900'000.0, 2'039'784.0, 3'900'000.0, 6'000'000.0  
##   
## ' 95%-CI (classic)



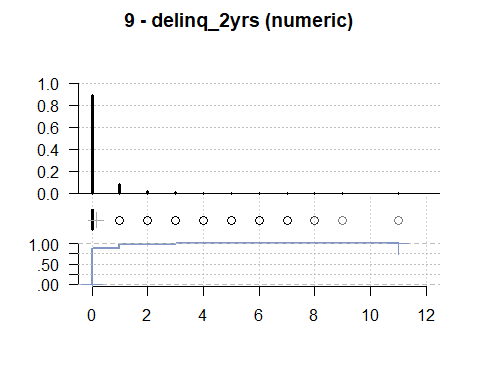
## ------------------------------------------------------------------------------   
## 7 - loan\_status (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 2 34'115 0.14 0.14  
## 100.0% 0.0% 85.7% 0.15  
##   
## .05 .10 .25 median .75 .90 .95  
## 0.00 0.00 0.00 0.00 0.00 1.00 1.00  
##   
## range sd vcoef mad IQR skew kurt  
## 1.00 0.35 2.45 0.00 0.00 2.04 2.18  
##   
##   
## value freq perc cumfreq cumperc  
## 1 0 34'115 85.7% 34'115 85.7%  
## 2 1 5'671 14.3% 39'786 100.0%  
##   
## ' 95%-CI (classic)



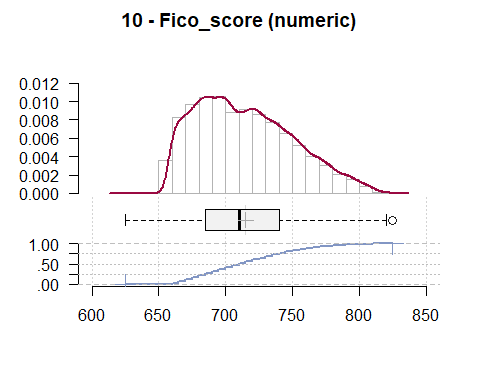
## ------------------------------------------------------------------------------   
## 8 - dti (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 2'868 184 13.318 13.252  
## 100.0% 0.0% 0.5% 13.383  
##   
## .05 .10 .25 median .75 .90 .95  
## 2.130 4.090 8.180 13.410 18.600 22.335 23.840  
##   
## range sd vcoef mad IQR skew kurt  
## 29.990 6.678 0.501 7.724 10.420 -0.028 -0.851  
##   
## lowest : 0.0 (184), 0.01 (3), 0.02 (5), 0.03 (2), 0.04 (3)  
## highest: 29.89, 29.92 (2), 29.93 (3), 29.95, 29.99  
##   
## ' 95%-CI (classic)



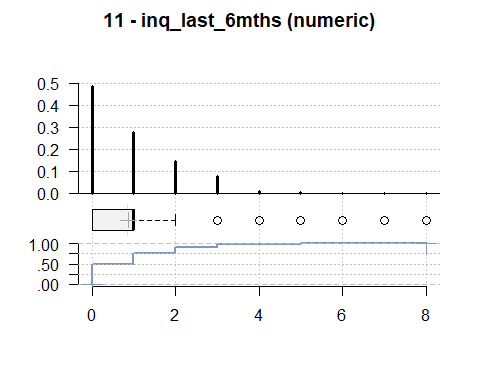
## ------------------------------------------------------------------------------   
## 9 - delinq\_2yrs (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 11 35'466 0.15 0.14  
## 100.0% 0.0% 89.1% 0.15  
##   
## .05 .10 .25 median .75 .90 .95  
## 0.00 0.00 0.00 0.00 0.00 1.00 1.00  
##   
## range sd vcoef mad IQR skew kurt  
## 11.00 0.49 3.36 0.00 0.00 5.02 39.36  
##   
##   
## value freq perc cumfreq cumperc  
## 1 0 35'466 89.1% 35'466 89.1%  
## 2 1 3'309 8.3% 38'775 97.5%  
## 3 2 688 1.7% 39'463 99.2%  
## 4 3 221 0.6% 39'684 99.7%  
## 5 4 62 0.2% 39'746 99.9%  
## 6 5 22 0.1% 39'768 100.0%  
## 7 6 10 0.0% 39'778 100.0%  
## 8 7 4 0.0% 39'782 100.0%  
## 9 8 2 0.0% 39'784 100.0%  
## 10 9 1 0.0% 39'785 100.0%  
## 11 11 1 0.0% 39'786 100.0%  
##   
## ' 95%-CI (classic)



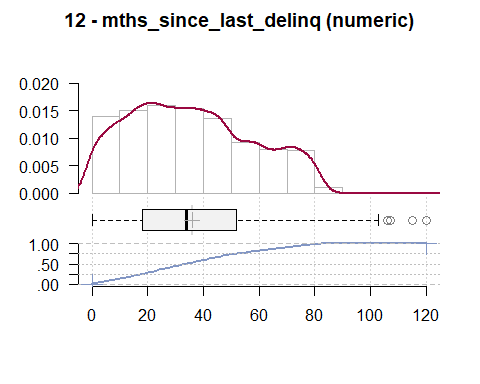
## ------------------------------------------------------------------------------   
## 10 - Fico\_score (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 36 0 715.00 714.65  
## 100.0% 0.0% 0.0% 715.35  
##   
## .05 .10 .25 median .75 .90 .95  
## 665.00 670.00 685.00 710.00 740.00 765.00 780.00  
##   
## range sd vcoef mad IQR skew kurt  
## 200.00 35.84 0.05 37.06 55.00 0.46 -0.56  
##   
## lowest : 625.0, 630.0, 660.0 (1'420), 665.0 (1'617), 670.0 (1'672)  
## highest: 805.0 (190), 810.0 (125), 815.0 (28), 820.0 (19), 825.0 (3)  
##   
## heap(?): remarkable frequency (5.4%) for the mode(s) (= 700)  
##   
## ' 95%-CI (classic)



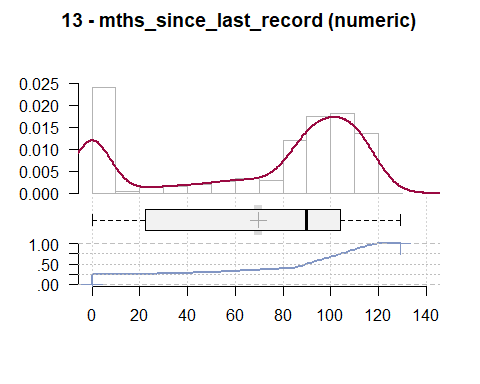
## ------------------------------------------------------------------------------   
## 11 - inq\_last\_6mths (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 9 19'337 0.87 0.86  
## 100.0% 0.0% 48.6% 0.88  
##   
## .05 .10 .25 median .75 .90 .95  
## 0.00 0.00 0.00 1.00 1.00 2.00 3.00  
##   
## range sd vcoef mad IQR skew kurt  
## 8.00 1.07 1.23 1.48 1.00 1.39 2.56  
##   
##   
## value freq perc cumfreq cumperc  
## 1 0 19'337 48.6% 19'337 48.6%  
## 2 1 10'986 27.6% 30'323 76.2%  
## 3 2 5'824 14.6% 36'147 90.9%  
## 4 3 3'053 7.7% 39'200 98.5%  
## 5 4 326 0.8% 39'526 99.3%  
## 6 5 146 0.4% 39'672 99.7%  
## 7 6 64 0.2% 39'736 99.9%  
## 8 7 35 0.1% 39'771 100.0%  
## 9 8 15 0.0% 39'786 100.0%  
##   
## ' 95%-CI (classic)



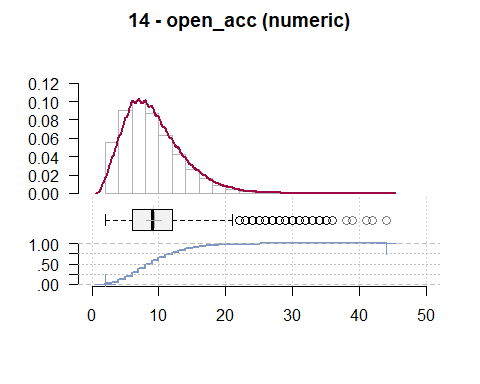
## ------------------------------------------------------------------------------   
## 12 - mths\_since\_last\_delinq (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 14'059 25'727 95 443 35.90 35.54  
## 35.3% 64.7% 1.1% 36.27  
##   
## .05 .10 .25 median .75 .90 .95  
## 3.00 8.00 18.00 34.00 52.00 69.00 75.00  
##   
## range sd vcoef mad IQR skew kurt  
## 120.00 22.02 0.61 25.20 34.00 0.31 -0.84  
##   
## lowest : 0.0 (443), 1.0 (30), 2.0 (101), 3.0 (145), 4.0 (153)  
## highest: 103.0 (2), 106.0, 107.0, 115.0, 120.0  
##   
## ' 95%-CI (classic)



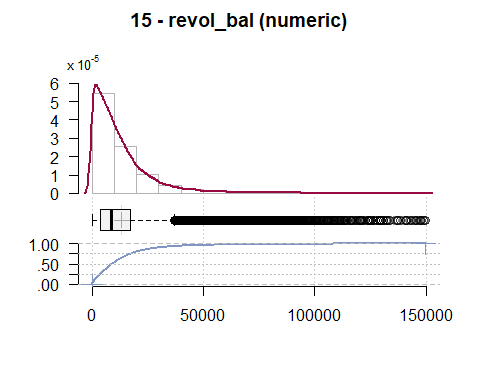
## ------------------------------------------------------------------------------   
## 13 - mths\_since\_last\_record (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 2'791 36'995 111 670 69.75 68.12  
## 7.0% 93.0% 1.7% 71.37  
##   
## .05 .10 .25 median .75 .90 .95  
## 0.00 0.00 22.50 90.00 104.00 113.00 115.00  
##   
## range sd vcoef mad IQR skew kurt  
## 129.00 43.81 0.63 29.65 81.50 -0.72 -1.15  
##   
## lowest : 0.0 (670), 5.0, 6.0, 7.0, 11.0 (2)  
## highest: 117.0 (47), 118.0 (36), 119.0 (10), 120.0, 129.0  
##   
## heap(?): remarkable frequency (24.0%) for the mode(s) (= 0)  
##   
## ' 95%-CI (classic)



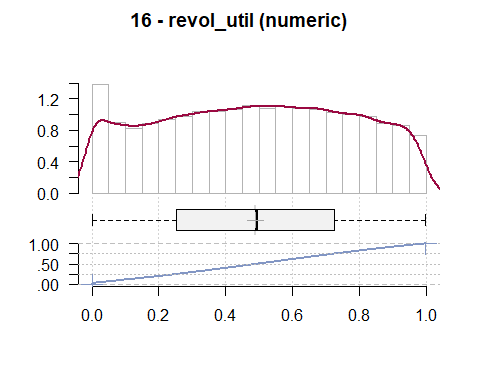
## ------------------------------------------------------------------------------   
## 14 - open\_acc (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 40 0 9.29 9.25  
## 100.0% 0.0% 0.0% 9.34  
##   
## .05 .10 .25 median .75 .90 .95  
## 3.00 4.00 6.00 9.00 12.00 15.00 17.00  
##   
## range sd vcoef mad IQR skew kurt  
## 42.00 4.40 0.47 4.45 6.00 1.00 1.68  
##   
## lowest : 2.0 (608), 3.0 (1'496), 4.0 (2'346), 5.0 (3'185), 6.0 (3'954)  
## highest: 38.0, 39.0, 41.0, 42.0, 44.0  
##   
## heap(?): remarkable frequency (10.1%) for the mode(s) (= 7)  
##   
## ' 95%-CI (classic)



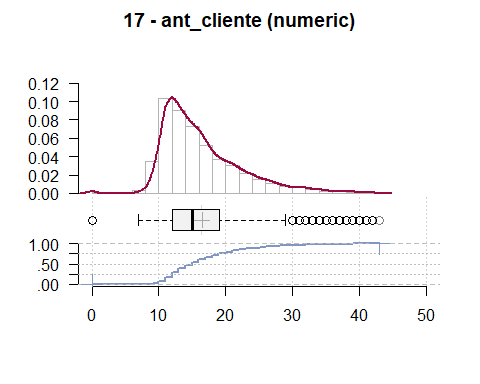
## ------------------------------------------------------------------------------   
## 15 - revol\_bal (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 21'738 996 13'391.98 13'235.80  
## 100.0% 0.0% 2.5% 13'548.17  
##   
## .05 .10 .25 median .75 .90 .95  
## 321.25 1'117.00 3'704.25 8'859.50 17'065.00 29'168.00 41'677.50  
##   
## range sd vcoef mad IQR skew kurt  
## 149'588.00 15'894.64 1.19 8'948.23 13'360.75 3.19 14.87  
##   
## lowest : 0.0 (996), 1.0 (12), 2.0 (5), 3.0 (6), 4.0 (3)  
## highest: 148'804.0, 148'829.0, 149'000.0, 149'527.0, 149'588.0  
##   
## ' 95%-CI (classic)



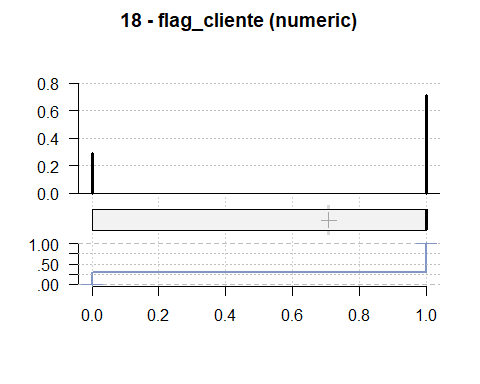
## ------------------------------------------------------------------------------   
## 16 - revol\_util (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'736 50 1'089 980 0.4886 0.4858  
## 99.9% 0.1% 2.5% 0.4914  
##   
## .05 .10 .25 median .75 .90 .95  
## 0.0270 0.0854 0.2540 0.4930 0.7240 0.8790 0.9360  
##   
## range sd vcoef mad IQR skew kurt  
## 0.9990 0.2834 0.5800 0.3484 0.4700 -0.0349 -1.1059  
##   
## lowest : 0.0 (980), 0.0001, 0.0003, 0.0004, 0.0005  
## highest: 0.995 (24), 0.996 (24), 0.997 (31), 0.998 (23), 0.999 (26)  
##   
## ' 95%-CI (classic)



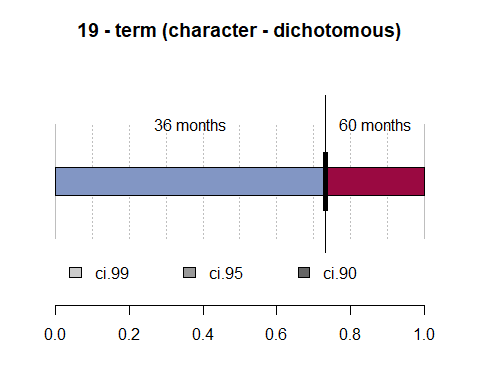
## ------------------------------------------------------------------------------   
## 17 - ant\_cliente (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 28'174 11'612 38 90 16.49 16.42  
## 70.8% 29.2% 0.2% 16.56  
##   
## .05 .10 .25 median .75 .90 .95  
## 10.00 11.00 12.00 15.00 19.00 25.00 29.00  
##   
## range sd vcoef mad IQR skew kurt  
## 43.00 5.99 0.36 4.45 7.00 1.27 1.91  
##   
## lowest : 0.0 (90), 7.0 (42), 8.0 (140), 9.0 (505), 10.0 (1'445)  
## highest: 39.0 (50), 40.0 (37), 41.0 (44), 42.0 (20), 43.0  
##   
## heap(?): remarkable frequency (10.7%) for the mode(s) (= 12)  
##   
## ' 95%-CI (classic)



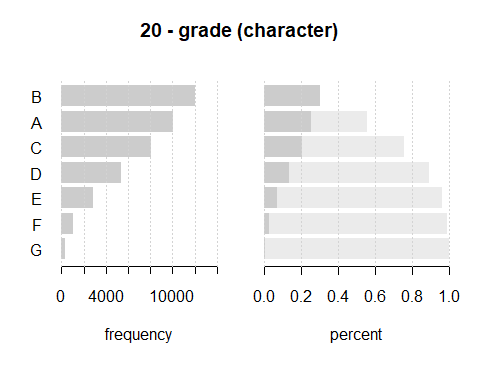
## ------------------------------------------------------------------------------   
## 18 - flag\_cliente (numeric)  
##   
## length n NAs unique 0s mean meanCI'  
## 39'786 39'786 0 2 11'612 0.71 0.70  
## 100.0% 0.0% 29.2% 0.71  
##   
## .05 .10 .25 median .75 .90 .95  
## 0.00 0.00 0.00 1.00 1.00 1.00 1.00  
##   
## range sd vcoef mad IQR skew kurt  
## 1.00 0.45 0.64 0.00 1.00 -0.92 -1.16  
##   
##   
## value freq perc cumfreq cumperc  
## 1 0 11'612 29.2% 11'612 29.2%  
## 2 1 28'174 70.8% 39'786 100.0%  
##   
## ' 95%-CI (classic)



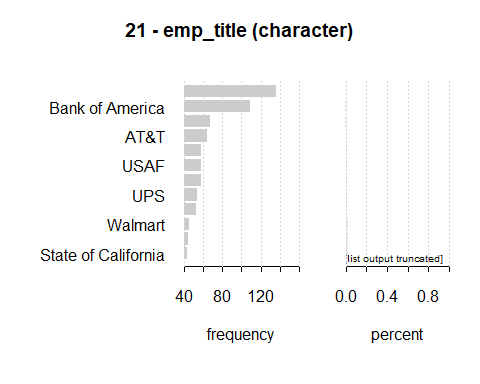
## ------------------------------------------------------------------------------   
## 19 - term (character - dichotomous)  
##   
## length n NAs unique  
## 39'786 39'786 0 2  
## 100.0% 0.0%   
##   
## freq perc lci.95 uci.95'  
## 36 months 29'096 73.1% 72.7% 73.6%  
## 60 months 10'690 26.9% 26.4% 27.3%  
##   
## ' 95%-CI (Wilson)



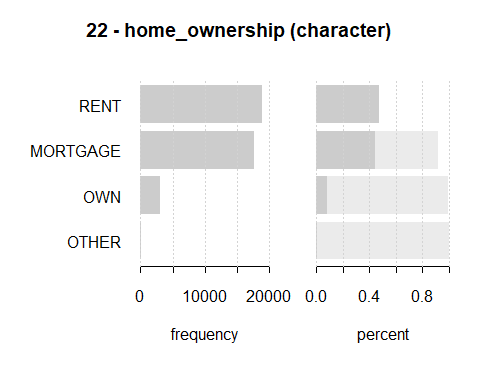
## ------------------------------------------------------------------------------   
## 20 - grade (character)  
##   
## length n NAs unique levels dupes  
## 39'786 39'786 0 7 7 y  
## 100.0% 0.0%   
##   
## level freq perc cumfreq cumperc  
## 1 B 12'035 30.2% 12'035 30.2%  
## 2 A 10'085 25.3% 22'120 55.6%  
## 3 C 8'111 20.4% 30'231 76.0%  
## 4 D 5'325 13.4% 35'556 89.4%  
## 5 E 2'858 7.2% 38'414 96.6%  
## 6 F 1'054 2.6% 39'468 99.2%  
## 7 G 318 0.8% 39'786 100.0%



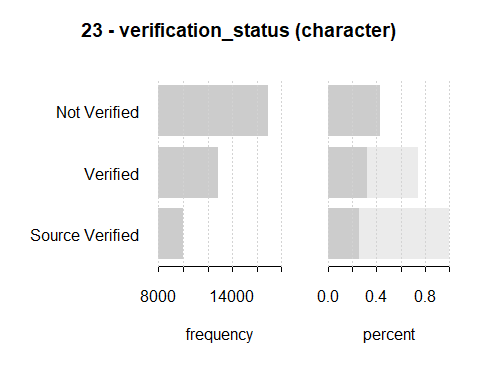
## ------------------------------------------------------------------------------   
## 21 - emp\_title (character)  
##   
## length n NAs unique levels dupes  
## 39'786 37'325 2'461 28'660 28'660 y  
## 93.8% 6.2%   
##   
## level freq perc cumfreq cumperc  
## 1 US Army 136 0.4% 136 0.4%  
## 2 Bank of America 109 0.3% 245 0.7%  
## 3 IBM 67 0.2% 312 0.8%  
## 4 AT&T 64 0.2% 376 1.0%  
## 5 Kaiser Permanente 57 0.2% 433 1.2%  
## 6 USAF 57 0.2% 490 1.3%  
## 7 Wells Fargo 57 0.2% 547 1.5%  
## 8 UPS 53 0.1% 600 1.6%  
## 9 US Air Force 52 0.1% 652 1.7%  
## 10 Walmart 45 0.1% 697 1.9%  
## 11 Lockheed Martin 44 0.1% 741 2.0%  
## 12 State of California 43 0.1% 784 2.1%  
## ... etc.  
## [list output truncated]



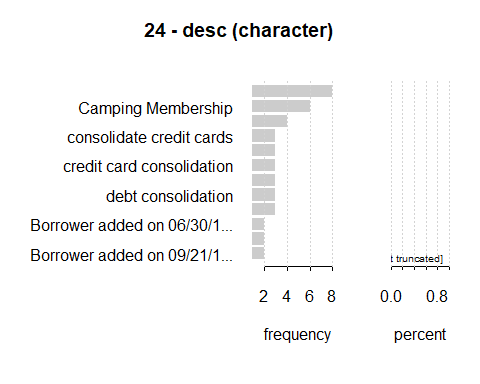
## ------------------------------------------------------------------------------   
## 22 - home\_ownership (character)  
##   
## length n NAs unique levels dupes  
## 39'786 39'786 0 4 4 y  
## 100.0% 0.0%   
##   
## level freq perc cumfreq cumperc  
## 1 RENT 18'918 47.5% 18'918 47.5%  
## 2 MORTGAGE 17'703 44.5% 36'621 92.0%  
## 3 OWN 3'064 7.7% 39'685 99.7%  
## 4 OTHER 101 0.3% 39'786 100.0%



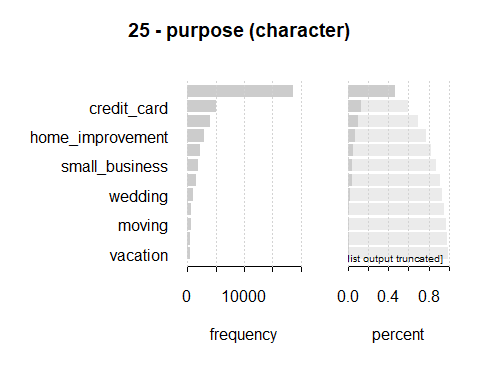
## ------------------------------------------------------------------------------   
## 23 - verification\_status (character)  
##   
## length n NAs unique levels dupes  
## 39'786 39'786 0 3 3 y  
## 100.0% 0.0%   
##   
## level freq perc cumfreq cumperc  
## 1 Not Verified 16'926 42.5% 16'926 42.5%  
## 2 Verified 12'844 32.3% 29'770 74.8%  
## 3 Source Verified 10'016 25.2% 39'786 100.0%



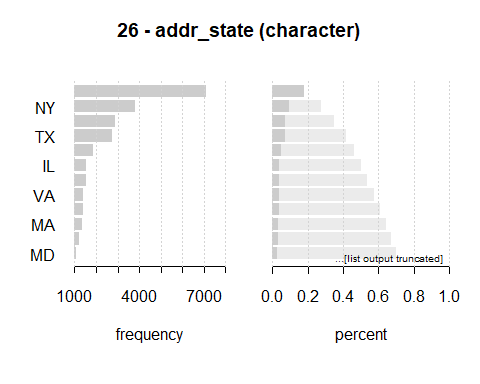
## ------------------------------------------------------------------------------   
## 24 - desc (character)  
##   
## length n NAs unique levels dupes  
## 39'786 26'610 13'176 26'562 26'562 y  
## 66.9% 33.1%   
##   
## level  
## 1 Debt Consolidation  
## 2 Camping Membership  
## 3 Personal Loan  
## 4 consolidate credit cards  
## 5 consolidate debt  
## 6 credit card consolidation  
## 7 credit card debt consolidation  
## 8 debt consolidation  
## 9 personal loan  
## 10 Borrower added on 06/30/11 > Debt Consolidation<br/>  
## 11 Borrower added on 08/31/11 > Presently embarking on the construction of a Business-class Social Network website potentially larger than other Social Networking sites. This website will cater to a Global audience of entrepreneurs, business professionals and individuals of all professions, industries and backgrounds. If successful, the revenue generated from this website will be exponential. Currently employed in the banking industry for 2 years and have been self-employed for 2.5 yrs. Planning to maintain full-time job in the interim while the website is being built and starts to generate substantial revenue. Any and all contributions are greatly appreciated.<br/>null  
## 12 Borrower added on 09/21/11 > Debt consolidation<br/>  
## freq perc cumfreq cumperc  
## 1 8 0.0% 8 0.0%  
## 2 6 0.0% 14 0.1%  
## 3 4 0.0% 18 0.1%  
## 4 3 0.0% 21 0.1%  
## 5 3 0.0% 24 0.1%  
## 6 3 0.0% 27 0.1%  
## 7 3 0.0% 30 0.1%  
## 8 3 0.0% 33 0.1%  
## 9 3 0.0% 36 0.1%  
## 10 2 0.0% 38 0.1%  
## 11 2 0.0% 40 0.2%  
## 12 2 0.0% 42 0.2%  
## ... etc.  
## [list output truncated]



## ------------------------------------------------------------------------------   
## 25 - purpose (character)  
##   
## length n NAs unique levels dupes  
## 39'786 39'786 0 14 14 y  
## 100.0% 0.0%   
##   
## level freq perc cumfreq cumperc  
## 1 debt\_consolidation 18'676 46.9% 18'676 46.9%  
## 2 credit\_card 5'137 12.9% 23'813 59.9%  
## 3 other 4'001 10.1% 27'814 69.9%  
## 4 home\_improvement 2'985 7.5% 30'799 77.4%  
## 5 major\_purchase 2'188 5.5% 32'987 82.9%  
## 6 small\_business 1'831 4.6% 34'818 87.5%  
## 7 car 1'551 3.9% 36'369 91.4%  
## 8 wedding 948 2.4% 37'317 93.8%  
## 9 medical 695 1.7% 38'012 95.5%  
## 10 moving 583 1.5% 38'595 97.0%  
## 11 house 382 1.0% 38'977 98.0%  
## 12 vacation 381 1.0% 39'358 98.9%  
## ... etc.  
## [list output truncated]

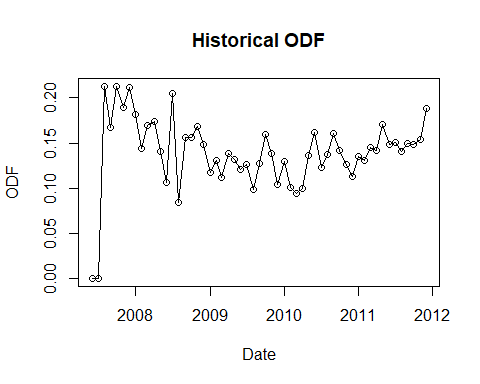


## ------------------------------------------------------------------------------   
## 26 - addr\_state (character)  
##   
## length n NAs unique levels dupes  
## 39'786 39'786 0 50 50 y  
## 100.0% 0.0%   
##   
## level freq perc cumfreq cumperc  
## 1 CA 7'105 17.9% 7'105 17.9%  
## 2 NY 3'817 9.6% 10'922 27.5%  
## 3 FL 2'872 7.2% 13'794 34.7%  
## 4 TX 2'734 6.9% 16'528 41.5%  
## 5 NJ 1'855 4.7% 18'383 46.2%  
## 6 IL 1'525 3.8% 19'908 50.0%  
## 7 PA 1'519 3.8% 21'427 53.9%  
## 8 VA 1'408 3.5% 22'835 57.4%  
## 9 GA 1'399 3.5% 24'234 60.9%  
## 10 MA 1'344 3.4% 25'578 64.3%  
## 11 OH 1'226 3.1% 26'804 67.4%  
## 12 MD 1'055 2.7% 27'859 70.0%  
## ... etc.  
## [list output truncated]

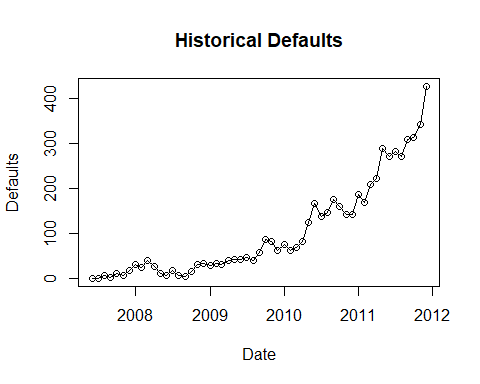


Vamos a ver algunos gráficos de evolución de los préstamos que tenemos disponibles en la base de datos. Por ejemplo, la evolución de la tasa de defaults a lo largo del tiempo:

Hist\_defaults<-sqldf("SELECT issue\_d,  
   
 count(loan\_status) as N,  
   
 sum(loan\_status) as Defaults,  
   
 avg(loan\_status) as ODF  
   
 from Loans group by issue\_d")  
  
  
plot(Hist\_defaults$issue\_d, Hist\_defaults$ODF, main="Historical ODF",xlab="Date",ylab="ODF")  
lines(Hist\_defaults$issue\_d, Hist\_defaults$ODF)



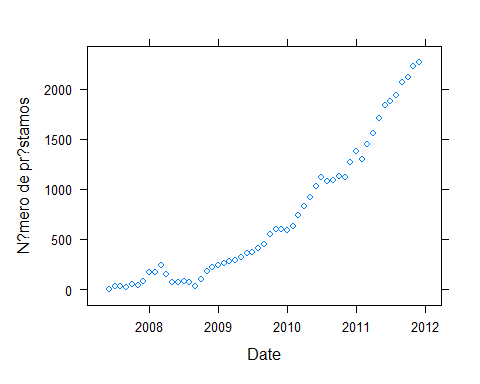
plot(Hist\_defaults$issue\_d, Hist\_defaults$Defaults, main="Historical Defaults",xlab="Date",ylab="Defaults")  
lines(Hist\_defaults$issue\_d, Hist\_defaults$Defaults)



Tenemos dos cosas aqui: - Por un lado, la tasa de default ha permanecido más o menos constante, ¿verdad? - Pero por otro lado vemos que los defaults han aumentado mucho. ¿Qué ha podido suceder?

Vamos entonces a ver también la evolución en el número de préstamos:

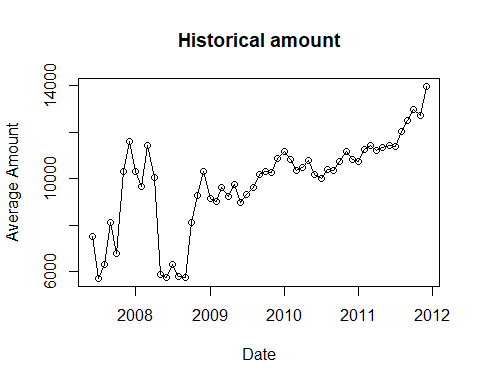
evolucion\_prestamos<-sqldf("SELECT issue\_d,  
   
 count(loan\_amnt) as Num\_prestamos  
   
 from Loans group by issue\_d")  
   
library(lattice)  
xyplot(Num\_prestamos ~ issue\_d, data=evolucion\_prestamos,ylab="N?mero de pr?stamos", xlab="Date",auto.key = TRUE)



Vemos que aunque el número de defaults haya aumentado en gran cantidad, el número de préstamos concedidos también lo ha hecho, manteniendo un número de defaults constante debido a este motivo.

También la evolución del importe medio concedido:

evolucion\_importe<-sqldf("SELECT issue\_d,  
   
 avg(loan\_amnt) as Average\_amount  
   
 from Loans group by issue\_d")  
  
  
plot(evolucion\_importe$issue\_d, evolucion\_importe$Average\_amount, main="Historical amount",xlab="Date",ylab="Average Amount")  
lines(evolucion\_importe$issue\_d, evolucion\_importe$Average\_amount)



Ahora vamos a ver si tenemos las variables lo suficientemente informadas y si nos interesa mantener todas las variables por este motivo. Primero calculamos el número de registros missing por variable:

data=Loans  
ncol=rep(nrow(data) ,each=ncol(data))  
nmsg=as.integer(as.character(as.vector(apply(data, 2, function(x) length(which(is.na(x)))))))  
missingdata=as.data.frame(cbind(colnames=names(data),ncol,nmsg))  
# missingdata$nmsg=as.numeric(levels(missingdata$nmsg))[missingdata$nmsg]  
missingdata$nmsg<-as.numeric(missingdata$nmsg)  
missingdata=cbind(missingdata,percmissing=(missingdata$nmsg/ncol\*100))  
  
print(nmsg)

## [1] 0 0 0 0 0 0 2461 1078 0 0 0 0  
## [13] 0 13176 0 0 0 0 11612 0 0 25727 36995 0  
## [25] 0 50 11612 0

print(missingdata)

## colnames ncol nmsg percmissing  
## 1 id 39786 0 0.0000000  
## 2 loan\_amnt 39786 0 0.0000000  
## 3 term 39786 0 0.0000000  
## 4 int\_rate 39786 0 0.0000000  
## 5 installment 39786 0 0.0000000  
## 6 grade 39786 0 0.0000000  
## 7 emp\_title 39786 2461 6.1855929  
## 8 emp\_length 39786 1078 2.7094958  
## 9 home\_ownership 39786 0 0.0000000  
## 10 annual\_inc 39786 0 0.0000000  
## 11 verification\_status 39786 0 0.0000000  
## 12 issue\_d 39786 0 0.0000000  
## 13 loan\_status 39786 0 0.0000000  
## 14 desc 39786 13176 33.1171769  
## 15 purpose 39786 0 0.0000000  
## 16 addr\_state 39786 0 0.0000000  
## 17 dti 39786 0 0.0000000  
## 18 delinq\_2yrs 39786 0 0.0000000  
## 19 earliest\_cr\_line 39786 11612 29.1861459  
## 20 Fico\_score 39786 0 0.0000000  
## 21 inq\_last\_6mths 39786 0 0.0000000  
## 22 mths\_since\_last\_delinq 39786 25727 64.6634495  
## 23 mths\_since\_last\_record 39786 36995 92.9849696  
## 24 open\_acc 39786 0 0.0000000  
## 25 revol\_bal 39786 0 0.0000000  
## 26 revol\_util 39786 50 0.1256723  
## 27 ant\_cliente 39786 11612 29.1861459  
## 28 flag\_cliente 39786 0 0.0000000

Vamos a ver las variables que tengan más de un 10% de valores missing:

print(missingdata[missingdata$percmissing>10,])

## colnames ncol nmsg percmissing  
## 14 desc 39786 13176 33.11718  
## 19 earliest\_cr\_line 39786 11612 29.18615  
## 22 mths\_since\_last\_delinq 39786 25727 64.66345  
## 23 mths\_since\_last\_record 39786 36995 92.98497  
## 27 ant\_cliente 39786 11612 29.18615

Hay que tratar de analizar los missing de las variables, ¿pueden aportar algo de información sobre la muestra? ¿Los registros vacios se comportan mejor o peor que la media de defaults de la cartera?. La primera de las variables (earliest\_cr\_line), nos sirvió de ayuda para crear la variable ant\_cliente. El número de missings coincide por este motivo. La variable earliest\_cr\_line ya no nos sirve de utilidad realmente:

Loans$earliest\_cr\_line<-NULL

Anteriormente vimos que la tasa de malos alcanzaba el 14.25%. ¿Cuál es la tasa de malos en la muestra cuando la variable “ant\_cliente” viene a missing?:

sqldf("SELECT   
 avg(loan\_status) as FDO  
   
 from Loans where ant\_cliente is null")

## FDO  
## 1 0.146917

Parece que es ligeramente superior y que los clientes con el valor a missing se comportan ligeramente peor que la media. Podríamos asignar el valor medio de la variable a los valores faltantes pero estaríamos perdiendo información. También podemos suponer que son clientes nuevos y sin antigüedad, pudiendo asignar un cero como valor. Eliminar los registros no es muy buena opción, pues estaríamos suprimiendo el 29% de la base de datos.

Otra opción es borrar la variable. En este caso vamos a mantener la variable sin realizar nada sobre ella pues luego agruparemos la variable y crearemos un grupo con los valores missing.

Analicemos ahora la tasa de defaults en los casos faltantes de la variable “mths\_since\_last\_delinq” en la que recordemos que había un 65% de registros missing:

sqldf("SELECT   
 avg(loan\_status) as FDO  
   
 from Loans where mths\_since\_last\_delinq is null")

## FDO  
## 1 0.136588

En estos casos, los préstamos cuya variable tiene valores missing se comportaran mejor que la media de la cartera de nuestra muestra. Parece lógico pensar que valores missing podrían recoger aquellos préstamos cuyos solicitantes nunca han tenido atrasos en los pagos. Por lo tanto este tipo de “missing” puede ser realmente un dato a tener en cuenta

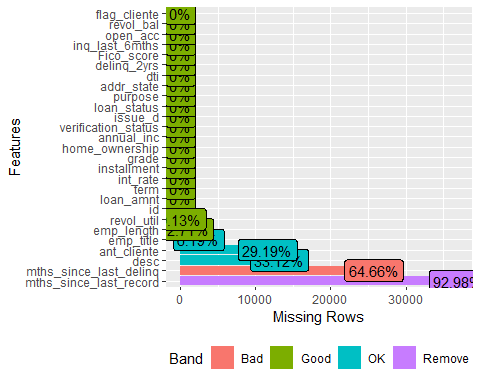
Por último, vemos la tasa de default en la variable “mths\_since\_last\_record”:

sqldf("SELECT   
 avg(loan\_status) as FDO  
   
 from Loans where mths\_since\_last\_record is null")

## FDO  
## 1 0.1375591

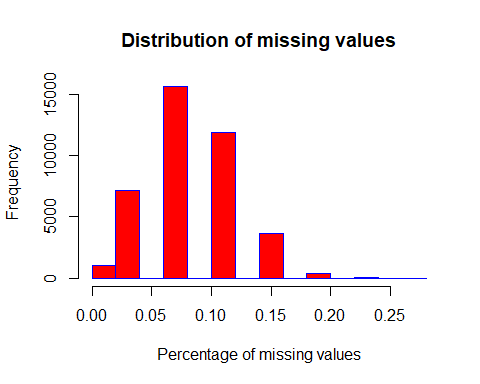
El racional viene a ser el mismo que la anterior variable. Otra manera de ver los valores missing:

plot\_missing(Loans)



Nos puede interesar suprimir aquellos registros que tengan más de un porcentaje de valores missing en todas las variables. Se podrían tratar de solicitudes de préstamos erróneas y que pueden afectar a nuestro modelo:

Loans$missingvalues<-rowSums(is.na(Loans[,2:27]))/26  
hist(Loans$missingvalues,main="Distribution of missing values",xlab="Percentage of missing values",border="blue", col="red",breaks=10)



summary(Loans$missingvalues)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00000 0.07692 0.07692 0.08807 0.11538 0.26923

Suelen eliminarse aquellos registros con un % de valores missing superior al 30%. En este caso no eliminaremos observaciones.

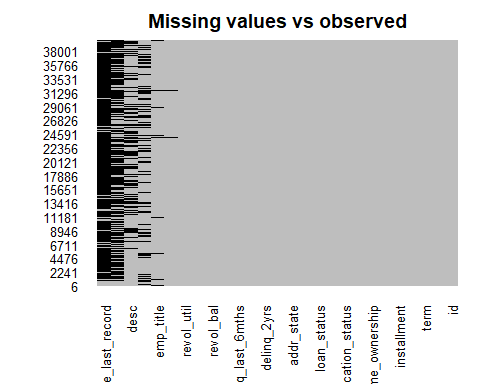
Loans$missingvalues<-NULL

Visualmente se pueden ver los valores faltantes con el siguiente paquete:

missmap(Loans, main = "Missing values vs observed",col=c("black", "grey"),,legend = FALSE)

## Warning: Unknown or uninitialised column: `arguments`.  
## Unknown or uninitialised column: `arguments`.

## Warning: Unknown or uninitialised column: `imputations`.



rm(list=setdiff(ls(), c("Loans","numeric\_vars","categorical\_vars")))  
  
save.image("Backup2.RData")

Para construir el modelo vamos a partir nuestros datos en muestra de desarrollo y validación. El 70% de los datos se usarán para desarrollar el modelo y el 30% restante para validar el modelo sobre una muestra diferente a la de construcción. Se pretende también que la selección aleatoria de la muestra mantenga la misma tasa de default en ambas muestras:

set.seed(1234)  
index = sample.split(Loans$loan\_status, SplitRatio = .70)  
train<-subset(Loans, index == TRUE)  
test<-subset(Loans, index == FALSE)

Vamos a comprobar el tamaño de ambas muestras y la tasa de default en cada una de ellas:

print("La muestra de entrenamiento tiene un total de:")

## [1] "La muestra de entrenamiento tiene un total de:"

nrow(train)

## [1] 27850

print("y tiene una tasa de defaults de:")

## [1] "y tiene una tasa de defaults de:"

mean(train$loan\_status)

## [1] 0.1425494

print("La muestra de validación tiene un total de:")

## [1] "La muestra de validación tiene un total de:"

nrow(test)

## [1] 11936

print("y tiene una tasa de defaults de:")

## [1] "y tiene una tasa de defaults de:"

mean(test$loan\_status)

## [1] 0.1425101

print("La muestra de entrenamiento tiene un tamaño del")

## [1] "La muestra de entrenamiento tiene un tamaño del"

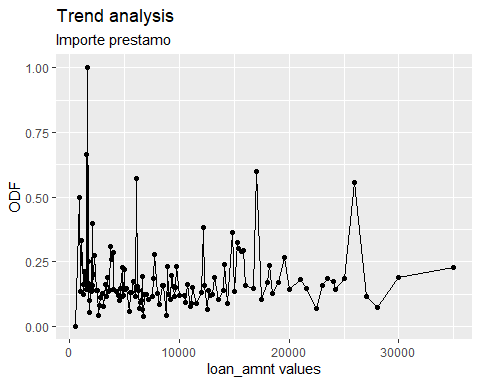
nrow(train)/nrow(Loans)

## [1] 0.699995

Hasta ahora hemos descrito los datos pero vamos a hacer un modelo que relacione esas variables con el default. ¿Como es esa relación de cada variable con la variable de desempeño de los préstamos?. Vamos a hacer un análisis gráfico inicial de cada variable:

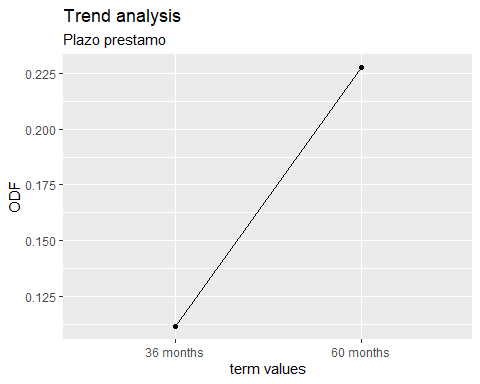
trends<-function(variable,label){  
  
  
temp <- train[,c(variable,"loan\_status")]  
  
if (class(temp[,1])=="numeric") {  
  
temp$quartile <- as.numeric(cut2(temp[,1], g=6)) #as.numeric to number the factors  
temp$quartileBounds <- cut2(temp[,1], g=6)  
temp$Interval <- paste(temp$quartile," - ",temp$quartileBounds,sep='')  
  
results<-sqldf("SELECT Interval,  
   
 count(loan\_status) as N,  
   
 sum(loan\_status) as Defaults,  
   
 avg(loan\_status) as ODF  
   
 from temp group by Interval")  
  
  
library(ggplot2)  
# Basic line plot with points  
ggplot(data=results, aes(x=Interval, y=ODF, group=1)) +  
 geom\_line()+  
 geom\_point() + labs(title="Trend analysis",   
 subtitle=paste(label,sep=''),  
 x=paste(variable, "values",sep=' '),  
 fill="loan\_status")  
  
} else {  
   
temp <- train[,c(variable,"loan\_status")]  
  
colnames(temp)[1]<-"var"  
  
results<-sqldf("SELECT var,  
   
 count(loan\_status) as N,  
   
 sum(loan\_status) as Defaults,  
   
 avg(loan\_status) as ODF  
   
 from temp group by var")  
  
results<-sqldf("SELECT \*  
   
 from results group by ODF")  
  
library(ggplot2)  
# Basic line plot with points  
ggplot(data=results, aes(x=var, y=ODF, group=1)) +  
 geom\_line()+  
 geom\_point() + labs(title="Trend analysis",   
 subtitle=paste(label,sep=''),  
 x=paste(variable, "values",sep=' '),  
 fill="loan\_status")  
  
  
}  
  
}  
  
trends("loan\_amnt","Importe prestamo")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



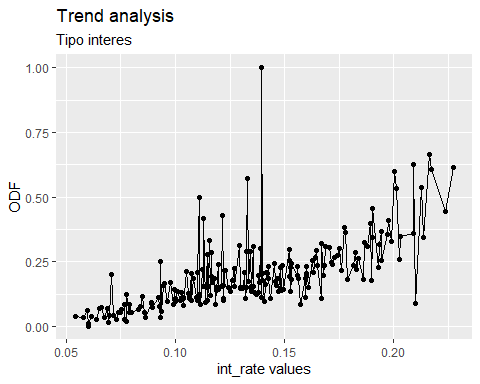
trends("term","Plazo prestamo")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



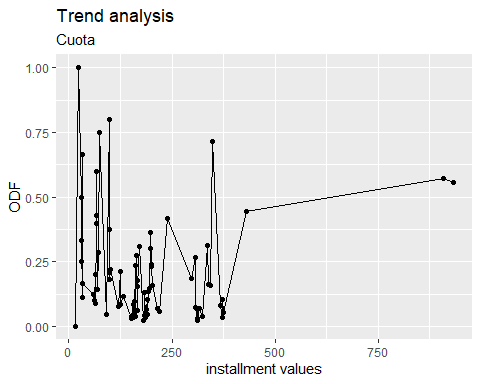
trends("int\_rate","Tipo interes")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



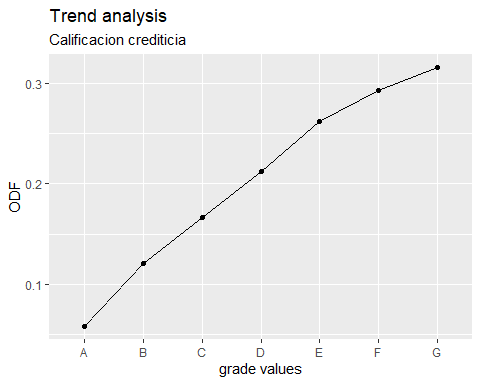
trends("installment","Cuota")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



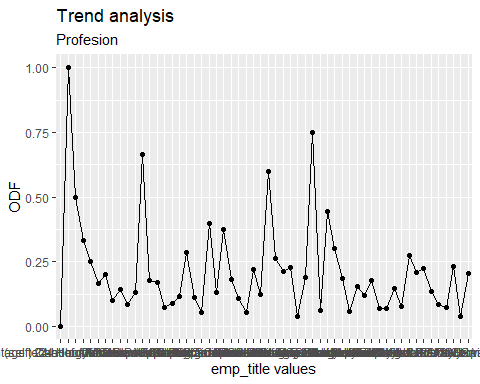
trends("grade","Calificacion crediticia")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



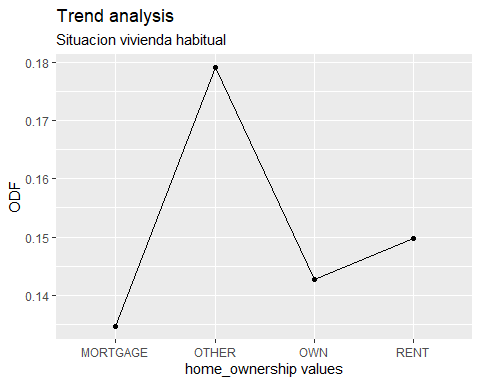
trends("emp\_title","Profesion")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



trends("home\_ownership","Situacion vivienda habitual")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

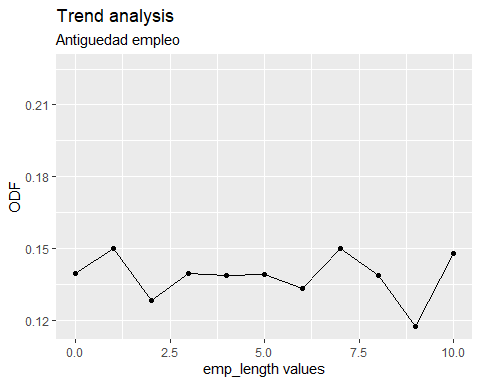


trends("emp\_length","Antiguedad empleo")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

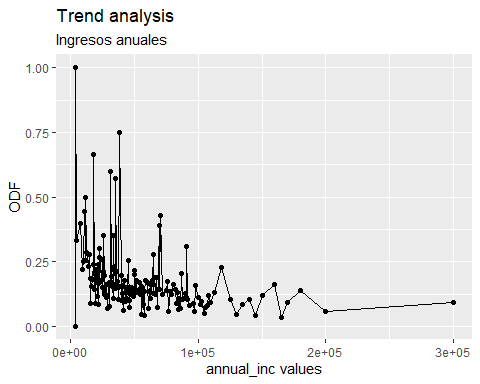
## Warning: Removed 1 row(s) containing missing values (geom\_path).

## Warning: Removed 1 rows containing missing values (geom\_point).



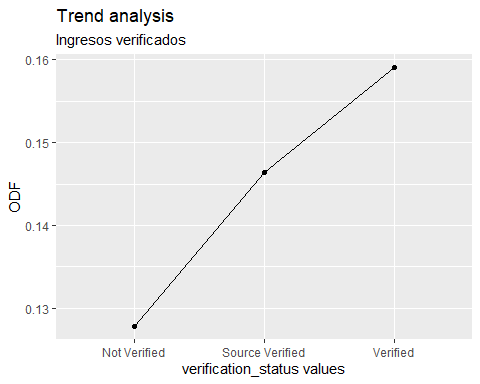
trends("annual\_inc","Ingresos anuales")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



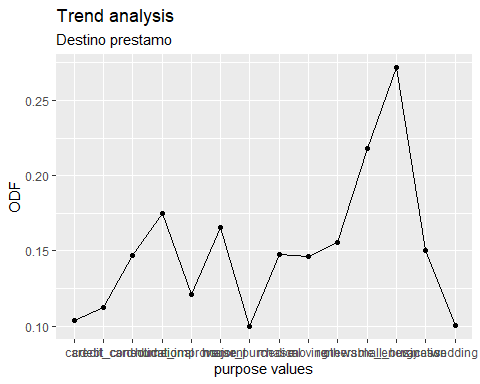
trends("verification\_status","Ingresos verificados")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



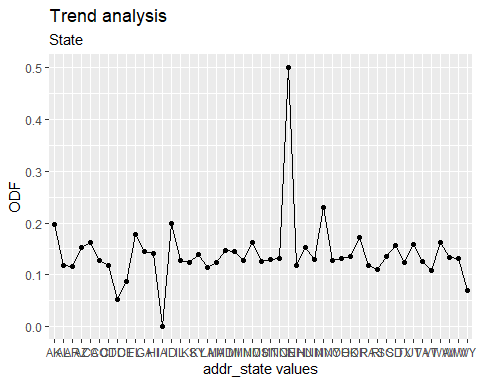
trends("purpose","Destino prestamo")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



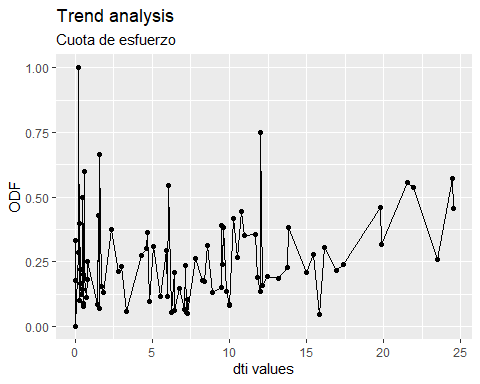
trends("addr\_state","State")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



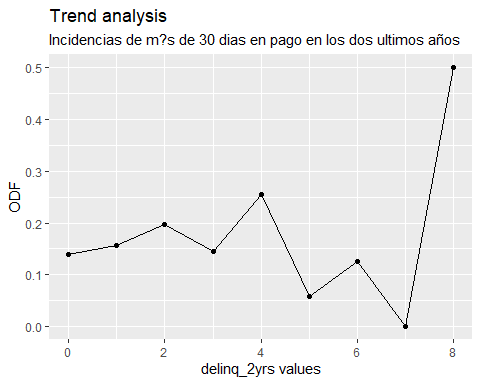
trends("dti","Cuota de esfuerzo")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



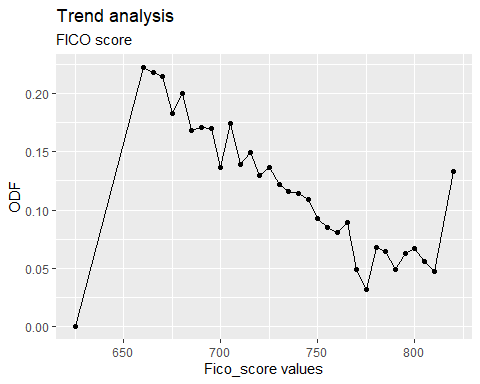
trends("delinq\_2yrs","Incidencias de m?s de 30 dias en pago en los dos ultimos años")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



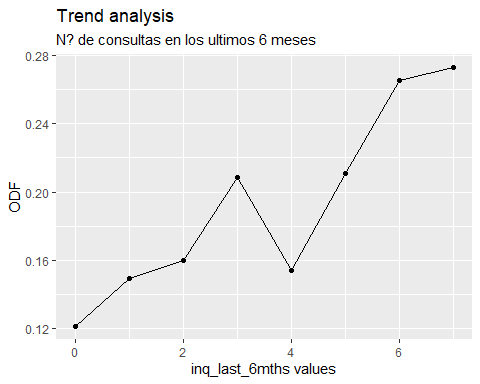
trends("Fico\_score","FICO score")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



trends("inq\_last\_6mths","N? de consultas en los ultimos 6 meses")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

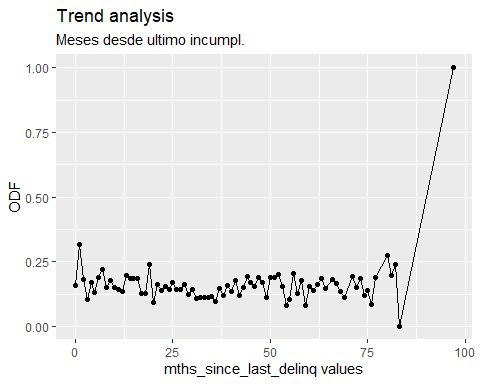


trends("mths\_since\_last\_delinq","Meses desde ultimo incumpl.")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

## Warning: Removed 1 row(s) containing missing values (geom\_path).

## Warning: Removed 1 rows containing missing values (geom\_point).

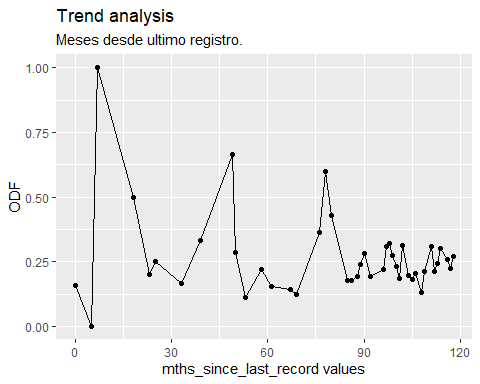


trends("mths\_since\_last\_record","Meses desde ultimo registro.")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

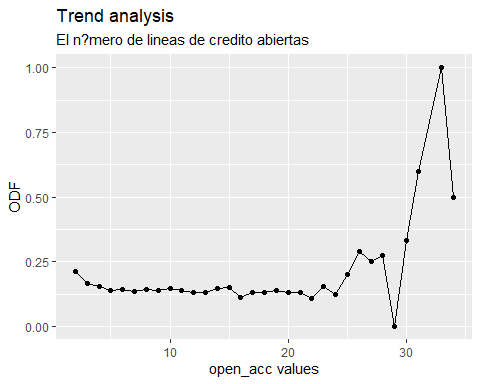
## Warning: Removed 1 row(s) containing missing values (geom\_path).

## Warning: Removed 1 rows containing missing values (geom\_point).



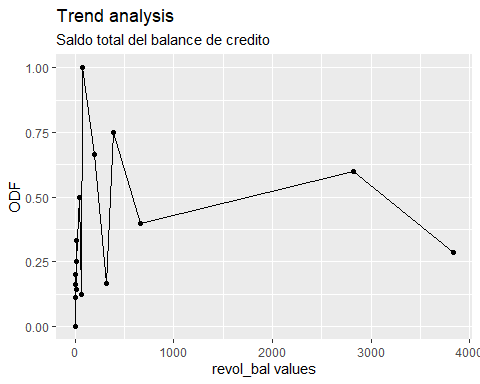
trends("open\_acc","El n?mero de lineas de credito abiertas ")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado



trends("revol\_bal","Saldo total del balance de credito")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

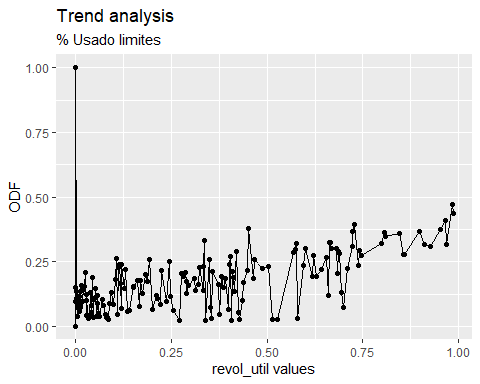


trends("revol\_util","% Usado limites")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

## Warning: Removed 1 row(s) containing missing values (geom\_path).

## Warning: Removed 1 rows containing missing values (geom\_point).

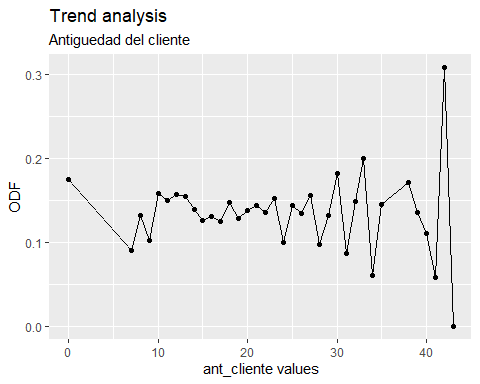


trends("ant\_cliente","Antiguedad del cliente")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

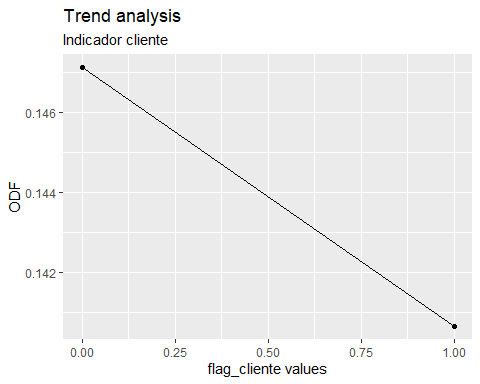
## Warning: Removed 1 row(s) containing missing values (geom\_path).

## Warning: Removed 1 rows containing missing values (geom\_point).



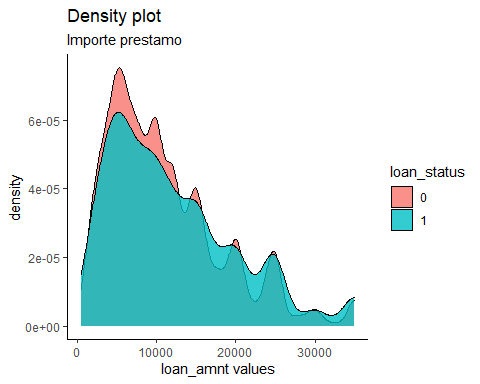
trends("flag\_cliente","Indicador cliente")

## Warning in if (class(temp[, 1]) == "numeric") {: la condición tiene longitud > 1  
## y sólo el primer elemento será usado

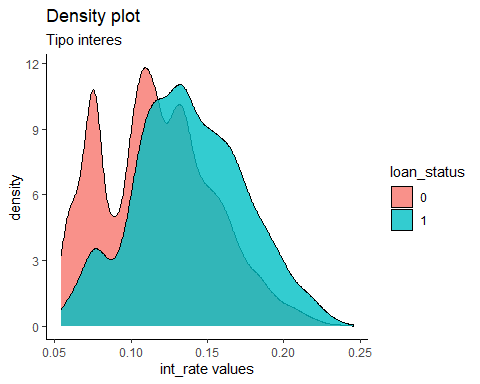


Otra manera de realizar el análisis es realizar gráficos de densidad para clientes “buenos” y “malos” en un único grafico para cada variable. Es decir, analizar gráficamente que valores toma cada variable para cada tipo de cliente. Vamos a hacerlo sobre las variables numéricas únicamente:

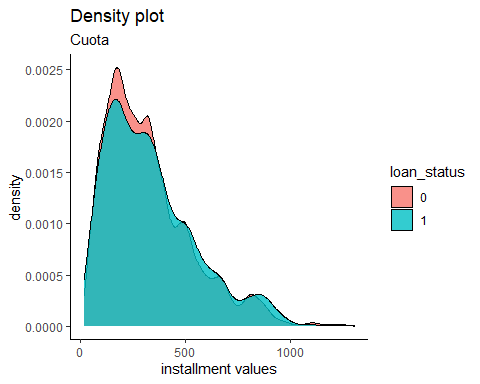
library(ggplot2)  
theme\_set(theme\_classic())  
  
  
#loan\_amnt  
  
ggplot(train, aes((loan\_amnt))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Importe prestamo",  
 x=paste("loan\_amnt values",sep=''),  
 fill="loan\_status")



#int\_rate  
  
ggplot(train, aes((int\_rate))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Tipo interes",  
 x=paste("int\_rate values",sep=''),  
 fill="loan\_status")

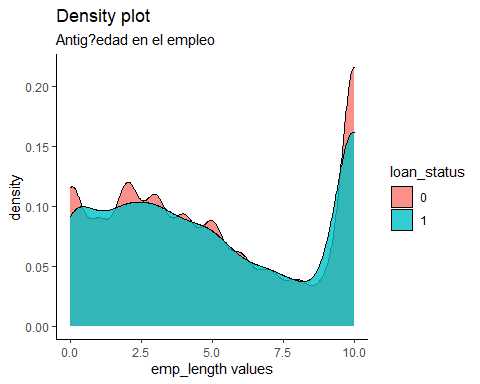


#installment  
  
ggplot(train, aes((installment))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Cuota",  
 x=paste("installment values",sep=''),  
 fill="loan\_status")

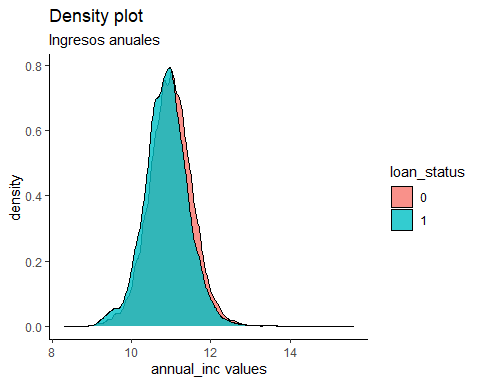


#emp\_length  
  
ggplot(train, aes((emp\_length))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Antig?edad en el empleo",  
 x=paste("emp\_length values",sep=''),  
 fill="loan\_status")

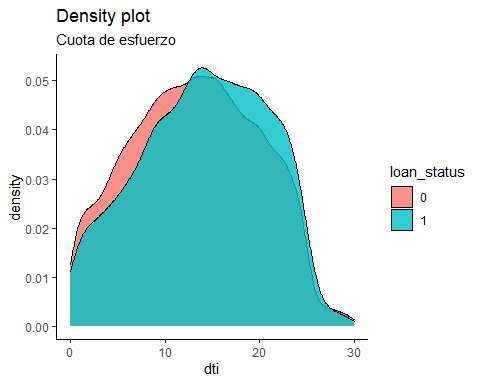
## Warning: Removed 745 rows containing non-finite values (stat\_density).



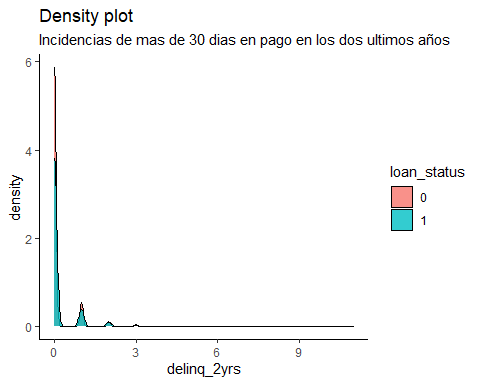
#anual\_inc  
  
ggplot(train, aes(log(annual\_inc))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Ingresos anuales",  
 x=paste("annual\_inc values",sep=''),  
 fill="loan\_status")



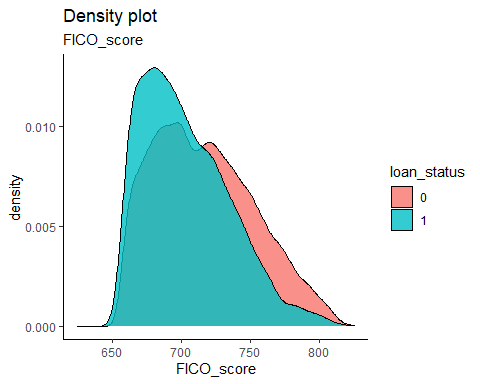
#dti  
  
ggplot(train, aes((dti))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Cuota de esfuerzo",  
 x=paste("dti",sep=''),  
 fill="loan\_status")



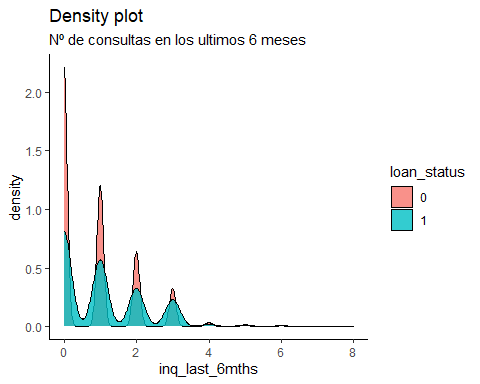
#delinq\_2yrs  
  
ggplot(train, aes((delinq\_2yrs))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Incidencias de mas de 30 dias en pago en los dos ultimos años",  
 x=paste("delinq\_2yrs",sep=''),  
 fill="loan\_status")



#FICO\_score  
  
ggplot(train, aes((Fico\_score))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="FICO\_score",  
 x=paste("FICO\_score",sep=''),  
 fill="loan\_status")

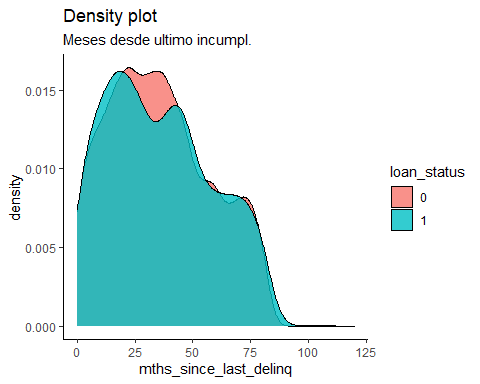


#inq\_last\_6mths  
  
ggplot(train, aes((inq\_last\_6mths))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Nº de consultas en los ultimos 6 meses",  
 x=paste("inq\_last\_6mths",sep=''),  
 fill="loan\_status")



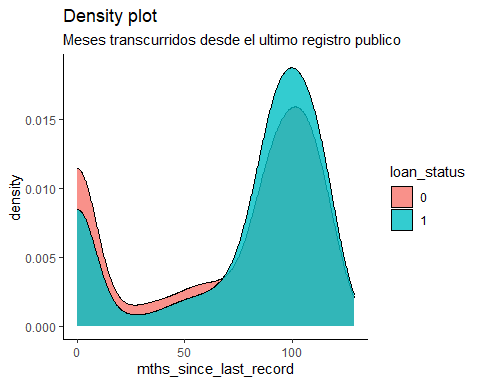
#mths\_since\_last\_delinq  
  
ggplot(train, aes((mths\_since\_last\_delinq))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Meses desde ultimo incumpl.",  
 x=paste("mths\_since\_last\_delinq",sep=''),  
 fill="loan\_status")

## Warning: Removed 18029 rows containing non-finite values (stat\_density).

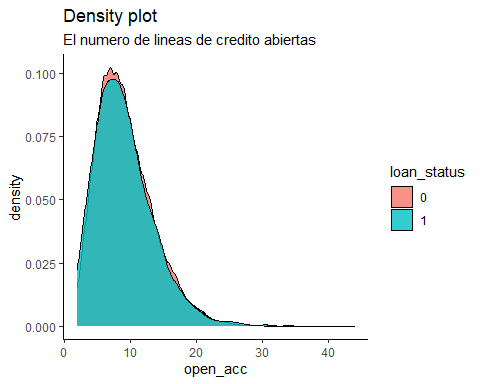


#mths\_since\_last\_record  
  
ggplot(train, aes((mths\_since\_last\_record))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Meses transcurridos desde el ultimo registro publico",  
 x=paste("mths\_since\_last\_record",sep=''),  
 fill="loan\_status")

## Warning: Removed 25898 rows containing non-finite values (stat\_density).

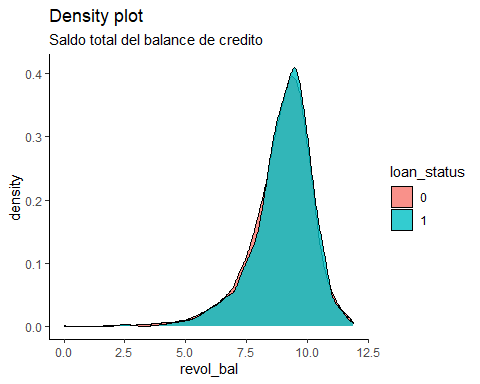


#open\_acc  
  
ggplot(train, aes((open\_acc))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="El numero de lineas de credito abiertas",  
 x=paste("open\_acc",sep=''),  
 fill="loan\_status")



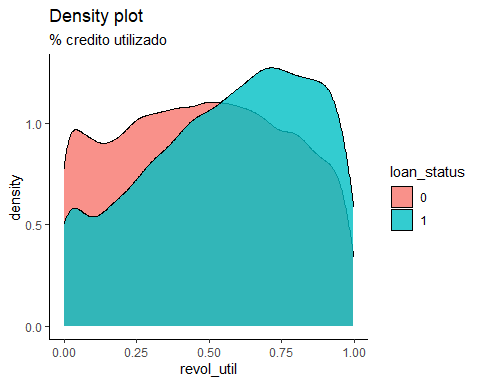
#revol\_bal  
  
ggplot(train, aes(log(revol\_bal))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Saldo total del balance de credito",  
 x=paste("revol\_bal",sep=''),  
 fill="loan\_status")

## Warning: Removed 722 rows containing non-finite values (stat\_density).



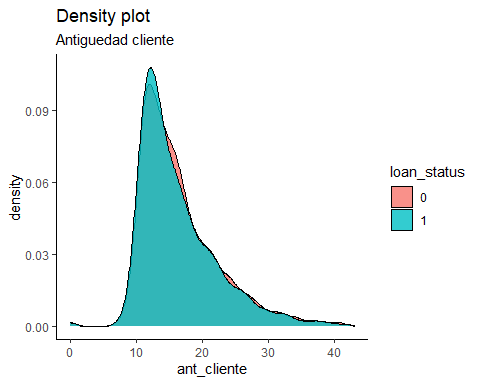
#revol\_util  
  
ggplot(train, aes((revol\_util))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="% credito utilizado",  
 x=paste("revol\_util",sep=''),  
 fill="loan\_status")

## Warning: Removed 35 rows containing non-finite values (stat\_density).

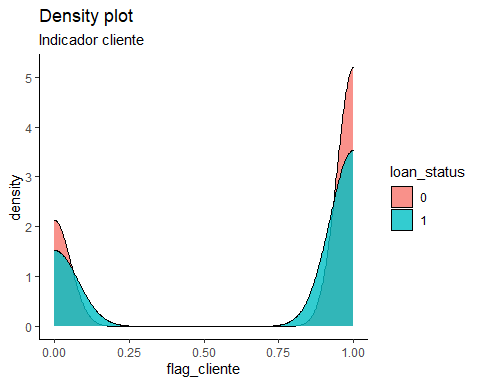


#ant\_cliente  
  
ggplot(train, aes((ant\_cliente))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Antiguedad cliente",  
 x=paste("ant\_cliente",sep=''),  
 fill="loan\_status")

## Warning: Removed 8157 rows containing non-finite values (stat\_density).



#flag\_cliente  
  
ggplot(train, aes((flag\_cliente))) + geom\_density(aes(fill=factor(loan\_status)),alpha=0.8) +   
 labs(title="Density plot",   
 subtitle="Indicador cliente",  
 x=paste("flag\_cliente",sep=''),  
 fill="loan\_status")



save.image("Backup3.RData")

De los gráficos anteriores podemos ver como hay algunas variables como el Fico score o el tipo de interés que parecen separar mejor a los clientes buenos y malos.

EJERCICIO 3: AGRUPACIÓN DE VARIABLES

Vamos a medir analiticamente cuál es la capacidad o poder discriminante de cada variable. Para usar el paquete que nos calcula el poder discriminante de las variables, es necesario considerar el default o el loan\_status al revés de lo que lo habiamos hecho, es decir un 0 si el préstamo resulto impagado y 1 en caso contrario:

source("woe\_functions.R")  
  
train$Default2<-as.numeric(as.character(train$loan\_status))  
train$Default2<-as.numeric(ifelse(train$Default2==1,0,1))  
test$Default2<-as.numeric(as.character(test$loan\_status))  
test$Default2<-as.numeric(ifelse(test$Default2==1,0,1))

Primero calculamos el poder predictivo de las variables numéricas:

##   
## | | | 0% | |--- | 5% | |----- | 11% | |-------- | 16% | |----------- | 21% | |------------- | 26% | |---------------- | 32% | |------------------ | 37% | |--------------------- | 42% | |------------------------ | 47% | |-------------------------- | 53% | |----------------------------- | 58% | |-------------------------------- | 63% | |---------------------------------- | 68% | |------------------------------------- | 74% | |--------------------------------------- | 79% | |------------------------------------------ | 84% | |--------------------------------------------- | 89% | |----------------------------------------------- | 95% | |--------------------------------------------------| 100%  
##

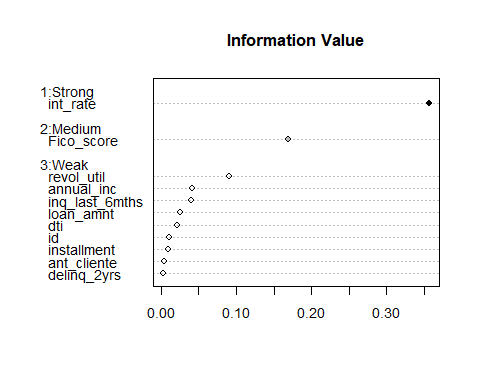
## Char IV Process  
## 3 int\_rate 0.3564 Numeric binning OK  
## 10 Fico\_score 0.1687 Numeric binning OK  
## 16 revol\_util 0.0904 Numeric binning OK  
## 6 annual\_inc 0.0416 Numeric binning OK  
## 11 inq\_last\_6mths 0.0402 Numeric binning OK  
## 2 loan\_amnt 0.0251 Numeric binning OK  
## 8 dti 0.0208 Numeric binning OK  
## 1 id 0.0105 Numeric binning OK  
## 4 installment 0.0098 Numeric binning OK  
## 17 ant\_cliente 0.0042 Numeric binning OK  
## 9 delinq\_2yrs 0.0033 Numeric binning OK  
## 5 emp\_length NA No significant splits  
## 7 loan\_status NA Uniques values < 5  
## 12 mths\_since\_last\_delinq NA No significant splits  
## 13 mths\_since\_last\_record NA No significant splits  
## 14 open\_acc NA No significant splits  
## 15 revol\_bal NA No significant splits  
## 18 flag\_cliente NA Uniques values < 5

El poder discriminante de cada variable teniendo en cuenta el IV se mide a través de la siguiente guía:

Information Value Predictive Power < 0.02: useless for prediction 0.02 to 0.1: Weak predictor 0.1 to 0.3: Medium predictor 0.3 to 0.5: Strong predictor >0.5 Suspicious or too good to be true

De acuerdo a los valores que hemos calculado:

smbinning.sumiv.plot(IV\_numeric, cex = 0.9)



La variable con más poder predictivo es el tipo de interés aplicable a los préstamos. Esta variable no suele ser empleada en este tipo de modelos de scoring pues es muy dependiente de la política de concesión de préstamos de la entidad, situación económica….

Cuando la función anterior devuelve el valor: “No significant splits”, indica que la variable no permite discriminar nada los clientes con problemas en el pago de los que no, por lo tanto podremos descartar estas variables.

Una de las variables flag\_cliente aparece con el mensaje “Uniques values of x < 5”. El paquete no realizara el calculo en valores numericos con menos de 5 valores unicos, ya que entiende que deberia ser casi una variable categorica:

table(train$flag\_cliente)

##   
## 0 1   
## 8157 19693

Vamos a tratar esta variable, convirtiéndola en categórica:

train$flag\_cliente<-as.factor(train$flag\_cliente)  
test$flag\_cliente<-as.factor(test$flag\_cliente)

Ahora calcularemos en poder predictivo de las variables categóricas (estas variables recordad que las tenemos creadas ya en el Ejercicio 1 de Exploración de Datos -línea aprox 446-).

Recordemos que tenemos una nueva variable categórica que acabamos de crear:

categorical\_vars<-c(categorical\_vars,"flag\_cliente")

Es necesario que las convirtamos en factor para el paquete:

NOTA IMPORTANTE: Para versiones de R anteriores a la 4.01 este código siguiente sí que funciona, En versiones posteriores da fallo así que lo haremos manualmente como se indica a continuación

#   
# for (m in 1:length(categorical\_vars)){  
#   
# var\_name<-as.character(categorical\_vars[m])  
# train[,var\_name]<-as.factor(train[,var\_name])  
# test[,var\_name]<-as.factor(test[,var\_name])  
# }  
#

Para versiones de R posteriores a 4.1 este código deberá utilizarse en lugar del anterior, ya que da fallo al parecer.

#Train  
train$term=as.factor(train$term)  
train$grade=as.factor(train$grade)  
train$emp\_title=as.factor(train$emp\_title)  
train$home\_ownership=as.factor(train$home\_ownership)  
train$verification\_status=as.factor(train$verification\_status)  
train$desc=as.factor(train$desc)  
train$purpose=as.factor(train$purpose)  
train$addr\_state=as.factor(train$addr\_state)  
train$flag\_cliente=as.factor(train$flag\_cliente)  
  
  
  
  
#Test  
test$term=as.factor(test$term)  
test$grade=as.factor(test$grade)  
test$emp\_title=as.factor(test$emp\_title)  
test$home\_ownership=as.factor(test$home\_ownership)  
test$verification\_status=as.factor(test$verification\_status)  
test$desc=as.factor(test$desc)  
test$purpose=as.factor(test$purpose)  
test$addr\_state=as.factor(test$addr\_state)  
test$flag\_cliente=as.factor(test$flag\_cliente)

NOTA: Para versiones superiores de R de 4.1 hay que referenciar la muestra train como as.data.frame() para que funcione el código. Como vemos mas abajo

Ahora calculemos el IV de estas variables:

IV\_cat<-matrix("NA",length(categorical\_vars),2)  
colnames(IV\_cat)<-c("Char","IV")  
  
  
for (o in 1:length(categorical\_vars)){  
   
 name<-as.character(categorical\_vars[o])   
 aux<-as.data.frame(train[,c("Default2",name)])  
 IV\_cat[o,1]<-name  
 IV\_cat[o,2]<-as.numeric(as.character(iv\_categorical(aux[,1],aux[,2])))  
   
}  
  
IV\_cat<-as.data.frame(IV\_cat)  
IV\_cat$IV<-as.numeric(as.character(IV\_cat$IV))  
print(IV\_cat)

## Char IV  
## 1 term 0.15742  
## 2 grade 0.32213  
## 3 emp\_title Inf  
## 4 home\_ownership 0.00376  
## 5 verification\_status 0.01234  
## 6 desc Inf  
## 7 purpose 0.06958  
## 8 addr\_state Inf  
## 9 flag\_cliente 0.00057

Vemos que hay tres variables con un valor “Inf”. La variable “desc”, como dijimos la obviamos. Para las variables “addr\_state” y “emp\_title” lo que ocurre es que son muy granulares y dan problemas en el cálculo. Estas variables podrían convertirse en numéricas y tratarlas como tal. Asignemos la tasa de default observada en cada valor de las variables y convirtamos las variables así en numericas. Empecemos con la variable indicativa del estado de residencia del solicitante:

detail\_state<-sqldf("SELECT addr\_state,  
  
 avg(loan\_status) as ODF\_addr\_state  
   
 from train group by addr\_state")  
  
detail\_state$ODF\_addr\_state<-round(detail\_state$ODF\_addr\_state,2)  
  
detail\_state<-detail\_state[order(detail\_state$ODF),]  
detail\_state

## addr\_state ODF\_addr\_state  
## 13 IA 0.00  
## 16 IN 0.00  
## 22 ME 0.00  
## 8 DC 0.05  
## 50 WY 0.07  
## 9 DE 0.09  
## 19 LA 0.11  
## 39 RI 0.11  
## 46 VT 0.11  
## 2 AL 0.12  
## 3 AR 0.12  
## 7 CT 0.12  
## 17 KS 0.12  
## 20 MA 0.12  
## 26 MS 0.12  
## 30 NH 0.12  
## 38 PA 0.12  
## 43 TX 0.12  
## 6 CO 0.13  
## 15 IL 0.13  
## 24 MN 0.13  
## 27 MT 0.13  
## 28 NC 0.13  
## 32 NM 0.13  
## 34 NY 0.13  
## 35 OH 0.13  
## 40 SC 0.13  
## 45 VA 0.13  
## 48 WI 0.13  
## 49 WV 0.13  
## 12 HI 0.14  
## 18 KY 0.14  
## 36 OK 0.14  
## 4 AZ 0.15  
## 11 GA 0.15  
## 21 MD 0.15  
## 23 MI 0.15  
## 31 NJ 0.15  
## 5 CA 0.16  
## 25 MO 0.16  
## 41 SD 0.16  
## 44 UT 0.16  
## 47 WA 0.16  
## 37 OR 0.17  
## 10 FL 0.18  
## 1 AK 0.20  
## 14 ID 0.20  
## 42 TN 0.20  
## 33 NV 0.23  
## 29 NE 0.50

unique(detail\_state$ODF\_addr\_state)

## [1] 0.00 0.05 0.07 0.09 0.11 0.12 0.13 0.14 0.15 0.16 0.17 0.18 0.20 0.23 0.50

Asignemos la ODF de cada estado en las muestras:

train<-merge(train,detail\_state,by.x=c("addr\_state"))  
  
test<-merge(test,detail\_state,by.x=c("addr\_state"))

El valor del IV de esta nueva variable numerica es:

smbinning.sumiv(train[,c("Default2","ODF\_addr\_state")], "Default2")

##   
## | | | 0% | |------------------------- | 50% | |--------------------------------------------------| 100%  
##

## Char IV Process  
## 1 ODF\_addr\_state 0.029 Numeric binning OK

Tras obtener el IV vemos que la variable del Estado no aportaría mucha información al modelo. (IV<0,01 weak predictor), no obstante la mantendremos y analizaremos más adelante.

La variable emp\_title es tan granular y diferente por cada solicitante que preferimos no considerarla en el modelo.

Respecto a la variable “purpose”, consideramos que también es muy granular y no queremos tener tantos grupos como categorías. Vamos a transformarla en variable numérica también a través de su ODF:

detail\_purpose<-sqldf("SELECT purpose,  
  
 avg(loan\_status) as ODF\_purpose  
   
 from train group by purpose")  
  
detail\_purpose$ODF\_purpose<-as.character(round(detail\_purpose$ODF\_purpose,2))  
detail\_purpose$ODF\_purpose<-as.factor(detail\_purpose$ODF\_purpose)  
  
detail\_purpose

## purpose ODF\_purpose  
## 1 car 0.1  
## 2 credit\_card 0.11  
## 3 debt\_consolidation 0.15  
## 4 educational 0.17  
## 5 home\_improvement 0.12  
## 6 house 0.17  
## 7 major\_purchase 0.1  
## 8 medical 0.15  
## 9 moving 0.15  
## 10 other 0.16  
## 11 renewable\_energy 0.22  
## 12 small\_business 0.27  
## 13 vacation 0.15  
## 14 wedding 0.1

train<-merge(train,detail\_purpose,by.x=c("purpose"))  
test<-merge(test,detail\_purpose,by.x=c("purpose"))

El nuevo IV de la variable purpose sería:

smbinning.sumiv(train[,c("Default2","ODF\_purpose")], "Default2")

##   
## | | | 0% | |------------------------- | 50% | |--------------------------------------------------| 100%  
##

## Char IV Process  
## 1 ODF\_purpose 0.07 Factor binning OK

El cálculo del IV se hace realizando un groupping de los valores de cada variable para encontrar cuál es la mejor separación entre clientes buenos y malos. Esta agrupación es luego empleada en la modelización, nuestras variables del modelo ya no serán las variables con sus valores originales, si no las variables transformadas en categorías o grupos como hemos visto en la teoría.

Los grupos son creados automáticamente y de manera estadística tratando de discriminar lo mejor posible entre los diferentes tipos de clientes (buenos y malos). Sin embargo estas agrupaciones para los analistas deben tener sentido economico.

Lo que se pretende ahora es ver si se cumple que los grupos que se crean automáticamente tienen sentido económico o hay que retocarles. Creamos la siguiente función para realizar el análisis:

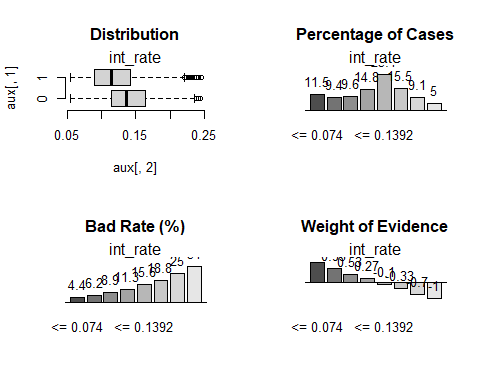
variables\_analysis<-function(variable){  
   
  
  
 if (class(train[,variable])=="numeric" | class(train[,variable])=="integer") {  
   
 name<-as.character(variable)   
 aux<-train[,c("loan\_status",name)]  
 aux$loan\_status<-as.numeric(ifelse(train$loan\_status==1,0,1))  
 groupping<-smbinning\_calc(df=aux,y="loan\_status",x=name,p=0.05)  
   
 par(mfrow=c(2,2))  
 boxplot(aux[,2]~aux[,1],   
 horizontal=T, frame=F, col="lightgray",main="Distribution")   
 mtext(name,3)   
  
 smbinning.plotty(groupping,option="dist",sub=name)   
 smbinning.plotty(groupping,option="badrate",sub=name)  
 smbinning.plotty(groupping,option="WoE",sub=name)  
  
 }  
 else {  
   
 name<-as.character(variable)   
 aux<-train[,c("loan\_status",name)]  
 aux$loan\_status<-as.numeric(ifelse(train$loan\_status==1,0,1))  
 groupping<-smbinning.factor(df=aux,y="loan\_status",x=name,maxcat = 8)  
   
 par(mfrow=c(2,2))  
  
 smbinning.plotty(groupping,option="dist",sub=name)   
 smbinning.plotty(groupping,option="badrate",sub=name)  
 smbinning.plotty(groupping,option="WoE",sub=name)  
  
 }  
}

Vamos a realizar el analisis sobre las variables que hemos visto que son mas predictivas anteriormente:

int\_rate Fico\_score revol\_util annual\_inc loan\_amnt dti ODF\_addr\_state term grade ODF\_purpose

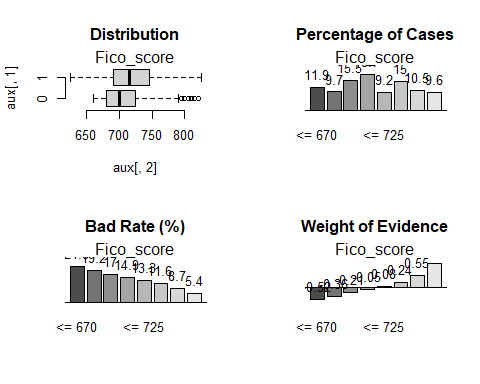
variables\_analysis("int\_rate")

## Warning: NAs introducidos por coerción

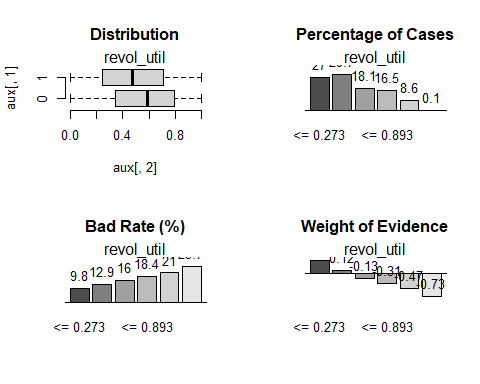


variables\_analysis("Fico\_score")

## Warning in smbinning\_calc(df = aux, y = "loan\_status", x = name, p = 0.05): NAs  
## introducidos por coerción

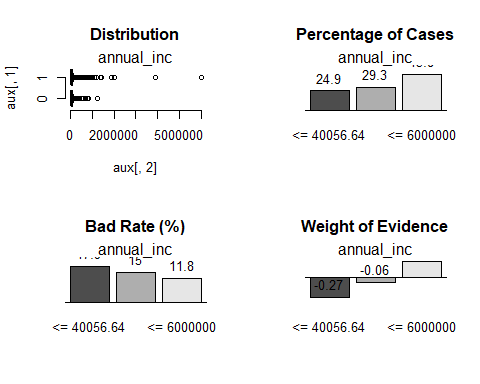


variables\_analysis("revol\_util")



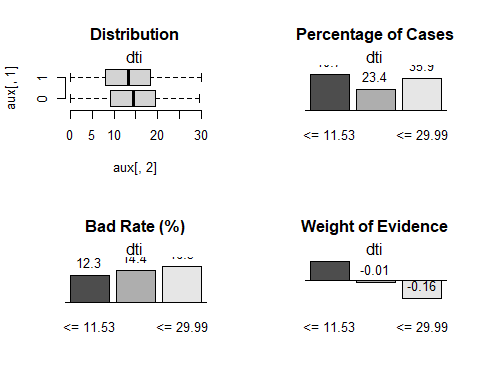
variables\_analysis("annual\_inc")

## Warning in smbinning\_calc(df = aux, y = "loan\_status", x = name, p = 0.05): NAs  
## introducidos por coerción



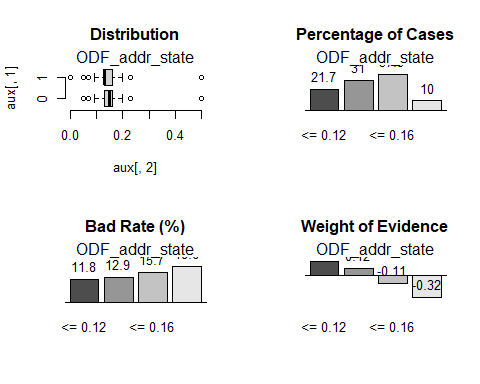
variables\_analysis("dti")

## Warning in smbinning\_calc(df = aux, y = "loan\_status", x = name, p = 0.05): NAs  
## introducidos por coerción

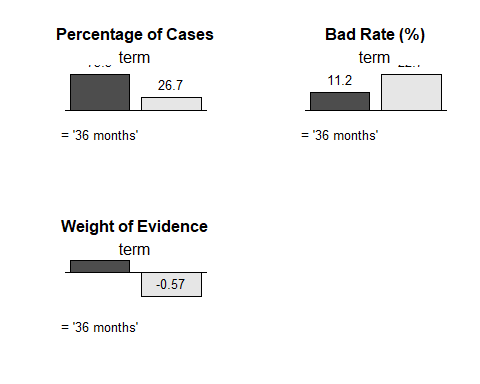


variables\_analysis("ODF\_addr\_state")

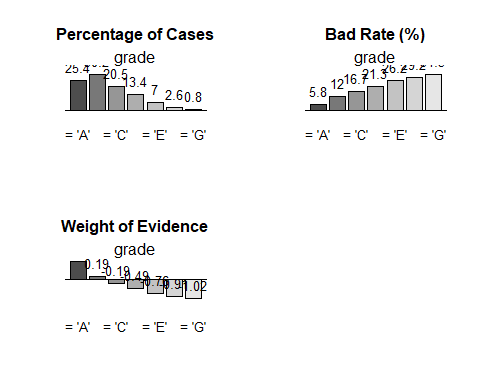
## Warning in smbinning\_calc(df = aux, y = "loan\_status", x = name, p = 0.05): NAs  
## introducidos por coerción



variables\_analysis("term")  
variables\_analysis("grade")

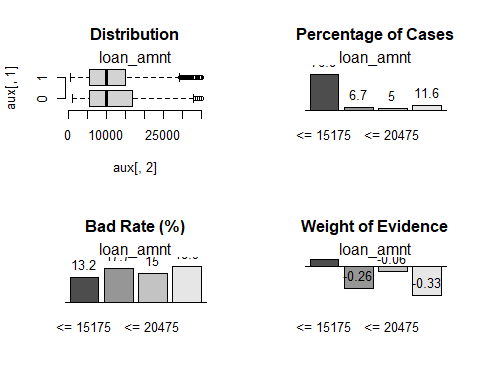
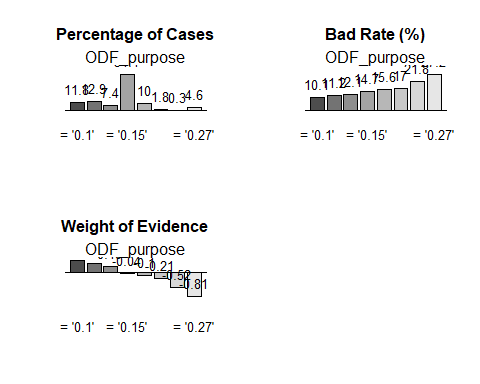


variables\_analysis("ODF\_purpose")



variables\_analysis("loan\_amnt")

## Warning in smbinning\_calc(df = aux, y = "loan\_status", x = name, p = 0.05): NAs  
## introducidos por coerción

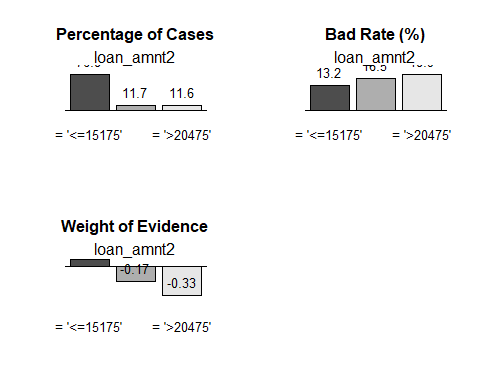


Como puede verse la variable loan\_amnt parece comportarse de manera extraña o al menos poco intuitiva de explicar. Cuanto mayor es el importe del préstamo mayor es la tasa de default, aunque hay un tramo (de 15.175 a 20.475 euros) que parece comportarse mejor que el anterior grupo). Vamos a unir los dos grupos intermedios para conseguir esa monotonocidad. Creamos una variable loan\_amnt2 que será la que usemos a partir de ahora:

train$loan\_amnt2<-as.factor(ifelse(train$loan\_amnt<=15175,"<=15175",ifelse(train$loan\_amnt<=20475,"<=20475",">20475")))  
  
test$loan\_amnt2<-as.factor(ifelse(test$loan\_amnt<=15175,"<=15175",ifelse(test$loan\_amnt<=20475,"<=20475",">20475")))

y vemos los resultados:

variables\_analysis("loan\_amnt2")



Una vez que vemos que los grupos tienen sentido y que variables queremos emplear, realizamos el agrupamiento en las muestras, creando una nueva variable transformada para cada variable de entrada. Usaremos dos funciones:

Groupping\_numeric<-function(variable){  
  
name<-as.character(variable)   
aux\_train<-train[,c("id","loan\_status",name)]  
aux\_test<-test[,c("id","loan\_status",name)]  
colnames(aux\_train)[3]<-"var\_name"  
colnames(aux\_test)[3]<-"var\_name"  
aux\_train$loan\_status<-as.numeric(ifelse(train$loan\_status==1,0,1))  
aux\_test$loan\_status<-as.numeric(ifelse(test$loan\_status==1,0,1))  
groupping<-smbinning\_calc(df=aux\_train,y="loan\_status",x="var\_name",p=0.05)  
  
table\_woe<-as.data.frame(groupping$ivtable)   
 for (i in 1:nrow(table\_woe)){  
   
 table\_woe$ID[i]<-i  
   
 }  
table\_woe$ID<-paste("0",table\_woe$ID,sep='')  
  
  
aux\_train<-smbinning.gen(aux\_train,groupping,"Bucket")  
aux\_train$ID<-substr(aux\_train$Bucket,1,2)  
  
aux\_test<-smbinning.gen(aux\_test,groupping,"Bucket")  
aux\_test$ID<-substr(aux\_test$Bucket,1,2)  
  
  
#table\_woe$Bucket<-paste(row.names(table\_woe),table\_woe$Cutpoint,sep=' ')  
#table\_woe$Bucket<-paste("0",table\_woe$Bucket,sep='')  
table\_woe<-table\_woe[,c("ID","Cutpoint","WoE")]  
  
aux\_train<-merge(aux\_train,table\_woe,by.x=c("ID"),all.x=TRUE)  
aux\_train$Bucket<-NULL  
aux\_test<-merge(aux\_test,table\_woe,by.x=c("ID"),all.x=TRUE)  
aux\_test$Bucket<-NULL  
  
miss\_woe<-table\_woe[grep("Missing",table\_woe$Cutpoint),]  
  
if (is.na(miss\_woe$WoE)==FALSE){  
  
value<- as.numeric(miss\_woe$WoE)  
#value<-as.numeric(gsub("NaN",0,value))  
   
} else { value <-0 }  
  
aux\_train$WoE[is.na(aux\_train$WoE)] <- value  
aux\_test$WoE[is.na(aux\_test$WoE)] <- value  
  
colnames(aux\_train)[ncol(aux\_train)]<-paste("WOE\_",variable,sep='')  
colnames(aux\_test)[ncol(aux\_test)]<-paste("WOE\_",variable,sep='')  
  
aux\_train$sample<-"train"  
aux\_test$sample<-"test"  
  
sample<-rbind(aux\_train,aux\_test)  
sample<-sample[,c("id",paste("WOE\_",variable,sep=''))]  
  
return(sample)  
  
  
}  
  
  
  
Groupping\_nominal<-function(variable){  
  
  
name<-as.character(variable)   
aux\_train<-train[,c("id","loan\_status",name)]  
aux\_test<-test[,c("id","loan\_status",name)]  
colnames(aux\_train)[3]<-"var\_name"  
colnames(aux\_test)[3]<-"var\_name"  
aux\_train$loan\_status<-as.numeric(ifelse(train$loan\_status==1,0,1))  
aux\_test$loan\_status<-as.numeric(ifelse(test$loan\_status==1,0,1))  
aux\_train$var\_name<-as.factor(aux\_train$var\_name)  
aux\_test$var\_name<-as.factor(aux\_test$var\_name)  
groupping<-smbinning.factor(df=aux\_train,y="loan\_status",x="var\_name",maxcat = 8)  
  
table\_woe<-as.data.frame(groupping$ivtable)   
 for (i in 1:nrow(table\_woe)){  
   
 table\_woe$ID[i]<-i  
   
 }  
table\_woe$ID<-paste("0",table\_woe$ID,sep='')  
  
  
aux\_train<-smbinning.factor.gen(aux\_train,groupping,"ID")  
  
aux\_test<-smbinning.factor.gen(aux\_test,groupping,"ID")  
  
  
  
table\_woe$ID<-paste(table\_woe$ID,table\_woe$Cutpoint,sep=' ')  
  
table\_woe<-table\_woe[,c("ID","Cutpoint","WoE")]  
  
aux\_train<-merge(aux\_train,table\_woe,by.x=c("ID"),all.x=TRUE)  
aux\_test<-merge(aux\_test,table\_woe,by.x=c("ID"),all.x=TRUE)  
  
  
miss\_woe<-table\_woe[grep("Missing",table\_woe$Cutpoint),]  
  
if (is.na(miss\_woe$WoE)==FALSE){  
  
value<- as.numeric(miss\_woe$WoE)  
#value<-as.numeric(gsub("NaN",0,value))  
   
} else { value <-0 }  
  
aux\_train$WoE[is.na(aux\_train$WoE)] <- value  
aux\_test$WoE[is.na(aux\_test$WoE)] <- value  
  
colnames(aux\_train)[ncol(aux\_train)]<-paste("WOE\_",variable,sep='')  
colnames(aux\_test)[ncol(aux\_test)]<-paste("WOE\_",variable,sep='')  
  
aux\_train$sample<-"train"  
aux\_test$sample<-"test"  
  
sample<-rbind(aux\_train,aux\_test)  
sample<-sample[,c("id",paste("WOE\_",variable,sep=''))]  
  
return(sample)  
  
  
}

Hacemos el agrupamiento aplicando las funciones anteriores:

train<-merge(train,Groupping\_numeric("Fico\_score"),by.x=c("id"))

## Warning in smbinning\_calc(df = aux\_train, y = "loan\_status", x = "var\_name", :  
## NAs introducidos por coerción

train<-merge(train,Groupping\_numeric("revol\_util"),by.x=c("id"))  
train<-merge(train,Groupping\_numeric("annual\_inc"),by.x=c("id"))

## Warning in smbinning\_calc(df = aux\_train, y = "loan\_status", x = "var\_name", :  
## NAs introducidos por coerción

train<-merge(train,Groupping\_numeric("dti"),by.x=c("id"))

## Warning in smbinning\_calc(df = aux\_train, y = "loan\_status", x = "var\_name", :  
## NAs introducidos por coerción

train<-merge(train,Groupping\_numeric("ODF\_addr\_state"),by.x=c("id"))

## Warning in smbinning\_calc(df = aux\_train, y = "loan\_status", x = "var\_name", :  
## NAs introducidos por coerción

train<-merge(train,Groupping\_nominal("grade"),by.x=c("id"))  
train<-merge(train,Groupping\_nominal("term"),by.x=c("id"))  
train<-merge(train,Groupping\_nominal("ODF\_purpose"),by.x=c("id"))  
train<-merge(train,Groupping\_nominal("loan\_amnt2"),by.x=c("id"))  
train<-merge(train,Groupping\_nominal("flag\_cliente"),by.x=c("id"))  
  
test<-merge(test,Groupping\_numeric("Fico\_score"),by.x=c("id"))

## Warning in smbinning\_calc(df = aux\_train, y = "loan\_status", x = "var\_name", :  
## NAs introducidos por coerción

test<-merge(test,Groupping\_numeric("revol\_util"),by.x=c("id"))  
test<-merge(test,Groupping\_numeric("annual\_inc"),by.x=c("id"))

## Warning in smbinning\_calc(df = aux\_train, y = "loan\_status", x = "var\_name", :  
## NAs introducidos por coerción

test<-merge(test,Groupping\_numeric("dti"),by.x=c("id"))

## Warning in smbinning\_calc(df = aux\_train, y = "loan\_status", x = "var\_name", :  
## NAs introducidos por coerción

test<-merge(test,Groupping\_numeric("ODF\_addr\_state"),by.x=c("id"))

## Warning in smbinning\_calc(df = aux\_train, y = "loan\_status", x = "var\_name", :  
## NAs introducidos por coerción

test<-merge(test,Groupping\_nominal("grade"),by.x=c("id"))  
test<-merge(test,Groupping\_nominal("term"),by.x=c("id"))  
test<-merge(test,Groupping\_nominal("ODF\_purpose"),by.x=c("id"))  
test<-merge(test,Groupping\_nominal("loan\_amnt2"),by.x=c("id"))  
test<-merge(test,Groupping\_nominal("flag\_cliente"),by.x=c("id"))

Vemos las nuevas variables creadas en las muestras:

colnames(train)

## [1] "id" "purpose" "addr\_state"   
## [4] "loan\_amnt" "term" "int\_rate"   
## [7] "installment" "grade" "emp\_title"   
## [10] "emp\_length" "home\_ownership" "annual\_inc"   
## [13] "verification\_status" "issue\_d" "loan\_status"   
## [16] "desc" "dti" "delinq\_2yrs"   
## [19] "Fico\_score" "inq\_last\_6mths" "mths\_since\_last\_delinq"  
## [22] "mths\_since\_last\_record" "open\_acc" "revol\_bal"   
## [25] "revol\_util" "ant\_cliente" "flag\_cliente"   
## [28] "Default2" "ODF\_addr\_state" "ODF\_purpose"   
## [31] "loan\_amnt2" "WOE\_Fico\_score" "WOE\_revol\_util"   
## [34] "WOE\_annual\_inc" "WOE\_dti" "WOE\_ODF\_addr\_state"   
## [37] "WOE\_grade" "WOE\_term" "WOE\_ODF\_purpose"   
## [40] "WOE\_loan\_amnt2" "WOE\_flag\_cliente"

colnames(test)

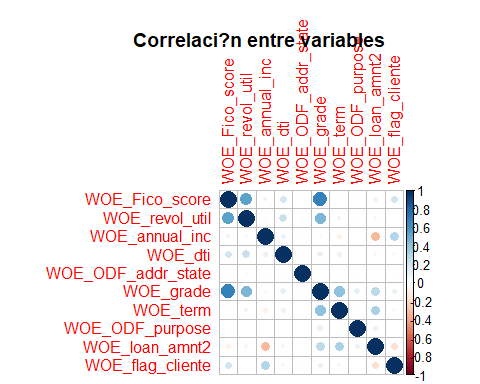
## [1] "id" "purpose" "addr\_state"   
## [4] "loan\_amnt" "term" "int\_rate"   
## [7] "installment" "grade" "emp\_title"   
## [10] "emp\_length" "home\_ownership" "annual\_inc"   
## [13] "verification\_status" "issue\_d" "loan\_status"   
## [16] "desc" "dti" "delinq\_2yrs"   
## [19] "Fico\_score" "inq\_last\_6mths" "mths\_since\_last\_delinq"  
## [22] "mths\_since\_last\_record" "open\_acc" "revol\_bal"   
## [25] "revol\_util" "ant\_cliente" "flag\_cliente"   
## [28] "Default2" "ODF\_addr\_state" "ODF\_purpose"   
## [31] "loan\_amnt2" "WOE\_Fico\_score" "WOE\_revol\_util"   
## [34] "WOE\_annual\_inc" "WOE\_dti" "WOE\_ODF\_addr\_state"   
## [37] "WOE\_grade" "WOE\_term" "WOE\_ODF\_purpose"   
## [40] "WOE\_loan\_amnt2" "WOE\_flag\_cliente"

save.image("Backup4.RData")

EJERCICIO 4: ESTIMACIÓN DEL MODELO

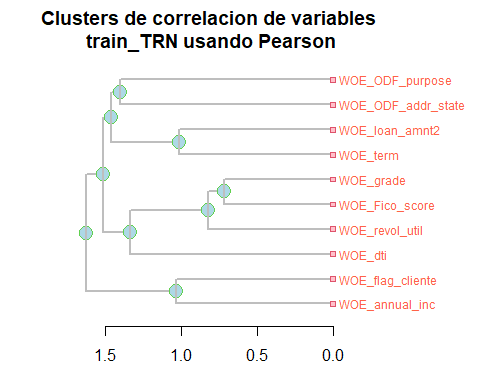
Aunque se ha realizado el análisis individual de cada variable ahora debemos combinarlas para que consigan separar a los clientes buenos de los malos de la mejor manera posible. Se pretende que los modelos sean parsimoniosos, es decir con el menor número posible de variables, para que sea comprensible y, por otro lado que no tenga variables correlacionadas. Vamos a ver las correlaciones existentes:

rm(list=setdiff(ls(), c("Loans","train","test")))  
  
variables\_relevantes<-names(train[,32:41])  
  
# Correlation matrix  
  
correlations<-cor(train[, variables\_relevantes], use="pairwise", method="pearson")  
  
corrplot(correlations, mar=c(0,0,1,0))  
title(main="Correlaci?n entre variables")



como se puede apreciar no hay una correlación muy elevada entre las variables, la variable que más correlación presenta es el Fico Score con el grade o calificación que proporciona LendingClub:

# Generar un cl?ster jer?rquico de variables.  
  
hc <- hclust(dist(correlations), method="average")  
  
# Generar el dendrograma.  
  
dn <- as.dendrogram(hc)  
  
op <- par(mar = c(3, 4, 3, 7.14))  
plot(dn, horiz = TRUE, nodePar = list(col = 3:2, cex = c(2.0, 0.75), pch = 21:22, bg= c("light blue", "pink"), lab.cex = 0.75, lab.col = "tomato"), edgePar = list(col = "gray", lwd = 2), xlab="Height")  
title(main="Clusters de correlacion de variables  
 train\_TRN usando Pearson")



par(op)

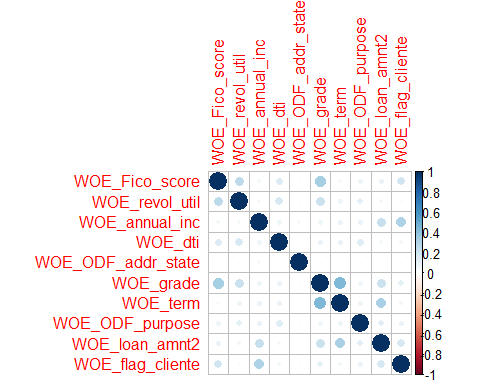
Las correlaciones entre variables no estarían correctamente calculadas ya que lo que tenemos ahora son trasnformaciones de la variable a 4 o 5 valores máximo en media, lo cual aumenta las probabilidades de correlación entre variables).

Por este motivo es más recomendable el uso de la V de Cramer como medida de asociación entre las variables:

get.V<-function(y){  
 library(vcd)  
 col.y<-ncol(y)  
 V<-matrix(ncol=col.y,nrow=col.y)  
 for(i in 1:col.y){  
 for(j in 1:col.y){  
 V[i,j]<-assocstats(table(y[,i],y[,j]))$cramer  
 }  
 }  
 return(V)  
}  
  
  
Cramer\_corr<-get.V(train[,variables\_relevantes])  
colnames(Cramer\_corr)<-colnames(train[,variables\_relevantes])  
rownames(Cramer\_corr)<-colnames(train[,variables\_relevantes])

Se considera que dos variables estan altamente correladas si tienen una V de Cramer superior a 0.4:

corrplot(Cramer\_corr)



Vamos a pasar a hacer diferentes modelos para ver cuál sería nuestro modelo elegido. Al tener 9 variables (el grade lo excluiremos) tendríamos la posibilidad de realizar 2^9 -1 modelos diferentes.

Normalmente, como contamos con un número más elevado de variables no calculamos todos los modelos. En estos casos empleamos una técnica denominada Branch and Bound, que permite calcular los mejores modelos sin necesidad de llevar a cabo cada uno de ellos por separado.

El paquete que permite hacer este procedimiento en R requiere de rJava y en ocasiones no funciona en todos los ordenadores que hemos probado, sobretodo en los ordenadores de empresa donde hay más problemas de permisos.

Por este motivo vamos a calcular todos los modelos posibles esta vez. Para esto vamos a realizar todas las combinaciones de variables solamente de 8 variables (interpretemos que los analistas o el MO del modelo no quiere modelos ni de más ni de menos variables)

Además de hacer las combinaciones, vamos a calcular cuál es el poder predictivo de cada modelo sobre cada una de las muestras (train and test) y vamos a omitir modelos con variables correlacionadas o cuyo coeficiente o signo en el modelo sea contrario a lo esperado:

combinations<-function(variables,comb){  
  
d <- combn(variables,comb) # All combinations 2 variables  
d <-as.data.frame(t(d))  
  
for (i in 1:nrow(d)){  
   
 d$id[i]<-paste(comb,"\_",i,sep='')  
}  
  
return(d)  
}  
  
variables<-names(train[,variables\_relevantes])  
  
  
  
combinats8<-combinations(variables,8)  
  
  
  
hacer\_modelos<-function(modelos){  
  
lista<-as.data.frame(modelos)  
temp<-matrix("NA",0,5)  
temp<-as.data.frame(temp)  
colnames(temp)<-c("id","Gini\_train","Gini\_test","test\_signos","test\_correlaciones")  
  
  
for (j in 1:(nrow(lista))){  
  
  
aux<-as.data.frame((as.data.frame(lista[j,])))  
  
for (i in 1:(ncol(aux))){  
  
  
colnames(aux)[i]<-as.character(aux[1,i])  
  
}  
  
ind<-names(aux[,1:ncol(aux)-1])  
temp2<-matrix("NA",1,5)  
temp2[1,1]<-lista[j,"id"]  
rm(aux)  
  
  
Logit\_model <- glm(loan\_status ~ ., data=train[, c("loan\_status",ind)],  
 family=binomial(link="logit"))  
  
pr\_logit\_train <- predict(Logit\_model, type="response", newdata=train[, c("loan\_status",ind)])  
  
pred <- prediction(pr\_logit\_train, train$loan\_status)  
pe\_Logit <- performance(pred, "tpr", "fpr")  
au\_Logit <- performance(pred, "auc")@y.values[[1]]  
  
temp2[1,2]<-as.numeric(round(2\*au\_Logit-1,4))  
  
  
pr\_logit\_test <- predict(Logit\_model, type="response", newdata=test[, c("loan\_status",ind)])  
  
pred <- prediction(pr\_logit\_test, test$loan\_status)  
pe\_Logit <- performance(pred, "tpr", "fpr")  
au\_Logit <- performance(pred, "auc")@y.values[[1]]  
  
temp2[1,3]<-as.numeric(round(2\*au\_Logit-1,4))  
  
  
coeficientes<-as.data.frame(Logit\_model$coefficient)  
coeficientes<-as.data.frame(coeficientes[coeficientes[,1]!="(Intercept)",])  
temp2[1,4]<-ifelse(max(coeficientes)<0,1,0)  
  
Correlations<-get.V(train[,ind])  
diag(Correlations)<-0  
test\_cor<-ifelse(abs(max(Correlations))<0.4,1,0)  
temp2[1,5]<-test\_cor  
colnames(temp2)<-colnames(temp)  
temp<-rbind(temp,temp2)  
rm(temp2)  
print("Modelo")  
print(j)  
print("de")  
print(nrow(lista))  
}  
return(temp)  
}

Procedemos a realizar los modelos (puede llevar algo de tiempo):

t <- proc.time()  
  
modelos8<-hacer\_modelos(combinats8)  
proc.time()-t   
  
#resultadosModelos<-rbind(modelos6,modelos7,modelos8)  
resultadosModelos<-modelos8  
head(resultadosModelos)  
  
save.image("Backup5.RData")

Vamos a filtrar los modelos que no cumplen las restricciones de signo negativo de los coeficientes (al hacerse la transformación a Woes todos los valores deberían venir en negativo) y de correlaciones, es decir las variables del modelo no deberían estar correlacionadas:

resultadosModelos<-resultadosModelos[resultadosModelos$test\_signos==1 & resultadosModelos$test\_correlaciones==1,]  
  
resultadosModelos$Gini\_train<-as.numeric(as.character(resultadosModelos$Gini\_train))  
resultadosModelos$Gini\_test<-as.numeric(as.character(resultadosModelos$Gini\_test))  
  
resultadosModelos<-resultadosModelos[order(-resultadosModelos$Gini\_train),]

Seleccionamos el mejor modelo:

resultadosModelos<-head(resultadosModelos,1)  
resultadosModelos

## id Gini\_train Gini\_test test\_signos test\_correlaciones  
## 7 8\_7 0.39 0.36 1 1

Vamos a ver que variables están dentro del modelo:

best\_model<-merge(resultadosModelos,combinats8,by.x="id")  
t(best\_model[,6:13])

## [,1]   
## V1 "WOE\_Fico\_score"   
## V2 "WOE\_revol\_util"   
## V3 "WOE\_annual\_inc"   
## V4 "WOE\_dti"   
## V5 "WOE\_ODF\_addr\_state"  
## V6 "WOE\_term"   
## V7 "WOE\_ODF\_purpose"   
## V8 "WOE\_loan\_amnt2"

Estimemos el modelo para ver los coeficientes:

variables<-c("WOE\_Fico\_score","WOE\_revol\_util","WOE\_annual\_inc","WOE\_ODF\_addr\_state","WOE\_term","WOE\_ODF\_purpose","WOE\_loan\_amnt2")  
  
Best\_Logit\_Model <- glm(loan\_status ~ ., data=train[, c("loan\_status",variables)],  
 family=binomial(link="logit"))  
  
summary(Best\_Logit\_Model)

##   
## Call:  
## glm(formula = loan\_status ~ ., family = binomial(link = "logit"),   
## data = train[, c("loan\_status", variables)])  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.480 -0.596 -0.463 -0.325 2.772   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.7996 0.0183 -98.59 < 0.0000000000000002 \*\*\*  
## WOE\_Fico\_score -0.8577 0.0540 -15.89 < 0.0000000000000002 \*\*\*  
## WOE\_revol\_util -0.4136 0.0685 -6.04 0.00000000157039 \*\*\*  
## WOE\_annual\_inc -1.3959 0.0946 -14.76 < 0.0000000000000002 \*\*\*  
## WOE\_ODF\_addr\_state -0.9459 0.1028 -9.20 < 0.0000000000000002 \*\*\*  
## WOE\_term -0.9391 0.0455 -20.66 < 0.0000000000000002 \*\*\*  
## WOE\_ODF\_purpose -0.9737 0.0670 -14.54 < 0.0000000000000002 \*\*\*  
## WOE\_loan\_amnt2 -0.9388 0.1288 -7.29 0.00000000000032 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 22813 on 27849 degrees of freedom  
## Residual deviance: 21166 on 27842 degrees of freedom  
## AIC: 21182  
##   
## Number of Fisher Scoring iterations: 5

save.image("Backup6.RData")

EJERCICIO 5: PUNTOS DE CORTE

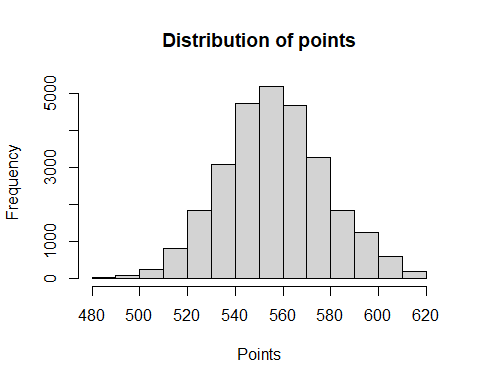
Podemos asignar una probabilidad de acuerdo a nuestro modelo para cada solicitante de nuestras muestras:

options(digits=4)  
  
PD <- predict(Best\_Logit\_Model, type="response", newdata=train[, c("loan\_status",variables)])  
  
# print(PD)  
  
Predictions\_Logit<-cbind(train,PD)

Vamos a transforman nuestro modelo en una tarjeta de puntuación, donde tener 500 puntos implicaría tener una probabilidad de incumpliento del 50%. Es decir existe un mal cliente por cada uno bueno.

Además, cada 20 puntos, se doblan los odds, es decir en los 520 puntos habría 2 clientes buenos por cada uno malo.

Puntos<-500  
Odds<-1  
Puntos\_doblar\_odds<-20  
  
Factor<-Puntos\_doblar\_odds/log(2)  
  
Offset<-Puntos-(Factor\*log(Odds))  
  
Predictions\_Logit$Points<-round(Offset+(log((1-PD)/PD))\*Factor,0)  
  
hist(Predictions\_Logit$Points,main="Distribution of points",xlab="Points")

 Imaginaros que el modelo antiguo, el cuál será remplazado por este modelo, aprueba automáticamente el 80% de las solicitudes de préstamos con un bad rate del 15%.

Vamos a partir nuestro dataset en 20 puntos de corte de puntuación y analizamos como podemos usar este modelo para sustituir al modelo teórico anterior y además controlar el % de admisión y la tasa de incumplimiento esperada:

Predictions\_Logit$bins <- cut(Predictions\_Logit$Points, 20, include.lowest=TRUE)  
  
aux<-sqldf("SELECT bins,  
 sum(loan\_status) as DEFAULTS,  
 count(loan\_status) as N,  
 avg(PD) as Avg\_pred\_PD  
 from Predictions\_Logit group by bins")  
  
aux$FDO<-aux$DEFAULTS/aux$N  
  
for (i in 1:nrow(aux)){  
 ifelse(i==1,aux$N\_acum[i]<-aux$N[i],aux$N\_acum[i]<-aux$N[i]+aux$N\_acum[i-1])  
 ifelse(i==1,aux$Def\_acum[i]<-aux$DEFAULTS[i],aux$Def\_acum[i]<-aux$DEFAULTS[i]+aux$Def\_acum[i-1])  
  
}  
  
aux$aceptance\_above<-1-(aux$N\_acum/sum(aux$N))  
defaults<-sum(aux$DEFAULTS)  
registros<-nrow(Predictions\_Logit)  
  
aux$defaults\_above<-(defaults-aux$Def\_acum)/registros  
  
aux<-aux[,c("bins","aceptance\_above","defaults\_above")]  
aux2<-sqldf("SELECT bins,  
 max(PD) as PD\_superior,  
 max(Points) as Points\_superior  
 from Predictions\_Logit group by bins")  
  
aux<-merge(aux,aux2,by.x="bins")

¿Qué quiere decir esta tabla que hemos obtenido? Veamos el primer punto: la columna PD\_superior es el punto de corte establecido en ese tramo. También encontramos los puntos equivalentes en el tramo.

Concretamente si rechazáramos todos los préstamos de solicitantes con un score inferior a 543 puntos, estaríamos aceptando un 75.22% de los solicitantes, lo que estaría generando una tasa de defaults del 7.69%.

En base a la información anterior nos planteamos una serie de preguntas:

1- Si queremos mantener la misma tasas de aceptación actual, ¿cuál sería el punto de corte que propondríais?

2- Y si queremos mantener la misma tasa de defaults como hasta ahora, ¿qué punto de corte podría establecerse?

muestra\_desarrollo<-cbind(train,Predictions\_Logit[,42:44])  
rm(list=setdiff(ls(), "muestra\_desarrollo"))  
  
save.image("Backup7.RData")

EJERCICIO 6: MUESTRAS OOT AND TTD

library(Hmisc)  
  
OOT\_sample <- read.csv("OOT\_sample.csv", stringsAsFactors=FALSE)  
  
unique(OOT\_sample$issue\_d)

## [1] "6/1/2012"

print("El poder predictivo del modelo en la muestra de desarrollo era:")

## [1] "El poder predictivo del modelo en la muestra de desarrollo era:"

(2\*rcorr.cens(muestra\_desarrollo$PD,muestra\_desarrollo$loan\_status)[1])-1

## C Index   
## 0.3875

print("El poder predictivo del modelo en la muestra OOT es:")

## [1] "El poder predictivo del modelo en la muestra OOT es:"

(2\*rcorr.cens(OOT\_sample$PD,OOT\_sample$loan\_status)[1])-1

## C Index   
## 0.3702

Parece que el poder predictivo en una muestra un poco mas reciente a la que se desarrollo el modelo, parece seguir siendo valido. Podemos evaluar si hay signos de fatiga en el modelo o de que no vaya a funcionar bien en una muestra todav?a mas reciente pero en la que aun no hay desempeño, es decir no podemos ver cual es el resultado de pago impago de los prestamos.

Se analiza solamente la estabilidad a traves de la metrica PSI Population Stability Index. Valores entre 0.1 y 0.25 indican que se ha producido una variacion leve en la distribucion de la poblacion en comparacion con la muestra de desarrollo. VAlores m?s altos que 0.25 indican que la distribucion de la poblacion para esa variable se ha modificado drasticamente.

Primero cargamos nuestra muestra TTD:

TTD\_sample <- read.csv("TTD\_sample.csv", stringsAsFactors=FALSE)  
  
unique(TTD\_sample$issue\_d)

## [1] "6/1/2013"

psi<-function(muestra\_des,muestra\_actual,variable){  
  
a1<-data.frame(table(muestra\_des[,c(variable)]))  
a2<-data.frame(table(muestra\_actual[,c(variable)]))  
  
a3<-merge(a1,a2,by=c("Var1"))  
colnames(a3)<-c(var,"des","actual")  
a3$perc\_des<-a3$des/sum(a3$des)  
a3$perc\_actual<-a3$actual/sum(a3$actual)  
a3$psi\_index<-(a3$perc\_actual-a3$perc\_des)\*log(a3$perc\_actual/a3$perc\_des)  
psi<-sum(a3$psi\_index)  
  
return(psi)  
  
}  
  
# ggplot(a3, aes(length, fill = veg)) + geom\_density(alpha = 0.2)

#   
# muestra\_des <- muestra\_desarrollo  
# muestra\_actual <- TTD\_sample  
# variable <- "WOE\_revol\_util"  
#   
# a1<-data.frame(table(muestra\_des[,c(variable)]))  
# a2<-data.frame(table(muestra\_actual[,c(variable)]))  
#   
# a3<-merge(a1,a2,by=c("Var1"))  
# colnames(a3)<-c("var","des","actual")  
# a3$perc\_des<-a3$des/sum(a3$des)  
# a3$perc\_actual<-a3$actual/sum(a3$actual)  
# a3$psi\_index<-(a3$perc\_actual-a3$perc\_des)\*log(a3$perc\_actual/a3$perc\_des)  
# psi\_value<-sum(a3$psi\_index)  
# psi\_value

Vamos a calcular el PSI para todas las variables del modelo:

print("WOE\_Fico\_score:")

## [1] "WOE\_Fico\_score:"

psi(muestra\_desarrollo,TTD\_sample,"WOE\_Fico\_score")

## [1] 0.1039

print("WOE\_revol\_util:")

## [1] "WOE\_revol\_util:"

psi(muestra\_desarrollo,TTD\_sample,"WOE\_revol\_util")

## [1] 0.2677

print("WOE\_annual\_inc:")

## [1] "WOE\_annual\_inc:"

psi(muestra\_desarrollo,TTD\_sample,"WOE\_annual\_inc")

## [1] 0.00347

print("WOE\_ODF\_addr\_state:")

## [1] "WOE\_ODF\_addr\_state:"

psi(muestra\_desarrollo,TTD\_sample,"WOE\_ODF\_addr\_state")

## [1] 0.005697

print("WOE\_term:")

## [1] "WOE\_term:"

psi(muestra\_desarrollo,TTD\_sample,"WOE\_term")

## [1] 0.04987

print("WOE\_ODF\_purpose:")

## [1] "WOE\_ODF\_purpose:"

psi(muestra\_desarrollo,TTD\_sample,"WOE\_ODF\_purpose")

## [1] 0.167

print("WOE\_loan\_amnt2:")

## [1] "WOE\_loan\_amnt2:"

psi(muestra\_desarrollo,TTD\_sample,"WOE\_loan\_amnt2")

## [1] 0.06478

En este caso vemos que la única variable con un PSI elevado sería la WOE\_revol\_util con un 26% de PSI que indicaría que no se aprecia estabilidad en la misma. En este caso, lo que tendríamos que hacer ya que el modelo funciona bien en muestras OOT (no varía el poder predictivo) sería realizar un seguimiento cercano al modelo sobre todo para ver si más variables dejan de ser estables.