R_SVM

Ulises Jose Bustamante Mora

2023-10-28

Importing the dataset

```
loan = read.csv("loan_data.csv")
head(loan)
```

```
##   credit.policy             purpose int.rate installment log.annual.inc
dti
## 1             1 debt_consolidation   0.1189      829.10       11.35041
19.48
## 2             1        credit_card   0.1071      228.22       11.08214
14.29
## 3             1 debt_consolidation   0.1357      366.86       10.37349
11.63
## 4             1 debt_consolidation   0.1008      162.34       11.35041
8.10
## 5             1        credit_card   0.1426      102.92       11.29973
14.97
## 6             1        credit_card   0.0788      125.13       11.90497
16.98
##   fico days.with.cr.line revol.bal revol.util inq.last.6mths
delinq.2yrs
## 1  737          5639.958     28854       52.1              0
0
## 2  707          2760.000     33623       76.7              0
0
## 3  682          4710.000      3511       25.6              1
0
## 4  712          2699.958     33667       73.2              1
0
## 5  667          4066.000      4740       39.5              0
1
## 6  727          6120.042     50807       51.0              0
0
##   pub.rec not.fully.paid
## 1       0              0
## 2       0              0
## 3       0              0
## 4       0              0
## 5       0              0
## 6       0              0
```

Data transformation

Changing into factors some variables

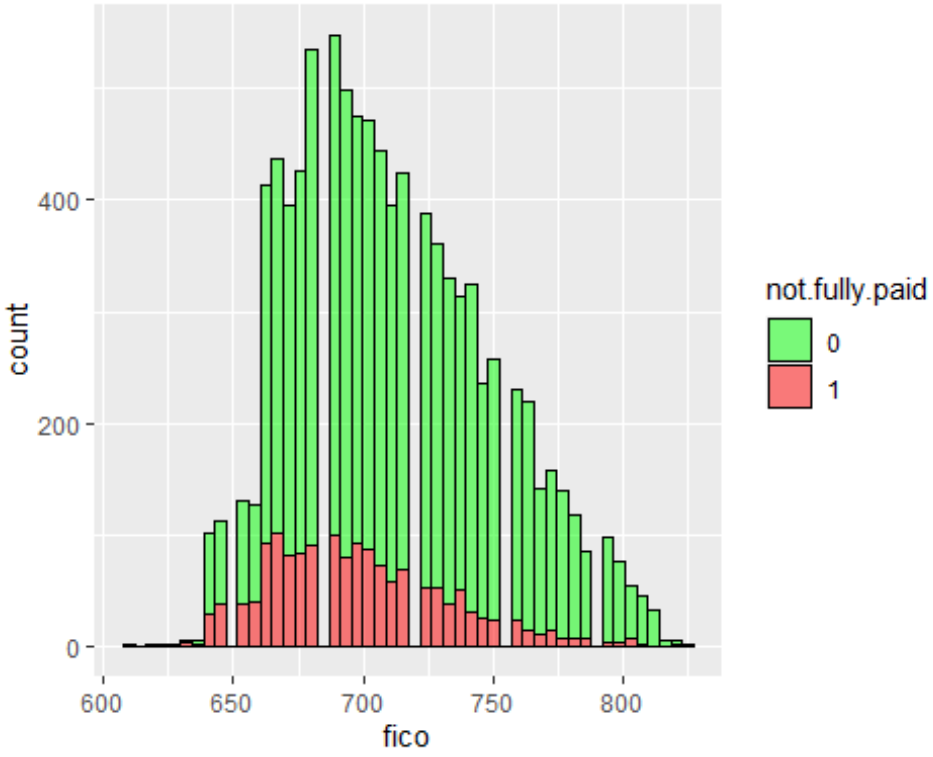
```
loan$credit.policy = factor(loan$credit.policy)
loan$inq.last.6mths = factor(loan$inq.last.6mths)
loan$delinq.2yrs = factor(loan$delinq.2yrs)
loan$pub.rec = factor(loan$pub.rec)
loan$not.fully.paid = factor(loan$not.fully.paid)

str(loan)
```

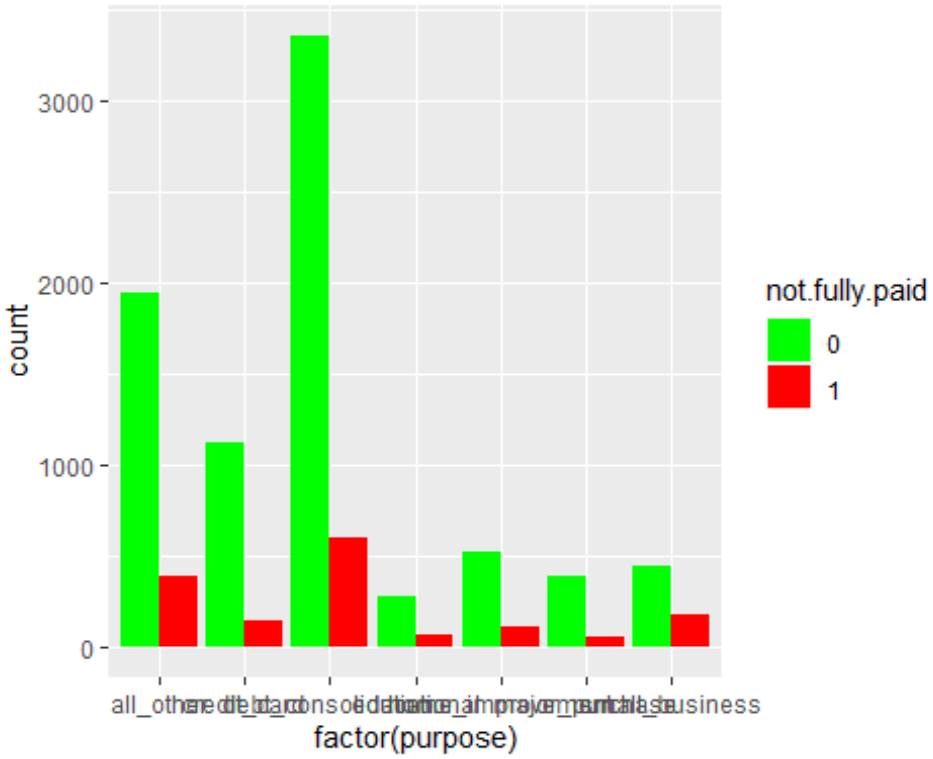
```
## 'data.frame':    9578 obs. of  14 variables:
##  $ credit.policy    : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2
...
##  $ purpose          : chr  "debt_consolidation" "credit_card"
"debt_consolidation" "debt_consolidation" "debt_consolidation" ...
##  $ int.rate         : num  0.119 0.107 0.136 0.101 0.143 ...
##  $ installment      : num  829 228 367 162 103 ...
##  $ log.annual.inc   : num  11.4 11.1 10.4 11.4 11.3 ...
##  $ dti              : num  19.5 14.3 11.6 8.1 15 ...
##  $ fico             : int  737 707 682 712 667 727 667 722 682 707 ...
##  $ days.with.cr.line: num  5640 2760 4710 2700 4066 ...
##  $ revol.bal        : int  28854 33623 3511 33667 4740 50807 3839
24220 69909 5630 ...
##  $ revol.util       : num  52.1 76.7 25.6 73.2 39.5 51 76.8 68.6 51.1
23 ...
##  $ inq.last.6mths   : Factor w/ 28 levels "0","1","2","3",..: 1 1 2 2
1 1 1 1 2 2 ...
##  $ delinq.2yrs      : Factor w/ 11 levels "0","1","2","3",..: 1 1 1 1
2 1 1 1 1 1 ...
##  $ pub.rec          : Factor w/ 6 levels "0","1","2","3",..: 1 1 1 1 1
1 2 1 1 1 ...
##  $ not.fully.paid   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 2 1 1
...
```

EDA

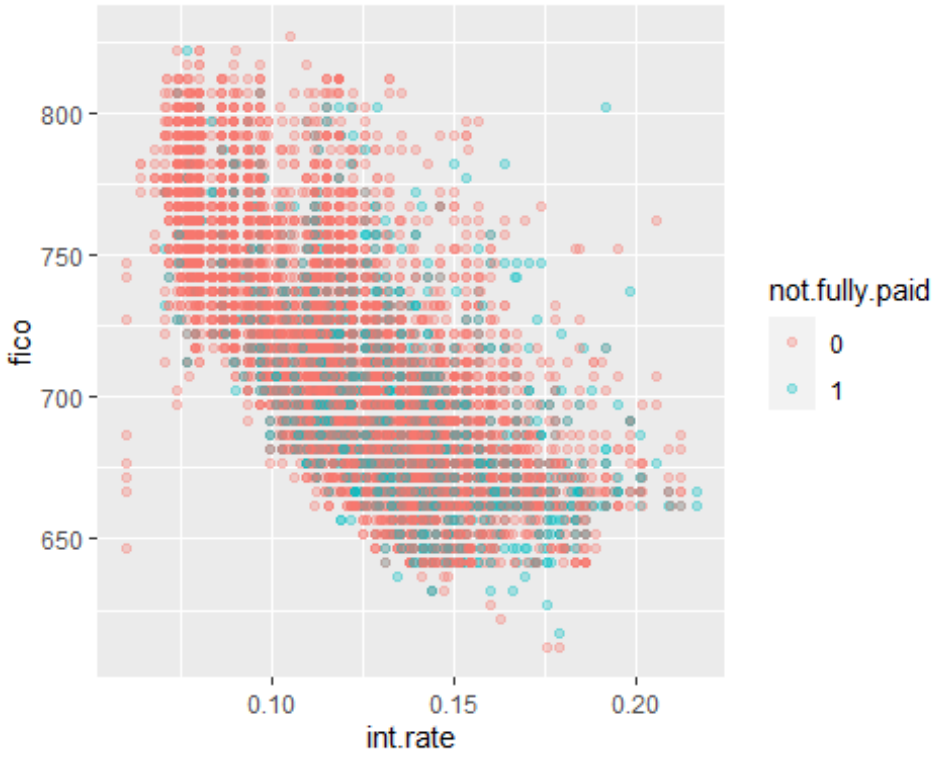
```
library(ggplot2)
ggplot(loan, aes(fico)) + geom_histogram(aes(fill = not.fully.paid),
color = "black", bins = 50, alpha = 0.5) + scale_fill_manual(values =
c("green", "red"))
```




```
ggplot(loan, aes(x = factor(purpose))) + geom_bar(aes(fill =
not.fully.paid), position = "dodge") + scale_fill_manual(values =
c("green", "red"))
```



```
ggplot(loan, aes(x = int.rate, y = fico)) +
geom_point(aes(color=not.fully.paid),alpha = 0.3)
```



Starting with the model creation

Splitting the data set

```
library(caTools)
sample = sample.split(loan$not.fully.paid, 0.7)
loanTrain = subset(loan, sample == T)
loanTest = subset(loan, sample == F)

print(dim(loanTest))

## [1] 2873   14

print(dim(loanTrain))

## [1] 6705   14
```

Creating the model

```
library(e1071)
model = svm(not.fully.paid~., data = loanTrain)
summary(model)

##
## Call:
## svm(formula = not.fully.paid ~ ., data = loanTrain)
```

```
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  2773
##
##  ( 1700 1073 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
```

Predictions

```
pre = predict(model, loanTest[1:13])
table(pre, loanTest$not.fully.paid)
```

```
##
## pre    0    1
##   0 2413  460
##   1    0    0
```

We got bad results because the gamma value was not the correct one. So let´s find out which is the best value.

```
tuned = tune(svm, train.x = not.fully.paid~., data = loanTrain,
             kernel = "radial",
             ranges = list(cost=c(100,200), gamma = c(0.1)))

summary(tuned)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##   100   0.1
##
## - best performance: 0.2067178
##
## - Detailed performance results:
##   cost gamma     error dispersion
## 1  100   0.1 0.2067178 0.01478311
## 2  200   0.1 0.2201408 0.01451925
```

```
#cost gamma
#100    0.1
```

Creadting again the model but with the best parameters.

```
tunedModel = svm(not.fully.paid~., data = loanTrain, cost = 100, gamma =
0.1)

tunedPre = predict(tunedModel, loanTest[1:13])

table(tunedPre, loanTest$not.fully.paid)

##
## tunedPre    0    1
##        0 2200  385
##        1  213   75
```