Credit_Validate.R

rodrigolima82

2019-05-30

```r
# Experimento DSA - Data Science Academy

# Carrega o dataset antes da transformacao
df <- read.csv("credito.csv")
#View(df)
str(df)
```

```
## 'data.frame':    999 obs. of  21 variables:
##  $ A11  : Factor w/ 4 levels "A11","A12","A13",..: 2 4 1 1 4 4 2 4 2 2 ...
##  $ X6   : int  48 12 42 24 36 24 36 12 30 12 ...
##  $ A34  : Factor w/ 5 levels "A30","A31","A32",..: 3 5 3 4 3 3 3 3 5 3 ...
##  $ A43  : Factor w/ 10 levels "A40","A41","A410",..: 5 8 4 1 8 4 2 5 1 1 ...
##  $ X1169: int  5951 2096 7882 4870 9055 2835 6948 3059 5234 1295 ...
##  $ A65  : Factor w/ 5 levels "A61","A62","A63",..: 1 1 1 1 5 3 1 4 1 1 ...
##  $ A75  : Factor w/ 5 levels "A71","A72","A73",..: 3 4 4 3 3 5 3 4 1 2 ...
##  $ X4   : int  2 2 2 3 2 3 2 2 4 3 ...
##  $ A93  : Factor w/ 4 levels "A91","A92","A93",..: 2 3 3 3 3 3 3 1 4 2 ...
##  $ A101 : Factor w/ 3 levels "A101","A102",..: 1 1 3 1 1 1 1 1 1 1 ...
##  $ X4.1 : int  2 3 4 4 4 4 2 4 2 1 ...
##  $ A121 : Factor w/ 4 levels "A121","A122",..: 1 1 2 4 4 2 3 1 3 3 ...
##  $ X67  : int  22 49 45 53 35 53 35 61 28 25 ...
##  $ A143 : Factor w/ 3 levels "A141","A142",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ A152 : Factor w/ 3 levels "A151","A152",..: 2 2 3 3 3 2 1 2 2 1 ...
##  $ X2   : int  1 1 1 2 1 1 1 1 2 1 ...
##  $ A173 : Factor w/ 4 levels "A171","A172",..: 3 2 3 3 2 3 4 2 4 3 ...
##  $ X1   : int  1 2 2 2 2 1 1 1 1 1 ...
##  $ A192 : Factor w/ 2 levels "A191","A192": 1 1 1 1 2 1 2 1 1 1 ...
##  $ A201 : Factor w/ 2 levels "A201","A202": 1 1 1 1 1 1 1 1 1 1 ...
##  $ X1.1 : int  2 1 1 2 1 1 1 1 2 2 ...
```

```r
# Nome das variáveis
# CheckingAcctStat, Duration, CreditHistory, Purpose, CreditAmount, SavingsBonds, Employment, Installme

# Aplicando Engenharia de Atributos em Variáveis Numéricas
source("src/ClassTools.R")
Credit <- read.csv("credito.csv", header = F, stringsAsFactors = F )
metaFrame <- data.frame(colNames, isOrdered, I(factOrder))
Credit <- fact.set(Credit, metaFrame)

# Balancear o número de casos positivos e negativos
Credit <- equ.Frame(Credit, 2)

# Transformando variáveis numéricas em variáveis categóricas
toFactors <- c("Duration", "CreditAmount", "Age")
maxVals <- c(100, 1000000, 100)
facNames <- unlist(lapply(toFactors, function(x) paste(x, "_f", sep = "")))
Credit[, facNames] <- Map(function(x, y) quantize.num(Credit[, x], maxval = y), toFactors, maxVals)
```

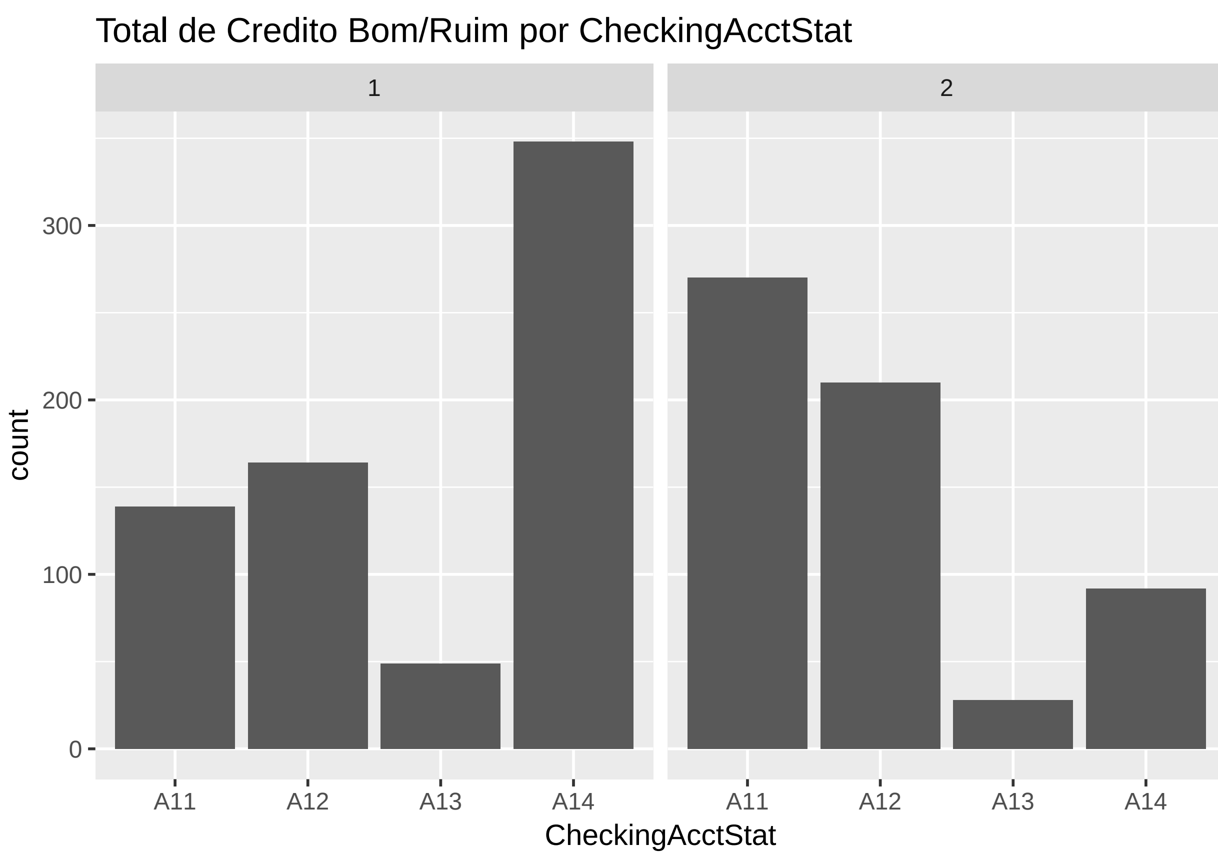
```
## [1]   0.0  17.6  31.2  44.8  58.4 100.0
## [1]       0.0    3884.8    7519.6   11154.4   14789.2 1000000.0
## [1]   0.0  30.2  41.4  52.6  63.8 100.0
```

```
# Análise Exploratória de Dados
# Plots usando ggplot2
library(ggplot2)
```

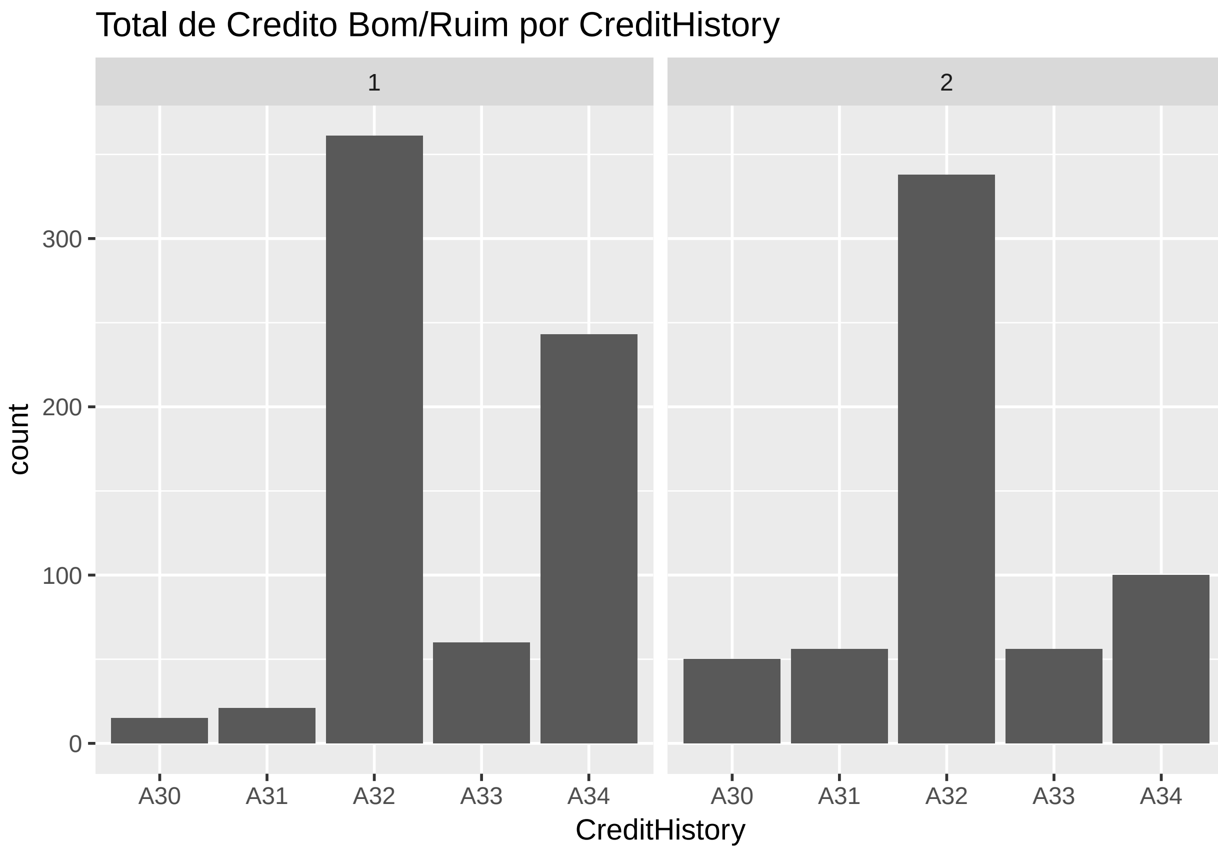
```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
lapply(colNames2, function(x){
  if(is.factor(Credit[,x])) {
    ggplot(Credit, aes_string(x)) +
      geom_bar() +
      facet_grid(. ~ CreditStatus) +
      ggtitle(paste("Total de Credito Bom/Ruim por",x))}})
```

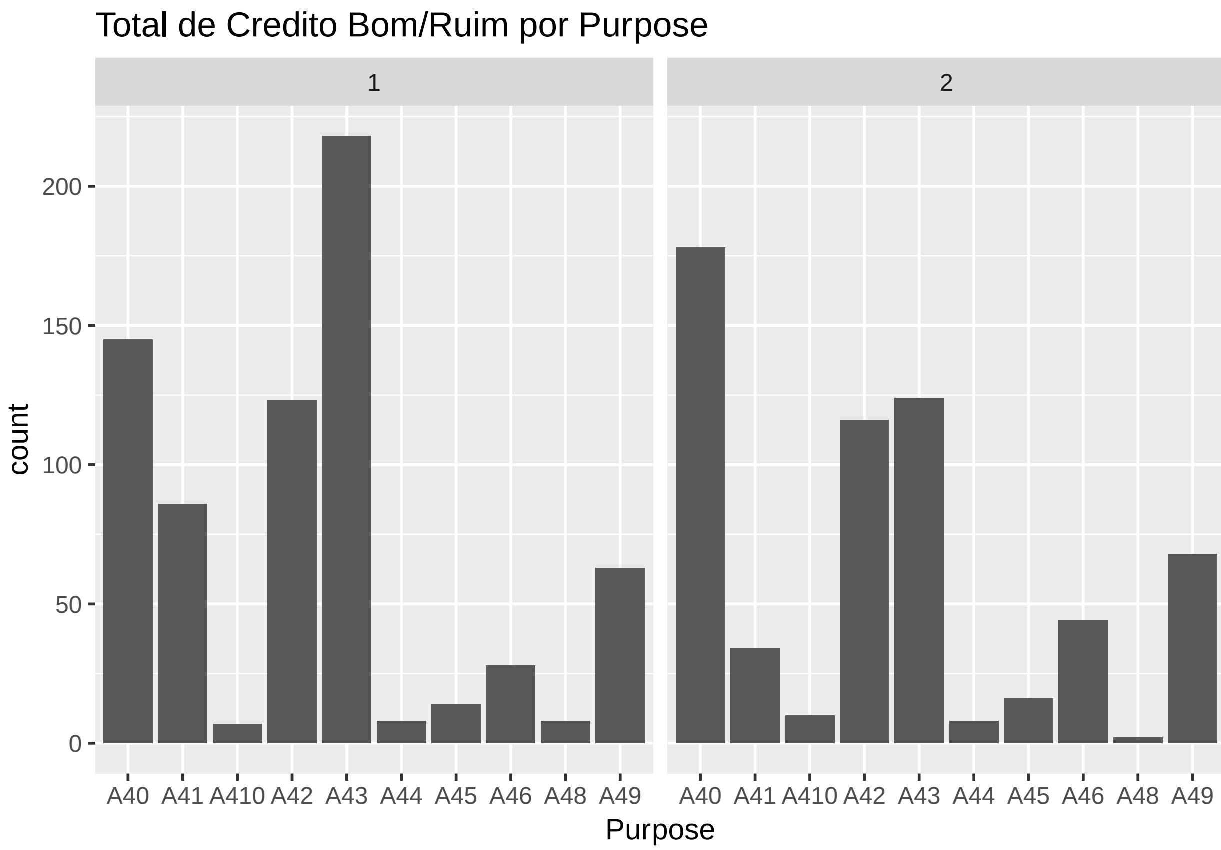
```
## [[1]]
```




```
##
## [[2]]
## NULL
##
## [[3]]
```

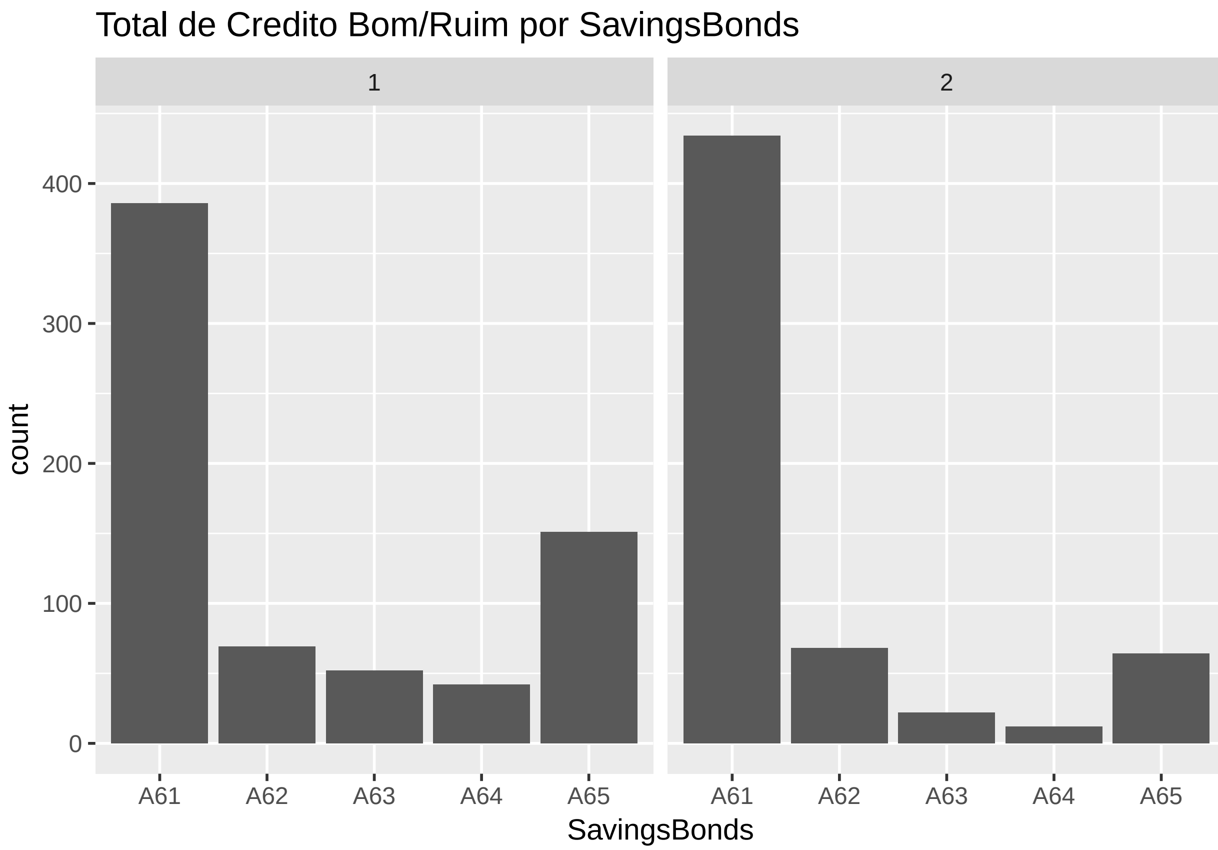

Total de Credito Bom/Ruim por CreditHistory
1
2
count
300
200
100
0
A30
A31
A32
A33
A34
A30
A31
A32
A33
A34
CreditHistory

```
##
## [[4]]
```

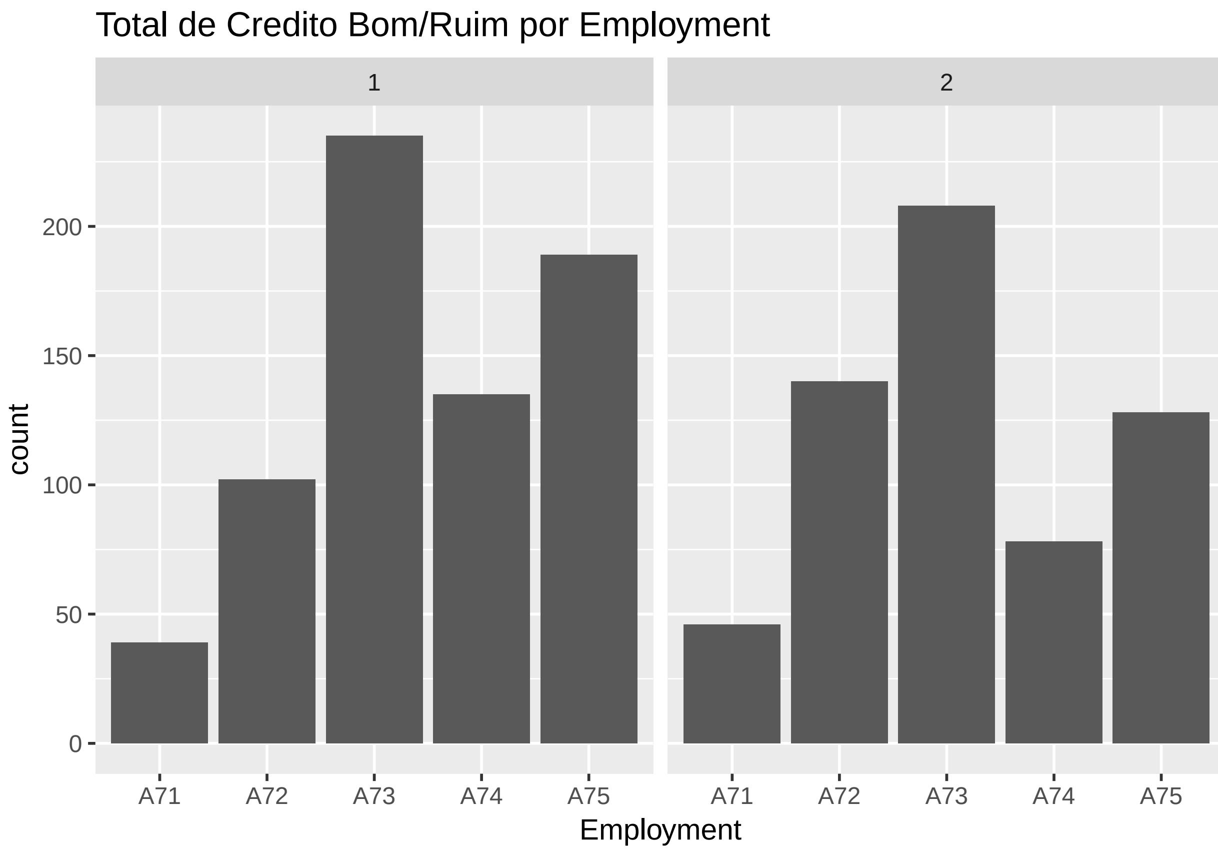


Total de Credito Bom/Ruim por Purpose
1
2
count
200
150
100
50
0
A40
A41
A410
A42
A43
A44
A45
A46
A48
A49
Purpose

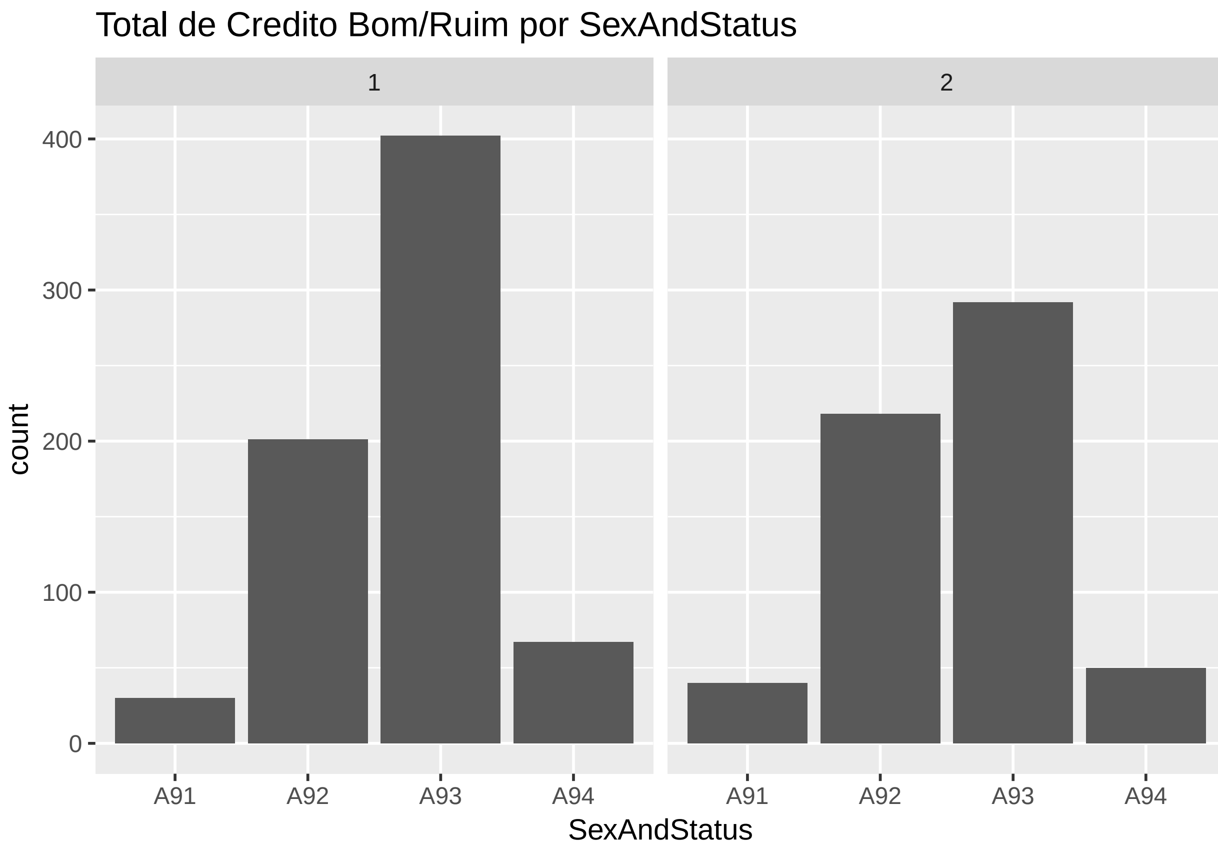

```
##
## [[5]]
## NULL
##
## [[6]]
```


Total de Credito Bom/Ruim por SavingsBonds
1
2
count
400
300
200
100
0
A61
A62
A63
A64
A65
A61
A62
A63
A64
A65
SavingsBonds

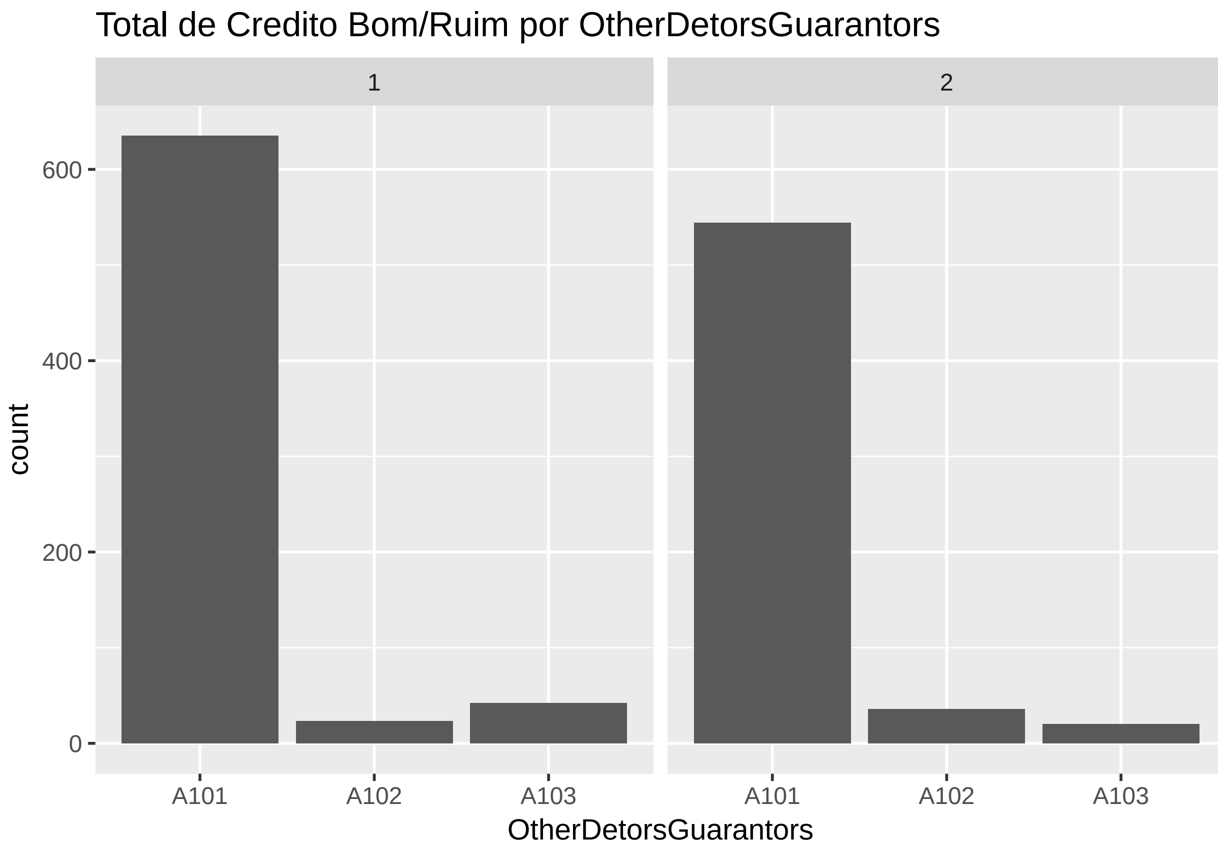
```
##
## [[7]]
```


Total de Credito Bom/Ruim por Employment
1
2
count
200
150
100
50
0
A71
A72
A73
A74
A75
A71
A72
A73
A74
A75
Employment

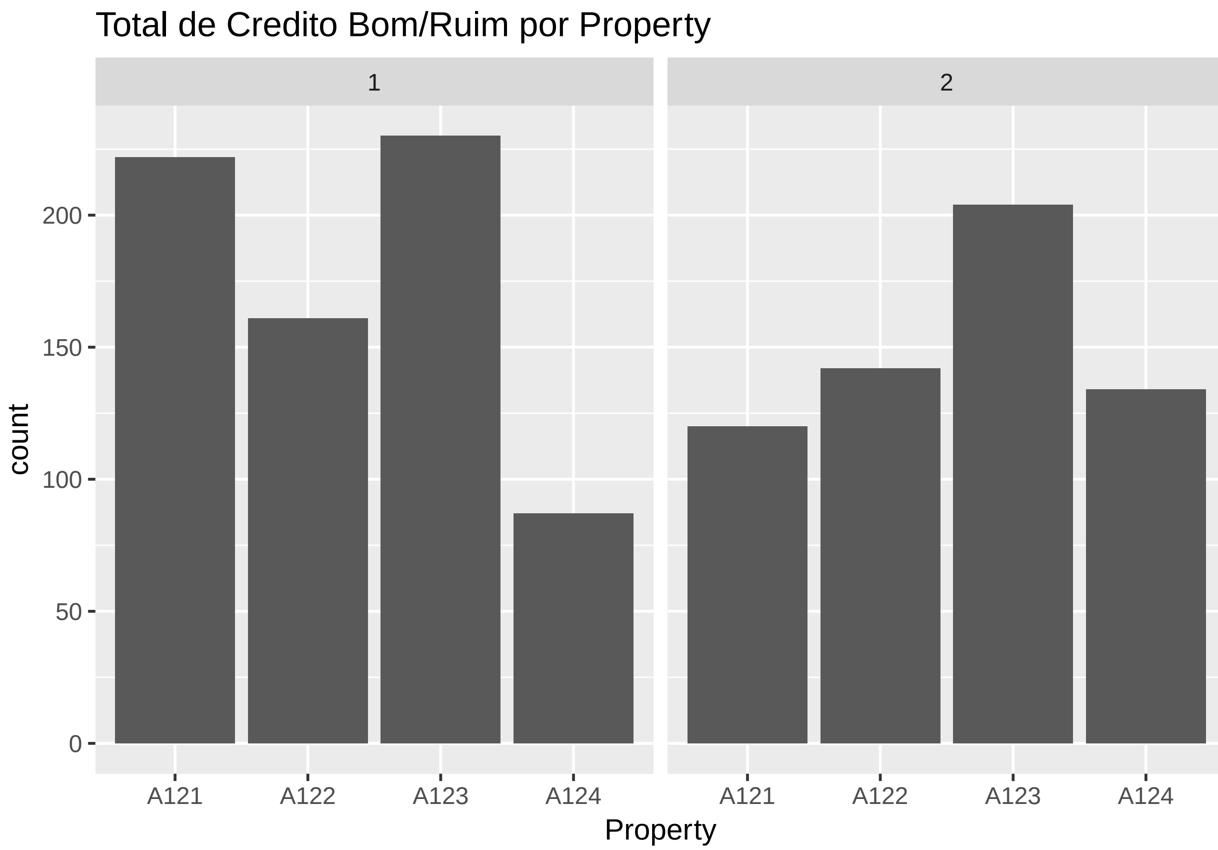
```
##
## [[8]]
## NULL
##
## [[9]]
```


Total de Credito Bom/Ruim por SexAndStatus
1
2
400
300
200
100
0
count
A91
A92
A93
A94
A91
A92
A93
A94
SexAndStatus

```
##
## [[10]]
```

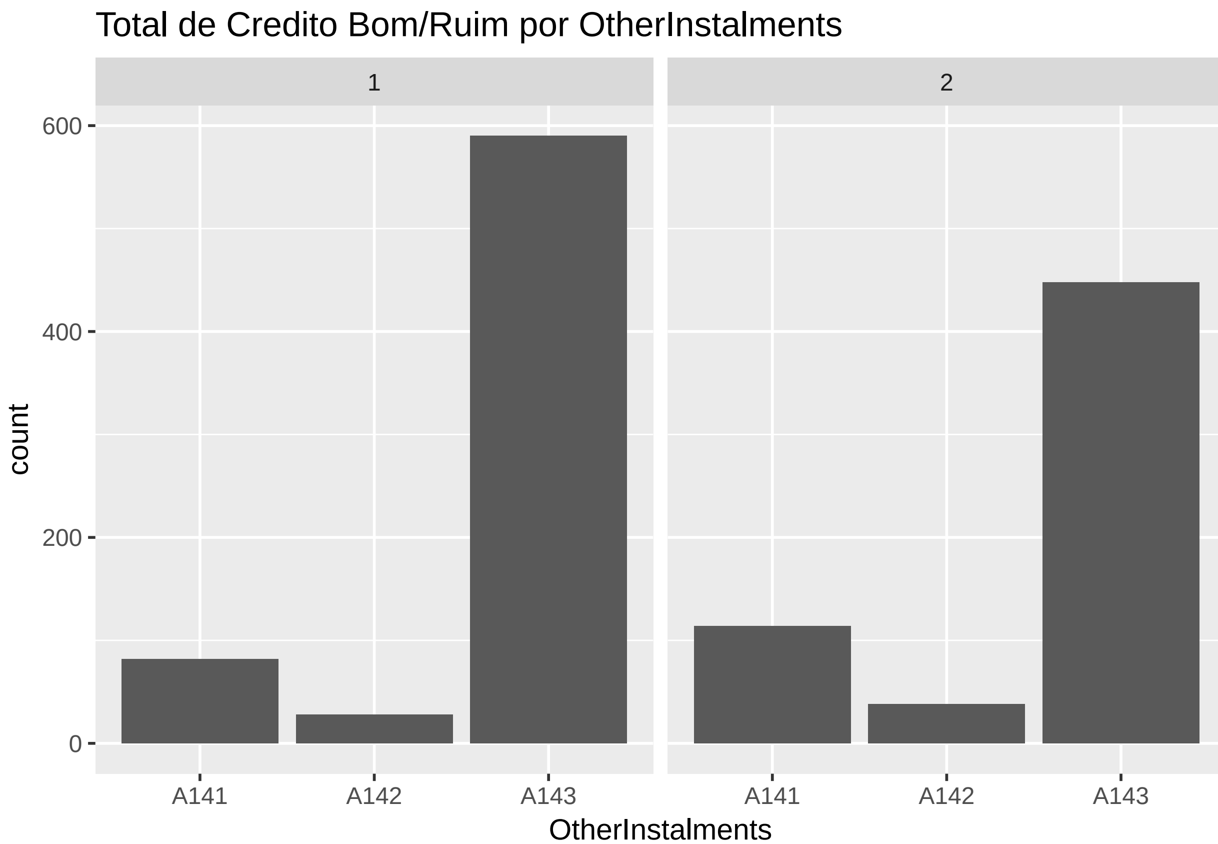

Total de Credito Bom/Ruim por OtherDetorsGuarantors
1
2
600
400
200
0
count
A101
A102
A103
A101
A102
A103
OtherDetorsGuarantors

```
##
## [[11]]
## NULL
##
## [[12]]
```



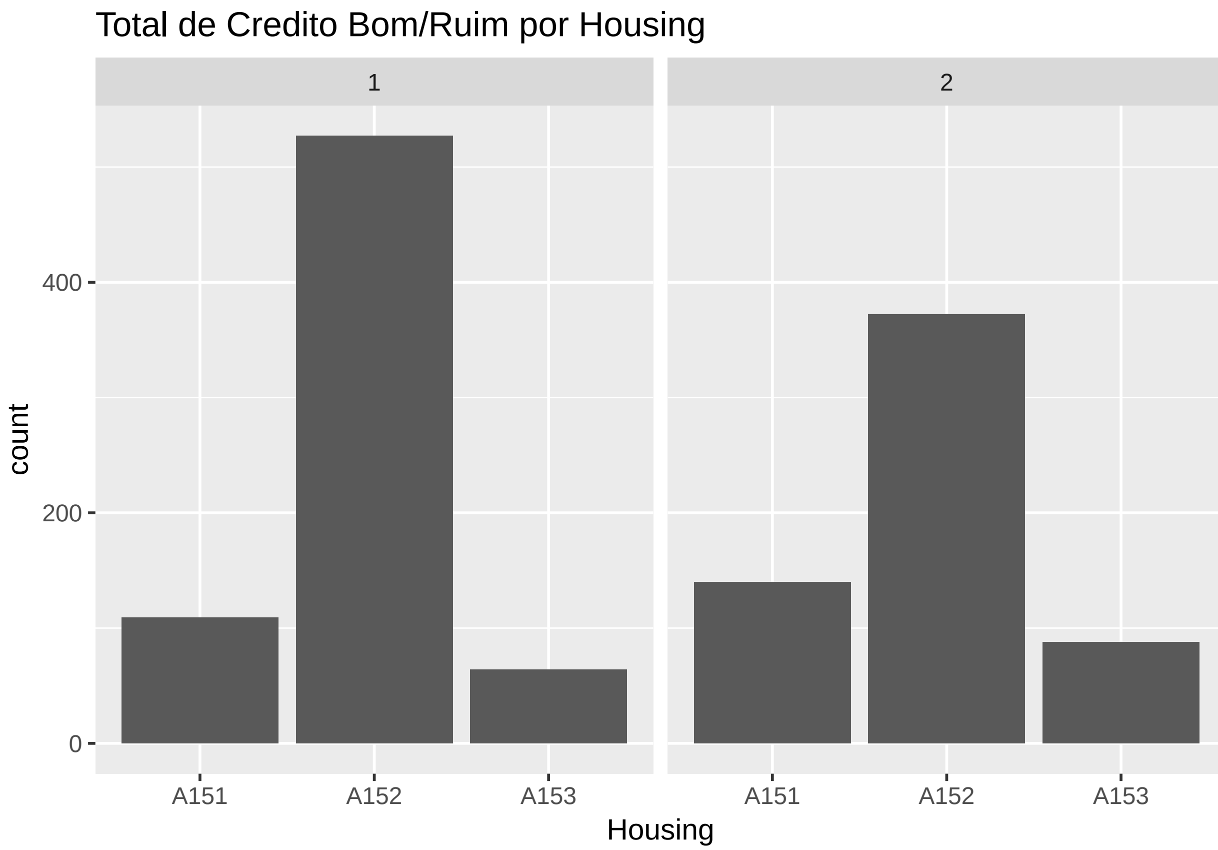
Total de Credito Bom/Ruim por Property
1
2
count
200
150
100
50
0
A121
A122
A123
A124
A121
A122
A123
A124
Property


```
##
## [[13]]
## NULL
##
## [[14]]
```

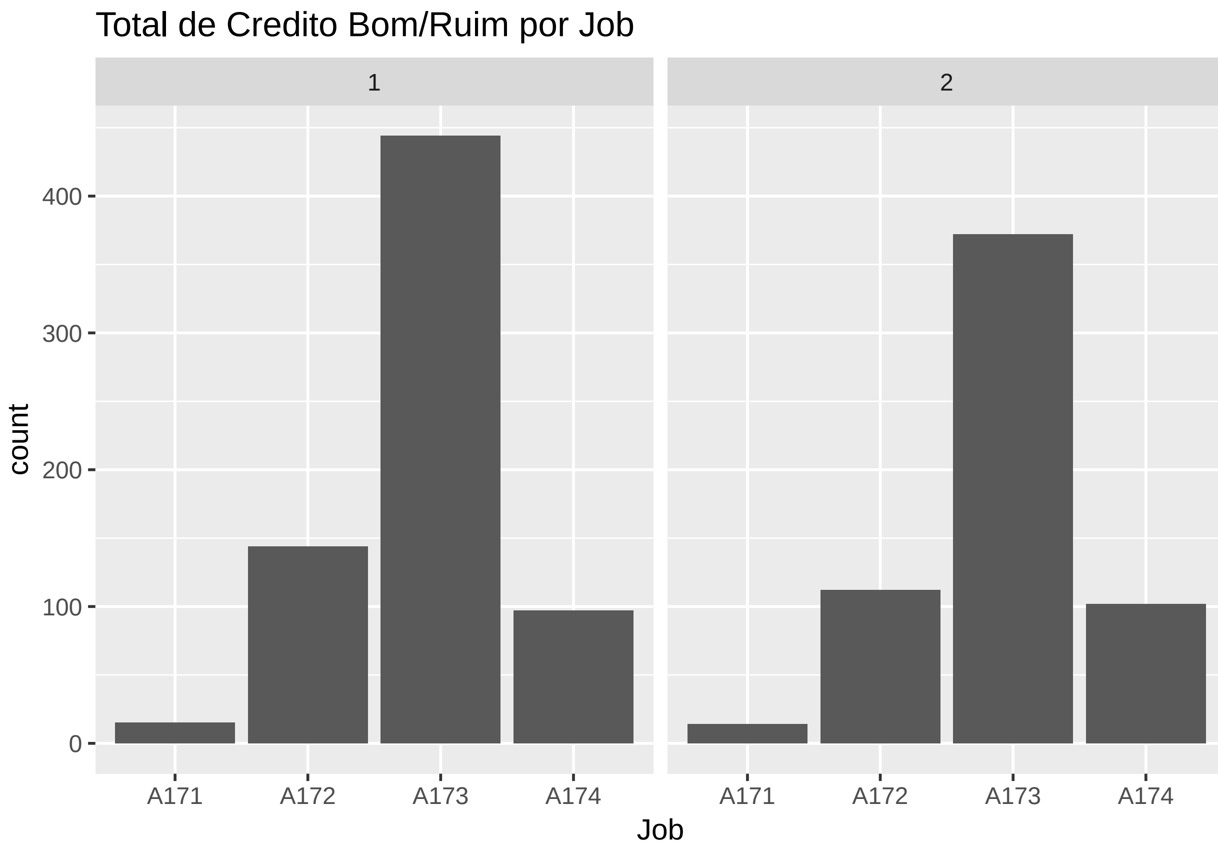


Total de Credito Bom/Ruim por OtherInstalments
1
2
600
400
200
0
count
A141
A142
A143
A141
A142
A143
OtherInstalments

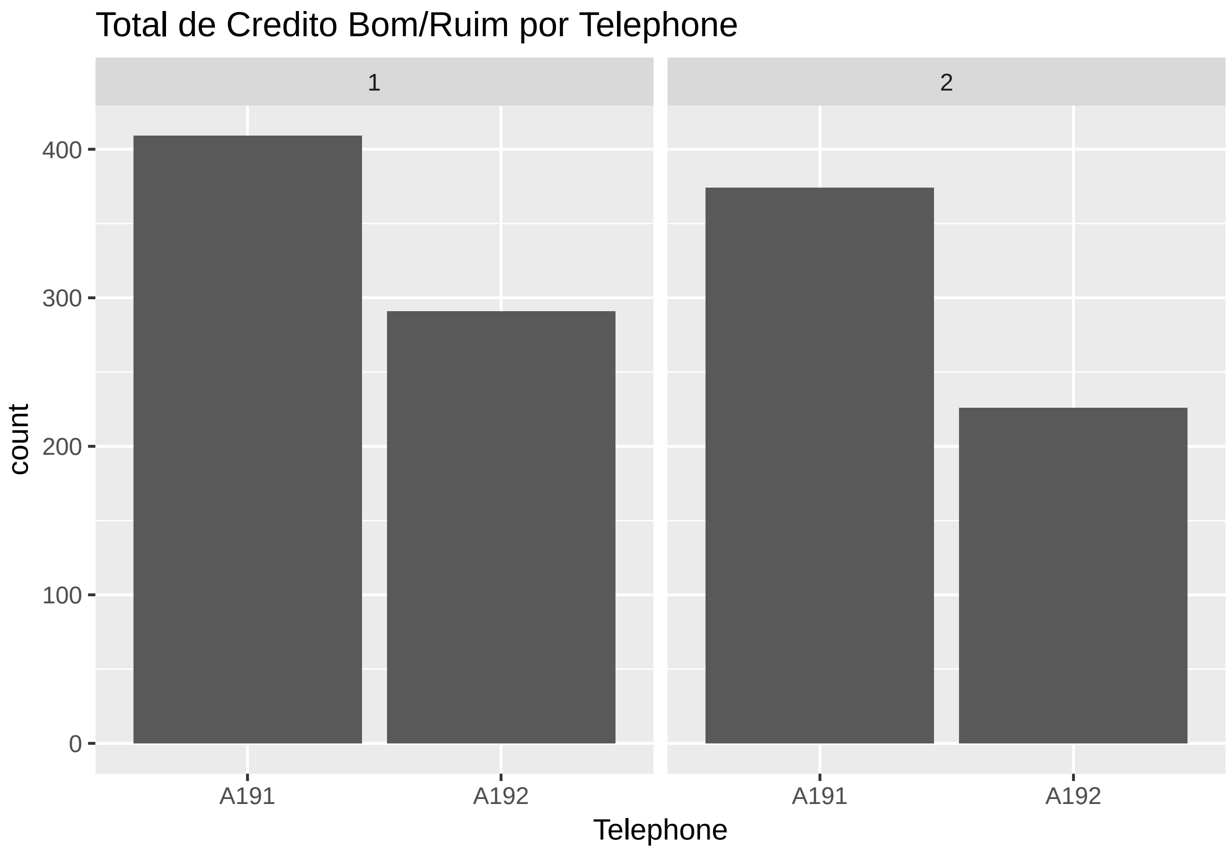

```
##
## [[15]]
```


Total de Credito Bom/Ruim por Housing
1
2
count
400
200
0
A151
A152
A153
A151
A152
A153
Housing

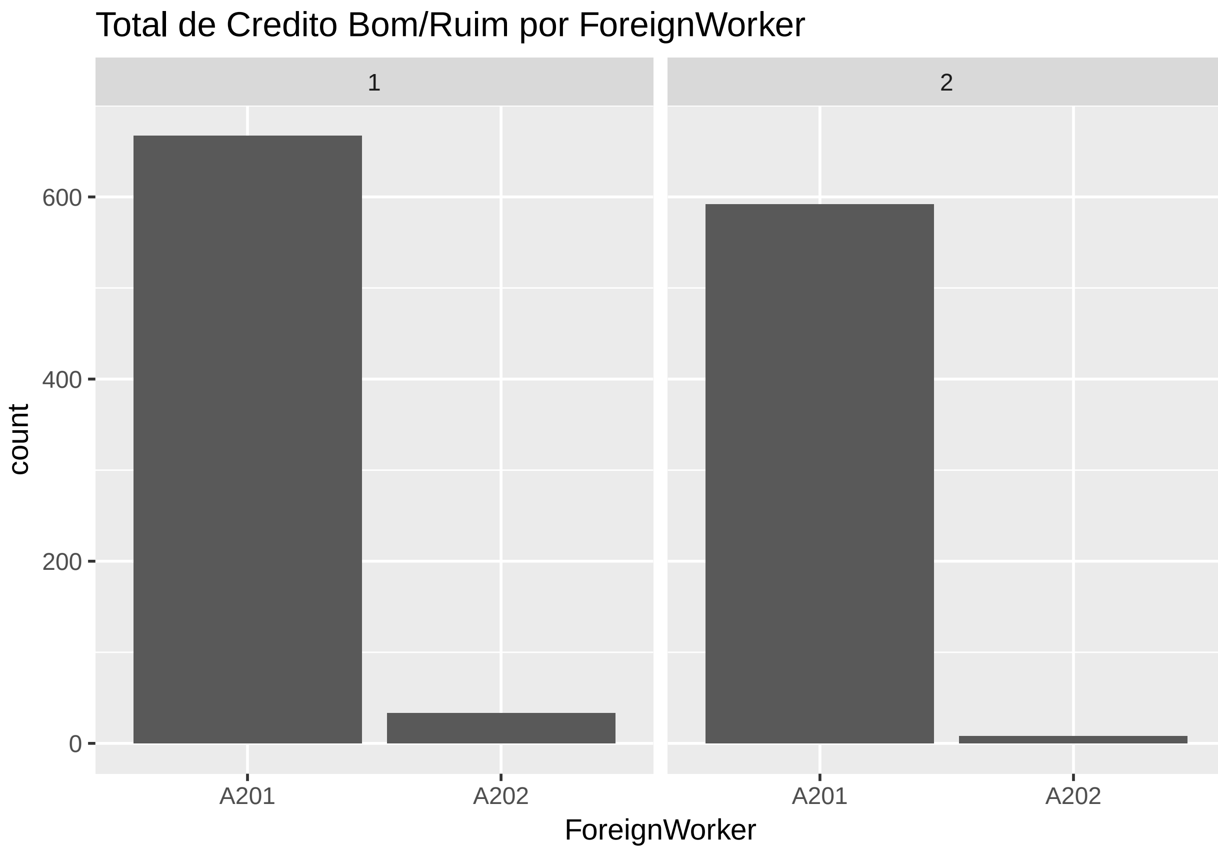
```
##
## [[16]]
## NULL
##
## [[17]]
```


Total de Credito Bom/Ruim por Job
1
2
400
300
200
100
0
count
A171
A172
A173
A174
A171
A172
A173
A174
Job

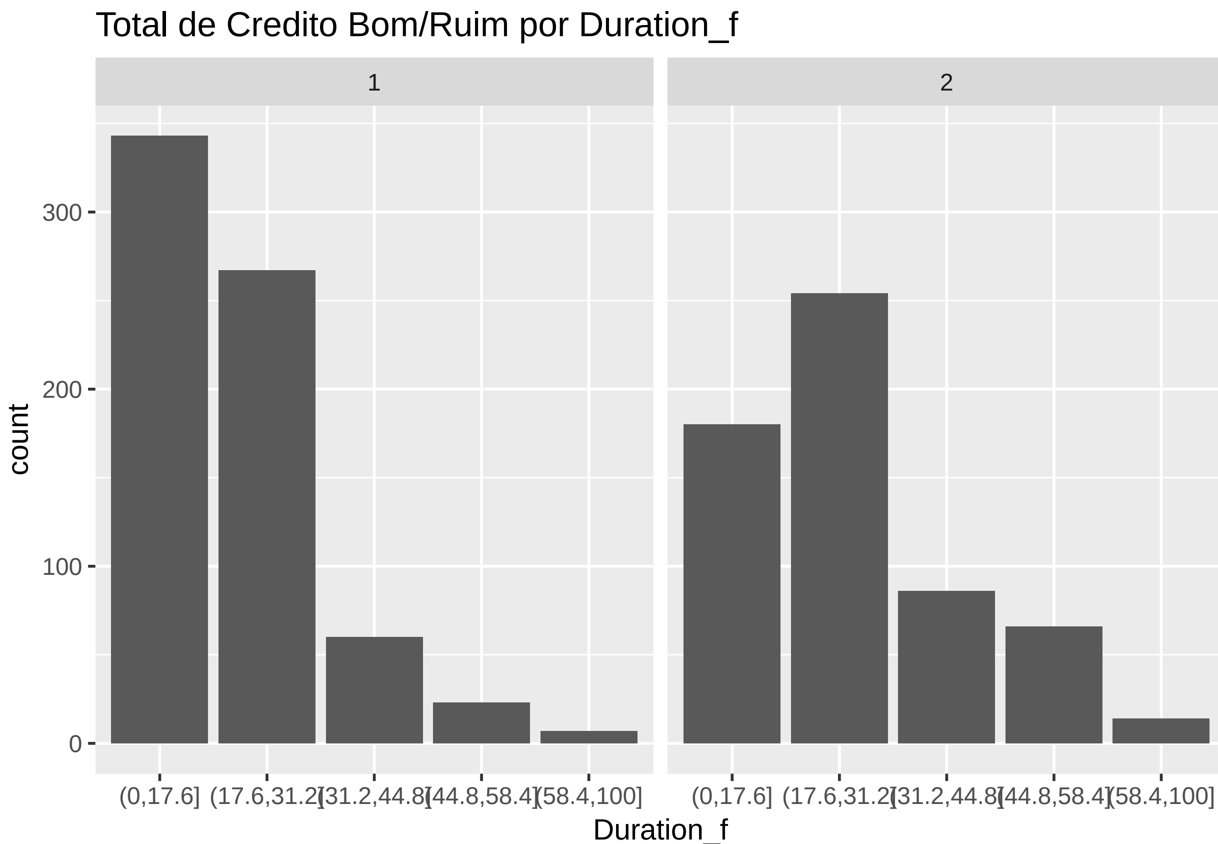
```
##
## [[18]]
## NULL
##
## [[19]]
```


Total de Credito Bom/Ruim por Telephone
1
2
400
300
200
100
0
count
A191
A192
A191
A192
Telephone

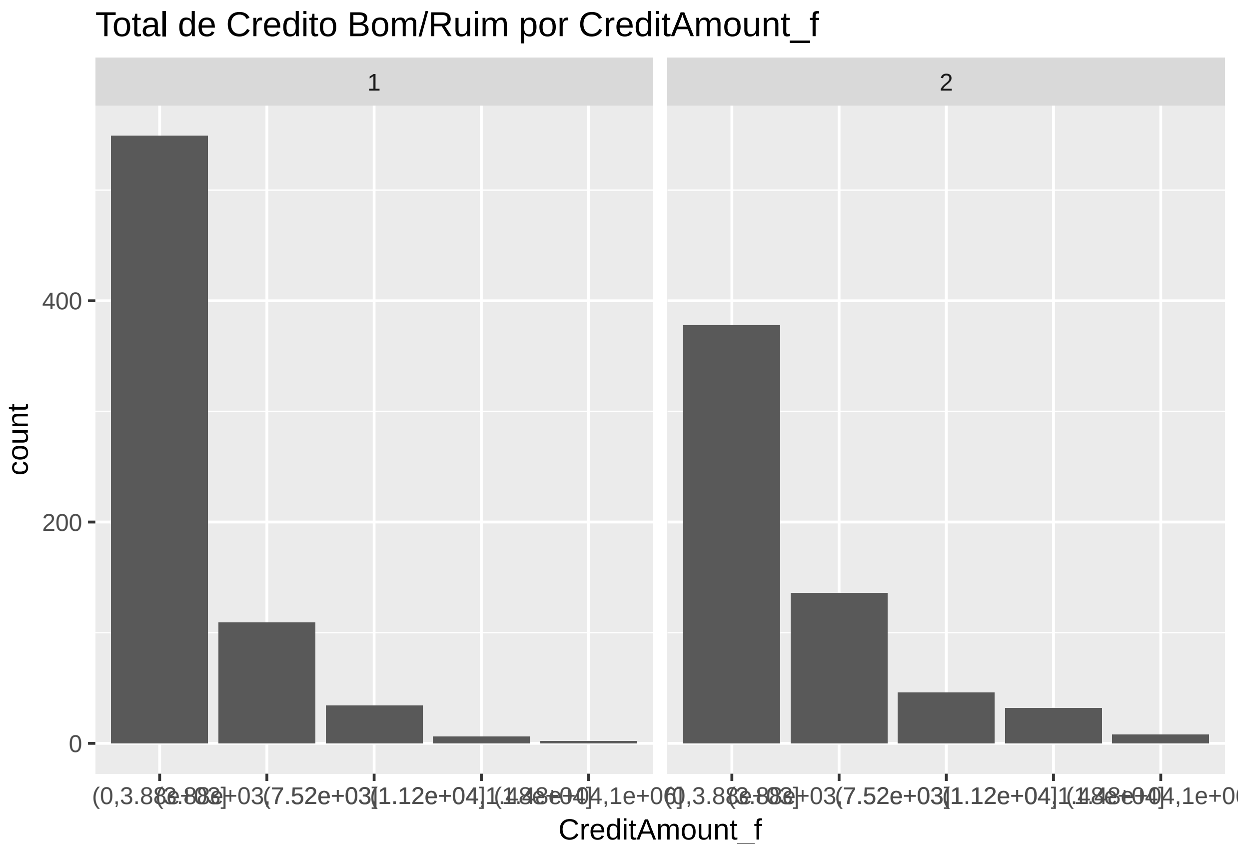
```
##
## [[20]]
```


Total de Credito Bom/Ruim por ForeignWorker
1
2
600
400
200
0
count
A201
A202
A201
A202
ForeignWorker

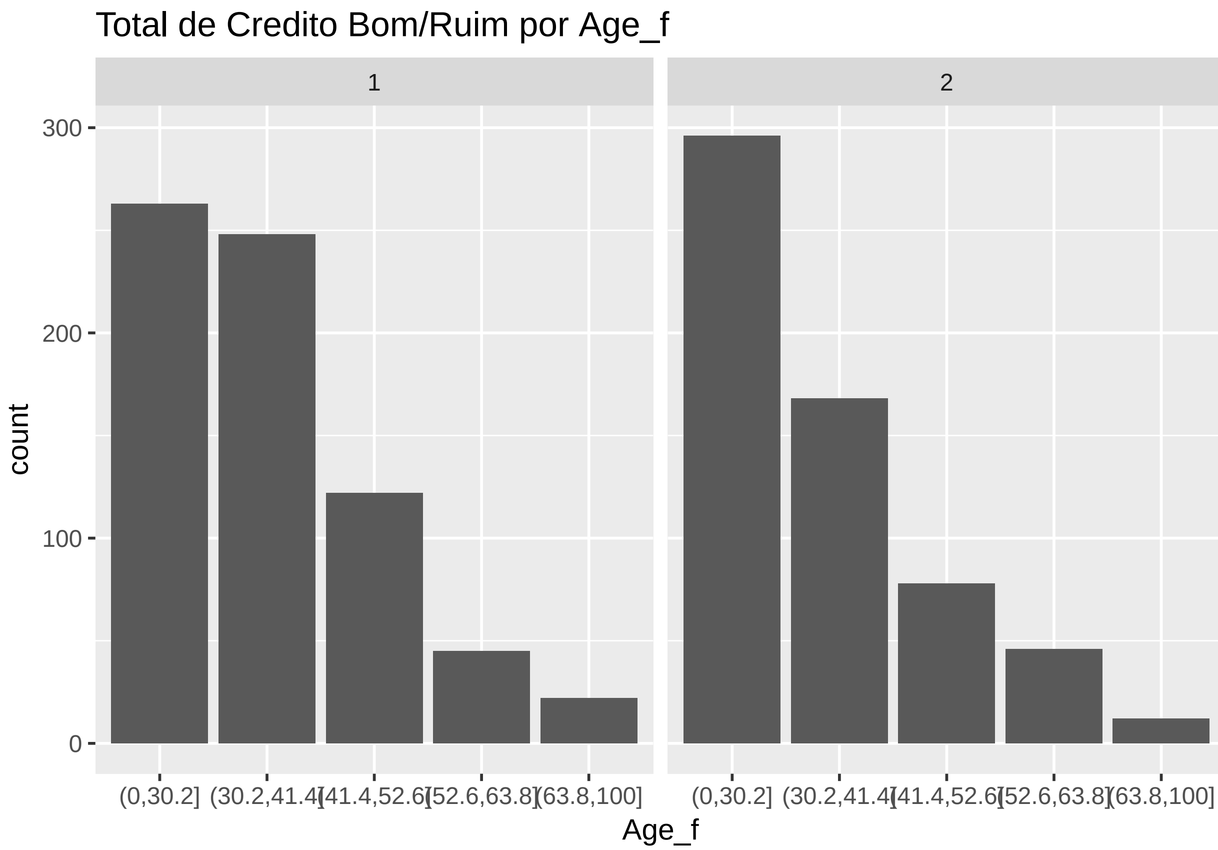
```
##
## [[21]]
```


Total de Credito Bom/Ruim por Duration_f
1
2
count
300
200
100
0
(0,17.6]
(17.6,31.2]
(31.2,44.8]
(44.8,58.4]
(58.4,100]
Duration_f

```
##
## [[22]]
```

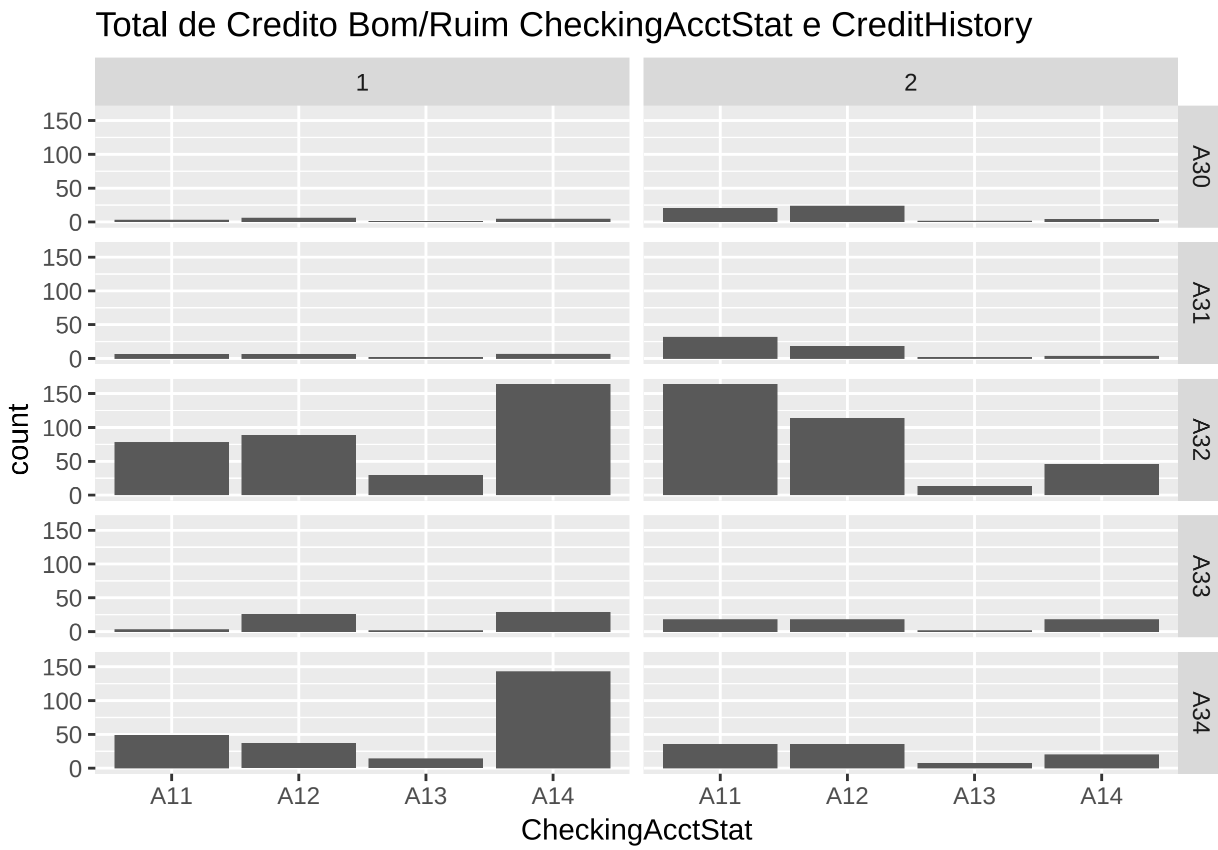

Total de Credito Bom/Ruim por CreditAmount_f
1
2
count
400
200
0
CreditAmount_f

```
##
## [[23]]
```



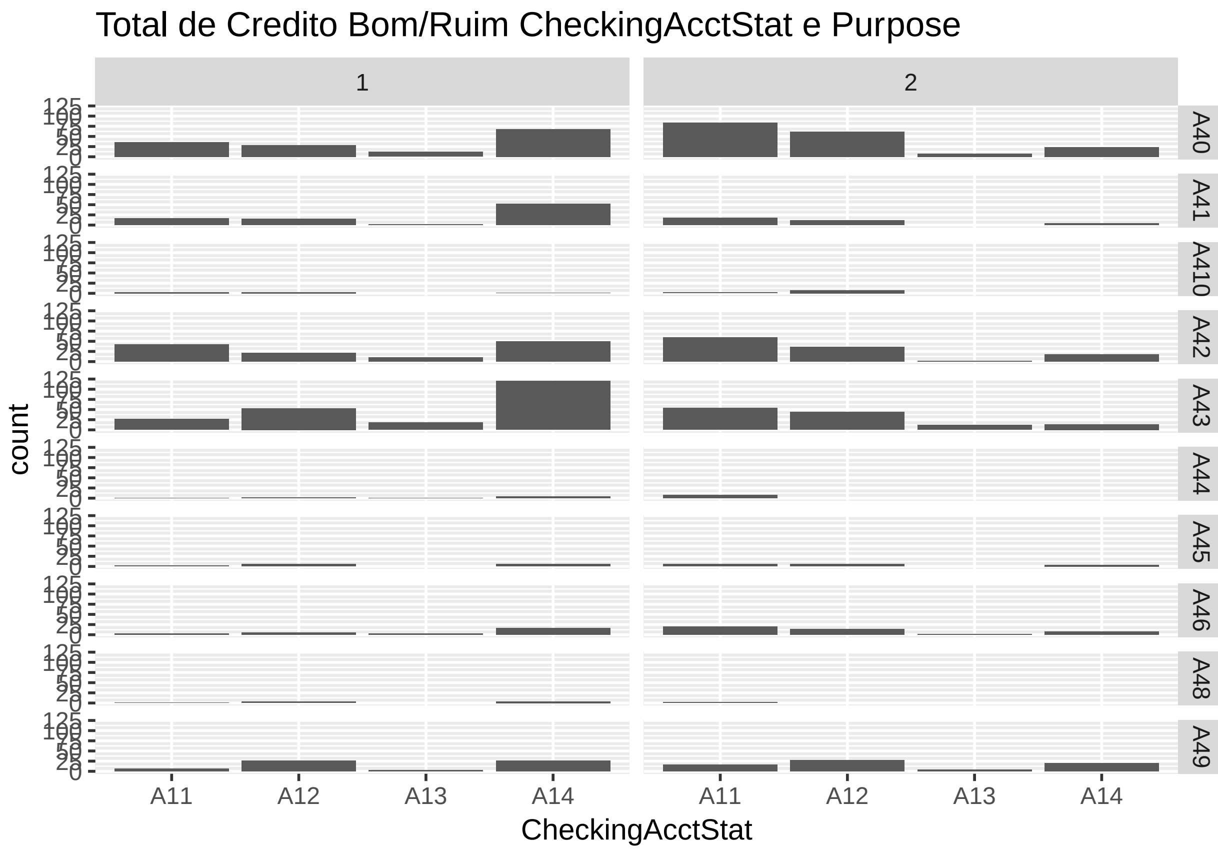

```
# Plots CreditStatus vs CheckingAcctStat
lapply(colNames2, function(x){
  if(is.factor(Credit[,x]) & x != "CheckingAcctStat") {
    ggplot(Credit, aes(CheckingAcctStat)) +
      geom_bar() +
      facet_grid(paste(x, " ~ CreditStatus"))+
      ggtitle(paste("Total de Credito Bom/Ruim CheckingAcctStat e",x))
  }})
```

```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
```

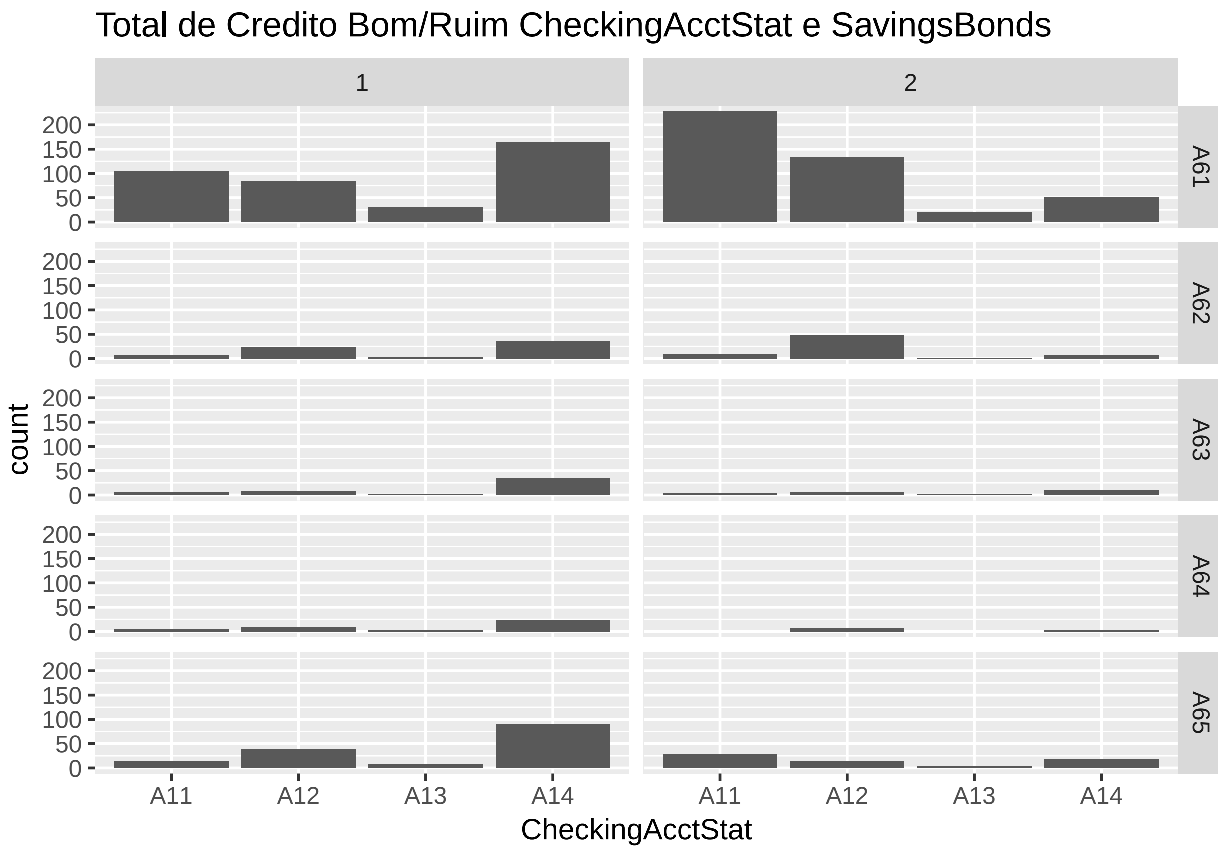


Total de Credito Bom/Ruim CheckingAcctStat e CreditHistory
1
2
A30
A31
A32
A33
A34
150
100
50
0
count
A11
A12
A13
A14
CheckingAcctStat


```
##
## [[4]]
```

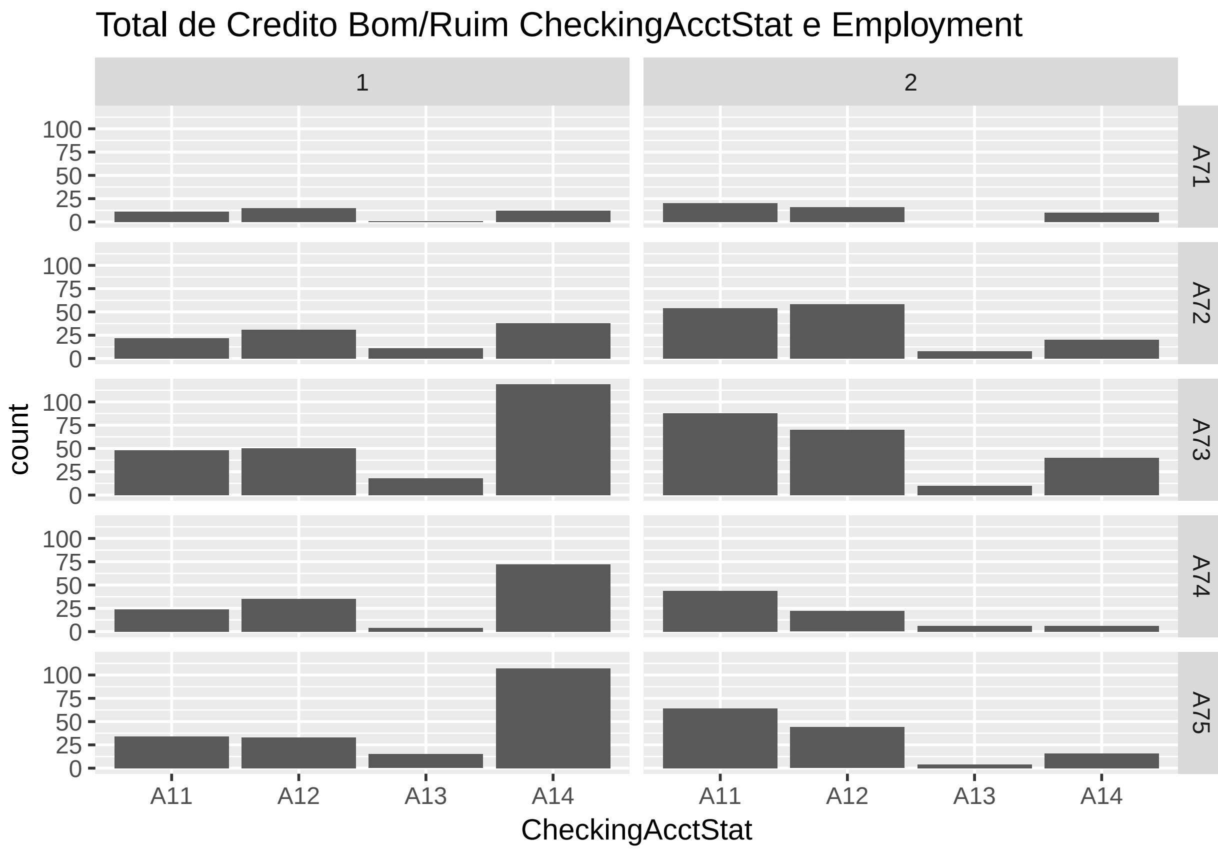

Total de Credito Bom/Ruim CheckingAcctStat e Purpose
1
2
A40
A41
A410
A42
A43
A44
A45
A46
A48
A49
count
125
100
75
50
25
0
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[5]]
## NULL
##
## [[6]]
```

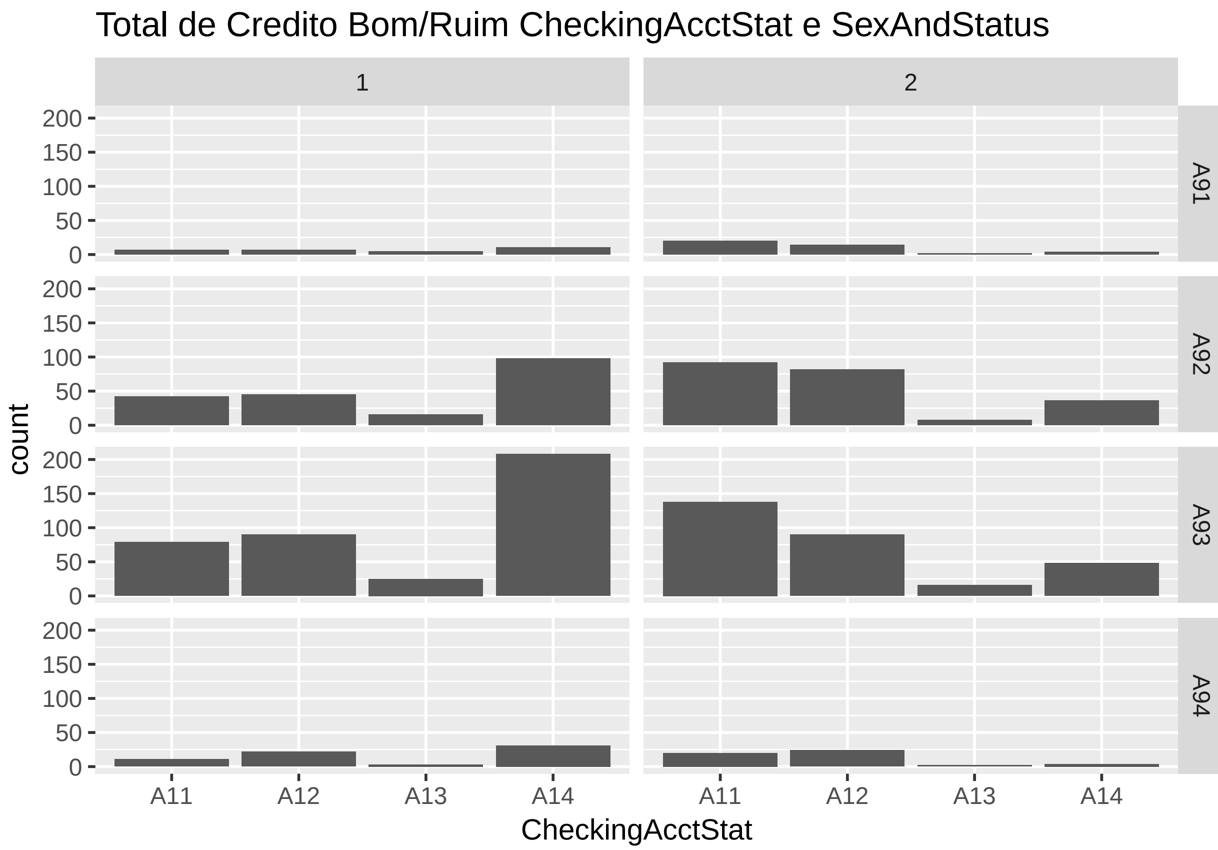


Total de Credito Bom/Ruim CheckingAcctStat e SavingsBonds
1
2
A61
A62
A63
A64
A65
count
0
50
100
150
200
A11
A12
A13
A14
CheckingAcctStat


```
##
## [[7]]
```

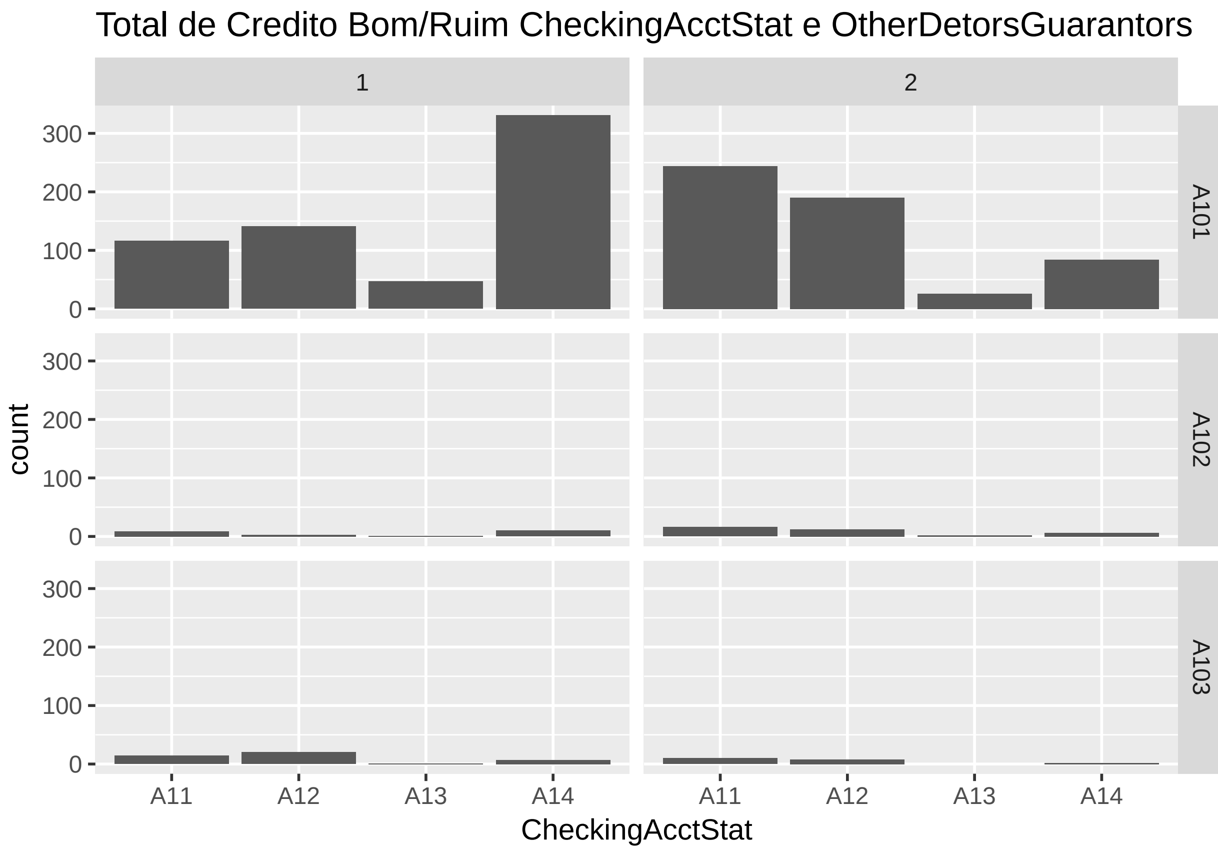

Total de Credito Bom/Ruim CheckingAcctStat e Employment
1
2
A71
A72
A73
A74
A75
count
100
75
50
25
0
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[8]]
## NULL
##
## [[9]]
```

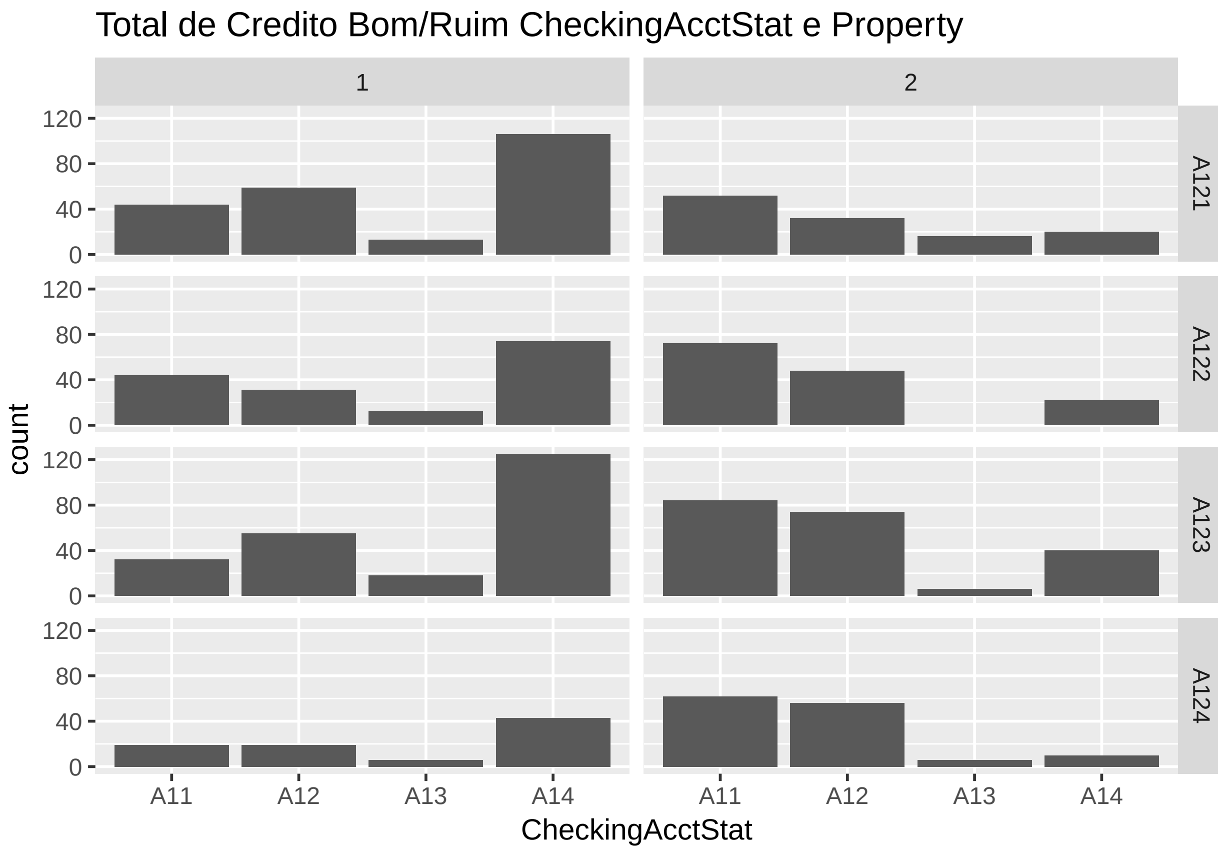


Total de Credito Bom/Ruim CheckingAcctStat e SexAndStatus
1
2
A91
A92
A93
A94
count
200
150
100
50
0
A11
A12
A13
A14
CheckingAcctStat

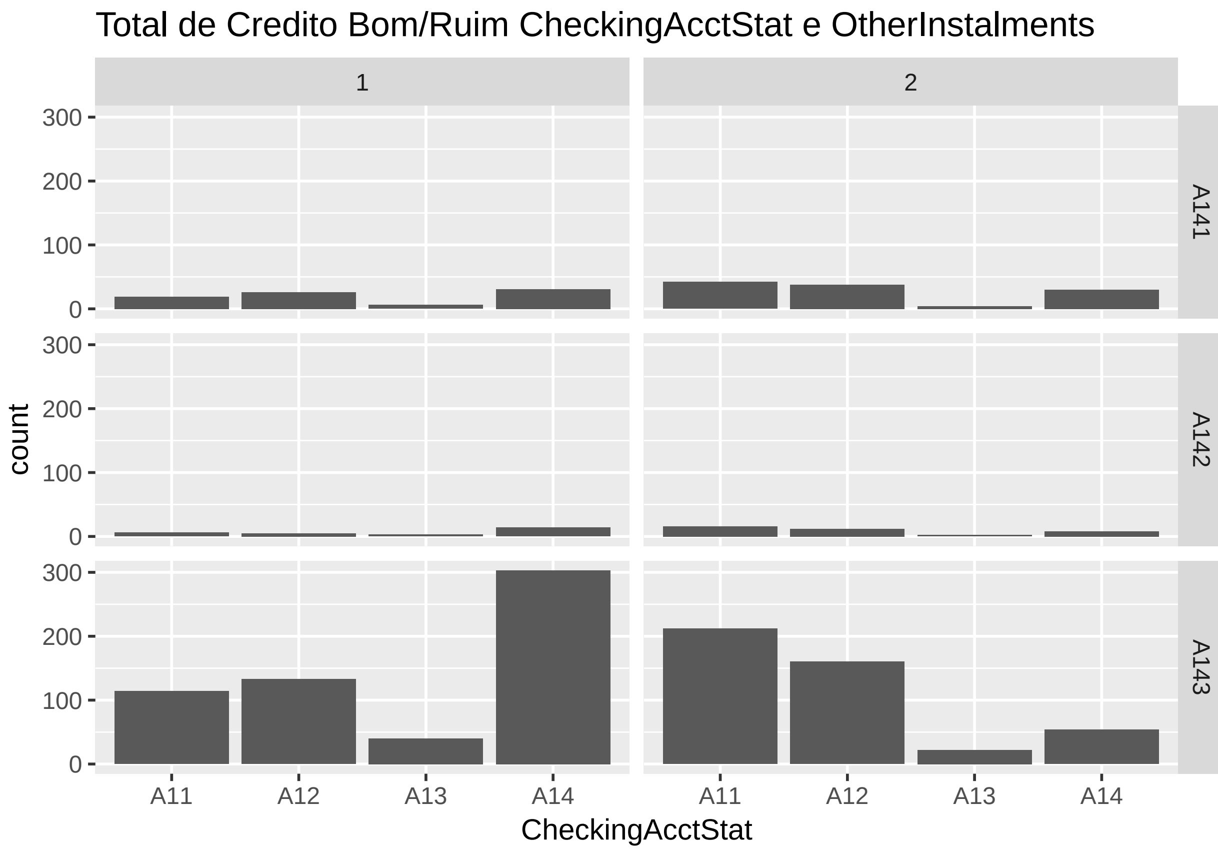

```
##
## [[10]]
```


Total de Credito Bom/Ruim CheckingAcctStat e OtherDetorsGuarantors
1
2
A101
A102
A103
count
300
200
100
0
A11
A12
A13
A14
CheckingAcctStat

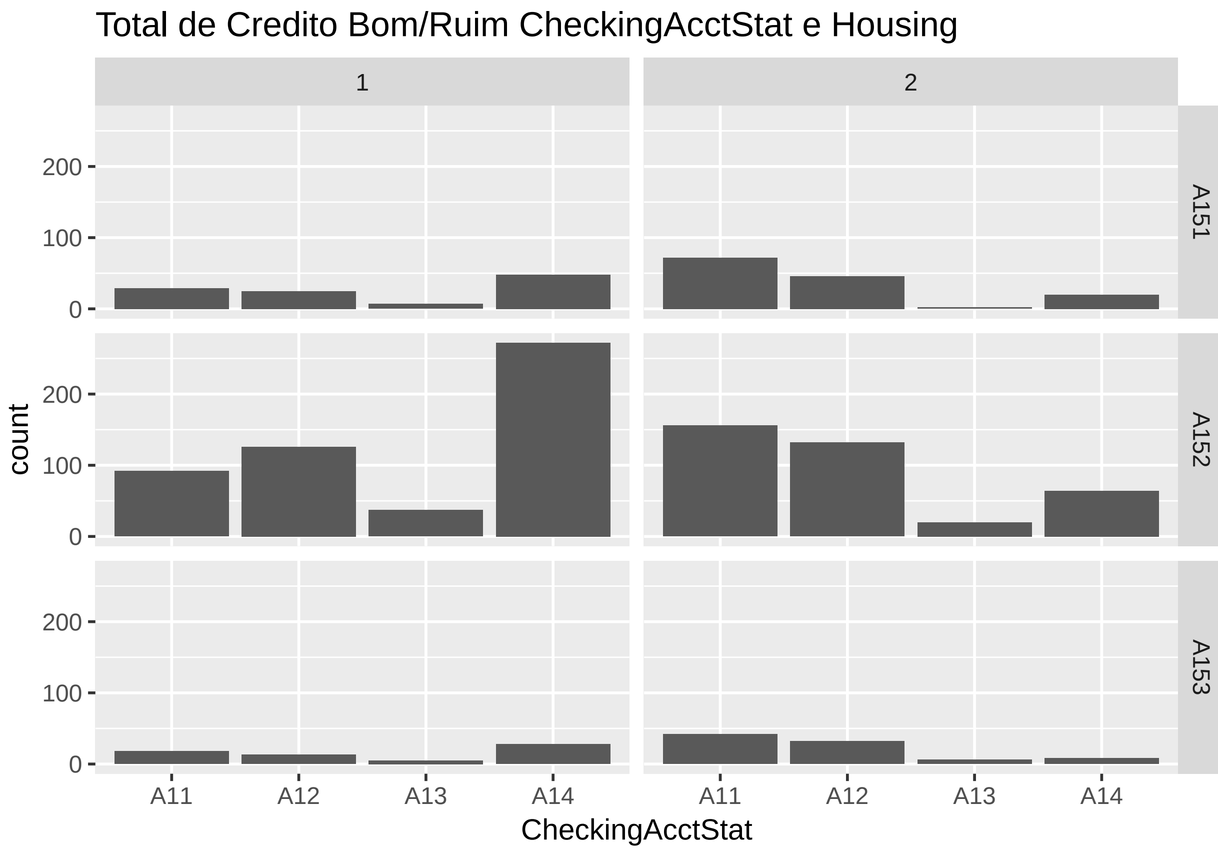
```
##
## [[11]]
## NULL
##
## [[12]]
```


Total de Credito Bom/Ruim CheckingAcctStat e Property
1
2
A121
A122
A123
A124
count
0
40
80
120
A11
A12
A13
A14
CheckingAcctStat

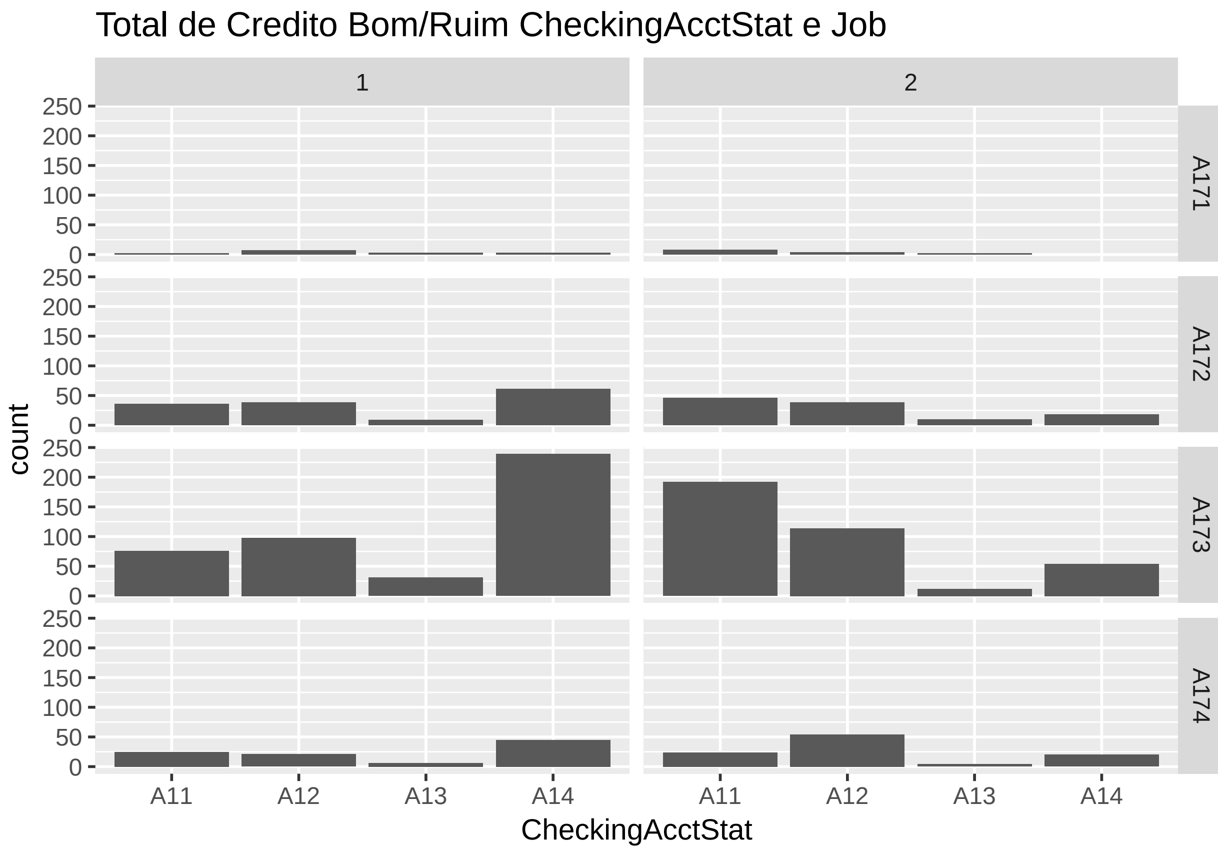
```
##
## [[13]]
## NULL
##
## [[14]]
```


Total de Credito Bom/Ruim CheckingAcctStat e OtherInstalments
1
2
A141
A142
A143
count
300
200
100
0
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[15]]
```

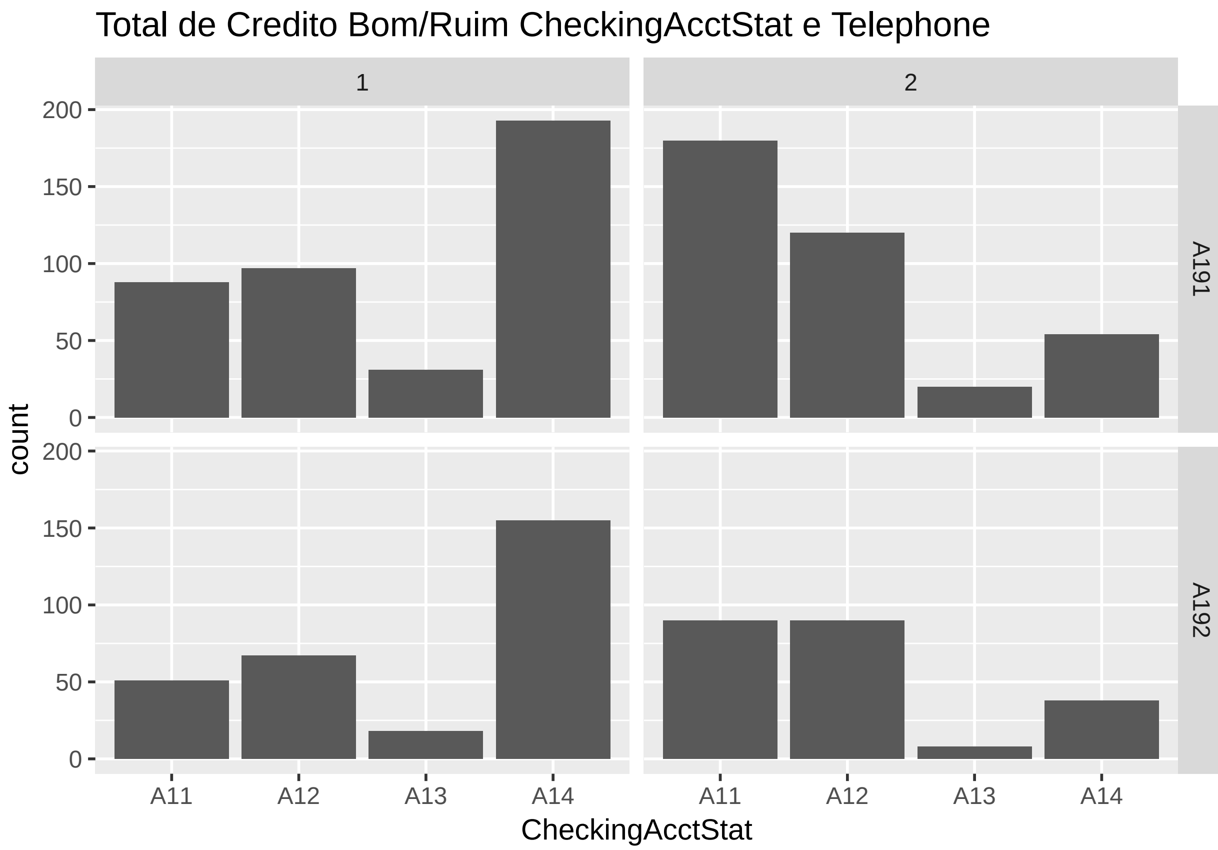

Total de Credito Bom/Ruim CheckingAcctStat e Housing
1
2
A151
A152
A153
count
0
100
200
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[16]]
## NULL
##
## [[17]]
```

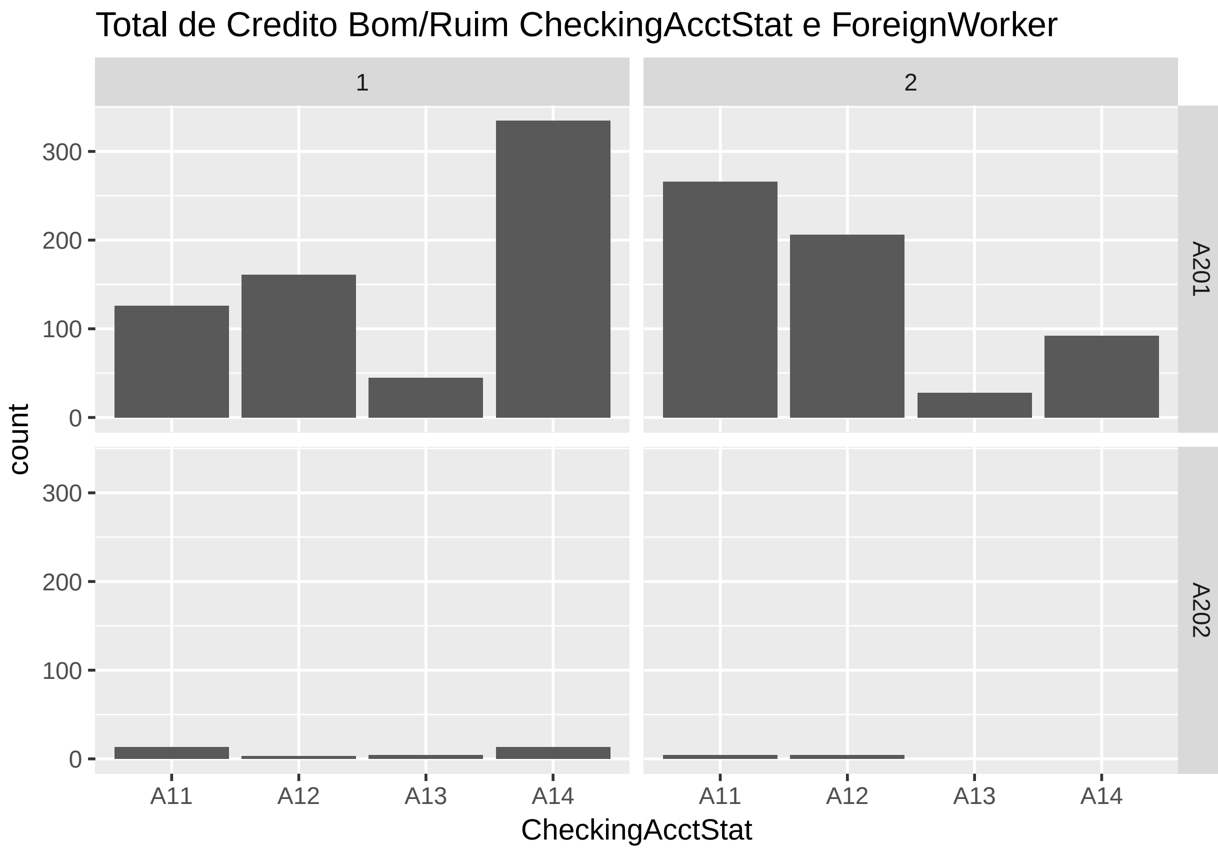


Total de Credito Bom/Ruim CheckingAcctStat e Job
1
2
A171
A172
A173
A174
count
250
200
150
100
50
0
A11
A12
A13
A14
CheckingAcctStat

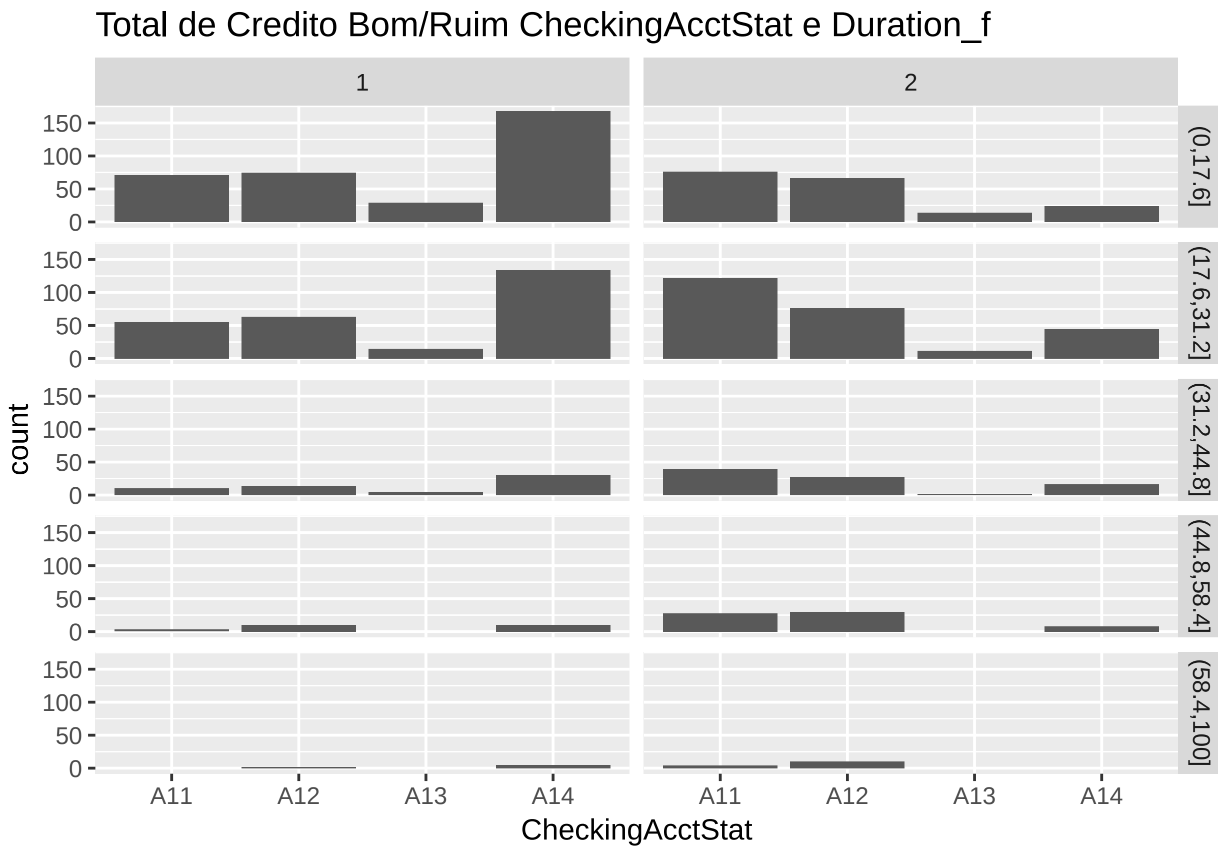

```
##
## [[18]]
## NULL
##
## [[19]]
```


Total de Credito Bom/Ruim CheckingAcctStat e Telephone
1
2
A191
A192
count
200
150
100
50
0
A11
A12
A13
A14
CheckingAcctStat

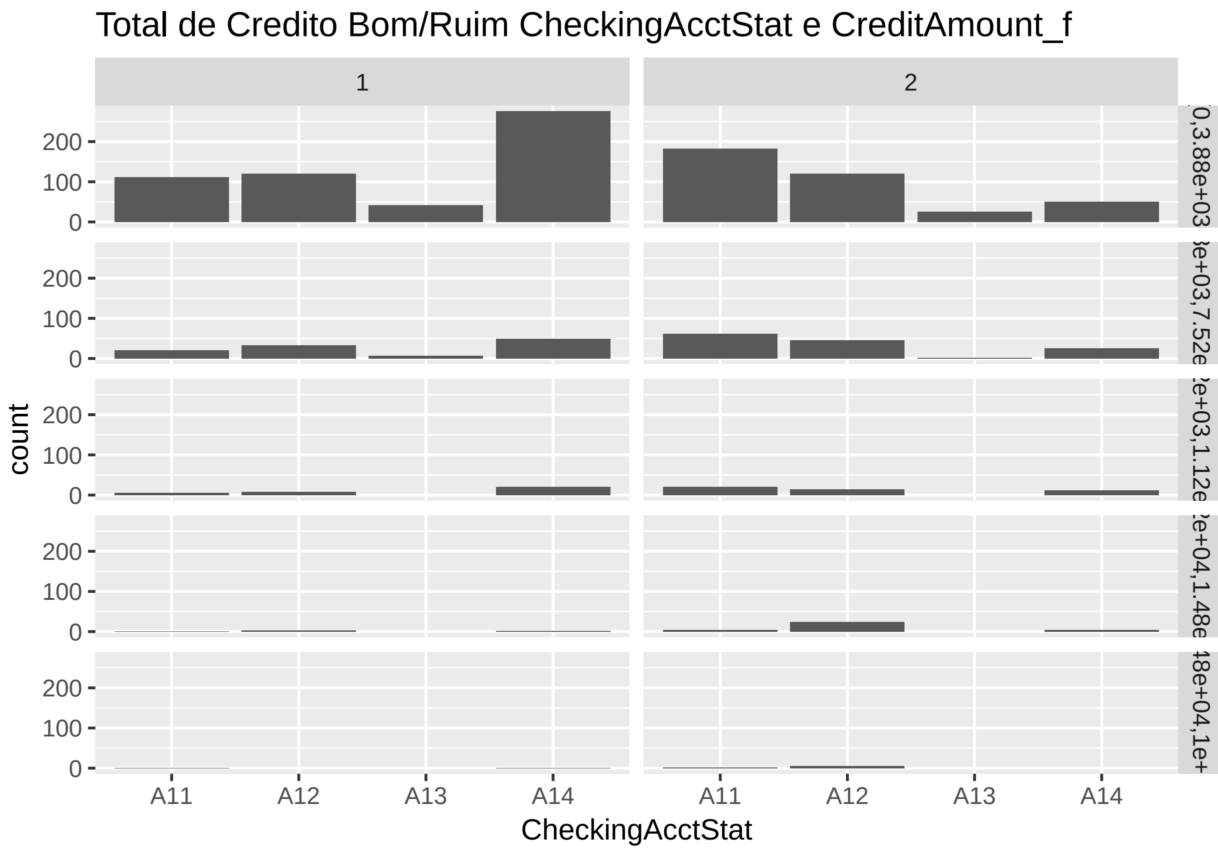
```
##
## [[20]]
```


Total de Credito Bom/Ruim CheckingAcctStat e ForeignWorker
1
2
A201
A202
count
300
200
100
0
300
200
100
0
A11
A12
A13
A14
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[21]]
```

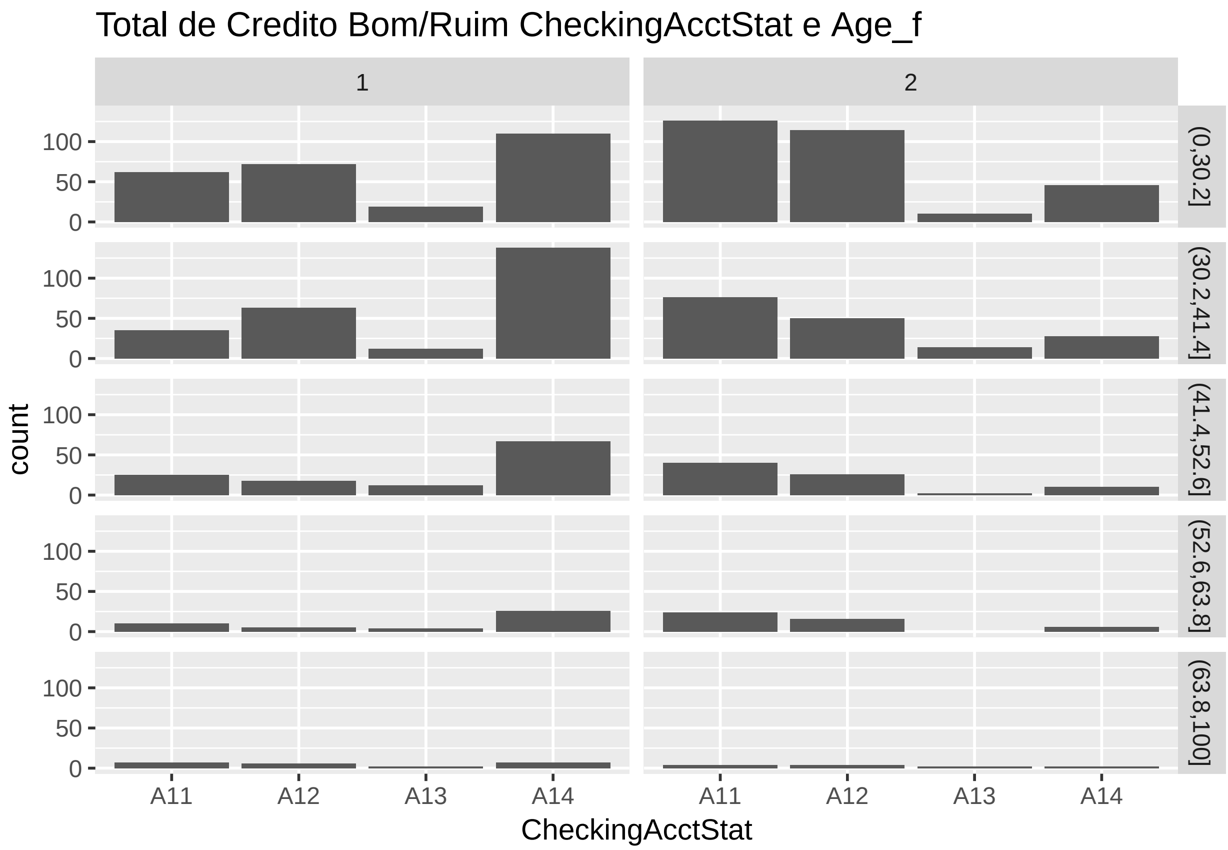

Total de Credito Bom/Ruim CheckingAcctStat e Duration_f
1
2
(0,17.6]
(17.6,31.2]
(31.2,44.8]
(44.8,58.4]
(58.4,100]
count
0
50
100
150
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[22]]
```



Total de Credito Bom/Ruim CheckingAcctStat e CreditAmount_f
1
2
count
200
100
0
A11
A12
A13
A14
CheckingAcctStat


```
##
## [[23]]
```

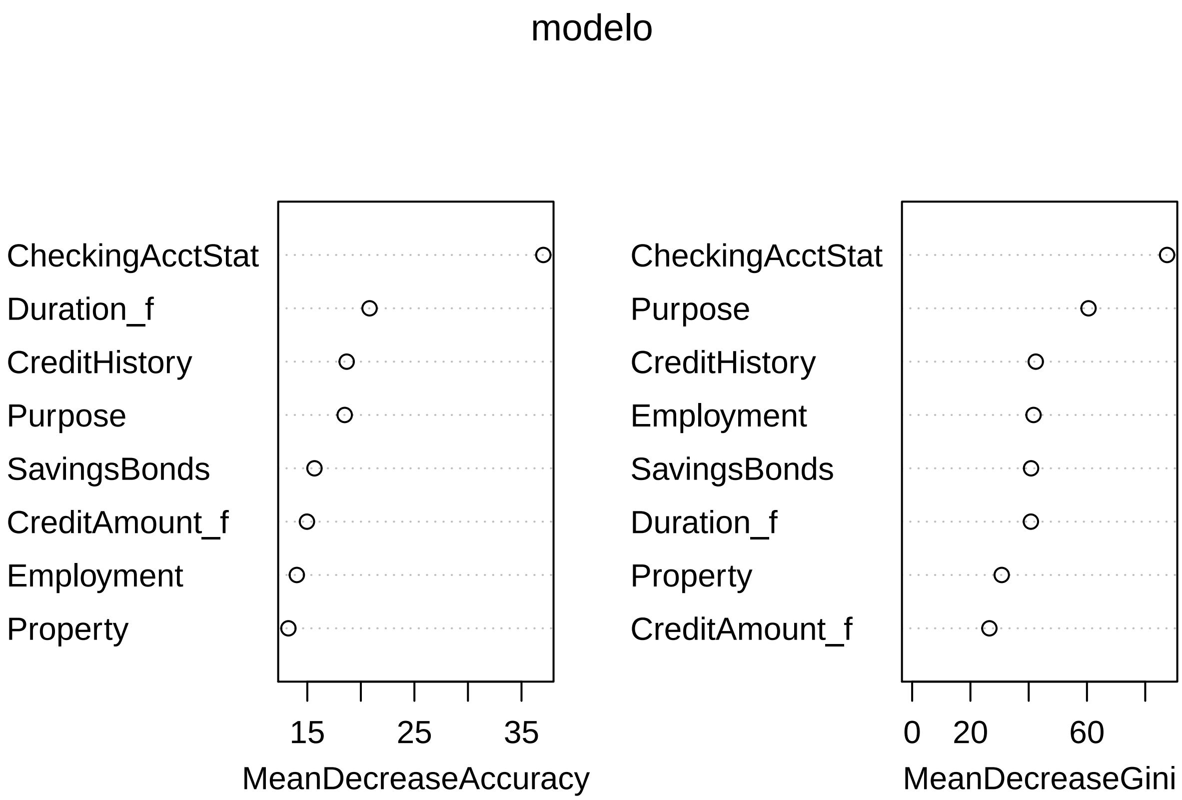



```r
# Feature Selection (Seleção de Variáveis)
# Modelo randomForest para criar um plot de importância das variáveis
library(randomForest)
```

```
## randomForest 4.6-14

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

##
## Attaching package: 'randomForest'

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

```r
modelo <- randomForest( CreditStatus ~ .
                        - Duration
                        - Age
                        - CreditAmount
                        - ForeignWorker
                        - NumberDependents
                        - Telephone
                        - ExistingCreditsAtBank
                        - PresentResidenceTime
                        - Job
                        - Housing
                        - SexAndStatus
                        - InstallmentRatePecnt
                        - OtherDetorsGuarantors
```

```
                                           - Age_f
                                           - OtherInstalments,
                                           data = Credit,
                                           ntree = 100, nodesize = 10, importance = T)

varImpPlot(modelo)
```




```
outFrame <- serList(list(credit.model = modelo))


# Criando o Modelo Preditivo no R
# Criar um modelo de classificação baseado em randomForest
library(randomForest)

# Cross Tabulation
table(Credit$CreditStatus)
```

```
##
##   1   2
## 700 600
```

```
# Funcao para gerar dados de treino e dados de teste
splitData <- function(dataframe, seed = NULL) {
  if (!is.null(seed)) set.seed(seed)
  index <- 1:nrow(dataframe)
  trainindex <- sample(index, trunc(length(index)/2))
  trainset <- dataframe[trainindex, ]
  testset <- dataframe[-trainindex, ]
  list(trainset = trainset, testset = testset)
```

```
}

# Gerando dados de treino e de teste
splits <- splitData(Credit, seed = 808)

# Separando os dados
dados_treino <- splits$trainset
dados_teste <- splits$testset

# Verificando o numero de linhas
nrow(dados_treino)
```

```
## [1] 650
```

```
nrow(dados_teste)
```

```
## [1] 650
```

```
# Construindo o modelo
modelo <- randomForest( CreditStatus ~ CheckingAcctStat
                        + Duration_f
                        + Purpose
                        + CreditHistory
                        + SavingsBonds
                        + Employment
                        + CreditAmount_f,
                        data = dados_treino,
                        ntree = 100,
                        nodesize = 10)

# Imprimindo o resultado
print(modelo)
```

```
##
## Call:
##  randomForest(formula = CreditStatus ~ CheckingAcctStat + Duration_f +      Purpose + CreditHistory +
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 26.15%
## Confusion matrix:
##     1   2 class.error
## 1 282  74   0.2078652
## 2  96 198   0.3265306
```

```
# Fazendo Previsões
# Previsões com um modelo de classificação baseado em randomForest
require(randomForest)

# Gerando previsões nos dados de teste
previsoes <- data.frame(observado = dados_teste$CreditStatus,
                        previsto = predict(modelo, newdata = dados_teste))



# Visualizando o resultado
```

```r
#View(previsoes)
#View(dados_teste)


# Calculando a Confusion Matrix em R (existem outras formas)
# Label 1 - Credito Ruim
# Label 2 - Credito Bom

# Formulas
Accuracy <- function(x){
  (x[1,1] + x[2,2]) / (x[1,1] + x[1,2] + x[2,1] + x[2,2])
}

Recall <- function(x){
  x[1,1] / (x[1,1] + x[1,2])
}

Precision <- function(x){
  x[1,1] / (x[1,1] + x[2,1])
}

W_Accuracy  <- function(x){
  (x[1,1] + x[2,2]) / (x[1,1] + 5 * x[1,2] + x[2,1] + x[2,2])
}

F1 <- function(x){
  2 * x[1,1] / (2 * x[1,1] + x[1,2] + x[2,1])
}

# Criando a confusion matrix.
confMat <- matrix(unlist(Map(function(x, y){sum(ifelse(previsoes[, 1] == x & previsoes[, 2] == y, 1, 0)
                             c(2, 1, 2, 1), c(2, 2, 1, 1))), nrow = 2)


# Criando um dataframe com as estatisticas dos testes
df_mat <- data.frame(Category = c("Credito Ruim", "Credito Bom"),
                     Classificado_como_ruim = c(confMat[1,1], confMat[2,1]),
                     Classificado_como_bom = c(confMat[1,2], confMat[2,2]),
                     Accuracy_Recall = c(Accuracy(confMat), Recall(confMat)),
                     Precision_WAcc = c(Precision(confMat), W_Accuracy(confMat)))

print(df_mat)
```

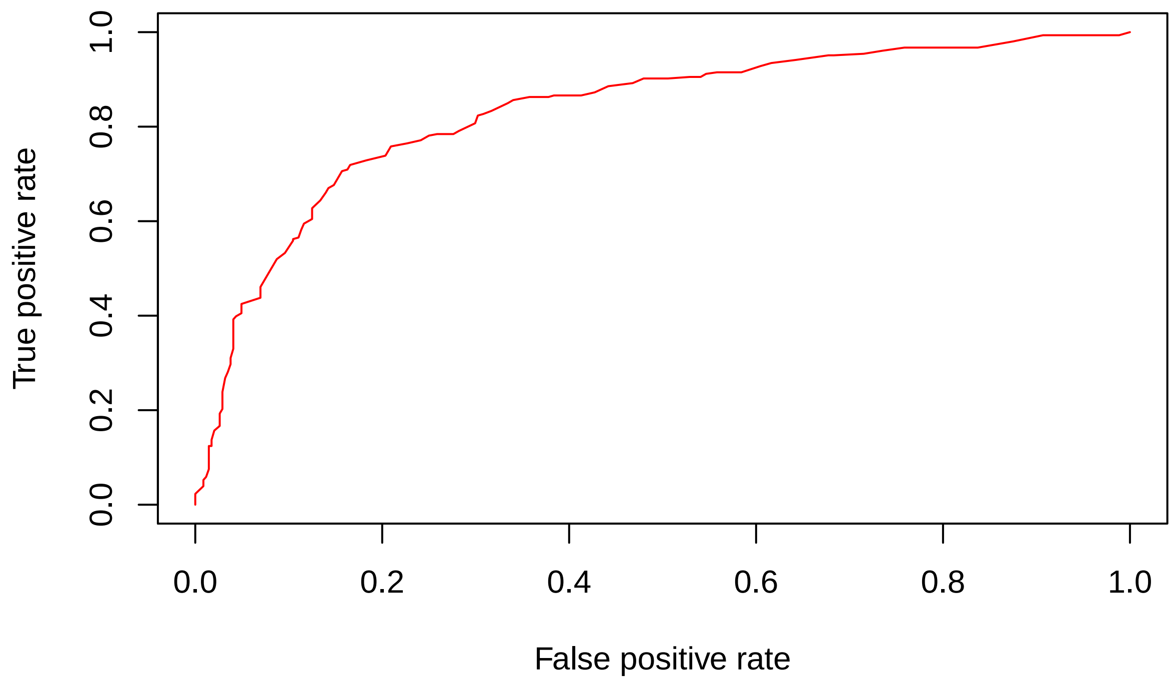
```
##       Category Classificado_como_ruim Classificado_como_bom
## 1 Credito Ruim                    216                    90
## 2  Credito Bom                     55                   289
##   Accuracy_Recall Precision_WAcc
## 1       0.7769231       0.797048
## 2       0.7058824       0.500000
```

```r
# Gerando uma curva ROC em R
library("ROCR")
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
## The following object is masked from 'package:stats':
##
##     lowess
```

```
# Gerando as classes de dados
class1 <- predict(modelo, newdata = dados_teste, type = 'prob')
class2 <- dados_teste$CreditStatus

# Gerando a curva ROC
pred <- prediction(class1[,2], class2)
perf <- performance(pred, "tpr","fpr")
plot(perf, col = rainbow(10))
```




```
# Gerando Confusion Matrix com o Caret
library(caret)
```

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

```
confusionMatrix(previsoes$observado, previsoes$previsto)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2
##          1 289  55
##          2  90 216
##
##                Accuracy : 0.7769
##                  95% CI : (0.7429, 0.8084)
##     No Information Rate : 0.5831
##     P-Value [Acc > NIR] : < 2e-16
##
```

```
##                  Kappa : 0.5495
##
##  Mcnemar's Test P-Value : 0.00475
##
##             Sensitivity : 0.7625
##             Specificity : 0.7970
##          Pos Pred Value : 0.8401
##          Neg Pred Value : 0.7059
##              Prevalence : 0.5831
##          Detection Rate : 0.4446
##    Detection Prevalence : 0.5292
##       Balanced Accuracy : 0.7798
##
##        'Positive' Class : 1
##
```

```r
# Otimizando o Modelo preditivo
# Modelo randomForest ponderado
# O pacote C50 permite que você dê peso aos erros, construindo assim um resultado ponderado
library(C50)

# Criando uma Cost Function
Cost_func <- matrix(c(0, 1.5, 1, 0), nrow = 2, dimnames = list(c("1", "2"), c("1", "2")))

# Cria o modelo
modelo_v2  <- C5.0(CreditStatus ~ CheckingAcctStat
                   + Purpose
                   + CreditHistory
                   + SavingsBonds
                   + Employment,
                   data = dados_treino,
                   trials = 100,
                   cost = Cost_func)

print(modelo_v2)
```

```
##
## Call:
## C5.0.formula(formula = CreditStatus ~ CheckingAcctStat + Purpose
##  + CreditHistory + SavingsBonds + Employment, data = dados_treino,
##  trials = 100, cost = Cost_func)
##
## Classification Tree
## Number of samples: 650
## Number of predictors: 5
##
## Number of boosting iterations: 100 requested;  3 used due to early stopping
## Average tree size: 15
##
## Non-standard options: attempt to group attributes
##
## Cost Matrix:
##     1 2
## 1 0.0 1
## 2 1.5 0
```

```r
# Dataframes com valores observados e previstos
previsoes_v2 <- data.frame(observado = dados_teste$CreditStatus,
                           previsto = predict(object = modelo_v2, newdata = dados_teste))

# Calculando a Confusion Matrix em R (existem outras formas).
# Label 1 - Credito Ruim
# Label 2 - Credito Bom

# Criando a confusion matrix.
confMat_v2 <- matrix(unlist(Map(function(x, y){sum(ifelse(previsoes_v2[, 1] == x & previsoes_v2[, 2] ==
                                c(2, 1, 2, 1), c(2, 2, 1, 1))), nrow = 2)


# Criando um dataframe com as estatisticas dos testes
df_mat <- data.frame(Category = c("Credito Ruim", "Credito Bom"),
                     Classificado_como_ruim = c(confMat_v2[1,1], confMat_v2[2,1]),
                     Classificado_como_bom = c(confMat_v2[1,2], confMat_v2[2,2]),
                     Accuracy_Recall = c(Accuracy(confMat_v2), Recall(confMat_v2)),
                     Precision_WAcc = c(Precision(confMat_v2), W_Accuracy(confMat_v2)))

print(df_mat)
```

```
##       Category Classificado_como_ruim Classificado_como_bom
## 1 Credito Ruim                    176                   130
## 2  Credito Bom                     74                   270
##   Accuracy_Recall Precision_WAcc
## 1       0.6861538      0.7040000
## 2       0.5751634      0.3811966
```

```r
# Gerando Confusion Matrix com o Caret
library(caret)
confusionMatrix(previsoes_v2$observado, previsoes_v2$previsto)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2
##          1 270  74
##          2 130 176
##
##                Accuracy : 0.6862
##                  95% CI : (0.6489, 0.7217)
##     No Information Rate : 0.6154
##     P-Value [Acc > NIR] : 0.0001018
##
##                   Kappa : 0.3637
##
##  Mcnemar's Test P-Value : 0.0001177
##
##             Sensitivity : 0.6750
##             Specificity : 0.7040
##          Pos Pred Value : 0.7849
##          Neg Pred Value : 0.5752
##              Prevalence : 0.6154
##          Detection Rate : 0.4154
```

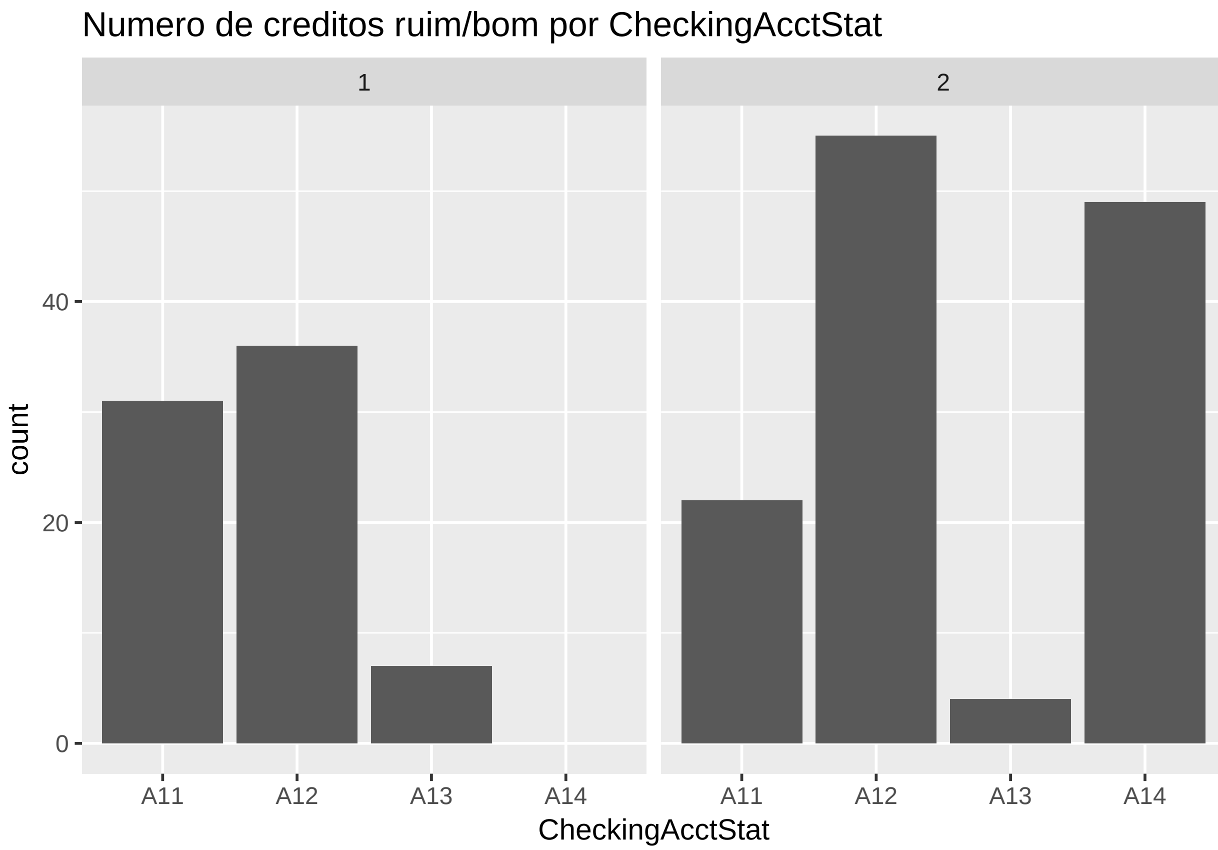
```
##    Detection Prevalence : 0.5292
##       Balanced Accuracy : 0.6895
##
##        'Positive' Class : 1
##
```

```r
# Analisando o resultado atraves de gráficos (bônus extra)

# Alterando atribuição da variável compFrame
compFrame <- previsoes_v2

# Usando o dplyr para filter linhas com classificação incorreta
require(dplyr)
```

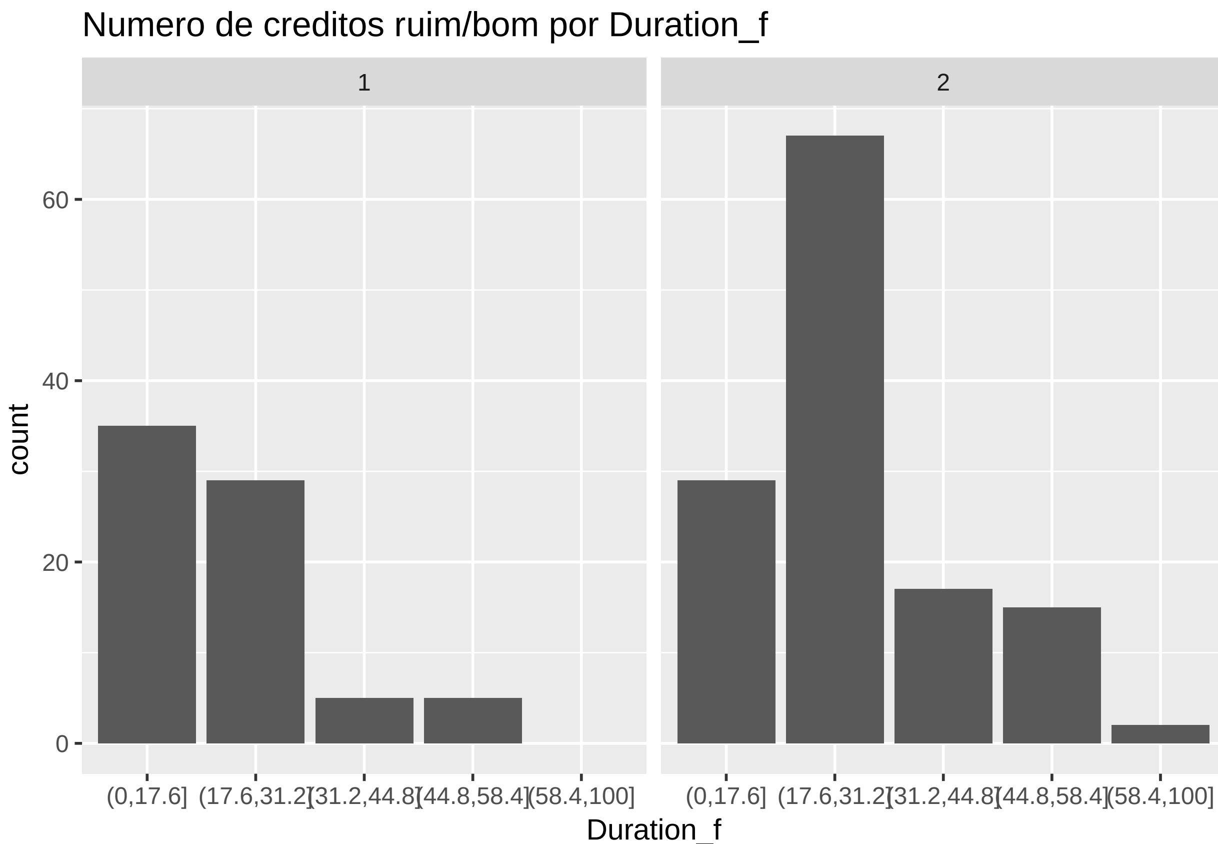
```
## Loading required package: dplyr

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:randomForest':
##
##     combine

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

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

```r
creditTest <- cbind(dados_teste, scored = compFrame[ ,2] )
creditTest <- creditTest %>% filter(CreditStatus != scored)

# Plot dos residuos para os niveis de cada fator
require(ggplot2)
colNames <- c("CheckingAcctStat", "Duration_f", "Purpose",
              "CreditHistory", "SavingsBonds", "Employment",
              "CreditAmount_f", "Employment")

lapply(colNames, function(x){
  if(is.factor(creditTest[,x])) {
    ggplot(creditTest, aes_string(x)) +
      geom_bar() +
      facet_grid(. ~ CreditStatus) +
      ggtitle(paste("Numero de creditos ruim/bom por",x))}})
```

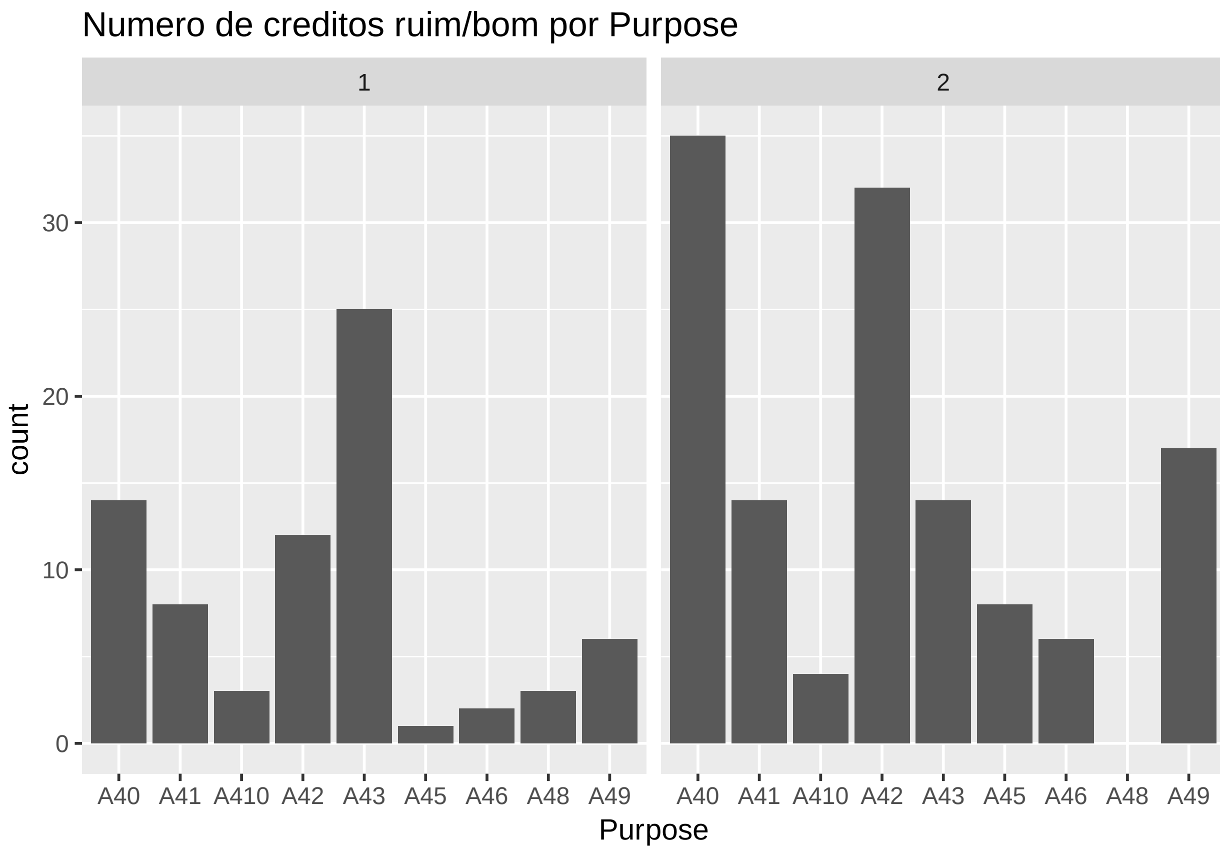
```
## [[1]]
```


Numero de creditos ruim/bom por CheckingAcctStat
1
2
count
40
20
0
A11
A12
A13
A14
A11
A12
A13
A14
CheckingAcctStat

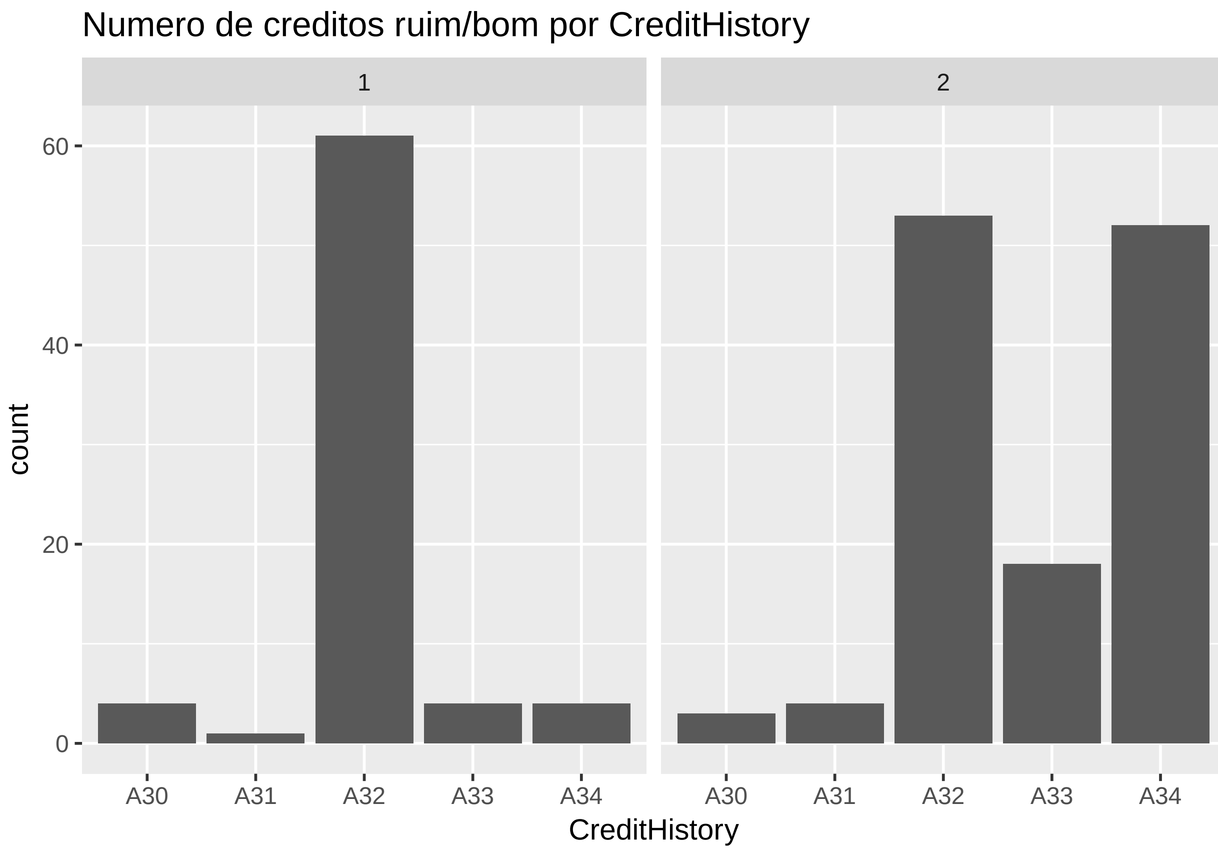
```
##
## [[2]]
```


Numero de creditos ruim/bom por Duration_f
1
2
count
60
40
20
0
(0,17.6]
(17.6,31.2]
(31.2,44.8]
(44.8,58.4]
(58.4,100]
Duration_f

```
##
## [[3]]
```

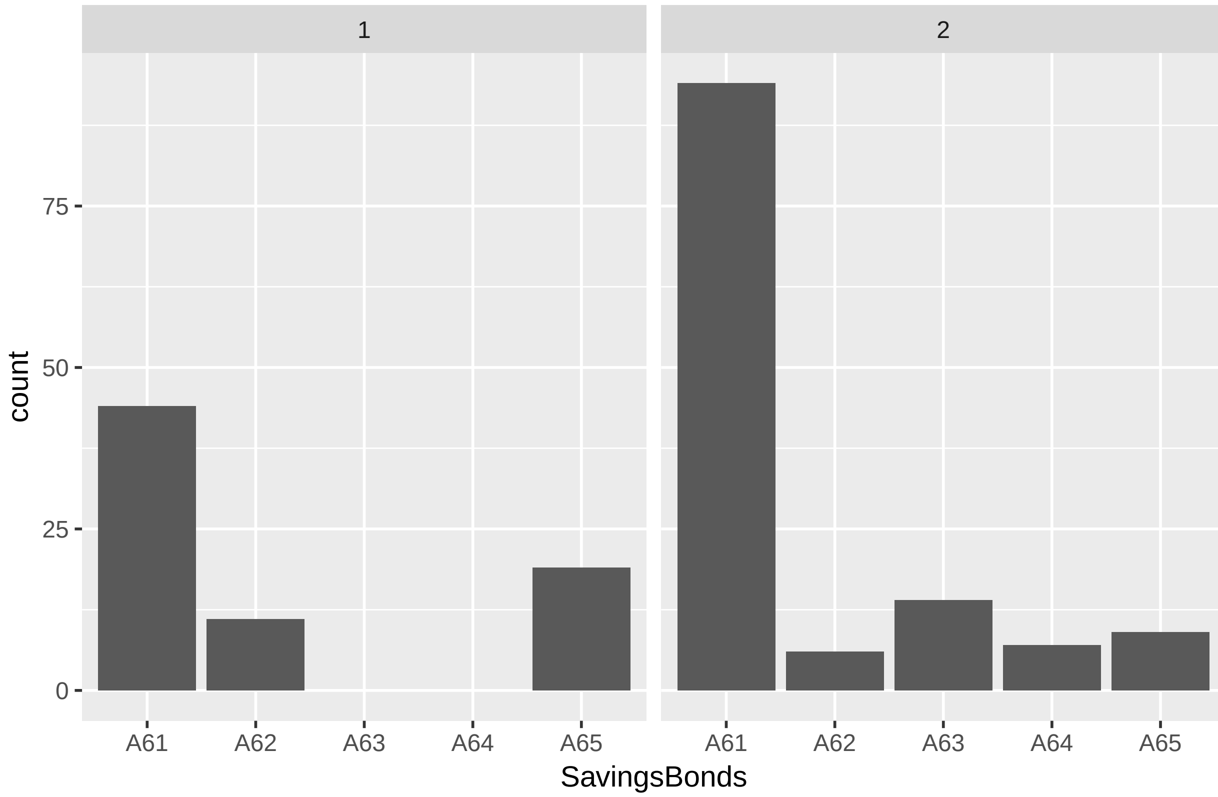

Numero de creditos ruim/bom por Purpose
1
2
count
30
20
10
0
A40
A41
A410
A42
A43
A45
A46
A48
A49
Purpose

```
##
## [[4]]
```

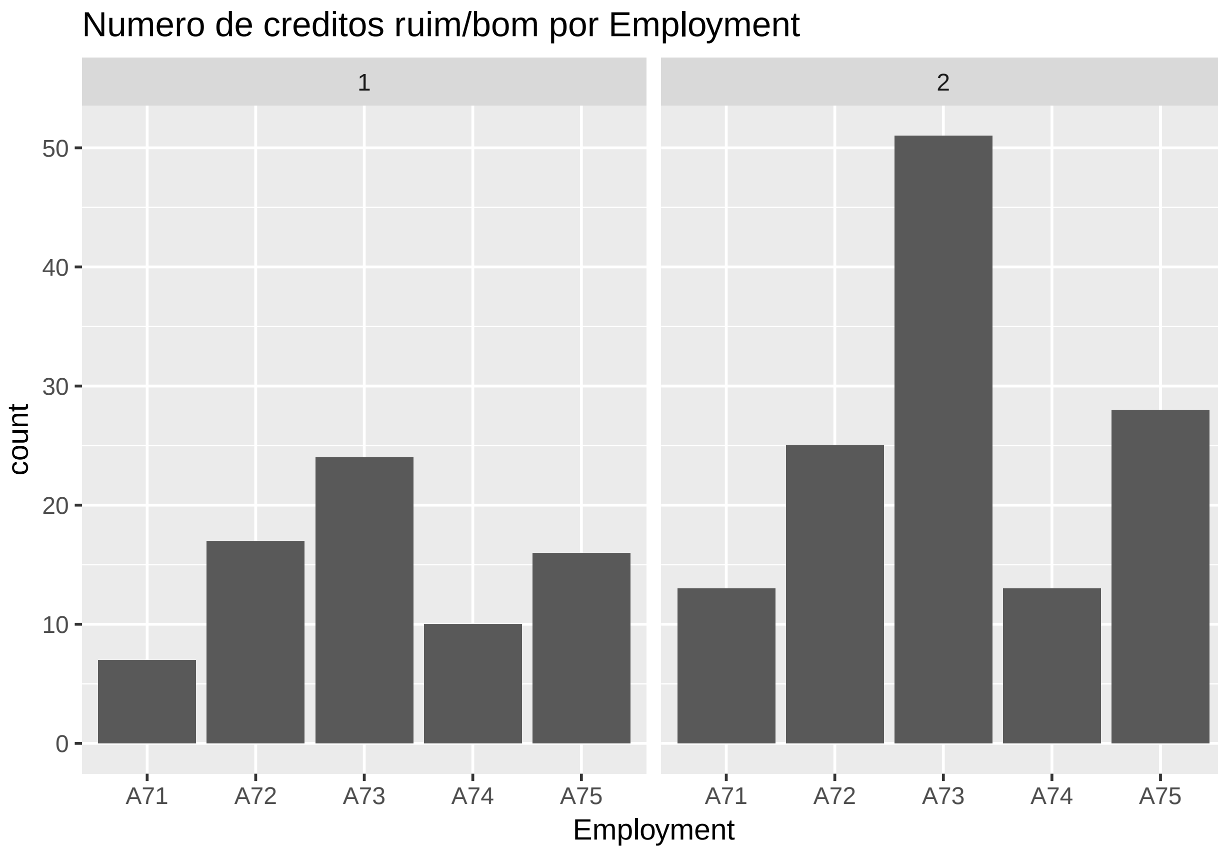

Numero de creditos ruim/bom por CreditHistory
1
2
count
60
40
20
0
A30
A31
A32
A33
A34
CreditHistory

```
##
## [[5]]
```

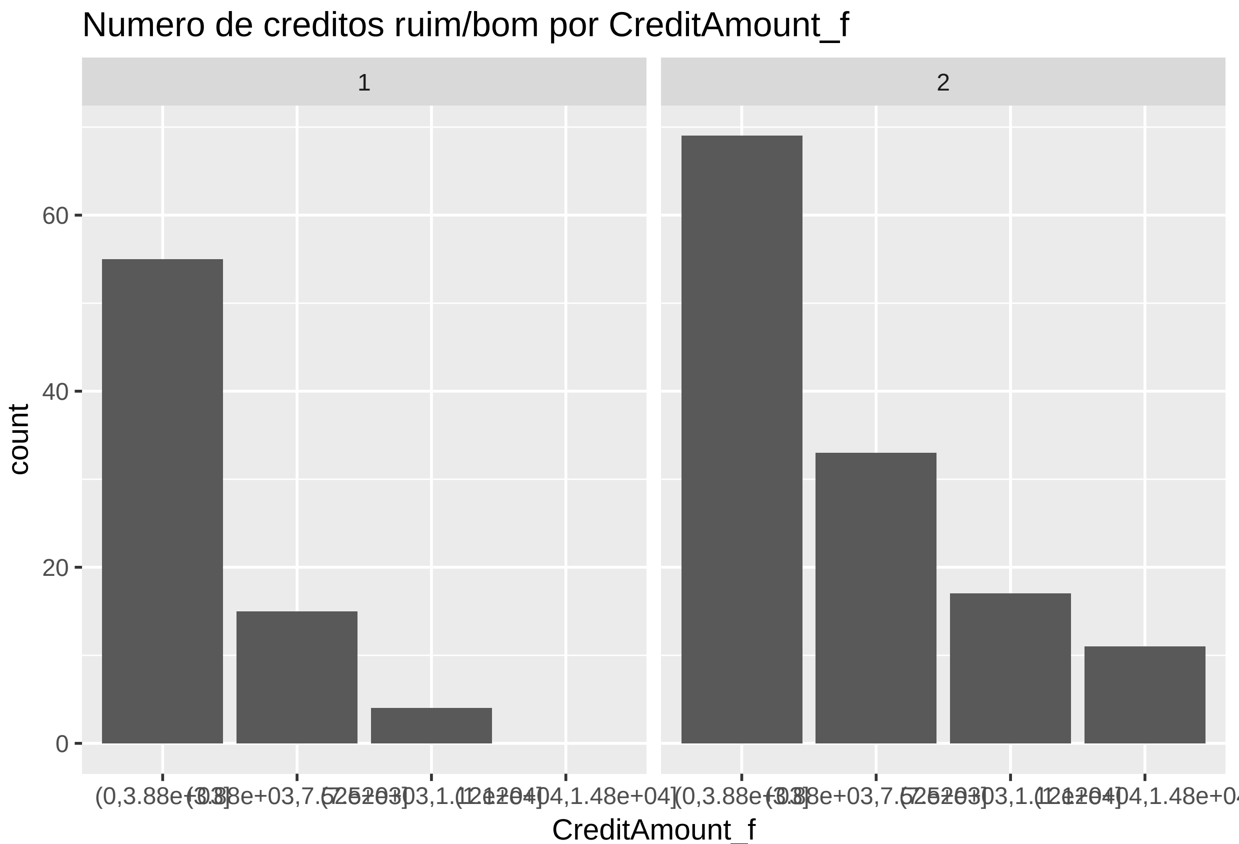
Numero de creditos ruim/bom por SavingsBonds



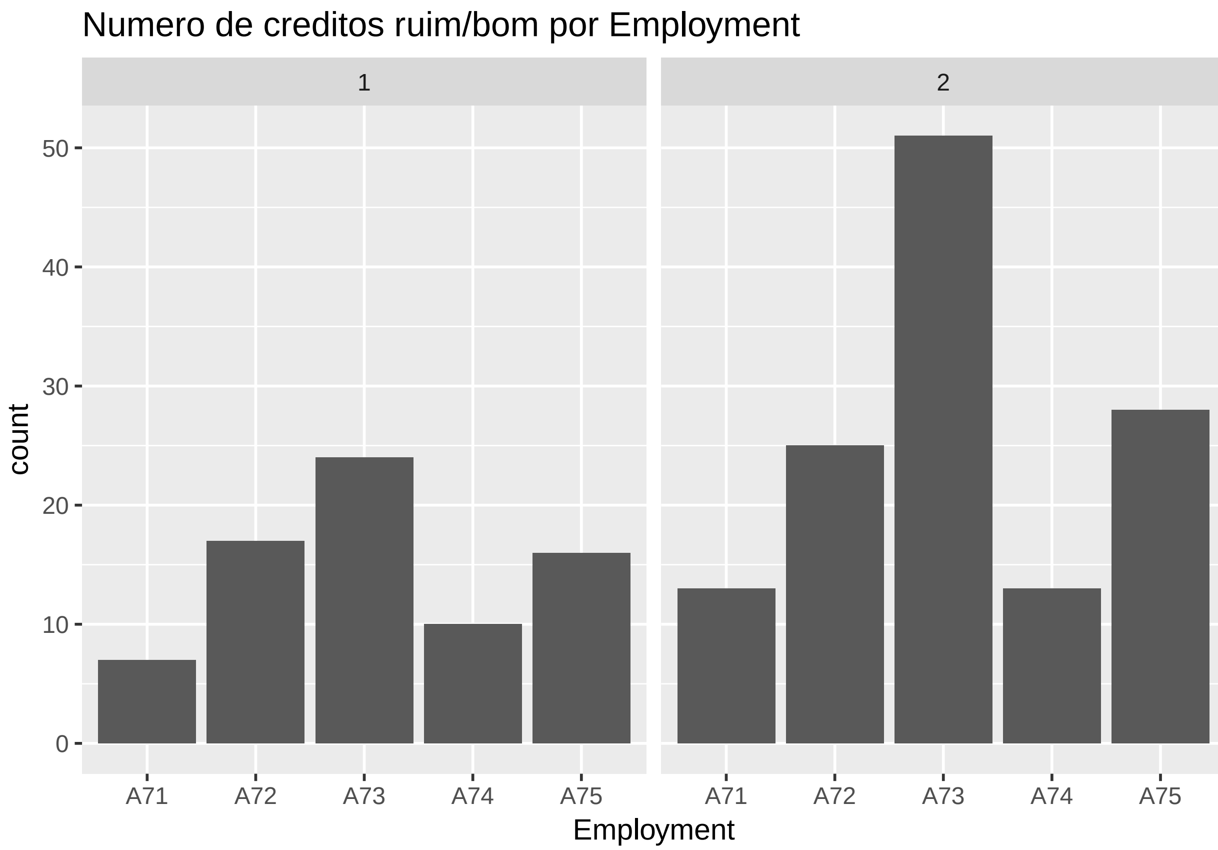

```
##
## [[6]]
```


Numero de creditos ruim/bom por Employment
1
2
50
40
30
20
10
0
count
A71
A72
A73
A74
A75
A71
A72
A73
A74
A75
Employment

```
##
## [[7]]
```

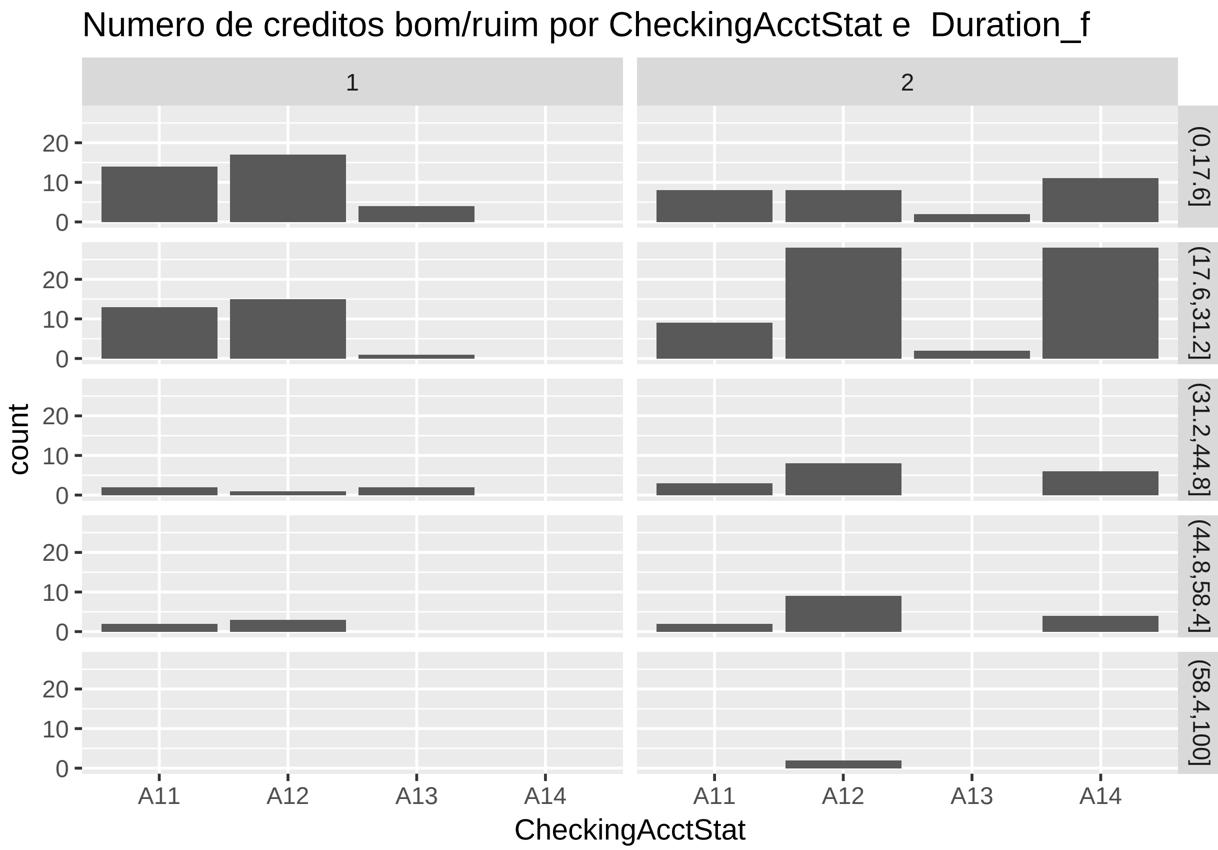

Numero de creditos ruim/bom por CreditAmount_f
1
2
count
60
40
20
0
CreditAmount_f

```
##
## [[8]]
```



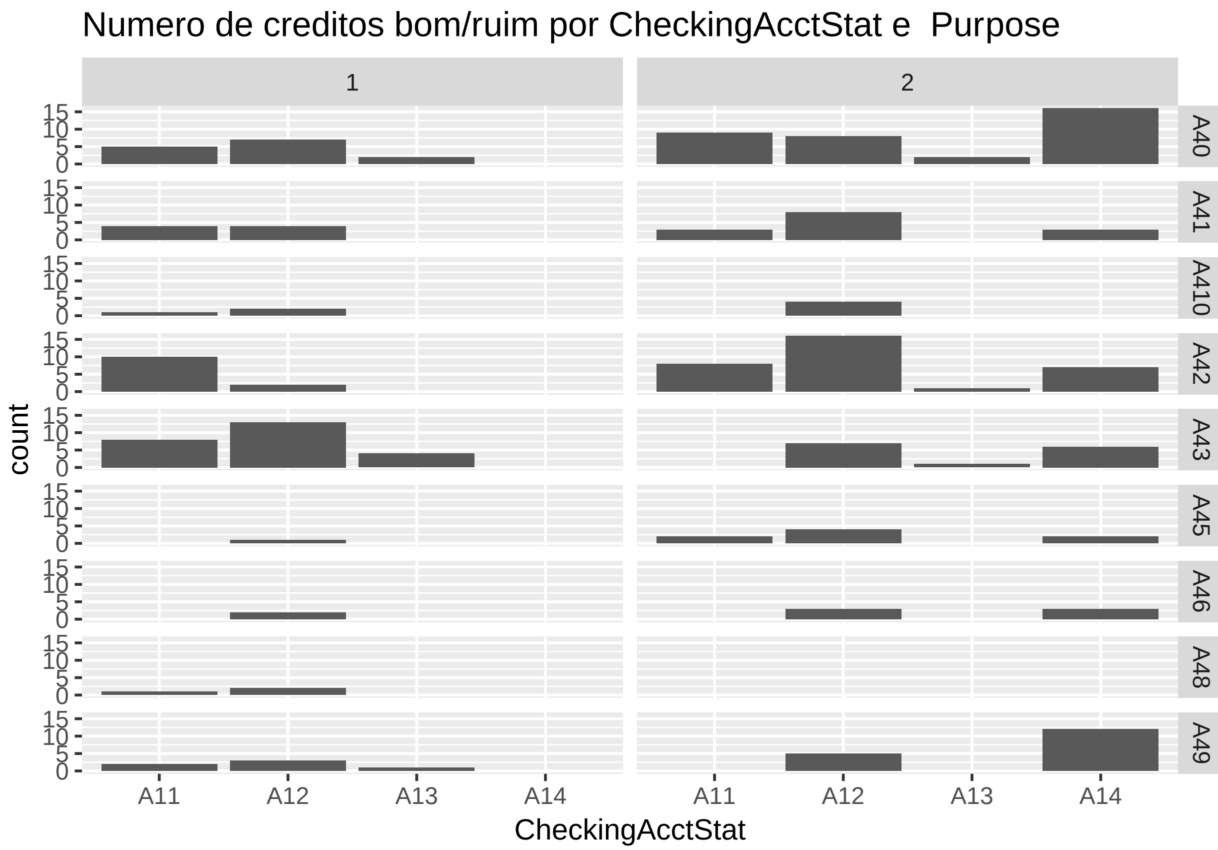

```
# Plot dos residuos condicionados nas variáveis CreditStatus vs CheckingAcctStat
lapply(colNames, function(x){
  if(is.factor(creditTest[,x]) & x != "CheckingAcctStat") {
    ggplot(creditTest, aes(CheckingAcctStat)) +
      geom_bar() +
      facet_grid(paste(x, " ~ CreditStatus"))+
      ggtitle(paste("Numero de creditos bom/ruim por CheckingAcctStat e ",x))
  }})
```

```
## [[1]]
## NULL
##
## [[2]]
```



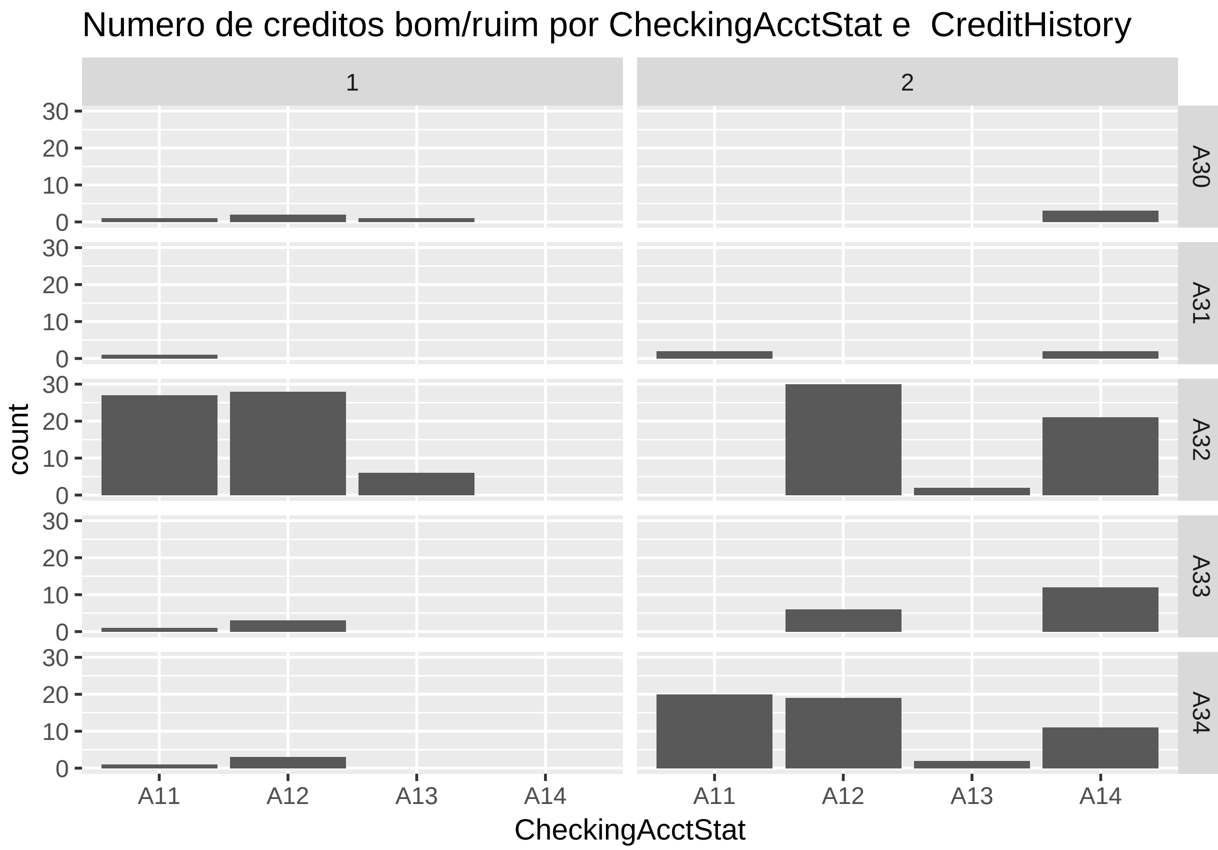
Numero de creditos bom/ruim por CheckingAcctStat e Duration_f
1
2
(0,17.6]
(17.6,31.2]
(31.2,44.8]
(44.8,58.4]
(58.4,100]
count
0
10
20
A11
A12
A13
A14
CheckingAcctStat


```
##
## [[3]]
```



Numero de creditos bom/ruim por CheckingAcctStat e Purpose
1
2
A40
A41
A410
A42
A43
A45
A46
A48
A49
count
15
10
5
0
A11
A12
A13
A14
CheckingAcctStat


```
##
## [[4]]
```


Numero de creditos bom/ruim por CheckingAcctStat e CreditHistory
1
2
A30
A31
A32
A33
A34
count
30
20
10
0
A11
A12
A13
A14
CheckingAcctStat

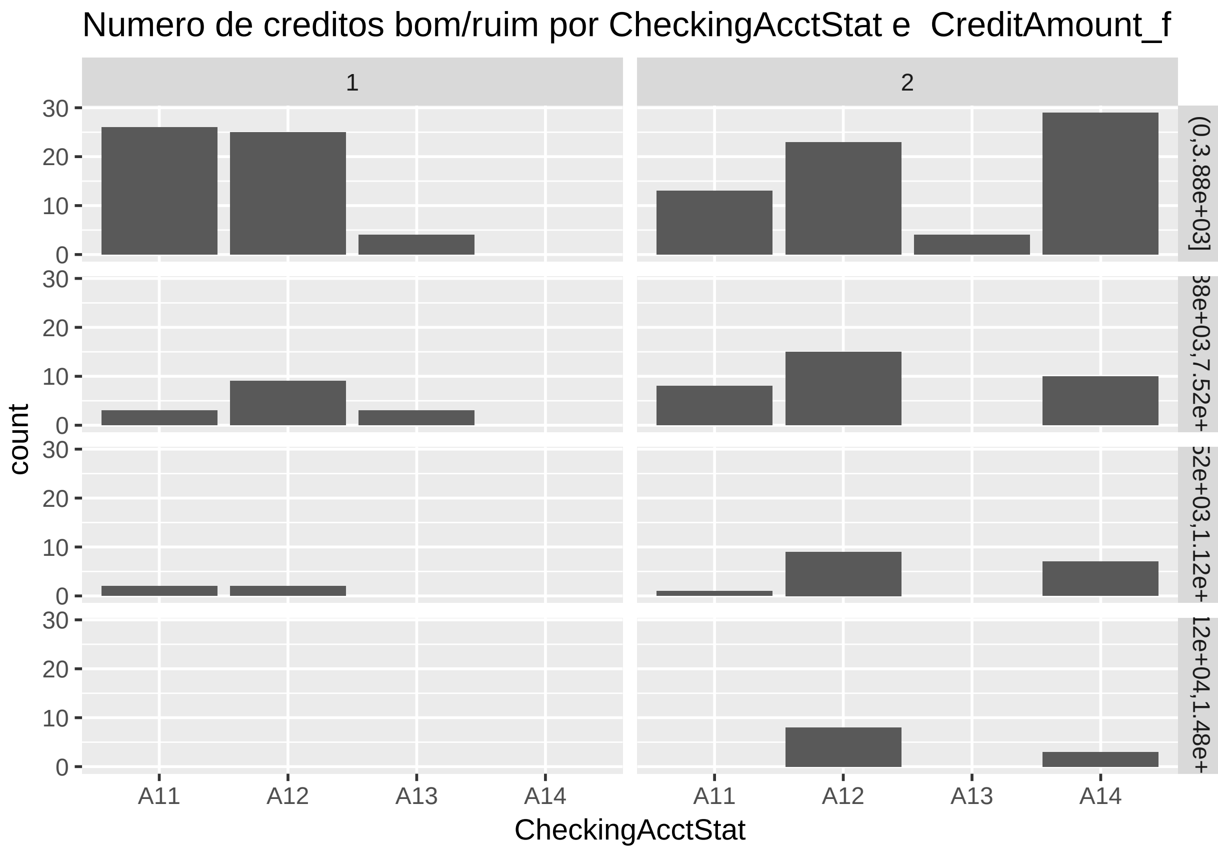
```
##
## [[5]]
```


Numero de creditos bom/ruim por CheckingAcctStat e SavingsBonds
1
2
A61
A62
A63
A64
A65
count
40
30
20
10
0
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[6]]
```


Numero de creditos bom/ruim por CheckingAcctStat e Employment
1
2
A71
A72
A73
A74
A75
count
20
15
10
5
0
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[7]]
```


Numero de creditos bom/ruim por CheckingAcctStat e CreditAmount_f
1
2
(0,3.88e+03]
38e+03,7.52e+
52e+03,1.12e+
12e+04,1.48e+
count
30
20
10
0
A11
A12
A13
A14
CheckingAcctStat

```
##
## [[8]]
```

Numero de creditos bom/ruim por CheckingAcctStat e Employment
1
2
A71
A72
A73
A74
A75
count
20
15
10
5
0
A11
A12
A13
A14
CheckingAcctStat