Risco de Crédito: Como uma Melhor Selecao de Variaveis Pode Ampliar a Eficacia do Modelo

Rafael Bicudo Rosa

24 de julho de 2018

Como a selecao de variaveis sozinha aumenta a eficacia de um modelo de classificacao qualquer

Este trabalho e uma releitura de um projeto integrante do curso Big Data Analytics com R e Microsoft Azure da Formacao Cientista de Dados. O objetivo e usar dados sobre analises de credito realizados na Alemanha, para, atraves de um modelo simples de classificacao para prever a qualidade do credito, ver a variacao de performance com uma melhor selecao de variaveis mais explicativas.

Os dados de credito incluem 1000 observacoes de concessao de credito, cada uma com 21 variaveis, sendo a ultima a classificacao do solicitante (bom ou mau pagador), e as restantes caracteristicas qualitativas e quantitativas sobre esses mesmos. Todas as informacoes foram retiradas do repositorio online da Universidade de Irvine, California (https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data), assim como uma melhor explicacao do significado das variaveis.

Etapa 1 - Coleta dos Dados

Assim como descrito acima, os dados serão retirados de um repositorio online contendo a base em si no formato table, e a informacao de cada uma das caracteristicas. Em seguida, as variaveis serao nomeadas e, por fim, ter-se-a a primeira visao do dataframe.

```
## Obtencao dos dados

# Carrega o dataset antes da transformacao
german_credit_l <- 'https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data
Credit <- read.table(german_credit_l)

# Nome das variaveis
names(Credit) <- c('CheckingAcctStat', 'Duration', 'CreditHistory', 'Purpose', 'CreditAmount',
                   'SavingsBonds', 'Employment', 'InstallmentRatePecnt', 'SexAndStatus',
                   'OtherDetorsGuarantors', 'PresentResidenceTime', 'Property', 'Age',
                   'OtherInstallments', 'Housing', 'ExistingCreditsAtBank', 'Job', 'NumberDependents',
                   'Telephone', 'ForeignWorker', 'CreditStatus')

# Analise do dataframe
str(Credit)
```

```
## 'data.frame':    1000 obs. of  21 variables:
##  $ CheckingAcctStat     : Factor w/ 4 levels "A11","A12","A13",..: 1 2 4 1 1 4 4 2 4 2 ...
##  $ Duration             : int  6 48 12 42 24 36 24 36 12 30 ...
##  $ CreditHistory        : Factor w/ 5 levels "A30","A31","A32",..: 5 3 5 3 4 3 3 3 3 5 ...
##  $ Purpose              : Factor w/ 10 levels "A40","A41","A410",..: 5 5 8 4 1 8 4 2 5 1 ...
##  $ CreditAmount         : int  1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...
##  $ SavingsBonds         : Factor w/ 5 levels "A61","A62","A63",..: 5 1 1 1 1 5 3 1 4 1 ...
##  $ Employment           : Factor w/ 5 levels "A71","A72","A73",..: 5 3 4 4 3 3 5 3 4 1 ...
##  $ InstallmentRatePecnt : int  4 2 2 2 3 2 3 2 2 4 ...
```

```
##  $ SexAndStatus         : Factor w/ 4 levels "A91","A92","A93",..: 3 2 3 3 3 3 3 3 1 4 ...
##  $ OtherDetorsGuarantors: Factor w/ 3 levels "A101","A102",..: 1 1 1 3 1 1 1 1 1 1 ...
##  $ PresentResidenceTime : int  4 2 3 4 4 4 4 2 4 2 ...
##  $ Property             : Factor w/ 4 levels "A121","A122",..: 1 1 1 2 4 4 2 3 1 3 ...
##  $ Age                  : int  67 22 49 45 53 35 53 35 61 28 ...
##  $ OtherInstallments    : Factor w/ 3 levels "A141","A142",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Housing              : Factor w/ 3 levels "A151","A152",..: 2 2 2 3 3 3 2 1 2 2 ...
##  $ ExistingCreditsAtBank: int  2 1 1 1 2 1 1 1 1 2 ...
##  $ Job                  : Factor w/ 4 levels "A171","A172",..: 3 3 2 3 3 2 3 4 2 4 ...
##  $ NumberDependents     : int  1 1 2 2 2 2 1 1 1 1 ...
##  $ Telephone            : Factor w/ 2 levels "A191","A192": 2 1 1 1 1 2 1 2 1 1 ...
##  $ ForeignWorker        : Factor w/ 2 levels "A201","A202": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CreditStatus         : int  1 2 1 1 2 1 1 1 1 2 ...
```

```
summary(Credit)
```

```
##  CheckingAcctStat    Duration    CreditHistory    Purpose
##  A11:274          Min.   : 4.0   A30: 40       A43    :280
##  A12:269          1st Qu.:12.0   A31: 49       A40    :234
##  A13: 63          Median :18.0   A32:530       A42    :181
##  A14:394          Mean   :20.9   A33: 88       A41    :103
##                   3rd Qu.:24.0   A34:293       A49    : 97
##                   Max.   :72.0                 A46    : 50
##                                                (Other): 55
##   CreditAmount   SavingsBonds Employment InstallmentRatePecnt SexAndStatus
##  Min.   :  250   A61:603      A71: 62    Min.   :1.000        A91: 50
##  1st Qu.: 1366   A62:103      A72:172    1st Qu.:2.000        A92:310
##  Median : 2320   A63: 63      A73:339    Median :3.000        A93:548
##  Mean   : 3271   A64: 48      A74:174    Mean   :2.973        A94: 92
##  3rd Qu.: 3972   A65:183      A75:253    3rd Qu.:4.000
##  Max.   :18424                           Max.   :4.000
##
##  OtherDetorsGuarantors PresentResidenceTime Property        Age
##  A101:907              Min.   :1.000        A121:282   Min.   :19.00
##  A102: 41              1st Qu.:2.000        A122:232   1st Qu.:27.00
##  A103: 52              Median :3.000        A123:332   Median :33.00
##                        Mean   :2.845        A124:154   Mean   :35.55
##                        3rd Qu.:4.000                   3rd Qu.:42.00
##                        Max.   :4.000                   Max.   :75.00
##
##  OtherInstallments Housing    ExistingCreditsAtBank   Job
##  A141:139          A151:179   Min.   :1.000         A171: 22
##  A142: 47          A152:713   1st Qu.:1.000         A172:200
##  A143:814          A153:108   Median :1.000         A173:630
##                               Mean   :1.407         A174:148
##                               3rd Qu.:2.000
##                               Max.   :4.000
##
##  NumberDependents Telephone  ForeignWorker  CreditStatus
##  Min.   :1.000    A191:596   A201:963      Min.   :1.0
##  1st Qu.:1.000    A192:404   A202: 37      1st Qu.:1.0
##  Median :1.000                             Median :1.0
##  Mean   :1.155                             Mean   :1.3
##  3rd Qu.:1.000                             3rd Qu.:2.0
##  Max.   :2.000                             Max.   :2.0
```

##

Etapa 2 - Limpeza e Preparacao dos dados

A partir do demonstrado acima, ve-se a existencia de algumas imperfeicoes, como a diferenca de grandezas entre as variaveis quantitativas, e algumas variaveis qualitativas como numericas, portanto se segue a uma etapa de ajustamento dos dados.

```
## Data Cleaning

# Definicao variavel de interesse
Credit[, 'CreditStatus'] <- factor(Credit[, 'CreditStatus'], labels = c('Good', 'Bad'))

# Funcao para automatizar "fatorizacao" das variaveis
to.factor <- function(df, features) {
  for (feature in features) {
    df[[feature]] <- as.factor(df[[feature]])
  }
  return(df)
}

# Criacao do string vector das variaveis a  serem fatorizadas e sua fatorizacao
categorical_vars <- c('CheckingAcctStat', 'CreditHistory', 'Purpose',
                      'SavingsBonds', 'Employment', 'InstallmentRatePecnt', 'SexAndStatus',
                      'OtherDetorsGuarantors', 'PresentResidenceTime', 'Property',
                      'OtherInstallments', 'Housing', 'ExistingCreditsAtBank', 'Job', 'NumberDependents'
                      'Telephone', 'ForeignWorker', 'CreditStatus')

Credit <- to.factor(Credit, categorical_vars)

# Funcao para automatizar normalizacao
scale.features <- function(df, variables){
  for (variable in variables){
    df[[variable]] <- scale(df[[variable]], center=T, scale=T)
  }
  return(df)
}

# Normalizacao das variaveis
numeric_vars <- c("Duration", "Age", "CreditAmount")
scale_Credit <- scale.features(Credit, numeric_vars)
```

Etapa 3 - Dividindo os dados em treino e teste

Com a preparacao dos dados concluida, pode-se prosseguir a separacao dos dados entre treino, para modelagem e exploracao, e teste, para verificacao da aprendizagem.

```
# Carregando pacotes necessarios
library(caret)
```

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

```
## Loading required package: ggplot2
```

```
# Separacao dos Sets de Treino e Teste
set.seed(666)
sample <- createDataPartition(scale_Credit$CreditStatus, times = 1, list = F, p = .6)
train_sample <- scale_Credit[sample, ]
test_sample <- scale_Credit[-sample, ]
```

Etapa 4 - Selecao das variaveis

Com todas as transformacoes concluidas, segue-se para o tema principal do projeto: selecao das variaveis mais explicativas. Para executar a tarefa, foi criada uma funcao para aplicar o metodo de selecao de variaveis recursiva usando modelos "Randon Forests", atraves do uso do pacote de Machine Learning Caret. A escolha do metodo se deve ao fato de ser um dos melhores algoritmos para modelos de classificacao. (Para mais informacoes sobre os processos disponiveis, checar a documentacao do pacote Caret: http://topepo.github.io/caret/recursive-feature-elimination.html#backwards-selection)

```
## Feature Selection

# Carregando pacotes necessarios
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
```

```
library(ggplot2)

# Funcao para selecao das variaveis
rfe.feature.selection <- function(num_iters=20, features, target){
  variable_sizes <- 1:10
  control <- rfeControl(functions = rfFuncs, method = "cv",
                        verbose = FALSE, returnResamp = "all",
                        number = num_iters)
  rfe_results <- rfe(x = features, y = target,
                     sizes = variable_sizes,
                     rfeControl = control)
  return(rfe_results)
}

# Executando a funcao e para obter features mais explicativas
rfe_results <- rfe.feature.selection(features = train_sample[,-21],
                                     target = train_sample[,21])

# Selecao das Features mais significates e visualizacao da significancia
rfe_results
```

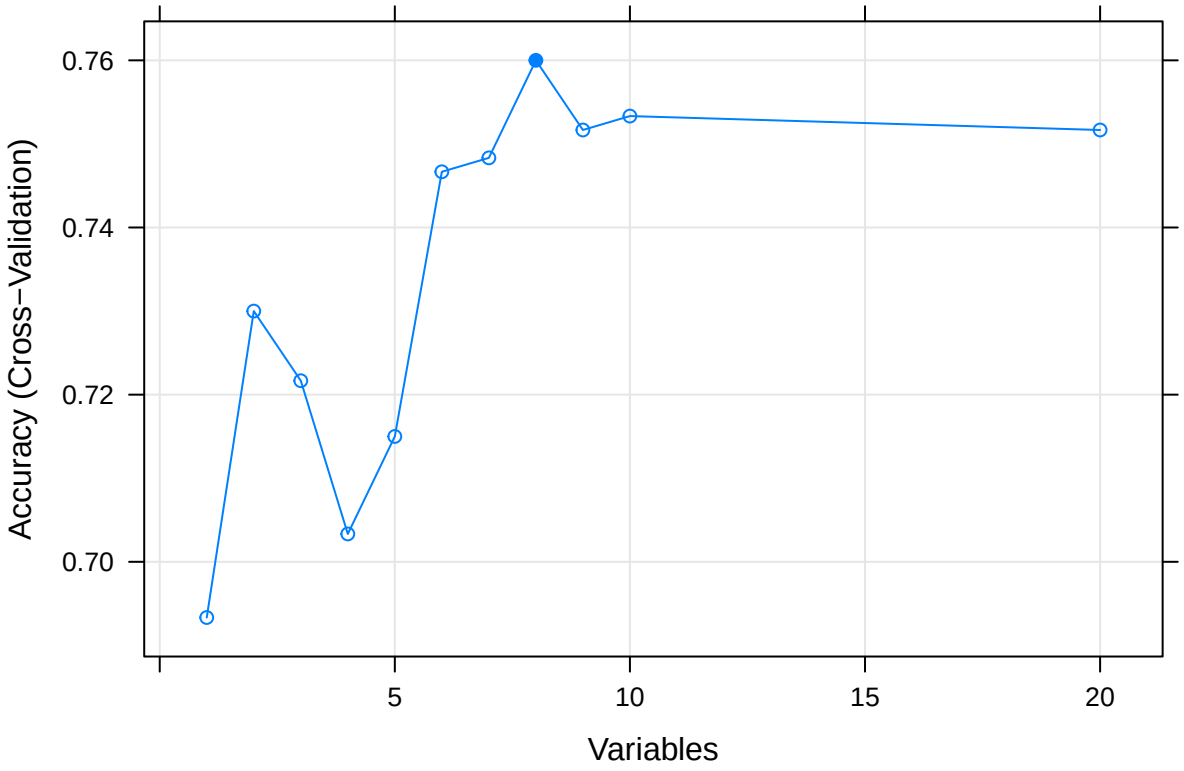
```
##
## Recursive feature selection
##
```

```
## Outer resampling method: Cross-Validated (20 fold)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          1   0.6933 0.2085    0.06719  0.2025
##          2   0.7300 0.2851    0.06389  0.2099
##          3   0.7217 0.2487    0.05437  0.1783
##          4   0.7033 0.2458    0.06831  0.1745
##          5   0.7150 0.2617    0.06163  0.1606
##          6   0.7467 0.3292    0.07446  0.2144
##          7   0.7483 0.3281    0.07452  0.2029
##          8   0.7600 0.3570    0.07383  0.2161        *
##          9   0.7517 0.3576    0.07452  0.1992
##         10   0.7533 0.3546    0.07829  0.2145
##         20   0.7517 0.3218    0.06965  0.1956
##
## The top 5 variables (out of 8):
##    CheckingAcctStat, Duration, CreditHistory, CreditAmount, SavingsBonds
```

```
varImp((rfe_results), scale = F)
```

```
##                         Overall
## CheckingAcctStat      20.486732
## Duration              10.622675
## CreditHistory          6.822450
## CreditAmount           6.802129
## SavingsBonds           6.477065
## OtherDetorsGuarantors  5.746643
## Purpose                5.509115
## OtherInstallments      4.806770
## Property               4.770802
## Employment             4.769671
```

```
optVariables <- rfe_results[["optVariables"]]
plot(rfe_results, type=c("g", "o"))
```

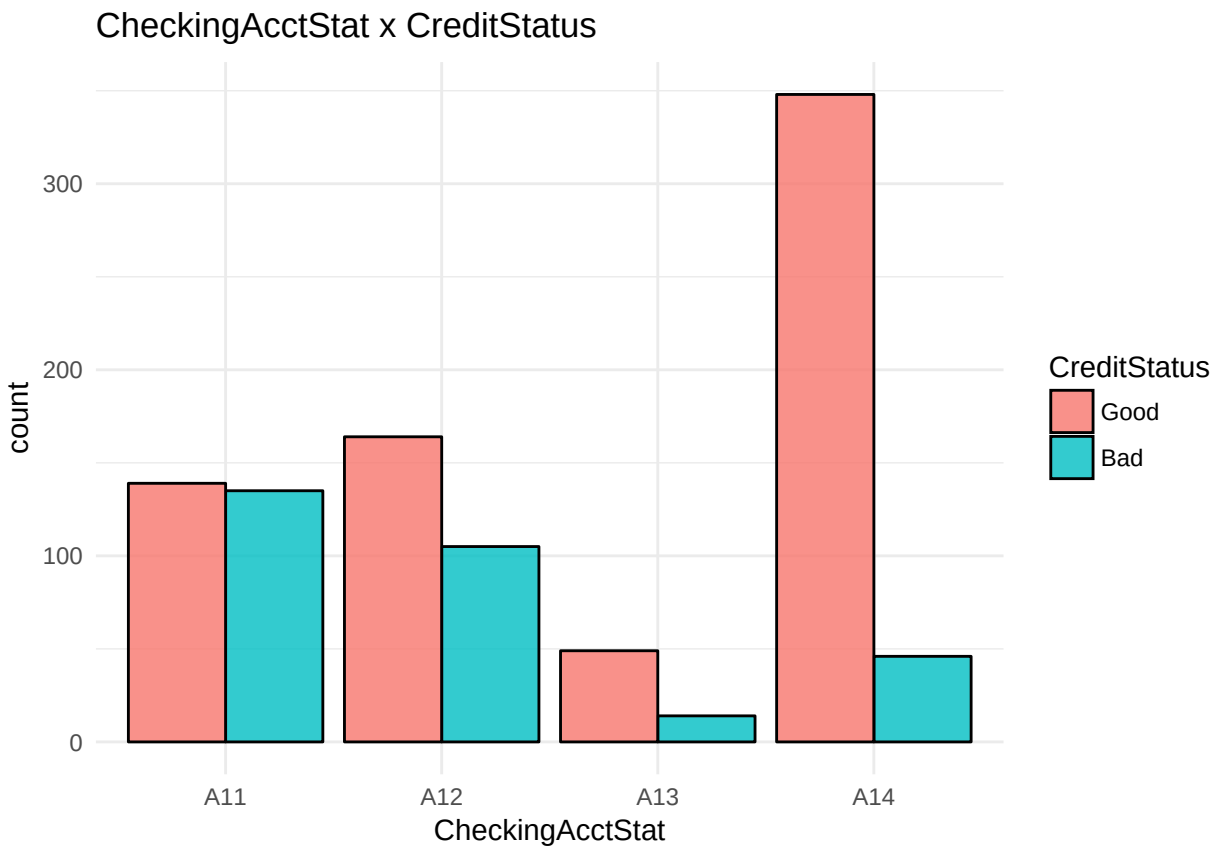



```
optVariables
```

```
## [1] "CheckingAcctStat"      "Duration"              "CreditHistory"
## [4] "CreditAmount"          "SavingsBonds"          "OtherDetorsGuarantors"
## [7] "Purpose"               "Employment"
```

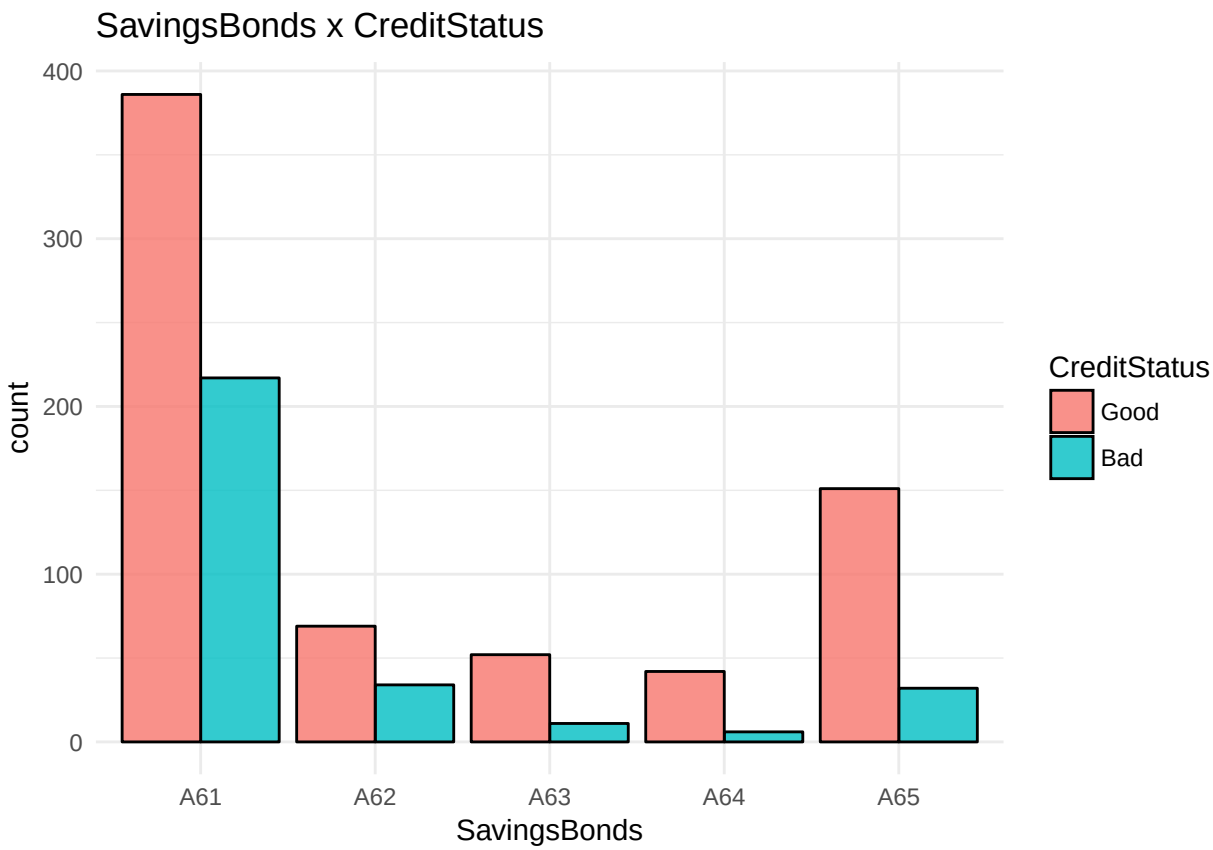
O grafico acima demonstra como o poder explicativo do modelo varia atraves da inclusao de mais caracteristicas, chegando ao seu numero ótimo e quais são essas. Para ilustrar melhor seu poder explicativo na pratica, seguem, abaixo, graficos entre as variveis explicativas citadas e nosso alvo.

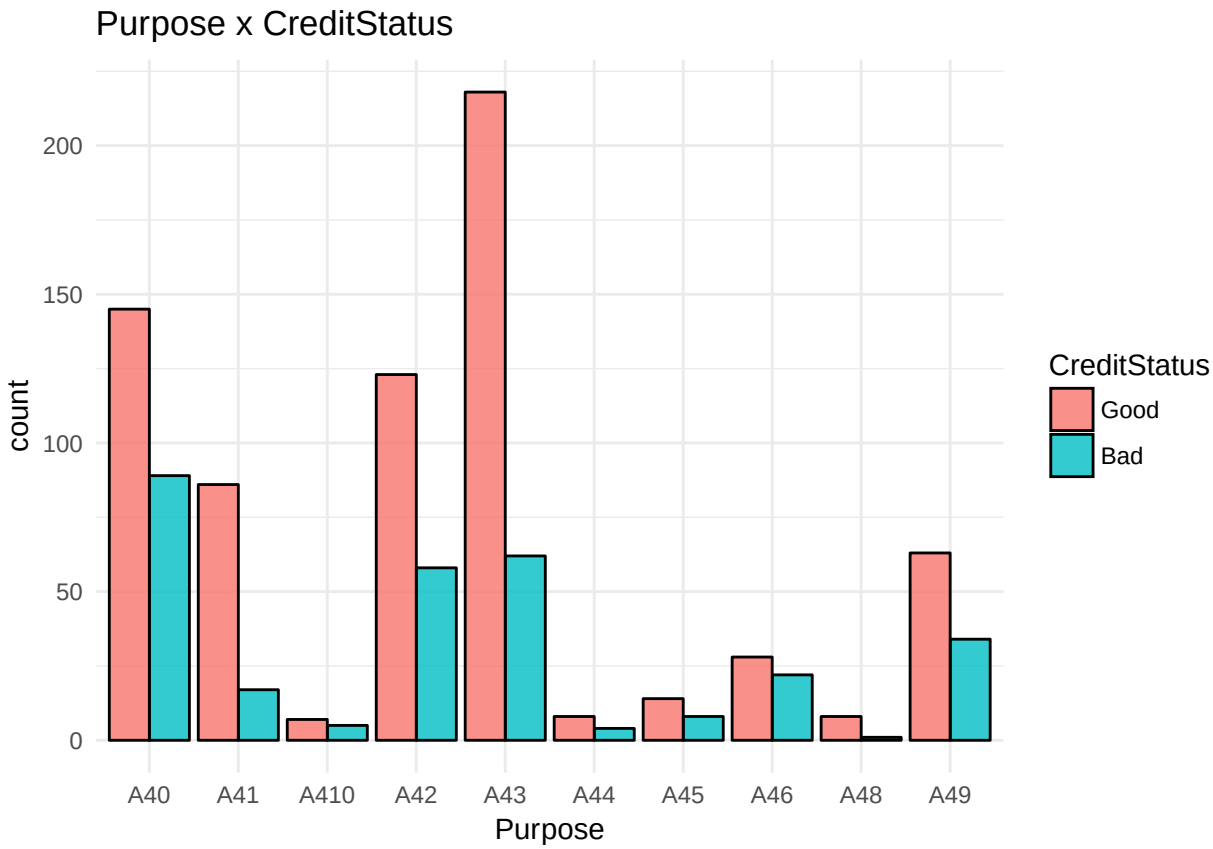
```
# Plot das variaveis otimas

# Categoricas
plots_cat<- list()
for (i in c('CheckingAcctStat', 'CreditHistory','OtherDetorsGuarantors', 'SavingsBonds', 'Purpose',
            'Employment')) {
  plots_cat[[i]] <- ggplot(Credit, aes_string(x = i, fill = 'CreditStatus')) +
  geom_bar(alpha=0.8, colour='black', position = 'dodge') + ggtitle(paste(i, 'x CreditStatus')) +
  theme_minimal()
  print(plots_cat[[i]])
}
```

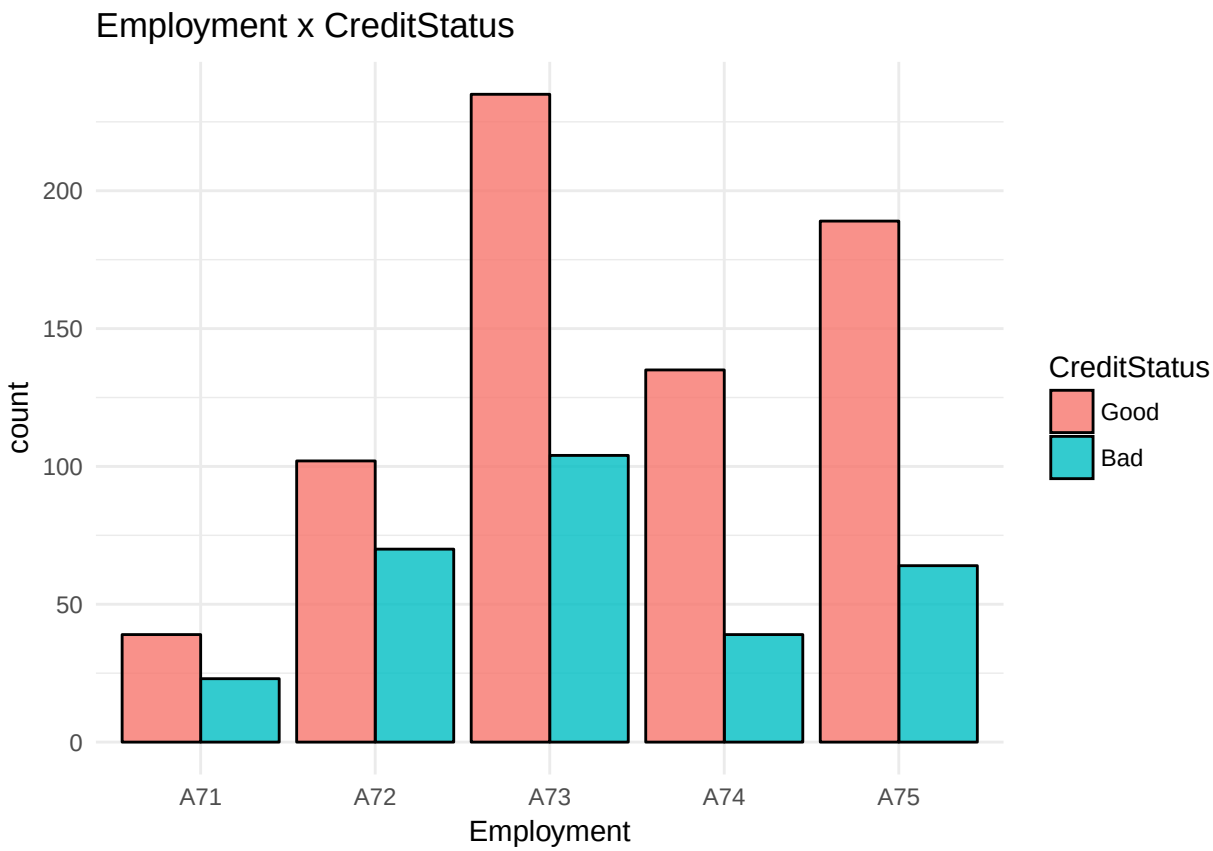

CheckingAcctStat x CreditStatus
300
200
100
0
count
A11
A12
A13
A14
CheckingAcctStat
CreditStatus
Good
Bad


CreditHistory x CreditStatus
count
300
200
100
0
A30
A31
A32
A33
A34
CreditHistory
CreditStatus
Good
Bad

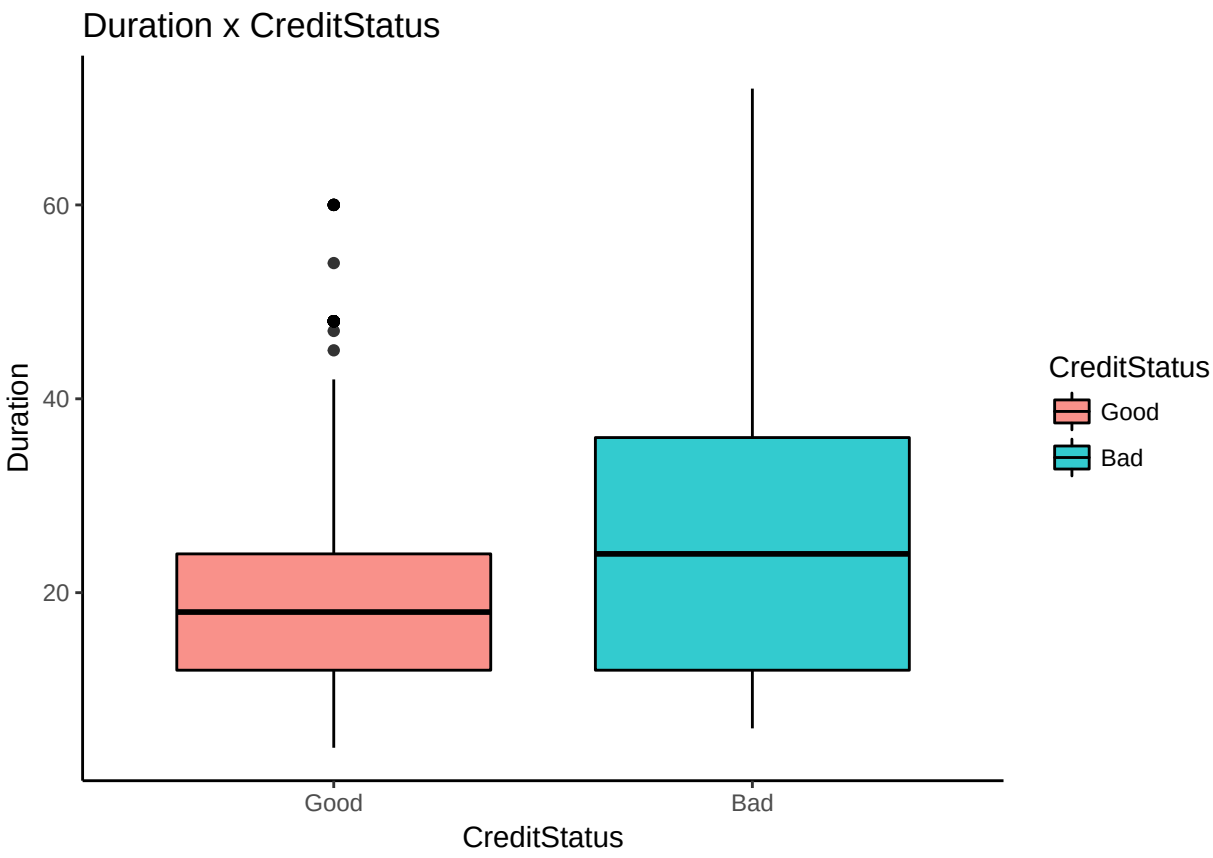

OtherDetorsGuarantors x CreditStatus
600
400
200
0
count
A101
A102
A103
OtherDetorsGuarantors
CreditStatus
Good
Bad

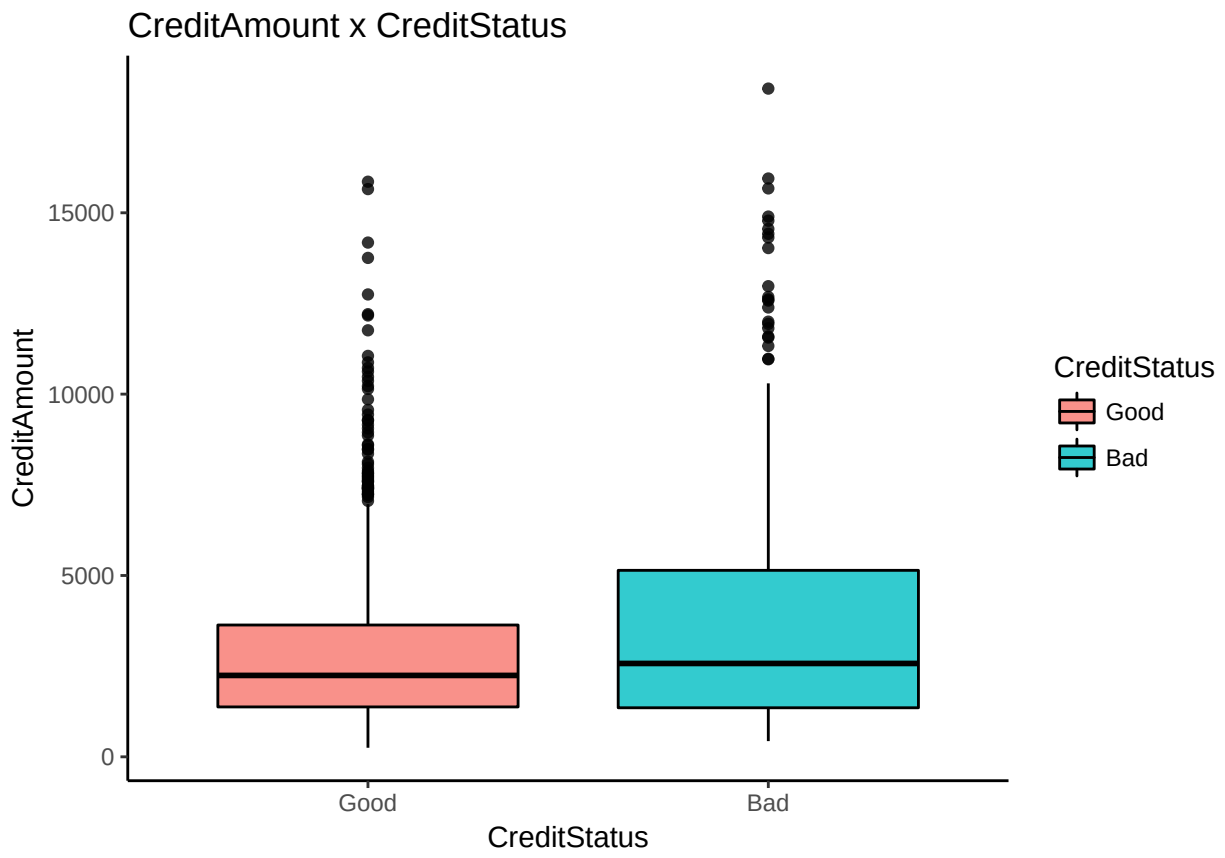

SavingsBonds x CreditStatus
400
300
200
100
0
count
A61
A62
A63
A64
A65
SavingsBonds
CreditStatus
Good
Bad


Purpose x CreditStatus
count
200
150
100
50
0
A40
A41
A410
A42
A43
A44
A45
A46
A48
A49
Purpose
CreditStatus
Good
Bad




```
# Continua (Duration, CreditAmount, Age)
plots_cont<- list()
for (i in c('Duration', 'CreditAmount')) {
  plots_cont[[i]] <- ggplot(Credit, aes_string(x = 'CreditStatus', y = i, fill = 'CreditStatus')) +
    geom_boxplot(alpha=0.8, colour='black', position = 'dodge') + ggtitle(paste(i, 'x CreditStatus')) +
    theme_classic()
  print(plots_cont[[i]])
}
```


Duration x CreditStatus
Duration
60
40
20
Good
Bad
CreditStatus
CreditStatus
Good
Bad



Etapa 5 - Criacao do controle dos modelos

Para prosseguir a criacao dos modelos de forma isenta, um controle unico sera criado que permitira usar as mesmas validacoes cruzadas em ambos os testes.

```
# Controles para certificar testes imparciais

# Particoes da data para reutilizar nos cvs
myFolds <- createFolds(train_sample$CreditStatus, k = 5)

# Controle dos modelos de treino
train_control <- trainControl(method = "cv",
                              index = myFolds,
                              returnResamp = "all",
                              classProbs = TRUE,
                              summaryFunction = twoClassSummary,
                              verboseIter = TRUE,
                              savePredictions = TRUE)
```

Etapa 6 - Execucao dos Modelos

```
## Modelo de classificacao sem feature selection
glmModel_full <- train(CreditStatus ~ ., data = train_sample,
                       method = 'glm',
```

```
                        metric = "ROC",
                        trControl = train_control)

# Predicao 1
glmModel_full_pred <-predict(glmModel_full, test_sample)

## Modelo de classificacao com feature selection
glmModel_fs <- train(CreditStatus ~
                        CheckingAcctStat +
                        Duration +
                        OtherDetorsGuarantors +
                        CreditAmount +
                        CreditHistory +
                        SavingsBonds +
                        Purpose +
                        Employment,
                        data = train_sample,
                        method = 'glm',
                        metric = "ROC",
                        trControl = train_control)

# Avaliacao 2
glmModel_fs_pred <-predict(glmModel_fs, test_sample)
```

Etapa 7 - Tabelas de Confusao

```
confusionMatrix(glmModel_full_pred, test_sample$CreditStatus)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Good Bad
##       Good  230  64
##       Bad    50  56
##
##                Accuracy : 0.715
##                  95% CI : (0.668, 0.7588)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.2758
##
##                   Kappa : 0.298
##  Mcnemar's Test P-Value : 0.2234
##
##             Sensitivity : 0.8214
##             Specificity : 0.4667
##          Pos Pred Value : 0.7823
##          Neg Pred Value : 0.5283
##              Prevalence : 0.7000
##          Detection Rate : 0.5750
##    Detection Prevalence : 0.7350
##       Balanced Accuracy : 0.6440
##
```

```
##        'Positive' Class : Good
##
```

```
confusionMatrix(glmModel_fs_pred, test_sample$CreditStatus)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Good Bad
##       Good  234  64
##       Bad    46  56
##
##                Accuracy : 0.725
##                  95% CI : (0.6784, 0.7682)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.1498
##
##                   Kappa : 0.3159
##  Mcnemar's Test P-Value : 0.1050
##
##             Sensitivity : 0.8357
##             Specificity : 0.4667
##          Pos Pred Value : 0.7852
##          Neg Pred Value : 0.5490
##              Prevalence : 0.7000
##          Detection Rate : 0.5850
##    Detection Prevalence : 0.7450
##       Balanced Accuracy : 0.6512
##
##        'Positive' Class : Good
##
```

Apos a rapida analise das Tabelas de Confusao, nota-se a melhora dos indicadores de qualidade dos modelos.

Etapa 8 - Curva ROC e avaliacao final do modelo

```
## Comparacao dos 2 modelos

## Plotagem das ROC Curves

# Pacote necesario
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 3.5.1
```

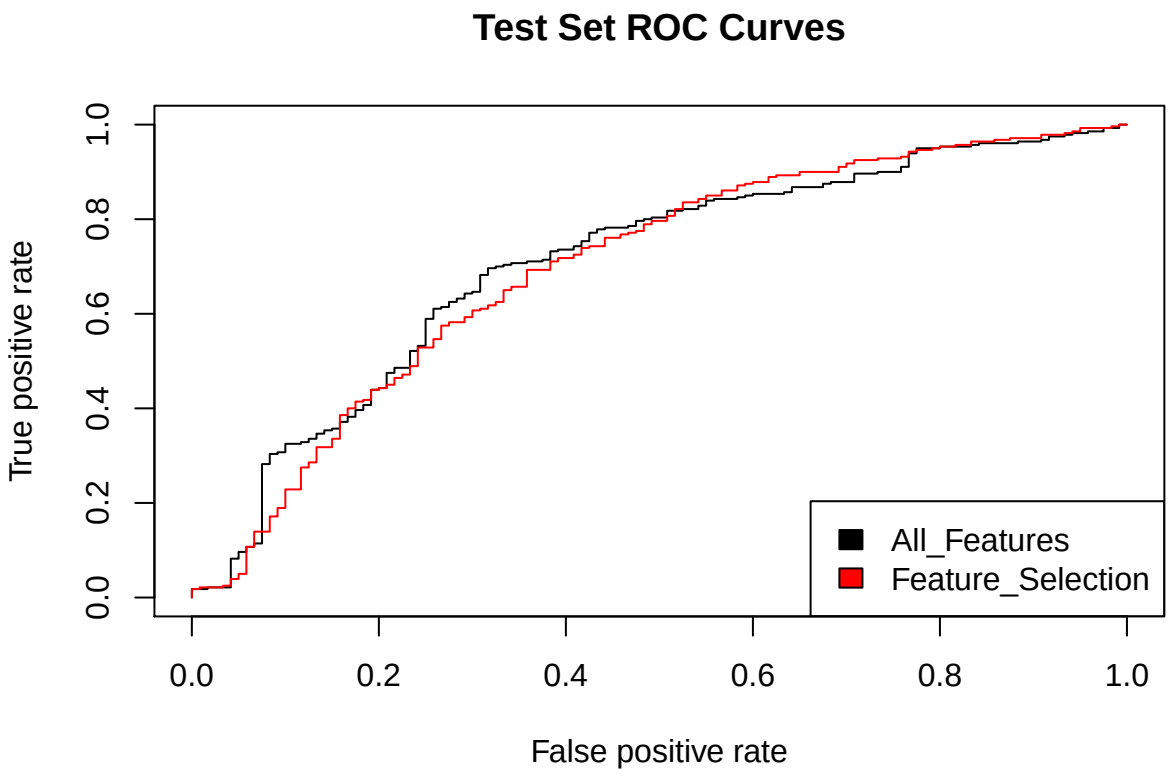
```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

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

```
# Lista com as predicoes
full_pred <- predict(glmModel_full, newdata = test_sample, type = "prob")
```

```
fs_pred <- predict(glmModel_fs, newdata = test_sample, type = "prob")
pred_list <- list(full_pred$Good, fs_pred$Good)

# Lista dos valores de fato (mesmo para todos)
m <- length(pred_list)
test_list <- rep(list(test_sample$CreditStatus), m)

# ROC curves
pred <- prediction(pred_list, test_list)
rocs <- performance(pred, "tpr", "fpr")
plot(rocs, col = as.list(1:m), main = "Test Set ROC Curves")
legend(x = "bottomright",
       legend = c("All_Features", "Feature_Selection"),
       fill = 1:m)
```

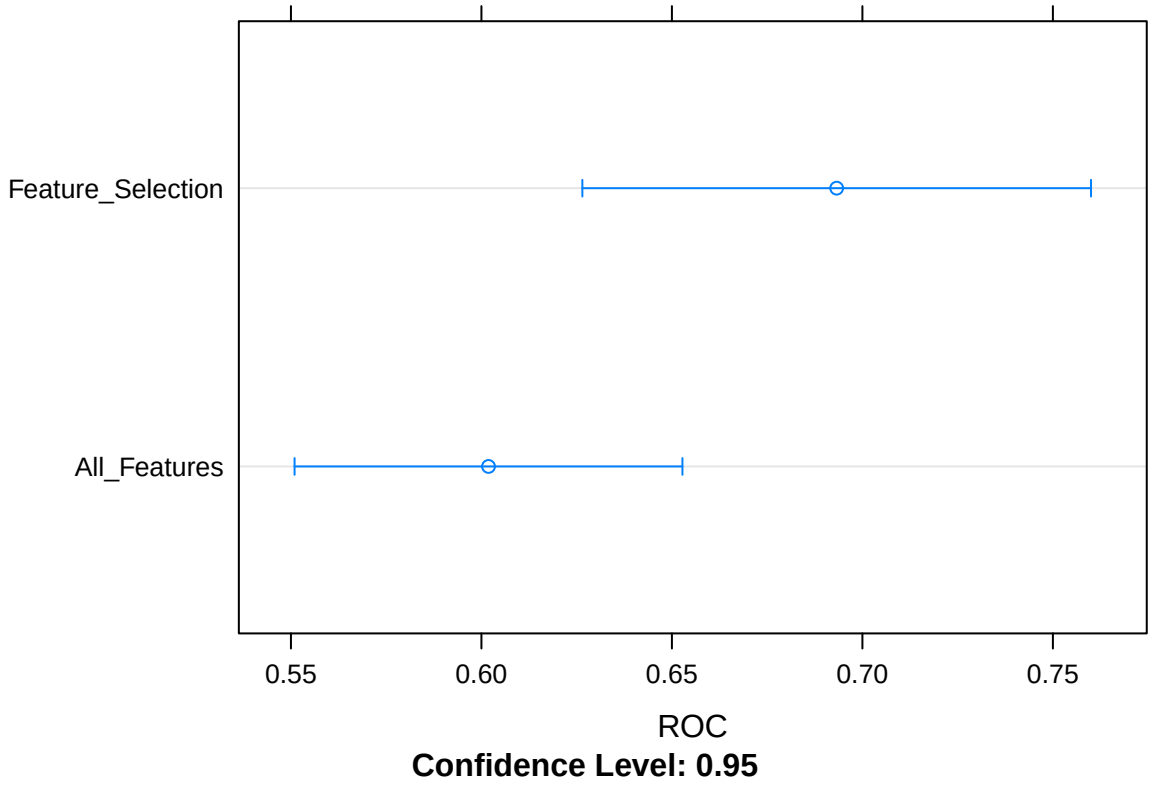



```
# Lista dos modelos
model_list <- list(All_Features = glmModel_full, Feature_Selection = glmModel_fs)

# Uso da funcao resamples
resamples <- resamples(model_list)
```

```
## Warning in resamples.default(model_list): 'All_Features' did not have
## 'returnResamp="final"; the optimal tuning parameters are used
```

```
## Warning in resamples.default(model_list): 'Feature_Selection' did not have
## 'returnResamp="final"; the optimal tuning parameters are used
```

```
# Dotplot
dotplot(resamples, metric  = "ROC")
```



Por fim ontem-se as representacoes graficas das ROC Curves de cada um dos modelos. Ao se analisar sua forma sobreposta, fica dificil de se extrair qualquer conclusao clara, no entanto, com o auxilios da funcao 'resamples', torna-se possivel fazer uma comparacao mais concisa a partir de cada uma das validacoes cruzadas usadas. Assim se consegue ter uma ideia mais clara de como somente uma selecao mais apurada de variaveis pode alterar a performance do modelo.

Fim

www.datascienceacademy.com.br