```
#
# Atividade de Aprofundamento
# Tarefa 3 = Logit Japan Credit Bank
#
# Empregue a base
# http://archive.ics.uci.edu/ml/datasets/credit+approval
#
# Antes de iniciar explore a documentação da base de dados no site.
#
# Altere o código abaixo para avaliar a acuracidade do modelo logístico
# empregando 20 partições sobre os dados (com 5 repetições).
#
# Em seguida responda a questões de 5 a 10 do questionário e POSTE SEU CÓDIGO
#
# As questões 5 a 7 correspondem a questões sobre preparação dos dados
# As questões 8 a 10 do modelo logistico e ROC
#
# dica: empregue o código da tarefa 2 como exemplo
#
install.packages("ROCR")
install.packages("dummies")
install.packages("caret")
install.packages("e1071")
library(ROCR)
library(dummies)
library(caret) # for Cross Validation functions
library(e1071)

# Leia os dados de
# https://archive.ics.uci.edu/ml/machine-learning-databases/credit-screening/crx.data
#
```

```
# dica: os dados não possuem cabeçalho


credit = read.csv('https://archive.ics.uci.edu/ml/machine-learning-databases/credit-screening/crx.data', header = FALSE)

cat("credit - rows and columns dataset:", nrow(credit) ," rows ", ncol(credit), "columns ", "\n")


# Elimine as linha com valores ausentes "?"

#

# dica: primeiro troque os valores "?" por NA e em seguida use na.omit()


credit[credit[]=="?"] <- NA

credit = na.omit(credit)


# Valores numéricos com NA aparecem como caracteres. Converta esses valores

# para numérico com as.numeric (V2, V14)

#


str(credit) # as colunas V2 e V14 estão como caracteres e não como numéricos

credit$V2 = as.numeric(credit$V2)

credit$V14 = as.numeric(credit$V14)


# Converta o atributo de classe para valores 0 e 1

#

credit$V16 = ifelse(credit$V16 == '+', 1, 0)

credit$V16 = as.numeric(credit$V16)


cat("credit - rows and columns after NA omit:", nrow(credit) ," rows ", ncol(credit), "columns ", "\n")


# Aplique o dummy encode para todos os atributos categóricos

#

# dica: empregue o comando dummy da library(dummies)
```

```
#

credit = cbind(credit, dummy('V1', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))

credit = cbind(credit, dummy('V4', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))

credit = cbind(credit, dummy('V5', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))

credit = cbind(credit, dummy('V6', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))

credit = cbind(credit, dummy('V7', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))

credit = cbind(credit, dummy('V9', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))

credit = cbind(credit, dummy('V10', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))

credit = cbind(credit, dummy('V12', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))

credit = cbind(credit, dummy('V13', data = credit, sep = ".", drop = TRUE, fun = as.integer,
verbose = FALSE))


cat("credit - rows and columns after dummy encode:", nrow(credit) ," rows ", ncol(credit),
"columns ", "\n")

head(credit)


#

# Logit com Cross Validation

# dica: empregue o código da tarefa 2 como exemplo

#


# Crie o arquivo de controle para 20 partições dos dados e 5 repetições : K-Fold

ctrl <- trainControl(method="repeatedcv", number= 20, repeats=5)


# Faça o treinamento logístico, nao esquecer de empregar preProcess = c("center", "scale")

fit <- train(V16~., data=credit,

        method="glm",
```

```
        family="binomial",
        trControl=ctrl,
        preProcess = c("center", "scale"))
fit

# Faça a predição para todos os valores de credit
predict_test = predict(fit, newdata=credit, type="raw")

# Converte predict_test para valores 0 e 1
predict_test = ifelse(predict_test > 0.5, 1,0)

# Construa a matriz de confusão
c_matrix = table(credit$V16, predict_test)

# Calcula a acuracidade
acc = sum(diag(c_matrix))/sum(c_matrix)*100
cat('Accuracy: ', acc, ' %', "\n")

# Plot da curva ROC
pr=prediction(as.numeric(predict_test),credit$V16)
prf=performance(pr, measure="tpr", x.measure="fpr")
plot(prf,colorize=TRUE)

# Calcule a área sob a curva ROC
auc=performance(pr, measure="auc")
auc=auc@y.values[[1]]
auc
```