Prever sobbreviventes no desastre do Titanic

luiz felipe

8/13/2020

Etapa 1 - Carregando conjunto de dados de treino e teste do titanic

Os dados foram retirados da plataforma kaggle do famoso desafio do titanic

```
# Carregando bibliotecas
require(dplyr)
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

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

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

```
require(ggplot2)
```

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

```
require(stringr)
```

```
## Loading required package: stringr
```

```
require(tidyr)
```

```
## Loading required package: tidyr
```

```
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
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
# Carregando conjunto de dados
titanic <- read.csv("train.csv", header = TRUE, stringsAsFactors = TRUE)
dados_teste_final <- read.csv("test.csv", header = TRUE, stringsAsFactors = TRUE)
```

Etapa 2 - Explorando dados

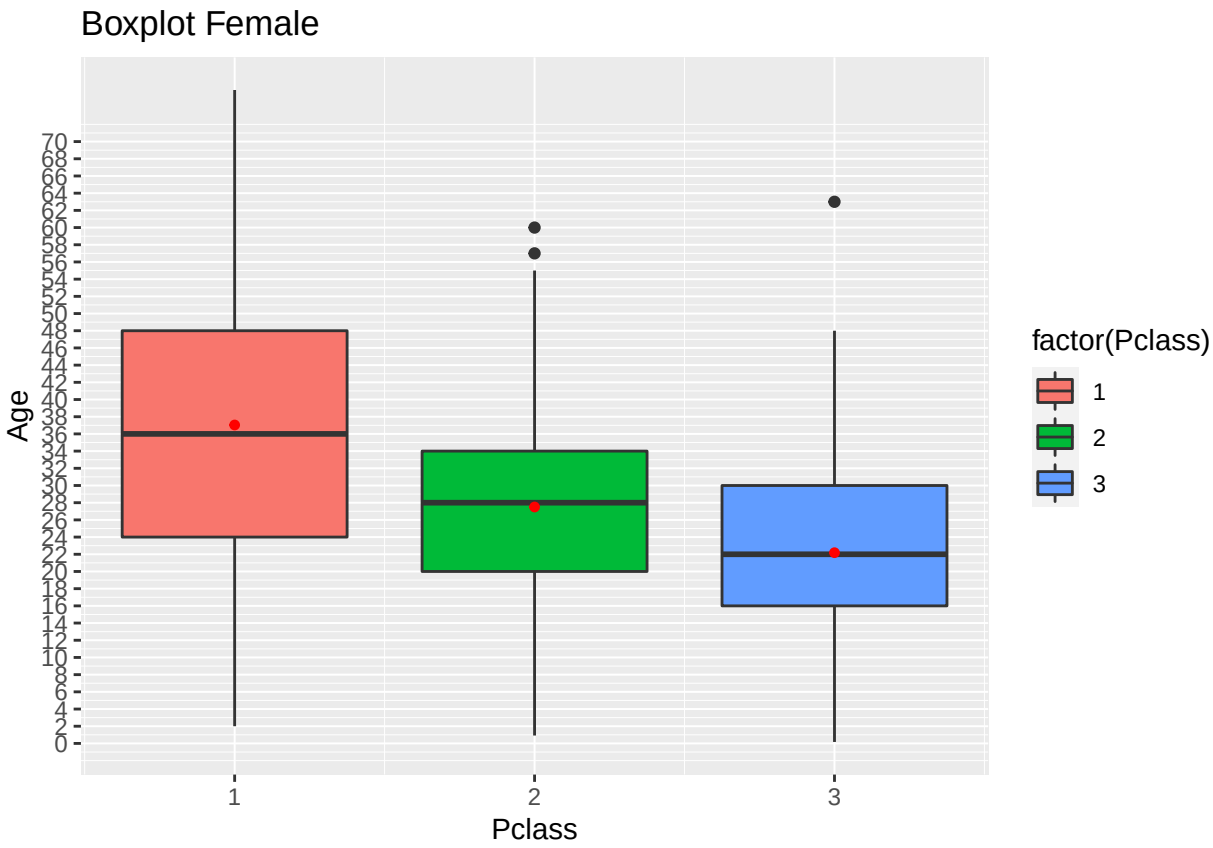
Para essa etapa, vamos criar um novo data set sem a variável a ser explicada

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : Factor w/ 891 levels "Abbing, Mr. Anthony",..: 109 191 358 277 16 559 520 629 417 58
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : Factor w/ 681 levels "110152","110413",..: 524 597 670 50 473 276 86 396 345 133 ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : Factor w/ 148 levels "","A10","A14",..: 1 83 1 57 1 1 131 1 1 1 ...
##  $ Embarked   : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0         177
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0           0           0
```

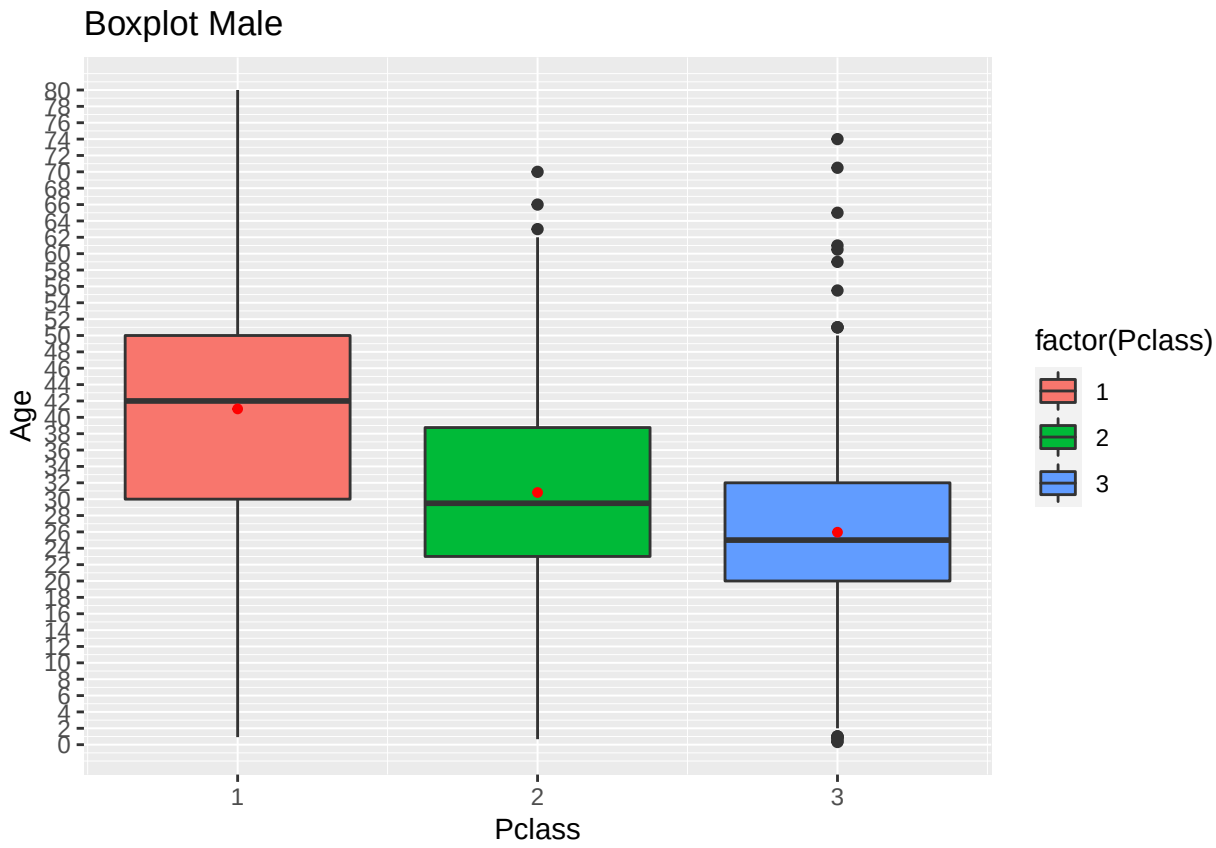
```
## Warning: Removed 78 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 78 rows containing non-finite values (stat_summary).
```



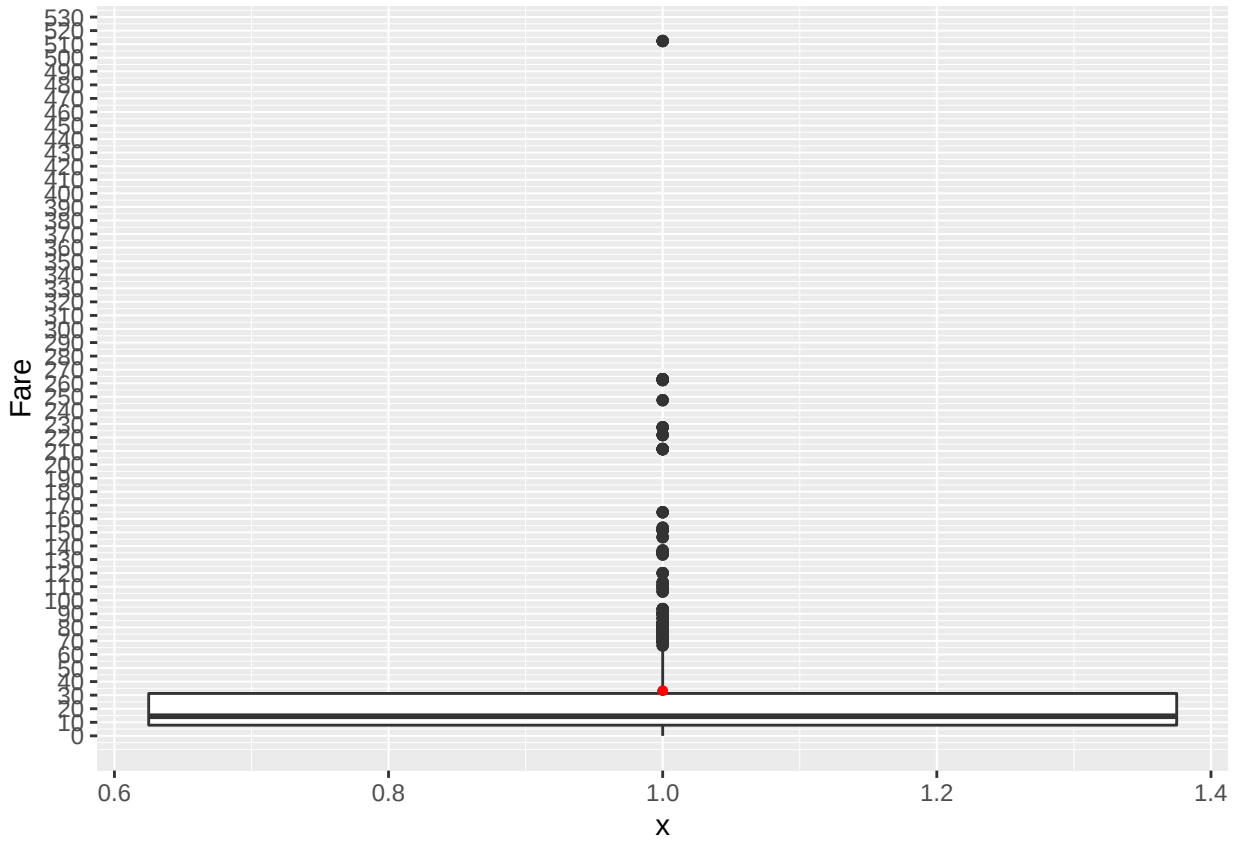

```
## Warning: Removed 185 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 185 rows containing non-finite values (stat_summary).
```




```
## Warning: Removed 1 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 1 rows containing non-finite values (stat_summary).
```



Etapa 3 - Engenharia de dados

Utilizando técnicas para modificar os dados

```
# Imputando valores para as idades (escolhemos o valor da mediana)
imputate.ages <- function(age, class, sex) {
  age.vec <- c()
  for(i in 1:length(age)) {
    if(is.na(age[i])) {
      if(class == 1 && sex == "female") {
        age.vec[i] = 36
      }
      else if(class == 2 && sex == "female") {
        age.vec[i] = 28
      }
      else if(class == 3 && sex == "female") {
        age.vec[i] = 22
      }
      else if(class == 1 && sex == "male") {
        age.vec[i] = 42
      }
      else if(class == 2 && sex == "male") {
        age.vec[i] = 29.5
      }
      else {
        age.vec[i] = 25
      }
```

```
    }
    else {
      age.vec[i] = age[i]
    }
  }
  return(age.vec)
}

ages <- imputate.ages(titanic$Age, titanic$Pclass, titanic$Sex)
titanic$Age <- ages

# Substituindo todos os pontos por caracter vazio na coluna Ticket
titanic$Ticket <- str_replace_all(titanic$Ticket, "\\.", "")

# Retirando os números e espaços da coluna Cabin
titanic$Cabin <- gsub('[0-9]+', '', titanic$Cabin)

# Substituindo o caracter espaço pelo caracter vazio da coluna Ticket no padrão
# TON/O 2
for(i in 1:length(titanic$Ticket)) {
  if(str_detect(titanic$Ticket[i], "(TON/O 2)")) {
    titanic$Ticket[i] <- str_replace(titanic$Ticket[i],
                                     "(?<=TON/O)\\s", '')
  }
  else {
    titanic$Ticket[i] <- titanic$Ticket[i]
  }
}

# Limpando alguns dados
for(i in 1:length(titanic$Ticket)) {
  if(str_detect(titanic$Ticket[i], "(STON/O)")) {
    titanic$Ticket[i] <- str_replace(titanic$Ticket[i],
                                     "(?<=)STON", "SOTON")
  }
  else if(str_detect(titanic$Ticket[i], "(SC/Pa)")) {
    titanic$Ticket[i] <- str_replace(titanic$Ticket[i],"(?<=)Paris","PARIS")
  }
  else if(str_detect(titanic$Ticket[i], "(Basle)")) {
    titanic$Ticket[i] <- str_replace(titanic$Ticket[i],"(?<=H)\\sBasle","")
  }
  else if(str_detect(titanic$Ticket[i], "(WEP)")) {
    titanic$Ticket[i] <- str_replace(titanic$Ticket[i],"(WEP)","WE/P")
  }
  else if(str_detect(titanic$Ticket[i], "(Fa)")) {
    titanic$Ticket[i] <- str_replace(titanic$Ticket[i],"(Fa)","")
  }
  else {
    titanic$Ticket[i] <- titanic$Ticket[i]
  }
}

# Colocando espaço nos Tickets que começam com número
```

```
for(i in 1:length(titanic$Ticket)) {
  if(str_detect(titanic$Ticket[i],"^[:digit:]")) {
    titanic$Ticket[i] <- str_replace(titanic$Ticket[i],"(?=[:digit:])", " ")
  }
  else {
    titanic$Ticket[i] <- titanic$Ticket[i]
  }
}

for(i in 1:length(titanic$Ticket)) {
  if(str_detect(titanic$Ticket[i],"^(A\\s)")) {
    titanic$Ticket[i] <- str_replace(titanic$Ticket[i],"(?<=A)\\s", "\\/")
  }
  else {
    titanic$Ticket[i] <- titanic$Ticket[i]
  }
}

# Separando a coluna Ticket em duas colunas
titanic <- separate(titanic, Ticket, c("TicketPrefix", "TicketNumber"), sep = "\\s")
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 4 rows [180, 272,
## 303, 598].
```

```
# Substituindo valor NA por caracter vazio
titanic$TicketNumber <- sapply(titanic$TicketNumber, function(x) ifelse(is.na(x), '', x))

# Deixando apenas uma letra
remove.letters <- function(vec) {
  lett <- c()
  for(i in 1:length(vec)) {
    if(nchar(vec[i]) > 1) {
      if(str_detect(vec[i],"\\s")) {
        lett[i] <- min(strsplit(vec[i]," ")[[1]])
      }
      else {
        lett[i] <- min(strsplit(vec[i],"")[[1]])
      }
    }
    else {
      lett[i] <- vec[i]
    }
  }
  return(lett)
}

titanic$Cabin <- remove.letters(titanic$Cabin)

# Eliminando letras maiores que G
remove.lgtT <- function(x) {
  let <- c()
  for(i in 1:length(x)) {
    if(x[i] > "G") {
      let[i] <- ""
```

```
    }
    else {
      let[i] <- x[i]
    }
  }
  return(let)
}

titanic$Cabin <- remove.lgtT(titanic$Cabin)

# Continuando a limpar os dados
for(i in 1:length(titanic$TicketPrefix)) {
  if(str_detect(titanic$TicketPrefix[i], "(SOC)")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "(?<=)SOC", "SO/C")
  }
  else if(str_detect(titanic$TicketPrefix[i], "^(A4)")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], titanic$TicketPrefix[i], "A/4")
  }
  else if(str_detect(titanic$TicketPrefix[i], "^(A5)")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "(?<=)A5", "A/5")
  }
  else if(str_detect(titanic$TicketPrefix[i], "(CA/SOTON)")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "(?<=)CA/SOTON", "CA")
  }
  else if(str_detect(titanic$TicketPrefix[i], "A/S")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "A/S", "A/5")
  }
  else if(str_detect(titanic$TicketPrefix[i], "FC")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], titanic$TicketPrefix[i], "FCC")
  }
  else if(str_detect(titanic$TicketPrefix[i], "(P/)")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "(?<=)P/PP", "PP")
  }
  else if(str_detect(titanic$TicketPrefix[i], "(SC)")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], titanic$TicketPrefix[i], "SC")
  }
  else if(str_detect(titanic$TicketPrefix[i], "SP")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "(?<=)SP", "SO")
  }
  else if(str_detect(titanic$TicketPrefix[i], "SO/PP")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "(?<=)SO/PP", "SO")
  }
  else if(str_detect(titanic$TicketPrefix[i], "SOP")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "(?<=)SOP", "SO")
  }
  else if(str_detect(titanic$TicketPrefix[i], "SW/PP")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], "(?<=)SW/PP", "PP")
  }
  else {
    titanic$TicketPrefix[i] <- titanic$TicketPrefix[i]
  }
}
```

```
for(i in 1:length(titanic$TicketPrefix)) {
  if(str_detect(titanic$TicketPrefix[i], "^A")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], titanic$TicketPrefix[i], "A")
  }
  else {
    titanic$TicketPrefix[i] <- titanic$TicketPrefix[i]
  }
}

for(i in 1:length(titanic$TicketPrefix)) {
  if(str_detect(titanic$TicketPrefix[i], "^LP")) {
    titanic$TicketPrefix[i] <- str_replace(titanic$TicketPrefix[i], titanic$TicketPrefix[i], "")
  }
  else {
    titanic$TicketPrefix[i] <- titanic$TicketPrefix[i]
  }
}

table(factor(titanic$TicketPrefix))
```

```
##
##                 A        C       CA      FCC     LINE       PC       PP
##       959       42        8       69       12        4       92        8
##        SC       SO     SO/C SOTON/O2 SOTON/OQ      W/C     WE/P
##        30        9        8       24       25       15        4
```

```
# Transformando TicketPrefix em número
titanic$TicketNumber <- as.numeric(titanic$TicketNumber)

# Subistitui valores NA por 0 na coluna Ticket Number
titanic$TicketNumber <- sapply(titanic$TicketNumber, function(x) ifelse(is.na(x), 0, x))

# Valores NA
sapply(titanic, function(x) sum(is.na(x)))
```

```
##  PassengerId       Pclass         Name          Sex          Age        SibSp
##            0            0            0            0            0            0
##        Parch TicketPrefix TicketNumber         Fare        Cabin     Embarked
##            0            0            0            1            0            0
```

```
# Substituindo o preço pela mediana
for(i in 1:nrow(titanic)) {
  if(is.na(titanic$Fare[i])) {
    titanic$Fare[i] <- 15
  }
  else {
    titanic$Fare[i] <- titanic$Fare[i]
  }
}
```

Criando modelo

Fazendo modificações no data set para criar os modelos

```
# Carregando dados
# Data set treino
```

```
titan1 <- read.csv("train.csv", header = TRUE, stringsAsFactors = FALSE)
titan2 <- cbind(Survived = titan1$Survived, titanic[1:891,])
names(titan2)[1] <- "Survived"
titanicTrain <- titan2
titanicTrain$TicketPrefix <- factor(titanicTrain$TicketPrefix)
titanicTrain$Cabin <- factor(titanicTrain$Cabin)
titanicTrain$Survived <- factor(titanicTrain$Survived)
titanicTrain$Pclass <- factor(titanicTrain$Pclass)

# Data set teste
titanicTest <- titanic[892:1309,]
titanicTest$TicketPrefix <- factor(titanicTest$TicketPrefix)
titanicTest$Cabin <- factor(titanicTest$Cabin)
titanicTest$Pclass <- factor(titanicTest$Pclass)

# Retirando algumas colunas que podem causar overfitting
# colunas a serem tiradas: Passengerid
# Dado um vetor do nome das colunas e um vetor de pattern retorna
# o índice desse padrão na lista
indice <- function(pattern, vectorPattern) {
  index <- c()
  for(i in 1:length(pattern)) {
    index[i] <- grep(pattern[i], vectorPattern,value = FALSE)
  }
  return(index)
}

# retirando a algumas colunas para criar o modelo
indices <- indice(c("Name", "PassengerId"),
                  names(titanicTrain))

titanic3 <- titanicTrain[,-indices]

sapply(titanic3, function(x) sum(is.na(x)))
```

```
##     Survived       Pclass          Sex          Age        SibSp        Parch
##            0            0            0            0            0            0
## TicketPrefix TicketNumber         Fare        Cabin     Embarked
##            0            0            0            0            0
```

```
# Criando vários modelos - a melhor performance é seed 92 e 0.9162011
RandomForestTitanic <- function(fraction_train, n) {
  Performance_models <- c()
  seeds <- c()
  seed <- 1
  for(i in 1:n) {
    seeds[i] <- seed
    set.seed(seed)
    rows <- sample(1:nrow(titanic3), fraction_train*nrow(titanic3), replace = FALSE)
    dados_treino <- titanic3[rows,]
    dados_teste <- titanic3[-rows,]
    while(length(levels(factor(dados_treino$Embarked))) != 4 | length(levels(factor(dados_treino$Cabin))
      seed <- seed + 1
      seeds[i] <- seed
```

```
      set.seed(seed)
      rows <- sample(1:nrow(titanic3), fraction_train*nrow(titanic3), replace = FALSE)
      dados_treino <- titanic3[rows,]
      dados_teste <- titanic3[-rows,]
    }
    dados_treino$Survived = factor(dados_treino$Survived)
    dados_teste$Survived = factor(dados_teste$Survived)
    model.rf <- randomForest(x = dados_treino[,-1],
                             importance = TRUE, proximity = TRUE,
                             y = dados_treino$Survived)
    model.pred <- predict(model.rf, dados_teste[,-1], type = "class")
    Performance_models[i] <- mean(model.pred == dados_teste[,1])
    seed <- seed + 1
  }
  return(PerformanceSeed <- list(Performance_models, seeds))
}

RandomForestTitanic(0.8,100)
```

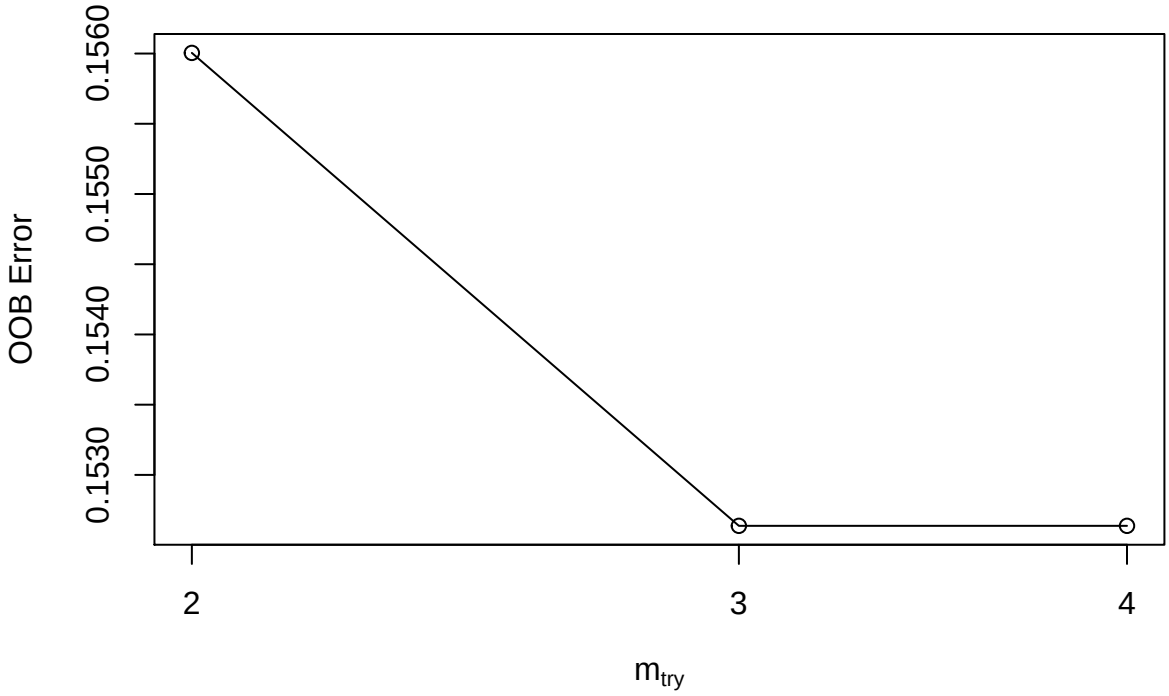
```
## [[1]]
##   [1] 0.8659218 0.7988827 0.8826816 0.8379888 0.8379888 0.8603352 0.8044693
##   [8] 0.8324022 0.8826816 0.8268156 0.8715084 0.8547486 0.8715084 0.8268156
##  [15] 0.8603352 0.8379888 0.8659218 0.8715084 0.8715084 0.7988827 0.8603352
##  [22] 0.8603352 0.8156425 0.8268156 0.8435754 0.8770950 0.8826816 0.8435754
##  [29] 0.8659218 0.8659218 0.8435754 0.8435754 0.8826816 0.8100559 0.8826816
##  [36] 0.8156425 0.8435754 0.8379888 0.8156425 0.8715084 0.8268156 0.8212291
##  [43] 0.8435754 0.8100559 0.8100559 0.8379888 0.8547486 0.8491620 0.8379888
##  [50] 0.8603352 0.8603352 0.8212291 0.8379888 0.8100559 0.8603352 0.8212291
##  [57] 0.8826816 0.8156425 0.8324022 0.8435754 0.8324022 0.8379888 0.8603352
##  [64] 0.8324022 0.8547486 0.8770950 0.8212291 0.8435754 0.8603352 0.8435754
##  [71] 0.8324022 0.8324022 0.7932961 0.8603352 0.8268156 0.8435754 0.8715084
##  [78] 0.8268156 0.8491620 0.8212291 0.8770950 0.9050279 0.8826816 0.8379888
##  [85] 0.8659218 0.8603352 0.8324022 0.8826816 0.8826816 0.8491620 0.8659218
##  [92] 0.8268156 0.8770950 0.8044693 0.8603352 0.8770950 0.8268156 0.8379888
##  [99] 0.8268156 0.8659218
##
## [[2]]
##   [1]   1   2   3   4   5   6   7  10  11  12  13  14  15  16  17  18  19  20
##  [19]  21  22  23  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39
##  [37]  41  42  43  44  45  47  48  49  50  51  52  53  55  56  57  58  59  60
##  [55]  61  62  63  64  65  66  67  68  69  70  71  72  73  75  76  77  78  79
##  [73]  81  82  83  84  86  87  88  89  90  92  94  95  96  97  99 100 101 102
##  [91] 103 104 105 106 107 108 109 110 111 112
```

Alterando o valor do mtry

```
# Testando o melhor valor para mtry
x <- titanic3[,2:11]
y <- titanic3[,1]
seed <- 1
set.seed(seed)
bestmtry <- tuneRF(x, y, stepFactor = 1.5, improve = 1e-5, ntreeTry = 500)
```

```
## mtry = 3  OOB error = 15.26%
```

```
## Searching left ...
## mtry = 2     OOB error = 15.6%
## -0.02205882 1e-05
## Searching right ...
## mtry = 4     OOB error = 15.26%
## 0 1e-05
```




```
print(bestmtry)
```

```
##       mtry  OOBError
## 2.OOB    2 0.1560045
## 3.OOB    3 0.1526375
## 4.OOB    4 0.1526375
```

Novo modelo

```
# Modelo com o melhor mtry
RandomForestTitanicmtry <- function(fraction_train, n) {
  Performance_models <- c()
  seeds <- c()
  seed <- 1
  for(i in 1:n) {
    seeds[i] <- seed
    set.seed(seed)
    rows <- sample(1:nrow(titanic3), fraction_train*nrow(titanic3), replace = FALSE)
    dados_treino <- titanic3[rows,]
    dados_teste <- titanic3[-rows,]
    while(length(levels(factor(dados_treino$Embarked))) != 4 | length(levels(factor(dados_treino$Cabin))
      seed <- seed + 1
      seeds[i] <- seed
      set.seed(seed)
      rows <- sample(1:nrow(titanic3), fraction_train*nrow(titanic3), replace = FALSE)
```

```
      dados_treino <- titanic3[rows,]
      dados_teste <- titanic3[-rows,]
    }
    dados_treino$Survived = factor(dados_treino$Survived)
    dados_teste$Survived = factor(dados_teste$Survived)
    model.rf <- randomForest(x = dados_treino[,-1],
                             importance = TRUE, proximity = TRUE,
                             y = dados_treino$Survived, mtry = 2)
    model.pred <- predict(model.rf, dados_teste[,-1], type = "class")
    Performance_models[i] <- mean(model.pred == dados_teste[,1])
    seed <- seed + 1
  }
  return(PerformanceSeed <- list(Performance_models, seeds))
}

RandomForestTitanicmtry(0.8,100)
```

```
## [[1]]
##   [1] 0.8659218 0.7932961 0.8715084 0.8491620 0.8324022 0.8659218 0.8100559
##   [8] 0.8379888 0.8603352 0.8156425 0.8826816 0.8435754 0.8491620 0.8156425
##  [15] 0.8715084 0.8379888 0.8715084 0.8603352 0.8659218 0.7877095 0.8435754
##  [22] 0.8491620 0.8212291 0.7988827 0.8324022 0.8603352 0.8770950 0.8268156
##  [29] 0.8603352 0.8379888 0.8435754 0.8268156 0.8547486 0.8156425 0.8435754
##  [36] 0.8100559 0.8547486 0.8379888 0.8268156 0.8603352 0.8435754 0.8212291
##  [43] 0.8435754 0.7932961 0.8100559 0.8379888 0.8435754 0.8435754 0.8100559
##  [50] 0.8547486 0.8715084 0.8212291 0.8435754 0.8044693 0.8603352 0.8379888
##  [57] 0.8659218 0.8212291 0.8324022 0.8268156 0.8379888 0.8491620 0.8491620
##  [64] 0.8212291 0.8603352 0.8547486 0.8156425 0.8547486 0.8547486 0.8491620
##  [71] 0.8379888 0.8268156 0.7932961 0.8491620 0.8324022 0.8491620 0.8659218
##  [78] 0.8379888 0.8379888 0.8156425 0.8770950 0.9106145 0.8770950 0.8435754
##  [85] 0.8826816 0.8715084 0.8212291 0.8770950 0.8826816 0.8603352 0.8715084
##  [92] 0.8379888 0.8547486 0.8044693 0.8435754 0.8603352 0.8379888 0.8379888
##  [99] 0.8100559 0.8659218
##
## [[2]]
##   [1]   1   2   3   4   5   6   7  10  11  12  13  14  15  16  17  18  19  20
##  [19]  21  22  23  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39
##  [37]  41  42  43  44  45  47  48  49  50  51  52  53  55  56  57  58  59  60
##  [55]  61  62  63  64  65  66  67  68  69  70  71  72  73  75  76  77  78  79
##  [73]  81  82  83  84  86  87  88  89  90  92  94  95  96  97  99 100 101 102
##  [91] 103 104 105 106 107 108 109 110 111 112
```

Aplicando os modelos aos dados de teste

```
# Aplicando os mesmos levels tanto para dados de treino e teste
levels(titanicTest$TicketPrefix) <- levels(titanicTrain$TicketPrefix)
levels(titanicTest$Cabin) <- levels(titanicTrain$Cabin)
levels(titanicTest$Embarked) <- levels(titanicTrain$Embarked)

# Dados de treino e teste
indicesx <- indice(c("Name", "PassengerId"),
                   names(titanicTest))
```

```r
titanTrain <- titanicTrain[,-indices]
titanTest <- titanicTest[,-indicesx]

# Criando modelo
model.rf <- randomForest(x = titanTrain[,-1],
                         importance = TRUE, proximity = TRUE,
                         y = titanTrain$Survived)

# Predição
model.pred <- predict(model.rf, titanTest, type = "class")

Survived <- model.pred
PassengerId <- titanicTest$PassengerId

Resultado <- as.data.frame(cbind(PassengerId, Survived))
Resultado$Survived <- ifelse(Resultado$Survived == 1, 0, 1)
```