ANÁLISE DOS DADOS COM R

Ferramentas de Machine Learning

James R. Hunter, PhD
Retrovirologia, EPM, UNIFESP

2023-10-17

TÉCNICAS DE CLASSIFICAÇÃO E MAIS

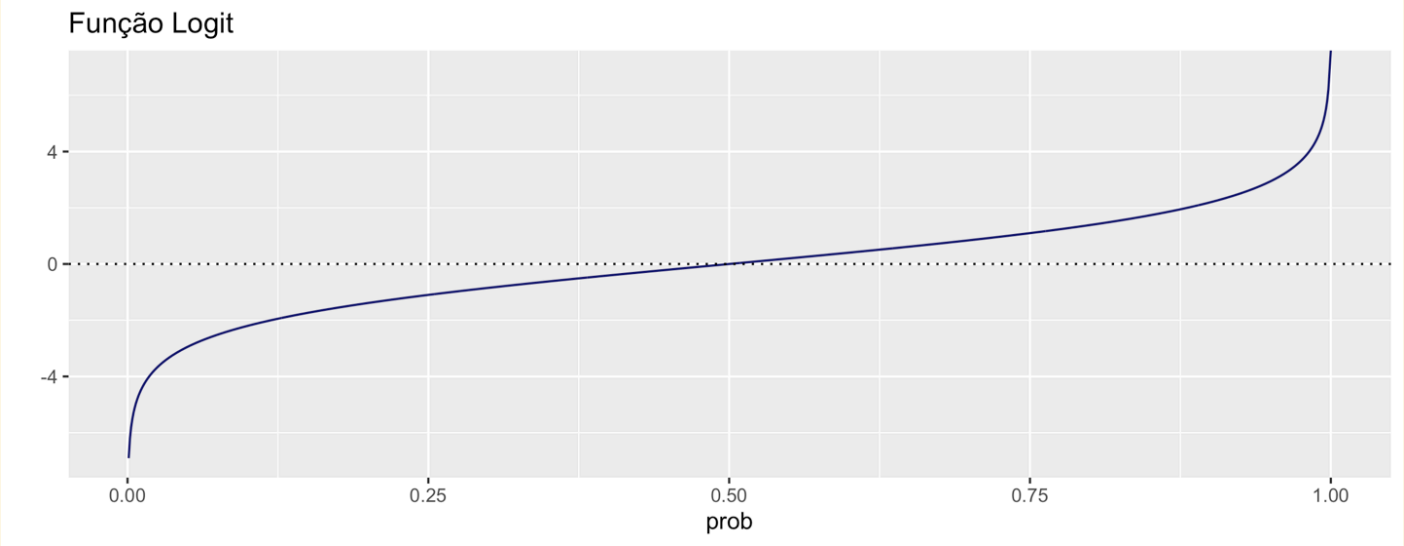
REGRESSÃO LOGÍSTICA

- Usado freqüentemente em bioestatística
- Extensão do conceito básico da regressão linear
- Variável dependente (Y) agora é **binomial**
  - Tem 2 estados:
    - TRUE; FALSE
    - 1 ; 0
    - “R5” ; “X4”
    - “infetado” ; “não infetado”
- As variáveis independentes podem ser numéricas ou categóricas

FUNÇÃO `logit`

- *log-odds*
- *odds* de um evento = $p/(1-p)$
  - probabilidade do evento dividido pela probabilidade que não acontecerá
- **logit** é o logaritmo natural dos odds

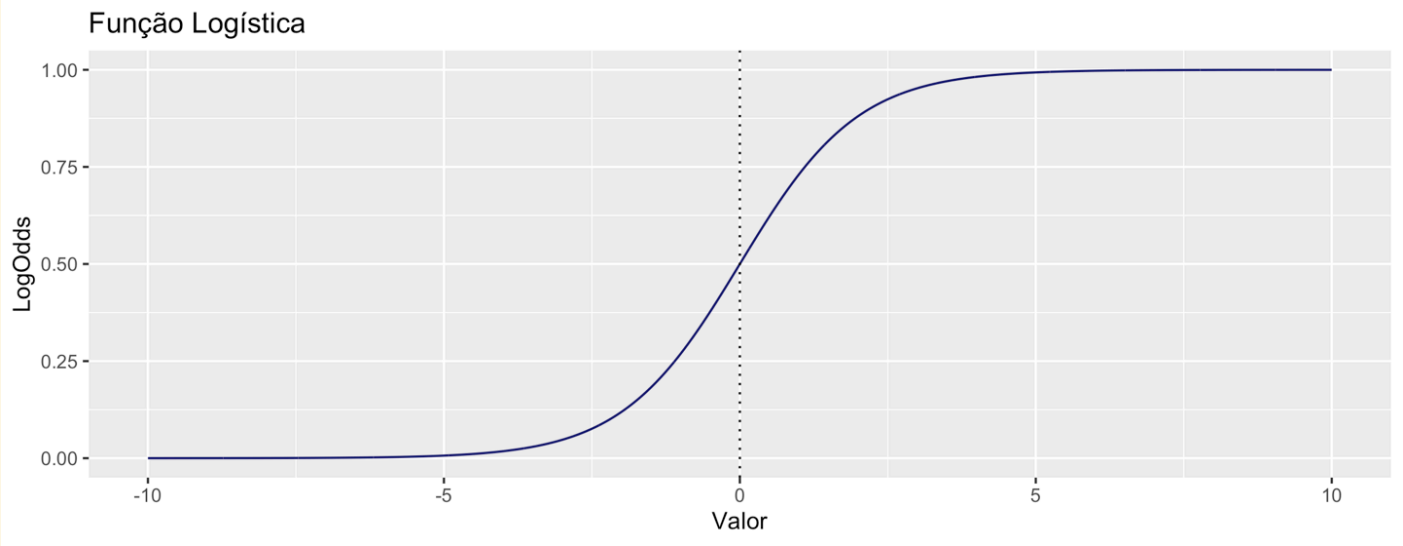
$$logit(p) = \frac{p}{1-p}$$


Função Logit
4
0
-4
0.00
0.25
0.50
0.75
1.00
prob

FUNÇÃO LOGÍSTICA

- Aplicamos função para as variáveis independentes (X)
  - Resultado: Variável dependente fica no intervalo entre 0 e 1
    - intervalo de probabilidades
- Função **Logística**
  - Inverso de **logit**
  - Aplicável a qualquer número

$$logit^{-1}(x) = \frac{1}{1 + e^{-x}}$$


Função Logística
1.00
0.75
0.50
0.25
0.00
LogOdds
-10
-5
0
5
10
Valor

COMPARAR RSL COM REGRESSÃO LOGÍSTICA

- Regressão Linear (usando notação de matrizes)

$$y = X\beta + \epsilon_i$$

- Regressão Logística

$$p(y_i = 1) = logit^{-1}(X_i\beta) + \epsilon_i$$

MODELOS LINEARES GERAIS (*GENERAL LINEAR MODELS*)

- Regressão logística faz parte de uma classe dos modelos: GLM
- Eles manipulam os matrizes diferente do modelo linear simples
  - Que é um caso especial dos GLM
- Outros modelos GLM: poisson (dados de contagem)
- Output seria semelhante com o output do regressão simples

EXEMPLO SIMPLES

- Estudo de 100 pacientes que têm ou não têm doença cardíaca coronária (CHD)
- Estudo interessado na relação entre a idade do paciente e a CHD
- Dados vêm de Hosmer & Lemeshow, *Applied Logistic Regression* (2a Ed.)
  - No arquivo `chdage.csv`

CARREGAR OS DADOS

```r
chdage <- read_csv(here::here("../chdage.csv")) %>%
  mutate(chd = factor(chd)) %>%
  mutate(chd = fct_recode(chd, negativo = "0", positivo = "1"))
glimpse(chdage)
```

```
Rows: 100
Columns: 3
$ id    <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
1…
$ idade <dbl> 20, 23, 24, 25, 25, 26, 26, 28, 28, 29, 30, 30, 30, 30, 30, 30,
…
$ chd   <fct> negativo, negativo, negativo, negativo, positivo, negativo,
nega…
```

ANALISE BÁSICA EXPLORATÓRIA

```
chdage %>%
  select(idade) %>%
  descr(transpose = TRUE,
        stats = c("mean", "sd", "min", "q1", "med", "q3",
                  "max", "iqr", "cv"))
```

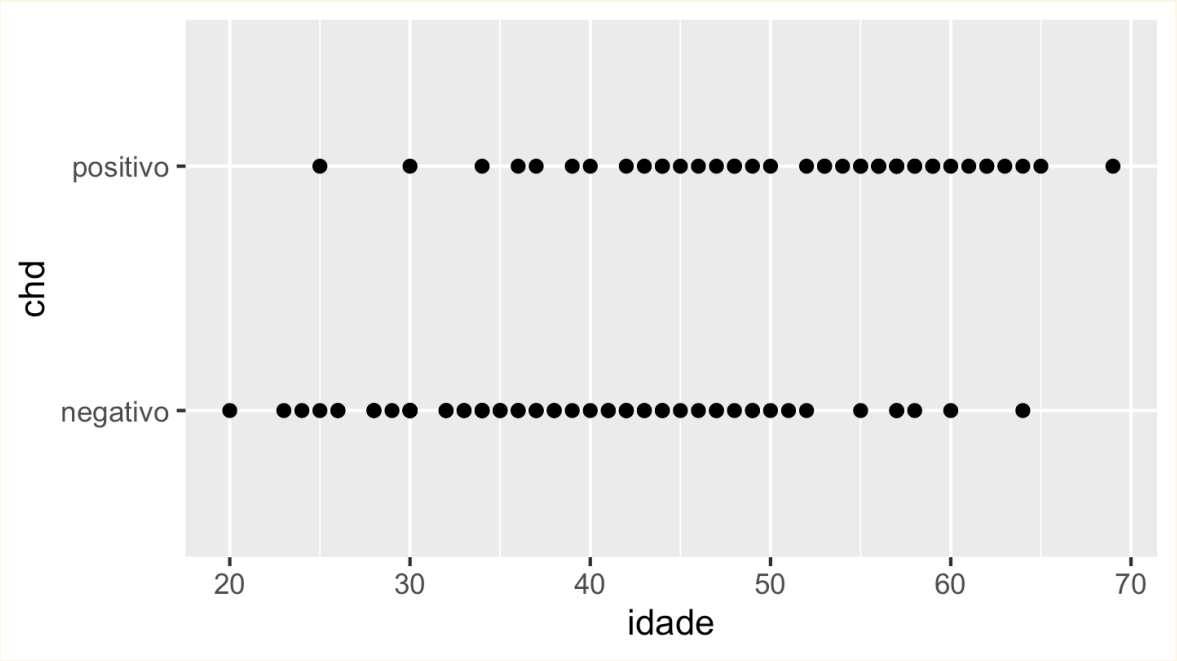
```
Descriptive Statistics
chdage$idade
N: 100

              Mean   Std.Dev      Min       Q1   Median       Q3      Max      IQR       CV
---------- ------- --------- ------- -------- -------- -------- -------- -------- -------
     idade   44.38     11.72   20.00    34.50    44.00    55.00    69.00    20.25    0.26
```

```
chdage %>%
  select(chd) %>%
  freq()
```

```
Frequencies
chdage$chd
Type: Factor

                Freq   % Valid   % Valid Cum.   % Total   % Total Cum.
-------------- ------ --------- -------------- --------- --------------
     negativo     57     57.00          57.00     57.00          57.00
     positivo     43     43.00         100.00     43.00         100.00
         <NA>      0                                0.00         100.00
        Total    100    100.00         100.00    100.00         100.00
```

SCATTERPLOT DE CHD E IDADE

```
1 chdscat <- ggplot(data = chdage, aes(y = chd, x = idade)) + geom_point()
2 chdscat
```



BOXPLOT DA IDADE

```
chdbox <- ggplot(data = chdage, aes(x = chd, y = idade, group = chd))
chdbox <- chdbox + geom_boxplot()
chdbox
```

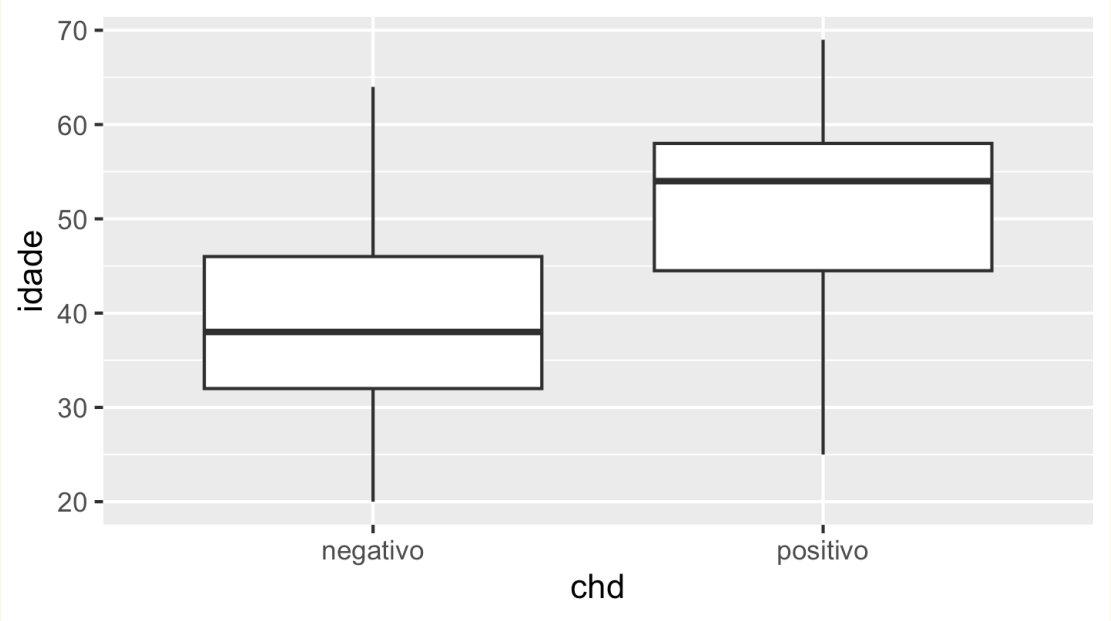
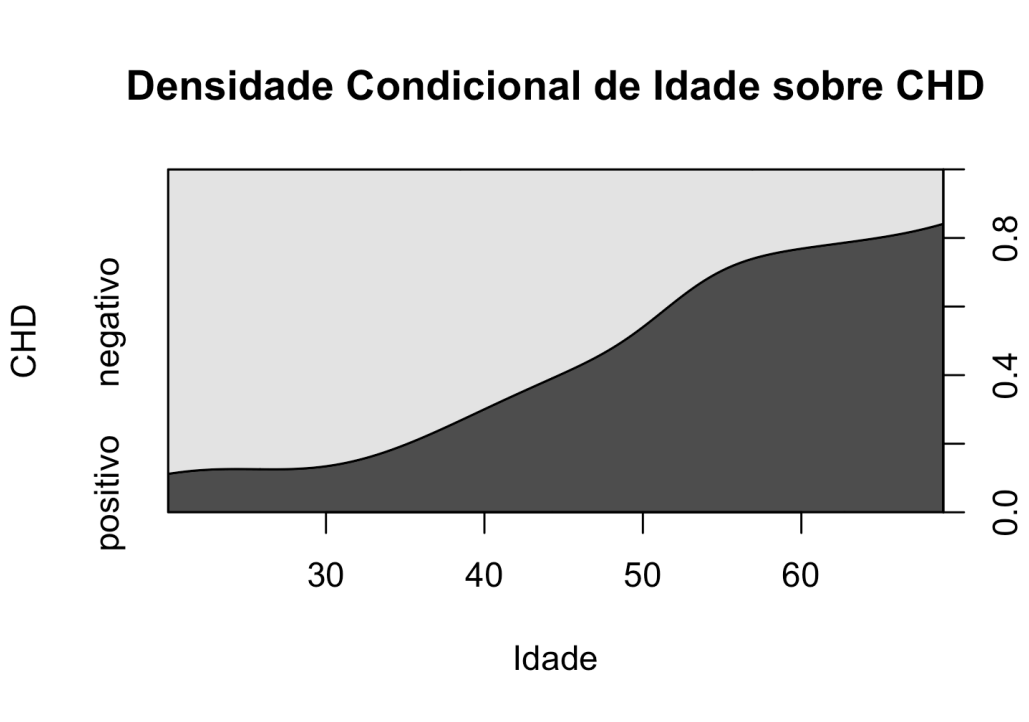


GRÁFICO DE DENSIDADE CONDICIONAL

- Também útil para entender como idade muda nas 2 categorias de CHD
- Mostra o número daqueles com a doença (chd = 1) para todos as idades
  - Numa forma continua

```r
cdplot(factor(chd) ~ idade, data = chdage,
       main = "Densidade Condicional de Idade sobre CHD",
       xlab = "Idade", ylab = "CHD")
```



MODELO

- Como o pacote `lm`, `glm` usa o formato de formula para especificar o modelo
  - variável dependente `~` variáveis independentes
  - variáveis independentes separados com `+`
- Fonte dos dados (`data =`)
- `Family` dos modelos (neste caso, `binomial`)
- Função `link` (neste caso, `logit`)

```
chdfit1 <- glm(chd ~  idade, data = chdage,
               family = binomial(link = "logit"))
```

RESULTADOS

- Obter os resultados como no `lm`, com `summary`
- Também podemos olhar nos coeficientes com um gráfico chamada `coefplot`
- Vem de pacote de mesmo nome

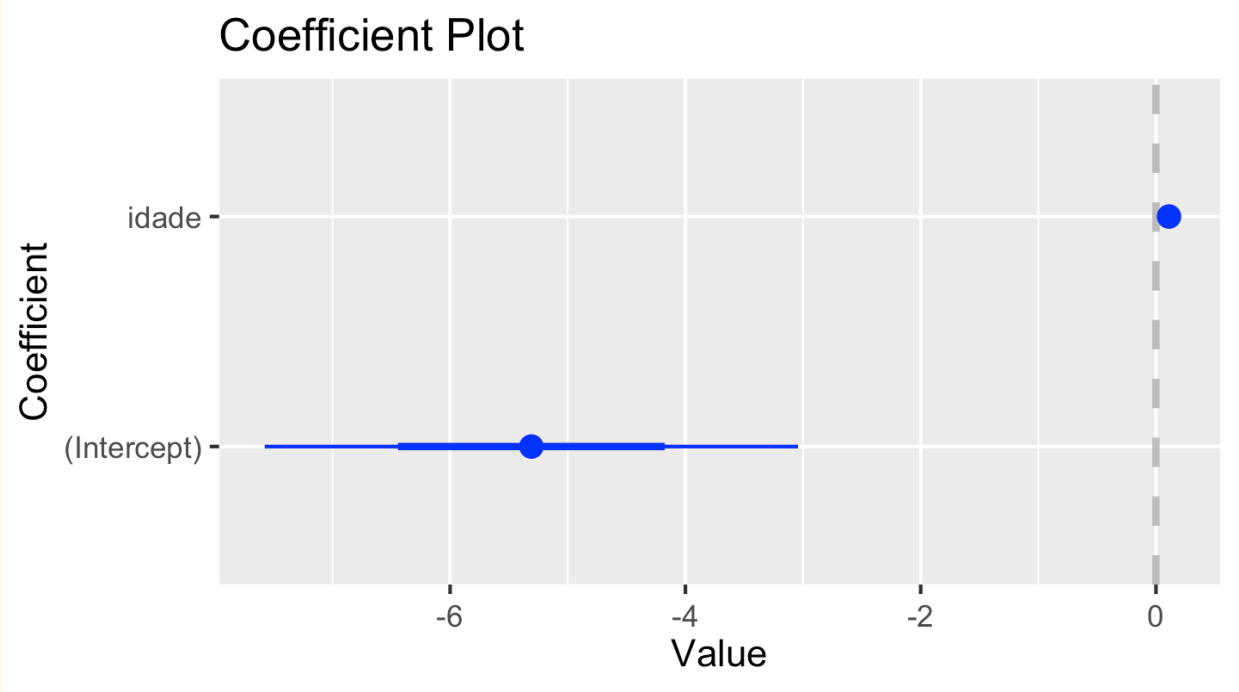
COEFICIENTES DO MODELO

```r
summary(chdfit1)
```

```
Call:
glm(formula = chd ~ idade, family = binomial(link = "logit"),
    data = chdage)

Coefficients:
            Estimate Std. Error z value   Pr(>|z|)
(Intercept) -5.30945    1.13365  -4.683 0.00000282 ***
idade        0.11092    0.02406   4.610 0.00000402 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 136.66  on 99  degrees of freedom
Residual deviance: 107.35  on 98  degrees of freedom
AIC: 111.35

Number of Fisher Scoring iterations: 4
```

PLOTAGEM DOS COEFICIENTES

```
1 coefplot::coefplot(chdfit1)
```



ENTENDER OS COEFICIENTES

- Parecido com o que nós conhecemos da regressão linear
- Os coeficientes em si representam os log odds que o resultado Y = 1.
- Pode ver no gráfico quais são positivos e quais negativos
- Gráfico indica também o tamanho do erro padrão para cada variável independente
- Para entender os coeficientes melhor, precisa calcular o *logit inverso*
- Este põe os coeficientes no intervalo entre 0 e 1
  - ou seja, probabilidade

LOGIT INVERSO

```
1 invlogit <- function(x) {
2    1/(1 + exp(-x))
3 }
4 invlogit(chdfit1$coefficients[2])
```

```
   idade
0.5277019
```

- Assim, podemos interpretar os resultados como probabilidades
- Com uma probabilidade acima de 50%, podemos dizer que idade tem uma relação positiva com a ocorrência de CHD

DESVIO E AIC

- 2a parte dos resultados são os equivalentes de R^2, medidas de qualidade do modelo
- Invés da variância, com `glm` falamos de desvio
- Queremos minimizar o *desvio residual*
- AIC = *Akaike's Information Criterion* (aqui = 111.3530927)
- AIC útil para comparar modelos
  - Nota menor melhor

ESTE MODELO

- Desvio Residual = 107.3530927
- AIC = 111.3530927

SEGUNDO MODELO PARA COMPARAÇÃO

- Modelo com Idade categórica – grupos de idade
- Esperança que podemos entender melhor as probabilidades relacionados aos grupos de idade mais específicos
  - Idosos mais propensos a CHD?
- Vamos usar `recode` do pacote `car`
  - Mais flexível que `recode` de `dplyr`

GRUPOS DE IDADE

```r
chdage$idgrp <- car::Recode(chdage$idade, "20:29 = '20-29'; 30:34 = '30-34'
                        35:39 = '35-39'; 40:44 = '40-44'; 45:49 = '45-49';
                        50:54 = '50-54'; 55:59 = '55-59'; 60:69 = '60-69'",
                        as.factor = TRUE)
```

MODELO DE GRUPOS

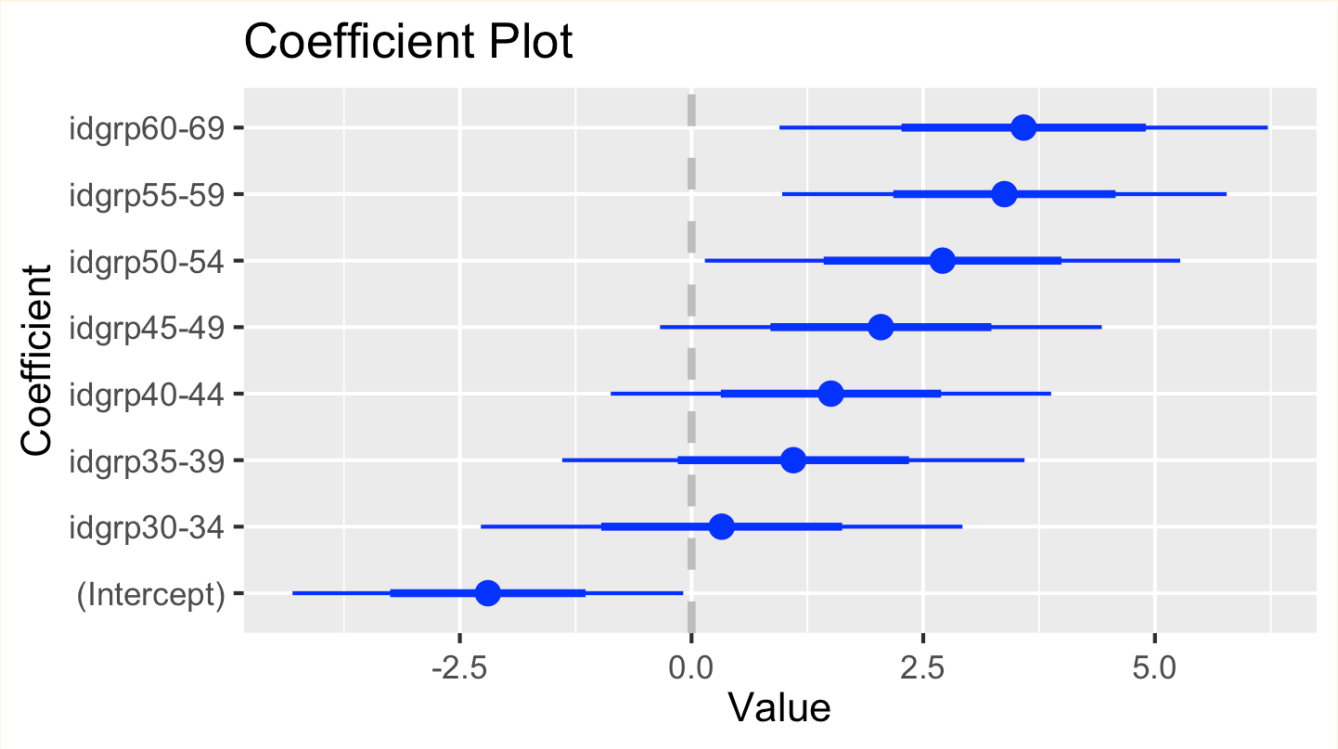
```r
chdfit2 <- glm(chd ~  idgrp, data = chdage,
               family = binomial(link = "logit"))
```

RESULTADOS

```
1 summary(chdfit2)
```

```
Call:
glm(formula = chd ~ idgrp, family = binomial(link = "logit"),
    data = chdage)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.1972     1.0540  -2.085  0.03710 *
idgrp30-34    0.3254     1.2992   0.250  0.80221
idgrp35-39    1.0986     1.2471   0.881  0.37837
idgrp40-44    1.5041     1.1878   1.266  0.20543
idgrp45-49    2.0431     1.1918   1.714  0.08649 .
idgrp50-54    2.7081     1.2823   2.112  0.03470 *
idgrp55-59    3.3759     1.1991   2.815  0.00487 **
idgrp60-69    3.5835     1.3175   2.720  0.00653 **
---
```

GRÁFICO DOS COEFICIENTES DO MODELO



IDOSOS TÊM ALTA PROBABILIDADE DE CHD

```
1  invlogit(coef(chdfit2)[5:8])
```

```
idgrp45-49 idgrp50-54 idgrp55-59 idgrp60-69
 0.8852459  0.9375000  0.9669421  0.9729730
```

QUAL MODELO PARECE MELHOR?

- Modelo 1 – Idade Numérica
  - Desvio Residual = 107.3530927
  - AIC = 111.3530927
- Modelo 2 – Idade Categórica
  - Desvio Residual = 107.9614654
  - AIC = 123.9614654
- AIC melhor no modelo numérico
- Mas, modelo categórico oferece mais informação sobre grupos de idade de interesse

EXEMPLO COM MÚLTIPLAS VARIÁVEIS INDEPENDENTES

OUTRO ESTUDO SOBRE CHD

- Pesquisadores querem identificar fatores causativos para CHD
- Covariados independentes
  - id (Número de identificação do caso)
  - idade (em anos)
  - bmi (índice de massa corporal em kg/m^2)
  - genero (0 = masculino, 1 = feminino)
- 65 casos
- Dados - `riscochd.RData`

```
Rows: 65
Columns: 5
$ id     <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
…
$ idade  <int> 75, 98, 91, 88, 56, 86, 93, 74, 56, 95, 64, 99, 68, 66, 95,
62,…
$ bmi    <dbl> 36.38134, 27.65790, 26.47878, 35.70601, 33.71147, 32.12082,
29.…
$ genero <fct> masculino, feminino, feminino, masculino, feminino, masculino,
…
$ chd    <fct> positivo, positivo, positivo, positivo, negativo, positivo,
pos…
```

ANÁLISE EXPLORATÓRIO

```
riscochd %>%
  select(idade, bmi) %>%
  descr(transpose = TRUE,
        stats = c("mean", "sd", "min", "q1", "med", "q3",
                  "max", "iqr", "cv"))
```

```
Descriptive Statistics
riscochd
N: 65

              Mean    Std.Dev      Min       Q1   Median       Q3      Max      IQR      CV
----------- ------- --------- ------- ------- -------- ------- ------- ------- ------
       bmi   28.42      5.36   16.78   25.18    28.06   31.47   44.94    6.30   0.19
     idade   71.38     17.67   33.00   56.00    74.00   84.00   99.00   28.00   0.25
```

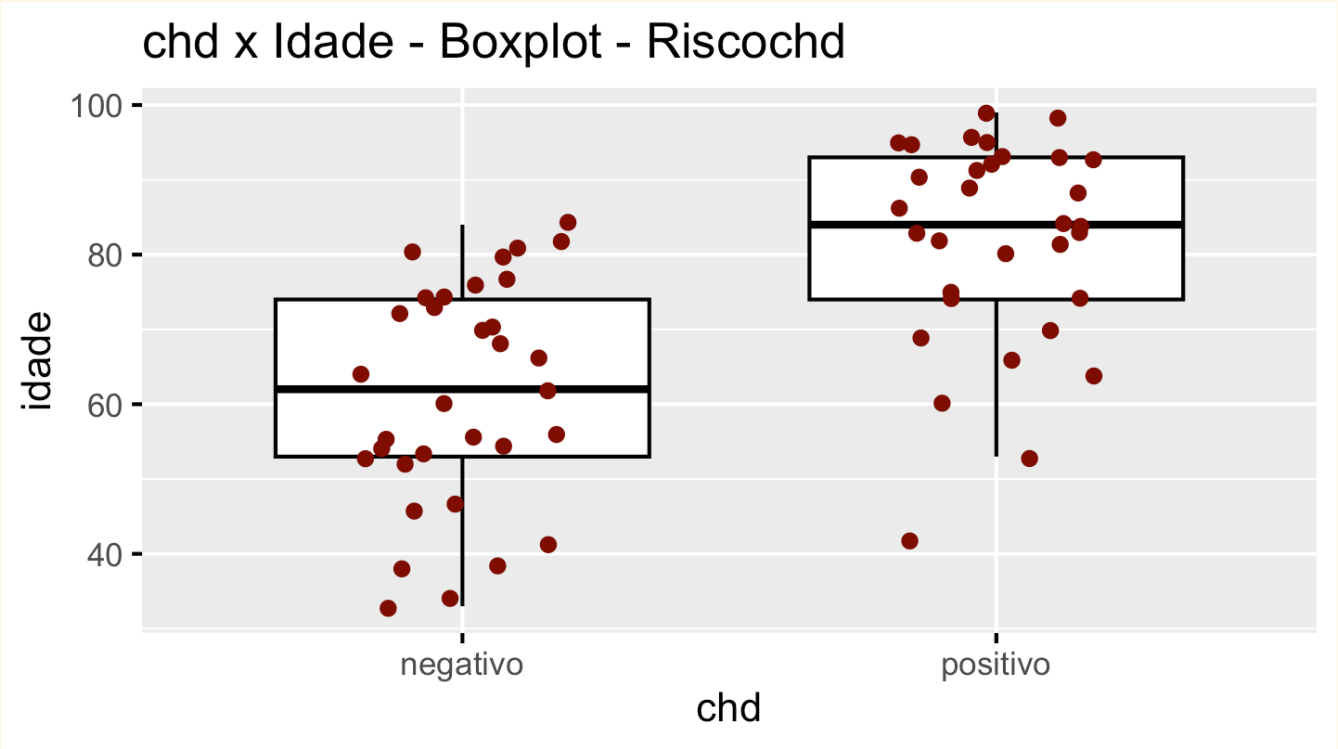
```r
riscochd %>%
  select(genero) %>%
  freq()
```

```
Frequencies
riscochd$genero
Type: Factor

                Freq   % Valid   % Valid Cum.   % Total   % Total Cum.
--------------- ------ --------- -------------- --------- --------------
      masculino     41     63.08          63.08     63.08          63.08
       feminino     24     36.92         100.00     36.92         100.00
           <NA>      0                               0.00         100.00
          Total     65    100.00         100.00    100.00         100.00
```

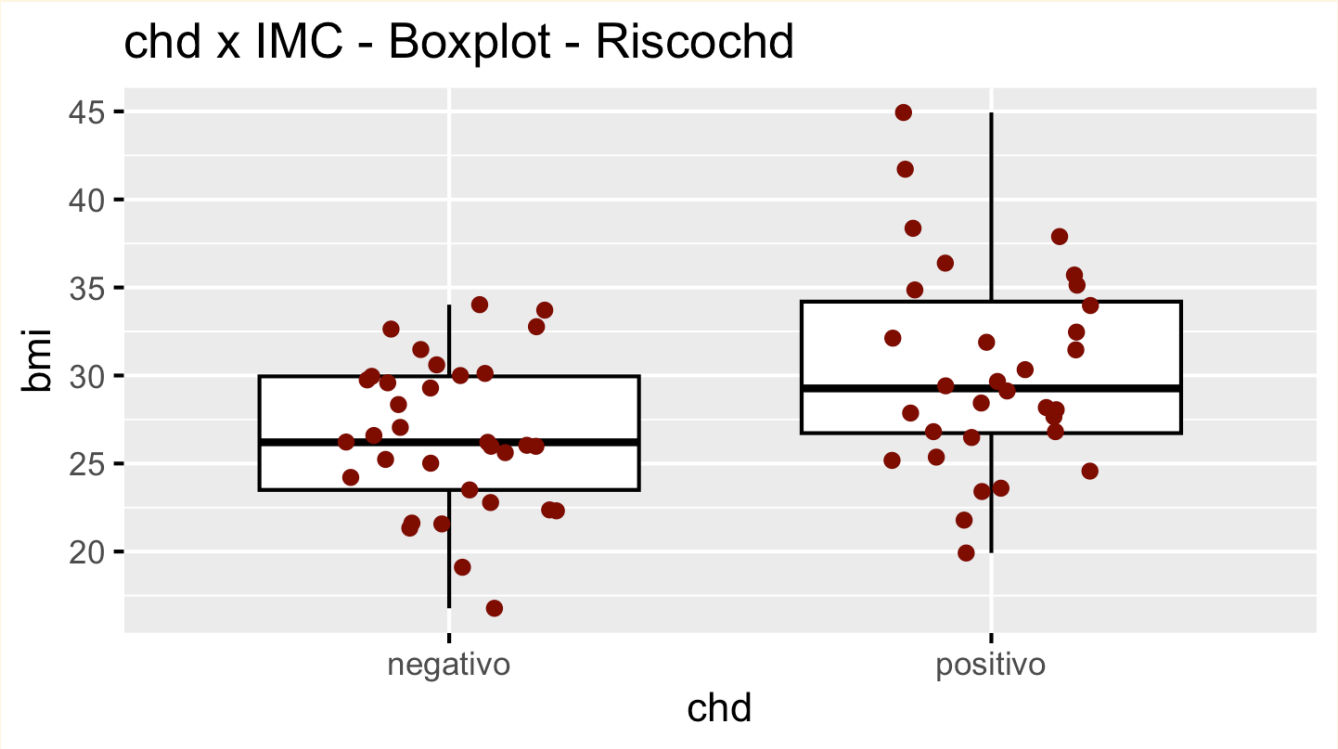
```r
riscochd %>%
  select(chd) %>%
  freq()
```

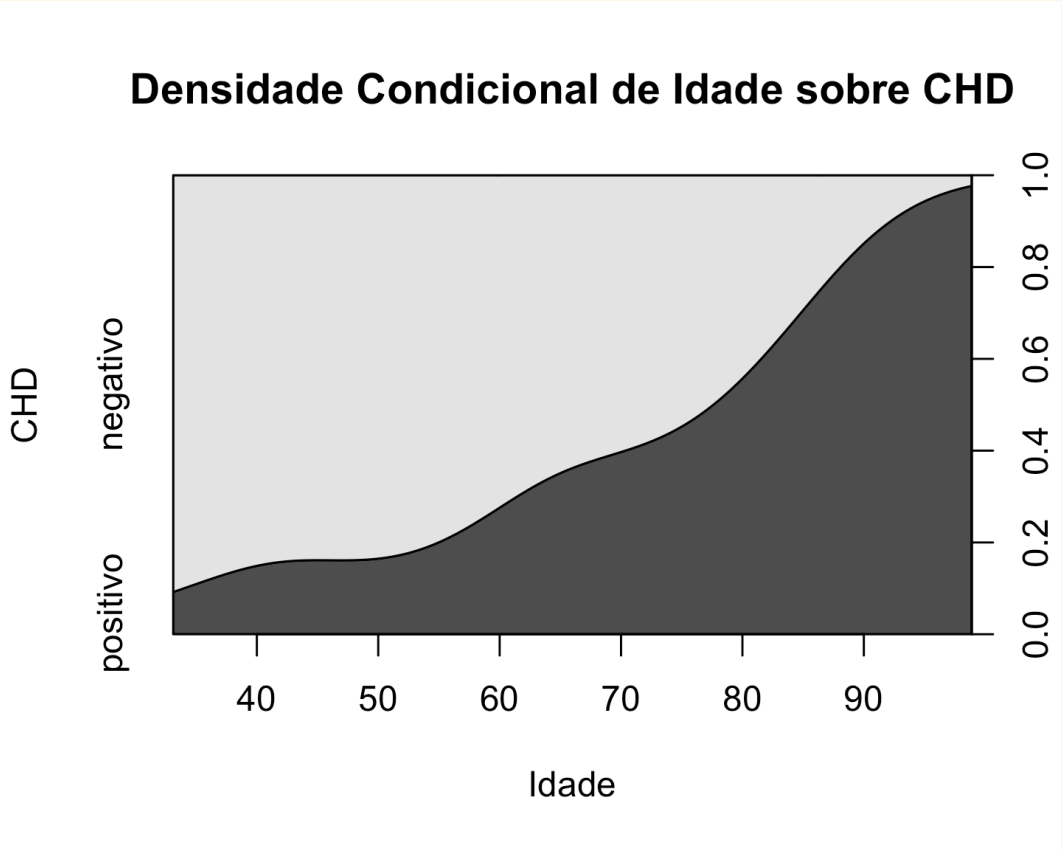
```
Frequencies
riscochd$chd
Type: Factor

               Freq   % Valid   % Valid Cum.   % Total   % Total Cum.
-------------- ------ --------- -------------- --------- --------------
      negativo     33     50.77          50.77     50.77          50.77
      positivo     32     49.23         100.00     49.23         100.00
          <NA>      0                               0.00         100.00
         Total     65    100.00         100.00    100.00         100.00
```

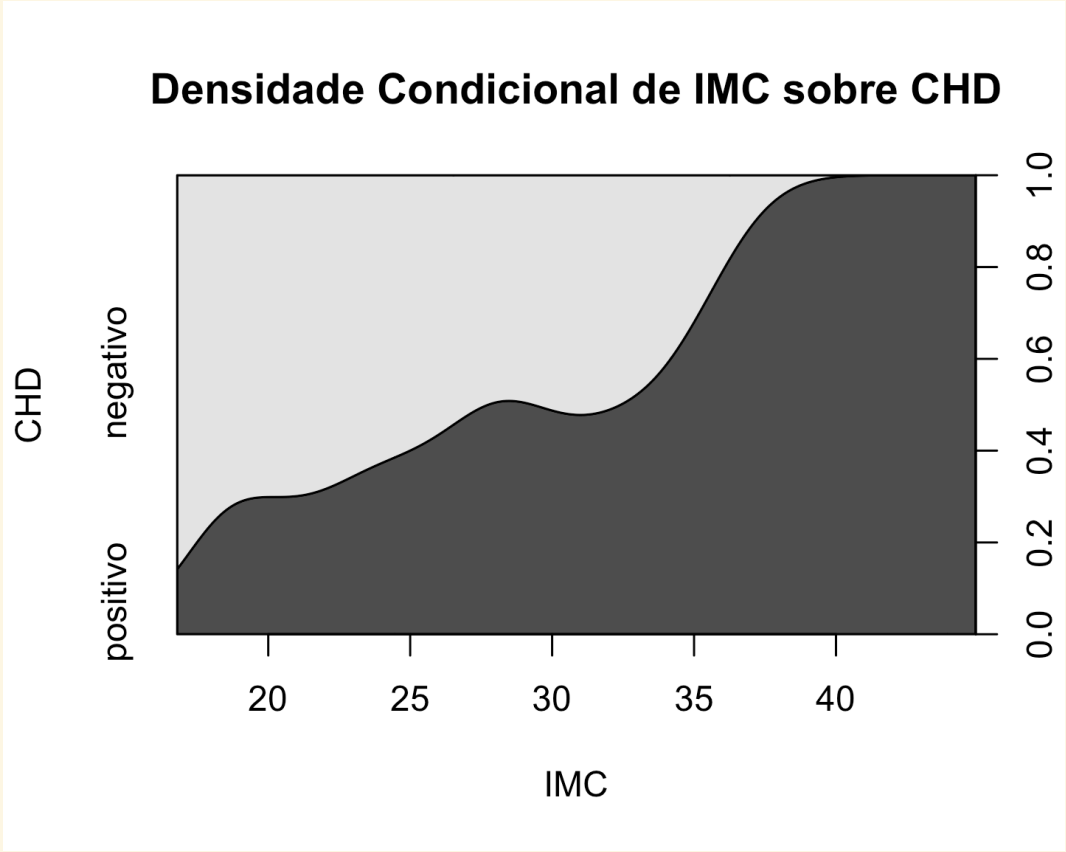
BOXPLOT DA IDADE



BOXPLOT DE IMC




Densidade Condicional de Idade sobre CHD
CHD
negativo
positivo
0.0
0.2
0.4
0.6
0.8
1.0
40
50
60
70
80
90
Idade


Densidade Condicional de IMC sobre CHD
CHD
negativo
positivo
0.0
0.2
0.4
0.6
0.8
1.0
20
25
30
35
40
IMC

MODELO 1 – TODAS AS VARIÁVEIS INDEPENDENTES

```r
chdfit3 <- glm(chd ~ idade + bmi + genero, data = riscochd,
               family = binomial(link = "logit"))
summary(chdfit3)
```

```
Call:
glm(formula = chd ~ idade + bmi + genero, family = binomial(link = "logit"),
    data = riscochd)

Coefficients:
               Estimate Std. Error z value  Pr(>|z|)
(Intercept)    -20.64336    5.06903  -4.072 0.0000465 ***
idade            0.14814    0.03822   3.876  0.000106 ***
bmi              0.34613    0.10189   3.397  0.000681 ***
generofeminino   0.45202    0.77568   0.583  0.560069
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 90.094  on 64  degrees of freedom
Residual deviance: 43.886  on 61  degrees of freedom
AIC: 51.886

Number of Fisher Scoring iterations: 6
```

MODELO 2 – USANDO SOMENTE A VARIÁVEL `idade`

```r
chdfit4 <- glm(chd ~ idade, data = riscochd,
               family = binomial(link = "logit"))
summary(chdfit4)
```

```
Call:
glm(formula = chd ~ idade, family = binomial(link = "logit"),
    data = riscochd)

Coefficients:
            Estimate Std. Error z value  Pr(>|z|)
(Intercept) -6.91677    1.79219  -3.859  0.000114 ***
idade        0.09495    0.02393   3.968 0.0000725 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 90.094  on 64  degrees of freedom
Residual deviance: 64.000  on 63  degrees of freedom
AIC: 68

Number of Fisher Scoring iterations: 5
```

SEGUNDO MODELO COMPARADO AO PRIMEIRO

- AIC aumentou com só `idade`
- Modelo piorou em qualidade

MODELO 3 – USANDO AS VARIÁVEIS `idade` E `bmi`

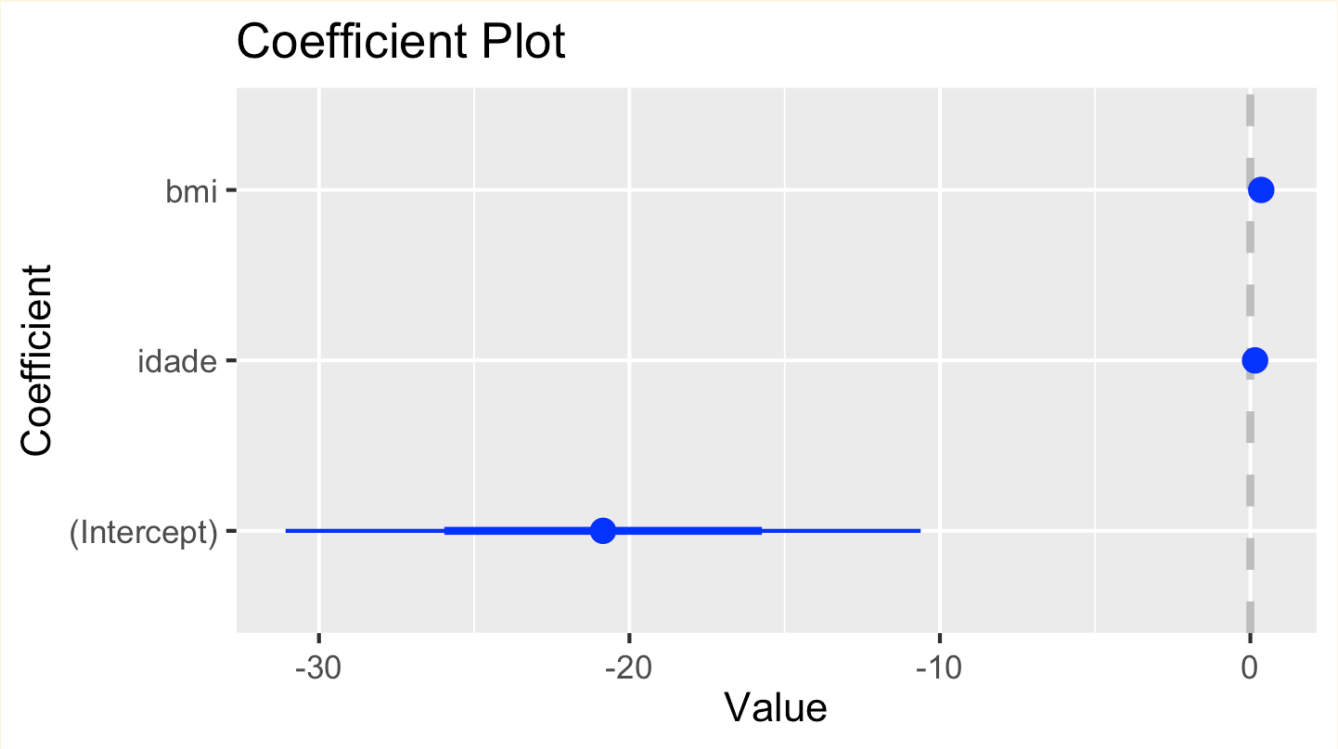
```r
chdfit5 <- glm(chd ~ idade + bmi, data = riscochd,
               family = binomial(link = "logit"))
summary(chdfit5)
```

```
Call:
glm(formula = chd ~ idade + bmi, family = binomial(link = "logit"),
    data = riscochd)

Coefficients:
             Estimate Std. Error z value  Pr(>|z|)
(Intercept) -20.84877    5.11434  -4.077 0.0000457 ***
idade         0.15229    0.03819   3.988 0.0000667 ***
bmi           0.35020    0.10196   3.435  0.000593 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 90.094  on 64  degrees of freedom
Residual deviance: 44.225  on 62  degrees of freedom
AIC: 50.225

Number of Fisher Scoring iterations: 6
```

DESEMPENHO DO NOVO MODELO

- De todos os três, tem o melhor AIC (50.2246163)
- Desvio residual fica muito perto (um pouco mais alto) do desvio do primeiro

GRÁFICO DE COEFICIENTES DO MODELO FINAL



RESULTADOS TRADUZIDOS EM PROBABILIDADE E ODDS

```r
paste("Relação de Odds:", exp(coef(chdfit5))) # Calculate the odds
```

```
[1] "Relação de Odds: 8.82049684688823e-10"
[2] "Relação de Odds: 1.16450293285125"
[3] "Relação de Odds: 1.4193517190987"
```

```r
exp(confint(chdfit5))
```

```
                  2.5 %         97.5 %
(Intercept) 6.449713e-15 0.000004578083
idade       1.092511e+00 1.272408489774
bmi         1.192024e+00 1.794849303037
```

```r
paste("Probabilidade de Ocorrência:", invlogit(chdfit5$coefficients))
```

```
[1] "Probabilidade de Ocorrência: 8.82049683910812e-10"
[2] "Probabilidade de Ocorrência: 0.538000163999444"
[3] "Probabilidade de Ocorrência: 0.586666133697776"
```

CONCLUSÃO SOBRE CONJUNTO DE `riscochd`

- Os 2 fatores no último modelo tem mais de 50% probabilidade de ser riscos para doenças cardíacas
- Modelos de regressão logística são difíceis de interpretar.
  - Log Odds, Odds ratios, AIC, etc.
- Modelo ainda muito importante e vai ser visto bastante

MODELO DE DIAGNOSE DE CÂNCER DE MAMA

- Dados vêm de Wisconsin dados sobre câncer de mama
- Características dos tumores de mama
- Variável dependente: diagnose (`diag`)

COIVARIÁVEIS - CARACTERÍSTICAS DOS TUMORES

- Vem de analise de imagens baseado na aspiração com agulha fina
- Características
  - Sample ID (code number)
  - Clump thickness
  - Uniformity of cell size
  - Uniformity of cell shape
  - Marginal adhesion
  - Single epithelial cell size
  - Number of bare nuclei
  - Bland chromatin
  - Number of normal nuclei
  - Mitosis

CARREGAR DADOS

```r
bc_data <- read.table(here::here("../breast-cancer-wisconsin-data.txt"),
                      header = FALSE,
                      sep = ",",
                      na.strings = "?")
colnames(bc_data) <- c("sample_code_number",
                       "clump_thickness",
                       "uniformity_of_cell_size",
                       "uniformity_of_cell_shape",
                       "marginal_adhesion",
                       "single_epithelial_cell_size",
                       "bare_nuclei",
                       "bland_chromatin",
                       "normal_nucleoli",
                       "mitosis",
                       "diag")


bc_data$diag <- ifelse(bc_data$diag == "2", "benign",
                       ifelse(bc_data$diag == "4", "malignant", NA))
```

DADOS

```
1 glimpse(bc_data)
```

```
Rows: 699
Columns: 11
$ sample_code_number          <int> 1000025, 1002945, 1015425, 1016277,
101702…
$ clump_thickness             <int> 5, 5, 3, 6, 4, 8, 1, 2, 2, 4, 1, 2, 5, 1,
…
$ uniformity_of_cell_size     <int> 1, 4, 1, 8, 1, 10, 1, 1, 1, 2, 1, 1, 3,
1,…
$ uniformity_of_cell_shape    <int> 1, 4, 1, 8, 1, 10, 1, 2, 1, 1, 1, 1, 3,
1,…
$ marginal_adhesion           <int> 1, 5, 1, 1, 3, 8, 1, 1, 1, 1, 1, 1, 3, 1,
…
$ single_epithelial_cell_size <int> 2, 7, 2, 3, 2, 7, 2, 2, 2, 2, 1, 2, 2, 2,
…
$ bare_nuclei                 <int> 1, 10, 2, 4, 1, 10, 10, 1, 1, 1, 1, 1, 3,
```

ANALISE DE NAS – DECISÃO SOBRE O QUE FAZER COM ELES

- Quantas NAs estão nos dados?

```
1 sum(is.na(bc_data))
```

```
[1] 16
```

- São todos na variável `bare_nuclei`

QUANTAS AMOSTRAS PERDEMOS SE RETIRAR OS NAS?

```
1 glue::glue("Número de casos perdidos: ", nrow(bc_data[is.na(bc_data), ]))
```

```
Número de casos perdidos: 16
```

```
1 glue::glue("Tamanho da base final: ", dim(drop_na(bc_data))[1])
```

```
Tamanho da base final: 683
```

OPÇÕES PARA RESOLVER NAS

- Eliminar casos com NA - `tidyr::drop_NA()`
- Preencher NAs com valores vizinhos - `tidyr::fill()`
  - Como feito com casos de tuberculose em Rússia
- Preencher com um outro valor - `tidyr::replace_na()`
  - Valor que você decide
  - Ex. 0 (`x <- x %>% mutate_all(replace_na, 0)`)
- Imputar valores com pacote `mice`

IMPUTAR VALORES COM `mice`

- Pacote e função `mice`
  - Multivariate Imputation by Chained Equations
- Cria dados imputados para dados incompletos multivariados
  - Gibbs Sampling (técnica bayesiana)
  - Gera valores plausíveis sintéticos dado as outras colunas no dataset
- Imputação introduza mais incerteza no modelo

```r
descr(bc_data$bare_nuclei, transpose = TRUE, # todos NA vem de bare_nuclei
      stats = c("mean", "sd", "med", "min", "max", "n.valid"))
```

```
Descriptive Statistics
bc_data$bare_nuclei
N: 699

                  Mean   Std.Dev   Median    Min     Max   N.Valid
---------------- ------ --------- -------- ------ ------- ---------
     bare_nuclei   3.54      3.64     1.00   1.00   10.00    683.00
```

```r
a_numero <- function(x) as.numeric(as.character(x))
mod_cols <- colnames(bc_data[2:10])
bc_data <- bc_data %>%
  mutate_at(mod_cols, ~a_numero(.), na.rm = TRUE)
dataset_impute <- mice::mice(bc_data[, 2:10],  print = FALSE)
bc_data <- cbind(diag = bc_data$diag, mice::complete(dataset_impute, 1))
descr(bc_data$bare_nuclei, transpose = TRUE, # todos NA vem de bare_nuclei
      stats = c("mean", "sd", "med", "min", "max", "n.valid"))
```

```
Descriptive Statistics
bc_data$bare_nuclei
N: 699

                  Mean   Std.Dev   Median    Min     Max   N.Valid
---------------- ------ --------- -------- ------ ------- ---------
     bare_nuclei   3.52      3.62     1.00   1.00   10.00    699.00
```

RESUMO DAS DIAGNOSES

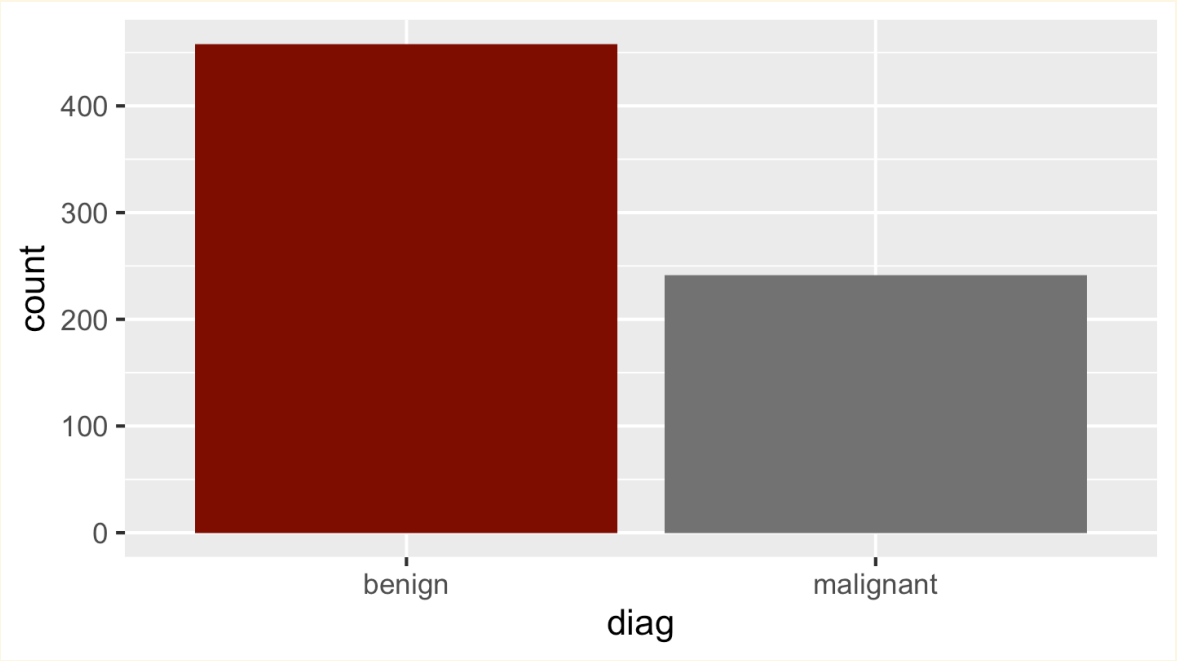
- Converter `diag` para um `factor`
- Quantos casos de benign e malignant têm?

```
1 bc_data$diag <- as.factor(bc_data$diag)
2  summary(bc_data$diag)
```

```
benign malignant
   458       241
```

GRÁFICO DAS DIAGNOSES

```
1 brgr1 <- ggplot(bc_data, aes(x = diag, fill = diag)) + geom_bar(fill = new_
2 brgr1
```



CLASSES DE `diag` DESEQUILIBRADAS

- Normalmente precisa um ajuste para tratar dessa desequilíbrio
- Não vamos fazer isso aqui

EXPLORAÇÃO DE ALGUMAS DAS COVARIÁVEIS

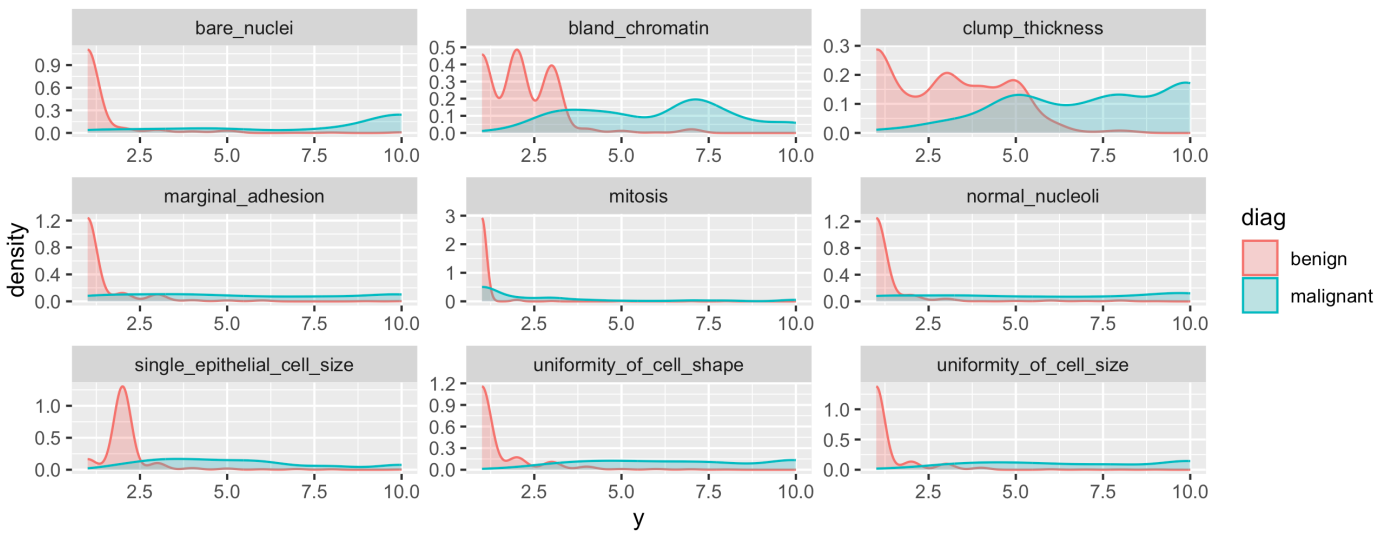
```r
bc_data %>%
  select(clump_thickness:mitosis) %>%
  descr(transpose = TRUE,
        stats = c("mean", "sd", "min", "q1", "med", "q3",
                  "max", "iqr", "cv"))
```

```
Descriptive Statistics
bc_data
N: 699

                                Mean   Std.Dev     Min      Q1   Median      Q3      Max     IQR      CV
------------------------------ ------ --------- ------ ------ -------- ------ -------- ------ ------
                    bare_nuclei   3.52      3.62   1.00   1.00     1.00   6.00    10.00   4.50   1.03
                 bland_chromatin   3.44      2.44   1.00   2.00     3.00   5.00    10.00   3.00   0.71
                 clump_thickness   4.42      2.82   1.00   2.00     4.00   6.00    10.00   4.00   0.64
               marginal_adhesion   2.81      2.86   1.00   1.00     1.00   4.00    10.00   3.00   1.02
                         mitosis   1.59      1.72   1.00   1.00     1.00   1.00    10.00   0.00   1.08
                 normal_nucleoli   2.87      3.05   1.00   1.00     1.00   4.00    10.00   3.00   1.07
     single_epithelial_cell_size   3.22      2.21   1.00   2.00     2.00   4.00    10.00   2.00   0.69
        uniformity_of_cell_shape   3.21      2.97   1.00   1.00     1.00   5.00    10.00   4.00   0.93
         uniformity_of_cell_size   3.13      3.05   1.00   1.00     1.00   5.00    10.00   4.00   0.97
```

GRÁFICO DAS COVARIÁVEIS COM A DIAGNOSE

```
gr_covars <- gather(bc_data, x, y, clump_thickness:mitosis) %>%
  ggplot(aes(x = y, color = diag, fill = diag)) +
    geom_density(alpha = 0.3) +
    facet_wrap( ~ x, scales = "free", ncol = 3)
```

```
1 gr_covars
```


bare_nuclei
bland_chromatin
clump_thickness
marginal_adhesion
mitosis
normal_nucleoli
single_epithelial_cell_size
uniformity_of_cell_shape
uniformity_of_cell_size
density
y
diag
benign
malignant

ANÁLISE DE COMPONENTES PRINCIPAIS (PCA)

- PCA – técnica para agrupar variáveis
- Neste caso
  - Mostra que os níveis de diagnose formam espaços coerentes
- Usa pacote `pcaGoPromoter` de Bioconductor
  - Pacote faz PCA e tem funções para ajudar na interpretação

GRÁFICO DE CORRELAÇÃO

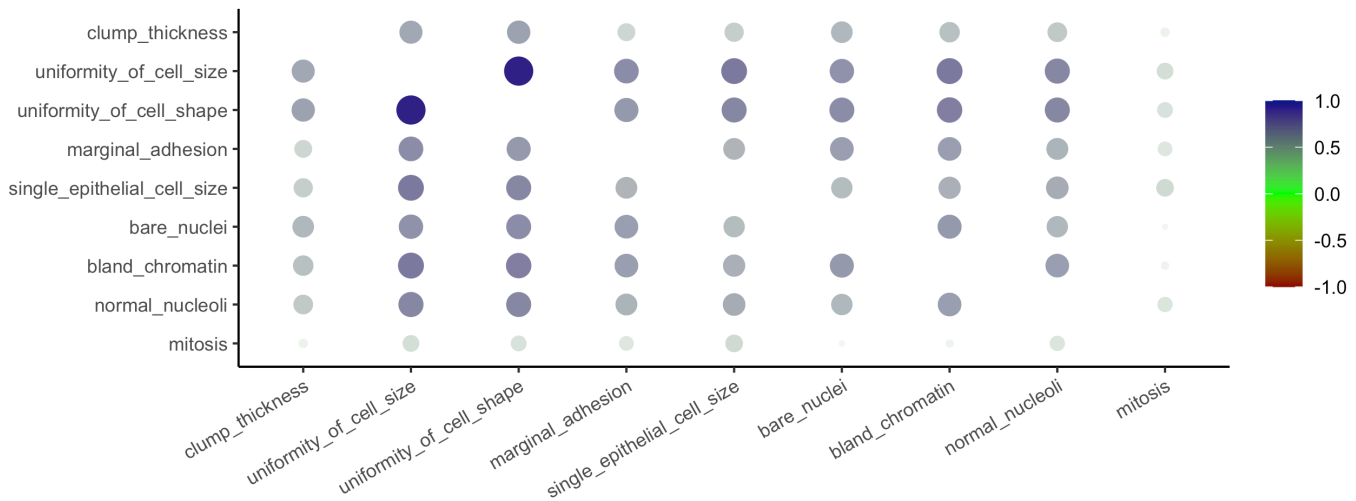
- Existem fortes ou fracas associações entre as covariáveis?
- Uso do pacote `corrr`

```
corrdf <- corrr::correlate(bc_data[,2:10])
cplot <- corrr::rplot(corrdf, legend = TRUE, colours =
                 c("darkred", "green", "darkblue"))
cplot <- cplot + theme(axis.text.x =
                         element_text(angle = 30, hjust = 1, vjust = 1))
```

```r
corrdf |>
  gt() |>
  fmt_number(decimals = 3) |>
  cols_width(clump_thickness:mitosis ~ px(75)) |>
  cols_label(clump_thickness = "clump",
             uniformity_of_cell_size = "unif_size",
             uniformity_of_cell_shape = "unif_shape",
             marginal_adhesion = "marg_ad",
             single_epithelial_cell_size = "single",
             bare_nuclei = "bare_nuc",
             bland_chromatin = "bland",
             normal_nucleoli = "normal",
             mitosis = "mitosis")
```

term	clump	unif_size	unif_shape	marg_ad	single	bare_nuc	bland	normal	mitosis
clump_thickness	NA	0.645	0.655	0.486	0.522	0.595	0.558	0.536	0.350
uniformity_of_cell_size	0.645	NA	0.907	0.706	0.752	0.694	0.756	0.723	0.459
uniformity_of_cell_shape	0.655	0.907	NA	0.683	0.720	0.715	0.736	0.719	0.439
marginal_adhesion	0.486	0.706	0.683	NA	0.600	0.668	0.667	0.603	0.418
single_epithelial_cell_size	0.522	0.752	0.720	0.600	NA	0.585	0.616	0.629	0.479
bare_nuclei	0.595	0.694	0.715	0.668	0.585	NA	0.680	0.587	0.339
bland_chromatin	0.558	0.756	0.736	0.667	0.616	0.680	NA	0.666	0.344
normal_nucleoli	0.536	0.723	0.719	0.603	0.629	0.587	0.666	NA	0.428
mitosis	0.350	0.459	0.439	0.418	0.479	0.339	0.344	0.428	NA

```
1 cplot
```


clump_thickness
uniformity_of_cell_size
uniformity_of_cell_shape
marginal_adhesion
single_epithelial_cell_size
bare_nuclei
bland_chromatin
normal_nucleoli
mitosis
1.0
0.5
0.0
-0.5
-1.0

TREINAMENTO E TESTE – DADOS SEPARADOS

PACOTE `caret`

- Funções para apoiar machine learning
- Pode conduzir todo a análise dentro de `caret`
- No grupos dos pacotes iniciais

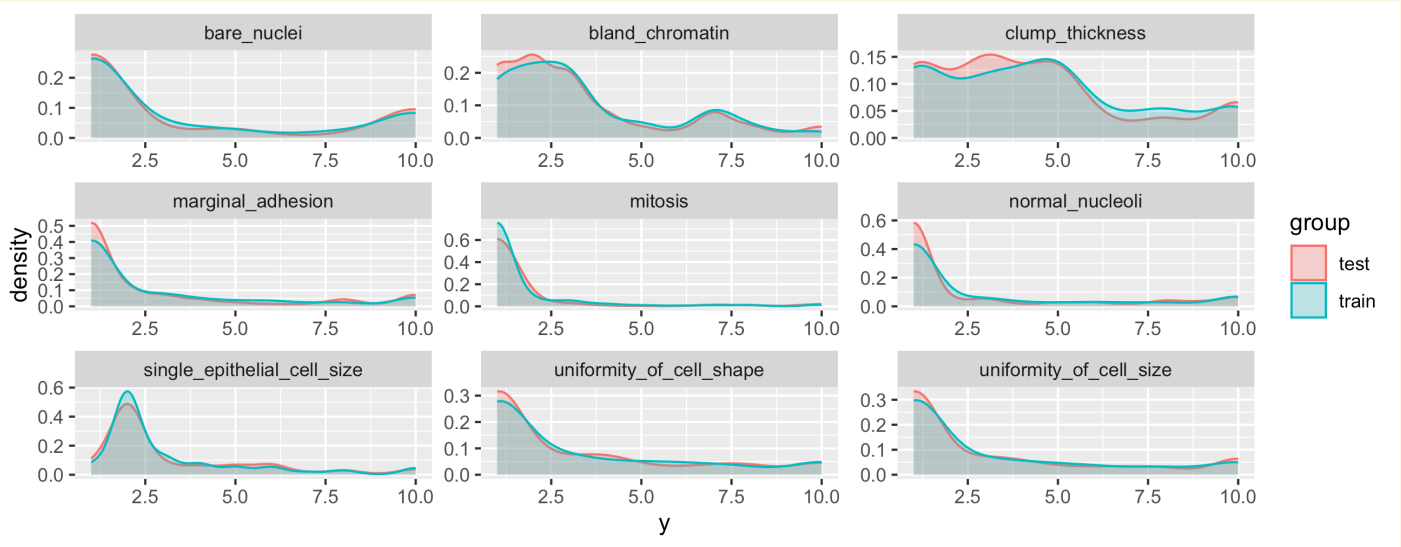
SEPARAR TREINAMENTO E TESTES

- Utilizar função `caret::createDataPartition()` para criar bases separadas
  - 1 para treinamento do modelo
  - 1 para testes
- Especificar (`p`) porcentagem de dados colocado na base de treinamento
- `createDataPartition()` estratifica os dados baseada nas proporções da variável y

CRIAR AS BASES TREINAMENTO E TESTES

```r
set.seed(42)
indice <- caret::createDataPartition(bc_data$diag, p = 0.7, list = FALSE)
train_data <- bc_data[indice, ] # use os índices para o treinamento
test_data <- bc_data[-indice, ] # use os outros para testes
```

AS BASES REFLETEM OS MESMOS DADOS?



EXEMPLOS DOS TIPOS DE MODELOS

- Regressão Logística
  - Ex: GLM
- Classificação com Arvores
  - Árvores recursivas de particionamento e regressão (pacote `rpart`)
  - Florestas Aleatórias (“Random Forests”)
- Todos com `caret`

CONTROLE DE TREINAMENTO

- Antes de iniciar o passo de treinar o modelos, precisamos decidir qual tipo de validação queremos usar
  - bootstrap, k-fold cross validation
- Especificar através da função `caret::trainControl()`
- Queremos usar *10-fold cross validation*
- Se pudermos repetir o processo de cross validation, faz a seleção do modelo ainda mais forte
  - Repetiremos 10 vezes

trainControl()

```r
set.seed(42)
control <- trainControl(method = "repeatedcv",
                        number = 10,
                        repeats = 10,
                        savePredictions = TRUE,
                        verboseIter = FALSE)
```

VARIÁVEL DEPENDENTE: `diag` (*BENIGN* OU *MALIGNANT*)

- Qual tipo de análise mais relacionado?
- Regressão logística

TREINAMENTO DO MODELO – REGRESSÃO LOGÍSTICA

```r
model_glm <- caret::train(diag ~ .,
                          data = train_data,
                          method = "glm",
                          preProcess = c("scale", "center"),
                          trControl = control)
```

MODELO

```
1 model_glm
```

```
Generalized Linear Model

490 samples
  9 predictor
  2 classes: 'benign', 'malignant'

Pre-processing: scaled (9), centered (9)
Resampling: Cross-Validated (10 fold, repeated 10 times)
Summary of sample sizes: 441, 441, 441, 441, 441, 441, ...
Resampling results:

  Accuracy   Kappa
  0.9555192  0.9012518
```

RESUMO DOS RESULTADOS DO MODELO

```
Call:
NULL

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)                -1.14976    0.30964  -3.713 0.000205 ***
clump_thickness             1.43477    0.41163   3.486 0.000491 ***
uniformity_of_cell_size    -0.36810    0.64784  -0.568 0.569899
uniformity_of_cell_shape    1.25713    0.72821   1.726 0.084288 .
marginal_adhesion           0.78093    0.35178   2.220 0.026424 *
single_epithelial_cell_size -0.07587    0.36520  -0.208 0.835421
bare_nuclei                 1.21462    0.35984   3.375 0.000737 ***
bland_chromatin             1.22395    0.43208   2.833 0.004616 **
normal_nucleoli             0.26938    0.36432   0.739 0.459659
mitosis                     0.99522    0.47810   2.082 0.037379 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 631.346  on 489  degrees of freedom
Residual deviance:  98.023  on 480  degrees of freedom
```

O MODELO PODE PREDIZER OS RESULTADOS DE TREINAMENTO E DE TESTE?

- Função `predict()`
  - com modelo e valores para ser usados para previsão
- Aplicado a base de `train` como exemplo
- Mais interessante – base de `test`
  - Modelo nunca viu esses dados antes
- **Teste ácido**

PREVISÕES

```
predtr <- predict(model_glm, train_data)
predtest <- predict(model_glm, test_data)
tabyl(predtest) %>% adorn_pct_formatting()
```

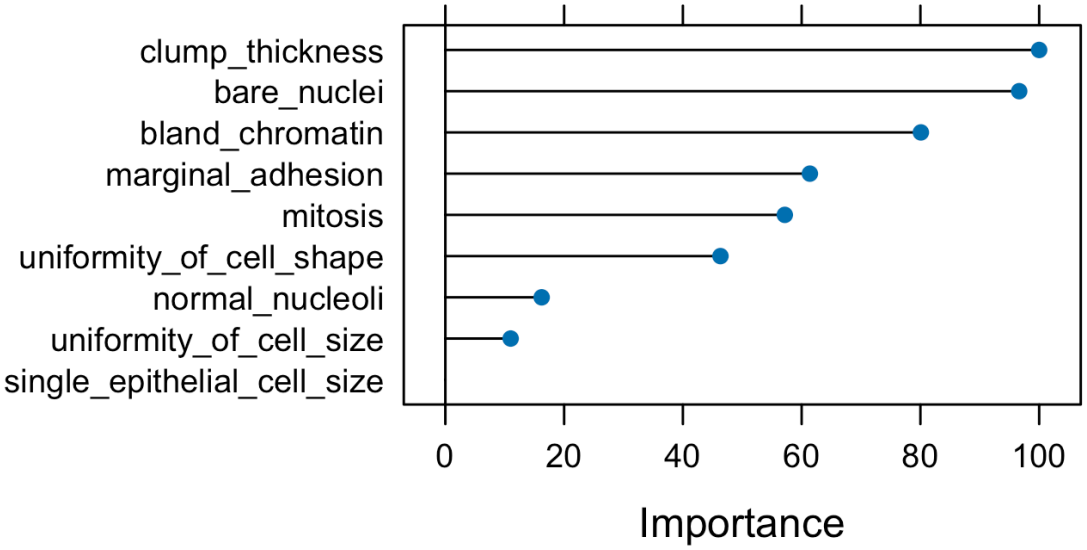
```
  predtest   n percent
    benign 138   66.0%
 malignant  71   34.0%
```

```
tabyl(predtr) %>% adorn_pct_formatting()
```

```
   predtr   n percent
   benign 321   65.5%
malignant 169   34.5%
```

QUAIS VARIÁVEIS TÊM IMPORTÂNCIA PARA O MODELO

```
1 plot(caret::varImp(model_glm))
```

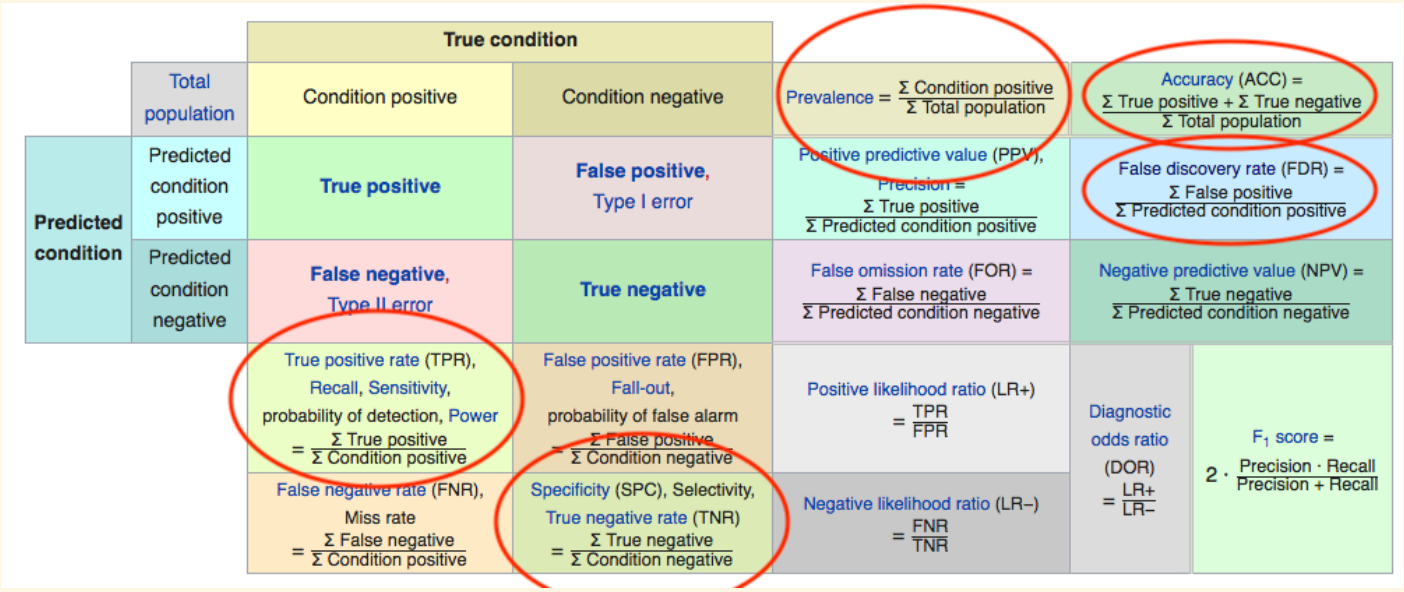


MATRIZ DE CONFUSÃO - UMA TABELA DE VERDADE

		True condition	
	Total population	Condition positive	Condition negative
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error
	Predicted condition negative	**False negative**, Type II error	**True negative**

- Maneira de comparar as previsões com a verdade
- Se as previsões não são corretas, tem ou Erro de Tipo I ou Tipo II
  - Tipo I - Falso positivo
  - Tipo II - Falso negativo

CÁLCULOS POSSÍVEIS COM A MATRIZ DE CONFUSÃO



		True condition				
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$	F_1 score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$		

PREVISÕES COM OS DADOS DE TESTE – MATRIZ DE CONFUSÃO

```
1 confusionMatrix(predtest, test_data$diag, positive = "malignant")
```

```
Confusion Matrix and Statistics

          Reference
Prediction  benign malignant
  benign       135         3
  malignant      2        69

               Accuracy : 0.9761
                 95% CI : (0.9451, 0.9922)
    No Information Rate : 0.6555
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9469

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9583
            Specificity : 0.9854
         Pos Pred Value : 0.9718
         Neg Pred Value : 0.9783
             Prevalence : 0.3445
         Detection Rate : 0.3301
   Detection Prevalence : 0.3397
```

PREVISÕES COM OS DADOS DE TREINAMENTO – MATRIZ DE CONFUSÃO

```
1 confusionMatrix(predtr, train_data$diag, positive = "malignant")
```

```
Confusion Matrix and Statistics

           Reference
Prediction  benign malignant
  benign       311        10
  malignant     10       159

               Accuracy : 0.9592
                 95% CI : (0.9377, 0.9749)
    No Information Rate : 0.6551
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9097

 Mcnemar's Test P-Value : 1
```

“RECEIVER OPERATING CHARACTERISTIC” (ROC) VALIDAÇÃO DO MODELO

- Desenvolvido ao início da WWII para discriminar o que foi o sinal recebido pela nova tecnologia, *radar*
  - Avião ou pássaro
- Mede *sensibilidade* vs. *especificidade* de um modelo
- *Sensibilidade* = % do resultado positivo correto
  - Teste mede % dos resultados positivos das pessoas com uma doença
  - Taxa de previsões positivas certas (“True positive rate”, TPR)
- *Especificidade* = % do resultado negativo correto
  - Teste mede % dos resultados negativos das pessoas sem uma doença

AUC (ÁREA ABAIXO DA CURVA)

- AUC mede quanto porcentagem da área do gráfico a curva do modelo ROC cobre
- 100% quer dizer que o modelo é perfeitamente sensível e especifico
- 50% quer dizer que o resultado é puramente aleatório
- Modelos com AUC maiores prevem melhor que eles com AUC menores
- Pergunta:
  - Como calcular área abaixo de uma curva qualquer em matemática?

ROC EM R

- 2 Pacotes
  - `pROC`
  - `ROCR`
- Iguais (basicamente)
- Faremos aqui `pROC`
  - Comando principal – `roc`

`pROC::roc()`

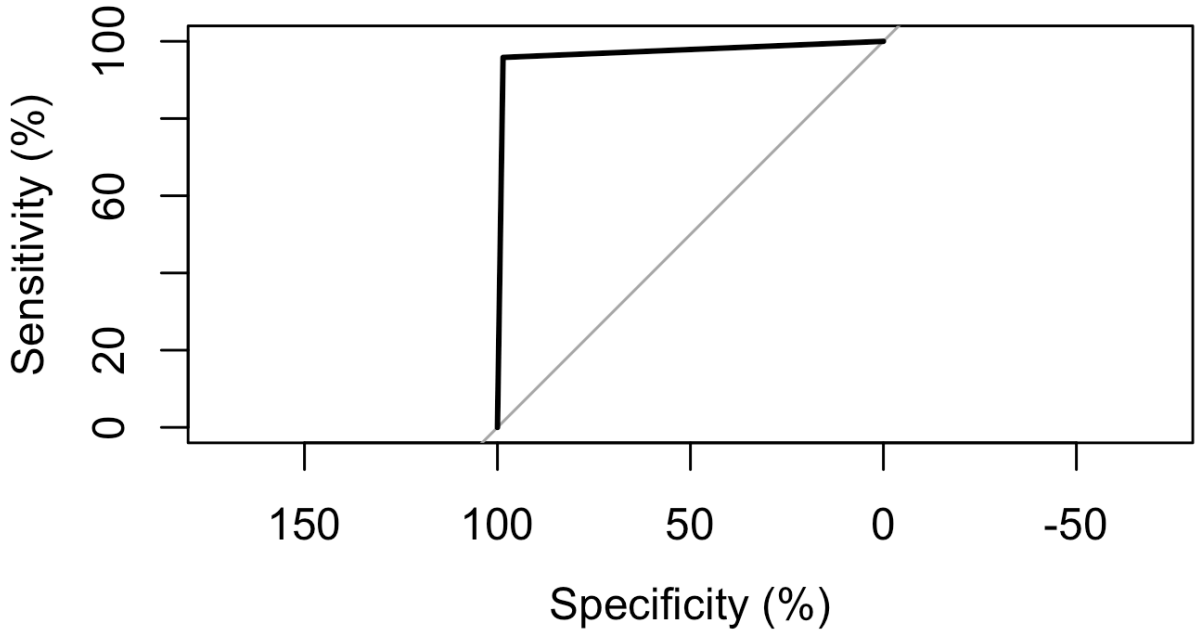
- Compara as previsões contra as observações
- Previsões precisam ser numéricas (não `factor`)
- Use as opções seguintes:
  - `plot = TRUE, percent = TRUE, ci = TRUE, grid = TRUE`
- Produz um gráfico e dados sobre o AUC

CHAMADA E ESTATÍSTICAS

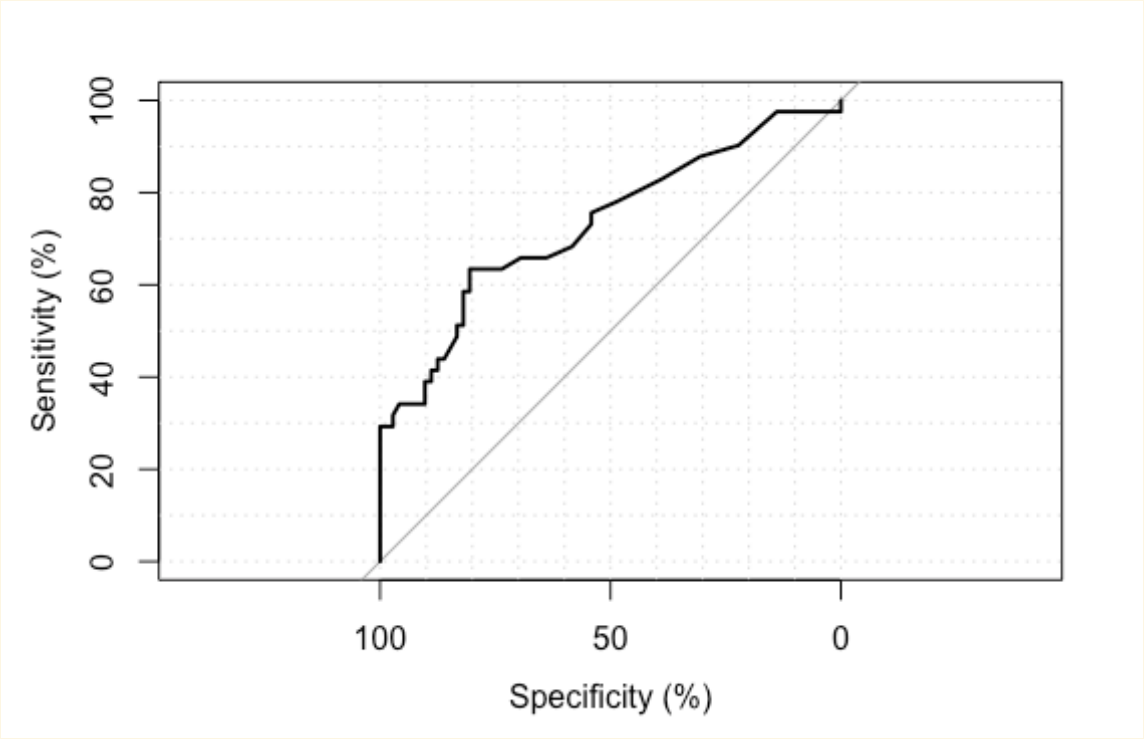
```r
## colocar predtest na faixa de 0:1 (atualmente 1:2)
predtestroc <- as.numeric(predtest) -1
rocteste <- pROC::roc(response = test_data$diag,
                      predictor = predtestroc,
                      levels = c("benign", "malignant"),
                      plot = FALSE, percent = TRUE,
                      ci = TRUE, grid = TRUE)
rocteste
```

```
Call:
roc.default(response = test_data$diag, predictor = predtestroc,     levels =
c("benign", "malignant"), percent = TRUE, ci = TRUE,     plot = FALSE, grid =
TRUE)

Data: predtestroc in 137 controls (test_data$diag benign) < 72 cases
(test_data$diag malignant).
Area under the curve: 97.19%
95% CI: 94.65%-99.72% (DeLong)
```

GRÁFICO



OUTRA CURVA ROC COM DADOS MAIS VARIÁVEIS



Curva ROC

LIMITES DA DECISÃO SOBRE `diag`

- Onde no gráfico fica a troca ótima?
  - No ponto mais para cima e para esquerda
- `pROC::coords()` pode calcular este ponto
- Precisa dar as seguintes informações a função:
  - nome de objeto de ROC
  - Palavra “best”
  - Coordenados para retornar a você (“threshold”)

LIMITES DE NOSSO MODELO

```
1 pROC::coords(rocteste, "best", ret = "threshold")
```

```
  threshold
1       0.5
```

ARVORES DE DECISÃO

NOVO MODELO - MODELOS DE ARVORE - `rpart`

- Modelos que constroem arvores de decisão
- Excelentes para problemas de classificação
- Pacote `rpart`
- Gráficos mostra como escolha das classes está sendo feita
  - Gráfico vem do pacote `rpart.plot`

COMO FUNCIONA UMA ARVORE

- Cf. Kuhn & Johnson, *Applied Predictive Modeling* (2013)
- Feita de *nodos* e *ramos*
- Ramos conectam nodos até que chegar num nodo terminal
- Algoritmo cria uma serie de partilhas (divisões) baseado em testes lógicos aninhados
- Os testes lógicos definem a previsão que o modelo faria com novos dados

EXEMPLO DE UMA REGRA DE UMA ARVORE

```
if Predictor A >= 1.7 then
|   if Predictor B >= 202.1 then Outcome = 1.3
|   else Outcome = 5.6
else Outcome 2.5
```

ARVORES SÃO UMA TÉCNICA DE MACHINE LEARNING POPULAR

- Interpretação fácil
- Podem lidar com muitas covariáveis de vários tipos
- Não precisa descrever exatamente a relação entre
  - Variável dependente
  - Variáveis independentes
- NA's não criam problemas
- Mas, tem desvantagens também
  - São instáveis (pequena mudança numa variável pode cause grande mudança no resultado)
  - Exatidão de previsões não tão boa que outros tipos de modelos

FUNCIONAMENTO DO MODELO DE ARVORE

- Algoritmo divide os dados em grupos menores que são mais homogêneos com a dependente
- 3 Critérios para divisão
  - Qual variável de previsão para usar para o “split”
  - Profundidade da arvore
  - A equação de previsão nos nodos terminais
- Metodologia de `rpart` vem de Breiman et. al (1984)
  - Classification and regression tree (CART)

PARÂMETROS CHAVES PARA `rpart`

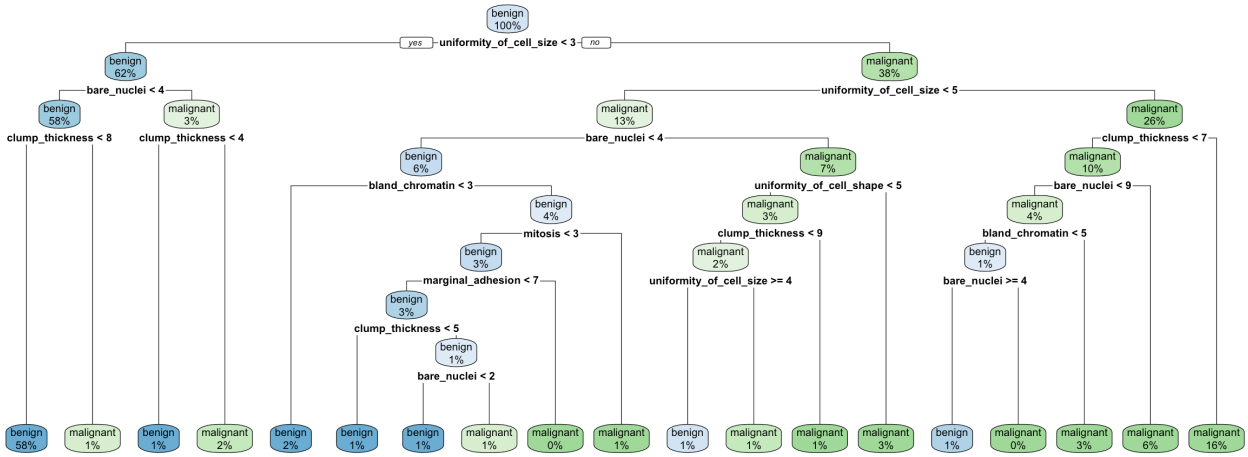
- `method`
  - Para classificação: “class”
  - Para regressão: “anova”
- `control`
  - Vai chamar `rpart.control` explicito
  - `xval`: número de cross-validations
  - `minbucket`: número mínimo de observações em um nodo terminal
- `parms` – parâmetros para dividindo os casos
  - Só usado para classificação
  - `information`

rpart MODELO DE CÂNCER DE MAMA

```
pacman::p_load(rpart, rpart.plot)
set.seed(42)
fitree1 <- rpart::rpart(diag ~ .,
            data = train_data,
            method = "class",
            control = rpart.control(xval = 10,
                                    minbucket = 2,
                                    cp = 0),
             parms = list(split = "information"))
```

ARVORE

```
1 rpart.plot(fitree1, extra = 100)
```



RESUMO DO MODELO DE `rpart`

```
1 summary(fitree1, cp = 1)
```

```
Call:
rpart::rpart(formula = diag ~ ., data = train_data, method = "class",
    parms = list(split = "information"), control = rpart.control(xval = 10,
        minbucket = 2, cp = 0))
  n= 490

           CP nsplit  rel error    xerror       xstd
1 0.757396450      0 1.00000000 1.0000000 0.06226029
2 0.038461538      1 0.24260355 0.3017751 0.03999748
3 0.020710059      3 0.16568047 0.2130178 0.03417389
4 0.008875740      5 0.12426036 0.1893491 0.03236108
5 0.005917160      8 0.09467456 0.1952663 0.03282693
6 0.004437870     11 0.07692308 0.1952663 0.03282693
7 0.001972387     15 0.05917160 0.2011834 0.03328412
8 0.000000000     18 0.05325444 0.2071006 0.03373300
```

PREVISÕES COM A ARVORE

```r
predtest <- predict(fitree1, newdata = test_data, type = "class")
prop.table(table(predtest))
```

```
predtest
   benign malignant
0.6698565 0.3301435
```

CONFUSION MATRIX – ARVORE

```r
confusionMatrix(predtest, test_data$diag, positive = "malignant")
```

```
Confusion Matrix and Statistics

          Reference
Prediction  benign malignant
  benign       135         5
  malignant      2        67

               Accuracy : 0.9665
                 95% CI : (0.9322, 0.9864)
    No Information Rate : 0.6555
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9251

 Mcnemar's Test P-Value : 0.4497
```

ROC DADOS

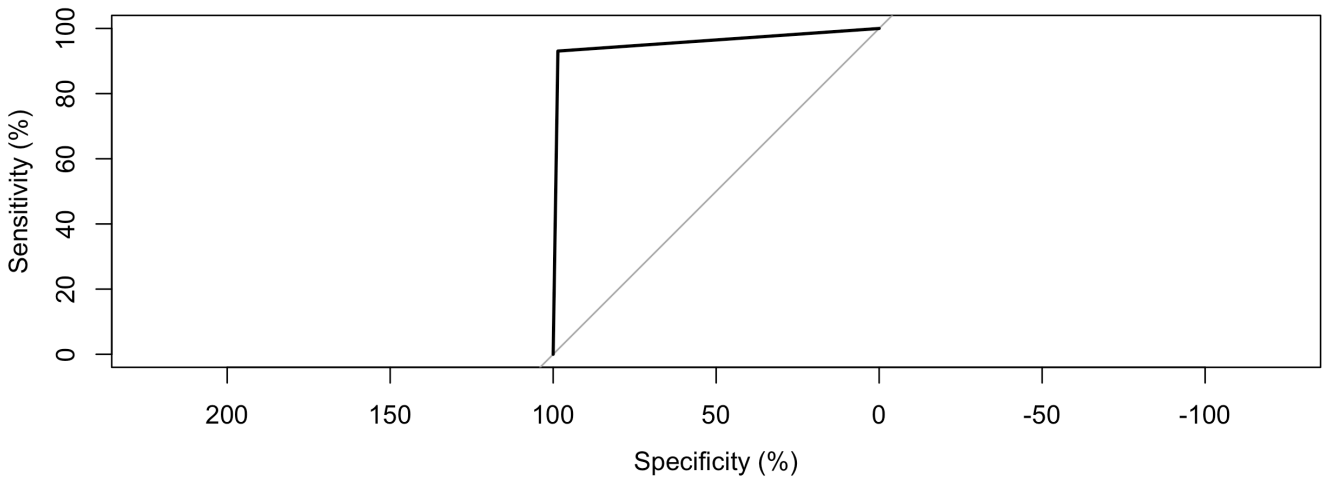
```r
## colocar predtest na faixa de 0:1 (atualmente 1:2)
predtestroc <- as.numeric(predtest) -1
rocteste <- pROC::roc(response = test_data$diag,
                      predictor = predtestroc,
                      levels = c("benign", "malignant"),
                      plot = FALSE, percent = TRUE,
                      ci = TRUE, grid = TRUE)
rocteste
```

```
Call:
roc.default(response = test_data$diag, predictor = predtestroc,     levels =
c("benign", "malignant"), percent = TRUE, ci = TRUE,     plot = FALSE, grid =
TRUE)

Data: predtestroc in 137 controls (test_data$diag benign) < 72 cases
(test_data$diag malignant).
Area under the curve: 95.8%
95% CI: 92.67%-98.92% (DeLong)
```

```r
pROC::coords(rocteste, "best", ret = "threshold")
```

```
  threshold
1       0.5
```

GRÁFICO


Sensitivity (%)
0
20
40
60
80
100
200
150
100
50
0
-50
-100
Specificity (%)

ARVORES MAIS ROBUSTAS – RANDOM FORESTS

- Random Forests elaborado como algoritmo por Breiman em 2000
- Ideia básica: Combinando resultados de muitas arvores vai produzir uma arvore final melhor

> Grow many deep regression trees to randomized versions of the training data, and average them. *Efron & Hastie, 2016*

- “Randomized versions” – pode ser bootstrapping ou outras técnicas de re-amostragem

ALGORITMO DE RANDOM FORESTS

1 Select the number of models to build, m
2 **for** $i = 1$ *to* m **do**
3 Generate a bootstrap sample of the original data
4 Train a tree model on this sample
5 **for** *each split* **do**
6 Randomly select k ($< P$) of the original predictors
7 Select the best predictor among the k predictors and partition the data
8 **end**
9 Use typical tree model stopping criteria to determine when a tree is complete (but do not prune)
10 **end**

Algorithm 8.2: Basic Random Forests

Kuhn & Johnson (2013)

RANDOM FORESTS EM R

- Pacote `randomForest`
- Formato:

```
randomForest(y ~ xvars, data = dados, ntrees = 1000,
             importance = TRUE)
```

- `y` deve ser expressa como `factor` para classificação
- Argumentos chaves:
  - `ntrees`: número de arvores para a calcular; deve ser muito maior que o número das covariáveis
  - `importance = TRUE`: para calcular os valores para importância dos variáveis

RANDOM FORESTS APLICADO AO CÂNCER DE MAMA

```
arvores = 100
rffit <- randomForest::randomForest(as.factor(diag) ~ ., data = train_data,
                      ntree = arvores, importance = TRUE, proximity = TRUE)
rffit
```

```
Call:
 randomForest(formula = as.factor(diag) ~ ., data = train_data,      ntree =
arvores, importance = TRUE, proximity = TRUE)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 3

        OOB estimate of  error rate: 4.08%
Confusion matrix:
          benign malignant class.error
benign       310        11  0.03426791
malignant      9       160  0.05325444
```

N.B. Confusion Matrix aqui é dos dados de treinamento

OOB ERROR????

- “Out of Bag”
  - Para todos as arvores, os erros associados com os valores não utilizados no treinamento do modelo
  - Os valores excluídos durante validação cruzada

PREVISÕES COM A RANDOM FOREST

```r
predtest <- predict(rffit, newdata = test_data, type = "class")
prop.table(table(predtest))
```

```
predtest
   benign malignant
0.6555024 0.3444976
```

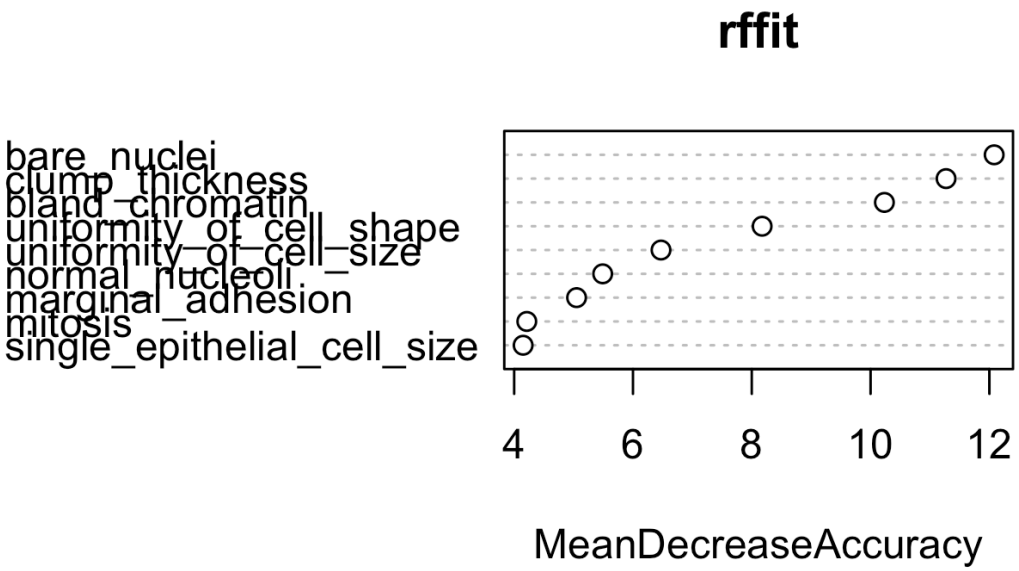
DESEMPENHO DE RANDOM FOREST

```r
confusionMatrix(predtest, test_data$diag)
```

```
Confusion Matrix and Statistics

           Reference
Prediction  benign malignant
  benign       135         2
  malignant      2        70

               Accuracy : 0.9809
                 95% CI : (0.9517, 0.9948)
    No Information Rate : 0.6555
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9576

 Mcnemar's Test P-Value : 1
```

IMPORTÂNCIA DAS VARIÁVEIS

```
1 randomForest::varImpPlot(rffit, type = 1)  ## NB, função dentro de randomFo
```



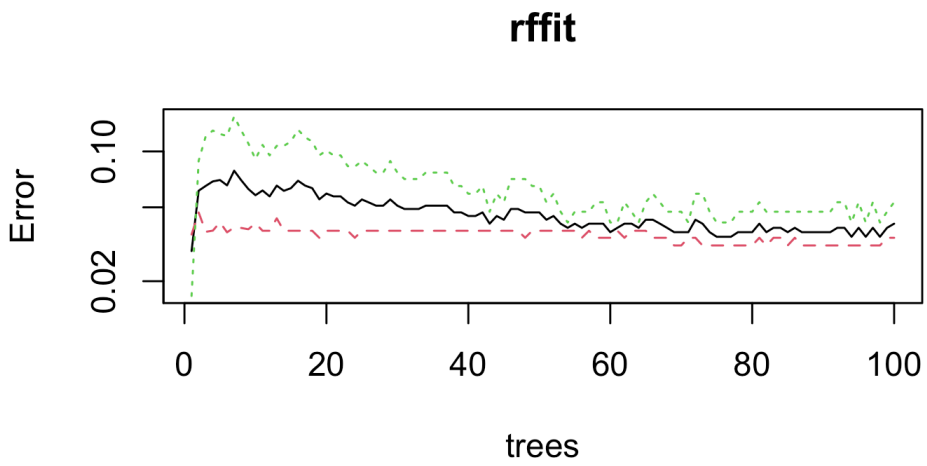
O QUE QUER DIZER “MEAN DECREASE ACCURACY”

- Através de todos as arvores - A variável causa uma perda de precisão no modelo
- Variáveis que podem causar perda de precisão são mais importantes
- Exemplos:
  - “bare nuclei” é a mais importante porque pode causar mais perda
  - “mitosis” é o menos importante, porque qualquer valor que assuma não vai afetar o resultado do modelo, `diag`

CONTROLE DE ERROS

- Gráfico de redução de MSE com o número de arvores calculadas

```
1 plot(rffit, log = 'y')
```



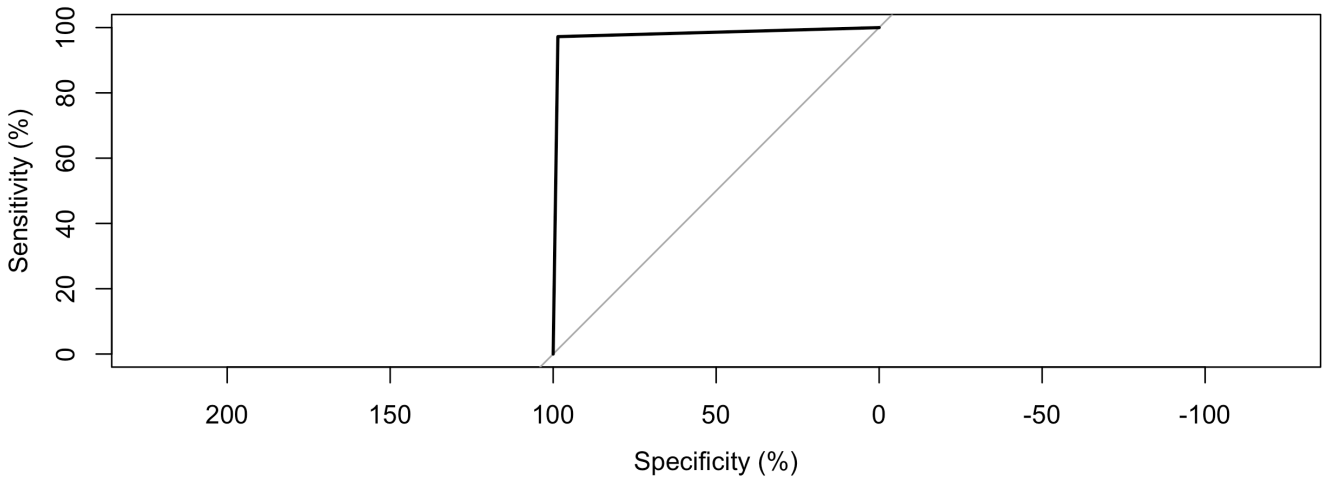
CURVA ROC E AUC PARA RANDOM FORESTS

```r
## colocar predtest na faixa de 0:1 (atualmente 1:2)
predtestroc <- as.numeric(predtest) -1
rocteste <- pROC::roc(response = test_data$diag,
                      predictor = predtestroc,
                      levels = c("benign", "malignant"),
                      plot = FALSE, percent = TRUE,
                      ci = TRUE, grid = TRUE)
rocteste
```

```
Call:
roc.default(response = test_data$diag, predictor = predtestroc,     levels =
c("benign", "malignant"), percent = TRUE, ci = TRUE,     plot = FALSE, grid =
TRUE)

Data: predtestroc in 137 controls (test_data$diag benign) < 72 cases
(test_data$diag malignant).
Area under the curve: 97.88%
95% CI: 95.72%-100% (DeLong)
```

```r
pROC::coords(rocteste, "best", ret = "threshold")
```

```
  threshold
1       0.5
```


Sensitivity (%)
0
20
40
60
80
100
200
150
100
50
0
-50
-100
Specificity (%)

FAZER RANDOM FORESTS COM `caret`

- Só precisa mudar o a especificação de `train`
- `method = "rf"`
- `caret` chama `randomForest` para fazer os cálculos
  - *wrapper* função
- Aqui vamos fazer `set.seed(42)` para ser consistente com os outros métodos

CALCULAR OS RANDOM FORESTS

```
set.seed(42)
model_rf <- caret::train(diag ~ .,
                         data = train_data,
                         method = "rf",
                         preProcess = c("scale", "center"),
                         trControl = control)
```

RESULTADOS BÁSICOS – RF – `caret`

```
Random Forest

490 samples
  9 predictor
  2 classes: 'benign', 'malignant'

Pre-processing: scaled (9), centered (9)
Resampling: Cross-Validated (10 fold, repeated 10 times)
Summary of sample sizes: 441, 441, 441, 441, 441, 441, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
  2     0.9653034  0.9239380
  5     0.9557107  0.9022051
  9     0.9516122  0.8929923

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 2.
```

- `mtry` - hiperparâmetro que é o número de variáveis que são candidatos a cada divisão da arvore
  - Sistema calcula o número ótimo baseado na precisão dos resultados

CALCULAR AS VARIÁVEIS IMPORTANTES

```r
imp <- model_rf$finalModel$importance # Guarda em unidades originais
importance <- varImp(model_rf, scale = TRUE) # Scale coloca em escala de 10
```

VARIÁVEIS IMPORTANTES – ESCALA ORIGINAL

- % das arvores em que a variável aparece

```
1 imp[order(imp, decreasing = TRUE), ]
```

```
    uniformity_of_cell_size    uniformity_of_cell_shape
                  39.473622                   35.949434
                bare_nuclei             bland_chromatin
                  35.733822                   30.663370
single_epithelial_cell_size             normal_nucleoli
                  22.501600                   22.420870
            clump_thickness           marginal_adhesion
                  17.265797                   12.082882
                    mitosis
                   3.293094
```

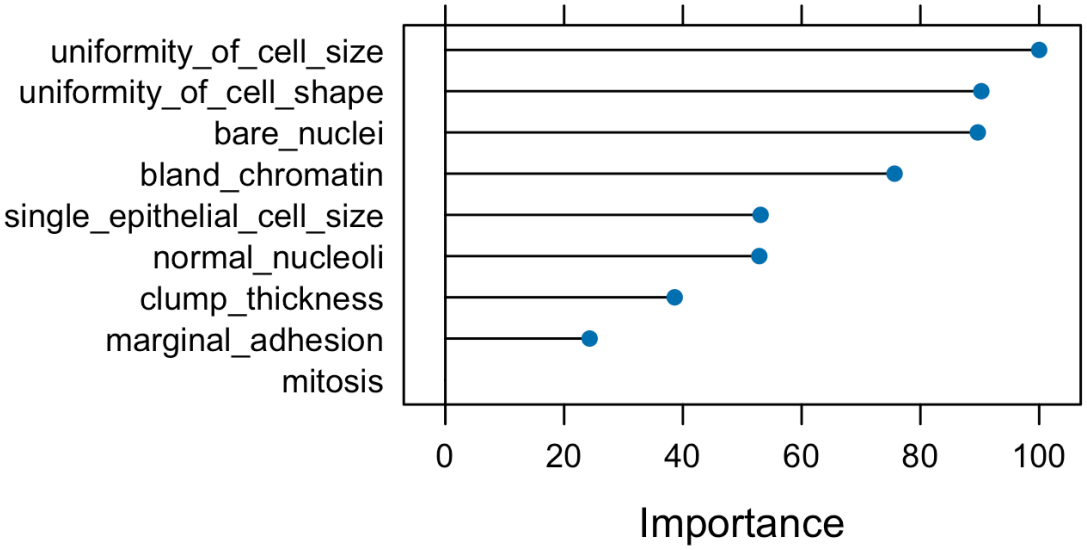
VARIÁVEIS IMPORTANTES - ESCALA 100 -> 0

```
1 importance
```

```
rf variable importance

                            Overall
uniformity_of_cell_size      100.00
uniformity_of_cell_shape      90.26
bare_nuclei                   89.66
bland_chromatin               75.65
single_epithelial_cell_size   53.09
normal_nucleoli               52.87
clump_thickness               38.62
marginal_adhesion             24.29
mitosis                        0.00
```

VARIÁVEIS IMPORTANTES – GRÁFICO

```
1 plot(importance)
```



PREVISÕES DO MODELO DE RF DE `caret`

```
predrfx <- predict(model_rf, test_data)
confusionMatrix(predrfx, test_data$diag)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  benign malignant
  benign       134         1
  malignant      3        71

               Accuracy : 0.9809
                 95% CI : (0.9517, 0.9948)
    No Information Rate : 0.6555
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9579

 Mcnemar's Test P-Value : 0.6171

            Sensitivity : 0.9781
            Specificity : 0.9861
         Pos Pred Value : 0.9926
         Neg Pred Value : 0.9595
             Prevalence : 0.6555
         Detection Rate : 0.6411
   Detection Prevalence : 0.6459
```

PREVISÕES NO FORMATO DE PROBABILIDADES

- `type = "prob"` de `predict() põe os valores em probabilidades
- Deixa você decidir qual seria o limite para diferenciar entre “benign” e “malignant”
  - Até agora, sempre foi 0.5

```
results <- tibble(actual = test_data$diag, predict = predict(model_rf, test_data, type = "prob"))
results$prediction <- ifelse(results$predict$benign > 0.5, "benign",
                             ifelse(results$predict$malignant > 0.5, "malignant", NA))
head(results %>% select(actual, prediction))
```

```
# A tibble: 6 × 2
  actual    prediction
  <fct>     <chr>
1 benign    benign
2 benign    benign
3 benign    benign
4 malignant malignant
5 benign    benign
6 malignant malignant
```

ACERTAMOS COM O NOVO MODELO?

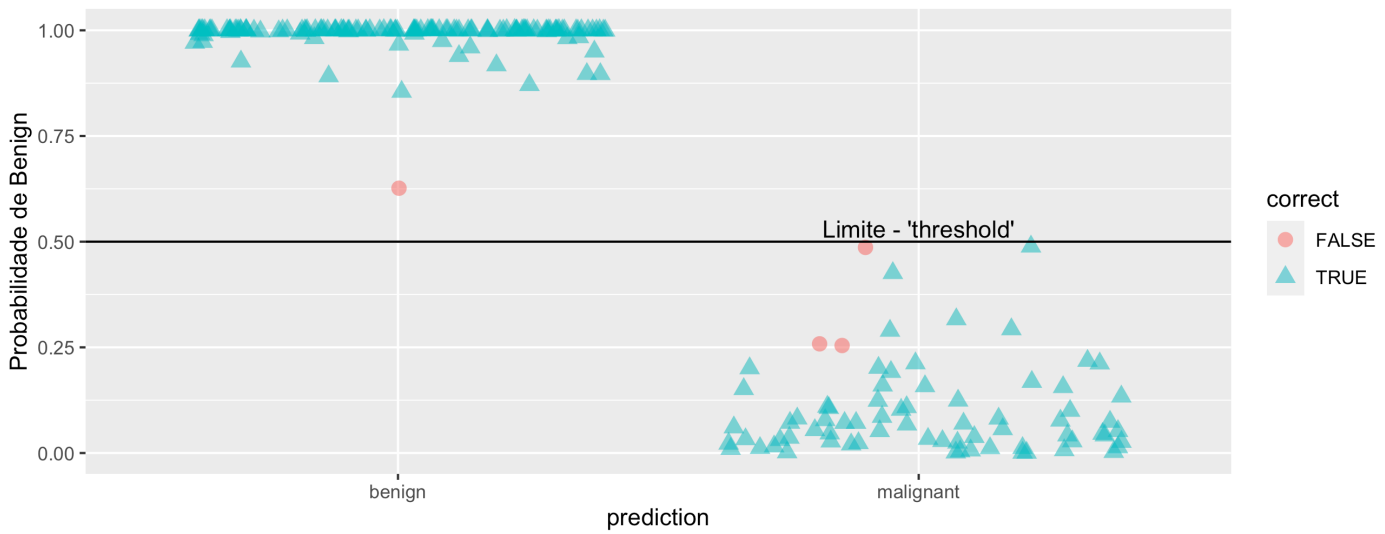
```
1 results$correct <- ifelse(results$actual == results$prediction, TRUE, FALSE
2 tabyl(results$correct) %>% adorn_pct_formatting()
```

```
results$correct   n percent
          FALSE   4    1.9%
           TRUE 205   98.1%
```

GRÁFICO DOS RESULTADOS

```
gr_rf <- ggplot(results, aes(x = prediction, y = predict$benign, color = co
  geom_jitter(size = 3, alpha = 0.6) +
  geom_hline(yintercept = 0.5) +
  ylab("Probabilidade de Benign") +
  annotate("text", x = 2, y = 0.53, label = "Limite - 'threshold'")
```

```
1 gr_rf
```


1.00
0.75
0.50
0.25
0.00
Probabilidade de Benign
Limite - 'threshold'
benign
malignant
prediction
correct
FALSE
TRUE

ESTE GRÁFICO MOSTRA

- Erro de “benign”
  - Perto a 0.50
- Erros de “malignant”
  - Mais espalhadas
  - Alguns com probabilidades bem perto a verdadeiro “malignant” (0.0)
- Mais confiança nas previsões de “benign”
- Parece que 0.5 é um bom “threshold” entre determinação de “benign” ou “malignant”
  - Discrimina bem

MUITO MAIS QUE PODEMOS FAZER

- Testar outros limites que 0.5
- Testar outros números das arvores
- ROC/AUC análise