ANÁLISE DOS DADOS COM R

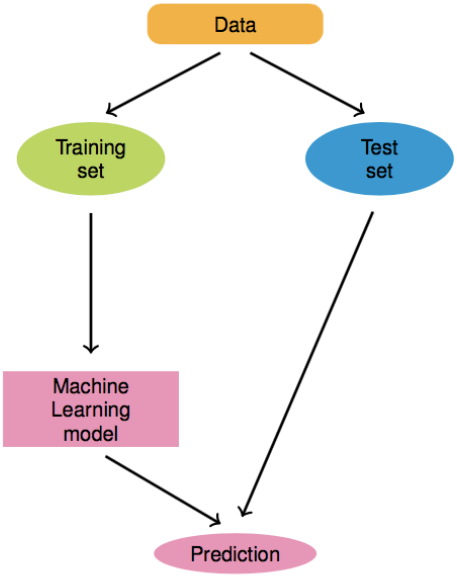
Ferramentas de Machine Learning

James R. Hunter, PhD
Retrovirologia, EPM, UNIFESP

2023-10-17

TIPOS DE *MACHINE LEARNING*

Supervised



Unsupervised

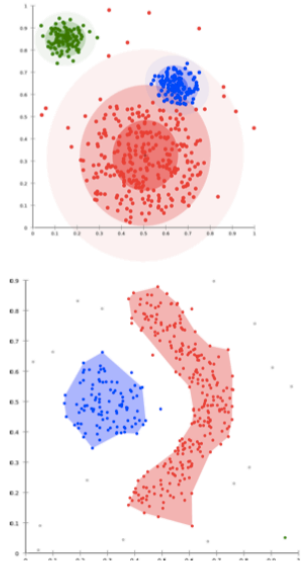


Image Source: Wikipedia

TREINAMENTO X TESTAGEM DOS MODELOS

- Divisão dos data frames em partes separadas
- Quer evitar *overfitting*
- **NUNCA, JAMAIS, USE OS MESMOS CASOS PARA TESTES QUE VOCÊ USOU PARA TREINAMENTO**

OVERFITTING

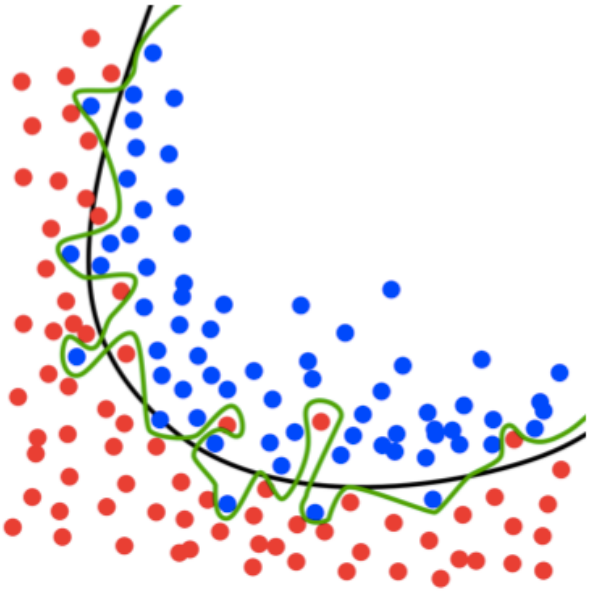


Image Source: Wikipedia

CARACTERÍSTICAS DOS MODELOS

- Covariáveis
  - Quantos são suficientes para construir um modelo
  - Número insuficiente – modelo não descreve suficiente a condição
  - Número demais – overfitting

FORTELECER O MODELO

- Bootstrapping
- k-fold Cross Validation
  - Tirar uma parte (fold) do grupo de treinamento
  - Treinar o modelo
  - Testar o modelo com os casos do treinamento
  - Faça o mesmo com as outras partes
  - Use como modelo final aquele que mostra melhor desempenho

MACHINE LEARNING EM MODELAGEM BIOLÓGICA/MÉDICA

- Tipicamente, projetos com “big data”
- Modelo pode fornecer informação rapidamente e corretamente
  - Médicos podem usar a informação para desenhar tratamentos ou diagnósticos
- Aplicação para medicina personalizada de precisão
- Exemplo:
  - Diagnostico de câncer de mama com ajuda de modelo informatizado

PODEMOS TER CONFIANÇA NOS MODELOS DE MACHINE LEARNING?

- Algoritmos de ML modelam interações de alto grau enter as variáveis
- Interpretação dos resultados de ML pode ser difícil
- A “caixa preta” dos algoritmos de ML escondem como eles fazem escolhas
  - Em alguns algoritmos (e.g. redes neurais)
- Assim, *precisamos modelos que significam algo* para os
  - Arquitetos
  - Usadores
- “Meaningful Models”

O QUE FAZ UM MODELO UM “MEANINGFUL MODEL”

- Poder generalizar baseado no modelo
- Responde à pergunta original
- … com suficiente precisão para ser confiável
- Grau de precisão depende do problema

COVARIÁVEIS – *FEATURES*

- As variáveis independentes
- Variáveis para treinar o modelo
- Selecionar as variáveis certas – **crucial**
- Mais features não necessariamente bom
  - Perigo de “overfitting”

VAMOS PÔR AS MÃOS NA MASSA

DADOS

- Continuar com os dados de `galton`
- Expandir a análise para incluir altura da mãe

```
1 glimpse(galton)
```

```
Rows: 898
Columns: 6
$ family <fct> 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5,
…
$ father <dbl> 78.5, 78.5, 78.5, 78.5, 75.5, 75.5, 75.5, 75.5, 75.0, 75.0,
75.…
$ mother <dbl> 67.0, 67.0, 67.0, 67.0, 66.5, 66.5, 66.5, 66.5, 64.0, 64.0,
64.…
$ sex    <fct> M, F, F, F, M, M, F, F, M, F, M, M, F, F, F, M, M, M, F, F, F,
…
$ height <dbl> 73.2, 69.2, 69.0, 69.0, 73.5, 72.5, 65.5, 65.5, 71.0, 68.0,
70.…
$ nkids  <int> 4, 4, 4, 4, 4, 4, 4, 4, 2, 2, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6,
…
```

MÉTODO NOVO QUE SEGUIMOS

- Método de *Machine Learning*
- Seguir a metodologia do pacote `caret`
- Passo 1
- Dividir os casos em 2 grupos: treinamento, testes
- Divisão aleatória
- Treinar o modelo com o grupo de treinamento
- Depois testar as previsões do modelo com os valores do grupo de testes
- Objetivo: fazer previsões corretas
  - Mais importante que a elegância do modelo

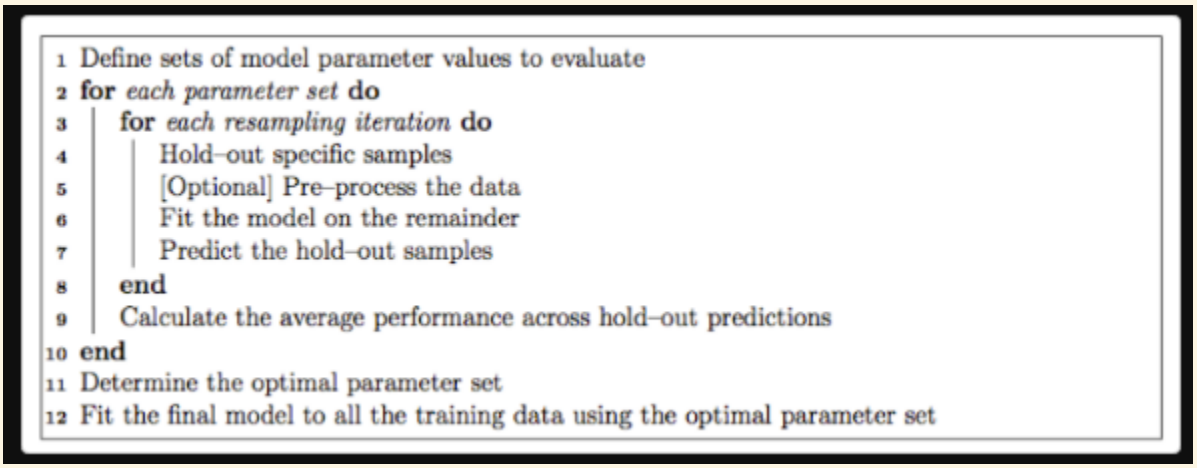
CARREGAR PACOTES NECESSÁRIOS PARA ESTE MÉTODO

- `caret` : *Classification And REgression Training*
- `ggpubr`: gráficos
- `broom` : funções para mostrar e comparar os modelos
- `nortest`: testes de normalidade estatística
- `janitor`: ajuda com tabelas

```
1 pacman::p_load(caret, ggpubr, broom, nortest, janitor)
```

PROCESSO DE `caret`

- Fornece um *workflow* eficiente para problemas de regressão e classificação
- Modelos construídos com a função `train`



```
Define sets of model parameter values to evaluate
for each parameter set do
    for each resampling iteration do
        Hold–out specific samples
        [Optional] Pre–process the data
        Fit the model on the remainder
        Predict the hold–out samples
    end
    Calculate the average performance across hold–out predictions
end
Determine the optimal parameter set
Fit the final model to all the training data using the optimal parameter set
```

`caret` DIVISÃO DOS DADOS

- função `createDataPartition()`
  - Dar para função a variável dependente `galton$height`
  - Proporção (`p`) que você quer na amostra de treinamento (70%)
    - Pode ser entre 50% e 70%
    - Mais pode causar *overfitting*
  - Função retorna os índices dos casos do conjunto de treinamento
  - Argumento `list = FALSE`

```r
set.seed(42)
indice <- createDataPartition(galton$height, p = 0.70, list = FALSE)
head(indice[, 1], 25)
```

```
 [1]  2  3  4  6  7  8  9 13 14 15 17 18 20 21 23 24 25 26 27 28 29 30 31 33
34
```

CRIAR `train_data` E `test_data`

- **VSS** lembre da virgula depois do `indice`
  - Por quê?
- Para `test_data`, você quer os dados que **NÃO** são de `train_data`
  - Assim, precisa usar o sinal de menos (**–**)

```
1 train_data <- galton[indice, ]
2 test_data <- galton[-indice, ]
```

VALIDAÇÃO CRUZADA (*CROSS-VALIDATION*)

- Validação do cálculo dos parâmetros do modelo utilizando pedaços dos casos cada repetição
- Evita necessidade de dividir o conjunto em 3 grupos (treinamento, validação, testes)
- Relacionado ao processo de *bootstrap* - reamostragem
- `caret` seleciona o modelo que tem o melhor desempenho

PROCESSO DE *K-FOLD* VALIDAÇÃO CRUZADA

- Dividir o grupo dos casos de treinamento em *k* subgrupos iguais
- Treinar o modelo com $k - 1$ dos folds
- Software testa este modelo com os casos do fold deixado fora e avalia desempenho (precisão)
- Repetir até tinha deixado fora todos os folds
- Pode repetir o processo inteiro um número das vezes

PRE-PROCESSAMENTO

- Se tiver traços das variáveis muito não normais
- Pode reduzir a não-normalidade das curvas com
  - Centralização (subtrair a média do valor)
  - Normalização (dividir valor centralizado por des. padrão)
- `caret` oferece essas opções

`train()` MODELO DAS ALTURAS

- `caret::train()` é a função que determina os parâmetros do modelo da regressão

```
fit_pai_mae <- caret::train(height ~ father + mother,
                            method = "lm",
                            data = train_data,
                            trControl = trainControl(method = "repeatedcv",
                                                     number = 5,
                                                     repeats = 10,
                                                     savePredictions = "none",
                                                     verboseIter = FALSE))
```

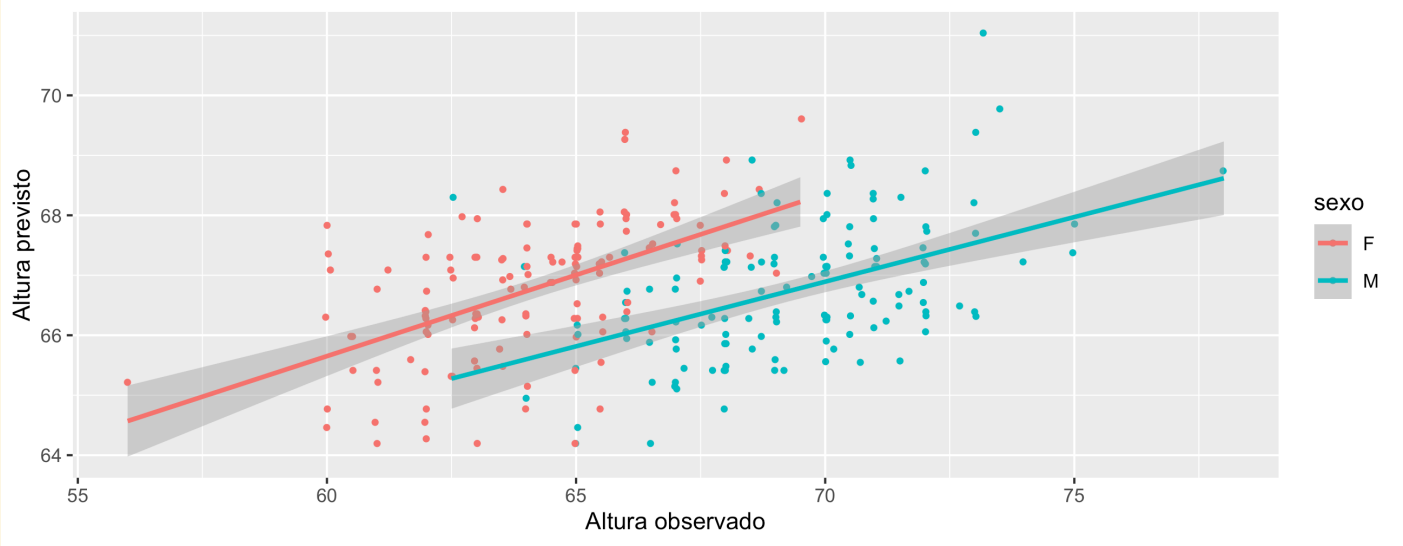
```r
summary(fit_pai_mae)
```

```
Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
   Min     1Q Median     3Q    Max
-8.480 -2.740 -0.179  2.807 11.699

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 23.59851    5.08952   4.637 4.31e-06 ***
father       0.37731    0.05589   6.751 3.34e-11 ***
mother       0.26601    0.05870   4.532 7.00e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

COMO FOI O DESEMPENHO DO MODELO?

- Aplicar o modelo aos dados do conjunto de `test_data`
  - Até agora, o modelo não tinha visto esses dados
  - Indica o que pode fazer com qualquer dados que mede a mesma fenômeno
  - `predict` calcula os valores previstos usando os parâmetros do modelo

```r
# previsões
prv <- predict(fit_pai_mae, test_data)
# comparar para preços observados
gg_pai_mae_1 <- data.frame(obs = test_data$height,
                           previs = prv,
                           sexo = test_data$sex) %>%
  ggplot(aes(x = obs, y = previs, color = sexo)) +
    geom_jitter(shape = 20) +
    geom_smooth(method = "lm") +
    labs(x = "Altura observado", y = "Altura previsto")
```


Altura previsto
70
68
66
64
55
60
65
70
75
Altura observado
sexo
F
M

QUANTA PRECISÃO TEVE O MODELO?

- Olhar a diferença entre os valores verdadeiros (observados) e os valores previstos pelo modelo
- Quantas dessas diferenças foram menores que um padrão razoável (? 2 polegadas)

```
pred <- predict(fit_pai_mae, test_data)
res <- tibble(pred = pred,
                  obs = test_data$height,
                  dif = obs - pred)
padrao_in <- 2
# teste de bom, ruim
res <- res %>%
  mutate(bomruim = ifelse(abs(dif) <= padrao_in, "bom", "ruim"))
tabyl(res$bomruim) %>% adorn_pct_formatting()
```

```
res$bomruim   n percent
        bom  95   35.6%
       ruim 172   64.4%
```

MODELO NÃO É BOM

- Precisão muito baixo
  - 36% dentro do padrão de 2 polegadas
- R^2 muito baixo (0.1023)
  - Só 10% da variância no modelo explicada pelas variáveis

PODEMOS FAZER MELHOR?

- Gênero pode ter um efeito
- Gênero é uma variável catégorica
- Regressão compara as distribuições dos números
- Mas pode incluir variáveis categóricas

INCLUSÃO DAS VARIÁVEIS CATEGÓRICAS EM REGRESSÃO

- Dividir a variável em "*dummy*" variáveis
  - 1 variável *dummy* para cada nível da variável categórica menos o primeiro nível
  - Se tiver 3 níveis (`alto`, `medio`, `baixo`), o sistema criaria 2 novas variáveis
    - `medio` e `baixo`
    - `alto` seria um valor de referência que representa o caso quando nenhum dos outros níveis está presente

```r
notas <- tibble(x = rep(c("alto", "media", "baixo"), 3),
                y = c(3, 2, 1, 3, 2, 1, 7, 5, 2))
summary(lm(y ~ x, data = notas))
```

```
Call:
lm(formula = y ~ x, data = notas)

Residuals:
    Min      1Q  Median      3Q     Max
-1.3333 -1.0000 -0.3333  0.6667  2.6667

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.3333     0.9813   4.416  0.00449 **
xbaixo       -3.0000     1.3878  -2.162  0.07390 .
xmedia       -1.3333     1.3878  -0.961  0.37377
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

INCLUIR `sex` NA REGRESSÃO

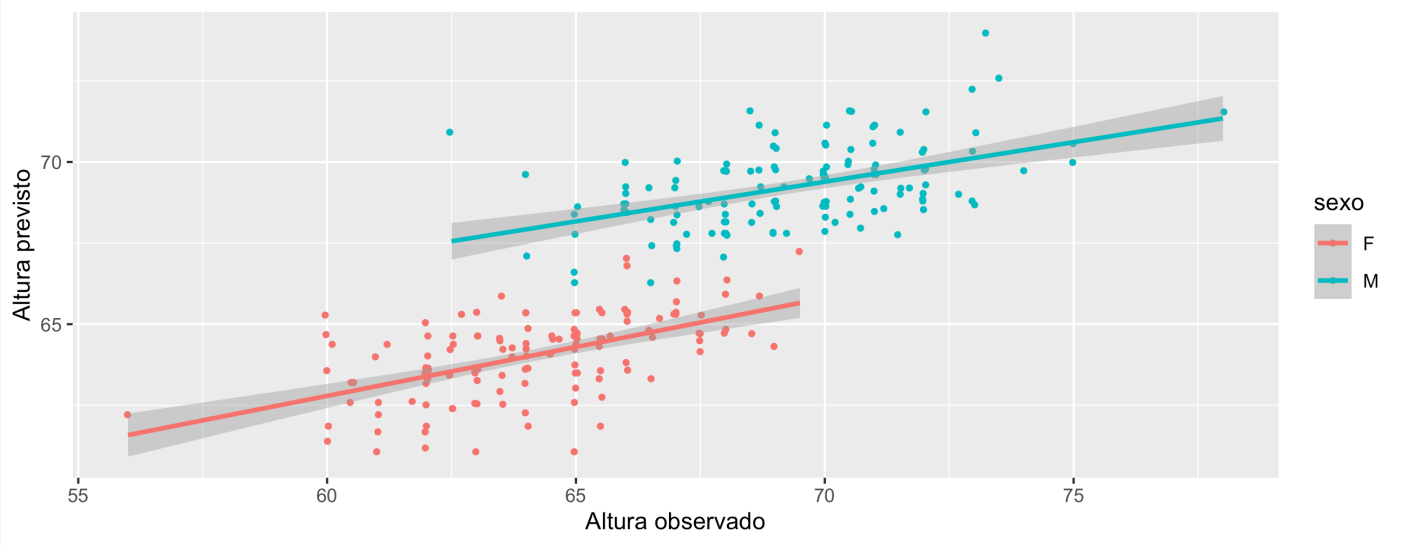
```
fit_pms <- caret::train(height ~ father + mother + sex,
                        method = "lm",
                        data = train_data,
                        trControl = trainControl(method = "repeatedcv",
                                                 number = 5,
                                                 repeats = 10,
                                                 savePredictions = "none",
                                                 verboseIter = FALSE))
```

```r
summary(fit_pms)
```

```
Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-9.4833 -1.5274  0.0932  1.5369  9.1510

Coefficients:
            Estimate Std. Error t value   Pr(>|t|)
(Intercept) 15.05115    3.29308   4.571 0.00000586 ***
father       0.40976    0.03604  11.369    < 2e-16 ***
mother       0.32157    0.03788   8.489    < 2e-16 ***
sexM         5.21288    0.17527  29.742    < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 ' ' 0.1 ' ' 1
```

DESEMPENHO DO MODELO

```r
# previsões
prv <- predict(fit_pms, test_data)
# comparar para preços observados
gg_pms_1 <- data.frame(obs = test_data$height,
                       previs = prv,
                       sexo = test_data$sex) %>%
  ggplot(aes(x = obs, y = previs, color = sexo)) +
    geom_jitter(shape = 20) +
    geom_smooth(method = "lm") +
    labs(x = "Altura observado", y = "Altura previsto")
```


Altura previsto
70
65
55
60
65
70
75
Altura observado
sexo
F
M

QUANTA PRECISÃO TEVE O MODELO?

```r
pred <- predict(fit_pms, test_data)
res_pms <- tibble(pred = pred,
                  obs = test_data$height,
                  dif = obs - pred)
padrao_in <- 2
# teste de bom, ruim
res_pms <- res_pms %>%
  mutate(bomruim = ifelse(abs(dif) <= padrao_in, "bom", "ruim"))
tabyl(res_pms$bomruim) %>% adorn_pct_formatting()
```

```
res_pms$bomruim   n percent
            bom 183   68.5%
           ruim  84   31.5%
```

RESULTADO

- Modelo consegue prever 69% das alturas dentro da padrão
  - Dobro do modelo anterior
- R^2 aumentou a 0.627 (muito)
- Gênero tem um papel importante na determinação das alturas das crianças
  - O modelo inclui esta caraterística

`varImp()` FUNÇÃO EM `caret`

- Função avalia a importância relativa das variáveis no modelo
- Mais importante - 100%
- Menos importante - 0%
- Nosso modelo 2

```
1 varImp(fit_pms)
```

```
lm variable importance

       Overall
sexM    100.00
father   13.55
mother    0.00
```

EXEMPLO FINAL - `gapminder`

- Pacote R derivado do site https://www.gapminder.org/
- Monitora condições socio-economicas no mundo
- Fruto das pesquisas do Hans Rosling e família
- Eles acham que pobreza no mundo pode ser eliminada por 2030
- Assiste o video: https://www.gapminder.org/videos/dont-panic-end-poverty/
- Empolgante!