Bootstrap Aggregation (Bagging)

Paulo C. Marques F. e Hedibert F. Lopes

Sexta-feira, 24 de Novembro de 2017

Insper

Função de distribuição empírica (1)

Insper

Suponha que $X_1, \ldots, X_n$ são variáveis aleatórias independentes e identicamente distribuídas tais que $X_1 \sim F$.

Se em uma realização desta *amostra aleatória* observamos os valores $x_1, \ldots, x_n$, a *função de distribuição empírica* $\hat{F}_n$ desta realização é uma função degrau, continua à direita, que salta $1/n$ em cada ponto observado.

$$\hat{F}_n(t) = \frac{1}{n} \sum_{i=1}^{n} I_{(-\infty, t]}(x_i).$$

Ou seja, $\hat{F}_n$ define uma distribuição discreta que coloca massa $1/n$ em cada ponto amostral observado.

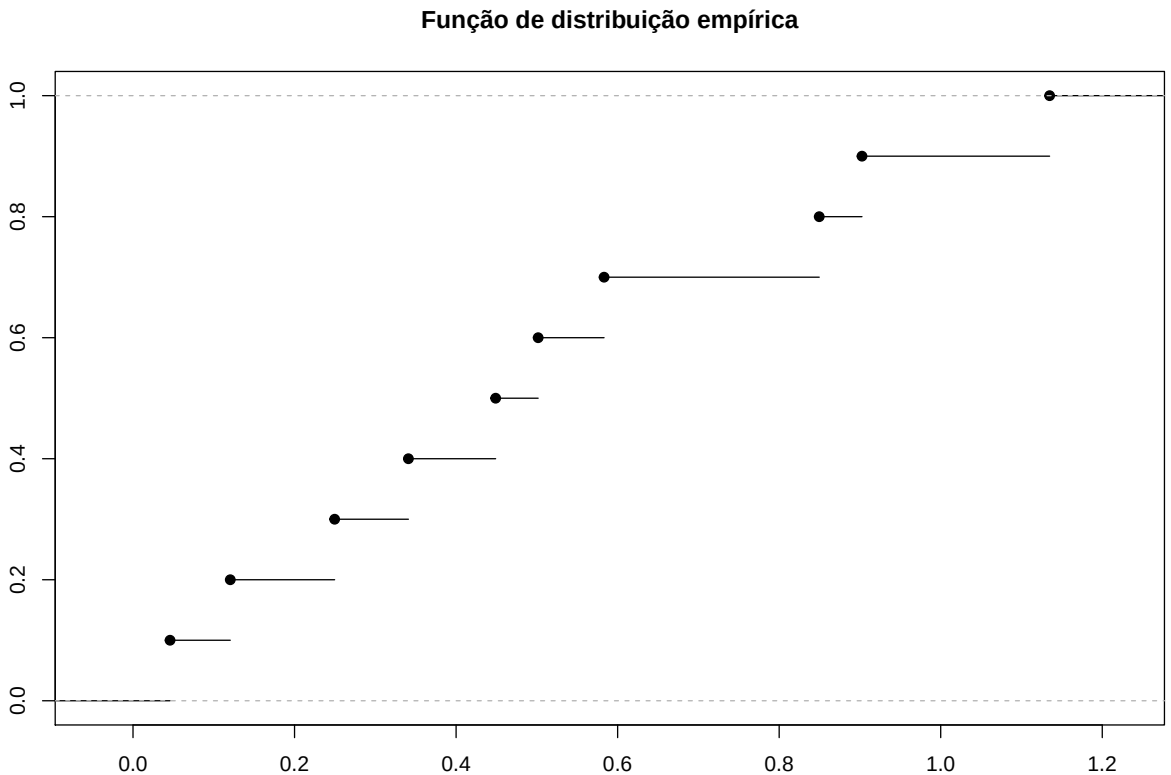
Função de distribuição empírica (2)

Insper

```
set.seed(4321)
n <- 10; alpha <- 2; beta <- 3
(x <- rgamma(10, shape = alpha, rate = beta))
```

```
##  [1] 0.34095385 0.04584761 0.90252132 0.44897133 0.12042893 0.58322005
##  [7] 1.13482011 0.24963663 0.50159632 0.84959536
```

```
plot(ecdf(x), main = "Função de distribuição empírica", xlab = "", ylab = "")
```



Função de distribuição empírica (3)

Insper

Antes de termos uma realização da amostra aleatória em mãos, podemos olhar a função de distribuição empírica, para cada t, como uma variável aleatória:

$$\hat{F}_n(t, \omega) = \frac{1}{n} \sum_{i=1}^{n} I_{(-\infty, t]}(X_i(\omega)).$$

Pela Lei Forte dos Grandes Números, temos para todo $t \in \mathbb{R}$ que

$$\frac{1}{n} \sum_{i=1}^{n} I_{(-\infty, t]}(X_i) \underset{n \to \infty}{\longrightarrow} \mathrm{E}[I_{(-\infty, t]}(X_1)] = \Pr\{X_1 \leq t\} = F(t),$$

quase certamente.

Função de distribuição empírica (3)

Insper

Antes de termos uma realização da amostra aleatória em mãos, podemos olhar a função de distribuição empírica, para cada t, como uma variável aleatória:

$$\hat{F}_n(t, \omega) = \frac{1}{n} \sum_{i=1}^{n} I_{(-\infty, t]}(X_i(\omega)).$$

Pela Lei Forte dos Grandes Números, temos para todo $t \in \mathbb{R}$ que

$$\frac{1}{n} \sum_{i=1}^{n} I_{(-\infty, t]}(X_i) \underset{n \to \infty}{\longrightarrow} \mathrm{E}[I_{(-\infty, t]}(X_1)] = \Pr\{X_1 \leq t\} = F(t),$$

quase certamente.

De fato, o resultado é ainda mais forte, pois Glivenko e Cantelli provaram no começo dos anos 1930 a convergência uniforme:

$$\sup_{t \in \mathbb{R}} |\hat{F}_n(t) - F(t)| \underset{n \to \infty}{\longrightarrow} 0,$$

quase certamente (demonstração no Billingsley).

Bootstrap (1)

Estes resultados de convergência inspiraram o método *bootstrap*, proposto por Efron no final dos anos 1970.

O *bootstrap* é um método de reamostragem, utilizado extensamente em Estatística Clássica e em Aprendizagem Estatística.

Para nós interessa a discussão do *bootstrap não paramétrico*. Denote a amostra aleatória da função de distribuição F por $X = (X_1, \ldots, X_n)$.

O princípio básico do *bootstrap* pode ser ilustrado com o seguinte exemplo.

Denote por θ a mediana de F e por M a mediana amostral obtida a partir de X:

$$M = \text{Med}\{X_1, \ldots, X_n\}.$$

Definindo $g(X, F) = (M - \theta)^2$, suponha que queremos utilizar M para estimar θ, e que desejamos conhecer, por exemplo, o erro quadrático médio $\text{E}[g(X, F)]$.

Bootstrap (2)

Insper

A ideia do *bootstrap* é aproximar $\mathrm{E}[g(X, F)]$ por $\mathrm{E}[g(X^*, \hat{F}_n)]$, em que X^* é uma amostra da função de distribuição empírica $\hat{F}_n$. Em geral, não conseguimos calcular $\mathrm{E}[g(X^*, \hat{F}_n)]$ diretamente, mas é fácil obter uma aproximação de Monte Carlo.

1. A partir da amostra observada $x_1, \ldots, x_n$ calculamos a função de distribuição empírica $\hat{F}_n$.
2. A partir de $\hat{F}_n$, geramos *amostras bootstrap*

$$x^{*(i)} = (x_1^{*(i)}, \ldots, x_n^{*(i)}),$$

   para $i = 1, \ldots, B$. Ou seja, cada amostra *bootstrap* é obtida simplesmente amostrando $\{x_1, \ldots, x_n\}$ uniformemente e com reposição n vezes.
3. Para cada amostra *bootstrap* calculamos $T^{(i)} = g(x^{*(i)}, \hat{F}_n)$.
4. A aproximação de Monte Carlo é dada por $\frac{1}{B}\sum_{i=1}^{B} T^{(i)}$.

Bootstrap (3)

Insper

```r
# sampling from the true distribution
N <- 10^4
true_median <- qgamma(0.5, shape = alpha, rate = beta)
mc_sample <- rgamma(N*n,  shape = alpha, rate = beta)
x_mc <- matrix(mc_sample, nrow = N, ncol = n)
eqm <- apply(x_mc, 1, function(y) (median(y) - true_median)^2)
approx_true_eqm <- mean(eqm)

# bootstrap sampling from observed x
B <- 10^4
x_bs <- matrix(nrow = B, ncol = n)
for (i in 1:B) x_bs[i,] <- sample(x, size = n, replace = TRUE)
theta_hat <- median(x)
T <- apply(x_bs, 1, function(y) (median(y) - theta_hat)^2)
bootstrap_estimate <- mean(T)

(sprintf("True EQM     = %.4f (approximately)", approx_true_eqm))
```

```
## [1] "True EQM     = 0.0269 (approximately)"
```

```r
(sprintf("Boostrap EQM = %.4f", bootstrap_estimate))
```

```
## [1] "Boostrap EQM = 0.0221"
```

Bootstrap (4)

Tradicionalmente, o *bootstrap* é utilizado pelos estatísticos clássicos para obter uma estimativa do erro padrão de um estimador complicado, para o qual não se conhece a expressão de um intervalo de confiança.

Há uma versão bayesiana do *bootstrap* proposta por Donald Rubin no começo dos anos 1980 (*Bayesian bootstrap*).

O *bootstrap* foi estendido para situações em que temos dependência, como o caso de uma série temporal estacionária (*block bootstrap*).

O uso do *bootstrap* em Aprendizagem Estatística se distancia muito do propósito original do método. De fato, para um usuário tradicional do *bootstrap*, tal uso revela-se surpreendente.

Em Aprendizagem Estatística o *bootstrap* é uma ferramenta fundamental que permite melhorar a performance preditiva de métodos de aprendizagem, tais como as árvores de classificação e regressão.

Árvores de classificação e regressão (1)

Insper

Vimos em aulas anteriores como crescer árvores de classificação e regressão.

Especulando sobre o aspecto cognitivo, há quem diga que as árvores resultantes refletem melhor a maneira de pensar de um ser humano do que, digamos, um modelo de regressão linear múltipla.

As árvores resultantes do processo de crescimento podem ser exibidas graficamente, e a regra decisão correspondente é interpretável até mesmo por um não especialista (se tivermos uma árvore não muito alta).

Em uma árvore, variáveis preditoras qualitativas são tratadas diretamente (as divisões são feitas considerando-se a separação dos possíveis valores da preditora em dois conjuntos disjuntos) sem a necessidade de se criar variáveis *dummy*.

Árvores de classificação e regressão (2)

Insper

Em geral, as árvores de classificação e regressão não têm a mesma performance preditiva de outros métodos, como as máquinas de vetores de suporte "kernelizadas".

Ademais, árvores não são muito robustas: uma pequena mudança nos dados de treinamento leva a uma grande mudança na árvore crescida.

Ou seja, nos termos das nossas aulas de fundamentos, algoritmos como o CART sofrem por ter grande variância.

Agregando-se muitas árvores de decisão com métodos como o *bagging* (agregação *bootstrap*), a performance preditiva das árvores pode ser melhorada substancialmente

Bagging (1)

A agregação *bootstrap*, ou *bagging*, foi proposta por Leo Breiman em 1996.

Um dos autores do CART, Breiman foi uma figura central em Aprendizagem Estatística, que intencionalmente se posicionou no imbricamento da maneira mais tradicional e probabilística de se fazer inferência (ao estilo dos estatísticos) e da visão mais alinhada à ciência da computação (orientada a algoritmos).

O *bagging* é um procedimento geral, não restrito apenas ao contexto de árvores de classificação e regressão, cujo propósito é reduzir a variância de um método de aprendizagem.

Uma maneira natural de se reduzir a variância de um método de aprendizagem seria tomar muitos conjuntos de treinamento e com cada um deles treinar regressores, cujas respostas seriam tomadas em média como sendo a resposta resultante do conjunto (*ensemble*) de regressores (daí o termo "método de *ensemble*").

Este caminho não é viável, pois em geral não temos acesso a múltiplos conjuntos de dados de treinamento.

Bagging (2)

A ideia fundamental do *bagging* é gerar amostras *bootstrap* a partir do conjunto original de dados de treinamento.

No *bagging*, em um problema de regressão, geramos B amostras *bootstrap* que farão o papel de dados de treinamento em cada regressor treinado.

Assim, treinamos os regressores $\hat{\psi}_1, \ldots, \hat{\psi}_B$ e o regressor agregado é a média

$$\hat{\psi}_{\text{bag}}(x) = \frac{1}{B} \sum_{i=1}^{B} \hat{\psi}_i(x).$$

Quando os regressores são treinados via algoritmo CART, não podamos as árvores resultantes. Crescemos árvores bem altas, que, individualmente, terão grande variância e viés baixo. O processo de *bagging* cuida da redução da variância.

Bagging (3)

Quando temos árvores de classificação, o *bagging* funciona de maneira análoga.

A diferença é que o classificador agregado $\hat{\varphi}_{\text{bag}}$ prevê segundo o "voto da maioria" dos classificadores $\hat{\varphi}_1, \ldots, \hat{\varphi}_B$ treinados a partir das B amostras *bootstrap*.

O *bagging* produz grandes ganhos de performance preditiva quando agregamos milhares de árvores, formando um único regressor ou classificador.

Erro *Out-of-Bag* (1)

Que tal uma estimativa grátis do erro de teste esperado?

Uma característica útil e curiosa do *bagging* é que há uma forma simples e barata de se estimar o erro de teste do regressor (ou classificador) agregado sem precisarmos recorrer a procedimentos de validação cruzada.

É possível mostrar que, em média, cada árvore crescida durante o procedimento do *bagging* utiliza aproximadamente dois terços [1] dos dados originais de treinamento.

Os dados do um terço remanescente de uma determinada árvore do *ensemble* são denominados observações *out-of-bag* (fora-da-sacola) da árvore em questão.

[1] Veja a demonstração no último slide.

Erro *Out-of-Bag* (2)

Podemos prever a resposta do i-ésimo dado de treinamento utilizando todas as árvores do *ensemble* nas quais este dado pertence às observações *out-of-bag*.

Para tanto, tomamos a média das respostas previstas para este dado (em um problema de regressão), ou o voto da maioria (em um problema de classificação), das árvores do *ensemble* em que o i-ésimo dado ficou fora da sacola.

Fazendo isto para cada um dos dados de treinamento obtemos uma estimativa válida do erro de teste esperado.

Quando temos muitos dados de treinamento este procedimento é vantajoso, pois a validação cruzada nestes casos seria extremamente onerosa.

Dados de *spam* (1)

Insper

Retomando os dados de *spam* que analisamos na aula de CART, podemos comparar a performance preditiva de uma única árvore de classificação com a performance de um *ensemble* de 1.000 árvores, construído via *bagging*.

```
db <- read.table("./spambase/spambase.data", sep = ",", header = FALSE)

colnames(db) <- c(
    # 1
    "make", "address", "all", "w_3d", "our", "over", "remove", "internet", "order", "mail",
    "receive", "will", "people", "report", "addresses", "free", "business", "email", "you",
    "credit", "your", "font", "w_000", "money", "hp", "hpl", "george", "w_650", "lab", "labs",
    "telnet", "w_857", "data", "w_415", "w_85", "technology", "w_1999", "parts", "pm", "direct",
    "cs", "meeting", "original", "project", "re", "edu", "table", "conference",
    # 2
    "ch_semi", "ch_lparen", "ch_lbrack", "ch_bang", "ch_dollar", "ch_hash",
    # 3
    "CAPAVE", "CAPMAX", "CAPTOT",
    # Spam = 1, Ham = 0
    "Class"
)

db$Class <- as.factor(ifelse(db$Class, "Spam", "Ham"))
```

Na aula de CART, com apenas uma árvore, conseguimos uma taxa de erro de classificação de 10,7%.

Dados de *spam* (2)

Insper

Para construir o classificador via *bagging* utilizamos a bilioteca `randomForest` do `R`, pois, como veremos na aula de amanhã, o classificador obtido via *bagging* é um caso particular de uma *Random Forest*.

```
library(randomForest)

set.seed(1234)

bag <- randomForest(Class ~ . , data = db, mtry = 57, ntree = 1000)

print(bag)
```

```
##
## Call:
##  randomForest(formula = Class ~ ., data = db, mtry = 57, ntree = 1000)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 57
##
##         OOB estimate of  error rate: 5.28%
## Confusion matrix:
##       Ham Spam class.error
## Ham  2684  104  0.03730273
## Spam  139 1674  0.07666851
```

Portanto, o classificador obtido via *bagging* apresenta (via estimativa *out-of-bag*) uma taxa de erro de classificação de 5,28%, menos da metade do que havíamos conseguido com apenas uma árvore.

Demonstração

Suponha que temos dados de treinamento $x_1, \ldots, x_n$, em que os x_i's têm valores distintos, e que geramos uma amostra *bootstrap* $x_1^*, \ldots, x_n^*$. Sem perda de generalidade, considere o valor x_1. A probabilidade de x_1 não ter sido o primeiro valor x_1^* incluído na amostra *bootstrap* é igual a $1-1/n$. Portanto, por independência, a probabilidade de x_1 não ter sido incluído na amostra *bootstrap* nenhuma vez é $(1-1/n)^n =: q_n$. Uma vez que $\lim_{n\to\infty}(1+t/n)^n = e^t$, temos que $\lim_{n\to\infty} q_n = e^{-1} \approx 36{,}8\%$. Assim, sendo n suficientemente grande, a probabilidade de x_1 ter sido incluído pelo menos uma vez na amostra *bootstrap* é aproximadamente 63,2% (que é o tal “aproximadamente dois terços”). Defina variáveis indicadoras U_i, tais que $U_i = 1$ se x_i foi incluído pelo menos uma vez na amostra *bootstrap*; e $U_i = 0$ caso contrário. O número de dados de treinamento incluídos pelo menos uma vez na amostra *bootstrap* é $N = \sum_{i=1}^n U_i$. Segue da linearidade da esperança que $\mathrm{E}[N] \approx n \times 63{,}2\%$.