Aula 3: Random Forests

Paulo C. Marques F.

Aula ministrada no Insper

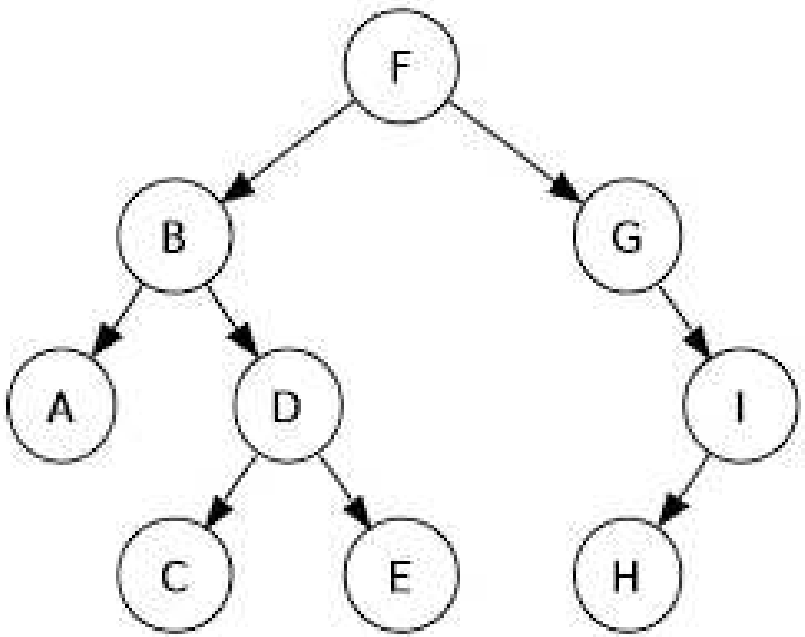
26 de Fevereiro de 2016

Árvores de classificação

- Estamos no mesmo contexto de aprendizagem supervisionada da primeira aula.
- Temos um vetor de $p \geq 1$ preditoras $X = (X_1, \ldots, X_p) \in \mathbb{R}^p$ (por exemplo, peso e maior comprimento);
- E uma resposta $Y \in \{1, \ldots, c\}$ (por exemplo, salmões e robalos).
- Uma diferença importante é que as árvores de classificação não dependem da escolha de uma métrica no espaço das preditoras.
- Precisamos apenas que seja possível ordenar cada variável preditora.
- As árvores de classificação fazem parte dos assim denominados "métodos não métricos de classificação".

Terminologia

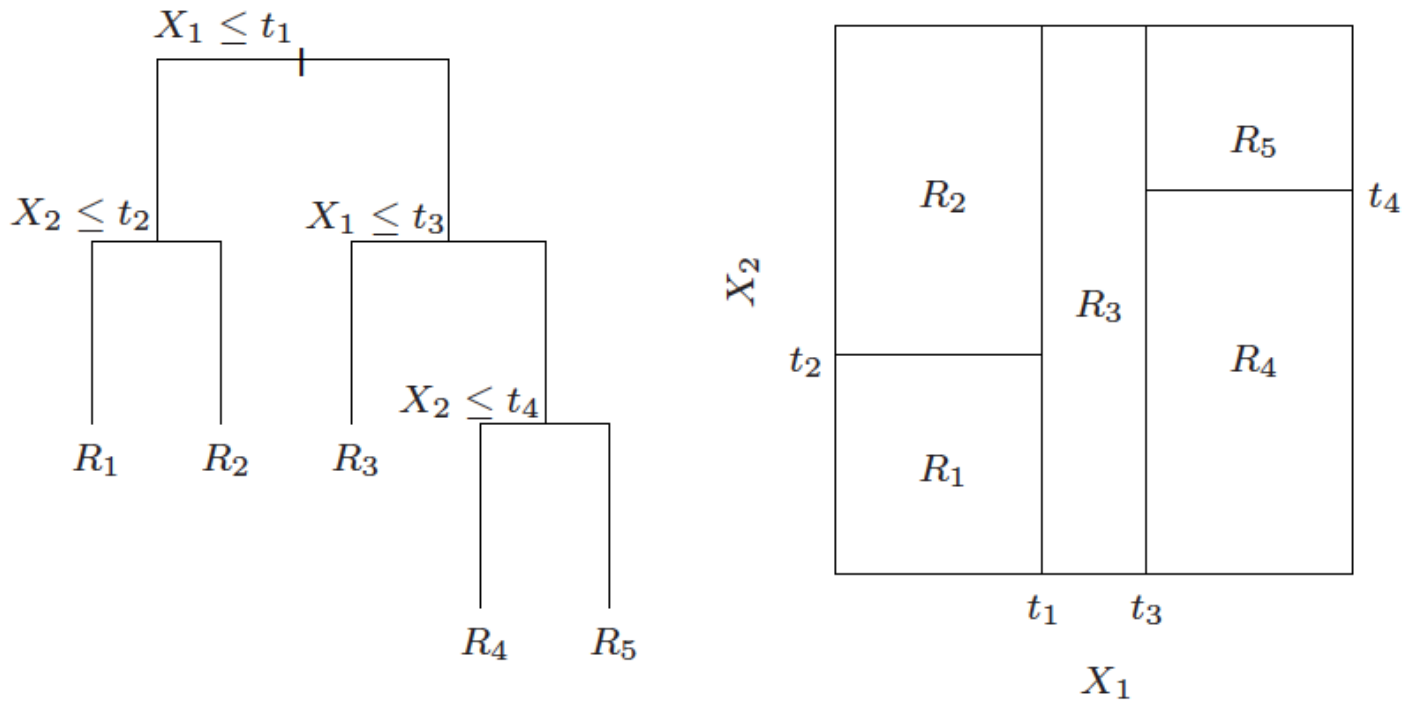
Na *árvore binária* abaixo, o nó F é a *raiz*, e os nós A, C, E e H são as *folhas* (ou *nós terminais*).



A árvore tem quatro níveis de *altura*. A *profundidade* do *nó interno* D é igual a dois.

Regiões determinadas pela árvore de classificação

Na figura abaixo (página 306 do ESL) temos uma árvore de classificação T e a partição do espaço de preditoras nas regiões correspondentes.



Cada nó não terminal de T define um "split" (divisão) em uma das preditoras. Cada folha de T corresponde a uma região retangular R_j.

Como classificar?

- Suponha que a árvore de classificação T do slide anterior nos foi dada *ex machina*.
- Suponha que temos dados de treinamento

$$(x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^2 \times \{0, 1\},$$

  para $i = 1, \ldots, n$.
- Tendo em mãos um novo $x_{n+1} \in \mathbb{R}^2$, inicialmente determinamos a qual região R_j pertence este vetor de preditoras x_{n+1}.
- Note que não precisamos examinar todas as regiões: basta descer a árvore a partir da raiz para saber a qual região x_{n+1} pertence!
- Isso é uma tremenda vantagem do ponto de vista computacional.
- Uma vez determinada a região R_j a qual x_{n+1} pertence, classificamos este exemplar como sendo da classe mais frequente entre os dados de treinamento pertencentes à mesma região R_j (voto da maioria).

Formalizando o classificador

- Dados de treinamento $(x_i, y_i) \in \mathbb{R}^p \times \{1, \ldots, c\}$, para $i = 1, \ldots, n$.
- A árvore de classificação T define as regiões retangulares $R_1, \ldots, R_m$ que particionam o espaço das preditoras $\mathbb{R}^p$.
- Seja $n_j = \sum_{i=1}^{n} I_{R_j}(x_i)$ o número de preditoras dos dados de treinamento pertencentes à região R_j, para $j = 1, \ldots, m$.
- A fração de exemplos da classe k na região R_j é igual a

$$\hat{p}_k(R_j) = \frac{1}{n_j} \sum_{\{i\,:\,x_i \in R_j\}} I_{\{k\}}(y_i),$$

  para $k = 1, \ldots, c$.
- A classe predita para a região R_j é $c_j = \arg\max_k \hat{p}_k(R_j)$, que é a proporção de exemplos (dados de treinamento) na região R_j que são da classe predominante.
- Classificador: $\varphi(x_{n+1}) = \sum_{j=1}^{n} c_j \, I_{R_j}(x_{n+1})$.

Como construir uma árvore de classificação?

- Como escolher cada uma das divisões ("splits")?
- Lembrando que em cada divisão precisamos escolher uma das preditoras e o ponto de separação.
- Que altura deve ter a árvore de classificação?
- Note que se a árvore for binária e balanceada, então em cada nível temos aproximadamente $2^{\text{nível}-1}$ nós e para cada um deles podemos escolher uma das p preditoras para fazer a divisão.
- Que algoritmo utilizar?

Algoritmo CART (1)

- CART: Classification and Regression Trees. Breiman el al. (1984).
- O algoritmo CART começa na raiz da árvore e efetua uma divisão, criando dois nós no próximo nível da árvore.
- Depois disso, descemos para o primeiro nível da árvore e repetimos o procedimento para os dois nós que foram criados.
- Continuamos da mesma maneira nos níveis seguintes.
- Em cada etapa, escolhemos a divisão que produz a maior queda no erro de classificação.
- O algoritmo CART cresce uma árvore alta e poda alguns dos seus ramos no final do processo.

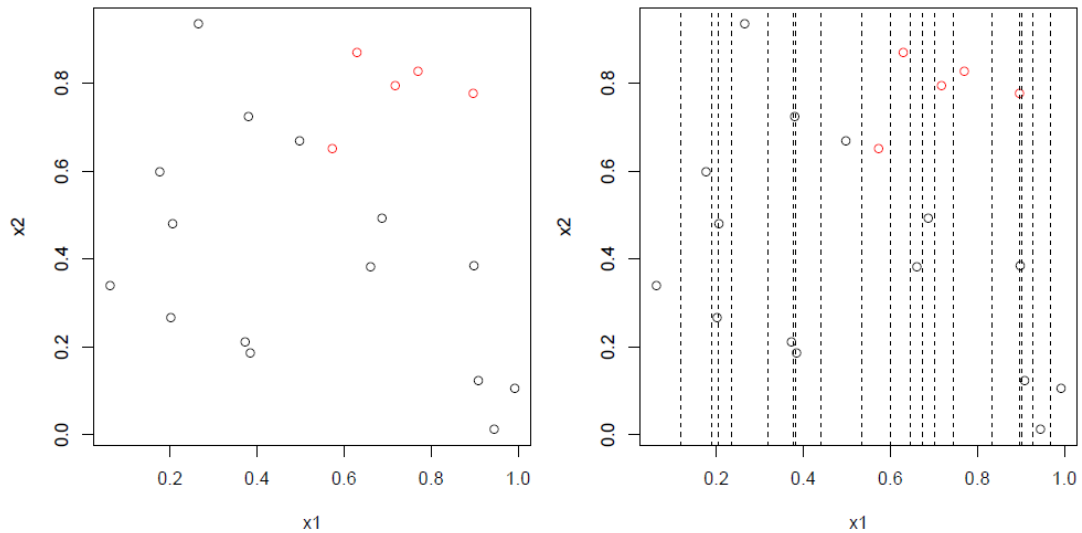
Algoritmo CART (2)

- Formalmente, o algoritmo CART começa na raiz da árvore e define as regiões disjuntas

$$R_1 = \{X \in \mathbb{R}^p : X_j \leq t\} \quad \text{e} \quad R_2 = \{X \in \mathbb{R}^p : X_j > t\}.$$

- Utilizando os dados de treinamento, fazemos a divisão escolhendo $\hat{j}$ e $\hat{t}$ tais que

$$(\hat{j}, \hat{t}) = \arg\min_{(j,t)} \left((1 - \hat{p}_{c_1}(R_1)) + (1 - \hat{p}_{c_2}(R_2)) \right).$$



Algoritmo CART (3)

- Procedemos de maneira análoga para os novos nós criados, até atingirmos algum critério de parada; por exemplo, quando tivermos apenas dados de treinamento de uma certa classe na nova região gerada.
- Este procedimento gera uma árvore T_0 que será podada: algumas de suas folhas serão colapsadas aos seus nós pais.
- Para uma árvore de classificação T, denote por $|T|$ o número de suas folhas e, para $\alpha \geq 0$, defina

$$C_\alpha(T) = \sum_{j=1}^{|T|}(1 - \hat{p}_{c_j}(R_j)) + \alpha|T|.$$

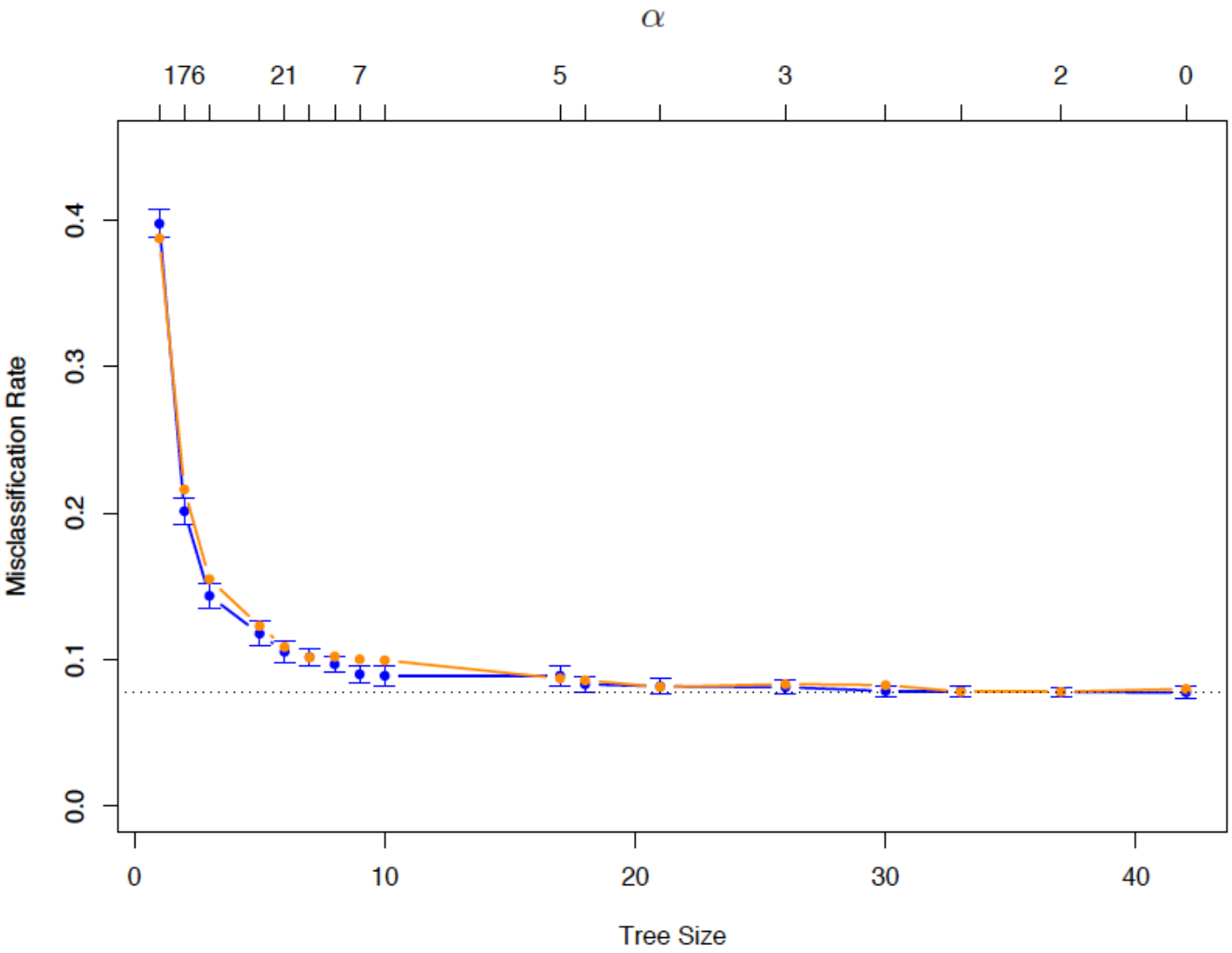
- O algoritmo CART escolhe a árvore T que minimiza $C_\alpha(T)$, sendo que α é escolhido por alguma forma de validação cruzada (geralmente, 5-fold ou 10-fold).
- Note que há uma forma de regularização contida na definição de C_α, uma vez que estamos penalizando árvores com muitas folhas.

Exemplo: classificação de e-mails em *spam* e *ham* (1)

- Exemplo discutido na página 300 do ESL.
- Queremos classificar e-mails em duas categorias: *spam* (lixo eletrônico) e *ham* (legítimo).
- Temos 4601 e-mails, sendo que 1813 são marcados como *spam*; foram definidas 57 variáveis preditoras.
- Temos 48 preditoras quantitativas com as porcentagens das ocorrências de palavras específicas, tais como `business`, `address`, `internet`, `free` etc.
- Mais 6 preditoras quantitativas com as porcentagens das ocorrências de caracteres específicos, tais como `;`, `$`, `!` etc.
- O tamanho médio das sequências ininterruptas de letras maiúsculas (`CAPAVE`).
- O tamanho da maior sequência ininterrupta de letras maiúsculas (`CAPMAX`).
- A soma dos comprimentos das sequências ininterruptas de letras maiúsculas (`CAPTOT`).

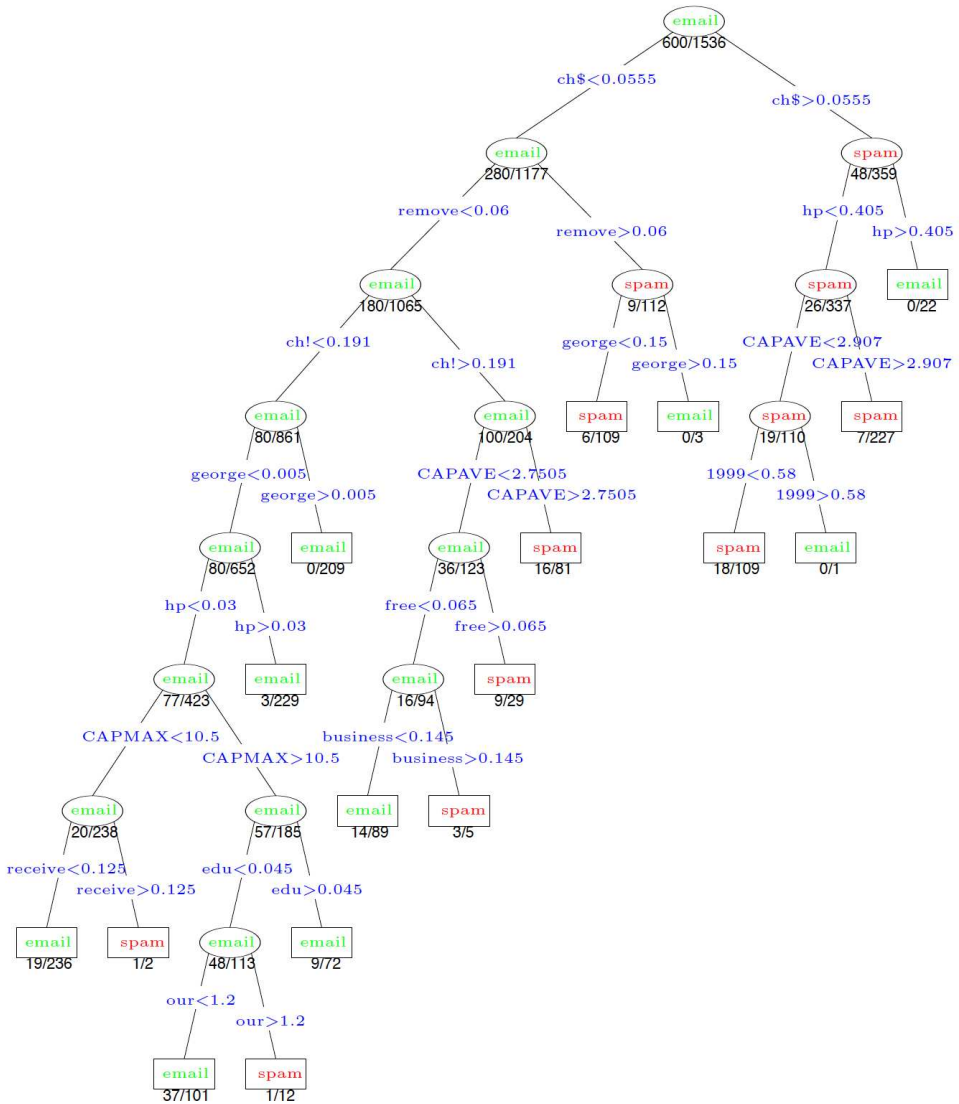
Exemplo: classificação de e-mails em *spam* e *ham* (2)

Página 314 do ESL.



Exemplo: classificação de e-mails em *spam* e *ham* (3)

Página 315 do ESL.



Detour: Bootstrap

- Suponha que $Z_1, \ldots, Z_n$ são variáveis aleatórias independentes e identicamente distribuídas (IID) com função de distribuição F.
- Função de distribuição empírica: $\hat{F}_n(t) = \sum_{i=1}^{n} I_{(-\infty,t]}(Z_i)$.
- Pela Lei Forte dos Grandes Números, temos que $\hat{F}_n(t) \to F(t)$ quase certamente, quando $n \to \infty$.
- Seja $M = g(Z_1, \ldots, Z_n)$ a mediana amostral.
- Suponha que queremos estimar $\mathrm{Var}[M]$ a partir dos dados $Z_1, \ldots, Z_n$.
- A idéia do bootstrap (Efron) é "substituir" F por $\hat{F}_n$, considerar $\tilde{Z}_1, \ldots, \tilde{Z}_n$ IID com função de distribuição $\hat{F}_n$, definir $\tilde{M} = g(\tilde{Z}_1, \ldots, \tilde{Z}_n)$, e aproximar $\mathrm{Var}[M]$ por $\mathrm{Var}[\tilde{M}]$, uma vez que podemos calcular $\mathrm{Var}[\tilde{M}]$ (em geral, via método de Monte Carlo).
- O bootstrap tem uma interpretação bayesiana (Rubin) em termos de um processo Dirichlet (Ferguson) centrado na função de distribuição empírica.

Bagging

- Método de "comitê" ou "ensemble" criado por Breiman (um dos autores do CART).
- Bagging = Bootstrap Aggregation.
- A partir dos dados de treinamento $(x_1, y_1), \ldots, (x_n, y_n)$, geramos dados bootstrap $(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_n, \tilde{y}_n)$ e construímos uma árvore de classificação via algoritmo CART.
- Repetindo este processo, produzimos uma floresta de árvores de classificação $\hat{\varphi}_1, \ldots, \hat{\varphi}_B$.
- A classificação de um novo exemplar x_{n+1} é feita pelo voto da maioria das B árvores de classificação da floresta.

Boosting

- Método de "comitê" ou "ensemble" criado por Schapire em 1990.
- Celebrado e premiado como uma das maiores descobertas de "Machine Learning" de todos os tempos (Prêmio Gödel de 2003).
- A idéia é partir de um classificador mediano e ir construindo classificadores melhores atribuindo pesos maiores aos exemplos que forem classificados incorretamente.
- Por exemplo, o algoritmo CART pode ser modificado para atribuir pesos a cada um dos dados de treinamento, redefinindo

$$\hat{p}_k(R_j) = \frac{\sum_{\{i\,:\,x_i \in R_j\}} w_i \cdot I_{\{k\}}(y_i)}{\sum_{\{i\,:\,x_i \in R_j\}} w_i}.$$

- No final do processo, a classificação é feita pelo conjunto dos classificadores obtidos.

Random Forests

- Método de "comitê" ou "ensemble" criado por Breiman.
- É uma variação do Bagging.
- A diferença é que, ao construir cada árvore de classificação da floresta a partir dos dados bootstrap, em cada split do CART nos restringimos a um subconjunto aleatório de m preditoras.
- Breiman (2001): "*Random Forests may also be viewed as a Bayesian procedure. Although I doubt this is a fruitfull line of exploration, if it could explain the bias reduction, I might become more of a Bayesian*".
- O entendimento teórico de todos os mecanismos envolvidos nas Random Forests é um problema em aberto, com uma série de resultados parciais bastante recentes. Veja, por exemplo, Denil et al. (2014).
- Todos os métodos de classificação discutidos nesta aula se aplicam *mutatis mutandis* a problemas de regressão.

Comparação usando os dados de *spam*

Página 589 do ESL.