

ANÁLISE INTELIGENTE DE DADOS (COB754)

MODELOS MÚLTIPLOS PREDITIVOS

LETÍCIA MARTINS RAPOSO

IDEIA

COMBINAR, DE ALGUMA FORMA, AS DECISÕES INDIVIDUAIS DE UM CONJUNTO DE MODELOS PREDITIVOS PARA PREDIZER NOVOS EXEMPLOS, A FIM DE REDUZIR A VARIÂNCIA OU VIÉS E MELHORAR A PREDIÇÃO.



MODELOS MÚLTIPLOS PREDITIVOS

Diferentes algoritmos de aprendizado exploram:

- Diferentes linguagens de representação;
- Diferentes espaços de procura;
- Diferentes funções de avaliação de hipóteses.

Como é possível explorar essas diferenças?

É possível desenvolver um conjunto de modelos que, trabalhando juntos, obtêm um melhor desempenho do que cada modelo trabalhando individualmente?

ERRO

O erro que surge de qualquer modelo pode ser dividido em três componentes matematicamente:

$$Err(x) = (E[\hat{f}(x)] - f(x))^2 + E[\hat{f}(x) - E[\hat{f}(x)]]^2 + \sigma_e^2$$

$$Err(x) = Viés^2 + Variância + Ruído$$

QUADRADO DO VIÉS: MEDE O QUÃO LONGE A PREDIÇÃO MÉDIA DO ALGORITMO DE APRENDIZADO (SOBRE TODOS OS POSSÍVEIS CONJUNTOS DE TREINAMENTO CUJO TAMANHO É IGUAL AO TAMANHO DO CONJUNTO DE TREINAMENTO DADO) ESTÁ DO VALOR CORRETO.

VARIÂNCIA: MEDE O QUÃO LONGE A PREDIÇÃO DO ALGORITMO DE APRENDIZADO ESTÁ DA PREDIÇÃO MÉDIA PARA DIFERENTES CONJUNTOS DE OBJETOS DE UM DADO TAMANHO.

RUÍDO: LIMITE INFERIOR DO ERRO ESPERADO DE QUALQUER ALGORITMO DE APRENDIZADO.

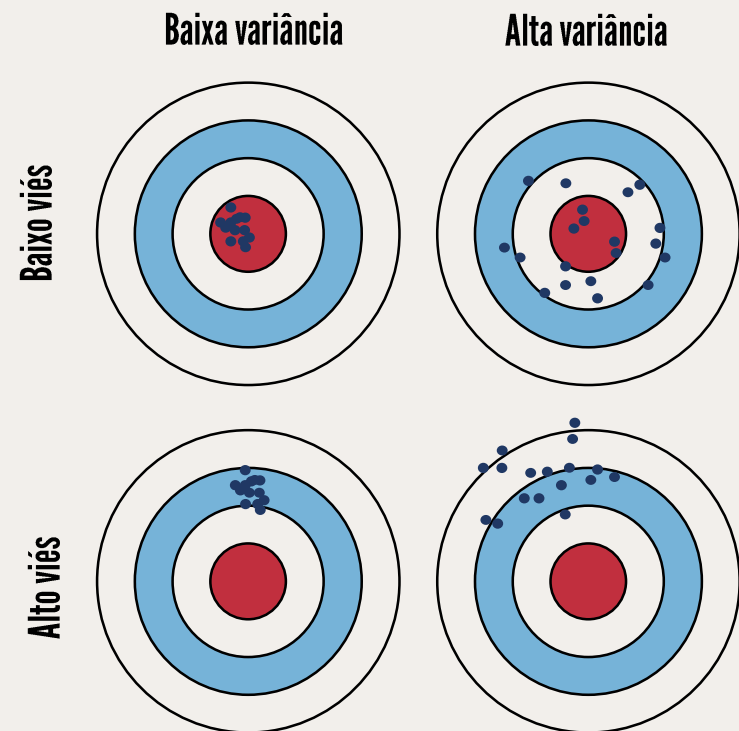
VIÉS VS. VARIÂNCIA

Viés: tendência que o modelo tem de aprender errado por não levar em consideração toda informação necessária (*underfitting*).

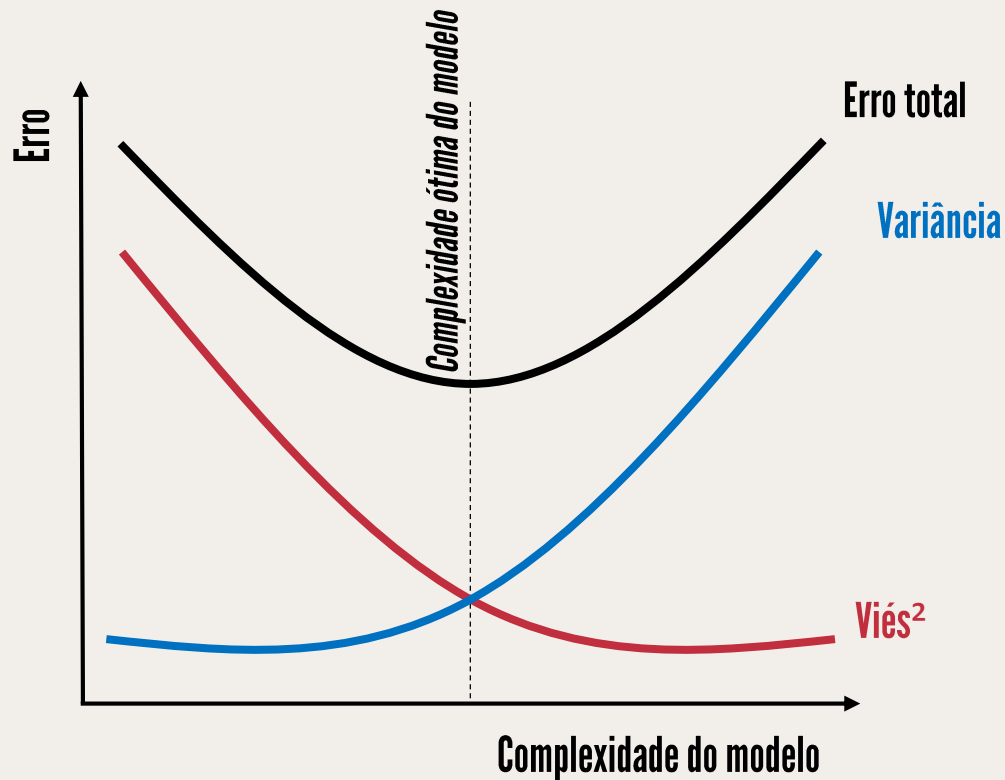
↑ viés: modelo com baixo desempenho que continua perdendo tendências importantes, pode não se ajustar bem aos dados.

Variância: tendência de aprender coisas aleatórias, ajustando modelos altamente flexíveis que acompanham o erro nos dados (*overfitting*).

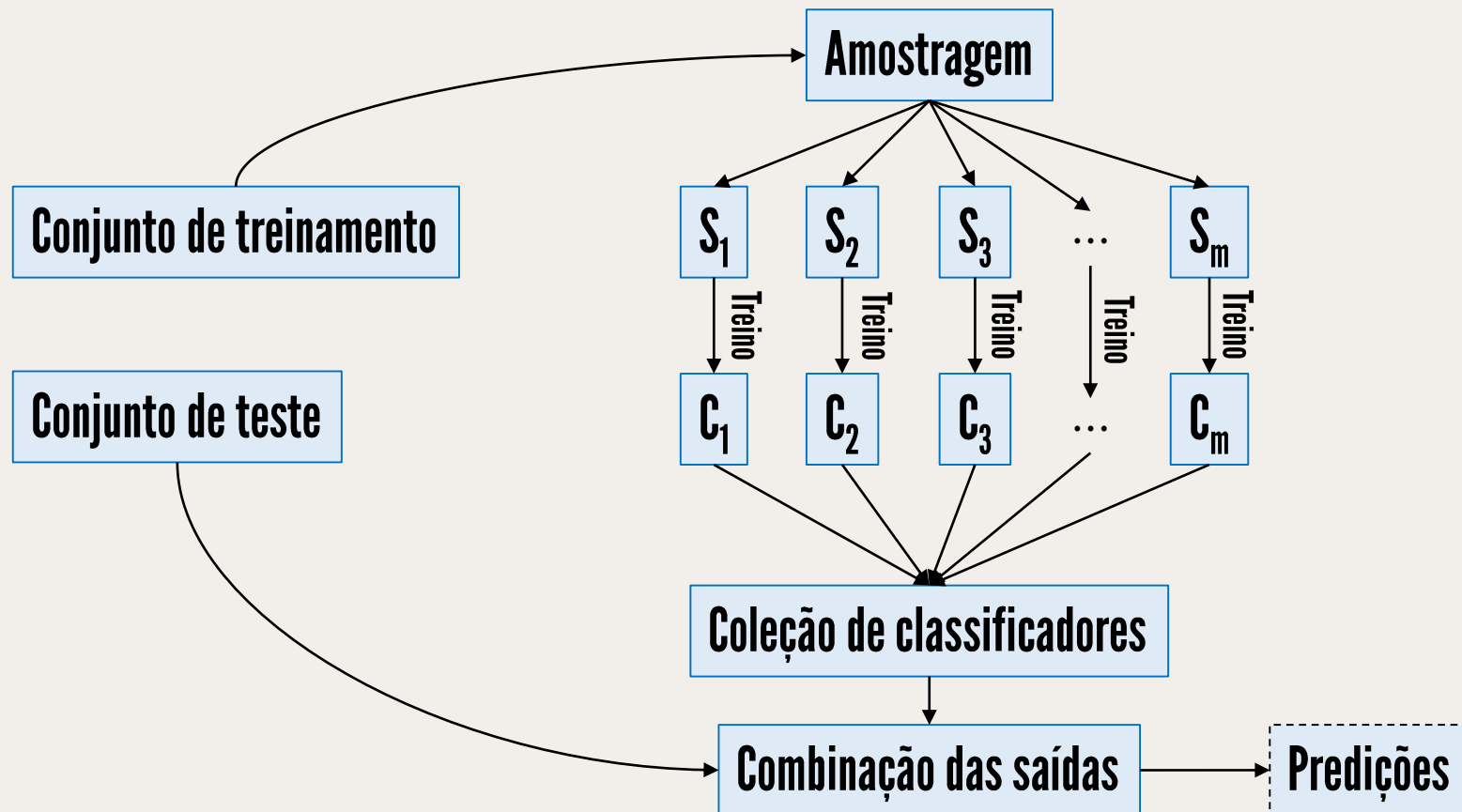
↑ variância: se ajustará ao seu conjunto de treinamento e terá um desempenho ruim para observações fora desse conjunto.



ERRO VS COMPLEXIDADE DO MODELO



VISÃO LÓGICA DO MÉTODO DE MODELOS MÚLTIPLOS PREDITIVOS



TIPOS DE MODELOS MÚLTIPLOS PREDITIVOS

HOMOGÊNEOS

UM ÚNICO ALGORITMO
DE APRENDIZADO DE
MÁQUINA, DIFERENTES
AMOSTRAS DO
CONJUNTO ORIGINAL
BAGGING e BOOSTING

HETEROGÊNEOS

DIFERENTES
ALGORITMOS DE
APRENDIZADO DE
MÁQUINA, AMOSTRAS
IGUAIS A DO CONJUNTO
ORIGINAL
STACKING

BAGGING (BOOTSTRAP AGGREGATING)

- Desenvolvido por Leo Breiman ("Bagging predictors," Machine Learning, 24(2):123-140, 1996).
- Baseia-se em criar preditores com amostras *bootstrap* dos dados, e depois combiná-los a fim de formar um melhor preditor.
- Aprendizado em paralelo dos preditores.
- Classificação: os preditores são combinados por meio de voto.
- Regressão: usa-se, comumente, a média dos preditores.
- O objetivo do *bagging* é reduzir a complexidade dos modelos que superajustam os dados.
 - Tem pouco efeito sobre o viés, mas reduz a variância, principalmente em funções não lineares.

BAGGING

O método pode melhorar o desempenho de preditores instáveis, que são basicamente preditores com alta variância.

Preditor instável: pequenas perturbações no conjunto de treinamento causam grandes variações nos preditores treinados com ele.

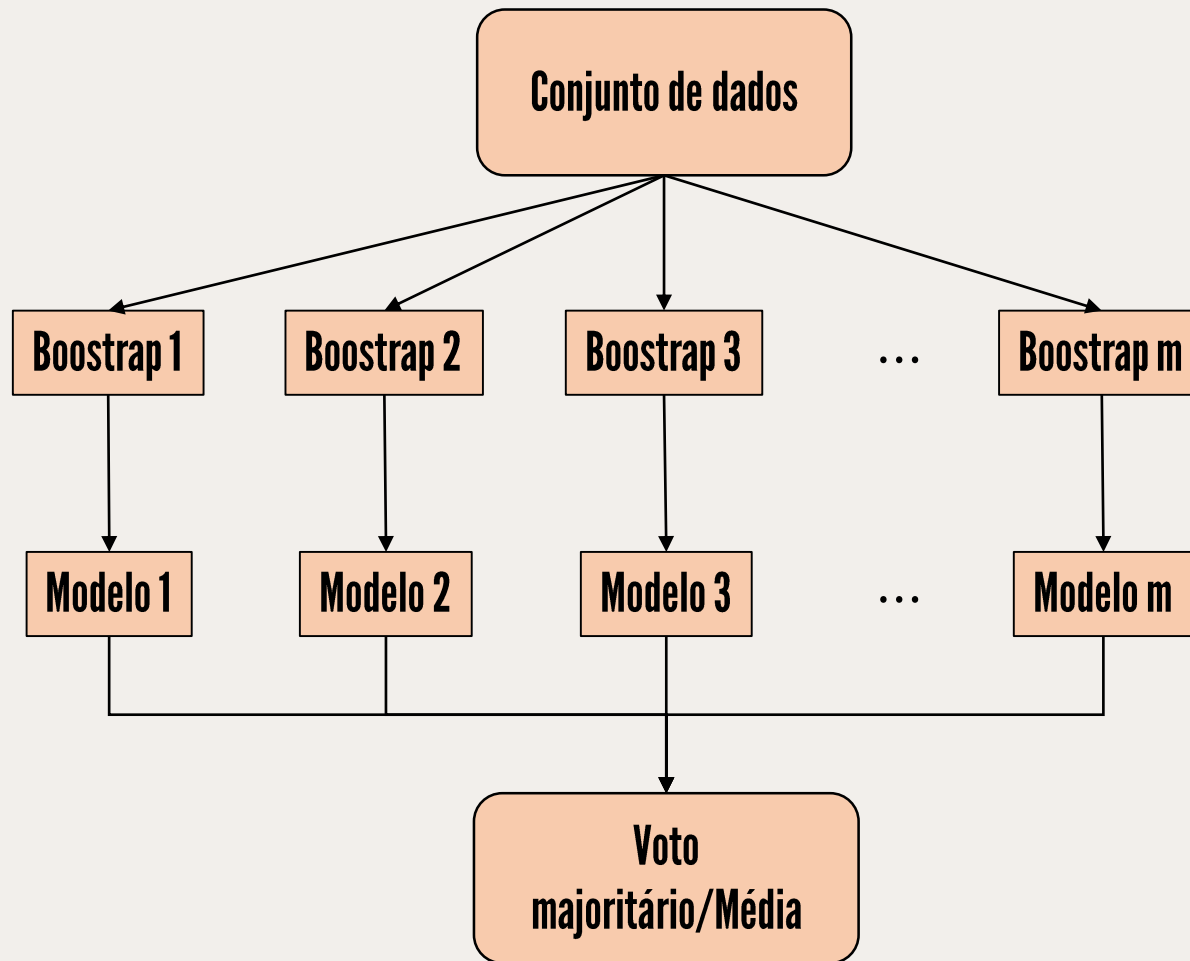
ALGORITMOS INSTÁVEIS	ALGORITMOS ESTÁVEIS
Redes Neurais	K-Nearest Neighbors
Árvores de Decisão	Análise Discriminante
Regressão	
	NÃO PODEM SER MELHORADOS POR BAGGING.

BAGGING

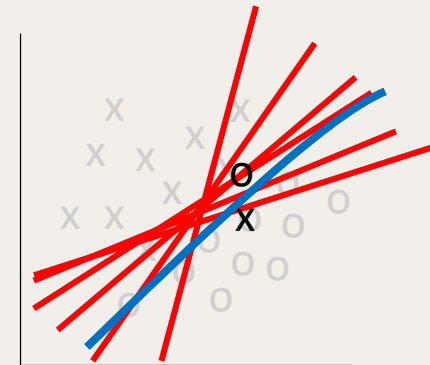
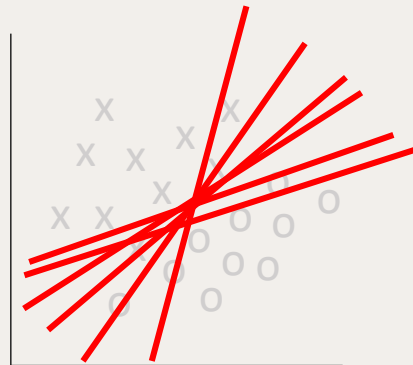
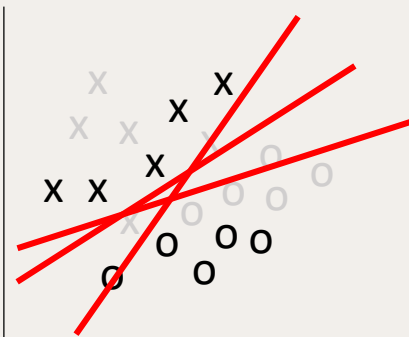
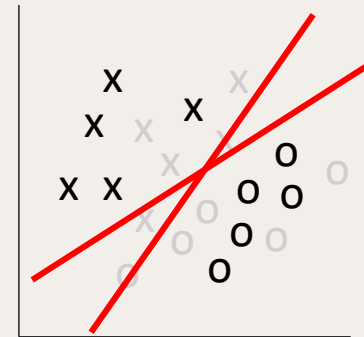
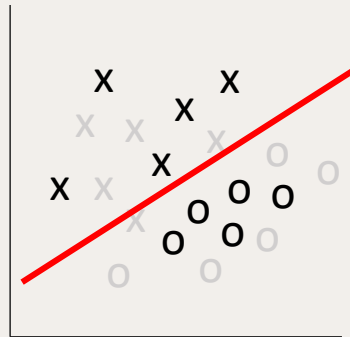
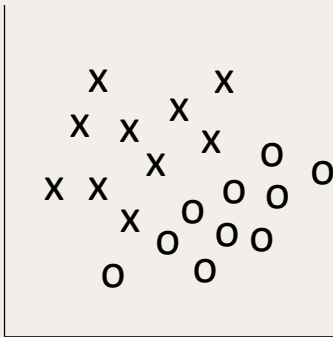
Considere uma amostra de treinamento $L = \{x_i, y_i\}_{i=1}^N$ e um preditor $\hat{d}_n(x)$, o qual é construído com base na amostra L .

1. Construa uma amostra *bootstrap* $L^* = \{x_i^*, y_i^*\}_{i=1}^N$;
2. Compute o preditor via *bootstrap* $d_n^*(x)$, utilizando o mesmo procedimento para construir $\hat{d}_n(x)$, mas com a amostra L^* ;
3. O preditor via *bagging* é dado por $\hat{d}_{n;B}(x) \approx \frac{1}{K} \sum_{k=1}^K d_{n;(k)}^*(x)$ no caso de regressão;
4. Em classificação, a classe predita pelo classificador via *bagging* é a mais votada pelos classificadores, em que $N_j = \#\{k; d_{n;(k)}^*(x) = j\}$ e o classificador via *bagging* é $\hat{d}_{n;B}(x) = \operatorname{argmax} N_j$.

BAGGING



EXEMPLO



SUBAGGING (SUBSAMPLE AGGREGATING)

- Proposta por Bühlmann e Yu (2002);
- Consiste em retirar amostras de tamanho $M < N$ da amostra original L de tamanho N , **sem reposição**, calcular o preditor em cada uma dessas amostras e depois combiná-los.
 - $M = aN$, a normalmente $\frac{1}{2}$.
- Atraente do ponto de vista computacional: \downarrow n° de pontos a serem ajustados pelos preditores ($M < N$).
- Bühlmann e Yu (2002) mostraram que:
 - *Half subagging* pode ser praticamente idêntico a *bagging* (para árvores de classificação binárias com apenas 2 nós terminais);
 - Escolher a muito pequeno pode levar a resultados piores do que o preditor sem *bagging*.

BOOSTING

- Técnica para combinar múltiplos modelos base (fracos) a fim de obter um modelo final com melhor desempenho (forte).
- Em cada iteração, o algoritmo atribui pesos maiores às observações incorretamente preditas na iteração anterior e essa etapa se repete até que o limite do algoritmo base seja atingido ou uma maior acurácia seja alcançada.
- O objetivo do *boosting* é aumentar a complexidade dos modelos que sofrem de alto viés, especialmente no caso de *underfitting*.
 - Tenta reduzir o erro nas predições → reduz o viés.

BOOSTING

APRENDIZADO SEQUENCIAL DOS PREDITORES

O 1º APRENDE A PARTIR DE TODO O CONJUNTO DE DADOS

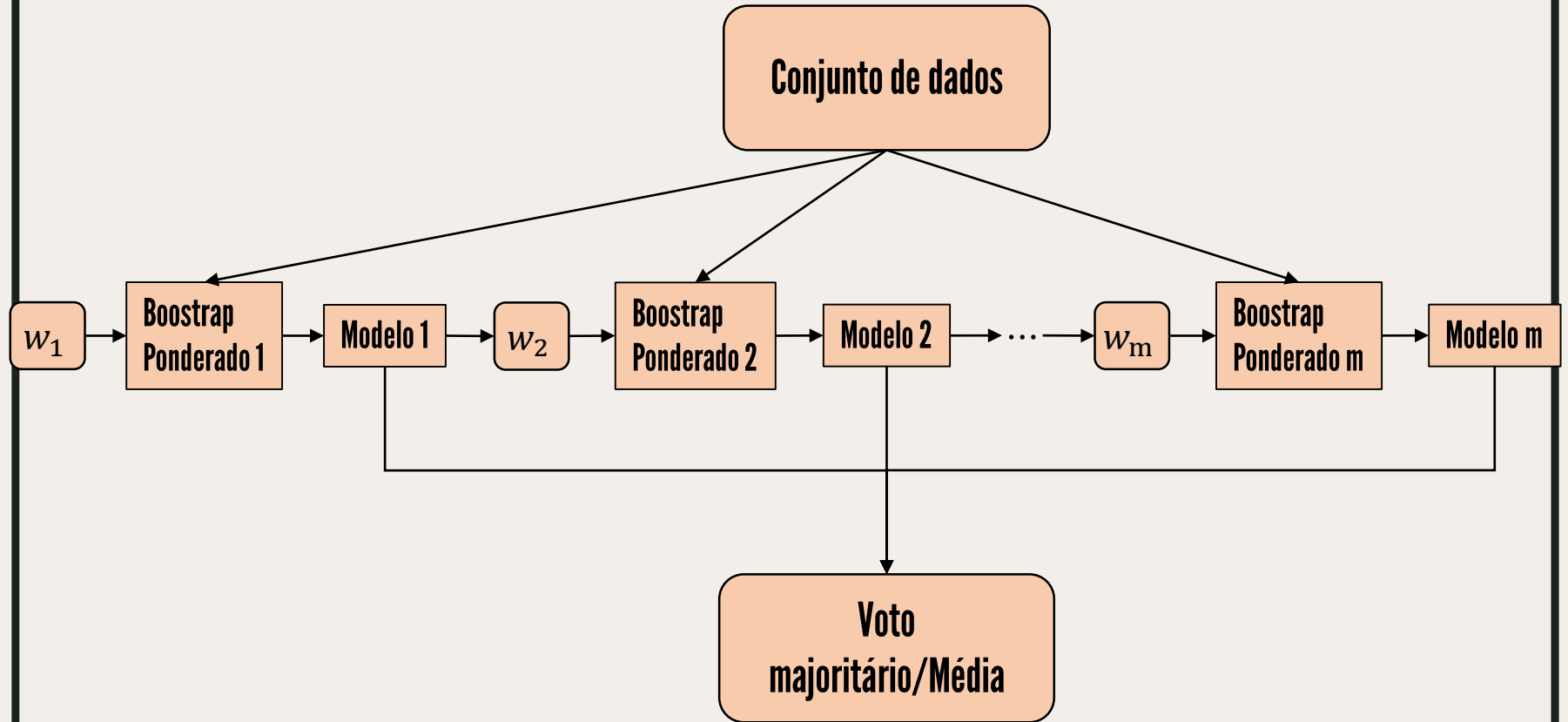


OS SEQUINTE APRENDEM A PARTIR DE UM CONJUNTO DE TREINAMENTO COM BASE NO DESEMPENHO DOS ANTERIORES



↑ PESOS DOS EXEMPLOS ERRONEAMENTE CLASSIFICADOS
↑ PROBABILIDADE DE APARECER NO CONJUNTO DE TREINAMENTO DO PRÓXIMO PREDITOR

BOOSTING



ADABOOST (ADAPTATIVE BOOSTING)

- Um dos mais bem sucedidos algoritmos de *Boosting*.
- Desenvolvido por Freund e Schapire (1997).
- Treina uma série de modelos sequencialmente.
- Em cada iteração, a distribuição de pesos é atualizada para indicar a importância do exemplo no conjunto de dados.

ADABOOST (PARA DUAS CLASSES)

Dado n exemplos (x_i, y_i) , em que $x \in X$ e $y \in \pm 1$:

- Inicialize os pesos (probabilidade de selecionar o exemplo i na amostra): $D_1(i) = \frac{1}{n}, i = 1, 2, \dots, n$;
- Repita para $t = 1, 2, \dots, T$:
 1. Obtenha uma amostra (*bootstrap*) usando os pesos $D(i)$ nos dados de treinamento;
 2. Treine o classificador $h_t(x): X \rightarrow \pm 1$ com a amostra;
 3. Calcule a soma dos pesos para os exemplos classificados erroneamente $\epsilon_t = \sum_{i: h_t(x_i) \neq y_i} D_t(i)$;
 4. Calcule α_t (chance de classificação errada);

$$\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$$

ADABOOST

5. Atualize os pesos $D_{t+1}(i) = \frac{D_t(i) \exp\{-\alpha_t y_i h_t(x_i)\}}{Z_t}$, $Z_t = \sum_k D_t(k) \exp\{-\alpha_t y_k h_t(x_k)\}$

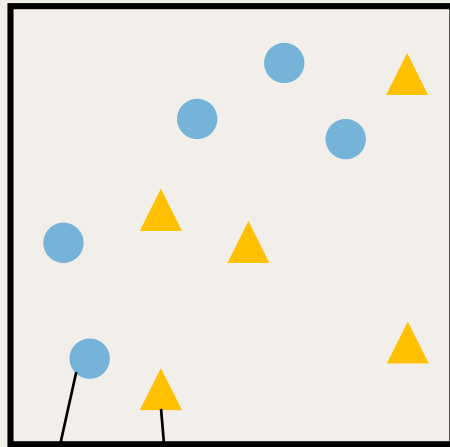
- Se o classificador acertar: +1 e +1 ou -1 e -1: $\exp(-) \rightarrow$ diminui;
- Se o classificador errar: +1 e -1: $\exp(+)$ \rightarrow aumenta;
- Saída do classificador final:

$$H_{final} = \text{sign}(\sum_t \alpha_t h_t(x)).$$

EXEMPLO

CONJUNTO DE DADOS ORIGINAL

D_1

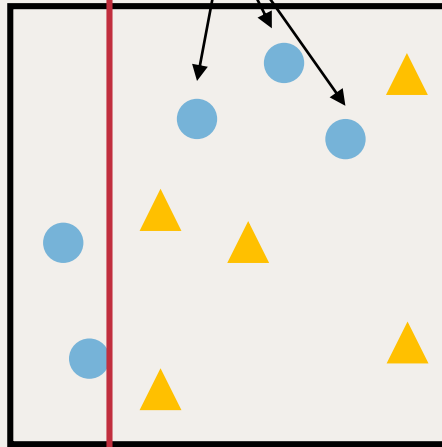


+1

-1

CASOS CLASSIFICADOS
ERRONEAMENTE

h_1

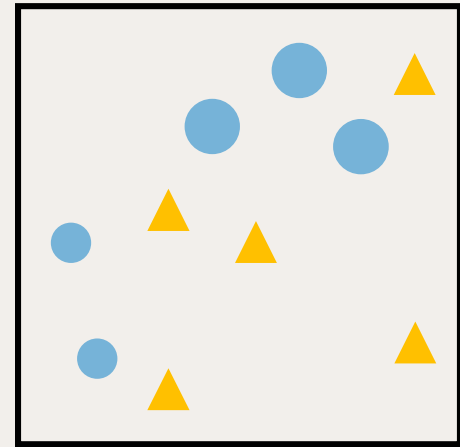


$$\varepsilon_1 = 0,30$$

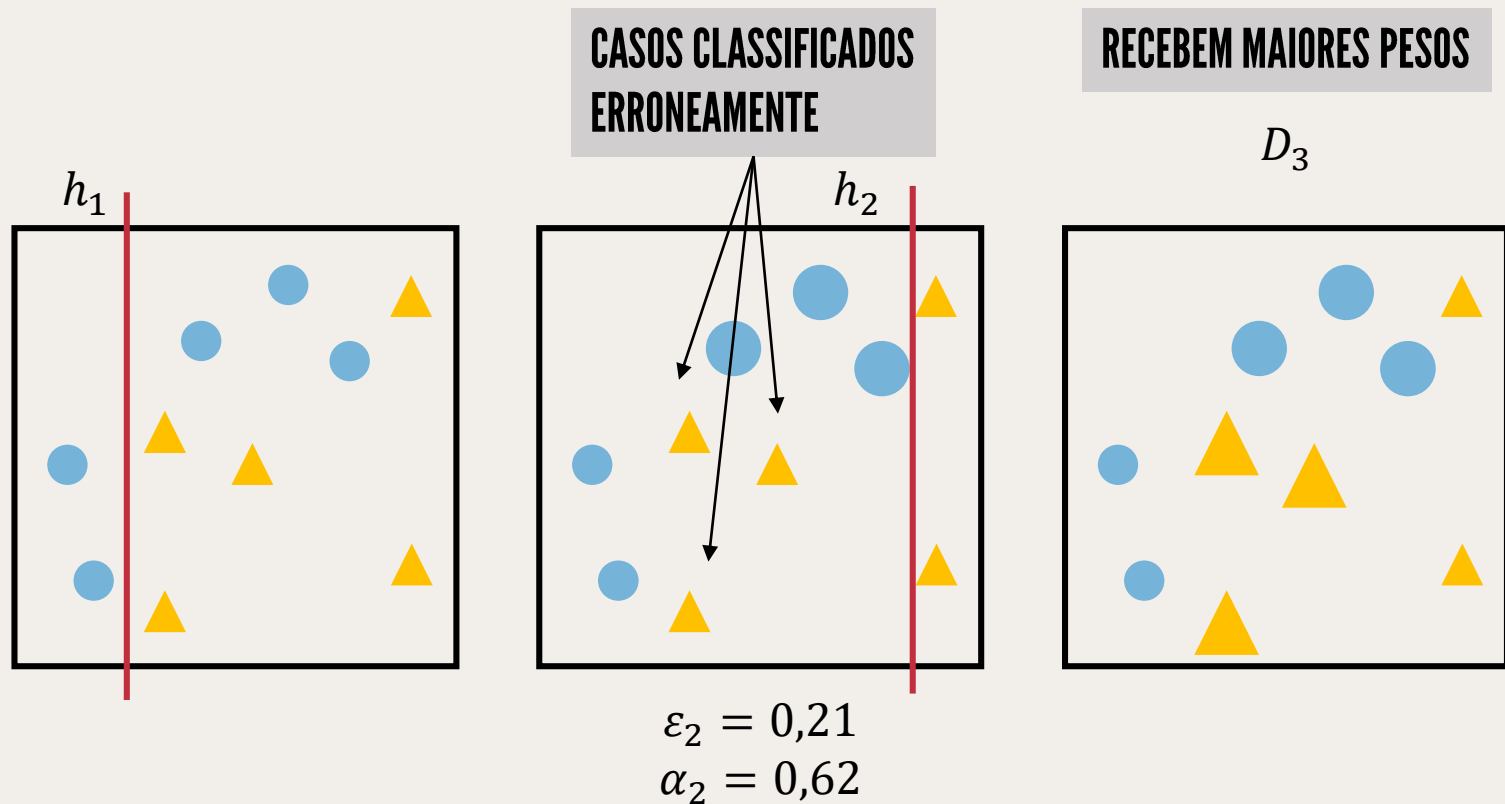
$$\alpha_1 = 0,42$$

RECEBEM MAIORES PESOS

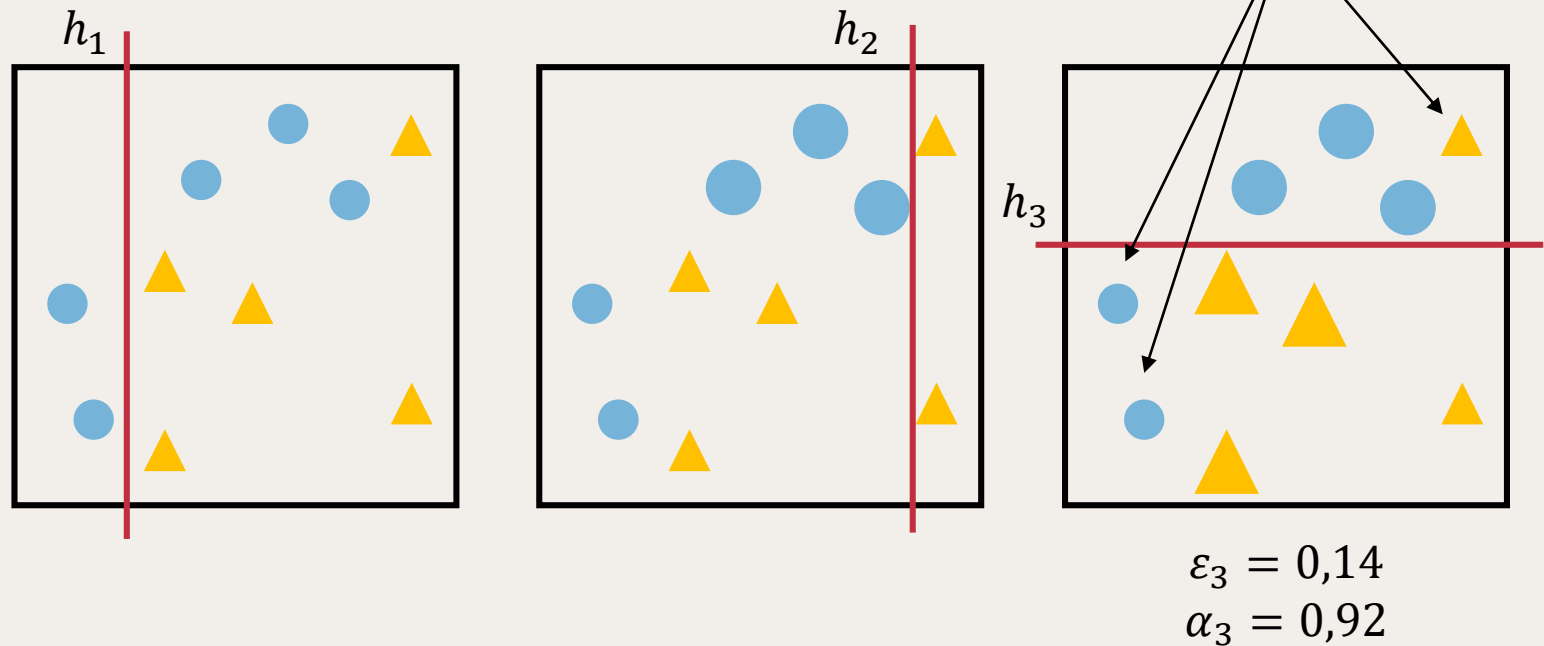
D_2



EXEMPLO

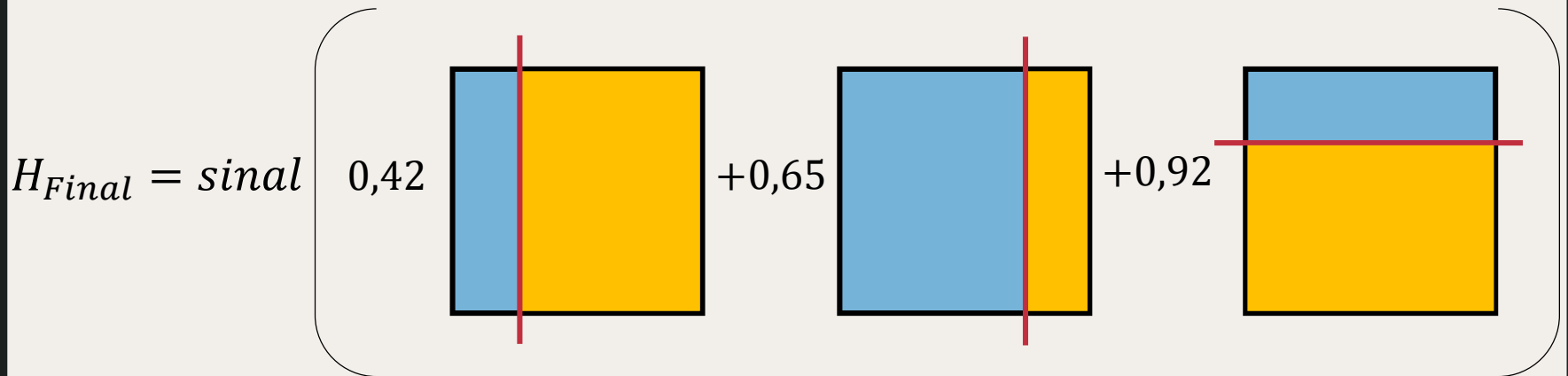


EXEMPLO

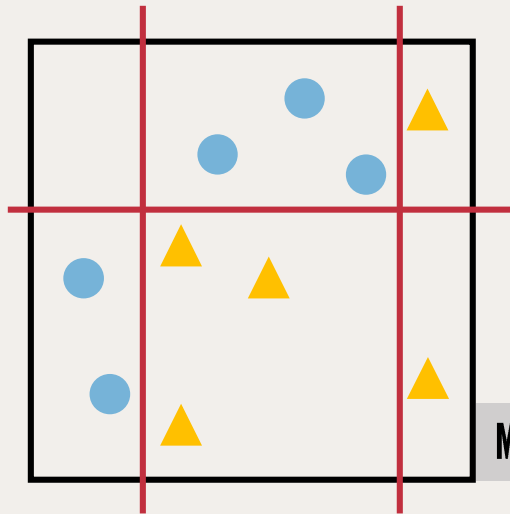


EXEMPLO



MODELOS SIMPLES



=



MODELO COMPLEXO

SE O SINAL FOR > 0 , PREDIZEMOS 
SE O SINAL FOR < 0 , PREDIZEMOS 

DIFERENÇAS ENTRE BAGGING E BOOSTING

BAGGING

- Qualquer elemento tem a mesma probabilidade de aparecer no novo conjunto de dados.
- Treinamento paralelo (cada modelo é construído independentemente).
- O resultado é obtido pela média das respostas dos N modelos (ou pelo voto majoritário).
- Treina e mantém.

BOOSTING

- As observações são ponderadas: algumas delas irão participar mais frequentemente.
- Constrói os modelos de forma sequencial: cada novo modelo é influenciado pelo desempenho do anterior.
- Atribui um conjunto de pesos para os N classificadores a fim de obter uma média ponderada de suas estimativas.
- Treina e avalia.

RANDOM FOREST (FLORESTA ALEATÓRIA)

- As árvores de decisão são excelentes preditores. No entanto, nem sempre generalizam bem.
 - Por ex.: poda da árvore → fica menor e mais generalizável.
- Um passo além, nesse processo, é o uso de métodos que utilizem técnicas como:
 - *Bagging*;
 - *Randomizing*: em cada método/técnica empregada, utiliza-se um conjunto diferente de variáveis.
- Um bom exemplo é a técnica de *Random Forest*, que apresenta excelentes características de acurácia, generalização para outras amostras que foram utilizadas no treinamento e capacidade de bom desempenho em pequenas amostras.

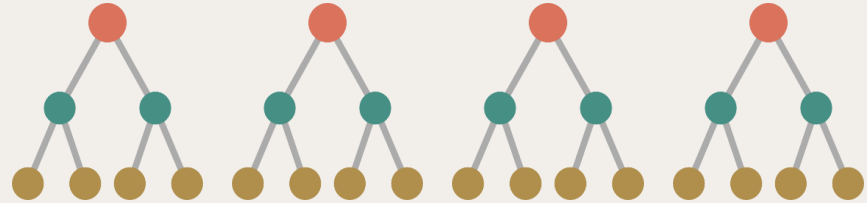
RANDOM FOREST (FLORESTA ALEATÓRIA)

RANDOM FOREST =
BAGGING DE ÁRVORES DE
DECISÃO
+
RANDOMIZAÇÃO



- EXECUTA TANTO TAREFAS DE REGRESSÃO QUANTO DE CLASSIFICAÇÃO.
- REALIZA REDUÇÃO DE DIMENSIONALIDADE.
- TRATA VALORES AUSENTES E DISCREPANTES.
- TIPO DE MÉTODO DE APRENDIZADO CONJUNTO, EM QUE UM GRUPO DE MODELOS FRACOS SE COMBINAM PARA FORMAR UM MODELO PODEROSO.

ALGORITMO



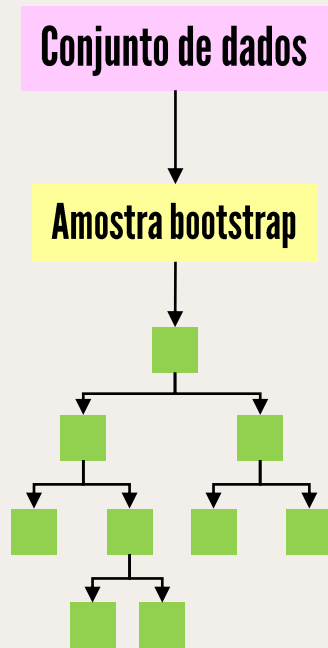
1. Para $b = 1$ a B :
 1. Obtenha uma amostra *bootstrap* Z^* de tamanho N a partir do conjunto de treinamento.
 2. Cresça uma árvore de decisão T_b a partir da amostra *bootstrap*, repetindo recursivamente os passos abaixo para cada nó da árvore, até o tamanho mínimo do nó n_{min} ser alcançado.
 1. Selecione m variáveis aleatoriamente a partir das p variáveis.
 2. Pegue a melhor variável para a divisão dentre as m .
 3. Divida o nó em dois nós filhos.
2. Finalize o conjunto de árvores $\{T_b\}_1^B$.

ALGORITMO

RECOMENDAÇÕES

Classificação: o valor padrão para m é $\lfloor \sqrt{p} \rfloor$ e o tamanho mínimo do nó é 1.

Regressão: o valor padrão para m é $\lfloor p/3 \rfloor$ e o tamanho mínimo do nó é 5.



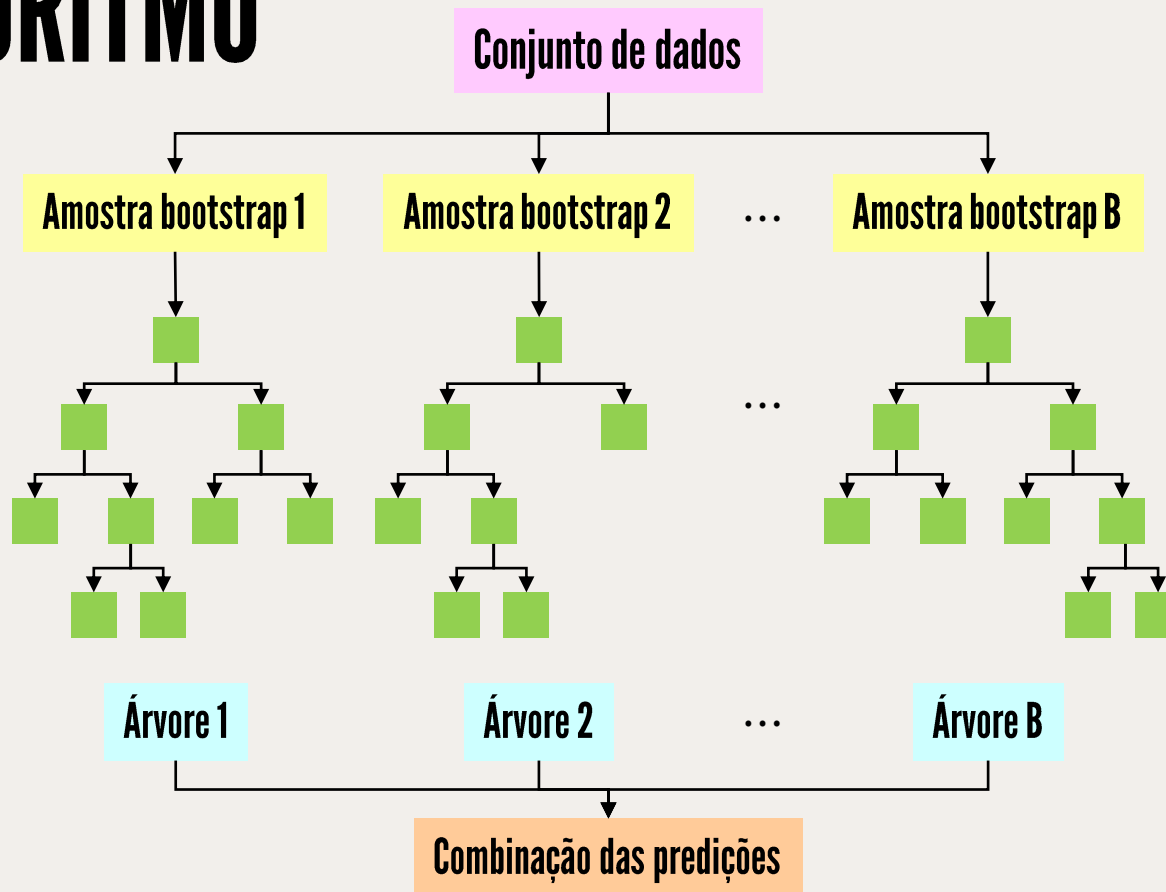
1. Obtenha uma amostra *bootstrap* Z^* de tamanho N a partir do conjunto de treinamento.

2. Cresça uma árvore de decisão T_b a partir da amostra *bootstrap*, repetindo recursivamente os passos abaixo para cada nó da árvore, até o tamanho mínimo do nó n_{min} ser alcançado.

1. Selecione m variáveis aleatoriamente a partir das p variáveis.
2. Pegue a melhor variável para a divisão dentre as m .
3. Divida o nó em dois nós filhos.

REPITA ESSA PROCESSO B VEZES E FINALIZE O CONJUNTO DE B ÁRVORES .

ALGORITMO



Para fazer uma predição de uma nova observação x :

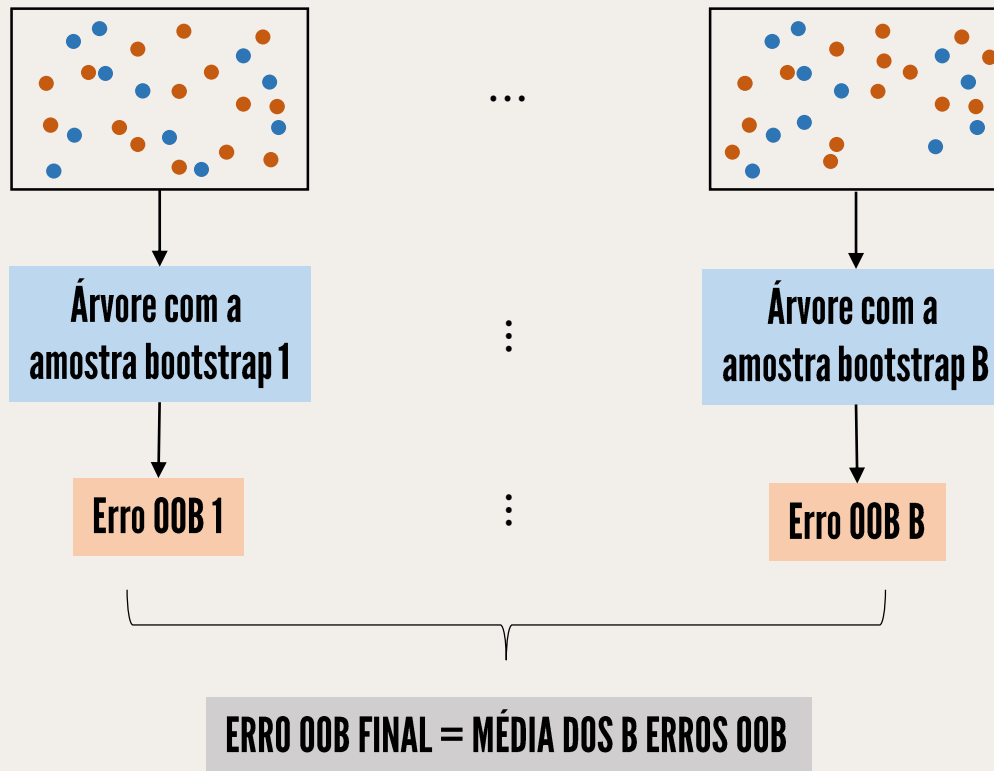
Regressão: $f_{rf}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x)$

Classificação: Sendo $\hat{c}_b(x)$ a classe predita pelo b -ésima árvore, então $\hat{c}_{rf}^B(x) = \text{voto major.} \{ \hat{c}_b(x) \}_1^B$.

AMOSTRAS “*OUT-OF-BAG*”

- Uma característica importante da *random forest* é o uso de amostras “*out-of-bag*” (OOB) para medir o erro de predição.
- OOB é o erro médio de predição em cada amostra de treinamento x_i , usando apenas as árvores que não tinham x_i em sua amostra *bootstrap*.
- Estimativas “*out-of-bag*” ajudam a evitar a necessidade de um conjunto de dados de validação independente.
- Uma vez que o erro OOB se estabilize, o treinamento pode ser finalizado.

AMOSTRAS “*OUT-OF-BAG*”



IMPORTÂNCIA DA VARIÁVEL

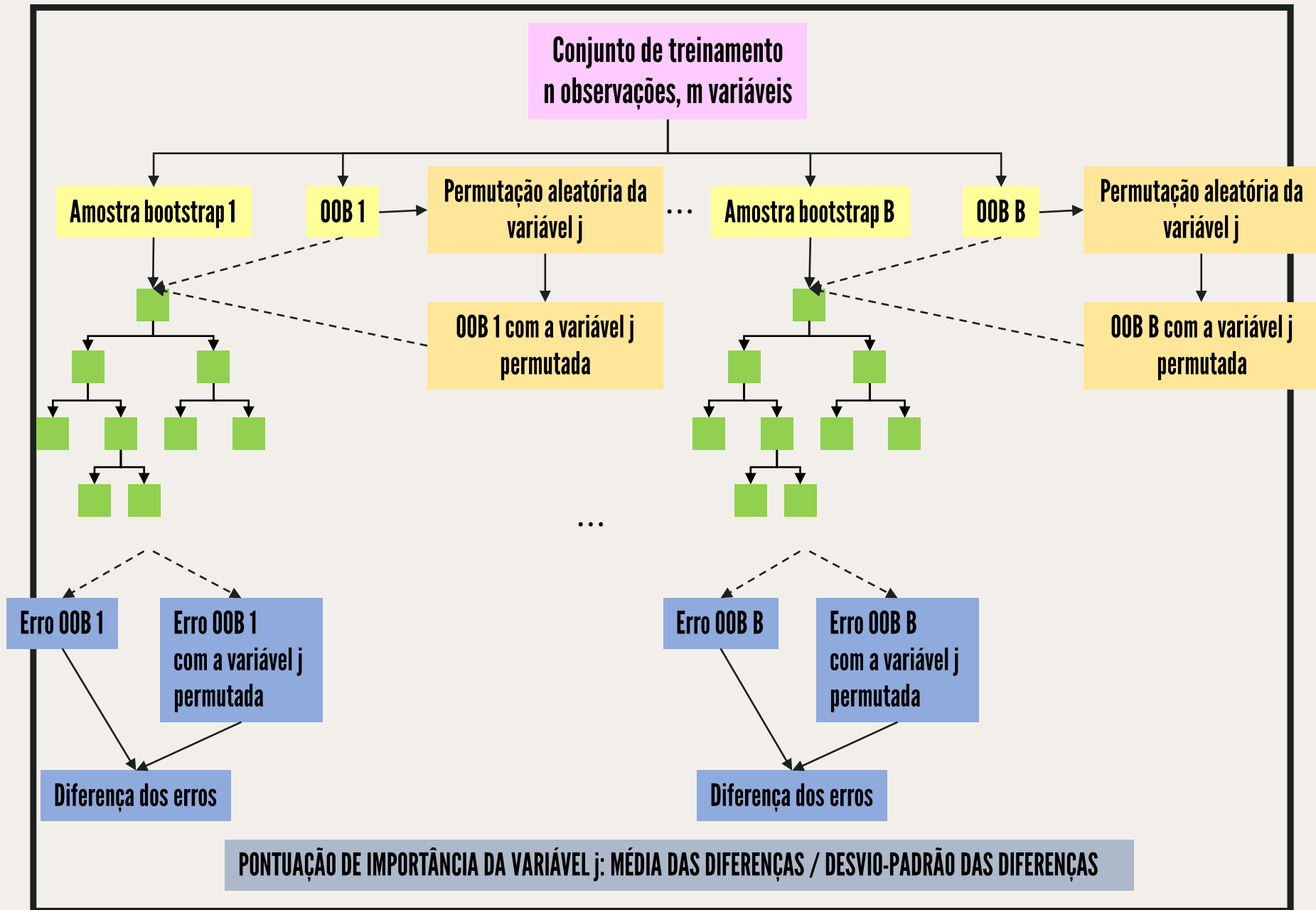
IMPUREZA DO NÓ: MÉTODO MAIS SIMPLES

- Calcula-se, em média, o quanto a partição gerada por uma variável reduziu as impurezas dos nós sobre todas as árvores.
 - Regressão: soma residual dos quadrados;
 - Classificação: índice Gini.
- Inclinado a preferir variáveis com mais categorias.

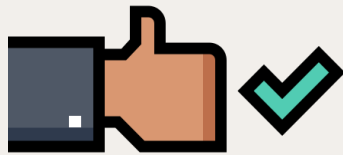
IMPORTÂNCIA DA VARIÁVEL

MÉTODO DE PERMUTAÇÃO

- Para cada árvore, o erro OOB é registrado.
- Para medir a importância de uma variável após o treinamento, os valores da j -ésima variável são permutados (mantendo a distribuição original) nos conjuntos OOB e o erro OOB é novamente computado neste conjunto de dados perturbado.
- A pontuação de importância da j -ésima variável é calculada pela média da diferença entre o erro OOB antes e depois da permutação em todas as árvores. A pontuação é normalizada pelo desvio padrão dessas diferenças.
- Assim, para variáveis sem importância, a permutação deve ter pouco ou nenhum efeito na acurácia do modelo, enquanto a permutação de variáveis importantes deve diminuí-la significativamente.



VANTAGENS



NA CLASSIFICAÇÃO,
REDUZEM O PROBLEMA DE
OVERFITTING.

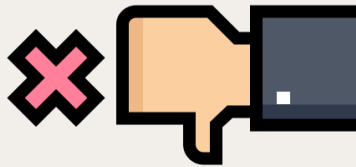
IDENTIFICA AS VARIÁVEIS
MAIS IMPORTANTES DO
CONJUNTO DE
TREINAMENTO.

EXTREMAMENTE FLEXÍVEL,
COM UMA ACURÁCIA MUITO
ALTA.

NÃO EXIGE PREPARAÇÃO
DOS DADOS DE ENTRADA.

A CONTRUÇÃO DAS ÁRVORES
PODE SER PARALELIZADA.

DESVANTAGENS



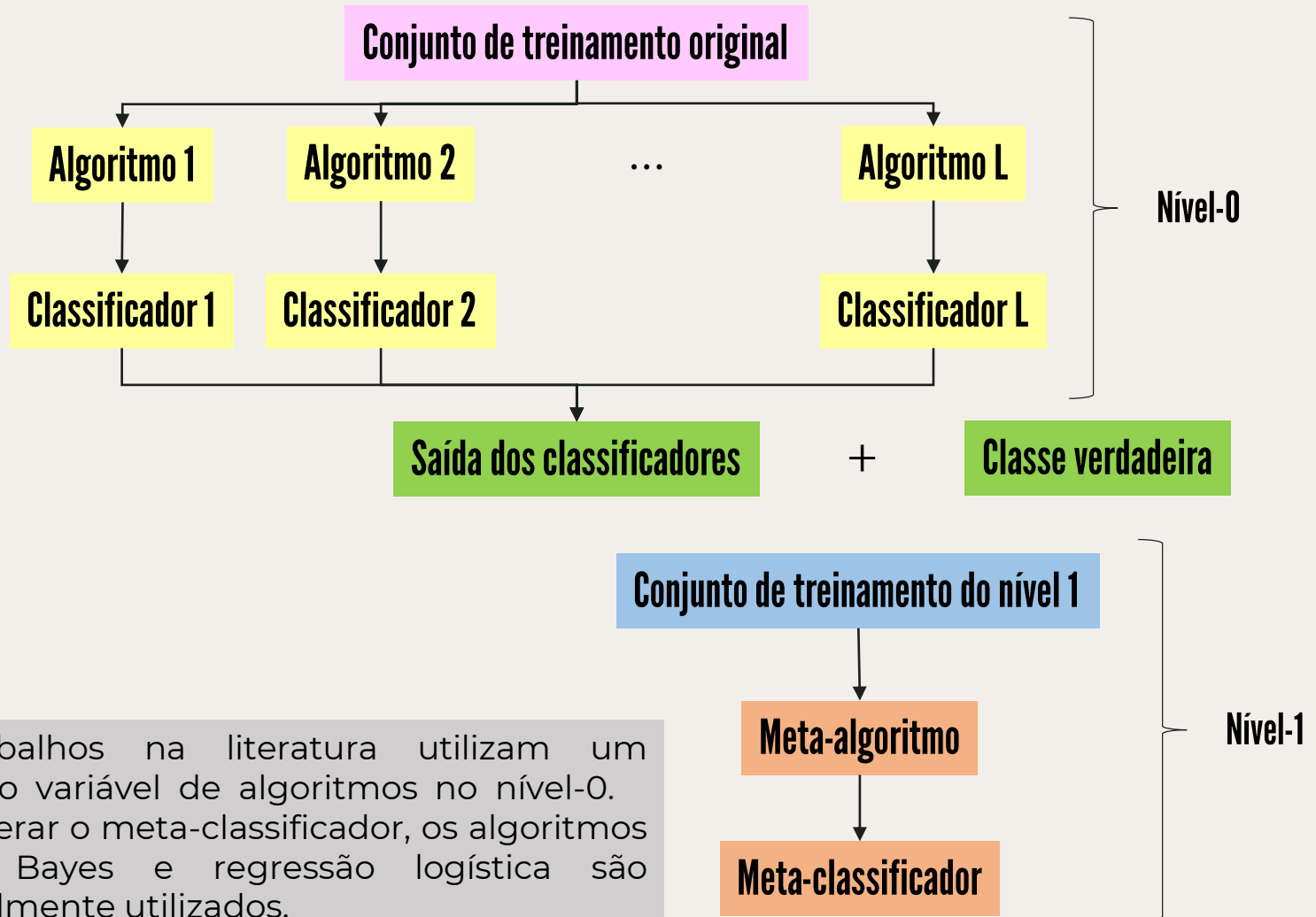
UMA QUANTIDADE GRANDE
DE ÁRVORES PODE TORNAR
O ALGORITMO LENTO E
INEFICIENTE PARA
PREDIÇÕES EM TEMPO REAL.

SÃO MENOS INTUITIVOS DO
QUE AS ÁRVORES DE
DECISÃO.

STACKING (STACKED GENERALIZATION)

- Desenvolvido por Wolpert (1992);
- Estruturado em duas camadas:
 - Nível-0:
 - Vários algoritmos de aprendizado recebem o mesmo conjunto de treinamento, gerando os classificadores de nível-0;
 - Nível-1:
 - Tem como entrada as previsões da camada anterior (nível-0), na qual um meta-algoritmo de nível-1 as combina para fornecer o meta-classificador final h^* .

STACKING



- Os trabalhos na literatura utilizam um número variável de algoritmos no nível-0.
- Para gerar o meta-classificador, os algoritmos *naïve* Bayes e regressão logística são normalmente utilizados.