

# Aprendizado de Máquina

## Aula 7: Ensembles

André C. P. L. F de Carvalho  
ICMC/USP

[andre@icmc.usp.br](mailto:andre@icmc.usp.br)



# Tópicos

- Ensembles
- Combinação de classificadores
  - Viés e variância
  - Boosting
  - Bagging
  - Stacking
  - Ensembles de árvores

# Ensembles

- Procuram melhorar acurácia preditiva combinando predições de múltiplos estimadores
  - Classificação
    - Constroem conjunto de classificadores a partir de dados de treinamento
      - Classificadores base
      - Classe do novo exemplo é definida pela agregação da predição dos múltiplos classificadores base
  - Também podem ser usados em tarefas de regressão e de agrupamento de dados

# Combinação de classificadores

- Condições necessárias para um bom desempenho
  - Diversidade
    - Classificadores base devem ser independentes e com diferentes vieses
      - Ideal: cometer erros diferentes
  - Acurácia preditiva
    - Desempenho dos classificadores base deve ser melhor que classificação aleatória
      - E do que classificar na classe majoritária

# Exemplo

- Sejam 3 classificadores induzidos para os mesmos dados, com acurácia 0.6
  - Se eles cometem os mesmos erros
    - Acurácia do ensemble será 0.6 (taxa de erro = 0.4)
  - Se eles são completamente independentes
    - Ensemble erra classificação apenas se pelo menos 2 classificadores erram na predição

$$erro_{ems} = \sum_{i=2}^3 \binom{3}{i} e^i (1-e)^{3-i}$$

Distribuição Binomial

e: taxa de erro  
(0.35 < 0.4)

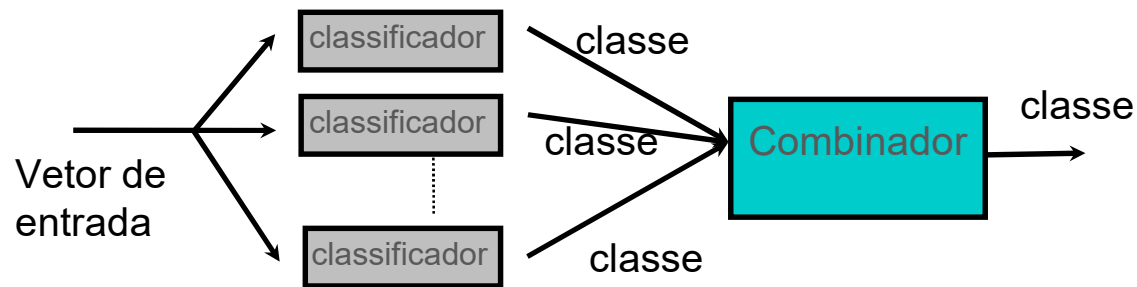
# Estruturas de combinação

- Combinação Paralela
  - Treinamentos independentes
  - Algoritmos aplicados a:
    - Mesmo conjunto dados
    - Conjuntos de dados formados por diferentes amostras do conjunto de dados original
    - Conjuntos de dados com diferentes atributos preditivos do conjunto de dados originais
  - Explora semelhanças e diferenças



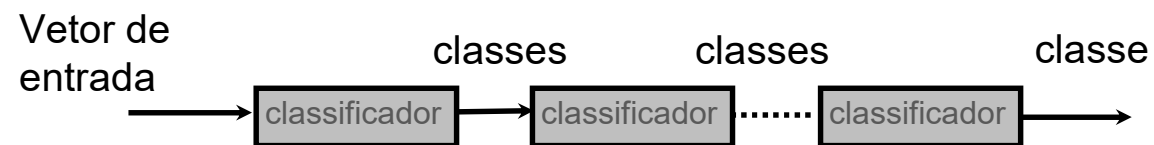
# Estruturas de combinação

- Combinação Paralela



# Estruturas de combinação

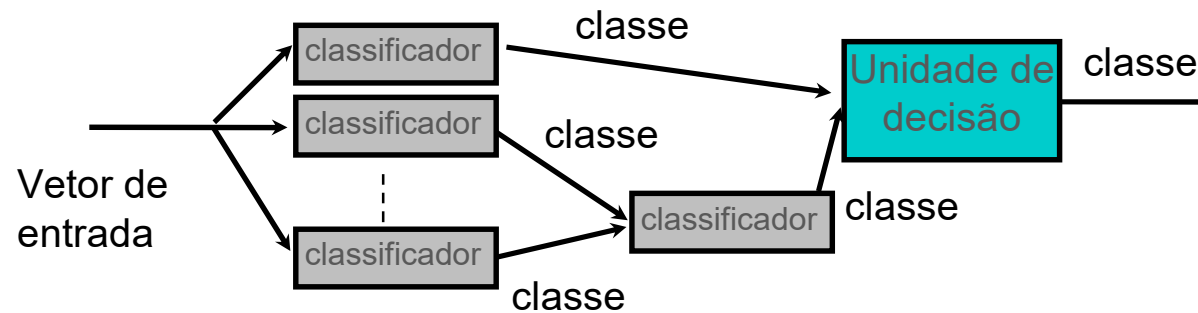
- Combinação em cascata (sequencial)
  - Saída de um classificador é utilizada como entrada para o próximo classificador
  - Não precisa combinar saídas
  - Problema: propagação de erro





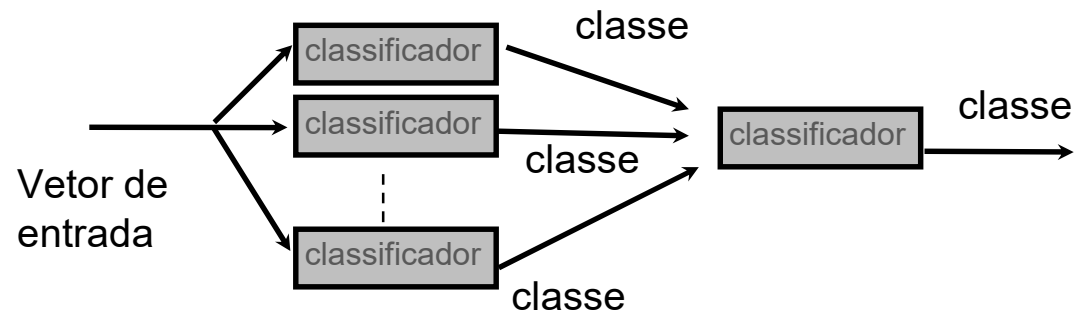
# Estruturas de combinação

- Combinação hierárquica
  - Mistura das combinações anteriores
    - Caso especial: stacking



# Estruturas de combinação

- Combinação hierárquica
  - Mistura das combinações anteriores
    - Caso especial: stacking

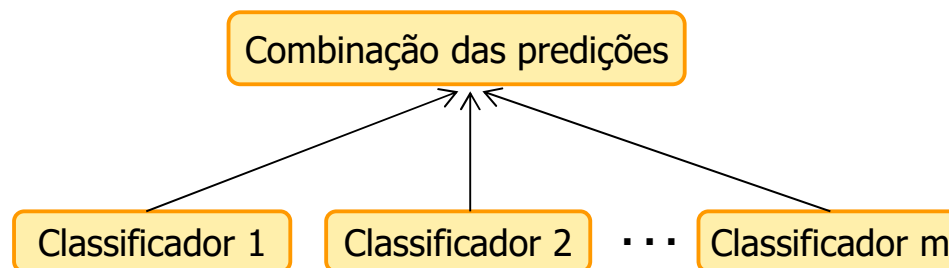


# Combinações paralelas

- Pode ocorrer pela manipulação de:
  - Conjunto de treinamento
    - Boosting e Bagging
  - Atributos preditivos
    - Ensemble de árvores
  - Rótulos das classes
    - Multiclasses e multirrótulo
  - Algoritmo de aprendizado
    - Modelos gerados por algoritmo(s)

# Combinação de previsões

- Voto (média)
- Voto (média) ponderado
- Algoritmo combinador (*stacking*)



# Decomposição viés-variância

- Em geral
  - Quanto mais forte a suposição de um classificador sobre o espaço de decisão, maior seu viés
    - Ex. Árvore podada faz suposição mais forte, por basear a fronteira em menos atributos
      - Menos consistente com os dados de treinamento
      - Tem maior viés (e menor variância)

# Decomposição viés-variância

- Em geral
  - Algoritmo de classificação gera modelos diferentes para mesmo conjunto de dados
  - Variabilidade do conjunto de treinamento leva a variância nos erros de predição
- Erro de modelo é definido por três componentes
  - Viés + variância + ruído



# Bagging (Bootstrap Agregating)

- Cada classificador é induzido por uma amostra diferente do conjunto de treinamento
  - Mesmo tamanho do conjunto original
  - Usa *bootstrap*
- Classe definida por votação
- Tende a reduzir variância associada com os classificadores base

# Bagging (Bootstrap Agregating)

- Indicado para classificadores instáveis
  - Pequena mudança nos dados de treinamento afeta modelo de classificação induzido
  - Redes neurais e árvores de decisão
    - Por que eles são instáveis?
- Não são indicados para classificadores estáveis
  - Erro geralmente causado por viés do classificador base
- Menos sensível a *overfitting* quando dados têm ruído

# Bagging

- Seja o conjunto de dados de treinamento  $\{x_1, x_2, x_3, x_4, x_5, x_6\}$

$x_1, x_6, x_3, x_5, x_3, x_1$

Amostra 1

$x_3, x_4, x_1, x_5, x_5, x_1$

Amostra 2

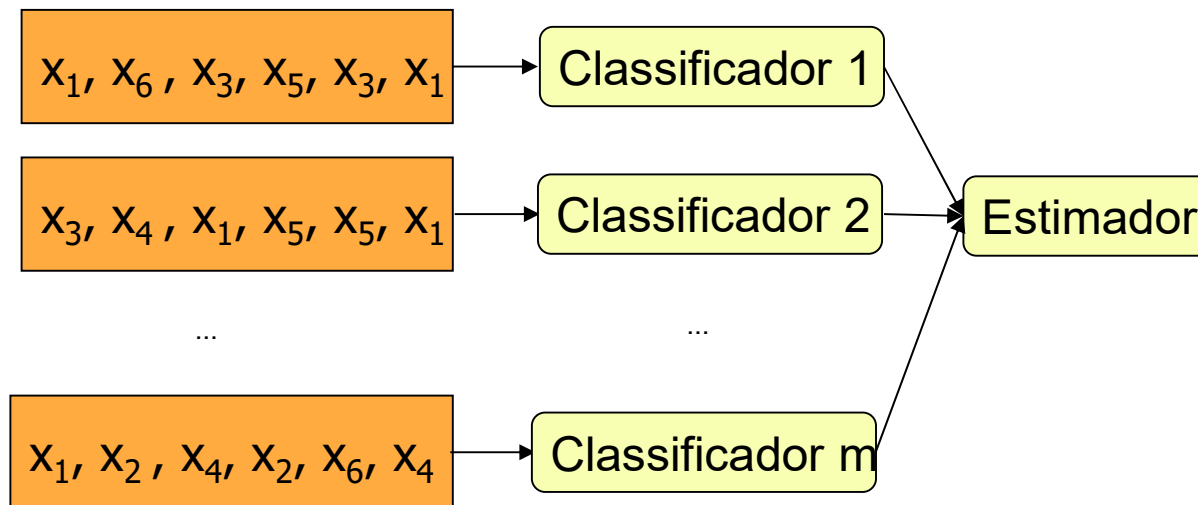
...

...

$x_1, x_2, x_4, x_2, x_6, x_4$

Amostra m

# Bagging



# Boosting

- Conjunto de técnicas
  - Adaboost é uma das mais conhecidas
- A cada iteração
  - Induz um classificador
  - Pondera cada exemplo do conjunto de dados completo pelo desempenho do classificador base
    - Quanto mais difícil de ser aprendido, maior o peso associado ao exemplo
      - Maior probabilidade de ser escolhido na próxima iteração
- Boosting funciona de forma semelhante a minimização por gradiente descendente

# Boosting

- Seja o conj. treinamento  $\{x_1, x_2, x_3, x_4, x_5\}$

Exemplos:	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
Pesos atuais:	0.2	0.2	0.2	0.2	0.2
Classificação:	C	I	C	C	I
Novos pesos:	0.2	0.4	0.2	0.2	0.4

Soma dos pesos = 1.0

C: correta  
I: incorreta

Exemplos:	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
Pesos atuais:	0.2	0.4	0.2	0.2	0.4
Classificação:	C	I	C	I	C
Novos pesos:	0.2	0.6	0.2	0.4	0.4

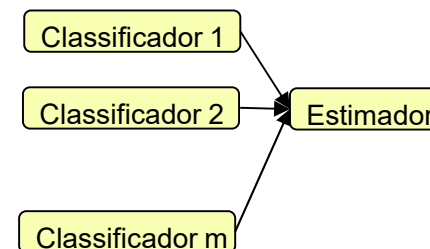


# Boosting

- Indicado para classificadores base fracos
  - Acurácia ligeiramente melhor que palpite aleatório
- Convergência rápida
- Pouco indicado para dados com ruídos e pequenos conjuntos de dados
  - Por focar em exemplos difíceis de serem classificados

# Stacking

- Um algoritmo estimador aprende a combinar previsões de modelos base
  - Modelos gerados por algoritmos base
  - Saídas combinadas por algoritmo estimador
    - Algoritmo de AM
- Algoritmos base podem ser:
  - Homogêneos
  - Heterógenos

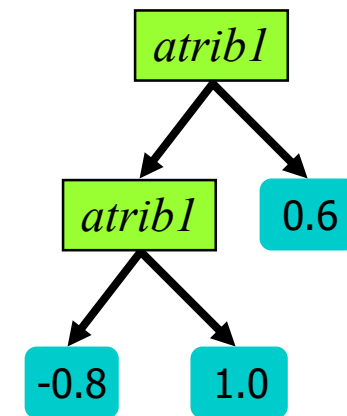


# Ensembles de ADs

- Combina a predição de várias árvores de decisão
- Duas principais abordagens:
  - Random forests
  - Extreme Gradient Boosting
  - Algoritmo CART ou baseado no CART

# Algoritmo CART

- Classification and regression trees
  - Árvore binária
  - Que pode ser usado para regressão ou classificação
    - Para classificação, cada nó folha possui uma classe
    - Para regressão, cada nó folha possui um valor real



# Random Forests (RFs)

- Combinar ADs, mas pode usar modelos gerados por qualquer algoritmo de AM
- Combina k ADs
  - Cada árvore é induzida usando um subconjunto aleatório dos atributos preditivos
    - Usado na escolha do atributo preditivo para cada nó
    - Hiper-parâmetros definem número de ADs e número de atributos preditivos para cada AD
  - Classificação ocorre por voto majoritário

# Algoritmo de treinamento

For  $i = 1$  to número de árvores:

- Extrair por bootstrap uma amostra dos dados de treinamento

- Construir uma árvore, repetindo de forma recursiva até um dado critério de parada (número de objetos no nó)

- Selecionar aleatoriamente  $m$  dos  $M$  atributos

- Escolher o melhor destes  $m$  atributos para dividir um nó

- Dividir o nó em dois nós filhos

Resultado é um ensemble de ADs

Uma predição para um novo objeto retorna:

- Média dos resultados, para regressão

- Classe mais votada, para classificação




# Random Forest Classifier

## Dados de treinamento

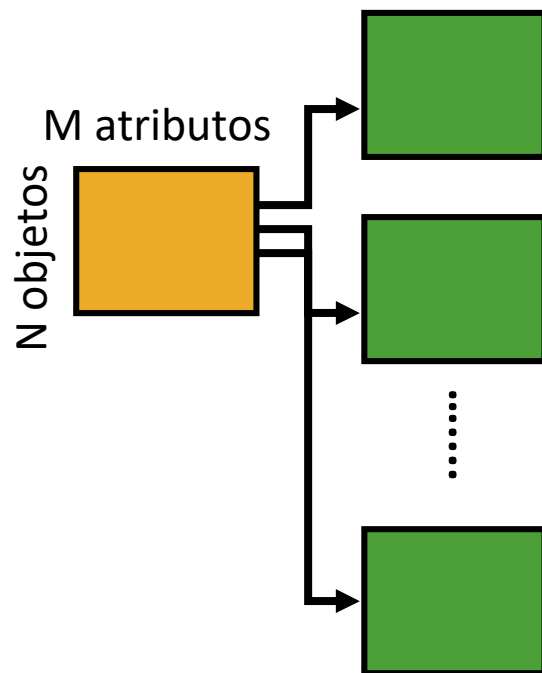
M atributos

N objetos



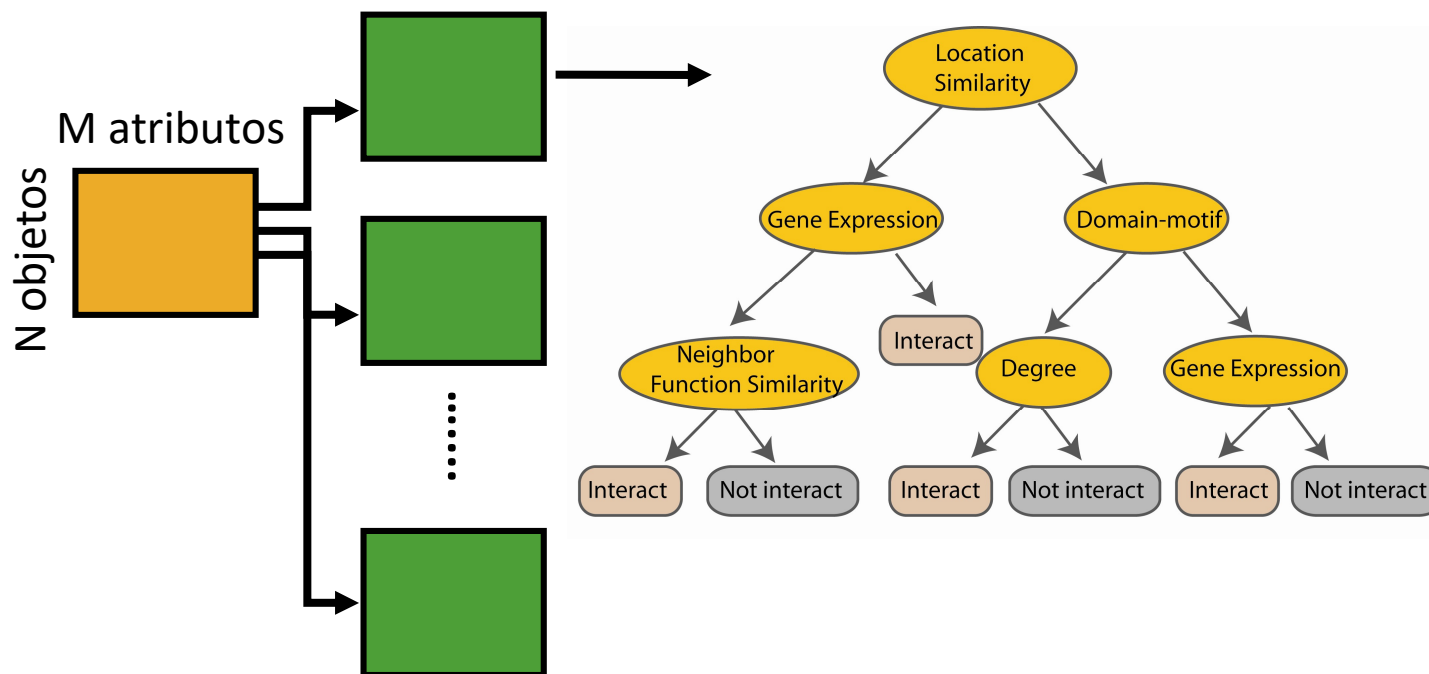
# Classificador Random Forest

Cria amostras do conjunto de treinamento usando bootstrap

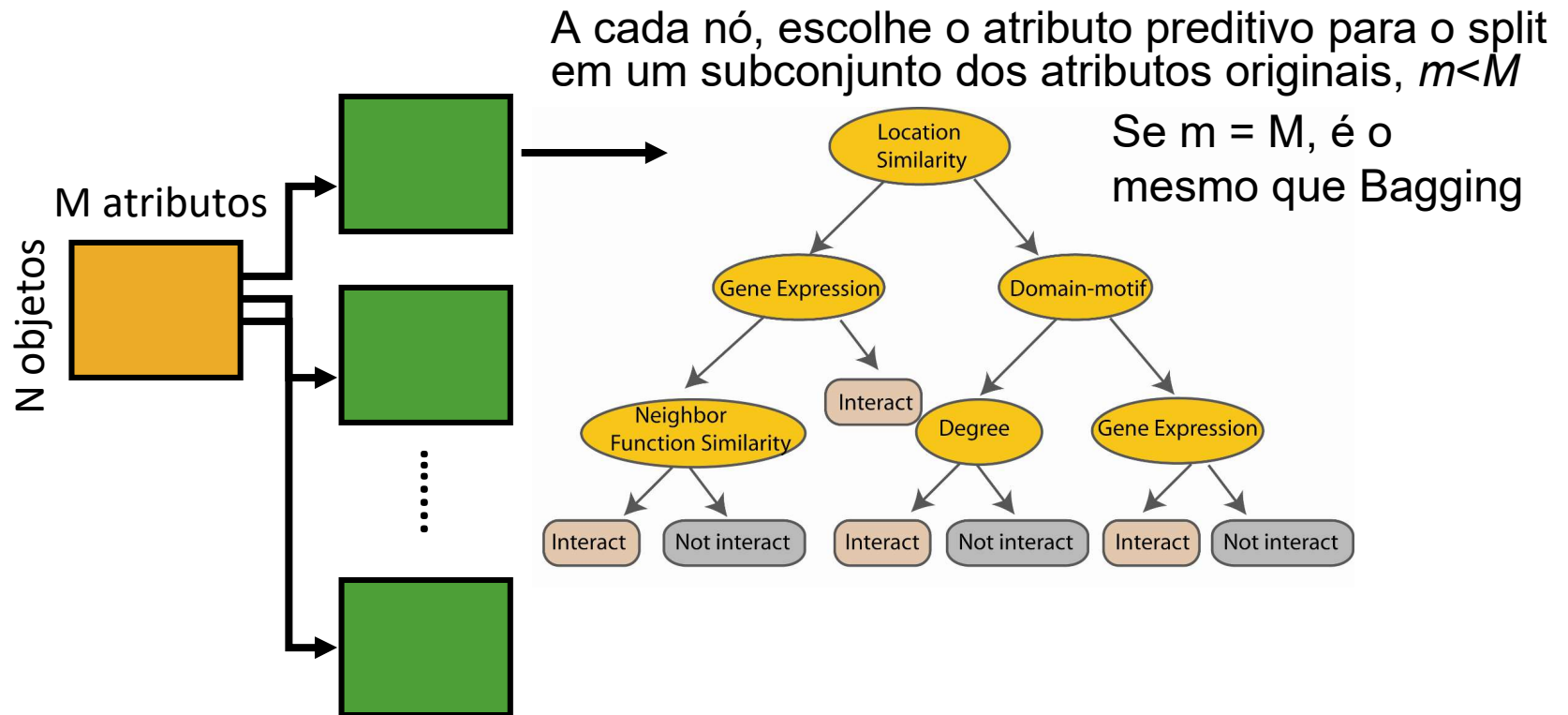


# Classificador Random Forest

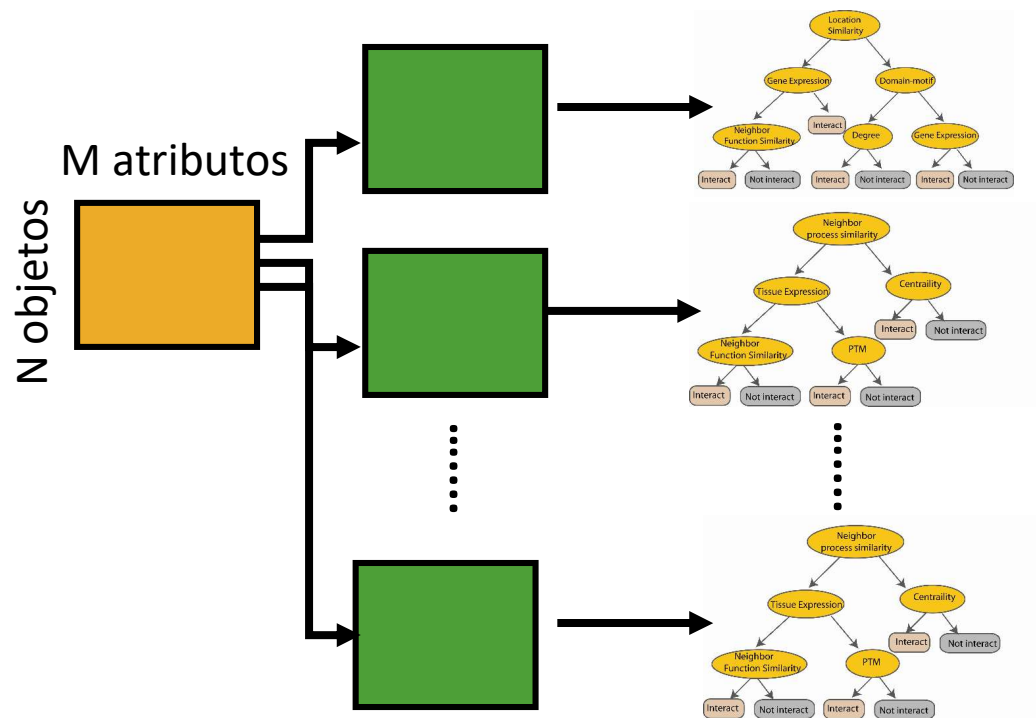
Constrói uma árvore de decisão



# Classificador Random Forest

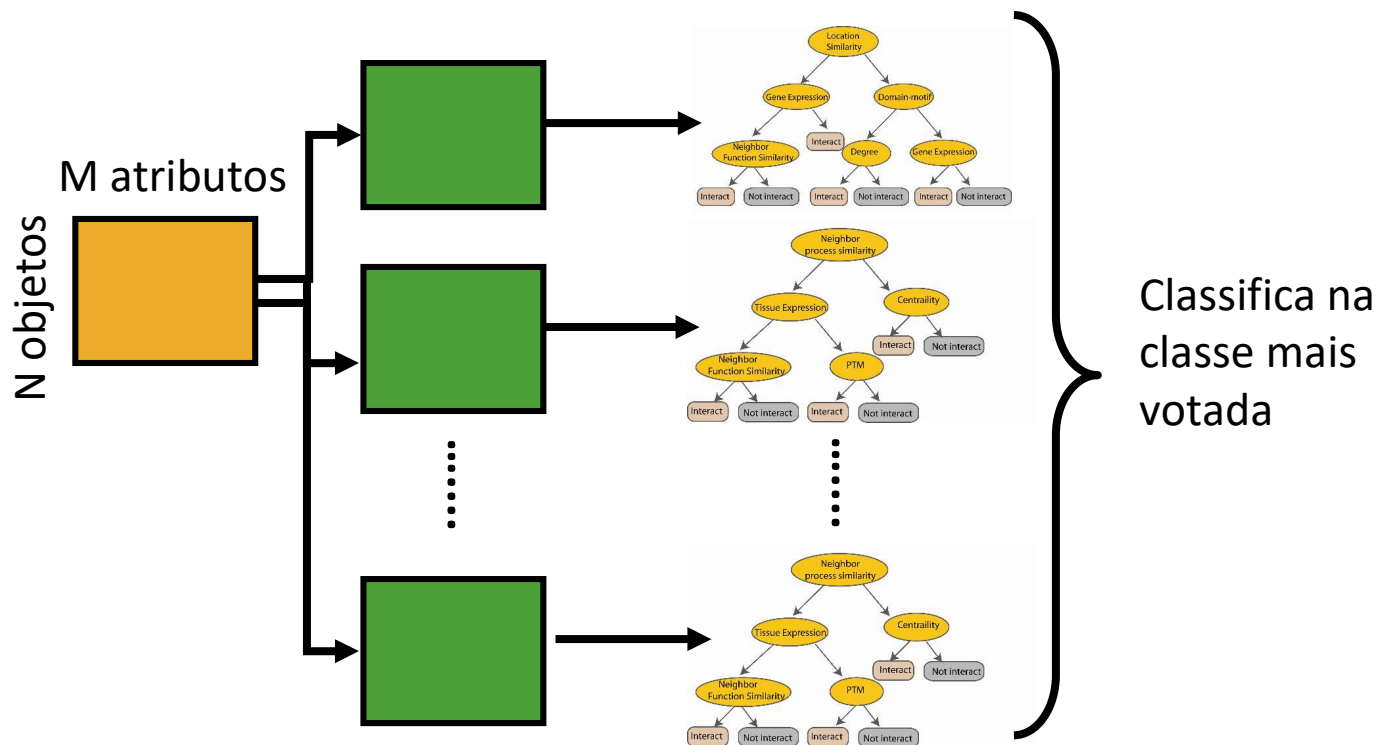


# Classificador Random Forest



Cria uma árvore de decisão de cada amostra de bootstrap

# Classificador Random Forest





# Random Forests (RFs)

- *Bagging* pode ser visto como um caso especial de RFs
  - RFs usa *bootstrap* de forma similar a *bagging* para selecionar exemplos de treinamento
- Várias alternativas para escolher aleatoriamente os atributos preditivos
  - Forest-RI (Random Input Selection)
  - Forest-RC (Random Combination)

# Random Forests (RFs)

- Forest – RI (*Random Input Selection*)
  - Seleciona aleatoriamente, para cada nó, um subconjunto de  $m$  atributos preditivos
  - Algoritmo CART é usado para crescer as árvores sem poda
  - Problema: conjunto de dados com poucos atributos preditivos
    - Pode selecionar atributos fortemente correlacionados

# Random Forests (RFs)

- Forest-RC (*Random Combination*)
  - Expande número de atributos criando combinações lineares aleatórias de atributos
    - A cada nó,  $m'$  combinações de  $m$  atributos são aleatoriamente geradas
      - Combina atributos utilizando pesos aleatoriamente gerados entre -1 e +1
      - Cada combinação é um novo atributo
  - Usada quando conjunto de dados tem poucos atributos preditivos

# Extreme Gradient Boosting

- XGBoost
- Combina árvores geradas pelo algoritmo CART
- Treinamento aditivo
  - Induz uma árvore
    - Inclui ela no ensemble
      - Induz próxima árvore
    - ...
- Pondera a resposta de cada árvore para reduzir complexidade do modelo final

# Observações

- Boosting de árvores de decisão (C4.5) muitas vezes funciona muito bem
- Uma árvore de decisão de nível 2~3 tem um bom equilíbrio entre eficácia e eficiência
- Random forest requer menos tempo de treinamento que outros algoritmos que apresentam desempenho semelhante
- XGBoost e random forest podem ser usados em tarefas de regressão

# Conclusão

- Combinação de estimadores em geral aumenta desempenho preditivo
  - E reduz variância
- As vezes chamado de meta-aprendizado
- Regressão
  - Média simples ou ponderada

Fim do  
apresentação