

# Aprendizado Supervisionado de Máquina

## Métodos de Classificação

João Carlos Xavier Júnior

[jcxavier@imd.ufn.br](mailto:jcxavier@imd.ufn.br)

# Treinamento e Teste

- ❑ Como treinar e testar os algoritmos?



<http://iasdcentralcampinas.org.br/treinamentos-apac/>

---

# Treinamento e Teste

- ❑ **Validação cruzada:** é uma técnica para avaliar a **capacidade de generalização** de um modelo, a partir de um conjunto de dados.
- ❑ Empregada em **problemas de predição**.
- ❑ Busca-se então estimar o quão **acurado** é este modelo na prática, ou seja, o seu desempenho para um **novo conjunto de dados**.
- ❑ Particiona o conjunto de dados em subconjuntos mutualmente exclusivos. Formas: *holdout* e *k-fold*.

---

# Treinamento e Teste

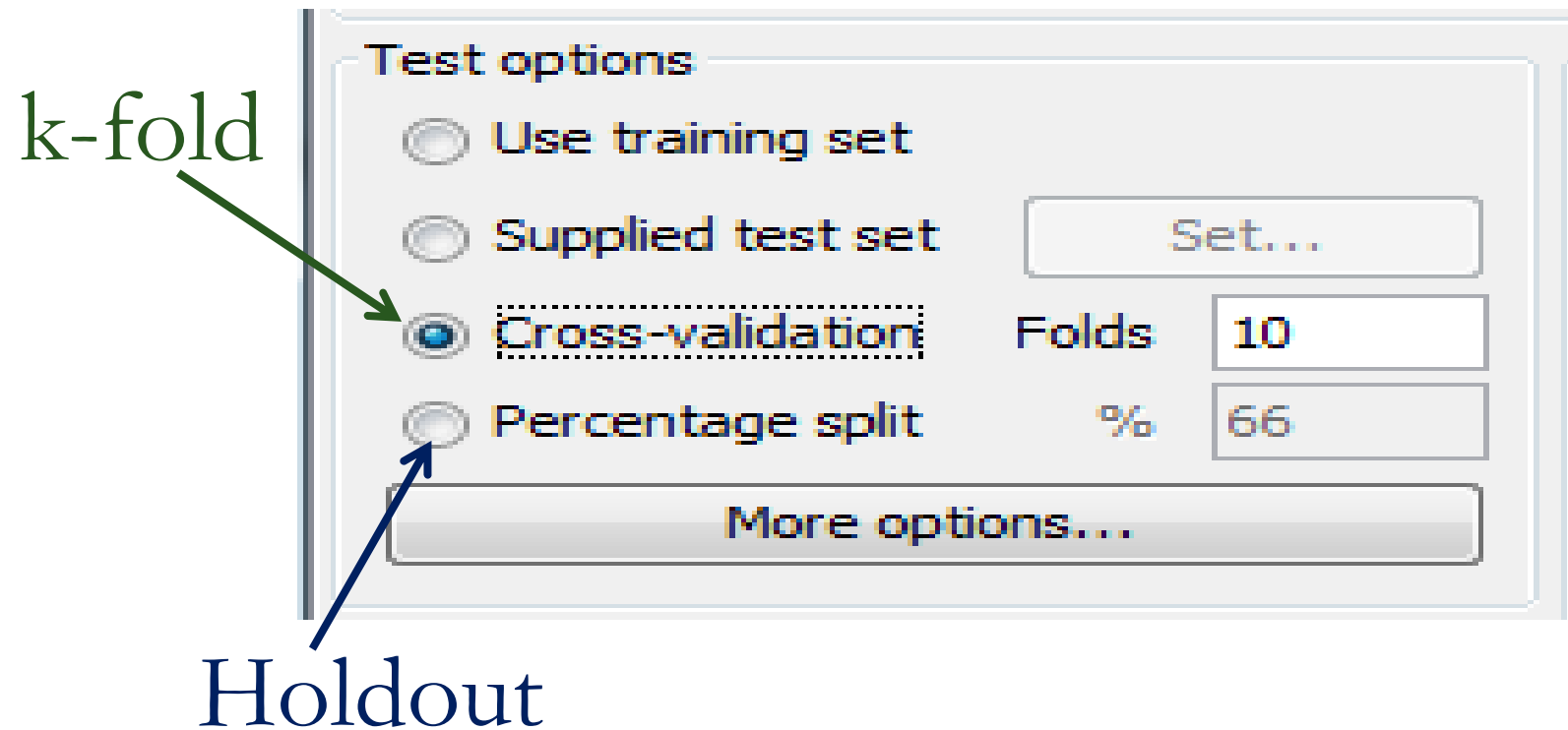
- ❑ **Método holdout:** consiste em dividir o conjunto total de dados em dois subconjuntos mutuamente exclusivos, um para **treinamento** e outro para **teste** (validação).
- ❑ O conjunto de dados pode ser **separado** em **quantidades** iguais ou não. É possível ter  $2/3$  dos dados para **treinamento** e o  $1/3$  restante para **teste**.
- ❑ Esta abordagem é indicada para **grande quantidade** de dados. Quando o conjunto é **pequeno**, o **erro calculado** na predição pode sofrer **muita variação**.

---

# Treinamento e Teste

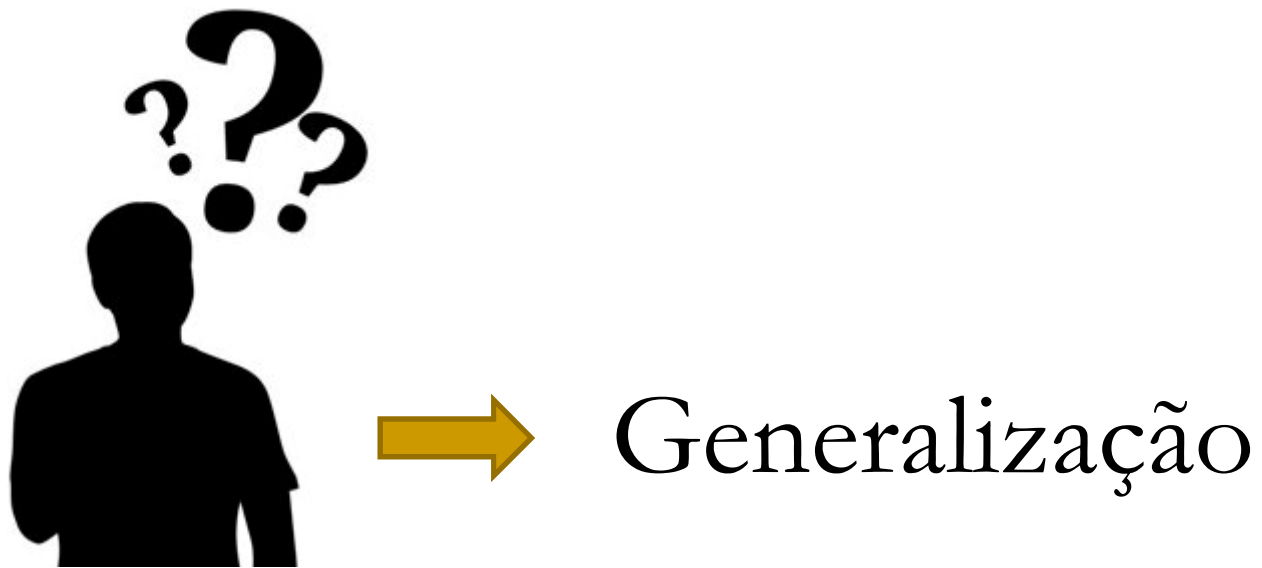
- ❑ **Método k-fold:** consiste em dividir o conjunto total de dados em  $k$  subconjuntos mutuamente exclusivos do mesmo tamanho.
- ❑ Um subconjunto é utilizado para teste e os  $k - 1$  restantes são utilizados para estimação dos parâmetros e calcula-se a acurácia do modelo.

# Treinamento e Teste



# Treinamento

- ❑ O que se busca alcançar com treinamento?



- ❑ **Generalização:** a habilidade de classificar padrões de teste que não foram utilizados durante o treinamento.

# O que é classificação?

- ❑ **Dado um conjunto de registros** (*dataset*):
  - Cada registro contém um conjunto de **atributos**, em que um dos atributos é o **atributo-meta** (**variável resposta**).
  - O conjunto de dados é dividido em dois subconjuntos: **conjunto de treinamento** para **construir o modelo** e **conjunto de teste** para **validar o modelo**.
- ❑ **Passo 1:** encontrar um modelo para o atributo-meta (ou atributo-classe) como uma função dos valores dos outros atributos.
- ❑ **Passo 2:** registros não conhecidos devem ser associados à classe com a maior precisão possível.



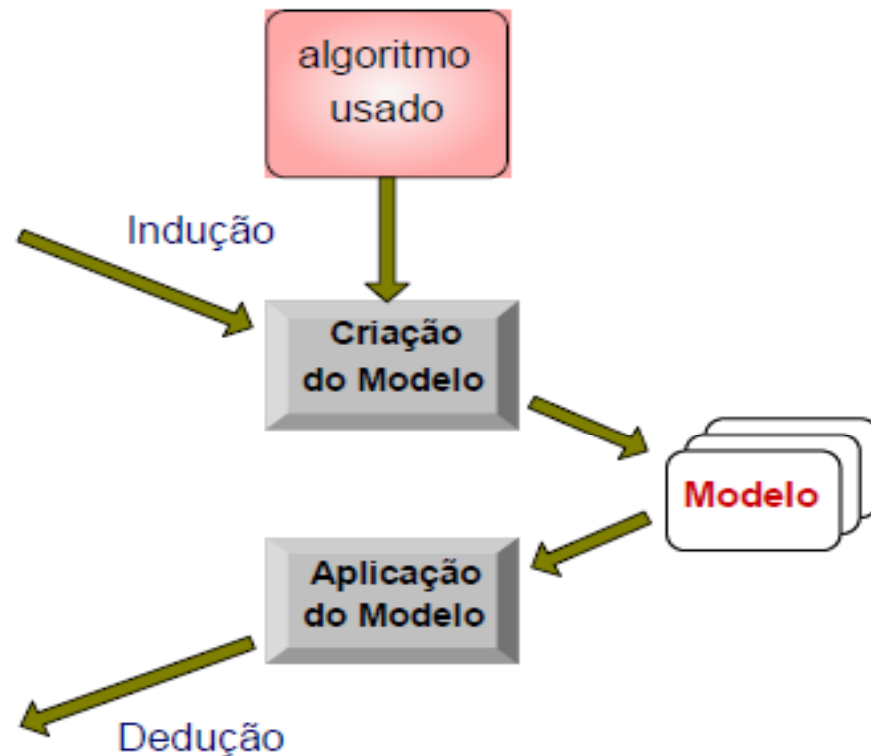
# O que é Classificação?

Tid	Atrib1	Atrib2	Atrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Conjunto de treinamento

Tid	Atrib1	Atrib2	Atrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Conjunto de Teste



# Onde aplicar Classificação?

- ❑ Classificar tumores como benigno ou maligno.
- ❑ Classificar transações de cartão de crédito como legítima ou fraudulenta.
- ❑ Classificar estruturas secundárias de proteínas como *alpha-helix*, *beta-sheet* ou *random coil*.
- ❑ Avaliar riscos de empréstimos, previsão de tempo, etc.
- ❑ Sistema de alerta de geada.
- ❑ Qualquer sistema que **tome decisão**.

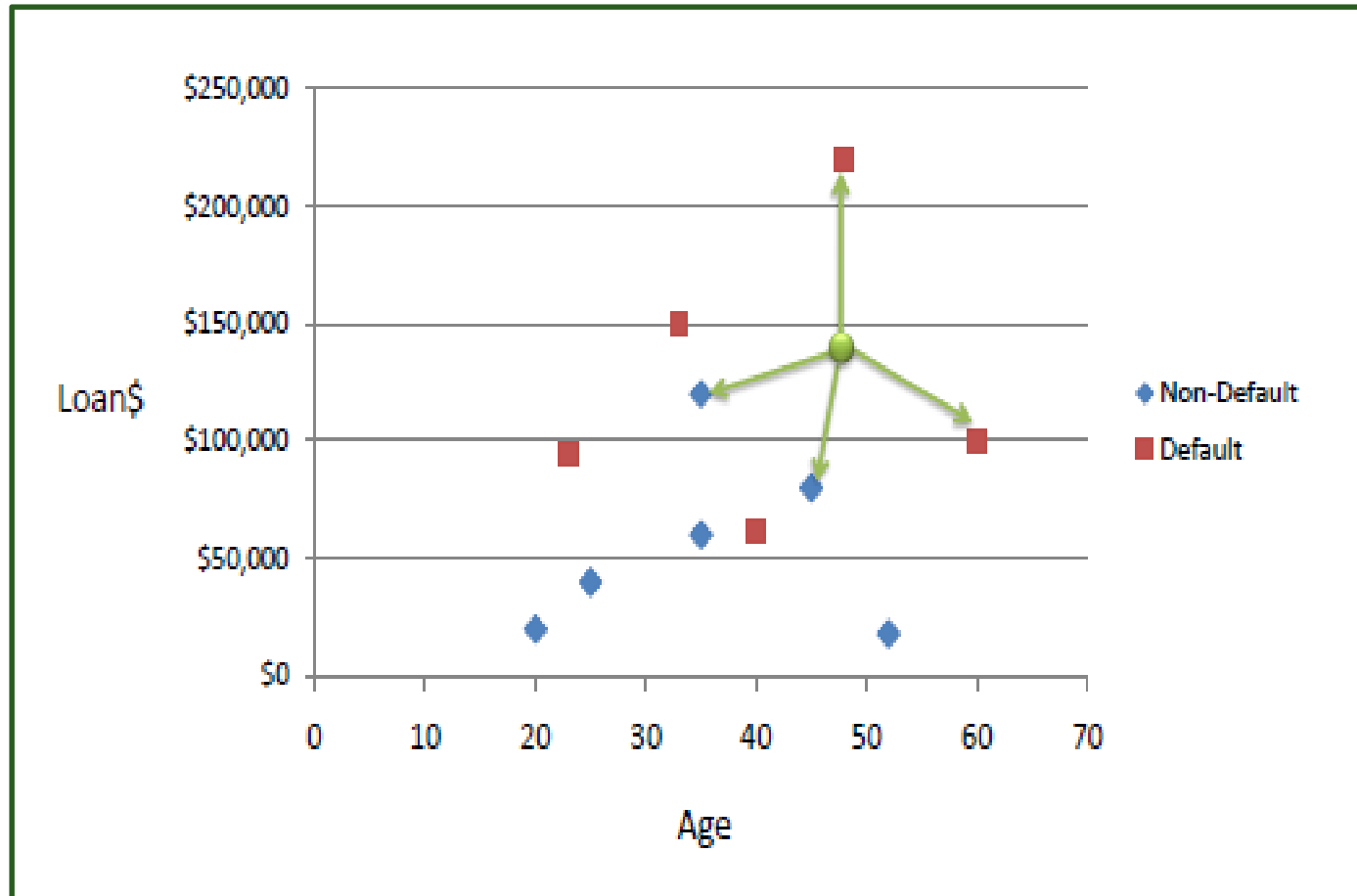
---

# Boas Características - Classificador

- ❑ Precisão
- ❑ Velocidade
  - Tempo para construir o modelo.
  - Tempo para usar o modelo.
- ❑ Robustez
  - Capacidade de lidar com ruídos e valores faltantes (missing).
- ❑ Escalabilidade
  - Eficiência em banco de dados residentes em disco.
- ❑ Interpretabilidade
  - Clareza fornecida pelo modelo.

# k-NN (k Nearest Neighbor)

# k-NN (k Nearest Neighbor)



[http://www.saedsayad.com/k\\_nearest\\_neighbors.htm](http://www.saedsayad.com/k_nearest_neighbors.htm)

---

# k-NN (k Nearest Neighbor)

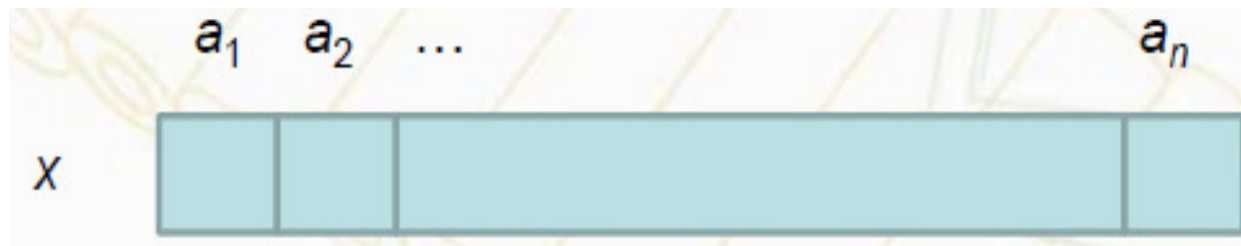
- ❑ Algoritmo de aprendizado mais simples.
- ❑ Algoritmo baseado em Instâncias.
- ❑ Este algoritmo supõe que todos os padrões (instâncias) são pontos no espaço n-dimensional  $R^n$ .
- ❑ Os vizinhos mais próximos de um padrão são definidos em termos da **distância Euclidiana**.
- ❑ A regra dos vizinhos mais próximos: classificar um ponto  $x$  atribuindo a ele o rótulo mais frequente dentre as  $k$  amostras mais próximas (esquema de votação).

# k-NN (k Nearest Neighbor)

- Vamos considerar uma instância arbitrária  $x$  que é descrita pelo vetor de características:

$$x = \langle a_1(x), a_2(x), \dots, a_n(x) \rangle$$

onde  $a_r(x)$  representa o valor do  $r$ -ésimo atributo da instância  $x$ .



<http://www.ppgia.pucpr.br/~alekoe/AM/2013/>

# k-NN (k Nearest Neighbor)

- Então a distância Euclidiana entre duas instâncias  $x_i$  e  $x_j$  é definida como  $d(x_i, x_j)$ , onde:

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^n (a_r(x_i) - a_r(x_j))^2}$$



# k-NN (k Nearest Neighbor)

## ❑ Algoritmo de Treinamento:

- Não há um treinamento explícito.
  - Classificam exemplos nunca vistos por meio de exemplos similares conhecidos.
  - Os próprios padrões são utilizados como base para a resposta do k-NN.
  - Método é denominado de *lazy*, pois necessitam manter os exemplos na memória para classificar novos exemplos.

## ❑ Para cada padrão de treinamento $\langle \mathbf{x}, f(\mathbf{x}) \rangle$ , faça:

- Adicione o exemplo a lista de exemplos\_de\_treinamento.

# k-NN (k Nearest Neighbor)

## □ Algoritmo de Classificação:

- Dado um padrão (instância) de consulta  $\mathbf{x}_q$  a ser classificado.
  - Seja  $\mathbf{x}_1, \dots, \mathbf{x}_k$  as  $k$  instâncias (padrões) do exemplos\_de\_treinamento que são mais próximos a  $\mathbf{x}_q$ .
  - Retorne a **classe** mais comum **entre os vizinhos**.

# k-NN (k Nearest Neighbor)

□ Um exemplo:

# Inst	Attr 1	Attr 2	Attr 3	Class
1	0,50	0,80	0,90	A
2	0,40	0,50	0,40	B
3	0,20	0,50	0,15	C
4	0,50	0,40	0,60	B
5	0,80	0,75	0,87	A

- Dado um exemplo:  $(0.67; 0.75; 0.58)$ , é fornecido que  $k=3$ .
- A qual classe este padrão pertence?

# k-NN (k Nearest Neighbor)

□ Para os cinco padrões ficaria:

$$\begin{aligned} 1: & \sqrt{(0.5-0.67)^2 + (0.8-0.75)^2 + (0.9-0.58)^2} = \sqrt{0.0289+0.025+0.1024} = \sqrt{0.1338} = 0.37 \\ 2: & \sqrt{(0.4-0.67)^2 + (0.5-0.75)^2 + (0.4-0.58)^2} = \sqrt{0.0729+0.0625+0.0324} = \sqrt{0.1678} = 0.41 \\ 3: & \sqrt{(0.2-0.67)^2 + (0.5-0.75)^2 + (0.15-0.58)^2} = \sqrt{0.2209+0.2025+0.1849} = \sqrt{0.6083} = 0.68 \\ 4: & \sqrt{(0.5-0.67)^2 + (0.4-0.75)^2 + (0.6-0.58)^2} = \sqrt{0.0289+0.1225+0.0004} = \sqrt{0.1518} = 0.39 \\ 5: & \sqrt{(0.8-0.67)^2 + (0.75-0.75)^2 + (0.87-0.58)^2} = \sqrt{0.0169+0+0.0841} = \sqrt{0.101} = 0.32 \end{aligned}$$

Os três vizinhos mais próximos

# k-NN (k Nearest Neighbor)

❑ Para os cinco padrões ficaria:

$$1: \sqrt{(0.5-0.67)^2 + (0.8-0.75)^2 + (0.9-0.58)^2} = \sqrt{0.0289+0.025+0.1024} = \sqrt{0.1338} = 0.37$$

$$2: \sqrt{(0.4-0.67)^2 + (0.5-0.75)^2 + (0.4-0.58)^2} = \sqrt{0.0729+0.0625+0.0324} = \sqrt{0.1678} = 0.41$$

$$3: \sqrt{(0.2-0.67)^2 + (0.5-0.75)^2 + (0.15-0.58)^2} = \sqrt{0.2209+0.2025+0.1849} = \sqrt{0.6083} = 0.68$$

$$4: \sqrt{(0.5-0.67)^2 + (0.4-0.75)^2 + (0.6-0.58)^2} = \sqrt{0.0289+0.1225+0.0004} = \sqrt{0.1518} = 0.39$$

$$5: \sqrt{(0.8-0.67)^2 + (0.75-0.75)^2 + (0.87-0.58)^2} = \sqrt{0.0169+0+0.0841} = \sqrt{0.101} = 0.32$$



**Classe A**

# k-NN (k Nearest Neighbor)

- ❑ E se houvesse empate, o que fazer?
- ❑ No exemplo anterior, se  $k = 4$ , haveria um empate.
- ❑ Solução:
  - Ponderar a contribuição de cada um dos  $k$  vizinhos de acordo com a distância do ponto  $\mathbf{x}_q$ ;
  - Maior peso para os vizinhos mais próximos;
  - Como ficaria a resposta para o exemplo anterior?

# k-NN (k Nearest Neighbor)

$$\begin{aligned} 1: & \sqrt{(0.5-0.67)^2 + (0.8-0.75)^2 + (0.9-0.58)^2} = \sqrt{0.0289+0.0025+0.1024} = \sqrt{0.1338} = 0.37 \\ 2: & \sqrt{(0.4-0.67)^2 + (0.5-0.75)^2 + (0.4-0.58)^2} = \sqrt{0.0729+0.0625+0.0324} = \sqrt{0.1678} = 0.41 \\ 3: & \sqrt{(0.2-0.67)^2 + (0.5-0.75)^2 + (0.15-0.58)^2} = \sqrt{0.2209+0.2025+0.1849} = \sqrt{0.6083} = 0.68 \\ 4: & \sqrt{(0.5-0.67)^2 + (0.4-0.75)^2 + (0.6-0.58)^2} = \sqrt{0.0289+0.1225+0.0004} = \sqrt{0.1518} = 0.39 \\ 5: & \sqrt{(0.8-0.67)^2 + (0.75-0.75)^2 + (0.87-0.58)^2} = \sqrt{0.0169+0+0.0841} = \sqrt{0.101} = 0.32 \end{aligned}$$

# k-NN (k Nearest Neighbor)

$$\begin{aligned} 1: & \sqrt{(0.5-0.67)^2 + (0.8-0.75)^2 + (0.9-0.58)^2} = \sqrt{0.0289+0.025+0.1024} = \sqrt{0.1338} = 0.37 \\ 2: & \sqrt{(0.4-0.67)^2 + (0.5-0.75)^2 + (0.4-0.58)^2} = \sqrt{0.0729+0.0625+0.0324} = \sqrt{0.1678} = 0.41 \\ 3: & \sqrt{(0.2-0.67)^2 + (0.5-0.75)^2 + (0.15-0.58)^2} = \sqrt{0.2209+0.2025+0.1849} = \sqrt{0.6083} = 0.68 \\ 4: & \sqrt{(0.5-0.67)^2 + (0.4-0.75)^2 + (0.6-0.58)^2} = \sqrt{0.0289+0.1225+0.0004} = \sqrt{0.1518} = 0.39 \\ 5: & \sqrt{(0.8-0.67)^2 + (0.75-0.75)^2 + (0.87-0.58)^2} = \sqrt{0.0169+0+0.0841} = \sqrt{0.101} = 0.32 \end{aligned}$$

Classe A

Classe B



---

# Overview

- ❑ Métodos de aprendizagem baseados em instâncias **não** necessitam formar uma **hipótese explícita** da função alvo sobre o espaço das instâncias.
- ❑ Eles formam uma aproximação local da função alvo para cada nova instância a “classificar”.
- ❑ O **k-NN** é um algoritmo baseado em instâncias para aproximar funções alvo de valor real ou de valor discreto, assumindo que as instâncias correspondem a pontos em um espaço  $d$ -dimensional.

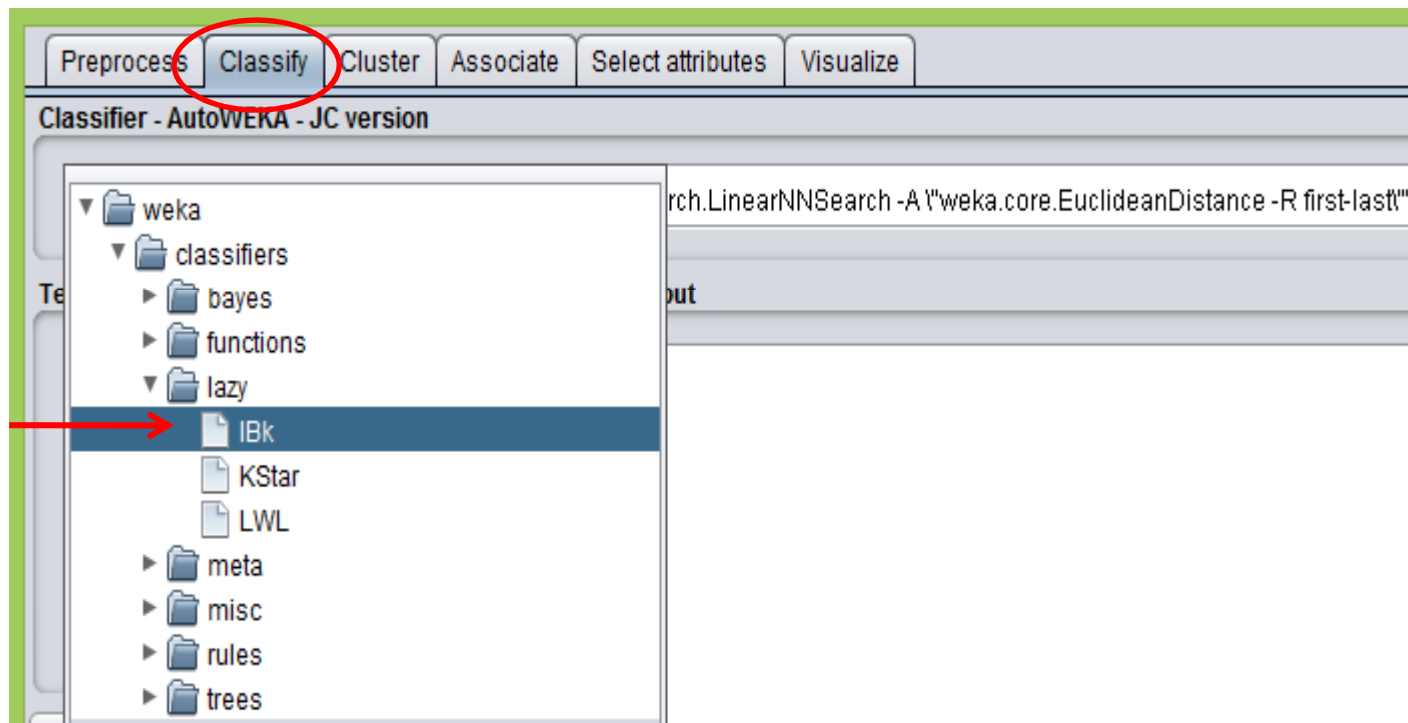
---

# Overview

- ❑ O valor da função alvo para um novo ponto é estimada a partir dos valores conhecidos dos  $k$  exemplos de treinamento mais próximos.
- ❑ **Vantagens:**
  - Habilidade para modelar funções alvo complexas por uma coleção de aproximações locais menos complexas.
  - A informação presente nos exemplos de treinamento nunca é perdida.
- ❑ **Dificuldades:**
  - Tempo?

# k-NN

- ❑ Utilizando **IBK** (WEKA):



# k-NN

## ❑ Configurando o IBK:

weka.classifiers.lazy.IBk

About

K-nearest neighbours classifier.

More

Capabilities

KNN 1

batchSize 100

crossValidate False

debug False

distanceWeighting No distance weighting

doNotCheckCapabilities False

meanSquared False

nearestNeighbourSearchAlgorithm Choose LinearNNSearch -A "weka.core.EuclideanDistance -R

# k-NN

## ❑ Configurando treinamento e teste:


**Test options**

☐ Use training set

☐ Supplied test set

☒ Cross-validation Folds

☐ Percentage split %




**Test options**

☐ Use training set

☐ Supplied test set

☐ Cross-validation Folds

☒ Percentage split %



# k-NN

## ❑ Analisando os resultados....

```
Classifier output

=== Run information ===

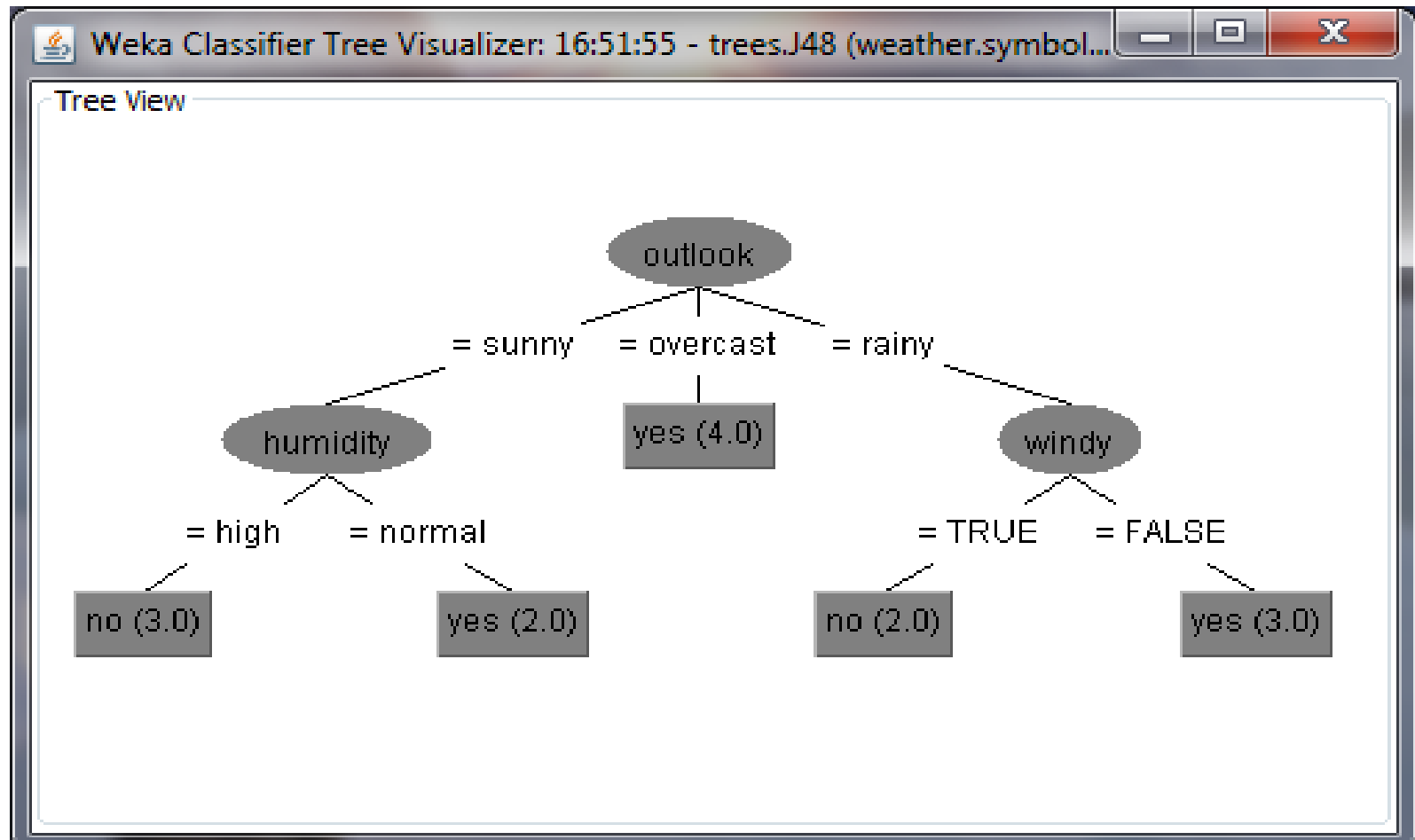
Scheme:      weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.
Relation:    Iris
Instances:    150
Attributes:   5
              a0
              a1
              a2
              a3
              class
Test mode:    split 70.0% train, remainder test
```

```
=== Summary ===
```

Correctly Classified Instances	43	95.5556 %
Incorrectly Classified Instances	2	4.4444 %
Kappa statistic	0.9331	
Mean absolute error	0.0409	
Root mean squared error	0.1702	
Relative absolute error	9.1927 %	
Root relative squared error	36.0807 %	
Total Number of Instances	45	

# Árvore de Decisão

# Árvores de Decisão





# Árvore de Decisão

## ❑ Definição:

- Um fluxograma com a estrutura de uma árvore.
- Nó interno representa um testes sobre um atributo.
- Cada ramo representa um resultado do teste.
- Folhas representam as classes.

## ❑ A geração de uma árvore consiste de duas fases:

- Construção da árvore (particionamento de atributos).
- Fase da poda (identifica e remove ruídos ou outliers).

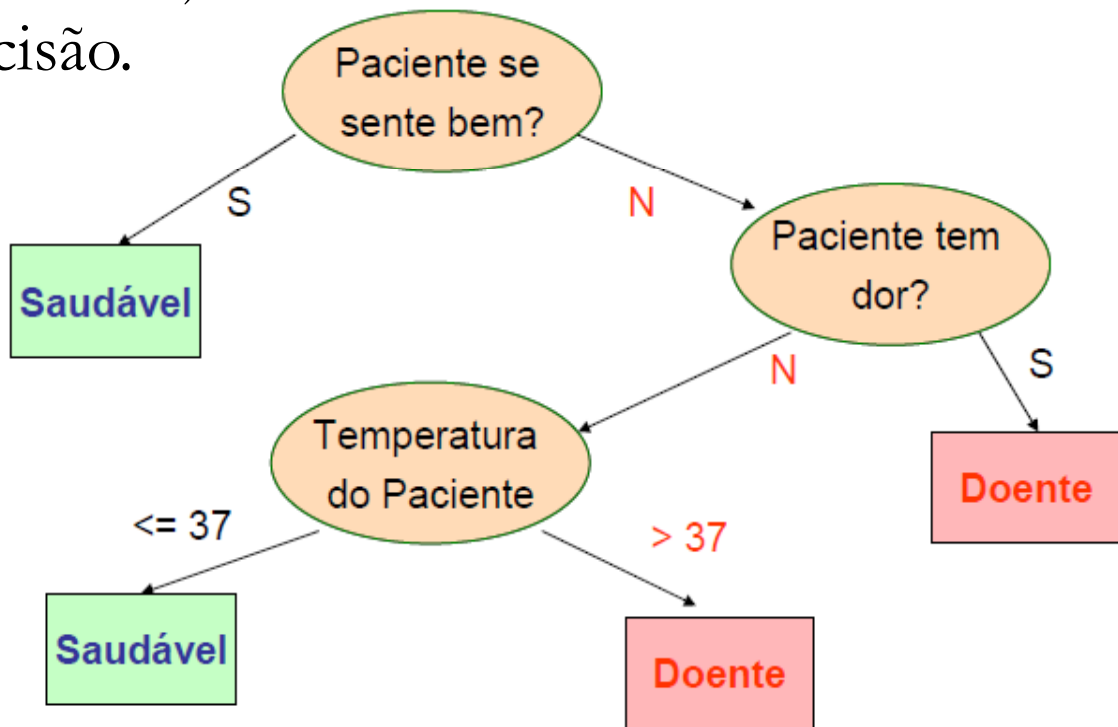
## ❑ Uso da árvore: classificação de amostras desconhecidas.

- Testa os valores dos atributos da amostra “contra” a árvore.

# Árvore de Decisão

## ❑ Funcionamento:

- Lista de perguntas → respostas “**sim**” ou “**não**”.
- Hierarquicamente arranjadas.
- Levam a uma decisão.



# Árvore de Decisão

## ❑ Geração de regras:

Se (paciente está bem = sim) então  
    classe = **saudável**

Senão

    Se (paciente tem dor = sim) então  
        classe = **doente**

    Fimse

Fimse

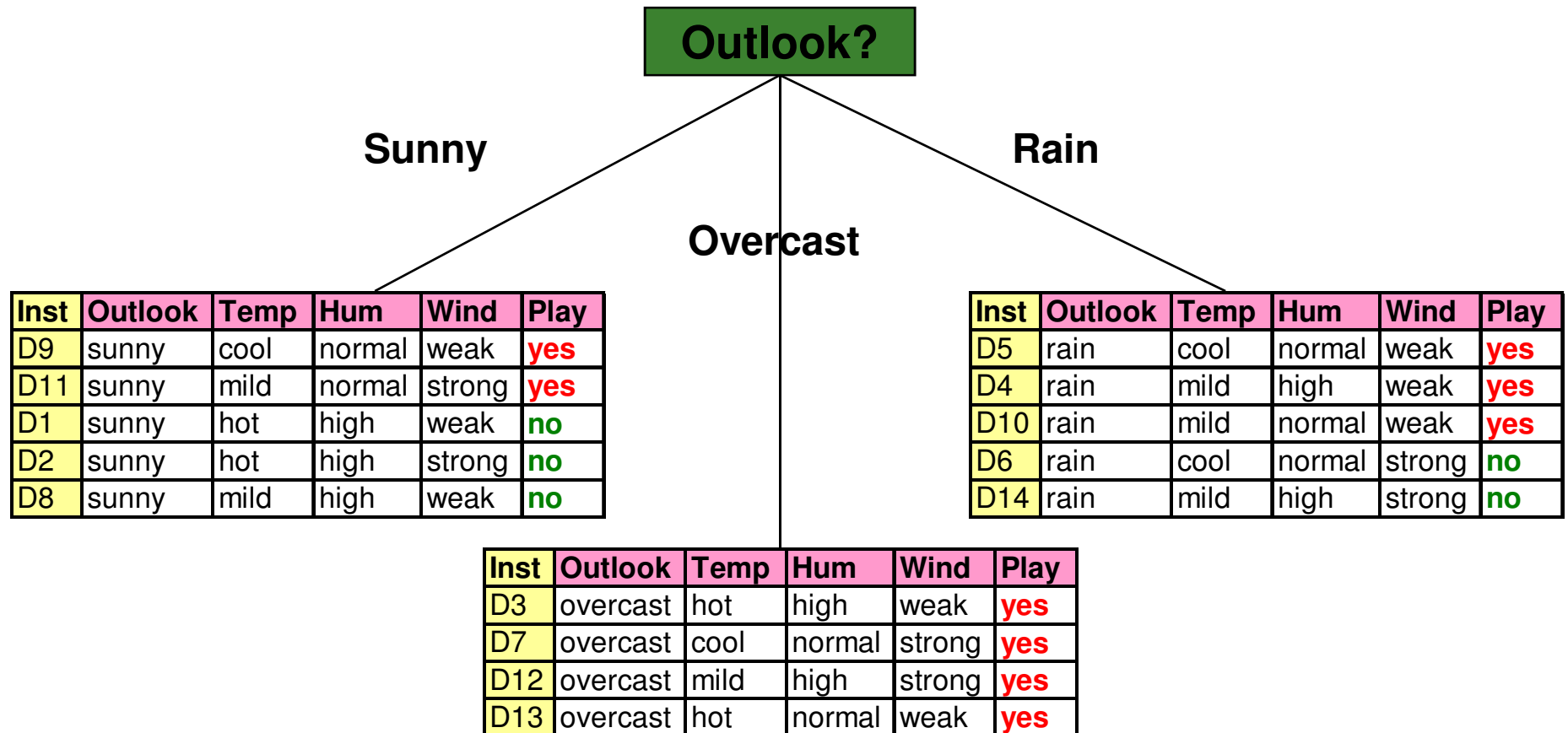
# Árvore de Decisão

## □ Treinamento:

Base de Dados “Tempo”

Instância	Outlook	Temperature	Humidity	Wind	Play
D1	sunny	hot	high	weak	no
D2	sunny	hot	high	strong	no
D3	overcast	hot	high	weak	yes
D4	rain	mild	high	weak	yes
D5	rain	cool	normal	weak	yes
D6	rain	cool	normal	strong	no
D7	overcast	cool	normal	strong	yes
D8	sunny	mild	high	weak	no
D9	sunny	cool	normal	weak	yes
D10	rain	mild	normal	weak	yes
D11	sunny	mild	normal	strong	yes
D12	overcast	mild	high	strong	yes
D13	overcast	hot	normal	weak	yes
D14	rain	mild	high	strong	no

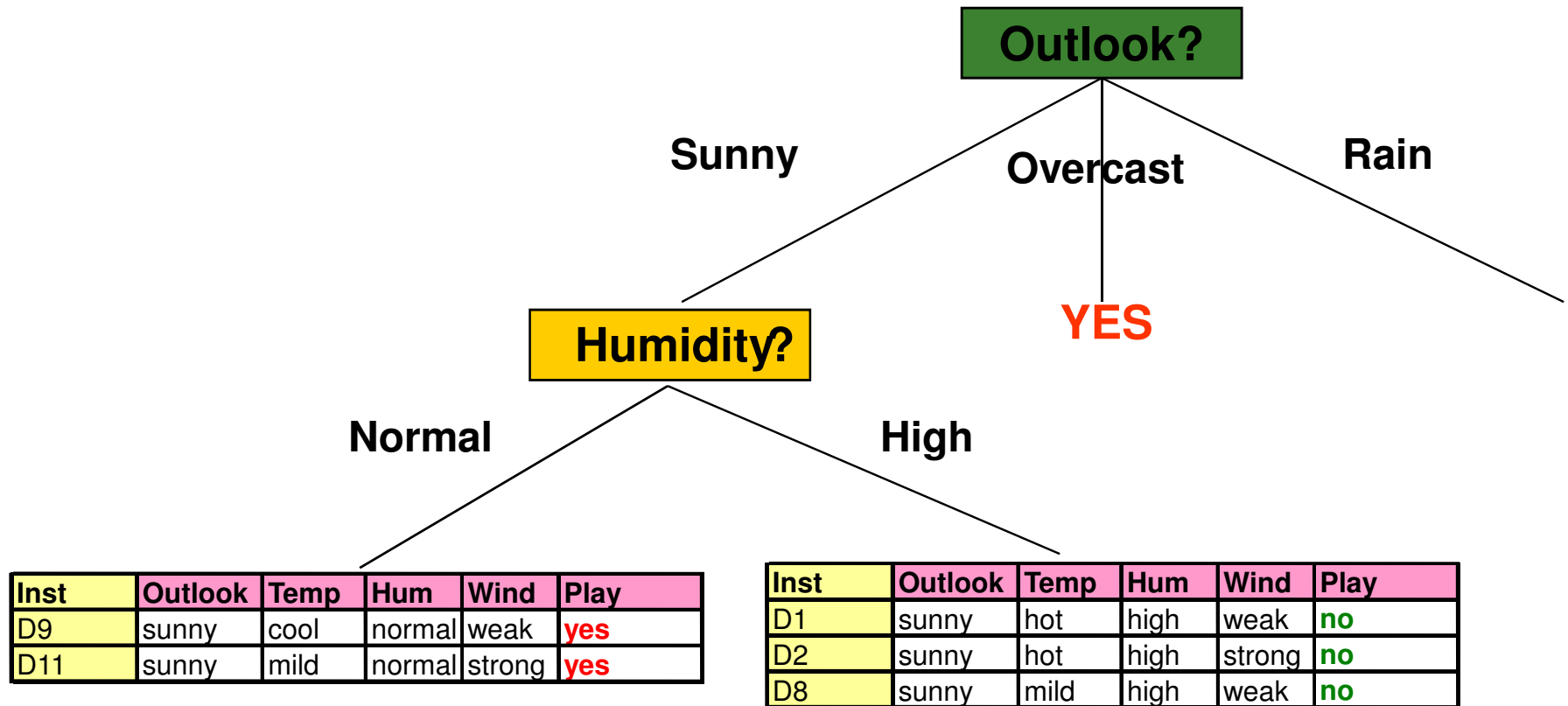
# Árvore de Decisão



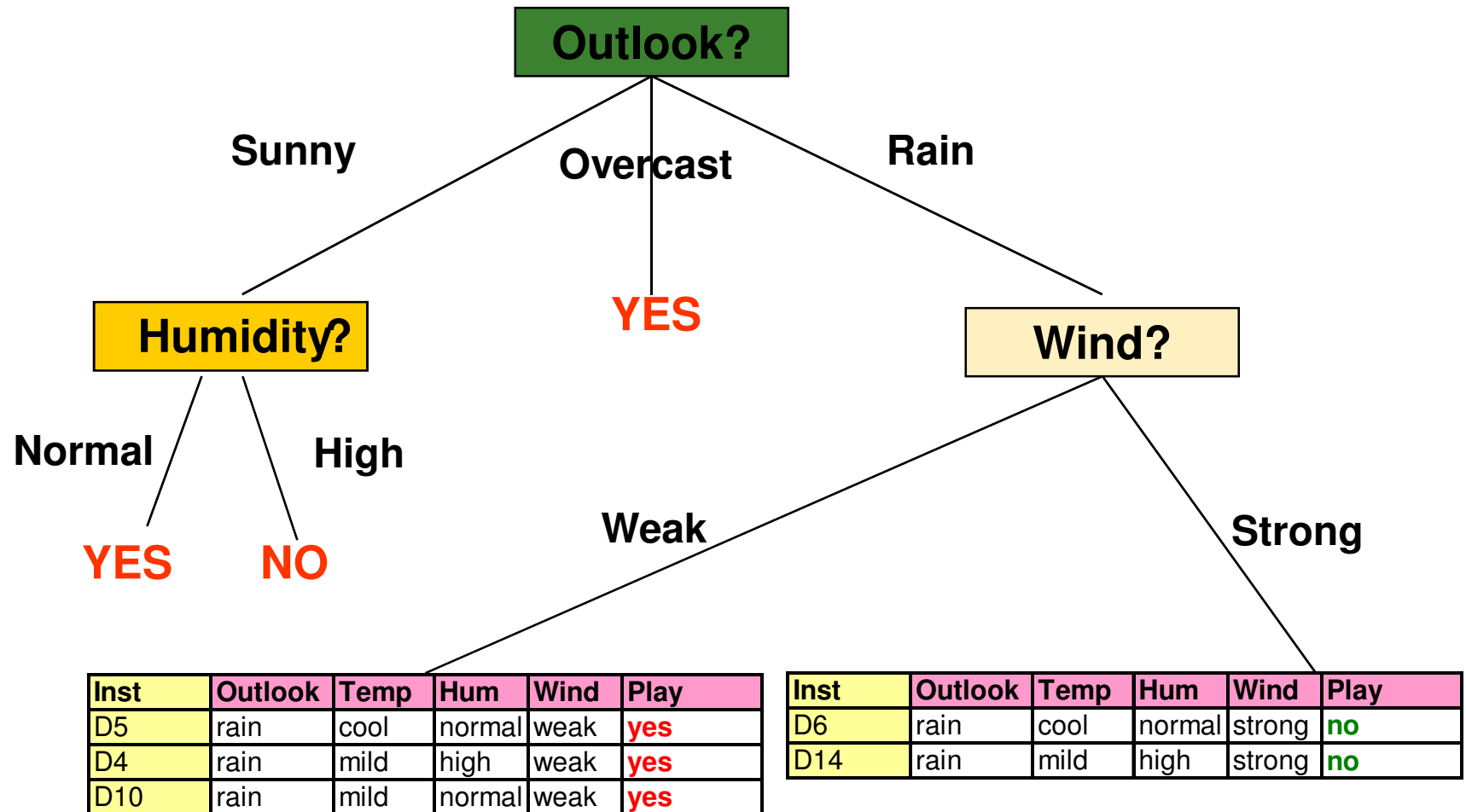
# Árvore de Decisão

Teste	Exemplo	Outlook	Temperature	Humidity	Wind	Play?
If outlook=sunny	D1	Sunny	Hot	High	Weak	No
	D2	Sunny	Hot	High	Strong	No
	D8	Sunny	Mild	High	Weak	No
	D9	Sunny	Cool	Normal	Weak	Yes
	D11	Sunny	Mild	Normal	Strong	Yes
If outlook=overcast	D3	Overcast	Hot	High	Weak	Yes
	D7	Overcast	Cold	Normal	Strong	Yes
	D12	Overcast	Mild	High	Strong	Yes
	D13	Overcast	Hot	Normal	Weak	Yes
If outlook=rain	D4	Rain	Mild	High	Weak	Yes
	D5	Rain	Cool	Normal	Weak	Yes
	D6	Rain	Cool	Normal	Strong	No
	D10	Rain	Mild	Normal	Weak	Yes
	D14	Rain	Mild	High	Strong	No

# Árvore de Decisão



# Árvore de Decisão

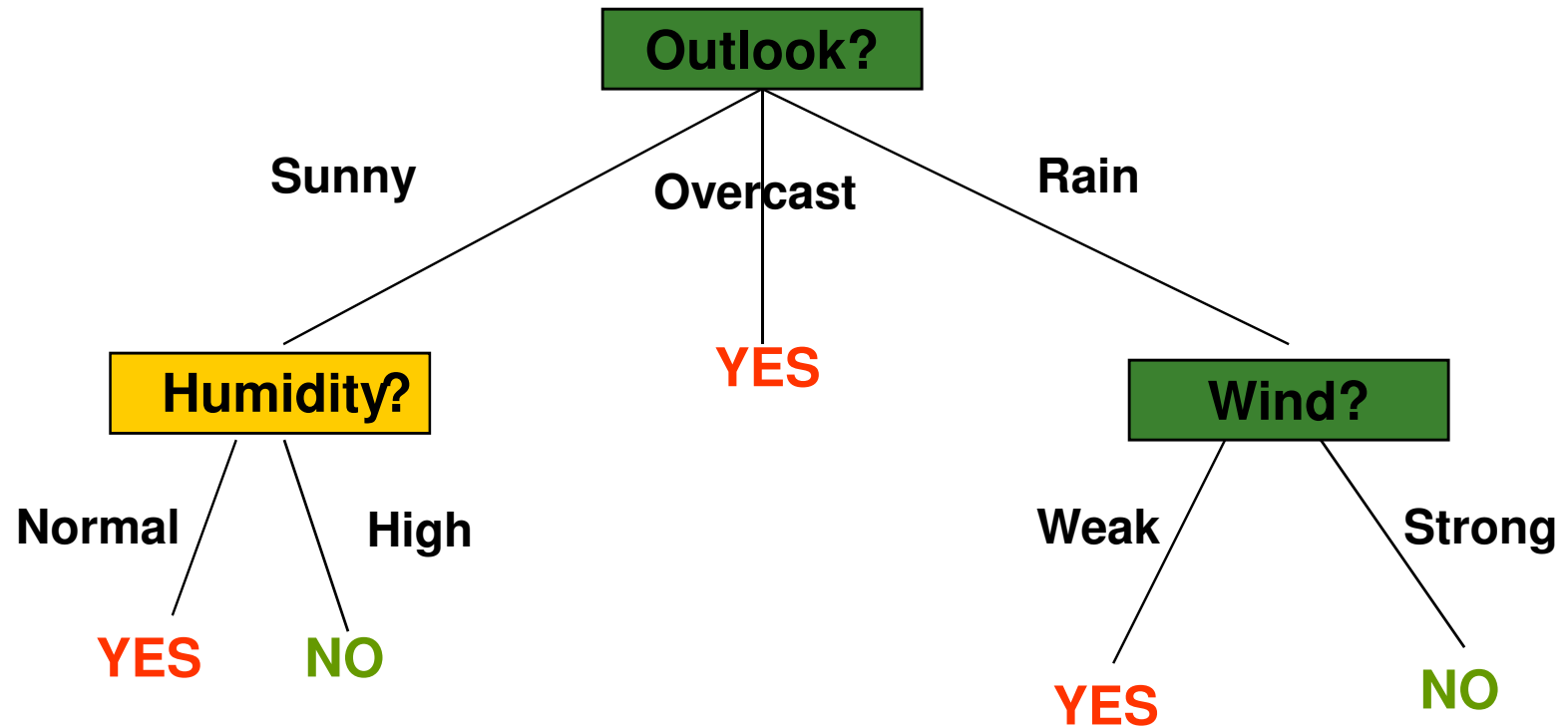




# Árvore de Decisão

Teste	Exemplo	Outlook	Temperature	Humidity	Wind	Play?
If outlook=sunny and humidity=high	D1	Sunny	Hot	High	Weak	No
	D2	Sunny	Hot	High	Strong	No
	D8	Sunny	Mild	High	Weak	No
If outlook=sunny and humidity=normal	D9	Sunny	Cool	Normal	Weak	Yes
	D11	Sunny	Mild	Normal	Strong	Yes
If outlook=overcast	D3	Overcast	Hot	High	Weak	Yes
	D7	Overcast	Cold	Normal	Strong	Yes
	D12	Overcast	Mild	High	Strong	Yes
	D13	Overcast	Hot	Normal	Weak	Yes
If outlook=rain and wind=strong	D6	Rain	Cool	Normal	Strong	No
	D14	Rain	Mild	High	Strong	No
If outlook=rain and wind=weak	D4	Rain	Mild	High	Weak	Yes
	D5	Rain	Cool	Normal	Weak	Yes
	D10	Rain	Mild	Normal	Weak	Yes

# Árvore de Decisão



# Árvore de Decisão

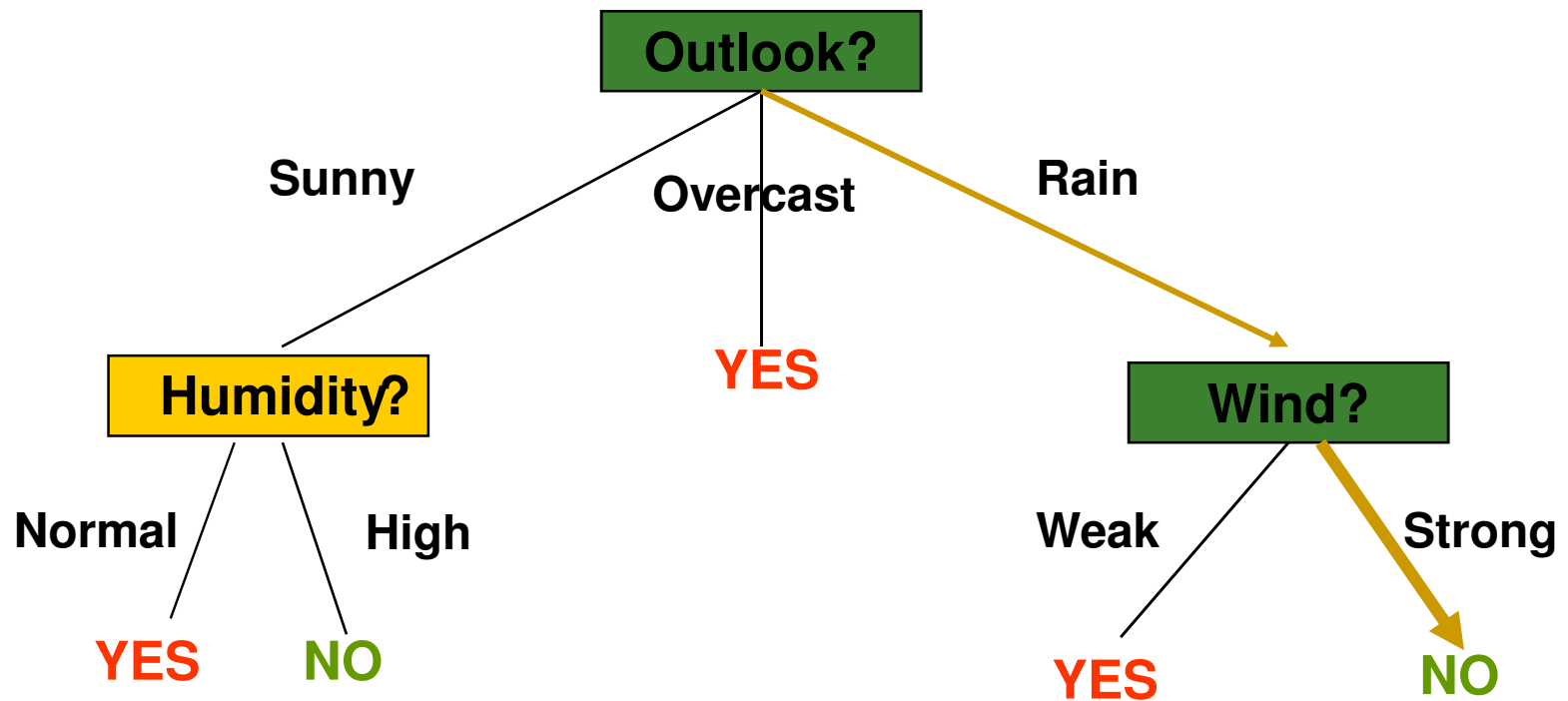
Teste	Exemplo	Outlook	Temperature	Humidity	Wind	Play?
If outlook=sunny and humidity=high	D1	Sunny	Hot	High	Weak	No
	D2	Sunny	Hot	High	Strong	No
	D8	Sunny	Mild	High	Weak	No
If outlook=sunny and humidity=nomal	D9	Sunny	Cool	Normal	Weak	Yes
	D11	Sunny	Mild	Normal	Strong	Yes
If outlook=overcast	D3	Overcast	Hot	High	Weak	Yes
	D7	Overcast	Cold	Normal	Strong	Yes
	D12	Overcast	Mild	High	Strong	Yes
	D13	Overcast	Hot	Normal	Weak	Yes
If outlook=rain and wind=strong	D6	Rain	Cool	Normal	Strong	No
	D14	Rain	Mild	High	Strong	No
If outlook=rain and wind=weak	D4	Rain	Mild	High	Weak	Yes
	D5	Rain	Cool	Normal	Weak	Yes
	D10	Rain	Mild	Normal	Weak	Yes

# Árvore de Decisão

❑ Como classificar a seguinte instância:

@data

rainy, hot, normal, strong, ???



# Algoritmos para árvores de decisão

## ❑ Algoritmo Básico:

- A árvore é construída recursivamente no sentido top-down (divisão para conquista).
- Os atributos são nominais (se numéricos, eles são discretizados).
- Atributos “**testes**” são selecionados com base em heurísticas ou medidas estatísticas (ex., ganho de informação).

## ❑ Condições de parada do particionamento:

- Todas as amostras de um nó pertencem a mesma classe.
- Não existem mais atributos para particionamento.

# Algoritmos mais conhecidos

- ❑ **ID3 (Iterative Dichotomiser 3) (Quilan,1986):**
  - Os atributos devem ser obrigatoriamente categóricos.
- ❑ **C4.5 (J48 no Weka) (Quilan, 1993):**
  - Um algoritmo para geração de árvores de decisão, sucessor do algoritmo ID3.
  - Considera atributos numéricos e categóricos.
- ❑ **CART (Classification And Regression Trees) (Breiman et al., 1984):**
  - Produz árvores de classificação ou regressão, dependendo se as variáveis são categóricas ou numéricas.

---

# Overview

## ❑ Vantagens:

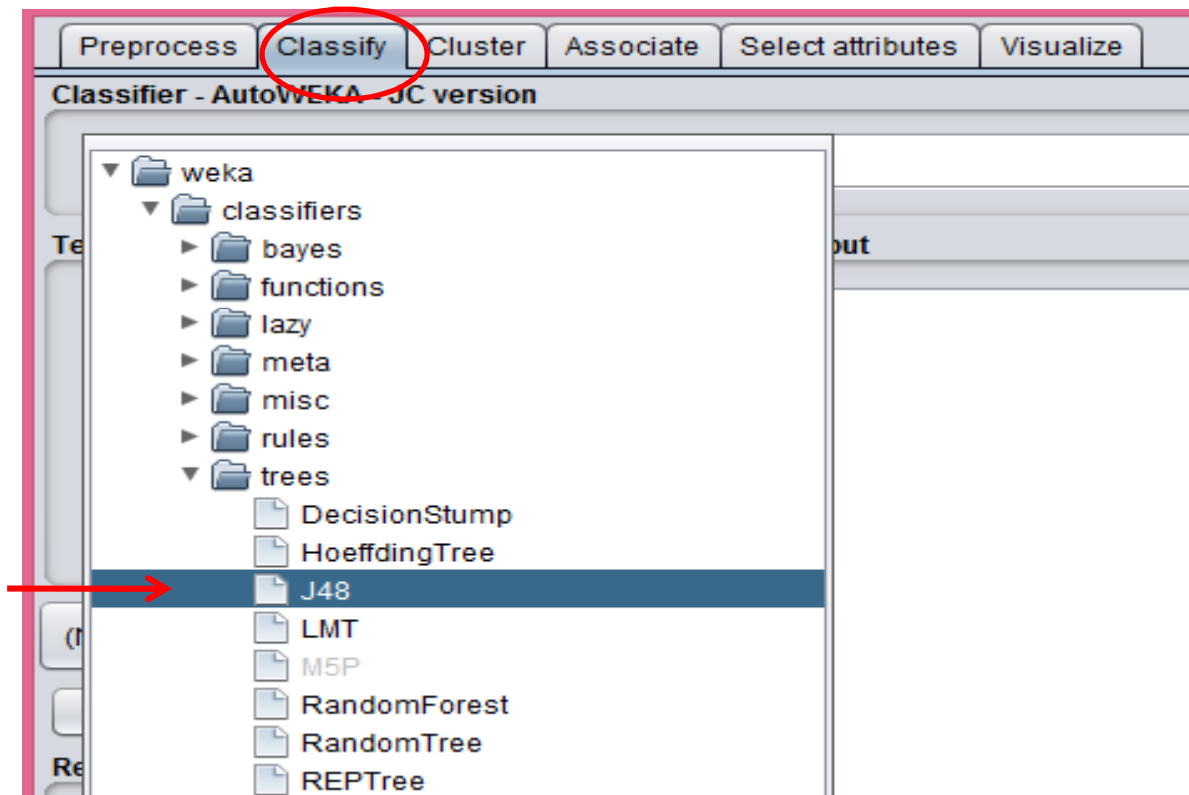
- Estrutura de fácil manipulação;
- Produzem modelos que podem ser facilmente interpretados por humanos;
- Muito rápido para classificar amostras desconhecidas.

## ❑ Desvantagens:

- Pouca robustez a dados de grande dimensão;
- Acurácia afetada por atributos pouco relevantes;
- Dificuldade em lidar com dados contínuos.

# Árvore de Decisão

- ❑ Utilizando **J48** (WEKA):





# Árvore de Decisão

## ❑ Analisando os resultados....

```
Classifier output

=== Run information ===

Scheme:      weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:    Iris
Instances:   150
Attributes:  5
              a0
              a1
              a2
              a3
              class
Test mode:   split 70.0% train, remainder test

=== Classifier model (full training set) ===

J48 pruned tree
-----
a3 <= 0.6: Iris-setosa (50.0)
a3 > 0.6
|   a3 <= 1.7
|   |   a2 <= 4.9: Iris-versicolor (48.0/1.0)
|   |   a2 > 4.9
|   |   |   a3 <= 1.5: Iris-virginica (3.0)
|   |   |   a3 > 1.5: Iris-versicolor (3.0/1.0)
|   |   a3 > 1.7: Iris-virginica (46.0/1.0)

Number of Leaves :    5
Size of the tree :    9
```

# Árvore de Decisão

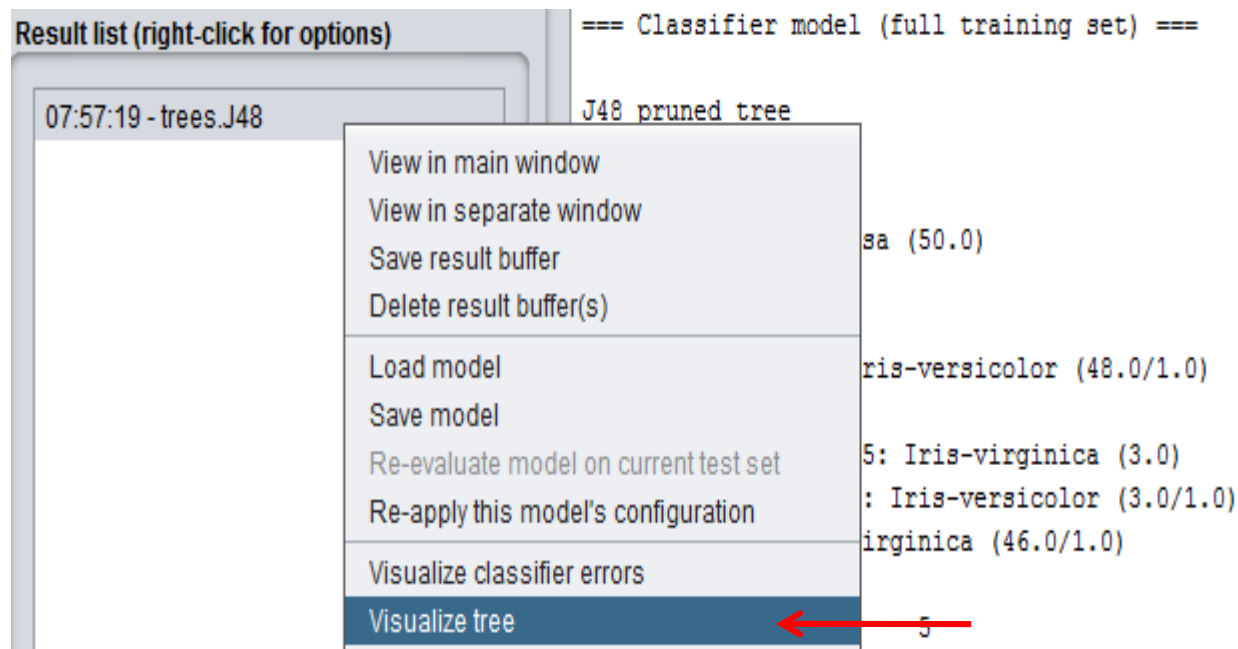
## ☐ Analisando os resultados....

=== Summary ===

Correctly Classified Instances	43	95.5556 %	←
Incorrectly Classified Instances	2	4.4444 %	←
Kappa statistic	0.9331		
Mean absolute error	0.0416		
Root mean squared error	0.1682		
Relative absolute error	9.3466 %		
Root relative squared error	35.6559 %		
Total Number of Instances	45		

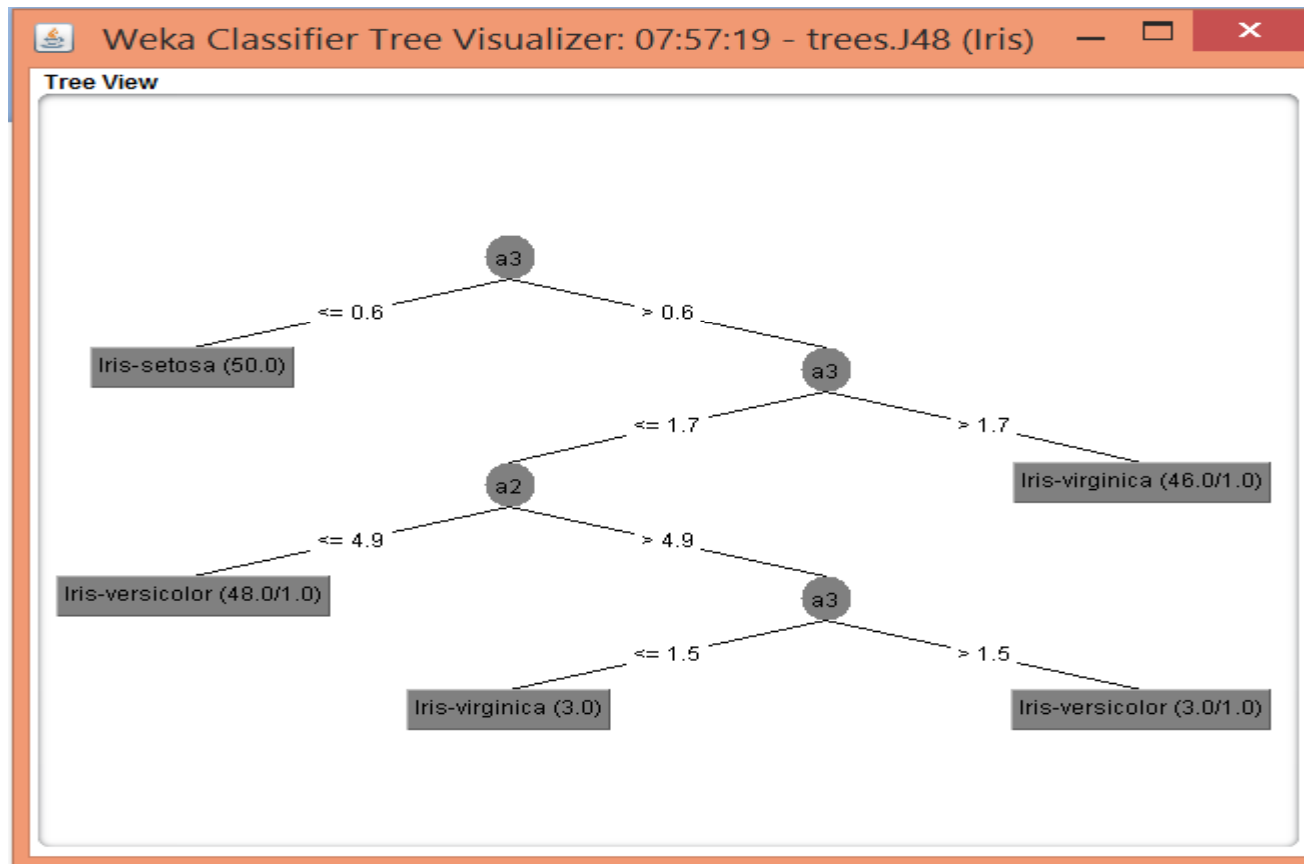
# Árvore de Decisão

## ❑ Visualizando a árvore....



# Árvore de Decisão

❑ Visualizando a árvore....



# Dúvidas ...

