# Classificação

Prof. Dra. Karina S. Machado
Prof. Dr. Eduardo N. Borges
Prof. Dr. Adriano Werhli

# Classificação de dados

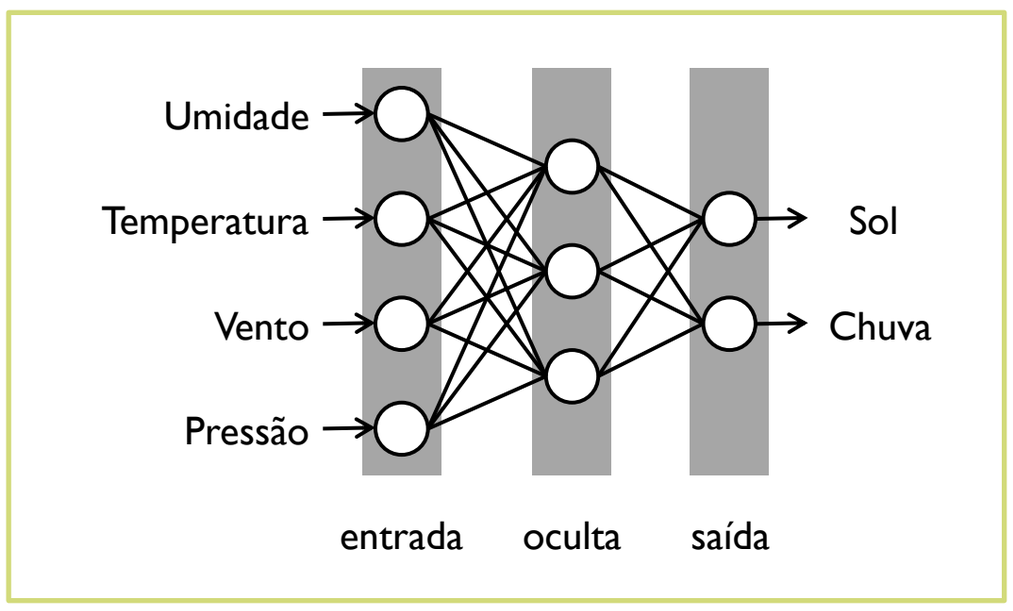
- Definição
  - Método supervisionado que determina um modelo para um determinado atributo classe que é função dos valores dos outros atributos. Pode ser utilizado para predizer a que classe de dados uma nova instância pertence (TAN; STEINBACH; KUMAR, 2005).
  - Processo de encontrar, através de aprendizado de máquina supervisionado, um modelo ou função que descreva diferentes classes de dados (HAN; KAMBER, 2006).

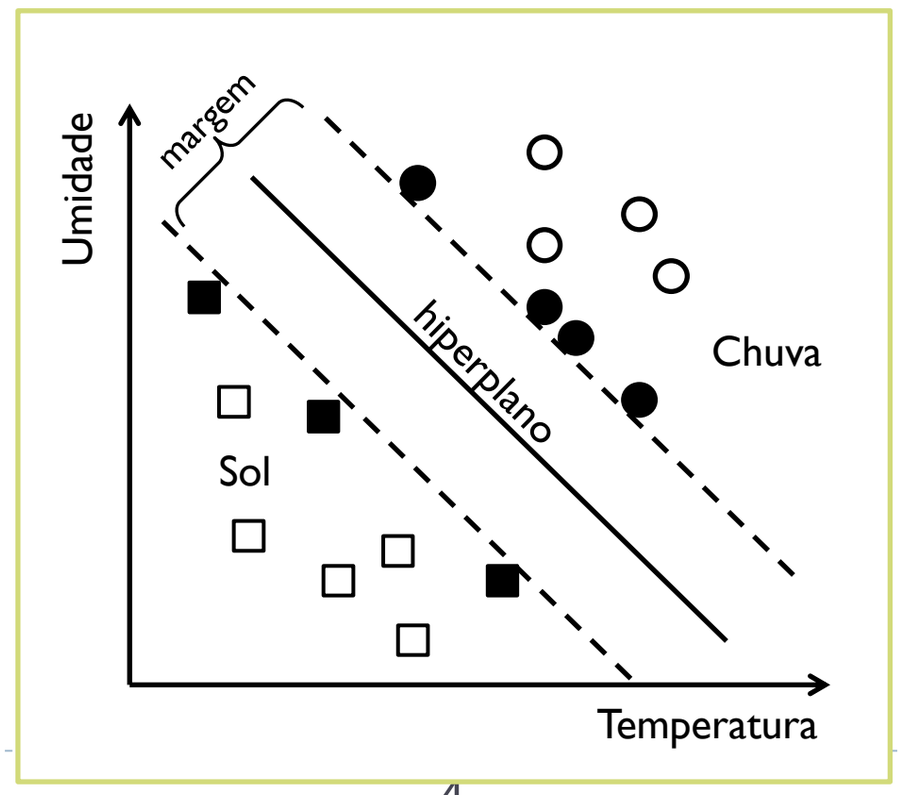
# Modelagem

- Algoritmos de classificação
  - Multilayer Perceptron (MLP)
  - Voted Perceptron
  - Sequential Minimal Optimization (SMO)
  - Naïve Bayes
  - Bayes Net
  - RIPPER
  - C4.5
  - AdaBoost.M1

# Modelagem

- Algoritmos de classificação
  - Multilayer Perceptron (MLP)
  - Voted Perceptron
  - Sequential Minimal Optimization (SMO)
  - Naïve Bayes



# Modelagem

- Algoritmos de classificação
  - Multilayer Perceptron (MLP)
  - Voted Perceptron
  - Sequential Minimal Optimization (SMO)
  - Naïve Bayes
  - Bayes Net
  - RIPPER
  - C4.5
  - AdaBoost.M1

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$

Com e sem dependência entre atributos

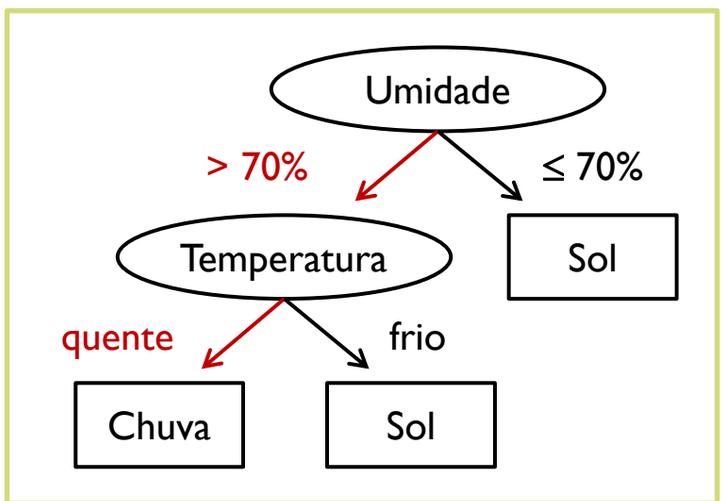
# Modelagem

- Algoritmos de classificação
  - Multilayer Perceptron (MLP)
  - Voted Perceptron
  - Sequential Minimal Optimization (SMO)
  - Naïve Bayes
  - Bayes Net
  - RIPPER
  - C4.5
  - AdaBoost.M1

$$r_i : (A_1 \, op \, v_1) \wedge (A_2 \, op \, v_2) \wedge \cdots \wedge (A_k \, op \, v_k) \rightarrow y_i$$

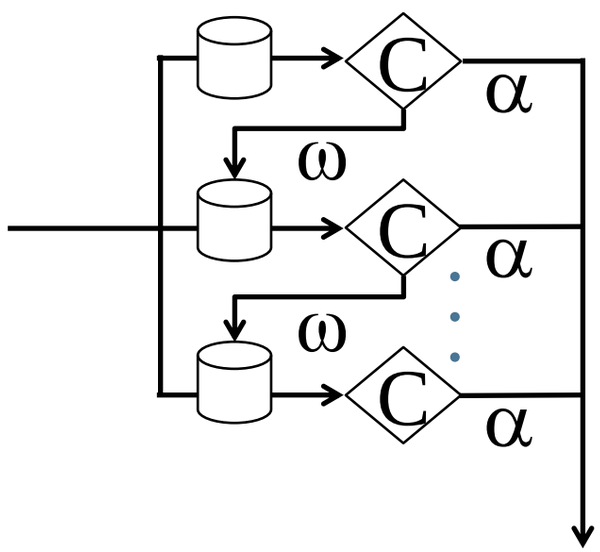
# Modelagem

- Algoritmos de classificação
  - Multilayer Perceptron (MLP)
  - Voted Perceptron
  - Sequential Minimal Optimization (SMO)
  - Naïve Bayes
  - Bayes Net
  - RIPPER
  - C4.5
  - AdaBoost.M1



# Modelagem

- Algoritmos de classificação
  - Multilayer Perceptron (MLP)
  - Voted Perceptron
  - Sequential Minimal Optimization (SMO)
  - Naïve Bayes
  - Bayes Net
  - RIPPER
  - C4.5
  - AdaBoost.M1



# “Contraceptive Methods Choice” - CMC

cmc.names **cmc.data**

```
24,2,3,3,1,1,2,3,0,1
45,1,3,10,1,1,3,4,0,1
43,2,3,7,1,1,3,4,0,1
42,3,2,9,1,1,3,3,0,1
36,3,3,8,1,1,3,2,0,1
19,4,4,0,1,1,3,3,0,1
38,2,3,6,1,1,3,2,0,1
21,3,3,1,1,0,3,2,0,1
27,2,3,3,1,1,3,4,0,1
45,1,1,8,1,1,2,2,1,1
38,1,3,2,1,0,3,3,1,1
42,1,4,4,1,1,1,3,0,1
44,4,4,1,1,0,1,4,0,1
42,2,4,1,1,0,3,3,0,1
38,3,4,2,1,1,2,3,0,1
26,2,4,0,1,1,4,1,0,1
48,1,1,7,1,1,2,4,0,1
39,2,2,6,1,1,2,4,0,1
```

**cmc.names** cmc.data

```
1. Title: Contraceptive Method Choice

2. Sources:
   (a) Origin:  This dataset is a subset of the 1987 National Indonesia
                Contraceptive Prevalence Survey
   (b) Creator: Tjen-Sien Lim (limt@stat.wisc.edu)
   (c) Donor:   Tjen-Sien Lim (limt@stat.wisc.edu)
   (c) Date:    June 7, 1997

3. Past Usage:
   Lim, T.-S., Loh, W.-Y. & Shih, Y.-S. (1999). A Comparison of
   Prediction Accuracy, Complexity, and Training Time of Thirty-three
   Old and New Classification Algorithms. Machine Learning. Forthcoming.
   (ftp://ftp.stat.wisc.edu/pub/loh/treeprogs/quest1.7/mach1317.pdf or
   (http://www.stat.wisc.edu/~limt/mach1317.pdf)

4. Relevant Information:
   This dataset is a subset of the 1987 National Indonesia Contraceptive
   Prevalence Survey. The samples are married women who were either not
   pregnant or do not know if they were at the time of interview. The
   problem is to predict the current contraceptive method choice
   (no use, long-term methods, or short-term methods) of a woman based
   on her demographic and socio-economic characteristics.

5. Number of Instances: 1473

6. Number of Attributes: 10 (including the class attribute)
```

# “Contraceptive Methods Choice” - CMC

cmc.names **cmc.data**

```
24,2,3,3,1,1,2,3,0,1
45,1,3,10,1,1,3,4,0,1
43,2,3,7,1,1,3,4,0,1
42,3,2,9,1,1,3,3,0,1
36,3,3,8,1,1,3,2,0,1
19,4,4,0,1,1,3,3,0,1
38,2,3,6,1,1,3,2,0,1
21,3,3,1,1,0,3,2,0,1
27,2,3,3,1,1,3,4,0,1
45,1,1,8,1,1,2,2,1,1
38,1,3,2,1,0,3,3,1,1
42,1,4,4,1,1,1,3,0,1
44,4,4,1,1,0,1,4,0,1
42,2,4,1,1,0,3,3,0,1
38,3,4,2,1,1,2,3,0,1
26,2,4,0,1,1,4,1,0,1
48,1,1,7,1,1,2,4,0,1
39,2,2,6,1,1,2,4,0,1
```

7. Attribute Information:

| | | |
|---|---|---|
| 1. Wife's age | (numerical) | |
| 2. Wife's education | (categorical) | 1=low, 2, 3, 4=high |
| 3. Husband's education | (categorical) | 1=low, 2, 3, 4=high |
| 4. Number of children ever born | (numerical) | |
| 5. Wife's religion | (binary) | 0=Non-Islam, 1=Islam |
| 6. Wife's now working? | (binary) | 0=Yes, 1=No |
| 7. Husband's occupation | (categorical) | 1, 2, 3, 4 |
| 8. Standard-of-living index | (categorical) | 1=low, 2, 3, 4=high |
| 9. Media exposure | (binary) | 0=Good, 1=Not good |
| 10. Contraceptive method used | (class attribute) | 1=No-use |
| | | 2=Long-term |
| | | 3=Short-term |

8. Missing Attribute Values: None

# ARFF - CMC

Tipos de dados
Numéricos, categóricos, ordinais, strings

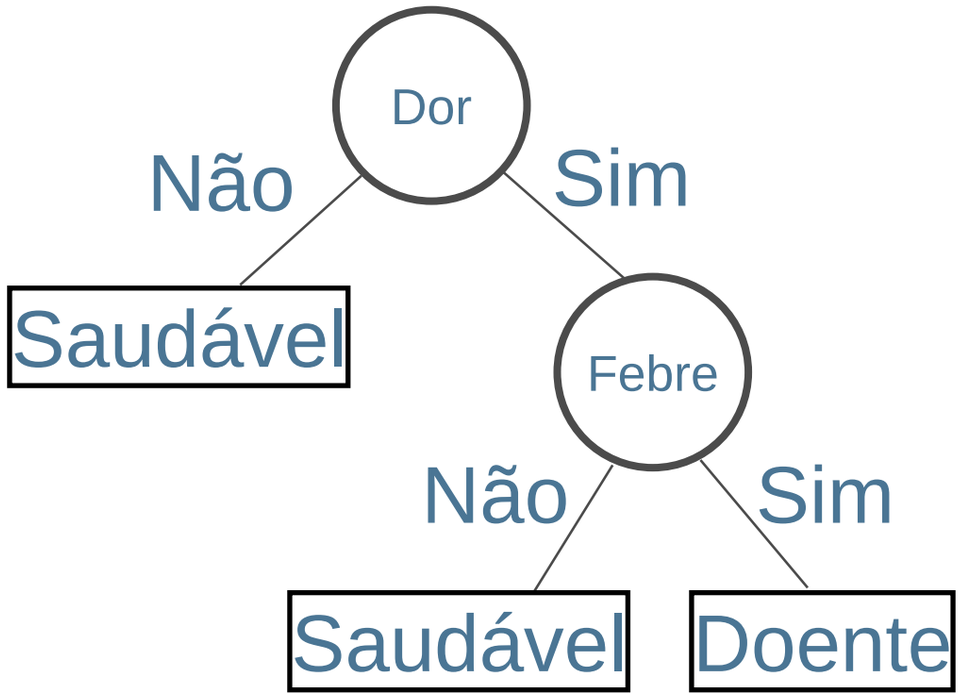
```
@relation CMC

@attribute Idade real
@attribute Educacao_Esposa {1,2,3,4}
@attribute Educacao_Marido {1,2,3,4}
@attribute Numero_Filhos real
@attribute Religiao_Esposa {0,1}
@attribute Esposa_Trabalha {0,1}
@attribute Ocupacao_Marido {1,2,3,4}
@attribute Padrao_Vida {1,2,3,4}
@attribute Exposicao_Midia {0,1}
@attribute Metodo_contraceptivo {1,2,3}

@data
24,2,3,3,1,1,2,3,0,1
```

# Árvores de decisão

- O que são árvores de decisão?
  - Um dos métodos mais usados e práticos para **inferência indutiva**
  - Indução a partir de um conjunto de **dados rotulados** (classificados)
  - A indução é feita baseada na abordagem **dividir para conquistar**
    - Um **problema** complexo é **dividido** em problemas mais simples, ou **subproblemas**
    - **Recursivamente** a mesma estratégia é aplicada a cada subproblema

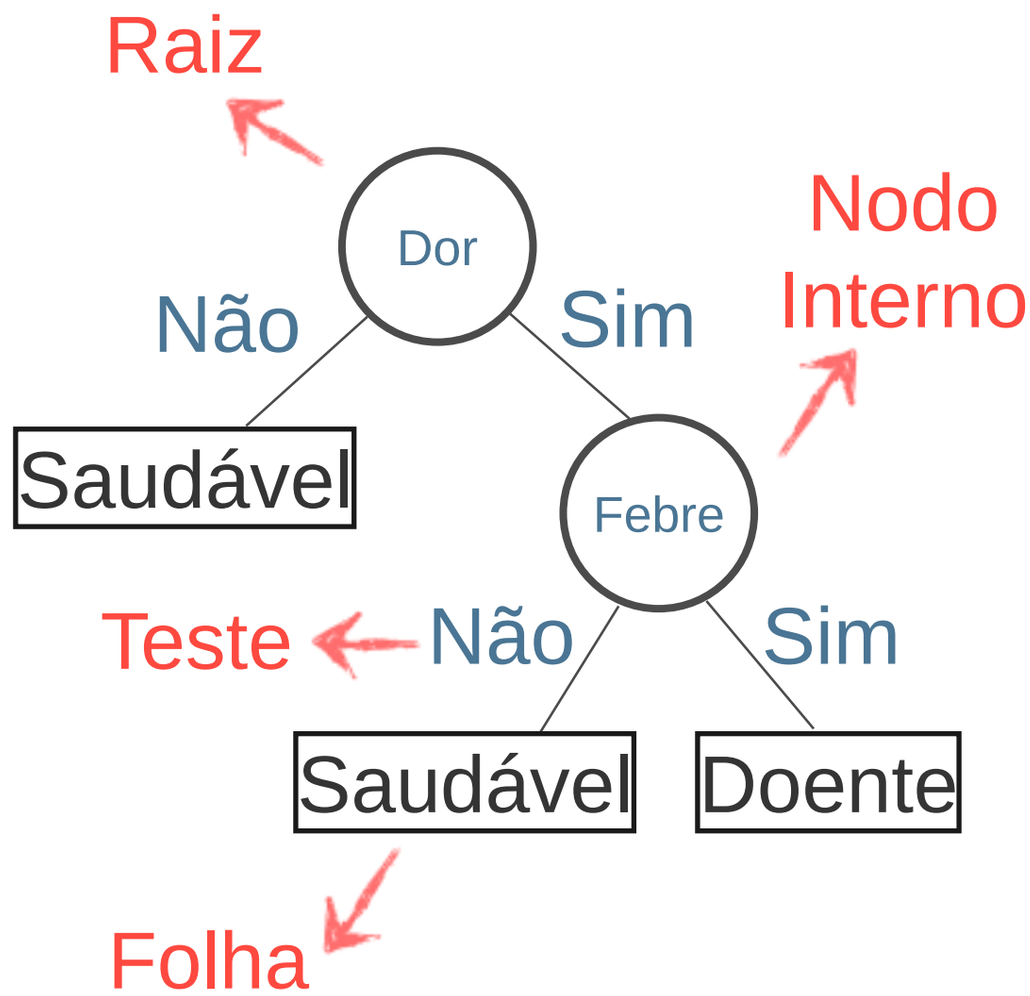
# Árvores de decisão

| Exemplo | Febre | Enjôo | Manchas | Dor | Diagnóstico |
|---|---|---|---|---|---|
| T1 | sim | sim | pequenas | sim | doente |
| T2 | não | não | grandes | não | saudável |
| T3 | sim | sim | pequenas | não | saudável |
| T4 | sim | não | grandes | sim | doente |
| T5 | sim | não | pequenas | sim | saúdavel |
| T6 | não | não | grandes | sim | doente |



# Árvores de decisão

- Estrutura hierárquica consistindo de nodos e arestas direcionados
  - Nodos raiz, internos e folha
- Cada nó de decisão contém um teste de atributo
- Cada ramo descendente corresponde a um possível valor desse atributo
- Cada folha está associada a uma classe
- Abordagem *top-down* a partir do nodo raiz
- Cada percurso da árvore (raiz-folha) corresponde a uma regra de classificação



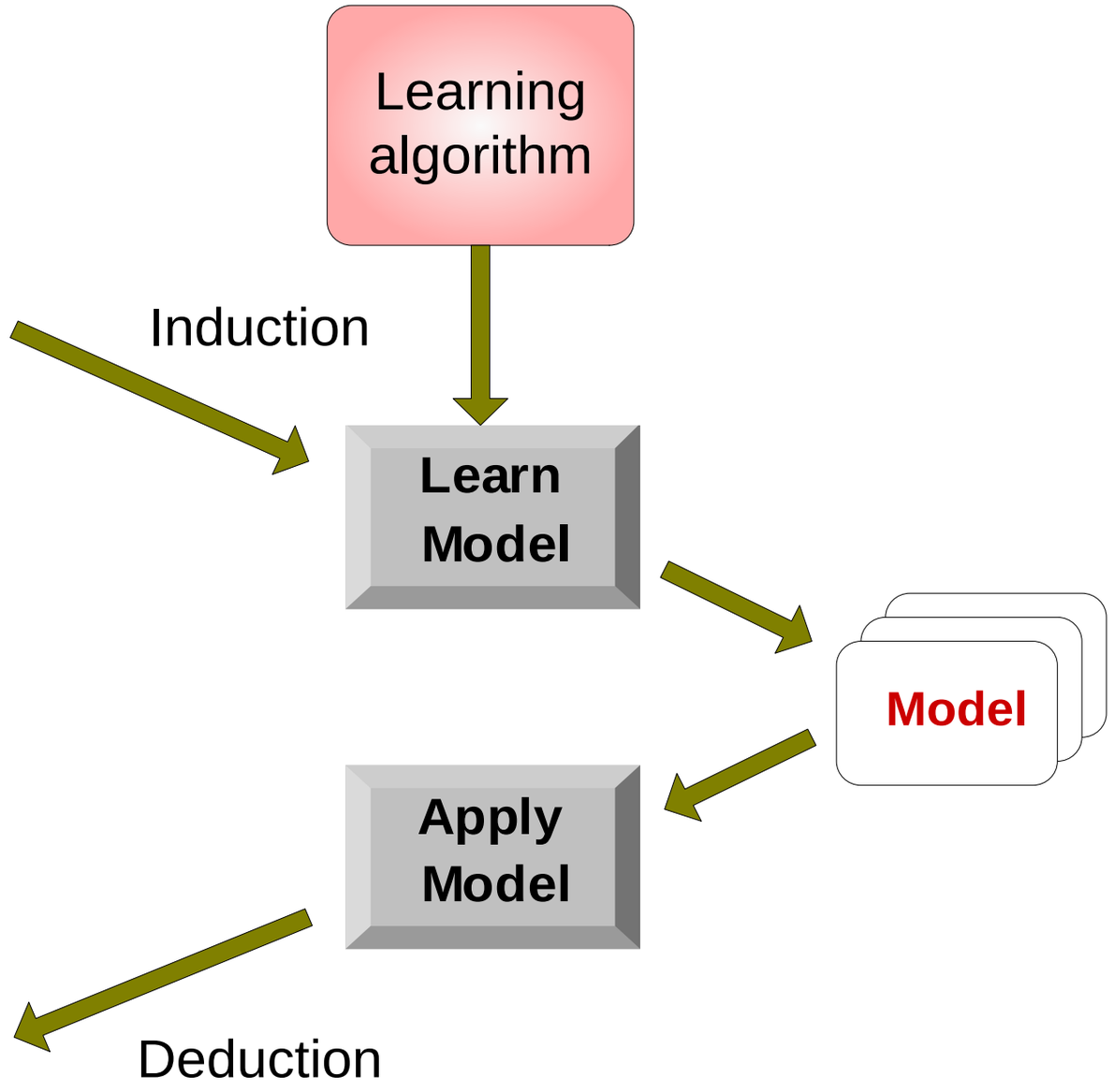
# Tarefa de Classificação

| *Tid* | Attrib1 | Attrib2 | Attrib3 | 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 |

Training Set

| *Tid* | Attrib1 | Attrib2 | Attrib3 | Class |
|---|---|---|---|---|
| 11 | No | Small | 55K | ? |
| 12 | Yes | Medium | 80K | ? |
| 13 | Yes | Large | 110K | ? |
| 14 | No | Small | 95K | ? |
| 15 | No | Large | 67K | ? |

Test Set

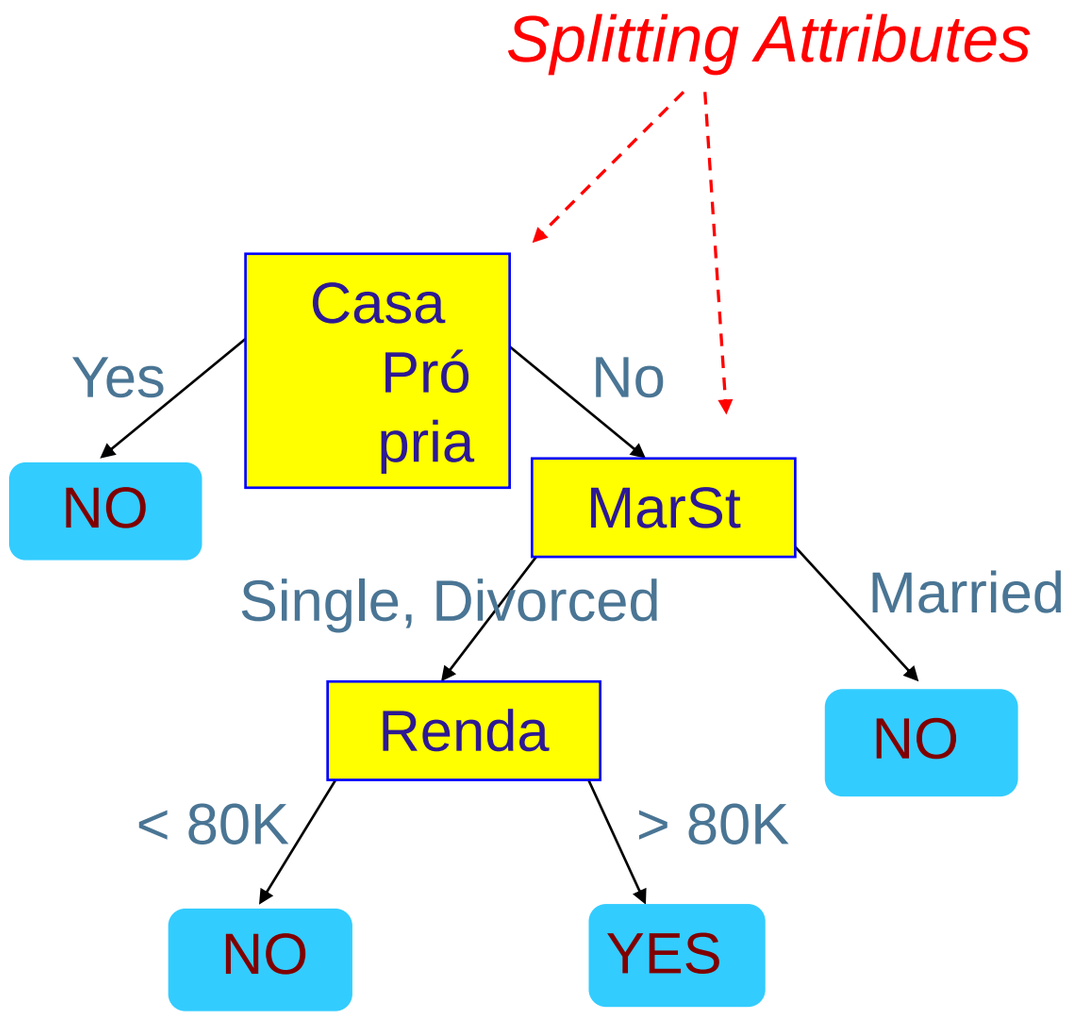


# Concessão de empréstimo

| *Tid* | Casa Própria | Marital Status | Renda Anual | Inadimplente |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |

categorical categorical continuous class

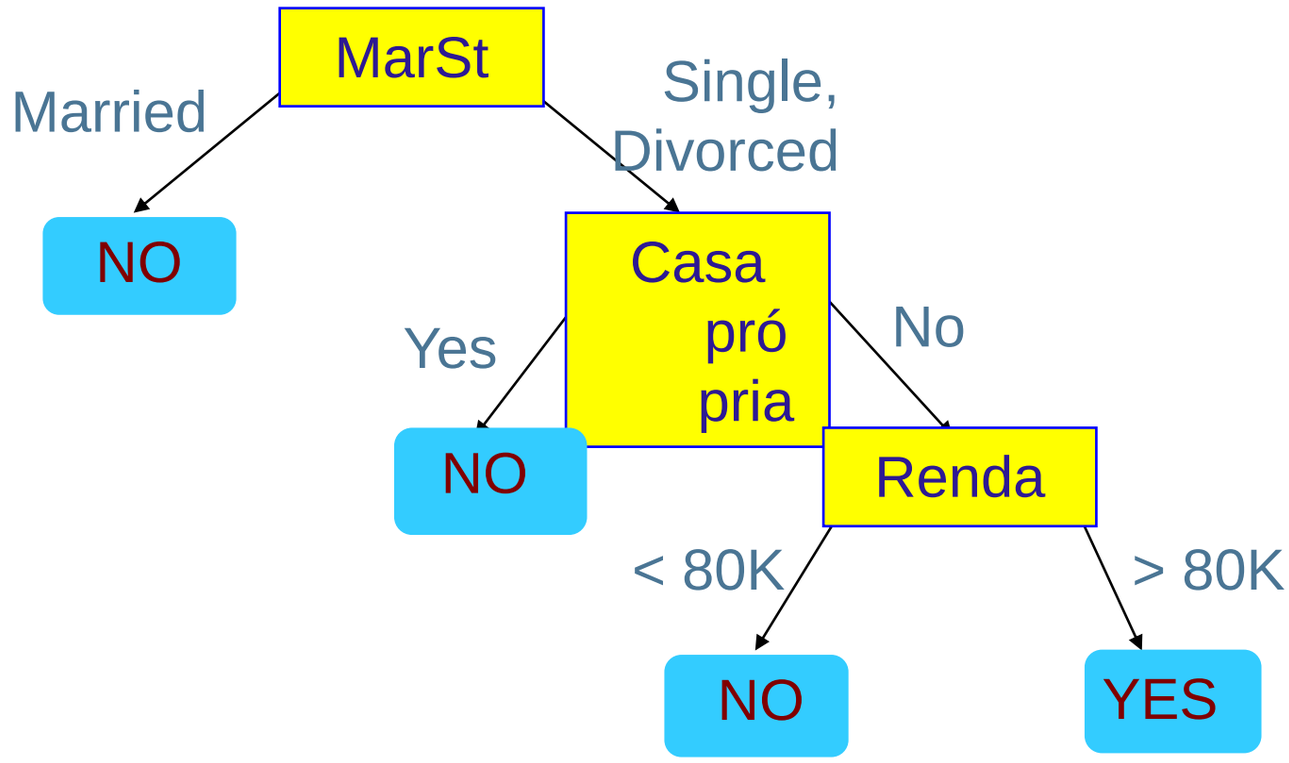
Training Data




Model: Decision Tree

# Another Example of Decision Tree

| *Tid* | Casa Própria | Marital Status | Renda Anual | Inadimplente |
|---|---|---|---|---|
| | categorical | categorical | continuous | class |
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |




Pode ter mais de uma árvore para o mesmo conjunto de dados

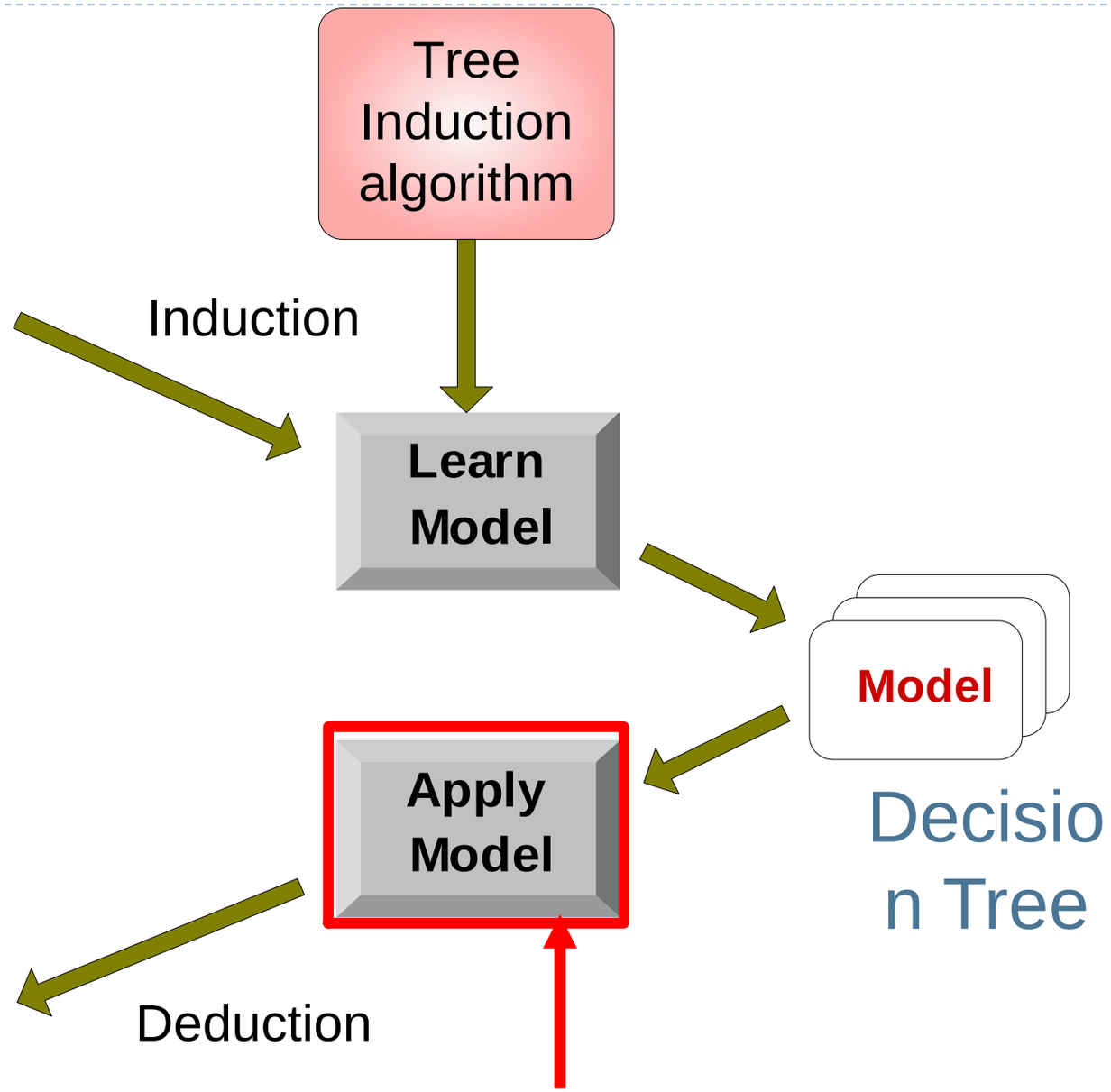
# Decision Tree Classification Task

| *Tid* | Attrib1 | Attrib2 | Attrib3 | 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 |

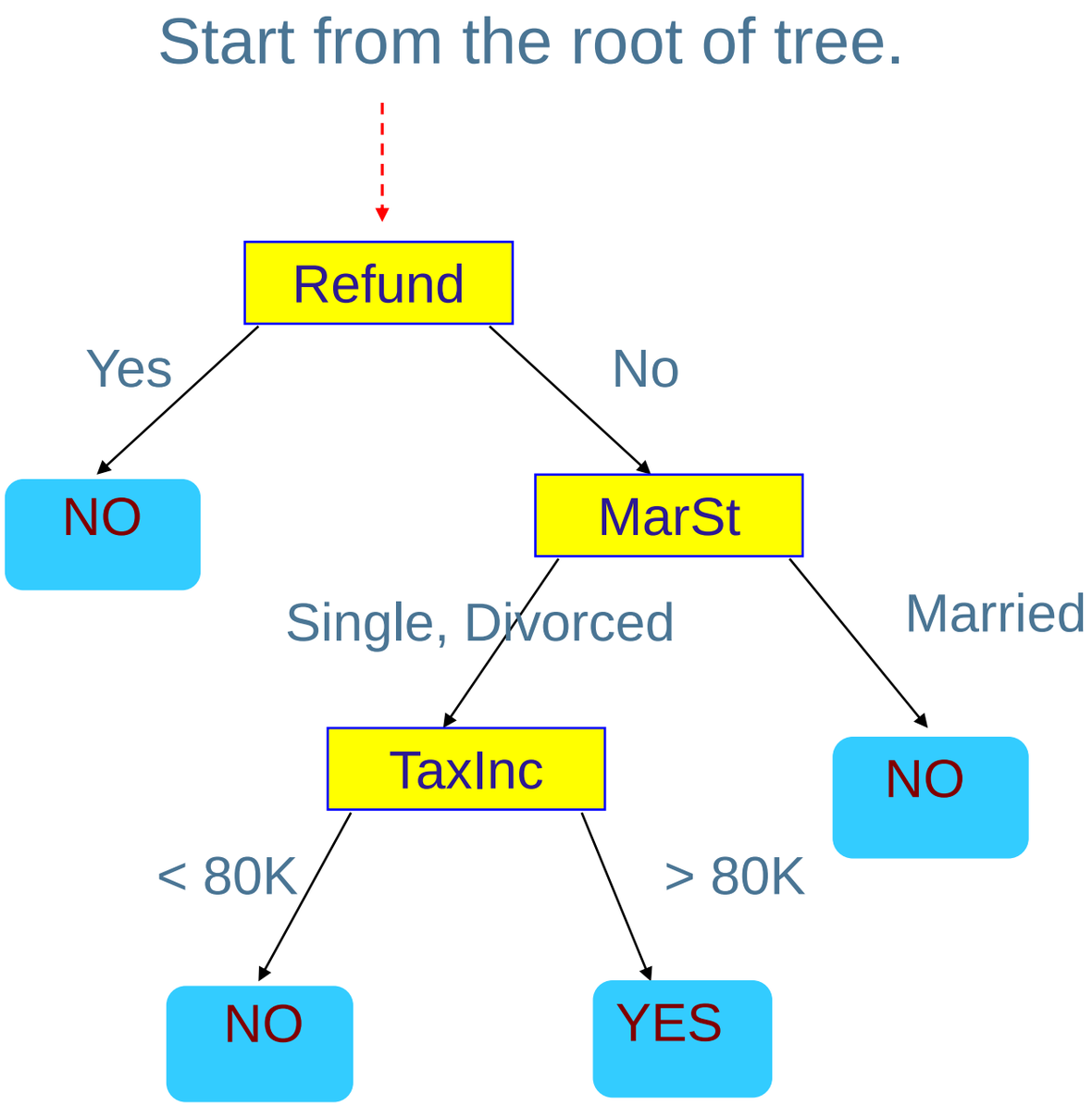
Training Set

| *Tid* | Attrib1 | Attrib2 | Attrib3 | Class |
|---|---|---|---|---|
| 11 | No | Small | 55K | ? |
| 12 | Yes | Medium | 80K | ? |
| 13 | Yes | Large | 110K | ? |
| 14 | No | Small | 95K | ? |
| 15 | No | Large | 67K | ? |

Test Set



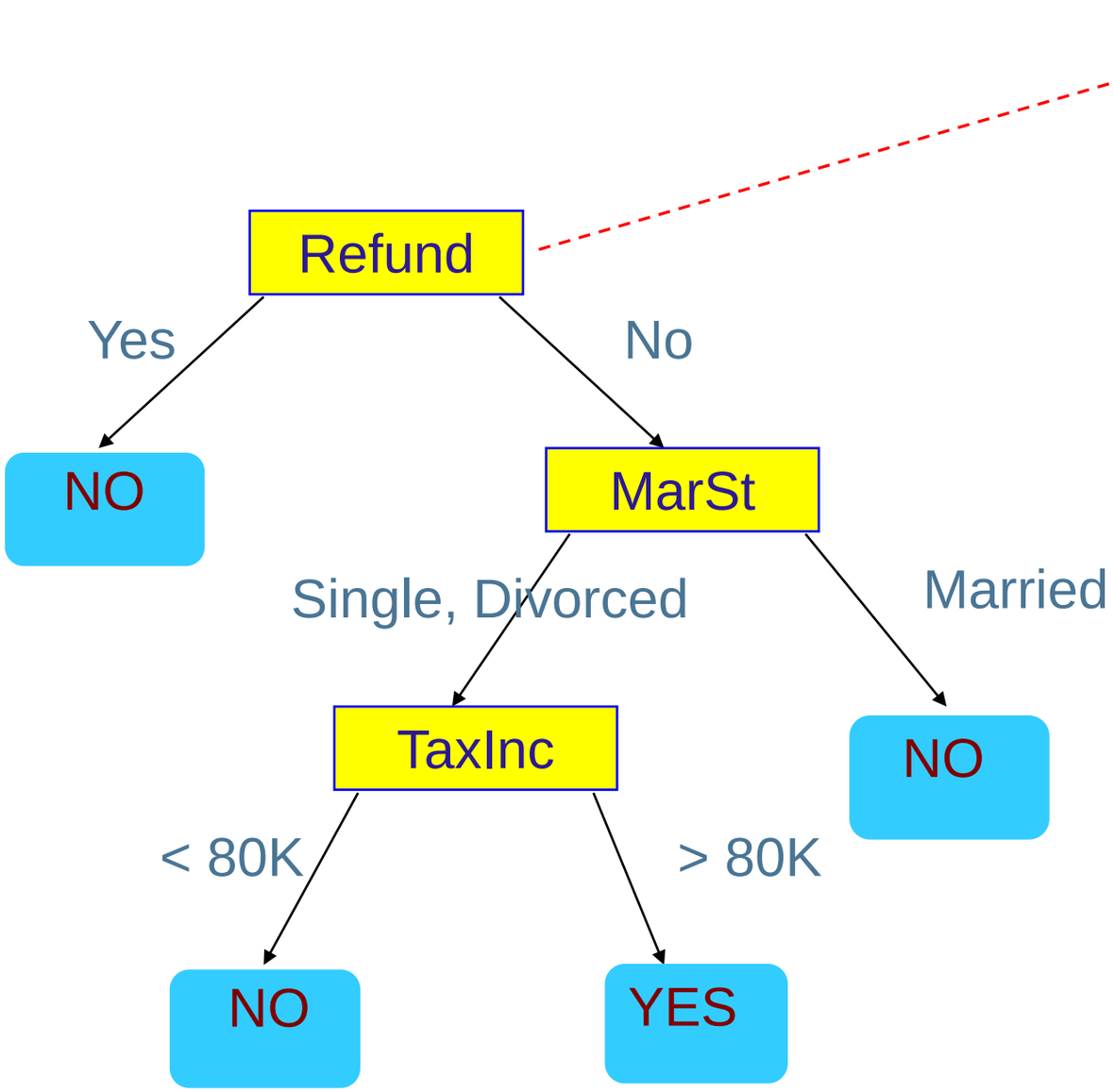
# Aplicando um modelo para testar os dados




| Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|
| No | Married | 80K | **?** |

Dados de teste

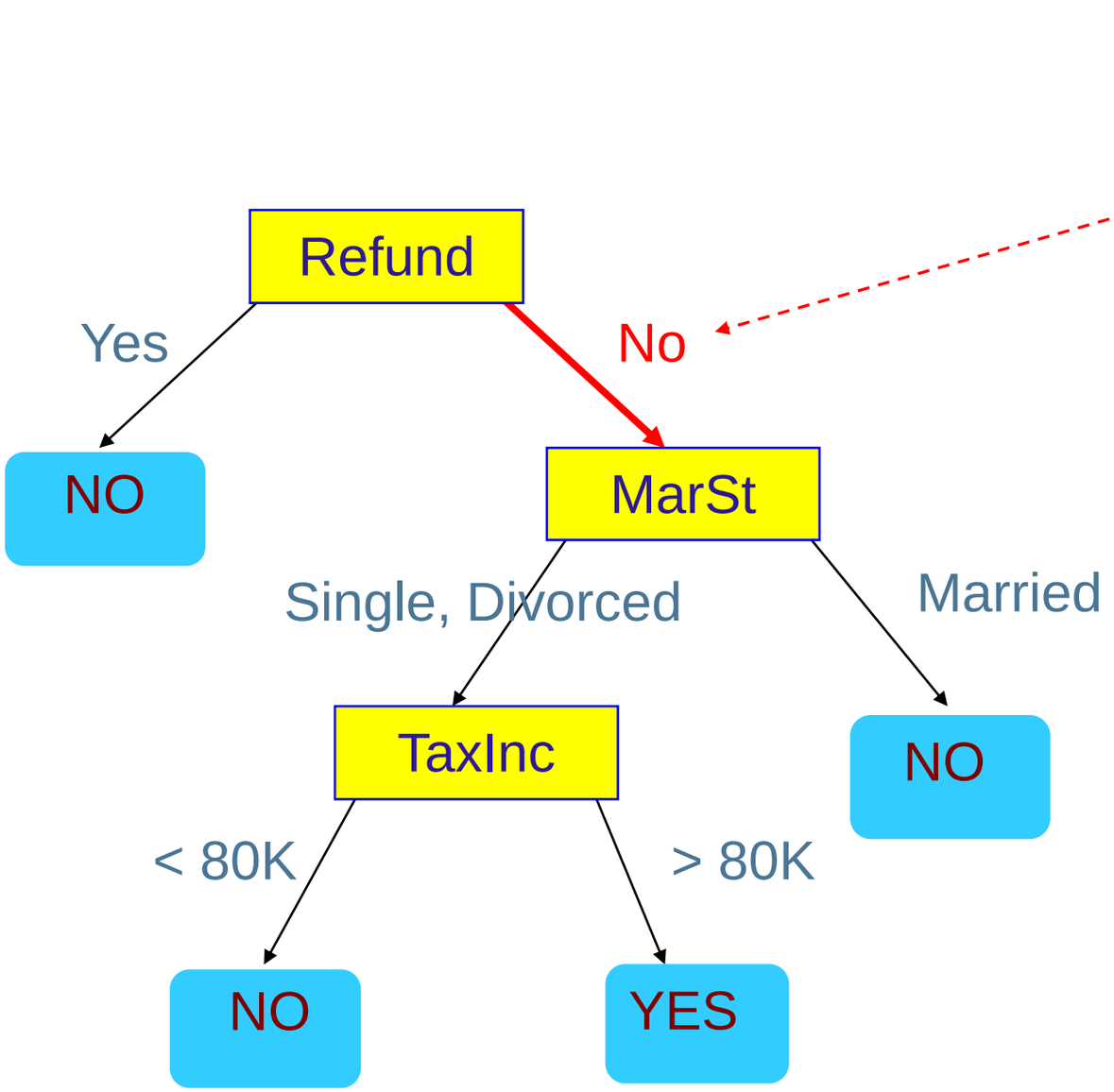
# Aplicando um modelo para testar os dados




| Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|
| No | Married | 80K | ? |

Test Data

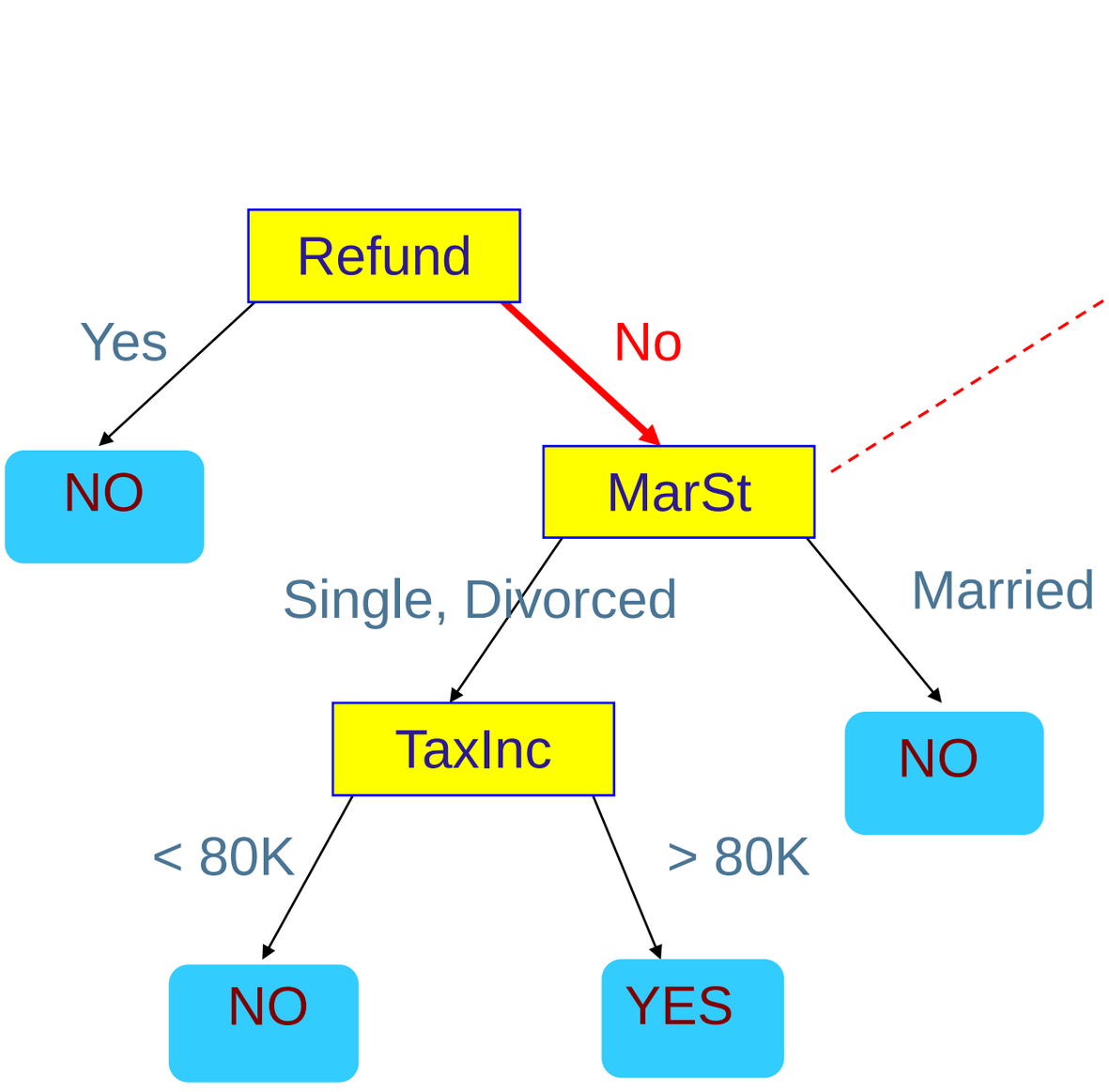
# Aplicando um modelo para testar os dados




| Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|
| No | Married | 80K | ? |

Test Data

# Aplicando um modelo para testar os dados



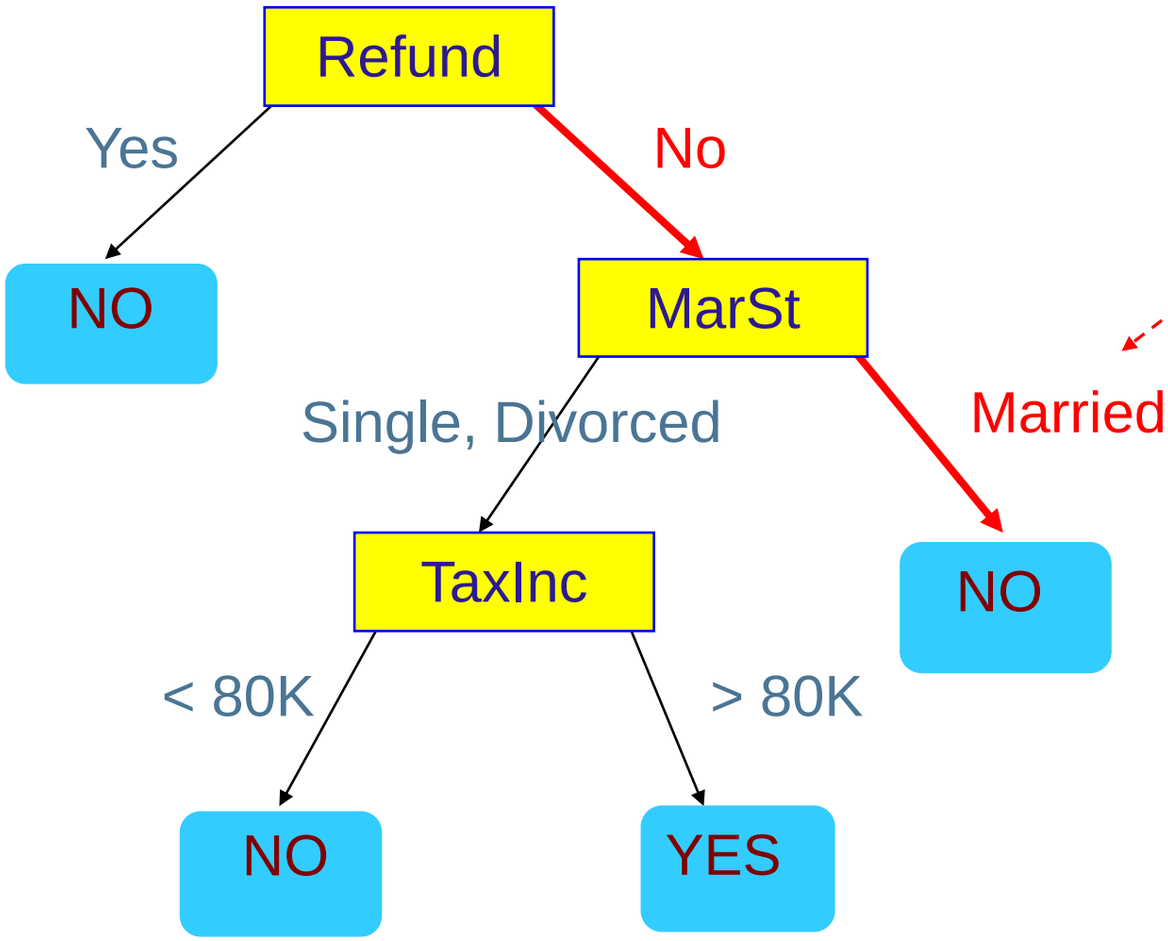

| Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|
| No | Married | 80K | ? |

Test Data

# Aplicando um modelo para testar os dados

Test Data

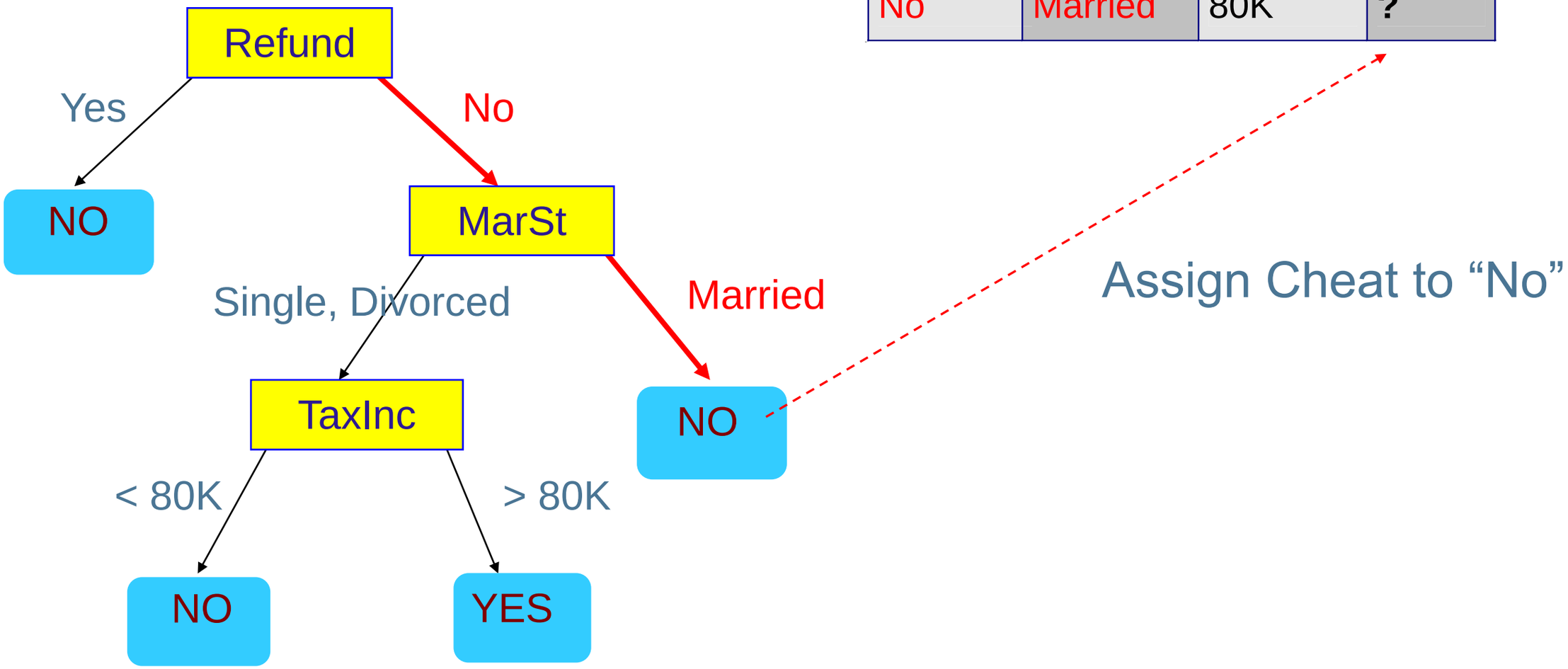
| Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|
| No | Married | 80K | ? |



# Aplicando um modelo para testar os dados

Test Data

| Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|
| No | Married | 80K | ? |



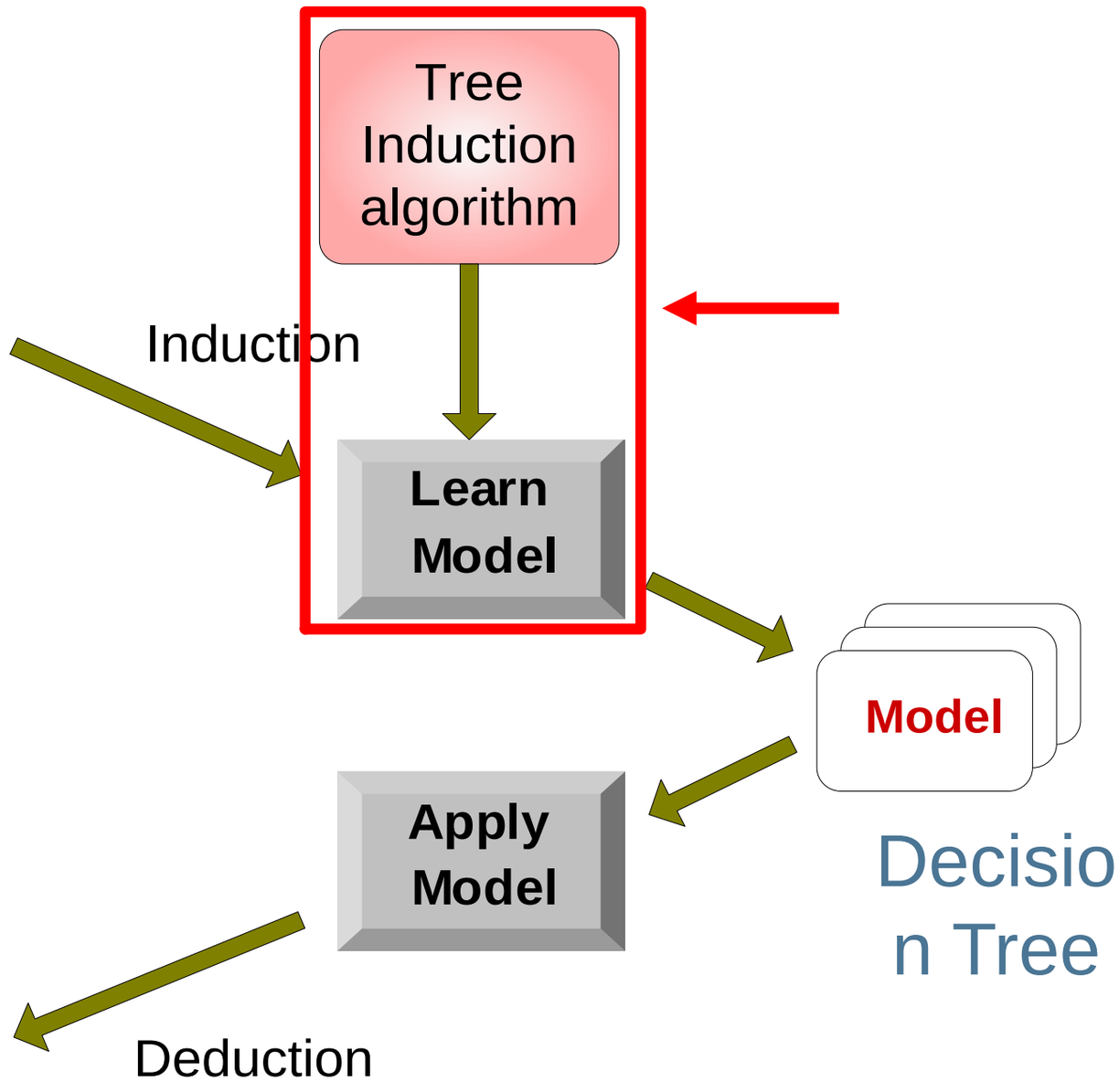
# Decision Tree Classification Task

| *Tid* | Attrib1 | Attrib2 | Attrib3 | 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 |

Training Set

| *Tid* | Attrib1 | Attrib2 | Attrib3 | Class |
|---|---|---|---|---|
| 11 | No | Small | 55K | ? |
| 12 | Yes | Medium | 80K | ? |
| 13 | Yes | Large | 110K | ? |
| 14 | No | Small | 95K | ? |
| 15 | No | Large | 67K | ? |

Test Set



# Indução de uma árvore de Decisão

- Muitos algoritmos
  - Hunt's Algorithm (one of the earliest)
  - CART
  - ID3, C4.5
  - SLIQ,SPRINT

# Algoritmo de Hunt

- Vamos ler no livro antes de discutir
- Peguem o livro na página 180 – 186.
- Após vamos discutir o que foi lido.

# Algoritmo básico (hunt)

1. Escolha um atributo
2. Estenda a árvore adicionando um ramo para cada valor do atributo
3. Passe os exemplos para as folhas (considerando o atributo escolhido)
4. Para cada folha
    1. Se todos os exemplos são da mesma classe, associe esta classe à folha
    2. Senão, repita os passos 1 a 4

# Hunt's Algorithm

- Árvore Inicial:

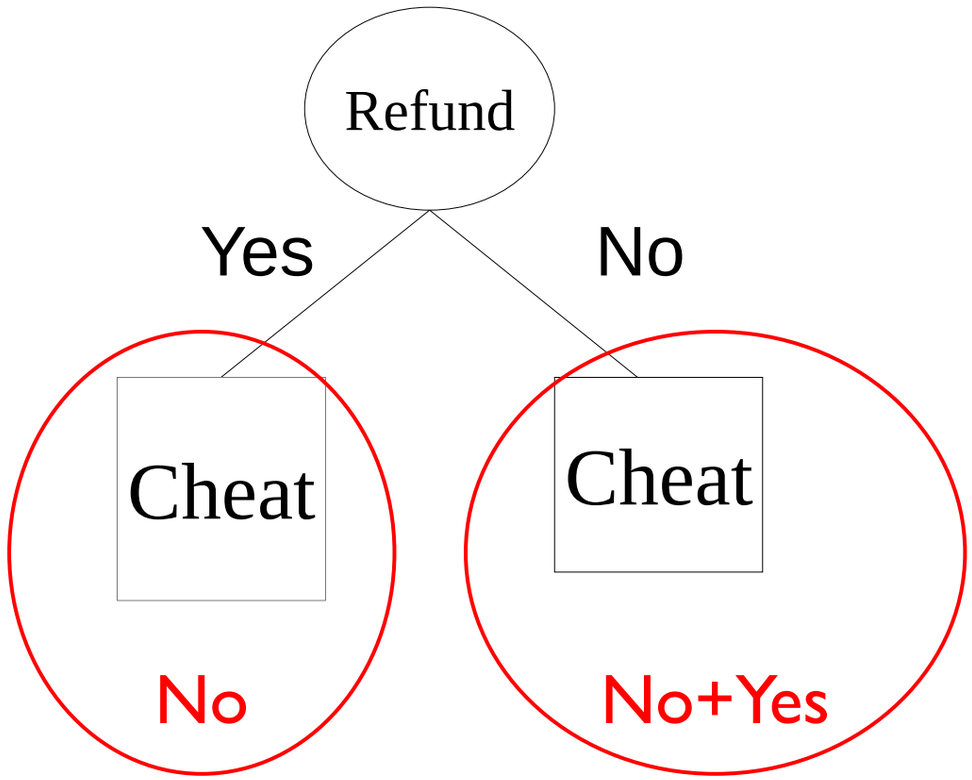
Cheat = NO

| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |

- => maioria de quem pega empréstimo paga seus débitos

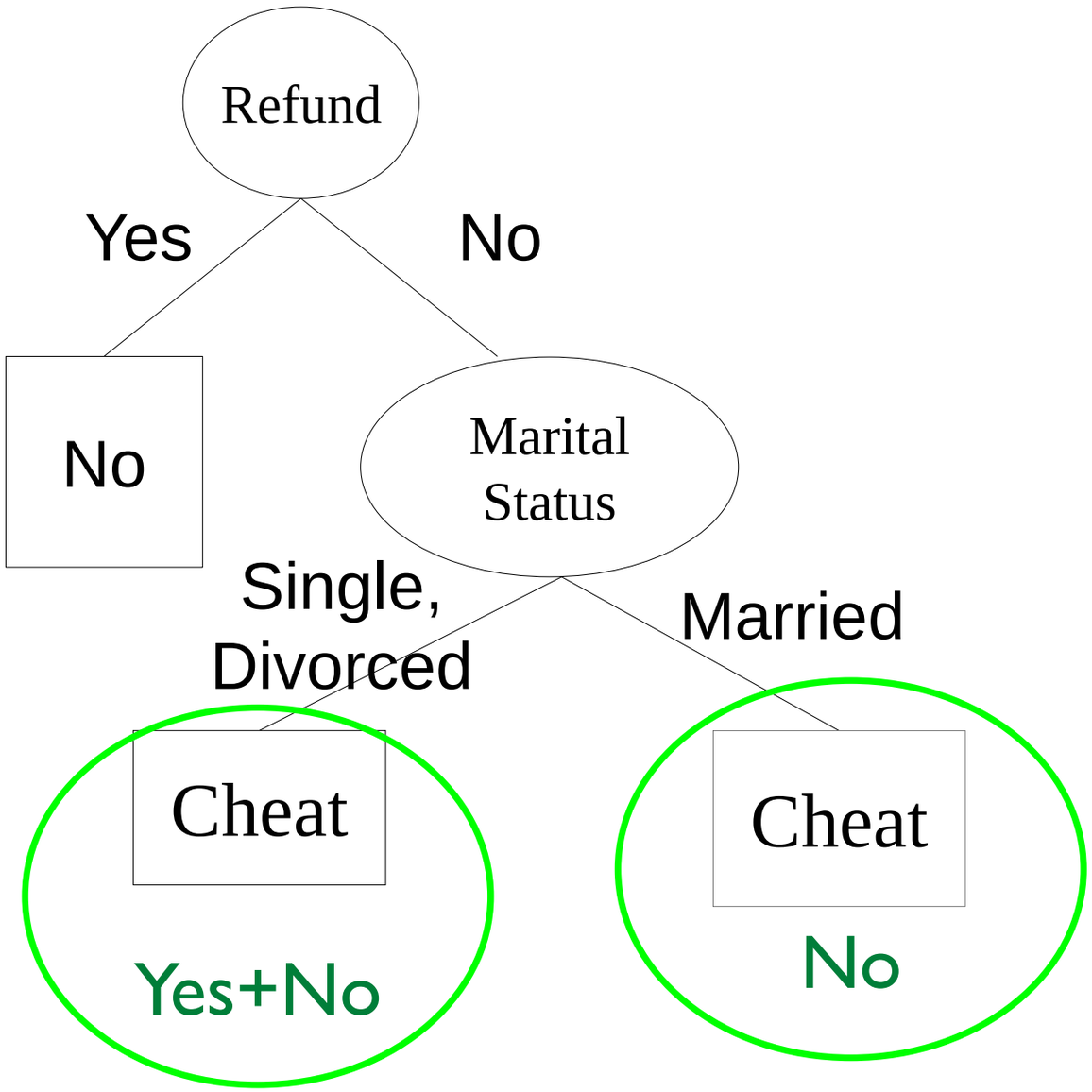
- Necessidade de refinar => a árvore contém registros da outra classe CHEAT = YES

# Hunt's Algorithm



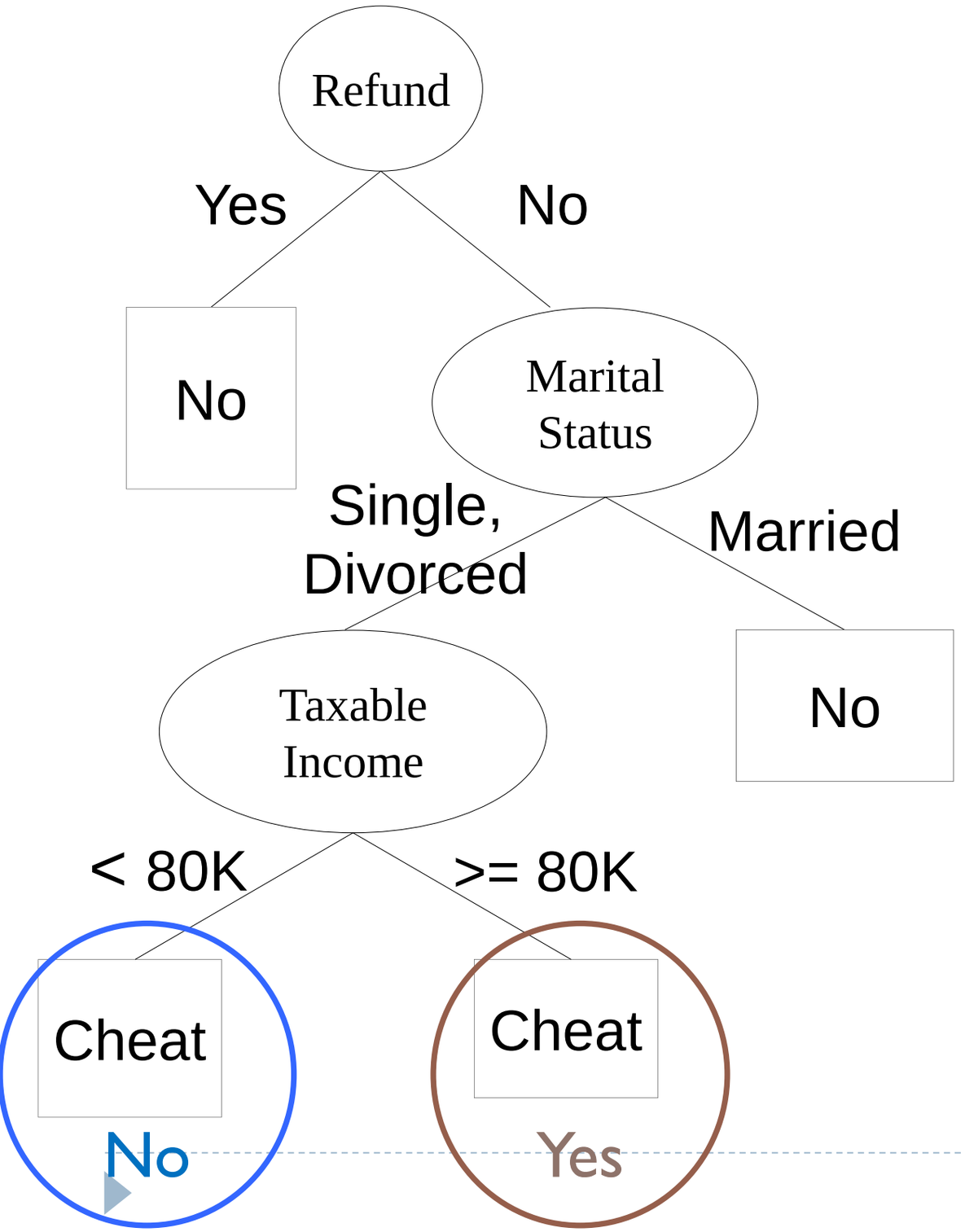

| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |

# Hunt's Algorithm



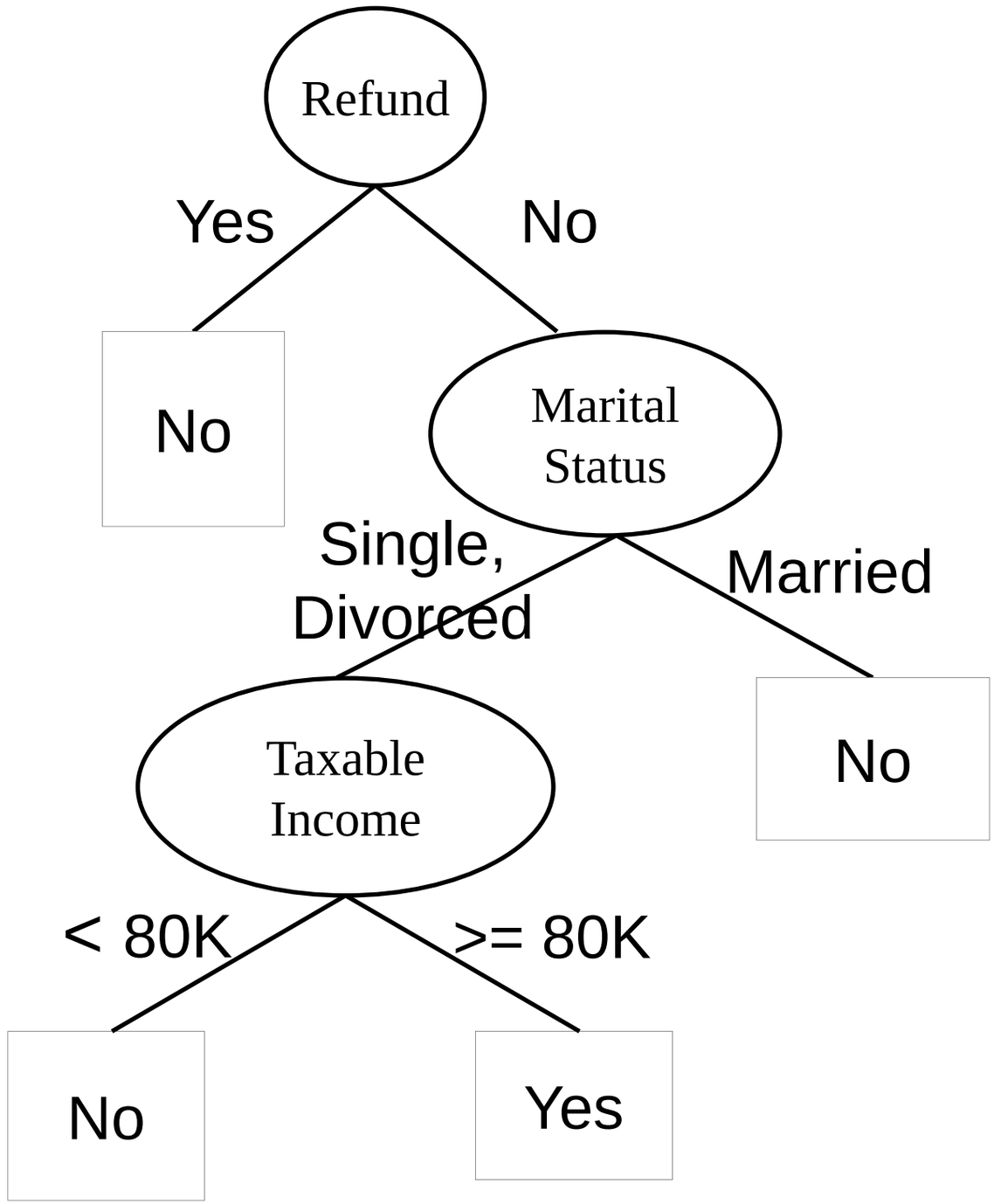

| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |

# Hunt's Algorithm




| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |

# Hunt's Algorithm




| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |

# Questões sobre a Indução

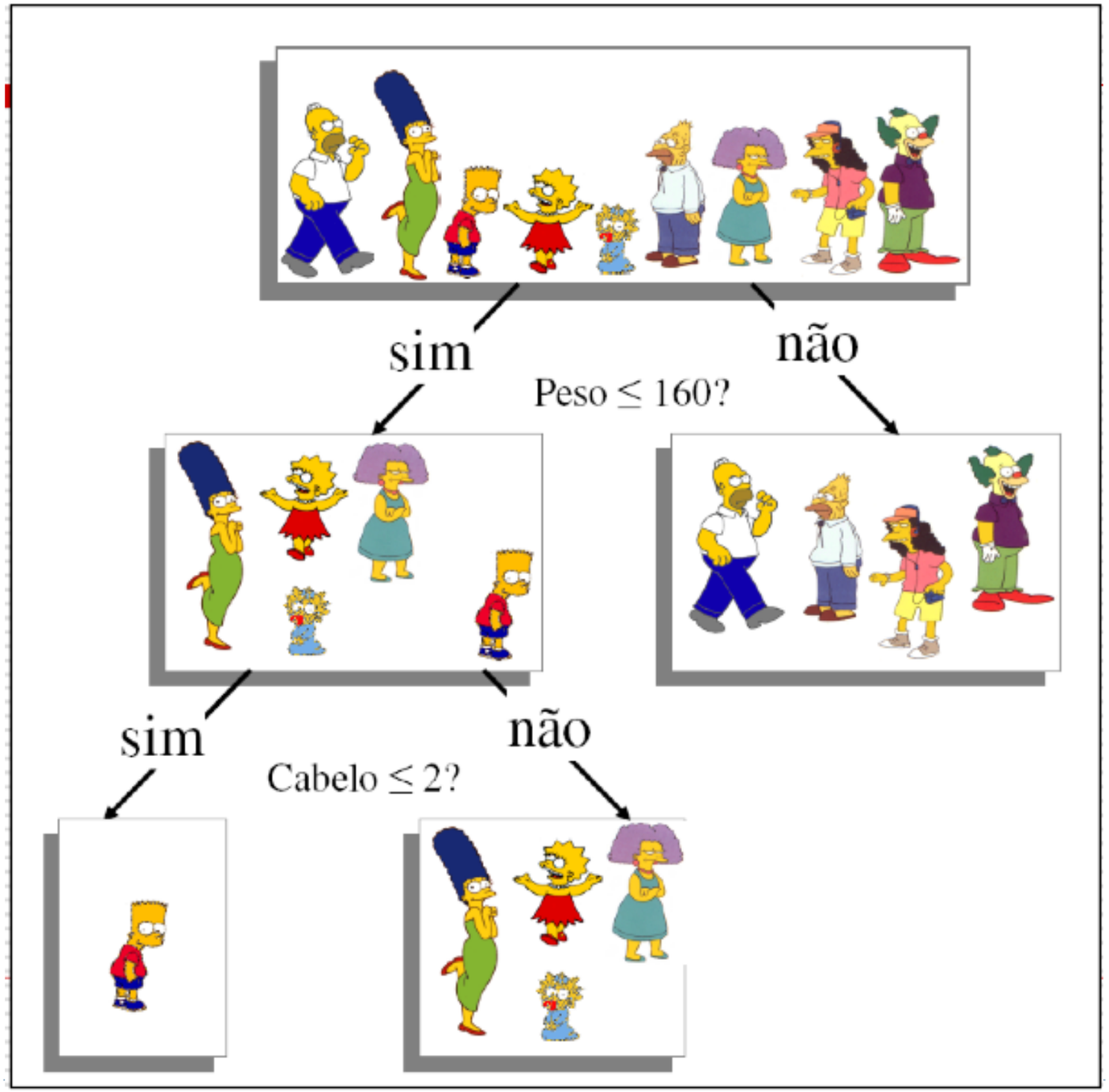
- Como os registros de treinamento devem ser tratados?
  - Cada passo recursivo do processo deve selecionar uma condição de teste de atributo para dividir os registros em subcojuntos menores. Qual o método???
- Como o procedimento de divisão deve parar??
  - Uma condição de parada é necessária para terminar o processo de crescimento de uma árvore. Uma estratégia é expandir até que só tenham elementos da mesma classe no nodo folha, mas nem sempre é possível.

- Como especificar a condição de teste de um atributo???

# exemplo: Como fazer o particionamento?

| Pessoa | Comprimento do Cabelo | Peso | Idade | Classe: Sexo |
|---|---|---|---|---|
| Homer | 0 | 250 | 36 | **M** |
| Marge | 10 | 150 | 34 | **F** |
| Bart | 2 | 90 | 10 | **M** |
| Lisa | 6 | 78 | 8 | **F** |
| Maggie | 4 | 20 | 1 | **F** |
| Abe | 1 | 170 | 70 | **M** |
| Selma | 8 | 160 | 41 | **F** |
| Otto | 10 | 180 | 38 | **M** |
| Krusty | 6 | 200 | 45 | **M** |

# exemplo: Como fazer o particionamento?

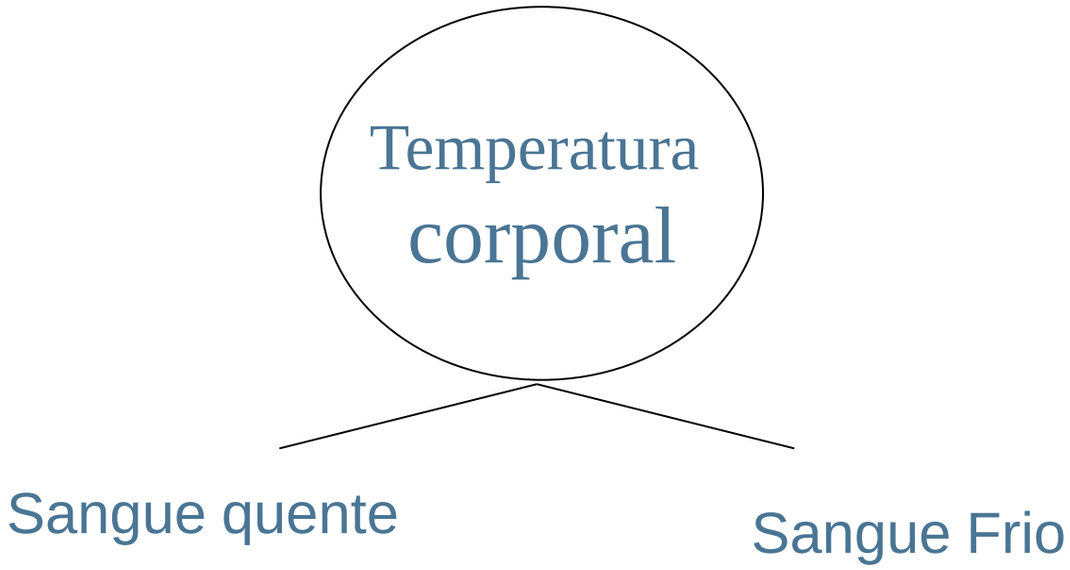


# Como especificar a condição de teste???

- Depende do tipo de atributo
  - Nominal
  - Ordinal
  - Contínuo
- Depende da forma que o atributo pode ser particionado
  - 2-way split
  - Multi-way split

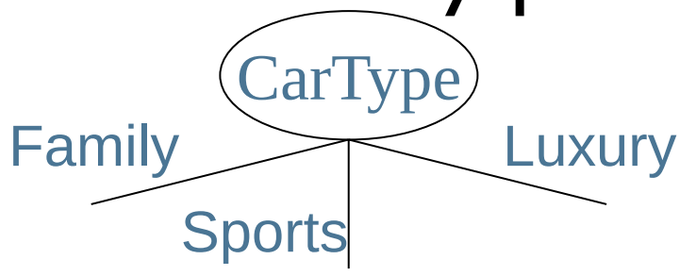
# Atributos binários

- A condição para um atributo binário gera 2 resultados possíveis:



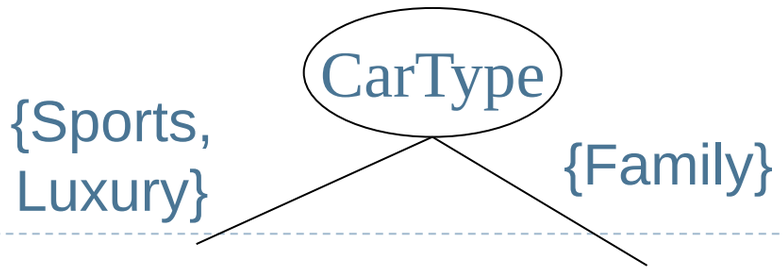
# Atributos Nominais e Ordinais

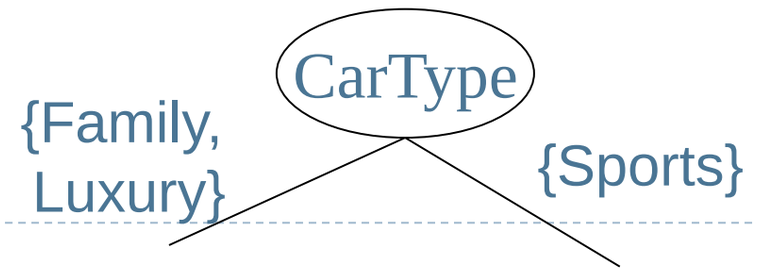
- Multi-way split: Use as many partitions as distinct values.




- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.

OR

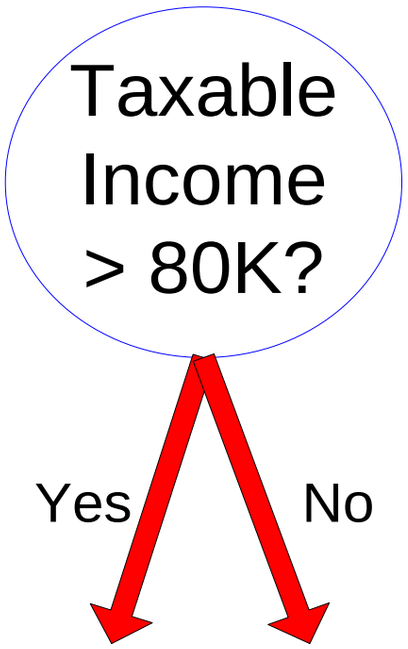


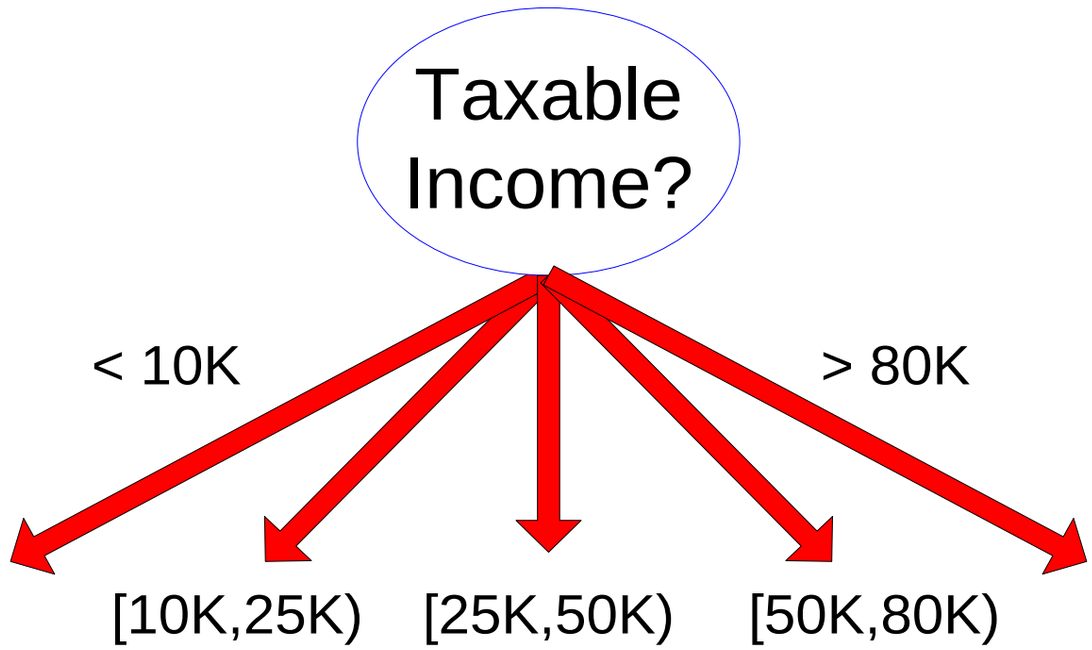
# Atributos Contínuos

- Discretization to form an ordinal categorical attribute
  - Static – discretize once at the beginning
  - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
- Binary Decision: $(A < v)$ or $(A \geq v)$
  - consider all possible splits and finds the best cut
  - can be more compute intensive

# Atributos Contínuos




(i) Binary split



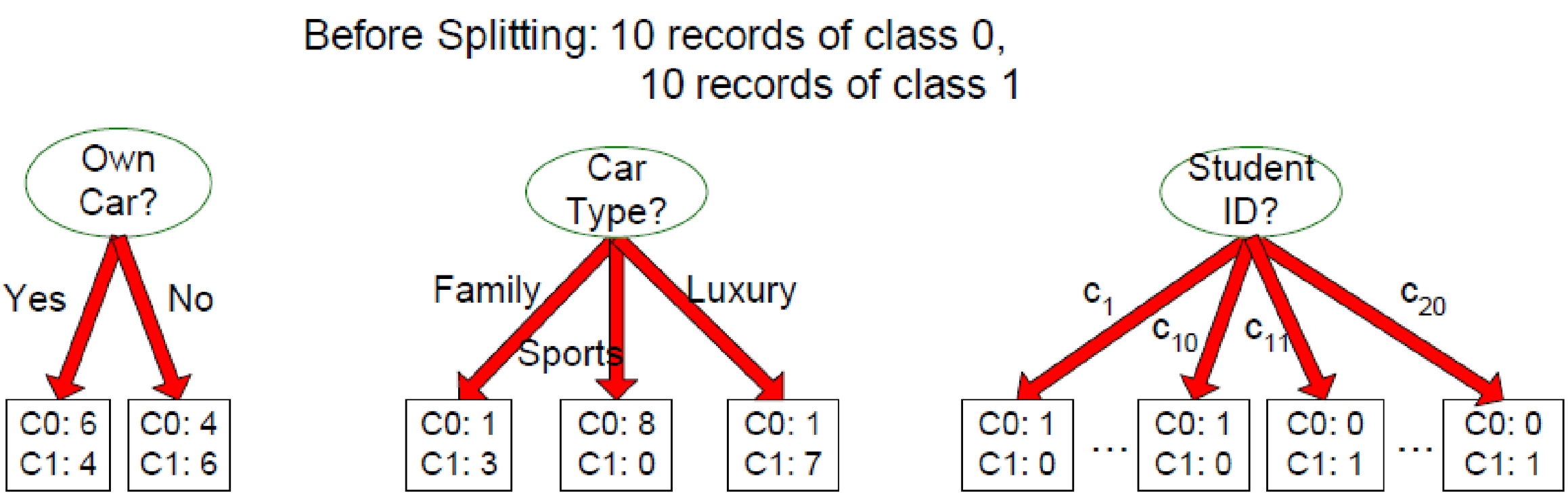

(ii) Multi-way split

- Como determinar a melhor divisão??

# atributo de particionamento

- Existem várias **possibilidades** para **escolha** do **atributo**
  - **Aleatória**: seleciona um atributo aleatoriamento
  - **Menos valores**: escolhe o atributo com menor número de valores possíveis
  - **Mais valores**: escolhe o atributo com o maior número de valores possíveis
  - **Ganho Máximo:** Selecione o atributo que possui o maior ganho de informação esperado, isto é, selecione o atributo que resultará no menor tamanho para as sub-árvores
  - **Indice GINI** (Breiman et al. 1984)
  - **Razão de Ganho** (Quinlan, 1993)

# Como determinar a melhor divisão?




Which test condition is the best?

# Como determinar a melhor divisão?

- São definidos em termos da distribuição da classe dos registros antes e depois da divisão
  - O ideal é ser o mais **heterogêneo**
  - É necessária uma medida de impureza.

| C0: 5<br>C1: 5 | C0: 9<br>C1: 1 |
|---|---|
| Non-homogeneous, | Homogeneous, |
| High degree of impurity | Low degree of impurity |
| Impureza > 0 | Impureza ~ 0 |

# Medidas de impureza

- Gini Index
- Entropy
- Misclassification error

# Examples for computing GINI

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

| C1 | **0** |
|---|---|
| C2 | **6** |

P(C1) = 0/6 = 0  P(C2) = 6/6 = 1

Gini = 1 – $P(C1)^2$ – $P(C2)^2$ = 1 – 0 – 1 = 0

| C1 | **1** |
|---|---|
| C2 | **5** |

P(C1) = 1/6  P(C2) = 5/6

Gini = 1 – $(1/6)^2$ – $(5/6)^2$ = 0.278

| C1 | **2** |
|---|---|
| C2 | **4** |

P(C1) = 2/6  P(C2) = 4/6

Gini = 1 – $(2/6)^2$ – $(4/6)^2$ = 0.444

Quando um nó *p* é dividido em *k* partições (filhos), a qualidade da divisão é calculada como,

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

onde, $n_i$ = número de registros no filho i,

n = número de registros no nó p.

Multi-way split

| | CarType | | |
|---|---|---|---|
| | Family | Sports | Luxury |
| C1 | 1 | 2 | 1 |
| C2 | 4 | 1 | 1 |
| Gini | 0.393 | | |

Two-way split
(find best partition of values)

| | CarType | |
|---|---|---|
| | {Sports, Luxury} | {Family} |
| C1 | 3 | 1 |
| C2 | 2 | 4 |
| Gini | 0.400 | |

| | CarType | |
|---|---|---|
| | {Sports} | {Family, Luxury} |
| C1 | 2 | 2 |
| C2 | 1 | 5 |
| Gini | 0.419 | |

$$\frac{5}{10}\left[1-\left(\frac{3}{5}\right)^2-\left(\frac{2}{5}\right)^2\right]+\frac{5}{10}\left[1-\left(\frac{1}{5}\right)^2-\left(\frac{4}{5}\right)^2\right]$$

# Examples for computing Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

| C1 | **0** |
|---|---|
| C2 | **6** |

P(C1) = 0/6 = 0  P(C2) = 6/6 = 1

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

| C1 | **1** |
|---|---|
| C2 | **5** |

P(C1) = 1/6  P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

5

| C1 | **2** |
|---|---|
| C2 | **4** |

P(C1) = 2/6  P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

# Algoritmo de indução

E = registros
F = atributos

CrescimentodaArvore(E,F)

1. **SE (cond_parada(E,F)= verdadeiro then**
2. Folha = criarNodo()
3. Folha.rotulo=classificar(E)
4. Retorna folha
5. **Senão**
6. Raiz=criarNodo()
7. Raiz.cond_teste=encontrar_melhor_divisão(E,F)
8. Atribuir V={v | v é um resultado possível de raiz.cond_teste}
9. Para cada v E V faça
10. Ev={e | raiz.cond_teste(e) = v e e pertence E}
11. Filho=crescimentodaarvore(Ev,F)
12. Adicionar filho como descendente de raiz e rotule a aresta
13. FimPara
14. FimSe
15. Retornar Raiz

# Critério de parada

- Stop expanding a node when all the records belong to the same class

- Stop expanding a node when all the records have similar attribute values

- Early termination (to be discussed later)

# Avaliação do modelo

```
Confusion Matrix and Statistics

                 Reference
Prediction        Iris.setosa Iris.versicolor Iris.virginica
  Iris.setosa              13               0              0
  Iris.versicolor           0              13              1
  Iris.virginica            0               0             12
```

```
Overall Statistics

               Accuracy : 0.9744
                 95% CI : (0.8652, 0.9994)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : < 2.2e-16
```

$$\text{recall} = \frac{\text{number of documents retrieved that are relevant}}{\text{total number of documents that are relevant}}$$

$$\text{precision} = \frac{\text{number of documents retrieved that are relevant}}{\text{total number of documents that are retrieved}}.$$

# Avaliação do modelo

```
Statistics by Class:

                     Class: Iris.setosa Class: Iris.versicolor
Sensitivity          1.0000                          1.0000
Specificity          1.0000                          0.9615
Pos Pred Value       1.0000                          0.9286
Neg Pred Value       1.0000                          1.0000
Prevalence           0.3333                          0.3333
Detection Rate       0.3333                          0.3333
Detection Prevalence 0.3333                          0.3590
Balanced Accuracy    1.0000                          0.9808
                     Class: Iris.virginica
Sensitivity                         0.9231
Specificity                         1.0000
Pos Pred Value                      1.0000
```

Recall (Pos)

Recall (Neg)

Precision (Pos)

Precision (Neg)

$$\text{recall} = \frac{\text{number of documents retrieved that are relevant}}{\text{total number of documents that are relevant}}$$

$$\text{precision} = \frac{\text{number of documents retrieved that are relevant}}{\text{total number of documents that are retrieved}}.$$

# Avaliação do modelo

| | Iris.setosa | Iris.versicolor | Iris.virginica |
|---|---|---|---|
| 1 | 1.000000e+00 | 1.925186e-09 | 2.256862e-10 |
| 2 | 9.999999e-01 | 7.484844e-08 | 2.807169e-09 |
| 3 | 1.000000e+00 | 9.242904e-09 | 3.748854e-08 |
| 4 | 9.999943e-01 | 5.176429e-06 | 5.014183e-07 |
| 5 | 1.000000e+00 | 5.880422e-10 | 2.821031e-09 |
| 6 | 1.000000e+00 | 9.492960e-09 | 1.110844e-08 |
| 7 | 1.000000e+00 | 5.964081e-09 | 4.617001e-10 |
| 8 | 9.999998e-01 | 1.998648e-07 | 8.309491e-09 |
| 9 | 9.999998e-01 | 1.998648e-07 | 8.309491e-09 |
| 10 | 1.000000e+00 | 9.540811e-09 | 2.391038e-09 |
| 11 | 9.999947e-01 | 4.744178e-06 | 5.297235e-07 |
| 12 | 9.999998e-01 | 2.356624e-07 | 1.045023e-08 |
| 13 | 1.000000e+00 | 1.352047e-10 | 9.563372e-10 |

...

39

# Cálculo de Precision e Recall

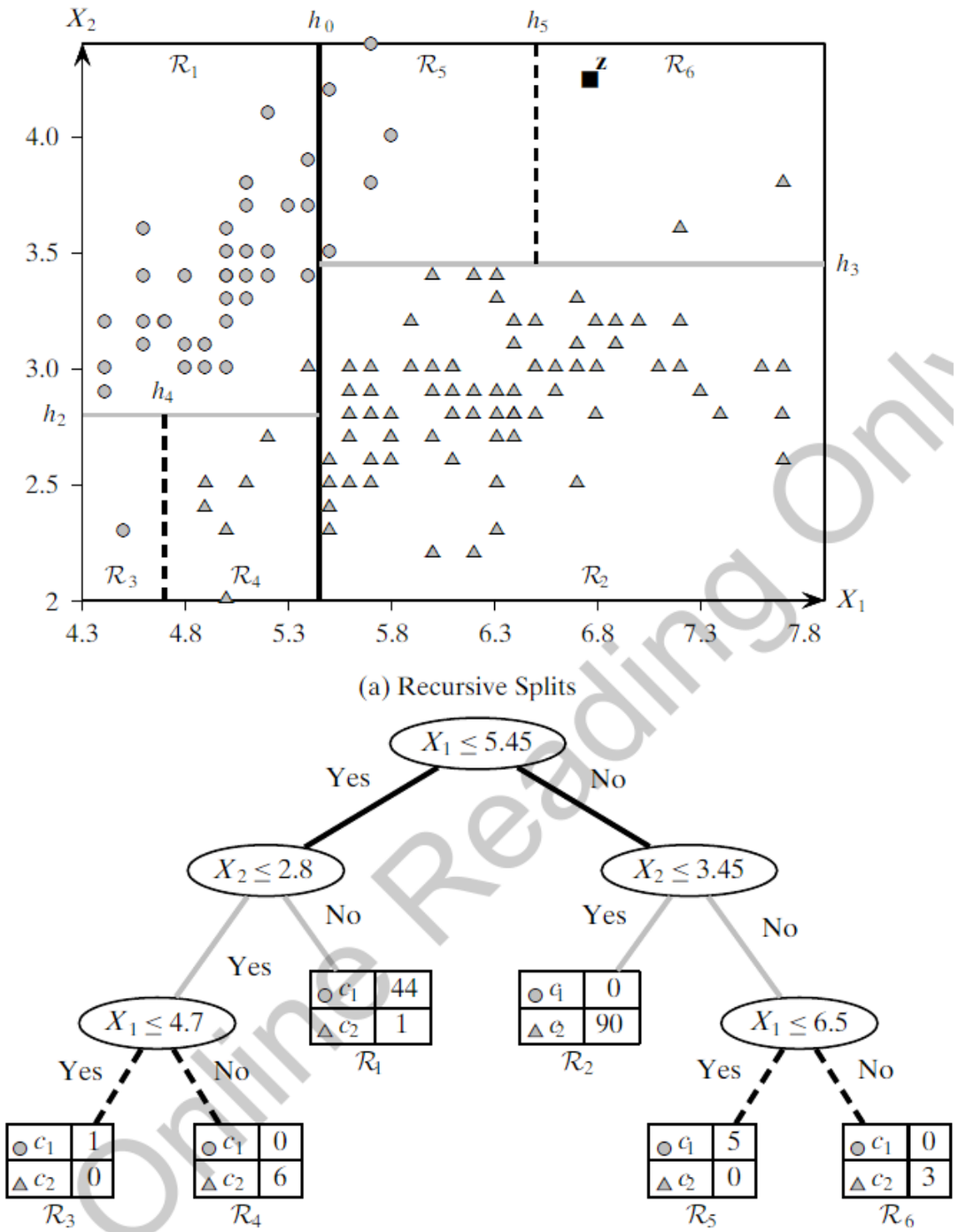
- Descubra como são calculados os valores de precision e recall para o exemplo realizado

- Mostre os cálculos de como voce chegou aos valores da tabela!!!

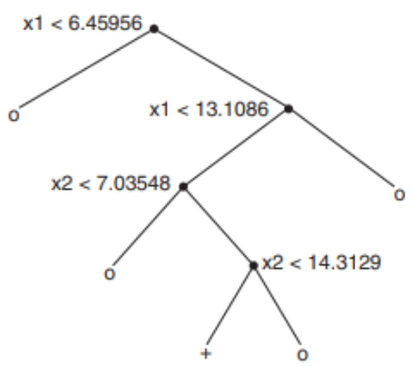
# Exercício da semana

- Explique cada uma das medidas de impureza abaixo, relacionando-as com a entropia
  - Information Gain
  - Gain Ratio
- Exercite o algoritmo de indução de árvores considerando o Information Gain como medida de impureza e a altura da árvore igual a 2 como critério de parada
  - Construa o modelo de classificação apresentando os cálculos que determinam a escolha dos atributos dos nós internos
  - Apresente a acurácia do modelo para o mesmo dataset

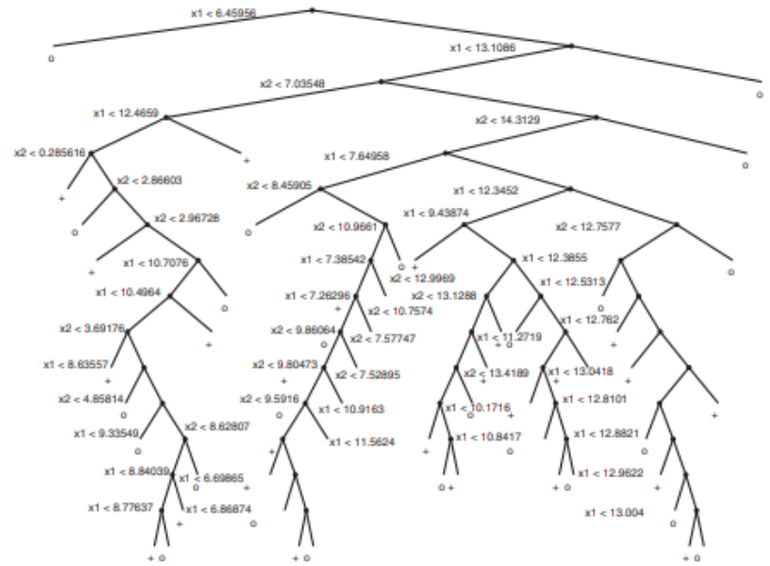
# Fronteiras de decisão



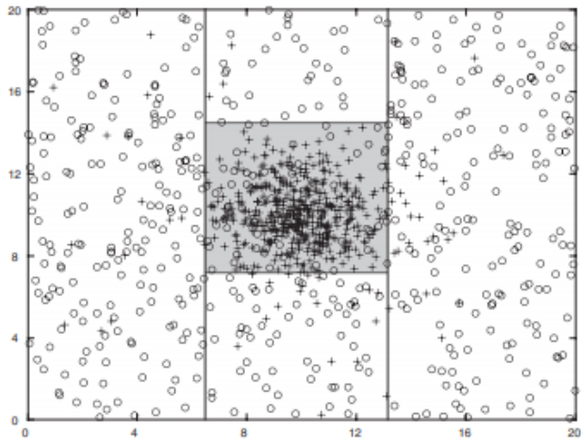
# Sobreajuste (overfitting)



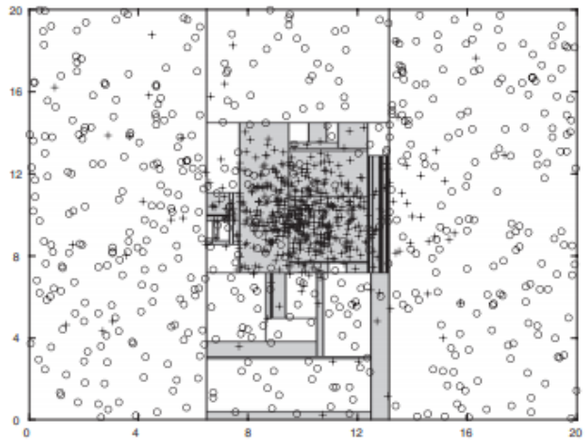

(a) Decision tree with 5 leaf nodes.



(b) Decision tree with 50 leaf nodes.



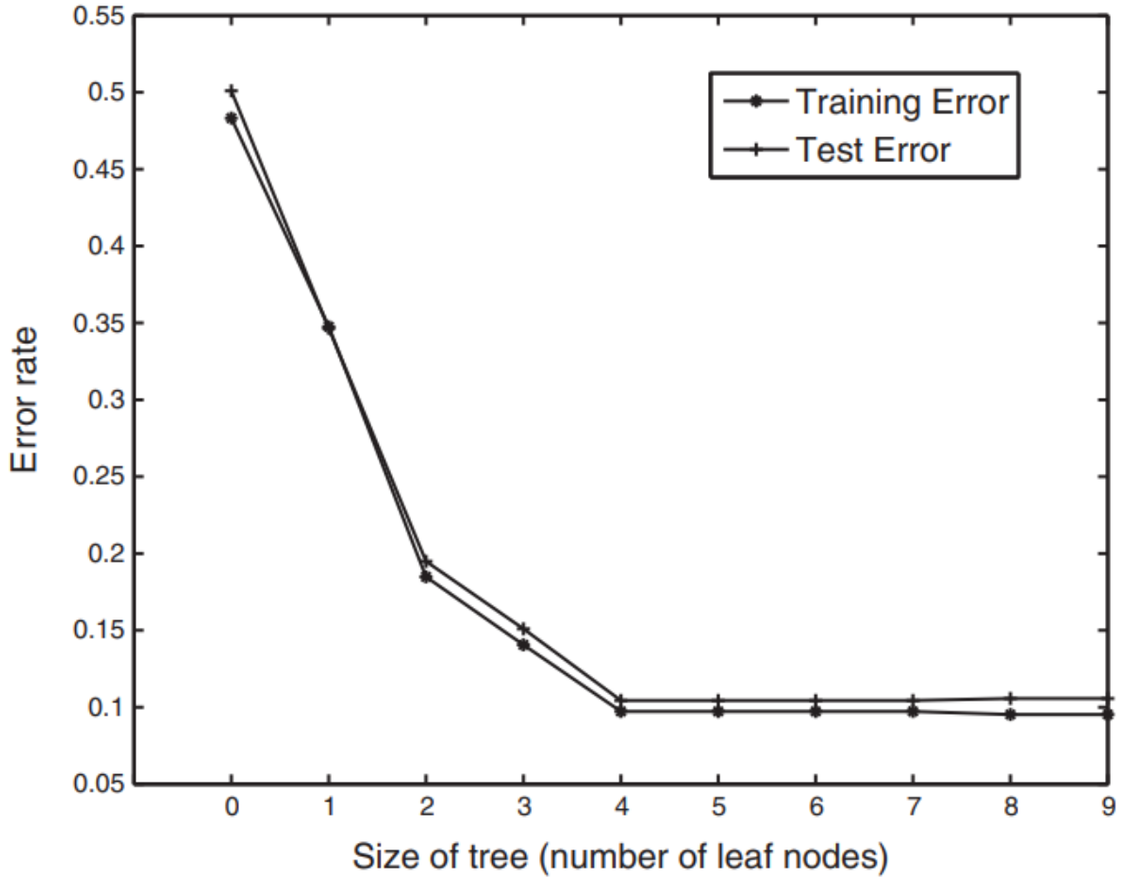
(c) Decision boundary for tree with 5 leaf nodes.



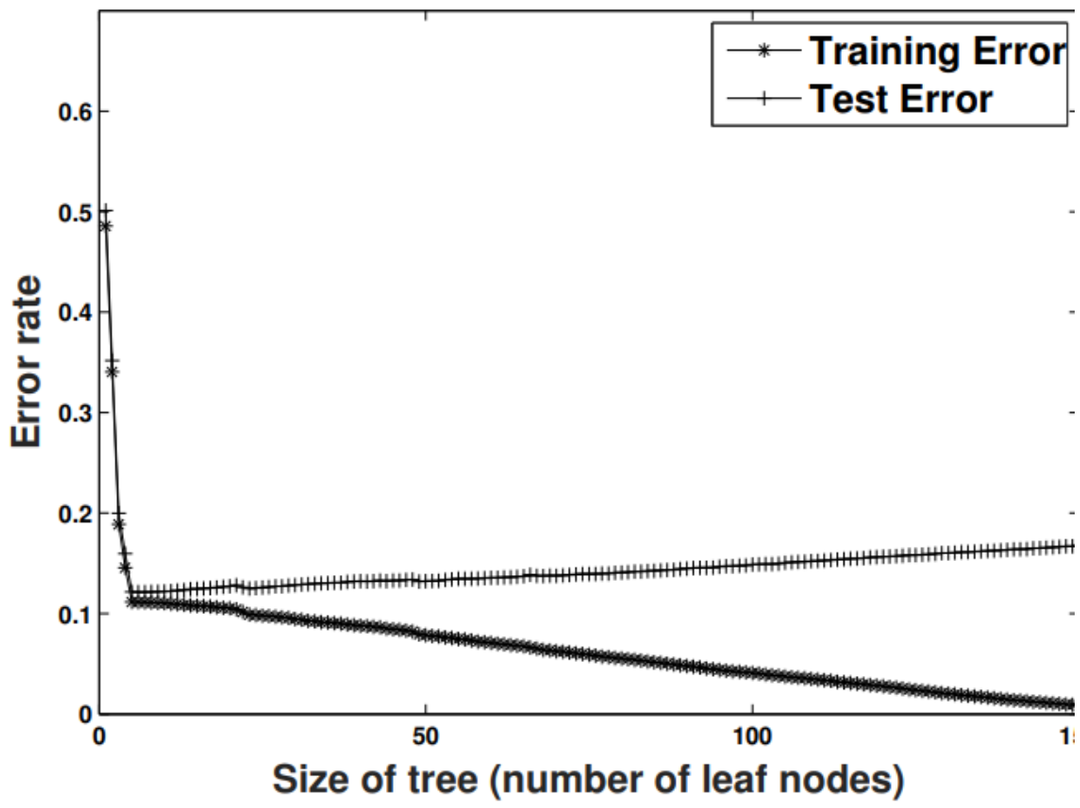
(d) Decision boundary for tree with 50 leaf nodes.

**Figure 3.24.** Decision trees with different model complexities

# Sobreajuste (overfitting)



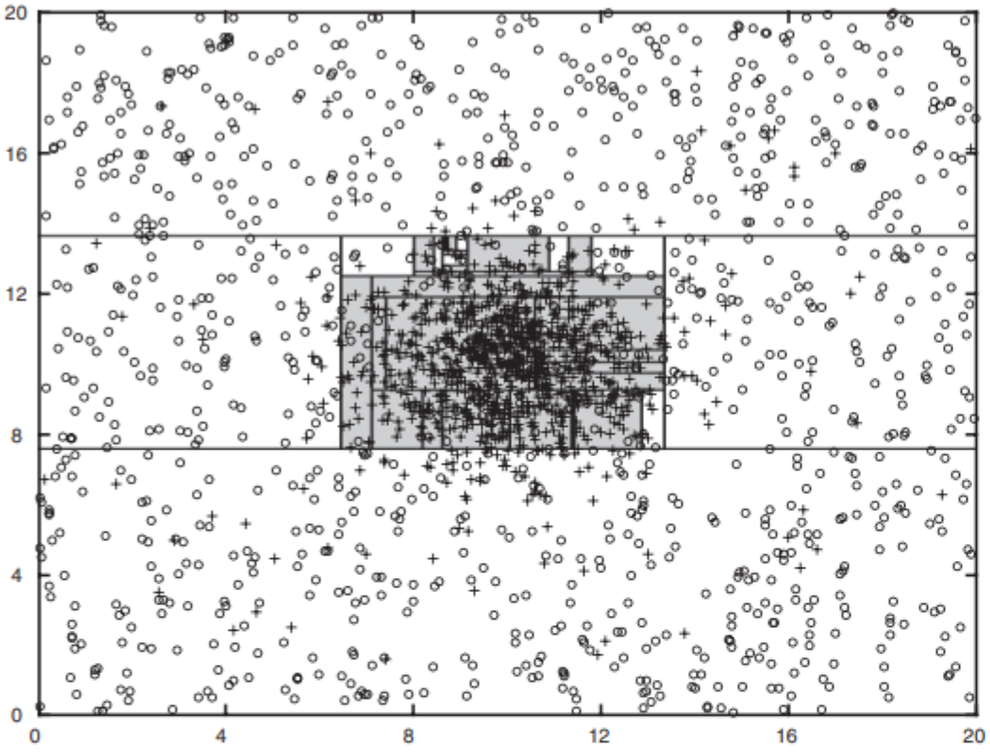

(a) Varying tree size from 1 to 8.



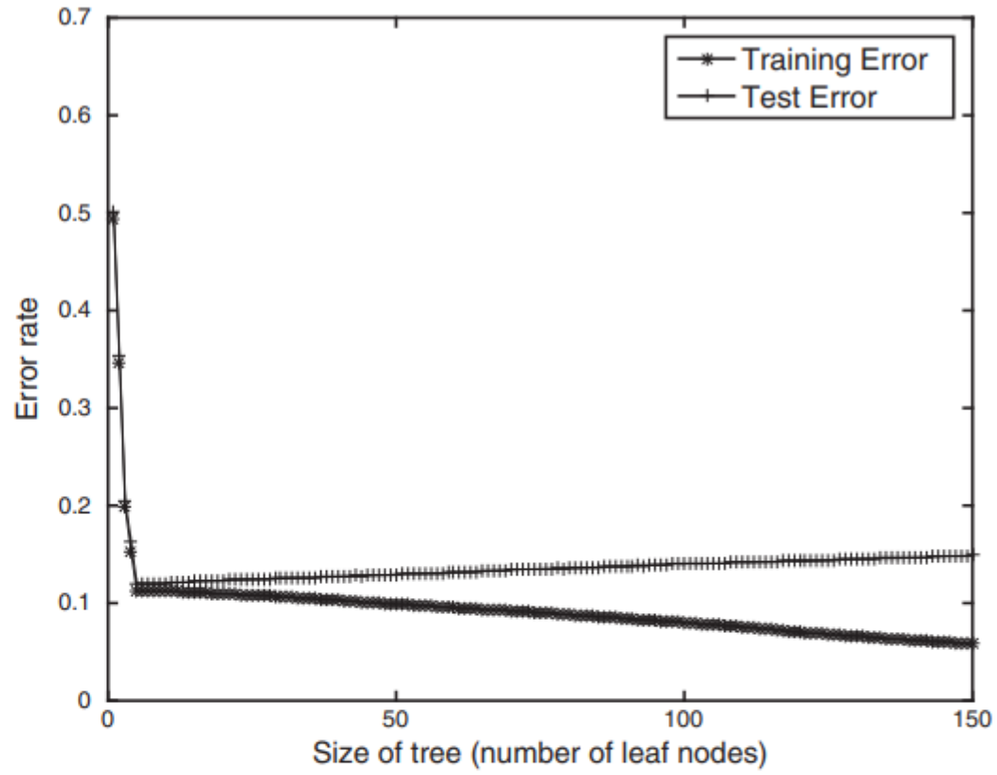

(b) Varying tree size from 1 to 150.

**Figure 3.23.** Effect of varying tree size (number of leaf nodes) on training and test errors.

# Sobreajuste (overfitting)




(a) Decision boundary for tree with 50 leaf nodes using 20% data for training.




(b) Training and test error rates using 20% data for training.

**Figure 3.25.** Performance of decision trees using 20% data for training (twice the original training size)