Aprendizado de Máquina:
Árvore de Decisão

Prof. Arnaldo Candido Junior
UTFPR – Medianeira

Visão geral

- **Arvores de decisão** (ADs)
  - Utilizam a estratégia de divisão e conquista
  - Recursivamente: dividir problemas difíceis em problemas mais simples
  - É uma das técnicas mais utilizadas

Visão geral (2)

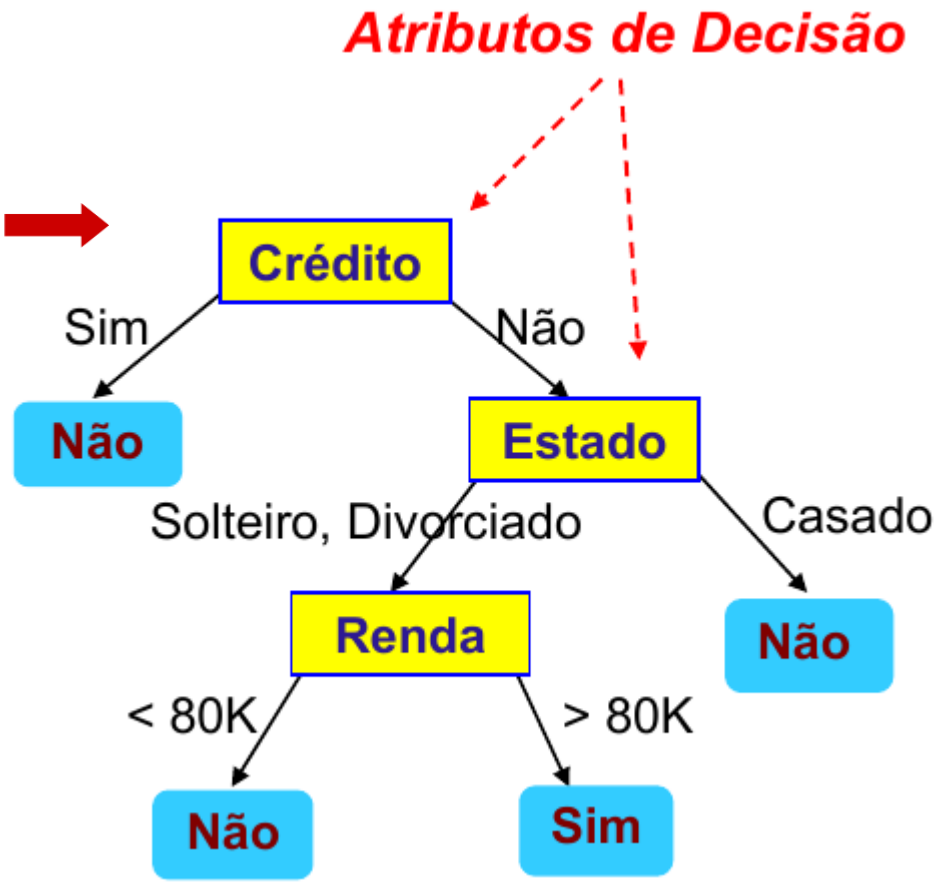
- Composta por:
  - Nó raiz: nenhum pai e com n ≥ 0 filhos
  - Nós intermediários: um pai e n ≥ 2 filhos
  - Nós folhas: um pai e nenhum filho

Exemplo: deve dinheiro?

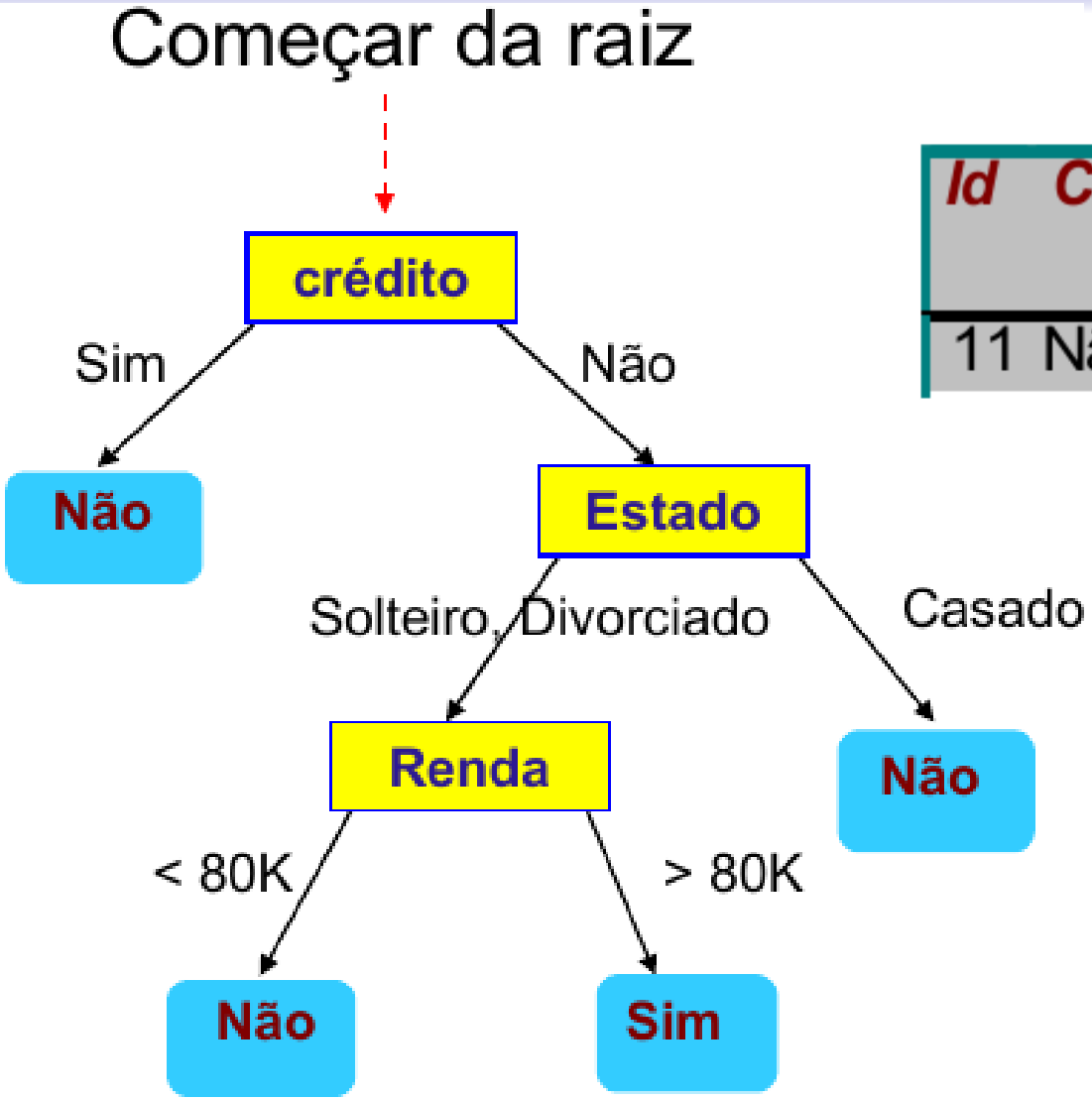
Dados de treinamento

Crédito	Estado Civil	Renda	Deve
Sim	Solteiro	125K	Não
Não	Casado	100K	Não
Não	Solteiro	70K	Não
Sim	Casado	120K	Não
Não	Divorciado	95K	Sim
Não	Casado	60K	Não
Sim	Divorciado	220K	Não
Não	Solteiro	85K	Sim
Não	Casado	75K	Não
Não	Solteiro	90K	Sim

Modelo induzido



Exemplo: deve dinheiro (2)



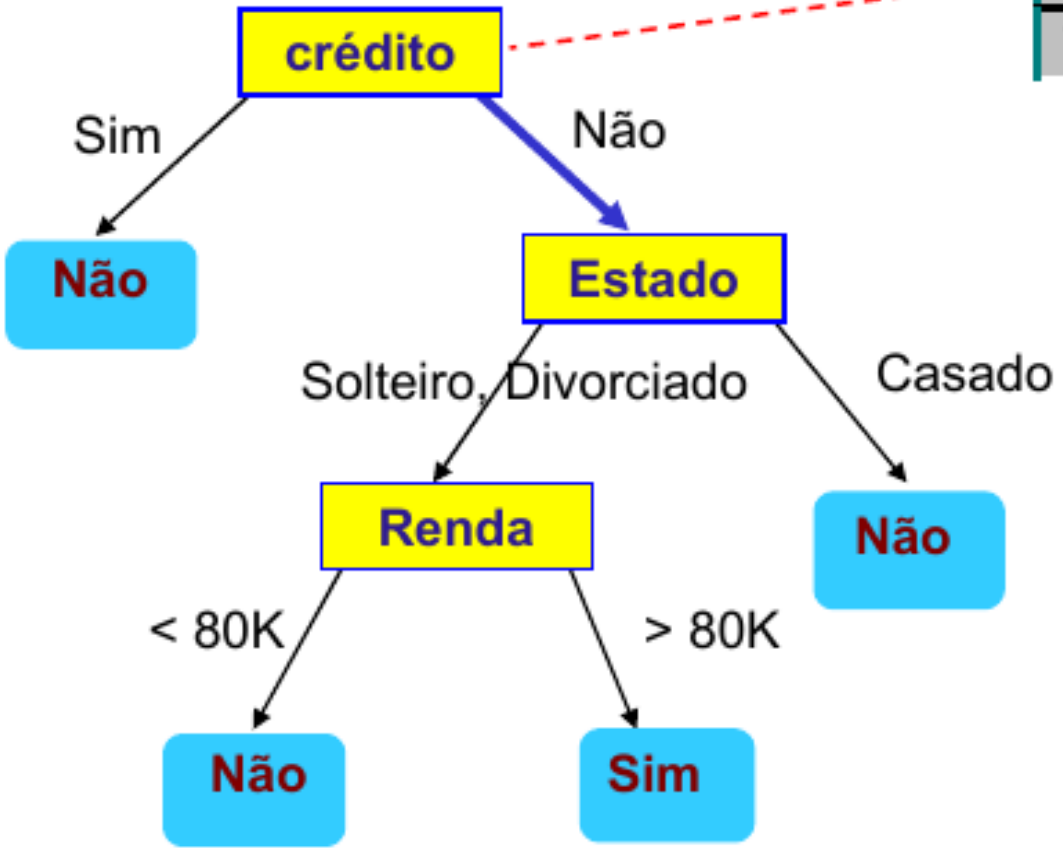
Dados de teste

Id	Crédito	Estado Civil	Renda	Deve
11	Não	Casado	80K	?

Exemplo: deve dinheiro (3)

Dados de teste

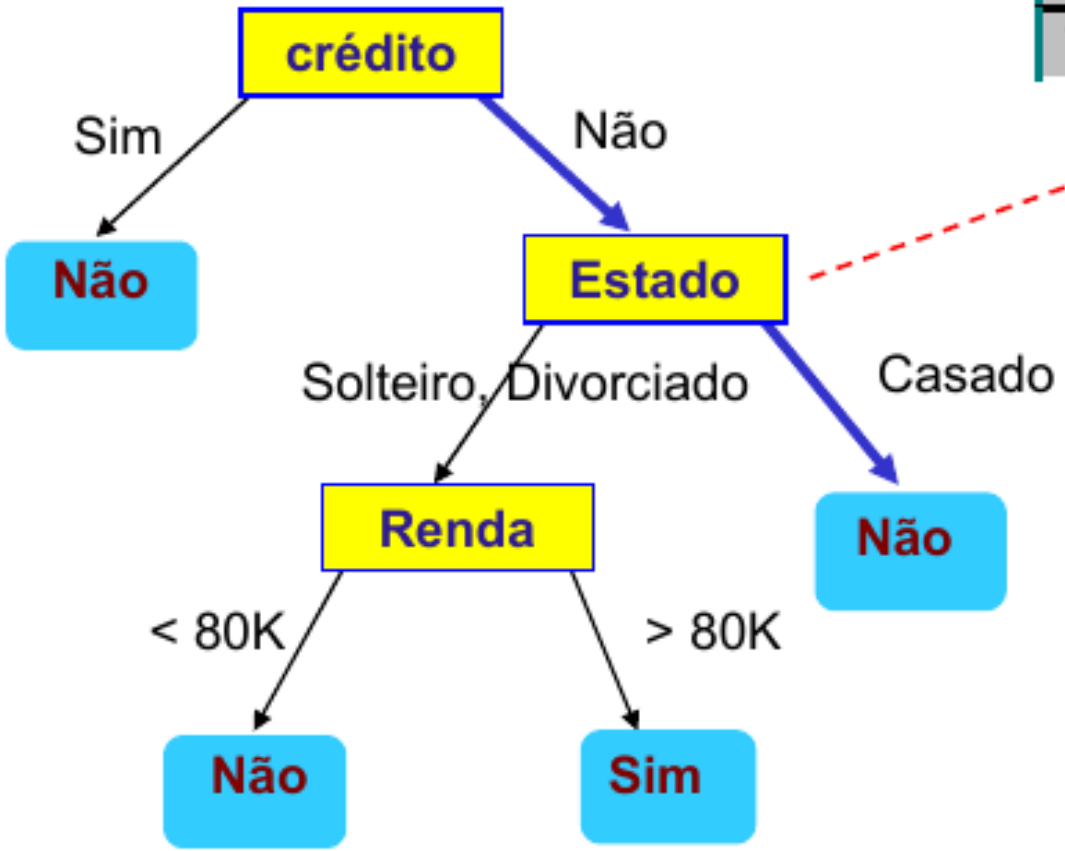
Id	Crédito	Estado Civil	Renda	Deve
11	Não	Casado	80K	?



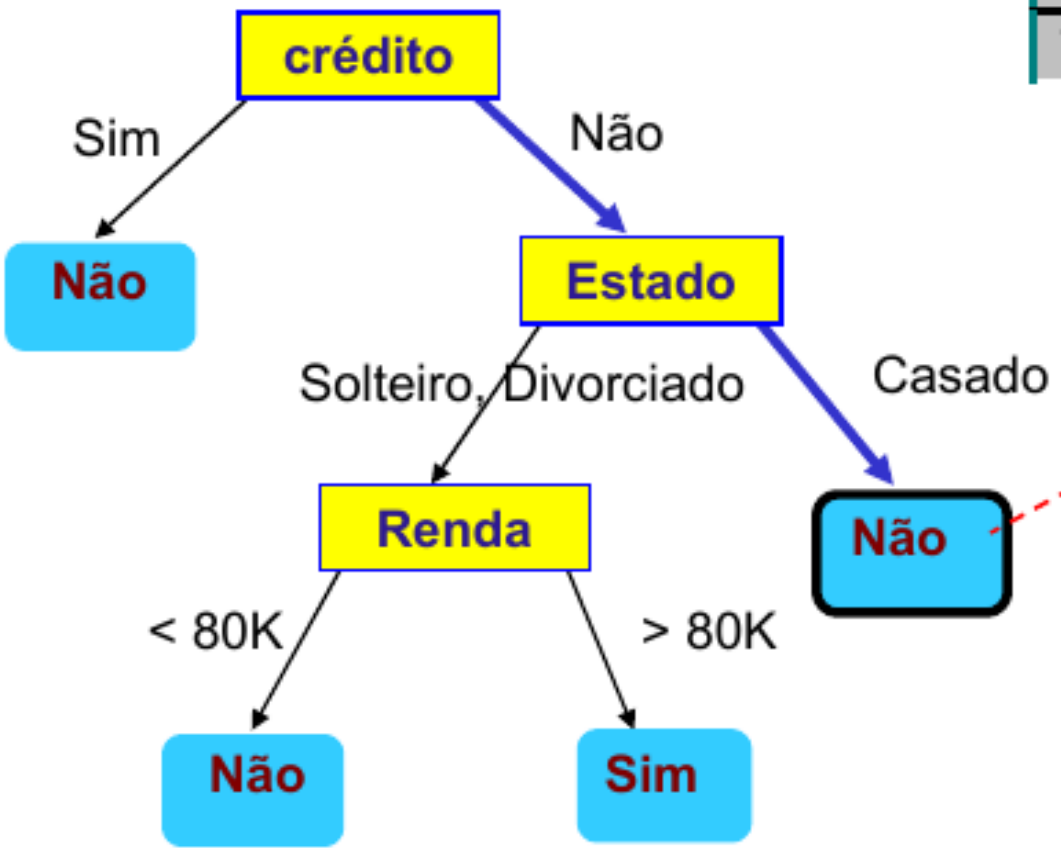
Exemplo: deve dinheiro (4)

Dados de teste

Id	Crédito	Estado Civil	Renda	Deve
11	Não	Casado	80K	?



Exemplo: deve dinheiro (5)



Dados de teste

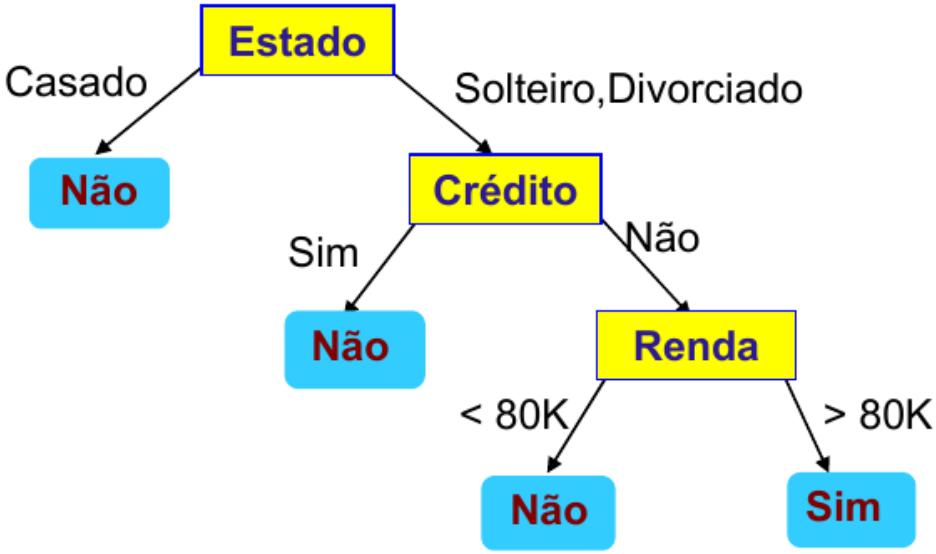
Id	Crédito	Estado Civil	Renda	Deve
11	Não	Casado	80K	?

Atribui classe **não**

Exemplo: deve dinheiro (6)

- Mais de uma árvore pode ser reajustada para os mesmos dados

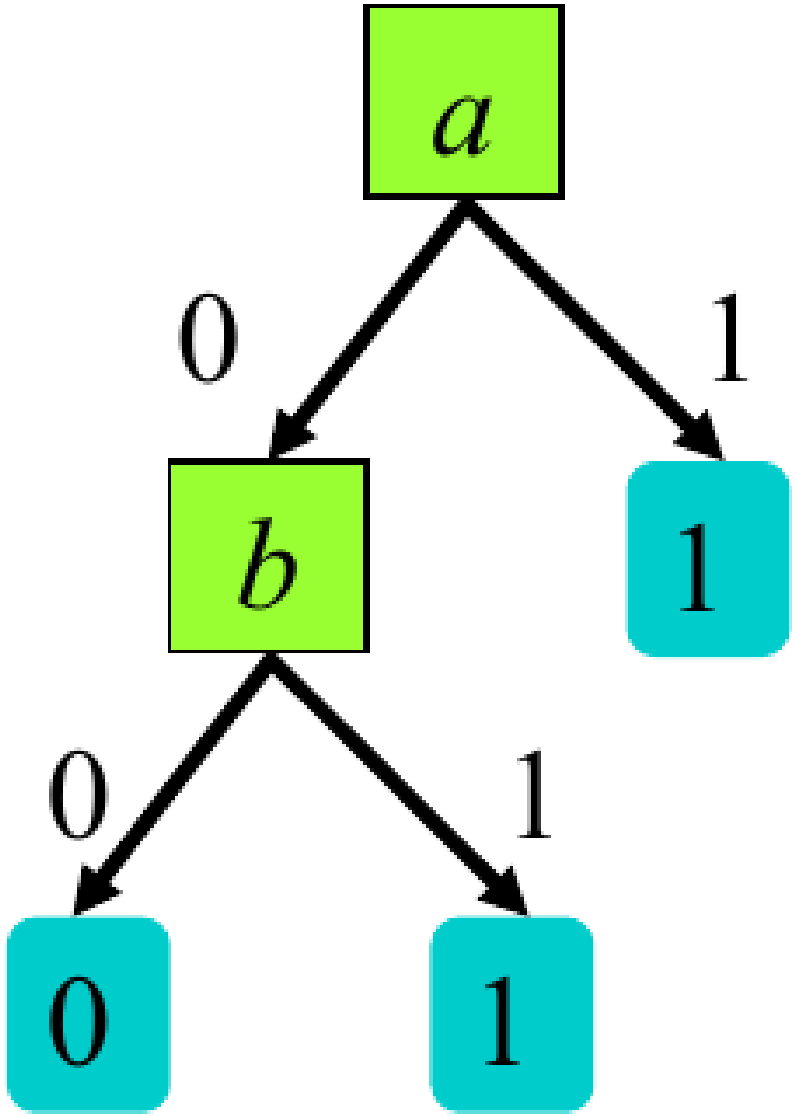
Crédito	Estado Civil	Renda	Deve
Sim	Solteiro	125K	Não
Não	Casado	100K	Não
Não	Solteiro	70K	Não
Sim	Casado	120K	Não
Não	Divorciado	95K	Sim
Não	Casado	60K	Não
Sim	Divorciado	220K	Não
Não	Solteiro	85K	Sim
Não	Casado	75K	Não
Não	Solteiro	90K	Sim



Exemplo: operações lógicas

a v b

a	b	a v b
0	0	0
0	1	1
1	0	1
1	1	1



Exercício 1

- Encontrar árvore de decisão para:
  - A AND b
  - A XOR b
  - (a AND b) OR (b AND c)

Indução de Árvores de Decisão (3)

- Geralmente usa estratégia gulosa
- Divide progressivamente objetos baseado em um atributo de teste
- Escolha do atributo normalmente tenta otimizar algum critério

Indução de Árvores de Decisão (2)

- Fase 1: **treinamento**
  - **Conj. treinamento**: permite induzir o modelo
  - **Conj. teste**: permite avaliar seu desempenho com dados não vistos durante indução
  - **Conj. validação**: semelhante a teste, mas o objetivo é podar a árvore
- Fase 2: **classificação**
  - Modelo induzido é utilizado em alguma aplicação do mundo real

Indução de Árvores de Decisão (3)

- **Algoritmo de Hunt** (**foco**)
  - Um dos primeiros
  - Base de vários algoritmos atuais
- **J48**: no Weka
  - Hunt → ID3 → C4.5 (J48)

Algoritmo de Hunt

Se todas as instâncias $\hat{x}_i$ do dataset X ↲
são da mesma classe y_j então
  t é um nó folha rotulado como y_j
Se X = ∅ então
  t é um nó folha rotulado pela ↲
  classe majoritária, y_m
Se existem x_i de diferentes classes ∈ X então
  Dividir X em subconjuntos X_1 e X_2 com ↲
  um atributo de teste
Repetir procedimento a cada subconjunto

- Obs: X = conjunto de instâncias que atingem o nó t

Algoritmo de Hunt (2)

Crédito	Estado Civil	Renda	Deve
Sim	Solteiro	125K	**Não**
Não	Casado	100K	**Não**
Não	Solteiro	70K	**Não**
Sim	Casado	120K	**Não**
Não	Divorciado	95K	**Sim**
Não	Casado	60K	**Não**
Sim	Divorciado	220K	**Não**
Não	Solteiro	85K	**Sim**
Não	Casado	75K	**Não**
Não	Solteiro	90K	**Sim**

Raiz:

classe default não (7/10)

Algoritmo de Hunt (3)

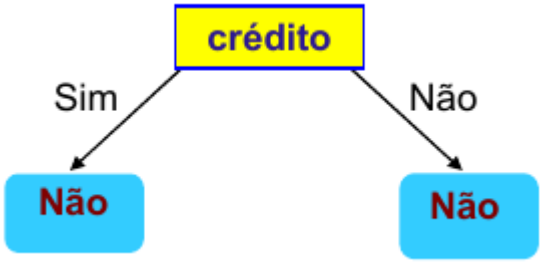
Crédito	Estado Civil	Renda	Deve
Sim	**Solteiro**	**125K**	**Não**
Não	Casado	100K	Não
Não	Solteiro	70K	Não
Sim	**Casado**	**120K**	**Não**
Não	Divorciado	95K	Sim
Não	Casado	60K	Não
Sim	**Divorciado**	**220K**	**Não**
Não	Solteiro	85K	Sim
Não	Casado	75K	Não
Não	Solteiro	90K	Sim

Nó crédito (nova raiz):
sim: classe não (3/3)

Algoritmo de Hunt (4)

Crédito	Estado Civil	Renda	Deve
Sim	Solteiro	125K	Não
Não	**Casado**	**100K**	**Não**
Não	**Solteiro**	**70K**	**Não**
Sim	Casado	120K	Não
Não	**Divorciado**	**95K**	**Sim**
Não	**Casado**	**60K**	**Não**
Sim	Divorciado	220K	Não
Não	**Solteiro**	**85K**	**Sim**
Não	**Casado**	**75K**	**Não**
Não	**Solteiro**	**90K**	**Sim**

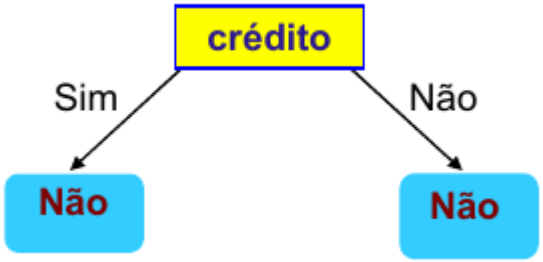
Nó crédito (nova raiz):
sim: classe não (3/3)
não: classe não (4/7)



Algoritmo de Hunt (5)

Crédito	Estado Civil	Renda	Deve
~~Sim~~	~~Solteiro~~	~~125K~~	~~Não~~
Não	Casado	100K	Não
Não	Solteiro	70K	Não
~~Sim~~	~~Casado~~	~~120K~~	~~Não~~
Não	Divorciado	95K	Sim
Não	Casado	60K	Não
~~Sim~~	~~Divorciado~~	~~220K~~	~~Não~~
Não	Solteiro	85K	Sim
Não	Casado	75K	Não
Não	Solteiro	90K	Sim

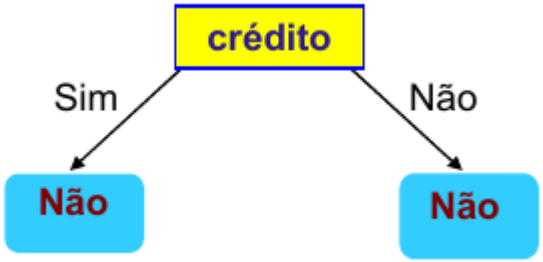
Acerto para crédito = sim já está ok.
Vamos seguir para crédito = não



Algoritmo de Hunt (6)

Crédito	Estado Civil	Renda	Deve
~~Sim~~	~~Solteiro~~	~~125K~~	~~Não~~
Não	**Casado**	**100K**	**Não**
Não	Solteiro	70K	Não
~~Sim~~	~~Casado~~	~~120K~~	~~Não~~
Não	Divorciado	95K	Sim
Não	**Casado**	**60K**	**Não**
~~Sim~~	~~Divorciado~~	~~220K~~	~~Não~~
Não	Solteiro	85K	Sim
Não	**Casado**	**75K**	**Não**
Não	Solteiro	90K	Sim

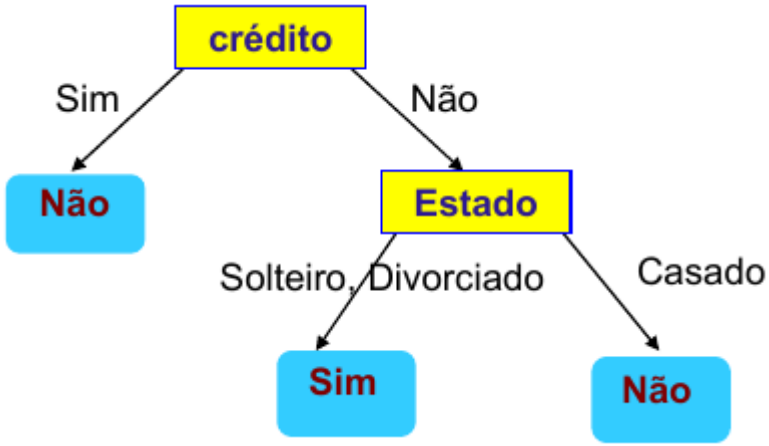
Nó estado
casado: classe não (3/3)



Algoritmo de Hunt (7)

Crédito	Estado Civil	Renda	Deve
~~Sim~~	~~Solteiro~~	~~125K~~	~~Não~~
Não	Casado	100K	Não
Não	**Solteiro**	**70K**	**Não**
~~Sim~~	~~Casado~~	~~120K~~	~~Não~~
Não	**Divorciado**	**95K**	**Sim**
Não	Casado	60K	Não
~~Sim~~	~~Divorciado~~	~~220K~~	~~Não~~
Não	**Solteiro**	**85K**	**Sim**
Não	Casado	75K	Não
Não	**Solteiro**	**90K**	**Sim**

Nó estado
casado: classe não (3/3)
solteiro/divorciado: classe sim (3/4)

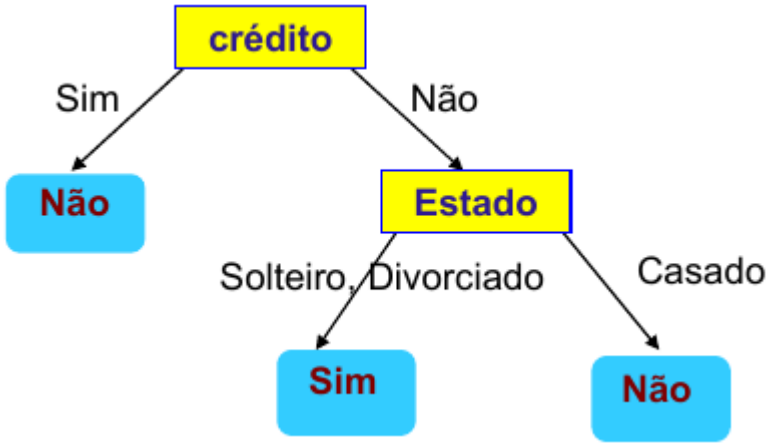


Algoritmo de Hunt (8)

Crédito	Estado Civil	Renda	Deve
~~Sim~~	~~Solteiro~~	~~125K~~	~~Não~~
~~Não~~	~~Casado~~	~~100K~~	~~Não~~
Não	Solteiro	70K	Não
~~Sim~~	~~Casado~~	~~120K~~	~~Não~~
Não	Divorciado	95K	Sim
~~Não~~	~~Casado~~	~~60K~~	~~Não~~
~~Sim~~	~~Divorciado~~	~~220K~~	~~Não~~
Não	Solteiro	85K	Sim
~~Não~~	~~Casado~~	~~75K~~	~~Não~~
Não	Solteiro	90K	Sim

Análises ok:
crédito = sim
crédito = não && estado = casado

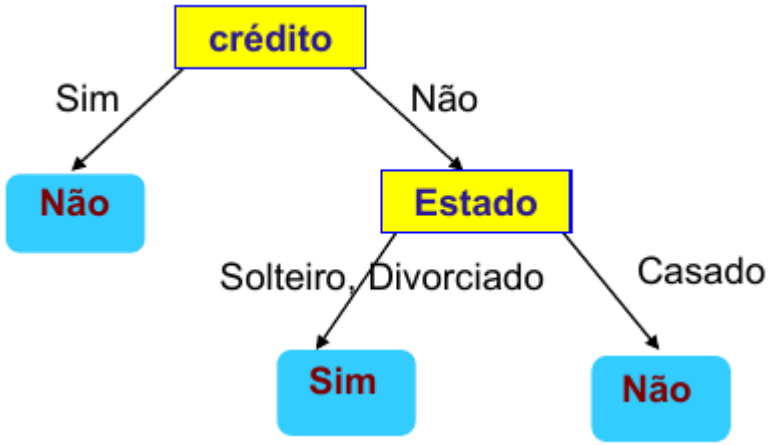
vamos seguir para
crédito = não && estado = sol/div.



Algoritmo de Hunt (9)

Crédito	Estado Civil	Renda	Deve
~~Sim~~	~~Solteiro~~	~~125K~~	~~Não~~
~~Não~~	~~Casado~~	~~100K~~	~~Não~~
Não	**Solteiro**	**70K**	**Não**
~~Sim~~	~~Casado~~	~~120K~~	~~Não~~
Não	Divorciado	95K	Sim
~~Não~~	~~Casado~~	~~60K~~	~~Não~~
~~Sim~~	~~Divorciado~~	~~220K~~	~~Não~~
Não	Solteiro	85K	Sim
~~Não~~	~~Casado~~	~~75K~~	~~Não~~
Não	Solteiro	90K	Sim

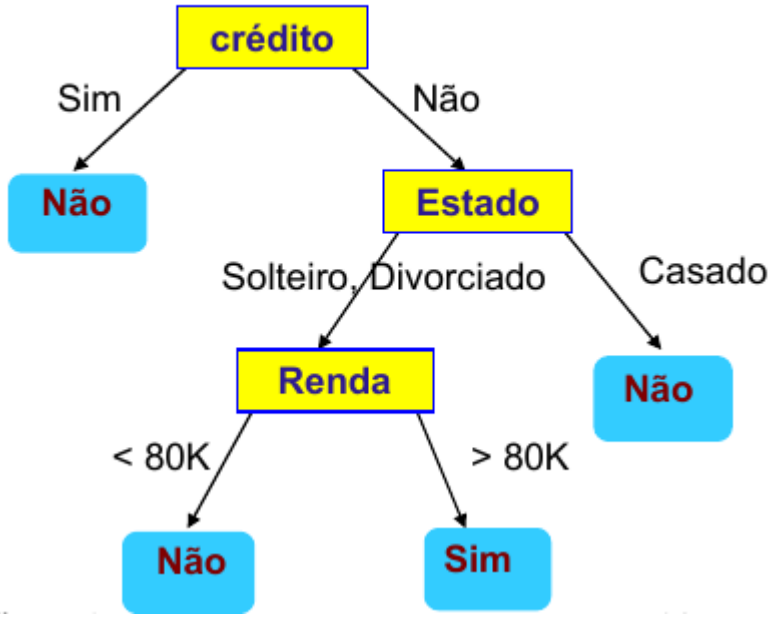
Nó renda
<80k = classe não (1/1)



Algoritmo de Hunt (10)

Crédito	Estado Civil	Renda	Deve
~~Sim~~	~~Solteiro~~	~~125K~~	~~Não~~
~~Não~~	~~Casado~~	~~100K~~	~~Não~~
Não	Solteiro	70K	Não
~~Sim~~	~~Casado~~	~~120K~~	~~Não~~
Não	**Divorciado**	**95K**	**Sim**
~~Não~~	~~Casado~~	~~60K~~	~~Não~~
~~Sim~~	~~Divorciado~~	~~220K~~	~~Não~~
Não	**Solteiro**	**85K**	**Sim**
~~Não~~	~~Casado~~	~~75K~~	~~Não~~
Não	**Solteiro**	**90K**	**Sim**

Nó renda
<80k = classe não (1/1)
>80k = classe sim (3/3)



Algoritmo de Hunt (11)

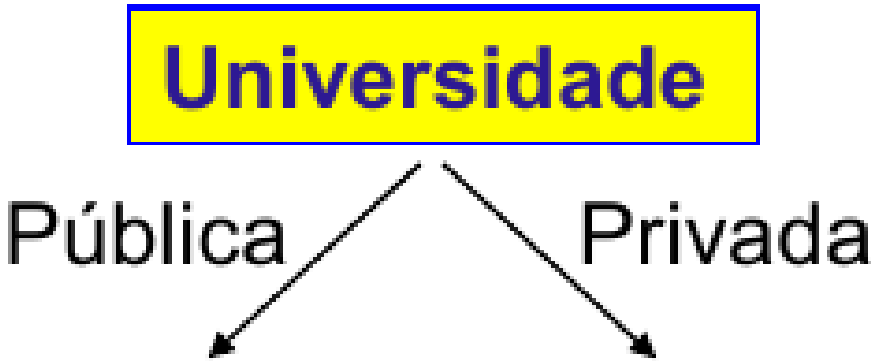
- Algoritmo de Hunt não especifica como:
  - Dividir atributos não binários
  - Escolher nó para dividir em cada nível
  - Escolher o número de divisões por nó: duas ou mais?
- Algoritmos derivados atacam essas questões

Dividindo atributos

- Divisão varia de acordo com as estratégias:
  - Binária: exatamente duas por nó (**foco**)
  - N-ária: um ramo para cada valor
  - Intermediária: alguns nós podem ficar agrupados

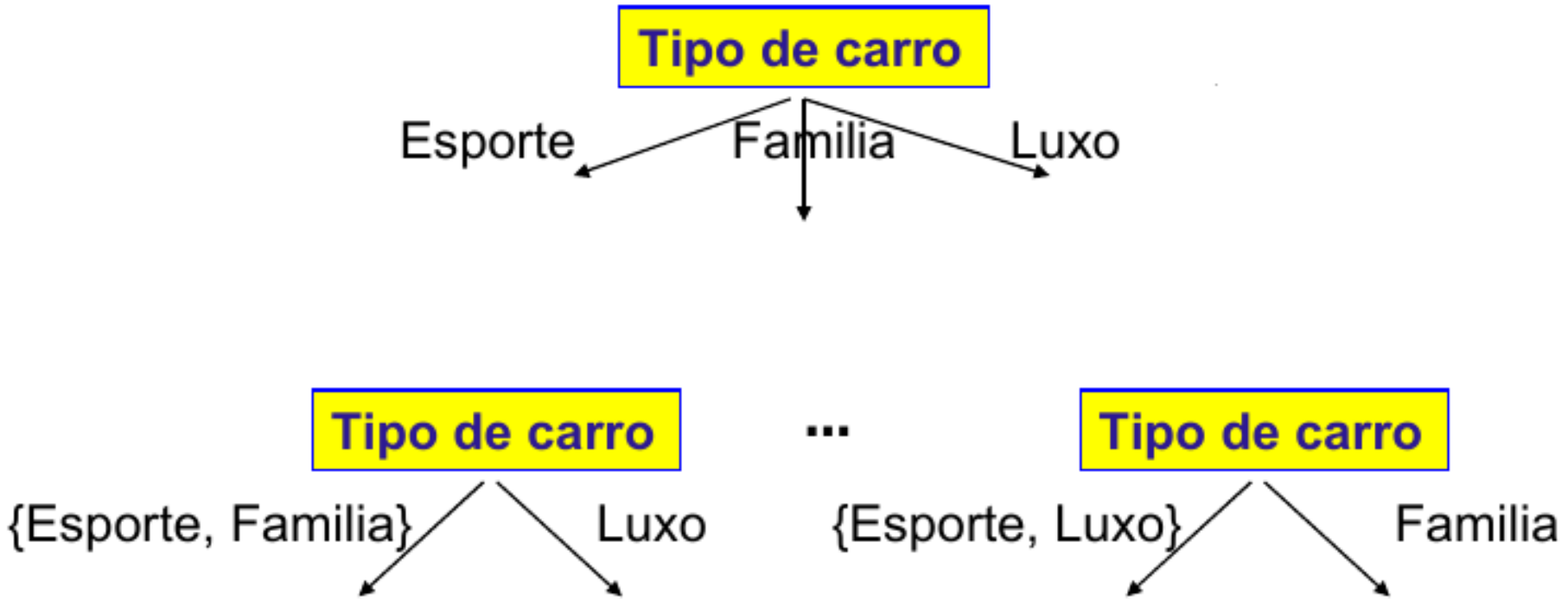
Dividindo atributos (2)

- **Atributo binário**: caso mais simples



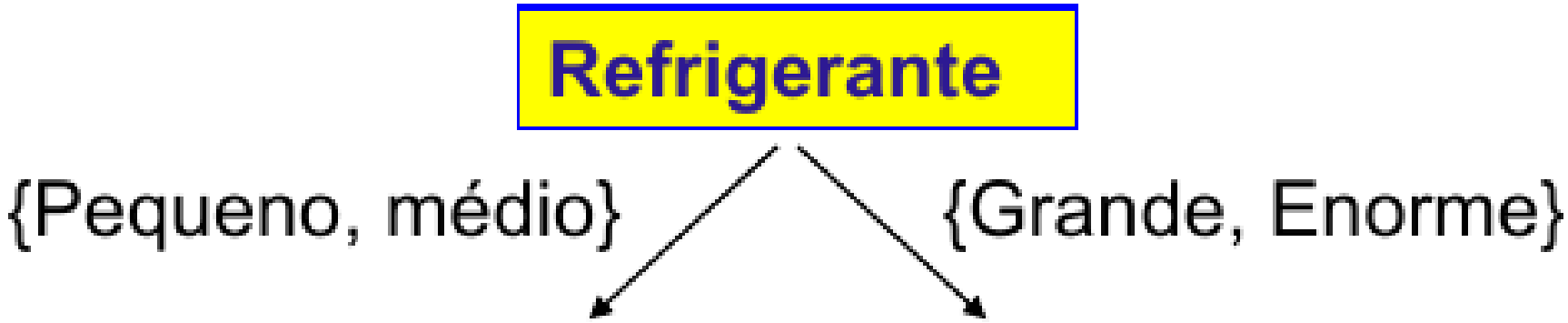
Dividindo atributos (3)

- **Atributo nominal**: depende do número de divisões desejado: binária (foco) ou múltipla



Dividindo atributos (4)

- **Atributo ordinal**: semelhante binária, mas relação de ordem deve ser respeitada

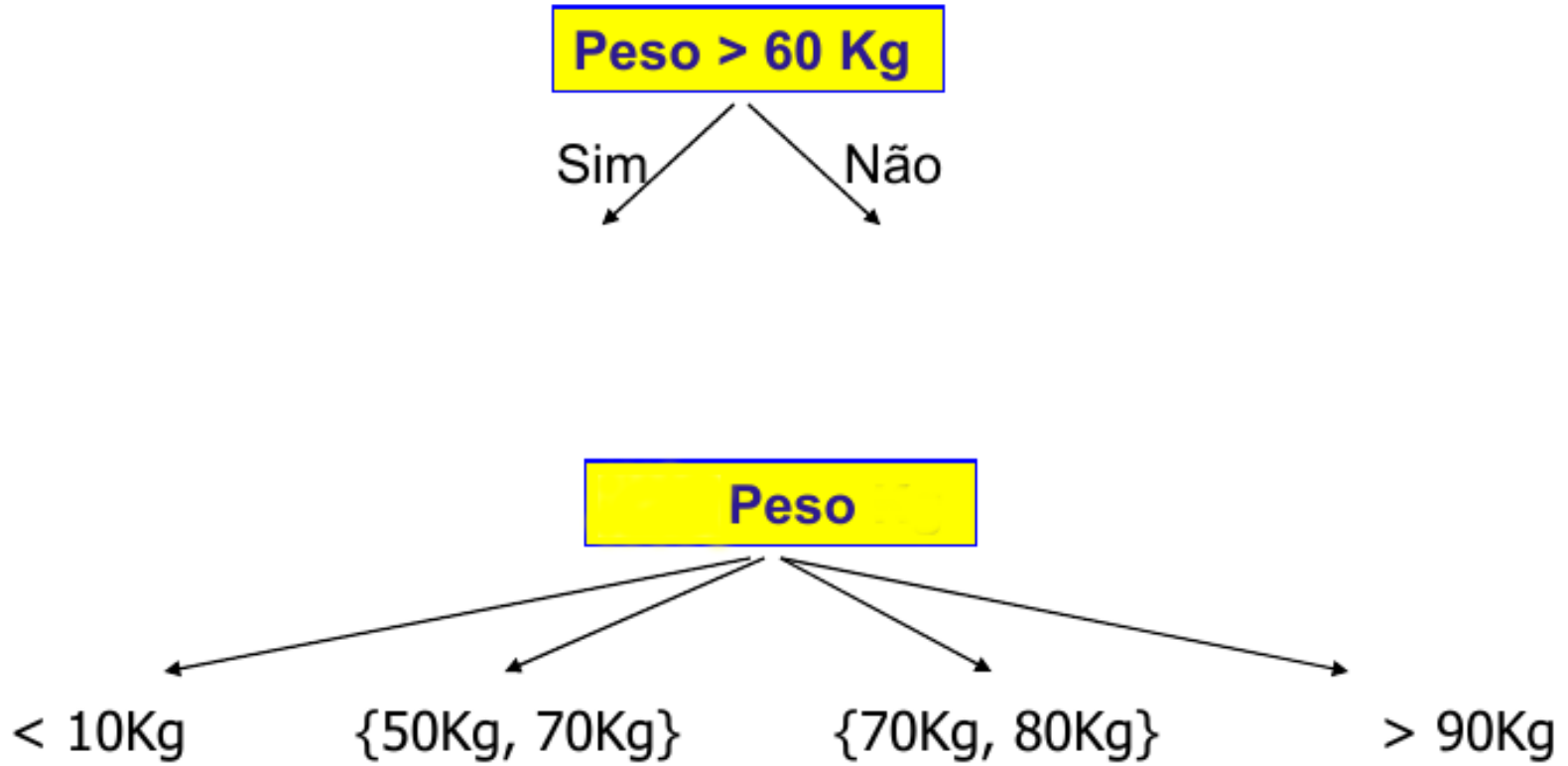


Dividindo atributos (5)

- **Atributo continuo**:
  - Usa estratégia de discretização
  - Escolher pontos de corte que gera conjuntos mais puros
  - Mais sobre pureza vs cortes adiante...

Dividindo atributos (6)

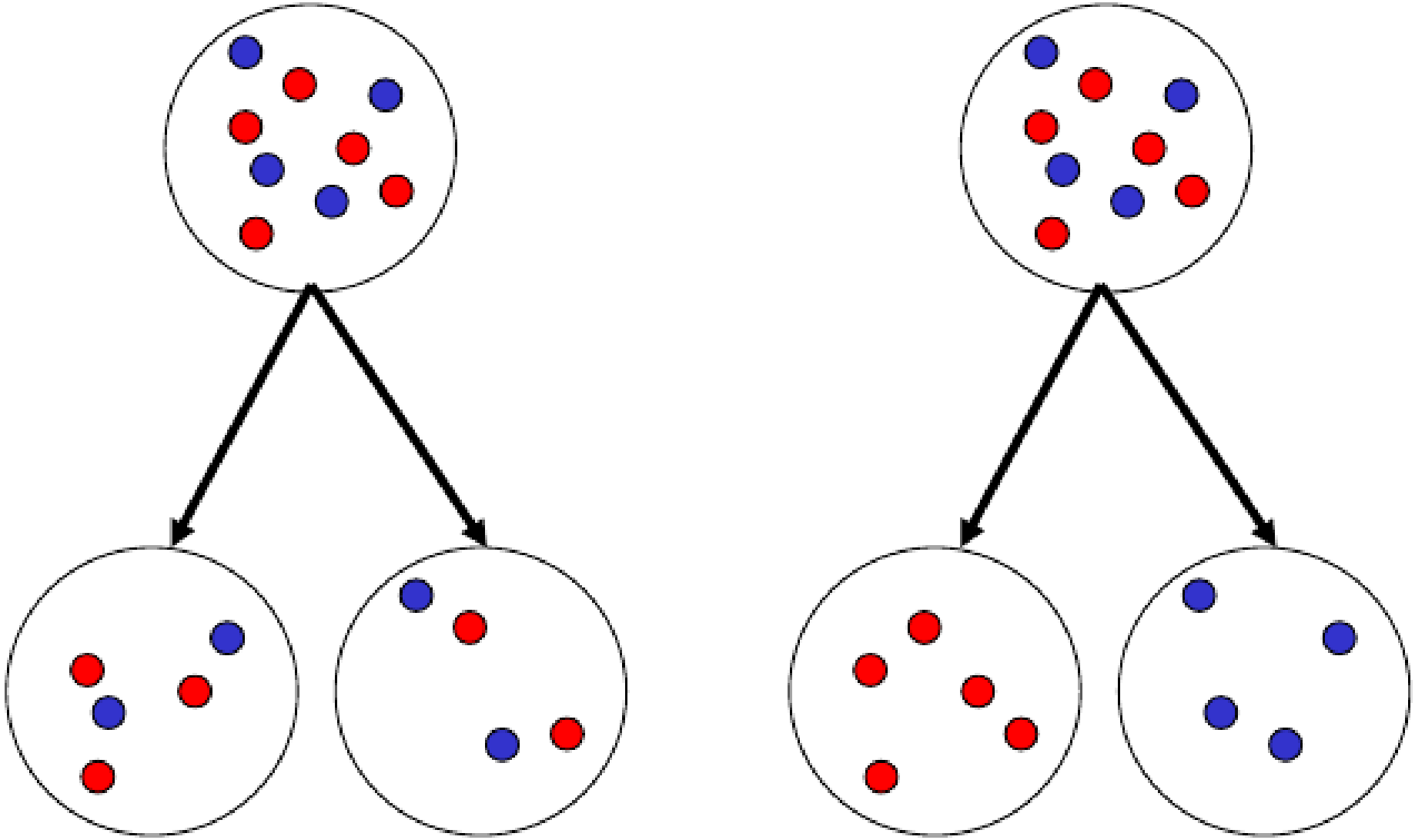
- **Atributo continuo**:



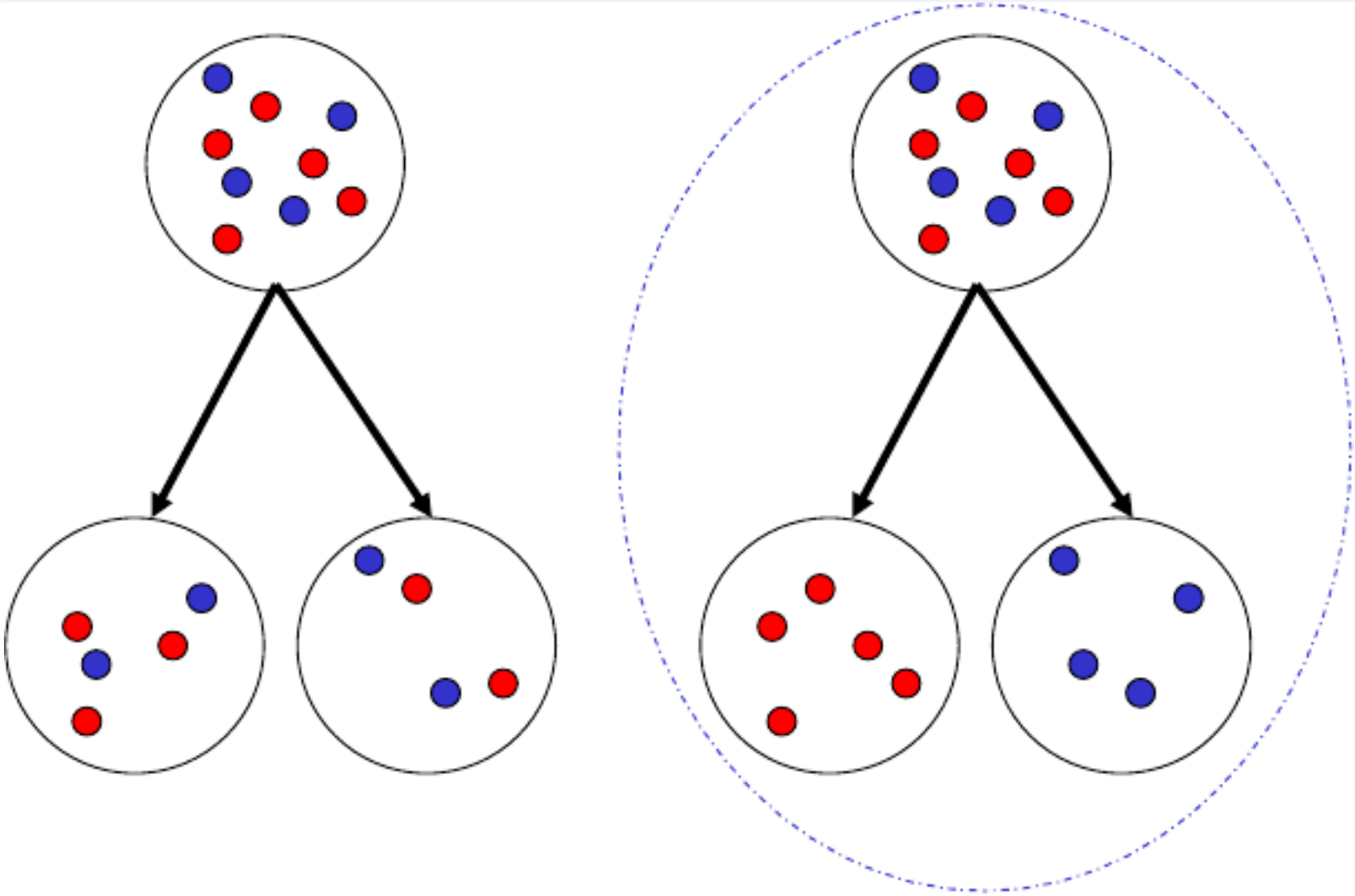
Escolhendo nós

- Idealmente: raiz deve trazer melhor divisão possível (estratégia gulosa)
- Precisamos de uma maneira de medir a qualidade:
  - (a) de um conjunto de dados
  - (b) da divisão de um conjunto de dados em conjuntos menores
- No slide a seguir, identifique qual a melhor divisão para um problema com a classe azul e a vermelha

Escolhendo nós (2)



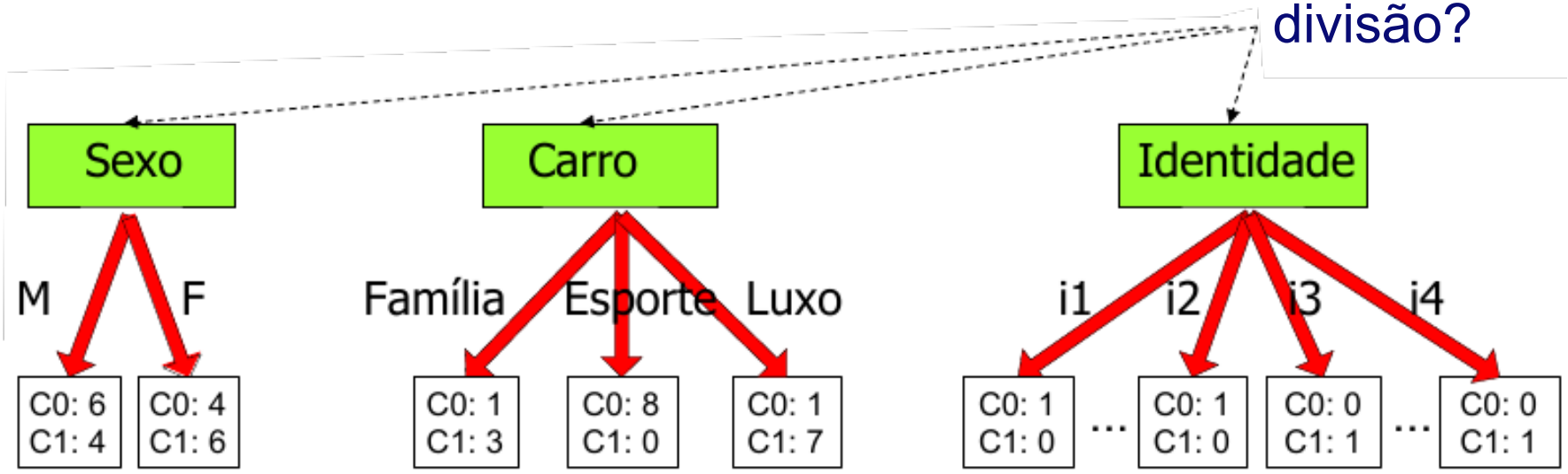
Escolhendo nós (3)



Escolhendo nós (4)

- Supor que D possui antes da divisão
  - 10 exemplos da classe 0 (C0: 10)
  - 10 exemplos da classe 1 (C1: 10)

Qual o melhor atributo para iniciar a divisão?



Medidas de impureza

- Desejamos que uma divisão gere subconjuntos mais puros o possível
- Para avaliar os subconjuntos, usamos **medidas de impureza**
  - Quanto menor seu valor, melhor
  - Desejamos minimizar a impureza

Medidas de impureza (2)

- Divisão deve preferir nós com distribuição mais homogênea (pura) de classes

C0: 5 C1: 5	C0: 9 C1: 1
Não homogênea Alto grau de impureza	Homogênea Baixo grau de impureza

- Focaremos três medidas: entropia, gini, erro de classificação

Medidas de impureza (3)

Entropia (t) = $-\sum_{y_i \in Y} p(y_i) \log_2 p(y_i)$

Gini (t) = $1 - \sum_{y_i \in Y} p(y_i)^2$

ErroClass (t) = $1 - \max_{y_i \in Y} p(y_i)$

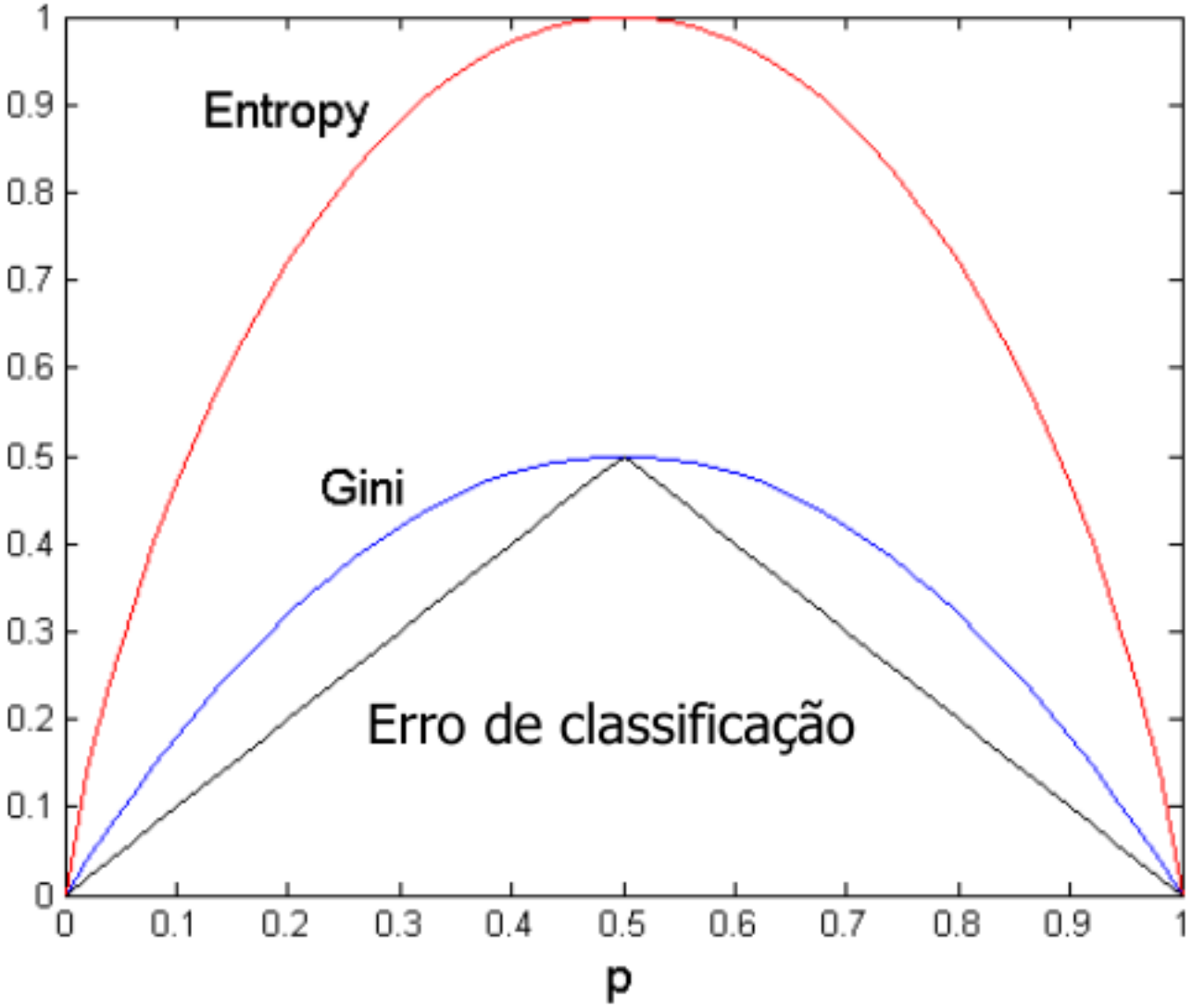
percentual $p(y_i)$ = $\frac{n(X, y_i)}{|X|}$

Medidas de impureza (4)

Onde:

- X: instâncias do subconjunto em questão
- Y: classes do subconjunto em questão
- y_i = a i-ésima classe
- $n(X, y_i)$ = número de instâncias da classe y_i no subconjunto em questão
- $0 log_2 0 = 0$

Medidas de impureza (5)



Medidas de impureza (6)

- Máximos: quando os dados estão igualmente distribuídos entre todas as classes
  - Gini: 1 – 1/|Y|
  - Erro de classificação: 1 – 1/|Y|
  - Entropia: $\log_2 |Y|$
- Mínimos: quando todos os dados pertencem a uma classe (pureza máxima)
  - Todas: 0

Medida Gini

- Calcular a medida de impureza Gini para os dados abaixo

$$Gini(t) = 1 - \sum_{y_i \in Y} p(y_i)^2$$

C1	0
C2	6
Gini=?	

C1	1
C2	5
Gini=?	

C1	2
C2	4
Gini=?	

C1	3
C2	3
Gini=?	

Exemplo Medida Gini (2)

$P(y_1) = 0/6 = 0$ $P(y_2) = 6/6 = 1$

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

$P(y_1) = 1/6$ $P(y_2) = 5/6$

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

$P(y_1) = 2/6$ $P(y_2) = 4/6$

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

$P(y_1) = 3/6$ $P(y_2) = 3/6$

$Gini = 1 – (3/6)^2 – (3/6)^2 = 0.500$

C1	**0**
C2	**6**
Gini=0.000	

C1	**1**
C2	**5**
Gini=0.278	

C1	**2**
C2	**4**
Gini=0.444	

C1	**3**
C2	**3**
Gini=0.500	

Entropia

- Medida utilizada em Teoria da Informação proposta por Shannon
  - Permite determinar a capacidade de comunicação de um canal em temos de bits
  - Diferente da entropia definida na física (termodinâmica)
    - Relacionada a quantidade de energia em um sistema físico

Entropia (2)

- Mede quanta aleatoriedade existe em um sinal ou em um evento aleatório
  - Quanta informação é carregada por um sinal
- Shannon não sabia como chamar sua medida
  - Pediu opinião a Newton
    - “Você deveria chamá-la de Entropia”
    - “Ninguém sabe o que é entropia realmente é, assim, em um debate você estará em posição de vantagem”

Exercício

- Fazer os cálculos para as medidas de entropia e de erro de classificação

$$\textit{Entropia (t)} = -\sum_{y_i \in Y} p(y_i) \log_2 p(y_i)$$

$$\textit{ErroClass(t)} = 1 - \max_{y_i \in Y} p(y_i)$$

C1	0
C2	6
E=?	

C1	1
C2	5
E=?	

C1	2
C2	4
E=?	

C1	3
C2	3
E=?	

C1	0
C2	6
Class=?	

C1	1
C2	5
Class=?	

C1	2
C2	4
Class=?	

C1	3
C2	3
Class=?	

Medida de Ganho

- As medidas ErroClass, Gini e Entropia dizem se um conjunto é puro ou não
- Uma divisão é boa quando a pureza dos filhos é maior que a pureza do pai
- A medida Ganho leva esse fator em conta
- Basicamente, escolhe-se o atributo que gere a divisão mais pura

Medida de Ganho (2)

$$\Delta = I(X_{pai}) - \sum_{i=1}^{n} \frac{|X_i|}{|X_{pai}|} I(X_i)$$

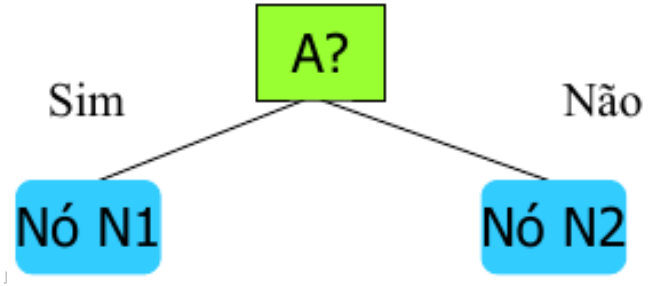
- X_{pai}: conjunto de instâncias do nó pai
- X_i : conjunto de instâncias em cada filho da divisão
- |X|: número de instâncias dentro do conjunto X
- I(X): uma medida de impureza aplicada a um dado conjunto

Medida de Ganho (3)

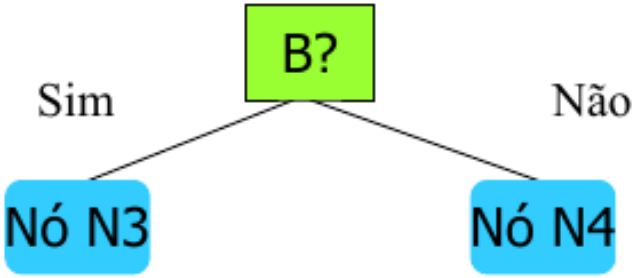
- A somatória calcula a impureza ponderada dos filhos
  - Conta a impureza de cada um, mas privilegia filhos com mais instâncias
- Δinfo: **ganho de informação**
  - Quando a medida de impureza é entropia

Exercício 2: calcular Gini's e Ganho

	Pai
C1	6
C2	6
Gini = 0.500	



	Nó 1	*Nó 2*
C1	4	2
C2	3	3
	??	??



	Nó 3	*Nó 4*
C1	1	5
C2	4	2
	??	??

Medida GiniP

- Quando um nó pai é dividido em k filhos, a qualidade da divisão é definida por:

$$GiniP = \sum_{i=1}^{n} \frac{|X_i|}{|X_{pai}|} Gini(X_i)$$

- Muito similar à medida ganho
  - Mas note que, devido a troca de sinal, está medida deve ser minimizada

Medida GiniP (2)

- GiniP é a abreviação de Gini da Média Ponderada
  - Usa a medida Gini
- Assim como o Ganho, também avalia a qualidade da divisão
  - Ganho deve ser maximizada; GiniP minimizada
  - Não leva nó pai em comparação: permitindo comparar divisões em pontos diferentes da hierarquia
- Usada pelos algoritmos CART, SLIQ, SPRINT

Exercício 3

- De acordo com GiniP, qual é a melhor divisão possível?

	Tipo de Carro		
	Família	Esporte	Luxo
C1	1	2	1
C2	4	1	1
GiniP	???		

	Tipo de Carro	
	{Esporte, Luxo}	{Familia}
C1	3	1
C2	2	4
GiniP	???	

	Tipo de Carro	
	{Esporte}	{Family, Luxury}
C1	2	2
C2	1	5
GiniP	???	

Exercício 3 (2)

	Tipo de Carro		
	Família	Esporte	Luxo
C1	1	2	1
C2	4	1	1
GiniP	0.393		

	Tipo de Carro	
	{Esporte, Luxo}	{Familia}
C1	3	1
C2	2	4
GiniP	0.400	

	Tipo de Carro	
	{Esporte}	{Família, Luxo}
C1	2	2
C2	1	5
GiniP	0.419	

Taxa de Ganho

- Ganho e GiniP acabam privilegiando divisões com o maior número de subgrupos
  - Como pode ser visto no slide anterior
- Conjuntos menores tendem a ser mais puros
  - Pense na pureza de um conjunto unitário
- Taxa de Ganho é uma melhoria na medida Ganho que penaliza divisões com muitos subgrupos
- A seguir, exemplo para entropia

Taxa de Ganho (2)

$$TG = \frac{\Delta_{info}}{IV}$$

$$IV = -\sum_{v_i \in x} p(v_i) \log_2(v_i)$$

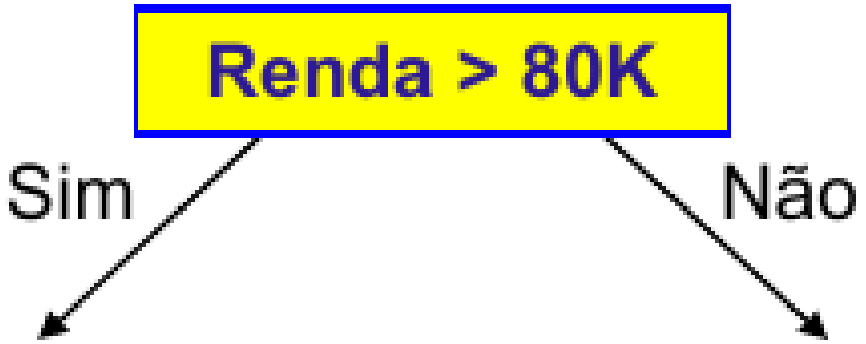
- Onde:
  - TG: taxa de ganho da divisão
  - Δ_{info}: ganho de informação da divisão
  - Continua no próximo slide...

Taxa de Ganho (3)

- … continuação:
  - IV: valor intrínseco da divisão
  - v_i: valores do atributo *x* utilizados na divisão
  - $p(v_i)$ = porcentagem das instâncias no nó pai que assumem o valor (v_i) para o atributo x
- Note que taxa de ganho lembra a entropia, mas a primeira é sobre valores de atributos e a segunda sobre as possíveis classes

Atributos contínuos

Crédito	Estado Civil	Renda	Deve
Sim	Solteiro	**125K**	Não
Não	Casado	**100K**	Não
Não	Solteiro	**70K**	Não
Sim	Casado	**120K**	Não
Não	Divorciado	**95K**	Sim
Não	Casado	**60K**	Não
Sim	Divorciado	**220K**	Não
Não	Solteiro	**85K**	Sim
Não	Casado	**75K**	Não
Não	Solteiro	**90K**	Sim



Atributos contínuos (2)

- Seleção de atributos para divisão
  - Binários: usar ganho ou giniP
  - Nominais/ordinais:
    - Divisão binária: ganho ou ginip
    - Divisão múltipla: taxa de ganho
  - **Contínuos**: mesma estratégia do caso anterior
    - Testando vários pontos de corte

Atributos contínuos (3)

- Estratégias de escolha de pontos de corte
  - Força bruta: método mais simples consistindo em testar vários pontos de corte e calcular qualidade da divisão obtida
  - Ordenar valores do dataset em ordem crescente: avaliar qualidade do corte entre cada par de valores
  - Ordenação otimizada: idem anterior; testar corte só testar pares com classes diferentes

Atributos contínuos (4)

Deve	Não	Não	Não	Sim	Sim	Sim	Não	Não	Não	Não
Renda										
Valores ordenados →	60	70	75	85	90	95	100	120	125	220
Posições de divisão →	55	65	72	80	87	92	97	110	122	172	230

	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Sim	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
Não	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
GiniP	0.420		0.400		0.375		0.343		0.417		0.400		*0.300*		0.343		0.375		0.400		0.420	

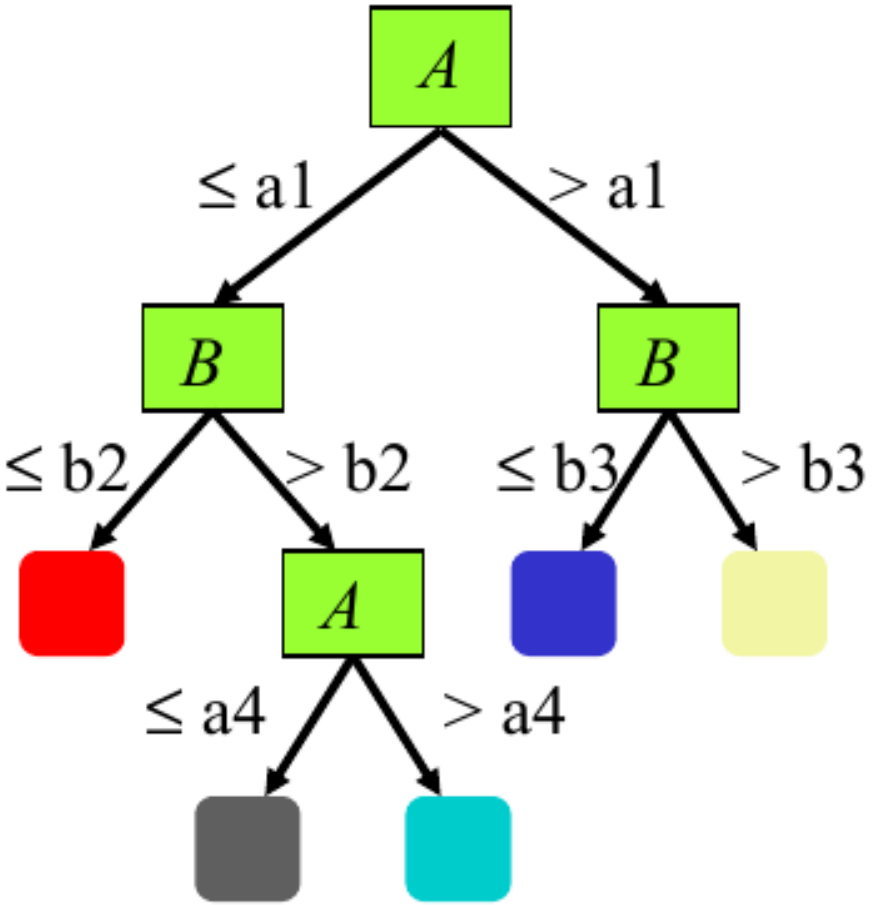
Treinamento

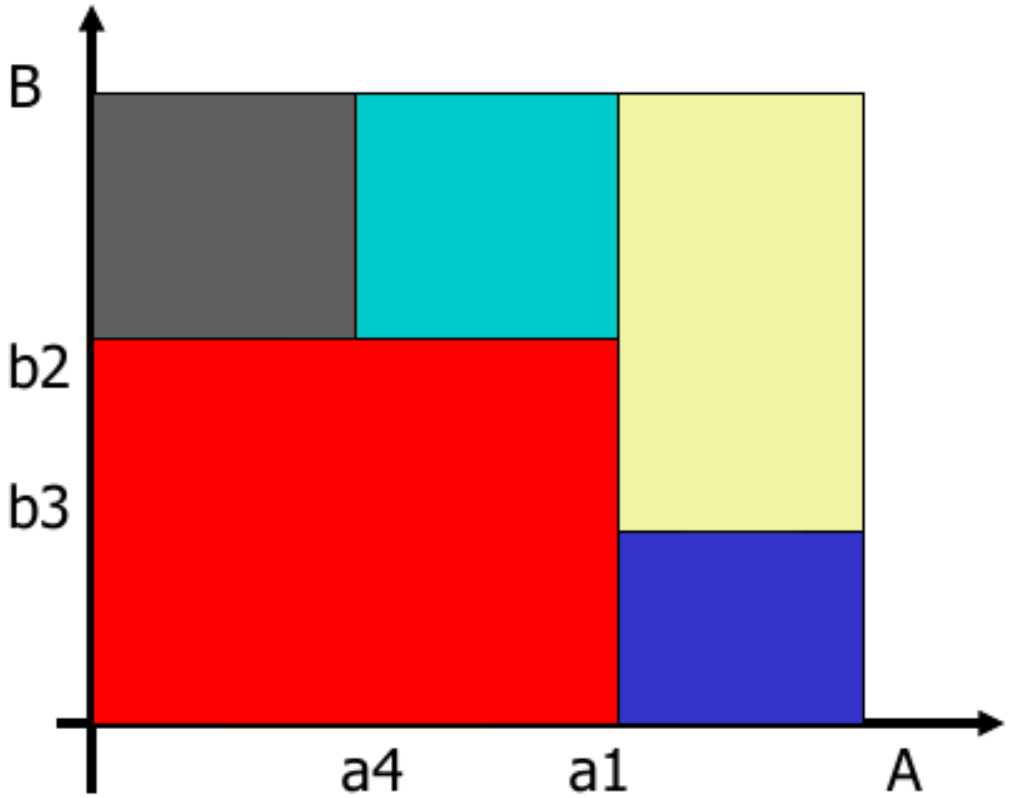
- Indução da árvore pode ser finalizada quando:
  - Quando os dados do nó atual têm o mesmo rótulo
  - Quando as instâncias no nó atual são duplicadas umas das outras (variando apenas a classe)
  - Quando o número de dados é menor que um dado valor
  - Todos os atributos já foram incluídos no caminho

Espaço de instâncias

- Cada percurso da raiz a um nó folha representa uma regra de classificação
- Cada folha esta associada a uma classe, corresponde a uma região do espaço de instâncias
- Hiperretângulos:
  - Interseção entre eles é um conjunto vazio
  - União é o espaço total

Espaço de instâncias (2)





Espaço de hipóteses (3)

- Tem como saída uma única hipótese
  - Não há backtracking
  - Mínimo local
- Espaço de hipóteses completo (sujeito a overfitting)
  - Hipercubos podem aproximar hiperplanos não parelelos aos eixos

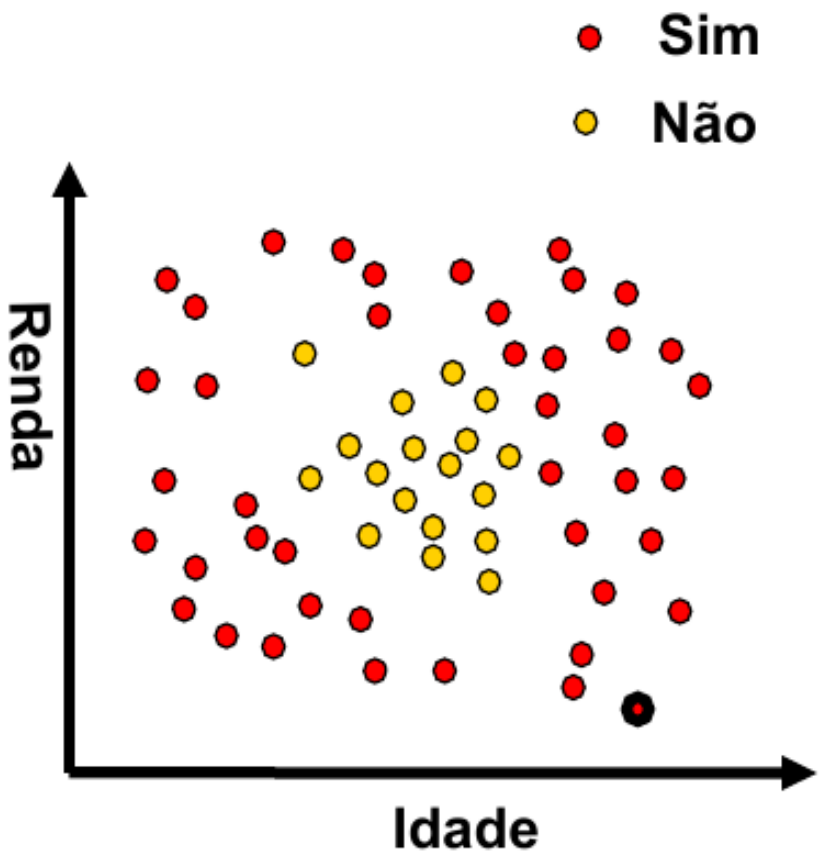
Exemplo 2

- Sejam os dados abaixo referentes a solicitações de crédito bancário
  - Construir uma árvore de decisão que classifica quem solicita crédito

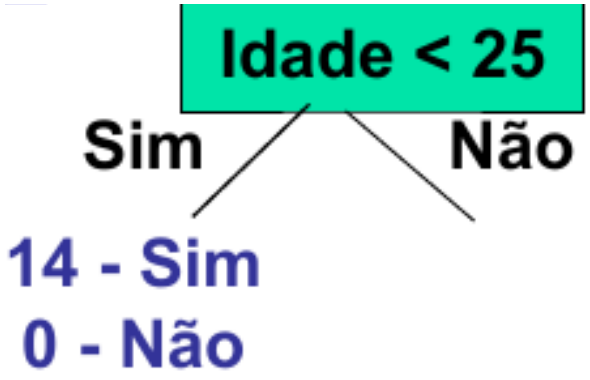
Idade	*Renda*	*Classe*
20	2000	Sim
30	5100	Não
60	5000	Sim
40	6000	Não
...	...	...

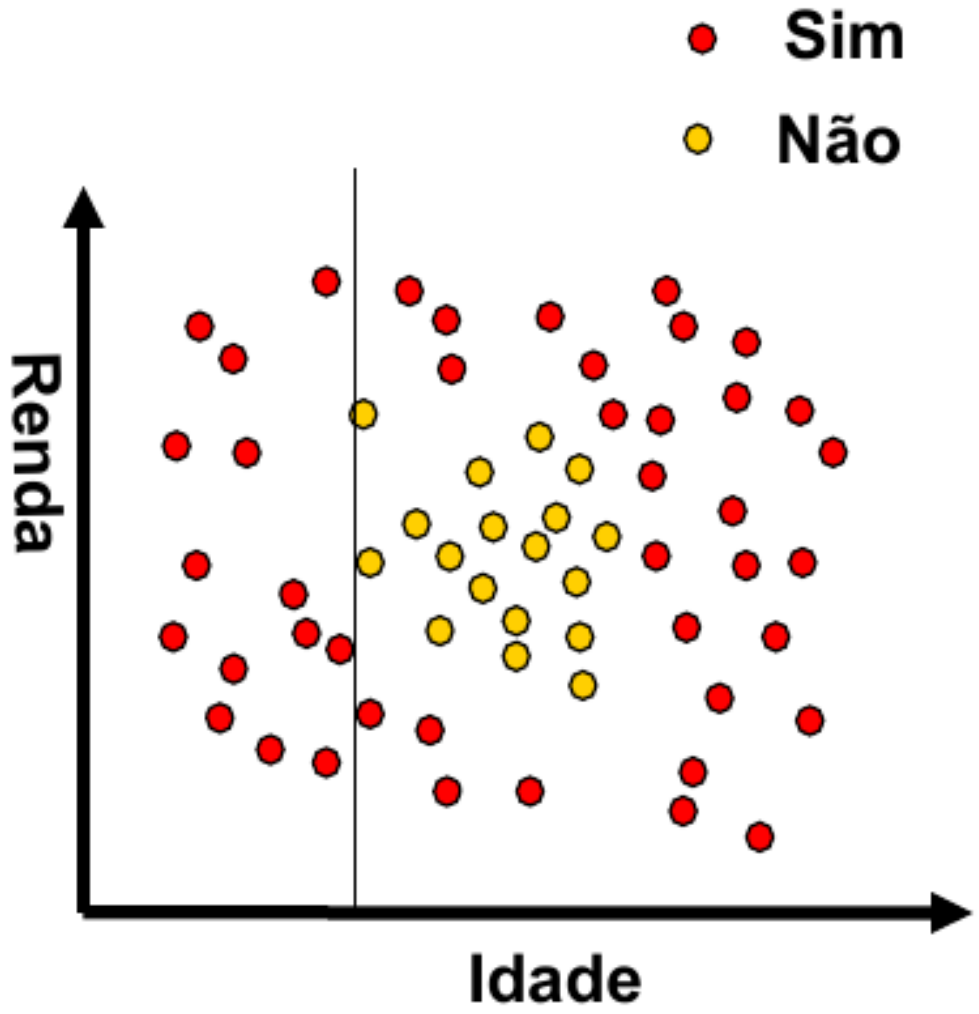
Indução

- Construir uma árvore de decisão que classifica quem solicitar crédito
  - Deve (Sim)
  - Não deve (Não)

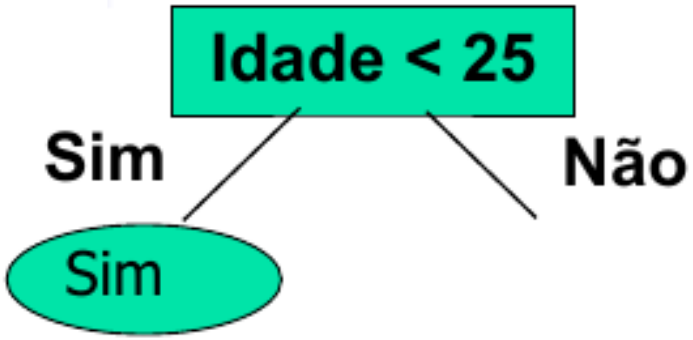


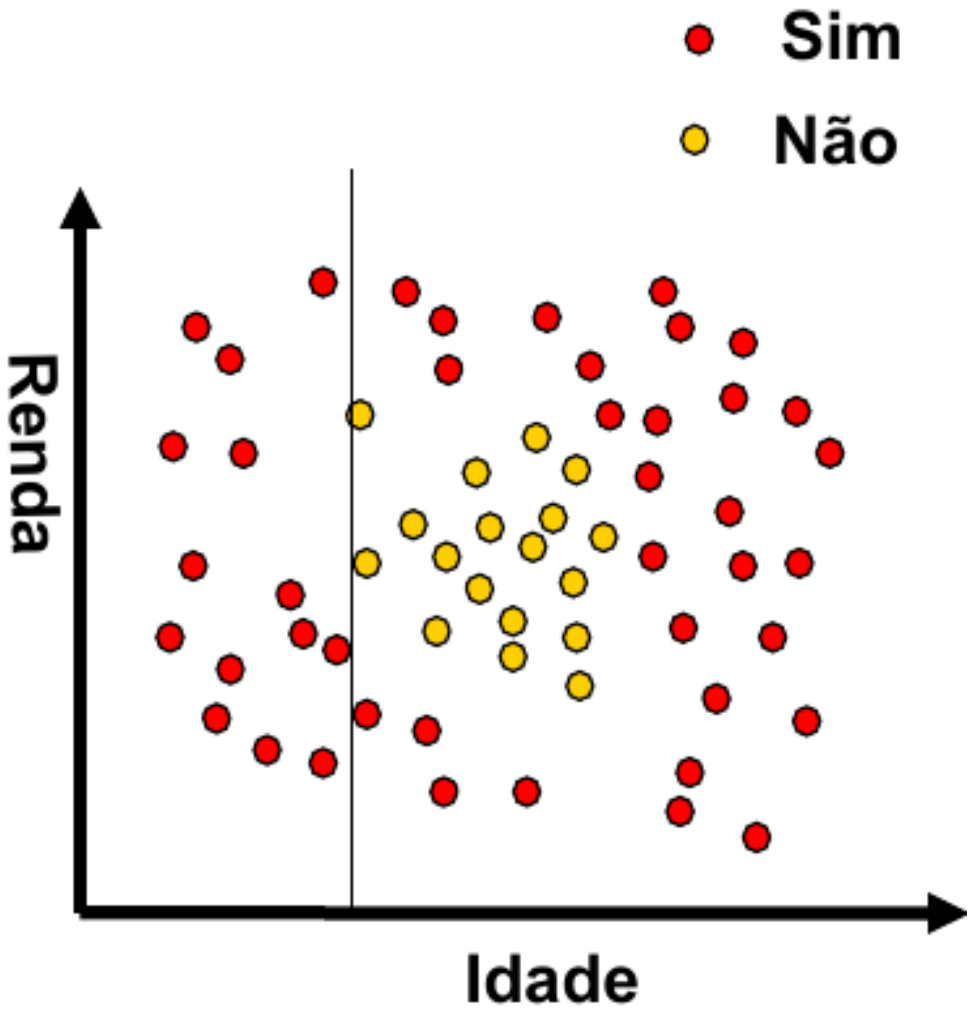
Indução (2)



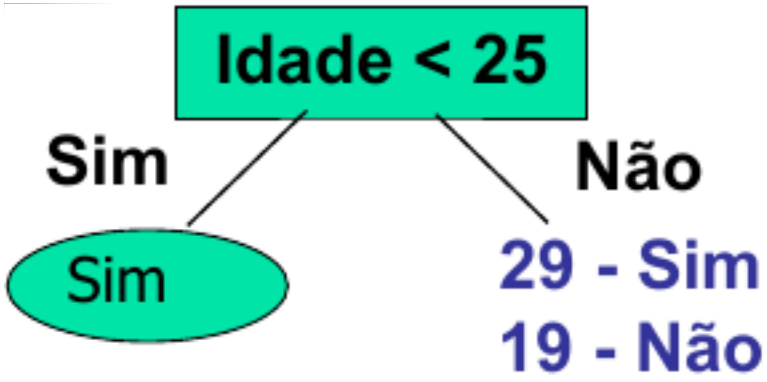


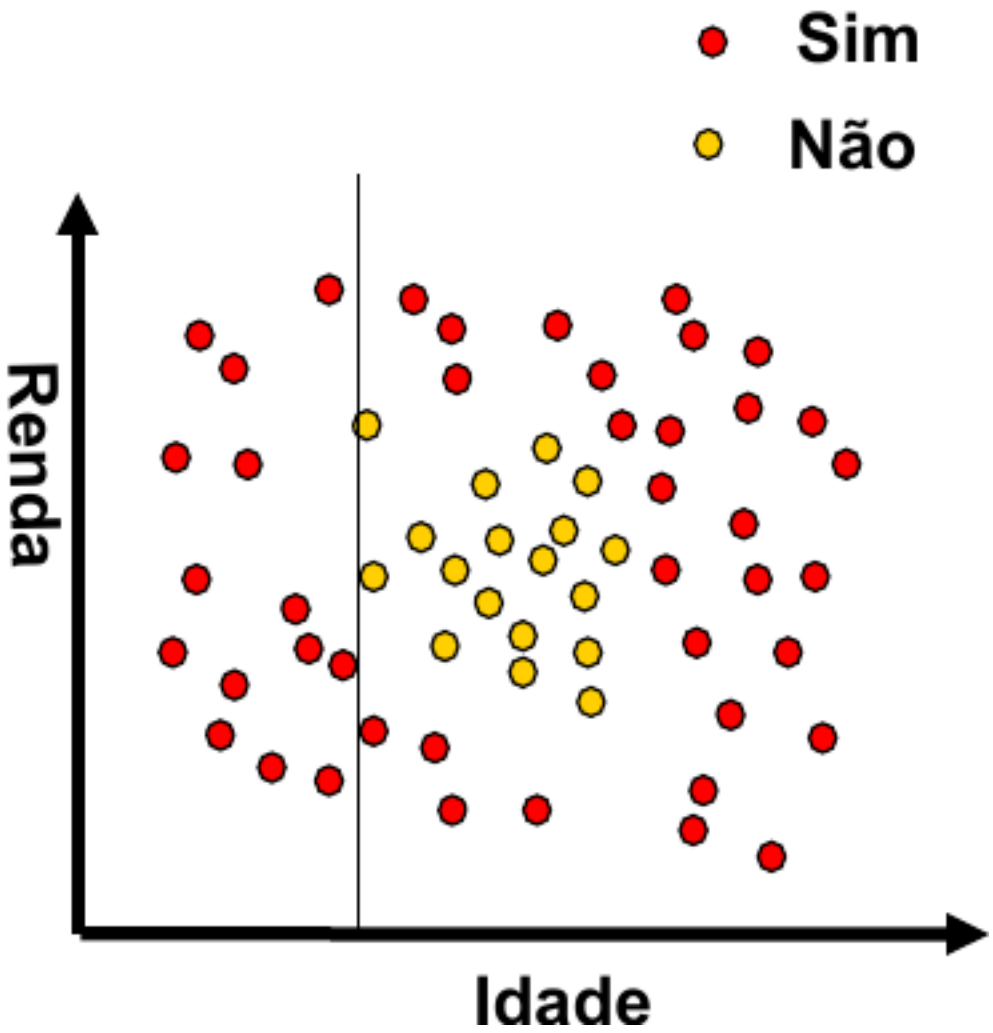
Indução (3)



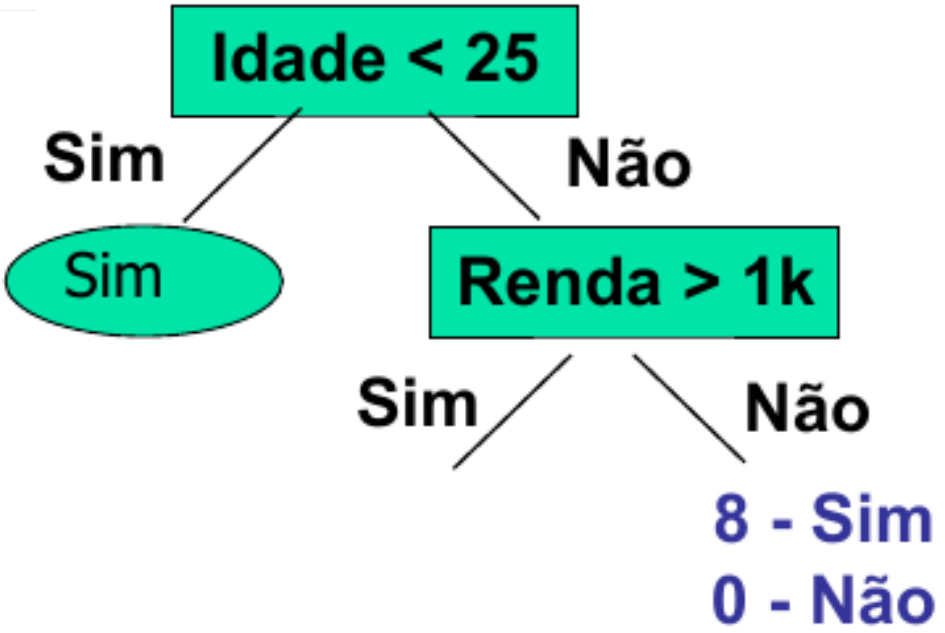


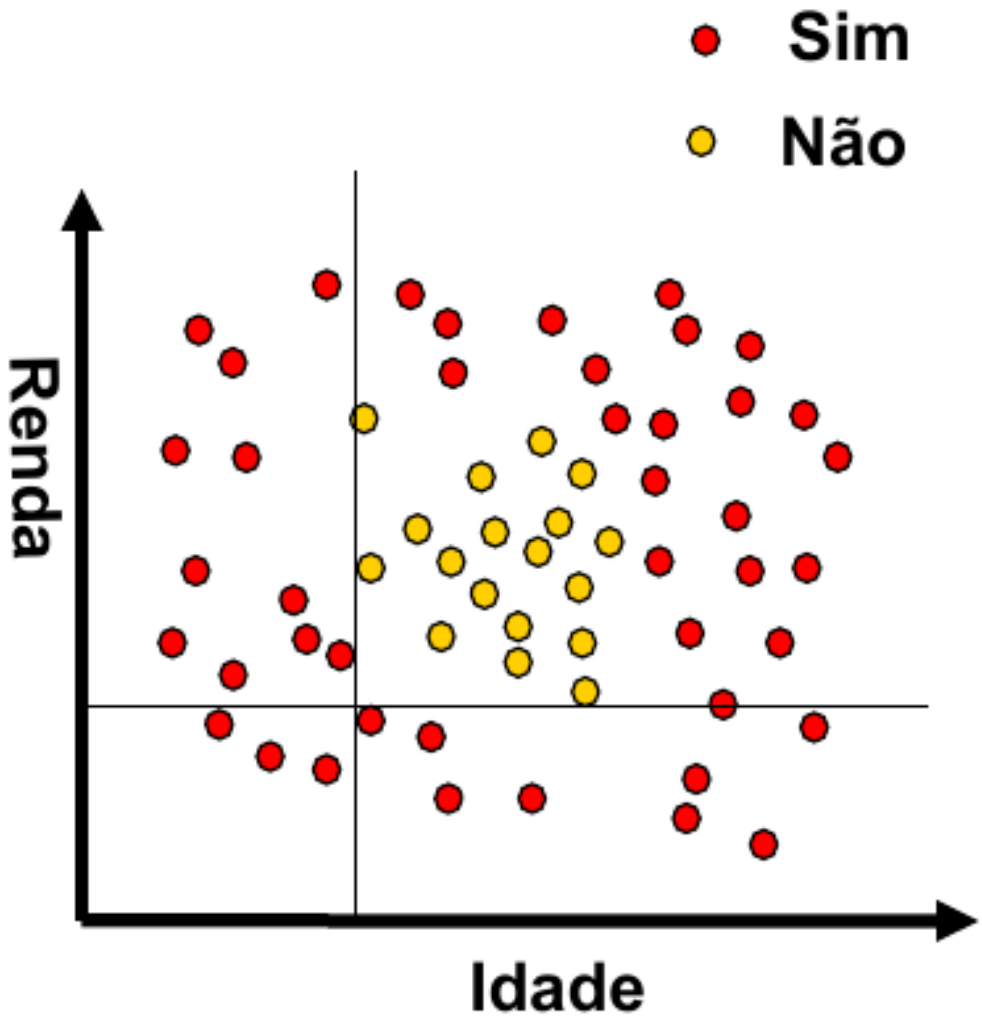
Indução (4)



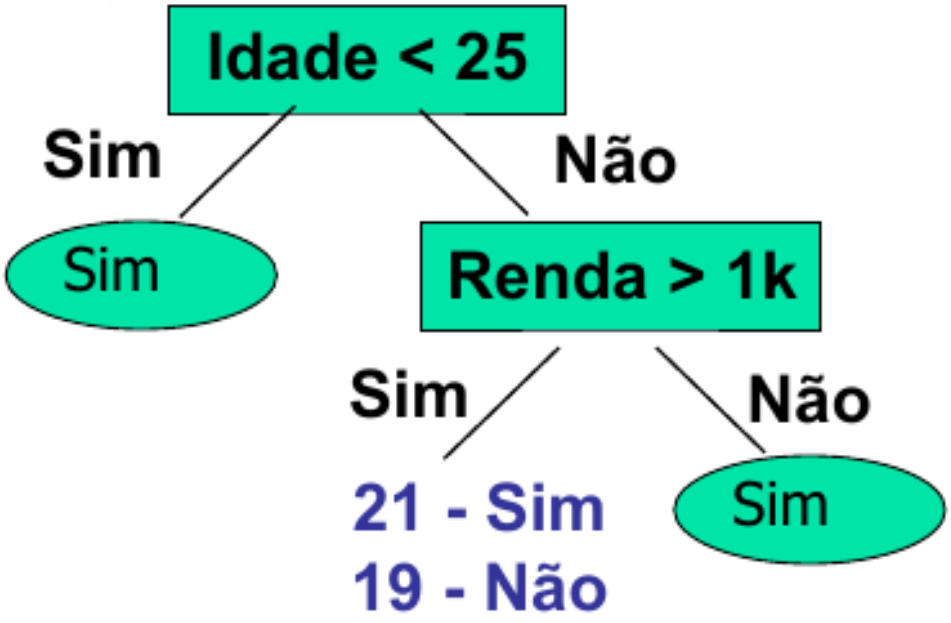


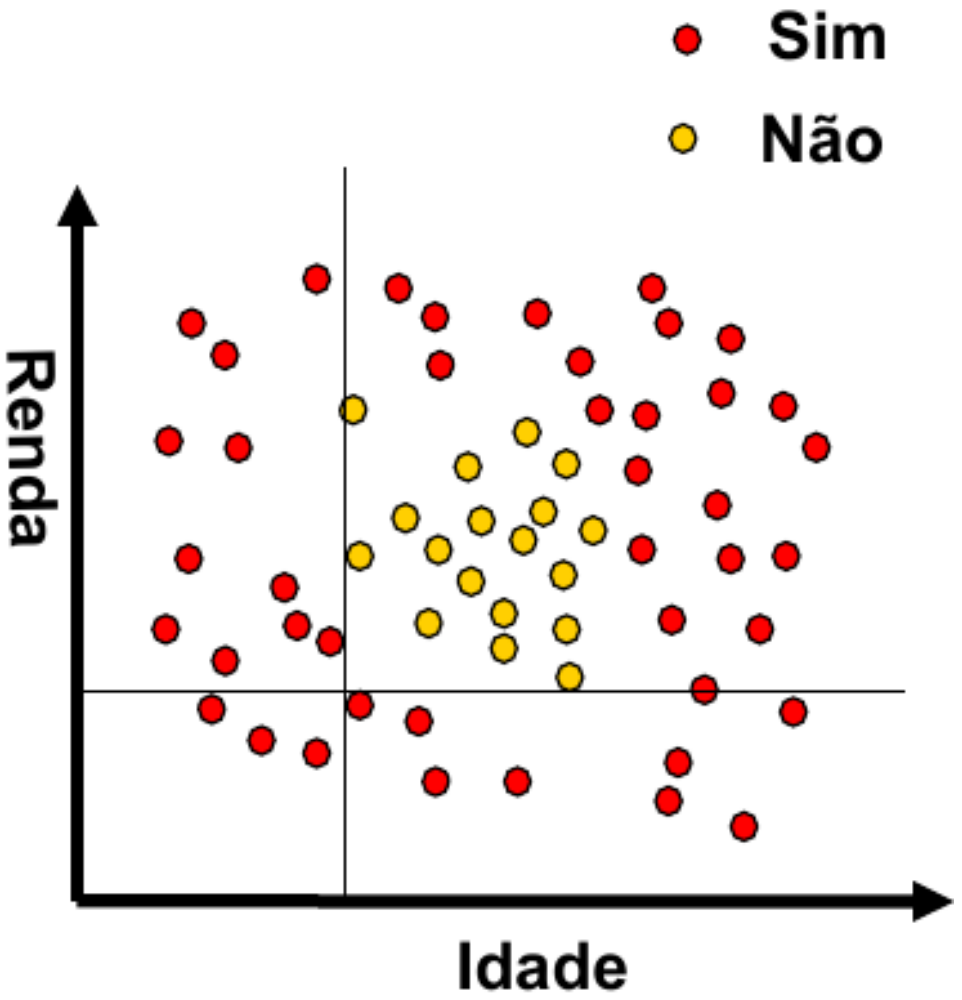
Indução (5)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
8 - Sim
0 - Não

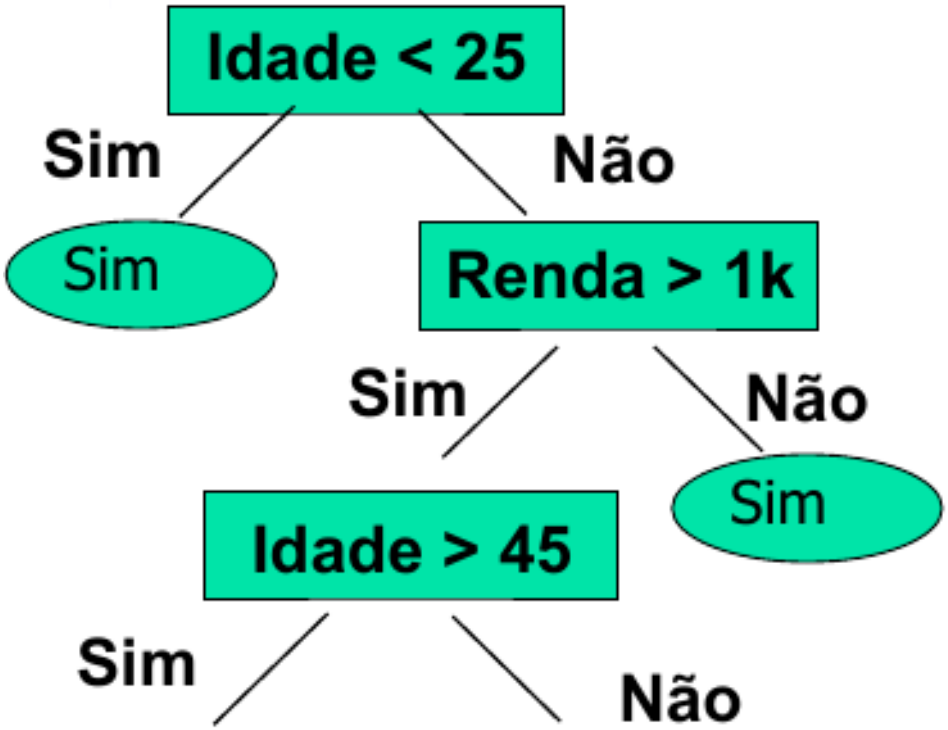

Sim
Não
Renda
Idade

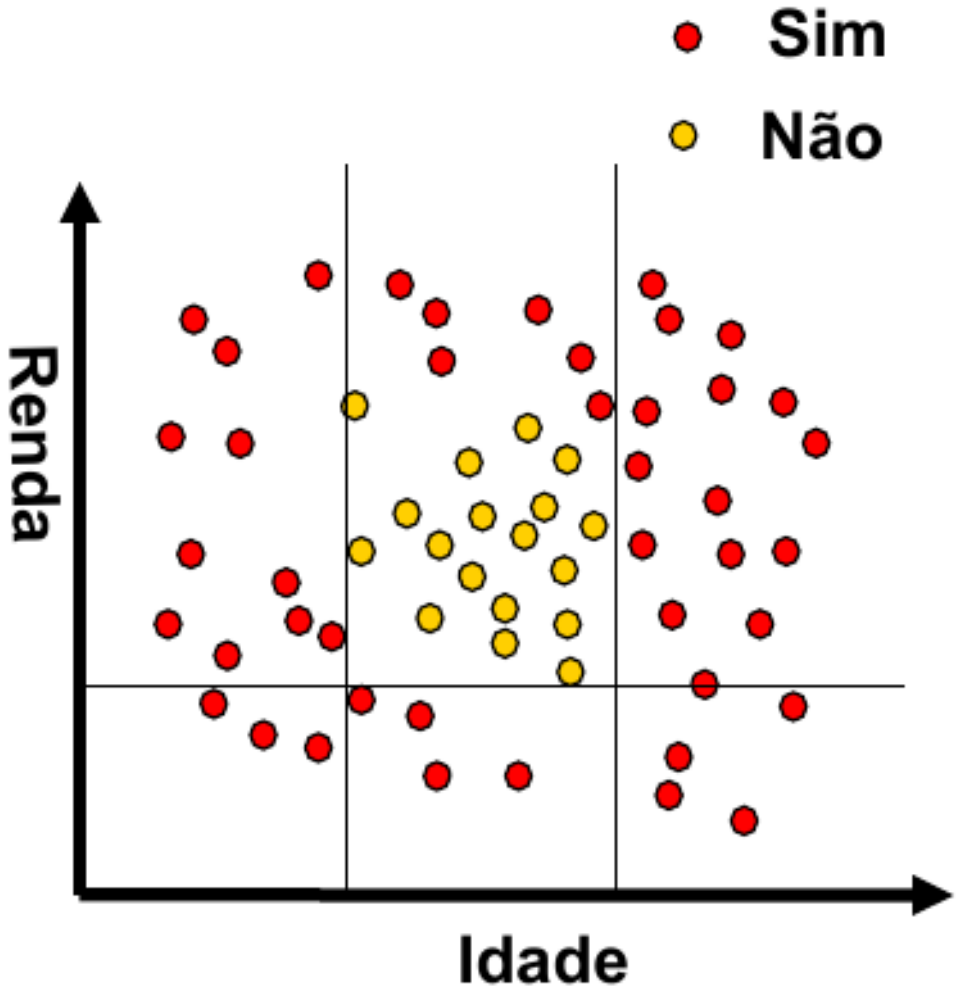
Indução (6)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
21 - Sim
19 - Não
Sim

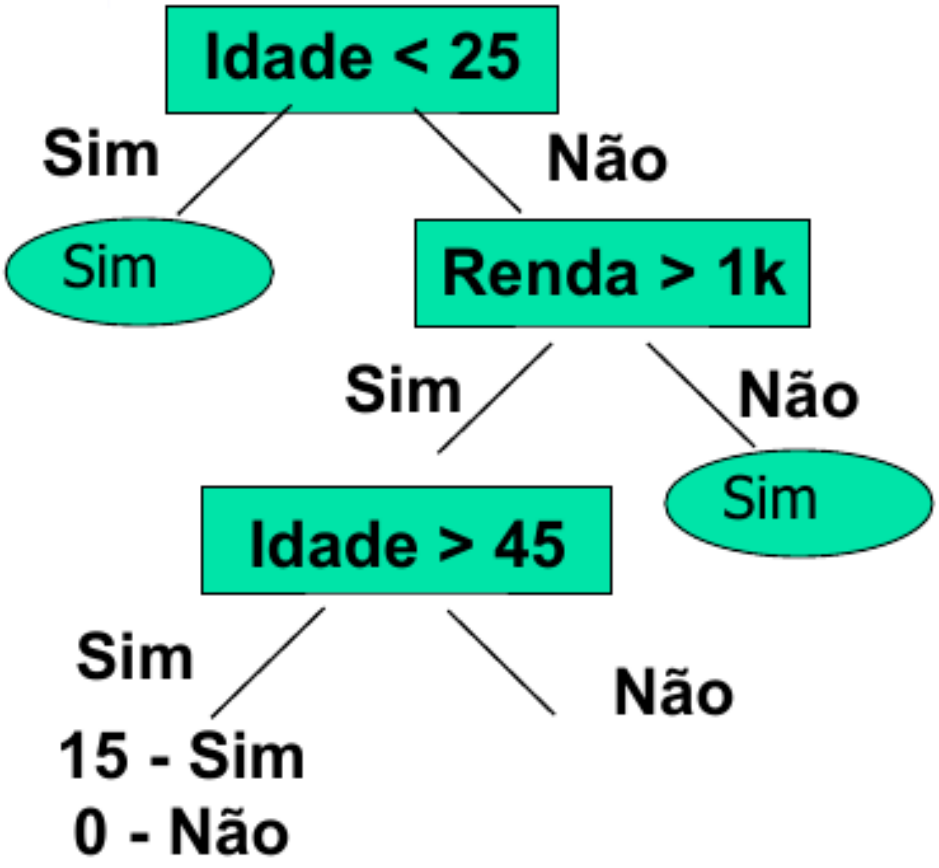

Sim
Não
Renda
Idade

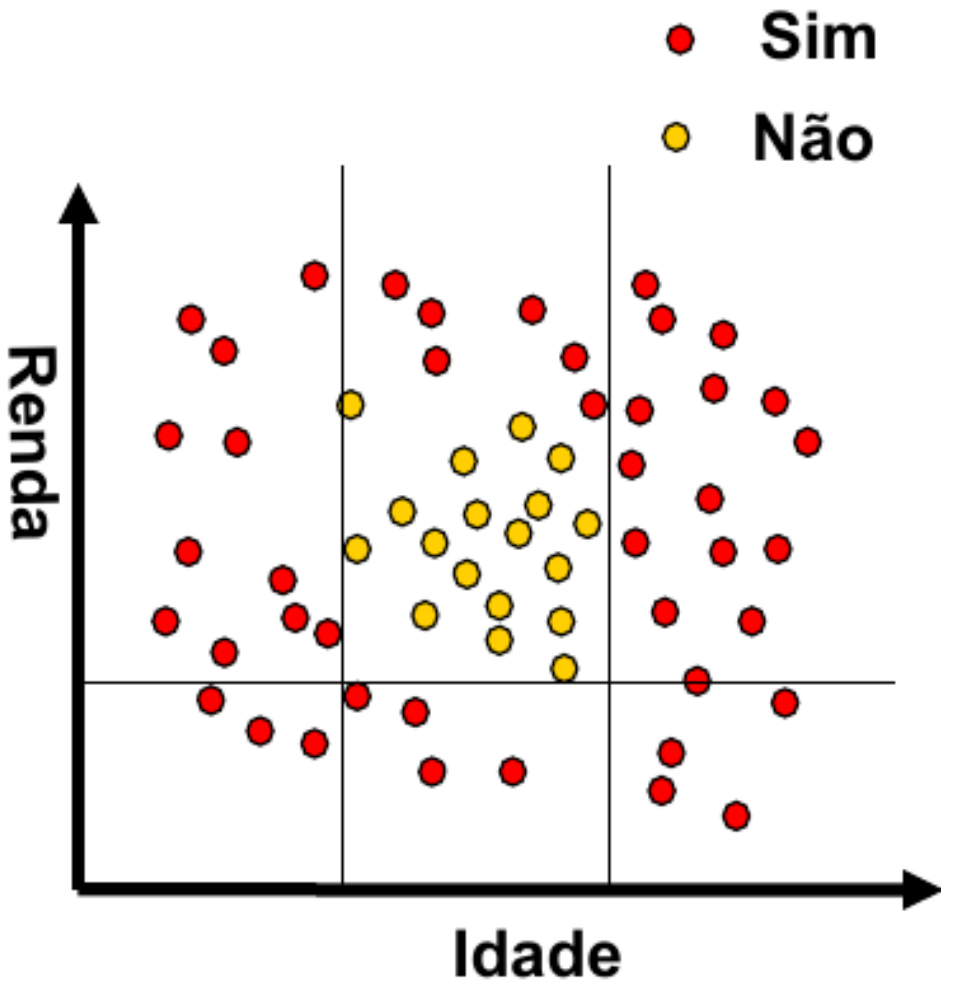
Indução (7)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
Sim
Idade > 45
Sim
Não

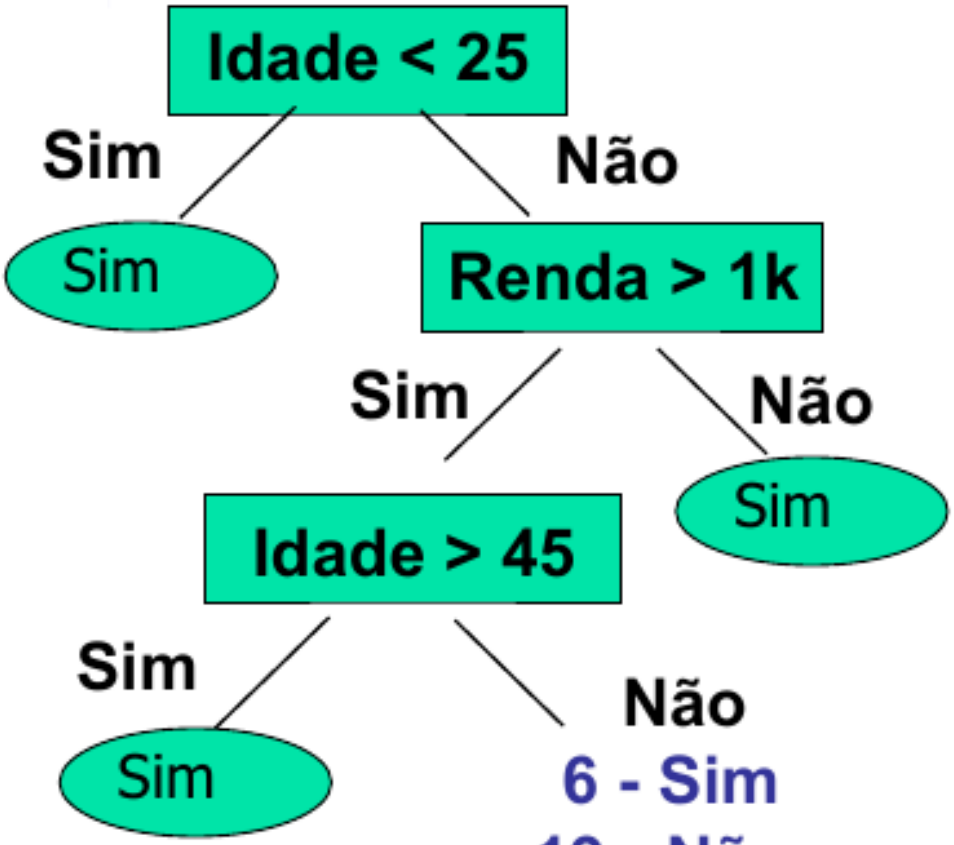

Sim
Não
Renda
Idade

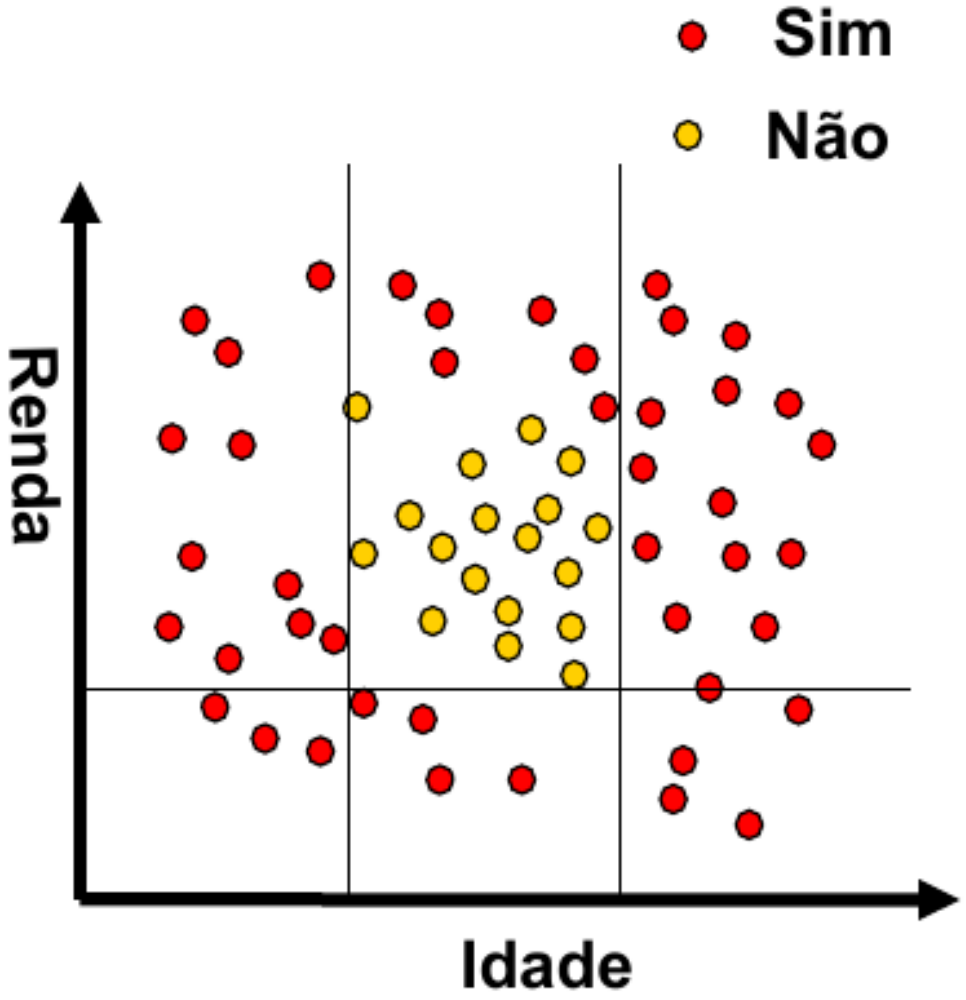
Indução (8)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
Sim
Idade > 45
Sim
Não
15 - Sim
0 - Não

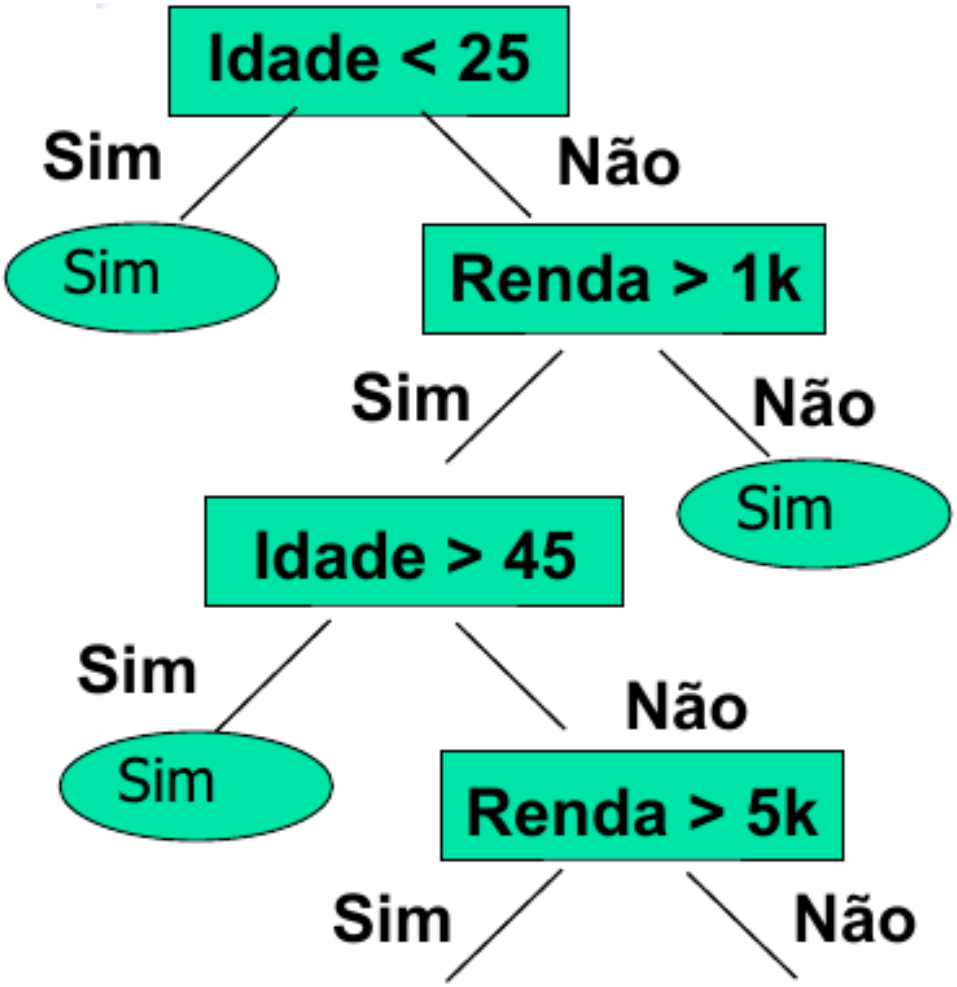

Sim
Não
Renda
Idade

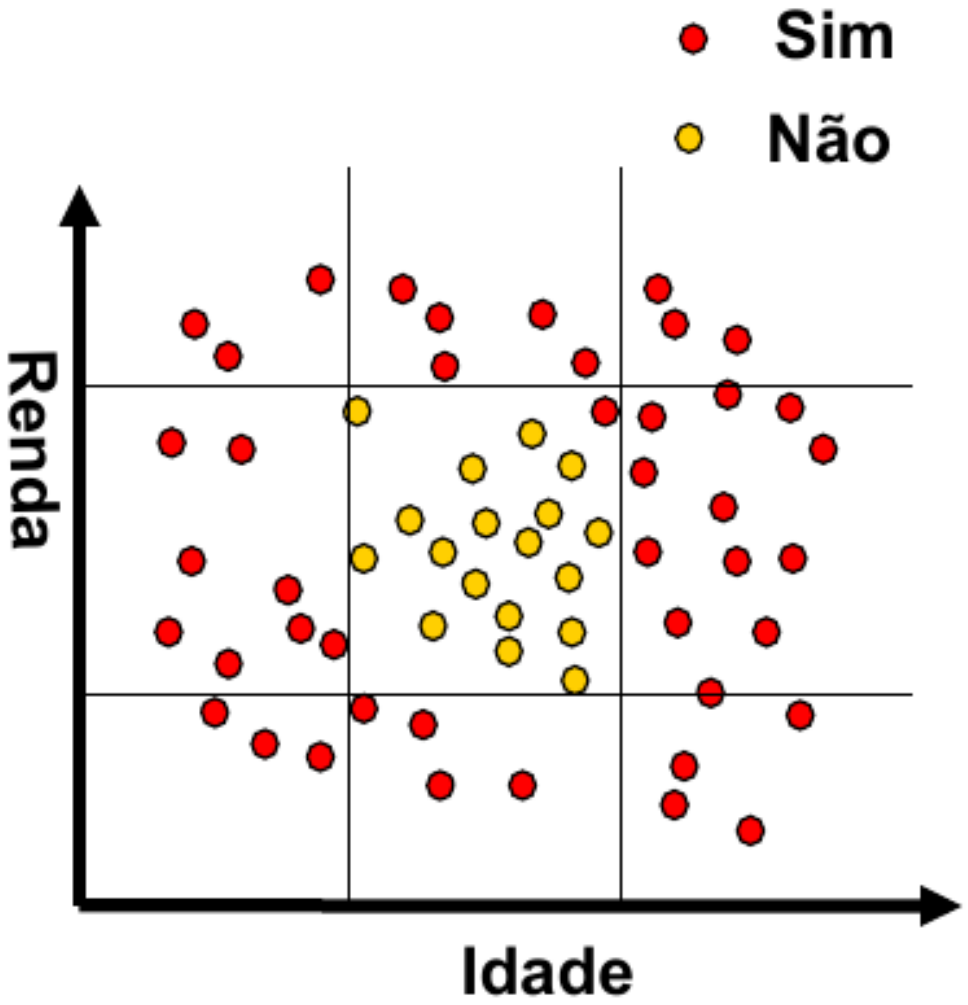
Indução (9)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
Sim
Idade > 45
Sim
Não
Sim
6 - Sim
19 - Não

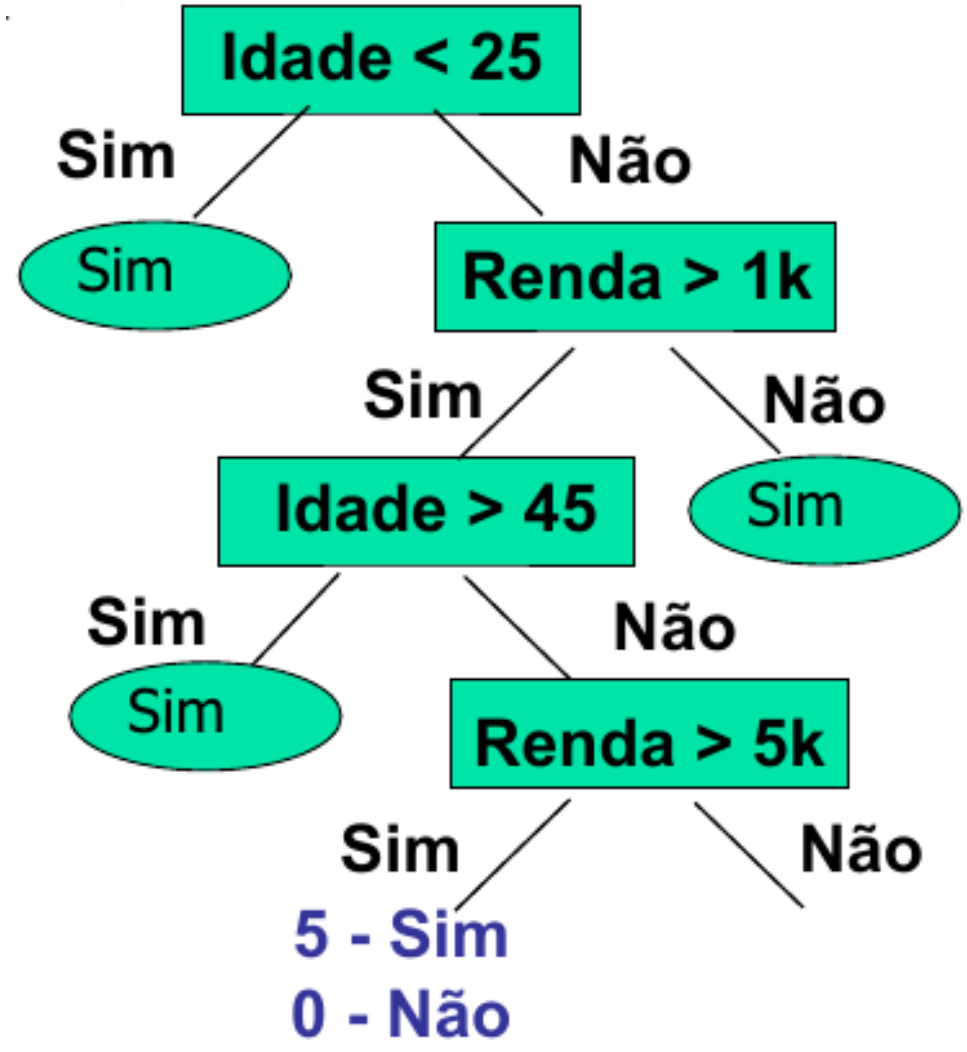

Sim
Não
Renda
Idade

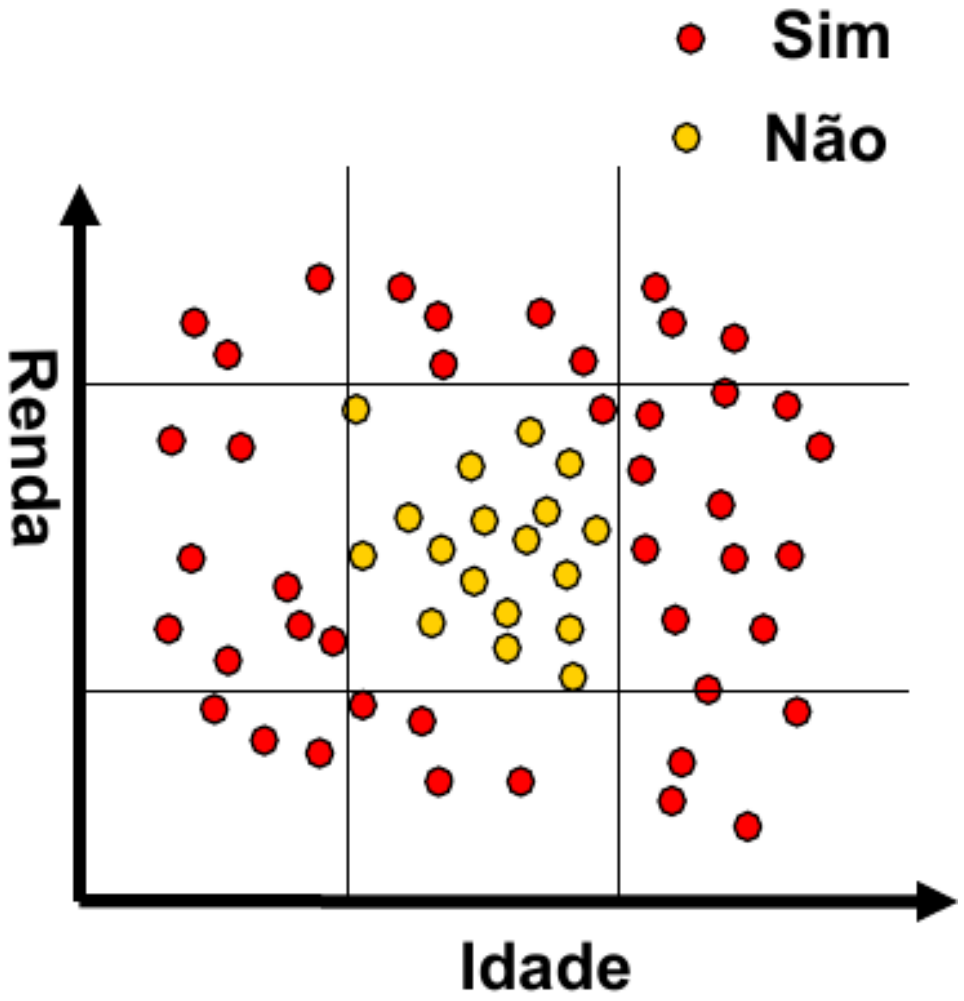
Indução (10)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
Sim
Idade > 45
Sim
Não
Sim
Renda > 5k
Sim
Não

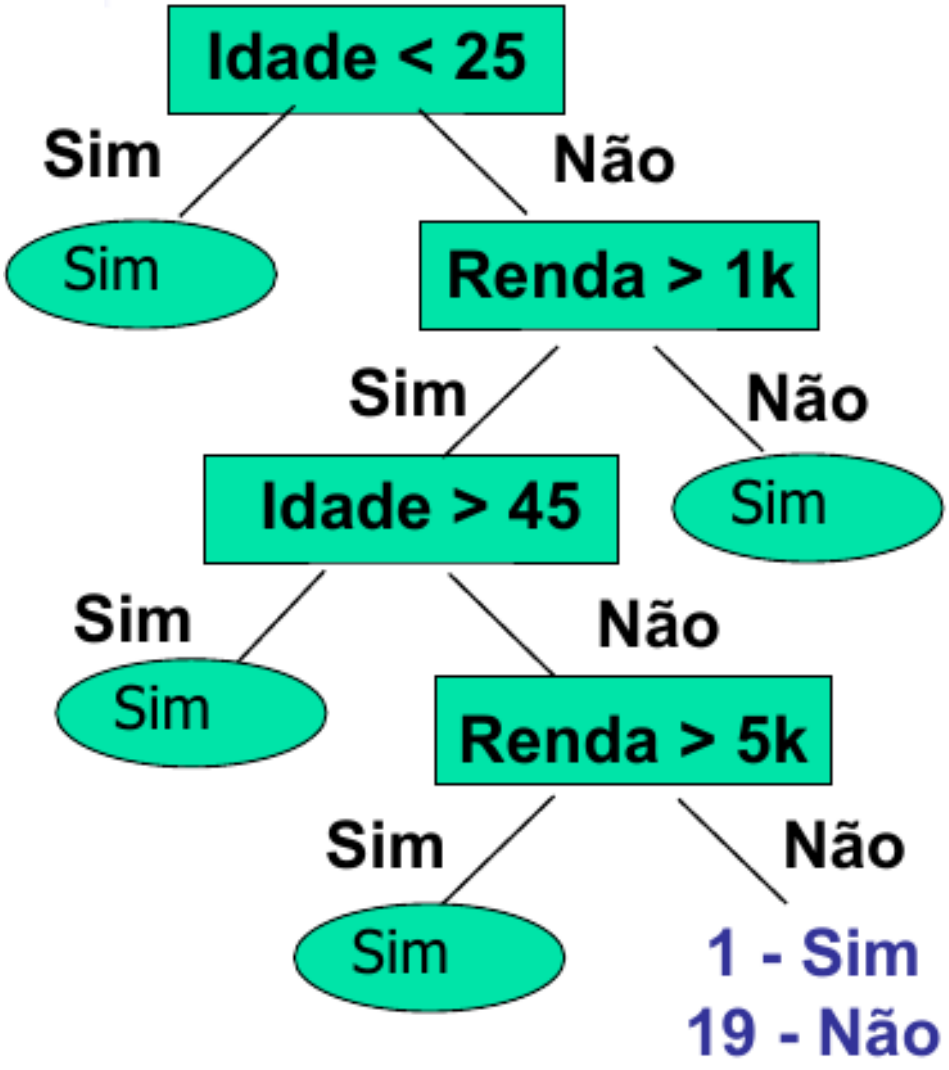

Sim
Não
Renda
Idade

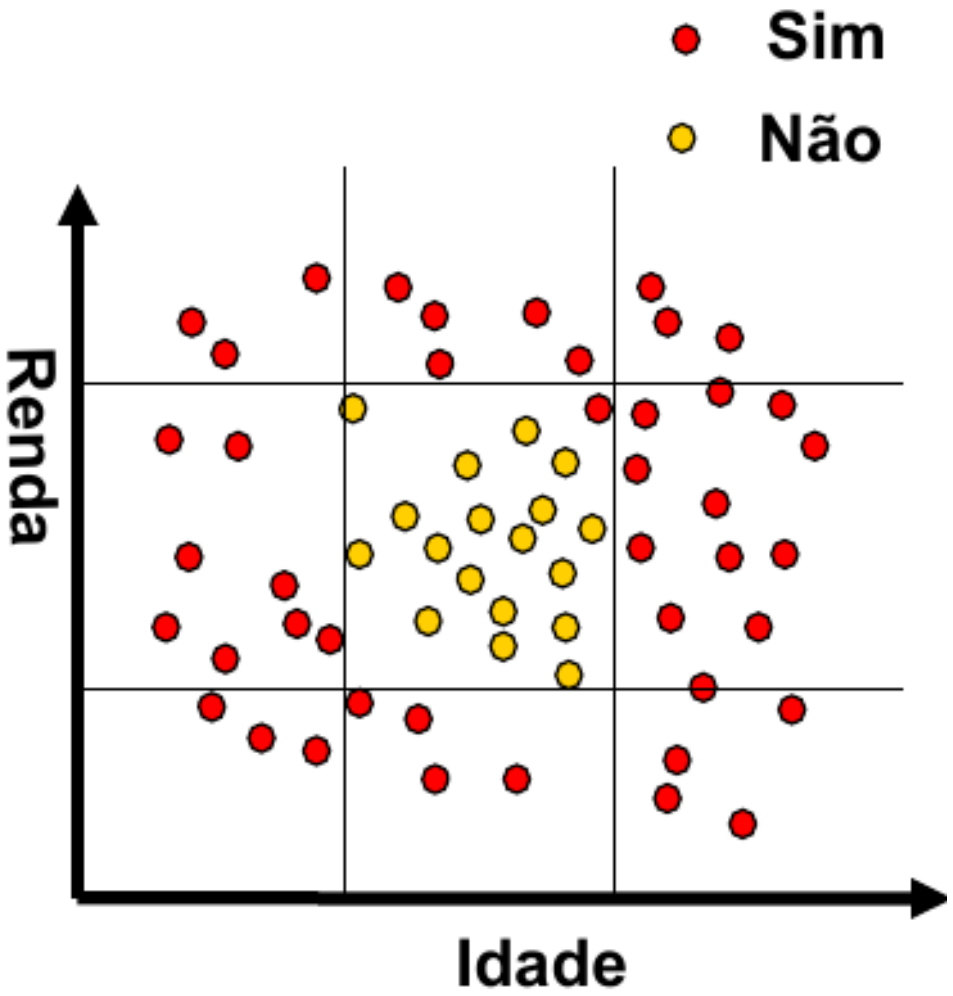
Indução (11)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
Idade > 45
Sim
Sim
Não
Sim
Renda > 5k
Sim
Não
5 - Sim
0 - Não

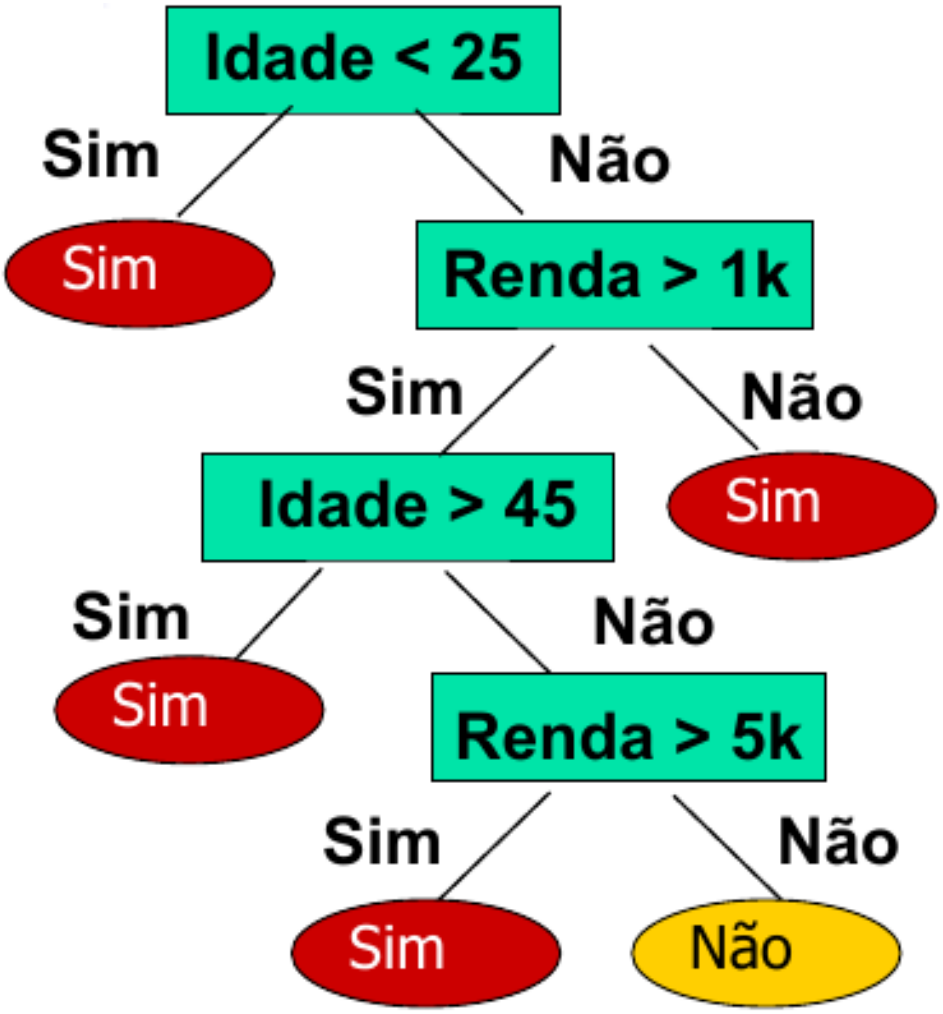

Sim
Não
Renda
Idade

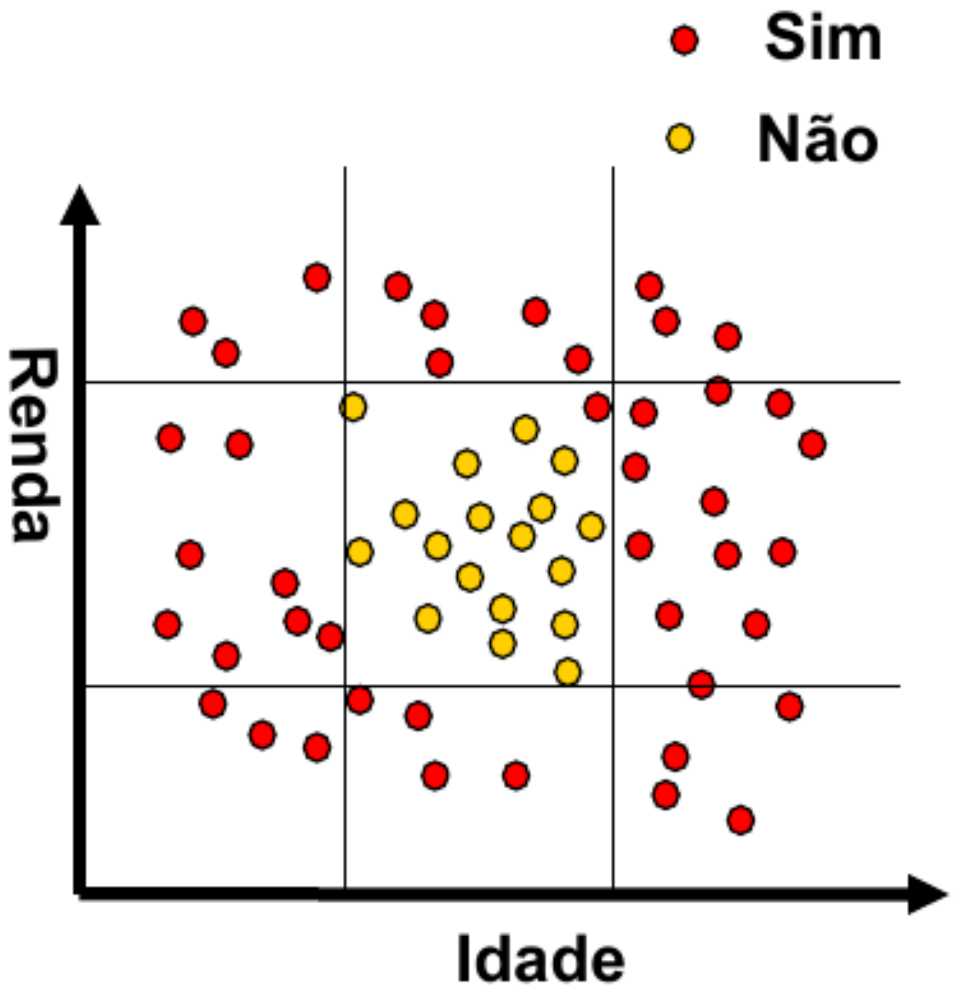
Indução (12)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
Idade > 45
Sim
Sim
Não
Sim
Renda > 5k
Sim
Não
Sim
1 - Sim
19 - Não


Sim
Não
Renda
Idade

Indução (13)


Idade < 25
Sim
Não
Sim
Renda > 1k
Sim
Não
Idade > 45
Sim
Sim
Não
Sim
Renda > 5k
Sim
Não
Sim
Não


Sim
Não
Renda
Idade

Vantagens e Desvantagens

- Vantagens
  - Indica quais atributos são mais importantes para a classificação
  - Atributos de entrada podem ser numéricos ou categóricos
- Desvantagem: hipercubos são paralelos aos eixos
  - Tem mais dificuldade de analisar classes cujos atributos são proporcionais ou inversamente proporcionais

Algoritmo C4.5 (J48)

- Indução simples por busca em profundidade
- Usa ganho de informação
- Ordena atributos contínuos em cada nó
- Todos os dados precisam caber na memória principal
- Inadequado para grandes conjuntos de dados
  - Ordenação será feita fora da memória principal

Overfitting

- Partição recursiva pode gerar árvores perfeitamente ajustadas aos dados
- Decisões são baseadas em conjuntos cada vez menores de dados
  - Níveis mais profundos podem ter muito poucos dados
    - Presença de ruído nos dados afeta bastante a decisão para esses nós
    - Reduz capacidade de generalização

Overfitting (2)

- Navalha de Ockham (Ockham's razor)
  - Quanto mais simples a solução, melhor
  - Preferir as hipóteses mais simples
    - Quando hipótese mais simples explica os dados, é pouco provável que seja coincidência
    - Explicação dos dados por uma hipótese mais complexa pode ser apenas uma coincidência
    - Árvore de decisão pode ser simplificada por poda

Poda de Árvore

- Elimina parte da árvore
- Pode ser realizada em duas etapas
  - Durante indução (pré-poda)
    - Parar o crescimento da árvore mais cedo
  - Após indução (pós-poda)
    - Crescer a árvore completa e depois podá-la
    - Mais lento, porém mais confiável

Algoritmo Simples de Poda

Percorrer a arvore em profundidade
Para cada nó i de decisão
    Ei = erro no nó
    ES = soma dos erros nos nós descendentes
    Se Ei ≤ ES
    Então nó Ei é transformado em nó folha

Usar conjunto de validação para a poda

Exercício

- Seja o seguinte cadastro de pacientes:

Nome	Febre	Enjôo	Manchas	Dores	Diagnóstico
João	sim	sim	pequenas	sim	doente
Pedro	não	não	grandes	não	saudável
Maria	sim	sim	pequenas	não	saudável
José	sim	não	grandes	sim	doente
Ana	sim	não	pequenas	sim	saudável
Leila	não	não	grandes	sim	doente

Exercício (2)

- Usando medidas Gini e GiniP, induzir uma árvore de decisão capaz de distinguir:
  - Pacientes potencialmente saudáveis (slide anterior)
  - Pacientes potencialmente doentes
- Testar a árvore para novos casos
  - (Luis, não, não, pequenas, sim)
  - (Laura, sim, sim, grandes, sim)

Conclusão

- Introdução
- Algoritmo de Hunt
- Medidas para selecionar divisão de atributos
- Ponto de referência
- Critério de parada
- Espaço de hipóteses
- Poda

Agradecimentos/referências

- Notas de aula do Prof. André de Carvalho (USP)