# Mineração de Dados
# 2018.2

## Árvore de decisão

Thiago Ferreira Covões

(slides baseados no material do Prof. Eamonn Keogh [eamonn@cs.ucr.edu] e Prof. Eduardo R. Hruschka [USP])

# Inferindo regras rudimentares:

- 1R: aprende uma árvore de decisão de um nível.
  - Todas as regras usam somente um atributo.
- Versão Básica:
  - Um ramo para cada valor do atributo;
  - Para cada ramo, atribuir a classe mais frequente;
  - Taxa de erro de classificação: proporção de exemplos que não pertencem à classe majoritária do ramo correspondente;
  - Escolher o atributo com a menor taxa de erro de classificação;
    - Atributos nominais/categóricos;
    - Há vários algoritmos de discretização para definir estratégias de corte nos valores dos atributos (<=, <, >, >=).

# Algoritmo 1R em pseudo-código:

**Para cada atributo:**

**Para cada valor do atributo gerar uma regra como segue:**

**Contar a freqüência de cada classe;**

**Encontrar a classe mais freqüente;**

**Formar uma regra que atribui à classe mais freqüente este atributo-valor;**

**Calcular a taxa de erro de classificação das regras;**

**Escolher as regras com a menor taxa de erro de classificação.**

# 1R para o problema *weather* :

| Outlook | Temp | Humidity | Windy | Play |
|---|---|---|---|---|
| Sunny | Hot | High | False | No |
| Sunny | Hot | High | True | No |
| Overcast | Hot | High | False | Yes |
| Rainy | Mild | High | False | Yes |
| Rainy | Cool | Normal | False | Yes |
| Rainy | Cool | Normal | True | No |
| Overcast | Cool | Normal | True | Yes |
| Sunny | Mild | High | False | No |
| Sunny | Cool | Normal | False | Yes |
| Rainy | Mild | Normal | False | Yes |
| Sunny | Mild | Normal | True | Yes |
| Overcast | Mild | High | True | Yes |
| Overcast | Hot | Normal | False | Yes |
| Rainy | Mild | High | True | No |

| Atributo | Regra | Erros | Total erros |
|---|---|---|---|
| Outlook | Sunny → No | 2/5 | 4/14 |
| | Overcast → Yes | 0/4 | |
| | Rainy → Yes | 2/5 | |
| Temp | Hot → No* | 2/4 | 5/14 |
| | Mild → Yes | 2/6 | |
| | Cool → Yes | 1/4 | |
| Humidity | High → No | 3/7 | 4/14 |
| | Normal → Yes | 1/7 | |
| Windy | False → Yes | 2/8 | 5/14 |
| | True → No* | 3/6 | |

* empate

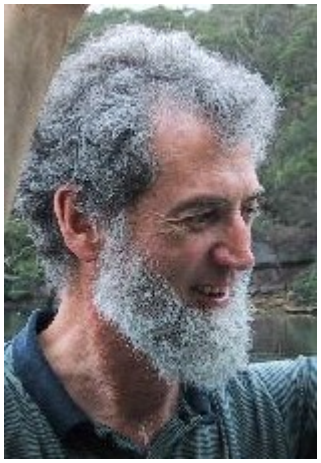
Qual seria a capacidade de generalização do modelo?

# Discussão para o 1R:

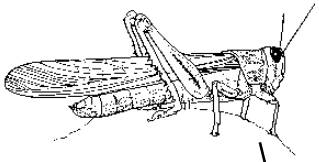
- 1R foi descrito por Holte (1993):
  - Contém uma avaliação experimental em 16 bases de dados;
  - Em muitos *benchmarks*, regras simples não são muito piores do que árvores de decisão mais complexas...
  - Complexidade de tempo?
- Atualmente usado para análise exploratória de dados
- Árvores de Decisão estendem essa ideia

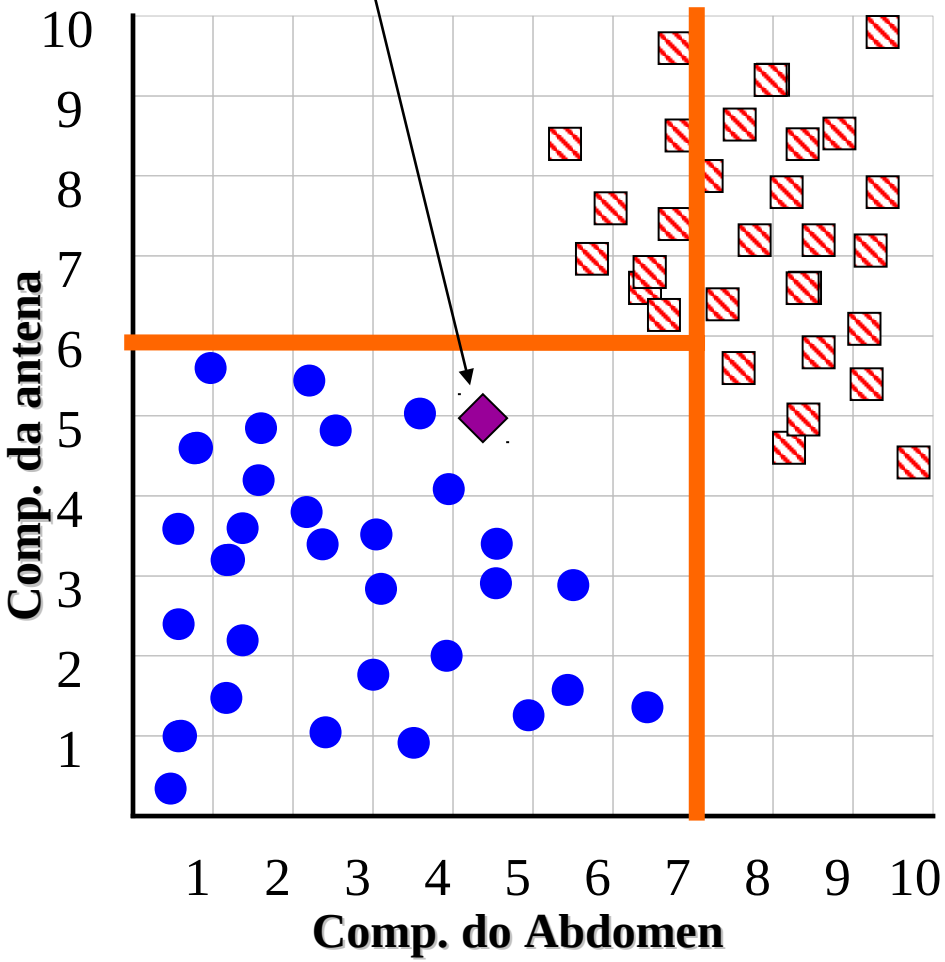
Holte, Robert C., Very Simple Classification Rules Perform Well on Most Commonly Used Datasets, *Machine Learning* 11 (1), pp. 63-90, 1993.

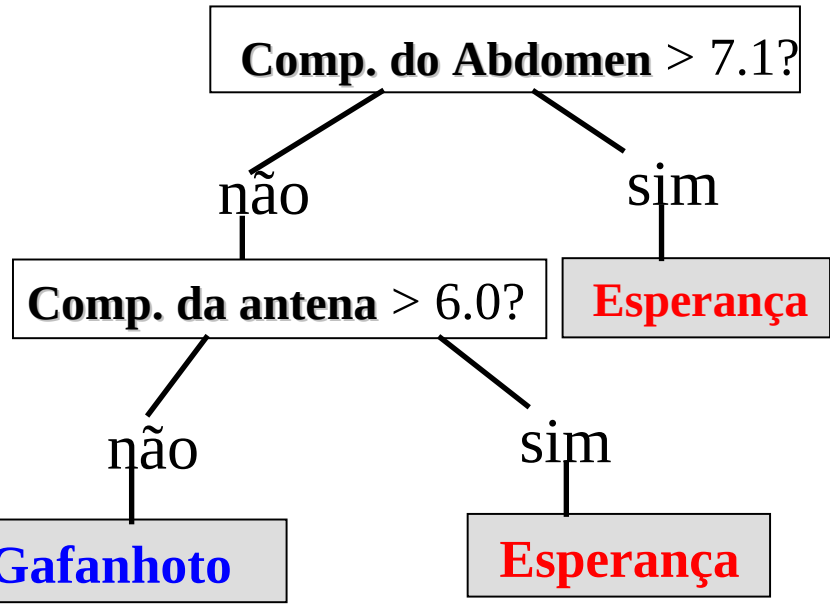
# Classificador:Árvore de decisão



Ross Quinlan





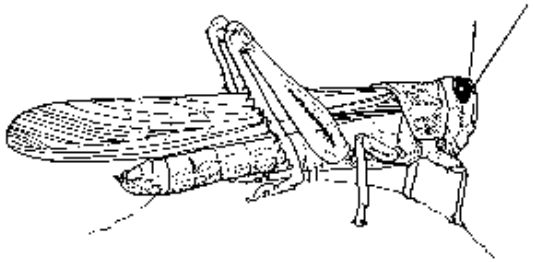



# Árvores de Decisão

- Métodos para aproximar funções discretas, representadas por meio de uma árvore de decisão;
- Árvores de decisão podem ser representadas por conjuntos de regras “*se...então*”;
  - *compreensibilidade*;
- Muito utilizadas em aplicações práticas, principalmente em problemas de classificação.

**Antena mais curta que o corpo?**

**sim**



Gafanhoto

**não**

**3 tarso?**

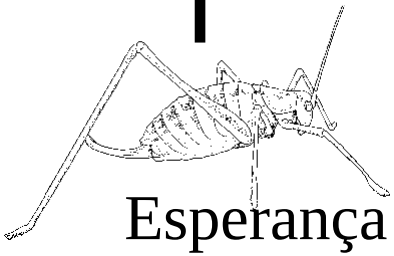
**sim**

Grilo

**não**

**Foretibia possui ouvidos?**

**sim**



Esperança

**não**

Grilo camelo

Árvores de decisão são mais antigas que a computação

# Classificação com Árvores de decisão

- Árvore de decisão
  - Uma estrutura de fluxo parecida com uma árvore
  - Nós internos denotam um teste ou atributo
  - Ramos representam um resultado do teste
  - Nós folha representam rótulos de classe ou distribuição de classes
- Geração de Árvores de decisão consiste de 2 fases
  - Construção da árvore
    - No início, todos os exemplos de treinamento estão na raiz
    - Particiona exemplos recursivamente baseando-se nos atributos selecionados
  - Poda da árvore
    - Identificar e remover ramos que refletem ruído ou outliers
- Uso da Árvore de decisão: Classificação de um exemplo desconhecido
  - Testa-se os valores dos atributos do exemplo na Árvore de decisão

# Como construir Árvores de decisão?

- Algoritmo básico (guloso)
  - Árvore é construída de cima pra baixo, recursivamente, no método de divisão e conquista
  - No início, todos os exemplos de treinamento estão na raiz da árvore
  - Exemplos são particionados recursivamente, baseando-se nos atributos selecionados.
  - Os atributos teste são selecionados com base em uma heurística ou medida estatística (por exemplo, ganho de informação)

# Como construir Árvores de decisão?

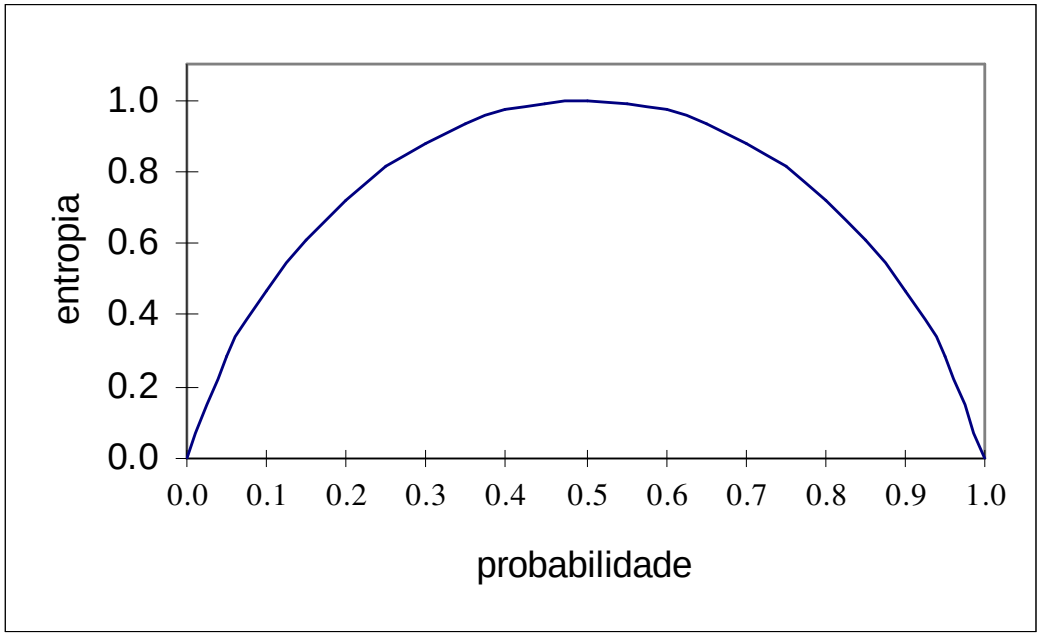
- Condições para parar o particionamento
  - Todos os exemplos para um dado nó pertencem a uma mesma classe
  - Não há mais atributos restantes para particionamento – votação da maioria é utilizada para classificar a folha
  - Não existem mais exemplos disponíveis
  - Número mínimo de exemplos (parâmetro) foi alcançado

# Ganho de informação como critério de divisão

- Selecione o atributo com o maior ganho de informação (ganho de informação é a redução esperada da entropia).
- Assuma que há C classes
  - Seja o conjunto de exemplos **X,** contendo $|X^c|$ elementos da classe *c*
  - A quantidade de informação necessária para decidir se um exemplo arbitrário em *S* pertence a uma das classes é definido como
  - Assume-se que 0*log(0) = 0

$$entropia(\mathcal{X}) = -\sum_{c=1}^{C} \frac{|\mathcal{X}^c|}{|\mathcal{X}|} \log_2 \frac{|\mathcal{X}^c|}{|\mathcal{X}|}$$

# Função “probabilidade x entropia” para classificação booleana:




- Lembrando que $\log_2 1=0$ e definindo $\log_2 0=0$

# Ganho de informação na indução de Árvores de decisão

- Assuma que usando o atributo A, um conjunto atual será particionado em um número de conjuntos filhos
  - $T = \{T_1, \ldots, T_k\}$
  - Atributo nominal: um ramo para cada valor possível
  - Atributo contínuo: definir um limiar e separar entre {<=,>}

# Ganho de informação na indução de Árvores de decisão

- Assuma que usando o atributo A, um conjunto atual será particionado em um número de conjuntos filhos
  - $T = \{T_1, \ldots, T_k\}$

- A informação codificada que será ganha pelo ramo em $A$

$$info(\mathcal{X}, T) = \sum_{T_i \in T} \frac{|T_i|}{|T|} entropia(T_i)$$

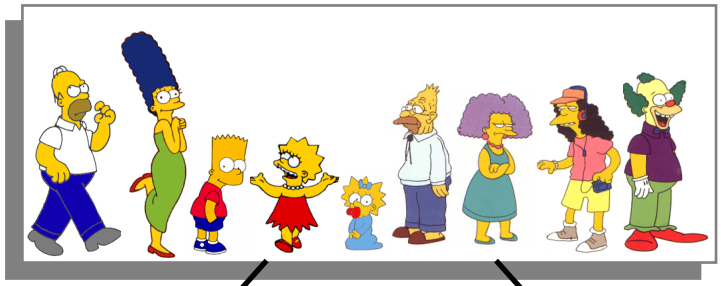
$$gain(\mathcal{X}, T) = entropia(\mathcal{X}) - info(\mathcal{X}, T)$$

| Pessoa | Comp. cabelo | Peso | Idade | Classe |
|---|---|---|---|---|
|  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** |

| | | | | |
|---|---|---|---|---|
|  Comic | 8” | 290 | 38 | **?** |

| Pessoa | Comp. cabelo | Peso | Idade | Classe |
|---|---|---|---|---|
|  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** |

Dividir por comprimento de cabelo <= 5”, peso <= 160 ou idade <= 40?



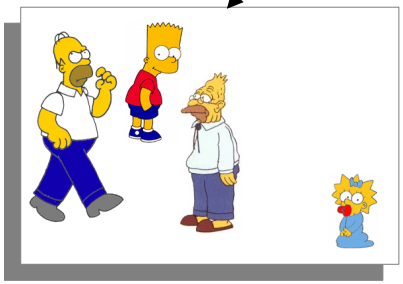
$$Entropia(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$

*Entropia*(4**F**,5**M**) = -(4/9)$\log_2$(4/9) - (5/9)$\log_2$(5/9)
= **0.9911**

sim

não

Comp. cabelo <= 5?

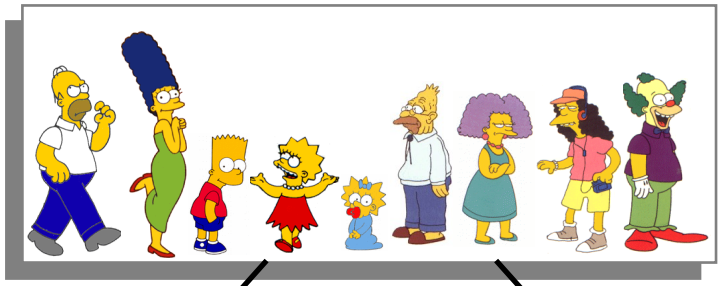


Vamos tentar dividir usando o atributo *Comp. cabelo*

*Entropia*(1**F**,3**M**) = -(1/4)$\log_2$(1/4) - (3/4)$\log_2$(3/4)
= ***0.8113***

*Entropia*(3**F**,2**M**) = -(3/5)$\log_2$(3/5) - (2/5)$\log_2$(2/5)
= ***0.9710***

*Gain*(Comp. cabelo <= 5) = **0.9911** – (4/9 * **0.8113** + 5/9 * **0.9710** ) = **0.0911**



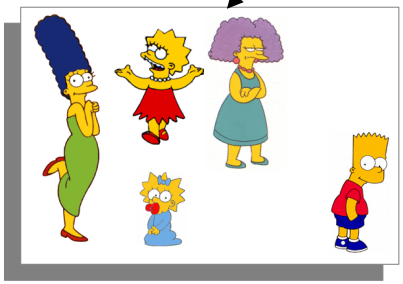
$$Entropia(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$

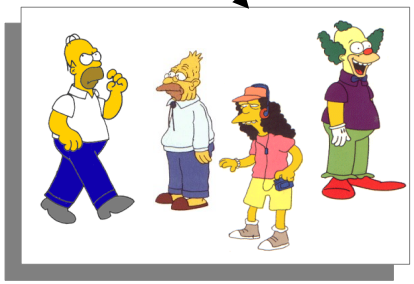
*Entropia*(4**F**,5**M**) = -(4/9)$\log_2$(4/9) - (5/9)$\log_2$(5/9)
= **0.9911**

sim

não

Peso <= 160?



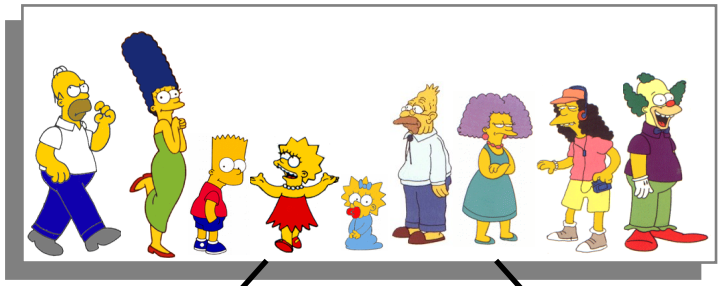


Vamos tentar dividir usando o atributo *Peso*

*Entropia*(4**F**,1**M**) = -(4/5)$\log_2$(4/5) - (1/5)$\log_2$(1/5)
= **0.7219**

*Entropia*(0**F**,4**M**) = -(0/4)$\log_2$(0/4) - (4/4)$\log_2$(4/4)
= **0**

*Gain*(Peso <= 160) = **0.9911** – (5/9 * **0.7219** + 4/9 * **0** ) = **0.5900**



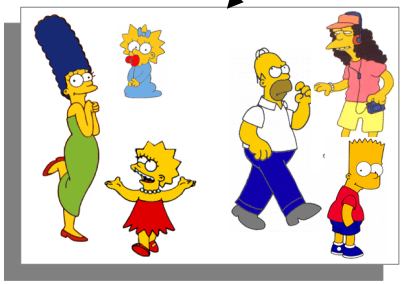
$$Entropia(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$

*Entropia*(4**F**,5**M**) = -(4/9)$\log_2$(4/9) - (5/9)$\log_2$(5/9)
= **0.9911**

sim

não

Idade <= 40?



Vamos tentar dividir usando o atributo *Idade*

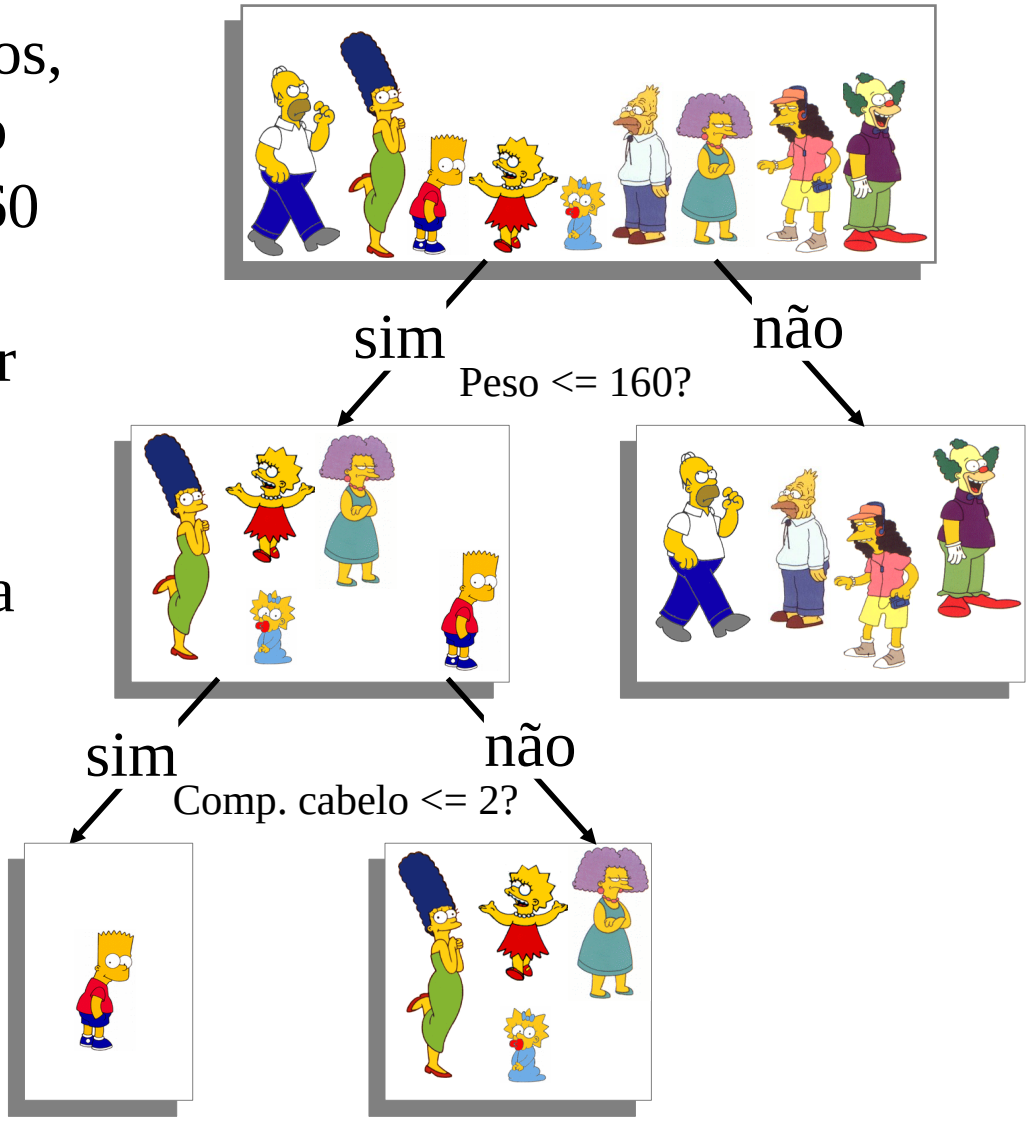
*Entropia*(3**F**,3**M**) = -(3/6)$\log_2$(3/6) - (3/6)$\log_2$(3/6)
= **1**

*Entropia*(1**F**,2**M**) = -(1/3)$\log_2$(1/3) - (2/3)$\log_2$(2/3)
= **0.9183**

*Gain*(Idade <= 40) = **0.9911** – (6/9 * **1** + 3/9 * **0.9183** ) = **0.0183**

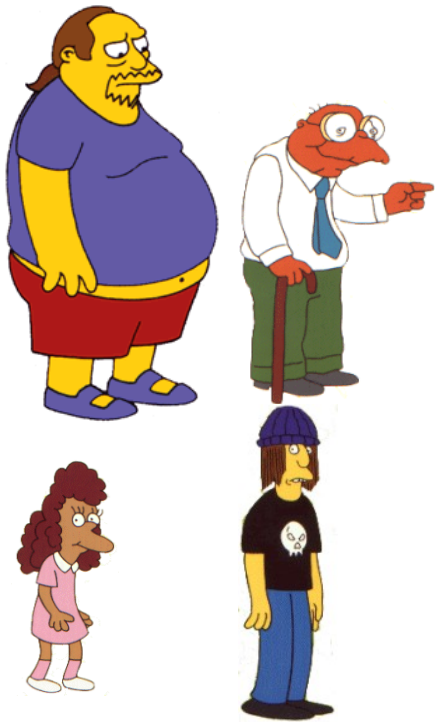
Das 3 características que tínhamos, *Peso* foi a melhor. Mas enquanto as pessoas que pesam mais de 160 são perfeitamente classificadas (como homens), com peso menor que 160 os exemplos não são perfeitamente classificados. Portanto, simplesmente usamos a recursividade!

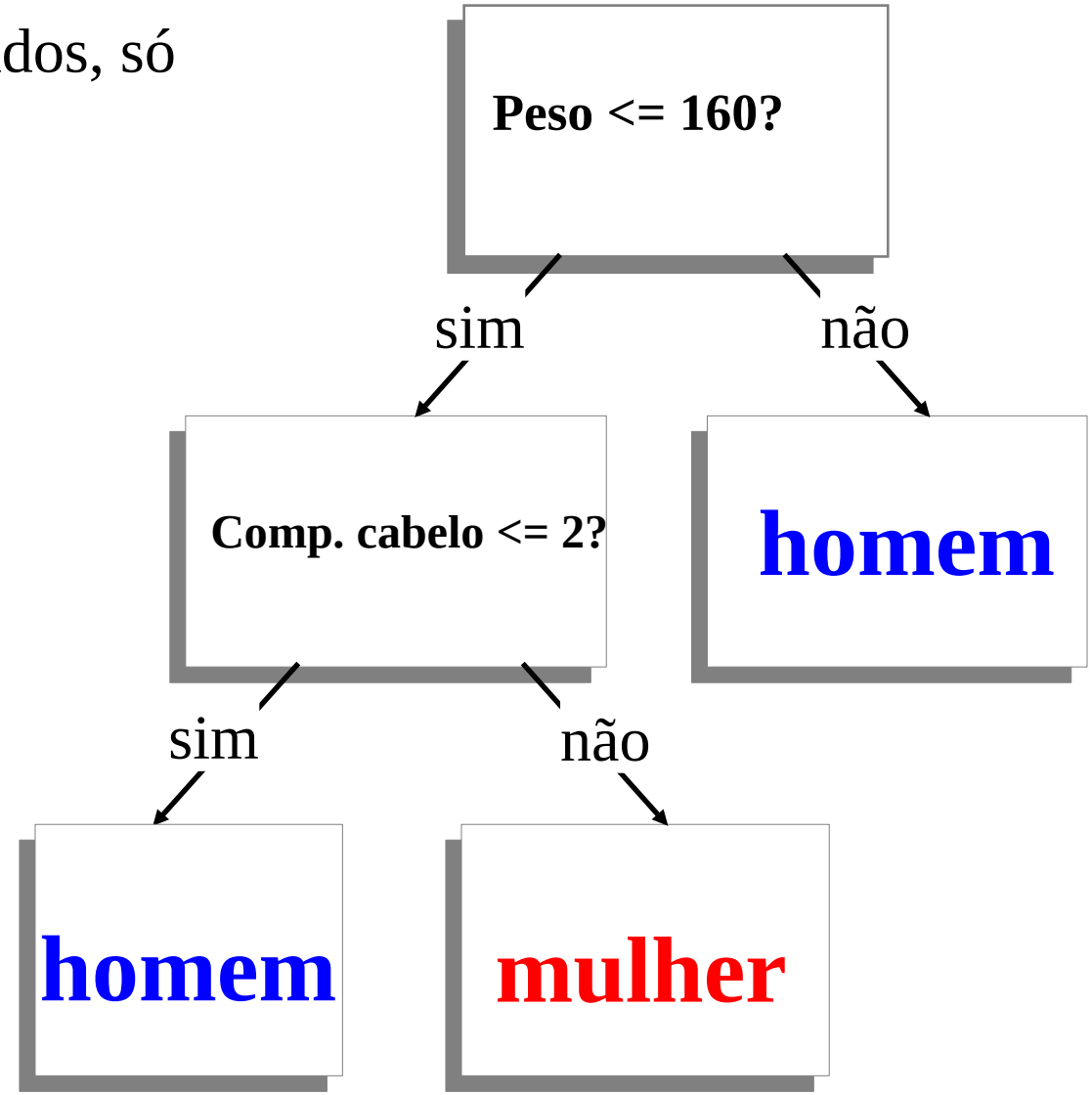
Desta vez descobrimos que podemos dividir *Comp. cabelo*, e está pronto!



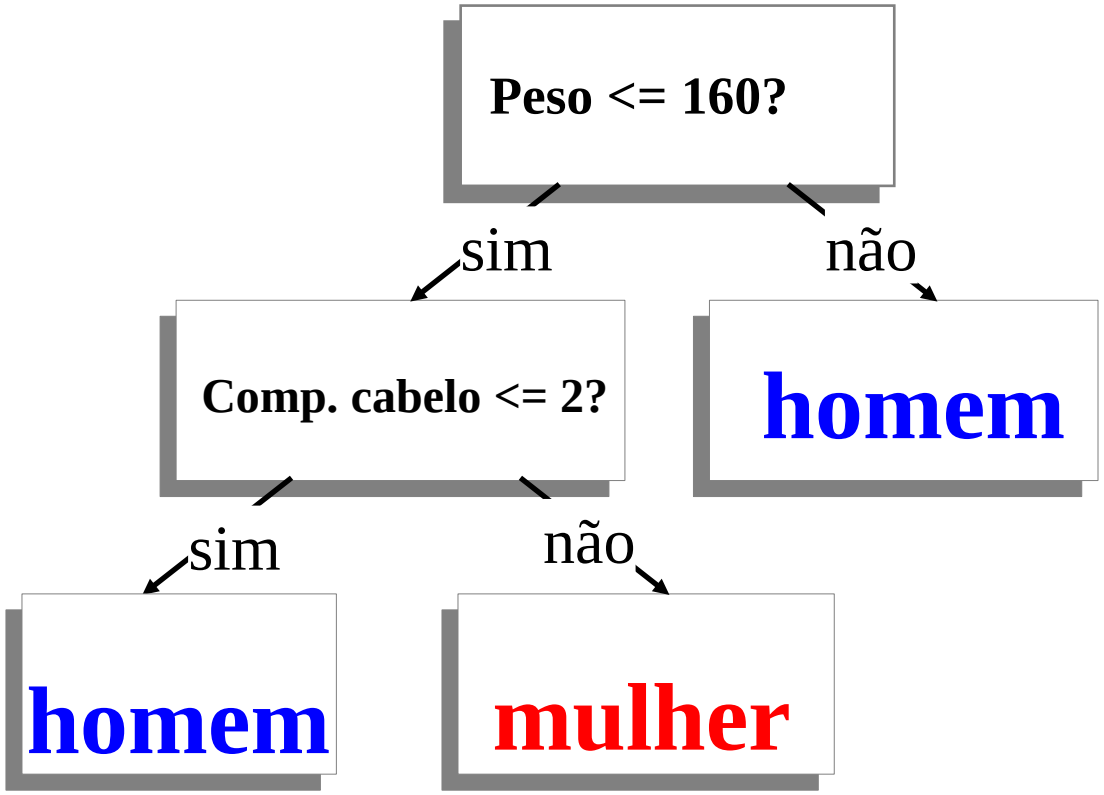
Não precisamos manter os dados, só as condições de teste.

Como essas pessoas seriam classificadas?





É trivial converter árvores de decisão em regras…

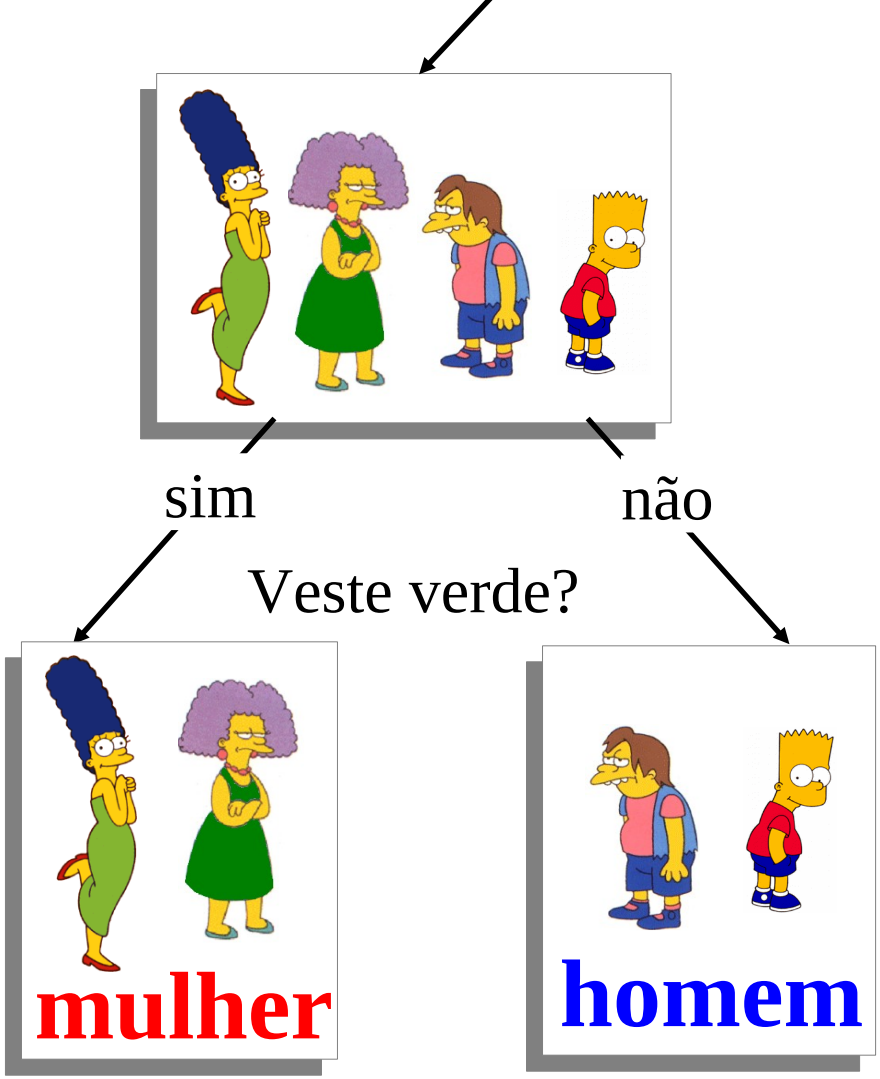



**Regras para classificar homens/mulheres**

**se** *Peso* **maior que** 160, classificar como **homem**
**senão se** *Comp. cabelo* **menor que ou igual a** 2, classificar como **homem**
**senão** classificar como **mulher**

Os exemplos que vimos foram realizados em pequenos conjuntos de dados. Entretanto, com pequenos conjuntos de dados há um grande risco de super adequação dos dados (overfitting)

Quando se tem poucos dados, há muitas regras de divisão que classificarão perfeitamente os dados, mas que não generalizarão o conhecimento para futuros conjuntos de dados.



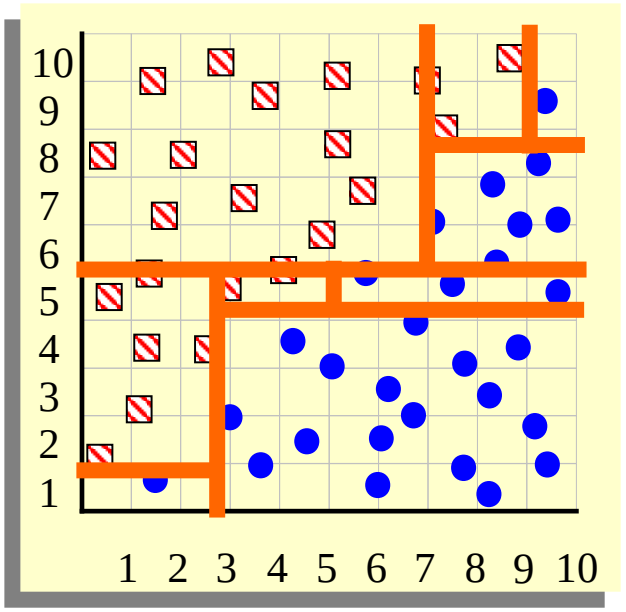

Por exemplo, a regra “Veste verde?” classifica perfeitamente os dados, assim como “O nome da mãe é Jacqueline?”, assim como “Tem sapatos azuis”…

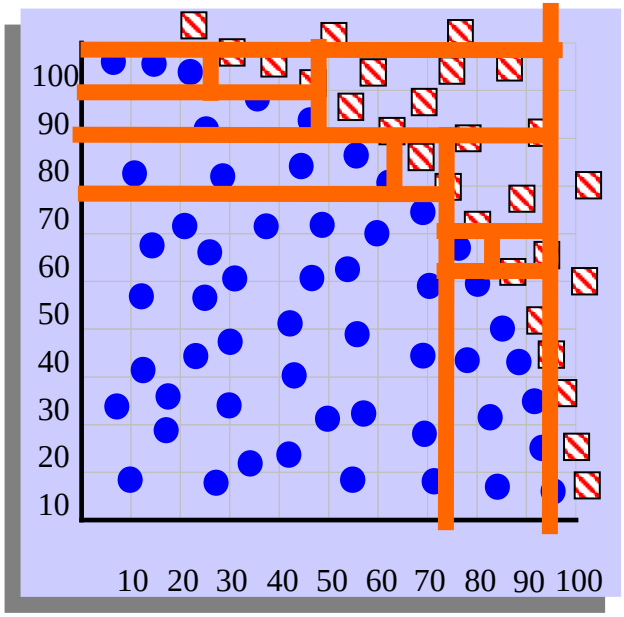
# Como evitar overfitting na classificação?

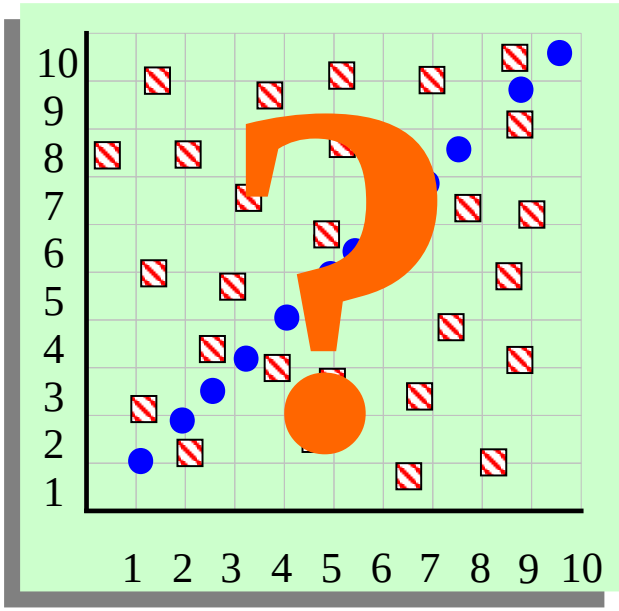
- A árvore gerada pode super adequar os dados de treinamento
  - Ramos demais: alguns podem refletir anomalias devido a ruídos ou outliers
  - Resultado é pobre em acurácia para exemplos não vistos
- Duas abordagens para evitar overfitting
  - Poda prévia: Pare a construção da árvore cedo – não divida um nó se isto fizer com que a medida de avaliação caia abaixo de um limiar
    - Difícil escolher um limiar apropriado
  - Poda posterior: Remova ramos da árvore completa – realize uma sequência progressiva de podas
    - Use um conjunto de dados diferente dos dados de treinamento para decidir qual é a melhor árvore podada

Qual dos dois “Problemas do Pombo” podem ser resolvidos por uma árvore de decisão?

1) Árvore profunda e serrilhada
2) Inútil
3) Árvore profunda e serrilhada




A árvore de decisão tem dificuldade com atributos correlatos

# Vantages e desvantagens de árvores de decisão

- Vantagens:
  - Fácil de entender
  - Fácil de gerar regras
- Desvantagens:
  - Sensível a ruídos
  - Pode apresentar overfitting
  - Classifica por meio de particionamentos retangulares (portanto não trata características correlatas muito bem)
  - Podem ser bem grandes – podar pode ser necessário

# Extensões

- Podem ser induzidas árvores com um número mínimo de objetos nas folhas
- Critério de ganho de informação tem um viés para atributos com muitos valores (existem diversos critérios na literatura)
  - Por exemplo: normalizar pela entropia do particionamento (gainRatio)
- Em cada folha pode-se ter um modelo simples para realizar predições
  - Modelo local
  - Naïve Bayes é um frequentemente utilizado nesse contexto
- Possível utilizar para regressão (discutido em outra aula)

# Extensões

- Discretização
  - Pode ser utilizado para discretizar um atributo contínuo considerando a classe
- Valores ausentes
  - Pode ser usado para estimar (compreensibilidade é uma vantagem)
  - No teste pode verificar os caminhos possíveis e ponderar pelas probabilidades observadas

# Para pensar

- Bias x Variância
  - Considerando esses dois critérios, como você classificaria os algoritmos vistos até o momento?