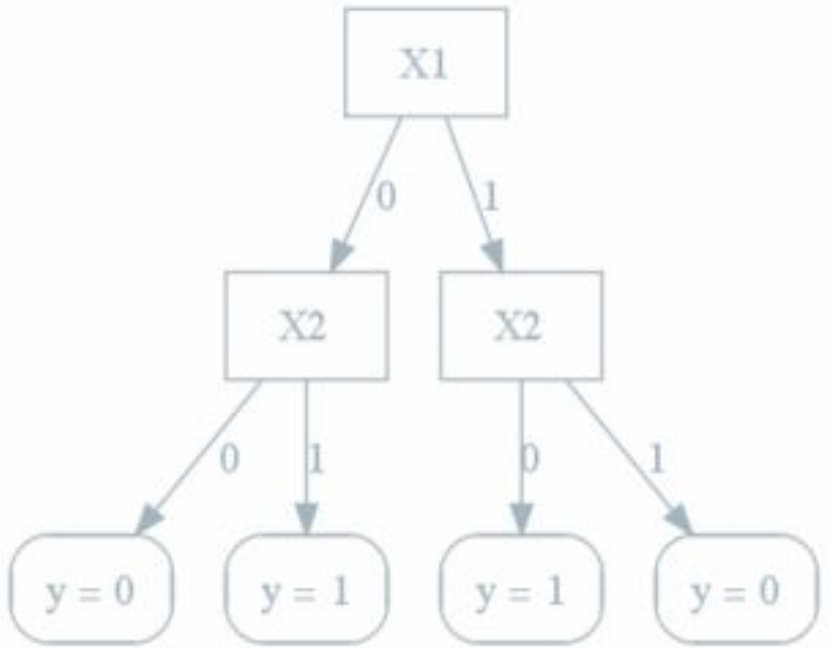


Aula 10

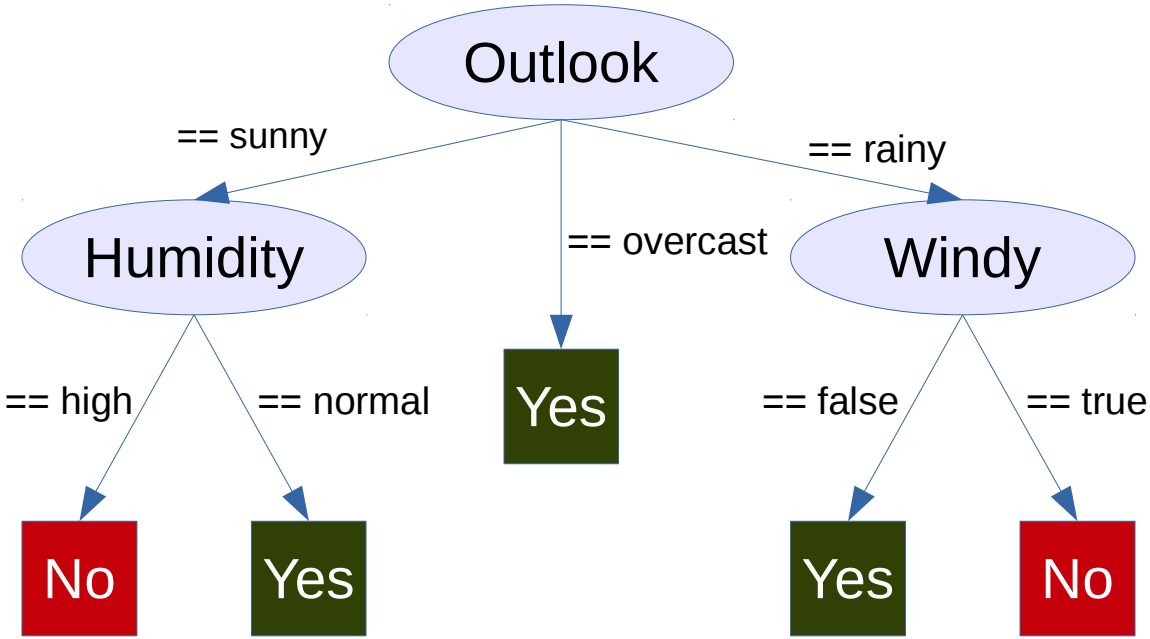
Árvores de Decisão

Rafael Geraldeli Rossi

Conteúdo

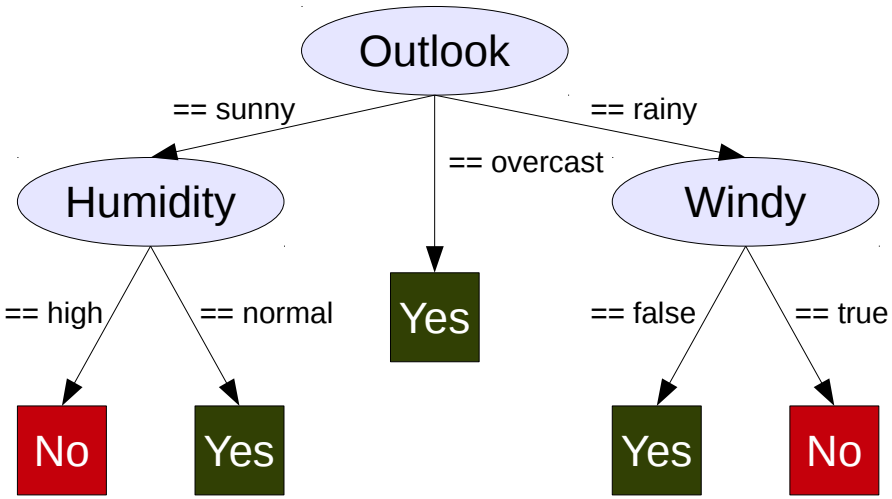
Introdução

- Uma árvore de decisão é uma estrutura na qual cada nó interno corresponde a um teste em um atributo, cada ramificação representa a saída de um teste, e cada nó folha representa um rótulo de classe



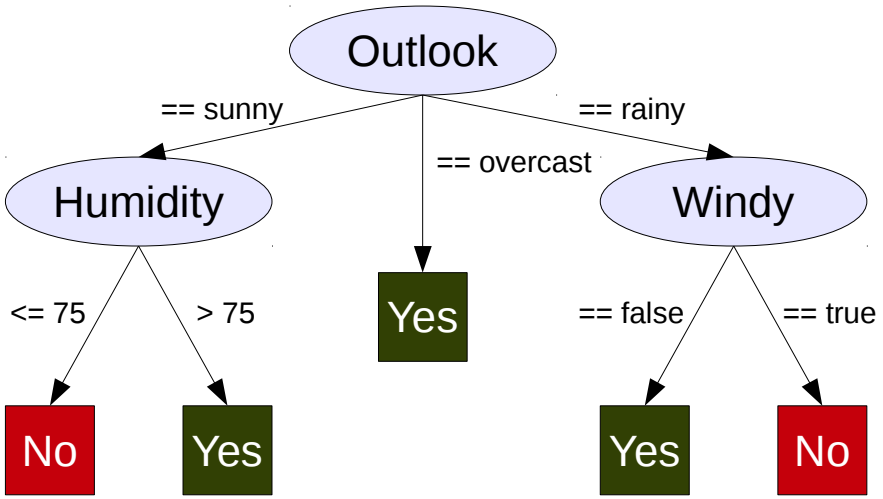
Introdução

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**



Introdução

Outlook	Temperature	Humidity	Windy	Classe
Sunny	85	85	FALSE	NO
Sunny	80	90	TRUE	NO
Overcast	83	86	FALSE	YES
Rainy	70	96	FALSE	YES
Rainy	68	80	FALSE	YES
Rainy	65	70	TRUE	NO
Overcast	64	65	TRUE	YES
Sunny	72	95	FALSE	NO
Sunny	69	70	FALSE	YES
Rainy	75	80	FALSE	YES
Sunny	75	70	TRUE	YES
Overcast	72	90	TRUE	YES
Overcast	81	75	FALSE	YES
Rainy	71	91	TRUE	NO



Introdução

- Paradigma de aprendizado **simbólico**, ou seja, os padrões gerados são facilmente interpretados

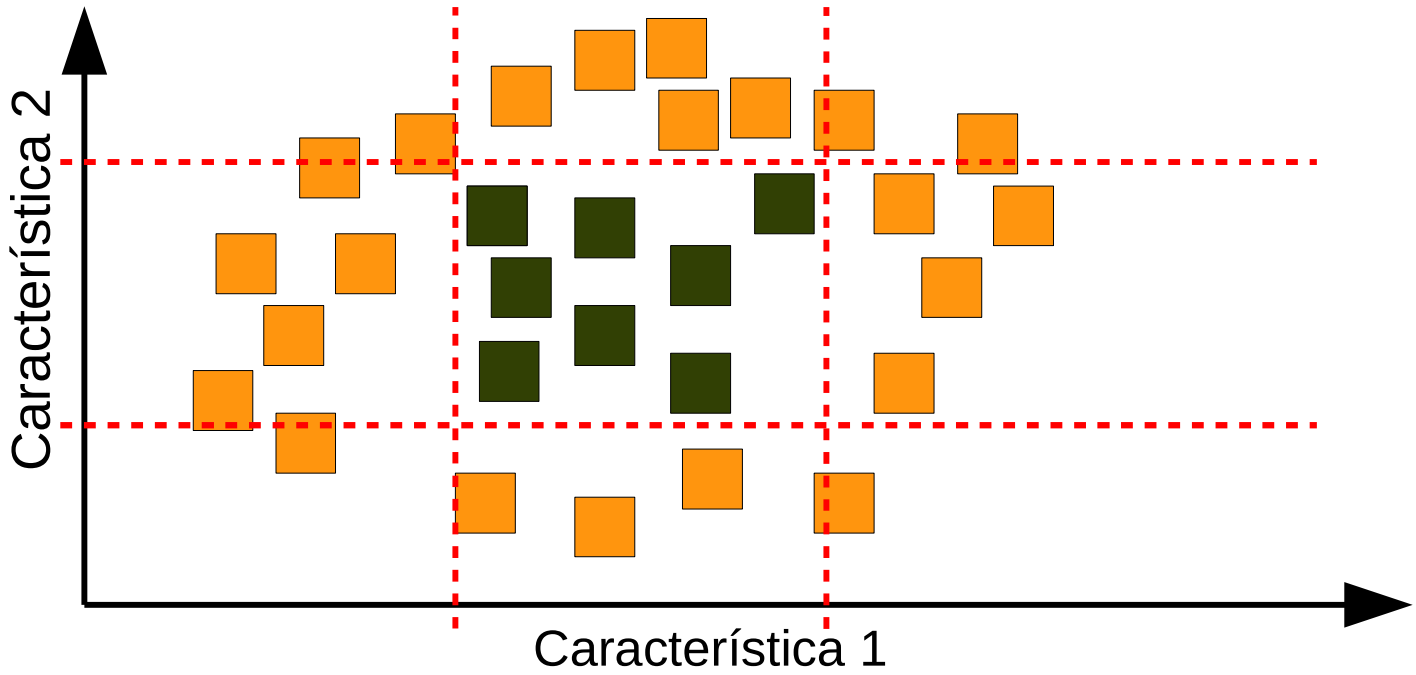
- Algoritmos para a indução de árvores de decisão
  - ID3 (Iterative Dichotomiser) [Quinlan, 1986]
  - **C4.5** (sucessor do ID3) [Quinlan, 1993]
  - Classification and Regression Trees (CART) [Breiman et al., 1984]

Introdução

- **Algoritmos de indução de árvores de decisão normalmente adotam uma estratégia GULOSA e to tipo top-down** $\rightarrow$ da raíz para as folhas
  - Diminuição da busca no espaço de hipótese $\rightarrow$ impraticável gerar todas as possíveis árvores de decisão para conjuntos de dados com algumas dezenas de atributos
  - Estratégia de dividir e conquistar de maneira recursiva - o conjunto de treinamento é recursivamente particionado em subconjuntos conforme a árvore é construída
  - **LEMBRETE**: nem sempre a estratégia gulosa irá levar a melhor árvore

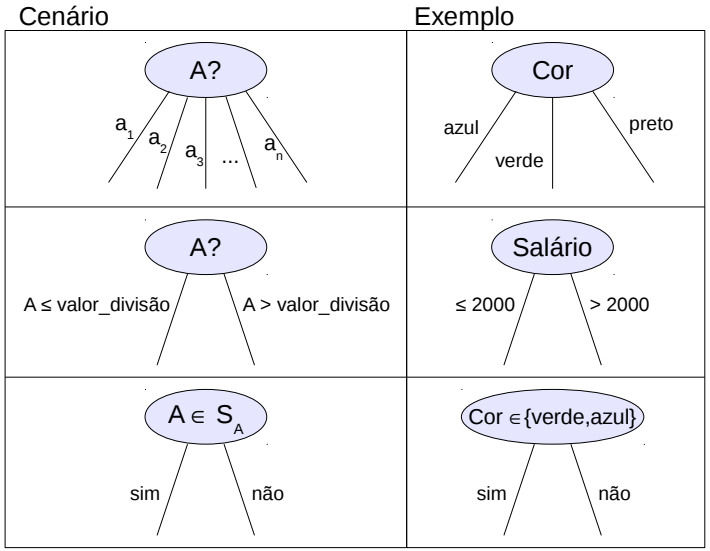
Introdução

- Uma árvore de decisão gera hiperplanos de separação perpendiculares aos eixos



Introdução

- As divisões, ou testes, podem dividir o conjunto de exemplos de acordo com cada valor de um atributo (discreto), se é maior ou menor que um valor de um atributo (numérico), ou se pertence ou não a um subconjunto de valores de um atributo (discreto)



Introdução

TABLE 4.3: Number of Possible Splits Based on Attribute Type

Attribute Type	Binary Split	Multiway Split
Binary	1	1 (same as binary)
Categorical (unordered)	$\frac{2^k-2}{2} = 2^{k-1} - 1$	one k-way split
Numerical (ordered)	$k - 1$	one k-way split

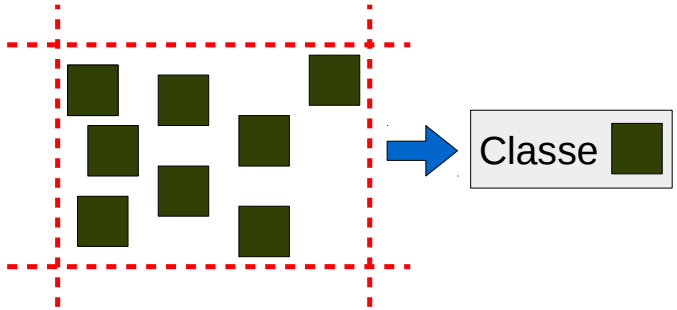
[Aggarwal, 2015]

Indução

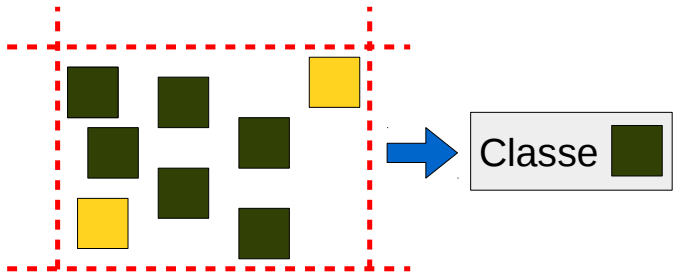
Algoritmo **Gerar_árvore_decisão**

- **Entrada**
  - D: conjunto de exemplos de treinamento
  - *lista_atributos*: conjunto de atributos candidatos
  - *método_seleção_atributos*: determina o critério de divisão que melhor particiona o conjunto de treinamento em relação às classes
  - *profundidade máxima*: determiná um número máximo de níveis da árvore de decisão
- **Saída**: Árvore de decisão

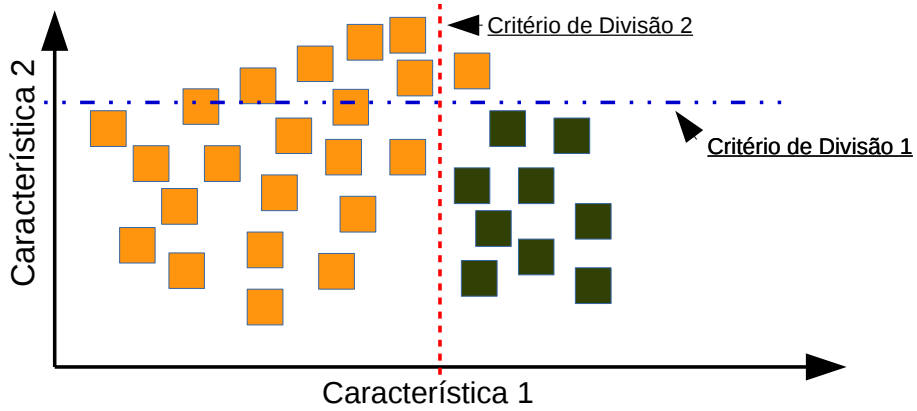
- **Método**
  1. crie um nó N
  2. **se** (os exemplos em D são todos da mesma classe C) **então**
  3. retorne N como um nó folha rotulado com a classe C



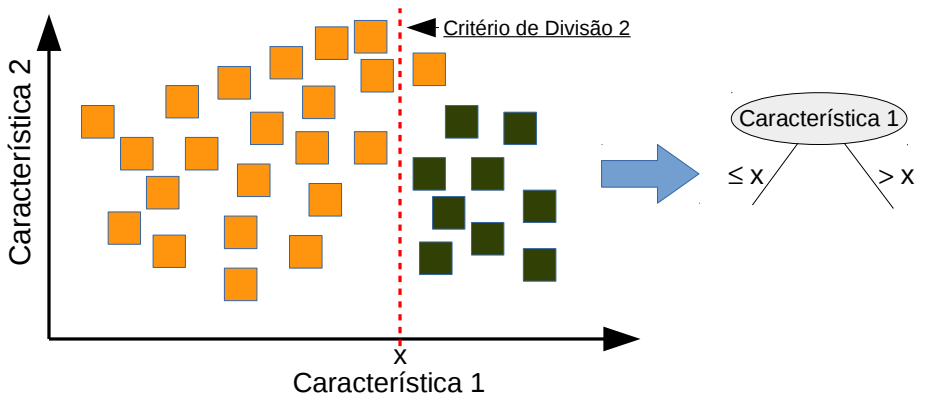
  4. **se** (*lista_atributos* está vazia) **ou** (profundidade máxima atingida) **então**
  5. retorne N como um nó folha rotulado com a classe majoritária em D



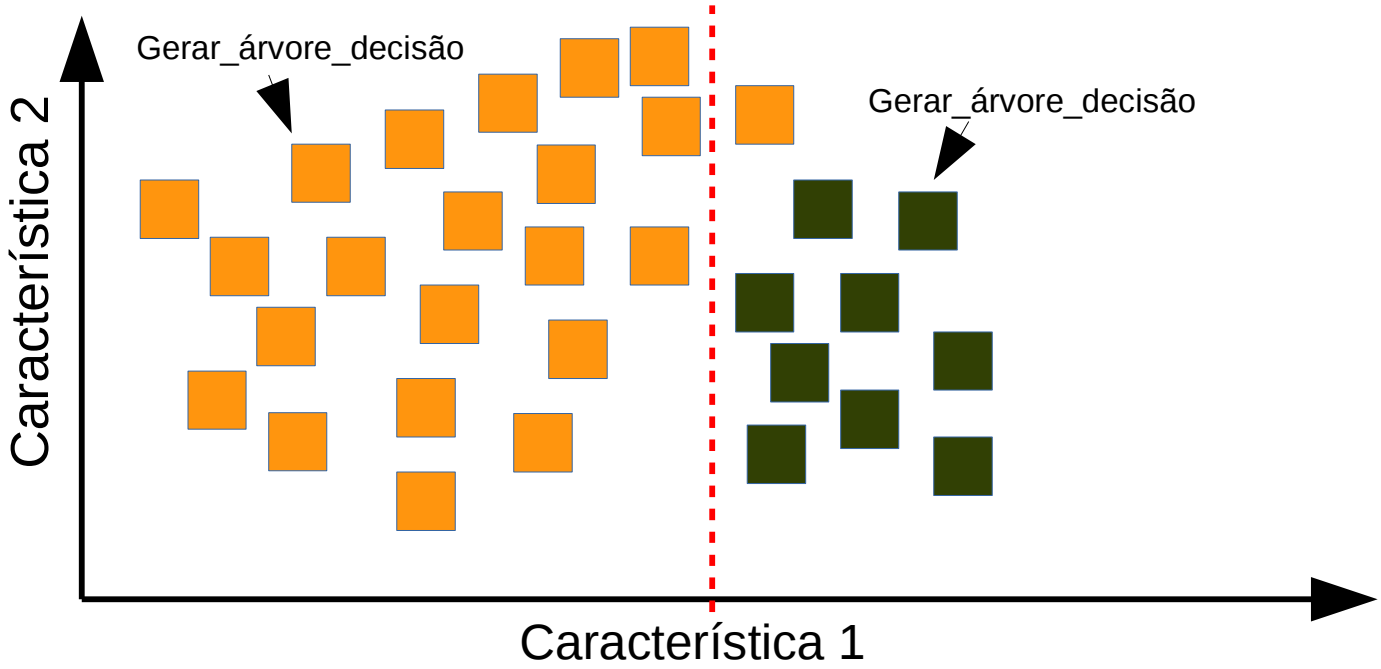
- **Método**
  6. aplique *Método_seleção_atributo*(*D*, *lista_atributos*) para **encontrar** o melhor *critério_divisão*

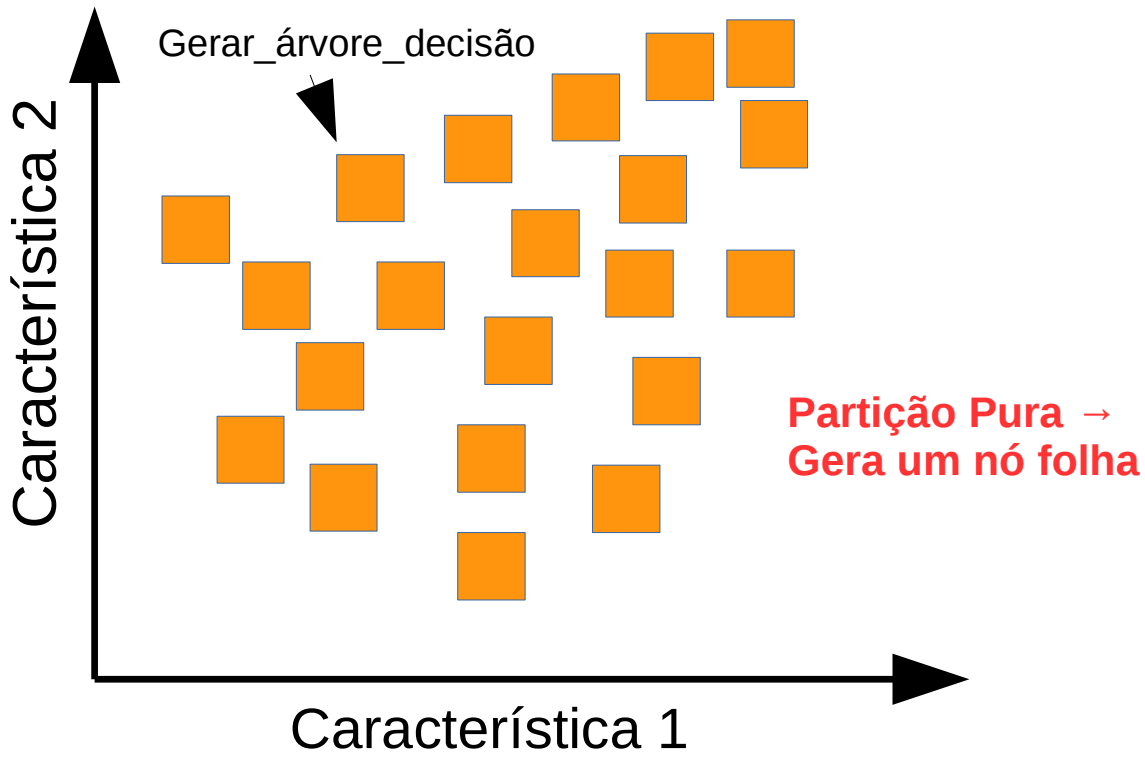


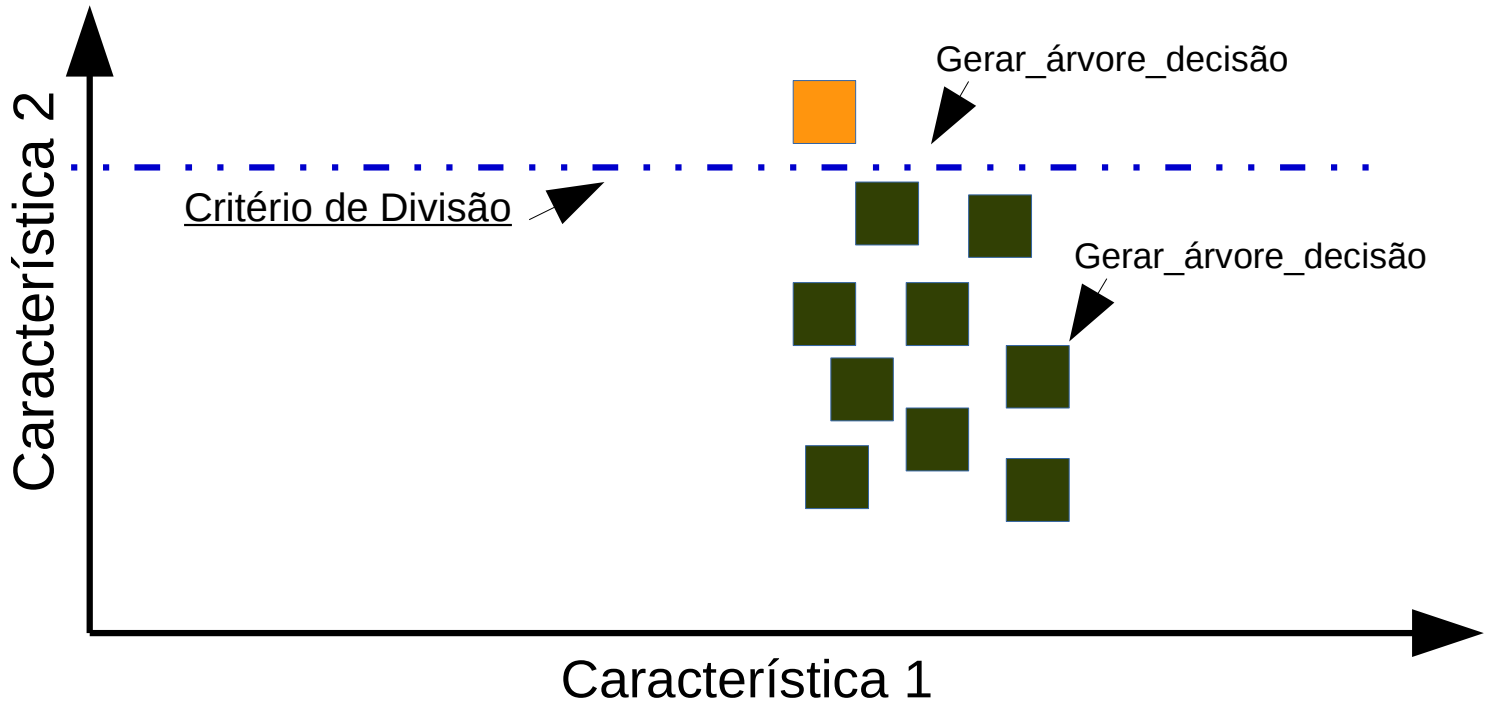
  7. rotule o nó *N* com o atributo de melhor *critério_divisão*



- **Método**
  8. **se** *atributo_divisão* é discreto **e** divisões múltiplas são permitidas **então**
  9. *lista_atributos* $\leftarrow$ *lista_atributos* - *atributo_divisão*
  10. **para cada** valor j do *critério_divisão*
  11. seja D_j os conjunto de exemplos em D que contém j
  12. **se** D_j está vazio **então**
  13. atribua um nó folha rotulado com a classe majoritária em D ao nó N
  14. **senão**
  15. anexe o nó retornado por *Gerar_árvore_decisão*(D_j, *lista_atributos*) ao nó N
  16. **fim para**
  17. retorne N


Gerar_árvore_decisão
Gerar_árvore_decisão
Característica 2
Característica 1


Gerar_árvore_decisão
Característica 2
Característica 1
Partição Pura →
Gera um nó folha


Característica 2
Característica 1
Gerar_árvore_decisão
Critério de Divisão
Gerar_árvore_decisão

Algorithm 4.1 Recursive Top-Down Decision Tree Induction

Input: Data set X, Attribute set A

```
1: tree ← BUILDSUBTREE(X, A, 0);

2: function BUILDSUBTREE(X', A', depth)
3:     if STOP(X', depth) then
4:         return CreateNode(nullRule, majorityClass(X'));
5:     else
6:         rule ← FINDBESTSPLITTINGRULE(X', A');
7:         attr ← attributeUsed(rule);
8:         node ← CreateNode(rule, nullClass);
9:         if rule "exhausts" attr then
10:            remove attr from A';
11:        end if
12:        DataSubsets ← ApplyRule(X', rule);
13:        for X_i ∈ DataSubsets do
14:            child ← BUILDSUBTREE(X_i, A', depth + 1);
15:            node.addChild(child);
16:        end for
17:        return node;
18:    end if
19: end function
```

[Aggarwal, 2015]

Critérios para Seleção de Atributos

- A função `Método_seleção_atributo` determina o critério de divisão
  - Qual atributo será utilizado como teste em um nó da árvore
  - É escolhido o atributo que melhor separa os exemplos considerando as classes
  - Idealmente, após as partições os nós folhas devem ser os mais "puros" possíveis, ou seja, devem haver apenas exemplos de uma única classe
  - No caso de atributos contínuos, cada ponto de divisão deve ser considerado

Critérios para Seleção de Atributos

- **Uma medida de seleção de atributos é uma heurística para selecionar um critério de divisão que melhor separa os dados considerando suas classes $\rightarrow$ geram partições mais "puras"**

- As medidas geram um *ranking* para os atributos, na qual o melhor atributo no *ranking* é escolhido como critério de divisão

- Para entender as medidas de seleção de atributos apresentadas à seguir, serão utilizados as seguintes notações
  - D: conjunto de treinamento
  - $|D|$: quantidade de exemplos do conjunto de treinamento
  - C: conjunto de classes, tal que $C = (C_1, C_2, \ldots, C_m)$
  - $C_{i,D}$: conjunto de tuplas da classe C_i em D
  - $|C_{i,D}|$: quantidade de exemplos da classe C_i em D

Ganho de Informação

- Usado no algoritmo ID3
- O atributo com maior ganho de informação é escolhido como atributo de divisão para um nó da árvore
- Este atributo é o que minimiza a informação necessária para classificar as tuplas nas partições resultantes – maior "pureza"
- A informação necessária para classificar uma tupla em D é dada por (Entropia)

$$Info(D) = -\sum_{i=1}^{m} p_i log_2(p_i) \quad (1)$$

na qual p_i é a probabilidade, diferente de zero, de uma tupla em D pertencer a classe C_i, ou seja $|C_{i,D}|/|D|$

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**

P(No) = 5/14 = 0,35

P(Yes) = 9/14 = 0,65

- Caso $p_i = 0$, assume-se que $p_i log_2(p_i) = 0$
- DICA: para quem não têm calculadora que realize cálculos com log_2

$$log_2 a = \frac{log_{10} a}{log_{10} 2} = \frac{log_{10} a}{0,301}$$

- Suponha que queremos particionas os exemplos em D considerando um atributo A com v valores distintos, ou seja, $A = \{a_1, a_2, \ldots, a_v\}$
- Se A possui valores discretos, cada um dos valores corresponde as saídas de um nó na árvore de decisão
- O atributo A será utilizado para separar D em v partições, $\{D_1, D_2, \ldots, D_v\}$, na qual D_j contém as tuplas em D cujo valor do atributo A é a_j

- Para verificar a informação necessária para classificar um exemplo em D baseado na partição gerada pelo atributo A usa-se

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j) \quad (2)$$

- O termo $\frac{|D_j|}{|D|}$ age como peso para a j-ésima partição
- Quanto menor a informação "ainda" requerida, maior a puridade das partições

Calcule a medida *Info* para o seguinte conjunto de dados:

Renda	**Idade**	**Emprestar**
Alta	Jovem	Sim
Alta	Adulto	Sim
Média	Idoso	Sim
Média	Jovem	Não
Baixa	Adulto	Não
Baixa	Idoso	Não

- O *ganho de informação* é definido como a diferença entre a informação original, isto é, baseada apenas nas proporções das classe, e a informação obtida após o particionamento utilizando o atributo A

$$Gain(A) = Info(D) - Info_A(D) \tag{3}$$

- O atributo A que produz o maior ganho de informação ($Gain(A)$) é escolhido como critério de divisão para o nó N

Razão de Ganho

- Utilizado no algoritmo C4.5

- A medida do ganho de informação é tendenciosa para atributos discretos com muitos valores
  - Se considerarmos o atributo *ID*, cada valor de *ID* irá ocorrer com uma única classe
  - Portanto, cada partição gerada por cada valor de *ID* irá gerar uma partição pura, ou seja, $Info_{ID}(D) = 0$

Razão de Ganho

- A *Razão de Ganho* realiza uma ponderação no ganho de informação utilizando o que se chama de "informação de divisão"

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times log_2(\frac{|D_j|}{|D|}) \qquad (4)$$

que representa a informação potencial gerada por dividir o conjunto de treinamento D em v partições

- A Razão de Ganho é definida por

$$GainRation(A) = \frac{Gain(A)}{SplitInfo_A(D)} \tag{5}$$

- O atributo com maior razão de ganho é selecionado como atributo de divisão
- Normalmente uma constante é adicionada ao valor de $SplitInfo_A(D)$ para evitar $SplitInfo_A(D) = 0$

Gini Index

- Utilizado no algoritmo CART
- O valor da medida Gini Index para a partição D é calculada da seguinte forma

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2 \tag{6}$$

na qual p_i é a probabilidade de uma tupla em D pertencer a classe C_i, dado por $|C_{i,D}|/|D|$

- O valor de Gini Index para um atributo A com 2 valores diferentes é computado da seguinte forma

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \quad (7)$$

- A redução da impureza ocorrida por uma divisão em D utilizando o atributo A é dada por

$$\Delta Gini(A) = Gini(D) - Gini_A(D) \quad (8)$$

Comparação entre as medidas

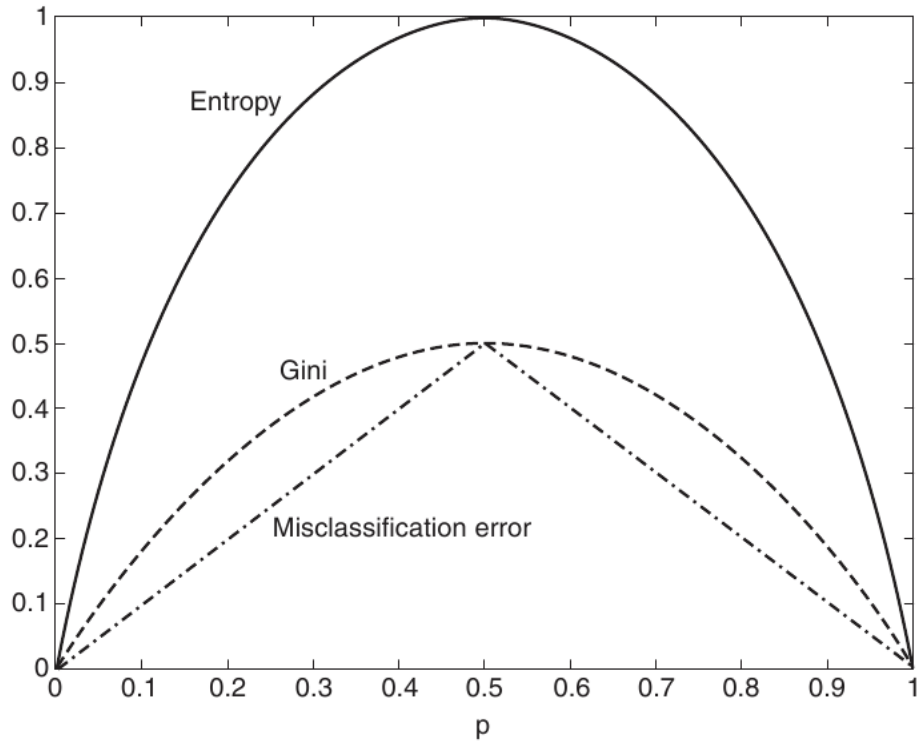


Figura: Comparação entre medidas de impureza para problemas de classificação binários

Nó N_1	**Contagem**
Classe = 0	0
Classe = 1	6

$Gini = 1 - (0/6)^2 - (6/6)^2 = 0$
$Entropia = -(0/6)log_2(0/6) - (6/6)log_2(6/6) = 0$

Nó N_2	**Contagem**
Classe = 0	1
Classe = 1	5

$Gini = 1 - (1/6)^2 - (5/6)^2 = 0,278$
$Entropia = -(1/6)log_2(1/6) - (5/6)log_2(5/6) = 0,650$

Nó N_3	**Contagem**
Classe = 0	3
Classe = 1	3

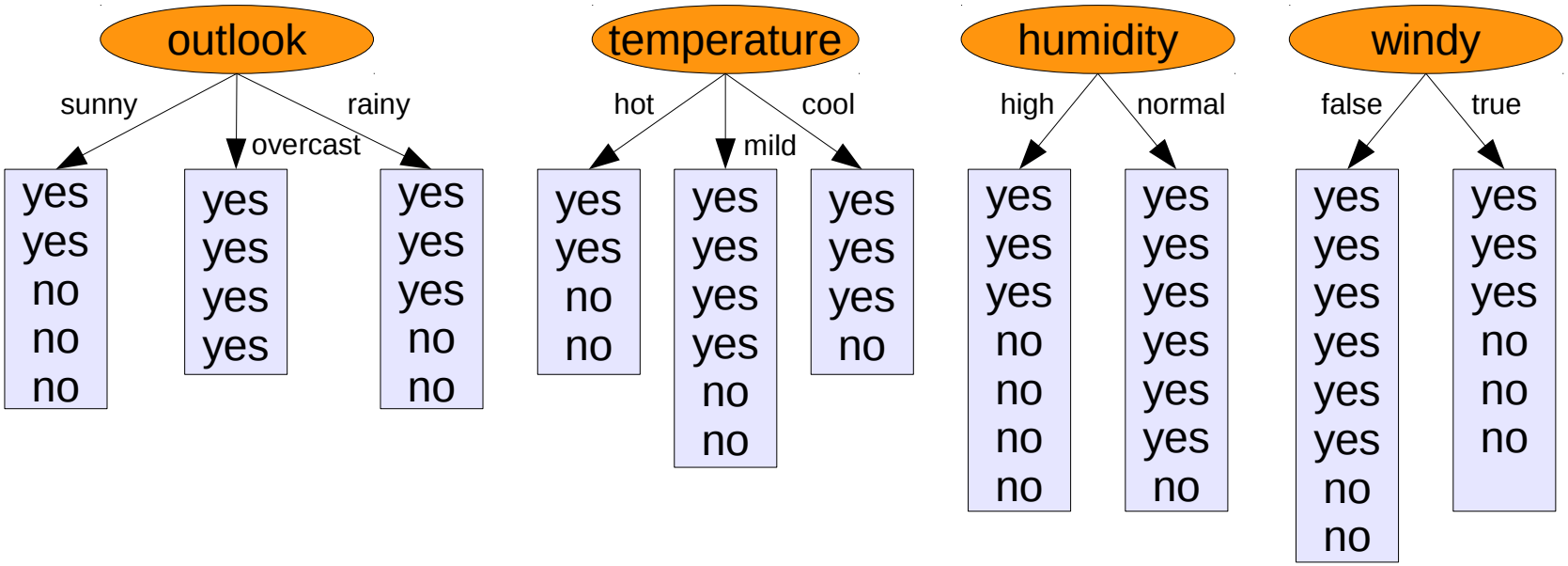
$Gini = 1 - (3/6)^2 - (3/6)^2 = 0,5$
$Entropia = -(3/6)log_2(3/6) - (3/6)log_2(3/6) = 1$

Exemplo

- Vamos considerar o conjunto de dados *Weather*

Tabela: Conjunto de dados *Weather* [Witten and Frank, 2005].

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**


outlook
sunny
overcast
rainy
yes yes no no no
yes yes yes yes
yes yes yes no no
temperature
hot
mild
cool
yes yes no no
yes yes yes yes no no
yes yes yes no
humidity
high
normal
yes yes yes no no no no
yes yes yes yes yes yes no
windy
false
true
yes yes yes yes yes yes no no
yes yes yes no no no

- $info(D) = info([9,5]) = -(9/14)log_2(9/14) - (5/14)log_2(5/14) = 0,92$

- $info_{outlook}(D) = info_{outlook}([2,3],[4,0],[3,2]) =$
  $(5/14)info([2,3]) + (4/14)info([4,0]) + (5/14)info([3,2])$
- $info_{outlook}(D) = (5/14)0,96 + (4/14)0 + (5/14)0,96 = 0,68$
- $gain_{outlook}(D) = info(D) - info_{outlook}(D) = 0,92 - 0,64 = 0,28$

- $info_{temperature}(D) = info_{temperature}([2,2],[4,2],[3,1]) =$
  $(4/14)info([2,2]) + (6/14)info([4,2]) + (4/14)info([3,1])$
- $info_{temperature}(D) = (4/14)1 + (6/14)0,9 + (4/14)0,81 = 0,90$
- $gain_{temperature}(D) = info(D) - info_{temperature}(D) = 0,92 - 0,89 = 0,03$

- $info_{humidity}(D) = info_{humidity}([3,4],[6,1]) = (7/14)info([3,4]) + (7/14)info([6,1])$
- $info_{humidity}(D) = (7/14)0,97 + (7/14)0,57 = 0,77$
- $gain_{humidity}(D) = info(D) - info_{humidity}(D) = 0,92 - 0,77 = 0,15$

- $info_{windy}(D) = info(windy)([6,2],[3,3]) = (8/14)info([6,2]) + (6/14)info[3,3]$
- $info_{windy}(D) = (8/14)0,81 + (6/14)1 = 0,89$
- $gain_{windy}(D) = info(D) = info_{windy} = 0,92 - 0,87 = 0,05$

- Portanto, o atributo *outlook* é selecionado como atributo de divisão
- Como não foram selecionados nós anteriormente, o atributo *outlook* será a raiz da árvore de decisão
- Vale ressaltar que a partição gerada por $outlook = overcast$ gera uma partição pura, portanto, pode-se gerar um nó folha para este ramo

Tabela: Conjunto de dados *Weather* considerando *outlook* = *sunny*

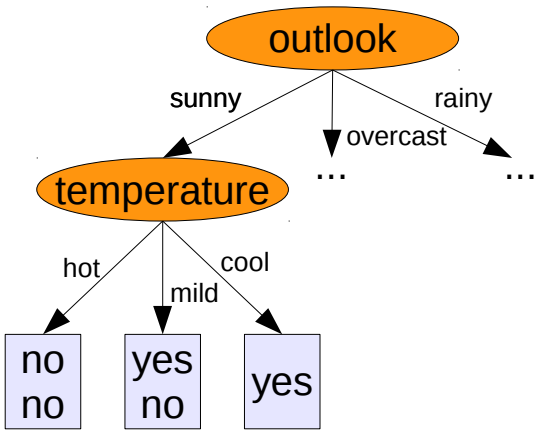
Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	**No**
Sunny	Hot	High	True	**No**
Sunny	Mild	High	False	**No**
Sunny	Cool	Normal	False	**Yes**
Sunny	Mild	Normal	True	**Yes**

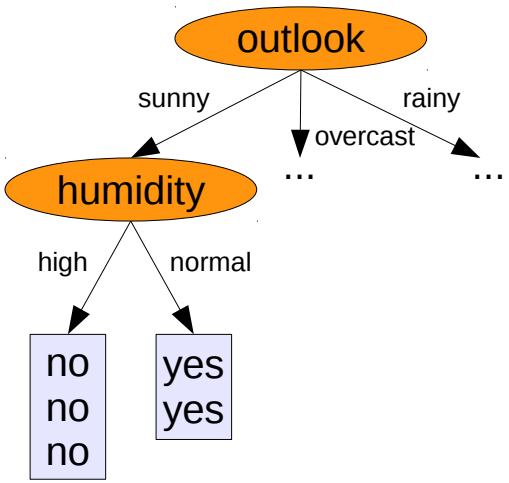
Tabela: Conjunto de dados *Weather* considerando *outlook* = *overcast*.

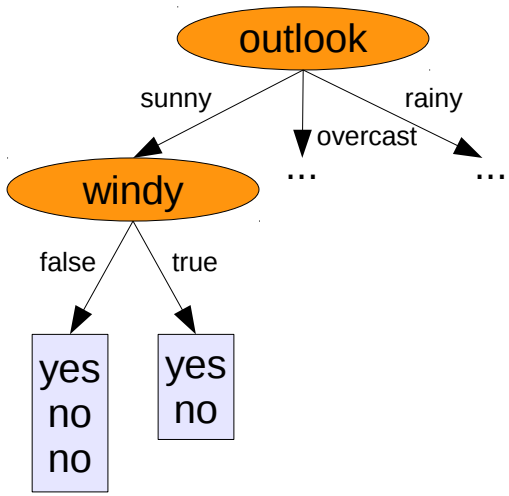
Outlook	Temp	Humidity	Windy	Play
Overcast	Hot	High	False	**Yes**
Overcast	Cool	Normal	True	**Yes**
Overcast	Mild	High	True	**Yes**
Overcast	Hot	Normal	False	**Yes**

Tabela: Conjunto de dados *Weather* considerando *outlook* = *rainy*.

Outlook	Temp	Humidity	Windy	Play
Rainy	Mild	High	False	**Yes**
Rainy	Cool	Normal	False	**Yes**
Rainy	Cool	Normal	True	**No**
Rainy	Mild	Normal	False	**Yes**
Rainy	Mild	High	True	**No**


outlook
sunny
overcast
rainy
...
...
temperature
hot
mild
cool
no
no
yes
no
yes

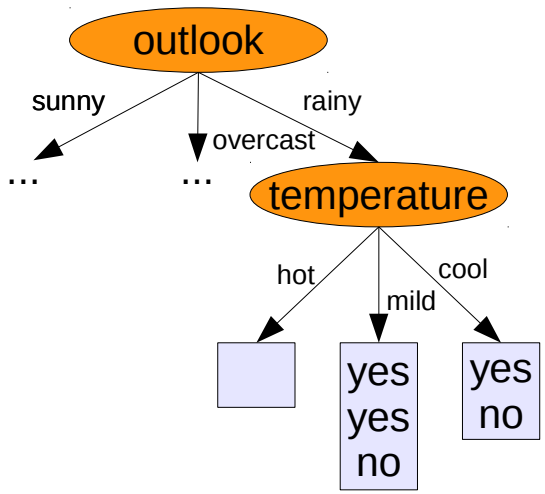

outlook
sunny
overcast
rainy
...
...
humidity
high
normal
no
no
no
yes
yes

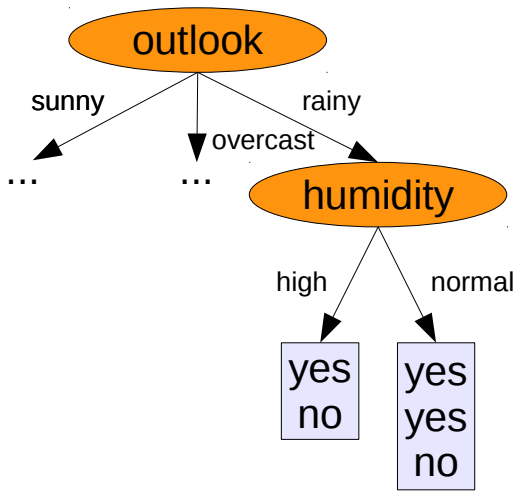

outlook
sunny
overcast
rainy
...
...
windy
false
true
yes
no
no
yes
no

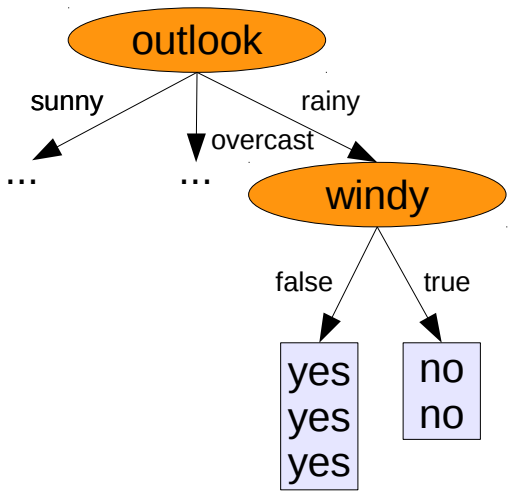
- $info(D_{outlook=sunny}) = info([2,3]) = 0,94$
- $info_{temperature}(D_{outlook=sunny}) = info_{temperature}([0,2],[1,1],[1,0]) =$ $(2/5)info([0,2]) + (2/5)info([1,1]) + (1/5)info(1,0)$
- $info_{temperature}(D_{outlook=sunny}) = (2/5)0 + (2/5)1 + (1/5)0 = 0,4$
- $gain_{temperature}(D_{outlook=sunny}) = info(D_{outlook=sunny}) - gain_{temperature}(D_{outlook=sunny}) = 0,54$
- $info_{humidity}(D_{outlook=sunny}) = info_{humidity}([0,3],[2,0])$
- $info_{humidity}(D_{outlook=sunny}) = (3/5)info([0,3]) + (2/5)info([2,0]) = 0$
- $gain_{humidity}(D_{outlook=sunny}) = info(D_{outlook=sunny}) - gain_{humidity}(D_{outlook=sunny}) = 0,94$

...

- O atributo *humidity* apresenta o maior ganho de informação dado $outlook = sunny$
- Vale ressaltar que para $outlook = sunny$, as divisões do atributo *humidity* geram partições puras, portanto, criam-se nós folhas para estes ramos

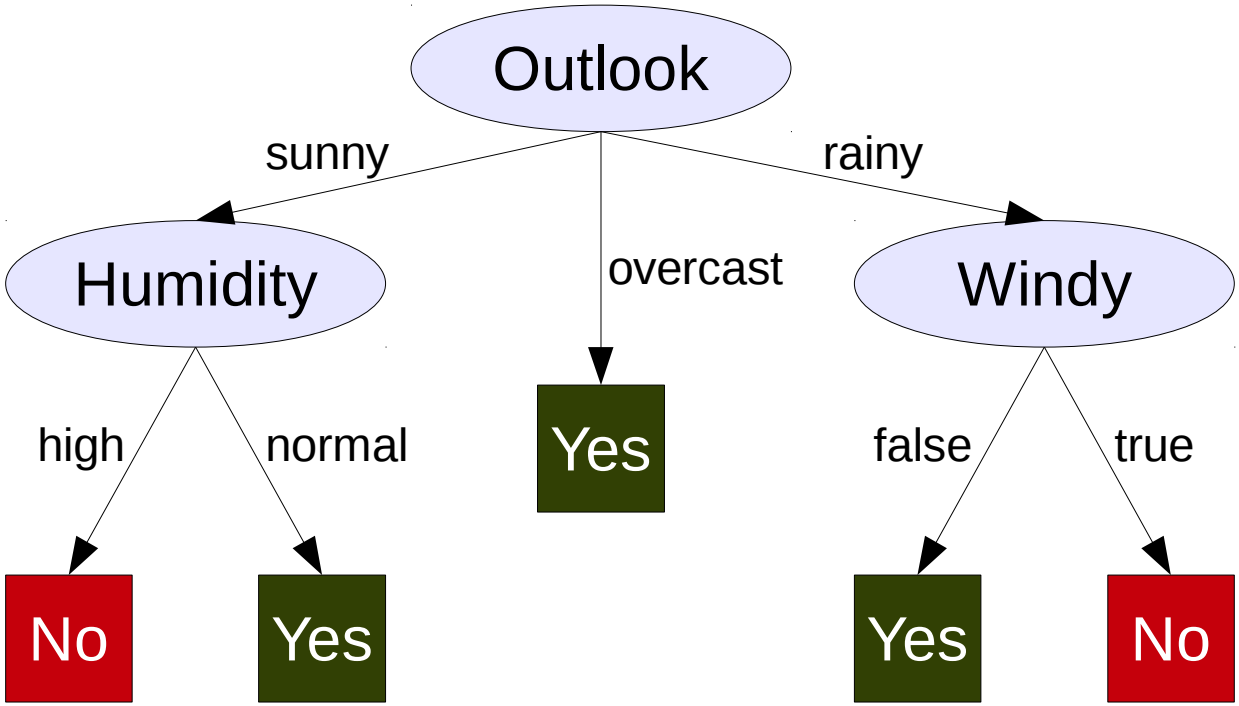

outlook
sunny
overcast
rainy
...
...
temperature
hot
mild
cool
yes
yes
no
yes
no


outlook
sunny
overcast
rainy
...
...
humidity
high
normal
yes
no
yes
yes
no


outlook
sunny
overcast
rainy
...
...
windy
false
true
yes
yes
yes
no
no

- O atributo *windy* apresenta o maior ganho de informação dado $outlook = rainy$
- Vale ressaltar que para $outlook = rainy$, as divisões do atributo *windy* geram partições puras, portanto, criam-se nós folhas para estes ramos

Árvore de decisão final



Exemplo do cálculo da entropia para atributos númericos

Tabela: Conjunto de dados *Weather* com alguns atributos numéricos

Outlook	Temperature	Humidity	Windy	Play
sunny	85	85	false	no
sunny	80	90	true	no
overcast	83	86	false	yes
rainy	70	96	false	yes
rainy	68	80	false	yes
rainy	65	70	true	no
overcast	64	65	true	yes
sunny	72	95	false	no
sunny	69	70	false	yes
rainy	75	80	false	yes
sunny	75	70	true	yes
overcast	72	90	true	yes
overcast	81	75	false	yes
rainy	71	91	true	no

- Calculando a entropia para o atributo *temperature*
  - Ordenação dos valores com suas respectivas classes
  - Considerando uma divisão binária, com 12 valores temos 11 possíveis divisões
  - Se considerarmos que não devemos separar os itens da mesma classe, temos 8 possíveis divisões
  - Para cada divisão, devemos calcular a entropia
  - O ponto de divisão pode ser a média entre dois números consecutivos

Tabela: Valores do atributo *temperature* ordenados

64	65	68	69	70	71	72	75	80	81	83	85
yes	no	yes	yes	yes	no	no yes	yes yes	no	yes	yes	no

- Vamos considerar o ponto de divisão 71,5 (média entre os valores 71 e 72)
- Neste caso os valores dos ramos da árvore seriam $\leq 71,5$ e $> 71,5$
- A entropia desta divisão é dada por $info([4,2],[5,3])$, uma vez que para valores $\leq 71,5$ temos 4 ocorrências da classe `yes` e 2 ocorrências da classe `no`, e para valores $> 71,5$ temos 5 ocorrências da classe `yes` e 3 ocorrências da classe `no`

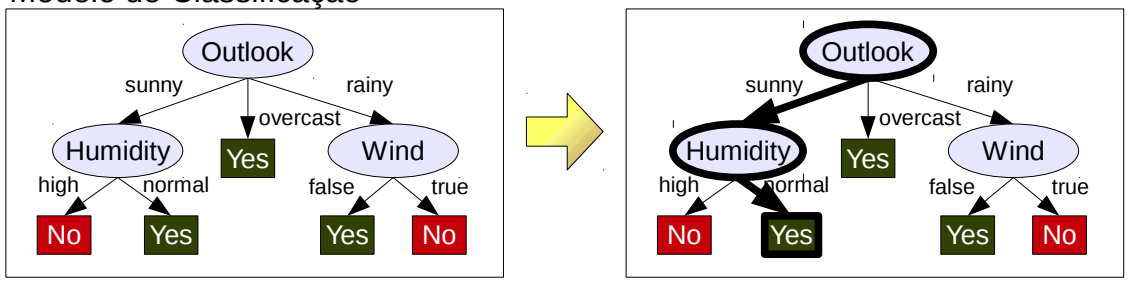
$$info([4,2],[5,3]) = (6/14)info([4,2]) + (8/14)info([5,3]) = 0,93$$

Classificação

- Dado uma instância X, na qual a classe é desconhecida, os valores dos atributos são testados nos nós da árvore de decisão
- Um caminho é traçado da raiz à um nó folha, que representa a classe predita pela árvore de decisão

Outlook	Temp	Humidity	Windy	Classe
Sunny	Mild	Normal	True	????

Modelo de Classificação



Sunny	Mild	Normal	True	Yes

Poda da Árvore

- Quando uma árvore de decisão é construída, muitos ramos podem refletir anomalias dos dados de treinamento devido a ruídos ou *outliers*
- Podem causar *overfitting* nestes tipos de dados
  - Super-ajuste aos dados de treinamento
  - Podem causar baixa performance preditiva
- Solução: **podar a árvore**

Poda da Árvore

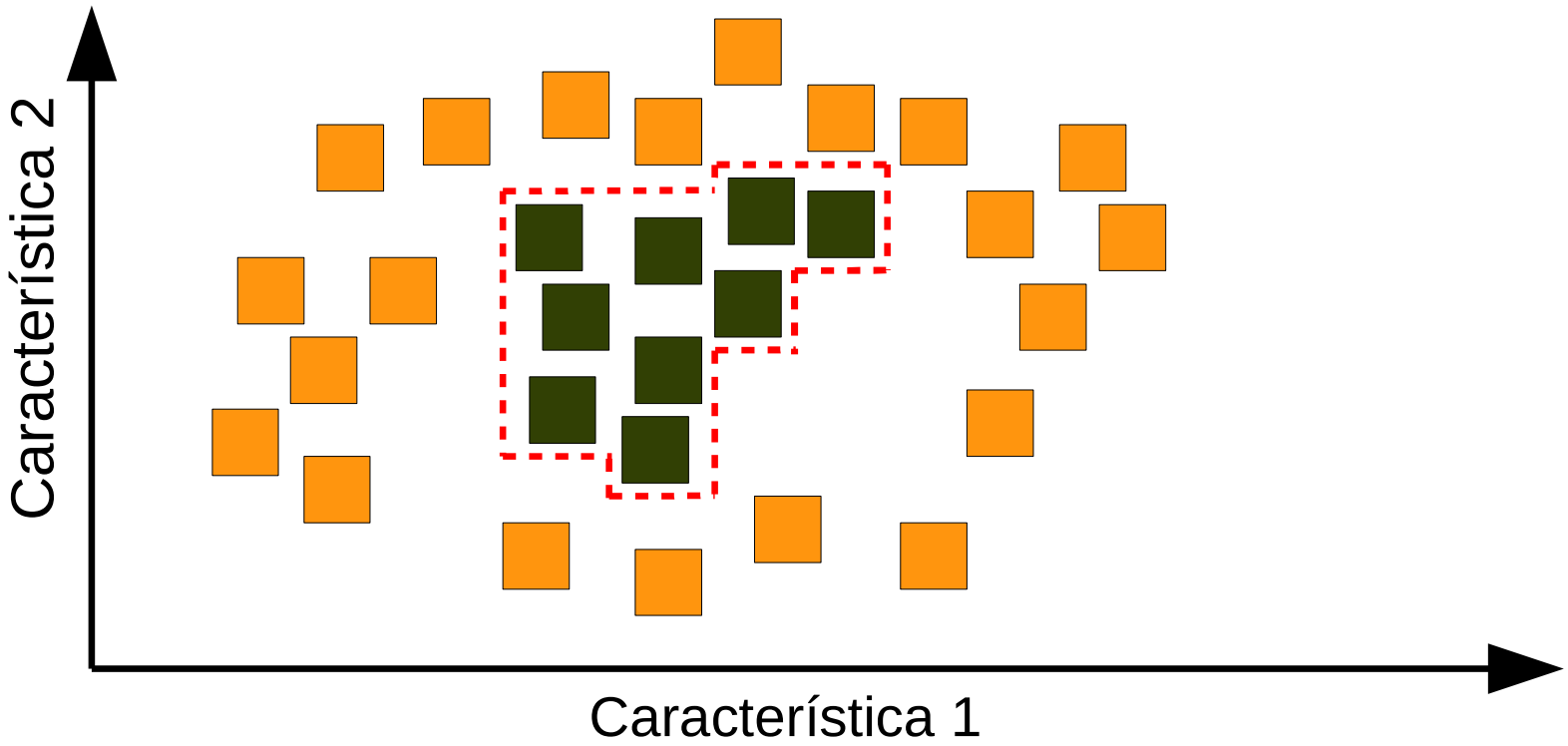


Figura: Exemplo de um modelo de classificação super-ajustado aos dados de treinamento

Poda da Árvore

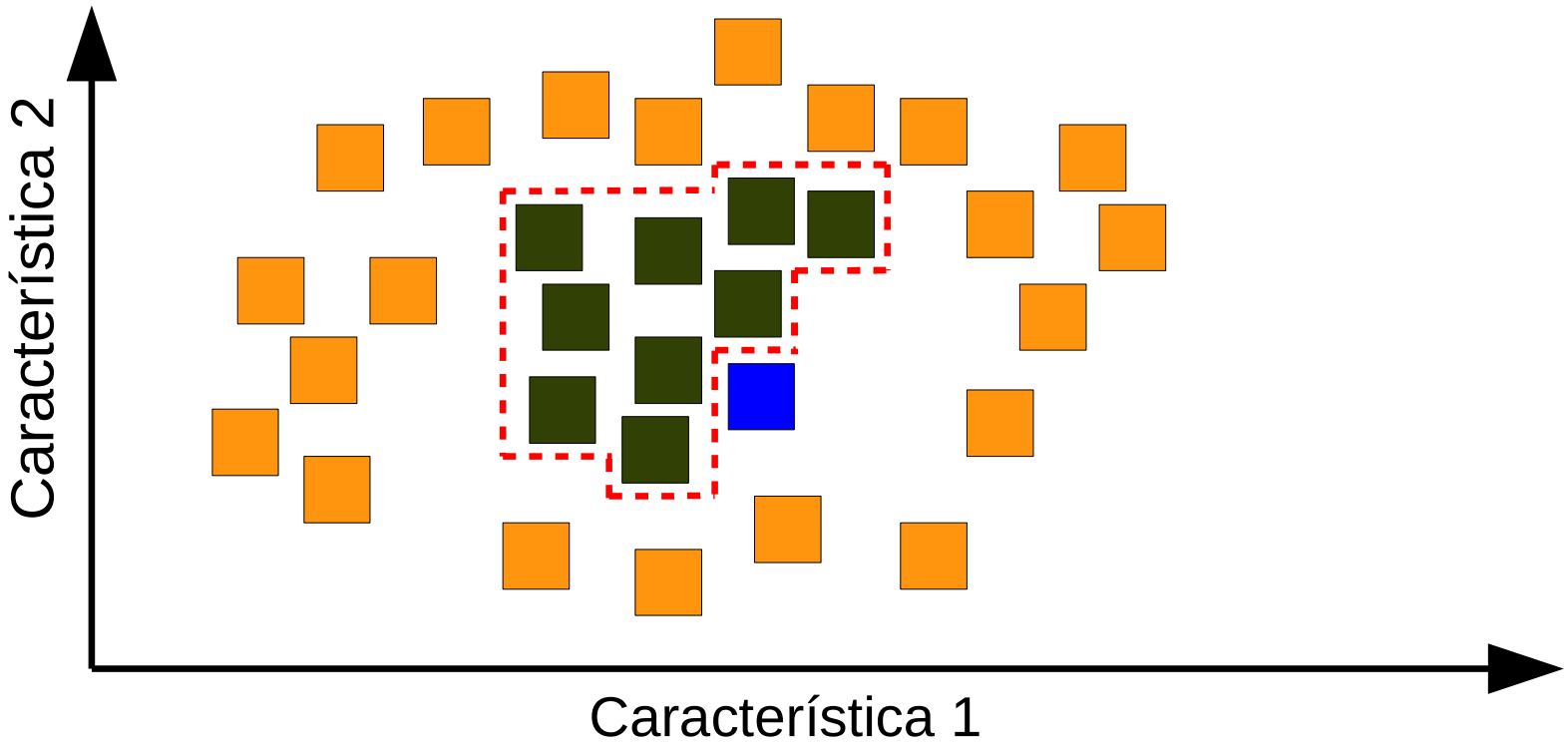


Figura: Exemplo de uma classificação incorreta devido ao super-ajuste do modelo de classificação aos dados de treinamento

Poda da Árvore

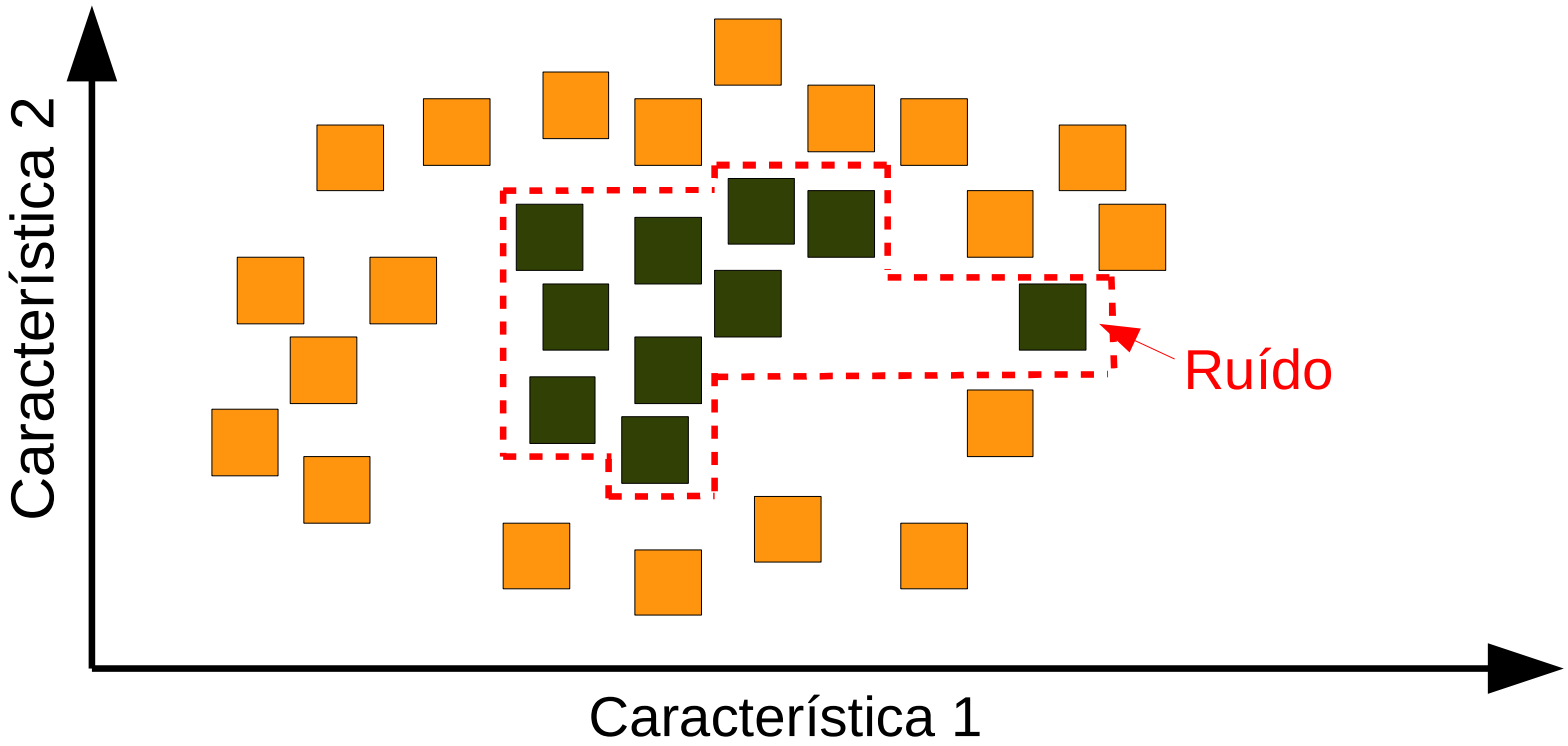


Figura: Exemplo de um modelo de classificação considerando ruídos

Poda da Árvore

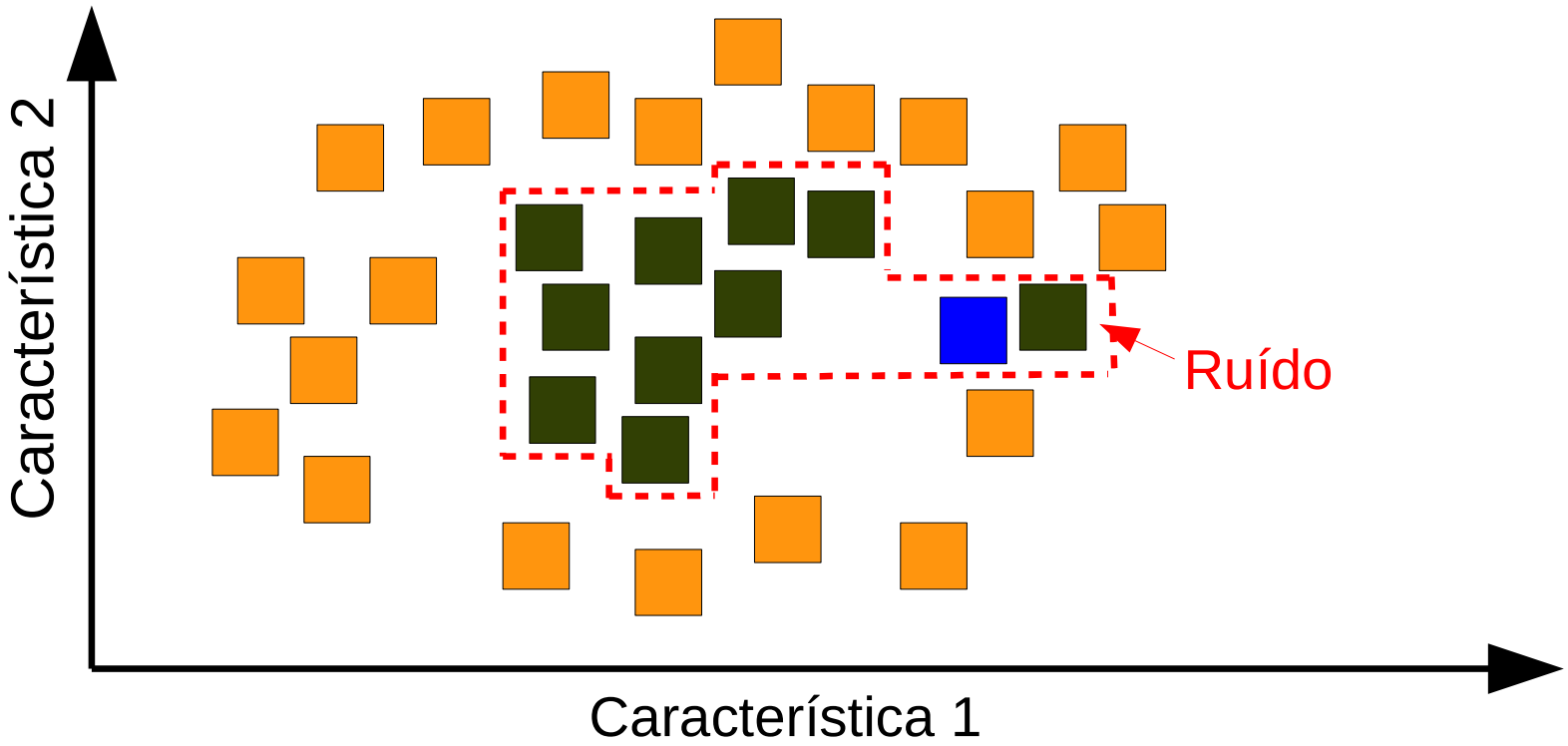


Figura: Exemplo de uma classificação incorreta devido ao ajuste do modelo de classificação a um ruído

Poda da Árvore

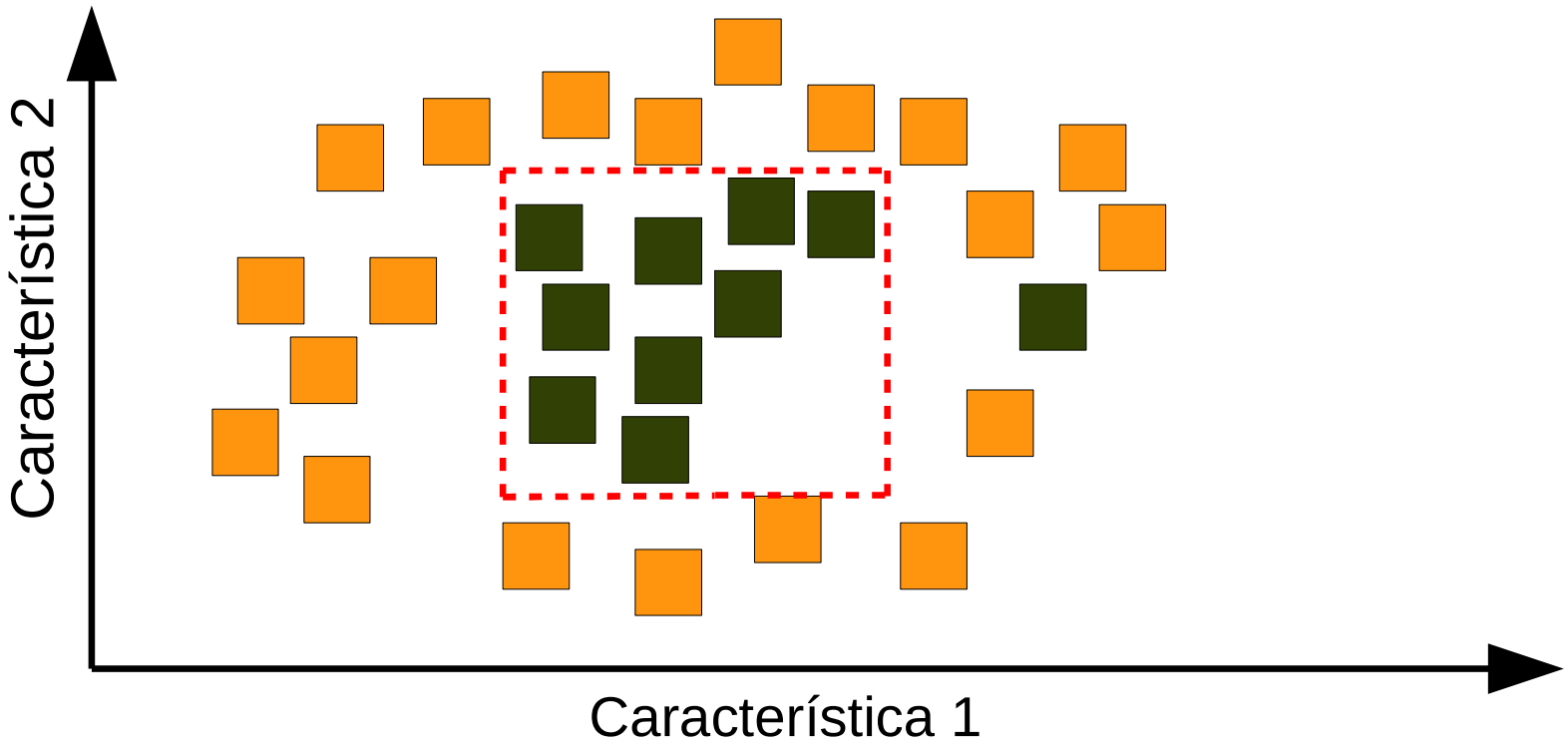


Figura: Exemplo de um modelo de classificação após a poda

Poda da Árvore

- Tipicamente são utilizadas medidas estatísticas para remover ramos pouco confiáveis
- As árvores podadas tendem a ser menores e menos complexas, portanto mais fáceis de serem compreendidas
- Duas abordagens:
  - Pré-poda
  - Pós-poda
- A pós-poda possui um custo computacional maior, mas geralmente produz melhores resultados

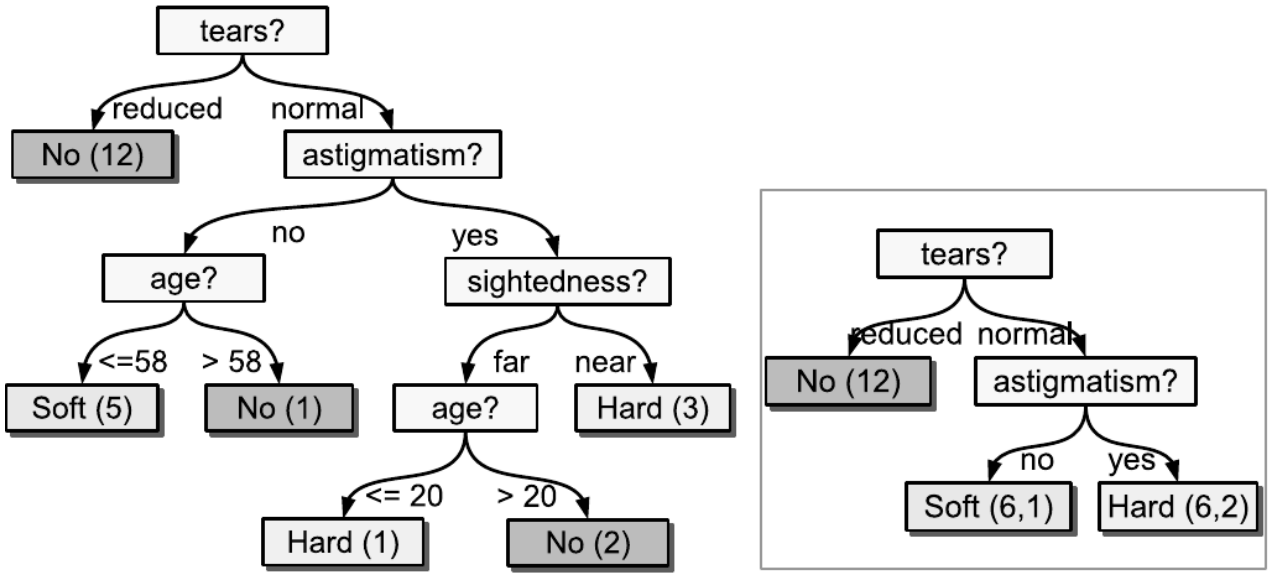
Poda da Árvore

- Pré-poda
  - A divisão de um subconjuntos de exemplos não é realizada, e o nó folha representará a classe mais frequente do conjunto de exemplos representado pelo nó
  - Medidas como Ganho de Informação, Gino Índex, etc., podem ser utilizadas para medir a qualidade de uma partição
  - Se a partição gerar um valor de medida abaixo de um limiar, então não é realizada nenhuma partição
    - Altos valores de limiar podem gerar árvores muito simplificadas
    - Baixos valores de limiar podem resultar em pouco/nenhuma simplificação na árvore

Poda da Árvore

- Pós-poda
  - Remove subárvores de uma árvore induzida completa
  - Uma subárvore é removida removendo seus ramos e os substituindo por nós folha
  - O nó folha é rotulado com a classe mais frequente dos exemplos cobertos pela subárvore
  - O algoritmo utiliza um método chamado de "poda pessimista"
    - Utiliza a taxa de erro do conjunto de treinamento para avaliar a poda das subárvores
    - A taxa de erro obtido no conjunto de treinamento é otimista, por isso, uma penalidade é adicionada
    - A taxa de erro estiver abaixo de um limiar ou intervalo de confiança, a poda é realizada

Poda da Árvore



(a) CART algorithm

(b) Pruned version

Figura: Exemplo de um modelo de classificação após a poda [Aggarwal, 2015]

Material Complementar

- Árvores de Decisão

  http://www.cin.ufpe.br/~if684/EC/aulas/Aula-arvores-decisao-SI.pdf

- Árvore de decisão. Exemplo completo.
  https://www.youtube.com/watch?v=_ICNdRrl68k

- Entropia e Aprendizagem de Árvores de Decisão - C4.5

  https://www.youtube.com/watch?v=qPbimX0R5vg

Material Complementar

- Decision Tree Algorithm — Explained

  https://towardsdatascience.com/decision-tree-algorithm-explained-83beb6e78ef4

- The Simple Math behind 3 Decision Tree Splitting criterions

  https://towardsdatascience.com/

  the-simple-math-behind-3-decision-tree-splitting-criterions-85d4de2a75fe

- Decisin Tress Explained

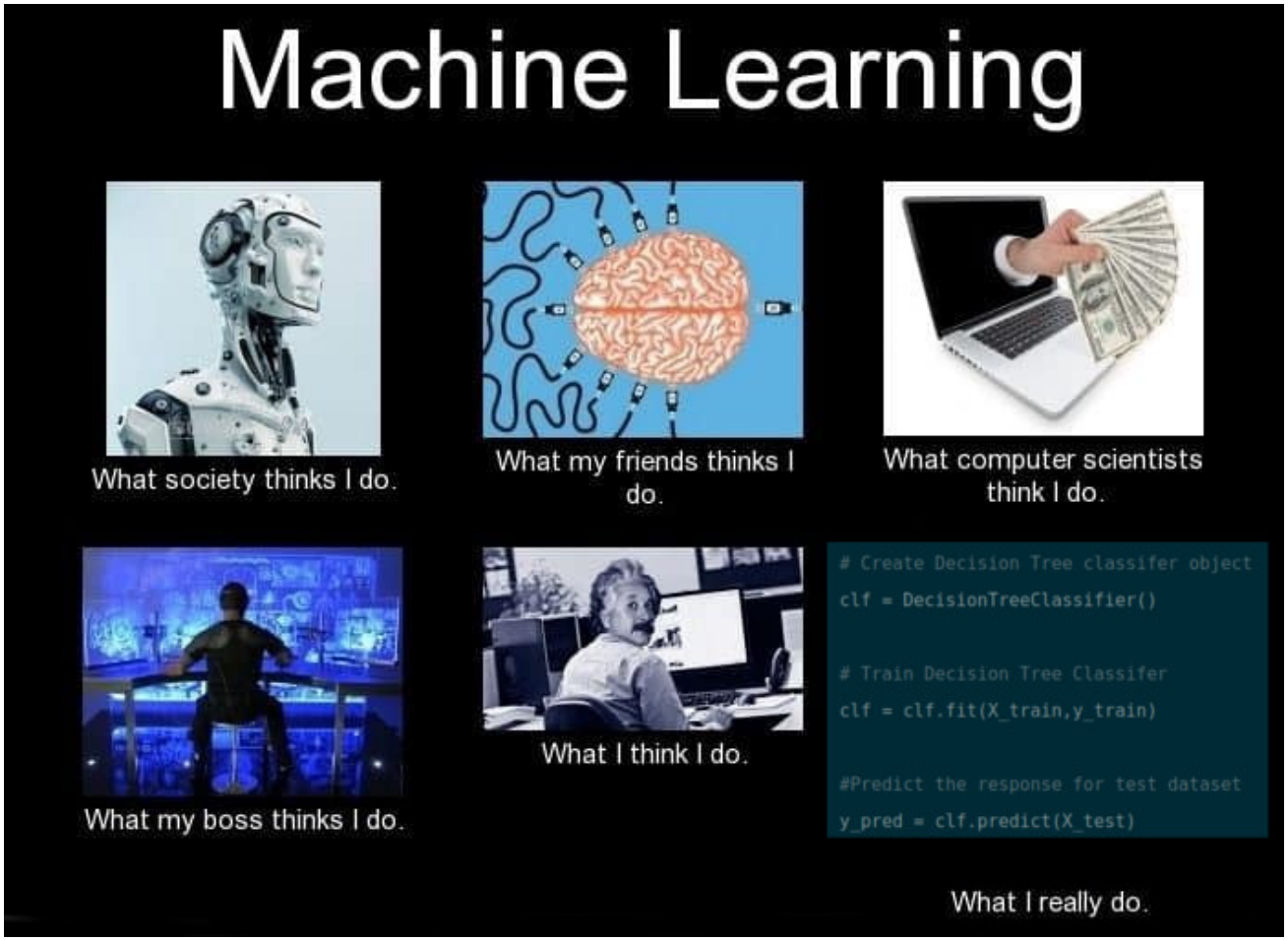
  https://towardsdatascience.com/scikit-learn-decision-trees-explained-803f3812290d

- How to Visualize a Decision Tree from a Random Forest in Python using Scikit-Learn

  https://towardsdatascience.com/

  how-to-visualize-a-decision-tree-from-a-random-forest-in-python-using-scikit-learn-38ad2d75f21c

Imagem do Dia



Inteligência Artificial
http://lives.ufms.br/moodle/

Rafael Geraldeli Rossi
rafael.g.rossi@ufms.br

Slides baseados em [Han et al., 2011], [Tan et al., 2005], [Witten and Frank, 2005] e [Aggarwal, 2015]

Referências Bibliográficas I


Aggarwal, C. (2015).
Data Classification: Algorithms and Applications.
Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. CRC Press.


Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984).
Classification and Regression Trees.
Wadsworth.

Referências Bibliográficas II


Han, J., Kamber, M., and Pei, J. (2011).
Data Mining: Concepts and Techniques.
The Morgan Kaufmann Series in Data Management Systems.
Elsevier.


Quinlan, J. R. (1986).
Induction of decision trees.
Mach. Learn., 1(1):81–106.


Quinlan, J. R. (1993).
C4.5: programs for machine learning.
Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Referências Bibliográficas III


Tan, P.-N., Steinbach, M., and Kumar, V. (2005).
Introduction to Data Mining.
Addison-Wesley.


Witten, I. H. and Frank, E. (2005).
Data Mining: Practical machine learning tools and techniques.
Morgan Kaufmann, 2 edition.