

Aprendizagem de Máquina

Alessandro L. Koerich

Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal do Paraná (UFPR)

ÁRVORES DE DECISÃO

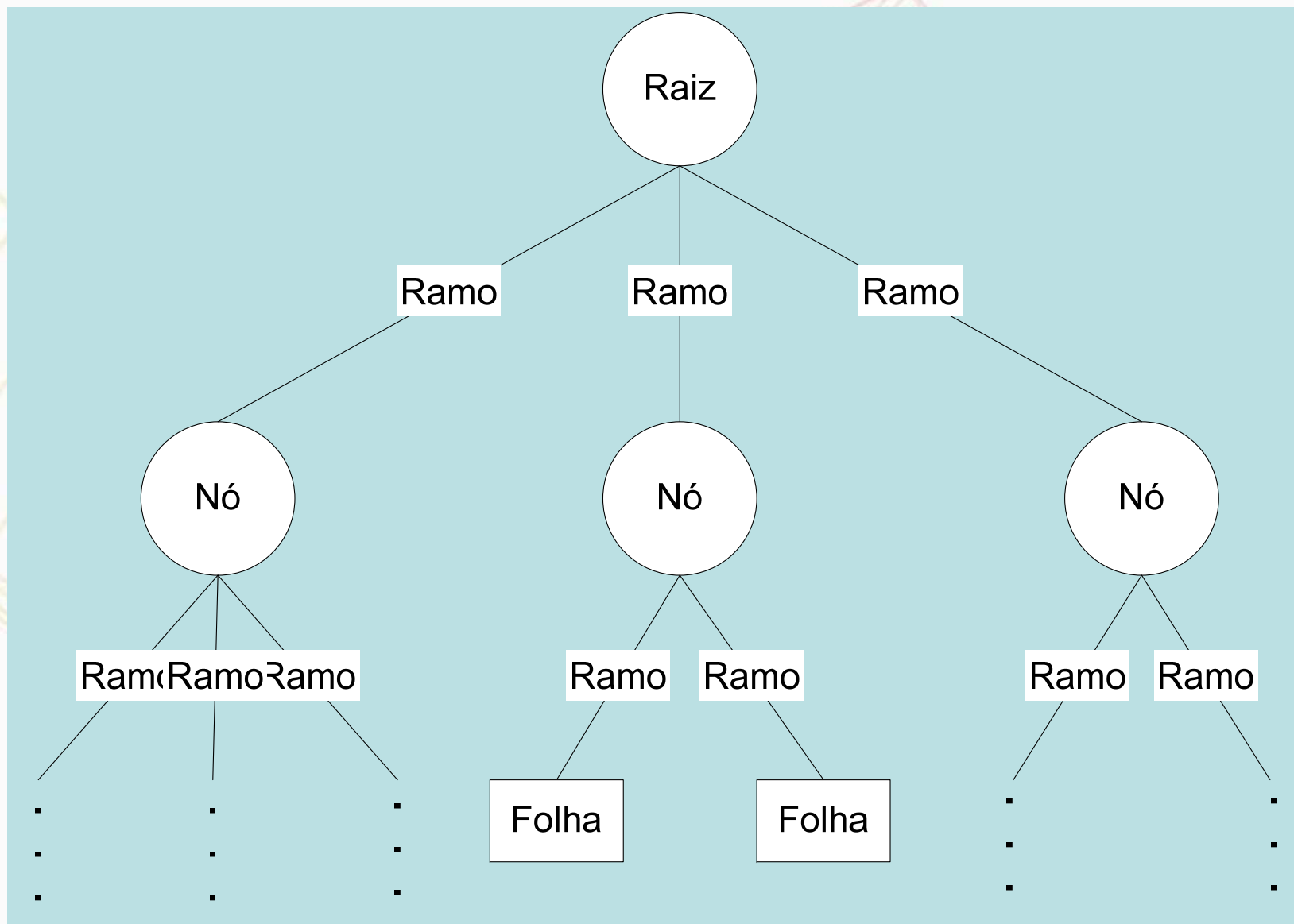
Plano de Aula

- Introdução
- Representação de Árvores de Decisão
- Algoritmo de Aprendizagem *ID3*
- Entropia e Ganho de Informação
- Exemplos
- Aspectos na Aprendizagem de Árvores de Decisão
- Resumo

Referências

- Duda R., Hart P., Stork D. *Pattern Classification* 2ed. Wiley Interscience, 2002. Capítulo 8.
- Mitchell T. *Machine Learning*. WCB McGraw–Hill, 1997. Capítulo 3.

Árvore



Introdução

- Aprendizagem de árvores de decisão é um método prático de aprendizagem indutiva.
- É um método para a aproximação de **funções de valor discreto**
- É um método relativamente robusto a ruídos nos dados
- Utiliza um *bias* indutivo: preferência por árvores menores.

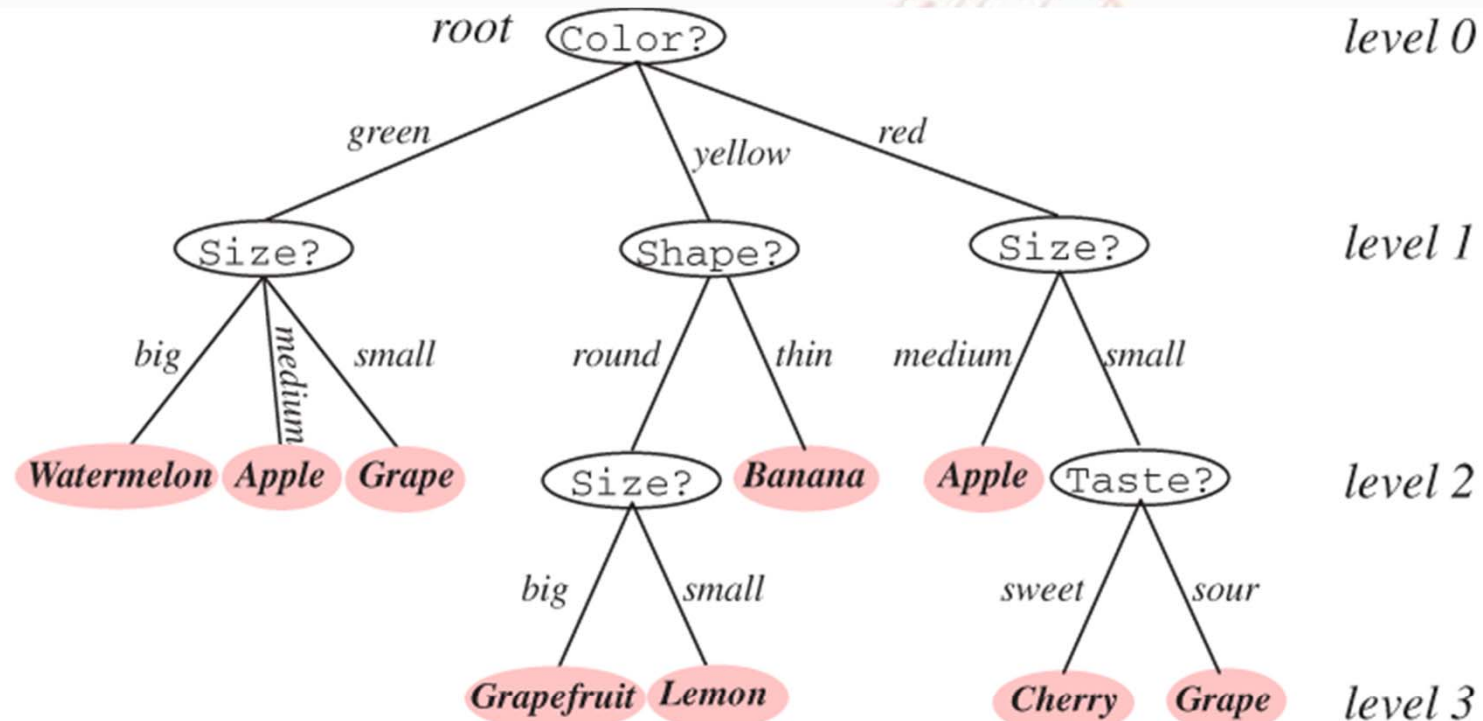
Introdução

- A função aprendida é representada por uma **Árvore de Decisão**.
- Árvores de decisão também podem ser representadas como conjuntos de regras SE-ENTÃO (IF-THEN).
- É um dos métodos de aprendizagem mais conhecidos
- Aplicações: Diagnóstico médico, análise de risco de crédito, mineração de dados, etc.

Representação de Árvores de Decisão

- Árvores de decisão classificam instâncias ordenando-as árvore abaixo, a partir da raiz até alguma folha.
- Cada **nó** da árvore especifica o **teste** de algum atributo da instância
- Cada **ramo** partindo de um nó corresponde a um dos **valores possíveis dos atributos**.

Exemplo

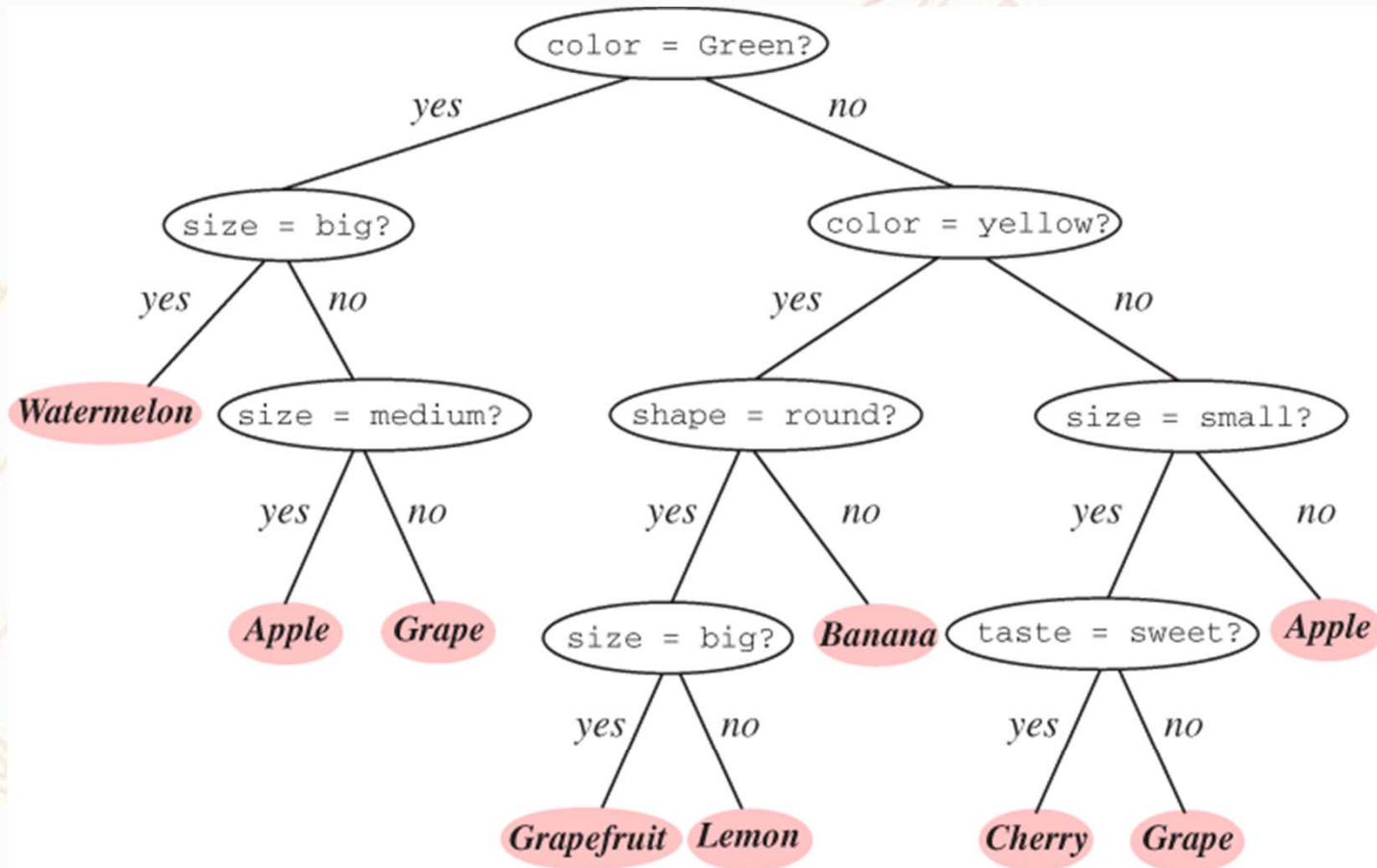


Classification in a basic decision tree proceeds from top to bottom. The questions asked at each node concern a particular property of the pattern, and the downward links correspond to the possible values. Successive nodes are visited until a terminal or leaf node is reached, where the category label is read. Note that the same question, **Size?**, appears in different places in the tree and that different questions can have different numbers of branches. Moreover, different leaf nodes, shown in pink, can be labeled by the same category (e.g., **Apple**).

Representação de Árvores de Decisão

- Uma instância é classificada inicialmente pelo **nó raiz**, testando o atributo especificado por este nó.
- Em seguida, movendo-se através do ramo correspondendo ao valor do atributo no exemplo dado.
- Este processo é repetido para a sub-árvore originada no novo nó.

Exemplo



A tree with arbitrary branching factor at different nodes can always be represented by a functionally equivalent binary tree—that is, one having branching factor $B = 2$ throughout, as shown here. By convention the “yes” branch is on the left, the “no” branch on the right. This binary tree contains the same information and implements the same classification as that in Fig. 8.1.

Quando Considerar Árvores de Decisão

- Instâncias descritas por **pares atributo–valor**.
Instâncias são descritas por um conjunto fixo de atributos (e.g. Temperature) e seus valores (e.g. Hot)
- Função alvo tem **valores discretos de saída**.
Classificação booleana (Yes ou No) para cada exemplo ou mais de duas possibilidades (*Uva, Laranja, Melancia*).

Quando Considerar Árvores de Decisão

- Hipóteses disjuntivas podem ser necessárias. Árvores de decisão representam naturalmente expressões disjuntivas.
- Dados de treinamento podem conter erros e valores de atributos faltantes.

Algoritmo Básico para Aprendizagem de Árvores de Decisão

- **Algoritmo base:** ID3 e seu sucessor, o C4.5.
- O algoritmo ID3 “aprende” árvores de decisão construindo-as de cima para baixo (*top-down*), começando com a questão:

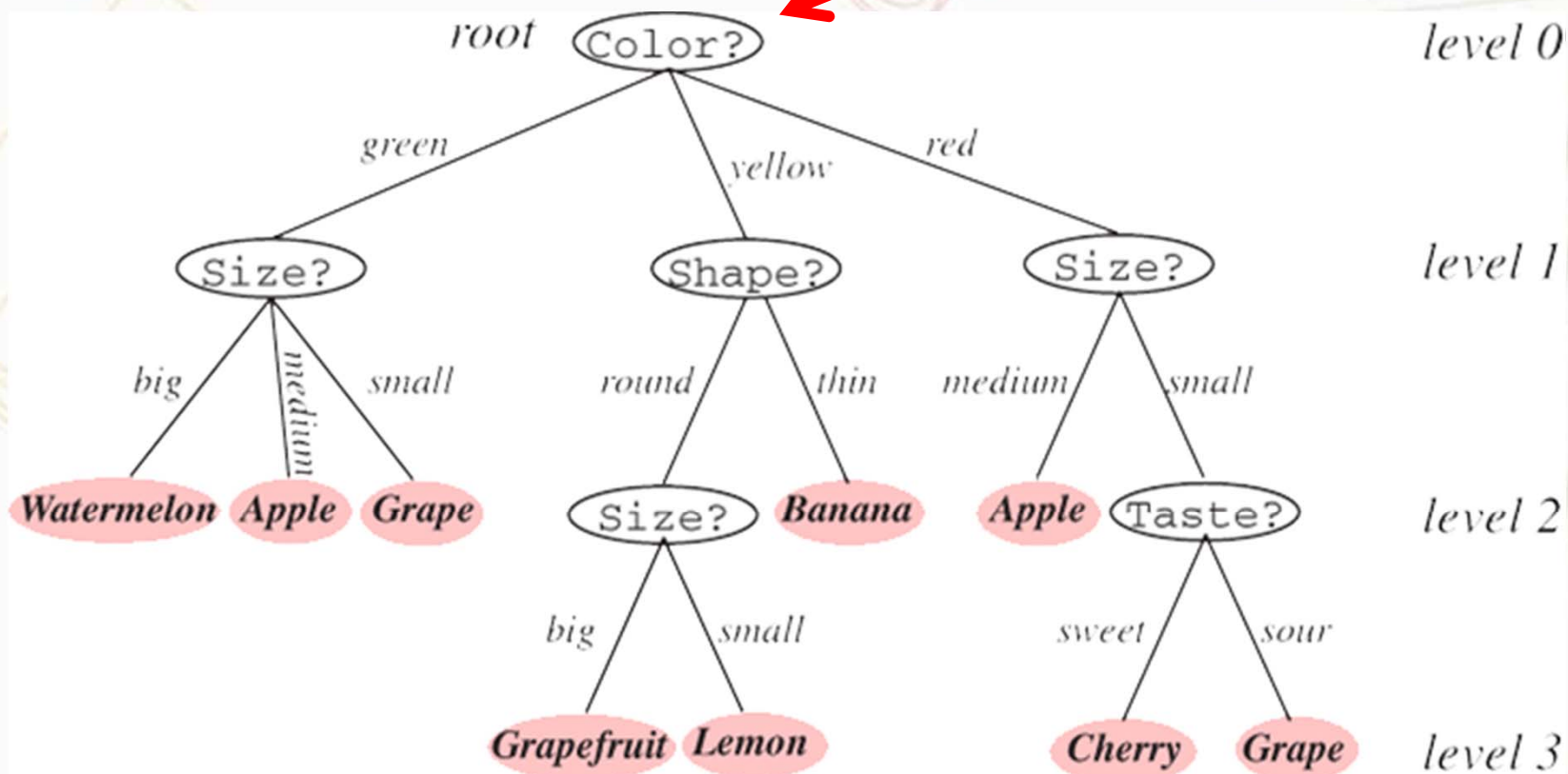
“Qual atributo deve ser testado na raiz da árvore?”

- Para responder esta questão, cada atributo da instância é avaliado usando um teste estatístico para determinar quão bem ele sozinho classifica os exemplos de treinamento.

Exemplo

Color? Size? Taste? Shape?

Melhor Atributo !!!



Algoritmo Básico para Aprendizagem de Árvores de Decisão

- O **melhor atributo** é selecionado e usado como teste na **raiz** da árvore.
- Um descendente do nó raiz é então criado para cada valor possível deste atributo e os exemplos de treinamento são ordenados para o nó descendente apropriado.

Algoritmo Básico para Aprendizagem de Árvores de Decisão

- O processo é repetido usando exemplos de treinamento associados com cada nó descendente para selecionar o melhor atributo para testar naquele ponto da árvore.
- Busca gulosa (greedy) por uma árvore de decisão aceitável, na qual o algoritmo nunca recua para reconsiderar escolhas prévias.

Algoritmo Básico para Aprendizagem de Árvores de Decisão

ID3(*Examples*, *Target_attribute*, *Attributes*)

Examples are the training examples. *Target_attribute* is the attribute whose value is to be predicted by the tree. *Attributes* is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given *Examples*.

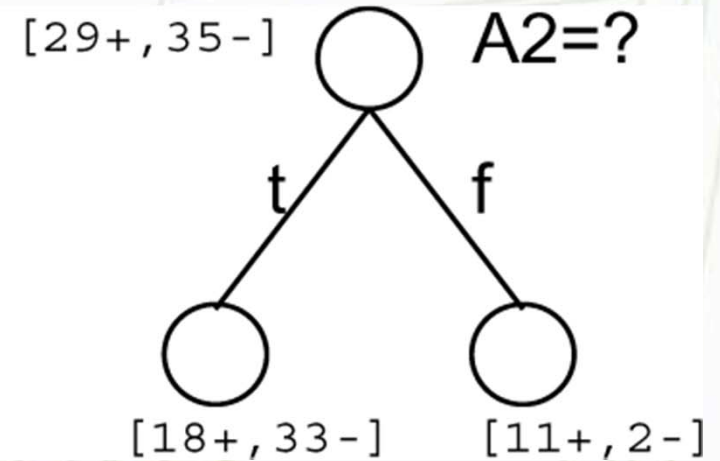
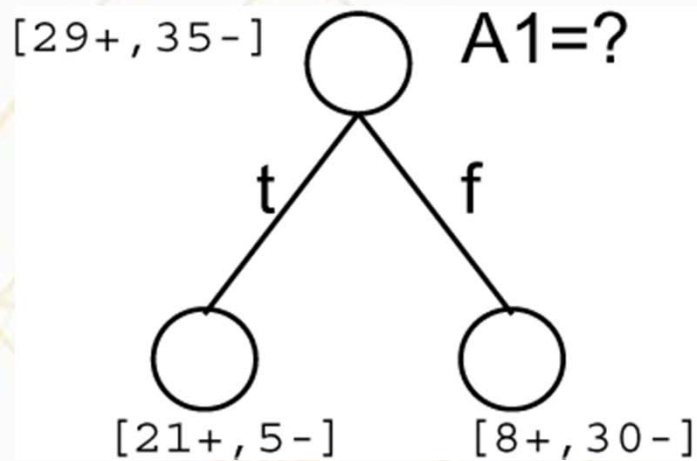
- Create a *Root* node for the tree
- If all *Examples* are positive, Return the single-node tree *Root*, with label = +
- If all *Examples* are negative, Return the single-node tree *Root*, with label = -
- If *Attributes* is empty, Return the single-node tree *Root*, with label = most common value of *Target_attribute* in *Examples*
- Otherwise Begin
 - $A \leftarrow$ the attribute from *Attributes* that best* classifies *Examples*
 - The decision attribute for *Root* $\leftarrow A$
 - For each possible value, v_i , of A ,
 - Add a new tree branch below *Root*, corresponding to the test $A = v_i$
 - Let $Examples_{v_i}$ be the subset of *Examples* that have value v_i for A
 - If $Examples_{v_i}$ is empty
 - Then below this new branch add a leaf node with label = most common value of *Target_attribute* in *Examples*
 - Else below this new branch add the subtree
ID3($Examples_{v_i}$, *Target_attribute*, $Attributes - \{A\}$)
- End
- Return *Root*

Algoritmo Básico para Aprendizagem de Árvores de Decisão

- **Escolha Central:** selecionar qual atributo testar em cada nó da árvore.
- Devemos selecionar:
 - Atributo que é mais útil para classificar os exemplos.
 - Como sabemos qual é o mais útil?
- Medida Quantitativa:
 - **Ganho de Informação** = mede quão bem um atributo separa os exemplos de treinamento de acordo com a classificação alvo.

Indução Top-Down de Árvores de Decisão

- Qual é o melhor atributo ?



Entropia

- Caracteriza a (im)pureza de uma coleção arbitrária de exemplos.
- Dado uma coleção S contendo exemplos positivos (+) e negativos (–) de algum conceito alvo, a entropia de S relativa a esta classificação booleana é:

$$\text{Entropia}(S) \equiv -p_+ \log_2 p_+ - p_- \log_2 p_-$$

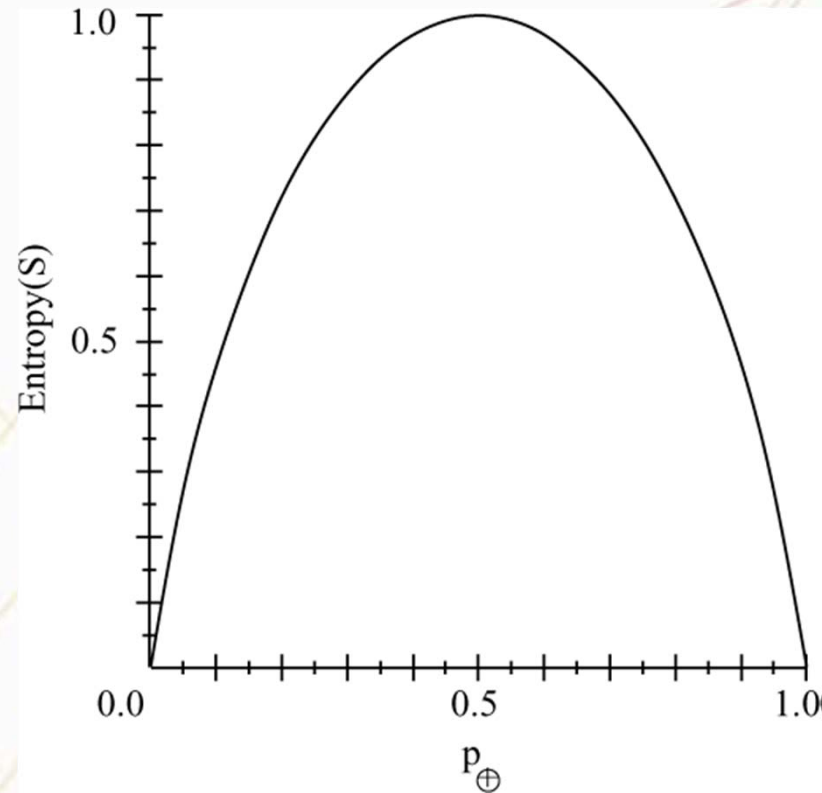
- p_+ é a proporção de exemplos positivos em S
- p_- é a proporção de exemplos negativos em S

Entropia

- Exemplo: Sendo S uma coleção de 14 exemplos de algum conceito booleano, incluindo 9 exemplos positivos e 5 negativos $[9+, 5-]$.
- A entropia de S relativa a classificação booleana é:

$$\begin{aligned} \textit{Entropia} ([9+, 5-]) &= -\left(\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14}\right) \\ &= 0.940 \end{aligned}$$

Entropia



- A função entropia relativa a uma classificação booleana, como a proporção, p_{+} de exemplos positivos varia entre 0 e 1.

Entropia

- *Entropia*(S) = número de bits esperados necessários para codificar classe (+ ou –) de membros tirados aleatoriamente de S (sob o código ótimo, de comprimento mais curto).
- Por que? Da teoria da informação: código de tamanho ótimo atribui $-\log_2 p$ bits para mensagens tendo probabilidade p .

Entropia

- Generalizando para o caso de um atributo alvo aceitar c diferentes valores, a entropia de S relativa a esta classificação c -classes é definida como:

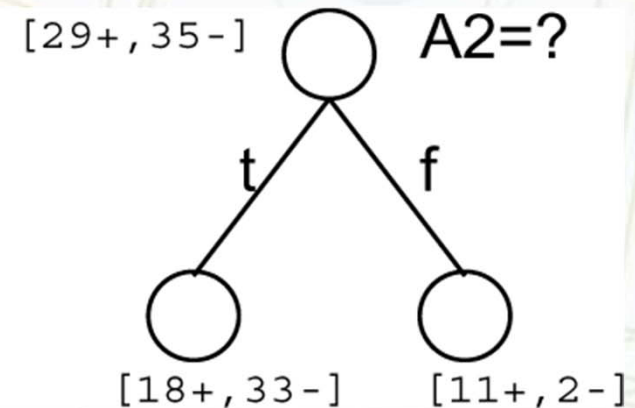
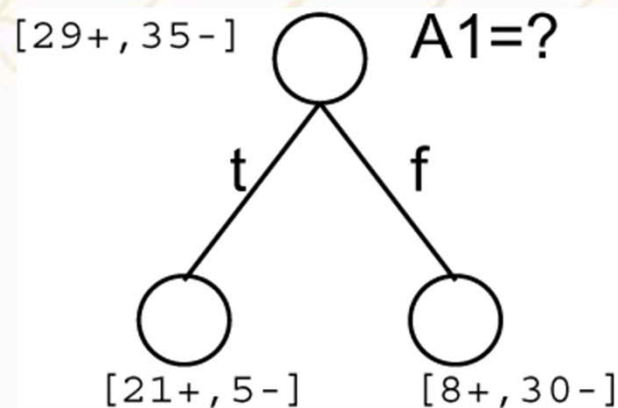
$$\textit{Entropia} (S) \equiv \sum_{i=1}^c - p_i \log_2 p_i$$

onde p_i é a proporção de S pertencendo a classe i .

Ganho de Informação

- $Gain(S, A) =$ redução esperada na entropia devido a ordenação sobre A , ou seja, a redução esperada na entropia causada pela partição dos exemplos de acordo com este atributo A .

$$Gain(S, A) \equiv Entropia(S) - \sum_{v \in \text{Valores}(A)} \frac{|S_v|}{|S|} Entropia(S_v)$$



Ganho de Informação

- S é uma coleção de (dias) exemplos de treinamento descritos por atributos incluindo *Wind*. Temos 14 exemplos.

$$\text{Values}(\text{Wind}) = \text{Weak}, \text{Strong}$$

$$S = [9+, 5-]$$

$$S_{\text{Weak}} \leftarrow [6+, 2-]$$

$$S_{\text{Strong}} \leftarrow [3+, 3-]$$

$$\begin{aligned} \text{Gain}(S, \text{Wind}) &= \text{Entropy}(S) - \sum_{v \in \{\text{Weak}, \text{Strong}\}} \frac{|S_v|}{|S|} \text{Entropy}(S_v) \\ &= \text{Entropy}(S) - (8/14) \text{Entropy}(S_{\text{Weak}}) \\ &\quad - (6/14) \text{Entropy}(S_{\text{Strong}}) \\ &= 0.940 - (8/14)0.811 - (6/14)1.00 \\ &= 0.048 \end{aligned}$$

Tarefa de Aprendizagem



- Vamos considerar a tarefa de **aprender o conceito** “*dias nos quais eu jogo tennis*”
- A tabela abaixo apresenta um conjunto de dias, cada um representado por um conjunto de características (ou atributos ou *features*)

<i>Outlook</i>	<i>Temperature</i>	<i>Humidity</i>	<i>Wind</i>	<i>PlayTennis</i>
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes



Tarefa de Aprendizagem

Atributos (ou Características ou *Features*)

Atributo Alvo
ou
Conceito Alvo



<i>Outlook</i>	<i>Temperature</i>	<i>Humidity</i>	<i>Wind</i>	<i>PlayTennis</i>
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes



Valor dos Atributos

Valor do Atributo Alvo
ou
Valor do Conceito Alvo

Tarefa de Aprendizagem

- No treinamento queremos encontrar a relação entre “o valor dos atributos” e o “valor do atributo alvo/conceito alvo”
- Uma vez “treinado”, dado um dia qualquer que é representado pelos 4 atributos, desejamos saber o valor do conceito alvo.

<i>Outlook</i>	<i>Temperature</i>	<i>Humidity</i>	<i>Wind</i>	<i>PlayTennis</i>
Sunny	Hot	High	Weak	???
Sunny	Hot	High	Strong	???
Overcast	Hot	High	Weak	???
Rain	Mild	High	Weak	???

Tarefa de Aprendizagem

- O atributo *PlayTennis* indica se eu jogo ou não tennis naquele dia.

Qual é a tarefa de aprendizagem ?

Aprender a prever o valor de *PlayTennis* para um dia qualquer baseando-se apenas nos valores dos outros atributos (*Outlook, Temperature, Humidity, Wind*).

Exemplo Ilustrativo

- Atributo alvo: *PlayTennis* (Yes, No)

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Exemplo Ilustrativo

- Primeiro passo: criação do nó superior da árvore de decisão.

Qual atributo deve ser testado primeiro na árvore?

- Determinar o ganho de informação (*Gain*) para cada atributo candidato (i.e. *Outlook*, *Temperature*, *Humidity* e *Wind*)
- Selecionar aquele cujo ganho de informação é o mais alto.

Exemplo Ilustrativo

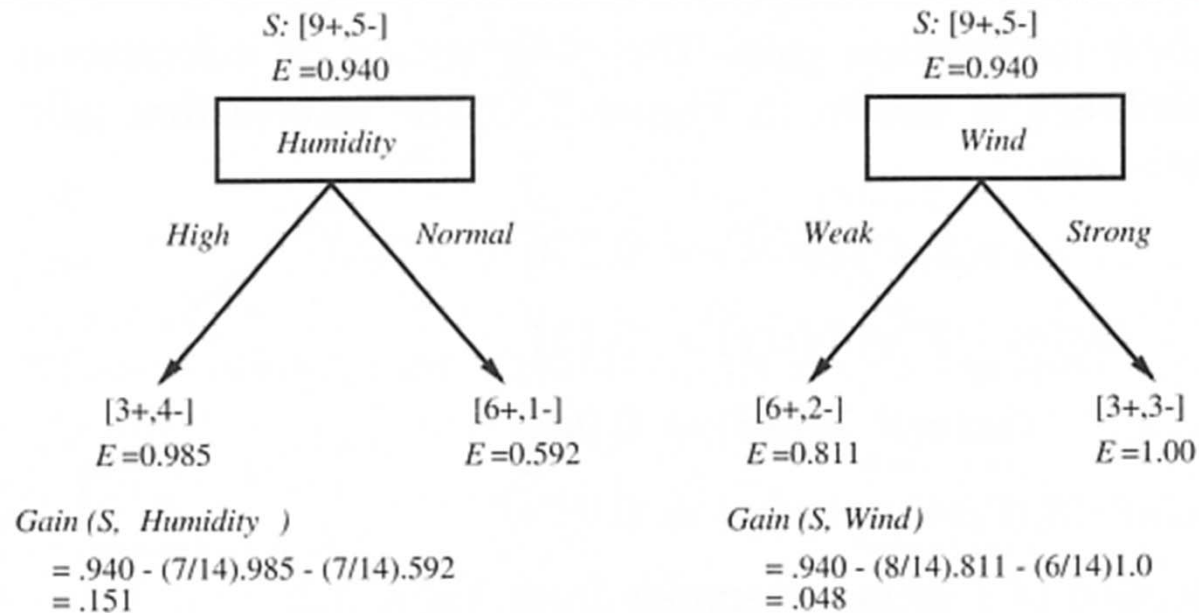


FIGURE 3.3

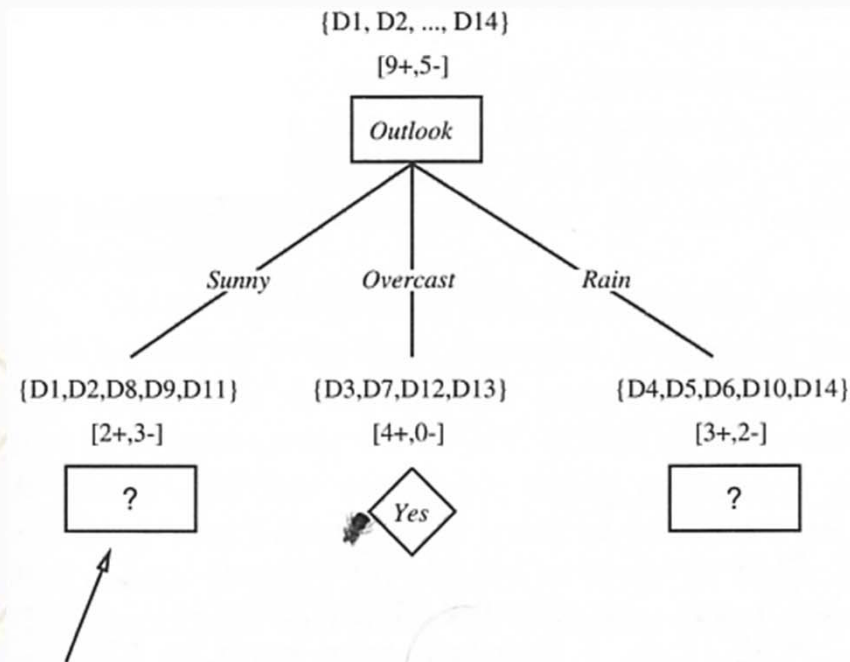
Humidity provides greater information gain than *Wind*, relative to the target classification. Here, E stands for entropy and S for the original collection of examples. Given an initial collection S of 9 positive and 5 negative examples, $[9+, 5-]$, sorting these by their *Humidity* produces collections of $[3+, 4-]$ (*Humidity* = *High*) and $[6+, 1-]$ (*Humidity* = *Normal*). The information gained by this partitioning is .151, compared to a gain of only .048 for the attribute *Wind*.

Exemplo Ilustrativo

Exemplo:

- $Gain(S, Outlook) = 0.246$
 - $Gain(S, Humidity) = 0.151$
 - $Gain(S, Wind) = 0.048$
 - $Gain(S, Temperature) = 0.029$
- Ou seja, o atributo *Outlook* fornece a melhor predição do atributo alvo, *PlayTennis*, sobre os exemplos de treinamento (Fig 3.4)

Exemplo Ilustrativo



Which attribute should be tested here?

$$S_{\text{sunny}} = \{D1, D2, D8, D9, D11\}$$

$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = .970 - (3/5) 0.0 - (2/5) 0.0 + .970$$

$$\text{Gain}(S_{\text{sunny}}, \text{Temperature}) = .970 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = .570$$

$$\text{Gain}(S_{\text{sunny}}, \text{Wind}) = .970 - (2/5) 1.0 - (3/5) .918 = .019$$

FIGURE 3.4

The partially learned decision tree resulting from the first step of ID3. The training examples are sorted to the corresponding descendant nodes. The *Overcast* descendant has only positive examples and therefore becomes a leaf node with classification *Yes*. The other two nodes will be further expanded, by selecting the attribute with highest information gain relative to the new subsets of examples.

Exemplo Ilustrativo

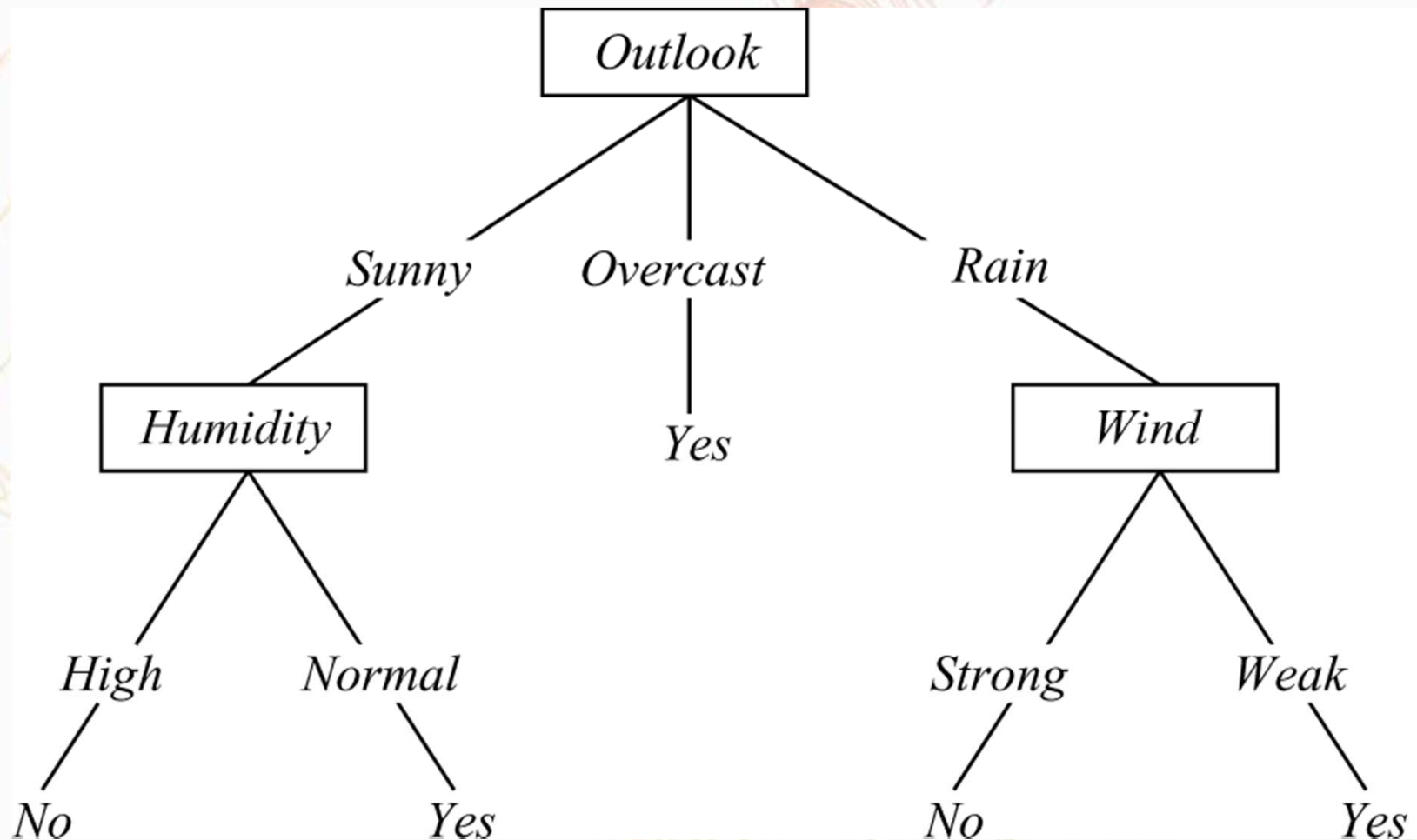
- O processo para selecionar um novo atributo e particionar os exemplos de treinamento é repetido para cada nó descendente não terminal
- São utilizados somente os exemplos de treinamento associados com este nó.
- Atributos que foram incorporados anteriormente a árvore são excluídos → qualquer atributo deve aparecer somente uma vez ao longo de qualquer caminho na árvore.

Exemplo Ilustrativo

- Este processo continua até que uma das seguintes condições seja atendida:
 1. Todos os atributos já estejam incluídos ao longo deste caminho da árvore;
 2. Os exemplos de treinamento associados com este nó folha tenham todos o mesmo valor de atributo alvo.
- A árvore final é ...

Exemplo Ilustrativo

- Árvore de decisão final.



Árvore de Decisão para PlayTennis

- Representação de árvores de decisão:
 - Cada nó interno testa um atributo
 - Cada ramo corresponde ao valor do atributo
 - Cada folha atribui uma classificação

Árvore de Decisão para *PlayTennis*

- Em geral, árvores de decisão representam uma disjunção de conjunções de restrições sobre valores dos atributos das instâncias.
- Cada caminho entre a raiz da árvore e uma folha corresponde a uma conjunção de testes de atributos e a própria árvore corresponde a uma disjunção destas conjunções.
- Exemplo:
 - $(Outlook=Sunny \wedge Humidity=Normal)$
 - ✓ $(Outlook=Overcast)$
 - ✓ $(Outlook=Rain, Wind=Weak)$

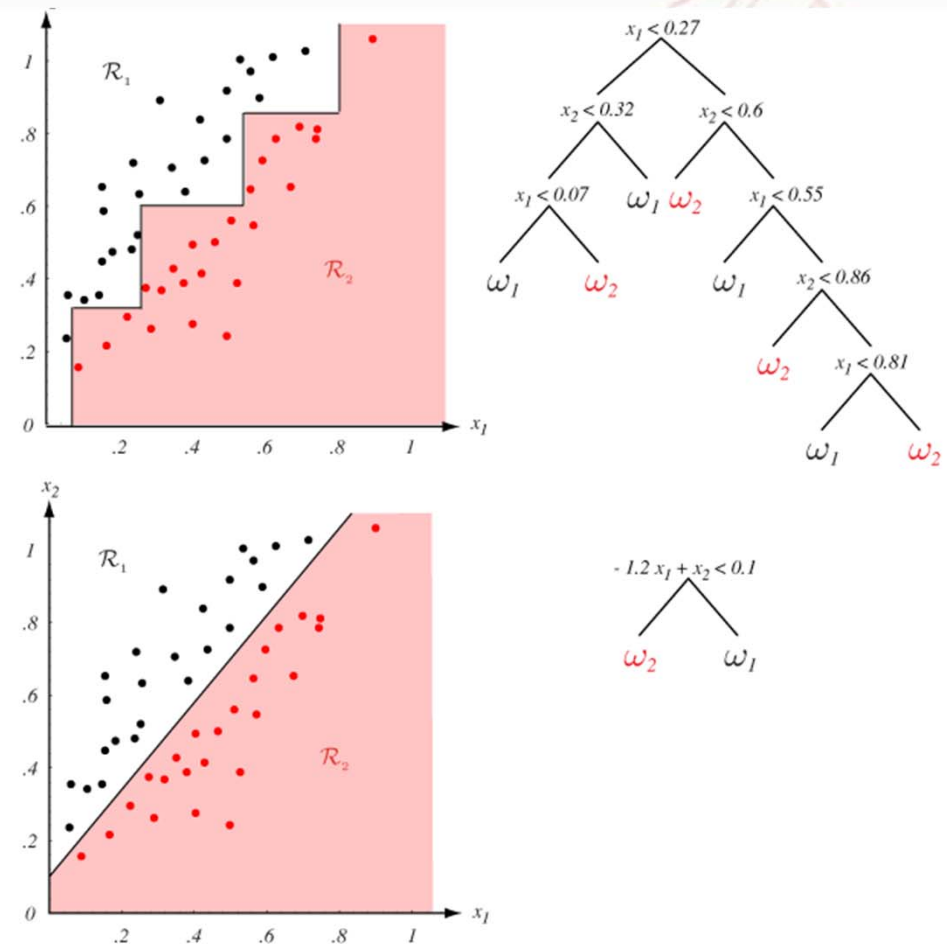
Bias Indutivo no ID3

- Dada uma coleção de exemplos de treinamento, existem geralmente várias árvores de decisão consistentes com os exemplos.

Qual árvore deve ser escolhida ????

- A preferência é por árvores mais curtas e por aquelas com atributos de alto ganho de informação próximos da raiz
- *Occam's razor* prefere as hipóteses mais curtas (mais simples) que se ajustam aos dados ...

Superfície de Separação



If the class of node decisions does not match the form of the training data, a very complicated decision tree will result, as shown at the top. Here decisions are parallel to the axes while in fact the data is better split by boundaries along another direction. If, however, "proper" decision forms are used (here, linear combinations of the features), the tree can be quite simple, as shown at the bottom.

Aspectos na Aprendizagem de Árvores de Decisão

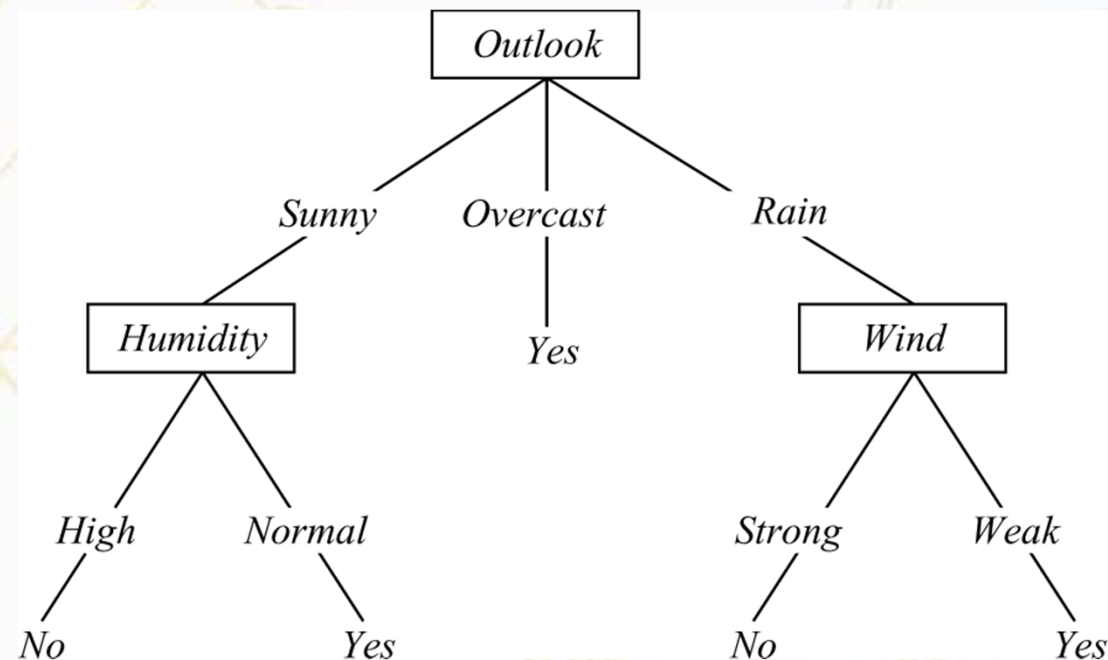
- Aspectos práticos na aprendizagem de árvores de decisão:
 - Crescimento da profundidade de árvores de decisão
 - Manipulação de atributos contínuos
 - Escolha de uma medida apropriada para a seleção de atributos
 - Manipulação de atributos com diferentes custos
 - Melhoria da eficiência computacional

Sobreajuste em Árvores de Decisão

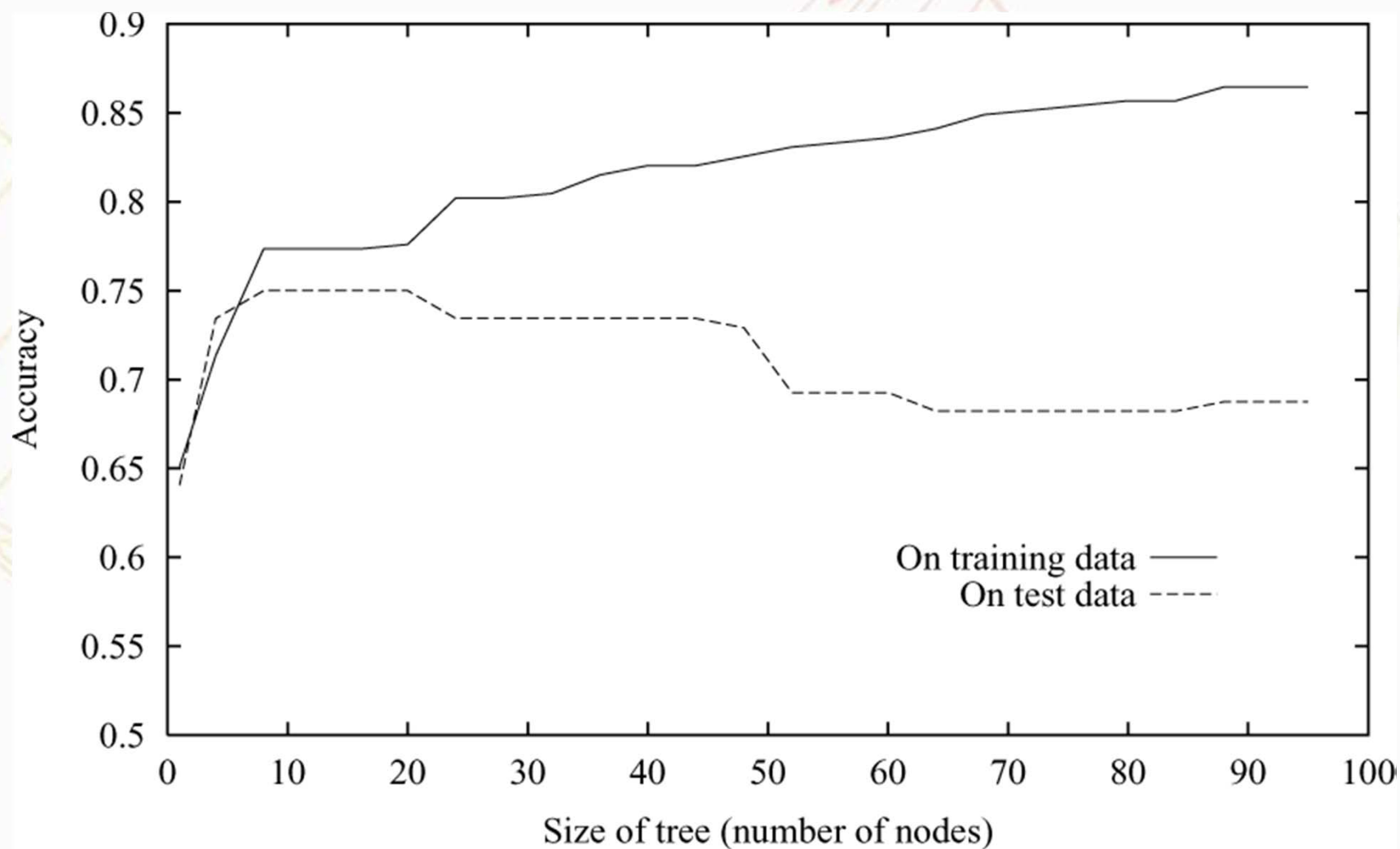
- Considere a adição de ruído no exemplo de treinamento #15.

< Sunny, Hot, Normal, Strong, PlayTennis = No >

- Qual o efeito na árvore anterior?



Sobreajuste no Treinamento de Árvores de Decisão



Evitando Sobreajuste

- Como podemos evitar o sobreajuste?
 - Parar o crescimento quando a partição de dados não for estatisticamente significativa
 - Desenvolver uma árvore completa e então fazer uma poda.
- Como selecionar a melhor árvore
 - Medida de performance sobre os dados de treinamento
 - Medida de performance sobre um conjunto de dados de validação

Erro de Poda Reduzido

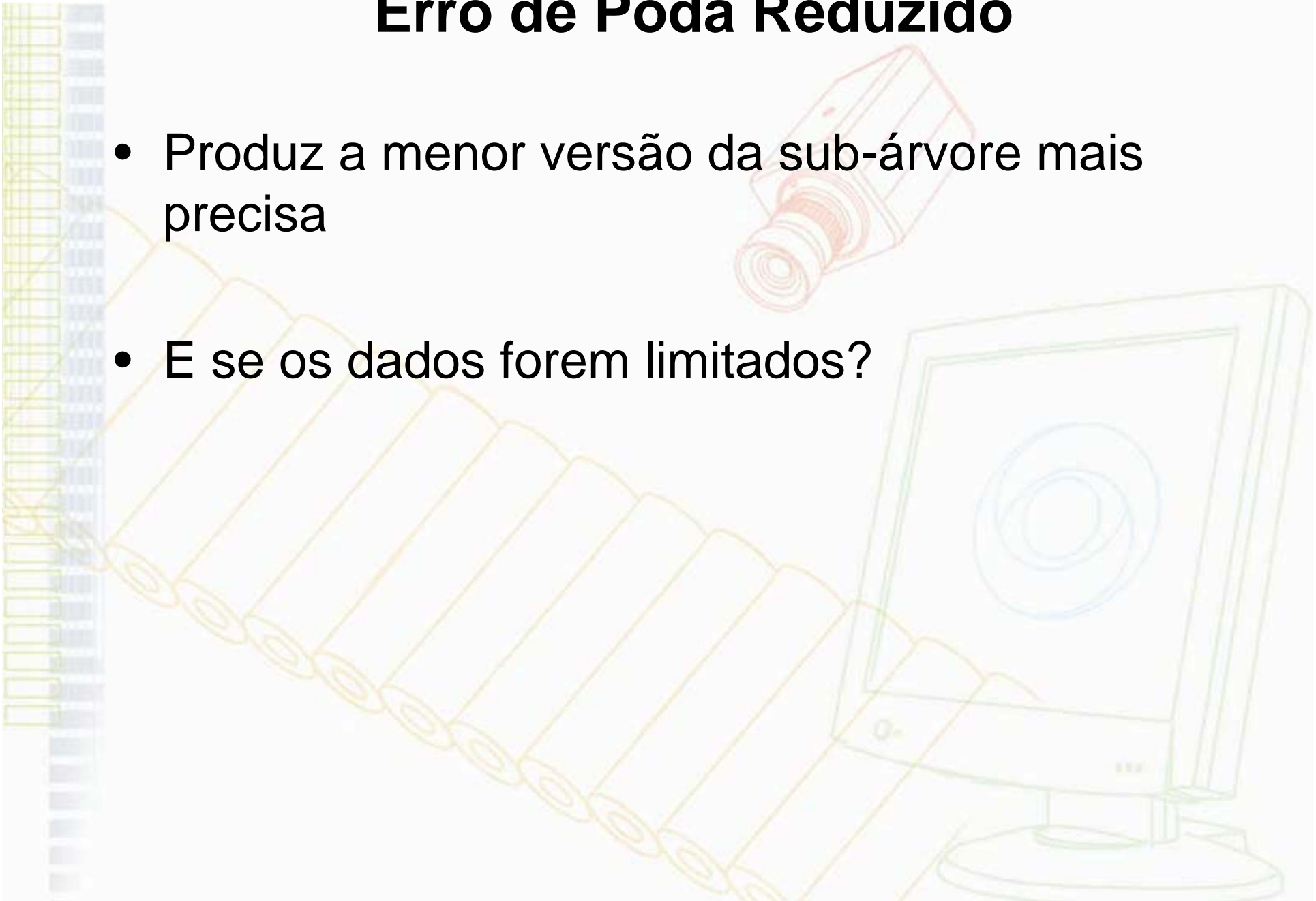
- *Podar um nó de decisão* → consiste em remover a sub-árvore enraizada naquele nó, tornando-o um nó folha.
- Atribuir a este nó, a classificação mais comum dos exemplos de treinamento afiliados com aquele nó.
- Nós são removidos somente se a árvore aparada resultante não apresenta um comportamento pior do que a original sobre o conjunto de validação

Erro de Poda Reduzido

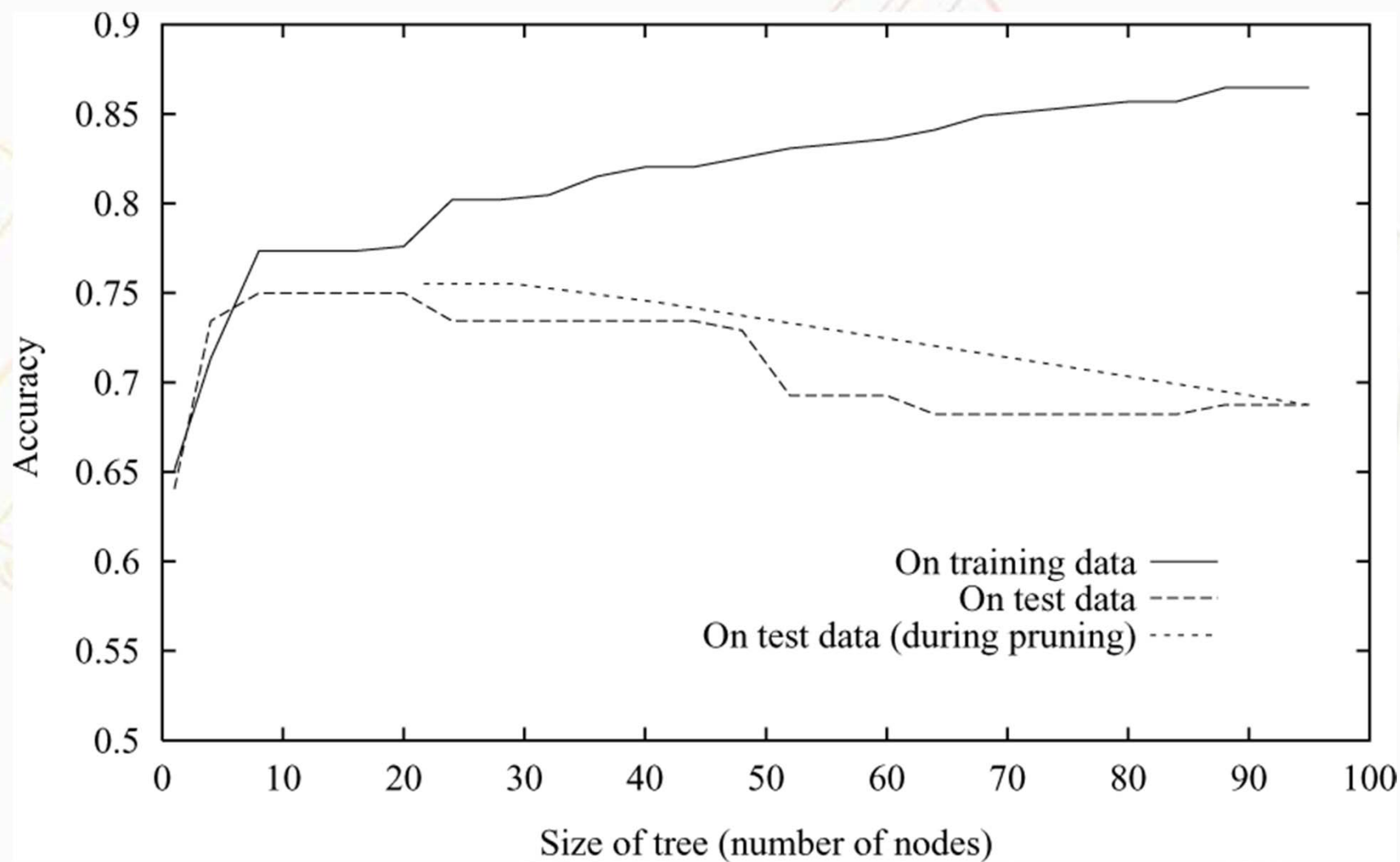
- Particionar os dados em conjuntos de validação e treinamento
- Faça até que uma redução (poda) adicional seja prejudicial
 1. Avaliar o impacto sobre o conjunto de validação da poda de cada nó possível, mais aqueles abaixo dele
 2. Remover “gulosamente” aquele que melhora mais a precisão sobre o conjunto de validação

Erro de Poda Reduzido

- Produz a menor versão da sub-árvore mais precisa
- E se os dados forem limitados?



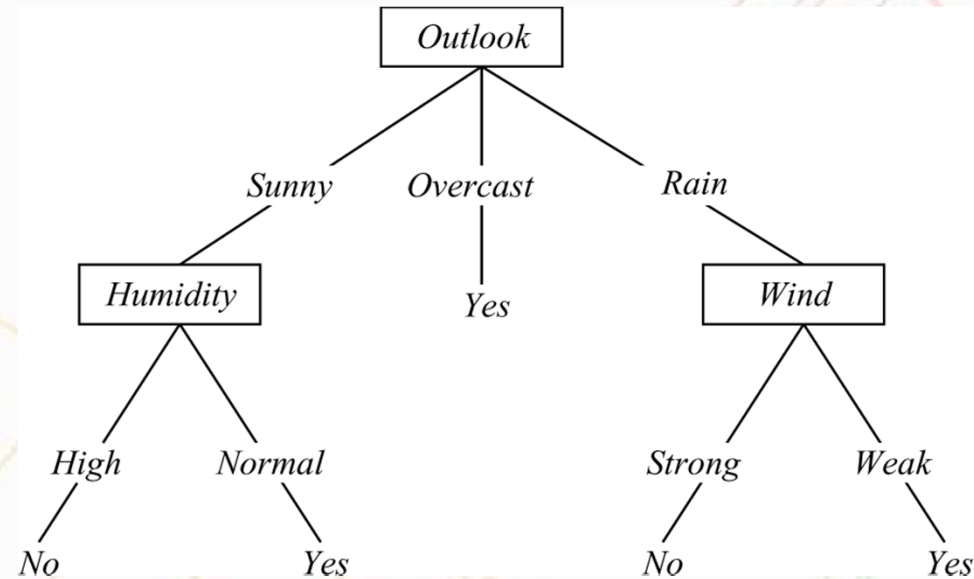
Efeito do Erro de Poda Reduzido



Pós-Redução (Poda) da Regra

1. Converter a árvore em um conjunto de regras equivalente
 2. Podar cada regra independentemente das outras
 3. Ordenar as regras finais em uma seqüência desejável para o uso
- Talvez seja o método usando com mais freqüência. (e.g. C4.5)

Convertendo uma Árvore em Regras



IF (*Outlook* = *Sunny*) ^ (*Humidity* = *High*)
THEN *PlayTennis* = *No*

IF (*Outlook* = *Sunny*) ^ (*Humidity* = *Normal*)
THEN *PlayTennis* = *Yes*

Convertendo uma Árvore em Regras

- Vantagens de converter uma árvore de decisão em regras antes da poda:
 - Permite distinguir entre os diferentes contextos onde os nós de decisão são utilizados.
 - Remove a distinção entre atributos de testes que ocorrem próximos da raiz da árvore e aqueles que ocorrem próximos das folhas.
 - Melhora a leitura humana. Regras são geralmente mais fáceis para pessoas entenderem

Atributos de Valor Contínuo

Na definição da ID3 temos as restrições:

- 1. Atributo alvo deve ter valor discreto
- 2. Os atributos testados nos nós de decisão devem também ser de valor discreto.

A segunda restrição pode ser removida.

- Definir dinamicamente novos atributos de valor discreto que particionam o valor do atributo contínuo em um conjunto discreto de intervalos.
- A = atributo de valor contínuo \rightarrow criar um novo atributo A_c que é verdadeiro se $A < c$ e falso caso contrário.
- Como identificar c ???

Atributos de Valor Contínuo

Exemplo: Incluir o atributo de valor contínuo *Temperature* na descrição dos exemplos de treinamento para a tarefa de aprendizagem anterior.

Supondo que os exemplos de treinamento associados a um nó particular são

<i>Temperature:</i>	40	48	60	72	80	90
<i>PlayTennis:</i>	No	No	Yes	Yes	Yes	No

- Escolher um limiar c que produza o maior ganho de informações → Identificar exemplos adjacentes que diferem na classificação alvo.

Atributos de Valor Contínuo

<i>Temperature:</i>	40	48	60	72	80	90
<i>PlayTennis:</i>	No	No	Yes	Yes	Yes	No

Dois limiares candidatos:

- $c_1 = (48+60)/2 = 54$
- $c_2 = (80+90)/2 = 85$
- O ganho de informação pode ser calculado para cada um destes atributos candidatos: $Temperature_{>54}$ e $Temperature_{>85}$ e o melhor pode ser selecionado ($T_{>54}$)

Resumo

- Aprendizagem de árvores de decisão fornece um método prático para a aprendizagem de conceito e para a aprendizagem de outras funções de valor discreto
- A família de algoritmos ID3 infere árvores de decisão expandindo-as a partir da raiz e descendo, selecionando o próximo melhor atributo para cada novo ramo de decisão adicionado a árvore.

Prós

- Simplicidade para compreensão e interpretação
 - árvores de decisão são facilmente compreendidas após uma breve explicação.
- Os dados não necessitam de pré-processamento
 - outras técnicas normalmente exigem normalização de dados.
- Lidam tanto com dados numéricos quanto categóricos
 - outras técnicas normalmente lidam somente com um único tipo de variável.

Prós

- Emprega um modelo “caixa branca”
 - Se uma dada situação é observável em um modelo, a explicação para a condição é facilmente feita através da lógica booleana.
- Possibilidade de validar um modelo através de testes estatísticos.
 - é possível avaliar a confiabilidade do modelo.
- Robustez.
 - Bom desempenho mesmo se as suposições iniciais do modelo de dados forem violadas.
- Bom desempenho em grandes conjuntos de dados em um tempo curto.
 - Grandes quantidades de dados podem ser analisados utilizando recursos computacionais comuns.

Contras (Limitações)

- O problema de aprender uma árvore de decisão ótima é NP-Completo.
 - Os algoritmos práticos de aprendizagem de árvore de decisão são baseados em heurísticas (e.g. algoritmo guloso) onde decisões ótimas locais são tomadas em cada nó.
- O aprendizado de árvores de decisão pode criar árvores muito complexas que não generalizam bem os dados.
 - Sobreajuste (Overfitting!!). Mecanismos de poda são necessários para evitar este problema.

Contras (Limitações)

- Alguns conceitos são difíceis de serem aprendidos, pois, árvores de decisão não expressa-os facilmente.
 - Problemas XOR, paridade e multiplexador
 - Nestes casos as árvores de decisão se tornam proibitivamente grandes.
- Para dados com variáveis categóricas, com diferentes níveis, o ganho de informação é tendencioso em favor daqueles atributos que possuem mais níveis.