Inteligência Computacional [2COP229]

Mestrado em Ciência da Computação

Sylvio Barbon Jr
barbon@uel.br

Tema

Aula 3
Árvores de Decisão e Random Forest

Sumário

- Árvores de Decisão e Regressão
- Indução de Árvores
- Estratégias de Poda
- Vantagens e Desvantagens
- Ensemble Learning
- Floresta Aleatória (Random Forest)

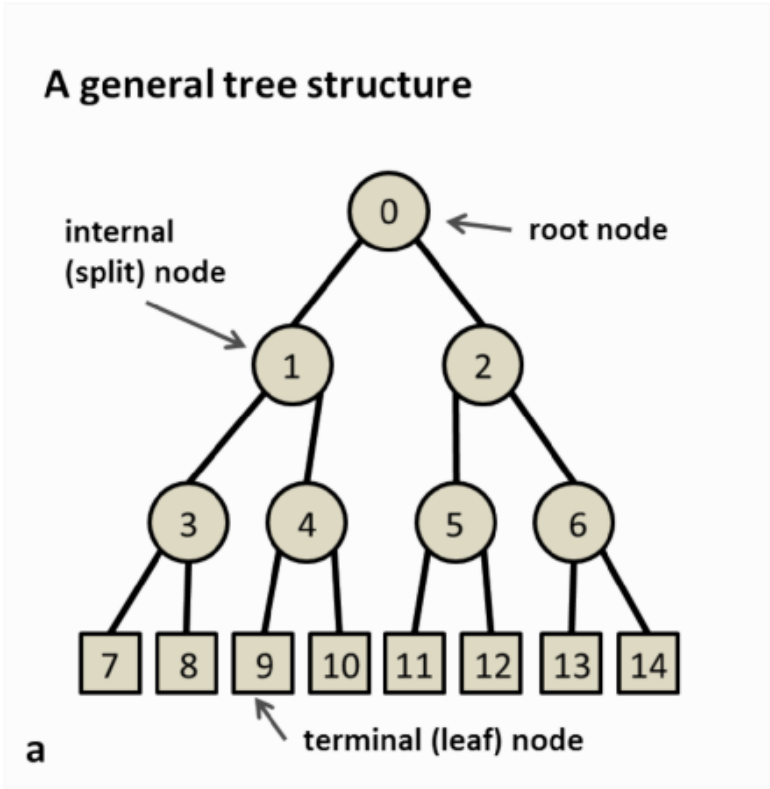
Introdução

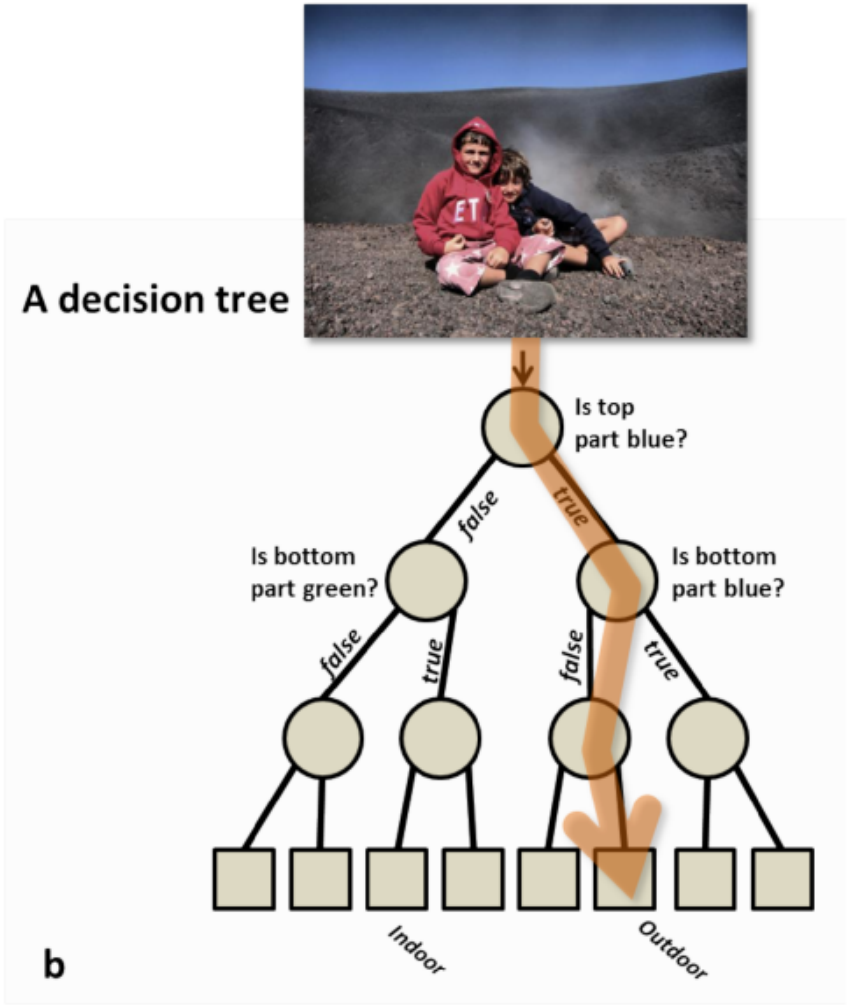
- O problema de Aprendizado de Máquina pode ser formulado como um problema de procura num espaço de possíveis soluções.
- Uma árvore de decisão é um problema de **dividir para conquistar** para resolver um problema de decisão baseado em procura.
- Um problema complexo é reduzido em problemas mais simples recursivamente avaliado.
- As soluções para os subproblemas podem ser combinadas, em forma de uma árvore, para reduzir uma solução de um problema complexo.

Introdução

- Exemplos de algoritmos baseados em árvores:
  1. ID3 (1979)
  2. CHAID (1980)
  3. Assistant (1987)
  4. CART (1984)
  5. C4.5 (1993)
  6. J48 (2001)
  7. Random Forest(2001)

Introdução

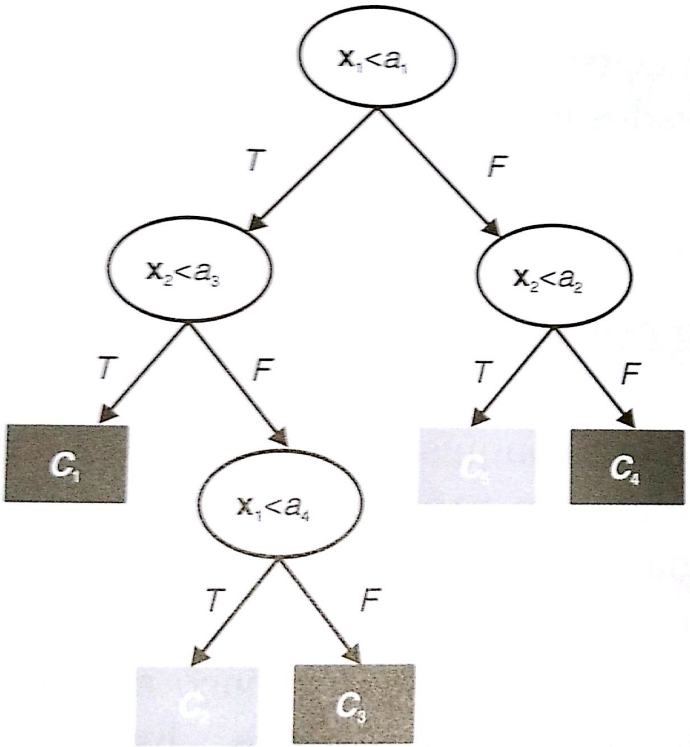

A general tree structure
internal
(split) node
root node
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
terminal (leaf) node
a

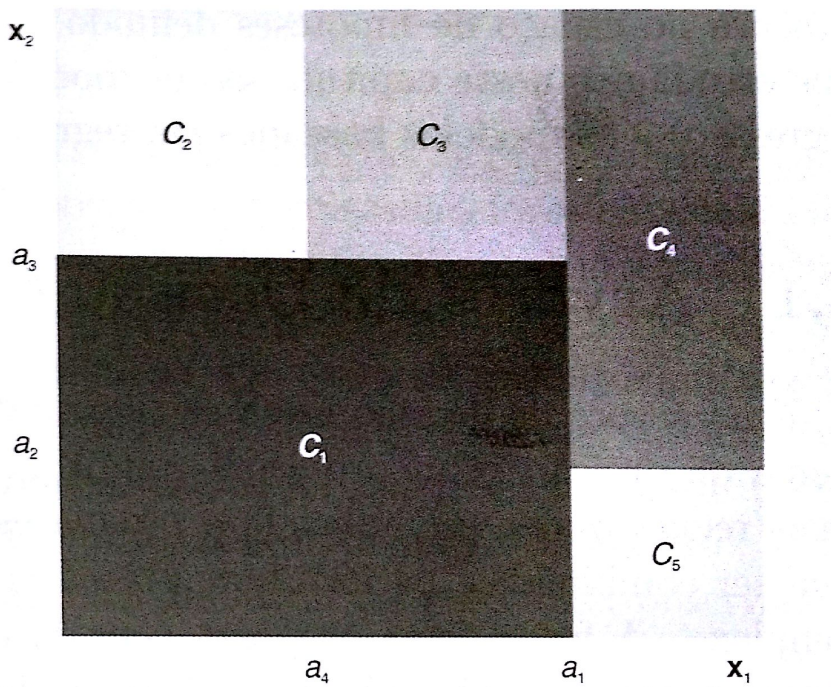

A decision tree
Is top
part blue?
false
true
Is bottom
part green?
Is bottom
part blue?
false
true
false
true
Indoor
Outdoor
b

Introdução

- Formalmente uma árvore é um grafo acíclico direcionado em que cada nó é um nó de divisão, com dois ou mais sucessores, ou um nó folha.
- Um nó folha é rotulado como uma função.
- Um nó de divisão contém um teste condicional baseado nos valores dos atributos. Exemplo
  - Idade ≥ 12
  - Cor $\in \{\text{azul, verde}\}$
  - $0{,}3 + 0{,}5 * x_1 - 0,5 * x_2 \leq 0$
- Uma árvore deve abranger todos os espeços de instâncias.
- Forma Normal Disjunta (FND) é o formalismo que descreve que para cada ramo (percurso entre raiz e folha) são conjunções de condições e os ramos individuais disjunções. Esses classificadores geram **hiper-retângulos**.

Introdução

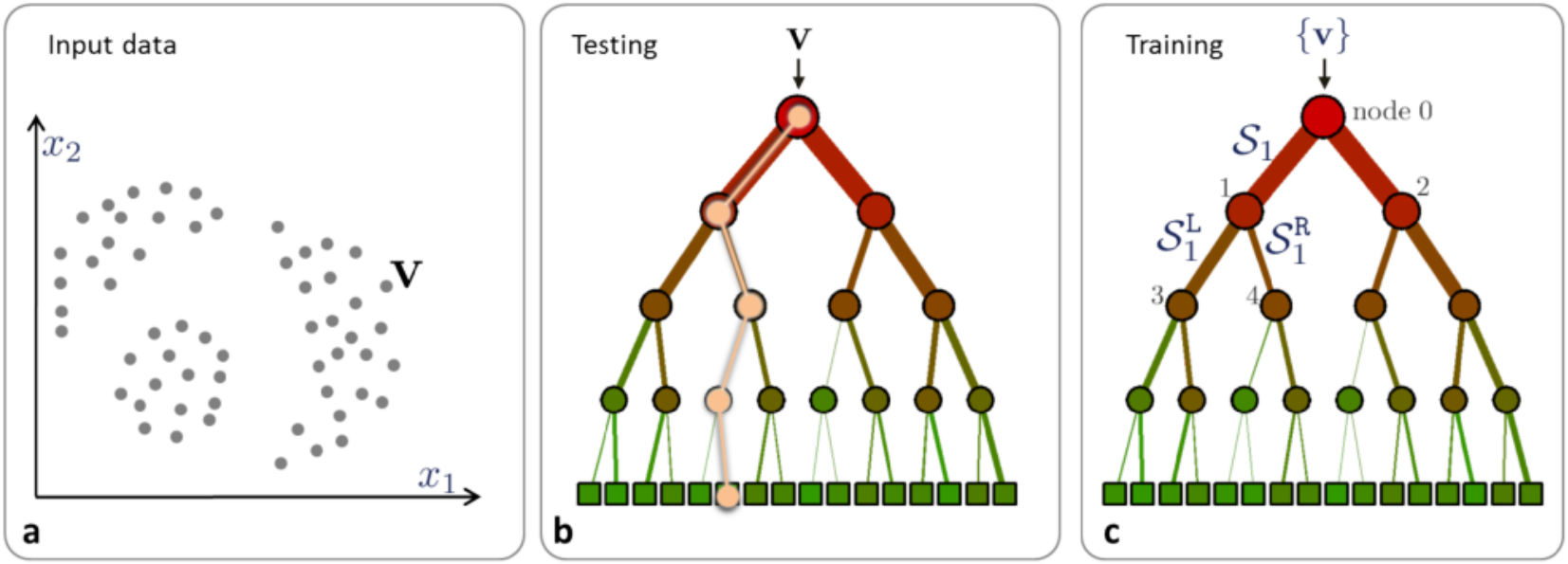

$x_1 < a_1$
T
F
$x_2 < a_3$
$x_2 < a_2$
T
F
T
F
C_1
$x_1 < a_4$
C_4
T
F
C_3


x_2
C_2
C_3
C_4
a_3
a_2
C_1
C_5
a_4
a_1
x_1

Indução de Árvores

- A construção da árvore é um problema de decisão minimal (em termos de números de nós), sendo um problema NP-COMPLETO.
- Características de **hill-climbing** sem **back-tracking**, assim é suscetível a ser uma solução ótima localmente e não ser a global.
- A vantagem é que é uma solução **linear** para os n número de exemplos.
- Entre as regras de divisão para classificação temos como principal guia a abordagem "goodness of split", que indica quão bem um atributo discrimina a classe.
- As funções de mérito indicam quais conjuntos de atributos tem utilidade para a separação das classes.

Introdução


Input data
x_2
x_1
v
a
Testing
v
b
Training
{v}
node 0
$\mathcal{S}_1$
1
2
$\mathcal{S}_1^L$
$\mathcal{S}_1^R$
3
4
c

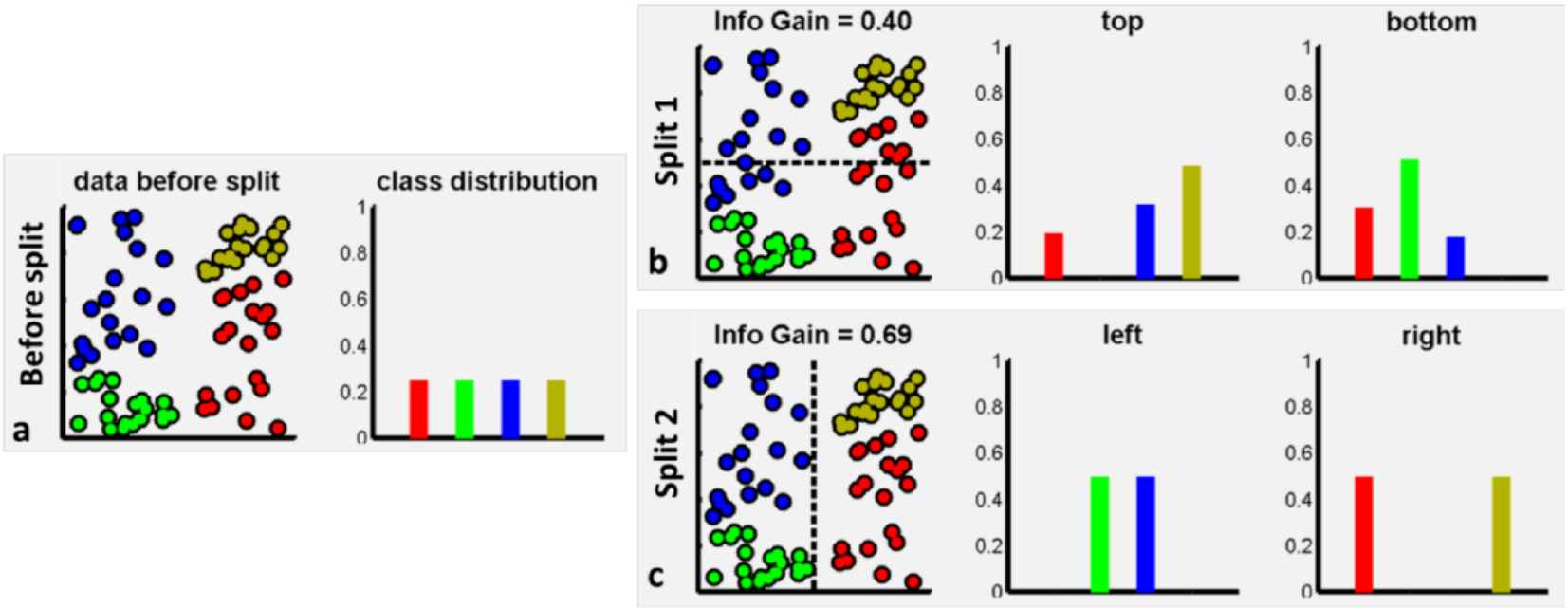
Indução de Árvores

- Existem as seguintes funções para realizar a avaliação da função mérito:
  1. Diferença entre a distribuição no nó pai e subconjuntos obtidos baseados em proporções de classe (ex. entropia).
  2. Diferença entre os subconjuntos divididos com base na proporção como distância ou ângulo, enfatizando a disparidade entre os subconjuntos.
3. Medidas estatísticas independentes entre classe e subconjunto, por exemplo χ^2.
- As regras de divisão baseadas no **Ganho de Informação** são as mais populares, utilizadas por exemplo, nos algoritmos J48 e C4.5.

Indução de Árvores

- Ganho de Informação tem como kernel a **Entropia**.
- A Entropia mede a aleatoriedade dos dados e é calculada como:
  $H(A) = -\sum_i p_i \times log_2\ p_i$, onde p é a probabilidade de obsevar um dadao valor A.
- A cada nó de decisão, o atributo que mais reduz a aleatoriedade da variável alvo será escolhido para dividir os dados.
- O Ganho de Informação mede a redução na Entropia.
- O Ganho de informação é obtido como:
  $IG(A, p, q) = I(p, q) - E(A, p, q)$, sendo p e q o número de objetos de duas classes diferentes.

Introdução


data before split
class distribution
Before split
a
Info Gain = 0.40
top
bottom
Split 1
b
Info Gain = 0.69
left
right
Split 2
c
1
0.8
0.6
0.4
0.2
0

Exemplo

Suponha que o problema de decisão é quando alguém joga ou não um dado esporte pelas condições do tempo. O problema é definido por quatro atributos de entrada: TEMPO, TEMPERATURA, UMIDADE e VENTO. O conjunto de treinamento contém 14 exemplos que descrevem observações de indivíduos (Joga) dada as condições do tempo.

Qual o valor que melhor discrimina as classes?

Exemplo

Tempo	Temperatura	Umidade	Vento	Joga
Chuvoso	71	91	Sim	Não
Ensolarado	69	70	Não	Sim
Ensolarado	80	90	Sim	Não
Nublado	83	86	Não	Sim
Chuvoso	70	96	Não	Sim
Chuvoso	65	70	Sim	Não
Nublado	64	65	Sim	Sim
Nublado	72	90	Sim	Sim
Ensolarado	75	70	Sim	Sim
Chuvoso	68	80	Não	Sim
Nublado	81	75	Não	Sim
Ensolarado	85	85	Não	Não
Ensolarado	72	95	Não	Não
Chuvoso	75	80	Não	Sim

Exemplo

- A entropia da classe para o conjunto de exemplos é:
  - p(Joga=Sim) $= 9/14$
  - p(Joga=Não) $= 5/14$
  - H(Joga) $= -9/14 \times log_2(9/14) - 5/14 \times log_2(5/14) = 0,94$

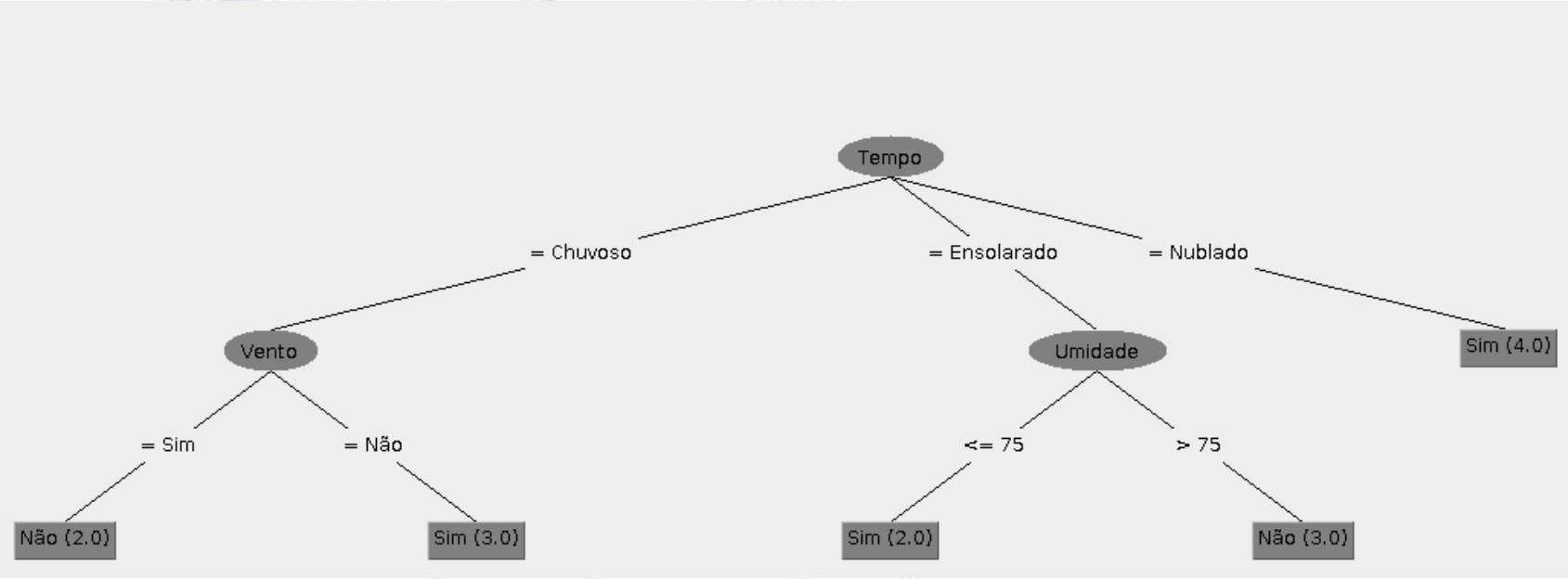
Exemplo

- Ganho de Informação para **Atributo Nominal**:
  - Deve-se dividir o conjunto pelos valores do atributo:
  - p(Joga=SIM||Tempo=Ensolarado)= 2/5
  - p(Joga=Não||Tempo=Ensolarado)= 3/5
  - H(Jogar||Tempo = Ensolarado) = $-2/5 \times log_2(2/5) - 3/5 \times log_2(3/5) = 0,971$
  - Outras partições:
  - H(Jogar||Tempo = Nublado) = 0,0
  - H(Jogar||Tempo = Chuvoso) = 0,971
  - A entropia ponderada para o **tempo** é: $H(Tempo) = 5/14 \times 0,971 + 4/14 \times 0 + 5/14 \times 0,971 = 0,693$
  - $IG(Tempo) = 0,940 - 0,693 = 0,247$

Exemplo

- Ganho de Informação para **Atributo Contínuo** usa a estratégia para conjuntos em que atributo $\leq$ valor e atributo $>$ valor. Considere o teste Temperatura $= 70{,}5$.
  - Temperatura $> 70{,}5 =$ $Verdadeiro\{SIM(5), Não(4)\}, Falso\{Sim(4), Não(1)\}$
  - p(Joga=Sim$\|$Temperatura$\leq$70,5)$= 4/5$
  - p(Joga=Não$\|$Temperatura$\leq$70,5)$= 1/5$
  - p(Joga=Sim$\|$Temperatura$>$70,5)$= 5/9$
  - p(Joga=Não$\|$Temperatura$>$70,5)$= 4/9$
  - H(Joga$\|$Temperatura$\leq$70,5) $=$ $-4/5 \times log_2(4/5) - 1/5 \times log_2(1/5) = 0,721$
  - H(Joga$\|$Temperatura 70,5) $=$ $-5/9 \times log_2(5/9) - 4/9 \times log_2(4/9) = 0,991$
  - H(Temperatura) $= 5/14 \times 0{,}721 + 9/14 \times 0{,}991 = 0{,}895$
  - IG(Temperatura) $= 0{,}940$ - $0{,}895 = 0{,}045$

Introdução


Tempo
= Chuvoso
= Ensolarado
= Nublado
Vento
Umidade
Sim (4.0)
= Sim
= Não
<= 75
> 75
Não (2.0)
Sim (3.0)
Sim (2.0)
Não (3.0)

Estratégias de Poda

- A poda é uma importante estratégia em domínios ruidosos.
- Sem a poda, as árvores induzidas podem classificar novos objetos de maneira incorreta.
- Nós profundos refletem mais o conjunto de treinamento (overfitting) e aumentam o erro devido a variância do classificador.
- Outro ponto que estimula o uso de podas é que uma árvore grande é de difícil compreensão.
- **Podar** uma árvore é trocar nós profundos por folhas, minimizando possíveis problemas.

Estratégias de Poda

- A vantagem da poda se torna mais evidente quando se classificam novos exemplos não usados no processo de construção da árvore.
- Os métodos de poda podem ser divididos em dois grupos:
  - Pré-poda: Conta com regras de parada e previnem a construção daqueles ramos que não melhorariam o poder preditivo.
  - Pós-poda: Método mais comum baseado no erro estático e backed-up. "Construir e podar uma árvore é mais lento, porém confiáve.

Vantagens do Uso de Árvores

- Flexibilidade: Árvores não assumem nenhuma distribuição para os dados, são não paramétricos e criam decisões sobre todo o espaço de busca.
- Robustez: É invariante a transformações monótonas de variáveis de entrada, a sensibilidade a outliers é reduzida.
- Seleção de Atributos: O processo de construção da árvores seleciona os atributos e sua relevância no modelo de decisão.
- Interpretabilidade: Decisões complexas se tornar pequenas, mais simples e locais.
- Eficiência: É um algoritmo guloso, top-down sem back-tracking, tendo sua complexidade linear com o número de exemplos.

Desvantagens do Uso de Árvores

- Replicação: Duplicação de uma sequencia de testes em ramos distintos da árvore.
- Valores ausentes: A árvore é uma hierarquia de testes, se um valor é desconhecido não é possível decidir.
- Atributos contínuos: O gargalo do algoritmo são os atributos contínuos. Alguns autores até sugerem a discretização dos valores contínuos.
- Instabilidade: Pequenas variações no conjunto de treinamento geram grandes transformações na árvore.

Lista de Referências

1. C.M. Bishop, "Pattern Recognition and Machine Learning", Springer, 2006
2. R. Duda, P. Hart, D. Stork, "Pattern Classification", second edition, 2000.
3. T. Hastie, R. Tibshurani, and J.H. Friedman, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Spinger Series in Statistics, 2001.
4. N. Cristianini and J. Shawe-Taylor, "An Introduction to Support Vector Machines", Cambridge Univ. Press, 2000.
5. B. D. Ripley, "Pattern Recognition and Neural Networks", Cambridge University Press, 1996.