Árvore de Decisão

Inteligência Artificial
PCS3438

Anna Helena Reali Costa
Escola Politécnica da USP
Engenharia de Computação (PCS)

Aprendendo pelas observações

- **Ideia**: percepção deve ser usada não somente para a atuação imediata, mas também para melhorar a habilidade do agente em agir no futuro! ➔ *aprendizado*
- Aprendizado decorre das interações do agente com o mundo e pela observação do agente de seu próprio processo de decisão.

Aprendizado – por que?

- Capacidade de aprender é parte fundamental do conceito de **inteligência**.
- Um agente aprendiz é mais flexível ➔ aprendizado permite lidar com situações novas (mundo é dinâmico). Dá **autonomia** ao agente.
- Aprendizado facilita tarefa do projetista ➔ programar apenas o essencial

Como construir programas (agentes) que automaticamente melhoram com sua experiência?

Aprendizagem – paradigmas (I)

- **Aprendizado supervisionado**
  - Pares corretos de **entrada/saída** podem ser observados (ou demonstrados por um supervisor).
  - Um **erro** relativo é calculado entre a ação que deve ser tomada idealmente (saída desejada) e a ação efetivamente escolhida pelo agente (saída executada). Este erro é usado para atualizar a tomada de decisão do agente aprendiz.

Aprendizagem – paradigmas (II)

- **Aprendizado por reforço**
  - Apenas uma **indicação de desempenho** (geralmente, indicação de quão bom ou ruim é o estado atual resultante) é comunicada ao agente, normalmente de modo intermitente e apenas quando situações dramáticas são atingidas (***feedback* indireto, com retardo**).

Aprendizagem – paradigmas (III)

- **Aprendizado não-supervisionado**
  - Nenhum tipo de informação é comunicada ao agente; não há "pistas" sobre as saídas corretas.
  - Geralmente utilizam-se de **regularidades**, propriedades estatísticas dos dados sensoriais, etc.
    - Ex: Clustering (agrupamento).

Aprendizado Indutivo

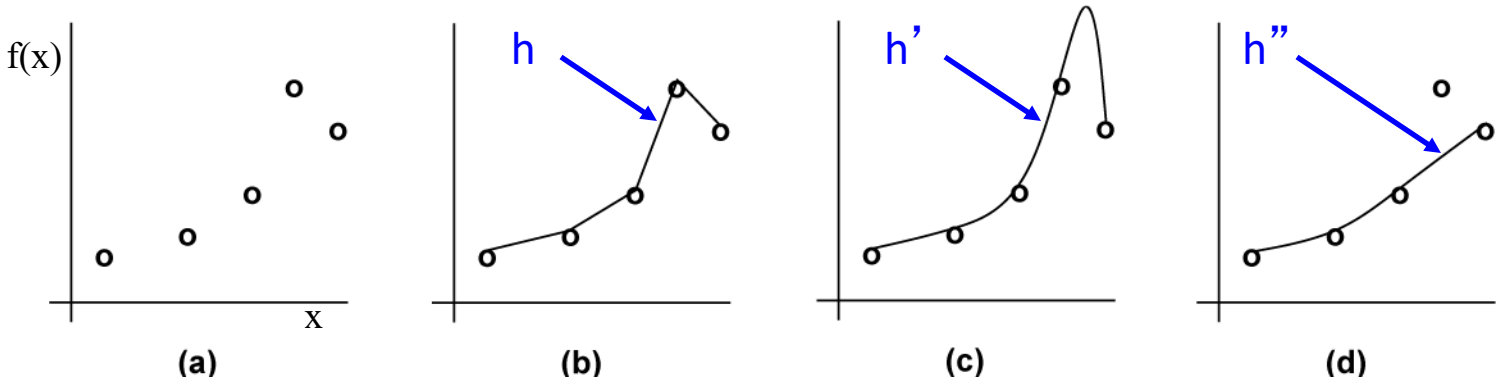
- Paradigma: aprendizado **supervisionado**.
- Funcionamento: inferência de uma regra geral (***hipótese***) a partir de exemplos particulares ➔ **generalização**
- Eficácia diretamente proporcional à quantidade de exemplos.

Aprendizado Indutivo

- Abordagem:
  - **incremental**: atualiza hipótese a cada novo exemplo
    - mais flexível, situada... Porém a ordem de apresentação é importante!
  - **não incremental**: gera hipótese a partir de todo conjunto de exemplos
    - mais eficiente e prática.
- Métodos:
  - simbólicos (ex:ID3),
  - não-simbólicos (ex:NN).

Aprendizado Indutivo

- *X*: entrada; *f(X)*: saída desejada
- Exemplo (par de treinamento) = *(X, f(X))*
- Objetivo: aprender uma função ***h*** (*hipótese*) que aproxime *f*.



Aprendizado Indutivo

Classificação: a variável de saída assume valores categóricos ou qualitativos (*class labels*).

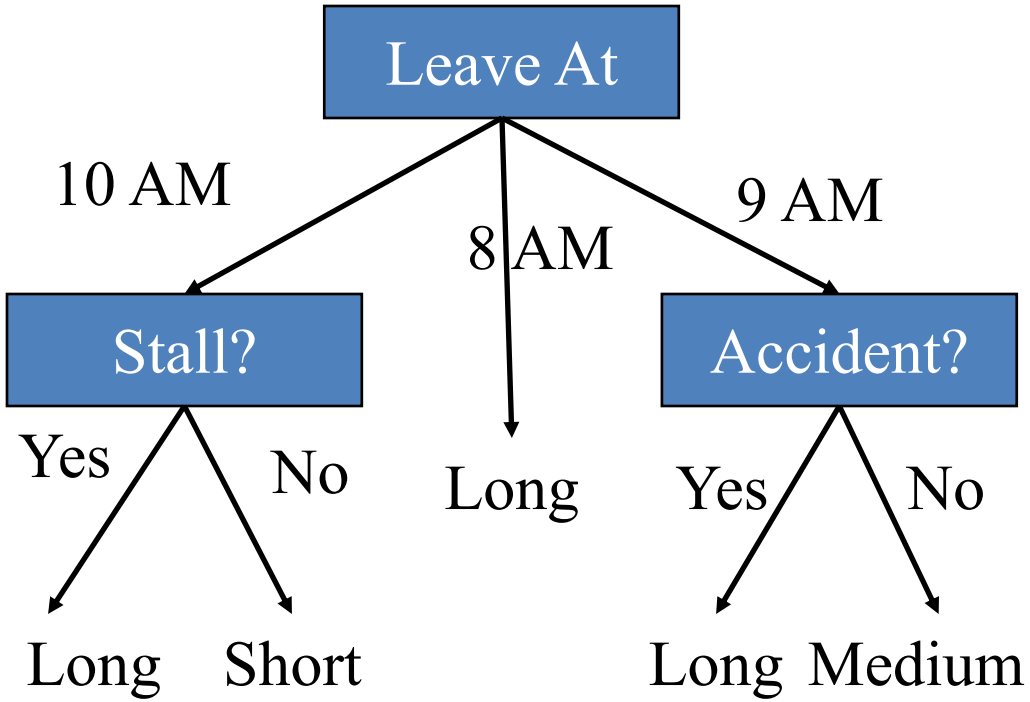
$$\hat{Y} = \hat{f}(X), \; y \in C = \{1, 2, \ldots j\},$$

$$|C| = \textit{número de classes}$$

Árvore de Decisão

- Aprendizado indutivo, supervisionado, simbólico, não incremental.
- Função aprendida: representada por uma árvore de decisão (representa funções booleanas)
  - ou conjunto de regras IF-THEN
- Entrada: objeto ou situação descrita por um conjunto de propriedades/atributos;
- Saída: “decisão” sobre o alvo

Árvores de Decisão como Regras



Ex.: Classificação

```
if hour == 8am
    commute time = long
else if hour == 9am
    if accident == yes
      commute time = long
    else
      commute time = medium
else if hour == 10am
    if stall == yes
      commute time = long
    else
      commute time = short
```

Como criar uma Árvore de Decisão

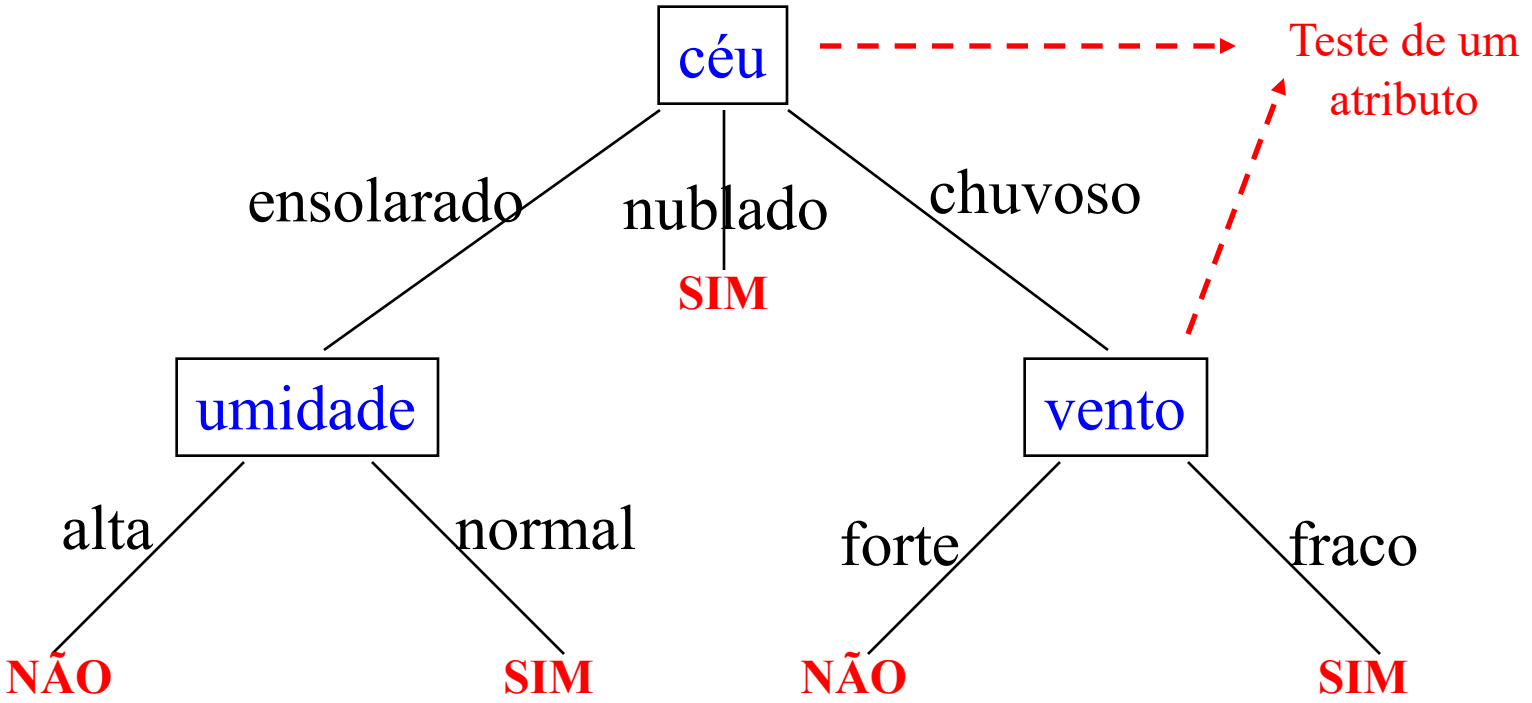
- Primeiro define-se uma lista de atributos que podemos medir (**preditores**)
- Em seguida escolhe-se um **atributo alvo** que desejamos predizer (saída)
- Finalmente, cria uma tabela de experiências que lista o que foi observado (**conjunto de treinamento**)

Vamos ver um exemplo com árvores de classificação…..

Exemplos de treinamento para o alvo *JogarTênis*

Ex	Céu	Temperatura	Umidade	Vento	JogarTênis
X1	Ensolarado	Quente	Alta	Fraco	NÃO
X2	Ensolarado	Quente	Alta	Forte	NÃO
X3	Nublado	Quente	Alta	Fraco	SIM
X4	Chuvoso	Boa	Alta	Fraco	SIM
X5	Chuvoso	Fria	Normal	Fraco	SIM
X6	Chuvoso	Fria	Normal	Forte	NÃO
X7	Nublado	Fria	Normal	Forte	SIM
X8	Ensolarado	Boa	Alta	Fraco	NÃO
X9	Ensolarado	Fria	Normal	Fraco	SIM
X10	Chuvoso	Boa	Normal	Fraco	SIM
X11	Ensolarado	Boa	Normal	Forte	SIM
X12	Nublado	Boa	Alta	Forte	SIM
X13	Nublado	Quente	Normal	Fraco	SIM
X14	Chuvoso	Boa	Alta	Forte	NÃO

Conceito a aprender: devo jogar tênis?



Devo jogar tênis se: (céu = ensolarado $\wedge$ umidade = normal)
$\vee$ (céu = nublado) $\vee$ (céu = chuvoso $\wedge$ vento = fraco)

Formulação

- O objetivo é construir um classificador

$$\hat{Y} : X \rightarrow C$$

tal que

$$P(\hat{Y} \neq Y)$$

é tão pequeno quanto possível.

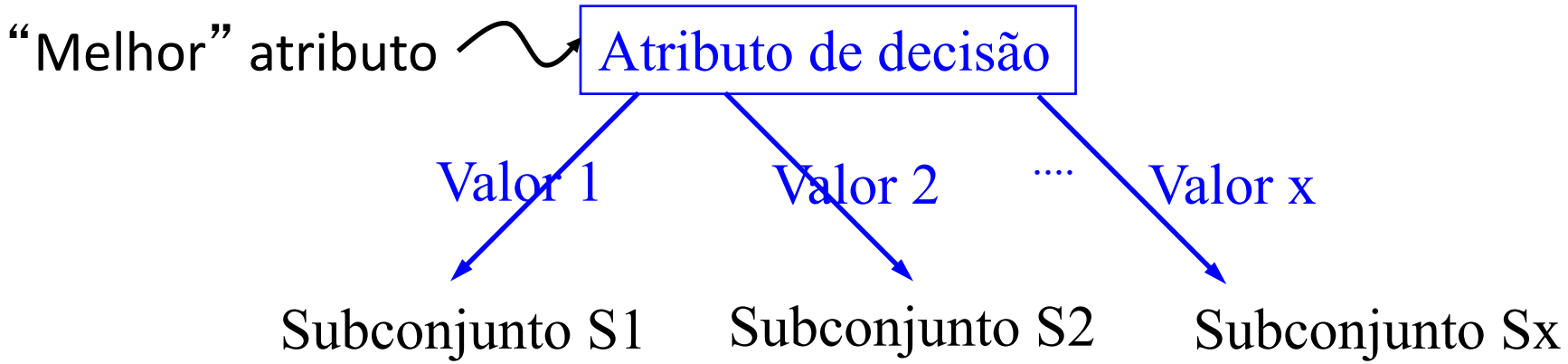
Focaremos no algoritmo ID3 desenvolvido por Ross Quinlan em 1975

Inductive Decision Trees – algoritmo: **`ID3(exemplos, alvo, atributos)`**

- Crie nó *Raiz* para a árvore
- Se todos *exemplos* são positivos, retorne *Raiz* com rótulo +
- Se todos *exemplos* são negativos, retorne *Raiz* com rótulo –
- Se *atributos* for vazio, retorne *Raiz* com rótulo "valor mais comum do *alvo* nos *exemplos (V+comum)*"
- A ← atributo de *atributos* que melhor classifica *exemplos*
- Atributo de decisão da *Raiz* ← A
- Para cada valor possível **vi** de A:
  - Adicionar aresta abaixo de *Raiz* para A=**vi**
  - Definir *exemplos-vi* como o subconjunto de *exemplos* onde A=**vi**
  - Se *exemplos-vi* for vazio, então adicionar nesta aresta um nó-folha com rótulo "(V+comum)"
  - Senão adicionar nesta aresta ***ID3(exemplos-vi, alvo, atributos – {A})***
- Retorne *Raiz*

ID3 – exemplo

Exemplos S: p positivos e n negativos

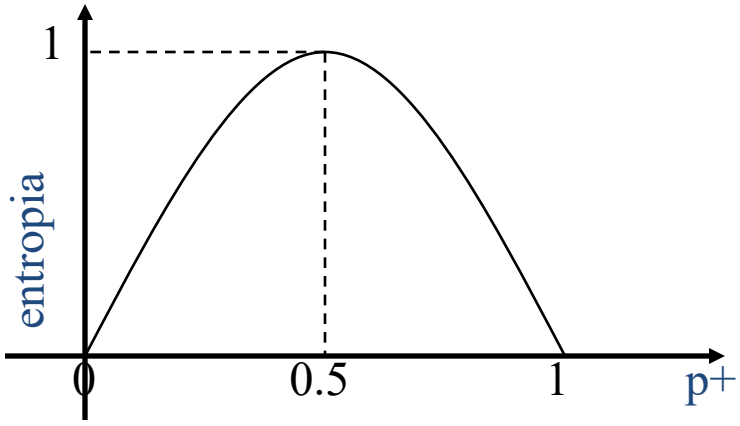


$S = S1 \cup S2 \cup .... Sx$

Qual atributo é o melhor classificador?

- Medida baseada em **ganho de informação**, calculado pela **entropia**
- **Entropia**: medida de "impureza" numa coleção de exemplos de treinamento S
  - Entropia = 0: todos membros da mesma classe
  - Entropia = 1: coleção com mesmo número de + e –

Entropia(S) = – (p+) $\log_2$ (p+) – (p–) $\log_2$ (p–)



p+: proporção de exemplos positivos em S

p – : proporção de exemplos negativos em S

OBS: 0 $\log_2$ 0 = 0

Entropia: exemplo

Entropia(S) = – (p+) log_2 (p+) – (p–) log_2 (p–)

- Ex: se S tem 14 exemplos, sendo 9 positivos e 5 negativos, vem:

  Entropia(S) = – (9/14) $\log_2$ (9/14) –

  – (5/14) $\log_2$ (5/14) = 0.940

- Se atributo pode ter c-valores (não só + e –), então:

$$\text{entropia(S)} = \sum_{i=1}^{c} (- \text{pi} \log_2 \text{pi})$$

OBS: 0 log_2 0 = 0

Ganho de Informação

- Mede a redução esperada na entropia, causada pela partição nos exemplos segundo um atributo.
- Ganho(S,A): ganho de informação de um atributo A, relativo à coleção de exemplos S.

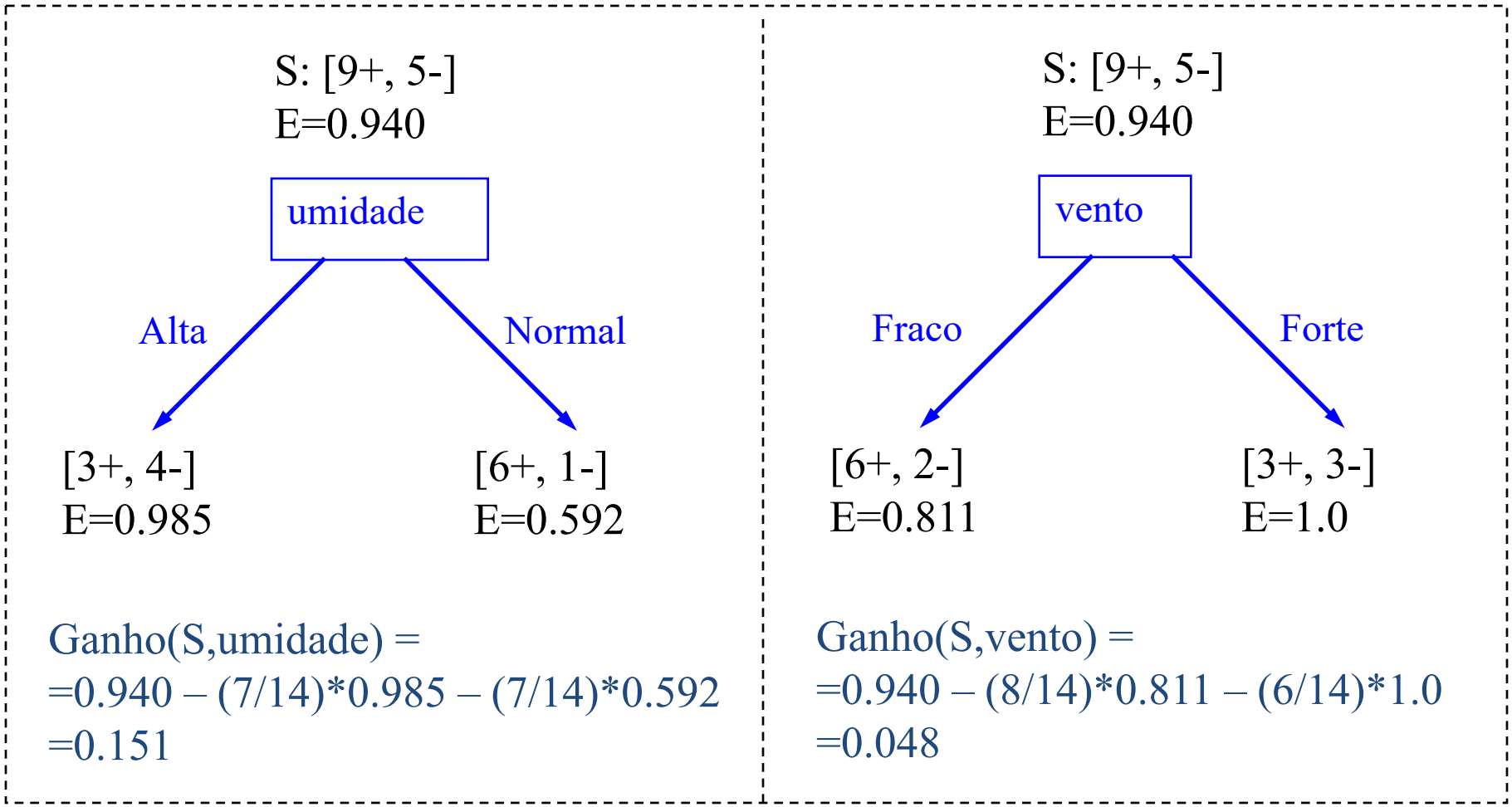
$$\text{Ganho}(S,A) = \text{entropia}(S) - \sum_{v \in \text{valores}(A)} (|Sv| / |S|)\ \text{entropia}(Sv)$$

valores(A): todos possíveis valores do atributo A

Sv: subconjunto de S no qual A tem valor v

$$Sv = \{s \in S \mid A(s) = v\}$$

Ex: qual melhor atributo classificador?

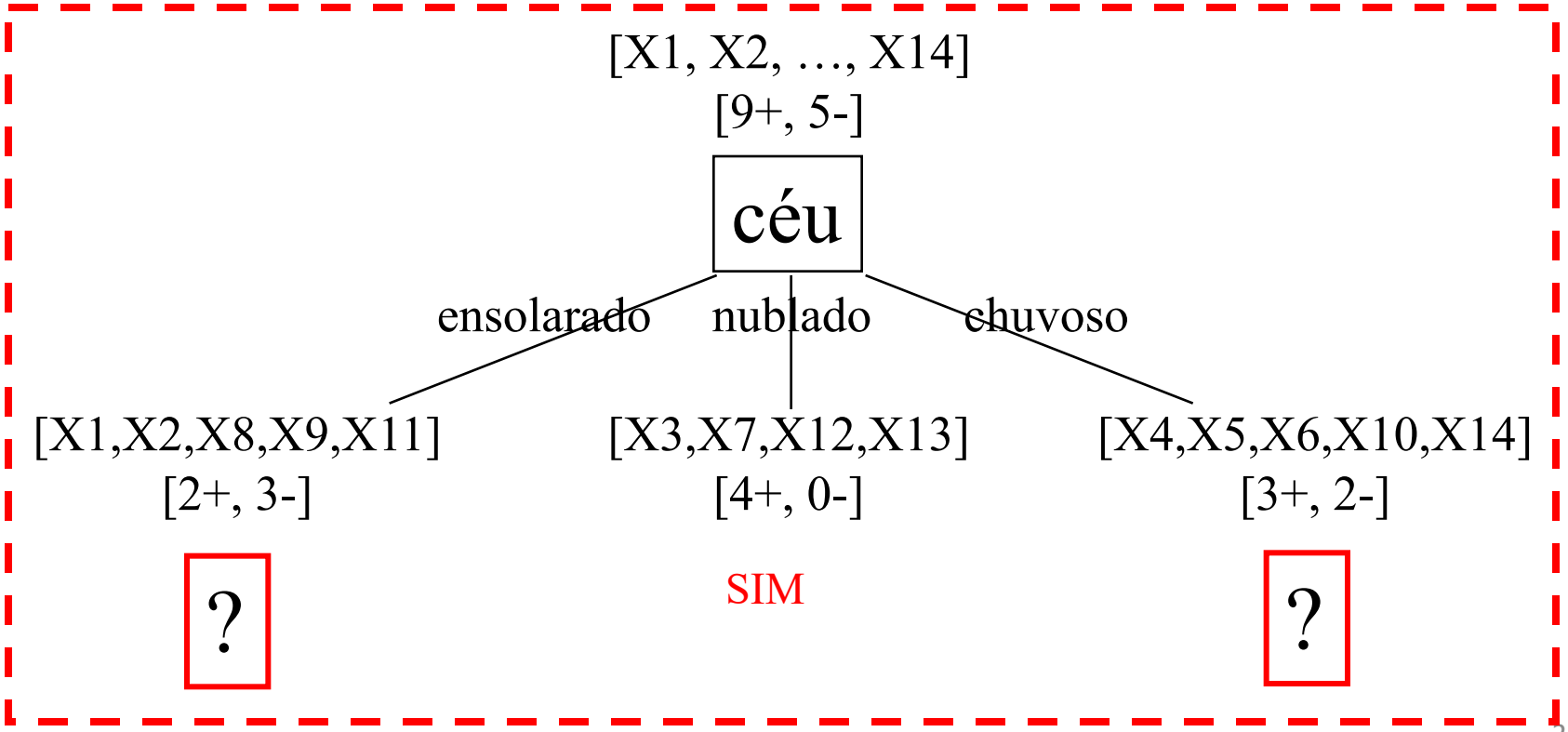


ID3: aprendizado não-incremental

- analisa todos os exemplos para decidir atributo classificador

Construção da árvore de decisão com ID3

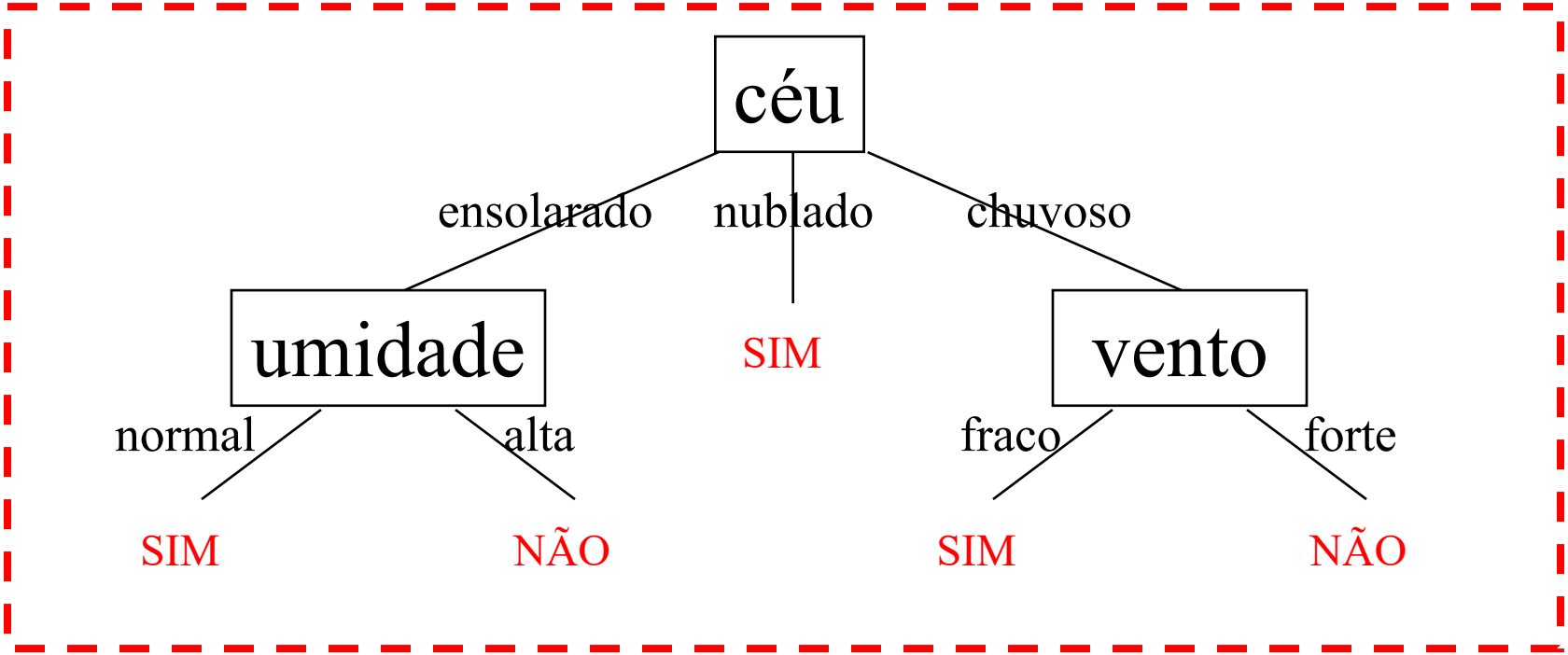
Ganho(S,céu) = 0.246; Ganho(S,umidade) = 0.151
Ganho(S,vento) = 0.048; Ganho(S,temperatura) = 0.029
➔ Para S, céu é melhor (maior ganho)!



Construção da árvore de decisão com ID3

Repete passo para Sv1 = {X1,X2,X8,X9,X11} e Sv2 = {X4,X5,X6,X10,X14}

E assim por diante até chegar às folhas da árvore de decisão.



Características do ID3

- Preferência por árvores pequenas:
  - sua busca no espaço de hipóteses aumenta a árvore somente até o tamanho necessário para classificar o conjunto de exemplos de treinamento disponível.
- Coloca mais perto da raiz aqueles atributos que oferecem o maior ganho de informação.

Problemas gerais

- Estratégia de aumentar a árvore o mínimo necessário pode trazer ***problemas*** quando:
  1. Há ruído nos dados;
  2. Atributos são insuficientes;
  3. Número de exemplos de treinamento é pequeno (não representativo da função buscada)

  - Soluções possíveis:
    - Cada folha é rotulada com a classificação majoritária;
    - Folhas indicam probabilidade de ocorrência de cada classificação (relativo à frequência da classificação).

Overfitting = hiperespecialização

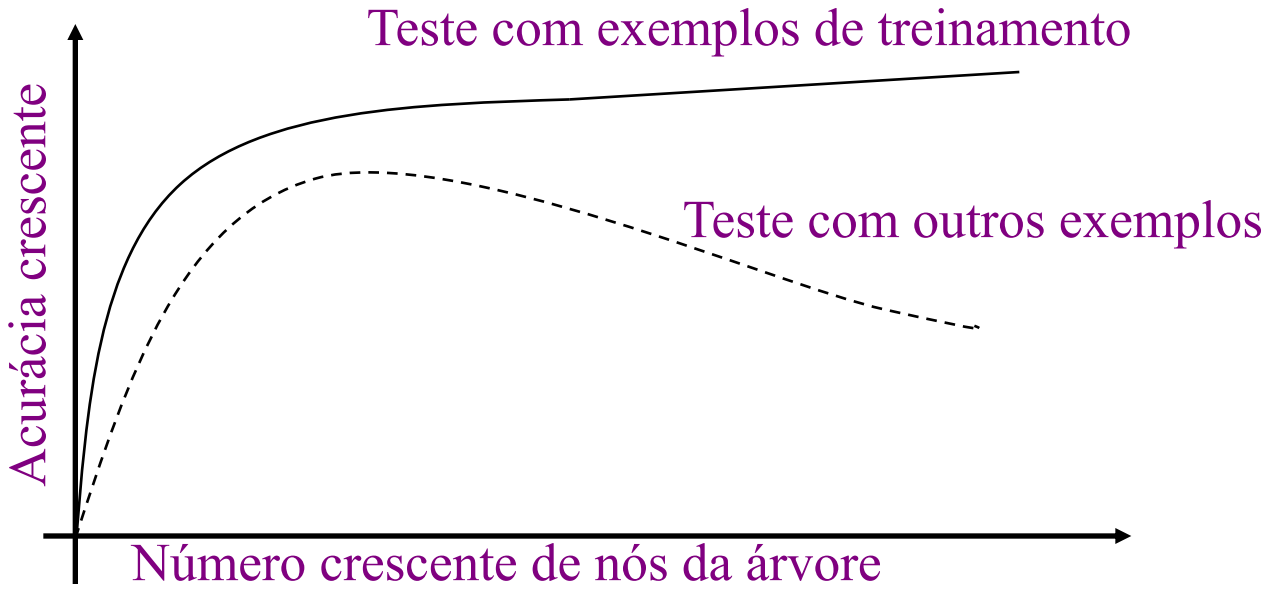
➔ problema de **todos** algoritmos de aprendizado!

Definição: dado um espaço de hipóteses H, uma hipótese $h \in H$ ***overfits*** os dados de treinamento se existir uma outra hipótese $h' \in H$, tal que **h** tem menor erro que **h'** no conjunto de treinamento, ***mas*** **h'** tem menor erro que **h** sobre a distribuição total de instâncias (incluindo instâncias fora do conjunto de treinamento).

➔ *Como detectar atributos irrelevantes?*

➔ *Quão grande deve ser o ganho de informação para que o correspondente atributo seja um nó na árvore?*

Impacto do *overfitting* numa aplicação típica de aprendizado por árvore de decisão:



- Conforme ID3 adiciona mais nós para crescer a árvore de decisão, a acurácia da árvore (medida sobre os exemplos de treinamento) cresce monotonicamente.
- Entretanto, quando medida sobre um conjunto de dados independente do conjunto de treinamento, a acurácia primeiro cresce e, depois, decresce.

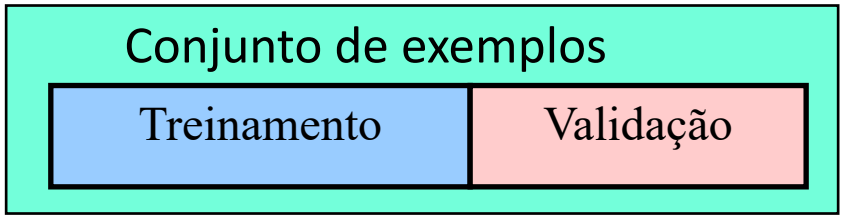
Overfitting – Uma Solução

- Solução 1: Parar de crescer a árvore antes de alcançar o ponto de classificação perfeita dos exemplos de treinamento.

  ➔ mas, quando parar?

  Validação cruzada: tenta estimar quão bem a hipótese corrente irá predizer dados ainda não recebidos (“vistos”).

Validação Cruzada



- Algoritmo:
  1) Divide o conjunto de exemplos em dois subconjuntos: conjuntos de treinamento (CT) e de validação (CV);
  2) Usa indução (ID3) para gerar hipótese H sobre CT;
  3) Mede percentagem de erro de H aplicada à CV;
  4) Quando erro de H em CV aumentar, para construção de H.
  - Repete passos 1-3 com diferentes tamanhos de CV e CT, e tendo elementos escolhidos aleatoriamente
- **Tamanho do conjunto de treinamento:** pode-se calcular a média da percentagem de acerto da hipótese atual em CV e determinar a **curva de aprendizagem** para o domínio em questão.
  - Espera-se que a qualidade da predição cresça com o crescimento do tamanho do conjunto de treinamento.

Overfitting – Outra Solução

- **Solução 2:** Abordagens que provoquem o *overfitting* e **depois poda a árvore** (*post-pruning*)
  - **Método do Erro Reduzido:**
    - considera cada nó como candidato a folha (elimina sub-árvore abaixo dele), com classificação a ele associada como a mais comum;
    - o nó se torna folha (nova árvore, menor) sempre que a acurácia da classificação não diminuir em relação à árvore original, usando o conjunto de validação.

Trees: Pros

- Fácil de explicar
- Fácil de interpretar
- Lida com preditores qualitativos
- Eficiente – teste rápido
- Lida com múltiplas classes
- Pode ser transformada em regras
- Faz seleção de atributos (*feature selection*)
- Geralmente resulta em modelos compactos

Trees: Cons

- Instabilidade – sensível aos pontos de treinamento
- Acurácia
- Grandes amostras de treinamento geram árvores grandes
- Diferentes amostras de treinamento geram árvores diferentes
- Propaga erros pela árvore
- Pode não desempenhar bem quando houver estrutura nos dados que não é bem capturada pelos cortes horizontais e verticais
- Não há formas de capturar as interações entre as variáveis (pois lida com uma variável por vez)

References

- **Russel, S.; Norvig, P. Inteligência Artificial, 2a. Edição, 2004, cap.18.**
- Mitchell, T.M. Machine learning. McGraw-Hill, 1997. Cap. 3
- James, G.; Witten, D.; Hastie, T; Tibshirani, R. An introduction to statistical learning. Springer, 2013. Cap. 8.
- www.wisdom.weizmann.ac.il/~vision/courses/2003_2/**multipleTrees.ppt**
- www.cse.lehigh.edu/~munoz/CSE497/classes/**Storey_DecisionTrees.ppt**
- http://www.saedsayad.com/decision_tree.htm
- https://class.stanford.edu/c4x/HumanitiesScience/StatLearning/asset/trees.pdf