Inteligência Artificial

Aula 19

Profª Bianca Zadrozny

http://www.ic.uff.br/~bianca/ia

Aprendizagem a partir de observações

Capítulo 18 – Russell & Norvig

Seções 18.1 a 18.3

Aprendizagem Indutiva Clássica

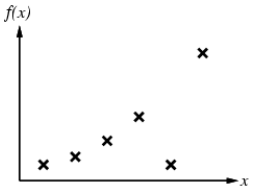
- Recebe como entrada o valor correto de uma **função desconhecida** para **entradas específicas** e tenta recuperar a função.
- Dada uma coleção de exemplos de **f**, retornar uma função **h** que se aproxime de **f**
  - **f** é a função-alvo
  - **h** é a hipótese
  - (x, f(x)) são exemplos de entrada

Exemplos de Aplicações de Aprendizagem

- Detecção de spam
  - $\boldsymbol{x}$: descreve o conteúdo de mensagens de e-mail, em geral utiliza-se uma dimensão por palavra, além de outros atributos como número de palavras erradas, número de links, etc.
  - $\boldsymbol{f(x)}$: indica se a mensagem é spam ou não (em geral fornecido pelos usuários).
- Reconhecimento de caracteres
  - $\boldsymbol{x}$: pixels da imagem em que o caracter aparece.
  - $\boldsymbol{f(x)}$: indica o caracter.
- Diagnóstico de doenças
  - $\boldsymbol{x}$: descreve o resultado de exames do paciente.
  - $\boldsymbol{f(x)}$: indica se o paciente tem a doença ou não.
- Classificação de proteínas
  - $\boldsymbol{x}$: seqüência de aminoácidos da proteína.
  - $\boldsymbol{f(x)}$: indica a família a qual a proteína pertence.

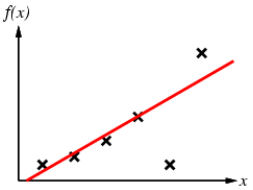
Aprendizagem Indutiva

- Construir/ajustar *h* que concorde com *f* no conjunto de treinamento
- ***h* é consistente se concorda com todos os exemplos dados**
- E.g., ajuste de curva:



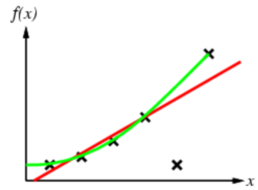
Aprendizagem indutiva

- Construir/ajustar *h* que concorde com *f* no conjunto de treinamento
- ***h* é consistente se concorda com todos os exemplos dados**
- E.g., ajuste de curva:



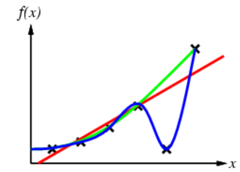
Aprendizagem indutiva

- Construir/ajustar *h* que concorde com *f* no conjunto de treinamento
- ***h* é consistente se concorda com todos os exemplos dados**
- E.g., ajuste de curva:



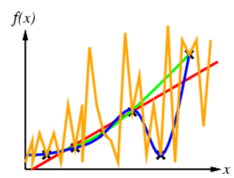
Aprendizagem indutiva

- Construir/ajustar *h* que concorde com *f* no conjunto de treinamento
- ***h* é consistente se concorda com todos os exemplos dados**
- E.g., ajuste de curva:



Aprendizagem indutiva

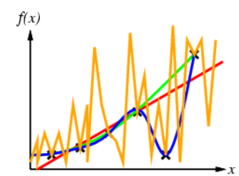
- Construir/ajustar *h* que concorde com *f* no conjunto de treinamento
- ***h* é consistente se concorda com todos os exemplos dados**
- E.g., ajuste de curva:



Aprendizagem indutiva

- Como escolher entre as várias hipóteses disponíveis?
  - **Lâmina de Ockam (*Ockam's razor*)**: prefira a hipótese ***mais simples*** consistente com os dados

hipóteses que são mais complexas que os dados estão deixando de extrair algum padrão dos dados!



Espaço de hipótese

- Deve-se ter em mente que a possibilidade de encontrar uma hipótese simples e consistente depende do espaço de hipótese escolhido.
  - Ex. Ax + b + c sen x
- Um problema de aprendizagem é realizável se o espaço de hipótese contém a função verdadeira
  - nem sempre é possível de decidir pois a função verdadeira não é conhecida

Aprendizagem em árvores de decisão

- **Indução de árvores de decisão:** uma das forma mais simples (e mais bem sucedidas) de aprendizagem automática.
- **Árvore de decisão**: toma como entrada um objeto ou situação descritos por um conjunto de **atributos** e retorna uma decisão -- valor de saída previsto, dada uma entrada.

Árvores de decisão

Exemplo: decidir se devo esperar por uma mesa em um restaurante, dados os atributos:

1. ***Alternate***: há um restaurante alternativo na redondeza?
2. ***Bar***: existe um bar confortável onde se esperar?
3. ***Fri/Sat***: hoje é sexta ou sábado ?
4. ***Hungry***: estou com fome?
5. ***Patrons***: numero de pessoas no restaurante (***None***, ***Some***, ***Full***)
6. ***Price***: faixa de preços ($, $$, $$$)
7. ***Raining***: está a chover?
8. ***Reservation***: temos reserva?
9. ***Type***: tipo do restaurante (***French***, ***Italian***, ***Thai***, ***Burger***)
10. ***WaitEstimate***: tempo de espera estimado (0-10, 10-30, 30-60, >60)

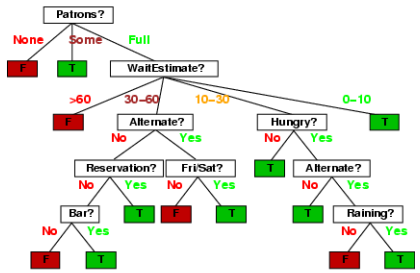
Induzindo árvores de decisão a partir de exemplos

- Exemplos descritos por valores de atributos (discretos, ou contínuos)
- E.g., exemplos de situações em que o autor do livro não esperará por uma mesa:

Example	Attributes									Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

- Classificação dos exemplos em positivo (T) ou negativo (F)

Árvores de decisão



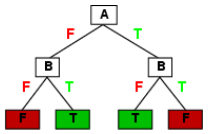
Árvores de decisão

- Uma árvore de decisão chega a uma decisão executando uma **sequência** de testes.
- Cada **nó interno** da árvore corresponde a um **teste** do valor de uma das propriedades, e as ramificações a partir do nó são identificadas com os valores possíveis do teste.
- Cada **nó de folha** especifica o **valor** a ser retornado se aquela folha for alcançada.

Expressividade

- Qualquer função booleana pode ser escrita como uma árvore de decisão

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



- Trivialmente, há uma árvore de decisão consistente para qualquer conjunto de treinamento com um caminho para cada exemplo.
  - Isso seria uma representação exponencialmente grande
- Devemos procurar por árvores de decisão mais compactas.

Algoritmo de Aprendizagem de Árvores de Decisão (informal)

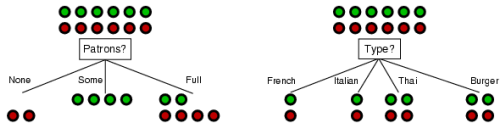
- Se existem alguns exemplos positivos e alguns negativos, escolha o **melhor** atributo para dividi-los.
- Se todos os exemplos restantes forem positivos (ou todos negativos), então terminamos: podemos responder *Sim* ou *Não*.
- Se não resta nenhum exemplo, nenhum exemplo desse tipo foi observado. Então retorna-se um valor-padrão calculado a partir da classificação de maioria no pai do nó.

Algoritmo de Aprendizagem de Árvores de Decisão (informal)

- Se não resta nenhum atributo, mas há exemplos positivos e negativos, temos um problema.
- Isso acontece quando
  - alguns dados estão incorretos; dizemos que existe **ruído** nos dados;
  - os atributos não fornecem informações suficientes;
  - o domínio não é completamente determinístico.
- Saída simples: utilizar uma votação pela maioria

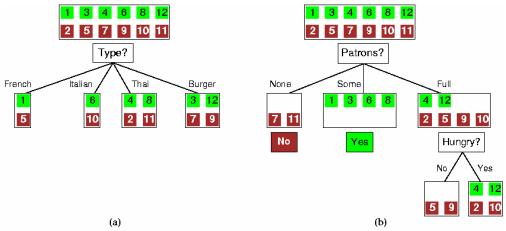
Escolha de atributos

- Idéia: um bom atributo divide os exemplos em subconjuntos que (preferivelmente) são “todos positivos" ou ”todos negativos"



- *Patrons?* é um atributo melhor do que *Type* para ser raiz.

Escolha de atributos



Como definir o que é um atributo melhor ?

- A escolha de atributos deve minimizar a profundidade da árvore de decisão;
  - Escolher um atributo que vá o mais longe possível na classificação exata de exemplos;
  - Um atributo perfeito divide os exemplos em conjuntos que são todos positivos ou todos negativos.
- Solução: medir os atributos a partir da quantidade esperada de informações fornecidas por ele.

Como definir o que é um atributo melhor ?

- O atributo “patrons” não é perfeito, mas é bastante bom; o atributo “type” é completamente inútil.
- Precisamos definir uma medida formal de “bastante bom” e “bastante ruim”.
- A medida deve ter seu valor máximo quanto o atributo for perfeito e seu valor mínimo quando o atributo for inútil.

Utilizando teoria da informação

- A teoria da informação mede o conteúdo de informação em bits.
  - um bit é o suficiente para responder a uma pergunta sim/não sobre a qual não se tem nenhuma idéia a respeito (como o lançamento de uma moeda)

Utilizando teoria da informação

- Em geral, se as respostas possíveis v_i tem probabilidade $P(v_i)$, então o conteúdo de informação ***I*** da resposta real é dado por:

  Conteúdo de Informação (Entropia):

  $I(P(v_1), \ldots, P(v_n)) = \Sigma^n_{i=1} -P(v_i) \log_2 P(v_i)$
- No caso do lançamento de uma moeda imparcial:
- $I(1/2,1/2) = -1/2 \log_2 1/2 - 1/2 \log_2 1/2 =$ **1 bit**

Utilizando teoria da informação

Para um conjunto de treinamento contendo ***p*** exemplos positivos e ***n*** exemplos negativos :

$$I(\frac{p}{p+n}, \frac{n}{p+n}) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

No exemplo do restaurante temos n=p=6, portanto precisamos de 1 bit de informação no total.

Ganho de Informação

- Dado um atributo A, podemos medir quantas informações ainda precisamos *depois* deste atributo;
  - Qqr atributo A divide o conjunto de treinamento E em subconjuntos $E_1, \ldots, E_v$ de acordo com seus valores para A, onde A pode ter v valores distintos.
  - Cada subconjunto E_i tem p_i exemplos positivos e n_i exemplos negativos;
  - Assim seguindo essa ramificação, precisaremos de

    $I(p_i/(p_i + n_i), n_i/(p_i + n_i))$

    bits de informação para responder a pergunta.

Ganho de informação

- Um exemplo escolhido ao acaso a partir do conjunto de treinamento tem o i-esimo valor para o atributo com probabilidade $(p_i + n_i)/(p+n)$ ("peso do atributo").
- e assim, em média, depois de testar o atributo A, temos:

$$remainder(A) = \sum_{i=1}^{v} \frac{p_i+n_i}{p+n} I(\frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i})$$

bits de informação para classificar o exemplo...

Ganho de informação

- Um exemplo escolhido ao acaso a partir do conjunto de treinamento tem o i-esimo valor para o atributo com probabilidade $(p_i + n_i)/(p+n)$ ("peso do atributo").
- e assim, em média, depois de testar o atributo A, temos:

$$remainder(A) = \sum_{i=1}^{v} \frac{p_i+n_i}{p+n} I(\frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i})$$

bits de informação para classificar o exemplo...

"Dentre os exemplos deste atributo, qual é o grau de discernimento to dele."

Ganho de informação

- O ganho de informação a partir do teste deste atributo é a diferença entre o <u>*requisito de informações original*</u> e o <u>*novo requisito*</u>:

$$IG(A) = I(\frac{p}{p+n}, \frac{n}{p+n}) - remainder(A)$$

- A heurística é então escolher o atributo com o maior valor de ganho de informação (IG).

Ganho de informação

Para o conjunto de treinamento, $p = n = 6$, $I(6/12, 6/12) = 1$ bit

Considerando os atributos *Patrons* e *Type*, e os outros tb:

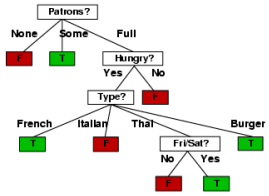
$$IG(Patrons) = 1 - [\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I(\frac{2}{6}, \frac{4}{6})] = .0541 \text{ bits}$$

$$IG(Type) = 1 - [\frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4})] = 0 \text{ bits}$$

Patrons possui o maior valor de IG de todos os atributos e, portanto, é escolhido como raiz da árvore de decisão aprendida pelo algoritmo DTL

Resolvendo o exemplo

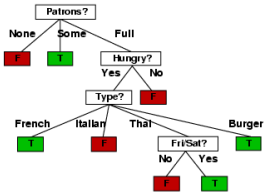
- Árvore de decisão aprendida dos 12 exemplos:



- Substancialmente mais simples do que a árvore "verdadeira";
- Não há nenhuma razão para uma solução mais complexa (e.g incluindo *Rain* e *Res*), pois todos os exemplos já foram classificados.

Resolvendo o exemplo

- Árvore de decisão aprendida dos 12 exemplos:



- Com mais exemplos seria possível induzir uma árvore mais semelhante à árvore original;
- Esta árvore nunca viu um exemplo de espera 0-10
  - portanto, pode cometer um engano...
  - Como avaliar o desempenho do algoritmo?

Avaliação de desempenho

- Como sabemos que $h \approx f$?
  - Um algoritmo de aprendizagem é bom se produz hipóteses que fazem um bom trabalho de previsão de classificações de exemplos não vistos
- Método de validação:
  1. Coletar um grande número de exemplos;
  2. Dividi-lo em **conjunto de treinamento** e **conjunto de teste;**
  3. Aplicar o algoritmo de aprendizagem ao conjunto de treinamento, gerando uma hipótese ***h***
  4. Medir a porcentagem de exemplos no conjunto de teste que são corretamente classificados por ***h***
  - Repetir as etapas 1 a 4 para diferentes tamanhos de conjuntos de treinamento e diferentes conjuntos de teste de cada tamanho selecionados aleatoriamente

Avaliação de desempenho

- Experimente *h* em novos conjuntos de teste.
- Curva de aprendizagem = % de classificações corretas sobre o conjunto de teste como uma função do tamanho do conjunto de treinamento