Laboratório de Inteligência Computacional (LABIC)

http://labic.icmc.sc.usp.br/

Probably Approximately Correct (PAC)

Maria Carolina Monard & Gustavo Batista

Universidade de São Paulo
Campus de São Carlos - SP, Brasil
email: {mcmonard, gbatista}@icmc.sc.usp.br

Apresentação baseada na dissertação de mestrado de Gustavo Batista

O Modelo Probably Approximately Correct (PAC)

- A performance de um classificador é geralmente medida através de sua taxa de erro.
- Se um número ilimitado de casos para treinar está disponível então a taxa de erro aparente será a taxa de erro verdadeira.
- Infelizmente, conjuntos reais de exemplos são finitos e geralmente pequenos.

- Desta forma, uma indagação importante a ser respondida é

  Quantos casos são necessários para que a taxa de erro aparente se torne efetivamente a taxa de erro verdadeira?

- Algumas respostas para indagações como esta são fornecidas pela área denominada *Teoria Computacional de Aprendizado*[1].

- A idéia geral é a seguinte:

  – Para qualquer hipótese que estiver *muito errada* com alta probabilidade, após processar um pequeno número de exemplos, este fato será observado pois essa hipótese fará predições incorretas.

[1] Computational Learning Theory

– Assim, para qualquer hipótese que é consistente com um número suficientemente grande de exemplos de treinamento, é pouco provável que ela esteja *muito errada*, ou seja, essa hipótese deve ser Provavelmente Aproximadamente Correta[2].

– Um ponto importante a ser considerado é a relação entre o conjunto de treinamento e o conjunto de teste, pois interessa que a hipótese proposta seja aproximadamente correta também no conjunto de teste, e não somente no conjunto de treinamento.

– Nesta teoria é assumido que tanto o conjunto de treinamento quanto o conjunto de teste são retirados aleatoriamente do mesmo conjunto de exemplos usando a mesma distribuição de probabilidade.

[2]Probably Aproximately Correct – PAC

– Esta condição, denominada de *estacionária*, é muito importante em aprendizado-PAC pois permite realizar uma conexão entre o passado — conjunto de exemplos de treinamento utilizado pelo algoritmo para aprender uma hipótese — e o futuro, isto é, classificar corretamente novos exemplos. Ou seja,

> *o modelo PAC define aprendizado em termos do poder preditivo da hipótese encontrada pelo algoritmo de aprendizado.*

- Como mostrado a seguir, é possível aplicar ao algoritmo esta medida de sucesso porque é considerado que os exemplos de treinamento são retirados independentemente de uma distribuição de probabilidades fixa D e seu poder preditivo é também medido em relação a essa mesma distribuição D.
- Com as considerações acima, é possível responder à seguinte pergunta

  Quantos exemplos de treinamento são necessários para que a hipótese h encontrada pelo algoritmo seja PAC?

- Sejam
  - $\mathbf{X} = \{X_1, X_2, X_3, \ldots\}$ o conjunto de todos os possíveis exemplos;
  - D a distribuição da qual os exemplos são retirados;
  - $\mathbf{H} = \{h_1, h_2, h_3, \ldots\}$ o conjunto das possíveis hipóteses;
  - m o número de exemplos no conjunto de treinamento.
- Observar que $\mathbf{X}$ e $\mathbf{H}$ podem também ser finitos.

- Assumindo que a *verdadeira* função f_v é um elemento de $\mathbf{H}$, isto é, $f_v \in \mathbf{H}$, o erro em um exemplo $X_i \in \mathbf{X}$ da hipótese h em relação a verdadeira função f_v dada a distribuição D sobre os exemplos pode ser definido como a probabilidade de h ser diferente de f_v no exemplo $X_i \in \mathbf{X}$

$$erro(h) = P(h(X_i) \neq f_v(X_i) | X_i \, retirado \, de \, D)$$

- A hipótese h é denominada *aproximadamente correta* se

$$erro(h) \leq \epsilon$$

- ϵ uma constante pequena. A idéia geral é mostrar que após o algoritmo ter visto m exemplos, todas as hipóteses consistentes serão, com alta probabilidade, aproximadamente corretas.

- A Figura 1 ilustra esta idéia. As hipóteses aproximadamente corretas são aquelas no espaço de hipóteses $\mathbf{H}$ que se encontram *perto*, a uma distância ϵ, da verdadeira hipótese f_v, isto é, dentro de um círculo — denominado ϵ-*círculo* — cujo centro encontra-se a verdadeira função f_v.

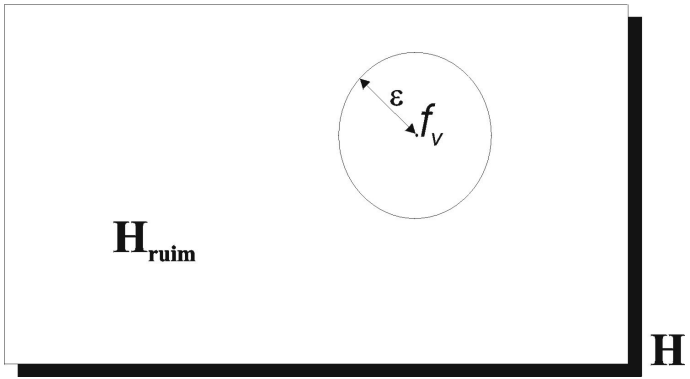

Figura 1: ϵ Círculo Delineando o Conjunto de Hipóteses PAC

- Assim, o conjunto $\mathbf{H}$ de hipóteses pode ser considerado como o conjunto de hipóteses aproximadamente corretas, que são aquelas dentro do ϵ-*círculo*, e as hipóteses restantes que serão denominadas $\mathbf{H}_{ruim}$.
- Para todo $h_r \in \mathbf{H}_{ruim}$, como $erro(h_r) > \epsilon$, pois h_r encontra-se fora do ϵ-*círculo*, então a probabilidade de h_r cobrir qualquer exemplo é

$$P(h_r \; cobrir \; um \; exemplo) \leq (1 - \epsilon)$$

- Então a probabilidade de *uma* hipótese *ruim* $h_r \in \mathbf{H}_{ruim}$ ser consistente com os primeiros m exemplos é

$$P(h_r \; cobrir \; m \; exemplos) \leq (1 - \epsilon)^m$$

- Porém, para $\mathbf{H}_{ruim}$ conter hipóteses consistentes pelo menos uma das hipóteses em $\mathbf{H}_{ruim}$ deve ser consistente.
- A probabilidade de que isto aconteça está limitada pela soma das probabilidades individuais.

$$\begin{aligned} P(\mathbf{H}_{ruim}\ conter\ uma\ hipótese\ consistente) &\leq |\mathbf{H}_{ruim}|(1-\epsilon)^m \\ &\leq |\mathbf{H}|(1-\epsilon)^m \end{aligned}$$

- Interessa então analisar as condições para reduzir a probabilidade deste evento tal que seja menor do que um número pequeno δ, ou seja

$$|\mathbf{H}|(1-\epsilon)^m \leq \delta$$

- Aplicando ln obtêm-se

$$\begin{aligned} ln(|\mathbf{H}|) + m\ ln(1-\epsilon) &\leq ln(\delta) \\ m &\leq \frac{ln(\delta) - ln(|\mathbf{H}|)}{ln(1-\epsilon)} \\ m &\geq \frac{1}{ln(1-\epsilon)}(-ln(\delta) + ln(|\mathbf{H}|) \\ m &\geq \frac{1}{ln(1-\epsilon)}(ln\frac{1}{\delta} + ln(|\mathbf{H}|) \end{aligned}$$

- Utilizando a inequação $(1-\epsilon) \leq e^{-\epsilon}$ obtém-se

$$m \geq \frac{1}{\epsilon}(ln\frac{1}{\delta} + ln|\mathbf{H}|) \qquad (1)$$

- Esta inequação mostra que se um algoritmo de aprendizado de máquina encontrar uma hipótese que é consistente com este número m de exemplos, então com probabilidade no mínimo de $1 - \delta$ essa hipótese tem erro máximo de ϵ
- ϵ é denominado *parâmetro de erro* e δ *parâmetro de confiança*.
- Ambos controlam os dois tipos de fracassos aos quais é susceptível o algoritmo de aprendizado no modelo PAC.
- O número requerido de exemplos m em função de ϵ e δ é denominado *complexidade da amostra* do espaço de estados.

- O critério mínimo não trivial a ser exigido do algoritmo é

$$0 < \epsilon < \frac{1}{2}$$
$$0 < \delta < \frac{1}{2}$$

- Considerando a inequação (1), aparentemente, a única incógnita a ser determinada é o tamanho do espaço de hipóteses $|\mathbf{H}|$.

- No caso de $|\mathbf{H}|$ ter cardinalidade finita e ignorando a eficiência computacional do algoritmo de aprendizado, ou, em outras palavras, assumindo que o algoritmo de aprendizado possui infinito poder computacional para processar a amostra m de exemplos, então, a inequação (1) afirma que se o algoritmo utilizar uma amostra finita $m = O(\frac{1}{\epsilon} log \frac{|\mathbf{H}|}{\delta})$ de exemplos e encontrar qualquer hipótese h consistente com esses exemplos[3] então h verifica o modelo PAC.
- Há uma diferença fundamental entre um conceito (que pode ser um conjunto ou uma função booleana, por exemplo) e sua representação (que é a codificação simbólica desse conjunto ou função).

[3]Observar que essa hipótese h pode até ser encontrada por busca exaustiva.

- Por exemplo, considerando a classe de conceitos definidos pelas atribuições que satisfazem uma função booleana f_b, um conceito dessa classe tanto pode ser representado pela própria fórmula f_b, pela sua tabela verdade, por uma árvore de decisão, bem como por outra fórmula booleana f'_b equivalente a f_b.
- Entretanto, ainda que todas elas sejam representações do mesmo conceito, o tamanho da representação de cada uma delas pode ser muito diferente. Por exemplo, a função booleana de paridade[4]

$$f(x_1, x_2, \ldots, x_n) = x_1 \oplus x_2 \oplus, \ldots, x_n$$

retorna 1 se e somente se um número par de entradas for 1.

[4] onde $\oplus$ denota a operação de ou-exclusivo.

- Esta função pode ser computada por um circuito digital – $h_{circuito}$ – constituído de portas $\neg$, $\vee$ e $\wedge$ cujo tamanho é limitado por um polinômio de grau n.
- Entretanto, a mesma função representada na Forma Normal Disjuntiva – h_{FND} – ou por uma árvore de decisão – h_{ad} – requer tamanho exponencial em n.
- Ou seja, as três hipóteses $h_{circuito}$, h_{FND} e h_{ad} são representações do mesmo conceito, no entanto

$$\begin{aligned} tamanho(h_{circuito}) &= O(n) \\ tamanho(h_{FND}) &= O(2^n) \\ tamanho(h_{ad}) &= O(2^n) \end{aligned}$$

- Problema análogo acontece com a função booleana majoritária[5] que retorna 1 se a maioria das entradas for 1.
- Em cada um desses exemplos foi fixado um esquema de representação (circuito digital, FND, árvore de decisão, etc . . .) para codificar conceitos, mas o tamanho da codificação (por exemplo número de portas de um circuito digital, número de literais de uma conjunção de funções booleanas, número de nós de uma árvore de decisão, etc . . .) pode ser muito diferente para esquemas diferentes.
- Em princípio, o modelo de aprendizado PAC somente leva em conta os exemplos e sua classificação. Ele não possui informação referente ao esquema de representação que está sendo utilizado pelo algoritmo.

[5] Majority function

- Entretanto, como mostrado pelos exemplos citados, o esquema de representação utilizado pelo algoritmo é muito importante já que o tempo de execução do algoritmo também depende desse esquema.
- Uma pergunta que surge é a seguinte

  Existe algum esquema de representação que é eficiente para todas as classes de funções?

- A resposta é negativa.

- Por exemplo, considerando o conjunto de todas as funções booleanas com n atributos

$$F_n = \{f_{n_1}, f_{n_2}, f_{n_3}, \ldots, f_{n_k}, \ldots\}$$

  definidas pela sua tabela verdade, então a cardinalidade $|F_n|$ desse conjunto é o número de tabelas verdade diferentes.

- Cada tabela verdade tem 2^n filas já que cada entrada está definida por n atributos.

- Assim, pode-se considerar que a coluna $n+1$ que especifica o valor dessa função – classe – está representada por um número de 2^n bits que define a função.

- Portanto, independentemente do esquema de representação utilizado, serão necessários, para algumas funções (na realidade para a maioria delas) 2^n bits para representá-las.
- Dessa forma, conclui-se que existem 2^{2^n} — um número exponencial —, de funções booleanas diferentes com n atributos, ou seja

$$|F_n| = 2^{2^n}$$

- Por exemplo, com somente $n = 3$ atributos booleanos existem $2^8 = 256$ funções booleanas diferentes — Tabela 1.

$\mathbf{X}$	x_1	x_2	x_3	f_{3_1}	f_{3_2}	f_{3_3}	$\ldots$	$f_{3_{10}}$	$\ldots$	$f_{3_{256}}$
$\mathbf{X_1}$	0	0	0	0	0	0		0		1
$\mathbf{X_2}$	0	0	1	0	0	0		1		1
$\mathbf{X_3}$	0	1	0	0	0	0		0		1
$\mathbf{X_4}$	0	1	1	0	0	0		0		1
$\mathbf{X_5}$	1	0	0	0	0	0		0		1
$\mathbf{X_6}$	1	0	1	0	0	0		0		1
$\mathbf{X_7}$	1	1	0	0	0	1		0		1
$\mathbf{X_8}$	1	1	1	0	1	0		1		1

Tabela 1: Funções Booleanas Diferentes com $n = 3$ atributos

- Somente duplicando o número de valores booleanos para $n = 6$, o número de funções booleanas diferentes cresce para $2^{64} = 1.84 * 10^{19}$
- Assim, considerando o caso geral do conjunto de todas as funções booleanas com n atributos, e com base na inequação (1), a complexidade da amostra m é

$$m \geq \frac{1}{\epsilon}(ln\frac{1}{\delta} + ln2^{2^n})$$
$$m \geq \frac{1}{\epsilon}(ln\frac{1}{\delta} + 2^n ln2)$$

portanto, para ϵ e δ fixos tem-se

$$m = O(2^n)$$

- Como o número de possíveis exemplos $\mathbf{X}$ é também 2^n, isto significa que qualquer algoritmo de aprendizado para o espaço de hipóteses de todas as funções booleanas não fará melhor que uma tabela *lookup* se simplesmente retornar uma hipótese que é consistente com todos os exemplos conhecidos.
- A Tabela 2 mostra o número de exemplos necessários em função de alguns valores de n, ϵ e δ para o espaço de hipóteses de todas as funções booleanas.

n	ϵ	δ	m
5	0.1	0.1	245
5	0.1	0.01	268
5	0.01	0.1	2450
5	0.01	0.01	2680
10	0.1	0.1	7123
10	0.1	0.01	7146
10	0.01	0.1	71230
10	0.01	0.01	71460

Tabela 2: Número de Exemplos para Aprendizado PAC no Espaço de Todas as Funções Booleanas

- A Figura 5 mostra graficamente o número mínimo de exemplos, m, em função do número de atributos, n e δ para $\epsilon = 0.01$ e 0.001.

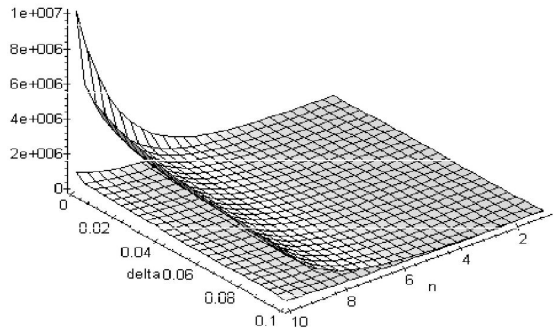

Figura 2: Número de Exemplos, m, em Função de n e δ para $\epsilon = 0.01$ e 0.001

- Desta forma, o algoritmo não terá capacidade de aprender a menos que considere um espaço de hipótese mais restrito.
- Mas, se o espaço de hipótese for restringido, pode acontecer que a verdadeira função f_v não faça parte desse espaço.
- De qualquer forma, pode-se pensar que na maioria dos casos não é necessário utilizar todo o poder de expressão das funções booleanas, e limitar o aprendizado a linguagens de descrição mais restritas.
- Por fim, deve-se ressaltar que a análise PAC é uma análise de pior caso.

- Para todas as possíveis distribuições resultantes de um conjunto de amostras, ela garante que os resultados de classificação serão corretos com uma pequena margem de erro.
- Embora este tipo de análise forneça limites teóricos interessantes para taxas de erro, mesmo para classificadores simples os resultados indicam que um grande número de casos é necessário para garantir a performance.
- Baseados nestes resultados teóricos, pode-se ficar desencorajado em tentar estimar a taxa de erro verdadeira para um classificador. Mas resultados melhores podem ser obtidos para problemas reais, uma vez que dada uma amostra de uma simples população, a tarefa está em estimar a taxa de erro para aquela população, não para todas as possíveis populações.

- Este tipo de análise requer muito menos casos, uma vez que uma única distribuição da população é considerada. Além disso, em vez de todos os casos serem usados para estimar a taxa de erro verdadeira, os casos podem ser particionados em dois grupos, alguns utilizados para projetar o classificador, outros para testar o classificador.
- Apesar de que esta forma de análise não fornece garantias de performance para todas as possíveis distribuições, ela fornece uma estimativa da taxa de erro verdadeira para a população sendo considerada.
- Ao contrário da análise PAC, o número de casos necessários, neste outro tipo de análise, para o conjunto de teste é surpreendentemente pequeno.