Padronização da Sintaxe e Informações sobre Regras Induzidas a Partir de Algoritmos de Aprendizado de Máquina Simbólico[1]

RONALDO CRISTIANO PRATI
JOSÉ AUGUSTO BARANAUSKAS
MARIA CAROLINA MONARD (ORIENTADORA)

USP — Universidade de São Paulo
ICMC — Instituto de Ciências Matemáticas e Computação
LABIC — Laboratório de Inteligência Computacional
Av. do Trabalhador São-Carlense, 400
Cx Postal 668 — CEP 13560-970 — São Carlos (SP)
{prati,jaugusto,mcmonard}@icmc.sc.usp.br

Resumo: O uso de algoritmos de Aprendizado de Máquina para tarefas de Aquisição de Conhecimento é um trabalho empírico, ou seja, trabalha com a experimentação e comparação de diversos algoritmos na tentativa de se descobrir um que melhor se adapte a um domínio específico. No entanto, a comparação entre esses algoritmos, para outros indicadores que não a precisão, não é uma tarefa trivial ou automática, pois cada algoritmo adota uma forma de representação de conhecimento diferente. Em geral, é possível transformar um classificador, obtido através de um algoritmo de aprendizado simbólico, o qual descreve o conhecimento induzido em uma forma diretamente interpretável por seres humanos, para um conjunto de regras. Essas regras podem ser avaliadas em conjunto, comparando o desempenho de cada classificador em relação a outros classificadores como uma "caixa preta", ou cada regra pode ser avaliada individualmente, quanto à qualidade, interessabilidade, novidade, entre outras medidas objetivas. Neste trabalho definimos uma sintaxe padrão de regras para representar classificadores simbólicos e implementamos uma biblioteca de ferramentas que transformam os classificadores simbólicos induzidos por algoritmos de Aprendizado de Máquina para essa sintaxe padrão. Também foi implementada uma biblioteca de ferramentas que calcula, a partir das três possíveis formas de aplicar um conjunto de regras, um a conjunto mínimo de valores pelo qual é possível derivar a maiorias das medidas de avaliação de regras propostas na literatura.

Palavras-Chave: Aprendizado de Máquina, Descoberta de Conhecimento, Avaliação de Regras

1 Introdução

Durante as últimas décadas, houve uma verdadeira explosão nas tecnologias computacionais e da informação. Com isso, uma grande quantidade de dados vem sendo gerada nas mais diferentes áreas do conhecimento humano, tais como medicina, biologia, finanças e marketing. O desafio de entender esses dados levou ao desenvolvimento de novas ferramentas no campo da estatística, e o aparecimento de novas áreas de pesquisa, tais como Mineração de Dados — DM [2] —, Redes Neurais Artificiais — RNA — Aprendizado de Máquina — AM — e Bioinformática.

[1] Trabalho realizado com apoio do CNPq e da FAPESP
[2] *Data Mining*

Um sistema de AM supervisionado é um programa (*indutor*) capaz de induzir uma descrição de um conceito (*classificador*) a partir de um conjunto de exemplos conhecidos e previamente rotulados com as suas respectivas classes. Em outras palavras, o classificador é uma generalização dos exemplos fornecidos ao indutor. O objetivo principal de um classificador é, dado um novo exemplo cuja classe é desconhecida, predizer a sua classe.

Durante muitos anos, muitos algoritmos de AM foram desenvolvidos, alguns utilizando forte embasamento teórico, empírico, ou uma combinação de ambos. Esses sistemas têm sido desenvolvidos utilizando diferentes paradigmas de aprendizado, tais como estatístico, conexionista, *instance-based*, genético e sistemas de aprendizado simbólico (Mitchell 1997). Em especial, sistemas de aprendizado simbólico são utilizados em situações em que os conceitos aprendidos precisam ser interpretados por humanos. O conhecimento induzido por algoritmos de AM simbólico é geralmente representado por árvores de decisão ou por um conjunto de regras de produção.

Diferentes algoritmos de AM podem ser aplicados à uma mesma amostra de dados e alguns desses algoritmos podem apresentar melhor desempenho que outros. Entretanto, deve ser observado que, para um dado domínio, não existem garantias que um determinado algoritmo seja necessariamente melhor que outro, ou seja, não existe uma análise matemática capaz de determinar, a priori, qual algoritmo terá um melhor desempenho em relação aos outros (Kohavi, Sommerfield & Dougherty 1997). Dessa forma, estudos experimentais são necessários.

No início, a preocupação principal das pesquisas em AM era obter a melhor precisão para o classificador com a disponibilidade de uma quantidade limitada de exemplos. Dessa forma, os classificadores eram tratados como uma "caixa preta". Além disso, cada um desses algoritmos possui um formato próprio para os dados de entrada bem como para representar o conhecimento induzido, tais como Árvores de Decisão ou Regras de Produção.

Dessa forma, a comparação e até mesmo a compreensão do conhecimento extraído a partir dos exemplos fornecidos, através da verificação das árvores de decisão ou regras induzidas não é uma tarefa trivial, uma vez que é necessário trabalhar com diferentes sintaxes para as hipóteses induzidas. A avaliação dessas árvores e regras também não é uma tarefa trivial, uma vez que, para cada regra ou ramo da árvore, são necessárias informações que permitam derivar facilmente medidas objetivas de precisão, qualidade e interesse dessas regras.

Alguns indutores apresentam essas informações, mas de uma forma própria, como por exemplo em forma de porcentagem em relação à quantidade de exemplos cobertos ou mesmo como um número absoluto. Somase a isso o fato de que alguns indutores mostram essas informações separadas por cada classe existente no conjunto de exemplos rotulado, outros não, enquanto que outros indutores não apresentam nenhuma dessas informações.

Além da variedade de informação extra disponibilizada pelos diversos indutores implementados, é importante notar que, geralmente, ela é decorrente da avaliação do classificador gerado no próprio conjunto de exemplos de treinamento. Dessa forma, os resultados são muito otimistas pois o maior interesse está em obter informações relacionadas com o comportamento futuro do classificador, isto é, utilizando novos exemplos não vistos pelo indutor durante a indução do classificador. Dessa forma, é possível realizar uma avaliação menos tendenciosa utilizando uma amostra diferente daquela utilizada para treinamento.

Neste trabalho nos concentramos em formular uma proposta de unificação da linguagem de representação de conceitos para os algoritmos de AM simbólico freqüentemente utilizados pela comunidade, bem como a sua implementação, através de uma biblioteca de *scripts* para a conversão da forma de representação do conhecimento induzido por esses indutores para um formato padrão de regras por nós proposto (Prati, Baranauskas & Monard 2001b). Além disso, também foram desenvolvidos *scripts* que, a partir de um conjunto de regras no formato padrão, e de um conjunto de exemplos (de teste), calcula um conjunto mínimo de informações para cada regra, de uma forma padronizada (Prati, Baranauskas & Monard 2001a). A

partir desse conjunto mínimo de informações, medidas de qualidade e interessabilidade de regras podem ser facilmente obtidas.

Essa biblioteca foi desenvolvida utilizando a linguagem de programação PERL (PERL 1999) e está integrada a um sistema computacional de maior porte, o Sistema DISCOVER (Baranauskas & Batista 2000), que está sendo desenvolvido no nosso laboratório de pesquisa — Labic[3] — para realizar, entre outros, extração automática e análise de conhecimento.

Quanto à organização, este trabalho está divido da seguinte forma: na Seção 2 é apresentada a forma mais comum que alguns indutores simbólicos representam o conhecimento. Na Seção 3, é apresentada uma introdução à avaliação de regras. Na Seção 4, é apresentada uma sintaxe padrão para a qual o conhecimento induzido pelos sistemas de AM simbólico é convertido. Na Seção 5 é apresentado uma extensão dessa sintaxe padrão que contempla os valores da matriz de contingência de regras, bem como os algoritmos para a avaliação de conjuntos de regras e, finalmente, na Seção 6, são apresentadas algumas considerações finais.

2 Formas de Representação do Conhecimento

Para se representar o conceito induzido por um algoritmo de AM é necessária uma linguagem de representação para o conhecimento. De uma forma geral, no caso de AM simbólico, as linguagens para esse tipo de representação se dividem em dois tipos: baseadas em atributos (proposicionais) e relacionais.

Os sistemas que utilizam linguagens baseadas em atributo são mais comuns na área de AM. Sistemas que utilizam linguagens de representação como regras de produção ou árvores de decisão podem ser tratados como variantes de linguagens proposicionais. Todos os algoritmos utilizados neste trabalho usam representação de conhecimento baseada em atributo.

A forma de representação do conhecimento induzido geralmente depende da maneira pela qual o algoritmo faz indução. Algoritmos da família TDIDT — Top Down Induction of Decision Trees — como o $\mathcal{ID}3$ ou o $\mathcal{C}4.5$ induzem classificadores no formato de árvores de decisão. Já algoritmos que induzem regras de produção diretamente a partir dos conjuntos de dados, como $\mathcal{CN}2$ ou $\mathcal{R}ipper$, fornecem como saída classificadores na forma *if* <condição> *then* <classe = C_i >. A Tabela 1 mostra a forma de representação do conhecimento induzido para os indutores utilizados neste trabalho.

Algoritmo	**Forma de Representação**	Referência
$\mathcal{C}4.5$	árvore de decisão	Quinlan (1988)
$\mathcal{C}4.5rules$	regras de produção	Quinlan (1988)
$\mathcal{C}5.0$	árvore de decisão e regras de produção	www.rulequest.com
$\mathcal{ID}3$	árvore de decisão	Quinlan (1986)
$\mathcal{CN}2$	regras de produção	Clark & Boswell (1991)
$\mathcal{OC}1$	árvore de decisão (oblíqua)	Murthy, Kasif & Salzberg (1994)
$\mathcal{R}ipper$	regras de produção	Cohen (1995)
$\mathcal{MC}4$	árvore de decisão	Auer, Holte & Maass (1995)
$\mathcal{T}2$	árvore de decisão	

Tabela 1: Formas de Representação de Alguns Algoritmos de AM

As duas seções seguintes descrevem, de modo simplificado, árvores de decisão e regras de produção. Para isso, é conveniente salientar que, em AM supervisionado, é dado ao algoritmo de aprendizado um conjunto de exemplos de treinamento contendo n exemplos classificados (rotulados) segundo uma classe de interesse.

[3] http://labic.icmc.usp.br

Um exemplo, também denominado *caso*, *registro* ou *dado* na literatura, é uma tupla de valores de atributos (ou um vetor de valores de atributos). Um exemplo descreve o objeto de interesse, tal como um paciente, dados médicos sobre uma determinada doença ou histórico de clientes de uma dada companhia.

Um *conjunto de exemplos* é composto por exemplos contendo valores de atributos bem como a classe associada. Na Tabela 2 é mostrado o formato padrão de um conjunto de exemplos T com n exemplos e m atributos. Nessa tabela, a linha i refere-se ao i-ésimo exemplo ($i = 1, 2, \ldots, n$) e a entrada x_{ij} refere-se ao valor do j-ésimo ($j = 1, 2, \ldots, m$) atributo X_j do exemplo i.

	X_1	X_2	$\cdots$	X_m	Y
T_1	x_{11}	x_{12}	$\cdots$	x_{1m}	y_1
T_2	x_{21}	x_{22}	$\cdots$	x_{2m}	y_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
T_n	x_{n1}	x_{n2}	$\ldots$	x_{nm}	y_n

Tabela 2: Conjunto de exemplos no formato atributo-valor

Como pode ser notado, exemplos são tuplas $T_i = (x_{i1}, x_{i2}, \ldots, x_{im}, y_i) = (\vec{x}_i, y_i)$ também denotados por (x_i, y_i), onde fica subentendido o fato que x_i é um vetor. A última coluna, $y_i = f(x_i)$, é a função que tenta-se predizer a partir dos atributos. Observa-se que cada x_i é um elemento do conjunto $\mathsf{dom}(X_1) \times \mathsf{dom}(X_2) \times \ldots \times \mathsf{dom}(X_m)$, onde $\mathsf{dom}(X_j)$ é o domínio do atributo X_j e y_i pertence a uma das k classes, isto é, $y_i \in \{C_1, C_2, \ldots, C_k\}$.

Usualmente, um conjunto de exemplos é dividido em dois subconjuntos disjuntos: o *conjunto de treinamento* que é usado para o aprendizado do conceito e o *conjunto de teste* usado para medir o grau de efetividade do conceito aprendido. Os subconjuntos são normalmente disjuntos para assegurar que as medidas obtidas utilizando o conjunto de teste sejam de um conjunto diferente do usado para realizar o aprendizado, tornando a medida estatisticamente válida.

A fim de ilustrar o trabalho realizado, será utilizado um conjunto de dados muito simples, o *viagem*, adaptado de (Quinlan 1988) por (Baranauskas & Monard 2000), descrito a seguir. Cada exemplo do conjunto de exemplos *viagem* consiste de medidas diárias sobre as condições do tempo, composto pelos seguinte atributos:

- aparência: assume os valores discreto “sol”, “nublado” ou “chuva”;
- temperatura: um valor numérico indicando a temperatura em graus Celsius;
- umidade: também um valor numérico indicando a porcentagem de umidade;
- ventando: assume valores discretos “sim” ou “não” indicando se é um dia com vento.

Cada dia (exemplo), está rotulado com “vá” se o tempo nesse dias estava bom o suficiente para uma viagem ao campo ou “não_vá” caso contrário, como mostrado na Tabela 3. Embora esse conjunto de exemplos possua apenas duas classes, é importante salientar que árvore de decisão e regras de produção podem trabalhar com qualquer número $k \geq 2$ de classes $\{C_1, C_2, \ldots, C_k\}$.

2.1 Árvores de Decisão

Uma Árvore de Decisão — AD — é uma estrutura de dados recursivamente definida como:

- um nó folha, que indica uma classe, ou

Exemplo No.	Aparência	Temperatura	Umidade	Ventando	Viajar?
T_1	sol	25	72	sim	vá
T_2	sol	28	91	sim	não_vá
T_3	sol	22	70	não	vá
T_4	sol	23	95	não	não_vá
T_5	sol	30	85	não	não_vá
T_6	nublado	23	90	sim	vá
T_7	nublado	29	78	não	vá
T_8	nublado	19	65	sim	não_vá
T_9	nublado	26	75	não	vá
T_{10}	nublado	20	87	sim	vá
T_{11}	chuva	22	95	não	vá
T_{12}	chuva	19	70	sim	não_vá
T_{13}	chuva	23	80	sim	não_vá
T_{14}	chuva	25	81	não	vá
T_{15}	chuva	21	80	não	vá

Tabela 3: Conjunto de exemplos *viagem*

- um nó de decisão, que contém um teste sobre o valor de um atributo. Para cada um dos possíveis valores do atributo tem-se um ramo para uma outra árvore de decisão (subárvore). Cada subárvore contém a mesma estrutura de uma árvore.

Uma representação gráfica de uma AD induzida pelo indutor $\mathcal{C}4.5$ para o conjunto de dados *viagem* é mostrada na Figura 1. Os círculos representam os nós de decisão, nas setas estão os possíveis resultados dos testes e os quadrados representam os nós folhas.

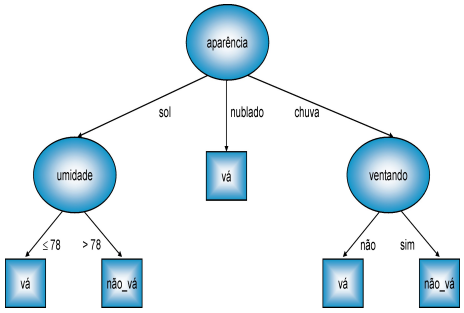

Figura 1: Representação Gráfica da Árvore de Decisão Gerada pelo Indutor $\mathcal{C}4.5$

Uma AD pode ser usada para classificar novos exemplos, começando a partir da raíz da árvore e descendo pelos nós de decisão até encontrar um nó folha. Quando um nó folha é encontrado, a classe para o novo exemplo é prevista com o rótulo do nó folha. É fácil perceber que a árvore pode ser representada como um conjunto de regras disjuntas, em que cada regra tem seu início na raíz da árvore até uma de suas folhas.

2.2 Regras de Produção

Uma regra é geralmente representada na forma

R: *if* <condição> *then* <classe = C_i >

onde <condição> é uma disjunção de conjunções de testes para os atributos da forma

$$X_i \ op \text{ valor}$$

e C_i é um dos possíveis valores para a classe.

Para facilitar a leitura, adotaremos uma representação mais genérica para qualquer regra R, onde

$$\text{R: } \underbrace{if \ < \text{condição} >}_{Body \text{ ou } B} \ then \ \underbrace{< \text{classe} = C_i >}_{Head \text{ ou } H}$$

passando a denotar uma regra como

$$Body \rightarrow Head$$

ou resumidamente $B \rightarrow H$.

O conjunto de exemplo de regras de produção que representa a mesma hipótese induzida expressa na linguagem de AD na Figura 1 pode ser vista na Figura 2.

```
IF aparencia = nublado
THEN CLASS = vá

IF aparência = sol
    AND umidade <= 78
THEN CLASS = vá

IF aparência = sol
    AND umidade > 78
THEN CLASS = não_vá

IF aparência = chuva
    AND ventando = sim
THEN CLASS = não_vá

IF aparência = chuva
    AND ventando = não
THEN CLASS = vá
```

Figura 2: Regras de Produção Obtidas a Partir da AD da Figura 1

3 Avaliação de Regras Induzidas por Algoritmos de Aprendizado de Máquina Simbólicos

O crescente uso e a diversidade de aplicações, tanto de algoritmos de Aprendizado de Máquina quanto o uso desses algoritmos na área de *Data Mining*, cria a necessidade de novas formas de avaliação/validação do conhecimento induzido (Weiss & Indurkhya 1998; Fayyad, Piatetsky-Shapiro, Smyth & Uthurusamy 1996). Essas formas de avaliação incluem medidas objetivas (obtidas a partir de diversas métricas propostas na literatura) e medidas subjetivas (obtidas através da inspeção do conhecimento induzido). Essas medidas fornecem informações sobre, entre outros, a precisão, qualidade e interessabilidade do conhecimento induzido (Horst 1999; Lavrač, Flach & Zupan 1999; Freitas 1998a; Freitas 1998b).

Como já mencionado, neste trabalho nos focalizamos no projeto e implementação de medidas objetivas de regras que represem o conhecimento induzido por algoritmos de AM simbólicos.

Os conceitos induzidos por algoritmos de AM simbólicos são geralmente representados por árvores de decisão ou conjuntos de regras. Como sempre é possível escrever uma árvore de decisão como um conjunto de regras disjuntas, deste ponto em diante o termo *regra* refere-se a uma regra extraída de uma árvore de decisão (regras disjuntas) ou uma regra diretamente induzida pelo algoritmo de AM.

Dizemos que um exemplo T_i é *coberto* por uma regra R se e somente se o exemplo satisfaz todas as condições da regra (todas as condições de uma regra são avaliadas como verdadeiras, dada a descrição do exemplo). Ou seja, um exemplo T_i é coberto por uma regra $R : B \rightarrow H$ se e somente se B é verdade. Por outro lado, um exemplo que não satisfaz a condição B da regra *não é coberto* pela regra.

Dizemos que um exemplo T_i é *corretamente coberto* por uma regra R se e somente se T_i é coberto pela regra e a classe y_i do exemplo é a mesma prevista pela regra. Ou seja, um exemplo T_i é corretamente coberto pela regra R se e somente se B é verdade e H é verdade. Entretanto, se o exemplo satisfaz a condição B da regra mas não satisfaz a condição H, o exemplo é *incorretamente coberto* pela regra. Um resumo dessas quatro possíveis situações pode ser visto na Tabela 4.

Exemplos satisfazendo ...	são ...
B	cobertos pela regra
$\overline{B}$	não cobertos pela regra
$B \wedge H$	cobertos corretamente pela regra
$B \wedge \overline{H}$	cobertos incorretamente pela regra

Tabela 4: Cobertura de uma Regra $B \rightarrow H$

3.1 Matriz de contingência

A matriz de contingência é uma generalização da matriz de confusão, que é a base padrão para calcular medidas de avaliação de hipóteses em problemas de classificação. A matriz de confusão é aplicada ao classificador como um todo, ou seja, o classificador é tratado como uma caixa preta. Já a matriz de contingência é calculada para cada regra que constitui o classificador simbólico.

Dados uma regra R e um exemplo $T_i = (\mathbf{x}_i, y_i)$ com a sua respectiva classe y_i, podemos aplicar a regra ao exemplo e comparar o resultado previsto pela cabeça H da regra com a verdadeira classe y_i do exemplo. Essa comparação resulta em quatro possíveis situações:

1. O exemplo é coberto corretamente pela regra, ou seja, B e H são verdade.
2. O exemplo é incorretamente coberto pela regra, ou seja, B é verdade mas H é falso.
3. O exemplo não é coberto pela regra mas a classe prevista pela cabeça H da regra é a mesma classe y_i do exemplo, ou seja, B é falso mas H é verdade.
4. O exemplo não é coberto pela regra e a classe prevista pela cabeça H da regra não é a mesma classe y_i do exemplo, ou seja, B é falso e H também é falso.

Aplicando a regra a um conjunto de teste T que contenha n exemplos, podemos derivar, para cada regra, a sua matriz de contingência (Tabela 5). A matriz de contingência também pode ser representada usando freqüências relativas, como mostrado na Tabela 6, onde os valores presentes na tabela de contingência foram

	H	$\overline{H}$	
B	bh	$b\overline{h}$	b
$\overline{B}$	$\overline{b}h$	$\overline{bh}$	$\overline{b}$
	h	$\overline{h}$	n

bh é o número de exemplos para os quais B é verdade e H é verdade.
$b\overline{h}$ é o número de exemplos para os quais B é verdade e H é falso.
$\overline{b}h$ é o número de exemplos para os quais B é falso, mas H é verdade.
$\overline{bh}$ é o número de exemplos para os quais B é falso e H é falso.
b é o número total de exemplos para os quais B é verdade.
$\overline{b}$ é o número total de exemplos para os quais B é falso.
h é o número total de exemplos para os quais H é verdade.
$\overline{h}$ é o número total de exemplos para os quais H é falso.
n é o número total de exemplos.

Tabela 5: Matriz de Contingência para uma Regra $R : B \rightarrow H$

	H	$\overline{H}$	
B	f_{bh}	$f_{b\overline{h}}$	f_b
$\overline{B}$	$f_{\overline{b}h}$	$f_{\overline{bh}}$	$f_{\overline{b}}$
	f_h	$f_{\overline{h}}$	1

Tabela 6: Matriz de Contingência com Freqüências Relativas para uma Regra $R : B \rightarrow H$

divididos por n, ou seja $f_\epsilon = \frac{\epsilon}{n}$. Neste trabalho, assumiremos a freqüência relativa $\frac{\epsilon}{n}$, associada ao evento ϵ, como uma estimativa de probabilidade para o evento ϵ, denotado como $P(\epsilon)$.

Usando como base a matriz de contingência, é possível definir a maioria das medidas propostas na literatura para avaliação de um conjunto de regras (Lavrač, Flach & Zupan 1999). Na Seção 3.2 são apresentadas algumas dessas medidas derivadas da matriz de contingência.

3.2 Medidas de Avaliação de Regras

Várias medidas já foram pesquisadas com a finalidade de auxiliar o usuário no entendimento e utilização do conhecimento adquirido por sistemas de AM simbólicos. As medidas apresentadas nesta seção foram compiladas em (Lavrač, Flach & Zupan 1999) e utilizam como base a matriz de contingência com freqüências relativas, definida na Seção 3.1.

Precisão (1): A precisão (*consistência* ou *confidência*) é uma medida do quanto uma regra é específica para o problema. A precisão pode ser definida como a probabilidade condicional de H ser verdade dado que B é verdade. Quanto maior, mais precisamente a regra cobre a classe em questão.

$$Acc(R) = P(H|B) = \frac{P(HB)}{P(B)} = \frac{f_{hb}}{f_b} \tag{1}$$

Erro (2): O erro de uma regra é definido como $1 - Acc(R)$. Quanto maior o erro, menos precisamente a

regra cobre a classe em questão.

$$Err(R) = 1 - Acc(r) = P(\overline{H}|B) = \frac{f_{\overline{h}b}}{f_b} \tag{2}$$

Confiança Negativa (3): é o correspondente à precisão, mas para os exemplos que não são cobertos pela regra. É definida como a probabilidade condicional de H ser falso dado que B também é falso.

$$NegRel(R) = P(\overline{H}|\overline{B}) = \frac{P(\overline{HB})}{P(\overline{B})} = \frac{f_{\overline{hb}}}{f_{\overline{b}}} \tag{3}$$

Sensitividade (4): Sensitividade (*completeza* ou *recall*) é uma medida do número (relativo) de exemplos da classe prevista em H cobertos pela regra. É definida como a probabilidade condicional de B ser verdade dado que H é verdade. Quanto maior a sensitividade, mais exemplos são cobertos pela regra.

$$Sens(R) = P(B|H) = \frac{P(HB)}{P(H)} = \frac{f_{hb}}{f_h} \tag{4}$$

Especificidade (5): é o correspondente à completeza, mas para os exemplos que não são cobertos pela regra R. É definida como a probabilidade condicional de B ser falso dado que H é falso.

$$Spec(R) = P(\overline{B}|\overline{H}) = \frac{P(\overline{HB})}{P(\overline{H})} = \frac{f_{\overline{hb}}}{f_{\overline{h}}} \tag{5}$$

Cobertura (6): Cobertura é uma medida do número (relativo) de exemplos cobertos pela regra R. É definida como a probabilidade de B ser verdade. Quanto maior a cobertura, maior o número de exemplos cobertos pela regra R.

$$Cov(R) = P(B) = f_b \tag{6}$$

Suporte (7): Suporte (*freqüência*) é uma medida do número (relativo) de exemplos cobertos corretamente pela regra R. É definido como a probabilidade de H e B serem verdade. Quanto maior o suporte, maior o número de exemplos da classe em questão que são cobertos corretamente pela regra R.

$$Sup(R) = P(HB) = f_{hb} \tag{7}$$

Novidade (8): Novidade pode ser definida como se a probabilidade de B e H ocorrerem juntos não puder ser inferida pelas probabilidades de B e H isoladamente, isto é, B e H não são estatisticamente independentes. A medida de novidade é obtida comparando o valor esperado $P(HB)$ com os valores de $P(H)$ e $P(B)$. Quanto mais o valor esperado diferir do observado, maior é a probabilidade que exista um correlação verdadeira e inesperada entre B e H. Pode ser demonstrado que $-0,25 \leq Nov(R) \leq 0,25$; quanto maior um valor positivo (mais próximo de 0,25) mais forte é a associação entre B e H enquanto que, quanto maior um valor negativo (mais próximo de -0,25), mais forte é a associação entre B e $\overline{H}$.

$$Nov(R) = P(HB) - P(H)P(B) = f_{hb} - f_h \cdot f_b \tag{8}$$

Satisfação (9): Satisfação é o aumento relativo na precisão entre a regra $B \rightarrow verdade$ e a regra $B \rightarrow H$. É uma medida mais indicada para tarefas voltadas à descoberta de conhecimento, sendo capaz de promover um equilíbrio entre regras com diferentes condições e conclusões.

$$Sat(R) = \frac{P(\overline{H}) - P(\overline{H}|B)}{P\overline{H}} = \frac{f_{\overline{h}} - \frac{f_{\overline{h}b}}{f_b}}{f_{\overline{h}}} \tag{9}$$

Precisão Relativa (10): A precisão relativa de uma regra mede o ganho de precisão obtido em relação à precisão de uma regra padrão $verdade \rightarrow H$, ou seja, que avalia B como verdade para todos os exemplos. Nesse caso, uma regra só interessa se melhorar a precisão da regra padrão.

$$RAcc(R) = P(H|B) - P(H) = \frac{f_{hb}}{f_b} - f_h \tag{10}$$

Confiança Negativa Relativa (11): É o análogo a precisão relativa para os exemplos que não são cobertos pela regra. Nesse caso, a regra padrão é $falso \rightarrow \overline{H}$.

$$RNegRel(R) = P(\overline{H}|\overline{B}) - P(\overline{H}) = \frac{f_{\overline{hb}}}{f_{\overline{b}}} - f_{\overline{h}} \tag{11}$$

Sensitividade Relativa (12): A sensitividade relativa mede o ganho de sensitividade obtido em relação à sensitividade de uma regra padrão $B \rightarrow verdade$, ou seja, uma regra que avalia H como verdade para todos os exemplos.

$$RAcc(R) = P(B|H) - P(B) = \frac{f_{hb}}{f_h} - f_b \tag{12}$$

Especificidade Relativa (13): É o análogo a sensitividade relativa para os exemplos que não são cobertos pela regra. Nesse caso, a regra padrão é $\overline{B} \rightarrow falso$.

$$RSpec(R) = P(\overline{B}|\overline{H}) - P(\overline{B}) = \frac{f_{\overline{hb}}}{f_{\overline{h}}} - f_{\overline{b}} \tag{13}$$

Um ponto importante referente as medidas relativas (medidas 10, 11, 12 e 13) é que elas dão mais informação sobre a utilidade de uma regra que as informações fornecidas pelas suas respectivas medidas absolutas (medidas 1, 3, 4 e 5). Por exemplo, se a precisão de uma regra é menor que a freqüência relativa da classe que a regra prediz, então a regra tem um desempenho ruim, independentemente de sua precisão absoluta.

Existe, no entanto, um problema com a precisão relativa, pois é fácil obter uma alta precisão relativa para regras muito específicas, ou seja, regras com uma baixa generalidade de $P(B)$.

Para contornar esse problema, é proposto em (Lavrač, Flach & Zupan 1999) uma variante das medidas relativas, na qual é atribuído um peso para cada uma dessas medidas. Este peso promove uma balanceamento entre a generalidade e a relatividade dessas medidas.

Dessa forma, a medida $RAcc(R)$ é multiplicada pelo seu "coeficiente de peso", $P(B)$, obtendo a nova medida Precisão Relativa com Peso (14), também conhecida na literatura como ganho. É possível mostrar que essa medida é equivalente a $Nov(R)$, ou seja, regras com alta precisão relativa também tem alta novidade e vice-versa.

$$WRAcc(R) = P(B)(P(H|B) - P(H)) = f_b\left(\frac{f_{hb}}{f_b} - f_h\right) \quad (14)$$

$$WRAcc(R) \equiv Nov(R) \quad (15)$$

Analogamente, as medidas Confiança Negativa (16), Sensitividade (17) e Especificidade (18) Relativas com Peso são obtidas multiplicando as correspondentes medidas relativas, $RNegRel(R)$, $RSens(R)$ e $RSpec(R)$, pelos coeficientes de pesos $P(\overline{B})$, $P(H)$ e $P(\overline{H})$, respectivamente. Lavrač também mostra que essas medidas são equivalentes entre si, ressaltando a importância da precisão relativa com peso como medida fundamental para a avaliação de regras, promovendo um balanceamento entre precisão e as outras medidas.

$$WRNegRel(R) = P((\overline{B}))P(\overline{H}|\overline{B}) - P(\overline{H}) = f_{\overline{b}}\left(\frac{f_{\overline{hb}}}{f_{\overline{b}}} - f_{\overline{h}}\right) \quad (16)$$

$$WRSens(R) = P(H)(P(B|H) - P(B)) = f_h\left(\frac{f_{hb}}{f_h} - f_b\right) \quad (17)$$

$$WRSpec(R) = P((\overline{H}))P(\overline{B}|\overline{H}) - P(\overline{B}) = f_{\overline{h}}\left(\frac{f_{\overline{hb}}}{f_{\overline{h}}} - f_{\overline{b}}\right) \quad (18)$$

$$WRAcc(R) \equiv WRNegRel(R) \equiv WRSens(R) \equiv WRSpec(R) \quad (19)$$

Muitas outras medidas para avaliação de regras podem ser encontradas na literatura. Por exemplo, em (Horst 1999) são descritas outras medidas de qualidade e interessabilidade de regras. Em (Bayardo & Agrawal 1999) são apresentadas medidas usadas na ordenação de regras de associação. Em (Freitas 1998a) e (Freitas 1998b) são apresentadas algumas medidas objetivas para se avaliar qualidade e surpresa de regras. Em (An & Cercone 1999) são apresentadas medidas derivadas empiricamente para a avaliação da qualidade de regras. Em geral, o número de nós presentes em uma árvore ou o número de testes presentes no corpo de uma regra são usados, normalmente, como medida de compreensibilidade de regras.

As medidas apresentadas neste trabalho, bem como outras existentes na literatura, podem ser combinadas ou avaliadas separadamente, para auxiliar o usuário no entendimento e utilização do conhecimento adquirido por sistemas de AM simbólico. Também vale lembrar que, mesmo que a precisão global do classificador não seja considerada boa, podem existir algumas regras boas em termos de qualidade, novidade, interessabilidade, etc. Este aspecto é muito importante na área de descoberta de conhecimento (KDD[4]) (Fayyad, Piatetsky-Shapiro, Smyth & Uthurusamy 1996).

3.3 Interpretação de um Conjunto de Regras

Árvores de Decisão dividem o espaço de descrição do problema em regiões disjuntas, isto é, cada exemplo é classificado por apenas um único ramo da árvore. As regras obtidas pela reescrita da árvore como um conjunto de regras preserva essa propriedade. Dessa forma, ao aplicarmos um exemplo a um conjunto de regras obtidas pela reescrita de uma árvore de decisão, uma única regra cobrirá aquele exemplo. O espaço de descrição também é divido de uma forma completa, isto é, cada ponto no espaço é atribuído a uma região. Na Figura 3 é mostrada a interpretação geométrica para uma árvore de decisão de duas classes (+ e o) e dois atributos (X_1 e X_2).

[4]Knowledge Data Discovery

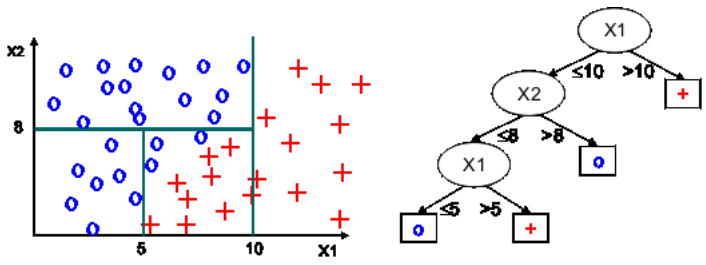

Figura 3: Interpretação Geométrica para uma Árvore de Decisão

Um ponto importante a ser considerado na avaliação de um conjunto de regras é a forma em que essas regras foram induzidas. Alguns algoritmos de AM que induzem regras diretamente dos dados criam um conjunto ordenado de regras, outros algoritmos criam um conjunto não ordenado de regras e alguns algoritmos ambos tipos de regras.

Para conjuntos ordenados, a ordem de aplicação das regras é fundamental. O exemplo deve ser avaliado pelo conjunto de regras, começando pela primeira até encontrarmos uma regra que cubra o exemplo. O exemplo deve ser classificado exclusivamente por essa regra, mesmo que as regras seguintes também cubram isoladamente o exemplo.

Já em conjuntos não ordenados, podemos aplicar o exemplo a todas as regras para calcular os valores da matriz de contingência, uma vez que a ordem de aplicação das regras não é importante. No caso de AD, como as regras obtidas pela reescrita de uma árvore de decisão são disjuntas, elas podem ser tratadas como regras não ordenadas, pois somente uma regra cobre o exemplo.

Quando um algoritmo de AM induz um conjunto regras não ordenadas a partir dos exemplos, as regras podem definir regiões sobrepostas no espaço de descrição. Na classificação de um exemplo, cada algoritmo deve definir a forma para classificar o exemplo quando mais de uma regra cobri-lo. Pode também acontecer que o espaço de descrição não seja completamente coberto pelo conjunto de regras. Para esses casos, os algoritmos de AM geralmente criam uma regra padrão (*default*), atribuindo ao exemplo que não é coberto por nenhuma das regras a classe com o maior número de exemplos no conjunto de dados de treinamento, ou a classe que foi menos coberta por todas as outras regras. Na Figura 4 é mostrada a interpretação geométrica para um conjunto de quatro regras não ordenadas e duas classes, + e o.

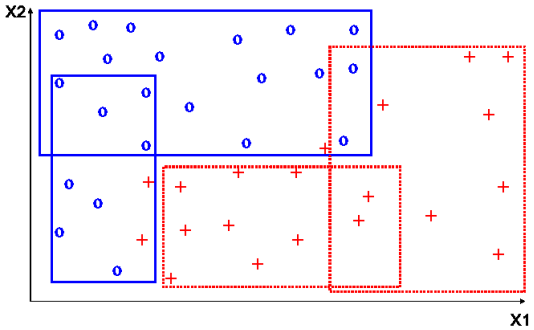

Figura 4: Interpretação Geométrica para um Conjunto de Regras Não Ordenadas

Um outro exemplo interessante são as regras induzidas pelo algoritmo de AM $\mathcal{C}4.5rules$. O $\mathcal{C}4.5rules$

utiliza uma árvore de decisão induzida pelo $\mathcal{C}4.5$ e então deriva um conjunto de regras. As regras são agrupadas por classe e são não ordenadas dentro de uma mesma classe, mas são ordenadas entre as classes. É muito importante notar que o $\mathcal{C}4.5rules$ não reescreve simplesmente a árvore de decisão para um conjunto de regras. Na verdade, ele generaliza as regras desconsiderando condições supérfluas.

4 O Formato Padrão $\mathcal{PBM}$

Um formato específico de regras de produção — o formato $\mathcal{PBM}$ — foi adotado neste trabalho para o qual são convertidos os classificadores induzidos pelos algoritmos de AM que fazem parte da biblioteca de conversão por nós desenvolvida.

O formato padrão $\mathcal{PBM}$ adotado para o desenvolvimento da biblioteca de conversão baseia-se no seguinte formato de regras:

$$\textit{if } \text{<condição>} \textit{ then } \text{<class} = C_i\text{>}$$

e é definido formalmente pela gramática $\mathcal{G} = < \vartheta_1, \Sigma_1, \delta_1, \varsigma >$, onde:

- $\vartheta_\mathbf{1}$ é o vocabulário dos símbolos não terminais: <regra>, <condição>, <classificação>, <fator>, <termo>, <comparação>, <combinação linear> e <somas>, <operador>, <atributo>, <atributo numérico> e <valor>.
- $\mathbf{\Sigma_1}$ é o vocabulário dos símbolos terminais: *if*, *then*, *class*, *and*, *or*, $X_1, X_2, ..., X_m, Y, \leq, \geq, \in, <, >, \neq$ e $=$.
- $\delta_\mathbf{1}$ é o conjunto das leis de formação da gramática $\mathcal{G}$, como é mostrado na Tabela 7.
- ς é o símbolo inicial da gramática: <regra>.

$\delta_1 =$	<regra>	$\Rightarrow$	`IF` <condição> `THEN` <classificação>
	<condição>	$\Rightarrow$	<fator> |
			<fator> `OR` <condição>
	<fator>	$\Rightarrow$	<termo> |
			<termo> `AND` <fator>
	<termo>	$\Rightarrow$	<comparação> |
			<combinação linear>
	<comparação>	$\Rightarrow$	<atributo> <operador> <valor>
	<combinação linear>	$\Rightarrow$	<somas> <operador> <valor>
	<somas>	$\Rightarrow$	<constante> $\times$ <atributo numérico> |
			<constante> $\times$ <atributo numérico> + <somas>
	<operador>	$\Rightarrow$	$\leq, \geq, \in, =, <, >, \neq$
	<classificação>	$\Rightarrow$	`CLASS` = <valor>

Tabela 7: Conjunto de Transições da Gramática $\mathcal{G}$

Dessa forma, o classificador induzido pelo indutor, depois de transformado para o formato padrão, se resume a um conjunto de regras *if* <condição> *then* <class = C_i>. O arquivo de saída que contém o resultado da conversão apresenta essas regras, enumerando-as (Figura 5). Deve se notar que tanto a ordem original das regras extraídas quanto a ordem dos testes de atributos em cada regra é mantida.

```
Standard Rules Conversor        Copyright (c) Ronaldo C. Prati
Inducer: <Inducer Name>         Input File: <Input File Name>
Date: <Date>

R0001 IF ...
    THEN ...

R0002 IF ...
    THEN ...

...
```

Figura 5: Formato do Arquivo de Saída Padrão

A biblioteca de conversão para a implementação do formato padrão foi desenvolvida utilizando-se a linguagem PERL (PERL 1999), trabalhando a partir dos arquivos de saída gerados pelos indutores. A saída, já no formato padrão, é armazenada em outro arquivo ou apresentada na tela. Atualmente, a biblioteca converte para o formato padrão nove algoritmos de AM — $\mathcal{ID}3$, $\mathcal{C}4.5$, $\mathcal{C}4.5rules$, $\mathcal{CN}2$, $\mathcal{OC}1$, $\mathcal{R}ipper$, $\mathcal{C}5.0/\mathcal{S}ee5$, $\mathcal{T}2$ e $\mathcal{MC}4$. Uma visão geral da biblioteca está representada na Figura 6.

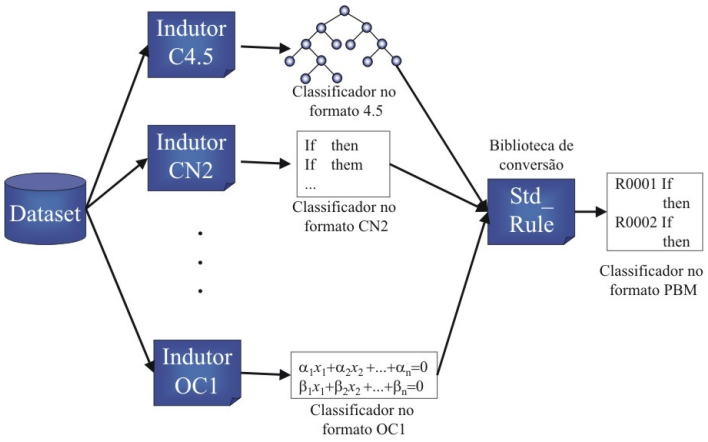

Figura 6: A biblioteca de conversão

5 Extensão do Formato Padrão de Regras

Além da conversão dos classificadores para o formato padrão, um dos objetivos deste trabalho é fornecer ao usuário ou pesquisador, de uma forma padronizada, um conjunto de informações para cada regra, sobre o qual se possa derivar facilmente as métricas aqui citadas, além de outras medidas.

Todas as medidas aqui apresentadas, bem como a maioria das medidas objetivas de regras encontradas na literatura, podem ser facilmente derivadas a partir da matriz de contingência. Assim, o formato padrão de regras, descrito na Seção 4, foi estendido, contendo agora também as freqüências relativas f_{bh}, $f_{b\overline{h}}$, $f_{\overline{bh}}$, e $f_{\overline{b}h}$, além do número de exemplos, n, utilizados na medição. As freqüências marginais, f_b, $f_{\overline{b}}$, f_h e $f_{\overline{h}}$ podem ser

facilmente obtidas pela soma das linhas e colunas da matriz de contingência (Tabela 6). Os valores absolutos das medidas também podem ser facilmente obtidos multiplicando as freqüências relativas por n. O formato padrão proposto para as informações adicionais contém agora a forma de avaliar as regras e duas listas com os seguintes elementos numéricos:

$$\left[f_{bh}, f_{b\overline{h}}, f_{\overline{b}h}, f_{\overline{b}\overline{h}}, n\right]$$

Esse conjunto de informações é obtido através da medição de um conjunto de regras no formato padrão de regras $\mathcal{PBM}$ utilizando um conjunto de exemplos fornecido pelo usuário, e adicionadas ao final de cada uma das regras.

A aplicação de regras para exemplos que contenham valores não especificados (desconhecidos) é dependente do domínio do problema e do algoritmo de AM utilizado para induzir essas regras. Para que esses valores não interfiram nas medições, neste trabalho assumimos como verdadeiro qualquer teste para um atributo com valor *desconhecido*. Essa avaliação foi implementada em uma função separada, podendo ser facilmente modificada caso o pesquisador ou usuário queira implementar uma nova abordagem. O conjunto de informações padrão é medido separadamente para os exemplos que contenham valores *desconhecidos* para os atributos avaliados pela regra, e adicionados ao final de cada regra precedidos de um ponto de interrogação (?).

Nas Figura 7 e 8 são mostrados, respectivamente, o arquivo com o conjunto de regras com informações calculadas para cada uma das regras e a gramática estendida que define o formato padrão de regras.

```
Standard Rules Conversor<version> Copyright (c) Ronaldo C. Prati
Inducer:  <Inducer Name> Input File:  <Input File Name>
Date:  <Date>

Rules Evaluated as <ORDERED|UNORDERED|INTER-CLASS ORDERED>
Names File:  <Names File Name> Data File:  <Data File Name>

R0001 IF <condição>
THEN CLASS = C_i  [f_bh, f_bh̄, f_b̄h, f_b̄h̄, n]  ?[f?_bh, f?_bh̄, f?_b̄h, f?_b̄h̄, n?]

R0002 IF <condição>
THEN CLASS = C_i  [f_bh, f_bh̄, f_b̄h, f_b̄h̄, n]  ?[f?_bh, f?_bh̄, f?_b̄h, f?_b̄h̄, n?]

...
```

Figura 7: Formato do Arquivo de Regras com Informações Padrão

No arquivo de saída também é incluída a expressão "*Rules Evaluated as*" seguido da informação de como o conjunto de regras foi avaliado: `ORDERED` se o conjunto de regras foi avaliado como ordenado, `UNORDERED` como não ordenado ou `INTER-CLASS ORDERED` como ordenado entre as classes. São também incluídas as expressões "*Names File:*", seguida do nome do arquivo contendo o arquivo de nomes, bem como a expressão "*Data File:*", seguida do nome do arquivo contendo os exemplos usados na avaliação. A saída do *script*, com as regras e as informações calculadas para cada regra, é armazenada em outro arquivo ou mostrada na tela. Uma visão geral da biblioteca está representada na Figura 9.

δ_1 =	<regra>	⇒	`IF` <condição> `THEN` <classificação> <extensão>
	<condição>	⇒	<fator> |
			<fator> `OR` <condição>
	<fator>	⇒	<termo> |
			<termo> `AND` <fator>
	<termo>	⇒	<comparação> |
			<combinação linear>
	<comparação>	⇒	<atributo> <operador> <valor>
	<combinação linear>	⇒	<somas> <operador> <valor>
	<somas>	⇒	<constante> × <atributo numérico> |
			<constante> × <atributo numérico> + <somas>
	<operador>	⇒	$\leq, \geq, \in, =, <, >, \neq$
	<classificação>	⇒	`CLASS` = <valor>
	<extensão>	⇒	<valores conhecidos> <valores desconhecidos>
	<valores conhecidos>	⇒	[<valor>, <valor>, <valor>, <valor>, <valor>]
	<valores conhecidos>	⇒	[<valor>, <valor>, <valor>, <valor>, <valor>]

Figura 8: Gramática do Formato Padrão de Regras Extendido

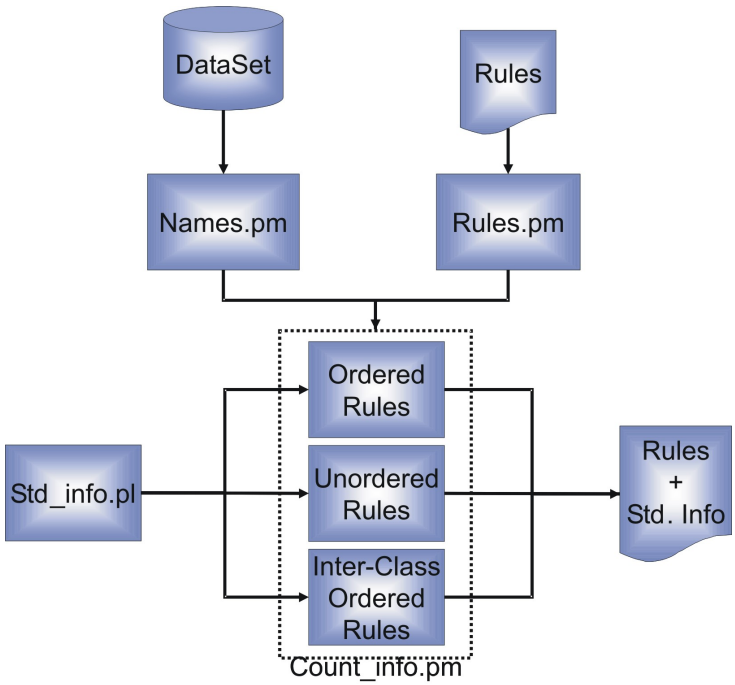

Figura 9: A Biblioteca para Cálculo das Informações Padrão

No cálculo dessas informações deve ser considerado a forma que as regras do classificador simbólico **h** foram induzidas pelo algoritmo de AM. O Módulo de Análise de Regras calcula essas informações para regras *unordered*, *ordered* e *interclass*, conforme descrito a seguir.

Unordered Para um conjunto de regras *unordered*, essas informações são calculadas verificando a cobertura

de cada regra do classificador simbólico $\mathbf{h} = \{R_1, ..., R_n\}$ em todo conjunto de exemplos S. O Algoritmo 1 mostra como é realizado o cálculo dessas informações para um conjunto de regras *unordered*.

Algoritmo 1 Cálculo de Informações para Avaliação de Regras *Unordered*

Require: $\mathbf{h} = \{R_1, ..., R_n\}$: Conjunto de Regras *Unordered*
$S = \{(\mathbf{x}_1,y_1),...,(\mathbf{x}_n,y_n)\}$: Conjunto de Exemplos
$L := [0, 0, 0, 0]$;
for all $R_u \in \mathbf{h}$ **do**
  for all $(\mathbf{x}_i,y_i) \in S$ **do**
    j:= cobertura($R_u(\mathbf{x}_i,y_i)$);
    Incrementar em 1 o elemento j da lista L;
  end for
end for
return L

A função cobertura($R_u(\mathbf{x}_i,y_i)$) retorna um número inteiro no intervalo $[1 \dots 4]$ que é usado como índice para realizar o elemento correspondente da lista L. O índice 1 corresponde a hb, 2 a $\overline{h}b$, 3 a $h\overline{b}$, 4 a $\overline{hb}$.

Ordered No caso de regras *ordered* existe um **else** implícito entre cada regra, assim o exemplo deve ser aplicado a partir da primeira regra, até que uma delas cubra esse exemplo (B é verdade para o exemplo). Esse exemplo deve ser coberto apenas por essa regra. Para as regras seguintes àquela que cobre o exemplo, as informações devem ser atualizadas incrementando-se os valores de $\overline{b}h$ e $\overline{bh}$ conforme a classe C_v do exemplo. O Algoritmo 2 mostra como é realizado o cálculo dessas informações para um conjunto de regras *ordered*.

Algoritmo 2 Cálculo de Informações para Avaliação de Regras *Ordered*

Require: $\mathbf{h} = \{R_1, ..., R_n\}$: Conjunto de Regras *Ordered*
$S = \{(\mathbf{x}_1,y_1),...,(\mathbf{x}_n,y_n)\}$: Conjunto de Exemplos
$L := [0, 0, 0, 0]$;
$first := false$;
for all $R_u \in \mathbf{h}$ **do**
  for all $(\mathbf{x}_i,y_i) \in S$ **do**
    while first = false **do**
      $j :=$ cobertura($R_u(\mathbf{x}_i,y_i)$);
      Incrementar em 1 o elemento j da lista L;
      if $j = 1$ or $j = 2$ **then**
        first:=true;
      end if
    end while
    $j :=$ cobertura($R_u(\mathbf{x}_i,y_i)$); {o exemplo já foi coberto por uma regra }
    if $j = 3$ or $j = 4$ **then**
      Incrementar em 1 o elemento j da lista L;
    end if
  end for
end for
return L

inter-class As regras *interclass* são separadas em C_v blocos para cada uma das $C_v, v = 1, ..., NCl$ diferentes classes, existindo um **else** implícito entre cada um desses blocos. Para regras em um mesmo bloco C_v, as informações são calculadas como descrito anteriormente para regras *unordered*. Ao ser encontrada uma regra que cubra o exemplo em um bloco C_v, as informações extraídas das regras subseqüentes devem ser atualizadas como descrito anteriormente para regras *ordered*, nos C_{NCl-v} blocos seguintes.

Por exemplo, a Figura 10 mostra que, para cada classe, existe um conjunto de subconjuntos de regras $\mathbf{h}' = \{\mathbf{h}'_1, ..., \mathbf{h}'_{NCl}\}$[5]. É importante observar que, além do **else** implícito entre cada um desses subconjuntos de regras, esses subconjuntos são formados conforme a ordem em que o algoritmo de AM induz as classes que compõem a hipótese $\mathbf{h}$. Cada elemento de $\mathbf{h}'$, ou seja, $\mathbf{h}'_1, ..., \mathbf{h}'_{NCl}$, é uma hipótese com as regras que predizem a mesma classe.

$$
\begin{array}{ll}
\mathbf{h}'_1 & \begin{cases} R_1 : Corpo \text{ then CLASS } = \alpha \\ R_2 : Corpo \text{ then CLASS } = \alpha \end{cases} \\
\textbf{else} & \\
\mathbf{h}'_2 & \begin{cases} R_3 : Corpo \text{ then CLASS } = \beta \\ R_4 : Corpo \text{ then CLASS } = \beta \\ R_5 : Corpo \text{ then CLASS } = \beta \end{cases} \\
\textbf{else} & \\
 & \vdots \\
\mathbf{h}'_{NCl} & \begin{cases} R_{u-2} : Corpo \text{ then CLASS } = \gamma \\ R_{u-1} : Corpo \text{ then CLASS } = \gamma \end{cases}
\end{array}
$$

com $\alpha, \beta, \gamma \in C = C_1, ..., C_{NCl}$.

Figura 10: Regras *Interclass* — Conjunto de Subconjuntos de Regras

Assim, o exemplo deve ser aplicado a todas as regras de $\mathbf{h}'_1$ e as informações são calculadas como descrito anteriormente para regras *unordered*. Se nenhuma regra de $\mathbf{h}'_1$ cobrir o exemplo, ele deve ser aplicado a $\mathbf{h}'_2$ e assim por diante. Ao ser encontrada uma regra em um $\mathbf{h}'_v$ que cobre o exemplo, as informações extraídas das regras subseqüentes são atualizadas como descrito anteriormente para regras *ordered*, nos $\mathbf{h}'_{NCl-v}$ subconjuntos seguintes.

6 Considerações Finais

A análise de um conjunto de regras induzidas por um algoritmo de Aprendizado de Máquina simbólico deve levar em consideração vários aspectos que não somente a precisão do classificador como uma "caixa preta". Por representarem o conhecimento de uma forma explícita e compreensível por seres humanos (regras ou árvores de decisão transformadas em regras, no contexto desse trabalho), diversas questões quanto a qualidade, interessabilidade, novidade, entre outros, que cada uma essas regras possam apresentar podem ser levantadas.

[5] $\mathbf{h}' = \mathbf{h}$ - regra *default*. A regra *default* é uma regra especial cujo corpo é vazio e a cabeça é a classe de maior freqüência no conjunto de exemplos. Assim, uma hipótese $\mathbf{h}$ sempre irá classificar um exemplo devido à existência da regra *default*.

Algumas dessa questões podem ser analisadas considerando medidas objetivas, calculadas a partir de um conjunto de exemplos. Esses exemplos devem ser, preferencialmente, desconhecidos do indutor (não utilizados como exemplos para a indução do classificador), para uma análise mais realística e menos tendenciosa dessas medidas.

O levantamento dessas medidas pode ser realizado através de um conjunto de informações sobre cada regra. No entanto, essa não é uma tarefa fácil quanto estamos comparando conjunto de regras provenientes de diversos indutores, uma vez que, geralmente, esses indutores apresentam essas informações, quando presentes, de uma maneira não uniforme.

Neste trabalho foram apresentados duas classes de ferramentas que se complementam. Na primeira, os classificadores induzidos por diversos algoritmos de aprendizado simbólico podem ser transformados em um formato padrão de regras, mas no qual não foi levado em conta a avaliação dessas regras. Na segunda, além de estender o formato padrão de regras, considerando agora as três possíveis formas de avaliar regras induzidas por algoritmos de AM, também implementamos a avaliação da matriz de contingência de cada uma dessas regras segundo esses três critérios, utilizando um conjunto de dados fornecido pelo usuário.

Na realidade, avaliam-se duas matrizes de contingência: uma para exemplos que não contém valores *desconhecidos* e outra para exemplos que contém valores *desconhecidos*. Deve ser observado que as informações adicionais fornecidas por essas duas matrizes de contingência permitem avaliar mais cuidadosamente as regras induzidas por diferentes algoritmos de AM simbólico.

Assim, com base neste novo formato padrão de regras, este trabalho têm como principal objetivo ajudar o pesquisador ou usuário na análise e compreensão das regras induzidas pelos indutores simbólicos mais usados na área de AM, fornecendo, de uma maneira padronizada, além do formato padrão de regras, um conjunto de informações pelas quais é possível derivar facilmente medidas de qualidade de regras, também de uma maneira padronizada.

Devido à facilidade que a abordagem de se trabalhar com um formato padrão de regras traz para as pesquisas em descoberta de conhecimento do nosso laboratório, o formato padrão de regras, por nós proposto, foi incorporado ao Sistema DISCOVER, que vem sendo desenvolvido em nosso laboratório para realizar entre outros, descoberta automática e análise de conhecimento.

Referências

An, A. & Cercone, N. (1999). An empirical study on rule quality measures. *Lecture Notes in Artificial Inteligence* (1711), 482–491.

Auer, P., Holte, R. & Maass, W. (1995). Theory and applications of agnostic pac-learning with small decision trees. In *Machine Learning: Procedings of the Twelfth International Conference*, Morgan Kaufmann. A. Prieditis and S. Russel.

Baranauskas, J. A. & Batista, G. E. A. P. A. (2000). O projeto DISCOVER: Idéias iniciais (comunicação pessoal).

Baranauskas, J. A. & Monard, M. C. (2000). Reviewing some machine learning concepts and methods. Technical Report 102, ICMC-USP. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/RT_102.ps.zip.

Bayardo, R. J. & Agrawal, R. (1999). Mining the most interesting rules. In *Fifth ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, pp. 145–154.

Clark, P. & Boswell, R. (1991). Rule induction with $\mathcal{CN}2$: Some recent improvements. In *Procedings of the Fifth European Conference*, Springer Verlag, pp. 151–163. Y. Kodratoff. http://www.cs.utexas.edu/users/pclark/papers/newcn.ps.

Cohen, W. W. (1995). Fast effective rule induction. In *Machine Learning: Procedings of the Twelfth International Conference*, San Francisco, CA, pp. 115–123. Morgan Kaufmam.

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. & Uthurusamy, R. (1996). *Advances in Knowledge Discovery and Data Mining*. AAAI Press.

Freitas, A. A. (1998a). A multi-criteria approach for the evaluation of rule interestingness. In *Proceedings of the International Conference on Data Mining*, Rio de Janeiro, RJ, pp. 7–20.

Freitas, A. A. (1998b). On objective measures of rule surprisingness. In *Principles of Data Mining & Knowledge Discovery: Proceedings of the Second European Symp. Lecture Notes in Artificial Intelligence*, Volume 1510, pp. 1–9.

Horst, P. S. (1999). Avaliação do conhecimento adquirido por algoritmos de aprendizado de máquina utilizando exemplos. Dissertação de Mestrado, ICMC-USP.

Kohavi, R., Sommerfield, D. & Dougherty, J. (1997). Data mining using $\mathcal{MLC}$++: A machine learning library in *C++*. *International Journal on Artificial Intellegence Tools*.

Lavrač, N., Flach, P. & Zupan, B. (1999). Rule evaluatin measures: A unifying view. *Lecture Notes in Artificial Inteligence* (1634), 174–185.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.

Murthy, S. K., Kasif, S. & Salzberg, S. L. (1994). A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research 2*(1), 1–32. http://www.cs.jhu.edu/~salzberg/jair94.ps.

PERL (1999). *Programming in PERL*. Morgan Kaufmann Publishers, Inc.

Prati, R. C., Baranauskas, J. A. & Monard, M. C. (2001a). Extração de informações padronizadas para a avaliação de regras induzidas por algoritmos de aprendizado de máquina simbólico. Technical Report 145, ICMC-USP. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/RT_145.ps.zip.

Prati, R. C., Baranauskas, J. A. & Monard, M. C. (2001b). Uma proposta de unificação da linguagem de representação de conceitos de algoritmos de aprendizado de máquina simbólicos. Technical Report 137, ICMC-USP. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/RT_137.ps.zip.

Quinlan, J. R. (1986). *Induction of Decision Trees*, Volume 1, pp. 81–106. Shavlik and Dietterich.

Quinlan, J. R. (1988). *C4.5 Programs for Machine Learning*. CA: Morgan Kaufmann.

Weiss, S. M. & Indurkhya, N. (1998). *Predictive Data Mining: A Practical Guide*. San Francisco, CA: Morgan Kaufmann.