Análise Formal do Aprendizado Supervisionado por Árvores de Decisão

Maurício Bellissimo Falleiros

Dissertação apresentada ao Instituto de Matemática e Estatística da Universidade de São Paulo para obtenção do grau de Mestre em Ciências da Computação.

Orientador: Flávio Soares Corrêa da Silva

São Paulo, 25 de julho de 2000.

Análise Formal do Aprendizado Supervisionado por Árvores de Decisão

Este exemplar corresponde à redação final da dissertação devidamente corrigida e defendida por Maurício Bellissimo Falleiros e aprovada pela banca examinadora.

São Paulo, 25 de julho de 2000.

Banca examinadora:

- Prof. Dr. Flávio Soares Corrêa da Silva (orientador) – IME-USP
- Prof. Dr. Júlio Michael Stern – IME-USP
- Prof^{ra}. Dr^a. Solange Oliveira Rezende – ICMC-SC-USP

Resumo

Nesta dissertação apresentamos duas vertentes da pesquisa em aprendizagem computacional, uma formal e outra empírica, destacando o modelo de análise “*Provavelmente Aproximadamente Correto*” (*PAC*) e o algoritmo *REAL* de indução de árvores de decisão sobre atributos de domínio real.

A seguir, levantamos a curva de aprendizagem do algoritmo *REAL* sobre uma base de dados padrão para testes de algoritmos de aprendizagem desta natureza e comparamos esta curva com as previsões teóricas dadas pelo modelo *PAC* e pelo modelo de Convergência Uniforme.

Fica evidente a grande lacuna entre estes resultados e então propomos algumas possibilidades de aprofundamento desta análise.

Abstract

In this dissertation we present two frameworks of the machine learning research, one formal and the other practical, emphasizing the analysis model “*Probably Approximately Correct*” (*PAC*) and the algorithm *REAL* of induction of decision trees on real-valued attributes.

We then build the learning curve of the algorithm *REAL* on a standard database for benchmarking learning algorithms of this nature and compare this curve and the theoretical predictions given by the *PAC* model and by the Uniform Convergence model.

The gap between these results becomes clear and then we propose some possibilities for a deeper approach.

Agradecimentos

Ao Prof. Flávio Soares Corrêa da Silva, pelo apoio, pelo respeito e pela paciência.

Aos Profs. Júlio Stern e Solange Rezende, pelo interesse e pelas sugestões e comentários sobre as versões preliminares deste trabalho.

Aos colegas Marcelo Lauretto e Fábio Nakano, pela cessão da implementação do REAL e da base de dados e pelas explicações e sugestões.

À minha família, pelos valores e pela educação privilegiada que me permitiram começar este trabalho.

À Renata, pelo interesse e incentivo, sem os quais eu não teria concluído este trabalho.

Aos colegas de empresa, pelo respeito e pelo apoio ao meu projeto pessoal.

Sumário

1	Introdução	1
1.1	Caracterização da aprendizagem	2
1.2	Modelos formais de análise	3
1.3	Objetivos	5
1.4	Organização	6
2	Noções preliminares	7
2.1	Ocorrências e resultados	8
2.2	Amostragem	8
2.3	Hipóteses e decisões	10
3	Aprendizagem PAC	13
3.1	Definição	13
3.2	Exemplo: algumas classes de funções booleanas	15
3.3	Espaços finitos	20
3.4	Listas de decisão	22
3.5	Árvores binárias de decisão	24
3.6	Outros modelos formais	28
3.6.1	Consultas estatísticas	28
3.6.2	A navalha de Occam	30
3.6.3	Convergência uniforme	32
3.6.4	Identificação no limite	34
3.7	Modelos de inferência estatística	35
3.7.1	Inferência bayesiana	35
3.7.2	Modelos físico-estatísticos	36
4	Abordagens práticas	39
4.1	Representação clássica de conceitos	40
4.2	Árvores de decisão	42
4.2.1	Um exemplo elementar	42
4.2.2	Seleção de atributos: a pergunta certa	46

4.2.3	Poda e pré-poda: evitando excessos	49
4.3	Um exemplo de sucesso: o C4.5	50
4.4	Algoritmo: REAL	53
4.5	Outras considerações sobre TDIDT	55
4.6	Outras famílias de algoritmos para árvores	57
4.7	Outras implementações	58
4.7.1	Programação em lógica indutiva	59
4.7.2	Redes neurais	62
5	Estudo de caso	65
5.1	Descrição da aplicação	65
5.2	Análise formal	68
5.3	Considerações adicionais	74
6	Conclusões	77

Lista de Figuras

3.1	Aprendiz de monômios	16
3.2	Aprendiz para expressões em $k\text{-}\mathcal{CNF}$	18
3.3	Aprendiz consistente para expressões em $k\text{-}\mathcal{DNF}$	21
3.4	Aprendiz consistente para listas de decisão em $k\text{-}\mathcal{LD}(\mathcal{M})$	24
3.5	Aprendiz consistente para árvore binária de posto r	27
4.1	Árvore de classificação de vertebrados: primeiro nó	43
4.2	Vertebrados: dois níveis	44
4.3	Vertebrados: mais uma expansão	44
4.4	Árvore final de classificação de vertebrados	45
4.5	Árvore obtida com critérios de seleção de atributos	47
4.6	Árvore para semiplano $x + y \leq 8$	56
5.1	Curva de aprendizagem do REAL	67
5.2	Curva de aprendizagem por árvores de posto 8	70
5.3	Limite superior de convergência uniforme: união ou intersecção de k regiões	71
5.4	Limite inferior de convergência uniforme: união ou intersecção de k regiões	72
5.5	Convergência uniforme para $\Re^{12}$	74

Lista de Tabelas

4.1	Amostra de treinamento para classificação de vertebrados . . .	43
4.2	Testes sobre uma nova amostra	46
4.3	Seleção de atributos por coesão e eficácia	47
4.4	Resultados com a nova árvore	48
5.1	Proporção de ocorrências da classe “1” nas amostras de treinamento	66

Capítulo 1

Introdução

Aprendizagem é a realização de modificações úteis no funcionamento de nossa mente. [Min89]

Aprendizagem computacional é o processo de alteração que um mecanismo computacional (programa ou sistema) realiza sobre a representação que tem de um problema, conforme tenha recebido informações sobre este problema, de forma a melhor solucioná-lo. Assim, um mecanismo com esta capacidade pode executar novas ocorrências de uma tarefa tanto melhor quanto mais informação tiver recebido.

De maneira geral, um mecanismo computacional convencional é um dispositivo que mapeia entradas em saídas de uma maneira sistemática, podendo ser visto como a implementação de uma função. Ele é programado para resolver uma classe de problemas e, a cada execução, recebe os dados da entrada e produz sempre a mesma saída. Esta característica é uma limitação muito forte quando a codificação do problema a ser resolvido é complexa demais ou, mais ainda, quando a relação entre a entrada e a saída não está ou não pode ser perfeitamente determinada. No contexto da aprendizagem computacional, um mecanismo computacional deve ser capaz de produzir uma saída tanto melhor quanto mais informação tenha recebido, construindo ou reconstruindo a representação que tem do problema ou da categoria de problemas que ele deve resolver. Valiant [Val84] diz que um programa aprende quando resolve ocorrências distintas da sua tarefa através de qualquer meio que não seja a programação explícita.

Um exemplo claro é a classificação de objetos: a entrada é composta de uma série de exemplos de como alguns objetos devem ser classificados em função de seus atributos mais a descrição de um objeto não classificado. Após o processamento da série de exemplos, o tal mecanismo deve classificar o novo objeto, fazendo-o tanto melhor quanto mais rica for a série de exemplos. Por esta capacidade de criar ou melhorar uma representação, este mecanismo será

chamado de *aprendiz*. Em contraponto, a pessoa ou outro mecanismo que fornece ao primeiro as informações para o aprendizado pode ser chamada de treinador ou *oráculo*.

Em vários casos de interesse prático, ocorrem situações em que esta capacidade é mais interessante do que codificar as informações disponíveis diretamente no programa ou simplesmente armazenar a série de exemplos e depois consultá-la:

- espaços de representação do problema muito grandes ou infinitos;
- sistemas complexos regidos por um número muito grande de variáveis, muitas vezes desconhecidas ou imensuráveis;
- deficiências de medição e de transmissão;
- apostas ou jogo contra um oponente;
- previsão de comportamento de sistemas temporais; e outros.

Algumas destas situações implicam que as informações disponíveis para o treinamento podem não ser completas ou conter erros, havendo até a possibilidade de que haja informações contraditórias. Em outros casos, as informações têm caráter estocástico.

1.1 Caracterização da aprendizagem

As aplicações de aprendizagem computacional podem ser caracterizadas por seus objetivos [Tho92]. Os objetivos mais comuns são o desenvolvimento de uma habilidade específica e a aquisição de conceitos determinados, mas podem ser a formação de teorias, a inferência indutiva, o reconhecimento de similaridades, entre outros. Tais aplicações também se distinguem pela forma como recebem informações e como trabalham sobre elas, sendo que a *aprendizagem por exemplos* e a *aprendizagem por observação* são os casos típicos de uma escala que classifica os métodos em supervisionados e não-supervisionados, ou algo intermediário, conforme as informações sejam mais ou menos explícitas sobre o alvo da aprendizagem. A noção do que é informação explícita e o que é implícita não é um consenso entre os pesquisadores da área. Thornton apresenta como exemplos, no caso geral de aprendizagem de uma relação entre dois espaços

explícito pares ordenados que exemplificam a relação;

intermediário grau de aproximação entre pares ordenados hipotéticos e a relação real;

implícito grau de aproximação entre uma relação hipotética e a relação real.

Muito em voga atualmente, é a segmentação por tipo de implementação, mesmo quando os casos de interesse prático não tenham uma fronteira bem definida com relação a este aspecto. Os segmentos mais proeminentes são o das redes neurais artificiais, o da indução de árvores de decisão e o da indução de programas lógicos e o mais antigo é a inferência estatística. Outras vertentes são o processamento genético de informação e a construção de sistemas especialistas.

Todos estes paradigmas de implementação dão bons resultados, em geral, dentro de uma ou algumas classes de problemas e a busca por resultados cada vez melhores é incessante. Entretanto, poucas vezes há a preocupação de determinar o grau de aproximação atingido em relação à melhor aproximação possível e, menos ainda, qual é o melhor grau de aproximação que uma classe de problemas admite em relação à perfeição, de maneira geral ou através de um determinado paradigma. Mesmo quando uma comparação é estabelecida empiricamente entre uma nova implementação e outras já conhecidas, aplicando-as todas ao mesmo problema e com as mesmas informações, o resultado só tem validade para este caso específico.

1.2 Modelos formais de análise

Um outro foco de pesquisa sobre aprendizagem computacional está nos modelos formais de análise, que se propõem a preencher esta lacuna, bem como trazer outras vantagens à busca de melhores implementações [Kea90, Wol95c]:

- definir *apreensibilidade*, uma característica do espaço em que se coloca o problema;
- estabelecer critérios para avaliação das implementações, tanto por comparação entre elas como em relação a uma medida de apreensibilidade do problema;
- permitir a transposição de melhorias e novas idéias entre as diferentes implementações;
- lançar novas luzes sobre o desenvolvimento destas;
- generalizar resultados empíricos;
- determinar objetivos tangíveis.

Também por este lado existem diversas vertentes, tendo origens diversas como a análise estatística bayesiana, a teoria de decisões, a análise de complexidade computacional e até modelos emprestados da física estatística. Alguns dos modelos mais expressivos são:

- a Análise Bayesiana;

- Aprendizado no Limite;
- a Dimensão VC (convergência uniforme);
- Aprendizado Provavelmente Aproximadamente Correto (PAC); e
- Aprendizado Exaustivo (Generalizador de Gibbs, física estatística).

Há entre todos eles uma estrutura comum que pode ser formalizada nas seguintes definições:

Um *espaço de ocorrências* (ou *atos ou circunstâncias*) descreve os objetos, problemas ou situações com que o aprendiz deverá se deparar durante e após o processo de aprendizagem; e um *espaço de resultados* descreve as decorrências possíveis de uma ocorrência. Um terceiro espaço, o *espaço de relações* contém as possíveis relações entre os dois primeiros espaços. O aprendiz deverá selecionar (ou construir) neste espaço uma relação que melhor aproxime a relação contida nas informações de treinamento, ou seja, uma que melhor represente as informações disponíveis.

No caso particular da classificação de objetos, o espaço de ocorrências é o espaço dos atributos que descrevem estes objetos e o espaço de resultados contém possíveis classificações para estes objetos, mas há uma grande variedade de situações que podem ser enquadradas nesta estrutura: desde a aproximação de uma função, cujo domínio é o espaço de ocorrências e cuja imagem é o espaço de resultados, até o comportamento mais adequado (resultado) para uma determinada situação (ocorrência). Um outro elemento muito freqüente é o *espaço de hipóteses* através do qual o aprendiz procura aproximar o espaço das relações quando este não é perfeitamente conhecido ou é intratável.

Todos os modelos formais lidam, em maior ou menor grau, com conceitos como

- a apreensibilidade do espaço das relações a serem aprendidas, ou seja, a relação entre a quantidade de informação dada ao aprendiz e a qualidade da aproximação que pode ser obtida;
- o tempo e o espaço de armazenamento gastos para tanto;
- a adequação dos espaços de ocorrências, de resultados e de relações com o problema;
- a adequação do espaço de hipóteses ao de relações; e
- a distribuição ou densidade de probabilidades inerente ao espaço de ocorrências.

Se, por um lado, os modelos de análise se propõem a resolver a falta de critérios percebida nas implementações, por outro têm também seus defeitos.

Em geral, apresentam pouca aderência às implementações de sucesso, pois trabalham sobre simplificações excessivas dos métodos de implementação. E, mais grave, são grosseiros em seus resultados, dando limites frouxos para a relação entre a quantidade de informação de treinamento e a qualidade da aproximação.

1.3 Objetivos

Daremos uma visão geral de algumas das abordagens empíricas e alguns dos modelos analíticos citados, com intuito de formar um panorama dos dois focos de atuação no campo da aprendizagem computacional. Um representante de cada foco será apresentado em maior detalhe e a proposta deste trabalho é juntá-los num estudo de caso para chegarmos a uma análise crítica de cada um. Outra motivação deste trabalho é ressaltar a carência e a necessidade de modelos formais adequados à implementação escolhida, já que ela tem sido largamente aplicada com sucesso de maneira empírica.

O modelo de aprendizagem *provavelmente aproximadamente correto* (ou PAC) foi escolhido pela clareza de suas definições e seus resultados, além de ter, como uma de suas preocupações, a complexidade computacional da solução de um problema. Outra questão importante, tratada no modelo PAC, é a uniformidade da aprendizagem e a generalidade dos resultados. Este modelo de aprendizagem apresenta resultados válidos para toda uma classe de problemas e não apenas para um problema específico.

As árvores de decisão, pelo lado das implementações, têm a vantagem de apresentar um resultado de mais fácil validação pelo *expert* humano, em contraposição a outros paradigmas como o das redes neurais, cuja representação do conhecimento não pode ser transmitida de volta ao usuário. Do ponto de vista de implementação, as árvores são relativamente fáceis e apresentam ótimo desempenho na avaliação das novas ocorrências de um problema para o qual foram treinadas.

Além disto, entre os paradigmas de sucesso, as árvores de decisão são as que mais carecem de análises formais, na literatura sobre o assunto, já que muitas extensões dos modelos formais têm sido motivadas pelas redes neurais e pela programação em lógica indutiva.

Para a realização do experimento e obtenção de resultados ilustrativos do lado prático da pesquisa em aprendizagem computacional, foi escolhido o algoritmo REAL, apresentado por Stern e outros [SNLR98b, SNLR98a], desenvolvido no mesmo Instituto de Matemática e Estatística da Universidade de São Paulo em que foi realizado o presente trabalho. A implementação realizada por Lauretto e Nakano [Lau96] foi adaptada para a avaliação da

iminência de sobrecarga numa central elétrica, a partir das leituras de diversos medidores, e a decisão de cortar ou manter o fornecimento de energia.

Nosso objetivo central foi levantar a curva de aprendizagem do REAL neste contexto, em função do tamanho da amostra e comparar este resultado às previsões dos modelos PAC e de Convergência Uniforme. A análise destes resultados ilustra a grande distância entre o desempenho obtido e as previsões teóricas.

Além de apontar algumas diferenças de enfoque que motivam estas divergências, sugerimos experimentos adicionais para explorar o enfoque de uniformidade dos modelos formais e outras formas de análise para explorar a especificidade de um experimento ou aplicação.

1.4 Organização

O Capítulo 2 apresenta a notação e os conceitos preliminares que serão utilizados pelo resto deste trabalho, dando ainda uma visão geral e intuitiva dos modelos formais, através de uma formulação geral do problema da aprendizagem computacional.

O Capítulo 3 detalha o modelo PAC de aprendizagem computacional e explora exemplos de sua aplicação. A seguir, descreve alguns outros modelos que podem ser considerados equivalentes ao PAC, outros que são extensões e generalizações deste e, finalmente, outros que são correlatos, embora com enfoques diferentes.

Algumas abordagens empíricas da aprendizagem computacional são apresentadas no Capítulo 4, com destaque para as árvores de decisão, mas incluindo outros paradigmas cujo sucesso está diretamente ligado ao desenvolvimento de modelos formais.

No Capítulo 5 são exibidos os testes realizados com o algoritmo escolhido e os resultados são analisados à luz das previsões dadas pelos os modelos formais escolhidos. Finalmente, o Capítulo 6 apresenta possibilidades que não foram exploradas nesta análise, para desenvolvimento futuro, e outros enfoques que podem vir a aprofundar os resultados deste trabalho.

Capítulo 2

Noções preliminares

Dentre os modelos de aprendizagem computacional, caracterizam-se como modelos de aprendizagem supervisionada aqueles em que a informação é fornecida ao aprendiz na forma de pares ordenados que exemplificam a relação a ser aprendida (às vezes, são incluídos pares que são contra-exemplo da relação). Por isto, são também caracterizados como modelos de *aprendizagem através de exemplos*. O objetivo do aprendizado através de exemplos é extrair informações relacionais ou causais de um conjunto de dados – a amostra de treinamento – e estruturá-las de forma a permitir a inferência de valores em conjuntos de dados menos completos – os conjuntos de teste e de produção.

Uma definição mais formal, seguindo a estrutura da definição da Seção 1.2, ainda que genérica e comum a vários modelos, com pequenas diferenças que serão exploradas adiante, é dada por Wolpert [Wol95c]:

Dados um espaço de ocorrências $\mathcal{X}$ e um espaço de resultados $\mathcal{Y}$ entre os quais há uma relação desconhecida φ (o conceito alvo da aprendizagem), um aprendiz, após examinar uma amostra de m pares $(x, y) \in \mathcal{X} \times \mathcal{Y}$, extraída sob uma determinada distribuição conjunta de probabilidades, e tendo ou não acesso a algum conhecimento prévio sobre a relação, deve ser capaz de produzir uma relação hipotética ϕ entre os dois espaços que seja uma boa aproximação da relação φ – ou seja, uma relação que a cada nova ocorrência x associe um resultado y da maneira mais próxima que o faria a relação desconhecida φ no maior número possível de vezes.

Para que possamos explorar alguns dos modelos de análise mais representativos deste segmento, este Capítulo formaliza, detalha e exemplifica alguns dos conceitos comuns contidos na definição acima. O Capítulo 3, a seguir, os desenvolve segundo os diversos modelos estudados.

2.1 Ocorrências e resultados

Um espaço de ocorrências $\mathcal{X}$ é um universo cujos entes podem ser descritos por um conjunto de atributos $\{a_1, a_2, \dots, a_n\}$. Cada atributo pode ter um domínio contínuo ou discreto, ordenado ou não, finito ou infinito, e corresponde a uma dimensão deste espaço. Um ponto $x \in \mathcal{X}$ é uma particular ocorrência no espaço $\mathcal{X}$, e o valor de cada coordenada deste ponto é o valor do atributo correspondente. Assim, x é uma ocorrência descrita por um conjunto de valores $\{v_1, v_2, \dots, v_n\}$, no qual v_i é o valor do atributo a_i .

Os espaços $\mathbb{R}^n$ e $\{0, 1\}^m$, por exemplo, podem ser tratados, respectivamente, como os espaços de ocorrências de pontos em um hiperespaço euclidiano e em um reticulado completo, cujos atributos são suas coordenadas. Certamente, podem representar objetos mais concretos, como as leituras de n medidores de grandezas diversas numa instalação física e a presença ou ausência de cada um dos m atributos que descrevem uma categoria de objetos. Quadros clínicos são entes descritos por sintomas e resultados de exames de natureza fisiológica de um paciente, e formam um outro espaço de ocorrências.

Também o espaço de resultados contém entes que podem ser descritos por um conjunto de atributos (novamente, podem ser contínuos ou discretos, finitos ou infinitos, ordenados ou não) e que estão de alguma forma relacionados aos entes do espaço de ocorrências: podem ser, por exemplo, rótulos de classificação, um outro ponto em um espaço qualquer, um vetor de componentes booleanas indicando a presença ou ausência de diversos atributos, um vetor associando à ocorrência em questão a probabilidade de que cada possível rótulo ou característica lhe seja apropriado, entre outras possibilidades. Seguindo com o exemplo dos quadros clínicos de pacientes, um espaço de resultados apropriado é $[0, 1]^n$, com cada dimensão associada a uma doença e o valor do resultado nesta dimensão indicando a probabilidade de sua presença no paciente a que se refere o quadro clínico. Um ponto no $\mathbb{R}^n$ pode ser classificado em pertencente ou não pertencente a uma determinada região; um ponto em $\{0, 1\}^m$ pode ser classificado em verificador ou falsificador de uma certa expressão booleana.

2.2 Amostragem

O conjunto $Z = \mathcal{X} \times \mathcal{Y}$ é chamado de *espaço de amostragem* e seus elementos são as unidades de informação disponíveis para o treinamento e a avaliação do aprendiz. Uma coleção $S = (s_1, s_2, \dots, s_m)$ de m exemplos $s_i \in Z$ é uma amostra de tamanho m . Um dos grandes objetivos dos modelos de análise

é a determinação do tamanho m que a amostra deve ter para garantir uma riqueza de informação suficiente para o sucesso do aprendiz. Na fase de avaliação do aprendiz, a amostra é apresentada ao aprendiz sem os resultados associados a cada ocorrência e estes serão comparados às respostas dadas pelo aprendiz. A fase de produção é semelhante, exceto por que os resultados não são, de fato, conhecidos, e são justamente o objetivo da aplicação da aprendizagem computacional.

Num acesso aleatório ao espaço de ocorrências $\mathcal{X}$, a ocorrência x tem uma probabilidade $\Pr\{x\}$ determinada por uma função d de distribuição ou de densidade de probabilidades, conforme $\mathcal{X}$ seja discreto ou contínuo. Muitas vezes esta função d é desconhecida, ou sujeita a restrições fracas, podendo pertencer a todo um espaço $\mathcal{D}$ de funções de distribuição ou de densidade de probabilidades. Algumas abordagens teóricas propõem a busca de soluções válidas para qualquer distribuição no espaço de distribuições $\mathcal{D}$, mas muitas vezes o ajustamento a subespaços mais realistas conduz a resultados mais aplicáveis. Outros modelos exigem a determinação a priori da distribuição de probabilidades no espaço de ocorrências, o que por sua vez restringe a abrangência do resultado e suas aplicações.

Mesmo sendo desconhecida, se a distribuição permanece fixa desde a fase de treinamento do aprendiz até sua aplicação, é possível dar maior peso às ocorrências mais prováveis, seja pela estimação desta distribuição baseada na amostra de treinamento, seja pela própria construção do aprendiz. Este fato pode ser usado de maneira ainda mais forte na análise do algoritmo, permitindo a obtenção de limites mais justos para o erro que será cometido ou, inversamente, para o número de exemplos necessários para se atingir a precisão desejada. A avaliação do algoritmo também será mais realista se penalizar mais fortemente as ocorrências mais prováveis do que as menos prováveis.

Uma amostra $S \in \mathcal{Z}^m$ é dita *consistente* se duas ocorrências iguais estão associadas a resultados iguais. Em muitos casos de interesse prático, isto não ocorre:

- a modelagem do problema pode ter deixado de considerar algum atributo relevante do espaço de ocorrências, permitindo que dois objetos efetivamente distintos deste espaço apresentem-se como idênticos;
- a amostra de treinamento (e mesmo as amostras de teste e de produção) podem conter erros de medição, leitura ou transmissão, interferências introduzidas por um adversário ou, ainda, erro do oráculo;
- o problema pode não ser determinístico.

Ainda, S é *consistente com um conceito* $c \in \mathcal{C}$ se os rótulos de todos os exemplos estão de acordo com o conceito c , ou seja, se não há erros ou ruídos.

Chama-se $\mathcal{S}(m, c)$ o conjunto das amostras de tamanho m consistentes com o conceito c .

Outro rótulo que se pode dar a uma amostra é o de *completa*, significando que ela contém todas as ocorrências possíveis, pelo menos uma vez. Outra vez, o interesse prático não está neste caso: espaços de ocorrência infinitos e série temporais, por exemplo, não admitem uma amostra completa.

Quando um problema permite uma amostra completa e consistente e esta amostra está disponível ou pode ser obtida, um algoritmo convencional dotado de boa capacidade de indexação e compressão de dados será muito mais útil que uma implementação de aprendizagem computacional ...

2.3 Hipóteses e decisões

Quando a relação entre os espaços de ocorrências $\mathcal{X}$ e de resultados $\mathcal{Y}$ é determinística e injetora, dá-se o caso geral de aprendizagem de funções, que inclui a aprendizagem para classificação de objetos segundo os atributos que o descrevem, a aproximação de funções e a indução de programas lógicos.

A modalidade mais simples é a determinação de pertinência a uma categoria: o aprendiz deve ser capaz de dizer se uma ocorrência pertence ou não a uma categoria. Esta modalidade é chamada de *aprendizagem de conceitos* e a categoria em questão é chamada de *conceito*. O espaço de resultados, também chamado de espaço de conceitos, é binário $\mathcal{Y} = \{0, 1\}$ e corresponde a uma bipartição c_1, c_2 do espaço de ocorrências, o que equivale a dizer:

$$\forall x \in \mathcal{X} \begin{cases} y(x) = 0 \rightarrow x \in c_1, \\ y(x) = 1 \rightarrow x \in c_2. \end{cases}$$

Aqui há um espaço auxiliar $\mathcal{H}$ que é chamado de *espaço de hipóteses* e é o conjunto dos subconjuntos do espaço de ocorrências $\mathcal{X}$. Neste caso, em que o espaço de resultados $\mathcal{Y}$ é binário, a hipótese construída ou selecionada pelo aprendiz é o subconjunto de $\mathcal{X}$ cujos elementos são as ocorrências que devem ser rotuladas com o resultado 1 (ou zero, alternativamente).

Para generalizar esta modalidade para resultados n -ários, podemos, alternativamente, dizer que $\mathcal{H}$ é o espaço das partições de $\mathcal{X}$ e que compete ao aprendiz selecionar a partição que lhe permite associar a cada resultado $y \in \mathcal{Y}$ uma parte $H \in \mathcal{H}$ de modo que

$$\forall x \in \mathcal{X}, x \in H \rightarrow y(x) = y(H).$$

Com isto, o aprendiz será capaz de dizer a qual categoria, em uma lista finita de possibilidades, pertence a ocorrência; e o caso binário é um caso particular em que as categorias são “conceito” e “não-conceito”.

Há casos em que a relação entre ocorrência e resultado é estocástica, como na previsão de comportamento futuro de um sistema, baseado em série histórica. Em geral, o espaço de ocorrências descreve situações e o de resultados descreve seus desfechos e o objetivo da aprendizagem é uma estratégia de decisão de custo mínimo face a uma situação atual. Um exemplo é a previsão do comportamento de papéis no mercado de ações: a partir da posição atual de um conjunto de indicadores econômicos e financeiros, o aprendiz deverá decidir pela compra ou não de uma ação que hoje não faz parte de um portfólio e, caso compre, por quanto tempo mantê-lo. Também poderá decidir pela venda ou não das ações que fazem parte do portfólio. Outra possibilidade é a determinação das probabilidades associadas a uma série de resultados possíveis para uma ocorrência. Assim, frente a um quadro clínico, o aprendiz deverá associar a cada disfunção fisiológica uma probabilidade. Também poderá determinar, com base nestas probabilidades, o melhor tratamento.

 

$$\begin{aligned}
  

Capítulo 3

Aprendizagem PAC

A aprendizagem PAC é uma vertente da Aprendizagem Supervisionada cujo objetivo é garantir a obtenção de uma hipótese que, muito provavelmente, seja uma boa aproximação para um conceito desconhecido. Daí o nome PAC – Provavelmente Aproximadamente Correto. O principal critério para esta garantia é a relação entre o tamanho da amostra de treinamento e o erro na aproximação; outro resultado importante na análise PAC é a complexidade computacional do algoritmo implementado.

Neste Capítulo, o modelo PAC de aprendizagem será examinado em detalhe. Nas Seções finais, são examinados outros modelos formais de aprendizagem computacional.

3.1 Definição

Na definição básica do modelo PAC [HW95], o espaço de ocorrências $\mathcal{X} = \{0, 1\}^n$ é o conjunto de todas as possíveis atribuições para n variáveis booleanas (ou atributos). Este modelo propõe a aprendizagem de conceitos, que são subconjuntos do espaço de ocorrências, bem como as hipóteses formadas pelo aprendiz. A noção de aproximação é dada pela consideração de que haja uma distribuição de probabilidade D sobre o espaço de ocorrências $\mathcal{X}$ dando a probabilidade de cada ocorrência. Assim, seja $error_c(h)$ o erro de uma hipótese h com respeito a um conceito c :

$$error_c(h) = \sum_{x \in h \Delta c} D(x).$$

Ou seja, $error_c(h)$ é a probabilidade de que $h(x)$ e $c(x)$ sejam diferentes em uma ocorrência x extraída aleatoriamente de $\mathcal{X}$ de acordo com D .

Para cada $n \geq 1$, seja C_n o espaço de conceitos sobre $\{0, 1\}^n$ e seja $\mathcal{C} = \{C_n\}_{n \geq 1}$. E sejam H_n e $\mathcal{H}$ definidos de maneira similar. Seja $\mathcal{D}$ o espaço de todas as distribuições de probabilidade possíveis sobre $\mathcal{X}$. Com isso, podemos definir *PAC-apreensibilidade* como:

O espaço de conceitos $\mathcal{C}$ é *PAC-apreensível* através do espaço de hipóteses $\mathcal{H}$ se existem um algoritmo A eficiente e um polinômio $p(\cdot, \cdot, \cdot)$ tais que

- para todo $n \geq 1$;
- para todo conceito $c \in C_n$;
- para todas as distribuições de probabilidade $D \in \mathcal{D}$ sobre $\mathcal{X}$; e
- para todos $\epsilon, \delta \in]0, 1[$,

se A recebe pelo menos $p(n, 1/\epsilon, 1/\delta)$ exemplos do conceito c , independentemente sorteados de $\mathcal{X}$ sob D , então, com probabilidade maior que $1 - \delta$, A devolve uma hipótese $h \in \mathcal{H}$ com $error_c(h) \leq \epsilon$. O menor polinômio p atendendo estas condições é chamado de *complexidade de amostra* do algoritmo A . A exigência de que A seja eficiente significa que este deve ter custo polinomial de tempo, sobre o tamanho da amostra e a dimensão dos espaços considerados.

Note-se que este é um modelo do tipo *pior caso*, já que o limite superior no tamanho da amostra necessária diz respeito a qualquer conceito c em $\mathcal{C}$ e a qualquer distribuição de probabilidades D em $\mathcal{D}$ sobre $\mathcal{X}$. Assim, ainda que fixemos o número de variáveis n no espaço de ocorrências e o parâmetro de confiança δ , a curva do erro ϵ em função do tamanho m da amostra de treinamento dada pela complexidade de amostra não é uma curva de aprendizagem, mas sim um envelope superior para todas as curvas de aprendizagem que podem ser obtidas variando-se o conceito alvo c e a distribuição de probabilidades D (para a confiança fixada δ). As curvas usualmente obtidas por meio de experimentação são curvas de aprendizagem para um determinado conceito a partir de uma amostra extraída sob uma particular distribuição no espaço de ocorrências e estarão sempre abaixo do envelope superior determinado pela complexidade de amostra.

Variações usuais desta definição, praticamente equivalentes, condicionam a apreensibilidade de um espaço de conceitos à existência de um algoritmo A cuja complexidade de amostra é definido em termos do número de exemplos exclusivamente positivos ou exclusivamente negativos, ao invés de exemplos quaisquer.

3.2 Exemplo: algumas classes de funções booleanas

Um dos exemplos mais diretos desta definição é a aprendizagem de conceitos conjuntivos, ou monômios, num espaço booleano de dimensão n [AB92, KV94]. O espaço de ocorrências é $\mathcal{X} = \{0, 1\}^n$ e os espaços de conceitos e de hipóteses são iguais, $\mathcal{C} = \mathcal{H} = M^n$ (o espaço dos monômios sobre $\mathcal{X}$). Por motivação, suponha que se quer ensinar um aprendiz a determinar, numa série de objetos, aqueles que são cadeiras e aqueles que não o são, com base numa lista de propriedades. O aprendiz deverá, a partir dos exemplos, determinar quais são os atributos da lista que são atributos de uma cadeira, quais não o são e quais são irrelevantes. Uma lista muito básica poderia ser algo como a que segue. Entre parênteses está a conclusão a que o aprendiz deveria chegar.

u_1 tem quatro pernas (sim)
 u_2 tem rabo (não)
 u_3 tem assento (sim)
 u_4 é vermelho (irrelevante)
 u_5 é vivo (não)

Assim, uma cadeira é determinada pela verificação do monômio $\langle u_1 \bar{u}_2 u_3 \bar{u}_5 \rangle$ e este é nosso conceito alvo.

A idéia básica do algoritmo, devido a Valiant [Val84], cf. [AB92], é simples: a hipótese inicial pressupõe que não há informação alguma – é a conjunção $\langle u_1 \bar{u}_1 u_2 \bar{u}_2 \dots u_n \bar{u}_n \rangle$ de todos os literais possíveis. Cada exemplo positivo $y = y_1 y_2 \dots y_n$ fornecido ao aprendiz lhe informa que se sua hipótese contém o literal $\bar{u}_j$ correspondente à negação de uma propriedade que o exemplo positivo apresenta ($y_j = 1$), ou inversamente, se contém o literal u_j correspondente à afirmação de uma propriedade que o exemplo positivo não apresenta ($y_j = 0$), sua hipótese o teria classificado como ocorrência não pertencente ao conceito alvo. Assim, o literal deve ser eliminado do monômio. O algoritmo é apresentado na Figura 3.1

Por ilustração, voltemos ao exemplo da cadeira, supondo que $m = 4$ e que a seqüência de exemplos obtidos é

$$\langle 10110, 1 \rangle, \langle 10011, 0 \rangle, \langle 10100, 1 \rangle, \langle 01011, 0 \rangle,$$

na qual o segundo componente de cada exemplo é a sua classificação. A seqüência de hipóteses geradas pelo algoritmo (ou seja, o monômio formado pelos literais presentes em U ao término de cada execução do laço sobre i) é

```

 $U := \{u_1, \bar{u}_1, \dots, u_n, \bar{u}_n\};$ 
para  $i$  de 1 até  $m$  faça
    obtenha um exemplo  $x_i$  do conceito alvo
    se o exemplo for positivo
        então para  $j$  de 1 até  $n$  faça
            se  $x_{ij} = 1$ 
                então remova  $\bar{u}_i$  de  $U$ , se presente
                senão remova  $u_i$  de  $U$ , se presente
            devolva o monômio formado pelos literais presentes em  $U$ 

```

Figura 3.1: Aprendiz de monômios

a seguinte:

$$\langle u_1 \bar{u}_2 u_3 u_4 \bar{u}_5 \rangle, \langle u_1 \bar{u}_2 u_3 u_4 \bar{u}_5 \rangle, \langle u_1 \bar{u}_2 u_3 \bar{u}_5 \rangle, \langle u_1 \bar{u}_2 u_3 \bar{u}_5 \rangle.$$

Observe que os exemplos negativos não alteram a hipótese. Ao fim, temos uma hipótese idêntica ao conceito alvo, o que significa que a amostra foi suficientemente boa para assegurar o sucesso total.

Pela construção do algoritmo, os literais que fazem parte do monômio alvo não são jamais removidos da hipótese. Assim, toda ocorrência negativa falsifica algum deles e, portanto, nenhuma será classificada como positiva. Por outro lado, após o processamento de um exemplo positivo, todos os literais falsificados por este exemplo são removidos e, assim, uma nova ocorrência deste exemplo será corretamente classificada. Com isto, é claro que se a amostra de treinamento contém todos os exemplos positivos, a hipótese será idêntica ao conceito alvo, como ocorreu no exemplo acima. Mas esta condição não é necessária: se o processamento de cada exemplo que contém alguma informação nova resulta na remoção de pelo menos um literal, n exemplos bem escolhidos poderão resolver o problema. Assim, o que nos falta é garantir um número de exemplos suficiente para que seja alta a probabilidade de ocorrência de um bom número de exemplos bastante informativos.

Considere um literal z que figura na hipótese h mas não no conceito alvo c [KV94]. Ele faz com que a h classifique como negativos apenas as ocorrências positivas de $\mathcal{X}$ para as quais $z = 0$ – estas ocorrências teriam feito o algoritmo remover z da hipótese, se houvessem aparecido na amostra de treinamento. Seja $p(z)$ a probabilidade total sob D (fixa porém indeterminada)

de que um tal z figure numa ocorrência positiva de $\mathcal{X}$:

$$p(z) = \Pr_{a \in D(\mathcal{X})} \{c(a) = 1 \wedge z \text{ é falso em } a\}.$$

Como todo erro da hipótese pode ser atribuído a pelo menos um literal z presente,

$$\text{error}_c(h) \leq \sum_{z \in h} p(z).$$

Diremos que um literal é prejudicial se $p(z) \geq \epsilon/2n$. Então, se h não contém qualquer literal prejudicial,

$$\text{error}_c(h) \leq \sum_{z \in h} p(z) \leq 2n(\epsilon/2n) = \epsilon,$$

considerando que h contém, no máximo, todos os $2n$ literais possíveis.

Agora, devemos limitar a probabilidade de que um literal prejudicial figure na hipótese, isto é, de que a hipótese seja uma aproximação ruim para o conceito alvo. A probabilidade de que um literal seja removido da hipótese após o processamento de um exemplo é $p(z)$ – a probabilidade de que ele apareça falsificado numa ocorrência positiva de $\mathcal{X}$. Assim, a probabilidade de que um literal z não seja removido da hipótese após o processamento de m exemplos é, no máximo, $(1 - p(z))^m$. Em particular, a probabilidade de que um literal z prejudicial não seja removido de h é, no máximo, $(1 - \epsilon/2n)^m$ e, por fim, a probabilidade de que qualquer literal prejudicial não seja removido de h é, no máximo, $2n(1 - \epsilon/2n)^m$, mais uma vez somando sobre todos os $2n$ literais possivelmente prejudiciais.

Para finalizar esta análise, devemos encontrar o valor de m que satisfaça $2n(1 - \epsilon/2n)^m \leq \delta$, sendo $1 - \delta$ a confiança desejada, ou seja, a probabilidade de que a hipótese tenha erro $\text{error}_c(h) \leq \epsilon$. Usando a desigualdade $1 - x \leq e^{-x}$ temos

$$2n(1 - \epsilon/2n)^m \leq 2ne^{-m\epsilon/2n} \leq \delta$$

e finalmente

$$m \geq (2n/\epsilon)(\ln(2n) + \ln(1/\delta)).$$

Assim, se o algoritmo processar pelo menos este número de exemplos, então com uma probabilidade de no mínimo $1 - \delta$ a hipótese h terá um erro de no máximo ϵ , com relação a qualquer conceito alvo $c \in \mathcal{C}$ sob qualquer distribuição $D \in \mathcal{D}$. Ainda, como o algoritmo leva tempo linear para processar cada exemplo, a complexidade computacional é limitada por mn .

Retomando a definição dada em 3.1, dizemos que este espaço de conceitos é PAC-apreensível. Além disto, é *propriamente* PAC-apreensível, pois a hipótese é expressa no mesmo espaço que os conceitos, isto é, na forma de um monômio. Note que a definição não faz esta exigência, mas admite um outro espaço de hipóteses.

Um espaço de conceitos um pouco mais abrangente que o dos monômios é o das expressões booleanas na *forma conjuntiva normal com até k literais em cada cláusula* ou $k\text{-CNF}$. O espaço dos conceitos conjuntivos é um caso particular: 1-CNF . Valiant [Val84] verifica que são suficientes $m = \frac{2}{\epsilon}[(2n)^{k+1} + \ln(\frac{1}{\epsilon})]$ exemplos positivos para que o erro da hipótese produzida pelo algoritmo da Figura 3.2 seja menor do que ϵ , com probabilidade maior que $1 - \epsilon$.

$h :=$ a conjunção de todas as possíveis cláusulas c_j de até k literais;
para i **de** 1 **até** m **faça**
 obtenha um exemplo positivo x_i do conceito alvo;
 para cada c_j ainda presente em h **faça**
 se c_j não contem literal algum entre os verificados em x_i
 então remova c_j de h ;
devolva a expressão $k\text{-CNF}$ h ;

Figura 3.2: Aprendiz para expressões em $k\text{-CNF}$

A idéia básica do algoritmo é a mesma do aprendiz de monômios. A hipótese h é sempre mais específica que o conceito alvo c : inicialmente, o conjunto das ocorrências que verificam h , é vazio; como toda ocorrência positiva verifica todas as cláusulas em c , nenhuma destas é removida de h pelo processamento dos exemplos positivos; e, finalmente, uma ocorrência que negue c , nega pelo menos uma das cláusulas em c , portanto nega a mesma em h . Por outro lado, após o processamento de um exemplo positivo, todas as cláusulas por ele negadas são removidas da hipótese h e jamais uma cláusula lhe é adicionada, de forma que h será sempre verificada pela ocorrência positiva.

A probabilidade de erro, sempre do tipo “falso negativo”, após o processamento de um certo número de exemplos positivos, é a soma das probabilidade das ocorrências positivas que negam pelo menos uma cláusula ausente em c

¹O próprio autor sugere que seria mais preciso, embora mais trabalhoso, trabalhar com dois parâmetros independentes ϵ e δ , como na definição apresentada na Seção 3.1 acima.

e não removida de h (certamente esta ocorrência não estava presente na amostra de exemplos positivos):

$$error_c(h) = \sum_{\substack{c(x)=1 \\ h(x)=0}} D(x)$$

Este valor está sempre bem definido em cada passo do algoritmo, ou seja, para cada estado intermediário da hipótese h , e é sempre decrescente. A probabilidade de que ocorra um exemplo positivo que negue a hipótese em seu estado corrente é exatamente a probabilidade de erro desta hipótese.

Ao fim do processamento, há duas possibilidades. A primeira é a de que, em algum ponto do processamento, a probabilidade de erro da hipótese h tornou-se menor que ϵ e, com isso, temos uma boa aproximação do conceito c . A outra é de que isto não ocorra e a hipótese h não é uma boa aproximação. Então, a probabilidade de erro da hipótese intermediária, que também é a probabilidade de ocorrência de um exemplo positivo que a negue, manteve-se maior ou igual a ϵ . Mas isto corresponde à execução de m experimentos de Bernoulli (cada exemplo positivo processado), cada um com probabilidade maior ou igual a ϵ de sucesso e à obtenção de menos do que $(2n)^{k+1}$ sucessos (o sucesso, aqui, é a exclusão de pelo menos uma cláusula da hipótese). Valiant demonstra, no mesmo artigo, que a probabilidade desta ocorrência é limitada superiormente por ϵ . Com isto, temos que

$$\Pr \{error_c(h) \leq \epsilon\} \geq 1 - \epsilon,$$

nos moldes da definição de apreensibilidade, com uma pequena diferença: o mesmo parâmetro ϵ é usado, independentemente, para limitar o erro admitido e, também, a probabilidade de que o primeiro seja maior que o admitido.

Agora, após provarem que a classe das expressões booleanas na *forma disjuntiva normal com três termos* ou *3-term-DNF* (disjunção de três monômios) não é PAC-apreensível através do próprio espaço de hipóteses, Kearns & Vazirani [KV94] mostram que conceitos desta classe podem ser representados através da classe *3-CNF*. A conversão de uma expressão em *3-term-DNF* em outra equivalente mas em *3-CNF* pode ser feita em tempo n -polinomial. O inverso nem sempre pode ser feito, pois *3-CNF* é mais expressivo que *3-term-DNF*, mas quando pode, o custo de tempo é da mesma ordem. Assim, se é garantido que os exemplos se referem a um conceito expressível em *3-term-DNF*, pode-se aprender uma expressão *3-CNF* para ele e convertê-la para a forma desejada. Portanto, a classe *3-term-DNF* é *impropriamente* PAC-apreensível através da classe *3-CNF*.

3.3 Espaços finitos

Conforme a definição inicial de PAC-apreensibilidade, esta é, ou não, uma propriedade de um espaço de conceitos. Ainda assim, a demonstração desta propriedade para um espaço pode depender da apresentação de um algoritmo e de uma argumentação específica sobre este. Mas há resultados intermediários mais gerais, a respeito de algumas classes de espaços de conceitos que podem aliviar esta exigência.

No primeiro exemplo da Seção 3.2 observamos que a hipótese produzida pelo algoritmo aprendiz é sempre coerente com a amostra, isto é, classificará corretamente qualquer ocorrência que já houver sido observada na fase de treinamento. Os exemplos seguintes daquela seção têm todos a mesma característica. Dizemos, em casos como estes, que o aprendiz é um *algoritmo consistente* para o espaço de conceitos que aprende. Exibir um algoritmo consistente com a amostra é uma tarefa bem mais simples do que demonstrar sua adequação à definição inicial e, para tirar proveito disto, há uma propriedade de um espaço de conceitos cuja presença garante que ele seja PAC-apreensível, dada a existência de um algoritmo consistente [AB92].

Para introduzi-la, é conveniente denotar por $\mathbf{H}[s]$ o conjunto de todas as hipóteses em $\mathcal{H}$ consistentes com a amostra s para um conceito desconhecido. Assim, um algoritmo A é consistente se e somente se produz uma hipótese $h = A(s) \in \mathbf{H}[s]$ para qualquer amostra s de qualquer conceito em $\mathcal{C}$.

Por outro lado, define-se o conjunto das hipóteses ϵ -ruins para um dado conceito alvo $c \in \mathcal{C}$ como

$$B_\epsilon = \{h \in \mathcal{H} : \text{error}_c(h) \geq \epsilon\}.$$

Um algoritmo consistente para $\mathcal{C}$ produz hipóteses em $\mathbf{H}[s]$ e a definição de PAC-apreensibilidade requer que esta hipótese muito pouco provavelmente pertença a B_ϵ . Isto conduz à definição de que $\mathcal{C}$ é *potencialmente apreensível* se, dados ϵ e δ reais positivos e menores do que 1, existe um inteiro positivo $m_0 = m_0(\epsilon, \delta)$ tal que se $m \geq m_0$,

$$\Pr \{s \in \mathcal{S}(m, c) : \mathbf{H}[s] \cap B_\epsilon = \emptyset\} \geq 1 - \delta,$$

para qualquer distribuição $D \in \mathcal{D}$ sobre $\mathcal{X}$ e qualquer conceito alvo $c \in \mathcal{C}$.

Agora, se um espaço de conceitos $\mathcal{C}$ é potencialmente apreensível e um algoritmo aprendiz A para este espaço é consistente, a condição $\mathbf{H}[s] \cap B_\epsilon = \emptyset$ significa que $\text{error}_c(h = A(s)) \leq \epsilon$ e, segundo a definição de potencialmente apreensível acima, isto ocorre com probabilidade não menor que $1 - \delta$.

Pode parecer tão complexo quanto, ou até mais do que a definição original, mas tem uma implicação muito forte: qualquer espaço de conceitos de

dimensão finita é potencialmente apreensível. Para chegar a este resultado, considerando que $\mathcal{C} \subseteq \mathcal{H}$ e observando que $\forall h_B \in B_\epsilon, \text{error}_c(h_B) \geq \epsilon$, a probabilidade de que h_B seja consistente com c , em uma ocorrência, é

$$\forall h_B \in B_\epsilon, \Pr \{x \in \mathcal{X} : h_B(x) = c(x)\} = 1 - \text{error}_c(h_B) \leq 1 - \epsilon,$$

e a de que o seja sobre toda uma amostra $s \in \mathcal{S}(m, c)$,

$$\forall h_B \in B_\epsilon, \Pr \{\forall x \in s : h_B(x) = c(x)\} \leq (1 - \epsilon)^m.$$

Assim, a probabilidade de que exista alguma hipótese simultaneamente em B_ϵ e em $\mathbf{H}[s]$ é

$$\Pr \{s \in \mathcal{S}(m, c) : \mathbf{H}[s] \cap B_\epsilon \neq \emptyset\} = |\mathcal{H}|(1 - \epsilon)^m.$$

Para que esta probabilidade seja menor do que δ ,

$$|\mathcal{H}|(1 - \epsilon)^m < |\mathcal{H}|e^{-m\epsilon} \leq \delta, \text{ ou}$$

$$m \geq m_0 = \frac{1}{\epsilon} \ln \left(\frac{|\mathcal{H}|}{\delta} \right).$$

Com este resultado, basta a exibição de um algoritmo consistente para demonstrar a PAC-apreensibilidade de qualquer espaço finito de conceitos. Como exemplo, considere a classe das expressões booleanas na *forma disjuntiva normal com até k literais em cada monômio* ou $k\text{-DNF}$. É uma classe finita, com $|k\text{-DNF}| \leq 2^{(2n)^k}$, e o algoritmo da Figura 3.3 é consistente para este espaço de conceitos, aprendendo apenas de exemplos negativos.

$U :=$ a disjunção de todos monômios relevantes de até k literais;

para i **de** 1 **até** m **faça**

 obtenha um exemplo negativo x_i do conceito alvo;

 remova de U todos os monômios u tais que $u(s)$ é verdadeiro;

devolva a expressão $k\text{-DNF}$ U ;

Figura 3.3: Aprendiz consistente para expressões em $k\text{-DNF}$

Monômios relevantes são os não identicamente falsos como $p_1\bar{p}_1$. A consistência do algoritmo é verificada da mesma forma que nos casos de aprendizado de monômios e de expressões $k\text{-CNF}$. A expressão inicial é identicamente verdadeira. A cada exemplo negativo são eliminados os termos que

são verificados na ocorrência em questão, de forma que a expressão resultante não seja verdadeira para esta ocorrência.

Para garantir a precisão e a confiança determinadas pelos parâmetros ϵ e δ , a amostra deve ter tamanho $m \geq m_0$ tal que ²

$$m_0 = \left\lceil \frac{1}{\epsilon} \ln \left(\frac{|k\text{-}\mathcal{DNF}|}{\delta} \right) \right\rceil \\ \leq \left\lceil \frac{1}{\epsilon} \left[(2n)^k \ln 2 + \ln \left(\frac{1}{\delta} \right) \right] \right\rceil.$$

Para a complexidade de tempo deste algoritmo basta verificarmos que para cada um dos m exemplos, devem ser analisados não mais do que $(2n)^k$ monômios ³.

O algoritmo aprendiz de monômios, apresentado na Seção 3.2, é consistente e também serve como exemplo. O espaço dos monômios sobre $\{0, 1\}^n$ tem tamanho 3^n , portanto bastam

$$m_0 = \left\lceil \frac{1}{\epsilon} \left[n \ln 3 + \ln \left(\frac{1}{\delta} \right) \right] \right\rceil$$

para o aprendiz: um cálculo muito mais simples resulta num limite linear em n , enquanto o método mais trabalhoso daquela Seção nos permitiu chegar somente a $O(n \ln n)$.

O algoritmo aprendiz para expressões em $k\text{-}\mathcal{CNF}$, da mesma Seção 3.2, também é consistente e este espaço tem tamanho não maior que $2^{(2n)^k}$. Assim, o número de exemplos que garante a alta probabilidade de uma boa aproximação é o mesmo que para $k\text{-}\mathcal{DNF}$ e é essencialmente igual ao resultado apresentado por Valiant [Val84], com a vantagem de que reintroduz o parâmetro de confiança δ independente do parâmetro de precisão ϵ .

Por outro lado, ainda que finito, o espaço de todas as expressões booleanas tem tamanho 2^{2^n} , o que torna o número de exemplos necessários (segundo este método) pouco útil na prática:

$$m_0 = \left\lceil \frac{1}{\epsilon} \left[2^n \ln 2 + \ln \left(\frac{1}{\delta} \right) \right] \right\rceil.$$

3.4 Listas de decisão

Espaços de representação muito potentes para expressões booleanas são o das Listas de Decisão e o das Árvore de Decisão [KV94]. Para definir estes

²Anthony & Biggs [AB92] exibem o resultado $m_0 = \left\lceil \frac{1}{\epsilon} [k \ln(2n) + \ln \left(\frac{1}{\delta} \right)] \right\rceil$, mas deixam a demonstração em aberto.

³São, no máximo $\binom{2n}{i}$ monômios para i fixo, e $1 + \sum_{1 \leq i \leq k} \binom{2n}{i} \leq (2n)^k$.

espaços, seja $\mathcal{V}$ um espaço de funções booleanas sobre $\mathcal{X} = \{0, 1\}^n$. Então,

$$\mathcal{LD}(\mathcal{V}) = \{l : l = (\langle f_1, b_1 \rangle, \langle f_2, b_2 \rangle, \dots, \langle f_r, b_r \rangle, b_0)\},$$

onde $f_i \in \mathcal{V}, b_i \in \{0, 1\} \forall i \in [1..r]$ e $b_0 \in \{0, 1\}$, é o espaço das listas de decisão baseadas em $\mathcal{V}$. O valor de uma lista de decisão l na ocorrência y é definido por

$$l(y) = \begin{cases} b_j, & \text{se existe } j = \min\{i : f_i(y) = 1\}; \\ b_0, & \text{caso contrário.} \end{cases}$$

Particularmente, para $k\text{-}\mathcal{M} = \{m : m \text{ é monômio de até } k \text{ literais}\}$, podemos definir $k\text{-}\mathcal{LD}(\mathcal{M})$ como o espaço das listas de decisão cujos termos são monômios de até k literais.

Para verificar o poder expressivo de $k\text{-}\mathcal{LD}(\mathcal{M})$, observe que qualquer expressão $k\text{-}\mathcal{DNF}$ pode ser representada neste espaço: escolha um ordem qualquer na qual avaliar os termos da expressão $k\text{-}\mathcal{DNF}$ e faça de cada um deles um termo da lista, associando a saída de todos eles ao valor *verdadeiro* e a saída final da lista ao valor *falso*. Agora, este espaço é fechado sob a complementação: se uma lista representa um conceito c , para representar $\neg c$ basta invertermos todas as saídas; assim, qualquer expressão em $k\text{-}\mathcal{CNF}$ também pode ser expressa por uma k -lista sobre $\mathcal{M}$. Ainda, para cada k , há funções expressas em $k\text{-}\mathcal{LD}(\mathcal{M})$ que não podem ser escritas em $k\text{-}\mathcal{CNF}$ ou $k\text{-}\mathcal{DNF}$.

Para aplicarmos o resultado da Seção 3.3 sobre $k\text{-}\mathcal{LD}(\mathcal{M})$, começamos pelo fato de que $|k\text{-}\mathcal{M}| \leq (2n)^k$ e este é o número máximo de termos da lista. Uma lista com menos termos do que isto é equivalente a uma lista com todos os termos possíveis, em que os termos excedentes são colocados ao final, associados ao mesmo valor b_0 da lista mais curta. Por outro lado, a repetição de termos pode ser desconsiderada: a segunda ocorrência de um termo jamais será avaliada, pois a avaliação terá terminado na primeira ocorrência. Agora, como cada termo pode ser associado a dois diferentes valores de saída, temos $2^{(2n)^k} (2n)^k!$ listas de decisão. Então, são suficientes $m \geq m_0$ exemplos para que um algoritmo consistente de indução de listas de decisão se enquadre na definição da Seção 3.1:

$$\begin{aligned} m_0 &= \left\lceil \frac{1}{\epsilon} \ln \left(\frac{2^{(2n)^k} (2n)^k!}{\delta} \right) \right\rceil \\ &\leq \left\lceil \frac{1}{\epsilon} \left[(2n)^k (\ln 2 + k \ln 2n) + \ln \left(\frac{1}{\delta} \right) \right] \right\rceil. \end{aligned}$$

O algoritmo dado na Figura 3.4, devido a Rivest [Riv87] (segundo [AB92, EH89]) é consistente para o espaço de conceitos representado por $k\text{-}\mathcal{LD}(\mathcal{M})$.

```

seja  $c$  o conceito a ser aprendido
 $l := ()$ ;
 $S := \{\langle x_i, b_i \rangle : 1 \leq i \leq m \text{ e } b_i = c(x_i)\}$ ;
 $F := \{f_i : f_i \in k\text{-}\mathcal{M}\}$ ;
enquanto  $S \neq \emptyset$  faça
    selecione  $f \in F$  e um bit  $t$ 
        tal que  $\forall \langle x, b \rangle \in S, f(x) = 1 \implies b = t$ ;
        se impossível aborte o laço;
    adicione  $\langle f, t \rangle$  a  $l$  antes de  $b_0$ ;
     $S_f := \{x \in S : f(x) = 1\}$ ;
     $S := S \setminus S_f$ ;
     $F := F \setminus \{f\}$ ;
se  $S \neq \emptyset$ 
    então ajuste  $b_0$  igual ao rótulo dos exemplos remanescentes em  $S$ ;
    senão ajuste  $b_0$  conforme a conveniência;
adicione  $b_0$  a  $l$ ;
devolva  $l$ ;

```

Figura 3.4: Aprendiz consistente para listas de decisão em $k\text{-}\mathcal{LD}(\mathcal{M})$

A demonstração deste algoritmo segue da definição de lista de decisão. A única dificuldade é a prova de que ou existe uma escolha para o comando **selecione** ... **tal que** ..., ou todos os exemplos remanescentes são exclusivamente positivos ou negativos. Mas, se o conceito alvo pertence à classe dos conceitos expressíveis por $k\text{-}\mathcal{LD}(\mathcal{M})$, a não existência de uma f que seja verificada por pelo menos um exemplo significa que os exemplos restantes ou são todos positivos ou todos negativos, caso em que a saída *default* da lista de decisão deve ser ajustada de acordo. Por outro lado, se há funções f verificadas por pelo menos um exemplo da amostra, então pelo menos uma delas é verificada por exemplos cujos rótulos b_i são todos iguais, do contrário, o conceito c não seria expressível por este espaço de representação.

3.5 Árvores binárias de decisão

Árvores binárias de decisão são estruturas em árvore [Wir89, CLR90] modificadas de forma que

- cada nó interno i está associado a uma função booleana f_i ;

- cada nó interno i tem uma subárvore associada ao valor falso de f_i e outra ao valor verdadeiro;
- cada nó terminal (folha) está associado a um valor de decisão $b \in \{0, 1\}$.

Uma ocorrência $x \in \mathcal{X}$ define um único caminho iniciado na raiz de uma árvore T e terminado num nó terminal. O valor de decisão associado a este nó terminal é o valor de $T(x)$ (segundo a convenção de que $0 = \text{falso}$ e $1 = \text{verdadeiro}$). O caminho é determinado pelo valor da função associada a cada nó interno: a partir de um nó i , ele segue pela subárvore correspondente a $f_i(x)$. Assim, se considerarmos apenas árvores em que as funções são simples literais e n é a dimensão do espaço de ocorrências, temos um caminho de $O(\log n)$ nós. Isto significa que a avaliação será extremamente rápida.

A definição de árvore binária de decisão pode ser formalizada recursivamente. Um simples nó associado ao valor de decisão $b \in \{0, 1\}$ é uma árvore binária de decisão sobre um espaço de funções booleanas $\mathcal{V}$:

$$T = \langle b \rangle.$$

Se T_0 e T_1 são árvores binárias de decisão sobre $\mathcal{V}$, então

$$T = \langle f, T_0, T_1 \rangle$$

é também uma árvore binária de decisão sobre $\mathcal{V}$ tal que

$$T(y) = \begin{cases} T_0(y), & \text{se } f(y) = 0; \\ T_1(y), & \text{se } f(y) = 1. \end{cases}$$

Cada $\langle \dots \rangle$ na expressão analítica de uma árvore binária de decisão corresponde a um nó interno. As constantes b_0, b_1 na expressão de um nó interno correspondem a nós terminais ou folhas, enquanto as árvores binárias de decisão T_0 e T_1 nesta expressão correspondem a subárvores do nó interno. Chamaremos de $\mathcal{AD}(\mathcal{V})$ o espaço das árvores de decisão sobre um espaço $\mathcal{V}$. Quando o espaço de base for do tipo $k\text{-}\mathcal{V}$, significando um limite k no número de literais presentes em uma cláusula ou termo, a notação será $k\text{-}\mathcal{AD}(\mathcal{V})$.

Por exemplo, tomemos $\mathcal{V} = 1\text{-}\mathcal{M}$, ou seja, monômios com apenas um literal. Então, um conceito $c = \langle u_1(u_2 \vee \bar{u}_3) \vee \bar{u}_1 u_3 \rangle$ sobre $\{0, 1\}^3$ pode ser representado em $1\text{-}\mathcal{AD}(\mathcal{M})$ por:

$$c = \langle u_1, \langle u_3, 0, 1 \rangle, \langle u_2, \langle u_3, 1, 0 \rangle, 1 \rangle \rangle.$$

A repetição de uma função f num caminho da raiz a alguma folha da árvore pode ser descartada, pois ela terá o mesmo valor a cada vez que for avaliada sobre a mesma ocorrência de $\mathcal{X}$. Após a avaliação de f , também não

faz sentido avaliar $\bar{f}$. Podemos, ainda, descartar as árvores incompletas, como fizemos com as listas de decisão, pelo mesmo motivo que naquele caso. Mesmo assim, o número de árvores possíveis em $1\text{-}\mathcal{AD}(\mathcal{M})$ é limitado inferiormente por 2^{2^n} , o que torna pouco interessante a aplicação direta do critério de apreensibilidade potencial.

Por outro lado, o *posto* (do inglês *rank*) de uma árvore é uma medida do balanceamento de uma árvore. Define-se o posto de um nó terminal igual a zero. Se os nós descendentes diretos de um mesmo ancestral têm o mesmo posto r , o posto do ancestral é $r + 1$; se têm postos diferentes r_0 e r_1 , o do ancestral é igual ao maior deles.

Então, seja $k\text{-}\mathcal{AD}(\mathcal{V})_r$ a classe das árvores binárias de decisão sobre $k\text{-}\mathcal{V}$ de posto menor ou igual a r . Kearns & Vazirani [KV94] propõem que toda expressão representável por uma árvore de decisão de posto r sobre $1\text{-}\mathcal{M}$, isto é, em $1\text{-}\mathcal{AD}(\mathcal{M})_r$, é representável também por uma lista de decisão sobre $r\text{-}\mathcal{M}$, ou seja, em $r\text{-}\mathcal{LD}(\mathcal{M})$, assim o mesmo resultado de apreensibilidade desta classe se aplica àquela.

Por outro lado, Ehrenfeucht & Haussler [EH89] fazem a conversão no sentido contrário. Uma lista em $k\text{-}\mathcal{LD}(\mathcal{M})$ pode ser transformada numa árvore em $1\text{-}\mathcal{AD}(\mathcal{M})_r$ pela transformação de cada conjunção da lista numa nova variável. A seguir propõem um algoritmo para aprendizagem através de $1\text{-}\mathcal{AD}(\mathcal{M})_r$ cujo caso particular $r = 1$ se reduz ao algoritmo para listas de decisão da Seção 3.4.

O algoritmo depende das seguintes definições. Seja $s \in \mathcal{S}(m, c)$ uma amostra de tamanho m para o conceito $c \in \mathcal{C}$, e $r(s)$ o menor posto de qualquer árvore consistente com s . Se v é uma variável do espaço de ocorrências, $s_0^v \subseteq s$ é o conjunto de todos os exemplos $\langle x, c(x) \rangle$ tais que $v = 0$ em x ; e $s_1^v \subseteq s$ é o conjunto dos exemplos tais que $v = 1$ em x . Se s_0^v e s_1^v são ambos não vazios, então v é uma variável *informativa*.

O algoritmo aprendiz tem um procedimento recursivo central, na Figura 3.5, que devolve uma árvore de posto r consistente com a amostra, se houver, em tempo $O(m(n+1)^{2r})$.

O algoritmo aprendiz obtém a árvore de menor posto consistente com a amostra executando o procedimento *FIND* acima para $r = 0, 1, 2, \dots$ até que uma árvore seja devolvida. A complexidade de tempo ⁴ é $O(m(n+1)^{2r(s)})$. O tamanho do espaço $1\text{-}\mathcal{AD}(\mathcal{M})_r$ é limitado por $(8n)^{\left(\frac{en}{r}\right)^r}$, assim, a complexidade de amostra é limitada superiormente por

$$m_0 = \frac{1}{\epsilon} \left[\left(\frac{en}{r} \right)^r \ln 8n + \ln \left(\frac{1}{\delta} \right) \right].$$

⁴ $O \left(\sum_{r=0}^{r(s)} O(m(n+1)^{2r}) \right) = O(m(n+1)^{2r(s)})$

função $FIND(s, r)$ – entradas: amostra s e parâmetro inteiro r

se $\forall \langle x, c(x) \rangle \in s, c(x) = 1$ então devolva a árvore $T = \langle 1 \rangle$;

se $\forall \langle x, c(x) \rangle \in s, c(x) = 0$ então devolva a árvore $T = \langle 0 \rangle$;

se $r = 0$ então devolva $NULO$;

para cada v informativo faça

$T_0^v = FIND(s_0^v, r - 1)$;

$T_1^v = FIND(s_1^v, r - 1)$;

se $T_0^v \neq NULO \wedge T_1^v \neq NULO$ então devolva $T = \langle v, T_0^v, T_1^v \rangle$;

se $T_0^v = NULO \wedge T_1^v = NULO$ então devolva $NULO$;

se $T_0^v = NULO$ então $T_0^v = FIND(s_0^v, r)$;

se $T_1^v = NULO$ então $T_1^v = FIND(s_1^v, r)$;

se $T_0^v = NULO \vee T_1^v = NULO$

então devolva $NULO$;

senão devolva $T = \langle v, T_0^v, T_1^v \rangle$;

devolva $NULO$;

Figura 3.5: Aprendiz consistente para árvore binária de posto r

O artigo vai ainda além, utilizando estes resultados para chegar a uma complexidade de amostra subexponencial, linear em $n^{O(\ln n)}$, para a aprendizagem de árvores binárias de decisão sobre $1-\mathcal{M}$ de qualquer posto, o que, de qualquer forma, não caracteriza a PAC-apreensibilidade de $1-\mathcal{AD}(\mathcal{M})$.

As definições desta seção podem ser estendidas para árvores q -árias de decisão, bastando, para isto, que $\mathcal{V}$ seja um espaço de funções q -valoradas e que cada nó interno contenha q sub-árvores. Ou seja, se

- $k-\mathcal{AD}(\mathcal{V})^q$ é o espaço destas árvores,
- $f \in k-\mathcal{V} : \mathcal{X} \mapsto \{0, 1, \dots, q-1\}$ e
- $0 \leq i \leq q-1 ; T_i \in k-\mathcal{AD}(\mathcal{V})^q$,

então

$$T = \langle f, T_0, T_1, \dots, T_{q-1} \rangle$$

é uma árvore q -ária de decisão sobre $\mathcal{V}$ tal que $f(x) = i \implies T(x) = T_i(x)$.

Entretanto, uma árvore q -ária pode ser transformada em uma árvore binária de decisão, com custo de tempo polinomial em n , considerando-se uma subárvore para um resultado possível e outra única para todos os outros, a ser expandida da mesma maneira, até que haja apenas dois resultados possíveis:

$$T = \langle f, T_0, \langle f, T_1, \langle f, T_2, \langle \dots \langle f, T_{q-2}, T_{q-1} \rangle \rangle \rangle \rangle \rangle.$$

3.6 Outros modelos formais

Nesta seção serão apresentados modelos semelhantes ao modelo PAC. Estes modelos propõem formulações equivalentes e estendem os resultados do modelo PAC para casos para os quais este, na sua versão elementar apresentada acima, não é adequado.

3.6.1 Consultas estatísticas

Uma das simplificações menos realistas do modelo básico da aprendizagem PAC é a suposição de que a amostra é consistente e correta, em relação ao conceito alvo. Em muitas áreas de aplicação da aprendizagem computacional, pode ocorrer a necessidade de que o aprendiz lide com a possibilidade de valores incorretos na amostra, correspondentes a erros de medição ou de classificação pelo usuário. Torna-se desejável, então, que o aprendiz tenha algum grau de tolerância a erros ocasionais.

Angluin e Laird, conforme [KV94], propuseram um novo modelo em que se admite um erro de classificação da ocorrência na amostra de treinamento, com probabilidade $0 \leq \eta < \frac{1}{2}$: a *aprendizagem por consultas estatísticas*. Este modelo é, na verdade, uma especialização e uma restrição ao modelo PAC: o aprendiz é proibido de examinar o rótulo de classificação de cada exemplo individual na amostra e deve derivar sua hipótese exclusivamente com base em estimativas de probabilidades sobre os valores dos atributos e dos rótulos na amostra.

Mantendo a suposição de conceitos booleanos, a diferença deste modelo em relação ao modelo PAC é que a classificação de cada exemplo tem probabilidade η de estar invertida. Assim, é natural que a complexidade de amostra deva ser tanto maior quanto mais η estiver próximo de $\frac{1}{2}$. Se este limite for atingido, os rótulos de classificação não mais contêm informação alguma e correspondem ao lançamento de uma moeda perfeita. Por simplicidade, Kearns & Vazirani [KV94] admitem que o aprendiz recebe como informação adicional um limite superior η_0 para a taxa de ruído efetivamente presente na amostra, embora mencionem que esta suposição pode ser removida. Com isto, a complexidade de amostra pode depender polinomialmente, também, de $\frac{1}{1-2\eta_0}$ (ou de $\frac{1}{1-2\eta}$, após a remoção desta suposição).

Para motivar a necessidade de algumas modificações nos algoritmos vistos anteriormente, basta lembrar que o algoritmo aprendiz de monônios remove da hipótese um literal que aparece falsificado num exemplo positivo. Se há a possibilidade de algum exemplo negativo estar erroneamente classificado como positivo, o aprendiz poderá cometer o erro de remover um literal que não deveria ser removido. Como o algoritmo não prevê a reinclusão de literais

na hipótese, este erro será permanente e fatal.

Considere então as seguintes idéias para um algoritmo aprendiz de monômios, ainda no contexto livre de ruído. Para cada literal z , seja $p_0(z)$ a probabilidade de que z esteja negado numa ocorrência $x \in \mathcal{X}$ sob $D \in \mathcal{D}$. Se $p_0(z)$ é pequena, z pode, intuitivamente, ser ignorado, dado que aparece quase sempre verificado. Agora, seja $p_{01}(z)$ a probabilidade de que z esteja negado em uma ocorrência $x \in \mathcal{X}$ sob $D \in \mathcal{D}$ que seja um exemplo positivo do conceito $c \in \mathcal{C}$. Então, se $p_{01}(z)$ é grande, é desejável não incluir z na hipótese, já que há grandes chances de que ele ocorra negado num exemplo positivo de c e uma hipótese que contenha z vai classificar tal ocorrência como negativa. Diremos que z é *significante* se $p_0(z) \geq \frac{\epsilon}{8n}$ e é *danoso* se $p_{01}(z) \geq \frac{\epsilon}{8n}$. Observe que, como $p_{01}(z) \leq p_0(z)$, todo literal danoso é também significativo.

A seguir, verificamos que se h é a conjunção de todos os literais significativos e não danosos, então $\Pr\{error_c(h) \leq \epsilon\} \geq 1 - \delta$. Primeiro, considere o erro do tipo falso positivo na classificação de uma ocorrência pela hipótese: $c(x) = 0 \wedge h(x) = 1$. Este erro ocorre quando um literal figura em c mas não em h , e é negado na ocorrência. Tal literal só pode ser insignificante. Então, a probabilidade do falso positivo é, no máximo, a probabilidade de que um literal insignificante esteja negado numa ocorrência. E esta, pela definição de significativo acima e pelo limite da união (*union bound*), é limitada superiormente por $2n \frac{\epsilon}{8n} = \frac{\epsilon}{4}$. Inversamente, o erro do tipo falso negativo, $c(x) = 1 \wedge h(x) = 0$, ocorre quando um literal negado na ocorrência figura em h mas não em c . Como h não contém literais danosos, a probabilidade de ocorrência de algum literal assim é limitada superiormente por $2n \frac{\epsilon}{8n} = \frac{\epsilon}{4}$. Finalmente, como estes dois tipos de erro cobrem todas as possibilidades de erro, $error_c(h) \leq \frac{\epsilon}{4} + \frac{\epsilon}{4} = \frac{\epsilon}{2}$.

Estas idéias sugerem um novo algoritmo PAC-aprendiz eficiente para monômios, ainda no contexto livre de ruído. As probabilidades $p_0(z)$ e $p_{01}(z)$ para cada literal podem ser estimadas por $\hat{p}_0(z)$ e $\hat{p}_{01}(z)$ com erro aditivo não superior a $\frac{\epsilon}{8n}$, a partir de uma amostra de treinamento suficientemente grande (mas ainda polinomial em n , $\frac{1}{\epsilon}$ e $\frac{1}{\delta}$). Então, com auxílio dos limites de Chernoff (expostos no apêndice de [KV94]), chegamos a $\Pr\{error_c(h) \leq \epsilon\} \geq 1 - \delta$.

Com esta motivação, [KV94] e [Kea93] formalizam o modelo em que as consultas estatísticas são simuladas sobre a amostra e demonstram que, mesmo na presença de ruídos de classificação, os requisitos da definição de aprendizagem são satisfeitos. O tamanho da amostra é feito suficientemente grande para garantir a precisão das consultas estatísticas, conforme um parâmetro derivado dos parâmetros originais ϵ e δ . Muitos espaços de conceitos PAC-apreensíveis na definição básica também o são neste modelo, mas este não se sobrepõe ao modelo original pois há espaços de conceitos que não são

apreensíveis neste modelo mas o são no modelo PAC básico.

Desenvolvimentos adicionais neste modelo, desenvolvidas em [Kea93] e referências aí incluídas, tratam de problemas cada vez mais realísticos ou voltados para aplicações específicas, como taxa de ruído variável no tempo, ruído nos valores dos atributos e o ruído malicioso, que não é exatamente um ruído aleatório, mas uma interferência introduzida por um adversário nos valores e rótulos observados, com o intuito de dificultar ou impossibilitar a aprendizagem. Os autores, em [KV94], mencionam também trabalhos sobre tolerância a ruídos na aprendizagem *on-line* e na aprendizagem de conceitos probabilísticos.

3.6.2 A navalha de Occam

O modelo PAC de aprendizagem parte da premissa que os exemplos são extraídos do espaço de ocorrências sob uma distribuição *fixa* de probabilidades para conseguir medir o poder de predição da hipótese. Um outro modelo explora a capacidade de uma hipótese de explicar sucintamente uma observação, isto é, uma amostra. O processo de aprendizagem, entretanto, tem como objetivo maior produzir teorias e modelos com poder preditivo sobre ocorrências ainda não conhecidas. Nesta Seção, veremos que, quando restrito ao mesmo contexto probabilístico do modelo PAC, este modelo tem tal poder [KV94].

Uma formulação moderna da doutrina pregada pelo teólogo medieval William Ockham (ou William de Occam) é: “a complexidade deve ser explorada ao nível mínimo necessário para a explicação do observado”. É com base neste princípio, chamado de *Navalha de Occam*, que se formulou o modelo que leva o mesmo nome.

Em termos gerais, o que este modelo propõe é a busca de um algoritmo que produza uma hipótese $h \in \mathcal{H}$ consistente com a amostra s de tamanho m para um conceito $c \in \mathcal{C}$ com a propriedade de crescer lentamente com o tamanho $|\mathcal{X}|$ do espaço de ocorrências, com o tamanho m da amostra s de treinamento (ou seja, com os inversos dos parâmetros de precisão ϵ e de confiança δ) e com o tamanho da menor representação possível do conceito c no espaço $\mathcal{H}$ de representação das hipóteses.

Para formalizar a noção de tamanho de uma representação, pode-se determinar o alfabeto $\mathcal{A}$ em que as representações de um espaço são expressas e codificá-lo em notação binária, usando palavras de $\lceil \log_2 |\mathcal{A}| \rceil$ bits. Às vezes, um espaço pode admitir várias representações para um mesmo conceito. Ainda assim, definimos $\text{tam}(c)$ como o tamanho da menor representação de um conceito no espaço, usando a codificação binária acima; o tamanho de uma hipótese é anotado $\text{tam}(h)$

A definição formal de *Algoritmo Occam* é a seguinte.

Sejam $\alpha \geq 0$ e $0 \leq \beta < 1$ constantes. Então, L é um algoritmo (α, β) -Occam para o espaço de conceitos $\mathcal{C}$ usando o espaço de hipóteses $\mathcal{H}$ se recebe uma amostra $s \in \mathcal{S}(m, c)$ e devolve uma hipótese $h \in \mathcal{H}$ consistente com a amostra e com tamanho limitado por $\text{tam}(h) \leq (n \times \text{tam}(c))^\alpha m^\beta$. Tal algoritmo será dito *eficiente* se seu tempo de execução é limitado por um polinômio em m , n e $\text{tam}(c)$.

A hipótese produzida por um algoritmo Occam é sucinta no sentido de que sua expressão num alfabeto determinado é menor do que a expressão da amostra inteira no mesmo alfabeto. Isto é o que determina o expoente β do tamanho da amostra na condição que restringe o tamanho da hipótese. Esta condição é muito mais frouxa do que seria a exigência de que $\text{tam}(h)$ dependesse somente de $\text{tam}(c)$, e isto torna viável a aprendizagem de muitas classes que, de outra forma, seria um problema computacionalmente complexo. Isto porque o problema de encontrar a menor representação de um conceito num espaço é um problema NP-completo [AB92].

Kearns & Vazirani [KV94] demonstram que um algoritmo (α, β) -Occam eficiente para $\mathcal{C}$ usando $\mathcal{H}$ produz, com probabilidade superior a $1 - \delta$, hipóteses com erro inferior a ϵ em tempo polinomial nos inversos destes dois parâmetros e em n , como exige a definição de PAC-apreensibilidade. A complexidade de amostra é dada pela expressão

$$m_0 = \left\lceil a \left[\frac{1}{\epsilon} \ln \left(\frac{1}{\delta} \right) + \left(\frac{(n \times \text{tam}(c))^\alpha}{\epsilon} \right)^{\frac{1}{1-\beta}} \right] \right\rceil,$$

em que a é uma constante positiva.

Os mesmos autores estabelecem outra relação, mais próxima da teoria da apreensibilidade potencial exposta na Seção 3.3, exigindo que

$$m_0 = \left\lceil \frac{1}{b\epsilon} \left[\ln |\mathcal{H}_m| + \ln \left(\frac{1}{\delta} \right) \right] \right\rceil,$$

onde b é uma constante positiva e $\mathcal{H}_m$ é o espaço das hipóteses tais que $\text{tam}(h) \leq m$.

A demonstração da segunda segue a mesma linha da apreensibilidade potencial. Se h_B é uma determinada hipótese ϵ -ruin, a probabilidade de que seja consistente com uma amostra de tamanho m do conceito c é não maior que $(1 - \epsilon)^m$. Então, a probabilidade de que qualquer hipótese ϵ -ruin seja consistente com uma amostra de tamanho m é não maior que $|B_\epsilon|(1 - \epsilon)^m$ ou,

mais forte ainda, não maior que $|\mathcal{H}_m|(1 - \epsilon)^m$. Esta probabilidade deve ser menor que δ e, isolando m (usando ainda que $\ln(1/(1 - \epsilon)) = \Theta(\epsilon)$), obtém-se a expressão acima.

A partir desta, a primeira é facilmente obtida. Se se trata de um algoritmo (α, β) -Occam, toda hipótese que produzir será menor que $(n \times \text{tam}(c))^\alpha m^\beta$, e portanto $|\mathcal{H}_m| \leq 2^{(n \times \text{tam}(c))^\alpha m^\beta}$. Substitui-se este valor na segunda expressão acima e, para satisfazê-la basta um m maior que o dobro que qualquer uma das parcelas. E, escolhendo $a = 2/b$, temos a primeira expressão.

Observe que a expressão obtida aqui é essencialmente igual à obtida na Seção 3.3. Agora, já vimos que o algoritmo aprendiz para $k\text{-}\mathcal{LD}(\mathcal{M})$ apresentado na Seção 3.4 é consistente e lhe aplicamos o resultado da apreensibilidade potencial. Considerando que qualquer lista de decisão pode ser codificada em $O(n \ln n)$ bits, podemos diminuir m_0 para

$$m_0 = \left\lceil c_1 \frac{1}{b\epsilon} \left[n \ln n + \ln \left(\frac{1}{\delta} \right) \right] \right\rceil,$$

eliminando o expoente k da dependência em n .

3.6.3 Convergência uniforme

Diversos modelos vistos até agora baseiam-se na cardinalidade do espaço de hipóteses para determinar a apreensibilidade de um espaço de conceitos: a complexidade de amostra, ou seja, o tamanho mínimo da amostra para garantir a precisão e a confiança desejadas, depende, em geral, de $\ln |\mathcal{H}|$. Então temos um problema quando $\mathcal{X}$ é um espaço infinito. Blumer, Ehrenfeucht, Haussler e Warmuth [BEHW89] substituem esta dependência em $|\mathcal{H}|$ pela *dimensão Vapnik-Chervonenkis* ou $VCdim$ deste espaço, afirmando que existe um algoritmo aprendiz para $\mathcal{C}$ usando $\mathcal{H}$ se e somente se $VCdim(H)$ é finita.

A apreensibilidade de um espaço de conceitos, no contexto da convergência uniforme, tem essencialmente a mesma definição que no modelo PAC, isto é, a existência de um algoritmo aprendiz que produza hipóteses que, provavelmente ($p \geq 1 - \delta$), sejam uma boa aproximação ($\text{error}_c(h) \leq \epsilon$) para qualquer conceito alvo $c \in \mathcal{C}$, a partir de uma amostra, extraída de $\mathcal{X}$ sob uma distribuição de probabilidades determinada, rotulada de acordo com o conceito alvo e cujo tamanho dependa não mais que polinomialmente de n , $\frac{1}{\epsilon}$ e $\frac{1}{\delta}$. O sentido de *uniforme* nas expressões “convergência uniforme” e “uniformemente apreensível” se refere ao fato de que as boas propriedades do algoritmo aprendiz devem valer uniformemente para qualquer conceito alvo no espaço de conceitos e para qualquer distribuição de probabilidades do espaço de ocorrências – como no modelo PAC.

A dimensão VC de um espaço $\mathcal{C}$ (ou $VCdim(\mathcal{C})$) é definida como a seguir. Dados um espaço de conceitos $\mathcal{C}$ sobre um espaço de ocorrências $\mathcal{X}$ e uma m -amostra $s \in \mathcal{X}^m$, seja $\Pi_{\mathcal{C}}(s)$ o conjunto de todos os subconjuntos de s obteníveis como a intersecção entre s e um conceito $c \in \mathcal{C}$. Isto é, $\Pi_{\mathcal{C}}(s) = \{s \cap c : c \in \mathcal{C}\}$.

Se $\Pi_{\mathcal{C}}(s) = 2^s$, isto é, se cada subconjunto de s está contido em algum conceito $c \in \mathcal{C}$, diz-se que s é quebrada por $\mathcal{C}$. Então, $VCdim(\mathcal{C})$ é a cardinalidade da maior amostra s que é quebrada por $\mathcal{C}$. Se amostras arbitrariamente grandes podem ser quebradas por $\mathcal{C}$, diz-se que $VCdim(\mathcal{C})$ é infinita. Note-se que $s = \emptyset$ sempre é quebrada, por qualquer espaço de conceitos, assim, $VCdim(\mathcal{C}) \geq 0$. Observe-se, ainda, que $\Pi_{\mathcal{C}}$ é uma propriedade de cada amostra em particular, e pode haver amostras quebráveis por $\mathcal{C}$ maiores que outras não quebráveis. Então, a existência de uma única amostra s quebrável por $\mathcal{C}$ garante que $VCdim(\mathcal{C}) \geq |s|$.

Como exemplo, considere $\mathcal{X} = \mathbb{R}$ e $\mathcal{C}$ o espaço $\mathcal{I}_{\mathbb{R}}$ dos intervalos, fechados ou abertos, sobre $\mathbb{R}$. Então, para qualquer 2-amostra $s = \{x_1, x_2\}$, é possível encontrar intervalos que contenham, respectivamente, $\{x_1\}$, $\{x_2\}$, $\{x_1, x_2\}$ e $\{\}$. Por outro lado, para nenhuma 3-amostra $s = \{x_1, x_2, x_3\}$, considerando-se sem perda de generalidade que $x_1 \leq x_2 \leq x_3$, haverá um intervalo que contenha x_1 e x_3 mas não x_2 . Com isto, $VCdim(\mathcal{I}_{\mathbb{R}}) = 2$.

Generalizando para $\mathcal{X} = \mathbb{R}^n$, seja $\mathcal{C}$ o espaço $\mathcal{I}_{\mathbb{R}^n}$ dos n -paralelepípedos sobre $\mathbb{R}^n$, pode-se verificar que uma amostra contendo os $2n$ pontos localizados nos centros das faces do n -cubo unitário é quebrada por $\mathcal{I}_{\mathbb{R}^n}$. Por outro lado, se a amostra contém $2n+1$ pontos, então seja R o n -paralelepípedo que contém todos os pontos da amostra: como R tem apenas $2n$ faces, deve haver pelo menos um ponto x que está ou no interior de R ou divide uma face com outro ponto. Portanto, qualquer conceito que inclua os outros pontos, inclui x , e não há conceito correspondente a $s \setminus \{x\}$. Assim, $VCdim(\mathcal{I}_{\mathbb{R}^n}) = 2n$.

No caso de espaços finitos, observe que são necessários 2^d conceitos distintos para quebrar uma d -amostra. Então, nenhuma amostra de cardinalidade maior que $\log_2 |\mathcal{C}|$ pode ser quebrada. Portanto, $VCdim(\mathcal{C}_{finito}) \leq \log_2 |\mathcal{C}|$.

Antes de dar limites para a complexidade de amostra, os mesmos autores definem ainda a trivialidade de um espaço de conceitos: um espaço de conceitos $\mathcal{C} \subseteq 2^{\mathcal{X}}$ é trivial se contém apenas um conceito ou, ainda, se contém dois conceitos disjuntos c_1 e c_2 tais que $c_1 \cup c_2 = \mathcal{X}$. Não será necessário mais do que um exemplo para aprender os conceitos em espaços assim.

Agora, se $\mathcal{C}$ é uma classe de conceitos não trivial e *bem comportada*⁵

⁵Os autores definem o que é um espaço de conceitos bem comportado no apêndice do artigo citado e mencionam que “virtuamente qualquer espaço de conceitos interessante no contexto da aprendizagem computacional é bem comportado”.

então:

1. $\mathcal{C}$ é uniformemente apreensível se e somente se $VCdim(\mathcal{C}) = d \leq \infty$;
2. Se $VCdim(\mathcal{C}) = d \leq \infty$, então
 - (a) para $0 \leq \epsilon \leq 1$ e tamanho de amostra $m \geq m_0$:

$$m_0 = \max \left\{ \frac{4}{\epsilon} \log_2 \left(\frac{2}{\delta} \right), \frac{8d}{\epsilon} \log_2 \left(\frac{13}{\epsilon} \right) \right\},$$

qualquer algoritmo consistente para $\mathcal{C}$ é um aprendiz para $\mathcal{C}$.

- (b) para $0 \leq \epsilon \leq \frac{1}{2}$ e tamanho de amostra $m \leq m_{min}$:

$$m_{min} = \max \left\{ \frac{1-\epsilon}{\epsilon} \ln \left(\frac{1}{\delta} \right), d(1 - 2(\epsilon(1 - \delta) + \delta)) \right\},$$

nenhum algoritmo é um aprendiz uniformemente convergente para $\mathcal{C}$, usando qualquer espaço de hipóteses $\mathcal{H}$.

Observe que a dependência em $d = VCdim(\mathcal{C})$ só figura num dos termos de $\max(\dots)$. O que ocorre aqui não é o mesmo que no exemplo da Seção 3.2 sobre o espaço das expressões $k\text{-}\mathcal{CNF}$, em que Valiant [Val84] substituiu deliberadamente o parâmetro de confiança pelo de precisão. Se a dimensão VC do espaço de conceitos for pequena e, também, a precisão (ϵ alto), o primeiro termo domina a expressão. Caso o segundo termo domine a expressão, os autores demonstram que o parâmetro de confiança é obedecido.

O maior acréscimo que este modelo trouxe aos modelos formais de aprendizagem é o limite mínimo para o tamanho da amostra, abaixo do qual se demonstra ser impossível obter a convergência uniforme. Obviamente, um determinado algoritmo para aprendizagem de um conceito específico em $\mathcal{C}$ e sob uma distribuição de probabilidades conhecida das ocorrências pode desempenhar bem sua função com um número ainda menor de exemplos, mas esta é uma questão explicitamente fora do escopo deste modelo, como sugere o próprio nome.

3.6.4 Identificação no limite

Gold (cf. [Mel92, BG96]) considera o problema do aprendizado de linguagens, mas seu formalismo pode ser aplicado à aprendizagem de conceitos, de maneira geral. O processo de aprendizagem se dá em passos discretos, em cada um dos quais o aprendiz recebe um novo elemento de informação sobre o conceito alvo; tendo recebido o elemento de informação mais recente o aprendiz seleciona ou constrói uma hipótese consistente com todos os elementos

de informação já recebidos. Dois protocolos de transmissão de informações são possíveis: uma sequência de exemplos a ser fornecida, passo a passo, ao aprendiz; ou consultas que o aprendiz faz a um informante ou oráculo.

O conceito alvo é dito *identificado no limite* se após um número finito de passos, todas as hipóteses fornecidas pelo aprendiz são iguais. Um espaço de conceitos é dito *identificável no limite* se existe um algoritmo aprendiz que identifica no limite qualquer conceito do espaço, em um número finito de passos, com qualquer sequência de treinamento.

Um exemplo geral é o dos espaços enumeráveis de conceitos. O aprendiz toma uma lista dos conceitos numa ordem qualquer e, a cada exemplo, descarta os conceitos contraditos pelo elemento de informação que acabou de receber. Todos os conceitos diferentes do conceito alvo e que figuram na lista antes deste serão descartados e o conceito alvo jamais será descartado.

Outra forma geral é a da progressão do específico para o geral, que é o caso do aprendiz de monômios da Seção 3.2. O aprendiz é um mecanismo de busca que começa com a hipótese mais específica possível e vai generalizando para incluir os exemplos positivos encontrados.

Infelizmente, este modelo não exige que o aprendiz saiba quando chegou a uma hipótese correta, ou mesmo suficientemente boa, nem fornece qualquer medida de erro.

3.7 Modelos de inferência estatística

Os modelos formais vistos até agora são do tipo *pior caso*, no sentido de que os limites obtidos para a complexidade de amostra devem ser estabelecidos para qualquer conceito num espaço de conceitos e para qualquer distribuição de probabilidades no espaço de ocorrências. Esta Seção apresenta modelos de aprendizagem baseados na inferência estatística, que levam em conta as informações sobre os espaços de ocorrências e de conceitos disponíveis na particular amostra de treinamento.

3.7.1 Inferência bayesiana

O modelo de *aprendizagem bayesiana*, baseado na inferência bayesiana, objetiva encontrar a melhor hipótese para aproximar um conceito alvo com base não só nos exemplos e na estrutura do espaços de ocorrências e de hipóteses, mas também num conhecimento prévio do domínio do problema, que se traduz no uso da distribuição de probabilidades *a priori* das hipóteses ou dos conceitos e, ainda, da distribuição de probabilidades sobre as possíveis amostras, condicionadas às probabilidades de cada hipótese ou conceito [Che95].

Formalmente, a estrutura da indução bayesiana tem como passo inicial a determinação

- do espaço de hipóteses (ou espaço de representação) $\mathcal{H}$;
- da probabilidade *a priori* das diversas hipóteses: $\Pr\{h_i : h_i \in \mathcal{H}\}$;
- da probabilidade das possíveis amostras, condicionadas às das hipóteses: $\Pr\{s|h_i : s \in \mathcal{S}(m, c)\}$.

Então, pelo Teorema de Bayes, a probabilidade de cada hipótese h_i condicionada à amostra s é

$$\Pr\{h_i|s\} = \frac{\Pr\{h_i\} \Pr\{s|h_i\}}{\sum_j \Pr\{h_j\} \Pr\{s|h_j\}}.$$

Por fim, roda-se o algoritmo escolhido para procurar a hipótese que maximiza este valor ou, alternativamente, uma para a qual este valor seja suficientemente bom - balanceando a otimalidade da hipótese com o tempo de busca.

Este modelo deixa bem claro a importância do espaço de representação escolhido: embora seja possível encontrar a melhor hipótese neste espaço, ela pode ser tão inadequada para o problema prático a que deve ser aplicada que a conclusão é imediata – o espaço de representação foi mal escolhido. Por outro lado, este modelo não torna claro o número de exemplos necessários para garantir um bom aprendizado. Wolpert [Wol95b] apresenta o que chama de *Formalismo Bayesiano Estendido*, em que pretende preencher esta lacuna, mas não torna claro o resultado.

3.7.2 Modelos físico-estatísticos

O surgimento de modelos empíricos de aprendizagem computacional com número muito grande de parâmetros – como as redes neurais e os pesos de suas diversas interconexões – motivou a analogia entre o espaço dos parâmetros do modelo e o espaço de configuração de sistemas físicos complexos, com os estudados pela termodinâmica e pela mecânica estatística [Tis95].

Nesta analogia, o erro da hipótese corrente é associado à energia do sistema, e o número de hipóteses consistentes com a amostra de treinamento é associado à sua entropia. Se a hipótese é representada por um vetor $\vec{h}$ cujas componentes são os valores dos parâmetros do modelo, então:

$$\text{error}_{\vec{c}}(\vec{h}) = \sum_{i=1}^m \epsilon(\vec{c}(x_i), \vec{h}(x_i)),$$

onde ϵ é uma medida do erro da hipótese sobre uma ocorrência, é o erro da hipótese sobre a amostra. Idealmente, desejamos minimizar o erro da hipótese sobre todo o espaço de ocorrências, e assim:

$$error_{\vec{c}}(\vec{h}) = \int_{x \in \mathcal{X}} dD(x) \epsilon(\vec{c}(x), \vec{h}(x)).$$

A aprendizagem, usualmente, se dá pelo método do gradiente decrescente. A expressão

$$\frac{\partial \vec{h}}{\partial t} = -\nabla_{\vec{h}} error_{\vec{c}}(\vec{h})$$

dá a correção a ser realizada sobre as componentes de $\vec{h}$ a cada evolução da hipótese.

A possibilidade de amostras inconsistentes, ou de presença de ruído é tratada pela inclusão de um termo associado à temperatura. Isto implica a adição de um termo aleatório, representado por um ruído branco, na expressão da evolução da hipótese, o que por sua vez, diminui as chances de que a busca termine num mínimo local do espaço de configuração.

Se este termo decresce lentamente, durante o treinamento, o mínimo global da função energia do sistema pode ser atingido. Entretanto, se o erro da hipótese numa ocorrência é zero quando a hipótese e o conceito alvo coincidem, a energia total zero do sistema pode indicar um ajuste excessivo do aprendiz às particularidades da amostra e, possivelmente, ao ruído presente. Assim, a manutenção de uma “temperatura” finita e positiva pode ser benéfica ao processo de aprendizagem.

Além disto, a temperatura introduz uma energia livre no sistema, associada na analogia termodinâmica ao potencial de Gibbs F . A relação entre este potencial, a energia devida ao erro, a temperatura T e a entropia S

$$F = error_{\vec{c}}(\vec{h}) - TS$$

permite avaliar a expectativa do erro da hipótese sobre a amostra e relacioná-lo à expectativa de erro sobre todo o espaço de ocorrências. Além disto, se o número de exemplos na amostra é proporcional ao número de parâmetros do modelo, ambas as expectativas podem ser limitadas, embora, mais uma vez, a relação não fique clara no artigo citado.

Capítulo 4

Abordagens práticas

Grande parte da recente história da aprendizagem computacional está baseada na construção e no refinamento de algoritmos e estruturas de dados para representação do conhecimento de forma aplicável a situações que exigem este conhecimento.

Muitas vertentes desta história são baseadas nas clássicas teorias sobre o conhecimento, expostas pela primeira vez por Sócrates, conforme os diálogos *Meno* e *Parmênides* de Platão e por Aristóteles, em seu *Organon*. E, embora a *representação clássica de conceitos* [Tho92] já não seja um modelo plausível de representação mental, ainda é uma direção segura e concreta para a derivação mecânica de conceitos. Segundo esta teoria fundamental, a representação de um conceito é uma descrição sumária de uma classe de objetos. Esta descrição é feita a partir de um conjunto necessário e suficiente de atributos e seus respectivos valores compartilhados por todos os membros da classe. Assim, um algoritmo clássico de aprendizagem de conceitos seria a construção de uma lista de valores possíveis de atributos e a eliminação daqueles não verificados nos membros conhecidos da classe. Alternativamente, pode-se construir a lista pela observação dos valores dos atributos verificados em todos os membros conhecidos da classe. Mecanismos de *generalização* e *especialização* de atributos e seus valores, pilares das ciências relacionadas à aprendizagem animal e humana, deram origem a diversas linhas de pesquisa em aprendizagem computacional, como a indução de regras de produção, de listas de decisão e de árvores de decisão.

Outra possibilidade, também derivada de teorias sobre o funcionamento do cérebro animal, é o esquema de *recompensa ou punição* atribuídos a decisões do algoritmo em sua fase de aprendizagem – analogamente ao desenvolvimento de uma criança, uma fase em que a recompensa não pode ser tal que congele o aprendizado nem a punição tal que destrua o aprendiz. Há, ainda, a vertente conexionista, cujas vedetes são as redes neurais, es-

truturas em que pequenas unidades de programação têm suas interligações estimuladas ou inibidas segundo um esquema mais complexo de recompensas e punições [Mel92]. Esta categoria de algoritmos pode ser classificada, alternativamente, como um caso de *ajuste de curvas*, classe cujo exemplo mais simples é a regressão linear. Aqui, a aprendizagem se dá através do ajuste de parâmetros numéricos que determinam o funcionamento de uma unidade de processamento. Outro caso de origem biológica é o dos algoritmos genéticos, originalmente chamados algoritmos de aprendizagem por evolução simulada, em que gerações de algoritmos não aptas a resolver os problemas de um ambiente são eliminadas enquanto as aptas são preservadas e submetidas a repetidas mutações. Nota-se neste processo, um conceito mais abstrato – a aprendizagem através da *busca num espaço de conceitos*.

Por fim, outra vertente baseada em modelos matemáticos abstratos é a da *otimização* em que uma estrutura representativa de conhecimento é transformada em uma nova, mais útil, mais eficiente, mais compacta ou qualquer combinação destas características desejáveis. O exemplo fundamental, neste caso, é o da indução de programas lógicos.

Apesar das diversas tentativas de se construir uma taxonomia dos algoritmos de aprendizagem computacional, as relações e as características comuns são tantas que se torna mais prático examinar algumas categorias de algoritmos mais detalhadamente para formar um panorama mais consistente de suas possibilidades, com ênfase sobre as árvores de decisão, categoria da implementação que será analisada sob a ótica dos modelos formais nos capítulos seguintes.

A escolha das árvores de decisão para maior aprofundamento e posterior análise se deve à carência de análises formais para este tipo de implementação na literatura a respeito de aprendizagem computacional. O algoritmo C4.5 [Qui87, Qui96] foi escolhido para exemplificar as diversas considerações que se fazem necessárias na construção de um algoritmo de indução de árvores de decisão pela vasta gama de variações que já sofreu e pela ampla literatura a seu respeito. Por outro lado, o algoritmo REAL [SNLR98b, SNLR98a, Lau96] foi utilizado no experimento prático pela disponibilidade, proximidade e apoio de seus construtores.

4.1 Representação clássica de conceitos

Considerando-se os atributos relevantes a uma determinada categorização como eixos independentes de um espaço, uma categoria de objetos corresponde a uma região neste mesmo espaço.

Se tomamos os atributos “cor” e “forma geométrica” (consideradas aqui

apenas as regulares, por aproximação), podemos dizer que as maçãs são objetos vermelhos, amarelos ou verdes, de forma aproximadamente esférica. Nenhum objeto azul ou com a forma de um prisma hexagonal é uma maçã. Os lugares geométricos determinados por estes valores, no espaço *cor* $\times$ *forma*, não têm intersecção com a região determinada pela forma e possíveis cores da maçã.

O desenvolvimento de uma criança fornece um outro exemplo prático: ao longo de sua fase de crescimento, seu pediatra mede sua estatura e seu peso, anotando-os em diagramas já preparados, muito comuns nos consultórios pediátricos. Um destes diagramas é um plano bidimensional, cujos eixos são as medidas da altura e do peso da criança. O papel graduado tem uma faixa demarcada por duas curvas crescentes, indicando as medidas consideradas normais para uma criança sadia. Há, ainda, um diagrama mais elaborado, semelhante ao primeiro, em que o quadrante positivo do plano é dividido em faixas diagonais, restringindo a categoria das crianças de desenvolvimento físico normal em cada faixa de idade.

Se as curvas limitantes desta região fossem perdidas (ou, talvez, se tivessem sua validade questionada), uma amostra razoavelmente significativa de crianças consideradas normais, em desenvolvimento físico, seria suficiente para reconstruí-las (ou determinar sua validade).

Conceitos como este trazem uma complicação, dada pelo fato de que os atributos em questão têm valores reais, mas quando se trata de representação, gráfica ou computacional, esta natureza está limitada pela precisão dos instrumentos.

Um maior número de atributos acaba por tornar impossível a representação gráfica da região correspondente a um conceito, mas não invalida a idéia: podemos incluir entre os atributos avaliados para determinação do conceito “maçã” o grau de doçura ao paladar, a dureza do material de que é composta, o reino natural a que pertence e uma infinidade de outros: ainda assim, “maçã” é uma hiper-região neste espaço multidimensional.

Os exemplos da Seção 3.2 são todos algoritmos desta natureza, com a particularidade de que os atributos são todos bivalorados. Uma expressão booleana define um conceito na mesma medida em que define uma região de pontos que a verificam, num reticulado completo de ordem n , para um espaço de ocorrências de dimensão n .

Note que expressões booleanas mais complicadas que um monômio podem definir regiões desconexas, o que não invalida o método geral. Basta que a linguagem de representação admita a representação de conjuntos desconexos.

Esta modalidade de algoritmos realiza uma generalização, já que parte dos exemplos e contra-exemplos do conceito e abstrai os atributos relevantes para determinação do conceito e os valores destes atributos que determinam

se um objeto pertence ou não ao conceito.

4.2 Árvores de decisão

A definição de árvores de decisão foi apresentada na Seção 3.5, mas o problema mais difícil foi deixado em aberto: a escolha ou construção da árvore de decisão mais adequada para um determinado objetivo de classificação, com base numa amostra.

Nesta seção examinaremos em detalhe os algoritmos com a seguinte forma geral.

Inicialmente, todos os exemplos da amostra são associados ao nó raiz de uma árvore que só tem este nó. Um teste aplicável aos elementos do espaço de ocorrências é selecionado como critério de partição da amostra e é aplicado a cada exemplo. Cada resultado possível deste teste, juntamente com os exemplos que dão tal resultado no teste, é associado a um nó descendente do nó raiz. Cada nó descendente da raiz sofre o mesmo processo, recursivamente.

A recursão é terminada, obviamente, quando todos os exemplos associados a um nó têm a mesma classificação. O nó é deixado como terminal e recebe esta classificação como rótulo. Os nós correspondentes a resultados possíveis de um teste mas sem nenhum representante na amostra também não podem ser expandidos e são rotulados como nulos. Critérios de parada e de rotulação mais refinados serão examinados mais adiante.

Esta forma geral é chamada de “*Top-Down Induction of Decision Trees*”, ou *TDIDT*, o que poderia ser traduzido por “Indução de Árvores de Decisão de Cima para Baixo” ou “*Indução de Árvores de Decisão da Raiz para as Folhas*”. Outras famílias de algoritmos de construção ou seleção de árvores serão examinadas na Seção 4.6.

4.2.1 Um exemplo elementar

Como exemplo, vamos considerar o espaço que contém os animais vertebrados, descritos pela seguinte lista de atributos:

- a_1 aptidão para mergulho {mergulha, afoga-se}
- a_2 meio em que se movimenta {ar, terra, água}
- a_3 aparelho respiratório {pulmões, brânquias}
- a_4 cobertura da pele ou couro {pelos, penas, escamas, nenhum}

a_5 massa (padrão: 50kg) {menor, maior}
 a_6 produz veneno {sim, não}
 a_7 forma de movimentação {salta, anda, rasteja, nada, voa}

Desejamos determinar a Classe dos vertebrados com base nestes atributos. Uma amostra possível é dada pela Tabela 4.1.

<i>Animal</i>	a_1	a_2	a_3	a_4	a_5	a_6	a_7	<i>Classe</i>
baleia	mergulha	água	pulmões	nenhum	maior	não	nada	<i>mamífero</i>
pingüim	mergulha	terra	pulmões	penas	menor	não	anda	<i>ave</i>
gaivota	mergulha	ar	pulmões	penas	menor	não	voa	<i>ave</i>
jacaré	mergulha	terra	pulmões	nenhum	maior	não	rasteja	<i>réptil</i>
tubarão	mergulha	água	brânquias	nenhum	maior	não	nada	<i>peixe</i>
tainha	mergulha	água	brânquias	escamas	menor	não	nada	<i>peixe</i>
leão	afoga-se	terra	pulmões	pelos	maior	não	anda	<i>mamífero</i>
cavalo	afoga-se	terra	pulmões	pelos	maior	não	anda	<i>mamífero</i>
morcego	afoga-se	ar	pulmões	pelos	menor	não	voa	<i>mamífero</i>
rato	mergulha	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>
canguru	afoga-se	terra	pulmões	pelos	maior	não	salta	<i>mamífero</i>
pardal	afoga-se	ar	pulmões	penas	menor	não	voa	<i>ave</i>
águia	afoga-se	ar	pulmões	penas	menor	não	voa	<i>ave</i>
cobra	mergulha	terra	pulmões	escamas	menor	sim	rasteja	<i>réptil</i>
lagarto	afoga-se	terra	pulmões	escamas	menor	não	rasteja	<i>réptil</i>
sapo	mergulha	terra	pulmões	nenhum	menor	sim	salta	<i>anfíbio</i>

Tabela 4.1: Amostra de treinamento para classificação de vertebrados

Uma maneira elementar de induzir uma árvore de decisão a partir desta amostra é particioná-la segundo os atributos na ordem em que são apresentados, terminando o processo quando cada partição contém apenas elementos da mesma classe. Assim, separando-se os animais conforme sua aptidão para o mergulho, temos uma primeira partição, na Figura 4.1.

Animais que ...

mergulham baleia, pingüim, gaivota, jacaré, tubarão, tainha, rato, cobra, sapo
afogam-se leão, cavalo, morcego, canguru, pardal, águia, lagarto

Figura 4.1: Árvore de classificação de vertebrados: primeiro nó

Na Figura 4.2 a seguir, particionamos cada grupo conforme seu meio.

Neste passo, já podemos observar um problema: não há um animal aquático que se afogue. Pode parecer óbvio, mas não para um algoritmo. A

Quanto à aptidão para o mergulho ...

mergulham e quanto ao meio em que se movem ...

água baleia, tubarão, tainha

terra pingüim, jacaré, rato, cobra, sapo

ar gaivota

afogam-se e quanto ao meio em que se movem ...

terra leão, cavalo, canguru, lagarto

ar morcego, pardal, águia

água (vazio)

Figura 4.2: Vertebrados: dois níveis

alternativa natural é rotular as ocorrências que fizeram seu caminho na árvore até aqui como de classe desconhecida.

Prosseguindo com o aparelho respiratório, encontraremos algumas partes que contêm apenas animais de uma mesma classe. Podemos rotulá-las e parar de particioná-las. Encontraremos também mais alguns resultados desconhecidos. O resultado é apresentando na Figura 4.3.

Quanto à aptidão para o mergulho ...

mergulham e quanto ao meio em que se movem ...

água e quanto ao aparelho respiratório ...

têm **pulmões** baleia » Classe dos Mamíferos

têm **brânquias** tubarão, tainha » Classe dos Peixes

terra pingüim, jacaré, rato, cobra, sapo

ar gaivota » Classe das Aves

afogam-se e quanto ao meio em que se movem ...

terra e quanto ao aparelho respiratório ...

têm **pulmões** leão, cavalo, canguru, lagarto

têm **brânquias** (vazio) » Classe desconhecida

ar e quanto ao aparelho respiratório ...

têm **pulmões** morcego, pardal, águia

têm **brânquias** (vazio) » Classe desconhecida

água (vazio) » Classe desconhecida

Figura 4.3: Vertebrados: mais uma expansão

Enfim, a Figura 4.4 apresenta o resultado final, após prosseguirmos enquanto necessário para determinar a classe correspondente a uma divisão.

Ao fim do processo, torna-se desnecessário manter registro dos exemplos que deram origem às divisões.

Animais que ...

mergulham e quanto ao meio em que se movem ...

água e quanto ao aparelho respiratório ...

têm pulmões *Classe dos Mamíferos*

têm brânquias *Classe dos Peixes*

terra e quanto ao aparelho respiratório ...

têm pulmões e quanto à cobertura do couro ou pele ...

têm penas *Classe das Aves*

têm pelos *Classe dos Mamíferos*

têm escamas *Classe dos Répteis*

sem cobertura e quanto à massa média de um adulto ...

maior que 50kg *Classe dos Répteis*

menor que 50kg *Classe dos Anfíbios*

têm brânquias *Classe desconhecida*

ar *Classe das Aves*

afogam-se e quanto ao meio em que se movem ...

terra e quanto ao aparelho respiratório ...

têm pulmões e quanto à cobertura do couro ou pele ...

têm pelos *Classe dos Mamíferos*

têm penas *Classe desconhecida*

têm escamas *Classe dos Répteis*

sem cobertura *Classe desconhecida*

têm brânquias *Classe desconhecida*

ar e quanto ao aparelho respiratório ...

têm pulmões e quanto ao aparelho respiratório ...

têm pelos *Classe dos Mamíferos*

têm penas *Classe das Aves*

têm escamas *Classe desconhecida*

sem cobertura *Classe desconhecida*

têm brânquias *Classe desconhecida*

água *Classe desconhecida*

Figura 4.4: Árvore final de classificação de vertebrados

Agora, a Tabela 4.2 mostra o resultado da aplicação desta árvore a 16 outras ocorrências de vertebrados, com a mesma distribuição de Classes da amostra de treinamento. A Classe resultante é apresentada na última coluna.

<i>Animal</i>	<i>a₁</i>	<i>a₂</i>	<i>a₂</i>	<i>a₄</i>	<i>a₅</i>	<i>a₆</i>	<i>a₇</i>	<i>Classe</i>	<i>Resultado</i>
urso	afoga-se	terra	pulmões	pelos	maior	não	anda	<i>mamífero</i>	<i>mamífero</i>
gato	afoga-se	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>	<i>mamífero</i>
cão	afoga-se	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>	<i>mamífero</i>
preguiça	afoga-se	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>	<i>mamífero</i>
castor	mergulha	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>	<i>mamífero</i>
golfinho	mergulha	água	pulmões	nenhum	maior	não	nada	<i>mamífero</i>	<i>mamífero</i>
galinha	afoga-se	terra	pulmões	penas	menor	não	anda	<i>ave</i>	desconhecido
pato	mergulha	terra	pulmões	penas	menor	não	anda	<i>ave</i>	<i>ave</i>
ema	afoga-se	terra	pulmões	penas	maior	não	anda	<i>ave</i>	desconhecido
sanhaço	afoga-se	ar	pulmões	penas	menor	não	voa	<i>ave</i>	<i>ave</i>
crocodilo	mergulha	terra	pulmões	nenhum	maior	não	rasteja	<i>réptil</i>	<i>réptil</i>
camaleão	mergulha	terra	pulmões	nenhum	menor	não	rasteja	<i>réptil</i>	desconhecido
lagartixa	mergulha	terra	pulmões	nenhum	menor	não	rasteja	<i>réptil</i>	desconhecido
baiacu	mergulha	água	branquias	nenhum	menor	sim	nada	<i>peixe</i>	<i>peixe</i>
salmão	mergulha	água	brânquias	escamas	menor	não	nada	<i>peixe</i>	<i>peixe</i>
rã	mergulha	terra	pulmões	nenhum	menor	não	salta	<i>anfíbio</i>	<i>anfíbio</i>

Tabela 4.2: Testes sobre uma nova amostra

4.2.2 Seleção de atributos: a pergunta certa

O exemplo anterior é um caso muito grosseiro de indução de árvores de decisão. É possível produzir uma árvore muito melhor usando critérios para escolha do teste a ser realizado em cada nó, de maneira que sejam avaliados os atributos mais relevantes na determinação da classificação de um elemento e seja minimizado o peso dos atributos que têm pouca correlação com a classificação.

Numa primeira abordagem, cada teste possível sobre os elementos incidentes num nó é pontuado segundo a coerência das partes resultantes de sua aplicação, e o melhor pontuado é selecionado. Se um teste divide a amostra corrente em grupos disjuntos quanto à classificação de seus elementos, ele é provavelmente um bom candidato. E, inversamente, se cada classificação tem representantes em todos os grupos resultantes da partição por este teste, ele é provavelmente um péssimo teste.

No exemplo anterior, poderíamos definir dois critérios de pontuação e avaliar os atributos pela soma dos dois:

coesão proporção dos exemplos cuja classe é mantida totalmente coesa
eficácia proporção dos exemplos cuja classe é mantida coesa e determina um nó folha

e, então, teríamos como primeiro critério o aparelho respiratório, conforme a

Tabela 4.3.

<i>Critério</i>	<i>Coesão</i>	<i>Eficácia</i>	<i>Total</i>
a_1	3/16	0/16	3/16
a_2	6/16	0/16	6/16
a_3	16/16	2/16	18/16
a_4	5/16	9/16	14/16
a_5	5/16	0/16	5/16
a_6	13/16	0/16	13/16
a_7	6/16	3/16	9/16

Tabela 4.3: Seleção de atributos por coesão e eficácia

Apenas uma das ramificações teria de ser reparticionada e aplicando-se o mesmo critério, o teste escolhido seria o atributo a_4 . Novamente, apenas uma das ramificações seria reparticionada, pelo atributo a_7 e, então, teríamos todas os nós terminais rotulados. A árvore resultante teria sido a representada na Figura 4.5.

Quanto ao aparelho respiratório ...

brânquias *Classe dos Peixes*

pulmões e quanto à cobertura têm ...

pelos *Classe dos Mamíferos*

penas *Classe das Aves*

escamas *Classe dos Répteis*

nenhuma e quanto à forma de movimentação ...

rasteja *Classe dos Répteis*

salta *Classe dos Anfíbios*

nada *Classe dos Mamíferos*

anda *Classe desconhecida*

voa *Classe desconhecida*

Figura 4.5: Árvore obtida com critérios de seleção de atributos

Observe que esta árvore é degenerada e, no máximo, uma subárvore de cada nó interno é também um nó interno. Isto ocorre pela distribuição dos valores dos atributos selecionados entre os exemplos da amostra. Estes atributos foram privilegiados por terem alguns valores muito característicos de apenas uma classe. Todos os peixes e apenas os peixes têm brânquias (na verdade, anfíbios também, mas deixam de usá-las após as primeiras fases de seu desenvolvimento). Pelos e penas também são características exclusivas de mamíferos e aves, respectivamente, além de serem muito poucos os casos em que não ocorrem.

É importante notar, ainda, que se não houvesse esta degenerescência e dois nós descendentes do mesmo ancestral tivessem de ser expandidos, cada um poderia ser expandido por um critério diferente do outro, já que o critério de seleção dos atributos é aplicado aos exemplos presentes em cada nó.

Novamente, a Tabela 4.4 mostra o resultado da aplicação desta árvore às mesmas 16 novas ocorrências de vertebrados, com a Classe resultante na última coluna.

<i>Animal</i>	<i>a₁</i>	<i>a₂</i>	<i>a₄</i>	<i>a₄</i>	<i>a₅</i>	<i>a₆</i>	<i>a₇</i>	<i>Classe</i>	<i>Resultado</i>
urso	afoga-se	terra	pulmões	pelos	maior	não	anda	<i>mamífero</i>	<i>mamífero</i>
gato	afoga-se	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>	<i>mamífero</i>
cão	afoga-se	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>	<i>mamífero</i>
preguiça	afoga-se	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>	<i>mamífero</i>
castor	mergulha	terra	pulmões	pelos	menor	não	anda	<i>mamífero</i>	<i>mamífero</i>
golfinho	mergulha	água	pulmões	nenhum	maior	não	nada	<i>mamífero</i>	<i>mamífero</i>
galinha	afoga-se	terra	pulmões	penas	menor	não	anda	<i>ave</i>	<i>ave</i>
pato	mergulha	terra	pulmões	penas	menor	não	anda	<i>ave</i>	<i>ave</i>
ema	afoga-se	terra	pulmões	penas	maior	não	anda	<i>ave</i>	<i>ave</i>
sanhaço	afoga-se	ar	pulmões	penas	menor	não	voa	<i>ave</i>	<i>ave</i>
crocodilo	mergulha	terra	pulmões	nenhum	maior	não	rasteja	<i>réptil</i>	<i>réptil</i>
camaleão	mergulha	terra	pulmões	nenhum	menor	não	rasteja	<i>réptil</i>	<i>réptil</i>
lagartixa	mergulha	terra	pulmões	nenhum	menor	não	rasteja	<i>réptil</i>	<i>réptil</i>
baiacu	mergulha	água	branquias	nenhum	menor	sim	nada	<i>peixe</i>	<i>peixe</i>
salmão	mergulha	água	brânquias	escamas	menor	não	nada	<i>peixe</i>	<i>peixe</i>
rã	mergulha	terra	pulmões	nenhum	menor	não	salta	<i>anfíbio</i>	<i>anfíbio</i>

Tabela 4.4: Resultados com a nova árvore

Com esta árvore não houve erros de classificação. Também, há muito menos nós terminais sem rótulo. Os critérios podem ser ainda mais refinados do que isto, levando em conta medidas estatísticas, como a correlação entre os valores de um mesmo atributo, entre os valores de atributos distintos e entre os valores dos atributos e a classificação desejada. Há outros, ainda, que se valem da teoria da informação, utilizando conceitos como entropia da informação e o princípio da descrição de menor comprimento.

Atributos contínuos e multivalorados

Um viés de alguns critérios de seleção de atributos é o privilégio concedido a atributos com maior variedade de valores. No exemplo desta Seção, vemos que isto ocorre com o critério de eficácia. No caso de atributos discretos, a própria natureza do atributo e de seus valores pode conter, de fato, informação com alto grau de correlação com a classificação das ocorrências. No

mesmo exemplo, observamos que apenas aves têm penas e apenas mamíferos têm pelos.

Por outro lado, atributos contínuos, cujo domínio é numérico, não podem ser tratados da mesma maneira. Se cada valor presente na amostra der origem a uma subárvore, possivelmente, cada uma delas terá apenas uma ocorrência e o poder preditivo da árvore resultante é praticamente nulo. Nenhum valor que não ocorre na amostra será classificado. Assim, estes atributos devem ser testados com base na pertinência do valor a um intervalo. O quinto atributo no exemplo dos vertebrados, a massa, por exemplo, foi testada com base num valor padrão predeterminado, que divide o domínio em dois intervalos. Entretanto o valor limiar foi escolhido sem qualquer critério com base teórica que o justifique.

4.2.3 Poda e pré-poda: evitando excessos

Outra questão a ser analisada na indução da árvore é a representatividade dos exemplos de uma parte da amostra associada a um nó: conforme as sucessivas partições vão sendo construídas, o número de casos incidentes num nó vai diminuindo e pode deixar de ser um tamanho razoável para uma amostra confiável. Ainda mais importante é o viés introduzido na distribuição subjacente ao espaço de ocorrências, já que os exemplos que seguiram pelo caminho em questão, na árvore, podem não ter a mesma distribuição [Qui87, Lau96].

Por vezes, esta avaliação é feita após o excesso cometido, o que não tem implicações graves, dado que o processo de construção da árvore de decisão só se dá por terminado após esta avaliação e as medidas cabíveis. Esta fase da construção de uma árvore de decisão é chamada de *poda*. Em geral, compara-se a probabilidade de erro da árvore em seu estado atual com a probabilidade de erro da árvore podada e com o nó terminal resultante, antes interno, associado a uma classificação entre as possíveis. Se a situação atual for mais precisa, ela é mantida. Caso contrário, aceita-se a poda e repete-se o procedimento. Outra idéia é chamada de *pré-poda* ou *poda prévia* e consiste na instituição de um critério que, uma vez atingido, impede a partição dos exemplos incidentes num nó, mesmo que haja classificações diferentes.

Nos dois casos, surge o problema: *como rotular este nó, que contém exemplos com rótulos diversos?* Esta questão sugere um tipo de critério de parada: se for possível garantir a precisão desejada rotulando o nó corrente de alguma maneira, então isto é feito; caso contrário, a expansão prossegue. Outro tipo de critério é a simples aplicação adiantada do critério da poda posterior: por um lado, atribui-se o melhor rótulo possível a um nó; por outro, ele é expandido. Os dois resultados são comparados quanto à precisão e o melhor

é escolhido. Se foi a expansão, o processo continua.

O conceito de poda trata naturalmente do problema da presença de ruídos. Uma árvore que dê conta destas anomalias é, em geral, uma árvore complexa e detalhada. Tanto a poda posterior quanto a pré-poda fazem com que a árvore resultante admita alguma “impureza” na classificação dos exemplos incidentes num nó terminal, desde que não seja muito significativa. Podemos afirmar que a presença de ruído, para ainda permitir o processo de aprendizagem, deve ser menos significativa que a informação efetivamente contida nos exemplos. Assim, em algum momento, os efeitos do ruído serão absorvidos pela poda, posterior ou prévia.

4.3 Um exemplo de sucesso: o C4.5

Para ilustrar os conceitos discutidos até aqui, sobre árvores de decisão, examinaremos alguns detalhes de um algoritmo de indução bastante conhecido. Quinlan [Qui96] apresenta e compara duas versões subsequentes (*Releases 7 e 8*) de seu algoritmo C4.5 para indução de árvores de decisão, em que os valores dos atributos contínuos são particionados em apenas dois intervalos. A idéia geral para determinação do limiar é ordenar os exemplos segundo o valor do atributo contínuo em questão. Então, cada par de valores adjacentes e distintos sugere um candidato a limiar: o valor médio entre eles.

A seleção do teste num nó interno se dá pela comparação, sob um critério de partição, de todos os possíveis testes, considerados todos os atributos discretos e todos os possíveis limiares de todos os atributos contínuos.

O critério de partição usado na primeira versão apresentada é chamado de *gain ratio* ou *razão de ganho*, que é definido como a seguir. Sejam s a amostra a ser particionada, g o número de diferentes classificações dos exemplos e $p(s, j)$ a proporção de ocorrências em s que pertencem à j -ésima classificação. A incerteza residual sobre a classe a que pertence uma ocorrência qualquer de s é expressa como

$$Info(s) = - \sum_{j=1}^g p(s, j) \log_2 p(s, j)$$

e o ganho de informação obtido por um teste t com k possíveis resultados é dado pela expressão

$$Gain(s, t) = Info(s) - \sum_{i=1}^k \frac{|s_i|}{|s|} Info(s_i),$$

na qual s_i é a parte da amostra s cujas ocorrências dão o i -ésimo resultado no teste t . O potencial de informação obtida por uma partição é baseado no conhecimento de qual será a parte correspondente a uma ocorrência após a partição:

$$Split(s, t) = - \sum_{i=1}^k \frac{|s_i|}{|s|} \log_2 \frac{|s_i|}{|s|}.$$

O critério da *razão de ganho* seleciona, entre os teste cujo *ganho de informação* é maior ou igual que a média deste valor, aquele com a melhor *razão de ganho*:

$$GainRatio(s, t) = Gain(s, t) / Split(s, t).$$

Esta estratégia de partição recursiva resulta em árvores consistentes com a amostra s , se possível. Mas em aplicações práticas, os dados são, muitas vezes, imprecisos ou ruidosos, e isto leva à produção de árvores que tentam modelar esta imprecisão. A maioria dos sistemas de indução podam, conforme a Seção 4.2.3, a árvore obtida identificando as subárvores com baixo poder preditivo e substituindo-as por nós terminais.

Atributos contínuos com muitos valores distintos podem ser privilegiados, no critério acima, em relação aos atributos discretos e, mesmo, em relação aos outros contínuos com menos valores distintos. A versão seguinte do C4.5 de Quinlan propõe uma correção deste viés, que é baseada no princípio da *Descrição de Menor Comprimento* ou MDL (*Minimum Description Length*), brevemente apresentado a seguir [QR89].

Considere um sujeito transmissor e outro receptor, ambos possuindo uma lista ordenada de ocorrências. O transmissor conhece, ainda, a classe a que pertence cada ocorrência. O transmissor codifica e envia ao receptor uma teoria de como classificar as ocorrências. Como esta teoria pode ser imperfeita, ele deve também identificar as exceções e indicar como a predição da teoria sobre suas respectivas classificações deve ser corrigida. O comprimento total da mensagem é, então, a soma do número de bits necessários para codificar a teoria (custo da teoria) com o número de bits necessários para identificar e corrigir as exceções (custo das exceções). O transmissor pode ter uma ampla gama de escolhas, desde uma teoria simples que deixa muitos erros a serem tratados como exceção, até outras extremamente complexas e com poucas exceções. O princípio MDL determina a escolha da teoria que minimiza o comprimento total da mensagem, isto é, a soma dos custos da teoria e das exceções.

Assim, este princípio induz um balanço entre a complexidade da teoria e a precisão sobre a amostra de treinamento. O custo das exceções associados a

uma amostra s é equivalente a $|s|Info(s)$, de maneira que $|s|Gain(s, t)$ mede a redução no custo de exceções quando s é particionado pelo teste t . Por outro lado, a partição de s desta forma requer a transmissão de uma teoria mais complexa que envolva a definição do teste t . Ao passo que o teste sobre um atributo discreto pode ser especificado pela simples nomeação do atributo, um teste sobre um atributo contínuo envolve a transmissão do limiar; se h é o número de valores distintos do atributo considerado, na amostra, então pode haver $h - 1$ possíveis limiares, e serão necessários $\log_2(h - 1)$ bits adicionais.

Com isto, a segunda versão apresentada (*Release 8*) recebe as seguintes modificações. Primeiro, o ganho de informação de um teste sobre um atributo contínuo é penalizado na medida do acréscimo de custo determinado pela transmissão do limiar:

$$Gain(s, t) = Info(s) - \sum_{i=1}^k \frac{|s_i|}{|s|} Info(s_i) - \log_2 \frac{h - 1}{|s|}.$$

Limiares com ganho negativo, ou seja, cujo ganho original não supera a penalização são descartados. Consequentemente, são totalmente desconsiderados os atributos contínuos cujos possíveis limiares são todos descartados. Além disto, como o potencial de informação de uma partição é maximizado quando o limiar divide a amostra na metade, a intensidade do efeito da penalidade acima sobre a razão de ganho varia com o limiar, sendo menor exatamente neste caso. A segunda modificação é determinar o limiar de cada atributo contínuo apenas através do máximo ganho de informação e, então, voltar a usar o critério da razão de ganho para escolher o melhor teste considerando, além dos atributos discretos, apenas os limiares já determinados para os atributos contínuos.

Claramente, é possível que, mesmo a amostra contendo ocorrências de várias classificações, não apresente nenhum teste possível, segundo os critérios acima e isto determinada outro critério de parada na recursão. A rotulação do nó deve seguir algum critério que minimize a probabilidade de erro, conforme a Seção 4.2.3.

Uma consequência da partição de atributos contínuos em apenas dois intervalos é a possível necessidade de reavaliar o mesmo atributo, em outro nó mais externo. Quinlan experimenta, ainda no mesmo artigo, um esquema de discretização em que antes de iniciar a expansão da árvore, os atributos contínuos são todos discretizados. A idéia fundamental desta discretização é, para cada atributo contínuo, encontrar o melhor limiar, particionar a amostra segundo este limiar e reparticionar cada parte, recursivamente até que um critério de parada seja satisfeito. Os mesmos critérios usados nos *Releases 7* e *8* do C4.5 são utilizados: o melhor limiar para cada uma das partições recursivas dos valores de um atributo contínuo é escolhido de modo a maximizar

o ganho ajustado de informação. A recursão para quando nenhum limiar tem ganho ajustado positivo. Os resultados comparativos apresentados não são favoráveis a esta discretização prévia, em termos de precisão, embora as árvores assim obtidas tenham sido menores.

O autor menciona ainda outras idéias, com melhorias de desempenho em algumas aplicações.

- o esquema original é aplicado até que a amostra incidentes num nó seja pequena, em relação à amostra original. A partir deste ponto, a discretização prévia é aplicada;
- atributos discretos tem seus valores agrupados e então biparticionados segundo os mesmo critérios aplicados aos contínuos;
- a discretização prévia é realizada através de uma busca no espaço de partições possíveis.

4.4 Algoritmo: REAL

Esta seção descreve os critérios de discretização e seleção de atributos e de parada na expansão da árvore utilizados pelo algoritmo REAL (*Real-valued Attribute classification tree Learning algorithm*) apresentado em [Lau96] e [SNLR98b, SNLR98a].

Seja j a classe majoritária entre os $|s|$ exemplos associados a um nó, dos quais k não pertencem à classe j . Se q , com distribuição Bayesiana $D(c) = \Pr\{q \leq c\}$, é a probabilidade de classificação incorreta de uma ocorrência neste nó, então define-se a medida de convicção $100(1 - cm)\%$, na qual

$$cm = \min_c |\Pr\{q \leq c\} - 1| = g(c),$$

e $g(c)$ é uma função convexa de $[0, 1]$ em $[0, 1]$. A convexidade de g dá a medida da cautela do aprendiz “ao fazer afirmações fortes”. Uma boa escolha é $g(c) = c^r$, onde $r \geq 1$ é um parâmetro de convexidade (ou de cautela). A expressão da distribuição D implementada é a função beta incompleta derivada da distribuição de Bernoulli:

$$\begin{aligned} B(s, k, q) &= \binom{|s|}{k} q^k (1 - q)^{|s| - k} \\ D(c, s, k) &= \frac{\int_{q=0}^c B(s, k, q)}{\int_{q=0}^1 B(s, k, q)} \\ &= \text{betainc}(c, k + 1, |s| - k + 1) \end{aligned}$$

Então, cm é dada por

$$\begin{aligned} cm(s, k) &= c|f(c) = 0 \\ f(c, s, k) &= 1 - g(c) - D(c, s, k) \\ &= 1 - c^r - \text{betainc}(c, k + 1, |s| - k + 1) \end{aligned}$$

O REAL define a função de perda associada à partição de uma amostra através do atributo v discretizado em w intervalos $I_1, I_2, \dots, I_w$ como

$$perda = \sum_{i=1}^w |s_i| cm(s_i, k_i),$$

onde $|s_i|$ é o número de exemplos no i -ésimo intervalo definido pela w -partição dos valores do atributo e k_i é o número de exemplos erroneamente classificados neste mesmo intervalo.

Para a categorização de um atributo contínuo, os exemplos presentes num nó v são ordenados segundo o valor do atributo e previamente agrupados, conforme os adjacentes tenham a mesma classificação. A seguir, os intervalos assim induzidos são reagrupados de modo a minimizar a perda global determinada pela expansão do nó através deste atributo.

O ganho determinado pelo reagrupamento dos intervalos $I_{h+1}, I_{h+2}, \dots, I_{h+j}$, em relação ao agrupamento inicial é dado por

$$ganho(h, j) = \sum_{i=h+1}^j perda(v, W_i) - perda(v, W).$$

O REAL calcula, iterativamente, o ganho de todos os possíveis reagrupamentos de $3 \leq j \leq J_{max}$ agrupamentos e efetiva aquele que dá o maior ganho. A iteração é terminada quando não há mais reagrupamentos possíveis com ganho positivo.

Após determinar a melhor discretização para cada atributo, o algoritmo seleciona, para expandir o nó, o atributo que dá o menor valor da função de perda, e realiza a expansão segundo esta discretização. Após esta expansão, pode haver nós adjacentes, com classificações majoritárias dos respectivos exemplos diferentes, que seriam melhor expandidos se fossem reagrupados do que se fossem expandidos isoladamente. Este passo adicional é realizado caso os intervalos correspondentes aos nós, na discretização selecionada, não satisfaçam $cm \leq crv$ tal que $crv \in [0, 1]$ é um parâmetro de convicção.

Um nó é definido como terminal quando não houver atributo cuja melhor discretização diminua a função de perda por, no mínimo, um fator $\epsilon > 0$. Se isto ocorrer com um nó resultante de um reagrupamento após a expansão,

conforme o parâmetro *crv*, o reagrupamento é desfeito. Em qualquer caso, o nó terminal é rotulado pela classe majoritária dos exemplos no nó.

Como os critérios envolvidos na expansão da árvore já envolvem uma comparação da árvore resultante desta expansão com a árvore atual, não há necessidade de uma poda posterior. A possível presença de ruído ou de casos anômalos será absorvida pelo reagrupamento de intervalos e de nós resultantes.

4.5 Outras considerações sobre TDIDT

Em ambos os algoritmos analisados acima, a seleção do atributo a ser testado em cada nó interno considera todos os atributos disponíveis. Isto é justificado pelo fato de que, no primeiro caso, os valores são biparticionados e podem ainda conter relação com as diversas classificações presentes em cada grupo; no segundo caso, o agrupamento de intervalos também acarreta a mistura de diferentes classificações num nó, e uma nova divisão usando o mesmo atributo pode ser mais vantajosa que outros testes.

No caso de árvores booleanas, apresentado na Seção 3.5, um atributo não precisa ser reavaliado pois, após a primeira avaliação, todos os exemplos incidentes em cada subárvore terão o mesmo resultado, numa reavaliação. Atributos categóricos, mesmo que não binários, se resultarem em tantas subárvores quantos valores possíveis tiver, também não precisam ser reavaliados, pelo mesmo motivo.

Outra característica relevante destes algoritmos é a avaliação de cada atributo isoladamente. Um problema resultante desta característica é o possível descarte de relações delicadas entre os atributos. Dois atributos que, isolados, não informam coisa alguma sobre a classificação de uma ocorrência podem, juntos, ser determinantes. Este argumento é citado contra a pré-poda e, mesmo, contra a poda posterior simples. Uma expansão mais profunda da árvore, sem poda, pode acabar incluindo testes sobre os dois atributos correlacionados, e um esquema de poda mais sofisticado, em que um nó interno é substituído não por um nó terminal, mas por uma de suas subárvores, pode trazer esta correlação à tona [Mar97].

Estas características estão interrelacionadas e, em muitos casos, a presença de uma exige a outra. No caso de espaços de ocorrências euclidianos, por exemplo, a imposição de um único atributo testado em cada nó implica uma partição ortogonal do espaço de ocorrências. O semiplano euclidiano $x + y \leq 8$, por exemplo, terá de ser aproximado por uma série de cortes ortogonais através de testes diferentes sobre o mesmo atributo, conforme a Figura 4.6.

padrão unitário. Assim, a atribuição do valor médio zero ao atributo, nos exemplos em que seu valor é desconhecido, não afeta o valor das combinações lineares. As informações de normalização são gravadas em cada nó, de forma que as ocorrências futuras, nas fase de testes e de aplicação, possam ter seus valores normalizados segundo os mesmos critérios.

4.6 Outras famílias de algoritmos para árvores

O método TDIDT de indução de árvores de decisão é particularmente ágil porque uma única árvore é gerada, sem a verificação de alternativas. Uma vez que um nó é determinado, interno com seu teste ou terminal com seu rótulo, todas as árvores que diferem da árvore corrente naquele nó são imediatamente descartadas. Além disto, é um método não incremental, no sentido de que toda a informação disponível para a aprendizagem é fornecida de uma só vez.

Utgoff, Berkman e Clouse [UBC97] apresentam um método eficiente de reestruturação de uma árvore de decisão e o usam em dois métodos diferentes de indução de árvores de decisão.

O primeiro é a indução incremental de árvores de decisão (ITI, do inglês *Incremental Tree Induction*). Sua grande vantagem sobre o processamento em lote (não-incremental) é que muitas aplicações práticas recebem novas informações durante o uso e o custo de reconstruir a árvore do início, acumulando todas as informações existentes será, em geral, bem maior do que o custo de atualização da árvore já existente conforme os novos exemplos disponíveis (possivelmente, o custo total das atualizações será maior que o da construção de uma árvore, mas certamente será menor que o custo total de todas as reconstruções). Por outro lado, toda a informação uma vez utilizada na construção ou atualização da árvore deve ser guardada.

Toda vez que um novo exemplo é fornecido, ele é classificado segundo a árvore corrente e os nós pelos quais passa são marcados. Então, os testes selecionados para cada nó interno são revisados segundo os mesmos critérios de construção da árvore. Se houver necessidade de alterar o teste, então os exemplos de todas as folhas abaixo do nó são utilizados para a revisão de todas as subárvores do nó. Caso contrário, apenas a subárvore pela qual o novo exemplo passou é submetida ao processo.

Este método admite algumas variações como considerar todos os novos exemplos ou apenas aqueles com que a árvore corrente está em discordância; ou, ainda, executar os passos de classificação de novos exemplos e marcação dos nós, adiando a revisão até um momento determinado, que pode ser aquele em que a árvore revisada será finalmente utilizada na classificação de uma ocorrência não classificada.

O segundo método proposto é, na verdade, uma variante da família TDIDT. A grande diferença é que naquela família, o melhor teste para um nó interno é escolhido através de uma métrica externa à árvore, enquanto neste método as diferentes possíveis expansões de um nó são efetivamente realizadas e as árvores resultantes são avaliadas diretamente. Por isto, o método é chamado de indução de árvores por métrica direta (DTMI, do inglês *Direct Metric Tree Induction*). Para diminuir o custo computacional de tal esquema, a mesma métrica indireta dos algoritmos tradicionais da TDIDT pode ser utilizada para a escolha do melhor teste sobre cada atributo e estes são os candidatos à seleção por métrica direta.

Diversas métricas diretas são sugeridas, podendo também ser combinadas:

- expectativa do número de testes na classificação de uma nova ocorrência;
- número de folhas da árvore;
- menor codificação (princípio da descrição de menor comprimento);
- expectativa do custo do erro, ponderada pela distribuição no espaço de ocorrências e, possivelmente, pelo custo de cada tipo de erro;
- expectativa do custo de avaliação, levando em conta que alguns testes podem ser mais dispendiosos que outros.

4.7 Outras implementações

Olhando para os modelos anteriormente descritos do ponto de vista da lógica, observa-se que tanto as árvores de decisão quanto as implementações sugeridas no capítulo 3 descrevem o conhecimento adquirido em uma linguagem proposicional. Por outro lado, diversas áreas de aplicação possíveis para a aprendizagem computacional exigem uma representação relacional do conhecimento, o que não é possível, ou é muito mais complicado, numa lógica proposicional. Alguns exemplos são o raciocínio temporal e/ou espacial, o planejamento e o agendamento de tarefas, o raciocínio qualitativo, as linguagens naturais, análise de imagens e outros sistemas envolvendo complexas relações estruturais como o mapeamento de DNA, a síntese de proteínas e o tratamento de sinais – sons, vozes, ressonâncias etc.

Outra limitação das implementações examinadas até agora é a impossibilidade de informar ao algoritmo aprendiz algum conhecimento prévio sobre o domínio do problema de aprendizagem. O máximo que se pode fazer é determinar o espaço de hipóteses cuja representação seja mais adequada do que outras, e isto é um conhecimento sobre o espaço de conceitos em que se encontra o conceito a ser aprendido, mas não sobre o conceito em particular.

Nesta seção, examinaremos dois outros paradigmas de implementação, frequentemente associados aos modelos formais vistos no Capítulo 3, que têm demonstrado bons resultados nas áreas em que a lógica proposicional não é adequada. Particularmente, o primeiro deles, a seguir, admite de maneira muito transparente o conhecimento prévio sobre o problema como entrada para o aprendiz.

4.7.1 Programação em lógica indutiva

A *Programação em Lógica* (PL) é um formalismo fundamentado na lógica matemática e, com grande simplicidade, pode simular o raciocínio dedutivo. Seu uso corrente se dá em ambientes como o *PROLOG*, uma linguagem de programação em que os programas tem a forma de um conjunto de fórmulas do cálculo de predicados de primeira ordem e, particularmente, restritas à forma das cláusulas de Horn. Um conjunto de regras de inferência é utilizado para efetuar a computação de um programa; particularmente, em *PROLOG*, este conjunto tem uma única regra – a regra da resolução [Apt87, Mug91].

A *Programação em Lógica Indutiva* (ILP) é a inteseccção entre a PL convencional e a aprendizagem computacional, particularmente no contexto da aprendizagem supervisionada. Seus objetivos, herdados da aprendizagem computacional, são o desenvolvimento de técnicas para a indução de hipóteses a partir de observações (exemplos) e conhecimento prévio e a síntese de novo conhecimento a partir da experiência [MR94]. O uso da lógica computacional permite a transposição de duas limitações, críticas em algumas áreas de aplicação, das técnicas clássicas da aprendizagem computacional: o uso de um formalismo lógico restrito, essencialmente proposicional, e a dificuldade no uso de conhecimento prévio.

Uma formalização possível é a seguinte. O algoritmo aprendiz recebe, como entrada:

- $\mathcal{P}$ um conjunto de possíveis cláusulas;
- E^+ um conjunto de exemplos positivos;
- E^- um conjunto de exemplos negativos;
- B um programa consistente.

O conjunto $\mathcal{P}$ pode ser dado declarativamente, cláusula a cláusula, ou na forma de elementos e regras para sua formação, como literais ou, ainda mais elementar, predicados e variáveis que podem ser utilizados. Ele representa o espaço de hipóteses. Os exemplos, positivos e negativos, são fatos básicos, isto é, literais que não contêm variáveis, apenas predicados t -ários e t constantes como parâmetros. O programa B é a forma de apresentar ao

aprendiz algum conhecimento prévio sobre o problema [BG96, Mug91]. Estes elementos devem satisfazer duas condições:

Satisfatibilidade prévia: $B \wedge E^- \not\models \square$

Necessidade prévia: $B \not\models E^+$

A partir desta entrada, o aprendiz deve encontrar um programa $P \in \mathcal{P}$ tal que $B \wedge P$ é um programa lógico consistente, coerente e que deduza os exemplos positivos e contradiga os negativos:

Suficiência posterior: $B \wedge P \models E^+$

Satisfatibilidade posterior: $B \wedge P \wedge E^- \not\models \square$

Como exemplo, considere-se a aprendizagem de relações familiares próximas. O aprendiz saberá, de antemão, que o avô de um elemento é o pai de um dos genitores do elemento. Então ¹,

$\mathcal{P}$ todas as cláusulas de Horn utilizando as variáveis X, Y e Z , que definem o predicado binário *parent* utilizando os predicados binários *father*, *mother*, *parent* e *grandfather*

$B \left\{ \begin{array}{l} grandfather(X, Y) \leftarrow father(X, Z), parent(Z, Y). \\ father(Antonio, Maria) \leftarrow . \\ mother(Maria, Ricardo) \leftarrow . \\ mother(Maria, Marina) \leftarrow . \\ father(Reinaldo, Ivan) \leftarrow . \\ father(Ivan, Daniel) \leftarrow . \\ father \end{array} \right.$

$E^+ \left\{ \begin{array}{l} grandfather(Antonio, Ricardo) \leftarrow . \\ grandfather(Antonio, Marina) \leftarrow . \\ grandfather(Reinaldo, Daniel) \leftarrow . \end{array} \right.$

$E^- \left\{ \begin{array}{l} \leftarrow grandfather(Ricardo, Antonio). \\ \leftarrow grandfather(Marina, Ricardo). \\ \leftarrow grandfather(Daniel, Ivan). \end{array} \right.$

Parent na cláusula que define *grandfather* em B não está definido, o que caracteriza a necessidade prévia. O algoritmo aprendiz pode produzir a seguinte hipótese, que seria perfeita:

$$P = \left\{ \begin{array}{l} parent(X, Y) \leftarrow mother(X, Y). \\ parent(X, Y) \leftarrow father(X, Y). \end{array} \right.$$

¹ Os predicados são representados pelo nome em inglês para evitar a acentuação.

A situação descrita e exemplificada acima caracteriza a semântica *normal* para a ILP. Outra situação possível é a que caracteriza a semântica *não-monotônica* para a ILP. Nesta, os exemplos negativos são impliticamente obtidos através de um tipo de *Suposição de Mundo Fechado* em que tudo o que não foi afirmado é falso [RD94], e a preocupação maior é com as propriedades dos exemplos, em contraposição às novas relações que podem ser obtidas no contexto da semântica normal, com objetivos de classificação e predição. Formalmente, os exemplos positivos são considerados um modelo para alguma teoria representável em $\mathcal{P}$ e o objetivo do aprendizado é encontrar uma hipótese maximal (isto é, um subconjunto maximal de $\mathcal{P}$) que seja verdadeira no modelo. O exemplo a seguir ilustra este contexto.

$\mathcal{P}$ todas as cláusulas possíveis utilizando a variável X e os predicados unários *human*, *male* e *female*.

$$\mathbf{E}^+ \begin{cases} \text{male}(\text{Daniel}) \leftarrow . \\ \text{female}(\text{Marina}) \leftarrow . \\ \text{human}(\text{Daniel}) \leftarrow . \\ \text{human}(\text{Marina}) \leftarrow . \end{cases}$$

Uma solução possível seria

$$\mathbf{P} = \begin{cases} \leftarrow \text{female}(X), \text{male}(X). \\ \text{human}(X) \leftarrow \text{female}(X). \\ \text{human}(X) \leftarrow \text{male}(X). \\ \text{female}(X), \text{male}(X) \leftarrow \text{human}(X). \end{cases}$$

Muitos resultados teóricos em ILP obtidos inicialmente foram negativos, em termos de complexidade computacional. Isto se deu, em geral, devido ao tamanho do espaço de hipóteses $\mathcal{P}$. Após o amadurecimento das pesquisas na área, surgiram algoritmos com heurísticas de busca apropriadas para favorecer o desempenho e outras idéias como a restrição do espaço de hipóteses.

De Raedt e Džeroski [RD94], por exemplo, apresentam um resultado positivo de PAC-apreensibilidade no contexto não-monotônico para o espaço das teorias clausais com cláusulas de até k literais, cada um de tamanho até j , ou jk -CT. O algoritmo apresentado enumera todas cláusulas possíveis e, para cada exemplo, elimina aquelas que são falsas no modelo representado pelo exemplo. Como o algoritmo é consistente (no sentido da apreensibilidade) e o espaço jk -CT é finito, este espaço é PAC-apreensível e aquele algoritmo é um PAC-aprendiz eficiente para o espaço.

4.7.2 Redes neurais

As redes neurais são estruturas de processamento inspiradas no funcionamento das células nervosas animais, particularmente na organização delas no cérebro humano. Esta Seção ilustra o conceito através do seu modelo mais simples [AB92, dSB98], embora diversas variantes tenham sido criadas.

Em uma primeira aproximação, um neurônio é modelado como uma *máquina de limiar linear* h_ω . Considere a representação dada pela sobrejeção $\mathbb{R}^{n+1} \mapsto \mathcal{H}$ tal que um estado $\omega = (\alpha_1, \dots, \alpha_n, \theta)$ defina uma *função de ativação* $h_\omega : \mathbb{R}^n \mapsto \{0, 1\}$ dada por

$$h_\omega(\vec{x}) = \begin{cases} 1, & \text{se } \sum_{i=1}^n \alpha_i x_i \geq \theta; \\ 0, & \text{caso contrário.} \end{cases}$$

O nome limiar linear vem do fato de que esta função divide linearmente os pontos de $\mathbb{R}^n$ em duas regiões, uma de cada lado do hiperplano $\sum_{i=1}^n \alpha_i x_i \geq \theta$. Este “neurônio” pode representar, por exemplo, as funções lógicas *e*, *ou* e *não*, através dos seguintes estados em $\mathbb{R}^3$ e $\mathbb{R}^2$:

e $\omega_e = (1, 1, 2)$

ou $\omega_{ou} = (1, 1, 1)$

não $\omega_{n\tilde{a}o} = (-1, 0)$

Uma rede neural artificial é uma estrutura composta por neurônios interconectados. Cada neurônio pode ter suas entradas sujeitas a condições exteriores à rede – os nós de entrada – ou alimentadas pelas saídas de outros neurônios. A saída de um neurônio por sua vez, está conectada ao exterior da rede – nós de saída – ou alimenta a entrada de outros neurônios. Os neurônios que não se comunicam com o exterior são chamados nós internos. A rede é descrita pelo conjunto Ω dos estados ω_i de cada um de seus nós.

Um modelo simples de rede é aquele em que as conexões entre os nós não formam ciclos, um neurônio é exclusivamente interno, de saída ou de entrada, e há apenas um nó de saída. Outros modelos mais complexos admitem ciclos nas conexões e saída da rede composta pelas saídas de vários neurônios.

Uma outra característica sujeita a muitas variantes é a função de ativação h_ω . Ela pode ser multivalorada ou mesmo real, ao invés de binária. Sua própria definição não precisa ser exatamente uma combinação linear das entradas, podendo ser qualquer outra função delas, sendo mais comum ser uma função do valor da combinação linear.

Associa-se cada entrada da rede a uma variável no espaço de ocorrências e cada saída a uma variável no espaço de resultados. O espaço de hipóteses é o conjunto de todas as configurações Ω possíveis dos estados ω individuais de cada nó.

Uma vez definida a estrutura de uma rede, a criação exaustiva dos estados Ω para a solução de cada problema não é atraente do ponto de vista da aprendizagem computacional.

O problema da aprendizagem computacional por redes neurais é justamente a modificação dos estados iniciais, aleatórios ou sugeridos, conforme os exemplos de uma amostra de treinamento lhe são apresentados. Cada exemplo é composto de um vetor com os valores das variáveis de entradas e outro com os valores das variáveis de saída. Uma regra de atualização dos estados compara a saída obtida da rede quando submetida a uma entrada com a saída desejada e determina as alterações nos valores dos estados, iterativamente, até que a rede produza saídas consistentes com a amostra de treinamento ou que uma taxa de erro tolerável seja atingida.

1. 在 1949 年以前，中国是一个半殖民地半封建国家，政治、经济、文化各方面都受到外国势力的控制。新中国成立后，中国实现了民族独立，建立了人民民主专政的国家。在党的领导下，中国进行了社会主义改造，建立了社会主义制度。在社会主义建设过程中，中国取得了巨大的成就，综合国力不断增强，人民生活水平不断提高。

2. 在 1949 年以前，中国是一个半殖民地半封建国家，政治、经济、文化各方面都受到外国势力的控制。新中国成立后，中国实现了民族独立，建立了人民民主专政的国家。在党的领导下，中国进行了社会主义改造，建立了社会主义制度。在社会主义建设过程中，中国取得了巨大的成就，综合国力不断增强，人民生活水平不断提高。

3. 在 1949 年以前，中国是一个半殖民地半封建国家，政治、经济、文化各方面都受到外国势力的控制。新中国成立后，中国实现了民族独立，建立了人民民主专政的国家。在党的领导下，中国进行了社会主义改造，建立了社会主义制度。在社会主义建设过程中，中国取得了巨大的成就，综合国力不断增强，人民生活水平不断提高。

4. 在 1949 年以前，中国是一个半殖民地半封建国家，政治、经济、文化各方面都受到外国势力的控制。新中国成立后，中国实现了民族独立，建立了人民民主专政的国家。在党的领导下，中国进行了社会主义改造，建立了社会主义制度。在社会主义建设过程中，中国取得了巨大的成就，综合国力不断增强，人民生活水平不断提高。

5. 在 1949 年以前，中国是一个半殖民地半封建国家，政治、经济、文化各方面都受到外国势力的控制。新中国成立后，中国实现了民族独立，建立了人民民主专政的国家。在党的领导下，中国进行了社会主义改造，建立了社会主义制度。在社会主义建设过程中，中国取得了巨大的成就，综合国力不断增强，人民生活水平不断提高。

Capítulo 5

Estudo de caso

Em geral, a apresentação de algoritmos de aprendizagem computacional é baseada na comparação do desempenho do algoritmo proposto com outros já consolidados, quando submetidos a um mesmo problema. Muitas vezes, vários problemas de teste são utilizados, com resultados diversos conforme a natureza do problema. Esta característica determina a qualidade relativa dos diversos métodos de aprendizagem, tendo grande valor na comparação de variantes de uma mesma linha.

O objetivo deste trabalho, ao contrário, é a análise do desempenho do algoritmo em relação à variação do tamanho da amostra de treinamento – a curva de aprendizagem – como forma de avaliação da qualidade absoluta da aprendizagem.

5.1 Descrição da aplicação

O algoritmo REAL foi originalmente apresentado para apoio a decisões de compra e venda de títulos no mercado de ações [Lau96, SNLR98b, SNLR98a]. Este contexto apresenta uma série de características que dificultam uma primeira análise formal, como a obtenção de mais de uma classificação – a própria decisão de compra de um determinado título, quanto tempo mantê-lo, parâmetros para venda antes do término deste tempo (variações acentuadas no valor do título). Outra questão cuja complexidade está além do escopo deste trabalho é a natureza temporal das informações e sua dependência de fatores externos ao modelo. Com isto, se um modelo formal indica a necessidade de informações que abrangem um longo período de tempo para garantia da curva de aprendizagem, então instabilidades externas, como as de origem política e macroeconômica, podem invalidar a suposição de uma distribuição de probabilidades fixa sobre as ocorrências, na amostra de treinamento.

Os artigos [SNLR98b, SNLR98a] apresentam também uma aplicação do algoritmo ao conjunto de testes de *van Cutsem*, que corresponde ao problema da decisão de cortar ou manter o funcionamento de uma estação de distribuição de energia elétrica. A característica de aprendizagem do problema é a determinação das medições dos aparelhos de controle da estação que indicam a iminência de uma sobrecarga. Desta forma, a estação pode ser desligada nesta situação, preservando-a dos desgastes decorrentes da sobrecarga. Um outro objetivo é minimizar a ocorrência de desligamentos desnecessários da estação, que tem outros custos para os usuários da estação. Optamos por analisar esta aplicação porque ela não apresenta algumas das complicações adicionais do problema do mercado de ações, que tornariam esta primeira análise inviável.

Uma ocorrência, neste contexto, é um vetor com as leituras registradas em 28 aparelhos de medição ligados à estação. Não há informação sobre o significado físico de cada leitura e cada uma delas é simplesmente um número real. Os exemplos são constituídos de uma ocorrência mais a indicação de sobrecarga iminente ou não, equivalentes aos rótulos “2” e “1”, respectivamente, quando a estação está na situação descrita pela ocorrência.

A partir de uma base de dez mil exemplos, foram geradas independentemente 150 amostras de treinamento, 10 de cada tamanho, através de seleção aleatória com distribuição uniforme sobre a base inicial. A proporção das classes nesta base é de 6421 ocorrências da classe 1 (64,2%) e o restante da classe 2. A distribuição uniforme utilizada na geração das bases parciais resultou numa proporção semelhante em todos os casos, conforme a Tabela 5.1, que apresenta a proporção de exemplos da classe “1” em cada uma delas, bem como os seus respectivos tamanhos.

Tam.	Cj. 1	Cj. 2	Cj. 3	Cj. 4	Cj. 5	Cj. 6	Cj. 7	Cj. 8	Cj. 9	Cj. 10
166	60,2%	60,2%	65,7%	65,7%	62,7%	60,8%	66,9%	57,8%	62,1%	63,9%
344	59,3%	59,3%	63,4%	64,2%	64,5%	63,4%	64,8%	63,4%	58,7%	59,3%
534	61,6%	61,6%	60,5%	63,9%	66,3%	61,4%	43,8%	63,7%	57,7%	61,6%
740	63,1%	63,5%	63,5%	62,0%	66,4%	61,4%	64,9%	65,1%	59,1%	62,0%
960	62,5%	63,0%	63,0%	62,9%	65,7%	61,3%	64,1%	65,2%	59,4%	61,6%
1200	62,2%	61,9%	62,9%	62,9%	63,3%	62,3%	63,3%	64,1%	59,4%	61,8%
1456	62,4%	61,4%	62,8%	62,8%	65,5%	62,5%	63,0%	63,9%	59,6%	62,9%
1736	62,9%	62,2%	63,4%	64,2%	64,2%	62,4%	62,5%	64,2%	60,5%	63,0%
2043	63,0%	62,3%	63,7%	64,2%	62,6%	62,6%	62,5%	64,2%	61,2%	63,0%
2380	63,5%	62,3%	62,7%	64,5%	62,0%	62,0%	62,1%	64,5%	62,0%	63,7%
2750	63,3%	62,6%	63,0%	63,7%	62,3%	62,5%	62,5%	62,4%	63,2%	63,6%
3156	63,3%	62,5%	63,2%	63,3%	62,4%	62,5%	63,9%	62,1%	63,3%	63,9%
3601	63,6%	62,7%	63,0%	63,7%	62,9%	62,5%	63,7%	61,9%	63,2%	63,8%
4116	63,0%	62,9%	63,0%	63,9%	63,2%	62,6%	63,4%	62,6%	63,4%	63,8%
4680	62,8%	63,0%	62,7%	63,9%	63,3%	63,1%	63,5%	62,4%	63,8%	63,4%

Tabela 5.1: Proporção de ocorrências da classe “1” nas amostras de treinamento

Cada amostra foi usada para treinar o REAL, dando origem a uma árvore diferente e cada uma delas foi testada contra os dez mil exemplos da base completa e a quantidade de erros cometidos foi registrada. O parâmetro de convexidade do REAL foi fixado em 2.5 e o parâmetro de confiança em 0.1. Este parâmetro não tem o mesmo significado que o parâmetro de confiança do modelo PAC de análise formal. A Figura 5.1 apresenta a curva de aprendizagem resultante: taxa de erro em função do tamanho da amostra de treinamento.

Figura 5.1: Curva de aprendizagem do REAL

A ocorrência de taxas de erro diferentes sobre uma mesma amostra de testes para árvores induzidas a partir de amostras de treinamento diferentes mas de mesmo tamanho mostra que hipóteses diferentes foram obtidas. Particularmente para amostras pequenas, houve grande variação da taxa de erro, o que indica que hipóteses muito diferentes foram geradas, algumas mais próximas do que outras do conceito alvo. Este fato mostra a sensibilidade do algoritmo às particularidades de cada amostra, dificultando o estabelecimento de limites seguros para a taxa de erro.

Por outro lado, as taxas de erro menos esparsas a medida que a amostra é maior não implicam a indução de árvores parecidas, mas permitem supor que a taxa efetiva de erro, sobre ocorrências futuras, está melhor determinada.

Isto ocorre porque amostras maiores tendem a ser mais homogêneas quanto à informação que contêm, quando extraídas sob uma mesma distribuição de probabilidades.

De qualquer maneira, a apresentação de algoritmos indutivos, em geral, é acompanhada de

- embasamento teórico que garante a obtenção da melhor hipótese possível, em relação aos critérios de medida propostos,
- testes com dados padronizados que mostram a viabilidade desta teoria e a comparação com outras teorias já estabelecidas, em relação a uma ou mais aplicações.

Embora este procedimento permita um avanço rápido da qualidade das implementações e seu uso em aplicações práticas, outras questões são deixadas de lado. Se o objetivo da aprendizagem computacional é produzir algoritmos que se adaptem ao contexto em que serão usados e que possam identificar novos conceitos a partir dos exemplos que lhe sejam apresentados, torna-se necessário ter uma medida de como o algoritmo se comportará em relação a todo o espaço dos conceitos que lhe podem ser apresentados. É neste contexto que se destaca o papel primordial dos modelos formais de aprendizagem computacional. No caso estudado, esta necessidade pode ser observada num exemplo tão simples quanto ir de um bairro a outro numa cidade: estações elétricas tem especificações técnicas diferentes, são construídas com tecnologias e materiais variados (estes ainda podem sofrer variações de qualidade) e estão sujeitas a cargas de uso diversas.

Outro aspecto a ser examinado é em que medida a melhor hipótese obtida segundo determinados critérios – postulados por determinada implementação – é suficientemente boa, em relação ao desejável e em relação ao possível? Do ponto de vista da aprendizagem computacional, o possível está ligado tanto à boa escolha do espaço de hipóteses quanto à quantidade de informação que pode estar presente numa amostra qualquer de determinado tamanho.

5.2 Análise formal

Tendo em mente as considerações anteciores, seguem algumas suposições sobre o contexto da aplicação apresentada e a busca de resultados segundo os modelos apresentados no Capítulo 3.

Em primeiro lugar, enfatizamos a vasta maioria de resultados baseados na consistência da hipótese com a amostra de treinamento e, conseqüentemente, a consistência interna da própria amostra. Outra suposição, menos restritiva,

é a de que o espaço de ocorrências corresponde a $\mathbb{R}^{28}$, já que temos como atributos das ocorrências as leituras de 28 medidores ligados à estação elétrica. Esta suposição poderia ser incrementada pelas seguintes considerações:

- as grandezas medidas têm valores finitos, de forma que seria possível supor um espaço limitado;
- o registro das leituras tem precisão limitada e, com isto, o espaço de ocorrências efetivo é discreto, ao invés de contínuo.

Entretanto, em razão da existência e facilidade de aplicação de resultados interessantes para espaços euclidianos, estas possibilidades serão descartadas em favor das suposições mais genéricas. Particularmente, a segunda possibilidade introduz problemas de contagem que, em geral, resultam em expressões contendo coeficientes binomiais e estes, por sua vez, terminam por dar em números muito distantes dos resultados práticos obtidos, embora de comportamento polinomial.

Árvores booleanas

Uma primeira idéia é aplicar os resultados de [EH89], expostos na Seção 3.5. Como o resultado para árvores quaisquer é subexponencial, não o consideraremos. O resultado polinomial depende do posto das árvores admitidas.

Observando as árvores produzidas pelo REAL, vemos que a maior árvore obtida nos diversos testes tem altura 9, e mesmo neste caso, nem todos os nós terminais com esta profundidade. Assim, o posto de todas as árvores obtidas é menor ou igual a 8. As árvores não são necessariamente binárias, mas a transformação de uma árvore q -ária em outra binária, convertendo os intervalos em uma série de limiares binários, é invariante quanto ao posto da árvore. Assim, supomos que o conceito de sobrecarga na estação elétrica pode ser representado por árvores binárias de posto menor ou igual a 8.

Observa-se, no gráfico da Figura 5.2 que este resultado ainda não tem aplicações práticas: são necessárias dezenas ou centenas de bilhões de exemplos para garantir que o espaço de conceitos em questão possa ser PAC-aprendido através de árvores de posto 8, enquanto o resultado prático indica o aprendizado com precisão razoável para amostras da ordem de milhares de exemplos. O parâmetro de confiança está fixado em $\delta = 0.1$.

Convergência Uniforme

De caráter mais prático é a suposição de que o conceito alvo pode ser caracterizado por uma região ou pela união/intersecção de um número pequeno

Figura 5.2: Curva de aprendizagem por árvores de posto 8

de regiões contínuas do espaço de ocorrências, que é euclidiano. Esta suposição é motivada pelo fato de que os valores das grandezas físicas medidas, individualmente ou em relação com outras, é que determinam o estado de sobrecarga. Estas grandezas físicas têm caráter contínuo, por isto não faz muito sentido falar no valor específico de uma delas e sim em uma faixa de valores. Uma motivação adicional é que as árvores produzidas pelo algoritmo baseiam-se na pertinência dos valores dos atributos a faixas de valores.

Assim, podemos falar em uma região hiper-esférica ou em uma união de algumas regiões desta forma. Outra possibilidade é a intersecção de semi-espaços, que podem determinar um sólido fechado ou aberto, limitado somente em algumas direções. Se, por um lado, esta configuração permite dar conta da idéia, intuitiva, de que algumas das faixas significativas podem se estender ao infinito, ela permite apenas conceitos determinados por uma única região contígua do espaço. Enquanto que a união de hiper-esferas admite a detecção de várias regiões significativas, embora não permita a caracterização de regiões infinitas. Esta limitação pode ser posta de lado levando-se em conta que as grandezas medidas estão, na verdade, limitadas e que, embora valores além destes limites também indiquem a sobrecarga, eles seriam atingidos de forma contínua, isto é, passando antes pela região limitada e

sinalizando a sobrecarga iminente.

Em particular, como a árvore de decisão testa apenas uma variável em cada nó, determinando faixas significativas de valores, as regiões que ela consegue caracterizar são, na verdade, limitadas por hiper-superfícies ortogonais às variáveis testadas.

De qualquer maneira, conforme [BEHW89] e referências ali citadas, tanto a intersecção de k semi-espacos quanto as uniões de k hiper-esferas ou de k hiper-paralelepípedos em um espaço euclidiano de dimensão n são espaços de conceitos cuja dimensão de Vapnik-Chervonenkis ($VCdim$) é dada por $k(n + 1)$.

Assim, supondo a existência de um algoritmo aprendiz consistente com a amostra de treinamento para os espaços de conceitos acima, o gráfico da Figura 5.3 exhibe as curvas de aprendizagem para diversos valores de k . O parâmetros de confiança foi fixado em $\delta = 0.01$.

Figura 5.3: Curva de aprendizagem por convergência uniforme para intersecção de k semi-espacos ou união de k hiper-sólidos

Novamente, os valores obtidos para o tamanho da amostra que garante a PAC-apreensibilidade do espaço de conceitos são muito maiores do que os valores efetivamente utilizados pelo REAL, embora bem menores do que os relativos a árvores booleanas.

Estas diferenças são devidas principalmente ao caráter de uniformidade da definição de apreensibilidade, presente tanto no modelo PAC quanto no modelo da Convergência Uniforme, e ainda em diversas variantes, apresentados no Capítulo 3: a complexidade de amostra é o número de exemplos que garante a precisão e a confiança desejados *para qualquer conceito no espaço de conceitos em questão*. Ocasionalmente, um conceito pode ser bem aprendido (no sentido de precisão e confiança) com um número menor de exemplos. Também não significa que os limites superiores mínimos conhecidos não possam ser ainda melhorados.

Uma vantagem do modelo de Convergência Uniforme é a apresentação de limites inferiores para o tamanho da amostra, em função da precisão e da confiança. Novamente, cabe ressaltar que esta curva se refere à apreensibilidade do espaço de conceitos e, assim, os limites significam que não é possível garantir a precisão e a confiança desejadas para todos os conceitos no espaço de conceitos, qualquer que seja o aprendiz consistente, se a amostra tiver tamanho menor. Isto é, há pelo menos um conceito no espaço cuja aprendizagem não pode ser realizada a partir de tão poucos exemplos aleatórios.

Figura 5.4: Limite inferior de convergência uniforme para intersecção de k semi-espacos ou união de k hiper-sólidos

A Figura 5.4 mostra o limite inferior para a curva de aprendizagem, com-

parada à curva do REAL. Observe, nesta Figura 5.4, que o REAL produz aproximações melhores que a qualidade mínima prevista pelo modelo de Convergência Uniforme com $k = 8$, com amostras muito pequenas. Novamente, isto não invalida qualquer um dos dois: o modelo diz apenas que não é possível garantir a aprendizagem de todos os conceitos neste espaço com tal precisão, mas não impede que alguns conceitos sejam bem aprendidos. Ainda assim, o resultado indica que muito provavelmente o conceito de sobrecarga na estação elétrica não é tão complexo quanto a união ou intersecção de $k = 8$ regiões.

A curva não decrescente após um certo tamanho mínimo indica a impossibilidade de se garantir uniformemente precisão e confiança absolutas.

Um enfoque mais específico

Com algumas considerações adicionais sobre o problema em questão, podemos tentar reduzir o valor de $VCdim(\mathcal{C})$ de modo a obter um limite superior para a curva de aprendizagem mais próximo dos valores empíricos.

Examinando cada uma das árvores geradas nos diversos testes vemos que nenhuma utilizou testes sobre mais do que 12 variáveis distintas. Com isto, podemos limitar nosso espaço de ocorrências relevante a $\mathfrak{R}^{12}$, o que reduz bastante os valores da dimensão VC do espaço de conceitos, nas suposições acima e, conseqüentemente, o número de exemplos necessários para garantir determinados valores para precisão, conforme a Figura 5.5, com o parâmetro de confiança ainda fixado em $\delta = 0.01$.

Por outro lado, uma árvore de decisão pode sempre ser rotulada segundo uma amostra de tamanho igual ao número de nós terminais da árvore. Entretanto, um exemplo adicional causará o acúmulo de dois exemplos num único nó terminal e, com isto, a árvore não poderá ser consistente com pelo menos um conceito no espaço de conceitos: um que rotule de maneira oposta as duas ocorrências acumuladas no mesmo nó terminal.

Este critério dispensa a necessidade de suposições sobre as características do espaço de conceitos, mas melhora pouco os resultados numéricos apresentados até aqui. O REAL não tem limitação quanto ao número de subárvores em que um nó interno pode ser expandido. Assim, também não tem um limite para o número de nós terminais. Recorrendo aos resultados empíricos obtidos, observamos árvores de até 55 nós terminais, o que dará números próximos aos obtidos com a suposição da união ou intersecção de regiões contínuas sobre $\mathfrak{R}^{28}$ com $k = 2$.

Figura 5.5: Convergência uniforme para $\mathcal{R}^{12}$

5.3 Considerações adicionais

Os resultados obtidos neste capítulo estão longe de ser conclusivos sobre a garantia de sucesso na aprendizagem do problema proposto. Entretanto, ilustram claramente a grande distância entre os resultados pretendidos pelos modelos formais e a implementação estudada. Esta se mostra muito mais eficiente do que se poderia prever.

Fica claro o baixo grau de adequação dos modelos formais aos problemas práticos. Há pouco espaço nos modelos para considerações específicas sobre o problema em questão, devido a uma preocupação excessiva com a generalidade dos resultados, quando muitas aplicações poderiam se valer de resultados menos gerais.

De qualquer maneira, observa-se uma grande dificuldade dos modelos quanto a amostras muito pequenas mas, à medida que seu tamanho aumenta, as distâncias diminuem consideravelmente e, sob suposições suficientemente simplificadoras, a ordem de grandeza da complexidade de amostra torna-se mais e mais próxima daquela obtida empiricamente.

Uma variação da análise aqui realizada que seria muito útil é o levantamento da curva de aprendizagem do algoritmo estudado para diversos con-

ceitos no mesmo espaço, conforme sugerido na descrição do problema. Neste caso específico, a aprendizagem das condições de sobrecarga iminente para outras estações de energia. Mais além, estaria a aplicação do algoritmo a problemas de natureza física ainda mais diversa, mas representável no mesmo espaço de hipóteses.

A parametrização de diferentes graus de convicção, possível na implementação do REAL, também pode influir na precisão e na confiança obtidas, mas também não foi explorada para manter o experimento num grau de simplicidade compatível com as suposições simplificadoras realizadas e com os objetivos do trabalho.

Capítulo 6

Conclusões

Neste trabalho, apresentamos uma visão geral do problema da aprendizagem computacional, destacando as duas frentes de desenvolvimento e a defasagem entre elas. Por um lado, foram apresentados diversos modelos formais de análise do problema, cujo objetivo comum é determinar o quanto pode ser aprendido em um determinado contexto e qual o volume de informação necessário para a garantia da precisão desejada. O modelo PAC foi apresentado mais detalhadamente por ter a maior clareza nas definições e a maior quantidade de resultados derivados, variantes e generalizações, na linha que representa. Esta é a linha da análise do tipo “pior caso”, em que os resultados são válidos para qualquer conceito possível num determinado espaço, com qualquer distribuição de probabilidades das ocorrências e dos conceitos. Outros modelos formais, na linha da inferência estatística, levam em conta a probabilidade dos diversos conceitos possíveis e a particular distribuição no espaço de ocorrências.

A outra frente de desenvolvimento foi apresentada na forma de algumas linhas correntes de implementação, com destaque para as árvores de decisão. Entre as linhas apresentadas, esta é a que mais carece de resultados formais. Além dos bons resultados empíricos, a clareza das generalizações obtidas pode ter contribuído para o abandono parcial desta questão, observado nos trabalhos pesquisados desta área. Com isto, fica em aberto a questão de quão boa pode ser a aproximação ao conceito alvo através de implementações deste tipo. Não há como demonstrar que a obtenção de aproximações satisfatórias está garantida, nem como determinar o volume de informação necessário para esta garantia.

Na parte dedicada ao estudo do problema prático, foram feitas várias suposições e simplificações, de maneira que fosse possível identificar alguns limites para a relação entre a complexidade da amostra e a precisão obtida no processo de aprendizagem. Entretanto, mesmo com estas simplificações

os resultados teóricos para esta relação estão muito distantes dos números obtidos empiricamente.

Com isto, pudemos ilustrar ao mesmo tempo o baixo grau de aplicabilidade dos resultados teóricos a problemas práticos de grande importância e a lacuna deixada pelos métodos empíricos de aprendizagem.

Os trabalhos mais recentes na linha dos modelos formais têm exibido maior preocupação com sua aplicabilidade, passando a considerar questões como

- a presença de imprecisões nos valores e na rotulação da amostra de treinamento;
- a impossibilidade ou inadequação da consistência do algoritmo com a amostra;
- a complexidade dos diversos conceitos em um espaço de conceitos;
- a particular distribuição subjacente ao espaço de ocorrências; e
- a probabilidade *a priori* dos conceitos possíveis num espaço de conceitos.

Algumas das linhas de implementação, por sua vez, estão mais diretamente ligadas a análise formal, seja por terem sua origem diretamente ligada ao problema desta análise, seja por produzirem hipóteses que não são diretamente validáveis por alguém que conhece o contexto da aplicação. Deste último caso, são exemplos as redes neurais que, embora possam ser validadas com resultados numéricos de testes, se apresentam como caixas pretas em relação à forma como o conhecimento foi aprendido. Muitas generalizações e variantes dos modelos formais, tanto da família do modelo PAC [Hau95] quanto da inferência estatística [Tis95], foram produzidas para aplicação direta a este modelo de implementação, embora continuem existindo diferenças entre os resultados previstos e os efetivamente alcançados.

Algumas linhas de desenvolvimento que se abrem diante do problema da compatibilização entre implementação e análise. A determinação da complexidade do espaço de conceitos representáveis por árvores de decisão de determinado tipo é a extensão mais imediata da análise realizada no Capítulo 5. Ela poderia levar em conta características estruturais dos espaços de ocorrências e de conceitos, como foi feito naquele Capítulo, ou, por outro lado, considerar o tipo de critério que é aplicado na construção da árvore, bem como as características das árvores possíveis segundo estes critérios. Por fim, resultados formais e generalizações obtidos para outras linhas de implementação podem ser adaptadas à linha das árvores de decisão.

Referências Bibliográficas

- [AB92] M. Anthony and N. Biggs. *Computational Learning Theory*. Cambridge University Press, Cambridge (Reino Unido), 1992.
- [Apt87] K. R. Apt. Introduction to logic programming. Technical Report CS-R8741, Centre for Mathematics and Computer Science, Amsterdam (Holanda), 1987.
- [BEHW89] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4):929–965, October 1989.
- [BG96] F. Bergadano and D. Gunetti. *Inductive Logic Programming From machine learning to software engineering*. The MIT Press, Cambridge, MA (EUA), 1996.
- [BU95] C. E. Brodley and P. E. Utgoff. Multivariate decision trees. *Machine Learning*, 19:45–77, 1995.
- [Che95] P. Cheeseman. On bayesian model selection. In Wolpert [Wol95a], pages 315–330.
- [CLR90] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA (EUA), 1990.
- [dSB98] Rodrigo de Salvo Braz. Alto nível em redes neurais. Dissertação de mestrado, Instituto de Matemática e Estatística da Universidade de São Paulo, São Paulo, SP, 1998.
- [EH89] A. Ehrenfeucht and D. Haussler. Learning decision trees from random examples. *Information and Computation*, 82:231–246, 1989.

- [Hau95] D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. In Wolpert [Wol95a], pages 37–116.
- [HW95] D. Haussler and M. Warmuth. The probably approximately correct (PAC) and other learning models. In Wolpert [Wol95a], pages 17–36.
- [Kea90] M. J. Kearns. *The Computational Complexity of Machine Learning*. MIT Press, Cambridge, MA (EUA), 1990.
- [Kea93] M. Kearns. Efficient noise-tolerant learning from statistical queries. In *Proceedings of the 25th ACM Symposium on the Theory of Computing*, pages 392–401. ACM Press, 1993.
- [KV94] M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA (EUA), 1994.
- [Lau96] Marcelo S. Laurotto. Árvores de classificação para escolha de estratégias de operação em mercados de capitais. Dissertação de mestrado, Instituto de Matemática e Estatística da Universidade de São Paulo, São Paulo, SP, 1996.
- [Mar97] J. Kent Martin. An exact probability metric for decision tree splitting and stopping. *Machine Learning*, 28:257–291, 1997.
- [Mel92] C. Mellish. Machine learning. Outline lecture notes, University of Edinburgh, Edimburgo (Escócia), 1992.
- [Min89] Marvin Minsky. *A Sociedade da Mente*. Francisco Alves, Rio de Janeiro, RJ, 1989.
- [MR94] S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19(20):629–679, 1994.
- [Mug91] S. Muggleton. Inductive logic programming. *New Generation Computing*, 8(4):295–318, 1991.
- [QR89] J. R. Quinlan and R. L. Rivest. Inferring decision trees using the minimum description length principle. *Information and Computation*, 80, 1989.

- [Qui87] J. R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27, 1987.
- [Qui89] J. R. Quinlan. Unknown attribute values in induction. In *Proceedings of the Sixth International Workshop on Machine Learning*, Ithaca, NY (EUA), 1989. Morgan Kaufmann.
- [Qui96] J. R. Quinlan. Improved use of continuous attributes in C4.5. *Journal of Artificial Intelligence Research*, 4, 1996.
- [RD94] L. De Raedt and S. Dzeroski. First order jk-clausal theories are PAC-learnable. *Artificial Intelligence*, 70:375–392, 1994.
- [Riv87] R. L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.
- [SNLR98a] J. M. Stern, F. Nakano, M. S. Lauretto, and C. O. Ribeiro. REAL: Algoritmo de aprendizagem para atributos reais e estratégias de operação em mercado. In *Proceedings of IBERAMIA'98 - Sixth Iberoamerican Conference on Artificial Intelligence*, Lisboa (Portugal), 1998.
- [SNLR98b] J. M. Stern, F. Nakano, M. S. Lauretto, and C. O. Ribeiro. REAL: Real attribute learning algorithm. In *World Multiconference on Systemics, Cybernetics and Informatics - SCI'98 Proceedings*, volume 2, pages 315–321, 1998.
- [Tho92] C. Thornton. Machine learning. Lecture notes, University of Edinburgh, Edimburgo (Escócia), 1992.
- [Tis95] N. Tishby. Statistical physics models of supervised learning. In Wolpert [Wol95a], pages 215–242.
- [UBC97] P. E. Utgoff, N. C. Berkman, and J. A. Clouse. Decision tree induction based on efficient tree restructuring. *Machine Learning*, 29:5–44, 1997.
- [Val84] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11), 1984.
- [Wir89] N. Wirth. *Algoritmos e Estruturas de Dados*. Prentice-Hall, Rio de Janeiro, RJ, 1989.

- [Wol95a] D. H. Wolpert, editor. *The Mathematics of Generalization – Proceedings of the SFI/CNLS Workshop on Formal Approaches to Supervised Learning*, volume 20 of *SFI Studies in the Sciences of Complexity*, Reading, MA (EUA), 1995. Santa Fe Institute, Addison-Wesley.
- [Wol95b] D. H. Wolpert. The relationship between PAC, the statistical physics framework, the bayesian framework, and the VC framework. In *The Mathematics of Generalization – Proceedings of the SFI/CNLS Workshop on Formal Approaches to Supervised Learning* [Wol95a], pages 117–214.
- [Wol95c] D. H. Wolpert. The status of supervised learning science circa 1994: The search for a consensus. In *The Mathematics of Generalization – Proceedings of the SFI/CNLS Workshop on Formal Approaches to Supervised Learning* [Wol95a], pages 1–10.