

# Data Mining

UM GUIA PRÁTICO

RONALDO GOLDSCHMIDT  
EMMANUEL PASSOS

# Data Mining

UM GUIA PRÁTICO

**Conceitos, técnicas, ferramentas, orientações e aplicações**



4ª Tiragem



© 2005, Elsevier Editora Ltda.

Todos os direitos reservados e protegidos pela Lei 9.610 de 19/02/1998.

Nenhuma parte deste livro, sem autorização prévia por escrito da editora, poderá ser reproduzida ou transmitida sejam quais forem os meios empregados: eletrônicos, mecânicos, fotográficos, gravação ou quaisquer outros.

*Copidesque:*

Elisa Rosa de Alencar

*Editoração Eletrônica:*

Estúdio Castellani

*Revisão Gráfica:*

Roberto Mauro dos Santos Facce

*Projeto Gráfico*

Elsevier Editora Ltda.

A Qualidade da Informação.

Rua Sete de Setembro, 111 – 16º andar

20050-006 Rio de Janeiro RJ Brasil

Telefone: (21) 3970-9300 FAX: (21) 2507-1991

E-mail: [info@elsevier.com.br](mailto:info@elsevier.com.br)

*Escritório São Paulo:*

Rua Quintana, 753/8º andar

04569-011 Brooklin São Paulo SP

Tel.: (11) 5105-8555

ISBN 13: 978-85-352-1877-0

ISBN 10: 85-352-1877-7

**Nota:** Muito zelo e técnica foram empregados na edição desta obra. No entanto, podem ocorrer erros de digitação, impressão ou dúvida conceitual. Em qualquer das hipóteses, solicitamos a comunicação à nossa Central de Atendimento, para que possamos esclarecer ou encaminhar a questão.

Nem a editora nem o autor assumem qualquer responsabilidade por eventuais danos ou perdas a pessoas ou bens, originados do uso desta publicação.

Central de atendimento

Tel: 0800-265340

Rua Sete de Setembro, 111, 16º andar – Centro – Rio de Janeiro

e-mail: [info@elsevier.com.br](mailto:info@elsevier.com.br)

site: [www.campus.com.br](http://www.campus.com.br)

CIP-Brasil. Catalogação-na-fonte.  
Sindicato Nacional dos Editores de Livros, RJ

---

G575d

Goldschmidt, Ronaldo

Data mining : um guia prático / Ronaldo Goldschmidt,  
Emmanuel Passos. – Rio de Janeiro : Elsevier, 2005 –

4ª Reimpressão.

ISBN 85-352-1877-7

1. Data mining. 2. Banco de dados. 3. Sistema de  
informação. I. Passos, Emmanuel. II. Título.

---

05-2124.

CDD 006.3

CDU 004.89

# Prefácio

*Descoberta de Conhecimento* talvez seja, neste instante, o tópico de maior relevância prática em toda a área de Informática.

No início, os computadores serviam para nos livrar das tarefas de rotina: absorviam dados, faziam contas e imprimiam relatórios. Como alguns desses dados precisassem ser constantemente acessados para consulta e atualização por diferentes tipos de usuários, sendo muitas vezes compartilhados por mais de um sistema automatizado de informação, surgiu e, depois de alguns anos, passou a ser amplamente utilizada a tecnologia de bancos de dados. E a partilha de dados foi se intensificando progressivamente com as ligações através de redes.

Durante todo esse tempo, um sonho de todo especialista em sistemas foi o de ver as pessoas se tornarem mais racionais na tomada de decisões, pelo menos a partir do dia em que pudessem contar com um acesso rápido e fácil a esses vastos repositórios de dados. Mas o sonho nem sempre se realiza, infelizmente. Examinar grandes volumes de dados brutos requer esforço, interpretá-los corretamente pode exigir talento e um nível de capacidade técnica não trivial, até mesmo para os especialistas envolvidos.

É nesse ponto que a Informática vem em nosso socorro, passando, de certa forma, a iniciativa e a responsabilidade pelo exame e interpretação dos dados ao computador. Descobrir conhecimento é extrair dos dados o que eles implicam em termos de *riscos* – a evitar – e *oportunidades* – a serem aproveitadas. Finalmente a máquina pode ajudar nas questões em que as regras do negócio devem ficar explícitas e ser bem compreendidas. E não apenas na hora de tomar cada decisão, mas também para o planejamento das atividades a médio e longo prazos.

O presente livro dos professores Ronaldo Goldschmidt e Emmanuel Passos nos oferece um tratamento completo do tópico, cobrindo os aspectos práticos e fornecendo a base teórica. Não é uma simples coletânea e manual de uso de ferramentas disponíveis no mercado, embora também cubra essa parte. A grande distinção do livro (cf. Capítulo 7) é apresentar de forma sistemática metodologias para desenvolver aplicações bem sucedidas. O livro é de inestimável utilidade para profissionais de qualquer área, tanto na administração pública quanto empresarial. Mas também é um livro texto obrigatório para cursos de graduação ou pós-graduação em programas acadêmicos que envolvam Tecnologia da Informação, seja como assunto principal ou complementar.

Conheço os autores de longa data, principalmente o professor Emmanuel Passos. Egresso do mestrado em Informática da PUC-Rio em 1971, recebeu o grau de doutor em Engenharia de Sistemas e Computação da Universidade Federal do Rio de Janeiro (UFRJ), em 1981. Em 1991, seguiu programa de pós-doutoramento na Purdue University, nos Estados Unidos. Dedicou-se a várias linhas de pesquisa, sempre com ênfase em Inteligência Artificial, da qual é considerado um dos pioneiros em nosso país. Exerceu atividade de ensino em diversas instituições, entre as quais a própria PUC-Rio, onde tive o prazer de ser seu colega, na UFRJ e no Instituto Militar de Engenharia (IME). No IME, teve atuação destacada como coordenador de pós-graduação em Informática.

Nestes últimos anos, o professor Emmanuel vem-se dedicando em tempo integral à pesquisa e ao desenvolvimento de protótipos, no campo de Descoberta de Conhecimento, para o qual sua experiência em técnicas de Inteligência Artificial, tais como Redes Neurais, Lógica Nebulosa e Algoritmos Genéticos, o torna especialmente qualificado. No decorrer dos trabalhos, produziu e comprovou, por meio de aplicações, ambientes e ferramentas para Descoberta de Conhecimento. O esforço tem tido reconhecimento dentro e fora do Brasil; neste ano de 2005, teve aceito e publicado em prestigioso evento internacional um artigo encabeçado pelo outro autor deste livro, que, sob sua orientação completou em 2003 o doutorado em Engenharia Elétrica na PUC-Rio (Goldschmidt, R. R. ; Passos, E. P. L. ; Vellasco, M. M. B. R. "Hybrid Assistance in KDD Task definition". In: *1st WSEAS International Symposium on Data Mining*, 2005, Corfu Island – Greece. Proceedings of the 1st WSEAS International Symposium on Data Mining, 2005. v. I).

Devo ainda relatar meu contato com o próprio Ronaldo Goldschmidt, ex-aluno e desde então professor (IME, Centro Universitário da Cidade do Rio de Janeiro, Instituto Superior Tecnológico – FAETEC, Faculdades Integradas de Jacarepaguá) e também colaborador do professor Emmanuel em todos os projetos. Por duas vezes, convidei o professor Ronaldo a expor o tópico Descoberta de Conhecimento para meus alunos de doutorado e mestrado em Infor-

mática. Suas palestras, no melhor estilo de apresentação em formato PowerPo-  
int, foram seguidas por animadíssimos diálogos com os alunos. Posso assim afir-  
mar, para concluir, que essa clareza e boa organização dos itens, esse talento di-  
dático em suma, já observado por mim quanto ao professor Emmanuel, confir-  
mou-se em seu jovem associado. E transparece no texto do livro, tornando sua  
leitura compreensível e agradável.

*Antonio L. Furtado*

*Professor titular do Departamento de Informática da  
Pontifícia Universidade Católica do Rio de Janeiro*

# Introdução

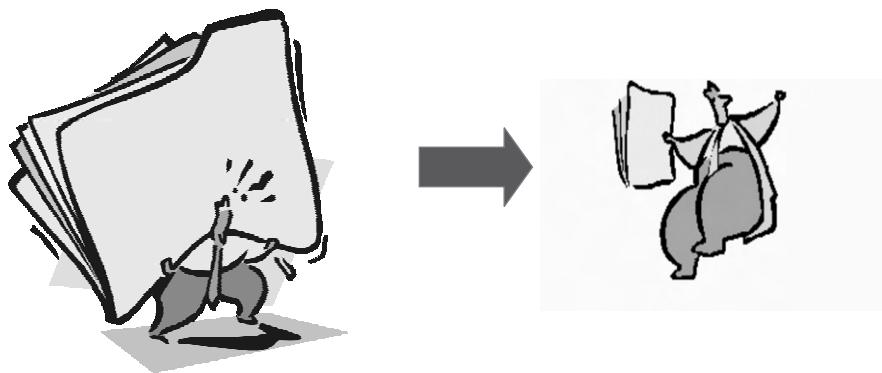
## KDD – Descoberta de Conhecimento em Bases de Dados: Uma Visão Geral

Os constantes avanços na área da Tecnologia da Informação têm viabilizado o armazenamento de grandes e múltiplas bases de dados. Tecnologias como a Internet, sistemas gerenciadores de banco de dados, leitores de códigos de barras, dispositivos de memória secundária de maior capacidade de armazenamento e de menor custo e sistemas de informação em geral são alguns exemplos de recursos que têm viabilizado a proliferação de inúmeras bases de dados de natureza comercial, administrativa, governamental e científica.

Atualmente, dados científicos em projetos de pesquisa, tais como missões espaciais da NASA e o Projeto do Genoma Humano, têm alcançado proporções gigantescas. Empresas como FedEx, Wal-Mart, UPS, Banco do Brasil, Caixa Econômica Federal e Sendas possuem bases de dados da ordem de centenas de terabytes de informações.

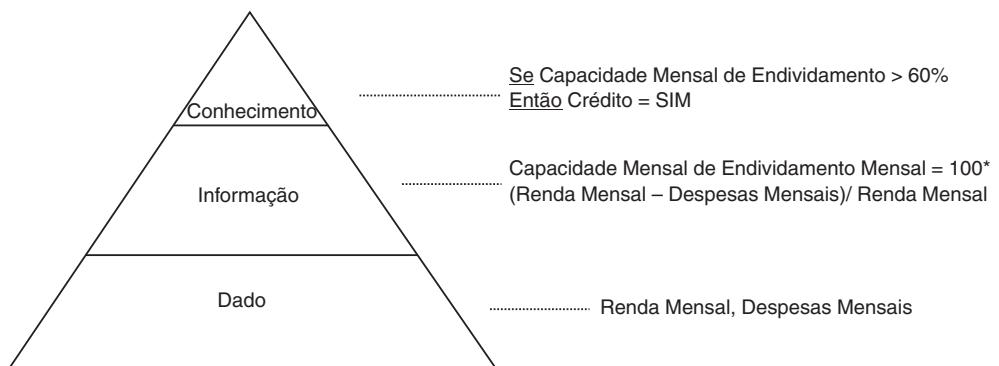
Diante desse cenário, naturalmente surgem algumas questões como: “O que fazer com todos os dados armazenados?”, “Como utilizar o patrimônio digital em benefício das instituições?”, “Como analisar e utilizar de maneira útil todo o volume de dados disponível?”, entre outras.

A análise de grandes quantidades de dados pelo homem é inviável sem o auxílio de ferramentas computacionais apropriadas. Portanto, torna-se imprescindível o desenvolvimento de ferramentas que auxiliem o homem, de forma automática e inteligente, na tarefa de analisar, interpretar e relacionar esses dados para que se possa desenvolver e selecionar estratégias de ação em cada contexto de aplicação.



Para atender a este novo contexto, surge uma nova área denominada Descoberta de Conhecimento em Bases de Dados (*Knowledge Discovery in Databases – KDD*), que vem despertando grande interesse junto às comunidades científica e industrial. A expressão *Mineração de Dados*, mais popular, é, na realidade, uma das etapas da Descoberta de Conhecimento em Bases de Dados. Ambas serão mais detalhadas adiante.

Neste momento, para proporcionar um melhor entendimento do problema, é importante destacar as diferenças e a hierarquia entre dado, informação e conhecimento, conforme ilustra a Figura 1.1.



**Figura 1.1.** Hierarquia entre Dado, Informação e Conhecimento.

Os dados, na base da pirâmide, podem ser interpretados como itens elementares, captados e armazenados por recursos da Tecnologia da Informação. No exemplo apresentado, consideremos uma base de dados de uma financeira que armazene a renda mensal e as despesas mensais de seus clientes.

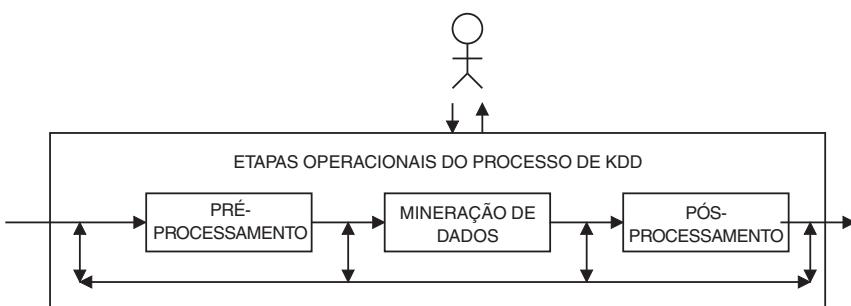
As informações representam os dados processados, com significados e contextos bem definidos. Diversos recursos da Tecnologia da Informação são utili-

zados para facilmente processar dados e obter informações. No exemplo, a capacidade mensal de endividamento é uma informação calculada a partir dos dados de renda e despesas mensais de cada cliente. Indica um valor percentual do quanto um cliente da financeira pode contrair de empréstimos em relação à sua renda mensal.

No topo da pirâmide está o conhecimento, padrão ou conjunto de padrões cuja formulação pode envolver e relacionar dados e informações. No exemplo, o conhecimento encontra-se representado na forma de uma regra de produção. Regras de produção têm a forma *SE <condições> ENTÃO <conclusões>* e, como outras formas de representação do conhecimento, serão mais bem explicadas ao longo do texto. Em geral, o conhecimento não pode ser abstraído das bases de dados por recursos tradicionais da Tecnologia da Informação. A busca por novos conhecimentos a partir dos dados é o tema principal deste livro.

O termo KDD foi formalizado em 1989 em referência ao amplo conceito de procurar conhecimento a partir de bases de dados. Uma das definições mais populares foi proposta em 1996 por um grupo de pesquisadores (Fayyad et al., 1996a): “KDD é um processo, de várias etapas, não trivial, interativo e iterativo, para identificação de padrões comprehensíveis, válidos, novos e potencialmente úteis a partir de grandes conjuntos de dados”.

A Descoberta de Conhecimento em Bases de Dados é caracterizada como um processo composto por várias *etapas operacionais*. A Figura 1.2 apresenta um resumo pragmático das etapas operacionais executadas em processos de KDD. Neste resumo, a etapa de pré-processamento compreende as funções relacionadas à captação, à organização e ao tratamento dos dados. A etapa de pré-processamento tem como objetivo a preparação dos dados para os algoritmos da etapa seguinte, a Mineração de Dados. Durante a etapa de Mineração de Dados, é realizada a busca efetiva por conhecimentos úteis no contexto da aplicação de KDD. A etapa de pós-processamento abrange o tratamento do conhecimento obtido na Mineração de Dados. Tal tratamento, nem sempre necessário, tem como objetivo viabilizar a avaliação da utilidade do conhecimento descoberto (Fayyad et al., 1996a).



**Figura 1.2.** *Etapas Operacionais do Processo de KDD.*

De uma maneira geral, a complexidade do processo de KDD está na dificuldade em perceber e interpretar adequadamente inúmeros fatos observáveis durante o processo e na dificuldade em conjugar dinamicamente tais interpretações de forma a decidir quais ações devem ser realizadas em cada caso (Goldschmidt, 2003). Cabe ao analista humano a árdua tarefa de orientar a execução do processo de KDD.

Na definição formal de KDD apresentada anteriormente, o termo *iterativo* indica a necessidade de atuação do homem como responsável pelo controle do processo. O homem utiliza os recursos computacionais disponíveis em função da análise e da interpretação dos fatos observados e resultados obtidos ao longo do processo.

O termo *iterativo*, por outro lado, sugere a possibilidade de repetições integrais ou parciais do processo de KDD na busca de resultados satisfatórios por meio de refinamentos sucessivos.

A expressão *não trivial* alerta para a complexidade normalmente presente na execução de processos de KDD. Maiores considerações sobre as dificuldades inerentes ao processo de KDD encontram-se na seção “Ferramentas de KDD” do Capítulo 2.

Ainda considerando a definição de KDD, um padrão deve ser interpretado como um conhecimento representado segundo as normas sintáticas de alguma linguagem formal (Fayyad et al., 1996a). Um padrão comprehensível refere-se, portanto, a um padrão representado em alguma forma de representação do conhecimento que possa ser interpretada pelo homem.

A expressão *padrão válido* indica que o conhecimento deve ser verdadeiro e adequado ao contexto da aplicação de KDD.

Um *padrão novo* deve acrescentar novos conhecimentos aos conhecimentos existentes no contexto da aplicação de KDD.

E, finalmente, um *conhecimento útil* é aquele que pode ser aplicado de forma a proporcionar benefícios ao contexto da aplicação de KDD.

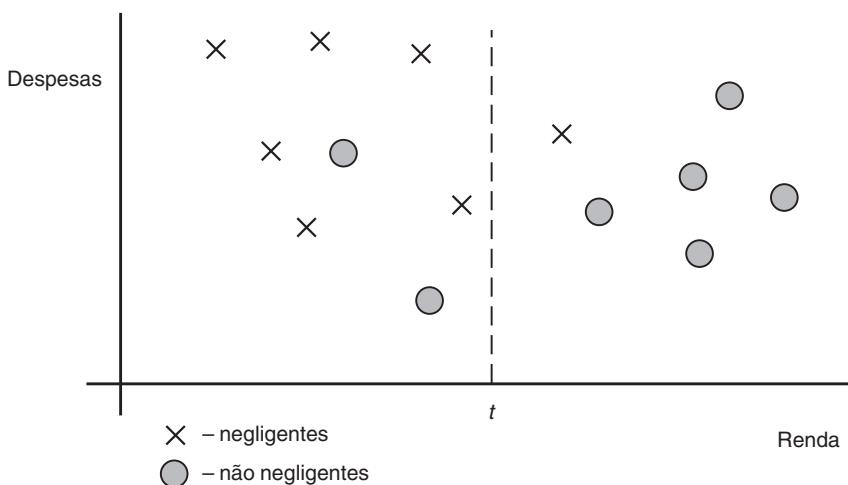
Consideremos para fins ilustrativos o exemplo na área de concessão de crédito da Figura 1.3 (Naliato, 2001). Cada ponto do plano cartesiano representa um dos clientes de uma financeira. O conjunto de pontos forma a base de fatos (banco de dados). A fim de facilitar a visualização do problema, foram utilizados apenas três atributos: renda, despesas e o comportamento do cliente com relação ao crédito recebido da financeira. Renda e despesas estão representadas pelos eixos do plano. O comportamento do cliente está representado por duas classes: X – clientes negligentes (não pagaram o crédito recebido) e O – Clientes pontuais (pagaram o crédito recebido). Genericamente falando, o problema pode envolver  $n$  atributos, sendo representado em um hiperespaço n-dimensional. Neste exemplo, a fim de melhor direcionar os seus recursos, seria útil para a financeira obter conhecimentos a partir dos dados existentes que permitissem discernir entre novos clientes os prováveis negligentes e os prováveis clientes pontuais.

Diversos modelos de conhecimento podem ser abstraídos do exemplo da Figura 1.3. Há diversas formas de separar os dois conjuntos de clientes. Um exemplo pode ser representado pela regra:

*SE Renda >= R\$ t*

*ENTÃO Cliente = Não negligente*

*SENÃO Cliente = Negligente*



**Figura 1.3.** Base de fatos de uma financeira hipotética.

A regra acima é um padrão compreensível pelo homem. Embora existam casos que não a satisfaçam, essa regra é válida para a maioria dos registros existentes na base. Uma das medidas normalmente utilizada na avaliação da qualidade de uma regra é a *acurácia*, também denominada *confiança* ou *precisão da regra*. Refere-se à proporção dos casos que satisfazem ao antecedente e ao consequente da regra em relação ao número de casos que satisfazem somente ao antecedente dessa regra. Neste exemplo, a separação dos conjuntos de clientes representada pela regra é linear. Cabe ressaltar, no entanto, que este problema não é linearmente separável, ou seja, os dois conjuntos (clientes negligentes e não negligentes) não podem ser separados por uma linha reta.

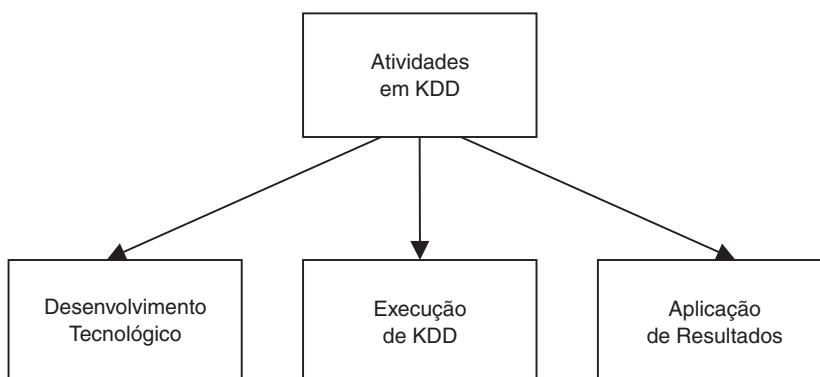
A Descoberta de Conhecimento em Bases de Dados é multidisciplinar e, historicamente, origina-se de diversas áreas, dentre as quais podem ser destacadas:

- Estatística
- Inteligência Computacional e Aprendizado de Máquina
- Reconhecimento de Padrões
- Banco de Dados

Com o propósito de melhor situar a área de KDD, a Figura 1.4 apresenta uma taxonomia das atividades na área da Descoberta de Conhecimento em Bases de Dados (Goldschmidt, 2003). Essa taxonomia mostra a diversidade de atividades relacionadas ao contexto de KDD.

As atividades na área de KDD podem ser organizadas em três grandes grupos: atividades voltadas ao desenvolvimento tecnológico, atividades de execução de processos de KDD e atividades envolvendo a aplicação de resultados obtidos em processos de KDD. A seguir encontram-se comentados os itens desta taxonomia.

- **Desenvolvimento Tecnológico** – Esse item abrange todas as iniciativas de concepção, aprimoramento e desenvolvimento de algoritmos, ferramentas e tecnologias de apoio que possam ser utilizados na busca por novos conhecimentos em grandes bases de dados.
- **Execução de KDD** – Esse item refere-se às atividades voltadas à busca efetiva de conhecimento em bases de dados. As ferramentas produzidas pelas atividades de desenvolvimento tecnológico são utilizadas na execução de processo de KDD.
- **Aplicação de Resultados** – Finalmente, uma vez obtidos modelos de conhecimento úteis a partir de grandes bases de dados, as atividades se voltam à aplicação dos resultados no contexto em que foi realizado o processo de KDD. Exemplos comuns de aplicação de resultados são as alterações em estratégias de negócios que tenham como objetivo procurar tirar proveito do conhecimento obtido. Tais alterações podem variar desde o posicionamento de produtos nas gôndolas de um mercado até políticas estratégicas corporativas (Agrawal et al., 1993; Goldschmidt & Passos, 2000; Godoy et al., 2003). O desenvolvimento de sistemas que utilizem conhecimentos extraídos de bases de dados tem propiciado valiosas ferramentas de apoio à decisão (Weiss & Indurkhy, 1998).



**Figura 1.4.** Taxonomia de Atividades na Área de KDD.

Ainda em termos históricos, a Mineração de Dados, usualmente utilizada para referenciar a Descoberta de Conhecimento em Bases de Dados, pode ser dividida em quatro gerações.

A primeira geração de Mineração de Dados (Piatetsky-Shapiro, 1999) apareceu nos anos 80 e consistia em ferramentas de análise voltadas a uma única tarefa, sem suporte às demais etapas do processo. Essas tarefas incluíam, em geral, a construção de classificadores usando ferramentas de Indução de Regras (por exemplo, C4.5) ou de Redes Neurais (por exemplo, *BackPropagation*), a descoberta de *clusters* (grupos) nos dados (por exemplo, *K-Means*), ou ainda a visualização de dados.

A segunda geração de sistemas de Mineração de Dados (Piatetsky-Shapiro, 1999) apareceu em 1995 com o desenvolvimento de ferramentas chamadas “*suites*”. Essas ferramentas eram dirigidas ao fato de que o processo de descoberta do conhecimento requer múltiplos tipos de análise dos dados. As “*suites*”, tais como *SPPS*, *Clementine*, *Intelligent Miner* e *SAS Enterprise Miner*, permitiam ao usuário realizar diversas tarefas de descoberta (geralmente classificação, clusterização e visualização) e suportavam transformação de dados. O Capítulo 6 comenta sobre diversas destas ferramentas de forma mais detalhada.

Embora a segunda geração de sistemas de Mineração de Dados enfatize a análise de dados, tais sistemas requerem conhecimento significativo da teoria estatística, não devendo ser usados diretamente pelo usuário, sem o auxílio de especialistas em análise de dados. Assim, surgiu a necessidade da terceira geração de Mineração de Dados (Piatetsky-shapiro, 1999) no final dos anos 90. Essas soluções são orientadas para a resolução de um problema específico da empresa, como, por exemplo, detecção de fraudes em cartão de crédito. Nesta geração, as interfaces são orientadas para o usuário e procuram esconder toda a complexidade da Mineração de Dados. O *HNC Software's Falcon* (Rainho, 2001) para detecção de fraudes em cartão de crédito é um exemplo deste tipo de sistema.

A quarta geração de Mineração de Dados compreende o desenvolvimento e a aplicação de técnicas e ferramentas que auxiliem o homem na própria condução do complexo processo de KDD (Goldschmidt, 2003). Também no Capítulo 6, algumas destas ferramentas encontram-se apresentadas e analisadas.

## Organização do Texto

Conforme o próprio título sugere, este livro reúne desde material teórico e formal até experiências e orientações práticas reais sobre como conduzir e executar aplicações na área de KDD. Cabe ressaltar que embora existam alguns poucos livros nacionais muito bons sobre Mineração de Dados, nenhum deles aborda com profundidade todo o processo de Descoberta de Conhecimento em Bases de Dados, lacuna que tentamos suprir com esta publicação.

Este livro tem como público alvo: pessoas ligadas à área de Tecnologia da Informação alocadas em empresas interessadas em utilizar dados históricos na

busca por novos conhecimentos, além de alunos de cursos de graduação e pós-graduação em Computação.

Assim sendo, o texto mescla uma abordagem conceitual e formal com linguagem acessível, recomendada a todos os tipos de leitores, seguido de informações de cunho mais prático, voltado ao público com interesse na aplicação da tecnologia.

Por razões óbvias, o texto pressupõe que o leitor esteja familiarizado com a área de Banco de Dados. Mesmo assim, o Anexo I apresenta uma revisão sucinta sobre DataWarehouses.

O Capítulo 2 complementa a visão geral das áreas de KDD e Mineração de Dados, com conceitos básicos necessários aos capítulos subsequentes.

O Capítulo 3 enfoca a Descoberta de Conhecimento em Bases de Dados como processo, detalhando suas etapas e algumas das funções de KDD mais utilizadas. Um mesmo exemplo de banco de dados é considerado ao longo do capítulo de forma que o leitor possa acompanhar de forma encadeada todo o processo.

Algumas das principais tarefas de KDD são apresentadas e exemplificadas em detalhe no Capítulo 4.

O Capítulo 5 apresenta diversos métodos de Mineração de Dados. Para os leitores sem conhecimento em Inteligência Computacional, recomendamos, para um maior aproveitamento, que a leitura deste capítulo seja precedida pela leitura dos Anexos II, III e IV que contêm noções introdutórias sobre Redes Neurais, Lógica Nebulosa e Algoritmos Genéticos, respectivamente.

No Capítulo 6 são apresentadas e analisadas ferramentas de KDD. Estas se subdividem em ferramentas operacionais, voltadas à execução efetiva das funções de KDD e ferramentas de administração e controle do processo. De forma a facilitar a comparação dentre as ferramentas de cada grupo, dois modelos são apresentados e utilizados na descrição das ferramentas.

O Capítulo 7 trata de um tópico de grande importância no processo de Descoberta de Conhecimento em Bases de Dados: a caracterização de uma metodologia para orientar o processo de KDD. Tal metodologia é descrita em detalhe e alguns mecanismos de controle foram propostos.

No Capítulo 8 são apresentadas, a título ilustrativo, algumas das nossas principais experiências práticas reais vivenciadas em projetos envolvendo Mineração de Dados.

O Capítulo 9 apresenta um resumo das principais tendências na área da Mineração de Dados e fornece algumas orientações para os leitores que atuem ou pretendam atuar na área.

Além dos capítulos mencionados, o livro dispõe ainda de:

- Algumas indicações de sites interessantes na área.
- Um CD-ROM contendo uma versão de demonstração do Bramining, ferramenta operacional de KDD.
- Um anexo com orientações sobre como utilizar o Bramining, ilustrado com exemplos.

# O Processo de KDD: Conceitos Básicos

## Caracterização do Processo de KDD

Para uma melhor compreensão do processo de KDD é necessária uma apresentação dos principais elementos envolvidos em aplicações nesta área. Basicamente, uma aplicação de KDD é composta por três tipos de componentes: o problema em que será aplicado o processo de KDD, os recursos disponíveis para a solução do problema e os resultados obtidos a partir da aplicação dos recursos disponíveis em busca da solução do problema. A seguir estão detalhamentos e comentários sobre cada um dos referidos componentes.

- a) O *problema* a ser submetido ao processo de KDD. Este componente pode ser caracterizado por três elementos: conjunto de dados, o especialista do domínio da aplicação e objetivos da aplicação.
  - Todo *conjunto de dados* pode ser observado sob os aspectos intensional e extensional (Date, 2000; Elmasri & Navathe, 1989). O aspecto intensional se refere à estrutura ou ao esquema do conjunto de dados. Neste contexto encontram-se, portanto, as características ou atributos do conjunto de dados. Os casos ou registros compõem o aspecto extensional do conjunto de dados. Em geral, o processo de KDD pressupõe que os dados sejam organizados em uma única estrutura tabular bidimensional contendo casos e características do problema a ser analisado. É importante destacar que o processo de KDD não requer que os dados a serem analisados pertençam a DataWarehouses (conceitos básicos no Anexo I). No entanto, o tratamento e a consolidação dos dados necessários à estruturação e carga neste tipo de ambiente é extremamente útil e desejável ao processo de KDD.
  - O *especialista no domínio da aplicação* representa a pessoa ou o grupo de pessoas que conhece o assunto e o ambiente em que deverá ser realizada a aplicação de KDD. Em geral, pertencem a esta classe analistas de negócios

interessados em identificar novos conhecimentos que possam ser utilizados em sua área de atuação. Costumam deter o chamado conhecimento prévio sobre o problema (“*background knowledge*”). As informações prestadas pelas pessoas deste grupo são de fundamental importância no processo de KDD, pois influenciam desde a definição dos objetivos do processo até a avaliação dos resultados.

- Os *objetivos da aplicação* compreendem as características esperadas do modelo de conhecimento a ser produzido ao final do processo. Tais objetivos retratam, portanto, restrições e expectativas dos especialistas no domínio da aplicação acerca do modelo de conhecimento a ser gerado. A precisão mínima de um modelo de conhecimento é um exemplo de restrição ou expectativa acerca de um modelo de classificação. No exemplo de concessão de crédito em uma financeira, 85% poderia ser uma precisão mínima do modelo de conhecimento em prever corretamente o comportamento dos clientes no pagamento dos créditos a eles cedidos. Somente modelos com precisão acima deste limiar poderiam ser considerados aceitáveis para esta aplicação. Em geral, os objetivos de uma aplicação são definidos em função da opinião dos especialistas no domínio desta aplicação. No entanto, convém ressaltar que são muito comuns os casos de aplicações de KDD em que os objetivos não estejam bem definidos no início do processo (Engels, 1996; Engels et al., 1997; Wirth et al., 1997; Verdenius & Engels, 1997). Tais objetivos devem ser refinados ao longo do processo de KDD em função dos resultados intermediários obtidos.
- b) Os *recursos disponíveis* para solução do problema mencionado. Entre eles podem ser destacados: o especialista em KDD, as ferramentas de KDD e plataforma computacional disponível.
  - O *especialista em KDD* representa a pessoa ou o grupo de pessoas que possui experiência na execução de processos de KDD. Interage com o especialista no domínio da aplicação e direciona a condução do processo de KDD, definindo o que, como e quando deve ser realizada cada ação do processo. Suas atribuições variam desde a identificação e a utilização do conhecimento prévio existente sobre o problema até o direcionamento das ações do processo, que englobam a seleção e a aplicação das ferramentas disponíveis, além da avaliação dos resultados obtidos.
  - A expressão *ferramenta de KDD* está sendo empregada para designar qualquer recurso computacional que possa ser utilizado no processo de análise de dados. Pode ser desde um ambiente de *software* que integre diversas funcionalidades de tratamento e análise de dados até algoritmos isolados que possam ser adaptados ao processo de KDD.

- A *plataforma computacional*, conforme o próprio nome sugere, indica os recursos computacionais de *hardware* (processadores e memória) disponíveis para a execução da aplicação de KDD. São os equipamentos disponibilizados para o processo. Podem ser desde máquinas isoladas até mesmo ambientes computacionais paralelos. Quanto maior a capacidade de processamento e memória da plataforma computacional, maior a agilidade em obter resultados, proporcionando uma maior dinâmica ao processo de KDD.
- c) Os *resultados obtidos* a partir da aplicação dos recursos no problema. Compreende, fundamentalmente, os modelos de conhecimento descobertos ao longo da aplicação de KDD e o histórico das ações realizadas.
- A expressão *modelo de conhecimento* indica qualquer abstração de conhecimento, expresso em alguma linguagem, que descreva algum conjunto de dados (Fayyad et al., 1996a). Todo modelo de conhecimento deve ser avaliado com relação ao cumprimento das expectativas definidas nos objetivos da aplicação. É muito comum que, durante o processo de KDD, sejam realizadas comparações entre os modelos de conhecimento obtidos. Por exemplo, imaginemos que no caso dos empréstimos da financeira fossem obtidos diversos modelos. Suponhamos que todas as características desses modelos sejam iguais, a menos da acurácia. Consideremos ainda que todos os modelos tenham acurácia superior à mínima desejada. O modelo de conhecimento com maior precisão na classificação dos clientes possui maiores chances de ser eleito como principal resultado (produto final) gerado pelo processo de KDD.
  - Os *históricos* sobre como os modelos de conhecimento foram gerados também enquadram-se como resultados do processo de KDD. São de fundamental importância no controle do processo, pois permitem uma análise crítica e uma revisão das ações realizadas.

Conforme mencionado no capítulo anterior, a Descoberta de Conhecimento em Bases de Dados é caracterizada como um processo composto por três etapas operacionais básicas: Pré-processamento, Mineração de Dados e Pós-processamento.

A etapa de Pré-processamento comprehende todas as funções relacionadas à captação, à organização e ao tratamento dos dados. Essa etapa tem como objetivo a preparação dos dados para os algoritmos da etapa da Mineração de Dados. A seguir encontram-se comentadas as principais funções de pré-processamento dos dados. O Capítulo 3 apresenta um maior detalhamento destas funções.

- Seleção de Dados – Essa função, também denominada Redução de Dados, comprehende, em essência, a identificação de quais informações, dentre as bases de dados existentes, devem ser efetivamente consideradas durante o

processo de KDD. Por exemplo, o nome do cliente é uma informação totalmente irrelevante em uma aplicação de KDD cujo objetivo seja construir um modelo que preveja o comportamento de novos clientes quanto ao pagamento de futuros créditos a eles concedidos. Por outro lado, a data de nascimento de um cliente é fundamental em um modelo para estimar o valor de uma apólice de seguro de vida para este cliente. A seleção dos dados pode ter dois enfoques distintos: a escolha de atributos ou a escolha de registros que devem ser considerados no processo de KDD.

- Limpeza dos Dados – Abrange qualquer tratamento realizado sobre os dados selecionados de forma a assegurar a qualidade (completude, veracidade e integridade) dos fatos por eles representados. Informações ausentes, errôneas ou inconsistentes nas bases de dados devem ser corrigidas de forma a não comprometer a qualidade dos modelos de conhecimento a serem extraídos ao final do processo de KDD. Um exemplo simples de limpeza de dados seria a definição de um intervalo de possíveis valores para um determinado atributo. Caso surgisse qualquer valor diferente dos definidos no intervalo, o registro contendo esse dado poderia ser removido.
- Codificação dos Dados – Nessa função, os dados devem ser codificados para ficarem numa forma que possam ser usados como entrada dos algoritmos de Mineração de Dados. A codificação pode ser: Numérica – Categórica, que transforma valores reais em categorias ou intervalos; ou Categórica – Numérica, que representa numericamente valores de atributos categóricos.
- Enriquecimento dos Dados – A função de enriquecimento consiste em conseguir de alguma forma mais informação que possa ser agregada aos registros existentes, enriquecendo os dados, para que estes forneçam mais informações para o processo de descoberta de conhecimento. Podem ser realizadas pesquisas para complementação dos dados, consultas a bases de dados externas, entre outras técnicas.

Durante a etapa de Mineração de Dados é realizada a busca efetiva por conhecimentos úteis no contexto da aplicação de KDD. É a principal etapa do processo de KDD. Alguns autores se referem à Mineração de Dados e à Descoberta de Conhecimento em Bases de Dados como sinônimos. Envolve a aplicação de algoritmos sobre os dados em busca de conhecimento implícitos e úteis.

Na Mineração de Dados, são definidos as técnicas e os algoritmos a serem utilizados no problema em questão. Redes Neurais (Haykin, 1999), Algoritmos Genéticos (Davis, 1990), Modelos Estatísticos e Probabilísticos (Michie et al., 1994) são exemplos de técnicas que podem ser utilizadas na etapa de Mineração de Dados. A escolha da técnica depende, muitas vezes, do tipo de tarefa de KDD a ser realizada. A seguir algumas tarefas de KDD encontram-se comentadas. O Capítulo 4 apresenta um maior detalhamento destas tarefas.

- *Descoberta de Associação:* Abrange a busca por itens que freqüentemente ocorram de forma simultânea em transações do banco de dados. Um exemplo clássico e didático da aplicação desta tarefa é na área de marketing: durante um processo de descoberta de associações em sua vasta base de dados, uma grande rede de mercados norte-americana descobriu que um número razoável de compradores de fralda também comprava cerveja na véspera de finais de semana com jogos transmitidos pela televisão. Com uma análise mais detalhada sobre os dados, pode-se perceber que tais compradores eram, na realidade, homens que, ao comprarem fraldas para seus filhos, compravam também cerveja para consumo enquanto cuidavam das crianças e assistiam aos jogos na televisão durante o final de semana. Este exemplo ilustra a associação entre fraldas e cervejas. Esta empresa utilizou o novo conhecimento para aproximar as gôndolas de fraldas e cervejas na rede de mercados, incrementando assim a venda conjunta dos dois produtos. Algoritmos tais como o Apriori, GSP, DHP, entre outros, são exemplos de ferramentas que implementam a tarefa de descoberta de associações (Zaki, 2000).
- *Classificação:* Consiste em descobrir uma função que mapeie um conjunto de registros em um conjunto de rótulos categóricos predefinidos, denominados classes. Uma vez descoberta, tal função pode ser aplicada a novos registros de forma a prever a classe em que tais registros se enquadram. Como exemplo da tarefa de classificação, considere uma financeira que possua um histórico com os dados de seus clientes e o comportamento desses clientes em relação ao pagamento de empréstimos contraídos previamente. Considere dois tipos de clientes: clientes que pagaram em dia e clientes inadimplentes. São as classes do problema. Uma aplicação da tarefa de classificação consiste em descobrir uma função que mapeie corretamente os clientes, a partir de seus dados, em uma destas classes. Tal função, uma vez descoberta, pode ser utilizada para prever o comportamento de novos clientes que desejem contrair empréstimos junto à financeira. Essa função pode ser incorporada a um sistema de apoio à decisão que auxilie na filtragem e concessão de empréstimos somente a clientes classificados como bons pagadores. Redes Neurais, Algoritmos Genéticos, Lógica Indutiva são exemplos de tecnologias que podem ser aplicadas na tarefa de classificação (Michie et al., 1994).
- *Regressão:* Compreende a busca por uma função que mapeie os registros de um banco de dados em valores reais. Esta tarefa é similar à tarefa de classificação, sendo restrita apenas a atributos numéricos. Como exemplo de aplicações de regressão, pode-se citar: predição da soma da biomassa presente em uma floresta; estimativa da probabilidade de um paciente sobreviver, dado o resultado de um conjunto de diagnósticos de exames; predição do risco de determinados investimentos, definição do limite do

cartão de crédito para cada cliente em um banco; dentre outros. Estatística, Redes Neurais, dentre outras áreas, oferecem ferramentas para implementação da tarefa de regressão (Michie et al., 1994).

- *Clusterização:* Utilizada para separar os registros de uma base de dados em subconjuntos ou clusters, de tal forma que os elementos de um cluster compartilhem de propriedades comuns que os distingam de elementos em outros clusters. O objetivo nessa tarefa é maximizar similaridade intracluster e minimizar similaridade intercluster. Diferente da tarefa de classificação, que tem rótulos predefinidos, a clusterização precisa automaticamente identificar os grupos de dados aos quais o usuário deverá atribuir rótulos (Fayyad et al., 1996a). Por exemplo: uma empresa do ramo de telecomunicações pode realizar um processo de clusterização de sua base de clientes de forma obter grupos de clientes que compartilhem o mesmo perfil de compra de serviços. Na implementação desta tarefa podem ser utilizados algoritmos tais como: K-Means, K-Modes, K-Prototypes, K-Medoids, Kohonen, dentre outros.
- *Sumarização:* Essa tarefa, muito comum em KDD, consiste em procurar identificar e indicar características comuns entre conjuntos de dados (Weiss & Indurkha, 1998). Como exemplo considere um banco de dados com informações sobre clientes que assinam um determinado tipo de revista semanal. A tarefa de sumarização deve buscar por características que sejam comuns a boa parte dos clientes. Por exemplo: são assinantes da revista X, homens na faixa etária de 25 a 45 anos, com nível superior e que trabalham na área de finanças. Tal informação poderia ser utilizada pela equipe de marketing da revista para direcionar a oferta para novos assinantes. É muito comum aplicar a tarefa de sumarização a cada um dos agrupamentos obtidos pela tarefa de clusterização. Lógica Indutiva e Algoritmos Genéticos são alguns exemplos de tecnologias que podem ser aplicadas na implementação da tarefa de sumarização.
- *Detecção de Desvios:* Essa tarefa consiste em procurar identificar registros do banco de dados cujas características não atendam aos padrões considerados normais no contexto (Weiss & Indurkha, 1998). Tais registros são denominados “*outliers*”. Como exemplo considere um banco de dados com informações sobre compras de clientes no cartão de crédito. A tarefa de detecção de desvios deve buscar por compras cujas características diverjam do perfil normal de compra do dono do cartão. A Estatística fornece recursos para a implementação dessa tarefa.
- *Descoberta de Sequências:* É uma extensão da tarefa de descoberta de associações em que são buscados itens freqüentes considerando-se várias transações ocorridas ao longo de um período. Consideremos o exemplo das compras no supermercado. Se o banco de dados possui a identificação

do cliente associada a cada compra, a tarefa de descoberta de associação pode ser ampliada de forma a considerar a ordem em os produtos são comprados ao longo do tempo.

Várias dessas tarefas podem ser adaptadas, originando novas tarefas. Por exemplo, a tarefa de associação pode ser ajustada de forma a envolver a mineração de regras de associação generalizadas (Srikant et al., 1997). Por outro lado, composições entre diversas tarefas básicas (primárias) podem originar novas tarefas, mais complexas, úteis em vários processos de KDD (Goldschmidt, 2003). Por exemplo, é comum em aplicações de KDD a composição das tarefas de clusterização e classificação em uma nova tarefa que requer a definição de classes antes do início de um processo voltado à indução de um mecanismo classificador (Han, 1996; Weiss & Indurkhy, 1998).

A etapa de Pós-processamento abrange o tratamento do conhecimento obtido na Mineração de Dados. Tal tratamento, muitas vezes desnecessário, tem como objetivo facilitar a interpretação e a avaliação, pelo homem, da utilidade do conhecimento descoberto (Fayyad et al., 1996a). Entre as principais funções da etapa de pós-processamento estão: elaboração e organização, podendo incluir a simplificação, de gráficos, diagramas, ou relatórios demonstrativos; além da conversão da forma de representação do conhecimento obtido.

## **Macroobjetivos e Orientações do Processo de KDD**

Em geral, toda aplicação de KDD deve ser iniciada com um exame da base de dados. A partir desta análise e de entrevistas junto aos especialistas no domínio da aplicação, são definidos os objetivos a serem buscados ao longo do processo de KDD. Os objetivos da aplicação devem nortear todo o processo. Assim sendo, toda aplicação de KDD deve ser classificada em duas dimensões: quanto à orientação das ações a serem realizadas e quanto ao macroobjetivo pretendido.

A classificação de uma aplicação de KDD quanto à orientação das ações a serem realizadas pode ser:

- *Validação de hipóteses postuladas* – Nesse caso, o especialista da área em que se deseja realizar o processo de KDD apresenta alguma hipótese que deve ser comprovada ou refutada, mediante a análise dos dados.
- *Descoberta de conhecimento* – Nesse caso, enquadra-se a busca efetiva por conhecimentos a partir da abstração dos dados existentes.

Adicionalmente, a classificação de uma aplicação de KDD quanto ao macroobjetivo desejado pode ser:

- *Predição* – Nesse caso busca-se um modelo de conhecimento que permita, a partir de um histórico de casos anteriores, prever os valores de determinados atributos em novas situações.
- *Descrição* – Nesse caso busca-se por um modelo que descreva, de forma compreensível pelo homem, o conhecimento existente em um conjunto de dados.

É importante perceber que nos casos em que existir uma hipótese postulada, tal hipótese pode ou não ser utilizada em modelos de predição. Sua descrição, no entanto, já deve ter sido realizada por especialistas no domínio da aplicação de forma a permitir sua validação. Por outro lado, nos casos de orientações de ações para descoberta de conhecimento, ambas classificações quanto ao macroobjetivo podem ocorrer.

## Operações e Métodos de KDD

A expressão *Operação de KDD* se refere a qualquer função das etapas operacionais de KDD. Uma operação de KDD é, portanto, a especificação, no nível lógico, de uma função de KDD. Como operações de KDD, podem ser citadas (Bernstein et al., 2002; Han, 1996): Seleção de Dados, Limpeza de Dados, Codificação, Classificação, Descoberta de Associação, Regressão, Simplificação de Modelos de Conhecimento, dentre inúmeras outras.

As operações de KDD pertencentes à etapa de Mineração de Dados recebem a denominação especial de *Tarefas de KDD* ou ainda *Tarefas de Mineração de Dados*.

Uma operação de KDD pode ser primária ou composta. Uma operação primária de KDD é aquela que não pode ser desmembrada em outras operações de KDD. Por outro lado, uma operação de KDD composta pode ser desmembrada em duas ou mais operações primárias de KDD.

Na operação composta exemplificada a seguir, o símbolo “ $\rightarrow$ ” foi utilizado para indicar um encadeamento de operações primárias de KDD. A operação Clusterização  $\rightarrow$  Classificação denota uma operação composta pela seguinte sequência de operações primárias:

- 1) Clusterização do Conjunto de Dados.
- 2) Classificação dos registros com base nos clusters (agrupamentos) gerados.

No exemplo acima, a operação composta indica que primeiramente deve ser realizada a operação de clusterização do conjunto de dados de forma a separá-lo em  $K$  grupos de registros de dados similares. Uma vez concluída função de clusterização, os  $K$  grupos passam a representar classes. Cada registro do banco de dados passa a pertencer a um único grupo, ou ainda, a uma única classe. Com base nas novas informações, a segunda função pode ser realizada. Considera, para construção do classificador, as classes geradas pela operação de clusterização.

Um maior detalhamento das operações de KDD pode ser obtido nos Capítulos 3 (tarefas de Pré-processamento) e 4 (tarefas de Mineração de Dados).

Os *Métodos de KDD* são implementações específicas das operações de KDD. Um método de KDD corresponde a um algoritmo em particular. Por exemplo: o *Apriori* (Agrawal et al., 1993), a *Análise de Componentes Principais (PCA)* (Haykin, 1999) e o *Corte de Regras* (Han, 1996; Bernstein et al., 2002) são métodos que implementam, respectivamente, as operações de Descoberta de Associação, de Redução Vertical de Dados e de Simplificação de Modelos de Conhecimento. Maiores detalhes sobre estes e outros métodos de KDD podem ser obtidos nos Capítulos 3 (métodos de Pré-processamento) e 5 (métodos de Mineração de Dados).

## Técnicas de KDD

A expressão *Técnica de KDD* se refere a qualquer teoria que possa fundamentar a implementação de um método de KDD. Abaixo seguem algumas ilustrações deste conceito:

- A Teoria de Redes Neurais subsidia o desenvolvimento das *Máquinas de Suporte Vetorial (SVM – Support Vector Machines)* (Haykin, 1999), que, por sua vez, são aplicáveis em tarefas de Classificação.
- O Método *Rule Evolver* (Lopes et al., 1999) foi concebido a partir da Teoria de *Algoritmos Genéticos* (Davis, 1990) e pode ser aplicado em operações de Classificação e Sumarização (Fayyad et al., 1996a).
- O Método *NFHB-Class* (Gonçalves, 2001; Souza, 1999) baseia-se na combinação de princípios das Teorias da *Lógica Nebulosa* e das *Redes Neurais* e pode ser aplicado em operações de classificação.

Existem diversos tipos de técnicas e de algoritmos para Mineração de Dados. Podem ser subdivididos em: Técnicas Tradicionais, Técnicas Específicas e Técnicas Híbridas.

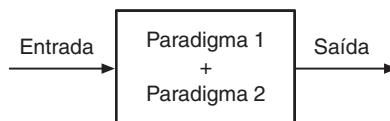
- a) *Técnicas Tradicionais*: São tecnologias que existem independente do contexto da Mineração de Dados. Em geral, produzem bons resultados também em aplicações de KDD. Como exemplo de tecnologias tradicionais podem ser citadas:
  - *Redes Neurais*: Uma Rede Neural Artificial (RNA) é uma técnica computacional que constrói um modelo matemático inspirado em um sistema neural biológico simplificado, com capacidade de aprendizado, generalização, associação e abstração. As RNAs tentam aprender padrões diretamente a partir dos dados através de um processo de repetidas apresentações dos dados à rede, ou seja, por experiência. Dessa forma, uma RNA

procura por relacionamentos, constrói modelos automaticamente, e os corrige de modo a diminuir seu próprio erro. Para uma introdução sobre Redes Neurais Artificiais, o leitor pode recorrer ao Anexo II.

- *Lógica Nebulosa (Fuzzy Logic)*: É uma técnica que permite construir sistemas que lidem com informações imprecisas ou subjetivas. Diferente da Lógica Clássica, a Lógica Nebulosa oferece flexibilidade na definição e na avaliação de conceitos. Para uma introdução sobre Lógica Nebulosa, o leitor pode recorrer ao Anexo III.
  - *Algoritmos Genéticos (AG)*: São modelos de otimização, inspirados na evolução natural e na genética, aplicados a problemas complexos de otimização. Problemas de otimização tipicamente envolvem três componentes: variáveis restrições e objetivos. As variáveis descrevem os vários aspectos do problema. As restrições indicam os valores que as variáveis podem ter. Por último, as funções objetivo são utilizadas para avaliar a solução. As variáveis, as restrições e as funções objetivo, descritas em um problema de otimização definem a geografia básica do espaço de busca, e determinam que técnicas podem ser usadas. Técnicas baseadas em modelos heurísticos como AG não podem garantir a solução ótima, porém podem conseguir soluções próximas, ou aceitáveis (subótimas). Além disso, AGs são indicados para problemas complexos com muitas variáveis e restrições ou com grandes espaços de busca. Para uma introdução sobre Algoritmos Genéticos, o leitor pode recorrer ao Anexo IV.
  - *Estatística*: Fornece diversos modelos e técnicas tradicionais para análise e interpretação de dados. Por exemplo: Redes Bayesianas, Análise Discriminante, Análise Exploratória de dados, dentre outros (Michie et al., 1994; Fayyad et al., 1996a; Engels & Theusinger, 1998).
- b) *Técnicas Específicas*: São técnicas desenvolvidas especificamente para aplicação em tarefas de KDD. Como exemplo de técnica específica, podemos citar o algoritmo Apriori, desenvolvido especificamente para a tarefa de Descoberta de Associação. Diversos algoritmos tais como DHP, Partition, Par-MaxEclat, dentre outros, foram originados a partir do Apriori. Um maior detalhamento sobre este algoritmo encontra-se no Capítulo 5.
- c) *Técnicas Híbridas*: Convém mencionar que técnicas podem ser combinadas de forma a gerar os chamados sistemas híbridos. Há várias formas de se definir sistemas híbridos. De uma forma simples, sistemas híbridos são aqueles que utilizam mais de uma técnica para a solução de um problema de modelagem. A grande vantagem desse tipo de sistema deve-se ao sinergismo obtido pela combinação de duas ou mais técnicas de modelagem. Este sinergismo reflete-se na obtenção de um sistema mais poderoso (em termos de poder de

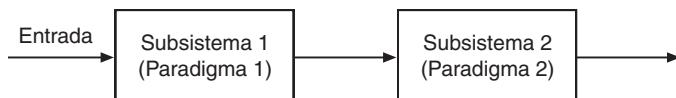
interpretação, de aprendizado, de estimativa de parâmetros, de generalização, dentre outros) e com menos deficiências. Existem três formas básicas de se associarem duas técnicas para a construção de sistemas híbridos (Souza, 1999).

- Híbrido Seqüencial – Neste modelo, um sistema com paradigma 1 atua como entrada de outro sistema com paradigma 2, como ilustrado na Figura 2.1. Um exemplo dessa combinação seria o caso de se usar um pré-processador fuzzy ou estatístico acionando uma rede neural. Essa é a forma mais fraca de hibridização, sendo que alguns pesquisadores nem a consideram como, efetivamente, um sistema híbrido.



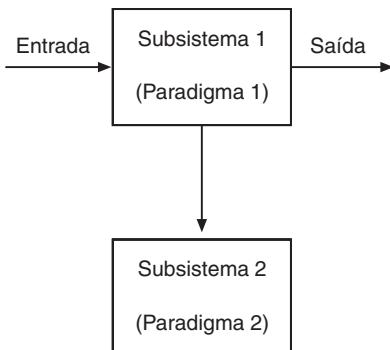
**Figura 2.1.** *Modelo Híbrido Seqüencial.*

- Híbrido Auxiliar – Nesse modelo, um subsistema constituído pela técnica do paradigma 2 é chamado pelo subsistema implementado pelo paradigma 1, retornando ou realizando alguma tarefa auxiliar. A Figura 2.2 ilustra este modelo de hibridização. Como exemplo desta combinação de paradigmas pode-se citar um sistema em que uma rede neural invoca um algoritmo genético para otimização de seus pesos, ou de sua estrutura. Nesse caso, tem-se um maior grau de hibridização em comparação com o híbrido seqüencial.



**Figura 2.2.** *Modelo Híbrido Auxiliar.*

- Híbrido Incorporado – Nessa forma de combinação entre os dois paradigmas não há uma separação nítida entre os dois subsistemas. Pode-se dizer que o primeiro paradigma contém o segundo e vice-versa. A Figura 2.3 traz uma representação deste modelo. Um exemplo deste modelo é o caso de um sistema neuro-fuzzy híbrido em que um sistema de inferência fuzzy é implementado segundo a estrutura de uma rede neural. Aqui a hibridização é a maior possível. Há sistemas em que a hibridização é de um grau tão elevado que não é possível a separação dos dois paradigmas.



**Figura 2.3.** *Modelo Híbrido Incorporado.*

## Ferramentas de KDD

De uma forma geral, a complexidade inerente ao processo de KDD decorre de diversos fatores que podem ser subdivididos em dois conjuntos: fatores operacionais e fatores de controle (Fayyad et al., 1996b; Hellerstein et al., 1999).

A necessidade de manipulação de grandes e heterogêneos volumes de dados, o tratamento de resultados representados em diferentes formatos e a dificuldade de integração de diversos algoritmos específicos são alguns exemplos de fatores operacionais. Atualmente, encontram-se comercialmente disponíveis diversas ferramentas que implementam ambientes integrados para facilitar a execução das etapas operacionais de KDD, minimizando as dificuldades decorrentes dos fatores operacionais. Estas ferramentas são referenciadas neste texto como ferramentas operacionais de KDD. SAS Enterprise Miner, PolyAnalyst, Clementine, SPSS, Intelligent Miner, WizRule e WizWhy são algumas dessas ferramentas (Goldschmidt et al., 2002a). Em geral essas ferramentas reúnem diversos métodos de Mineração de Dados podendo ser aplicadas em várias tarefas. O Capítulo 6 apresenta de forma resumida um estudo comparativo entre algumas das principais ferramentas operacionais de KDD existentes.

Por outro lado, os fatores de controle, mais complexos, envolvem considerações sobre como conduzir processos de KDD. Entre tais fatores, podem ser citados:

- A dificuldade na formulação precisa dos objetivos a serem alcançados em um processo de KDD. Não raro nem mesmo os especialistas no domínio da aplicação de KDD apresentam de forma clara suas expectativas quanto aos resultados do processo (Engels, 1996; Verdenius & Engels, 1997; Wirth et al., 1997).
- A dificuldade na escolha de um algoritmo de mineração de dados com potencial para geração de resultados satisfatórios. Tal dificuldade é in-

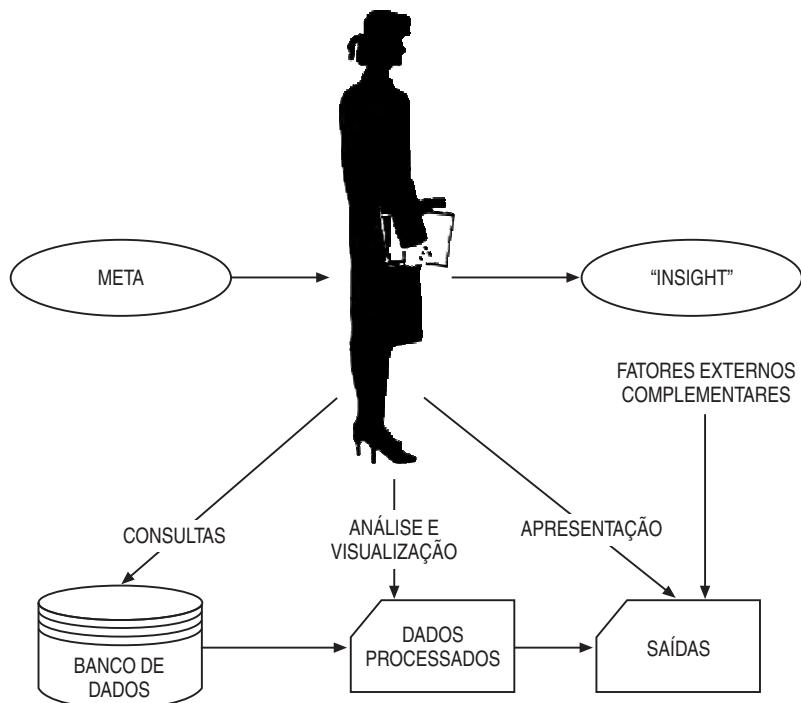
tensificada na medida em que surjam novos algoritmos com o mesmo propósito, aumentando a diversidade de alternativas. Em geral, a escolha dos algoritmos se restringe às opções conhecidas pelo analista de KDD, deixando muitas vezes de considerar outras alternativas promissoras (Brazdil et al., 2003).

- A dificuldade na escolha das alternativas de pré-processamento dos dados. Neste caso, a diversidade de algoritmos de pré-processamento é agravada não só pelo surgimento de novos algoritmos, mas pelas possibilidades de combinação entre eles. De forma análoga ao item anterior, alternativas de pré-processamento potencialmente adequadas podem sequer ser consideradas durante a escolha (Bernstein et al., 2002).
- A dificuldade freqüente na escolha cuidadosa da parametrização adequada a diversos algoritmos diante de cada nova situação. Esse problema pode facilmente aumentar o número de iterações do processo, na medida em que diversos algoritmos com diferentes parametrizações sejam experimentados na busca por resultados promissores (Han, 1996; Weiss & Indukhya, 1998; Zaki, 2000).
- Uma recorrente ruptura de concentração do analista de KDD causada muitas vezes pelos longos tempos de experimentação de inúmeras alternativas na busca por melhores resultados (Hellerstein et al., 1999).
- A incapacidade humana de memorização de resultados e alternativas processadas na medida em que o tempo passa e o número de experimentos realizados aumenta. Tal fato compromete ainda a desejável comparação entre alternativas e resultados necessária à tomada de decisão quanto a novas alternativas a serem avaliadas (Hellerstein et al., 1999).
- De uma maneira geral, a dificuldade em perceber e interpretar adequadamente inúmeros fatos observáveis ao longo de processos de KDD e a complexidade em conjugar dinamicamente tais interpretações de forma a decidir que ações devem ser realizadas em cada caso.

Pelo exposto acima, pode-se claramente perceber a importância de ferramentas que são capazes de auxiliar o homem no controle de aplicações de KDD. Metal, IKDD, IDEA, e Consultant são alguns exemplos de ferramentas de apoio ao controle do processo de KDD. Estas e algumas outras ferramentas de controle encontram-se resumidas e comparadas no Capítulo 6.

## O Papel do Usuário no Processo de KDD

O papel do usuário na condução dos processos de KDD é de grande importância e essa participação humana deve ser especializada. Os diversos fatores de controle enumerados na seção anterior demonstram tal importância.



**Figura 2.4.** Ser Humano como Elemento Central do Processo de KDD.

De uma forma geral, como ilustrado na Figura 2.4, cabe ao analista humano a árdua tarefa de orientar a execução do processo de KDD. Para tanto, diante de cada cenário, o homem utiliza sua experiência anterior, seus conhecimentos e sua intuição para interpretar e combinar subjetivamente os fatos de forma a decidir qual a estratégia a ser adotada (Fayyad et al., 1996b; Wirth et al., 1997). Os cenários envolvem, em geral, inúmeros aspectos tais como: fatos observados cuja origem e os níveis de detalhe são os mais diversos e difusos, resultados intermediários obtidos com várias ferramentas, opiniões de especialistas no contexto da aplicação, e conhecimentos previamente existentes (Buchanan, 2000; Bernstein et al., 2002).

No entanto, a formação de especialistas em KDD constitui-se em uma tarefa árdua, longa e exaustiva, pois requer não somente uma fundamentação teórica sobre a área, mas também a participação destes em inúmeras experiências práticas reais.

# Etapas do Processo de KDD

## Considerações Iniciais

Este capítulo tem como objetivo descrever as etapas operacionais do processo de KDD. Em cada etapa são apresentadas algumas das principais e mais utilizadas funções em aplicações desta natureza.

Com o propósito de ilustrar a aplicação e a integração das funções de KDD descritas neste capítulo, utilizamos o mesmo conjunto fictício de dados sobre clientes de uma financeira como exemplo. As Tabelas 3.1 e 3.2 apresentam a estrutura e o conteúdo deste conjunto, respectivamente.

**Tabela 3.1 Estrutura do conjunto de dados de clientes**

Atributo	Tipo de Dado	Descrição de Domínio
CPF	Char(11)	Número do Cadastro de Pessoa Física do Cliente. Formato: XXX.XXX.XXX-XX
Nome	VarChar(40)	Nome completo do Cliente
Sexo	Char(1)	M – Masculino F – Feminino
Data_Nasc	Date	Data de nascimento do Cliente. Formato: DD/MM/AAAA
Est_Civil	Char(1)	Estado civil do Cliente: C – Casado S – Solteiro V – Viúvo D – Divorciado O – Outro
Num_Dep	Integer	Quantidade de pessoas que dependem financeiramente do Cliente.
Renda	Real	Total dos rendimentos mensais do Cliente.
Despesa	Real	Total das despesas mensais do Cliente.

**Tabela 3.1 Continuação**

Atributo	Tipo de Dado	Descrição de Domínio
Tp_Res	Char(1)	Tipo da residência do Cliente: P – Própria A – Alugada E – Parentes F – Funcional O – Outro
Bairro_Res	VarChar(50)	Nome do Bairro em que o Cliente reside.
Result	Char(1)	Resultado do crédito concedido anteriormente ao cliente: A – Adimplente I – Inadimplente

**Tabela 3.2 Exemplos de instâncias de clientes**

Cpf	Nome	Sexo	Data_Nasc	Est_Civil	Num_Dep	Renda	Despesa	Tp_Res	Bairro_Res	Result
99999999999	José	M	05/05/1989	C	1	Null	1000	X	Centro	A
11111111111	Maria	F	10/07/1985	S	5	3000	2000	P	Urca	A
33333333333	Ana	F	15/08/1981	S	4	5000	3000	P	Leblon	A
55555555555	Pedro	M	10/09/1975	C	3	1500	1500	A	Centro	I
22222222222	Mario	M	20/02/1960	C	5	2500	1500	P	Leblon	A
00000000000	Manoel	M	13/10/1954	S	1	Null	1000	E	Barra	I
88888888888	Liza	F	27/04/1980	S	2	4500	3000	F	Urca	A
77777777777	Marisa	F	14/07/1980	C	3	1000	500	A	Recreio	I
66666666666	Carlos	M	06/11/1963	V	7	4300	4000	F	Leblon	A
44444444444	Paula	F	15/12/1982	C	3	1400	1000	A	Centro	I

É importante perceber ainda que as variáveis do problema, também denominadas atributos, podem ser classificadas sob dois aspectos distintos: quanto à representação de seus valores (tipo de dado) e quanto à natureza da informação (tipo de variável). Os tipos de dados indicam a forma em que eles estão armazenados. Os tipos de variáveis expressam a natureza com que a informação deve ser interpretada. Assim sendo, as variáveis podem ser classificadas em (Pyle, 1999):

- *Nominais ou Categóricas* – São variáveis utilizadas para nomear ou atribuir rótulos a objetos. Podem assumir valores pertencentes a um conjunto finito e pequeno de estados possíveis. Como exemplo pode-se citar o estado civil de uma pessoa: solteiro, casado, viúvo, divorciado etc. Nas variáveis nominais não há um ordenamento de seus valores. Não se pode dizer

que “solteiro” é menor que “viúvo”, por exemplo. Os valores de variáveis nominais podem ser representados por tipos de dados alfanuméricos.

- *Discretas* – Assemelham-se às variáveis nominais, mas os valores (estados) que elas podem assumir possuem um ordenamento, e este possui algum significado. O dia da semana é um bom exemplo deste tipo de variável, onde a “segunda-feira” vem após o “domingo” e antes da “terça-feira”, e assim sucessivamente. Podem ser intervalos em um espectro contínuo de valores. Por exemplo: faixa de renda e faixa etária, entre outros. De forma análoga às variáveis nominais, os valores de variáveis discretas podem ser representados por tipos de dados alfanuméricos.
- *Contínuas* – São variáveis quantitativas cujos valores possuem uma relação de ordem entre eles. O conjunto de valores de uma variável contínua pode ser finito ou infinito. Renda e Idade são exemplos de variáveis contínuas. Normalmente os valores das variáveis contínuas são representados por um tipo de dado numérico.

A Tabela 3.3 apresenta o enquadramento das variáveis de nosso exemplo conforme a classificação descrita acima.

**Tabela 3.3** Classificação das variáveis do conjunto de dados de clientes

Atributo	Tipo de Variável	Observações
CPF	Nominal	Não possui relação de ordem entre seus valores.
Nome	Nominal	Não possui relação de ordem entre seus valores.
Sexo	Nominal	Não possui relação de ordem entre seus valores.
Data_Nasc	Discreto	Uma variável do tipo data pode ser representada pelo número de dias transcorridos a partir de um determinado marco temporal. Envolve, portanto, uma relação de ordem entre seus valores. O conjunto de valores possíveis é finito.
Est_Civil	Nominal	Não possui relação de ordem entre seus valores.
Num_Dep	Discreto	Envolve uma relação de ordem entre os seus valores. O conjunto de valores desta variável é finito.
Renda	Contínuo	O conjunto de valores deste atributo, embora seja limitado, é teoricamente infinito, uma vez que sejam admitidas variações no número de casas decimais.
Despesa	Contínuo	Idêntico à variável Renda.
Tp_Res	Nominal	Não possui relação de ordem entre seus valores.
Bairro_Res	Nominal	Não possui relação de ordem entre seus valores.
Result	Nominal	Não possui relação de ordem entre seus valores.

## Pré-processamento

Conforme apresentado anteriormente, a etapa de pré-processamento comprehende as funções relacionadas à captação, à organização, ao tratamento e à preparação dos dados para a etapa da Mineração de Dados. Essa etapa possui fundamental relevância no processo de descoberta de conhecimento. Compreende desde a correção de dados errados até o ajuste da formatação dos dados para os algoritmos de Mineração de Dados a serem utilizados.

Para cada função de pré-processamento apresentada neste capítulo, foram indicados e discutidos alguns métodos. Para fins ilustrativos, os resultados da aplicação dos métodos no conjunto de clientes da Tabela 3.2 também encontram-se indicados e comentados.

Ainda ao longo desta seção serão apresentadas algumas heurísticas voltadas a orientações sobre quando determinadas operações devem ser utilizadas. Em Inteligência Computacional, denomina-se heurística a todo conhecimento que pode ser utilizado na simplificação de um problema.

## Seleção de Dados

Esta função comprehende, em essência, a identificação de quais informações, dentre as bases de dados existentes, devem ser efetivamente consideradas durante o processo de KDD.

Em geral, os dados encontram-se organizados em bases de dados transacionais que sofrem constantes atualizações ao longo do tempo. Assim sendo, recomenda-se que seja sempre feita uma cópia dos dados a fim de que o processo de KDD não interfira nas rotinas operacionais eventualmente relacionadas à base de dados.

Nos casos em que já existe uma estrutura de DataWarehouse, deve-se verificar a possibilidade de que esta seja utilizada no processo de KDD. Nos demais casos, é comum a congregação dos dados em uma única tabela. Tal fato justifica-se porque a maioria dos métodos de Mineração de Dados pressupõe que os dados estejam organizados em uma única, possivelmente muito grande, estrutura tabular bidimensional. Percebe-se, portanto, que o processo de KDD pode ocorrer independente da disponibilidade ou não de Data Warehouses.

A junção dos dados em uma única tabela pode ocorrer de duas formas:

- a) Junção Direta – Todos os atributos e registros da base de dados transacional são incluídos na nova tabela, sem uma análise crítica quanto a que variáveis e que casos podem realmente contribuir para o processo de KDD.
- b) Junção Orientada – O especialista no domínio da aplicação, em parceria com o especialista em KDD, escolhe os atributos e os registros com algum potencial para influir no processo de KDD. Recomenda-se que sejam desconsiderados somente atributos e registros sobre os quais se tenha uma visão clara quanto à inexistência de potencial de contribuição para o processo de KDD.

Convém mencionar que há situações em que, no início do processo, o especialista em KDD já recebe o conjunto de dados a ser analisado. Em nosso exemplo, estamos partindo deste tipo de situação em que a tabela de clientes foi fornecida sem uma análise da origem dos dados.

Assim, considerando que os dados estejam reunidos em uma mesma estrutura, a função de seleção dos dados pode ter dois enfoques distintos: a escolha de atributos ou a escolha de registros a serem considerados no processo de KDD.

## **Redução de Dados Horizontal**

A seleção por redução de dados horizontal é caracterizada pela escolha de casos. Entre as operações de redução de dados horizontal podem ser citadas: amostragem aleatória, eliminação direta de casos, segmentação do banco de dados e agregação de informações.

### *Segmentação do Banco de Dados*

Nesta operação, deve-se escolher um ou mais atributos para nortear o processo de segmentação. Suponhamos que em nosso exemplo desejamos apenas analisar os clientes que moram em residência própria. Tal operação poderia ser implementada por uma instrução de seleção em SQL do tipo:

```
SELECT *
FROM CLIENTE
WHERE TP_RES = "P";
```

O conjunto de dados resultante desta consulta se tornaria o conjunto a ser efetivamente considerado neste ponto em diante no processo de KDD.

### *Eliminação Direta de Casos*

Esta operação pode ser interpretada como uma variação da anterior (operação de complemento), e nela são especificados os casos a serem eliminados e não os casos que devem permanecer na análise. No mesmo exemplo enunciado acima, tal operação poderia ser implementada por uma instrução de exclusão em SQL do tipo:

```
DELETE FROM CLIENTE
WHERE TP_RES < > "P";
```

### *Amostragem Aleatória*

A operação de amostragem aleatória consiste em sortear da base de dados um número preestabelecido de registros de forma que o conjunto resultante possua

menos registros do que o conjunto original. Existem várias estratégias de amostragem aleatória. Para ilustrarmos algumas destas estratégias, suponhamos a existência de um grande conjunto de dados, contendo  $N$  tuplas, e que  $n$  seja o número de amostras desejadas ( $n < N$ ).

- *Amostragem Aleatória Simples sem Reposição:* Neste caso, todas as tuplas possuem a mesma probabilidade de seleção:  $1/N$ . Cada tupla selecionada é excluída do conjunto de dados original durante o processo de forma a evitar uma nova seleção.
- *Amostragem Aleatória Simples com Reposição:* Também neste caso, todas as tuplas possuem a mesma probabilidade de seleção:  $1/N$ . No entanto, cada tupla selecionada é mantida no conjunto de dados original durante o processo de forma que a mesma tupla possa ser selecionada novamente.
- *Amostragem de Clusters (Agrupamentos):* Neste caso, as tuplas devem estar agrupadas em  $M$  *clusters* de tal forma que possa ser realizada uma amostragem aleatória dentre os *clusters*. Por exemplo, as tuplas de um banco de dados normalmente são recuperadas uma página por vez. Cada página pode ser considerada um *cluster* de tal forma que a Amostragem de *Clusters* possa ser aplicada.
- *Amostragem Estratificada:* Neste caso, o conjunto de dados deve ser dividido em grupos disjuntos. A amostragem estratificada consiste em selecionar aleatoriamente um subconjunto de amostras de cada grupo. Essa abordagem auxilia na obtenção de amostras representativas, especialmente em situações que apresentam dados tendenciosos. Para ilustrar melhor a idéia, imaginemos os clientes de nosso exemplo separados em grupos por faixa etária. Aplicando a amostragem estratificada, asseguramos que mesmo os clientes da faixa etária com menor número de elementos sejam representados na amostra final.

### **Agregação de Informações**

Esta operação consiste em reunir os dados de forma a reduzir o conjunto de dados original. Na agregação de informações, dados em um nível maior de detalhamento são consolidados em novas informações com menor detalhe. Por exemplo: somar os valores de todas as compras de cada cliente, obtendo o total de despesas por ele realizadas durante um determinado período.

### **Redução de Dados Vertical**

A seleção de dados por redução de dados vertical é uma operação de pré-processamento muito importante no processo de KDD. Formalmente, sendo  $S$  um conjunto de dados com atributos  $A_1, A_2, A_3, \dots, A_n$ , o problema da redução

de dados vertical consiste em identificar qual das  $2^n$  combinações entre tais atributos deve ser considerada no processo de descoberta de conhecimento. Em outras palavras, a redução de dados vertical, também denominada redução de dimensão, é implementada pela eliminação ou pela substituição dos atributos de um conjunto de dados. Tem como objetivo procurar encontrar um conjunto mínimo de atributos de tal forma que a informação original seja preservada. Obviamente, quanto maior o valor de  $n$ , maior o desafio na escolha dos atributos: o número de possibilidades de subconjuntos de atributos cresce exponencialmente na medida em que  $n$  aumenta.

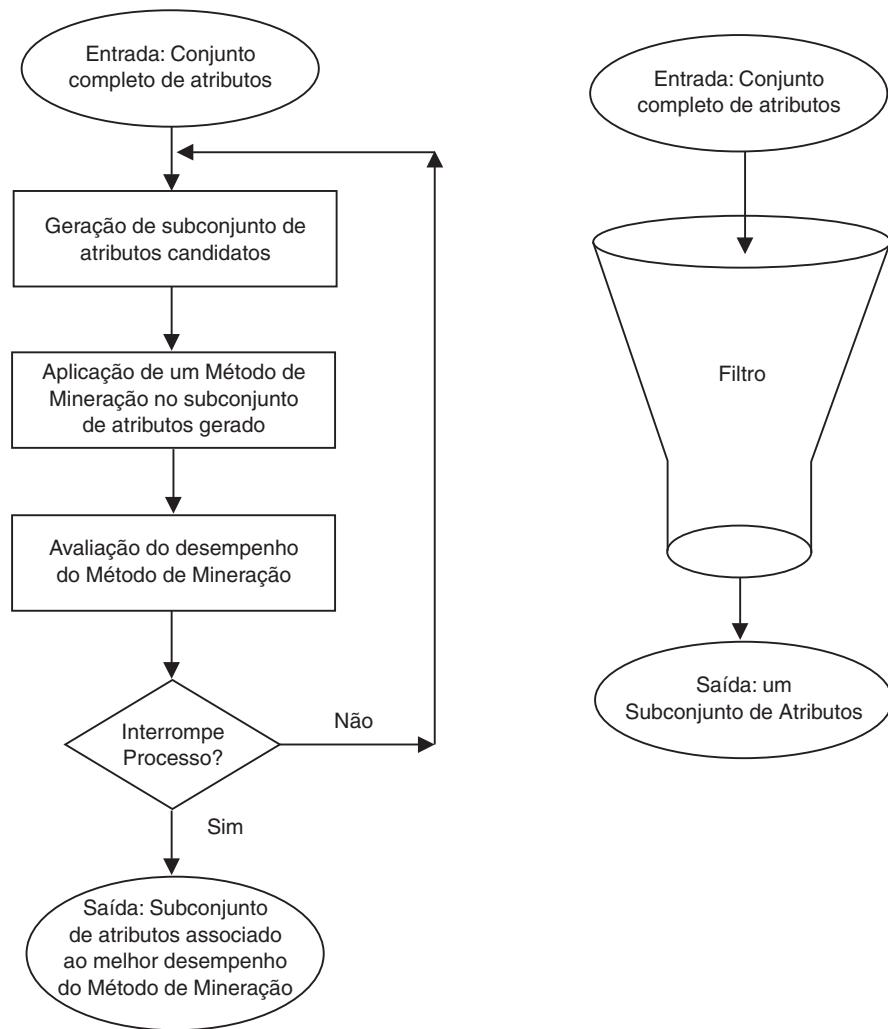
Entre as principais motivações para a aplicação da redução de dados vertical podem ser citadas: (a) um conjunto de atributos bem selecionado pode conduzir a modelos de conhecimento mais concisos e com maior precisão; (b) se o método de seleção dos atributos for rápido, o tempo de processamento necessário para utilizá-lo e, em seguida, aplicar o algoritmo de mineração de dados em um subconjunto dos atributos, pode ser inferior ao tempo de processamento para aplicar o algoritmo de mineração sobre todo o conjunto de atributos; (c) a eliminação de um atributo é muito mais significativa em termos de redução do tamanho de um conjunto de dados do que a exclusão de um registro.

Conforme exposto na Figura 3.1, existem duas abordagens para a redução de dados vertical, usualmente utilizadas em problemas de Classificação (Freitas, 2002):

- Abordagem Independente de Modelo (*Filter*) – Nessa abordagem, a seleção de atributos é realizada sem considerar o algoritmo de mineração de dados que será aplicado aos atributos selecionados.
- Abordagem Dependente de Modelo (*Wrapper*) – Essa abordagem consiste em experimentar o algoritmo de mineração de dados para cada conjunto de atributos e avaliar os resultados obtidos.

Em quaisquer das abordagens mencionadas acima, três estratégias clássicas e simples para escolha do conjunto de atributos podem ser utilizadas (Han & Kember, 1999):

- Seleção Seqüencial para a Frente (*Forward Selection*) – Essa seleção começa com o subconjunto de atributos candidatos vazio. O processo é iterativo. Cada atributo por vez é adicionado ao subconjunto de atributos candidatos, que é avaliado segundo alguma medida de qualidade. Ao final de cada iteração, é incluído no subconjunto de atributos candidatos, aquele atributo que tenha maximizado a medida de qualidade considerada.
- Seleção Seqüencial para Trás (*Backward Selection*) – É o processo contrário da Seleção Seqüencial para a frente: o subconjunto de atributos candidatos começa completo, com todos os atributos do problema. Então, cada



**Figura 3.1.** Abordagens para Seleção de Atributos em problemas de Classificação.

atributo é retirado do subconjunto, que é avaliado segundo alguma medida de qualidade. Ao final de cada iteração, é excluído do subconjunto de atributos candidatos, aquele atributo que tenha minimizado a medida de qualidade considerada.

- Combinação das Estratégias Anteriores – A seleção para a frente e a seleção para trás são combinadas de tal forma que, a cada passo do algoritmo, o algoritmo seleciona o melhor atributo (incluindo-o no subconjunto de atributos candidatos) e remove o pior atributo dentre os remanescentes do conjunto de atributos.

Cabe ressaltar que existem estratégias mais interessantes para a escolha do conjunto de atributos a ser utilizado (Han & Kember, 1999; Freitas, 2002). Algoritmos Genéticos, por exemplo, podem ser utilizados no processo de otimização do conjunto de atributos. Algoritmos para indução de Árvores de Decisão, tais como ID3 e C4.5, também podem ser aplicados para selecionar atributos em problemas de classificação. Uma árvore de decisão é construída a partir dos dados, de tal forma que todos os atributos que não apareçam na árvore são considerados irrelevantes para o problema.

Um exemplo de medida de qualidade muito utilizada pelos métodos de redução de dados vertical em problemas de classificação é a taxa de inconsistência gerada a partir do agrupamento dos registros que possuem os mesmos valores com relação aos atributos em análise (projeção sobre os atributos). Por exemplo, consideremos os atributos sexo e estado civil da relação de clientes da Tabela 3.2. Ao agruparmos os valores destes atributos, obtemos o resultado mostrado na Tabela 3.4. A primeira e a segunda linhas desta tabela ilustram a ocorrência de inconsistências: Para os mesmos valores de sexo e estado civil há em dois casos enquadramento na classe A e um caso na classe I. A terceira linha representa as 3 únicas mulheres solteiras, todas pertencentes à classe A, com nenhuma inconsistência. De forma, análoga, as três últimas linhas da tabela também não apresentam inconsistência.

**Tabela 3.4 Resultado do agrupamento dos atributos sexo e estado civil da Tabela 3.3**

Sexo	Est_Civil	Result	Count(*)
M	C	A	2
M	C	I	1
F	S	A	3
M	S	I	1
F	C	I	2
M	V	A	1

A seguir encontram-se citados e comentados alguns dos principais métodos de redução de dados vertical.

### **Eliminação Direta de Atributos**

Essa operação se refere à eliminação de atributos cujo conteúdo não seja relevante ao processo de KDD. A sua utilização depende do conhecimento prévio existente acerca do problema. No nosso exemplo, as informações de CPF e nome apenas particularizam os objetos, não contribuindo para uma análise de perfis de cliente. Essa operação pode ser implementada por uma instrução SQL do seguinte tipo:

```

SELECT
    SEXO,
    DATA_NASC,
    EST_CIVIL,
    NUM_DEP,
    RENDA,
    DESPESA,
    TP_RES,
    BAIRRO_RES,
    RESULT
FROM CLIENTE;

```

Duas heurísticas podem ser utilizadas para indicar que essa operação deve ser utilizada:

- Eliminar todos os atributos que apresentem valores constantes em todos os registros da base de dados. Atributos nesta situação não contribuem para distinguir os registros uns dos outros, sendo, portanto, dispensáveis do processo de KDD.
- Eliminar os atributos que sejam identificadores na base de dados em análise. Tal fato, ilustrado em nosso exemplo, justifica-se uma vez que essas informações são usadas para identificar unicamente cada registro da base de dados.

### *Análise de Componentes Principais*

A seguir encontra-se uma introdução intuitiva à Análise de Componentes Principais (PCA – Principal Component Analysis). Trata-se de uma técnica de baixo custo computacional, que pode ser aplicada a qualquer conjunto de dados numéricos com mais de duas dimensões. Maiores detalhes teóricos podem ser obtidos em (Haykin, 1999).

Suponhamos que o conjunto de dados em análise possua  $N$  tuplas ou vetores de dados, com  $k$  dimensões (atributos). A Análise de Componentes Principais, também chamada Karhunen-Loeve ou Método  $K-L$ , busca por  $c$  vetores orthonormais de dimensão  $k$  que possam ser utilizados para representar melhor os dados (sendo  $c \leq k$ ). Os dados originais são então projetados em um espaço de menor dimensão, resultado na redução dos dados. PCA combina, portanto, a essência dos atributos originais, criando um novo e menor conjunto de variáveis para representar os dados. Assim sendo, os dados iniciais são, então, projetados sobre as novas variáveis. O procedimento básico ocorre conforme descrito a seguir:

- Os dados de entrada são normalizados, de tal forma que os valores de todos os atributos pertençam a uma mesma faixa de valores (mais detalhes sobre normalização podem ser obtidos na seção “Normalização de Dados”, mais

adiante neste capítulo). Este passo ajuda a garantir que atributos que possuam domínios mais amplos não sejam priorizados em relação aos atributos com domínios menores.

- b) PCA computa  $c$  vetores ortonormais que forneçam uma base para os dados de entrada normalizados. Esses são vetores unitários (norma igual a 1) cuja direção é perpendicular em relação aos demais. Tais vetores são denominados componentes principais. Os dados de entrada podem ser escritos como combinação linear dos componentes principais.
- c) Os componentes principais são ordenados em ordem decrescente de variância. Os componentes principais servem, em essência, como um novo conjunto de eixos para os dados, fornecendo informações importantes sobre a variância dos dados.
- d) Uma vez que os componentes são ordenados de forma decrescente em relação à variância, o tamanho do conjunto de dados pode ser reduzido a partir da eliminação dos componentes mais fracos, ou seja, aqueles com menor variância. Usando os componentes principais mais fortes em termos de variância, pode-se reconstruir uma boa aproximação dos dados originais.

## Redução de Valores

A operação de redução de valores é uma alternativa interessante à opção de corte de atributos oferecida pela redução de dados vertical. Essa operação consiste em reduzir o número de valores distintos em determinados atributos, o que pode proporcionar um melhor desempenho a diversos algoritmos de Mineração de Dados, sobretudo àqueles que envolvem manipulações simbólicas e comparações lógicas dos dados. Com menos valores, menos comparações são feitas, reduzindo o tempo de processamento de tais algoritmos. Existem métodos de redução de valores contínuos e métodos de redução de valores nominais.

### *Redução de Valores Nominais*

Essa operação é aplicável somente a variáveis nominais. Variáveis nominais possuem um número finito (possivelmente grande) de valores distintos, sem ordenação entre esses valores. A seguir são apresentados dois métodos para geração de níveis de abstração (hierarquia) entre variáveis e valores nominais (conceitos).

- Identificação de Hierarquia entre Atributos – Neste caso, o especialista no domínio da aplicação deve apresentar a hierarquia existente entre os atributos. Por exemplo, considere um conjunto de dados que contenha informações sobre o endereço dos clientes: logradouro, bairro, cidade e unidade da federação. A hierarquia pode ser definida pelo especialista por meio de uma ordenação total entre esses atributos no esquema do conjunto de

dados: logradouro ⊂ bairro ⊂ cidade ⊂ unidade da federação. A partir desta especificação, pode-se estabelecer um nível de corte do detalhamento da informação. Supondo que apenas as informações sobre a cidade fossem de interesse na aplicação, as informações com níveis hierárquicos inferiores à cidade poderiam ser desconsideradas.

- Identificação de Hierarquia entre Valores – Neste outro caso, o especialista no domínio da aplicação deve apresentar possíveis generalizações para os valores de cada atributo. Por exemplo, considere um conjunto de dados que contenha informações sobre os produtos vendidos a cada cliente: tênis, sapato, sandália, bermuda, calça, camisa, paletó. Hierarquias podem ser definidas pelo especialista para indicar generalizações de conceito com relação aos valores existentes: tênis ⊂ calçado, sapato ⊂ calçado, sandália ⊂ calçado, bermuda ⊂ roupa, calça ⊂ roupa, camisa ⊂ roupa e paletó ⊂ roupa. As três primeiras generalizações indicam que tênis, sapato e sandália são casos particulares de calçado. Analogamente, as quatro últimas generalizações indicam que bermuda, calça, camisa e paletó são casos particulares do conceito roupa. A partir desta especificação, os valores originais podem ser substituídos pelas respectivas generalizações, reduzindo um domínio de 7 valores distintos para apenas 2.

### *Redução de Valores Contínuos (ou Discretos)*

Esse tipo operação pressupõe aplicação somente a variáveis contínuas ou discretas. Variáveis contínuas e discretas possuem uma relação de ordenação entre seus valores. A seguir são apresentados alguns métodos para redução do número de valores distintos em variáveis contínuas ou discretas. Todos os métodos expostos possuem uma etapa comum inicial voltada à ordenação dos valores existentes no conjunto de dados.

- Particionamento em Células (Bins) de mesma Cardinalidade (“*Equidepth Bins*”) – Nesse método, os valores são agrupados em células com o mesmo número de elementos em cada uma delas. A última célula pode conter mais valores em função de um número de valores que não seja múltiplo do número de células. Os valores originais são substituídos pela identificação de cada célula, gerando um novo conjunto de dados. A Figura 3.2 (a) mostra o resultado deste método quando aplicado à variável número de dependentes do nosso conjunto de dados de exemplo. A terceira célula possui 4 valores. As demais apenas 3.
- Redução de Valores pelas Medianas das Células (“*Bin Medians*”) – O mesmo procedimento descrito para o método acima é realizado neste método. Em seguida, é calculada a mediana de cada uma das células. Os valores originais são substituídos pela mediana associada a cada célula, gerando

um novo conjunto de dados. A Figura 3.2 (b) mostra o resultado desse método quando aplicado à variável número de dependentes do nosso conjunto de dados de exemplo.

- Redução de Valores pelas Médias das Células (“*Bin Means*”) – O procedimento desse método é análogo ao descrito para o método acima. Nesse caso, é calculada a média dos valores em cada uma das células. Os valores originais são substituídos pela média associada a cada célula, gerando um novo conjunto de dados. A Figura 3.2 (c) mostra o resultado deste método quando aplicado à variável número de dependentes do nosso conjunto de dados de exemplo.
- Redução de Valores pelos Limites das Células (“*Bin Boundaries*”) – O mesmo procedimento descrito para o primeiro método é realizado neste caso. Em seguida, os valores nos extremos das células são considerados. O procedimento calcula a distância de cada valor em relação aos extremos de cada célula. O valor original é substituído pelo valor do extremo mais próximo, gerando um novo conjunto de dados. A Figura 3.2 (d) mostra o resultado deste método quando aplicado à variável número de dependentes do nosso conjunto de dados de exemplo.
- Arredondamento de Valores – O arredondamento de valores, também chamado de aproximação de valores, é uma função comum em nosso dia-a-dia. O procedimento abaixo mostra uma alternativa para a noção elementar de arredondamento de valores em números inteiros. A variável  $x$  indica o valor original, que deve ser arredondado. A variável  $y$  recebe o resultado intermediário do procedimento de arredondamento. O parâmetro  $k$  é o número de casas decimais à direita a ser considerado no arredondamento.

$$\begin{aligned} y &= \text{int}(x/10^k) \\ \text{if } (\text{mod}(x, 10^k) \geq (10^k/2)) \text{ then } y &= y + 1 \\ x &= y * 10^k \end{aligned}$$

Considere como exemplos:

(i)  $x = 145, K=2$

*Primeiro passo:*  $y = \text{int}(145/10^2) = \text{int}(1,45) = 1$

*Segundo passo:* não é executado

*Terceiro passo:*  $x = 1 * 10^2 = 100$

(ii)  $x = 145, K=1$

*Primeiro passo:*  $y = \text{int}(145/10^1) = \text{int}(14,5) = 14$

*Segundo passo:*  $y = y + 1 = 14 + 1 = 15$

*Terceiro passo:*  $x = 15 * 10^1 = 150$

- Agrupamento de Valores (“*Clusterização*”) – Consiste em agrupar os valores de um atributo em *clusters* (grupos) levando em consideração a similaridade existente entre tais valores. O processo de *Clusterização*, descrito em detalhe no Capítulo 4, consiste em procurar minimizar a similaridade dos dados pertencentes ao mesmo *cluster* e maximizar a similaridade dos dados em *clusters* distintos. Uma vez concluído o processo de *Clusterização*, cada *cluster* pode passar a ser representado pela média dos valores a ele atribuídos. É importante ressaltar que o problema da redução de valores pode ser interpretado como um problema de otimização.

**Observação:** Com exceção do método de Arredondamento de Valores, os demais métodos podem ser adaptados para considerar classes que eventualmente ocorram em um problema. Dessa forma, os métodos devem ser aplicados apenas para os registros que pertençam a cada classe separadamente.

Duas heurísticas podem ser utilizadas para auxiliar na escolha de qual, dentre os métodos acima, deve ser utilizado:

- Quando o número de células for de moderado a grande, recomenda-se a utilização dos métodos de redução de valores pela mediana ou pela média.
- Quando o número de células for pequeno, recomenda-se a utilização do método de redução de valores pelos limites das células.

1	1	2	3	3	3	4	5	5	7
Bin1	Bin1	Bin1	Bin2	Bin2	Bin2	Bin3	Bin3	Bin3	Bin3

(a) Particionamento em Células de mesma Cardinalidade

1	1	2	3	3	3	4	5	5	7
1	1	1	3	3	3	5	5	5	5

(b) Redução de Valores pelas Medianas das Células

1	1	2	3	3	3	4	5	5	7
1,3	1,3	1,3	3	3	3	5,3	5,3	5,3	5,3

(c) Redução de Valores pelas Médias das Células

1	1	2	3	3	3	4	5	5	7
1	1	2	3	3	3	4	4	4	7

(d) Redução de Valores pelos Limites das Células

**Figura 3.2.** Resultados dos Métodos de Redução de Valores.

## **Limpeza**

Em aplicações reais, é comum que os dados sobre os quais se deseja extrair algum conhecimento estejam incompletos, ruidosos ou inconsistentes. Os dados são considerados incompletos se há informação ausente para determinados atributos ou ainda se há dados pouco detalhados. Dados ruidosos são dados errados ou que contenham valores considerados divergentes, denominados *outliers*, do padrão normal esperado. Dados inconsistentes são aqueles que contêm algum tipo de discrepância semântica entre si.

É importante perceber que a qualidade dos dados possui grande influência na qualidade dos modelos de conhecimento a serem abstraídos a partir destes dados. Quanto pior for a qualidade dos dados informados ao processo de KDD, pior será a qualidade dos modelos de conhecimento gerados (*GIGO – Garbage in, Garbage out*).

A etapa de pré-processamento envolve, dentre outras funções, a limpeza dos dados. A percepção sobre como os dados devem ser pré-processados a fim de melhorar a qualidade dos dados e, consequentemente, dos resultados da mineração constitui-se em uma questão de grande relevância no processo de KDD.

A fase de limpeza dos dados envolve uma verificação da consistência das informações, a correção de possíveis erros e o preenchimento ou a eliminação de valores desconhecidos e redundantes, além da eliminação de valores não pertencentes ao domínio. A execução dessa fase tem como objetivo, portanto, corrigir a base de dados, eliminando consultas desnecessárias que poderiam ser executadas futuramente pelos algoritmos de Mineração de Dados, afetando o desempenho destes algoritmos.

Em geral, os métodos de limpeza dos dados dependem do contexto da aplicação e pressupõem a caracterização dos domínios envolvidos. Assim sendo, a participação dos especialistas em KDD e dos especialistas na área da aplicação é essencial ao processo.

Um exemplo simples de limpeza de dados seria a definição de um intervalo de possíveis valores (domínio) para um determinado atributo. Caso surgisse qualquer valor diferente dos definidos no domínio, esse dado seria corrigido ou mesmo removido da base de dados.

A melhor maneira de evitar a poluição dos dados é organizando a entrada dos dados. Rotinas de crítica nas interfaces de entrada de dados dos sistemas de informação são de grande valor para evitar a poluição dos dados. No entanto, nem sempre tais rotinas estão disponíveis. Assim sendo, a seguir encontram-se apresentadas e comentadas algumas das principais funções de limpeza de dados usualmente utilizadas.

### **Limpeza de Informações Ausentes**

Essa função compreende a eliminação de valores ausentes em conjuntos de dados. Alguns métodos para preenchimento de valores ausentes estão descritos logo a seguir.

- *Exclusão de Casos* – Esse é o método mais simples para limpeza de informações ausentes. Consiste em excluir do conjunto de dados as tuplas que possuam pelo menos um atributo não preenchido. Em geral, esse método não é o mais adequado, a menos que a tupla a ser excluída contenha diversos atributos com valores ausentes. Quando a percentagem de valores ausentes por atributo varia significativamente na mesma base, esse método deve ser evitado, sob o risco de restarem poucos registros com todas as informações preenchidas.
- *Preenchimento Manual de Valores* – Em geral, essa abordagem demanda alto consumo de tempo e recursos, sendo muitas vezes inviável na prática (nos casos em que grandes volumes de dados possuam informações desconhecidas). Esse método pode ser implementado por meio de pesquisas junto às fontes de dados originais que procurem captar as informações ausentes. Nesse método, os dados pesquisados devem ser complementados via digitação.
- *Preenchimento com Valores Globais Constantes* – Esse método consiste em substituir todos os valores ausentes de um atributo por um valor padrão tal como “desconhecido” ou “null”. Esse valor padrão pode e deve ser especificado pelo especialista no domínio da aplicação. Cabe ressaltar, no entanto, que determinados algoritmos de mineração de dados podem assumir constantes padrões como valores recorrentes importantes. Assim sendo, embora simples, esse método não está entre os mais indicados.
- *Preenchimento com Medidas Estatísticas* – Medidas estatísticas podem ser empregadas como alternativa à utilização de constantes padrões no processo de preenchimento de valores ausentes. Como exemplos de medidas estatísticas para preenchimento de informações ausentes podem ser citados: *média* para atributos numéricos e *moda* para atributos categóricos. Uma variação desse método pode ser utilizada em problemas de classificação. Nesse caso, em vez de considerar todo o conjunto de dados, o preenchimento com medidas estatísticas fica restrito aos registros de cada classe. Assim sendo, o cálculo da medida estatística a ser utilizada no preenchimento de informações ausentes fica restrito aos registros que pertencem a cada classe. Em nosso exemplo, as informações de renda ausentes poderiam ser substituídas por: R\$3860,00 na classe A e R\$1300,00 na classe I.
- *Preenchimento com Métodos de Mineração de Dados* – Nesse caso, modelos preditivos podem ser construídos de forma a sugerir os valores mais prováveis a serem utilizados no preenchimento dos valores ausentes. Algoritmos de Mineração de Dados tais como Redes Neurais, Estatística (Modelos Bayesianos) e Árvores de Decisão são alternativas na construção destes modelos. É importante observar que, mesmo durante a etapa de pré-processamento, algoritmos de Mineração de Dados podem ser utilizados para preenchimento de valores ausentes.

**Observação:** Os 4 primeiros métodos podem tornar tendenciosos determinados atributos. O último método, no entanto, baseia-se em uma estratégia muito utilizada na prática. Em comparação aos demais métodos, utiliza a informação existente para predizer os valores ausentes. A utilização de atributos para prever os valores ausentes de outro atributo proporciona a chance de que relacionamentos eventualmente existentes entre tais atributos sejam preservados.

## Limpeza de Inconsistências

Essa função comprehende a identificação e a eliminação de valores inconsistentes em conjuntos de dados. Uma inconsistência pode envolver uma única tupla, ou um conjunto de tuplas. Inconsistência em uma única tupla ocorre quando houver divergência entre os valores desta tupla. Em nosso exemplo, a primeira tupla apresenta uma inconsistência: um cliente com idade inferior a 21 anos possui um crédito aprovado. Conforme pode ser observado, a participação do especialista no domínio da aplicação é fundamental na identificação de inconsistências. Esse processo demanda conhecimento prévio acerca do problema. Alguns métodos para limpeza de inconsistências estão descritos logo abaixo.

- *Exclusão de Casos* – Esse método, similar ao descrito na Limpeza de Valores Ausentes, é o mais simples para limpeza de informações inconsistentes. Consiste em excluir do conjunto de dados original, as tuplas que possuam pelo uma inconsistência. A identificação dos casos com inconsistência pode ser obtida por meio de consultas em SQL cujas restrições especifiquem o tipo de inconsistência a ser verificada.

Exemplo: Obtenha os clientes com menos de 21 anos para os quais tenha sido concedido crédito.

```
SELECT *
FROM CLIENTE
WHERE (YEAR(SYSDATE) - YEAR(DT_NC)) < 21 AND RESULT=A";
SYSDATE = "01/05/2005" (Data do dia)
```

- *Correção de Erros* – Esse método consiste em substituir valores errôneos ou inconsistentes identificados no conjunto de dados. Pode envolver desferir a correção manual até a atualização desses valores em um lote predeterminedado de registros, utilizando comandos de atualização de dados em ambientes relacionais.

## Limpeza de Valores não pertencentes ao Domínio

Essa função comprehende a identificação e a eliminação de valores que não pertençam ao domínio dos atributos do problema. Tal função pode ser considerada

um caso particular da operação de Limpeza de Inconsistências e demanda o conhecimento prévio do domínio de cada atributo. Em nosso exemplo, o valor “X” pertence ao domínio do atributo Tipo de Residência: a primeira tupla apresenta um valor inválido. Alguns métodos para limpeza de valores não pertencentes ao domínio estão descritos logo abaixo.

- *Exclusão de Casos* – Esse é o método mais simples para limpeza de valores fora do domínio. Consiste em excluir do conjunto de dados original, as tuplas que possuam pelo um valor fora do conjunto de valores válidos de cada atributo.

Exemplo: Obtenha os clientes com número de dependentes abaixo de zero.

```
SELECT *
FROM CLIENTE
WHERE QTDE_DEP < 0;
```

- *Correção de Erros* – Esse método consiste em substituir os valores inválidos identificados no conjunto de dados. Pode envolver desde a correção manual até a atualização destes valores em um lote predeterminado de registros utilizando comandos SQL.

## **Codificação**

Codificação de dados é operação de pré-processamento responsável pela forma como os dados serão representados durante o processo de KDD. Trata-se de uma atividade criativa que deve ser realizada repetidas vezes em busca melhores representações.

É importante compreender que os dados devem ser codificados de forma a atender às necessidades específicas dos algoritmos de Mineração de Dados. Por exemplo, uma rede neural requer que os dados estejam em uma representação numérica. Assim sendo, caso a base de dados a ser processada apresente valores nominais, estes devem ser codificados antes de serem submetidos à rede.

A maneira como a informação é codificada tem forte influência sobre o tipo de conhecimento a ser encontrado. Em essência, a codificação pode ser: *Numérica – Categórica*, que divide valores de atributos contínuos em intervalos codificados ou *Categórica – Numérica*, que representa valores de atributos categóricos por códigos numéricos.

### **Codificação: Numérica – Categórica**

- *Mapeamento Direto* – Consiste na simples substituição dos valores numéricos por valores categóricos. Por exemplo:

Sexo:

1 → M

0 → F

- *Mapeamento em Intervalos* – Também denominada *Discretização*, a representação em intervalos pode ser obtida a partir de métodos que dividam o domínio de uma variável numérica em intervalos. Alguns autores consideram o processo de Discretização como pertencente ao conjunto de operações voltadas à redução de valores das variáveis (Redução de Valores Contínuos). Existem diversos procedimentos para Discretização. A seguir encontram-se ilustrados alguns deles. Como exemplo considere o atributo renda com os seguintes valores, já organizados em ordem crescente: 1000, 1400, 1500, 1700, 2500, 3000, 3700, 4300, 4500, 5000.

- Divisão em intervalos com comprimentos definidos pelo usuário – Nesse caso, o usuário define o número de intervalos e escolhe o tamanho de cada deles. Por exemplo: três intervalos de comprimento 1600, 2800 e 1000. Observe que cada intervalo está representado por LI |– LS, que compreende todos os valores reais desde o limite inferior do intervalo (LI) até o limite superior do intervalo (LS), não incluso. Em termos matemáticos, intervalo fechado em LI e aberto em LS.

Intervalo	Freqüência (número de valores no intervalo)
1000  – 1600	3
1600  – 4400	5
4400  – 5400	2

- Divisão em intervalos de igual comprimento – Nesse caso, o usuário define somente o número de intervalos. O comprimento destes intervalos é calculado a partir do maior e do menor valor do domínio do atributo. Por exemplo: quatro intervalos. Como a faixa de valores vai de 1000 a 5000, faz-se  $R = 5000 - 1000 = 4000$ . Assim, cada intervalo terá comprimento 1000 ( $4000/4$ ).

Intervalo	Freqüência (número de valores no intervalo)
1000  – 2000	4
2000  – 3000	1
3000  – 4000	2
4000  – 5000	3

Uma variação comum a este procedimento consiste em utilizar um critério para definir a quantidade de intervalos. Em geral, estes critérios envolvem o número de elementos do domínio, incluindo as repetições de valores. Exemplo de critério: o número de intervalos ( $k$ ) é 5 se o número de amostras for inferior ou igual a 25. Nos demais casos, o número de intervalos é aproximadamente a raiz quadrada do número de amostras. A amplitude é expressa por  $R = X_{max} - X_{min}$  ( $R = 5000 - 1000 = 4000$ ). O comprimento de cada intervalo é obtido por  $h = R/k$  ( $h = 4000/5 = 800$ ). Seguindo este critério, temos a seguinte divisão:

Intervalo	Freqüência (número de valores no intervalo)
1000  – 1800	4
1800  – 2600	1
2600  – 3400	1
3400  – 4200	1
4200  – 5000	2

- c) Divisão em intervalos por meio de Clusterização – Consiste em agrupar os valores de um atributo em *clusters* (grupos) levando em consideração a similaridade existente entre tais valores. O processo de *Clusterização* está descrito em detalhe no Capítulo 4. Uma vez concluído o processo de Clusterização, cada *cluster* pode passar a ser representado por um intervalo delimitado pelo menor e pelo maior valor identificado no *cluster*. Esse procedimento requer que o usuário especifique previamente o número de *clusters* a ser considerado. É importante perceber que, de forma similar ao problema da redução de valores de um atributo, a Divisão de Intervalos por Clusterização pode ser interpretada como um problema de otimização. Assim sendo, métodos de otimização podem ser utilizados na implementação desse processo.

### Codificação: Categórica – Numérica

- *Representação Binária Padrão (Econômica)* – Nesta representação, cada valor categórico é associado a um valor de 1 até  $N$  e é representado por uma cadeia de dígitos binários. Por exemplo, se temos 5 possíveis valores, podemos representá-los com cadeias binárias de comprimento 3:

Valores Originais	Representação Binária Padrão
Casado	001
Solteiro	010
Viúvo	100
Divorciado	011
Outro	110

- *Representação Binária 1-de-N* – Nessa representação, o código 1-N tem um comprimento igual ao número de categorias discreteas permitidas para a variável, onde cada elemento na cadeia de bits é 0, exceto para um único elemento: aquele que representa o valor da categoria em questão. No exemplo:

Valores Originais	Representação Binária 1-de-N
Casado	00001
Solteiro	00010
Viúvo	00100
Divorciado	01000
Outro	10000

- *Representação Binária por Temperatura* – Essa representação é utilizada mais freqüentemente quando os valores discretos estão relacionados de algum modo. Por exemplo, uma variável discreta que pode ter um dos seguintes valores: fraco, regular, bom e ótimo. Nesse caso, existe uma graduação entre os valores. O valor ótimo é o melhor caso e o valor fraco, o pior. Assim sendo, a diferença entre os conceitos fraco e ótimo deve ser a maior possível entre os valores. Por outro lado, as diferenças entre valores adjacentes na escala devem ser iguais à menor diferença possível. Cada valor corresponde a um acréscimo de um bit igual a 1 na representação, conforme mostra a tabela abaixo.

Valores Originais	Representação Binária por Temperatura
Fraco	0001
Regular	0011
Bom	0111
Ótimo	1111

Uma medida de distância normalmente utilizada conjuntamente a esta representação é a distância de *Hamming* (*DH*), também conhecida como *City-Block*. Essa distância expressa a diferença entre duas cadeias de bits, adicionando uma unidade sempre que bits de mesma posição possuem valores distintos. A tabela a seguir mostra a distância de *Hamming* entre os conceitos.

DH	Fraco	Regular	Bom	Ótimo
Fraco	0	1	2	3
Regular	1	0	1	2
Bom	2	1	0	1
Ótimo	3	2	1	0

## ***Enriquecimento***

A fase de enriquecimento consiste em conseguir agregar mais informações aos registros existentes para que estes forneçam mais elementos para o processo de descoberta de conhecimento. A seguir estão comentadas algumas das operações mais usualmente utilizadas no processo de enriquecimento das bases de dados.

### **Pesquisas**

Estão incluídas nessa operação todas as iniciativas de enriquecimento que envolvem a captação de novas informações junto às fontes originais. Normalmente requerem a inclusão de novos atributos ou mesmo de novas tabelas nas bases de dados existentes. Diferem das operações de limpeza porque não estão restritas a preencher informações ausentes. Busca-se, no caso do enriquecimento, agregar novas informações. Muitas vezes inviável devido ao alto custo de implementação, as pesquisas podem ser realizadas considerando uma amostra do universo completo de casos.

Recomenda-se uma especial atenção ao processo de carga das informações captadas em pesquisas. Nos casos de pesquisas realizadas em papel, as interfaces para entrada de dados devem refletir os formulários utilizados de forma a facilitar a digitação. Adicionalmente, as devidas críticas de dados devem estar implementadas nas interfaces de forma a minimizar os esforços posteriores na limpeza das informações.

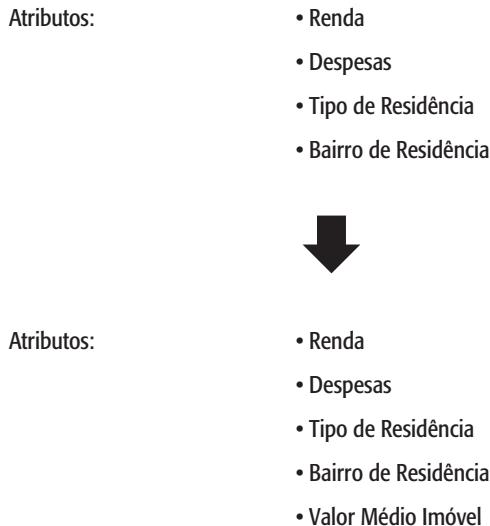
### **Consultas a Bases de Dados Externas**

O processo de enriquecimento pode ser realizado mediante a incorporação de informações fornecidas por outros sistemas. É muito comum a importação de informações advindas de outras bases de dados. A Figura 3.3 apresenta um exemplo de enriquecimento dos dados de clientes por meio da importação de informações provenientes de outra fonte de dados. Em adição aos dados existentes sobre clientes, alguns dados demográficos podem ser obtidos e incorporados ao conjunto existente, visando a definição dos preços médios dos imóveis dos clientes. Tais informações podem ser úteis em um contexto de análise e concessão de créditos. Por exemplo, para clientes que informaram morar em imóvel próprio, o valor médio dos imóveis no bairro de residência fornece um indicador da capacidade de aquisição patrimonial destes clientes.

### ***Normalização de Dados***

Essa operação consiste em ajustar a escala dos valores de cada atributo de forma que os valores fiquem em pequenos intervalos, tais como de -1 a 1, ou de 0 a 1. Tal ajuste faz-se necessário para evitar que alguns atributos, por apresentarem

uma escala de valores maior que outros, influenciem de forma tendenciosa em determinados métodos de Mineração de Dados. Abaixo estão apresentados, de forma resumida, alguns métodos de normalização de dados.



**Figura 3.3.** Exemplo de Enriquecimento de Dados.

## Normalização Linear

Também denominada normalização por interpolação linear, a normalização linear consiste em considerar os valores mínimo e máximo de cada atributo no ajuste de escala. Mapeia os valores de um atributo no intervalo fechado de 0 até 1. Mantém distâncias entre os dados normalizados que sejam proporcionais às distâncias entre os dados originais. Recomenda-se a utilização deste método somente nos casos em que exista a certeza de que o domínio do atributo está entre os valores mínimo e máximo considerados.

$$A' = (A - \text{Min}) / (\text{Max} - \text{Min}), \text{ onde:}$$

A' = valor normalizado;

A = valor do atributo a ser normalizado;

Min = valor mínimo do atributo a ser normalizado;

Max = valor máximo do atributo a ser normalizado;

Exemplo: Atributo “*Despesa*” da Tabela 3.2 após a normalização linear:

<b>CPF</b>	<b>Despesa_Normalizada</b>
99999999999	0,14
11111111111	0,43
33333333333	0,71
55555555555	0,29
22222222222	0,29
00000000000	0,14
88888888888	0,71
77777777777	0,00
66666666666	1,00
44444444444	0,14

## Normalização por Desvio Padrão

Também denominada normalização *Z-Score* ou *Zero Mean*, a normalização por desvio padrão considera a posição média dos valores de um atributo, assim como os graus de dispersão destes valores em relação à posição média. Esse método de normalização é útil quando os valores mínimo e máximo do atributo são desconhecidos.

$$A' = (A - X) / \sigma, \text{ onde:}$$

$A'$  = valor normalizado;

$A$  = valor do atributo;

$X$  = média entre os valores do atributo

$\sigma$  = desvio padrão;

Exemplo: Atributo “*Despesa*” da Tabela 3.2 após a normalização por desvio padrão:

<b>CPF</b>	<b>Despesa_Normalizada</b>
99999999999	-0,75
11111111111	0,13
33333333333	1,02
55555555555	-0,31
22222222222	-0,31

<b>CPF</b>	<b>Despesa_Normalizada</b>
000000000000	-0,75
888888888888	1,02
777777777777	-1,19
666666666666	1,90
444444444444	-0,75

### Normalização pela Soma dos Elementos

Consiste em dividir cada valor do atributo que esteja sendo normalizado pelo somatório de todos os valores de tal atributo. Uma desvantagem deste método é que determinados valores podem ser mapeados em valores muito pequenos.

$$A' = A / X, \text{ onde:}$$

A' = valor normalizado;

A = valor do atributo;

X = somatório de todos os valores do atributo

Exemplo: Atributo “*Despesa*” da Tabela 3.2 após a normalização pela soma dos elementos:

<b>CPF</b>	<b>Despesa_Normalizada</b>
999999999999	0,05
111111111111	0,11
333333333333	0,16
555555555555	0,08
222222222222	0,08
000000000000	0,05
888888888888	0,16
777777777777	0,03
666666666666	0,22
444444444444	0,05

### Normalização pelo Valor Máximo dos Elementos

Similar à normalização linear, esse método consiste em dividir cada valor do atributo que esteja sendo normalizado pelo maior valor dentre todos os valores de tal atributo.

$A' = A / \text{Max}$ , onde:

$A'$  = valor normalizado;

$A$  = valor original do atributo a ser normalizado;

$\text{Max}$  = valor máximo do atributo a ser normalizado;

Exemplo: Atributo “*Despesa*” da Tabela 3.2 após a normalização pelo valor máximo dos elementos:

CPF	Despesa_Normalizada
999999999999	0,25
111111111111	0,50
333333333333	0,75
555555555555	0,38
222222222222	0,38
000000000000	0,25
888888888888	0,75
777777777777	0,13
666666666666	1,00
444444444444	0,25

### Normalização por Escala Decimal

Esse método realiza o processo de normalização por meio do deslocamento do ponto decimal dos valores do atributo a ser normalizado. O número de casas decimais depende do maior valor absoluto do atributo em questão.

$A' = A / 10^j$ , onde:

$A'$  = valor normalizado;

$A$  = valor original do atributo a ser normalizado;

$j$  = menor inteiro tal que o maior valor absoluto normalizado seja inferior a 1

Exemplo: Atributo “*Despesa*” da Tabela 3.2 após a normalização por escala decimal:

CPF	Despesa_Normalizada
999999999999	0,10
111111111111	0,20
333333333333	0,30
555555555555	0,15

<b>CPF</b>	<b>Despesa_Normalizada</b>
22222222222	0,15
00000000000	0,10
88888888888	0,30
77777777777	0,05
66666666666	0,40
44444444444	0,10

## ***Construção de Atributos***

Essa operação consiste em gerar novos atributos a partir de atributos existentes. Os novos atributos são denominados atributos derivados. Como exemplo, podemos citar a criação de um atributo “idade” a partir do atributo “dt\_nc” (data de nascimento) e da data corrente do sistema.

1º Passo: Inclusão de um novo campo na tabela:

```
ALTER TABLE CLIENTE  
ADD FIELD (IDADE INTEGER);
```

2º Passo: Preenchimento do novo campo:

```
UPDATE CLIENTE SET  
IDADE = YEAR(SYSDATE) - YEAR(DT_NC);
```

SYSDATE = “01/05/2005” (Data do dia em que foi realizada a operação)

A importância desse tipo de operação é justificada pois novos atributos, além de expressarem relacionamentos conhecidos entre atributos existentes, podem reduzir o conjunto de dados, simplificando o processamento dos algoritmos de Mineração de Dados. A construção de atributos a partir da combinação de atributos existentes pode incorporar ao problema informações de relacionamentos entre os dados, que sejam úteis ao processo de KDD. É conveniente enfatizar que é muito comum a substituição dos atributos existentes pelos respectivos atributos derivados.

Como exemplo de operadores para a construção de atributos podem ser citados os operadores aritméticos (+, -, x, /).

## ***Correção de Prevalência***

Essa operação é muitas vezes necessária em tarefas de classificação. Consiste em corrigir um eventual desequilíbrio na distribuição de registros com determinadas características. Por exemplo, suponha que em nossa base de dados sobre cré-

dito somente 1% dos clientes não tenham quitado suas dívidas. Nesse caso, a descoberta de modelos de conhecimento voltados à classificação de novos clientes pode ser influenciada pela pouca ocorrência de maus pagadores.

O método de Amostragem Estratificada, descrito anteriormente (Redução de Dados Horizontal), é utilizado com freqüência no problema da correção de prevalência. Nesse caso, são selecionadas iguais quantidades de registros para as classes envolvidas.

Um outro método que pode ser utilizado nesta situação é o método de Replicação Aleatória de Registros. Esse método consiste em selecionar aleatoriamente e com reposição registros das classes com menor quantidade de amostras, de forma a equilibrar o volume de casos associados às diversas classes. A aplicação intensiva deste método em um mesmo problema pode tornar altamente tendenciosos os registros de determinadas classes, sobretudo quando há uma grande diferença na distribuição original dos registros.

Uma alternativa aos métodos indicados consiste em utilizar métodos de Mineração de Dados preparados para lidar com problemas de prevalência. É muito comum a utilização do conceito de matriz de custo para compensar o problema da prevalência. Em uma matriz de custo, o peso do erro associado aos registros cujas classes sejam menos numerosas, é normalmente maior, evitando com isso que os modelos de conhecimento abstraídos a partir dos dados sejam tendenciosos.

## **Partição do Conjunto de Dados**

A Mineração de Dados, conforme será mais bem detalhada na próxima seção, é a etapa do processo de KDD responsável pela abstração de modelos de conhecimento a partir dos dados existentes. A qualidade desses modelos precisa ser avaliada. A avaliação de um modelo de conhecimento requer a confrontação deste com dados visando à mensuração de algumas medidas que expressem a qualidade deste modelo. Para que essa avaliação seja isenta, os dados utilizados na construção do modelo não devem ser os mesmos utilizados na avaliação desse modelo. Portanto, pelo menos dois conjuntos de dados devem ser utilizados no processo de KDD: um conjunto de treinamento e um conjunto de testes. O conjunto de treinamento deve conter os registros a serem utilizados na construção do modelo de conhecimento. O conjunto de testes, conforme o próprio nome indica, deve conter os registros a serem utilizados na avaliação do modelo de conhecimento gerado. Como, em geral, o processo de KDD possui um conjunto de dados, a operação de partição do conjunto de dados em treinamento e teste assume grande importância. A seguir estão indicados alguns métodos utilizados na partição do conjunto de dados, com vistas à posterior avaliação dos modelos de conhecimento gerados.

- *Holdout* – Esse método divide aleatoriamente os registros em uma percentagem fixa  $p$  para treinamento e  $(1 - p)$  para teste, considerando nor-

malmente  $p > \frac{1}{2}$ . Embora não existam fundamentos teóricos sobre esta percentagem, valores tipicamente utilizados são:  $p = 2/3$  e  $(1 - p) = 1/3$  (Rezende, 2003). Essa abordagem é muito utilizada quando deseja-se produzir um único modelo de conhecimento a ser aplicado posteriormente em algum sistema de apoio à decisão.

- *Validação Cruzada com K Conjuntos (K-Fold CrossValidation)* – Esse método consiste em dividir aleatoriamente o conjunto de dados com  $N$  elementos em  $K$  subconjuntos disjuntos (*folds*), com aproximadamente o mesmo número de elementos ( $N / K$ ). Neste processo, cada um dos  $K$  subconjuntos é utilizado como conjunto de teste e os  $(K - 1)$  demais subconjuntos são reunidos em um conjunto de treinamento. Assim, o processo é repetido  $K$  vezes, sendo gerados e avaliados  $K$  modelos de conhecimento. Essa abordagem é muito utilizada quando se deseja avaliar a tecnologia utilizada na formulação do algoritmo de Mineração de Dados e quando a construção de um modelo de conhecimento para uso posterior não seja prioridade.
- *Validação Cruzada com K Conjuntos Estratificada (Stratified K-Fold CrossValidation)* – Aplicável em problemas de classificação, este método é similar à *Validação Cruzada com K Conjuntos*, sendo que ao gerar os subconjuntos mutuamente exclusivos, a proporção de exemplos em cada uma das classes é considerada durante a amostragem. Isso significa, por exemplo, que se o conjunto de dados original possui duas classes com distribuição de 20% e 80%, cada subconjunto também deverá conter aproximadamente esta mesma proporção de classes.
- *Leave-One-Out* – Esse método é um caso particular da *Validação Cruzada com K Conjuntos*, em que cada um dos  $K$  subconjuntos possui um único registro. É computacionalmente dispendioso e freqüentemente usado em pequenas amostras.
- *Bootstrap* – O conjunto de treinamento é gerado a partir de  $N$  sorteios aleatórios e com reposição a partir do conjunto de dados original (contendo  $N$  registros). O conjunto de teste é composto pelos registros do conjunto de dados original não sorteados para o conjunto de treinamento. Esse método consiste em gerar os conjuntos, abstrair e avaliar o modelo de conhecimento um número repetido de vezes, a fim de estimar uma média de desempenho do algoritmo de Mineração de Dados.

## Mineração de Dados

A Mineração de Dados é a principal etapa do processo de KDD. Nessa etapa ocorre a busca efetiva por conhecimentos novos e úteis a partir dos dados. Por este motivo, diversos autores referem-se à Mineração de Dados e ao Processo de KDD de forma indistinta, como se fossem sinônimos.

A execução da etapa de Mineração de Dados compreende a aplicação de algoritmos sobre os dados procurando abstrair conhecimento. Conforme mencionado no Capítulo 2, estes algoritmos são fundamentados em técnicas que procuram, segundo determinados paradigmas, explorar os dados de forma a produzir modelos de conhecimento. Neste livro, todo conhecimento abstraído ao longo do processo de KDD será interpretado e referenciado pela expressão *modelo de conhecimento*. A forma de representação do conhecimento em um modelo de conhecimento depende diretamente do algoritmo de Mineração de Dados utilizado.

Diversos conceitos importantes serão apresentados a seguir. Para melhor compreensão, tais conceitos serão introduzidos de forma ilustrada com um exemplo no contexto da análise de crédito já familiar ao leitor.

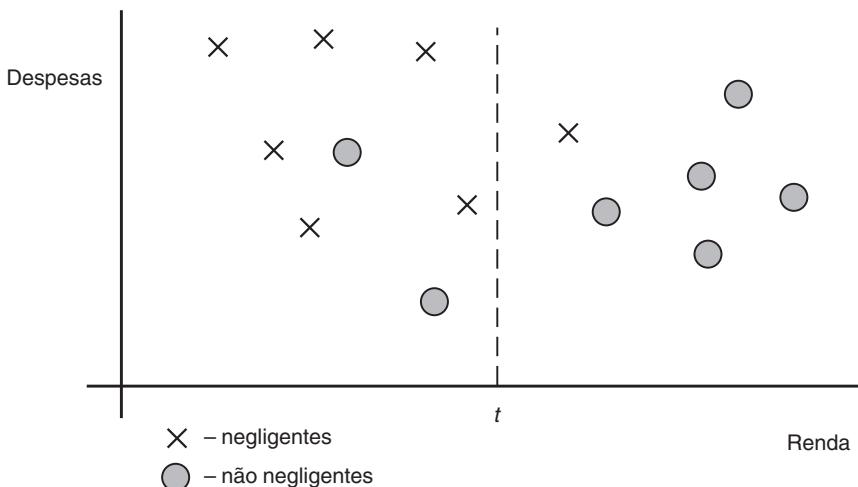
Todo *conjunto de dados* no processo de KDD corresponde a uma *base de fatos* ocorridos que devem ser interpretados como um *conjunto de pontos* em um *hiperespacô de dimensão K*. A dimensão da base de fatos é determinada pelo número de atributos do conjunto de dados em análise. A Figura 3.4 mostra o exemplo no contexto da análise de crédito na qual três informações estão representadas em um plano cartesiano. Os eixos correspondem aos atributos *Renda* e *Despesa*. Cada ponto representa um caso. O símbolo associado a cada caso (“círculo” ou “xis”) fornece a terceira informação, que corresponde ao comportamento do cliente quanto ao pagamento do crédito concedido.

Por outro lado, todo processo de KDD deve ser norteado por objetivos. Os *objetivos de um processo de KDD* compreendem a definição da *tarefa de KDD* a ser executada e da *expectativa* que os convededores do domínio da aplicação tenham com relação ao *modelo de conhecimento* a ser gerado. A partir dessas definições, o especialista em KDD tem condições de delinear que *tipos de padrões* devem ser abstraídos a partir dos dados.

Imaginemos em nosso exemplo que a financeira responsável pelos dados deseja obter um modelo de conhecimento que preveja o comportamento de futuros clientes quanto ao pagamento de suas dívidas com uma taxa máxima tolerável de erro de 5%. Esta intenção aliada à base de dados disponível nos conduz à *tarefa de classificação* dos clientes. Esta e outras tarefas de KDD encontram-se melhor descritas no Capítulo 4. Por ora, basta compreender que a tarefa de classificação consiste em gerar um modelo de conhecimento a partir do histórico de casos disponível que consiga, a partir dos dados de novos clientes, prever em qual classe de comportamento o cliente deverá se enquadrar.

O tipo de padrão desejado em nosso exemplo é uma função que separe da melhor forma possível a classe dos clientes negligentes da classe dos clientes não negligentes. São infinitas as possibilidades de funções, também denominadas *hipóteses*, que tenham como objetivo separar os dois conjuntos. Tais hipóteses pertencem a um *espaço de padrões*, normalmente infinito, que deve ser percorrido pelos algoritmos de Mineração de Dados em busca de elementos que aten-

dam a determinadas condições, que são as expectativas com relação ao modelo de conhecimento desejado. No exemplo, as hipóteses que interessam à financeira são aquelas que conduzem a uma *taxa de erro* de no máximo 5% dos casos analisados.



**Figura 3.4.** Exemplo de uma aplicação com “hiperespaço” de dimensão 2.

O conceito de medida de interesse é essencial ao processo de KDD por dois motivos principais: a) Medidas de interesse podem ser usadas após a etapa de Mineração de Dados (etapa de pós-processamento) a fim de ordenar ou filtrar os padrões descobertos de acordo com o grau de interesse associado a estes padrões; b) Medidas de interesse podem ser usadas para guiar o restringir o espaço de busca da Mineração de Dados, melhorando a eficiência da busca ao eliminar conjuntos de padrões que não satisfaçam a condições predeterminadas.

Existem basicamente dois tipos de medidas de interesse que podem ser associadas aos modelos de conhecimento em Mineração de Dados: *medidas de interesse objetivas* e *medidas de interesse subjetivas*.

As *medidas de interesse objetivas* são baseadas na estrutura dos padrões descobertos e nas estatísticas a eles relacionados. A taxa de erro, mencionada em nosso exemplo, é uma ilustração de *medida de interesse objetiva* acerca de modelos de conhecimento. Outras medidas de interesse serão citadas no Capítulo 4, juntamente com a apresentação das tarefas de Mineração de Dados.

As *medidas de interesse subjetivas* são em crenças que os especialistas no domínio da aplicação tenham com relação aos dados e aos modelos de conhecimento gerados. Surpresa (conhecimentos não esperados), contradições (conhecimentos que contrariem determinadas expectativas dos especialistas), ou ainda alternativas de ações estratégicas (conhecimentos que ofereçam informações estratégicas) são exemplos de medidas de interesse subjetivas. Padrões esperados

podem ser considerados interessantes caso confirmem suspeitas dos especialistas em determinados temas. A avaliação envolvendo este tipo de medida depende, muitas vezes, da visualização e da interpretação dos resultados obtidos, normalmente realizadas na etapa de pós-processamento.

Considere ainda que, por razões de segurança e confiabilidade do processo, a financeira deseja que o modelo de conhecimento seja transparente, de forma que os especialistas em crédito da instituição sejam capazes de ler, entender e analisar o conhecimento gerado. Em função desta expectativa, o especialista em KDD deve restringir o conjunto de algoritmos de Mineração de Dados aos algoritmos cujos modelos de conhecimento gerados sejam interpretáveis pelo homem. Algoritmos tais como *C4.5* e *Rough Sets* podem ser utilizados. Por outro lado, algoritmos tais como *Back-Propagation* e *K-NN* devem ser descartados diante deste tipo de expectativa. As descrições de alguns destes e de outros algoritmos aplicáveis em Mineração de Dados encontram-se no Capítulo 5.

Um outro fator que influencia na escolha dos algoritmos de Mineração de Dados a serem utilizados em cada problema diz respeito aos tipos de variáveis envolvidas. Determinados algoritmos possuem restrições quanto aos tipos de variáveis existentes no conjunto de dados. Neste caso, duas alternativas podem ser consideradas: a) elimina-se do conjunto de algoritmos de Mineração de Dados todos aqueles que forem incompatíveis com os tipos de variáveis envolvidas no problema; ou b) opta-se por utilizar um determinado algoritmo de Mineração de Dados e realizar todo o pré-processamento sobre o conjunto de dados de forma a torná-lo compatível com o algoritmo desejado.

Um conceito muito importante e muito utilizado em Mineração de Dados é a noção de *similaridade*. Uma vez que o conjunto de dados pode ser interpretado com um conjunto de pontos em um espaço k-dimensional, o conceito de *similaridade* entre dois pontos pode ser traduzido como a *distância* entre estes pontos. Quanto maior a similaridade, menor a distância entre os pontos.

O conceito de distância é formalizado como uma função  $D : E \times E \rightarrow \mathbb{R}$  (a cada par de pontos associa um valor real) que atende às seguintes restrições:

- $D(x,x) = 0$
- $D(x,y) = D(y,x)$
- $D(x,y) \leq D(x,z) + D(z,y)$

Exemplos comuns de distância são:

- Distância Euclideana:  $d(X, Y) = \sqrt{\sum_{i=1}^n (X_i - Y_i)^2}$

- Distância de Hamming (City-Block):  $d(X, Y) = \sum_{i=1}^n |X_i - Y_i|$
- Distância de Minkowski:  $d(X, Y) = (\sum_{i=1}^n |X_i - Y_i|^p)^{1/p}$

Diversos algoritmos de Mineração de Dados, com destaque para o *K-NN* (*K Nearest Neighbors* – *K* Vizinhos mais Próximos) utilizam o conceito de distância entre os registros do banco de dados. Uma descrição detalhada do algoritmo *K-NN* encontra-se no Capítulo 5.

Um outro conceito muito importante envolvido no processo de KDD e mais especificamente na etapa da Mineração de Dados refere-se à capacidade que determinados algoritmos têm de aprender a partir de exemplos. Tais algoritmos aprendem os relacionamentos eventualmente existentes entre os dados, retratando o resultado deste aprendizado nos modelos de conhecimento gerados.

As principais abordagens de aprendizado normalmente aplicadas em Mineração de Dados são: *aprendizado supervisionado* e *aprendizado não supervisionado*.

O *aprendizado supervisionado* comprehende a abstração de um modelo de conhecimento a partir dos dados apresentados na forma de pares ordenados (*entrada, saída desejada*). Por *entrada* entenda-se o conjunto de valores das variáveis de entrada do algoritmo para um determinado caso. A *saída desejada* corresponde ao valor que se espera que o algoritmo possa produzir sempre que receber os valores especificados em *entrada*. O *Back-Propagation* e *C4.5* são exemplos de algoritmos que utilizam a abordagem de aprendizado supervisionado. Algoritmos deste tipo necessitam de pelo menos dois conjuntos de dados: conjunto de treinamento e conjunto de teste. O modelo de conhecimento é abstraído a partir do conjunto de treinamento e avaliado a partir do conjunto de testes. Maiores detalhes sobre estes algoritmos podem ser obtidos no Capítulo 5.

No *aprendizado não supervisionado* não existe a informação da saída desejada. Os algoritmos partem dos dados, procurando estabelecer relacionamentos entre eles. Como exemplos clássicos de algoritmos que utilizam aprendizado não supervisionado estão o *K-Means* e o *Apriori*, ambos descritos também no Capítulo 5.

## Pós-processamento

Essa fase envolve a visualização, a análise e a interpretação do modelo de conhecimento gerado pela etapa de Mineração de Dados. Em geral, é nesta etapa que o especialista em KDD e o especialista no domínio da aplicação avaliam os resultados obtidos e definem novas alternativas de investigação dos dados. A seguir encontram-se indicadas algumas operações de pós-processamento.

## **Simplificações de Modelo de Conhecimento**

A simplificação de um modelo de conhecimento, conforme o próprio nome sugere, consiste em remover detalhes deste modelo de conhecimento de forma a torná-lo menos complexo, sem perda de informação relevante.

A representação de conhecimento por meio de regras é muito utilizada em KDD. Conjuntos com grandes quantidades de regras são de difícil interpretação.

Existem métodos voltados ao corte de regras. Esses métodos se baseiam em medidas de qualidade das regras tais como precisão e abrangência (Han e Kember, 2001).

Para ilustrar esta idéia, consideremos um modelo de conhecimento composto por regras da forma:  $X \rightarrow Y$ , onde X e Y são predicados (condições que podem se tornar verdadeiras ou falsas em função de cada registro da base de dados). Consideremos ainda as seguintes definições:

- Precisão ou Acurácia da Regra: É o percentual de registros da base de dados que ao satisfazerem ao antecedente da regra, satisfazem também ao consequente.

$$Acc = \frac{|X \wedge Y|}{|X|}$$

- Abrangência da Regra: É o percentual de registros da base de dados que ao satisfazerem ao consequente da regra, satisfazem também ao antecedente.

$$Abr = \frac{|X \wedge Y|}{|Y|}$$

É muito comum em Mineração de Dados que o usuário estabeleça limites mínimos de acurácia e abrangência para as regras, de tal forma a excluir do modelo de conhecimento gerado todas as regras que não satisfaçam a tais limites.

O corte de atributos em determinadas regras também constitui-se em uma alternativa para simplificação de modelos de conhecimento. O *ID3*, ou sua versão mais recente *C4.5*, são algoritmos que eliminam atributos e consequentemente conjuntos de regras baseados no conceito de entropia da Teoria da Informação (Quinlan, 1993). De uma forma simplificada, o grau de entropia de um conjunto de atributos expressa o grau de complexidade da informação contida no referido conjunto. Assim, quanto menor a entropia, menor a quantidade de informação codificada em um ou mais atributos. Em contrapartida, quanto maior a entropia de um conjunto de atributos, maior a relevância destes atributos na descrição do conjunto de dados.

De forma análoga ao descrito para regras, existem métodos de simplificação de modelos baseados na poda de árvores de decisão. Para um melhor en-

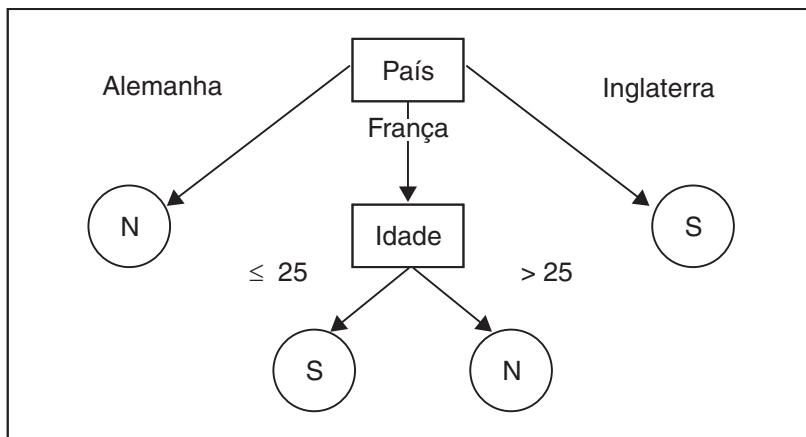
tendimento desta classe de algoritmos, é necessário compreender que existe uma correspondência entre regras e árvores de decisão, melhor comentada na próxima seção.

### **Transformações de Modelo de Conhecimento**

Muitas vezes, de forma a facilitar a análise de modelos de conhecimento, podem ser utilizados métodos de transformação sobre estes modelos. Esses métodos consistem basicamente na conversão da forma de representação do conhecimento de um modelo para outra forma de representação do mesmo modelo.

Um exemplo comum deste tipo de operação é a conversão de árvores de decisão em regras ou vice-versa. Uma árvore de decisão é um grafo em que cada nó não folha representa um predicado (condição) envolvendo um atributo e um conjunto de valores. Os nós folha correspondem à atribuição de um valor ou conjunto de valores a um atributo do problema. Nesta interpretação, cada caminho da árvore que parte do nó raiz e termina em um nó folha corresponde a uma regra da forma *SE <condições> ENTÃO <conclusão>*. As condições da regra correspondem a uma conjunção de todos os predicados existentes em um determinado caminho. A conclusão corresponde à atribuição do nó folha do caminho em questão. A Figura 3.5 ilustra uma árvore de decisão e o conjunto de regras correspondente.

<b>Sexo</b>	<b>País</b>	<b>Idade</b>	<b>Compra</b>
M	França	25	Sim
M	Inglaterra	21	Sim
F	França	23	Sim
F	Inglaterra	34	Sim
F	França	30	Não
M	Alemanha	21	Não
M	Alemanha	20	Não
F	Alemanha	18	Não
F	França	34	Não
M	França	55	Não



Se País=Alemanha Então Compra=Não

Se País=Inglaterra Então Compra=Sim

Se País=França e Idade  $\leq$  25 Então Compra=Sim

Se País=França e Idade  $>$  25 Então Compra=Não

**Figura 3.5.** Base de Dados original, Árvore de Decisão e Regras.

É importante enfatizar que a representação de modelos de conhecimento por meio de árvores de decisão é muito útil, pois, por se tratar de um diagrama, facilita a análise e validação do conhecimento.

### **Organização e Apresentação dos Resultados**

Conforme mencionado anteriormente, os modelos de conhecimento podem ser representados de diversas formas. Árvores, regras, gráficos em duas ou três dimensões, planilhas, tabelas e cubos de dados são muito úteis na representação de conhecimento. Em geral, as técnicas de visualização de dados estimulam a percepção e a inteligência humana, aumentando a capacidade de entendimento e associação de novos padrões. Oferecem, portanto, subsídios para a escolha dos passos seguintes a serem realizados no processo de KDD.

# Tarefas de KDD

## Considerações Iniciais

No contexto deste livro, uma tarefa de KDD equivale a uma operação de KDD que pertença à etapa de Mineração de Dados. Assim sendo, de forma análoga às operações de Mineração de Dados, as tarefas de KDD podem ser primárias ou compostas. Uma tarefa primária de KDD é aquela que não pode ser desmembrada em outras tarefas de KDD. Por outro lado, uma tarefa de KDD composta pode ser desmembrada em duas ou mais tarefas primárias de KDD.

Assim sendo, este capítulo descreve as tarefas de KDD mais comuns, primárias e compostas, na realização de processos de descoberta de conhecimento em bases de dados.

## Descoberta de Associações

A tarefa clássica de busca por regras de associação (também denominada de regras de associação ou regras associativas) foi introduzida em (Agrawal et al., 1993). Intuitivamente essa tarefa consiste em encontrar conjuntos de itens que ocorram simultaneamente e de forma freqüente em um banco de dados.

Como exemplo de uma aplicação da descoberta de regras de associação, considere o banco de dados da Tabela 4.1. Neste exemplo a tarefa consiste em descobrir produtos que sejam freqüentemente vendidos de forma conjunta.

Abaixo estão indicados dois exemplos de regras de associação. São implicações que indicam que a ocorrência do conjunto de itens do antecedente da regra tem propensão a levar à compra do conjunto de itens do conseqüente. A regra (1) indica que a compra de leite pode levar à compra de pão. Segundo a regra (2) a compra de pão e manteiga pode induzir a compra de café.

- (1) Leite → Pão
- (2) Pão ∧ Manteiga → Café

**Tabela 4.1** Relação das vendas de um Minimercado em um período

Transação	Leite	Café	Cerveja	Pão	Manteiga	Arroz	Feijão
1	Não	Sim	Não	Sim	Sim	Não	Não
2	Sim	Não	Sim	Sim	Sim	Não	Não
3	Não	Sim	Não	Sim	Sim	Não	Não
4	Sim	Sim	Não	Sim	Sim	Não	Não
5	Não	Não	Sim	Não	Não	Não	Não
6	Não	Não	Não	Não	Sim	Não	Não
7	Não	Não	Não	Sim	Não	Não	Não
8	Não	Não	Não	Não	Não	Não	Sim
9	Não	Não	Não	Não	Não	Sim	Sim
10	Não	Não	Não	Não	Não	Sim	Não

Uma alternativa para a representação de dados mostrada na Tabela 4.1 é denominada de formato *basket* (Agrawal et al., 1993). No formato basket, conforme pode ser observado na Tabela 4.2, a quantidade de itens que podem constar em uma transação não é limitada pelo número de atributos da relação.

**Tabela 4.2** Formato Basket da relação das vendas da Tabela 4.1

Transação	Item
1	Café
1	Pão
1	Manteiga
2	Leite
2	Cerveja
2	Pão
2	Manteiga
3	Café
3	Pão
3	Manteiga
4	Café
4	Leite
4	Pão
4	Manteiga
5	Cerveja

**Tabela 4.2 Continuação**

<b>Transação</b>	<b>Item</b>
6	Manteiga
7	Pão
8	Feijão
9	Arroz
9	Feijão
10	Arroz

Em (Agrawal et al., 1993), o autor introduziu o conceito de regras de associação com uma aplicação análoga à mencionada acima. No referido trabalho, foram consideradas grandes bases de dados operacionais no formato basket, contendo informações sobre cada transação de compra realizada ao longo de um determinado período. Uma transação de compra compreendia a relação de todos os produtos adquiridos por um cliente em um determinado instante. O principal objetivo daquele trabalho era desenvolver uma ferramenta que apresentasse um bom desempenho na identificação de regras de associação entre os vários tipos de produtos a partir das bases de dados mencionadas.

Formalmente, uma regra de associação é uma implicação da forma  $X \rightarrow Y$ , onde  $X$  e  $Y$  são conjuntos de itens tais que  $X \cap Y = \emptyset$ . Convém destacar que a interseção vazia entre antecedente e consequente da regras assegura que não sejam extraídas regras óbvias que indiquem que um item está associado a ele próprio.

*Transação* é o nome atribuído ao elemento de ligação existente em cada ocorrência de itens no banco de dados.

Uma associação é considerada *frequente* se o número de vezes em que a união de conjuntos de itens ( $X \cup Y$ ) ocorrer em relação ao número total de transações do banco de dados for superior a uma freqüência mínima (denominada *suporte mínimo*) que é estabelecida em cada aplicação. Busca-se, por meio do suporte, identificar que associações surgem em uma quantidade expressiva a ponto de ser destacada das demais existentes. No exemplo das Tabelas 4.1 e 4.2, as regras (1) e (2) possuem suporte 20% e 30%, respectivamente.

Uma associação é considerada *válida* se o número de vezes em que  $X \cup Y$  ocorrer em relação ao número de vezes que  $X$  ocorrer for superior a um valor denominado *confiança mínima*, e também estabelecido em cada aplicação. A medida de confiança procura expressar a qualidade de uma regra, indicando o quanto a ocorrência do antecedente da regra pode assegurar a ocorrência do consequente desta regra. As regras (1) e (2) possuem confiança 100% e 75%, respectivamente.

Denomina-se *K-itemset* a todo conjunto de itens com exatamente  $K$  elementos. As regras (1) e (2) apresentadas acima correspondem a 2-itemset e 3-itemset, respectivamente.

Assim sendo, a tarefa de Descoberta de Associações (Descoberta de Regras de Associação) pode ser definida formalmente como a busca por **regras de associação freqüentes e válidas** em um banco de dados, a partir da especificação dos parâmetros de suporte e confiança mínimos.

Os valores dos parâmetros de suporte e confiança mínimos devem ser especificados pelo especialista em KDD em conjunto com o especialista no domínio da aplicação.

Existem diversos algoritmos desenvolvidos especificamente para aplicação na tarefa de descoberta de associações, dentre eles: Apriori, DHP (Direct Hashing and Pruning), Partition, DIC (Dynamic Itemset Counting), Eclat, MaxEclat, Clique, MaxClique, Cumulate e EstMerge. Existem versões destes algoritmos para funcionamento em ambientes paralelos e distribuídos.

Todos os algoritmos mencionados, no entanto, possuem uma estrutura comum, inspirada na estrutura do algoritmo Apriori. Baseiam-se na propriedade de antimonotonicidade do suporte: “Um  $k$ -itemset somente pode ser freqüente se todos os seus  $(k-1)$ -subconjuntos forem freqüentes”. É fácil observar que o suporte de um conjunto de itens nunca pode crescer quando este é expandido para um conjunto com mais itens. Pode, na melhor hipótese, permanecer igual, ou simplesmente diminuir.

Considerando a importância do algoritmo Apriori na solução de tarefas de descoberta de associações, a sua estrutura encontra-se detalhada no Capítulo 5.

Um dos principais fatores de motivação para a tarefa de descoberta de regras associativas refere-se à possibilidade de incremento nas vendas de um determinado segmento comercial a partir de estratégias estabelecidas em função do conjunto de regras de associação extraído de grandes bases de dados históricas. Entre algumas das estratégias de venda passíveis de adoção a partir das regras de associação podem ser citadas: realização de promoções entre produtos; rearranjo da disposição dos produtos em prateleiras e gôndolas; reavaliação do rol de produtos oferecidos aos clientes, dentre outras.

## **Descoberta de Associações Generalizadas**

A descoberta de associações generalizadas é uma extensão da tarefa de descoberta de associações. A compreensão dessa tarefa depende da percepção de que é comum a existência de hierarquia e abstração entre conceitos. Por exemplo:

- O conceito roupa é uma generalização dos conceitos calça e camisa. Calça e camisa são tipos de roupa. O conceito roupa pertence a um nível de abstração que está acima do nível de abstração de calça e de camisa.
- Tênis e sapato são especializações do conceito calçado. Pertencem a um nível de abstração inferior ao nível de abstração de calçado.

Em muitas aplicações, torna-se difícil encontrar fortes associações (associações com alto suporte) entre itens de dados que pertençam a níveis inferiores ou mais primitivos de abstração de conceitos devido, principalmente, à pouca concentração de dados em espaços multidimensionais.

Assim sendo, a tarefa de descoberta de associações generalizadas não se resringe à busca por associações no nível mais primitivo de abstração. Leva em consideração a hierarquia conceitual eventualmente existente entre os itens de dados, de forma a identificar regras de associações que envolvam múltiplos níveis de abstração de conceitos. Exemplos de regras de associação generalizada:

camisa → sapato (nenhuma abstração)

roupa → sapato (uma abstração do lado esquerdo da regra)

camisa → calçado (uma abstração do lado direito da regra)

roupa → calçado (duas abstrações: uma do lado esquerdo e a outra do lado direito da regra)

Em geral, os algoritmos de descoberta de associações são adaptados de forma a incorporar estratégias para busca de associações generalizadas. Entre as principais estratégias de busca por regras de associação generalizadas podem ser citadas:

- Independente do Nível de Abstração – Consiste em percorrer todos os níveis da árvore de conceitos, sem utilizar conhecimento prévio acerca dos conjuntos de itens freqüentes para eliminar alternativas de busca. Essa estratégia demanda um maior volume de processamento.
- Máscara de Filtragem de um Item – Um item do  $i$ -ésimo nível hierárquico de conceitos é analisado, se e somente se, o seu nó filho do  $(i-1)$ -ésimo nível for freqüente. Em outras palavras, nessa abordagem, uma associação específica somente é analisada a partir de uma associação mais geral, que seja freqüente.
- Máscara de Filtragem de  $K$ -Itemsets – Um  $K$ -Itemset do  $i$ -ésimo nível hierárquico de conceitos é analisado, se e somente se, seus nós filhos ( $K$ -Itemsets) do  $(i-1)$ -ésimo nível forem freqüentes.

## Descoberta de Seqüências

A descoberta de seqüências também é uma extensão da tarefa de descoberta de associações. Essa tarefa considera o aspecto temporal entre as transações registradas no banco de dados.

Na descoberta de associações, os padrões a serem descobertos pertencem a cada transação. São denominados padrões intratransação. No caso da descoberta de seqüências, os padrões são denominados intertransação, pois diversas

transações devem ser analisadas em ordem cronológica de ocorrência. A busca por tais padrões é, evidentemente, mais complexa do que a busca por padrões intratransação.

Como exemplos de aplicações de descoberta de seqüências podem ser citados:

- A análise do histórico de itens comprados por consumidores ao longo de um período. A descoberta de quais itens os consumidores compram ao longo do tempo pode ser utilizada no marketing com a oferta de compras de forma direcionada aos interesses sazonais de cada consumidor ou de grupos de consumidores.
- A análise do histórico contendo a ordem dos acessos às páginas de um site pelos usuários da Internet. Essa análise pode permitir identificar páginas de interesse e atalhos de acesso a estas páginas. Com base nestas informações, os sites podem ser reestruturados de acordo com os interesses das pessoas que o acessam, tornando-os mais práticos e agradáveis.

Uma seqüência é uma lista ordenada de conjuntos de itens, caracterizada por objetos, rótulos temporais e eventos. Cada registro armazena ocorrências de eventos sobre um objeto em um instante de tempo particular. Notação:  $\langle S_1, S_2 \dots S_n \rangle$ , onde  $S_i$  é um conjunto de itens.

O conjunto de itens  $S_i$  é também chamado de elemento da seqüência. Cada elemento da seqüência é denotado por  $(x_1, x_2, \dots, x_m)$ , onde  $x_j$  é um item ou evento.

No exemplo das compras do mercado, apresentado na Tabela 4.3:

**Tabela 4.3 Relação das compras realizadas por cada cliente**

Identificação do Cliente	Identificação da Transação	Itens
1	114	A B
	232	B
	349	A B
2	150	A C
	386	A B C
	529	B
3	105	A
	307	B
	402	A
4	302	A B
	447	A
	596	B

- Os consumidores correspondem aos objetos e servem de ligação entre os diversos eventos temporais, que são as compras realizadas.
- Os itens comprados correspondem aos itens vinculados a cada evento temporal de compra, realizado por um consumidor.

Uma seqüênci  $\langle a_1 a_2 \dots a_n \rangle$  é uma **subseqüência (ou especialização)** de outra seqüênci  $\langle b_1 b_2 \dots b_n \rangle$  se existirem inteiros  $i_1 < i_2 < \dots < i_n$  tais que  $a_1 \subseteq b_{i1}$ ,  $a_2 \subseteq b_{i2}$ , ... e  $a_n \subseteq b_{in}$ . Exemplos:

- $\langle (3) (4, 5) (8) \rangle$  é uma subseqüência de  $\langle (7) (3, 8) (9) (4, 5, 6) (8) \rangle$ , pois  $(3) \subseteq (3, 8)$ ,  $(4, 5) \subseteq (4, 5, 6)$  e  $(8) \subseteq (8)$ .
- A seqüênci  $\langle (3) (5) \rangle$  não é uma subseqüência de  $\langle (3, 5) \rangle$  e vice-versa.

O **suporte (ou freqüência)** de uma seqüênci  $\alpha$  refere-se à proporção de objetos que contêm  $\alpha$ .

A seguir encontram-se citados alguns exemplos de seqüências observáveis na Tabela 4.3 e seus respectivos suportes:

Seqüênci	Suporte (%)
$\langle (A) \rangle$	100
$\langle (A) (A) \rangle$	100
$\langle (B) (A) \rangle$	75
$\langle (B) \rangle$	100
$\langle (A,B) \rangle$	75
$\langle (A) (B) \rangle$	100
$\langle (B) (B) \rangle$	75
$\langle (A,B) (B) \rangle$	75

Dado um limiar definido pelo usuário, denominado **suporte mínimo**, diz-se que uma seqüênci é **frequente** se esta ocorrer mais do que o suporte mínimo.

Uma *k*-seqüênci é uma seqüênci com exatamente *k* elementos.

Como exemplos de algoritmos para descoberta de seqüências podem ser citados os clássicos:

- GSP – Generalized Sequential Patterns
- MSDD – Multi Stream Dependency Detection
- SPADE – Sequential Pattern Discovery Using Equivalence Classes

De forma análoga aos algoritmos de descoberta de associações, os algoritmos acima se baseiam na propriedade de *antimonotonicidade* do suporte: “Uma *k*-seqüência somente pode ser freqüente se todas as suas  $(k-1)$ -subseqüências forem freqüentes”. O suporte de uma seqüência nunca pode crescer quando esta é expandida para uma seqüência com mais conjuntos de itens.

## Descoberta de Seqüências Generalizadas

A descoberta de seqüências generalizadas é uma extensão da tarefa de descoberta de seqüências. De forma análoga à descoberta de associações generalizadas, utiliza a hierarquia e a abstração entre conceitos eventualmente existentes em cada aplicação.

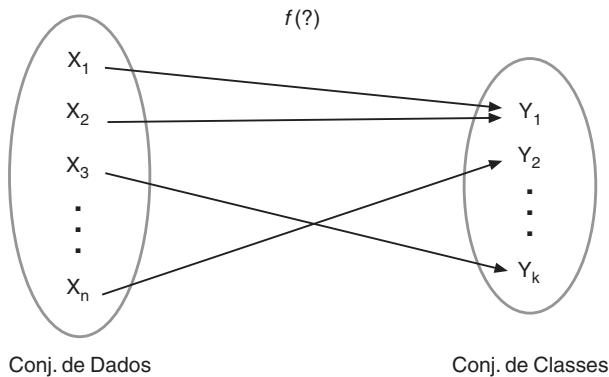
Considerando as abstrações entre os conceitos roupa, calça, camisa, tênis, sapato e calçado, e imaginando um banco de dados com estrutura similar àquela apresentada na Tabela 4.3, seguem alguns exemplos de seqüências generalizadas:

- <(roupa) (calçado)> envolve compras ocorridas em dois momentos distintos e duas generalizações.
- <(roupa) (sapato)> envolve compras ocorridas em dois momentos distintos e uma generalização.
- <(camisa) (sapato)> envolve duas compras em momentos distintos e nenhuma generalização.
- <(camisa, sapato)> envolve uma compra contendo dois itens e nenhuma generalização.
- <(roupa, calçado)> envolve uma compra contendo dois itens e duas generalizações.

## Classificação

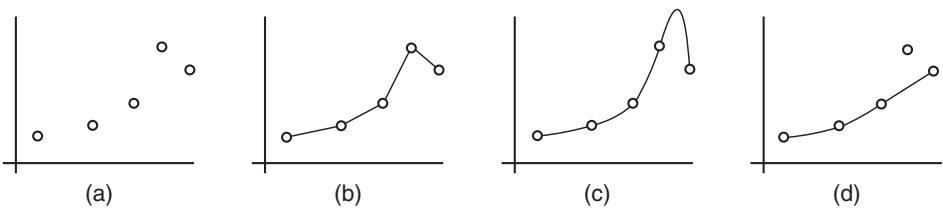
Uma das tarefas de KDD mais importantes e mais populares é a tarefa de classificação. Informalmente, conforme mostra a Figura 4.1, essa tarefa pode ser compreendida como a busca por uma função que permita associar corretamente cada registro  $X_i$  de um banco de dados a um único rótulo categórico,  $Y_j$ , denominado *classe*. Uma vez identificada, essa função pode ser aplicada a novos registros de forma a prever a classe em que tais registros se enquadram.

Com a finalidade de formalizar a tarefa de classificação, consideremos um par ordenado da forma  $(x, f(x))$ , onde  $x$  é um vetor de entradas  $n$ -dimensional e  $f(x)$  a saída de uma função  $f$ , desconhecida, aplicada a  $x$ . A tarefa de inferência indutiva consiste em, dada uma coleção de exemplos de  $f$ , obter uma função  $h$  que se aproxime de  $f$ . A função  $h$  é chamada de *hipótese* ou *modelo* de  $f$ . A Figura 4.2 apresenta geometricamente três hipóteses possíveis induzidas a partir do conjunto de exemplos fornecidos.



**Figura 4.1.** Associações entre registros de dados e classes.

Nos casos em que a imagem de  $f$  é formada por rótulos de classes, a tarefa de inferência indutiva é denominada classificação e toda hipótese  $h$  chamada de classificador. A identificação da função  $h$  consiste em um processo de busca no espaço de hipóteses  $H$ , pela função que mais se aproxime da função original  $f$ . Esse processo é denominado *aprendizado* (Russell e Norvig, 1995). Todo algoritmo que possa ser utilizado na execução do processo de aprendizado é chamado *algoritmo de aprendizado*. O conjunto de todas as hipóteses que podem ser obtidas a partir de um algoritmo de aprendizado  $L$  é representado por  $H_L$ . Cada hipótese pertencente ao  $H_L$  é representada por  $h_L$ .



**Figura 4.2.** Hipóteses de funções induzidas a partir dos exemplos de entradas e saídas.

A acurácia da hipótese  $h$  retrata a qualidade ou a precisão de  $h$  em mapear corretamente cada vetor de entradas  $x$  em  $f(x)$ . O conjunto de pares  $(x, f(x))$  utilizados na identificação da função  $h$  é denominado conjunto de treinamento. Por outro lado, o conjunto de pares  $(x, f(x))$  utilizados para avaliar a acurácia de  $h$  é denominado conjunto de testes. Assim, o algoritmo  $L$  pode ser interpretado como uma função tal que:

$L: T \rightarrow H_L$ , onde  $T$  é o espaço composto por todos os conjuntos de treinamento possíveis para  $L$ .

Cada algoritmo possui um *bias* indutivo que direciona o processo de construção dos classificadores. O *bias* indutivo de um algoritmo pode ser definido como o conjunto de fatores que coletivamente influenciam na seleção de hipóteses (Utgoff, 1986).

Em termos práticos, o *bias* de um algoritmo de aprendizado  $L$  afeta o processo de aprendizado de duas formas: restringe o tamanho do espaço de hipóteses  $H_L$ , e impõe uma ordem de preferência sobre as hipóteses em  $H_L$  (Bensusan, 1999).

Segundo o teorema *NFL* (*No Free Lunch Theorem*) (Wolpert, 1996), não existe um algoritmo de classificação que seja superior a todos os outros em qualquer problema de classificação. Isto significa que, a cada nova aplicação de KDD envolvendo a tarefa de classificação, os algoritmos disponíveis devem ser experimentados a fim de identificar aqueles que obtêm melhor desempenho.

Como exemplos nos quais a tarefa de classificação é aplicável, podem ser citados: análise de crédito, análise de risco em seguros, diagnóstico de doenças e prescrição de tratamento, análise de defeitos em equipamentos, entre inúmeros outros.

Conforme mencionado anteriormente, uma medida de desempenho de um classificador comumente utilizada é a *acurácia* ( $Acc(h)$ ), também denominada *precisão do classificador*:

$$Acc(h) = 1 - Err(h), \text{ onde:}$$

$Err(h)$  é denominada *taxa de erro* ou *taxa de classificação incorreta*:

$$Err(h) = \frac{1}{n} \sum_{i=1}^n \| y_i \neq h(i) \|$$

Na expressão acima:

- o operador  $\| E \|$  retorna 1 se a expressão  $E$  for verdadeira e 0, caso contrário;
- $n$  é o número de exemplos (registros da base de dados);
- $y_i$  é a classe real associada ao  $i$ -ésimo exemplo;
- $h(i)$  é a classe indicada pelo classificador para o  $i$ -ésimo exemplo.

Uma vez induzida uma hipótese (classificador) esta pode ser muito específica para o conjunto de treinamento utilizado. Caso este conjunto não seja suficientemente representativo, o classificador pode ter bom desempenho no conjunto de treinamento, mas não no conjunto de teste. Diz-se, neste caso, que o classificador ajustou-se em excesso ao conjunto de treinamento, ocorrendo um fenômeno denominado *overfitting*.

Por outro lado, quando o classificador ajusta-se muito pouco ao conjunto de treinamento, diz-se que ocorre um *underfitting*. Esse fenômeno costuma ocorrer em função de parametrizações inadequadas do algoritmo de aprendizado. Por exemplo, um número de neurônios insuficiente em uma rede neural, ou uma tolerância de erro excessivamente alta.

A completude de um classificador se refere à capacidade deste em classificar (apresentar uma resposta) a todos os exemplos da base de dados. A consistência, por outro lado, indica a capacidade do classificador em classificar corretamente os exemplos disponíveis no banco de dados.

A matriz de confusão de um classificador procura oferecer um detalhamento do desempenho do modelo de classificação, ao mostrar, para cada classe, o número de classificações corretas em relação ao número de classificações indicadas pelo modelo. Como mostrado na Tabela 4.4, os resultados são apresentados em duas dimensões: classes verdadeiras e classes preditas, para  $k$  classes distintas  $\{C_1, C_2, \dots, C_k\}$ . Cada elemento  $M(C_i, C_j)$  da matriz representa o número de exemplos da base de dados que tenham sido classificados na classe  $C_j$  e que pertençam efetivamente à classe  $C_i$ .

**Tabela 4.4** Matriz de Confusão de um Classificador – problema com  $k$  classes

Classes	Predita $C_1$	Predita $C_2$	...	Predita $C_k$
Verdadeira $C_1$	$M(C_1, C_1)$	$M(C_1, C_2)$	...	$M(C_1, C_k)$
Verdadeira $C_2$	$M(C_2, C_1)$	$M(C_2, C_2)$	...	$M(C_2, C_k)$
...	...	...	...	...
Verdadeira $C_k$	$M(C_k, C_1)$	$M(C_k, C_2)$	...	$M(C_k, C_k)$

Quando a matriz de confusão é aplicada a um problema com apenas duas classes, estas podem, por simplicidade, ser mapeadas em  $C_+$  (positiva) ou  $C_-$  (negativa). Assim, os exemplos recebem denominações especiais em função de sua classificação, conforme indica a Tabela 4.5.

**Tabela 4.5** Matriz de Confusão de um Classificador – problema com 2 classes

Classes	Predita $C_+$	Predita $C_-$
Verdadeira $C_+$	Verdadeiros Positivos	Falsos Negativos
Verdadeira $C_-$	Falsos Positivos	Verdadeiros Negativos

Conforme comentado no Capítulo 3, problemas de classificação podem apresentar-se com desigualdades na distribuição de registros pelas classes (prevalência). Em geral, os algoritmos de Mineração de Dados são fortemente influenciados pelas classes predominantes. O conceito de matriz de custos pode

ser utilizado em determinados algoritmos de aprendizado para compensar a prevalência. Nesta matriz, o custo, denotado por  $\text{Cost}(C_i, C_j)$  é um número que representa uma penalidade aplicada quando o classificador comete um erro ao rotular exemplos cuja classe verdadeira seja  $C_i$  como pertencentes à classe  $C_j$ , onde  $i, j = 1, 2, \dots, k$  e  $k$  é o número de classes.

Assim,  $\text{Cost}(C_i, C_j) = 0$  quando  $i = j$  e  $\text{Cost}(C_i, C_j) > 0$  quando  $i \neq j$ .

A fim de utilizar a matriz de custos, a equação da taxa de erro deve ser ajustada para:

$$\text{Err}(h) = \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n M(C_i, C_j) * \text{Cost}(C_i, C_j)$$

No Capítulo 5 encontram-se descritos vários algoritmos de aprendizado aplicáveis na tarefa de classificação, com destaque para: C4.5, Redes Neurais Back-Propagation, K-NN, Classificadores Bayesianos e Algoritmos Genéticos.

Por ora, para fins ilustrativos, consideremos a base de dados da Tabela 4.6 que contém dados sobre clientes e seu interesse por determinado tipo de literatura. O algoritmo de aprendizado C4.5, aplicado a esta base, geraria um classificador representado pelo conjunto de regras da Figura 4.3.

**Tabela 4.6** Clientes e suas compras em um tipo de literatura

Sexo	País	Idade	Compra
M	França	25	Sim
M	Inglaterra	21	Sim
F	França	23	Sim
F	Inglaterra	34	Sim
F	França	30	Não
M	Alemanha	21	Não
M	Alemanha	20	Não
F	Alemanha	18	Não
F	França	34	Não
M	França	55	Não

```

Se País=Alemanha Então Compra=Não
Se País=Inglaterra Então Compra=Sim
Se País=França e Idade ≤ 25 Então Compra=Sim
Se País=França e Idade > 25 Então Compra=Não

```

**Figura 4.3.** Conhecimento extraído pelo algoritmo C4.5 a partir dos dados da Tabela 4.6.

## Regressão

A tarefa de regressão compreende, fundamentalmente, a busca por funções, lineares ou não, que mapeiem os registros de um banco de dados em valores reais. Essa tarefa é similar à tarefa de classificação, sendo restrita apenas a atributos numéricos.

Como exemplo de aplicações de regressão, podemos citar: predição da soma da biomassa presente em uma floresta; estimativa da probabilidade de um paciente sobreviver, dado o resultado de um conjunto de diagnósticos de exames; predição do risco de determinados investimentos, definição do limite do cartão de crédito para cada cliente em um banco; dentre outros.

A regressão linear é a forma mais simples de regressão, onde a função a ser abstraída a partir dos dados é uma função linear. O número de variáveis, ou atributos, envolvido no problema varia de um caso para outro. Na situação mais simples, a regressão linear, denominada regressão linear bivariada, modela uma variável aleatória Y, chamada de variável dependente, como uma função linear de outra variável X, chamada variável independente:

$$Y = \alpha + \beta X,$$

Onde a variância da variável Y é assumida como constante, e  $\alpha$  e  $\beta$  são os coeficientes de regressão linear. Esses coeficientes podem ser obtidos a partir dos dados, por exemplo, pelo método dos mínimos quadrados, que busca minimizar o erro entre os dados reais e os dados estimados pela função. Assim, sendo  $n$  amostras dos dados:  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ , o método dos mínimos quadrados estima os coeficientes de regressão (Coeficientes de Regressão Linear) utilizando as seguintes fórmulas:

$$\beta = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\alpha = \bar{y} - \beta \bar{x}$$

Onde,  $y'$  e  $x'$  são as médias dos valores dos atributos  $y$  e  $x$ , respectivamente.

Consideremos como exemplo uma base de dados que possua informações sobre o tempo de experiência e o salário atual dos funcionários de uma empresa:

X (experiência em anos)	Y (salário anual em R\$ 1.000)
03	30
08	57
09	64
13	72
03	36
06	43
11	59
21	90
01	20
16	83

Aplicando o modelo de regressão linear aos dados acima, temos:

$$X' = 9,1$$

$$Y' = 55,4$$

$$\beta = \frac{(3 - 9,1)(30 - 55,4) + (8 - 9,1)(57 - 55,4) + \dots + (16 - 9,1)(83 - 55,4)}{(3 - 9,1)^2 + (8 - 9,1)^2 + \dots + (16 - 9,1)^2} = 3,7$$

$$a = 55,4 - (3,7)(9,1) = 21,7$$

$$\text{Portanto: } Y = 21,7 + 3,7 \cdot X$$

A regressão linear múltipla é uma extensão da regressão linear bivariada envolvendo mais de uma variável independente. Neste tipo de regressão, a variável dependente  $Y$  deve ser modelada como função linear de um vetor de características multidimensional. Assim, generalizando:

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k$$

O método dos mínimos quadrados também pode ser estendido para obter os  $(K + 1)$  coeficientes (Han & Kember, 2001).

Existem muitos problemas em que os dados não apresentam dependência linear entre si. Nesses casos, podem ser aplicadas técnicas de regressão não linear. Um exemplo deste tipo de técnica é a regressão polinomial que consiste, basicamente, em adicionar termos polinomiais ao modelo linear. Assim, aplicando-se transformações às variáveis, um modelo não linear pode ser convertido em um modelo linear, que pode, então, ser resolvido pelo método dos mínimos quadrados.

Estatística, Redes Neurais, dentre outras áreas, oferecem diversos métodos para implementação da tarefa de regressão (Michie et al., 1994).

## Sumarização

A tarefa de sumarização, também denominada descrição de conceitos, consiste em identificar e apresentar, de forma concisa e compreensível, as principais características dos dados contidos em um conjunto de dados.

Exemplos de aplicações envolvendo a tarefa de sumarização:

- Identificar as características dos assinantes de uma revista que residem na região sudeste do Brasil: “são em grande maioria, assinantes com faixa salarial de X reais, nível superior completo e que possuem residência própria”.
- Descrever o perfil dos meninos de rua da cidade do Rio de Janeiro: “são meninos que se encontram predominantemente na faixa etária X, cujos pais utilizam drogas e possuem na faixa de Y irmãos”.

Um conceito normalmente se refere a uma coleção de dados com pelo menos uma característica em comum. Por exemplo: Assinantes da revista XYZ na região Sudeste, meninos de rua na cidade do Rio de Janeiro, clientes inadimplentes, pacientes cardiolopatas, alunos de graduação, dentre muitos outros.

A sumarização dos dados não é uma simples enumeração dos dados. Busca gerar descrições para caracterização resumida dos dados e possivelmente comparação (discriminação) entre eles. Tais descrições são denominadas descrições de classe, quando o conceito a ser descrito se refere a uma classe de objetos.

A descrição de conceitos pode ser interpretada como uma generalização dos dados a partir das características mais relevantes dentre os registros analisados.

Um método muito utilizado na descrição de conceitos denomina-se Indução Orientada por Atributo. Esse método consiste da análise de medidas da Teoria da Informação, faz parte do algoritmo tradicional C4.5 e encontra-se descrito no Capítulo 5.

## Clusterização

A tarefa de *Clusterização*, também chamada de *Agrupamento*, é usada para partitionar os registros de uma base de dados em subconjuntos ou clusters, de tal

forma que elementos em um cluster compartilhem um conjunto de propriedades comuns que os distingam dos elementos de outros clusters. O objetivo desta tarefa é maximizar similaridade intracluster e minimizar similaridade intercluster. Diferente da classificação que tem rótulos predefinidos, a clusterização precisa automaticamente identificar os rótulos. Por esta razão, a clusterização é também denominada indução não supervisionada. A clusterização pode ser definida como uma das tarefas básicas da Mineração de Dados que auxilia o usuário a realizar agrupamentos naturais de registros em um conjunto de dados.

A análise de clusters envolve, portanto, a organização de um conjunto de padrões (usualmente representados na forma de vetores de atributos ou pontos em um espaço multidimensional – espaço de atributos) em clusters, de acordo com alguma medida de similaridade. Intuitivamente, padrões pertencentes a um dado cluster devem ser mais similares entre si (compartilham um conjunto de propriedades comuns) do que em relação a padrões pertencentes a outros clusters.

Em geral, o processo de clusterização requer que o usuário determine qual o número de grupos a ser considerado. Com base neste número, os registros de dados são então separados nos grupos de forma que registros similares fiquem nos mesmos grupos e registros diferentes em grupos distintos. Uma vez, tendo esses grupos, é possível fazer uma análise dos elementos que compõem cada um deles, identificando as características comuns aos seus elementos e, desta forma, podendo criar um rótulo que represente cada grupo.

A presença de dados distribuídos em um espaço de grande dimensionalidade (muitos atributos) dificulta a detecção de clusters. Os clusters podem estar imersos em algum subespaço do espaço de dados original.

Formalmente para o processo de clusterização, supõe-se a existência de  $n$  pontos de dados  $x_1, x_2, \dots, x_n$  tais que cada ponto pertença a um espaço  $d$  dimensional  $R^d$ . A tarefa de clusterização desses pontos de dados, separando-os em  $k$  clusters consiste em encontrar  $k$  pontos  $m_j$  em  $R^d$  de tal forma que a expressão:

$$\frac{\sum_i \min_j d^2(x_i, m_j)}{N}$$

seja minimizada, onde  $d^2(x_i, m_j)$  denota uma distância entre  $x_i$  e  $m_j$ . Os pontos  $m_j$  são denominados centróides ou médias dos clusters.

Informalmente, o problema descrito acima consiste em encontrar  $k$  centróides de clusters de tal forma que a distância entre cada ponto de dado e o centróide do cluster mais próximo seja minimizada. Pode-se perceber pelo exposto anteriormente que a tarefa de clusterização é um problema *NP-completo*.

Para que os algoritmos de clusterização possam efetuar sua tarefa é necessário que sejam utilizadas estruturas de dados capazes de armazenar os objetos a serem processados ou as informações sobre as relações entre estes. Algoritmos de clusterização que trabalham com dados armazenados na memória principal, normalmente, utilizam uma das seguintes estruturas de dados no seu processamento:

- Matriz de dados – As linhas representam cada um dos objetos a serem clusterizados e as colunas, os atributos ou características de cada objeto. Considerando  $n$  objetos cada qual com  $p$  atributos, obtém-se uma matriz  $n \times p$  como a matriz abaixo:

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1p} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2p} \\ x_{31} & x_{32} & x_{33} & \dots & x_{3p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{np} \end{bmatrix}$$

- Matriz de similaridade – Cada elemento da matriz representa a distância entre pares de objetos. Visto que a distância entre o objeto  $i$  e o objeto  $j$  é igual à distância entre o objeto  $j$  e o objeto  $i$ , não é necessário armazenar todas as distâncias entre os objetos. Aqui, considerando  $n$  objetos a serem clusterizados, obtém-se uma matriz quadrada de tamanho  $n \times n$  como a que segue:

$$D = \begin{bmatrix} 0 & & & & \\ d(1,2) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ d(n,1) & d(n,2) & d(n,3) & \dots & 0 \end{bmatrix}$$

O ponto  $d(i, j)$  representa a distância ou similaridade entre o objeto  $i$  e o  $j$ . Como as medidas de similaridade expressam o conceito de distância, estas são sempre números positivos. Quanto mais próximo de zero for  $d(i, j)$ , mais similares serão os objetos.

Quando um algoritmo que trabalha com matrizes de similaridade recebe uma matriz de dados, ele primeiro a transforma em uma matriz de similaridade antes de iniciar o processo de clusterização (Han & Kember, 2001).

Existem várias técnicas e métodos de clusterização. Entre os principais algoritmos de clusterização podem ser citados: K-Means, Fuzzy K-Means, K-Modes e K-Medoid. Esses algoritmos encontram-se comentados no Capítulo 5.

Cabe ressaltar que, para que se obtenha uma melhor clusterização dos dados, alguns requisitos que devem ser atendidos (Carlantonio, 2001) pelos algoritmos que implementam tal tarefa:

- Descobrir clusters com forma arbitrária – A forma dos clusters, considerando o espaço euclideano, pode ser esférica, linear, alongada, elíptica, cilíndrica, espiralada etc. Os métodos de clusterização baseados nas medidas de distância Euclideana ou Manhattan tendem a encontrar clusters esféricos de tamanho e densidade similares.
- Identificar clusters de tamanhos variados – Além da forma, alguns métodos tendem a fazer os clusters com tamanho homogêneo.
- Aceitar os diversos tipos de variáveis possíveis – Os métodos têm que ser capazes de lidar com variáveis contínuas discretas e nominais.
- Ser insensível à ordem de apresentação dos objetos – Um mesmo conjunto de objetos quando apresentado em diferentes ordenamentos deve conduzir aos mesmos resultados.
- Trabalhar com objetos com qualquer número de atributos (dimensões) – Os olhos humanos são bons para julgar a qualidade de clusters com até três dimensões. Os métodos devem manejar, com eficiência, objetos com altas dimensões e fornecer resultados comprehensíveis.
- Ser escalável para lidar com qualquer quantidade de objetos – Uma base de dados de grande porte pode conter milhões de objetos. Os métodos devem ser rápidos e escalonáveis em função do número de dimensões e da quantidade de objetos a serem clusterizados.
- Fornecer resultados interpretáveis e utilizáveis – As descrições dos clusters devem ser facilmente assimiladas. Em geral, os usuários esperam que os resultados dos clusters sejam interpretáveis, comprehensíveis e utilizáveis. Assim, é importante que os algoritmos utilizem representações simples.
- Ser robusto na presença de ruídos – A maioria das bases de dados do mundo real contém ruídos, dados desconhecidos ou errôneos. A existência deles não deve afetar a qualidade dos clusters obtidos.
- Exigir o mínimo de conhecimento para determinar os parâmetros de entrada – Os valores apropriados, são freqüentemente, desconhecidos e difíceis de determinar, especialmente, para conjuntos de objetos de alta dimensionalidade e de grande número de objetos. Em alguns métodos, os resultados do processo de clusterização são bastante sensíveis aos parâmetros de entrada.
- Aceitar restrições – Aplicações reais podem necessitar agrupar objetos de acordo com vários tipos de restrições. Assim sendo, os métodos devem encontrar grupos de dados cujas estruturas satisfaçam às restrições especificadas.
- Encontrar o número adequado de clusters – Encontrar o número natural de clusters de um conjunto de objetos é uma tarefa difícil. Muitos métodos precisam de um valor de referência, especificado pelo usuário.

Nenhuma técnica de clusterização atende a todos esses requisitos adequadamente, embora um trabalho considerável tenha sido feito para atender a cada aspecto separadamente (Carlantonio, 2001).

Os métodos de clusterização mais conhecidos e utilizados são os métodos por particionamento e os métodos hierárquicos.

Os algoritmos de clusterização por particionamento dividem a base de dados em  $k$  grupos, onde o usuário escolhe o número  $k$ . Inicialmente, estes algoritmos escolhem  $k$  objetos como sendo os centros dos  $k$  clusters. Os objetos são então divididos entre os  $k$  clusters de acordo com a medida de similaridade adotada, de modo que cada objeto fique no cluster que forneça o menor valor de distância entre o objeto e o centro do mesmo. Os algoritmos utilizam então, uma estratégia iterativa, que determina se os objetos devem mudar de cluster, fazendo com que cada cluster contenha somente elementos similares entre si.

Após a divisão inicial, há duas possibilidades na escolha do “elemento” que vai representar o centro do cluster, e que será a referência para o cálculo da medida de similaridade:

- Pode-se utilizar a média dos objetos que pertencem ao cluster em questão, também chamada de centro de gravidade do cluster. Essa é a abordagem conhecida como *k-means*, nome de um dos mais importantes algoritmos de clusterização.
- Pode-se também escolher como representante do cluster o objeto que se encontra mais próximo ao centro de gravidade do cluster. Essa abordagem é conhecida como *k-medoids*, e o elemento mais próximo ao centro é chamado de *medoid*.

Os métodos de clusterização por particionamento produzem agrupamentos simples. É importante destacar que esses algoritmos são efetivos se o número de clusters  $k$  puder ser razoavelmente estimado, se os clusters forem de forma convexa e possuírem tamanho e densidade similares.

Os métodos de clusterização por particionamento tentam fazer os  $k$  clusters tão compactos e separados quanto possível. Quando os clusters são compactos, densos e bastante separados uns dos outros, os métodos trabalham melhor; mas quando existem grandes diferenças nos tamanhos e geometrias dos diferentes clusters, os métodos por particionamento podem dividir desnecessariamente grandes clusters para minimizar a distância total calculada.

Os algoritmos de clusterização hierárquicos criam uma decomposição hierárquica da base de dados. A decomposição hierárquica é representada por um *dendrograma*, uma árvore que iterativamente divide a base de dados em subconjuntos menores até que cada subconjunto consista de somente um objeto.

Em tais hierarquias, cada nó da árvore representa um cluster da base de dados. O *dendrograma* pode ser criado de duas formas:

- Abordagem aglomerativa (bottom-up): Parte-se das folhas para a raiz. Coloca-se, inicialmente, cada objeto em seu próprio cluster (ou seja, todos os objetos estão separados), totalizando  $n$  clusters. Em cada etapa, calcula-se a distância entre cada par de clusters. Essas distâncias são, geralmente, armazenadas em uma matriz de similaridade. Então, são escolhidos 2 clusters com a distância mínima, juntando-os em seguida. Atualiza-se a matriz de similaridade. Esse processo continua até que todos os objetos estejam em um único cluster (o nível mais alto da hierarquia), ou até que uma condição de término ocorra (por exemplo, o número de clusters desejado tenha sido alcançado).
- Abordagem divisiva (top-down): Parte-se da raiz para as folhas. Inverte-se o processo por começar com todos os objetos em um único cluster. Em cada etapa, um cluster é escolhido e dividido em dois clusters menores. Esse processo continua até que se tenha  $n$  clusters ou até que uma condição de término aconteça.

## Previsão de Séries Temporais

Uma série temporal é um conjunto de observações de um fenômeno ordenadas no tempo. Podemos citar como exemplos de séries temporais: o consumo mensal de energia elétrica de uma casa, registrado durante um ano ou as vendas diárias de um produto no decorrer de um mês, dentre muitos outros.

A análise de uma série temporal é o processo de identificação das características, dos padrões e das propriedades importantes da série, utilizados para descrever em termos gerais o seu fenômeno gerador. Dentre os diversos objetivos da análise de séries temporais, o maior deles é a geração de modelos voltados à previsão de valores futuros.

Há quatro principais tipos de movimentos utilizados na caracterização de séries temporais:

- Movimentos de Tendência – Indicam a direção geral na qual o gráfico da série temporal se move ao longo do tempo.
- Movimentos Cíclicos – Referem-se às oscilações de uma curva, que podem ou não ser periódicas. Isso significa que os ciclos não precisam necessariamente seguir padrões exatos após intervalos de tempos iguais.
- Movimentos Sazonais – São movimentos que ocorrem devido a eventos que se repetem de tempos em tempos. Em geral, são padrões muito similares que ocorrem em determinadas épocas ou períodos. Por exemplo, aumento nas vendas de brinquedos nas semanas que antecedem ao dia das crianças.
- Movimentos Irregulares ou Randômicos – Esses movimentos são influenciados por eventos que ocorrem de forma aleatória. Por exemplo, chuvas intensas ou calor muito forte que influenciam na produção agrícola.

Uma série temporal pode ser interpretada como a decomposição dos quatro movimentos básicos mencionados acima.

Um método muito comum utilizado na determinação da tendência de uma série temporal é denominado **média móvel de ordem  $n$** . Esse método pode ser interpretado como um método de pré-processamento, pois consiste em suavizar os movimentos da série temporal, eliminando por meio da média dos valores, eventuais flutuações indesejadas nos valores da série:

$$\text{Cálculo do primeiro termo: } x_1 = \frac{y_1 + y_2 + \dots + y_n}{n},$$

$$\text{Cálculo do segundo termo: } x_2 = \frac{y_2 + y_3 + \dots + y_{n+1}}{n},$$

$$\text{Cálculo do terceiro termo: } x_3 = \frac{y_3 + y_4 + \dots + y_{n+2}}{n},$$

E assim por diante. Os  $y_i$  são os valores originais da série.

Uma variação deste método chama-se **média móvel ponderada de ordem  $n$** . Pode ser obtida aplicando-se a média aritmética ponderada aos valores da série. Os pesos são introduzidos com o objetivo de privilegiar mais os valores que ocupam determinadas posições na seqüência.

A média móvel perde os dados do início e do final das séries e pode algumas vezes gerar ciclos ou outros movimentos que não estejam presentes nos dados originais. Pode ainda, ser extremamente afetada pela presença de valores extremos.

O método dos mínimos quadrados é um método utilizado para modelar o comportamento de séries temporais. Consiste em procurar obter a curva mais próxima da série, ou seja, aquela que apresenta o menor desvio em relação à série original. O desvio é o somatório da diferença quadrática entre cada valor  $y_i$  do ponto  $(x_i, y_i)$  da curva e o seu correspondente na série.

A busca por similaridade em séries temporais envolve a identificação de seqüências de dados que apresentem pouca variação em relação ao padrão apresentado. São dois os tipos de busca por similaridade em séries temporais:

- Combinação de Subseqüências – Consiste em encontrar todas as seqüências de dados na série temporal que coincidam com um padrão apresentado.
- Aproximação de Seqüências – Busca encontrar as séries de dados que mais se aproximem da série em análise.

A previsão de séries temporais tem sido utilizada em diversos problemas do mundo real, reduzindo riscos gerados por incerteza e auxiliando o planejamento e a tomada de decisões, uma vez que a eficácia de uma decisão depende muitas vezes de eventos anteriores a ela mesma.

## **Detecção de Desvios**

A tarefa de detecção de desvios tem como objetivo identificar mudanças em padrões anteriormente percebidos. Sua aplicação vem crescendo de forma significativa nos últimos anos, sendo muito utilizada para detecção de fraudes em cartões de crédito, planos de saúde, arrecadação, dentre outras.

Diferentemente das demais tarefas de KDD em que a repetição de padrões é uma característica fundamental na busca por conhecimento, a detecção de desvios procura identificar padrões com pouca incidência e que sejam suficientemente distintos dos valores normalmente registrados.

O conceito de distância é também utilizado na detecção de desvios. Em geral, são especificados limiares de tolerância, de tal forma que, sempre que a distância entre o registro em análise e o padrão médio representativo da base de dados excede um destes limiares, tal registro é considerado como um desvio.

Para ilustração desta tarefa, considere um banco de dados de cartão de crédito que contenha os valores médios das compras realizadas pelos clientes em meses anteriores. Assim sendo, a ocorrência de compras cujo valor seja significativamente superior ao comportamento médio de consumo dos clientes pode ser um indicativo de roubo ou fraude, merecendo uma investigação específica.

A detecção de desvios pode ser on-line ou off-line. Na detecção de desvios on-line, mecanismos computacionais ativos devem monitorar a base de dados a fim de identificar a entrada de novos valores que sejam espúrios ou *outliers*. Somente novos dados são analisados. A tecnologia de Agentes Inteligentes é muito utilizada na detecção de desvios on-line. Na detecção de desvios off-line, o banco de dados é integralmente analisado na busca por *outliers*. Durante o processo de análise, não são incluídos novos dados na base.

## **Clusterização → Classificação**

A tarefa referenciada por “Clusterização → Classificação” é uma tarefa composta muito comum em aplicações de KDD. Encadeia as tarefas primárias de Clusterização e de Classificação.

Aplicável em situações em que os registros de dados não estejam enquadrados em classes predefinidas, esta tarefa consiste em:

- a) Agrupar os dados em função de sua similaridade. Utiliza-se, para tanto, algum método de clusterização de dados. Em geral, os métodos de clusterização incluem um novo atributo no conjunto de dados original, de forma que este novo atributo indique a qual cluster cada registro pertence.
- b) Cada rótulo de cluster passa a ser considerado como uma classe. No novo conjunto de dados, os registros estão enquadrados em classes. A partir de então, algoritmos de classificação podem ser aplicados de forma a gerar mo-

delos de conhecimento que possam prever a classificação de novos registros a partir das características dos dados.

Recomenda-se, para aplicação desta tarefa, que o usuário disponha de alguma ferramenta operacional de KDD que possua algoritmos de clusterização e classificação e que possa aplicá-los de forma integrada, evitando assim, eventuais dificuldades operacionais de manipulação dos conjuntos de dados original e final.

## **Clusterização → Sumarização**

A tarefa referenciada por “Clusterização → Sumarização” é do tipo composta e muito comum em aplicações de KDD. Encadeia as tarefas primárias de Clusterização e de Sumarização.

Aplicável em situações nas quais o conjunto completo de registros do banco de dados disponha de pouca similaridade entre seus elementos, esta tarefa consiste em:

- a) Agrupar os dados em função de sua similaridade. Utiliza-se, para tanto, algum método de clusterização de dados. Cada cluster passa a ser considerado isoladamente como um novo conjunto de dados. Para tanto, o conjunto de dados original é segmentado em tantos conjuntos quantos forem os clusters gerados.
- b) Cada novo conjunto de dados é então apresentado a um algoritmo de sumarização que descreve as principais características dos registros envolvidos naquele conjunto.

# Métodos de Mineração de Dados

## Considerações Iniciais

Embora neste capítulo estejam descritos diversos algoritmos e sua aplicação em Mineração de Dados, é importante destacar que vários deles são utilizados em muitos outros tipos de aplicações. Além disso, estes algoritmos apenas ilustram, mas não esgotam o universo de métodos de Mineração de Dados disponíveis.

A compreensão de determinados algoritmos requer um conhecimento prévio sobre algumas tecnologias tais como Redes Neurais, Lógica Nebulosa e Algoritmos Genéticos. Os Anexos II, III e IV apresentam um resumo dos principais fundamentos destas tecnologias.

Ao final deste capítulo, encontram-se indicadas quais tarefas de KDD podem ser implementadas pelos diversos métodos apresentados. Apenas para fins de organização da apresentação dos algoritmos, estes foram agrupados em uma taxonomia que considera o tipo de técnica/tecnologia utilizada por cada algoritmo em sua concepção.

Cada método de Mineração de Dados requer diferentes necessidades de pré-processamento (Morik, 2000). Tais necessidades variam em função do aspecto extensional da base de dados em que o método será utilizado. Em decorrência da grande diversidade de métodos de pré-processamento de dados, são muitas as alternativas possíveis de combinações entre métodos. A escolha dentre estas alternativas pode influenciar na qualidade do resultado do processo de KDD (Morik, 2000; Engels, 1996; Engels et al., 1997; Bernstein et al., 2002).

Os métodos de KDD, sendo os métodos de Mineração de Dados um caso particular, podem ser considerados *operadores* definidos a partir de *precondições* e *efeitos*. Uma *precondição* de um método de KDD é um *predicado* que estabelece um requisito que deve ser cumprido antes da execução do método. Um *efeito* de um método de KDD também é um *predicado* que descreve uma situação gerada após a aplicação do método. Um plano de ações de KDD válido é toda seqüência de métodos de KDD onde as *precondições* para execução de cada um dos métodos da seqüência sejam devidamente atendidas. Considerando os

métodos descritos em termos de precondições e de efeitos nas Tabelas 5.1 e 5.2, os planos de ações de KDD abaixo são exemplos de planos válidos.

P01 – Preenchimento Moda → Normalização Linear → Codificação Bin → Back-Prop

P02 – Preenchimento Moda → Normalização Máximo → Codificação Bin → Back-Prop

P03 – Preenchimento Moda → C4.5

**Tabela 5.1** *Back-Propagation e C4.5 representados por precondições e efeitos*

	<b>Descritores</b>	<b>Métodos</b>	
		<b>Back-Propagation</b>	<b>C4.5</b>
Precondições	Atributos Categóricos	Não	Não Importa
	Atributos Quantitativos	Sim	Não Importa
	Dados Não Normalizados	Não	Não Importa
	Valores Ausentes	Não	Não
Efeitos	Transparência	Não	Sim
	Representação do Conhecimento	Matriz de Pesos	Regras de Produção

**Tabela 5.2** *Exemplos de Precondições e Efeitos de Métodos de Preprocessamento*

	<b>Descritores</b>	<b>Métodos de Preprocessamento</b>		
		<b>Normalização Linear/Max</b>	<b>Preenchimento Moda</b>	<b>Codificação Binária</b>
Precondições	Atributos Categóricos	–	–	Sim
	Atributos Quantitativos	Sim	–	–
	Dados Não Normalizados	Sim	–	–
	Valores Ausentes	Não	Sim	Não
Efeitos	Atributos Categóricos	–	–	Não
	Atributos Quantitativos	–	–	Sim
	Dados Não Normalizados	Não	–	–
	Valores Ausentes	–	Não	–

## Métodos Baseados em Redes Neurais

Diversos modelos de Redes Neurais podem ser utilizados na implementação de métodos de Mineração de Dados. Esta seção tem como objetivo ilustrar, sem ter a pretensão de esgotar, o potencial de aplicação da tecnologia de Redes Neurais em tarefas de Mineração de Dados. Para os leitores não familiarizados com a tecnologia de Redes Neurais, recomenda-se uma leitura prévia do Anexo II, que apresenta uma introdução aos conceitos básicos mínimos necessários ao entendimento desta seção.

Classificação, Regressão, Previsão de Séries Temporais e Clusterização são exemplos de tarefas de Mineração de Dados que podem ser implementadas por métodos de Redes Neurais. Alguns modelos de Redes Neurais podem ser aplicados em mais de um tipo de tarefa de Mineração.

A topologia da rede neural varia em função do problema e da representação adotada para os dados. Em geral, em aplicações de Mineração de Dados, a camada de entrada do modelo neural recebe os dados pré-processados de cada registro de um banco de dados. A rede processa esses dados produzindo uma saída cuja natureza também varia em função da aplicação.

Em redes neurais com aprendizado supervisionado, a entrada corresponde aos atributos preditivos enquanto a saída do modelo corresponde ao atributo objetivo do problema. Assim sendo, o algoritmo de aprendizado pode estimar o erro, ou distância, entre a saída produzida pela rede e a saída desejada. Em função do erro calculado, o algoritmo ajusta os pesos das conexões da rede a fim de tornar a saída real tão próxima quanto seja possível da saída desejada. Modelos neurais deste tipo são muito úteis em geral para reconhecimento de padrões e, em particular, para tarefas de Mineração de Dados que envolvam previsão.

Por outro lado, redes neurais com aprendizado não supervisionado são adequadas para tarefas que envolvem descrição dos dados, como, por exemplo, a tarefa de Clusterização. Nestes casos, não há saída desejada.

A seguir estão descritos alguns algoritmos de aprendizado e indicadas a tarefas de Mineração de Dados em que estes algoritmos são aplicáveis.

### Back-Propagation

O algoritmo Back-Propagation, também conhecido como algoritmo de retro-propagação do erro, é um algoritmo de aprendizado supervisionado, cuja aplicação é adequada a tarefas de Mineração de Dados tais como Classificação, Regressão ou Previsão de Séries Temporais.

Esse algoritmo tem como objetivo minimizar a função de erro entre a saída gerada pela rede neural e a saída real desejada, utilizando o método do gradiente descendente.

A topologia de uma rede neural não linear cujo comportamento seja codificado pelo algoritmo Back-Propagation é, em geral, composta de uma camada de

entrada, uma camada de saída e um número arbitrário de camadas intermediárias. Cada neurônio de uma camada, com exceção da camada de entrada, encontra-se conectado a todos os neurônios presentes na camada imediatamente anterior à sua.

A fase de treinamento do algoritmo Back-Propagation desencadeia duas etapas, para cada padrão de entrada apresentado: processamento para a frente e processamento para trás.

No processamento para a frente, que é primeira etapa, o fluxo do processamento parte das unidades na camada de entrada em direção às unidades na camada de saída. Nesta etapa, os pesos sinápticos permanecem inalterados.

Os neurônios da camada de entrada recebem os valores do padrão de entrada. Em seguida, a função de ativação é aplicada, produzindo a saída de cada neurônio desta camada.

Os valores de entrada dos neurônios nas demais camadas são calculados pela regra de propagação, em geral o produto escalar:

$$\text{net}_j(n) = \sum w_{ji}(n)y_i(n)$$

Onde,

$w_{ji}(n)$  corresponde aos pesos das conexões que chegam ao neurônio  $j$  na  $n$ -ésima iteração.

$y_i(n)$  é o  $i$ -ésimo sinal de entrada do neurônio  $j$  na  $n$ -ésima iteração.

Em todos os neurônios, a função de ativação  $\phi$  deve ser diferenciável. A função de ativação é aplicada ao potencial de ativação de cada neurônio ( $\text{net}_j(n)$ ):

$$y_j(n) = \phi_j(\text{net}_j(n))$$

É muito comum a utilização da função logística sigmoidal:

$$y_j(n) = 1 / (1 + e^{-\text{net}_j(n)})$$

Uma vez geradas as saídas dos neurônios da camada de saída da rede, o algoritmo Back-Propagation inicia a segunda etapa do treinamento para o padrão apresentado. Neste momento, o sinal de erro produzido pelo neurônio  $j$  da camada de saída na iteração  $n$  é calculado por:

$$e_j(n) = d_j(n) - y_j(n)$$

O gradiente local de cada neurônio  $j$  da camada de saída no instante  $n$  é calculado pela expressão:

$$\delta_j(n) = e_j(n)\varphi'_j(\text{net}_j(n))$$

Esta etapa prossegue, passando os sinais de erro da direita para a esquerda, calculando o gradiente local associado a cada neurônio não pertencente à camada de saída (1), e ajustando os pesos das conexões (2) associadas a ele.

$$\delta_j(n) = \varphi'_j(\text{net}_j(n)) \sum \delta_k(n) w_{kj}(n) \quad (1)$$

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n) \quad (2)$$

Onde:

$\Delta w_{ji}(n)$  é o ajuste a ser adicionado ao peso da conexão entre o neurônio  $j$  e o neurônio  $i$ .

$\eta$  é uma constante denominada taxa de aprendizado, definida pelo usuário antes do início do treinamento da rede.

$\delta_j(n)$  é o gradiente local associado ao neurônio  $j$ .

$y_i(n)$  é o  $i$ -ésimo sinal de entrada do neurônio  $j$ .

Em geral são duas as condições de parada do treinamento de uma rede back-propagation: um número máximo de iterações definido pelo usuário ou a convergência da rede. Diz-se que uma rede neural converge quando o somatório dos erros dos neurônios da camada de saída tende a zero.

Ao final do treinamento de uma rede, os pesos das conexões entre os neurônios representam o conhecimento descoberto pela rede. Esse conjunto pode então ser utilizado pela rede para processar novos casos e, em função do conhecimento descoberto, apresentar resultados.

Uma vez que o conhecimento armazenado pela matriz de pesos de uma rede neural treinada não pode ser interpretado diretamente pelo homem, a qualidade do desempenho dessa rede deve ser avaliada por meio de experimentos a fim de verificar a adequação deste conhecimento na implementação da tarefa desejada.

## Kohonen

O mapa *Kohonen* pertence à classe das Redes Neurais Auto-organizáveis (também denominados Mapas Auto-organizáveis). Em uma Rede Neural Auto-organizável o treinamento é não supervisionado, geralmente baseado em uma forma de competição entre os elementos processadores. Entre as principais aplicações das Redes Auto-organizáveis estão:

- Tarefa de Clusterização – Tarefa na qual os dados de entrada devem ser agrupados em conjuntos que agregam padrões semelhantes.

- Detecção de Regularidades – Modelo em que o sistema deve extrair as características relevantes dos padrões de entrada.

O método de aprendizado mais comum nas Redes Auto-Organizáveis é denominado aprendizado por competição (*Competitive Learning*). Consiste em uma forma de aprendizado que divide o conjunto de padrões de entrada em grupos inerentes aos dados. Para tanto, esse método considera, em sua abordagem mais simples, que os neurônios de saída da rede competem entre si, resultando em apenas um neurônio vencedor (com maior ativação).

Em geral, Redes Neurais baseadas em Aprendizado Competitivo possuem as seguintes características:

- Topologia: Os neurônios são dispostos em uma única camada, estruturada em uma ou duas dimensões. Todos os neurônios desta camada recebem todos os valores dos padrões de entrada.
- Regra de Propagação: Produto escalar entre as entradas do elemento processador e os respectivos pesos associados às conexões afluentes ao referido elemento.
- Função de Ativação: Função Degrau, aplicada apenas ao neurônio vencedor, ou seja, somente o neurônio com maior potencial de ativação ( $net_j$ ) é submetido à função de ativação.
- Regra de Aprendizado:  $\Delta W_{ik} = \alpha(X_i - W_{ik})$  – Segundo essa regra, o aprendizado é não supervisionado, e ocorre de tal forma que a direção de atualização dos pesos minimiza a diferença entre o vetor de pesos e o vetor de entrada do elemento processador vencedor. O funcionamento dessa regra requer que os vetores  $X$  e  $W$ , estejam normalizados, sendo que somente os pesos do neurônio vencedor são atualizados. Essa regra de aprendizado tem como principal objetivo ajustar os pesos de tal forma que vetores de entrada similares ativem sempre o mesmo neurônio.

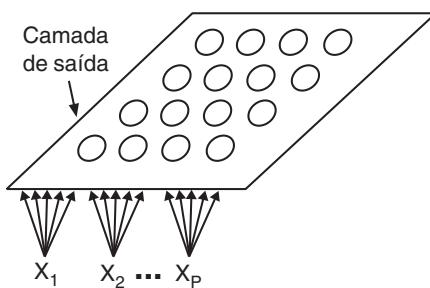
A restrição quanto à atualização dos pesos apenas do neurônio vencedor justifica-se porque, no caso de vetores normalizados, o processador vencedor é o que possui o vetor  $W$  mais próximo do vetor de entrada. A atualização dos pesos do processador vencedor faz com que o vetor  $W$  fique ainda mais próximo do vetor de entrada corrente. Desta forma, o vetor  $W$  fica cada vez mais representativo do grupo ao qual o padrão de entrada corrente pertence.

O parâmetro da regra de aprendizado deve ser sempre inferior a 1, sendo adaptativo e decrescente com o tempo. Desta forma, os pesos são ajustados para representar a classe ou grupo de padrões semelhantes e não apenas um padrão de entrada específico.

Os pesos devem ser inicializados de forma distribuída em relação à densidade dos vetores de entrada. Desta forma, pretende-se conseguir que maior dispo-

nibilidade de pesos na região com mais amostras de vetores de entrada. A inicialização aleatória dos pesos pode gerar processadores inúteis, que nunca vençam a competição, assim como processadores que representem mais de uma classe, e, eventualmente, mais de um processador para representar uma única classe.

No mapa de Kohonen os neurônios da camada de saída são organizados em uma estrutura bidimensional, conforme mostra a Figura 5.1. A atualização dos pesos é feita não somente para o neurônio vencedor  $k$ , mas também para os neurônios pertencentes a determinada vizinhança  $NE_k(t)$ , que deve ser reduzida com o tempo  $t$ . A Figura 5.2 ilustra o processo de redução da vizinhança  $NE_k(t)$ , na medida em que o tempo passa.



**Figura 5.1.** Mapa de Kohonen.

## Métodos Baseados em Algoritmos Genéticos

Genericamente os Algoritmos Genéticos são extremamente úteis em problemas complexos que envolvam otimização. Em particular, podem ser aplicados a diversas tarefas de Mineração de Dados. Recomenda-se ao leitor não familiarizado com Algoritmos Genéticos uma leitura prévia do Anexo III para uma introdução sobre o assunto.

Para ilustrar o potencial desta tecnologia em Mineração de Dados, considere inicialmente um exemplo em que desejamos utilizar um Algoritmo Genético para extrair regras que descrevam as características dos clientes de uma empresa de telecomunicações que pertençam a um determinado grupo (tarefa de sumarização). Considere ainda que cada cromossoma representa uma regra e que cada gene representa um atributo do banco de dados.

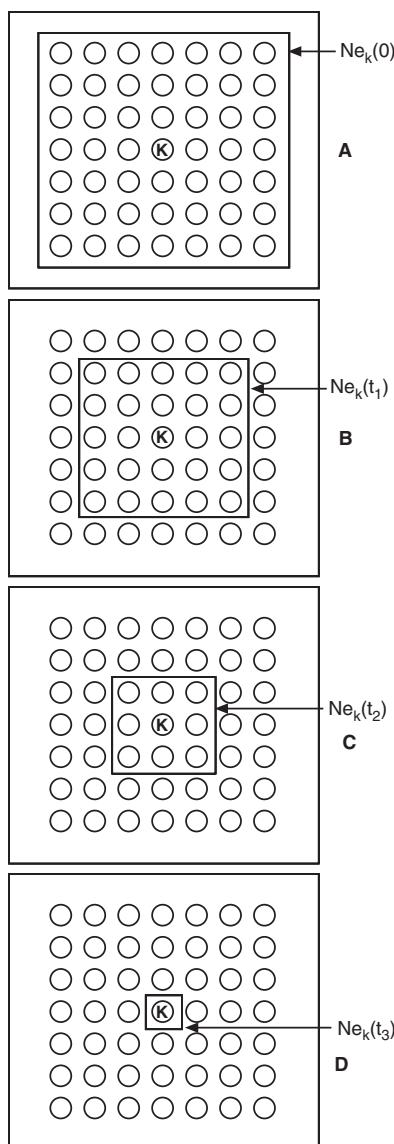
Regra:

- Se receita\_serviço 1 (Voz) entre 5000 e 7000
- e receita\_serviço 2 (Fax) entre 7000 e 8000
- e código\_atividade = 13 (Ind. Mat. Elétrico Eletrônico  
e Comunicação)
- e  $10 < \#_{\_Filiais} < 50$  (Número de Filiais entre 10 e 50)

e             $\#_{\text{Empreg}} > 100$  (Número de Empregados acima de 100)  
**Então**   Cliente do Grupo X

Cromossoma correspondente à regra acima:

Receita Serviço1 5000<R\$<7000	Receita Serviço2 7000<R\$<8000	CódAtividade = 13	10 < #_Filiais < 50	#_Empreg > 100



**Figura 5.2.** Processo de Redução da Vizinhança.

Cada regra pode ser avaliada com relação à sua acurácia e à sua abrangência. Para uma apresentação dos conceitos de acurácia e abrangência, consideremos uma regra  $R$  da forma: Se  $X$  então  $Y$ , onde  $X$  e  $Y$  representam predicados envolvendo atributos da base de dados.

A acurácia da regra acima pode ser traduzida como a proporção entre a quantidade de registros da base de dados que satisfazem ao antecedente e ao consequente da regra em relação à quantidade de registros que satisfazem ao antecedente. Matematicamente:

$$Acc(R) = \frac{|XeY|}{|X|}$$

A abrangência, por outro lado, expressa a proporção entre a quantidade de registros da base de dados que satisfazem às condições e ao consequente da regra em relação à quantidade de registros que satisfazem ao consequente.

$$Acc(R) = \frac{|XeY|}{|Y|}$$

Quanto maiores forem a acurácia e a abrangência de uma regra, melhor o indivíduo que representa esta regra. Essas medidas podem ser utilizadas para avaliar a qualidade das regras evoluídas pelo Algoritmo Genético.

O exemplo abaixo ilustra conceitualmente o cruzamento genético entre duas regras, gerando duas novas regras. Consideremos como operador de cruzamento o *crossover* de um ponto cujo ponto de corte tenha ocorrido na posição 2.

Pais:

Receita Serviço1 5000<R\$<7000	Receita Serviço2 7000<R\$<8000	CódAtividade = 13	10 < #_Filiais < 50	#_Empreg > 100
-----------------------------------	-----------------------------------	----------------------	---------------------	----------------

Receita Serviço1 3000<R\$<4000	Receita Serviço2 8500<R\$<9500	CódAtividade = 12	5 < #_Filiais < 10	#_Empreg > 100
-----------------------------------	-----------------------------------	----------------------	--------------------	----------------

Filhos:

Receita Serviço1 5000<R\$<7000	Receita Serviço2 7000<R\$<8000	CódAtividade = 12	5 < #_Filiais < 10	#_Empreg > 100
-----------------------------------	-----------------------------------	----------------------	--------------------	----------------

Receita Serviço1 3000<R\$<4000	Receita Serviço2 8500<R\$<9500	CódAtividade = 13	10 < #_Filiais < 50	#_Empreg > 100
-----------------------------------	-----------------------------------	----------------------	---------------------	----------------

## Rule Evolver

O Rule Evolver, ferramenta desenvolvida pelo Laboratório de Inteligência Computacional Aplicada do Departamento de Engenharia Elétrica da Pontifícia Universidade Católica do Rio de Janeiro, baseia-se em um modelo de Algoritmo Genético que pressupõe a evolução de regras cuja estrutura geral encontra-se descrita em (1). Os predicados  $C_i, i=1, \dots, n$  são condições envolvendo atributos denominados preditivos e a conclusão da regra envolve um atributo denominado objetivo que contém um valor fixo, não manipulado pelo Algoritmo Genético. Os atributos preditivos podem ser classificados quanto à natureza de seu domínio em atributos quantitativos ou atributos categóricos.

$$\underline{\text{SE}} \ C_1 \ \underline{\text{E}} \ C_2 \ \underline{\text{E}} \dots \ \underline{\text{E}} \ C_n \ \underline{\text{ENTÃO}} \ \text{Conclusão} \quad (1)$$

A representação utilizada pelo Rule Evolver, descrita na Figura 5.3, pressupõe que cada gene seja associado a um atributo preditivo e que compreenda dois valores reais: um valor inferior e um valor superior. Para atributos quantitativos, tais valores denotam o intervalo em que o respectivo atributo se encontra inserido. Para atributos categóricos, apenas o campo referente ao valor inferior é utilizado.



**Figura 5.3.** Representação do Cromossoma – Rule Evolver.

Para um melhor entendimento, suponha um atributo categórico com alfabeto de símbolos possíveis  $\{v_1, v_2, \dots, v_k\}$  de cardinalidade  $k$ . No Rule Evolver, cada valor desse atributo contido em um cromossoma qualquer deve preservar a representação real, adaptando-se apenas a interpretação desse valor para caracterização dos símbolos do atributo categórico efetivamente representados. Tal interpretação compreende a conversão do valor real para uma representação binária em que cada dígito indica a presença (1) ou ausência (0) do símbolo correspondente. Por exemplo, seja um atributo categórico denominado tipo de residência com alfabeto de cardinalidade 4 {própria, alugada, funcional, parentes}. A Tabela 5.3 indica o mapeamento entre os valores inteiros possíveis para o atributo e os símbolos efetivamente representados. Tal mapeamento utiliza o operador lógico de disjunção “ou”.

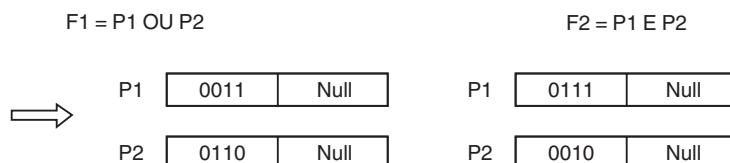
Os operadores genéticos disponíveis no sistema são:

- Crossover de um ponto, crossover de dois pontos, crossover uniforme, crossover de média, mutação simples, e mutação “don’t care”. São operadores aplicáveis aos genes referentes a atributos quantitativos.

**Tabela 5.3** Mapeamento entre os valores do atributo tipo de residência e os símbolos respectivamente representados

Alelo	Decodificação	Tipo de Residência
0	0000	Não informada (null)
1	0001	Própria
2	0010	Alugada
3	0011	Própria ou alugada
...	...	...
15	1111	Própria ou alugada ou parente ou funcional (don't care)

- b) Crossover lógico e mutação lógica são operadores genéticos voltados exclusivamente para aplicação nos genes referentes a atributos categóricos. Conforme exemplificado na Figura 5.4a, o crossover lógico implementa duas operações lógicas bit a bit entre as seqüências binárias correspondentes aos atributos categóricos dos cromossomas pais: uma operação “e” e uma operação “ou”. A mutação lógica, exemplificada na Figura 5.4b, implementa uma operação de inversão dos bits das seqüências binárias correspondentes aos atributos categóricos.



**Figura 5.4a.** Crossover Lógico.



**Figura 5.4b.** Mutação Lógica.

O critério de seleção de operadores baseia-se na interpolação linear das probabilidades de ocorrência de cada operador, ajustada em função da geração corrente. Tais probabilidades são parametrizadas pelo usuário do sistema conforme a aplicação.

O Rule Evolver dispõe de 10 funções de avaliação passíveis de escolha, pelo usuário da ferramenta. Algumas delas utilizam as medidas de acurácia e abrangência das regras. Conforme experimentos realizados, pode-se constatar uma melhor adequação de determinadas funções de avaliação para determinados tipos de base de dados.

Embora a utilização natural do Rule Evolver seja em tarefas de summarização, as regras evoluídas pela ferramenta para cada grupo podem ser reunidas em uma única base de conhecimento que pode ser utilizada com o propósito de implementar tarefas de classificação.

Mais informações sobre o Rule Evolver, assim como uma versão da própria ferramenta, podem ser obtidas no endereço [www.ica.ele.puc-rio.br](http://www.ica.ele.puc-rio.br).

### Exemplo de Algoritmo Genético para a Tarefa de Clusterização (Gonçalves e Pacheco, 2000)

O número de possibilidades que se podem agrupar  $n$  objetos em  $k$  grupos é dado pela seguinte expressão:

$$N(n, k) = \frac{1}{k!} \sum_{i=0}^k (-1)^i \binom{k}{i} (k-i)^n \quad (1)$$

Se tomarmos  $n=25$  objetos e  $k=5$  grupos teremos  $N(25,5) = 2.436.684.974.110.751$  maneiras de agrupar os 25 objetos em cinco grupos.

Se o número de *clusters* for desconhecido, a complexidade do espaço de busca aumenta consideravelmente. Neste caso, podemos ter os objetos agrupados em  $\sum_{i=1}^n N(n,k)$  possibilidades.

Para 25 objetos existem mais de  $4 \times 10^8$  possibilidades de *clusters*. Claramente é impraticável para um algoritmo de busca exaustiva encontrar a melhor solução nesse espaço de busca em um período de tempo computacionalmente viável. Podemos perceber pelo exposto a aplicabilidade de Algoritmos Genéticos neste tipo de tarefa.

Existem diversas possibilidades de representação de cromossomas em problemas de Clusterização.

Supondo os elementos  $x_1, x_2, x_3, x_4, x_5, x_6$ , agrupados em 2 clusters da seguinte forma:  $\{(x_1, x_3, x_6), (x_2, x_4, x_5)\}$ , podemos ter as seguintes representações de cromossomas mostradas na Figura 5.5.

<table border="1"><tr><td>1</td><td>2</td><td>1</td><td>2</td><td>2</td><td>1</td></tr></table>	1	2	1	2	2	1	<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td></tr></table>	1	0	1	0	0	1	0	1	0	1	1	0	<table border="1"><tr><td>1</td><td>3</td><td>6</td><td>7</td><td>2</td><td>4</td><td>5</td></tr></table>	1	3	6	7	2	4	5	<table border="1"><tr><td>1</td><td>2</td><td>3</td><td>6</td><td>4</td><td>5</td></tr></table>	1	2	3	6	4	5
1	2	1	2	2	1																													
1	0	1	0	0	1																													
0	1	0	1	1	0																													
1	3	6	7	2	4	5																												
1	2	3	6	4	5																													
(a)	(b)	(c)	(d)																															

**Figura 5.5.** Representações de cromossomas para o cluster  $\{(x_1, x_3, x_6), (x_2, x_4, x_5)\}$ . (a) grupamento de número; (b) matriz; (c) permutação com o caractere 7 representado como o separador dos dois clusters; (d) permutação greedy.

A representação por grupamento de número é um cromossoma cujo número de genes é a quantidade de elementos a serem agrupados, e o valor de cada gene representa o *cluster* ao qual cada elemento pertence.

A representação por matriz utiliza uma matriz  $n \times k$  para representar o cromossoma; onde cada linha corresponde ao *cluster* e cada coluna está associada a cada elemento. Cada coluna contém exatamente um “1”, pois cada elemento só pode pertencer a um único *cluster*.

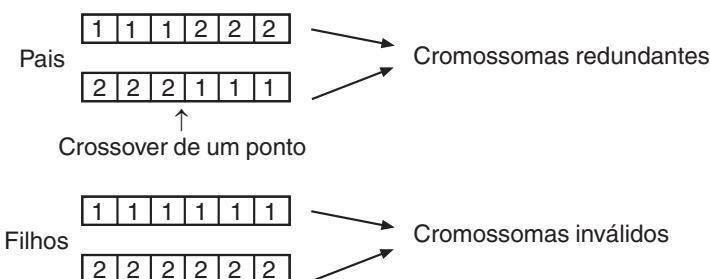
Permutação com separador é a representação do cromossoma que utiliza inteiros  $n+1$  até  $n+k+1$  (ou outros separadores apropriados) para indicar os *clusters* na permutação. Cada gene do cromossoma pode ser o número do elemento, ou um separador, e o tamanho do cromossoma é dado por  $n+k+1$ , onde  $n$  é o número de elementos e  $k$  o número de *clusters*.

A permutação *greedy* necessita de uma busca local no cromossoma para determinar a qual *cluster* o elemento pertence. Essa permutação utiliza os  $k$  primeiros genes do cromossoma para semear  $k$  *clusters*. Os genes restantes do cromossoma, na forma em que eles aparecem na permutação, são adicionados aos *clusters* com o centróide mais próximo.

Para o problema de Clusterização de  $k$  grupos, qualquer cromossoma que não represente os  $k$  grupos é uma representação inválida. Dessa forma: a representação por grupamento de número que não inclui todos os números de *clusters* como valores para os genes é inválida; a representação do cromossoma por matriz que possui uma linha contendo somente “0’s” é inválida; a representação do cromossoma por permutação com separador, com separadores adjacentes, ou com separadores no primeiro ou no último gene é inválida; e a representação do cromossoma por permutação *greedy* que contenha valores repetidos também é uma representação inválida.

O operador de *crossover* é utilizado para transferir material genético de uma geração para outra.

É necessário que o cromossoma gerado pelo processo de *crossover* seja verificado, a fim de se evitar cromossomas inválidos. Além de cromossomas inválidos, existem também cromossomas redundantes, isto é, cromossomas com codificações diferentes que representam na verdade a mesma solução. A Figura 5.6 mostra esses dois tipos de problemas.

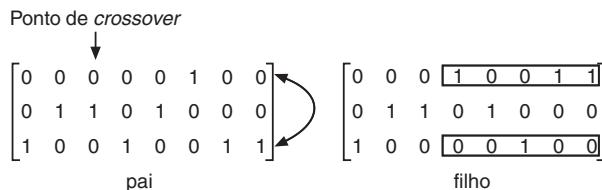


**Figura 5.6.** Representação do crossover de cromossomas redundantes gerando cromossomas inválidos.

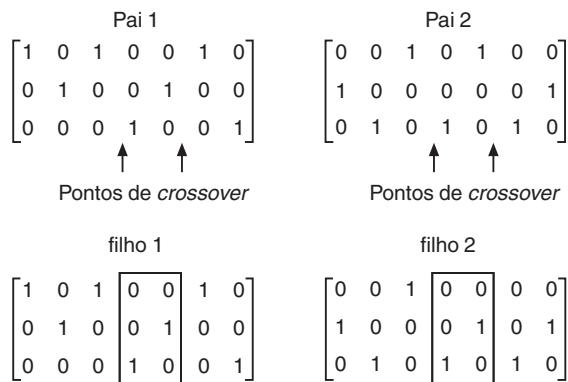
Na Figura 5.6, ambos cromossomas pais representam o mesmo *cluster*,  $\{ (x_1, x_3, x_6), (x_2, x_4, x_5) \}$ , embora os grupos sejam diferentes.

A seguir encontram-se relacionados alguns tipos de *crossover* que variam em função da representação utilizada do cromossoma:

- Representação do cromossoma por grupamento de número – *Crossover* de um ponto e *crossover* uniforme podem ser usados para essa representação de cromossoma. Entretanto, ambos operadores podem produzir cromossomas inválidos. Para evitar esse tipo de problema, pode-se usar o *crossover* baseado em ordem.
- Representação do cromossoma por matriz – Dois tipos de *crossover* podem ser utilizados para essa representação, são eles: *crossover* assexual de um ponto e *crossover* sexual de dois pontos. Ambos os *crossovers* podem gerar soluções inválidas, isto é, cromossomas com linhas somente de elementos “0’s”. As Figuras 5.7 e 5.8 mostram o *crossover* assexual de um ponto e o *crossover* sexual de dois pontos respectivamente.

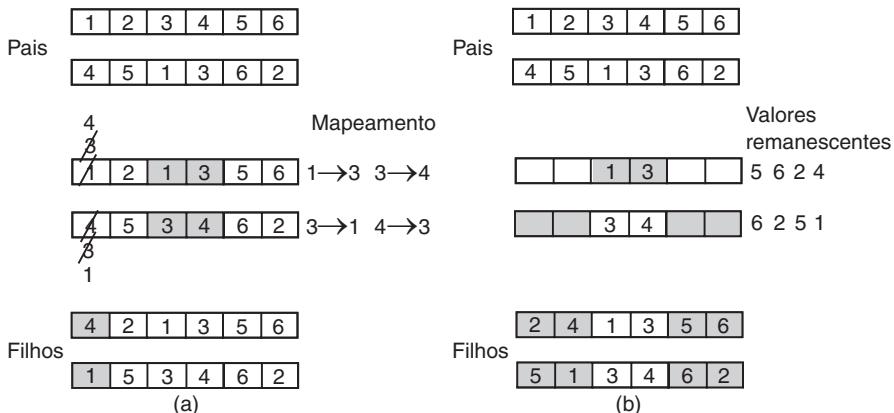


**Figura 5.7.** Crossover assexual de 1 ponto.



**Figura 5.8.** Crossover sexual de 2 pontos.

- Representação do cromossoma por permutação com separador – Para esse tipo de representação são recomendados dois tipos de *crossover*, são eles: *crossover* baseado em ordem e *crossover* de mapeamento parcial. Ambos podem ser observados através da Figura 5.9.

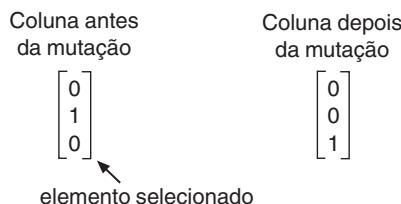


**Figura 5.9.** Operadores de crossover para restauração de cromossoma por permutação.  
(a) Crossover baseado em mapeamento parcial; (b) crossover baseado em ordem.

- Representação do cromossoma por permutação *greedy* – Os operadores de crossover também são: o crossover baseado em mapeamento parcial e crossover baseado em ordem mostrados pela Figura 5.9.

A mutação introduz novo material genético na população. Na tarefa de Clusterização, a mutação corresponde a mover objetos de um *cluster* para outro *cluster*.

- Representação do cromossoma por grupamento de número – A mutação utilizada para esta representação consiste em inverter cada bit do cromossoma com a probabilidade igual à taxa de mutação.
- Representação do cromossoma por matriz – Nesse caso, o processo de mutação ocorre da seguinte maneira: uma coluna da matriz sofre a mutação, ou seja, um elemento da coluna da matriz é escolhido aleatoriamente e é colocado o valor “1” para esse elemento, todos os outros elementos da coluna recebem o valor “0”. Se o elemento escolhido já era de valor “1” o operador não faz efeito. A Figura 5.10 ilustra esse operador de mutação.



**Figura 5.10.** Operador de mutação para representação do cromossoma por matriz.

- Representação do cromossoma por permutação com separador – A mutação utilizada nessa representação seleciona randomicamente dois elementos e os troca de posição. Para assegurar que não serão gerados cromossomas inválidos, não se pode trocar de posição o separador de grupos.
- Representação do cromossoma por permutação *greedy* – A mutação utilizada nesse caso é a mesma utilizada no caso da representação por permutação com separador.

As funções de avaliação que são comumente utilizadas para problemas envolvendo a tarefa de Clusterização são:

- Minimização do traço ( $W$ )
- Minimização do determinante de ( $W$ )
- Maximização do traço ( $BW^{-1}$ )

$$W = \sum_{i=1}^k \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_i)(x_{ij} - \bar{x}_i)' \quad (2)$$

$$B = \sum_{i=1}^k n_i (\bar{x}_i - \bar{x})(\bar{x}_i - \bar{x})' \quad (3)$$

Onde:

$k$  é o número de *clusters*,  $n_i$  é o número de elementos do *cluster*  $i$ ,  $x_{ij}$  é o  $j$ -ésimo objeto do *cluster*  $i$ , é o centróide do *cluster*  $i$ ,  $\bar{x}_i = \frac{1}{n_i} \sum_{l=1}^{n_i} x_{il}$  é a média de todos os objetos do conjunto de dados e  $\bar{x} = \frac{1}{n} \sum_{l=1}^n x_l$  significa a matriz transposta. As matrizes  $B$  e  $W$  são muito utilizadas em análise discriminante.

A minimização do traço ( $W$ ) é equivalente a minimização da soma do quadrado da distância Euclideana entre o indivíduo e o centróide do seu *cluster*.

O algoritmo tenta colocar os elementos nos *clusters* cuja distância do elemento ao *cluster* seja a menor possível. O processo termina quando não há mais arranjos de elementos que minimizem a função de avaliação.

## Métodos Baseados em Instâncias

A expressão “método baseado em instância” indica que o método, ao processar um novo registro, leva em consideração as instâncias ou os registros existentes na base de dados. Um dos principais métodos de Mineração de Dados baseados em instâncias é denominado *K-NN* (*K*-Nearest Neighbors ou, em português, *K*-Vizinhos mais Próximos), e encontra-se apresentado a seguir.

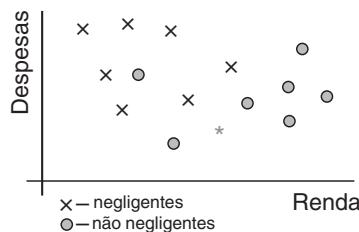
## K-NN

O método *K-NN* é muito utilizado em aplicações envolvendo a tarefa de classificação. Trata-se de um método de fácil entendimento e implementação e que não requer treinamento prévio para ser aplicado. O funcionamento do *K-NN* encontra-se descrito a seguir.

Considerando uma base de dados (base de referência) de um problema envolvendo a tarefa de classificação (que contém um atributo cujos valores são rótulos de classes predefinidas) e cada novo registro a ser classificado (registro da base de teste), os seguintes passos são executados:

- Cálculo da distância do novo registro a cada um dos registros existentes na base de referência. Para tanto, utiliza-se alguma métrica de distância. Exemplos de métricas de distância foram apresentadas no Capítulo 3.
- Identificação dos  $k$  registros da base de referência que apresentaram menor distância em relação ao novo registro (mais similares).
- Apuração da classe mais freqüente entre os  $k$  registros identificados no passo anterior.
- Comparação da classe apurada com a classe real, computando erro ou acerto do algoritmo. Este último passo só deve ser utilizado quando as classes dos novos registros são conhecidas e deseja-se avaliar o desempenho do método *K-NN* na base de dados em questão. Nas demais situações, este algoritmo se encerra no passo anterior.

Considere o exemplo no contexto da análise de crédito, cuja base de dados de referência encontra-se representada na Figura 5.11. Este conjunto está dividido em duas classes: os negligentes, representados com um “x” e os não negligentes, representados por um “•”. Nesta aplicação, deseja-se avaliar a possibilidade de concessão de crédito a novas solicitações.

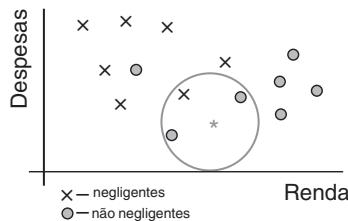


**Figura 5.11.** Conjunto contendo dados sobre clientes que receberam crédito.

Apresentando-se um novo registro (solicitante), representado por “\*”, calcula-se a distância entre o novo registro e todos os registros existentes na base de dados de referência. Assumindo que o número de  $k$  de vizinhos mais próxi-

mos seja 3, somente os 3 registros com menor distância ao novo registro são considerados.

Desta forma, avaliando os resultados na Figura 5.12, observa-se que a classe com maior ocorrência dentro da área delimitada pelo algoritmo  $k$ -NN foi “cliente não negligente”. Sendo assim, pela aplicação do algoritmo  $K$ -NN no exemplo apresentado, o crédito seria concedido ao solicitante.



**Figura 5.12.** Resultado do  $k$ -NN.

## Métodos Estatísticos

Diversos algoritmos de Mineração de Dados são fundamentados em princípios e teorias da Estatística. Nesta seção estão descritos alguns deles.

### Classificador Bayesiano Ingênuo

O Classificador Bayesiano Ingênuo baseia-se no Teorema de Bayes, estando relacionado ao cálculo de probabilidades condicionais. É aplicável, conforme o próprio nome sugere, em tarefas de classificação.

Formalmente, sejam  $X(A_1, A_2, \dots, A_n, C)$  um conjunto de dados,  $C_1, C_2, \dots, C_k$ , as classes do problema (valores possíveis do atributo  $C$ ) e  $R$  um novo registro que deve ser classificado. Sejam ainda  $a_1, a_2, \dots, a_n$  os valores que  $R$  assume em  $X$ .

O Classificador Bayesiano Ingênuo possui dois passos:

- 1) Calcular a probabilidade  $P(C=C_i/R)$ ,  $i=1,2,\dots,k$ .
- 2) Indicar como saída do algoritmo a classe  $C_j$  tal que  $P(C=C_j/R)$  seja máxima.

O problema reduz-se, portanto, ao cálculo das probabilidades condicionais  $P(C=C_i/R)$ ,  $i=1,2,\dots,k$ . Sabe-se que  $P(C=C_i/R)$  pode ser reescrito como:

$$P(C=C_i / A_1 = a_1 \text{ e } A_2 = a_2 \text{ e } \dots \text{ e } A_n = a_n)$$

Por outro lado, pelo teorema de Bayes, como  $P(A/B) = (P(B/A) * P(A)) / P(B)$ ,

$$\begin{aligned} P(C=C_i / A_1 = a_1 \text{ e } A_2 = a_2 \text{ e } \dots \text{ e } A_n = a_n) &= (P(A_1 = a_1 \text{ e } A_2 = a_2 \text{ e } \dots \text{ e } \\ &A_n = a_n / C=C_i)^* \\ &P(C=C_i)) / P(A_1 = a_1 \text{ e } A_2 = a_2 \text{ e } \dots \text{ e } A_n = a_n) \end{aligned}$$

O denominador na igualdade mostrado anteriormente será sempre o mesmo, independente da classe para a qual a probabilidade esteja sendo calculada. Assim sendo, para fins de comparação entre as probabilidades, o denominador  $P(A_1 = a_1 \text{ e } A_2 = a_2 \text{ e } \dots \text{ e } A_n = a_n)$  pode ser desprezado do cálculo. Desta forma, a expressão reduz-se para:

$$\frac{P(C=C_i / A_1 = a_1 \text{ e } A_2 = a_2 \text{ e } \dots \text{ e } A_n = a_n)}{P(A_1 = a_1 \text{ e } A_2 = a_2 \text{ e } \dots \text{ e } A_n = a_n / C=C_i)} = P(C=C_i)$$

O nome “ingênuo” no título do método decorre da premissa assumida pelo algoritmo de que os atributos serão sempre independentes entre si, o que, em muitos casos, não deverá ocorrer.

Da teoria das probabilidades, se dois eventos A e B são independentes, então  $P(AB) = P(A) * P(B)$ . Assim sendo a igualdade acima pode ser reescrita da seguinte forma:

$$\frac{P(C=C_i / A_1 = a_1 \text{ e } A_2 = a_2 \text{ e } \dots \text{ e } A_n = a_n)}{P(A_1 = a_1 / C=C_i) * P(A_2 = a_2 / C=C_i) * \dots * P(A_n = a_n / C=C_i)} = P(C=C_i)$$

Para ilustrar a aplicação deste método, considere o banco de dados abaixo:

Aparência	Temperatura	Umidade	Vento	Jogar Tênis
Ensolarado	Quente	Alta	Fraco	Não
Ensolarado	Quente	Alta	Forte	Não
Nublado	Quente	Alta	Fraco	Sim
Chuvoso	Moderado	Alta	Fraco	Sim
Chuvoso	Fresco	Normal	Fraco	Sim
Chuvoso	Fresco	Normal	Forte	Não
Nublado	Fresco	Normal	Forte	Sim
Ensolarado	Moderado	Alta	Fraco	Não
Ensolarado	Fresco	Normal	Fraco	Sim
Chuvoso	Moderado	Normal	Fraco	Sim
Ensolarado	Moderado	Normal	Forte	Sim
Nublado	Moderado	Alta	Forte	Sim
Nublado	Quente	Normal	Fraco	Sim
Chuvoso	Moderado	Alta	Forte	Não

Considerando o atributo Jogar Tênis como objetivo da classificação, pode-se observar que este problema tem duas classes: Jogar=Sim e Jogar=Não.

Os atributos  $A_i$  são Aparência, Temperatura, Umidade e Vento. Pergunta-se: Devo ou não jogar tênis em dia ensolarado, quente, de alta umidade e vento fraco?

Aplicando o Classificador Bayesiano Ingênuo, temos:

$$P(\text{Jogar=Sim} \mid \text{ensolarado, quente, alta umidade, vento fraco}) =$$

$$P(\text{ensolarado} \mid \text{Jogar=Sim}) * P(\text{quente} \mid \text{Jogar=Sim}) * P(\text{alta umidade} \mid \text{Jogar=Sim}) * P(\text{vento fraco} \mid \text{Jogar=Sim}) = 0,0071$$

$$P(\text{Jogar=Não} \mid \text{ensolarado, quente, alta umidade, vento fraco}) =$$

$$P(\text{ensolarado} \mid \text{Jogar=Não}) * P(\text{quente} \mid \text{Jogar=Não}) * P(\text{alta umidade} \mid \text{Jogar=Não}) * P(\text{vento fraco} \mid \text{Jogar=Não}) = 0,0274$$

A resposta do algoritmo seria Jogar=Não

## K-Means

O algoritmo *k-means* é um método popular da tarefa de Clusterização. Tomase, randomicamente,  $k$  pontos de dados (dados numéricos) como sendo os centróides (elementos centrais) dos clusters. Em seguida, cada ponto (ou registro da base de dados) é atribuído ao cluster cuja distância deste ponto em relação ao centróide de cada cluster é a menor dentre todas as distâncias calculadas. Um novo centróide para cada cluster é computado pela média dos pontos do cluster, caracterizando a configuração dos clusters para a iteração seguinte. O processo termina quando os centróides dos clusters param de se modificar, ou após um número limitado de iterações que tenha sido especificado pelo usuário.

O algoritmo *k-means* toma um parâmetro de entrada,  $k$ , e divide um conjunto de  $n$  objetos em  $k$  clusters tal que a similaridade intracluster resultante seja alta, mas a similaridade intercluster seja baixa. A similaridade em um cluster é medida em respeito ao valor médio dos objetos neste cluster (centro de gravidade do cluster).

A execução do algoritmo *k-means* consiste em, primeiro, selecionar aleatoriamente  $k$  objetos, que inicialmente representam cada um a média de um cluster. Para cada um dos objetos remanescentes, é feita a atribuição ao cluster ao qual o objeto é mais similar, baseado na distância entre o objeto e a média do cluster. A partir de então, o algoritmo computa as novas médias para cada cluster. Este processo se repete até que uma condição de parada seja atingida.

O funcionamento do algoritmo *k-means* encontra-se resumido na Figura 5.13. O algoritmo tenta determinar  $k$  partições que minimizem a função do erro quadrado. Apresenta bom desempenho quando os clusters são densos e compactos e bem separados uns dos outros (Carlantonio, 2001).

Cabe ressaltar que a necessidade do usuário ter que especificar  $k$ , o número de clusters, com antecedência pode ser visto como uma desvantagem. Em geral, diversos experimentos variando o valor de  $k$  devem ser realizados.

O método *k-means* não é adequado para descobrir clusters com formas não convexas ou clusters de tamanhos muito diferentes (Carlantonio, 2001).

Existem objetos que não seguem o comportamento geral dos dados. Esses objetos, que são diferentes ou inconsistentes em relação ao conjunto de dados formado, são chamados de ruídos (*outliers*). O método *k-means* é sensível a ruídos, visto que pequeno número de dados ruidosos pode influenciar, substancialmente, os valores médios dos clusters.

Muitas variações do método *k-means* são encontradas atualmente. Em geral, essas variações diferem na seleção das  $k$  médias iniciais, no cálculo da similaridade, ou na estratégia para calcular a média dos clusters.

As Figuras 5.14 e 5.15 ilustram a aplicação do algoritmo *k-means* em um arquivo com 20 registros de dados, considerando-se  $k = 3$ .

O algoritmo *k-means* é inicializado com os centros (médias) colocados em posições aleatórias. A busca pelo centro comum se faz de forma iterativa. Após essa inicialização, os objetos restantes são agrupados conforme a distância em que se encontram das médias.

## K-Modes

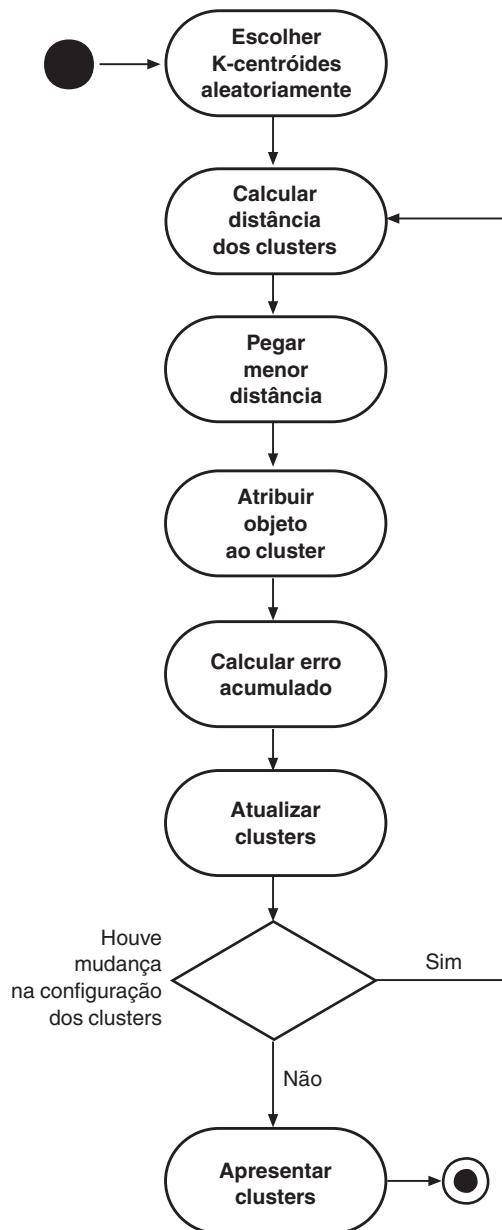
O algoritmo *k-modes* é uma variação do método *k-means*, só que utilizado para clusterização dados categóricos (nominais). Em geral, no lugar do cálculo da média, calcula-se a moda (valor que aparece com maior freqüência) dos objetos, usando medidas de similaridade para tratar objetos categóricos, e usando métodos baseados em freqüência para atualizar as modas dos clusters.

## K-Prototypes

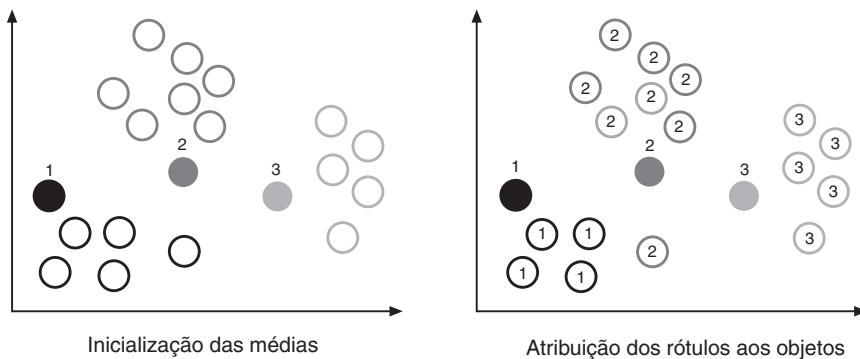
O método *k-prototypes* é a integração dos métodos *k-means* e *k-modes*. Esse método pode ser aplicado a bases de dados que contenham tanto atributos numéricos quanto atributos categóricos.

## K-Medoids

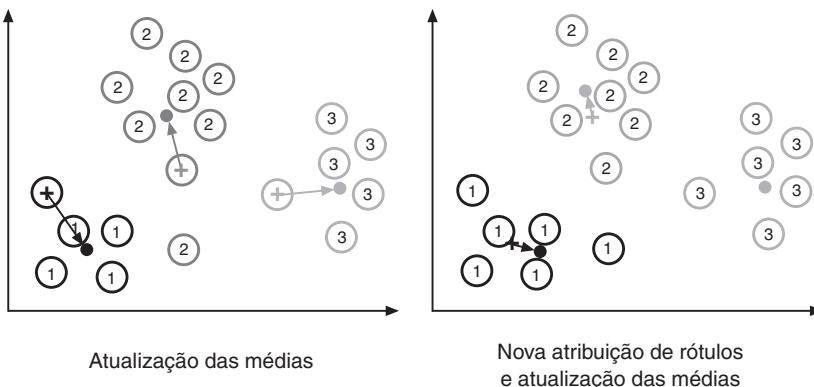
O algoritmo *k-medoids* baseia-se, primeiramente, em encontrar o *medoid* (objeto mais centralmente localizado em um cluster). Os objetos restantes são então clusterizados com o *medoid* ao qual ele é mais similar. Há então uma troca iterativa, de um *medoid* por um não *medoid*, visando à melhoria da clusterização. A qualidade é estimada usando uma função custo que mede a similaridade média entre os objetos e o *medoid* de seu cluster.

**Figura 5.13.** Algoritmo K-Means.

A diferença básica entre o *K-Means* e *K-Medoids* é que no segundo, cada cluster é representado por um dos registros pertencentes ao cluster. No primeiro, o elemento representante de cada cluster é a média dos registros que pertencem ao cluster.



**Figura 5.14.** Primeiros Passos do Algoritmo.



**Figura 5.15.** Passos Subseqüentes do Algoritmo.

## Métodos Específicos

Nesta seção estão referenciados alguns algoritmos desenvolvidos especificamente para implementar alguma tarefa de Mineração de Dados, com destaque especial para o algoritmo *Apriori*, voltado à Descoberta de Associações.

### Apriori

O *Apriori* é um algoritmo clássico de Mineração de Regras de Associação (Agrawal, 1993). Diversos algoritmos tais como *GSP*, *DHP*, *Partition*, *DIC*, *Eclat*, *MaxEclat*, *Clique* e *MaxClique* foram inspirados no funcionamento do *Apriori* e se baseiam no princípio da *antimonotonicidade* do suporte.

Segundo o princípio da *antimonotonicidade* do suporte “Um  $k$ -itemset somente pode ser freqüente se todos os seus  $(k-1)$ -itemsets forem freqüentes”. Assim sendo, a combinação de itemsets para gerar um novo itemset somente ocorre quando estes são freqüentes.

Os algoritmos mencionados acima podem ser decompostos basicamente em duas etapas:

- Encontrar todos os conjuntos de itens freqüentes (que satisfazem à condição de suporte mínimo).
- A partir do conjunto de itens freqüentes, gerar as regras de associação (que satisfazem à condição de confiança mínima).

Como a tarefa do item (a) demanda maior custo computacional e, uma vez gerados todos os conjuntos de itens freqüentes, a tarefa (b) se torna mais imediata, esforços de otimização têm sido concentrados na etapa (a).

Considere como exemplo o banco de dados abaixo. Em seguida, são descritas as etapas de funcionamento do algoritmo Apriori sobre este exemplo:

Transação	Leite	Café	Cerveja	Pão	Manteiga	Arroz	Feijão
1	Não	Sim	Não	Sim	Sim	Não	Não
2	Sim	Não	Sim	Sim	Sim	Não	Não
3	Não	Sim	Não	Sim	Sim	Não	Não
4	Sim	Sim	Não	Sim	Sim	Não	Não
5	Não	Não	Sim	Não	Não	Não	Não
6	Não	Não	Não	Não	Sim	Não	Não
7	Não	Não	Não	Sim	Não	Não	Não
8	Não	Não	Não	Não	Não	Não	Sim
9	Não	Não	Não	Não	Não	Sim	Sim
10	Não	Não	Não	Não	Não	Sim	Não

Inicialmente, o usuário deve definir os valores de suporte e confiança mínimos a serem considerados pelo Apriori. Consideremos para este exemplo que tenham sido definidos  $\text{MinSup}=0,3$  e  $\text{MinConf}=0,8$ . Os cálculos de suporte e confiança obedecem às fórmulas descritas no Capítulo 4, na tarefa de Descoberta de Associações.

A etapa (a) é iterativa. Em cada iteração são identificados *itemsets* freqüentes. No exemplo:

1ª Iteração – São identificados os 1-itemsets freqüentes (suporte maior que Min-Sup).

1-Itemsets	Suportes
Leite	0,2
Café	0,3
Cerveja	0,2
Pão	0,5
Manteiga	0,5
Arroz	0,2
Feijão	0,2

2<sup>a</sup> Iteração – São combinados os 1-itemsets freqüentes para gerar os 2-itemsets. Destes, são selecionados aqueles cujo suporte seja superior ao mínimo definido.

2-Itemsets	Suportes
Café, Pão	0,3
Café, Manteiga	0,3
Pão, Manteiga	0,4

3<sup>a</sup> Iteração – São combinados os 2-itemsets freqüentes para gerar os 3-itemsets. Destes, são selecionados aqueles cujo suporte seja superior ao mínimo definido.

3-Itemsets	Suportes
Café, Pão, Manteiga	0,3

Assim sendo, a lista de todos os k-itemsets com k2 resultante da etapa (a) foi:

- Café e Pão
- Café e Manteiga
- Pão e Manteiga
- Café, Pão e Manteiga

Esta lista é informada à etapa (b) que deve identificar as regras válidas, ou seja, aquelas regras cuja confiança seja superior à confiança mínima definida pelo usuário. Para isso, são geradas as regras possíveis a partir de cada *itemset* freqüente, assim como sua confiança.

- Itemset {Café, Pão}
  - SE café ENTÃO pão. Conf = 1,0.
  - SE pão ENTÃO café. Conf = 0,6.

- Itemset {Café, Manteiga}
 

SE café ENTÃO manteiga.	Conf = 1,0.
SE manteiga ENTÃO café.	Conf = 0,6.
  
- Itemset {Pão, Manteiga}
 

SE manteiga ENTÃO pão.	Conf = 0,8.
SE pão ENTÃO manteiga.	Conf = 0,8.
  
- Itemset {Café, Pão, Manteiga}
 

SE café, pão ENTÃO manteiga.	Conf = 1,0.
SE café, manteiga ENTÃO pão.	Conf = 1,0.
SE manteiga, pão ENTÃO café.	Conf = 0,75.
SE café ENTÃO pão, manteiga.	Conf = 1,0.
SE pão ENTÃO café, manteiga.	Conf = 0,6.
SE manteiga ENTÃO café, pão.	Conf = 0,6.

Após a filtragem das regras com confiança acima da confiança mínima, o Apriori produz como saída as seguintes regras de associação:

SE café ENTÃO pão.  
 SE café ENTÃO manteiga.  
 SE manteiga ENTÃO pão.  
 SE pão ENTÃO manteiga.  
 SE café,pão ENTÃO manteiga.  
 SE café, manteiga ENTÃO pão.  
 SE café ENTÃO pão, manteiga.

## Métodos Baseados em Indução de Árvores de Decisão

Alguns dos principais métodos de Mineração de Dados são baseados na construção de árvores de decisão a partir das bases de dados. Em geral a construção de uma árvore de decisão é realizada segundo alguma abordagem recursiva de particionamento da base de dados. A seguir encontra-se descrito o algoritmo C4.5, um exemplo clássico de método baseado na indução de árvores de decisão, bastante aceito e utilizado pela comunidade científica mundial.

### C4.5

O C4.5 é um dos mais tradicionais algoritmos na tarefa de Classificação (Quinlan, 1993). Inspirado no algoritmo ID3 (Quinlan, 1979), o método C4.5 procura abstrair árvores de decisão a partir de uma abordagem recursiva de particionamento das bases de dados. Utiliza, para tanto, conceitos e medidas da Teoria da Informação.

A fim de descrever o funcionamento do algoritmo C4.5, consideremos sua aplicação em um conjunto de dados  $S(A_1, A_2, \dots, A_n, C)$  em que  $C$  é o atributo objeto da tarefa de Classificação e que possui os seguintes valores:  $C_1, C_2, \dots, C_k$ , denominados classes do problema.

Uma Árvore de Decisão é um modelo de conhecimento em que cada nó interno da árvore representa uma decisão sobre um atributo que determina como os dados estão particionados pelos seus nós filhos. Inicialmente, a raiz da árvore contém toda a base de dados com exemplos misturados das várias classes. Um predicado, denominado ponto de separação, é escolhido como sendo a condição que melhor separa ou discrimina as classes. Tal predicado envolve exatamente um dos atributos do problema e divide a base de dados em dois ou mais conjuntos, que são associados cada um a um nó filho. Cada novo nó abrange, portanto, uma partição da base de dados que é recursivamente separada até que conjunto associado a cada nó folha consista inteiramente ou predominantemente de registros de uma mesma classe.

A maioria dos algoritmos baseados na abstração de árvores de decisão consiste em duas fases: Construção da Árvore de Decisão e Simplificação da Árvore de Decisão.

Na fase de Construção da Árvore de Decisão, uma árvore é gerada pelo particionamento recursivo dos dados de treinamento. O conjunto de treinamento é separado em duas ou mais partições usando restrições sobre os conjuntos de valores de cada atributo. O processo é repetido recursivamente até que todos ou a maioria dos exemplos em cada partição pertençam a uma classe. A árvore gerada abrange todo o conjunto de treinamento e é construída em largura. Assim, todos os nós em uma determinada altura da árvore devem ser processados antes do início do processamento do nível subsequente.

Há duas operações principais durante o processo de construção da árvore: (a) a avaliação dos pontos de separação de cada nó interno da árvore e a identificação de qual o melhor ponto de separação; e (b) a criação das partições usando o melhor ponto de separação identificado para os casos pertencentes a cada nó. Uma vez determinado o melhor ponto de separação de cada nó, as partições podem ser criadas pela simples aplicação do critério de separação identificado.

Para avaliação dos pontos de separação de cada nó interno da árvore, as seguintes medidas devem ser calculadas:

- Ganho de informação considerando a partição da base de dados associada ao nó em análise. Observa-se que, para o nó raiz, a base de dados está completa. Para esse cálculo utiliza-se a fórmula (1) sobre o atributo de classificação:

$$\text{info}(S) = -\sum_{j=1}^k \frac{\text{freq}(C_j, S)}{|S|} \times \log_2 \left( \frac{\text{freq}(C_j, S)}{|S|} \right) \text{bits}$$

Onde:

$S$  representa a partição da base de dados;

$freq(Cj, S)$  representa o número de vezes em que a classe  $Cj$  acontece em  $S$ ;

$|S|$  denota o número de casos do conjunto  $S$ ;

$k$  indica o número de classes distintas.

- Ganho de informação de cada atributo considerando a partição da base de dados associada ao nó em análise. Observa-se que, para o nó raiz, todos os atributos, com exceção do atributo de classificação, devem ser analisados. Para esse cálculo utilizam-se as fórmulas (2) e (3) sobre cada atributo:

$$info_x(T) = \sum_{i=1}^n \frac{|T_i|}{|T|} \times info(T_i) \quad (2)$$

Onde:

$T$  representa a quantidade de ocorrências na partição em análise;

$T_i$  representa a quantidade de ocorrências de uma classe contidas no conjunto  $T$ .

$$gain(X) = info(T) - info_x(T) \quad (3)$$

- Seleção do atributo com maior ganho de informação obtido sobre a partição em análise.

É importante ressaltar que o processo de avaliação de pontos de separação depende do domínio de cada atributo, que pode ser numérico ou categórico.

O processo de avaliação dos pontos de separação de atributos numéricos baseia-se em testes dicotômicos da forma  $A < v$ , onde  $v$  é um número real. Esse processo requer a ordenação dos exemplos de treinamento baseado nos valores do atributo em análise. Por exemplo, sejam  $v_1, v_2, \dots, v_n$ , valores ordenados de um atributo numérico  $A$ . Como qualquer valor entre  $v_i$  e  $v_{i+1}$  divide o conjunto nos mesmos dois subconjuntos, apenas  $(n - 1)$  possibilidades de separação precisam ser analisadas. Tipicamente, o ponto médio entre  $v_i$  e  $v_{i+1}$  é escolhido como ponto de separação. Pode ser observado, portanto, que o custo de avaliação das separações para um atributo numérico é dominado pelo custo de ordenação dos valores.

O processo de avaliação dos pontos de separação de atributos categóricos baseia-se em testes sobre cada atributo individualmente.

Para ilustrar o procedimento descrito acima, consideremos  $T$  o conjunto de dados abaixo contendo informações sobre as condições climáticas e a recomendação quanto a jogar ou não golfe em cada situação. Deseja-se construir uma ár-

vore de decisão que, baseada nas informações climáticas, ofereça uma recomendação quanto à prática de golfe.

Aparência	Temperatura (°F)	Umidade (%)	Vento	Jogar Golfe
Ensolarada	75	70	Sim	Sim
Ensolarada	80	90	Sim	Não
Ensolarada	85	85	Não	Não
Ensolarada	72	95	Não	Não
Ensolarada	69	70	Não	Sim
Nublada	72	90	Sim	Sim
Nublada	83	78	Não	Sim
Nublada	64	65	Sim	Sim
Nublada	81	75	Não	Sim
Chuvosa	71	80	Sim	Não
Chuvosa	65	70	Sim	Não
Chuvosa	75	80	Não	Sim
Chuvosa	68	80	Não	Sim
Chuvosa	70	96	Não	Sim

Neste problema, o atributo de classificação (*Jogar Golfe*) contém duas classes, *Sim* com nove ocorrências e *Não* com cinco ocorrências. Calcula-se a entropia do conjunto *T* completo, utilizando a fórmula (1):

$$info(golf) = -\frac{9}{14} \times \log_2\left(\frac{9}{14}\right) - \frac{5}{14} \times \log_2\left(\frac{5}{14}\right) = 0,94 bits$$

Considerando o atributo *Aparência*, devemos dividir o conjunto *T* em três partições, *T1*, *T2*, *T3*, representando os valor possíveis do atributo (*Ensolarada*, *Nublada* e *Chuvosa*). Com o cálculo da entropia deste atributo, utilizando a fórmula (2), temos:

$$\begin{aligned} info_x(outlook) &= \frac{5}{14} \times \left( -\frac{2}{5} \times \log_2\left(\frac{2}{5}\right) - \frac{3}{5} \times \log_2\left(\frac{3}{5}\right) \right) \\ &\quad + \frac{4}{14} \times \left( -\frac{4}{4} \times \log_2\left(\frac{4}{4}\right) - \frac{0}{4} \times \log_2\left(\frac{0}{4}\right) \right) \\ &\quad + \frac{5}{14} \times \left( -\frac{3}{5} \times \log_2\left(\frac{3}{5}\right) - \frac{2}{5} \times \log_2\left(\frac{2}{5}\right) \right) = 0,694 bits \end{aligned}$$

Calculando o ganho de informação do atributo *Aparência*, utilizando a fórmula (3) temos:

$$gain(outlook) = 0,940 - 0,694 = 0,246 bits$$

De forma análoga, o procedimento deve ser repetido para os demais atributos, sendo selecionado o atributo com maior ganho. Esta escolha determina o ponto de separação no nó raiz da árvore. A partir deste momento, a base de dados é particionada e o procedimento repetido para cada novo nó gerado. O exemplo a seguir mostra a partição final do conjunto de dados e a árvore de decisão correspondente.

### *Partição do Conjunto de Dados*

- Aparência=Ensolarada:

Umidade  $\leq 75$ : Jogar=Sim

Aparência	Temperatura (°F)	Umidade (%)	Vento	Jogar Golfe
Ensolarada	75	70	Sim	Sim
Ensolarada	69	70	Não	Sim

Umidade  $> 75$ : Jogar=Não

Aparência	Temperatura (°F)	Umidade (%)	Vento	Jogar Golfe
Ensolarada	80	90	Sim	Não
Ensolarada	85	85	Não	Não
Ensolarada	72	95	Não	Não

- Aparência=Nublada: Jogar=Sim

Aparência	Temperatura (°F)	Umidade (%)	Vento	Jogar Golfe
Nublada	72	90	Sim	Sim
Nublada	83	78	Não	Sim
Nublada	64	65	Sim	Sim
Nublada	81	75	Não	Sim

- Aparência=Chuvosa:

Vento=Sim: Jogar=Não

Aparência	Temperatura (°F)	Umidade (%)	Vento	Jogar Golfe
Chuvosa	71	80	Sim	Não
Chuvosa	65	70	Sim	Não

Vento=Não: Jogar=Sim

Aparência	Temperatura (°F)	Umidade (%)	Vento	Jogar Golfe
Chuvosa	75	80	Não	Sim
Chuvosa	68	80	Não	Sim
Chuvosa	70	96	Não	Sim

### Árvore de Decisão Construída

Aparência=Ensolarada:

Umidade  $\leq 75$ : Jogar=Sim

Umidade  $> 75$ : Jogar=Não

Aparência=Nublada: Jogar=Sim

Aparência=Chuvosa:

Vento=Sim: Jogar=Não

Vento=Não: Jogar=Sim

## Métodos Baseados em Lógica Nebulosa

Diversos métodos de Mineração de Dados foram adaptados de forma a incorporar a flexibilidade proporcionada pela Lógica Nebulosa. Entre eles podemos citar as versões nebulosas do K-Means e do C4.5. Nessas versões, os registros da base de dados podem pertencer a diversos clusters e classes simultaneamente, com diferentes graus de pertinência. A seguir encontra-se descrito o algoritmo Wang-Mendel, concebido para aplicação na tarefa de Previsão de Séries Temporais.

Para os leitores não familiarizados com a tecnologia da Lógica Nebulosa, recomenda-se uma leitura prévia do Anexo III, que apresenta uma introdução aos conceitos básicos mínimos necessários ao entendimento desta seção.

### Wang-Mendel

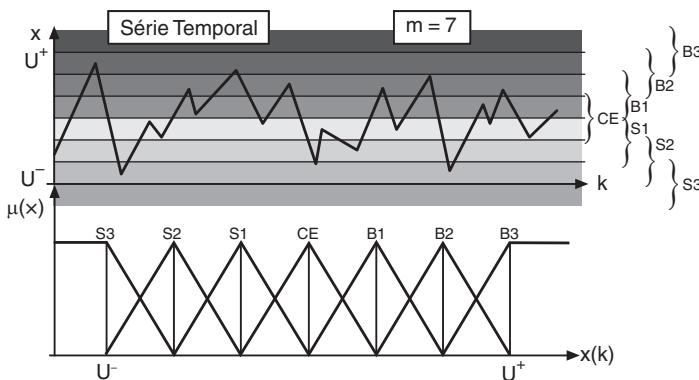
O Algoritmo Wang-Mendel foi concebido para aplicação em tarefas de Previsão de Séries Temporais utilizando Lógica Nebulosa. Esse método consiste em abs-

trair regras nebulosas a partir de conjuntos de dados históricos. Utilizamos esses dados históricos para definir os antecedentes e os consequentes das regras nebulosas.

Para uma apresentação formal do método Wang-Mendel, consideremos  $X(k)$ ,  $k=1,2,\dots$  uma série temporal, em que  $X(k) \in [U^-; U^+]$ . O objetivo desse método consiste em abstrair um conjunto de regras nebulosas que permita a partir de uma janela de  $n$  medidas de  $X(k)$  ( $X(k-n+1), X(k-n+2), \dots, X(k)$ ), determinar os valores de  $X$ ,  $t$  pontos à frente ( $X(k+1), X(k+2), \dots, X(k+t)$ ), sendo  $n$  e  $t$  valores positivos.

Como os valores a serem previstos dependem de  $n$  valores passados de  $X$ , cada regra nebulosa deverá possuir  $n$  antecedentes.

Divide-se a faixa de valores possíveis da série  $[U^-; U^+]$  em  $m$  conjuntos nebulosos de igual comprimento, devendo  $m$  ser um valor ímpar. A Figura 5.16 ilustra uma série temporal sendo dividida em sete conjuntos nebulosos.



**Figura 5.16.** Divisão da Série Temporal em Conjuntos Nebulosos.

O processo de geração das regras requer que sejam definidos o tamanho da janela  $n$  e o horizonte de previsão  $t$ . Ambos dependem da aplicação. Por exemplo, em uma aplicação para previsão de carga de 10 em 10 minutos, pode ser utilizada uma janela de 6 pontos, retratando a última hora anterior completa. O horizonte de previsão pode ser de um ponto, representando a previsão para os dez minutos seguintes.

O processo de geração das regras, segue, para cada regra  $j$ , três passos:

- Determina-se o grau de pertinência dos elementos de  $X^j$  – Considerando o exemplo, teríamos:

Antecedente:

$$X^j(1) = \begin{cases} S3 = 0.8 \\ S2 = 0.2 \end{cases} \quad X^j(2) = \begin{cases} S1 = 0.8 \\ CE = 0.2 \end{cases} \quad X^j(3) = \begin{cases} SI = 0.95 \\ CE = 0.05 \end{cases}$$

$$X^j(4) = \begin{cases} B1 = 0.9 \\ B2 = 0.1 \end{cases} \quad X^j(5) = \{CE = 0,1\} \quad X^j(6) = \begin{cases} B2 = 0.9 \\ B3 = 0.1 \end{cases}$$

Conseqüente:

$$X^j(7) = \begin{cases} S1 = 0.85 \\ CE = 0.15 \end{cases}$$

- Atribui-se a cada variável o conjunto com maior grau de pertinência:

$$X^j(1) = S3 = 0.8$$

$$X^j(2) = S1 = 0.8$$

$$X^j(3) = S1 = 0.95$$

$$X^j(4) = B1 = 0.9$$

$$X^j(5) = CE = 1.0$$

$$X^j(6) = B2 = 0.9$$

$$X^j(7) = S1 = 0.85$$

- Obtém-se uma regra para cada par entrada-saída – Considerando o exemplo, teríamos a seguinte regra:

$$SE \quad X^j(1) \text{ é } S3$$

$$E \quad X^j(2) \text{ é } S1$$

$$E \quad X^j(3) \text{ é } S1$$

$$E \quad X^j(4) \text{ é } B1$$

$$E \quad X^j(5) \text{ é } CE$$

$$E \quad X^j(6) \text{ é } B2$$

$$\text{ENTÃO} \quad X^j(7) \text{ é } S1$$

As regras obtidas são armazenadas em uma base de conhecimento para serem processadas posteriormente a partir de novas entradas.

Convém observar que devido à grande quantidade de dados, é comum obter-se regras conflitantes, ou seja, regras com o mesmo antecedente, mas conseqüentes distintos. Para lidar com esse problema, para a previsão de um novo valor da série, durante o processamento de cada nova janela associa-se um grau  $D(R^j)$  a cada regra, multiplicando-se o grau de pertinência de cada termo, do antecedente e do conseqüente. Utiliza-se, então, a regra que possui maior grau entre as regras conflitantes.

## Tarefas de KDD e Métodos de Mineração de Dados: Resumo

Esta seção apresenta um resumo das principais tarefas de KDD e algumas alternativas de métodos que podem ser utilizados. Convém mais uma vez enfatizar que a tabela a seguir não esgota o universo de métodos de Mineração de Dados que pode ser aplicado em cada tarefa de KDD.

**Tabela 5.4** *Métodos de Mineração de Dados que pode ser aplicado em cada tarefa de KDD*

Tarefas de KDD	Métodos de Mineração de Dados
Descoberta de Associações	Basic, Apriori, DHP, Partition, DIC, ASCX-2P
Descoberta de Associações Generalizadas	Basic, Apriori, DHP, Partition, DIC, ASCX-2P
Descoberta de Seqüências	GSP, MSDD, SPADE
Descoberta de Seqüências Generalizadas	GSP, MSDD, SPADE
Classificação	Redes Neurais (Ex: Back-Propagation, RBF), C4.5, Rough Sets, Algoritmos Genéticos (Ex: Rule Evolver), CART, K-NN, Classificadores Bayesianos
Regressão	Redes Neurais (Ex: Back-Propagation), Lógica Nebulosa
Sumarização	C4.5, Algoritmos Genéticos (Ex: Rule Evolver)
Clusterização	K-Means, K-Modes, K-Prototypes, Fuzzy K-Means, Algoritmos Genéticos, Redes Neurais (Ex: Kohonen)
Previsão de Séries Temporais	Redes Neurais (Ex: Back-Propagation), Lógica Nebulosa (Ex: Wang-Mendel)

# Ferramentas de KDD

## Considerações Iniciais

Conforme comentado em capítulos anteriores, a complexidade inerente ao processo de KDD decorre de diversos fatores que podem ser subdivididos em dois conjuntos: fatores operacionais e fatores de controle (Fayyad et al., 1996b; Hellerstein et al., 1999).

Este capítulo tem como objetivo apresentar algumas das principais ferramentas utilizadas para auxiliar o homem na execução do processo de KDD. Exemplos de ferramentas utilizadas no tratamento dos fatores operacionais estão mencionados na seção “Ferramentas Operacionais de KDD”. A seção “Ferramentas para Controle do Processo de KDD” apresenta algumas das principais pesquisas voltadas ao desenvolvimento de ferramentas para tratamento de aspectos ligados ao controle do processo de KDD.

De forma a facilitar a descrição das ferramentas operacionais e das ferramentas de controle, são descritos dois modelos cujo objetivo é apresentar de forma resumida as principais características de cada tipo de ferramenta.

## Ferramentas Operacionais de KDD

Entre alguns dos principais exemplos de dificuldades operacionais comuns em aplicações de KDD, podemos citar a necessidade de manipulação de grandes e heterogêneos volumes de dados, o tratamento de resultados representados em diferentes formatos e a dificuldade de integração de diversos algoritmos específicos. Atualmente, encontram-se comercialmente disponíveis diversas ferramentas que implementam ambientes integrados para facilitar a execução das etapas operacionais de KDD, procurando minimizar diversas das dificuldades operacionais existentes. SAS Enterprise Miner, PolyAnalyst, Darwin, SPSS, Intelligent Miner, WizRule e Bramining são algumas destas ferramentas cuja descrição encontra-se neste capítulo.

## **Modelo Comparativo**

Esta seção tem por objetivo descrever um modelo que permita a caracterização resumida de ambientes integrados para realização de aplicações de KDD, também denominados de *Ferramentas Operacionais de KDD*. O modelo descritor de ferramentas operacionais de KDD, além de destacar importantes características observáveis em ferramentas de KDD, pode ainda ser utilizado como instrumento facilitador da comparação entre tais ferramentas.

As características do modelo em questão foram agrupadas em três conjuntos: (a) características relacionadas ao problema; (b) características relacionadas aos recursos; (c) características relacionadas aos resultados.

As **características relacionadas ao problema** se referem às facilidades de acesso aos conjuntos de dados a serem utilizados no processo de KDD bem como às facilidades de integração entre esses conjuntos. Entre tais características, o modelo comprehende:

- *Acesso a Fontes de Dados Heterogêneas* – Essa característica tem como objetivo informar a capacidade da ferramenta em acessar conjuntos de dados em formatos e ambientes diversificados tais como SGBDs específicos, formato texto, planilhas, entre outros.
- *Integração de Conjuntos de Dados* – Considerando que diversas aplicações de KDD envolvem conjuntos de dados distintos que se relacionam, a capacidade de integração entre tais conjuntos se faz necessária em ambientes integrados de KDD. Por exemplo, em *Data Warehouses* implementados no formato estrela, estruturas de dados de apoio contêm descrições para os códigos existentes na tabela central.

As **características relacionadas aos recursos** buscam retratar quais as facilidades existentes na ferramenta de KDD para realização das tarefas necessárias durante o processo de KDD. Entre tais características estão:

- *Facilidade para Inclusão de Novas Operações* – Essa característica envolve o conceito de operação de KDD. Uma *operação de KDD* é uma especificação lógica de um conjunto de *métodos de KDD*. Por exemplo, *Classificação* é uma operação de KDD, que pode ser implementada por diversos métodos tais como: redes neurais backpropagation, classificadores bayesianos, C4.5, entre outros. Essa característica deve indicar se a ferramenta possui ou não meios para inclusão de novas operações de KDD.
- *Facilidade para Inclusão de Novos Métodos* – Conforme comentário anterior, uma mesma operação de KDD pode ser implementada por diversos métodos. A implementação de cada método tem como base uma ou mais técnicas, tais como Estatística, Redes Neurais, Algoritmos Genéticos, Ló-

gica Nebulosa, entre outras. Essa característica deve indicar se a ferramenta possui ou não meios para inclusão de novos métodos de KDD.

- *Recursos para Planejamento de Ações* – O atual estado da arte em KDD mostra que diversos métodos podem ser utilizados para realização de diversas tarefas ao longo do processo de KDD. Assim sendo, a escolha de seqüências adequadas de métodos e que devam ser utilizadas vem se tornando uma decisão cada vez menos trivial em aplicações de KDD. O processo de escolha de tais seqüências (denominadas Planos de Ação) é chamado de Planejamento de Ações. Assim sendo, essa característica tem como objetivo indicar se a ferramenta dispõe de recursos que auxiliem o planejamento das ações a serem realizadas ao longo das aplicações de KDD.
- *Processamento Paralelo/Distribuído* – Essa característica tem como objetivo indicar se a ferramenta dispõe de recursos computacionais que viabilizem a execução paralela e/ou distribuída de ações no processo de KDD.
- *Operações/Métodos Disponíveis* – Essa característica tem como objetivo relacionar os métodos disponíveis na ferramenta, assim como as operações de KDD por eles implementadas.

As características relacionadas aos resultados abrangem os meios utilizados para armazenamento e manipulação dos resultados produzidos ao longo do processo de KDD.

- *Estruturas para Armazenamento de Modelos de Conhecimento* – Essa característica tem como objetivo informar a capacidade da ferramenta em armazenar modelos de conhecimento com diferentes representações em estruturas de dados próprias que viabilizem a posterior manipulação de tais modelos.
- *Estruturas para Armazenamento de Históricos de Ações* – Essa característica tem como objetivo informar a capacidade da ferramenta em armazenar históricos contendo todas as ações realizadas ao longo das aplicações de KDD. A presença deste item em uma ferramenta de KDD viabiliza que, posteriormente, tais históricos sejam utilizados em uma análise crítica e de reavaliação das ações realizadas. A partir deste tipo de análise, metodologias e políticas de realização de KDD podem ser propostas e aperfeiçoadas.

### ***Alguns Exemplos de Ferramentas Operacionais de KDD***

Atualmente, existem diversas ferramentas que implementam ambientes integrados para facilitar a execução das etapas operacionais de KDD. A Tabela 6.1 apresenta algumas destas ferramentas. Cada uma delas encontra-se comentada e enquadrada no modelo descritor de ferramentas operacionais de KDD. Convém enfatizar que o detalhamento das ferramentas apresentou variações em decorrência da disponibilidade de informações publicadas pelos respectivos fabricantes.

**Tabela 6.1** Exemplos de ferramentas de KDD e sites para consulta

Nome	Tarefas de KDD – Exemplos	Fabricante
SPSS/Clementine	Classificação, Regras de Associação, Clusterização, Seqüências, Detecção de Desvios.	SPSS Inc. <a href="http://www.spss.com">www.spss.com</a>
PolyAnalyst	Classificação, Regressão, Regras de Associação, Clusterização, Sumarização, Detecção de Desvios.	Megaputer Intelligence <a href="http://www.megaputer.com">www.megaputer.com</a>
Weka	Classificação, Regressão, Regras de Associação.	University of Waikato <a href="http://www.cs.waikato.ac.nz">www.cs.waikato.ac.nz</a>
Darwin	Classificação	Thinking Machines <a href="http://en.wikipedia.org/wiki/thinking_machines">http://en.wikipedia.org/wiki/thinking_machines</a>
Intelligent Miner	Classificação, Regras de Associação, Seqüências, Clusterização, Sumarização.	IBM Corp. <a href="http://www.ibm.com">www.ibm.com</a>
WizRule	Sumarização, Classificação, Detecção de Desvios.	WizSoft Inc. <a href="http://www.wizsoft.com">www.wizsoft.com</a>
Bramining	Classificação, Regras de Associação, Regressão, Sumarização.	Graal Corp. <a href="http://www.graal-corp.com.br">www.graal-corp.com.br</a>
SAS Enterprise Miner	Classificação, Regras de Associação, Regressão, Sumarização.	SAS Inc. <a href="http://www.sas.com">www.sas.com</a>
Oracle Data Mining	Classificação, regressão, associação, clusterização, e mineração de textos	Oracle <a href="http://www.oracle.com">www.oracle.com</a>

### ***SPSS/Clementine***

O SPSS (*Statistical Package for the Social Sciences*) é um software que permite o gerenciamento e a análise estatística de dados. O SPSS é formado por uma família de produtos, que podem ser adquiridos e utilizados de forma independente, com destaque especial para o Clementine, cujo desempenho pode ser observado em diversas aplicações comerciais e científicas. O Clementine foi vencedor por duas vezes do prêmio *UK Government's SMART award for innovation* para inovações nas áreas da indústria e do comércio britânicos.

O SPSS possui uma estrutura de organização de dados própria mas permite a importação de dados em diversos formatos, tais como: FoxPro, Access, DBase, ACSII, XLS, Oracle e SQL Server.

**Tabela 6.2 Resumo das Características do SPSS/Clementine**

<b>Características</b>	<b>Valores</b>	
Acesso a Fontes de Dados Heterogêneas	Sim	
Integração de Conjuntos de Dados	Sim	
Facilidade para Inclusão de Novas Operações	Sim	
Facilidade para Inclusão de Novos Métodos	Sim	
Recursos para Planejamento de Ações	Sim	
Processamento Paralelo/Distribuído	Inf. não disponível	
Operações/Métodos Disponíveis	Visualização de Dados	Histogramas, Gráficos (de linha, de correlação, de pizza), Regiões de Decisão de Classificação
	Redução de Dados	Seleção, Amostragem, Merge, Aplicação de Filtros
	Limpeza de Dados	Substituição
	Codificação de Dados	Discretização e Transformação automática e manual
	Classificação	Árvore de Decisão, Métodos Estatísticos/Lineares, <i>Perceptron</i> , Regras de Indução
	Clusterização	K-Means, <i>Kohonen</i> , Regras de Associação
	Simplificação de Resultados	Poda de Árvore
	Organização de Resultados	Inf. não disponível
	Apresentação de Resultados	Tabela, Texto
Estruturas para Armazenamento de Modelos de Conhecimento	Sim	
Estruturas para Armazenamento de Históricos de Ações	Sim	

### *PolyAnalyst*

De forma análoga ao SPSS, o PolyAnalyst é uma família de produtos com aplicações específicas. Um destaque especial para o TextAnalyst, aplicado em Mineração de Textos.

**Tabela 6.3 Resumo das Características do PolyAnalyst**

<b>Características</b>	<b>Valores</b>
Acesso a Fontes de Dados Heterogêneas	Sim
Integração de Conjuntos de Dados	Sim
Facilidade para Inclusão de Novas Operações	Não

**Tabela 6.3** Continuação

Características		Valores
Facilidade para Inclusão de Novos Métodos		Não
Recursos para Planejamento de Ações		Sim
Processamento Paralelo/Distribuído		Não
Operações/Métodos Disponíveis	Visualização de Dados	Gráficos 3D interativos, Histogramas, Distribuição de Freqüências
	Redução de Dados	Aplicação de Filtros, Seleção Randômica
	Limpeza de Dados	Substituição
	Codificação de Dados	Conversão automática de tipos de dados compatíveis
	Classificação	Árvore de Decisão, <i>Nearest Neighbor</i> , Algoritmos Genéticos, Regressão Linear
	Clusterização	Inf. não disponível
	Simplificação de Resultados	Corte de Regras
	Organização de Resultados	Agregação de nós da árvore de decisão segundo um critério
	Apresentação de Resultados	Texto e Gráficos
Estruturas para Armazenamento de Modelos de Conhecimento		Sim
Estruturas para Armazenamento de Históricos de Ações		Sim

### Weka

O Weka é uma ferramenta de código aberto, aplicável em KDD, flexível, desenvolvida na linguagem Java pelo curso de Ciência da Computação da universidade de Waikato na Nova Zelândia.

A utilização da ferramenta pode ser realizada de diversas maneiras. Possui quatro diferentes implementações de interface, que permitem que todos os seus algoritmos sejam chamados diretamente via código Java. As interfaces são:

- 1) *Simple Client* – Nessa interface, a interação do usuário com o Weka ocorre por meio de linhas de comando. Requer um profundo conhecimento do programa, porém é extremamente flexível e ágil para usuários avançados.
- 2) *Explorer* – Trata-se da interface de utilização mais comum, e enquadra separadamente as etapas de pré-processamento (filtros), mineração de dados (associação, clusterização e classificação) e pós-processamento (apresentação de resultados).
- 3) *Experimenter* – Constitui um ambiente de experimentação, em que testes estatísticos podem ser conduzidos a fim de avaliar o desempenho de diferentes algoritmos de aprendizado.

- 4) *KnowledgeFlow* – É uma ferramenta gráfica ainda em desenvolvimento que permite o planejamento de ações, na construção de um fluxo de processos de KDD.

O Weka possui implementados diversos métodos de associação, classificação e clusterização. A inclusão ou remoção de novos métodos pode ser realizada de forma simples e rápida, o que torna a ferramenta customizável e expansível.

O Weka suporta a abertura direta de arquivos ARFF, CSV, C45. Porém, apenas consegue manipular os ARFF. Este é um arquivo ASCII usado para definir atributos e seus valores. Maiores detalhes sobre o formato ARFF podem ser encontrados em <http://www.cs.waikato.ac.nz/~ml/weka/arff.html>.

O Weka também permite a visualização gráfica dos dados em forma de histogramas, e a apresentação de resultados em árvores de decisão, diagramas de dispersão, além de prover modelos gráficos para montagem de redes neurais.

**Tabela 6.4 Resumo das Características do Weka**

Características	Valores	
Acesso a Fontes de Dados Heterogêneas	Sim	
Integração de Conjuntos de Dados	Não	
Facilidade para Inclusão de Novas Operações	Sim	
Facilidade para Inclusão de Novos Métodos	Sim	
Recursos para Planejamento de Ações	Sim	
Processamento Paralelo/Distribuído	Não	
Operações/Métodos Disponíveis	Visualização de Dados	Distribuição de Freqüências; Medidas de Dispersão; Histogramas
	Redução de Dados	Amostragem
	Limpeza de Dados	Substituição
	Codificação de Dados	Discretização automática e manual
	Classificação	Árvores de Decisão, Bayes, Redes Neurais...
	Clusterização	Simple-KMeans, Cobweb, FarthestFirst...
	Simplificação de Resultados	N/D
	Organização de Resultados	Agrupamento de Padrões; Ordenamento de Padrões
	Apresentação de Resultados	Conjunto de Regras; Árvores de Decisão
Estruturas para Armazenamento de Modelos de Conhecimento	Sim	
Estruturas para Armazenamento de Históricos de Ações	Sim	

## Darwin

Desenvolvido pela *Thinking Machines Corp.*, o Darwin é uma ferramenta disponível para operação nas plataformas *Windows*, *Sun Solaris*, e *HP-UX*.

Fornece interfaces simples, baseadas no modelo Windows, e oferece também uma implementação de vários algoritmos de Mineração de Dados, que podem ser executados paralelamente. Pode ser utilizado em conjunto com diversos SGDBs, incluindo o Oracle. É capaz de implementar o modelo de processo de KDD por completo. Acesso a arquivos texto e SAS também são permitidos. Códigos em C, C++ e Java podem ser exportados.

Também possui a flexibilidade para utilizar diversos algoritmos e escolher o mais adequado para um dado problema.

Outra característica é a sua abertura e modularidade, que permitem que sejam adicionados novos algoritmos de KDD. Possui recursos que simplificam a sua integração com outros produtos.

**Tabela 6.5 Resumo das Características do Darwin**

Características	Valores	
Acesso a Fontes de Dados Heterogêneas	Sim	
Integração de Conjuntos de Dados	Sim	
Facilidade para Inclusão de Novas Operações	Sim	
Facilidade para Inclusão de Novos Métodos	Sim	
Recursos para Planejamento de Ações	Sim	
Processamento Paralelo/Distribuído	Sim	
Operações/Métodos Disponíveis	Visualização de Dados	Histogramas, Gráficos (de pizza, de linha, de barra), Medidas de Dispersão, <i>Rotating Scatter, Conditional Plots</i>
	Redução de Dados	Projeção, Seleção, Amostragem
	Limpeza de Dados	Substituição
	Codificação de Dados	Discretização, Randomização, Transformação automática e manual
	Classificação	Árvore de Decisão (CART), Redes Neurais, <i>K-Nearst Neighbors</i>
	Normalização	Sim
	Clusterização	Não
	Simplificação de Resultados	Poda de árvore
	Organização de Resultados	Inf. não disponível
	Apresentação de Resultados	Através do MS Excel (Planilha, gráficos)
Estruturas para Armazenamento de Modelos de Conhecimento	Sim	
Estruturas para Armazenamento de Históricos de Ações	Inf. não disponível	

### *Intelligent Miner*

Fabricado pela IBM, O DB2 Intelligent Miner for Data, ou simplesmente Intelligent Miner, possui versões para operação nas Plataformas Windows, Solaris, AIX, OS/390, e OS/400.

O Intelligent Miner não é dependente do sistema da IBM, podendo também ser utilizado junto a outros SGDBs relacionais. A IBM oferece também o Intelligent Miner for Text, que realiza atividades de mineração em dados de texto, incluído a filtragem de e-mail e páginas Web.

Escalável e com suporte para várias plataformas, o pacote DB2 Intelligent Miner for Data oferece um conjunto de ferramentas apto a fornecer uma estrutura que suporta o processo iterativo de descoberta de conhecimentos.

O Intelligent Miner permite a utilização de algoritmos de mineração, de forma individual ou combinada, para solucionar problemas de KDD.

Também possui uma interface de programação de aplicativos que permite o desenvolvimento de aplicações personalizadas de Mineração de Dados.

**Tabela 6.6 Resumo das Características do Intelligent Miner**

Características	Valores	
Acesso a Fontes de Dados Heterogêneas	Sim	
Integração de Conjuntos de Dados	Sim	
Facilidade para Inclusão de Novas Operações	Sim	
Facilidade para Inclusão de Novos Métodos	Sim	
Recursos para Planejamento de Ações	Sim	
Processamento Paralelo/Distribuído	Sim	
Operações/Métodos Disponíveis	Visualização de Dados	Histogramas, Gráficos (de pizza, de linha, de barra)
	Redução de Dados	Cálculo de valores, Seleção, Amostragem, Aplicação de Filtros
	Limpeza de Dados	Substituição, Descarte
	Codificação de Dados	Discretização, Randomização, e Transformação automática e manual
	Classificação	Árvore de Decisão (CART modificado), Métodos Estatísticos, Redes Neurais
	Normalização	Não
	Clusterização	K-Means
	Simplificação de Resultados	Inf. não disponível

**Tabela 6.6** Continuação

Características		Valores
Operações/Métodos Disponíveis	Organização de Resultados	Agrupamento de Padrões, Ordenamento de Padrões
	Apresentação de Resultados	Gráficos (pizza, barras), Tabelas, Árvores, Clusters
Estruturas para Armazenamento de Modelos de Conhecimento		Sim
Estruturas para Armazenamento de Históricos de Ações		Sim

### *Wiz Rule*

O Wiz Rule é um software de mineração de dados israelense que foi desenvolvido para examinar e descrever conjuntos de dados, detectando possíveis erros dentre os dados analisados. As relações entre dados identificadas pelo WizRule podem ser apresentadas na forma de regras de produção ou como fórmulas matemáticas. Muito útil em aplicações envolvendo Detecção de Desvios, o WizRule abstrai regras e fórmulas, destacando os registros do banco de dados que não satisfazem ao conhecimento descoberto. Além disso, possui bom desempenho na identificação de possíveis erros de escrita nos valores das variáveis analisadas.

**Tabela 6.7** Resumo das Características do WizRule

Características		Valores
Acesso a Fontes de Dados Heterogêneas		Sim
Integração de Conjuntos de Dados		Não
Facilidade para Inclusão de Novas Operações		Não
Facilidade para Inclusão de Novos Métodos		Não
Recursos para Planejamento de Ações		Não
Processamento Paralelo/Distribuído		Informação não disponível
Operações/Métodos Disponíveis	Visualização de Dados	Relatórios e gráficos
	Redução de Dados	Não
	Limpeza de Dados	Não
	Codificação de Dados	Não
	Classificação	Informação não disponível
	Normalização	Não
	Clusterização	Não
	Simplificação de Resultados	Poda de árvore

**Tabela 6.7** Continuação

Características		Valores
Operações/Métodos Disponíveis	Organização de Resultados	Não
	Apresentação de Resultados	Através do MS Excel (Planilha, gráficos)
Estruturas para Armazenamento de Modelos de Conhecimento		Não
Estruturas para Armazenamento de Históricos de Ações		Informação não disponível

### Bramining

O Bramining é uma ferramenta nacional de mineração de dados produzida ao longo de três dissertações de mestrado realizadas na PUC-Rio e no IME. O Bramining disponibiliza ambiente para a realização do processo KDD e, conceitualmente, possui três níveis, denominados níveis funcionais. São eles: (a) Nível dos Métodos; (b) Nível das Operações; (c) Nível das Etapas.

O Nível dos Métodos é o nível funcional mais baixo e contém os métodos que se encontram disponíveis no Bramining para utilização durante a realização de aplicações de KDD. Os métodos são classificados em operações. Uma operação de KDD é uma especificação lógica de um grupo de métodos que têm a mesma finalidade. As operações disponíveis no Bramining compõem o nível funcional intermediário da ferramenta, denominado Nível das Operações. O Nível das Etapas é o nível funcional mais elevado. Nele as operações de KDD são agrupadas nas etapas do processo de KDD: Pré-processamento, Mineração de Dados e Pós-processamento.

O processamento iterativo e interativo de uma aplicação de KDD pelo Bramining requer, a cada ação, que seja definido o método que deverá implementar tal ação. A hierarquia mencionada anteriormente é utilizada para auxiliar usuários de KDD inexperientes a identificar, através de filtros, os métodos disponíveis que viabilizam a operação desejada. Uma vez selecionado o método desejado, a sua execução demanda a especificação da estrutura de entrada e dos parâmetros específicos do referido método. Cada método, após ser executado, apresenta uma interface específica com os resultados produzidos. Esses resultados podem ser novos conjuntos de dados ou mesmo modelos de conhecimento descobertos.

**Tabela 6.8** Resumo das Características do Bramining

Características	Valores
Acesso a Fontes de Dados Heterogêneas	Sim
Integração de Conjuntos de Dados	Não
Facilidade para Inclusão de Novas Operações	Sim

**Tabela 6.8** Continuação

	<b>Características</b>	<b>Valores</b>
Facilidade para Inclusão de Novos Métodos		Sim
Recursos para Planejamento de Ações		Não
Processamento Paralelo/Distribuído		Não
Operações/Métodos Disponíveis	Visualização de Dados	Distribuição de Freqüências; Medidas de Dispersão
	Redução de Dados	Amostragem
	Limpeza de Dados	Substituição
	Codificação de Dados	Discretização automática e manual
	Classificação	C4.5; Rough Sets; Back-Propagation
	Clusterização	C-Means
	Simplificação de Resultados	Corte de Regras; Poda de Árvore
	Organização de Resultados	Agrupamento de Padrões; Ordenamento de Padrões
	Apresentação de Resultados	Conjunto de Regras; Árvores de Decisão
Estruturas para Armazenamento de Modelos de Conhecimento		Sim
Estruturas para Armazenamento de Históricos de Ações		Sim

### **SAS Enterprise Miner**

O SAS é um software estatístico voltado à análise de dados. Possui uma família de produtos, que podem ser adquiridos e utilizados de forma independente, com destaque especial para o módulo de Mineração de Dados. Esse módulo dispõe de diversos algoritmos de análise, além de recursos para o planejamento de ações e encadeamento dos algoritmos.

**Tabela 6.9** Resumo das Características do SAS

	<b>Características</b>	<b>Valores</b>
Acesso a Fontes de Dados Heterogêneas		Sim
Integração de Conjuntos de Dados		Sim
Facilidade para Inclusão de Novas Operações		Sim
Facilidade para Inclusão de Novos Métodos		Sim
Recursos para Planejamento de Ações		Sim
Processamento Paralelo/Distribuído		Sim

**Tabela 6.9** Continuação

	<b>Características</b>	<b>Valores</b>
Operações/Métodos Disponíveis	Visualização de Dados	Gráficos (de linha, de barra de pizza, etc.) Histogramas
	Redução de Dados	Comandos SQL – LDD
	Limpeza de Dados	Checkagem automática de valores discrepantes, Substituição
	Codificação de Dados	Discretização e Transformação automática e manual
	Classificação	RBF, Perceptron Multi Camadas, CART, C4.5, Regressão Linear
	Clusterização	Kohonen, WARD
	Simplificação de Resultados	Poda de árvore, Seleção da melhor árvore
	Organização de Resultados	Ordenação e Seleção de Regras
	Apresentação de Resultados	Gráficos, Texto, Árvores
Estruturas para Armazenamento de Modelos de Conhecimento		Sim
Estruturas para Armazenamento de Históricos de Ações		Sim

### *Oracle Data Mining*

O ODM, Oracle Data Mining, é um software de Mineração de Dados em que todas as atividades de descoberta de conhecimento ocorrem no mesmo ambiente do gerenciador de banco de dados Oracle, provendo uma plataforma integrada simples, segura e escalável. Tal integração representa um diferencial importante em favor de sua utilização, pois, ao contrário das demais ferramentas, não requer a extração prévia dos dados para que estes sejam processados pelos métodos de KDD.

O ODM permite a realização de tarefas de classificação, regressão, associação, clusterização, e mineração de textos.

A infra-estrutura de análise de dados e desenvolvimento de aplicações integrada a Mineração de Dados é suportada através de ODM's Java API, ODM's PL/SQL API e ODM's graphical user interface.

**Tabela 6.10** Resumo das Características do Oracle Data Mining

	<b>Características</b>	<b>Valores</b>
Acesso a Fontes de Dados Heterogêneas		Sim
Integração de Conjuntos de Dados		Sim
Facilidade para Inclusão de Novas Operações		Sim

**Tabela 6.10** Continuação

<b>Características</b>		<b>Valores</b>
Facilidade para Inclusão de Novos Métodos		Sim
Recursos para Planejamento de Ações		Sim
Processamento Paralelo/Distribuído		Sim
Operações/Métodos Disponíveis	Visualização de Dados	ORACLE Data Miner's Graphical User Interface – ODM Data Miner's GUI - Interface gráfica para análise, aplicação e avaliação de modelos de mineração. Permite a geração de histogramas, gráficos, e tabelas
	Redução de Dados	Informação não disponível
	Limpeza de Dados	Informação não disponível
	Codificação de Dados	Informação não disponível
	Classificação	A classificação é realizada através da técnica de aprendizado supervisionado, utilizando os algoritmos de Classificação Bayesiana Ingênua, Máquinas de Suporte Vetorial e Mineração de Textos
	Clusterização	Algoritmos K-Means, e Orthogonal Partitioning Clustering
	Simplificação de Resultados	Informação não disponível
	Organização de Resultados	Informação não disponível
	Apresentação de Resultados	Apresentação de resultados por meio de gráficos, tabelas e histogramas
	Estruturas para Armazenamento de Modelos de Conhecimento	Informação não disponível
Estruturas para Armazenamento de Históricos de Ações		Sim

## Ferramentas para Controle do Processo de KDD

Certamente os fatores relacionados ao controle do processo de KDD contribuem de forma significativa para a notória complexidade inerente a este processo. Tais fatores envolvem considerações sobre como conduzir processos de KDD. Conforme comentado no Capítulo 2, entre os principais exemplos de fatores de controle do processo de KDD podem ser citados:

- A dificuldade na formulação precisa dos objetivos a serem alcançados em um processo de KDD.
- A dificuldade na escolha, dentre diversas possibilidades, de um algoritmo de mineração de dados com potencial para geração de resultados satisfatórios.

- A dificuldade na escolha, dentre inúmeras possibilidades, das alternativas de pré-processamento dos dados a serem utilizadas em cada situação.
- A dificuldade freqüente na escolha cuidadosa da parametrização adequada a diversos algoritmos diante de cada nova situação.
- Uma freqüente perda de concentração do analista por parte do analista de KDD causada muitas vezes pelos longos tempos de experimentação de inúmeras alternativas na busca por melhores resultados.
- A incapacidade humana de memorização de resultados e alternativas processadas na medida em que o tempo passa e o número de experimentos realizados aumenta.
- De uma maneira geral, a dificuldade em perceber e interpretar adequadamente inúmeros fatos observáveis ao longo de processos de KDD e a complexidade em conjugar dinamicamente tais interpretações de forma a decidir que ações devem ser realizadas em cada caso.

Atualmente, algumas ferramentas que possam auxiliar o homem no controle de aplicações de KDD encontram-se em fase de pesquisa e desenvolvimento. Metal, IKDD, IDEA, e Consultant são exemplos de ferramentas desta natureza e estão descritas nas próximas seções.

### **Modelo Comparativo**

O modelo para comparação de mecanismos de assistência ao controle do processo de KDD comprehende quatro conjuntos de características que se complementam: **Abrangência, Abordagem de Assistência, Versatilidade e Evolução, e Autonomia**. As descrições detalhadas de cada conjunto e de suas respectivas características encontram-se a seguir.

As características referentes à **Abrangência** de um trabalho nesta área buscam identificar que aspectos dos processos de KDD são considerados no escopo do mecanismo de assistência ou do trabalho estudado:

- **Etapas do processo de KDD** – Essa característica tem como objetivo especificar que etapa ou etapas do processo de KDD são assistidas pelo trabalho analisado, ou seja, qualquer subconjunto não vazio das etapas operacionais de KDD.
- **Tarefas de KDD** – Nessa característica são indicadas tarefas de KDD para as quais o trabalho analisado oferece algum tipo de assistência. Pode conter uma ou mais tarefas. Como exemplos de tarefas de KDD podem ser citadas (Fayyad et al., 1996a b; Weiss & Indurkhy, 1998; Han, 1996): *Classificação, Associação, Seqüência, Detecção de Desvios, Clusterização, Previsão de Séries Temporais*, entre outras.

- **Algoritmos e Técnicas de KDD** – Devem ser indicados nessa característica os algoritmos e as técnicas de KDD, também denominados métodos de KDD, para os quais o trabalho estudado oferece assistência.
- **Dimensões da Assistência em KDD** – Conforme mencionado anteriormente, todo processo de KDD requer um componente responsável pelas decisões sobre como conduzir tal processo. Uma análise das principais funções envolvidas neste tipo de decisão revela três grupos de funcionalidades (Engels, 1996; Verdenius & Engels, 1997): *assistência na definição de objetivos do processo de KDD*, *assistência no planejamento das ações* a serem realizadas ao longo do processo em busca dos objetivos definidos, e *assistência na execução das ações planejadas*. Cada um desses grupos será referenciado como uma dimensão de assistência. Assim sendo, esta característica indica o(s) momento(s) de atuação do mecanismo de assistência do trabalho analisado.

A **Abordagem de Assistência** busca retratar os recursos utilizados pelo mecanismo de assistência do trabalho analisado bem como sua forma de operação. As características envolvidas neste conjunto são:

- **Mecanismo de Assistência** – Essa característica tem como objetivo indicar quais são as bases computacionais para o funcionamento do mecanismo de assistência descrito.
- **Acoplamento entre Mecanismo de Assistência e de Execução** – Esse item busca retratar o grau de interdependência entre o mecanismo de assistência e o mecanismo de execução do processo de KDD. O valor *alto* indica que o mecanismo de assistência encontra-se intrinsecamente ligado ao mecanismo de execução. Percebe-se, neste caso, uma maior dificuldade de substituição do mecanismo de execução. O valor *baixo*, por sua vez, denota um maior grau de independência entre os mecanismos. Espera-se neste caso, portanto, uma maior facilidade em substituir o mecanismo de execução, sem a necessidade de alterações significativas no mecanismo de assistência.
- **Recursos de Paralelismo e Distribuição** – Esse item informa se o mecanismo de assistência possui uma configuração que permita sua atuação em ambientes de processamento paralelo e distribuído.
- **Suporte a iterações no Processo de KDD** – Nesse item é informado se o mecanismo analisado fornece assistência incremental na medida em que ocorram vários ciclos no processo de KDD. Assistência incremental refere-se à habilidade de auxílio em novas operações levando em consideração ações realizadas e resultados decorrentes de ciclos anteriores.
- **Utilização de Conhecimento sobre o Domínio da Aplicação** – Essa característica refere-se à habilidade do mecanismo de assistência em considerar conhecimentos prévios sobre o domínio da aplicação de KDD.

- **Utilização de Conhecimento sobre KDD** – Nessa característica deve ser expressa a habilidade do mecanismo de assistência em incorporar e utilizar conhecimento sobre como realizar processos de KDD. Convém enfatizar que este tipo de conhecimento independe do domínio da aplicação de KDD.
- **Forma de Representação do Conhecimento** – Esse item indica quais as formas de representação do conhecimento processadas pelo mecanismo de assistência. Essa informação assume caráter obrigatório sempre que o modelo contenha valor *sim* para uma das duas características imediatamente anteriores.
- **Tipo de Metaconhecimento** – Indica a origem do conhecimento sobre KDD previamente incorporado ao mecanismo de assistência. Conforme apresentado em (Brazdil, 1998; Brazdil & Soares, 2000), são três os tipos de metaconhecimento: *Teórico*, *Experimental* e *Híbrido*. O *Metaconhecimento Teórico* abrange todo tipo de conhecimento explicitamente fornecido por especialistas em KDD. O *Metaconhecimento Experimental*, por outro lado, compreende os conhecimentos extraídos a partir de registros históricos de experiências anteriores. A combinação de Metaconhecimento Teórico e Metaconhecimento Experimental denomina-se *Metaconhecimento Híbrido*.

O conjunto de características sobre **Versatilidade e Evolução** busca expressar, além do grau de independência entre o mecanismo de assistência e as aplicações que tenham sido descritas, a facilidade do mecanismo em adquirir novos conhecimentos, de forma automática ou não:

- **Independência do Mecanismo de Assistência e Aplicações** – Nesse item busca-se indicar a necessidade de alterações no mecanismo de assistência em questão a fim de que este possa ser adaptado a novas aplicações de KDD.
- **Facilidades para Incorporação de Novos Conhecimentos** – Essa característica indica se o mecanismo de assistência possui recursos que permitam a incorporação não automática de novos conhecimentos em sua estrutura. Tais recursos devem viabilizar a incorporação de novos conhecimentos sem a necessidade de alteração de códigos de programas.
- **Capacidade de Aprendizado** – Esse item tem como objetivo informar se o mecanismo de assistência possui recursos que o permitam aprender, de forma automática, sobre como conduzir processos de KDD a partir de experiências anteriores.

As características sobre **Autonomia** têm por função retratar a independência do mecanismo de assistência em relação ao analista humano, na tomada de deci-

são nas várias dimensões de assistência ao processo de KDD. Tais características encontram-se relacionadas e comentadas a seguir:

- **Interação Homem-Máquina na Definição de Objetivos de KDD** – Essa característica informa o tipo de participação humana na dimensão *Definição de Objetivos de KDD*. A Tabela 6.11 apresenta os valores possíveis para essa característica e os respectivos significados.
- **Interação Homem-Máquina no Planejamento de Ações de KDD** – Essa característica informa o tipo de participação humana na dimensão *Planejamento de Ações de KDD*. A Tabela 6.11 apresenta os valores possíveis para essa característica e os respectivos significados.
- **Interação Homem-Máquina na Execução de Ações de KDD** – Essa característica informa o tipo de participação humana na dimensão *Execução de Ações de KDD*. A Tabela 6.11 apresenta os valores possíveis para essa característica e os respectivos significados. O item *Etapas do Processo de KDD* do conjunto de características relacionadas à *Abrangência* contém, sempre que aplicável, as etapas cuja execução seja assistida pelo mecanismo em questão.

**Tabela 6.11 Valores das características referentes à Autonomia do Mecanismo de Assistência**

Valor	Descrição
Não se Aplica	Utilizado quando a dimensão em análise não constar no item <i>Dimensões da Assistência</i> do conjunto de características referentes à <i>Abrangência</i> .
Inexistente	Indica que a dimensão em análise é realizada pelo mecanismo de assistência sem o auxílio do homem.
Necessária	Indica que a dimensão em análise somente pode ser realizada pelo mecanismo de assistência com o auxílio do homem.
Possível	Indica que o mecanismo de assistência pode, como opção, requisitar a participação do homem durante na execução da função em análise.

### **Alguns Exemplos de Ferramentas de Controle em KDD**

Atualmente, existem poucas ferramentas desenvolvidas com o propósito de auxiliar o homem na condução do processo de KDD. Muitas delas pertencem a projetos de pesquisa e encontram-se ainda em fase de elaboração. A Tabela 6.12 indica os principais trabalhos associados ao desenvolvimento de algumas dessas ferramentas e o nível de assistência fornecido. Cada um deles encontra-se comentado e enquadrado no modelo descritor de mecanismos de assistência ao controle de processos de KDD. Convém enfatizar que o detalhamento dessas ferramentas também apresentou variações em decorrência da disponibilidade de informações publicadas pelos respectivos grupos de pesquisa.

**Tabela 6.12 Exemplos de pesquisas na elaboração de ferramentas de KDD e seus respectivos Níveis de Assistência**

Trabalhos	Dimensões de Assistência em KDD		
	Definição de Objetivos	Planejamento de Ações de KDD	Execução de Ações de KDD
(Livingston et al., 2000)			X
(Livingston, 2001)			X
(Livingston et al., 2001)			X
(Shen e Leng, 1996)			X
(Bernstein et al., 2002)		X	
(Suyama & Yamaguchi, 1998)		X	
(Gama & Brazdil, 1995)		X	
(Brazdil, 1998)		X	
(Brazdil & Soares, 2000)		X	
(Brazdil et al., 2003)		X	
(Soares et al., 2001)		X	
(Bensusan et al., 2000a, b)		X	
(Frunkranz & Petrak, 2001)		X	
(Kalousis & Theoharis, 1999)		X	
(Spiliopoulou et al., 1998)		X	
(Keller et al., 2000)		X	
(Pfahringer & Bensusan, 2001)		X	
(Morik, 2000)		X	
(Michie et al., 1994)		X	
(Brodley, 1993)		X	
(Goldschmidt et al., 2002d)		X	
(Goldschmidt et al., 2002b, c)	X		
(Goldschmidt et al., 2003)	X		
(Engels et al., 1997)	X	X	
(Jensen et al., 1999)	X	X	
(Kerber et al., 1998)	X	X	
(Amant & Cohen, 1998)		X	X
(Goldschmidt, 2003)	X	X	X

## Assistência à Execução de Ações em Aplicações de KDD

Entre as *Dimensões de Assistência em KDD*, a *Assistência à Execução de Ações em Aplicações de KDD* é certamente a que concentra maior volume de pesquisas. Grande parte destes trabalhos envolve a otimização dos parâmetros de algoritmos de Mineração de Dados (Goldberg, 1989; Fayyad et al., 1996a, b; Curran & O'Riordan, 2002). São estudos em que o espaço de busca se restringe ao espaço de valores possíveis para os parâmetros dos algoritmos. Nestes casos, o espaço de busca por padrões de conhecimento é percorrido pelos algoritmos de Mineração de Dados a partir dos parâmetros especificados pelos mecanismos de assistência. De outro lado, estão os esforços no desenvolvimento de assistência à execução que percorram diretamente o espaço de padrões de conhecimento. Assim sendo, optamos por citar e descrever de forma resumida apenas alguns dos principais trabalhos pertencentes ao segundo grupo.

Em (Livingston et al., 2000; Livingston, 2001; Livingston et al., 2001), os autores propuseram uma forma de assistência baseada nos conceitos de *Agenda* e *Justificativa* utilizados em (Lenat & Brown, 1984). Uma Agenda é um repositório em que são armazenadas tarefas a serem executadas pelo sistema. Tarefas são operações voltadas à descrição de atributos do conjunto de dados. A identificação, a inclusão e a ordenação de tarefas na estrutura de Agenda são realizadas com base em heurísticas previamente formuladas pelo homem.

O mecanismo proposto em (Livingston, 2001) encontra-se voltado especificamente para a realização da tarefa de KDD denominada Sumarização ou Descrição da Base de Dados (Fayyad et al., 1996a b; Weiss e Indurkhya, 1998). Para avaliação do mecanismo proposto, Livingston desenvolveu um protótipo denominado *HAMB – Heuristic Automatic Model Builder*. O HAMB opera de forma integrada ao RL (Provost & Buchanan, 1995), algoritmo para indução de regras que descrevam as características de atributos previamente definidos. Nessa arquitetura, primeiro o HAWB define o atributo a ser descrito. Em seguida aciona o RL para descrição do referido atributo. Convém ressaltar que o pré-processamento do conjunto de dados é um requisito necessário à execução do HAMB.

Na abordagem proposta em (Shen e Leng, 1996), o processo de descoberta de conhecimento baseia-se em *metaconsultas* que são aplicadas em um algoritmo de aprendizado relacional, produzindo todas as especializações possíveis das metaconsultas fornecidas. Uma metaconsulta, também denominada *meta-padrão*, é uma expressão da *Lógica de Segunda Ordem* (Russel & Norvig, 1995), que descreve o tipo de padrão a ser descoberto. Um exemplo de metapadrão que expressa transitividade encontra-se em (1.1). Em (1.2) é apresentado um exemplo de especialização de (1.1).

$$P(X,Y) \wedge Q(Y,Z) \Rightarrow R(X,Z) \quad (1.1)$$

$$\text{AutorDe}(X,Y) \wedge \text{EscritoEm}(Y,Z) \Rightarrow \text{PodeEscreverEm}(X,Z) \quad (1.2)$$

Em (Shen & Leng, 1996), o sistema gera todas as possíveis especializações a partir de metaconsultas avalia a freqüência dessas especializações, elimina aquelas consideradas pouco freqüentes, ordena as especializações remanescentes em função de suas freqüências e as apresenta ao usuário para seleção das relações aprendidas. Convém ressaltar que as descobertas realizadas por esse sistema são restritas ao tipo de conhecimento especificado pelo usuário como metaconsulta. Ressalta-se ainda que este sistema não utiliza conhecimento específico do domínio da aplicação e não possui recursos para identificar descobertas que poderiam ser mais interessantes segundo a óptica do usuário.

**Tabela 6.13** Trabalhos em Assistência à Execução de Ações de KDD

<b>Características da Assistência em KDD</b>	<b>(Livingston, 2001)</b>	<b>(Shen &amp; Leng, 1996)</b>
Etapas do Processo de KDD	Mineração de Dados	Mineração de Dados Pós-processamento
Algoritmos e Técnicas de KDD	RL (Provost & Buchanan, 1995)	Aprendizado Relacional e Lógica de 2ª Ordem
Tarefas de KDD	Sumarização	Sumarização
Mecanismo de Assistência	Agenda e Justificativa	Metaconsultas
Acoplamento Assistência – Execução	Alto	Alto
Recursos de Paralelismo e Distribuição	Não	Não
Supporte a iterações no Processo de KDD	Sim	Não
Conhecimento do Domínio da Aplicação	Sim	Não
Conhecimento de KDD	Sim	Sim
Representação do Conhecimento	Regras de Produção	Regras de Produção
Tipo de Metaconhecimento	Teórico	Teórico
Independência Assistência – Aplicações	Sim	Sim
Incorporação de Conhecimentos	Sim	Não
Capacidade de Aprendizado	Não	Não
IHM na Definição de Objetivos	Não se Aplica	Não se Aplica
IHM no Planejamento de Ações	Não se Aplica	Não se Aplica
IHM na Execução de Ações	Inexistente	Possível

### Assistência ao Planejamento de Ações em Aplicações de KDD

Os estudos sobre assistência ao planejamento de ações em KDD dividem-se em dois grupos.

Os estudos do primeiro grupo são orientados à análise da aderência de algoritmos de Mineração de Dados a conjuntos de dados. Considerando a crescente diversidade de algoritmos de mineração de dados, a definição de quais desses algoritmos possuem melhor desempenho em determinados problemas tem sido uma questão de grande relevância e interesse na comunidade científica (Aha, 1992; Brodley, 1993; Michie et al., 1994; Gama & Brazdil, 1995; Wolpert, 1996; Wolpert & Macready, 2001; Brazdil, 1998; Spiliopoulou et al., 1998; Nakhaeizadeh & Schnabl, 1998; Kalousis & Theoharis, 1999; Kalousis & Hilario, 2000; Keller et al., 2000; Brazdil & Soares, 2000; Bensusan et al., 2000a; Bensusan et al., 2000b; Pfahringer & Bensusan, 2001; Soares et al., 2001; Brazdil et al., 2003).

No segundo grupo encontram-se os estudos que têm como objetivos identificar e selecionar as possíveis alternativas de pré-processamento de dados para viabilizar a utilização de determinados algoritmos de mineração de dados.

As Tabelas 6.14 e 6.15 apresentam o enquadramento dos trabalhos voltados à assistência ao planejamento de ações de KDD.

Em (Bernstein et al., 2002; Suyama & Yamaguchi, 1998; Goldschmidt et al., 2002d) foram utilizadas *ontologias* (Russell & Norvig, 1995) para descrever os conjuntos de algoritmos de KDD e suas características. Em todos os trabalhos, as ontologias utilizadas procuraram representar os algoritmos de KDD a partir das suas precondições e dos seus efeitos.

Em (Bernstein et al., 2002), os autores implementaram um protótipo, chamado *IDEA*, que, baseado em ontologias de KDD, produz todos os planos de KDD válidos. Um plano de KDD válido especifica uma seqüência de algoritmos cujas precondições e efeitos são compatíveis. *IDEA* permite que usuários ordenem o conjunto de planos produzidos a fim de facilitar a escolha de um ou mais desses planos. Apenas tarefas de classificação, summarização e regressão foram tratadas neste trabalho.

Em (Suyama & Yamaguchi, 1998), os autores desenvolveram um sistema que procura combinar funções por meio de recursos de *programação genética* (Koza, 1992; Michalewicz, 1994) para geração de classificadores. Restringe-se, portanto, à tarefa de classificação. No contexto do referido trabalho, uma solução é uma seqüência de algoritmos que, ao ser aplicada, gera um classificador. A avaliação de cada solução demanda, portanto, a geração de um classificador seguida de uma bateria de testes. Observa-se no trabalho mencionado um intenso custo computacional na avaliação das soluções propostas.

Em (Goldschmidt et al., 2002d), os autores propuseram uma extensão ao trabalho (Bernstein et al., 2002). Além da utilização de ontologias para descrição das características dos algoritmos, foi proposta a incorporação de heurísticas para filtragem dos algoritmos em função de restrições especificadas pelos usuários.

Em (Gama & Brazdil, 1995; Spiliopoulou et al., 1998; Brazdil, 1998; Kalousis & Theoharis, 1999; Keller et al., 2000; Brazdil & Soares, 2000; Bensusan et al., 2000a, b; Soares et al., 2001; Frunkranz & Petrak, 2001; Pfahringer & Bensusan, 2001; Brazdil et al., 2003) foram propostos critérios para ordenação de algoritmos de classificação, baseados no desempenho desses algoritmos em experimentos prévios. Dessa forma, ao receber uma nova base de dados, os mecanismos propostos buscam bases de dados similares e apresentam, de forma ordenada, os algoritmos que obtiveram os melhores resultados em tais bases. Tais abordagens também se encontram restritas à tarefa de classificação. Destaque especial neste grupo de trabalhos para o projeto *MetaL* (Gama & Brazdil, 1995; Brazdil & Soares, 2000; Brazdil et al., 2003), desenvolvido na Universidade do Porto, em Portugal.

Em (Bensusan et al., 2000 a, b; Pfahringer & Bensusan, 2001; Godoy, 2004), os autores utilizaram um processo denominado *landmarking*. Esse processo consiste em aplicar algoritmos de aprendizado mais simples e rápidos, denominados *landmarkers*, às bases de dados de forma a extrair as respectivas medidas de desempenho. Tais medidas são utilizadas para estimar os desempenhos de outros algoritmos de aprendizado mais complexos. A motivação para a teoria de *landmarking* decorre da expectativa de que os *landmarkers* possam auxiliar na identificação e no mapeamento das áreas em que os algoritmos mais complexos tenham desempenho favorável.

Em (Michie et al., 1994), os autores discutiram a aderência de diversos algoritmos de classificação, buscando caracterizar sob que circunstâncias determinados algoritmos devem ser utilizados em detrimento de outros. Vários algoritmos, de natureza estatística, conexionista e de aprendizado de máquina, foram exaustivamente experimentados em cerca de 20 bases de dados distintas. A referência (Michie et al., 1994) foi um dos subprodutos do projeto *StatLog* e embora não tenha originado nenhum novo mecanismo computacional para seleção de algoritmos, tem contribuído de forma significativa em diversos estudos mais modernos sobre classificação.

Em (Morik, 2000), a autora analisa a influência de ações de pré-processamento de dados no desempenho dos métodos de mineração de dados.

**Tabela 6.14 Trabalhos em Assistência ao Planejamento de Ações de KDD**

<b>Características da Assistência em KDD</b>	<b>(Bernstein et al., 2002)</b>	<b>(Suyama &amp; Yamaguchi, 1998)</b>	<b>(Brazdil et al., 2003)</b>
Etapas do Processo de KDD	Todas	Todas	Mineração de Dados
Algoritmos e Técnicas de KDD	Diversos	Diversos	Diversos
Tarefas de KDD	Classificação Regressão Sumarização	Classificação	Classificação

**Tabela 6.14** Continuação

<b>Características da Assistência em KDD</b>	<b>(Bernstein et al., 2002)</b>	<b>(Suyama &amp; Yamaguchi, 1998)</b>	<b>(Brazdil et al., 2003)</b>
Mecanismo de Assistência	Planejamento	Programação Genética	Critérios de Ordenação por Desempenho
Acoplamento Assistência – Execução	Baixo	Baixo	Baixo
Recursos de Paralelismo e Distribuição	Não	Não	Não
Supporte a iterações no Processo de KDD	Não	Não	Não
Conhecimento do Domínio da Aplicação	Não	Não	Sim
Conhecimento de KDD	Sim	Sim	Sim
Representação do Conhecimento	Ontologias	Ontologias	Histórico de Desempenho
Tipo de Metaconhecimento	Teórico	Experimental	Experimental
Independência Assistência – Aplicações	Sim	Sim	Sim
Incorporação de Conhecimentos	Sim	Sim	Sim
Capacidade de Aprendizado	Não	Não	Não
IHM na Definição de Objetivos	Não se Aplica	Não se Aplica	Não se Aplica
IHM no Planejamento de Ações	Possível	Possível	Possível
IHM na Execução de Ações	Não se Aplica	Não se Aplica	Não se Aplica

Em (Brodley, 1993), foram apresentados exemplos de metaconhecimento teórico, expressos na forma de regras de produção, para guiar a seleção de algoritmos de classificação. Neste trabalho, a inclusão de novos algoritmos requer a reavaliação não automática das regras previamente incorporadas, de forma a viabilizar a inclusão de novas regras consistentes com o conjunto de regras original.

**Tabela 6.15** Trabalhos em Assistência ao Planejamento das Ações de KDD

<b>Características da Assistência em KDD</b>	<b>(Michie et al., 1994)</b>	<b>(Brodley, 1993)</b>	<b>(Goldschmidt et al., 2002d)</b>
Etapas do Processo de KDD	Mineração de Dados	Mineração de Dados	Todas
Algoritmos e Técnicas de KDD	Diversos	Diversos	Diversos
Tarefas de KDD	Classificação Regressão	Classificação	Diversas
Mecanismo de Assistência	Seleção de Métodos baseada em experiências prévias	Seleção de Métodos baseada em Heurísticas	Seleção de Métodos baseada em Heurísticas e Ontologias

**Tabela 6.15** Continuação

<b>Características da Assistência em KDD</b>	<b>(Michie et al., 1994)</b>	<b>(Brodley, 1993)</b>	<b>(Goldschmidt et al., 2002d)</b>
Acoplamento Assistência / Execução	Baixo	Baixo	Baixo
Recursos de Paralelismo e Distribuição	Não	Não	Não
Suporte a iterações no Processo de KDD	Não	Não	Não
Conhecimento do Domínio da Aplicação	Sim	Sim	Sim
Conhecimento de KDD	Sim	Sim	Sim
Representação do Conhecimento	Diversas	Regras de Produção	Ontologias e Regras de Produção
Tipo de Metaconhecimento	Híbrido	Teórico	Híbrido
Independência Assistência / Aplicações	Sim	Sim	Sim
Incorporação de Conhecimentos	Sim	Sim	Sim
Capacidade de Aprendizado	Não	Não	Não
IHM na Definição de Objetivos	Não se Aplica	Não se Aplica	Não se Aplica
IHM no Planejamento de Ações	Possível	Possível	Possível
IHM na Execução de Ações	Não se Aplica	Não se Aplica	Não se Aplica

### Assistência à Definição de Objetivos em Aplicações de KDD

Nas referências (Goldschmidt et al., 2003, 2002b, 2002c), os autores propuseram um modelo computacional de auxílio à prospecção de objetivos baseada em conceitos da *Teoria da Equivalência entre Atributos de Bancos de Dados* (Larson et al., 1989) e de *Espaços Topológicos* (Lipschutz, 1979). Essa abordagem procura representar bases de dados como a união de padrões definidos a partir dos metadados dos atributos destas bases. Uma vez que uma base de dados tenha sido mapeada na nova representação, o mecanismo de assistência instancia operações, sugerindo alternativas de tarefas de KDD potencialmente executáveis nesta base.

**Tabela 6.16** Assistência à Definição de Objetivos de KDD

<b>Características da Assistência em KDD</b>	<b>(Goldschmidt et al., 2003)</b>
Etapas do Processo de KDD	Nenhuma
Algoritmos e Técnicas de KDD	Nenhum
Tarefas de KDD	Diversas
Mecanismo de Assistência	Lembrança Semântica entre Atributos, Espaços Topológicos

**Tabela 6.16** Continuação

<b>Características da Assistência em KDD</b>	<b>(Goldschmidt et al., 2003)</b>
Acoplamento Assistência / Execução	Baixo
Recursos de Paralelismo e Distribuição	Não
Suporte a iterações no Processo de KDD	Não
Conhecimento do Domínio da Aplicação	Sim
Conhecimento de KDD	Sim
Representação do Conhecimento	Regras de Produção
Metaconhecimento	Híbrido
Independência Assistência / Aplicações	Sim
Incorporação de Conhecimentos	Sim
Capacidade de Aprendizado	Não
IHM na Definição de Objetivos	Possível
IHM no Planejamento de Ações	Não se Aplica
IHM na Execução de Ações	Não se Aplica

### Assistência à Definição de Objetivos e ao Planejamento de Ações em Aplicações de KDD

A Tabela 6.17 apresenta o enquadramento de trabalhos sobre assistência à definição de objetivos e assistência ao planejamento das ações necessárias à realização destes objetivos em aplicações de KDD.

Na referência (Jensen et al., 1999), encontra-se proposta uma linguagem de programação e um ambiente interativo para programação nesta linguagem. A idéia é que o especialista em KDD utilize o ambiente e a linguagem para programar os passos a serem realizados em cada aplicação. No programa, a responsabilidade pela execução de cada passo é atribuída a um agente (Maes, 1994; Hayes-Roth, 1995; Wooldridge & Jennings, 1995), que pode ser computacional ou humano.

Em (Engels et al., 1997; Verdenius & Engels, 1997), os autores utilizam um modelo de processos que permite a decomposição hierárquica de tarefas a partir de objetivos definidos pelo usuário humano em uma etapa não automatizada.

Encontra-se descrito em (Kerber et al., 1998) o conceito de *áreas de trabalho ativas*. Uma área de trabalho ativa é uma estrutura de dados utilizada para armazenar sequências de métodos de KDD que tenham sido bem-sucedidas. Neste mesmo trabalho, os autores utilizam hipertextos e recursos gráficos para apresentação de informações sobre as áreas de trabalho ativas, procurando orientar a realização de novos processos de KDD.

**Tabela 6.17 Assistência à Definição de Objetivos e ao Planejamento das Ações de KDD**

<b>Características da Assistência em KDD</b>	<b>(Engels et al., 1997)</b>	<b>(Jensen et al., 1999)</b>	<b>(Kerber et al., 1998)</b>
Etapas do Processo de KDD	Todas	Todas	Todas
Algoritmos e Técnicas de KDD	Diversos	Diversos	Diversos
Tarefas de Mineração de Dados	Diversas	Diversas	Diversas
Mecanismo de Assistência	Decomposição Hierárquica de Tarefas	Agenda de Tarefas	Áreas de Trabalho Ativas
Acoplamento Assistência/Execução	Baixo	Baixo	Baixo
Recursos de Paralelismo e Distribuição	Não	Sim	Não
Supporte a iterações no Processo de KDD	Não	Sim	Sim
Conhecimento do Domínio da Aplicação	Não	Não	Sim
Conhecimento de KDD	Sim	Sim	Sim
Representação do Conhecimento	Codificação em métodos	Associações entre Tarefas e Agentes	Hipertextos, Áreas de Trabalho Ativas
Tipo de Metaconhecimento	Teórico	Teórico	Teórico
Independência Assistência / Aplicações	Sim	Sim	Sim
Incorporação de Conhecimentos	Sim	Sim	Sim
Capacidade de Aprendizado	Não	Não	Não
IHM na Definição de Objetivos	Necessária	Necessária	Necessária
IHM no Planejamento de Ações	Possível	Necessária	Necessária
IHM na Execução de Ações	Inexistente	Inexistente	Inexistente

## Assistência ao Planejamento e à Execução de Ações em Aplicações de KDD

No trabalho descrito em (Amant & Cohen, 1997a, b), os autores utilizaram um conceito de *planejamento hierárquico com iniciativa de colaboração mista* aplicada à *Análise Exploratória de Dados*. O planejamento hierárquico refere-se à decomposição sucessiva de tarefas em tarefas menores. A expressão *iniciativa de colaboração mista* representa a possibilidade de realização de tarefas pelo próprio sistema ou pelo usuário humano. Em função do tipo de tarefa, das heurísticas utilizadas e do conhecimento envolvido, tanto o homem quanto o próprio sistema, ou mesmo ambos podem se candidatar à execução de cada nova tarefa.

Convém ressaltar que, em (Amant & Cohen, 1997a, b), a solução proposta restringe-se à tarefa de Sumarização ou Descrição da Base de Dados (Fayyad et al., 1996a; Weiss & Indurkha, 1998). O enquadramento do referido trabalho encontra-se na Tabela 6.18.

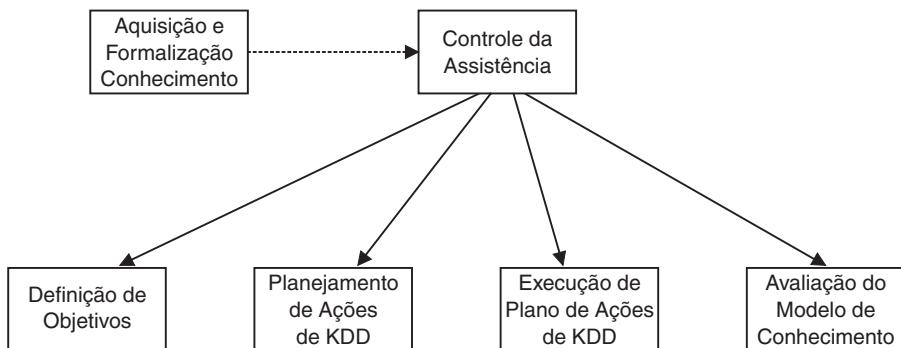
**Tabela 6.18 Assistência ao Planejamento e à Execução de Ações em KDD**

<b>Características da Assistência em KDD</b>	<b>(Amant &amp; Cohen, 1997a, b)</b>
Etapas do Processo de KDD	Todas
Algoritmos e Técnicas de KDD	Diversos
Tarefas de KDD	Sumarização
Mecanismo de Assistência	Decomposição Hierárquica de Tarefas e Análise Exploratória de Dados
Acoplamento Assistência / Execução	Baixo
Recursos de Paralelismo e Distribuição	Não
Supporte a iterações no Processo de KDD	Sim
Conhecimento do Domínio da Aplicação	Não
Conhecimento de KDD	Sim
Representação do Conhecimento	Codificado em Métodos de Análise Exploratória de Dados
Metaconhecimento	Teórico
Independência Assistência / Aplicações	Sim
Incorporação de Conhecimentos	Sim
Capacidade de Aprendizado	Não
IHM na Definição de Objetivos	Não se Aplica
IHM no Planejamento de Ações	Possível
IHM na Execução de Ações	Possível

### Assistência à Definição de Objetivos, ao Planejamento e à Execução de Ações em Aplicações de KDD – Máquina de IKDD

A Máquina de IKDD, proposta em (Goldschmidt, 2003), foi concebida de forma a ser utilizada, fundamentalmente, como ferramenta didática voltada à formação de profissionais técnicos na área da Descoberta de Conhecimento em Bases de Dados. A Figura 6.1 apresenta uma visão macrofuncional de sua estrutura. Cada macrofunção representa um tipo de assistência. A seguir encontram-se comentários sobre cada uma delas:

- Assistência à Aquisição e Formalização do Conhecimento – Essa função tem como objetivo viabilizar o processo de aquisição e formalização do conhecimento sobre KDD que será utilizado como *recurso* pela Máquina de IKDD. Por meio desta assistência, um especialista em KDD pode configurar, de forma personalizada, o conhecimento sobre KDD a ser processado pela máquina. Diferentes versões da Máquina de IKDD podem ser



**Figura 6.1.** Visão Macro Funcional da Máquina de IKDD.

configuradas e mantidas por diferentes especialistas em KDD com diferentes níveis de experiência. A configuração do conhecimento sobre KDD constitui-se em um requisito operacional ao funcionamento da Máquina de IKDD.

- Controle da Assistência – Uma vez configurados os recursos sobre KDD, a Máquina de IKDD pode oferecer assistência em aplicações de KDD. A criação de uma aplicação de KDD demanda a caracterização do problema associado. A função de Controle da Assistência auxilia neste processo. São informados o conjunto de dados e o conhecimento dependente do domínio da aplicação. Os objetivos da aplicação, caso sejam previamente conhecidos, também podem ser informados. Após a caracterização do problema de que trata a aplicação de KDD, o Controle da Assistência analisa o contexto, gera o *plano de IKDD* (ações de assistência) a ser utilizado na assistência ao processo de KDD e aciona as demais funções de assistência, na ordem especificada pelo *plano de IKDD*.
- Assistência à Definição de Objetivos – Essa assistência tem como função auxiliar os usuários da Máquina de IKDD a identificar tarefas de KDD viáveis em cada aplicação e a caracterizar expectativas quanto aos possíveis resultados. As expectativas quanto ao *modelo de conhecimento* a ser obtido em processos de KDD são representadas na forma de predicados e, juntas, compõem uma lista de restrições a serem satisfeitas pelo *modelo de conhecimento*. As abordagens propostas na formulação desta assistência baseiam-se em conceitos da *Teoria da Equivalência entre Atributo de Bancos de Dados* (Larson et al., 1989), em conceitos de *Espaços Topológicos* (Lipschutz, 1979) e em recursos de Redes Neurais Artificiais (Haykin, 1999) para viabilizar mapeamentos entre conjuntos de dados e operações de Mineração de Dados.
- Assistência ao Planejamento de Ações de KDD – Essa função visa a auxiliar o usuário na formulação e na escolha dos *planos de ações de KDD* a se-

rem utilizados no processo. As expectativas acerca do *modelo de conhecimento* a ser produzido podem ser utilizadas na simplificação do espaço de planos válidos. Os *predicados* que definem tais expectativas podem ser utilizados para filtrar métodos cujos efeitos satisfaçam a esses *predicados*. Esta função procura utilizar conhecimento prévio sobre o desempenho dos métodos filtrados para ordená-los segundo determinados critérios (Brazdil et al., 2003). Emprega, para tanto, medidas proporcionadas pela *Teoria Estatística* (Engels & Theusinger, 1998) e pela *Teoria da Informação* (Michie et al., 1994). Em seguida, a partir da lista ordenada de métodos, *planos de ações de KDD* são gerados. Esta função dispõe ainda de alguns critérios de ordenação que auxiliam o usuário na escolha entre os *planos de ações de KDD* gerados.

- Assistência à Execução do Plano de Ações de KDD – Essa assistência tem como objetivo auxiliar o usuário da Máquina de IKDD a utilizar os métodos especificados no *plano de ações de KDD*. O conteúdo e a função dessa assistência variam segundo as especificidades de cada método de KDD. Essa função de assistência pode oferecer mecanismos para avaliação dos resultados produzidos pelos métodos de KDD.
- Assistência à Avaliação do Modelo de Conhecimento – Trata-se de um caso especial de assistência para avaliação de resultados em que o resultado a ser avaliado é o *modelo de conhecimento*, objeto de busca do processo de KDD. Essa assistência também varia em função do método de KDD. Verifica se as expectativas inicialmente formuladas nos objetivos foram cumpridas com o *modelo de conhecimento* produzido pela aplicação de KDD. O não-cumprimento de determinadas expectativas pode ser utilizado como subsídio para o Controle da Assistência decidir acerca das estratégias a serem adotadas nos passos subsequentes do processo de KDD.

A Tabela 6.19 resume as principais características da Máquina de IKDD segundo o modelo descritor de mecanismos de assistência ao processo de KDD, facilitando a comparação da máquina com os principais trabalhos da área, mencionados anteriormente.

**Tabela 6.19 Resumo das Características da Máquina de IKDD**

Características da Assistência em KDD	Máquina de IKDD (Goldschmidt, 2003)
Etapas do Processo de KDD	Todas
Algoritmos e Técnicas de KDD	Diversos
Tarefas de KDD	Diversas
Dimensões da Assistência em KDD	Definição de Objetivos Planejamento das Ações de KDD Execução de Ações de KDD

**Tabela 6.19** Continuação

<b>Características da Assistência em KDD</b>	<b>Máquina de IKDD (Goldschmidt, 2003)</b>
Mecanismo de Assistência	<i>Teoria do Planejamento</i> <i>Teoria da Equivalência entre Atributos</i> <i>Espaços Topológicos</i> <i>Inteligência Computacional</i>
Acoplamento Assistência – Execução	Baixo
Recursos de Paralelismo e Distribuição	Sim
Suporte a iterações no Processo de KDD	Não
Conhecimento do Domínio da Aplicação	Sim
Conhecimento de KDD	Sim
Representação do Conhecimento	Diversas
Metaconhecimento	Teórico, Experimental e Híbrido
Independência Assistência – Aplicações	Sim
Incorporação de Novos Conhecimentos	Sim
Capacidade de Aprendizado	Sim
IHM na Definição de Objetivos	Necessária
IHM no Planejamento de Ações	Necessária
IHM na Execução de Ações	Necessária

# Uma Metodologia para KDD

## Considerações Iniciais

Conforme mencionado no capítulo anterior, a complexidade do processo de KDD deve-se, basicamente, à dificuldade de percepção, interpretação e conjugação adequada de inúmeros fatos que surgem durante o processo, aliada à diversidade de alternativas de métodos e parametrizações possíveis de experimentação.

A área de KDD carece de metodologias de trabalho que direcionem e optimizem esforços. Muitos autores se referem à área como sendo uma reunião de arte e conhecimento técnico. São recorrentes as questões sobre como executar um processo para a descoberta de algo que não se sabe exatamente o que seja.

Na formulação de metodologias de realização de KDD, o objetivo não é desenvolver ferramentas computacionais, mas preestabelecer conjuntos ordenados de regras e tarefas a serem seguidas pelo homem a fim de realizar processos de KDD e produzir resultados satisfatórios (Brachman & Anand, 1996; Wirth et al., 1997; Kerber, et al., 1998).

Assim sendo, este capítulo procura fornecer algumas orientações práticas sobre como conduzir processos de KDD. Para tanto, são indicados alguns modelos de documentação que, apoiados em uma linha básica de raciocínio, subsidiem a escolha de procedimentos a serem adotados diante da diversidade de situações e possibilidades.

A metodologia proposta neste capítulo retrata um pouco da experiência dos autores em diversas aplicações práticas reais. É importante destacar, no entanto, que tal metodologia não tem a pretensão de ser uma solução que esgote o universo ilimitado de possibilidades de alternativas de ação em processos de KDD. Apenas propõe uma forma de investigação diante deste universo. Cabe ressaltar ainda que esta metodologia pode e deve ser ajustada conforme as necessidades do leitor, na medida em que este adquira uma experiência cada vez maior na área.

Conforme ilustrado no capítulo anterior, ferramentas computacionais podem ser utilizadas para auxiliar no processo de KDD. A Máquina de IKDD apresentada no Capítulo 6, na seção “Assistência à Definição de Objetivos, ao Planejamen-

to e à Execução de Ações de KDD – Máquina de IKDD” é um exemplo de ferramenta computacional que fornece suporte à metodologia descrita neste capítulo.

## Visão Geral

A metodologia proposta neste capítulo será apresentada por meio de um conjunto de etapas que são detalhadas nas seções seguintes.

Considerando a complexidade normalmente inerente a processos de descoberta de conhecimento em bases de dados, a metodologia proposta utiliza como base princípios de planejamento de atividades. Assim, em função dos objetivos de cada aplicação de KDD, os passos do processo de descoberta de conhecimento são planejados antes do início de sua execução. A aplicação da metodologia de KDD proposta divide-se basicamente em quatro momentos:

O primeiro momento envolve a definição sobre “o que fazer” diante da base de dados apresentada. Neste momento, devem ser executadas as etapas de “*Levantamento Inicial*” e de “*Definição de Objetivos*”.

O Levantamento Inicial compreende um exame preliminar da base de dados, procurando obter informações sobre a natureza e o propósito dos dados a serem analisados.

Na definição de objetivos, devem ser identificadas quais tarefas de Mineração de Dados são viáveis, procurando adequá-las de forma a atender às necessidades e expectativas do usuário do domínio da aplicação. Neste momento, devem ser formulados alguns requisitos quanto ao modelo de conhecimento a ser produzido. Mais de um objetivo pode ser constituído.

A partir da escolha de um objetivo, a abordagem proposta é direcionada para a definição sobre “como fazer”. Corresponde à etapa de “*Planejamento de Atividades*”. Nesta etapa devem ser definidas as alternativas de planos de ação associados ao objetivo escolhido. Um plano de ação é uma seqüência válida de métodos de KDD. Essa metodologia sugere que os planos de ação sejam constituídos a partir de cada método de mineração de dados aplicável à tarefa de KDD associada ao objetivo selecionado.

Para cada plano de ação selecionado, iniciam-se os trabalhos de execução ordenada das ações previstas no plano. A etapa de “*Execução dos Planos de Ação*” corresponde à aplicação propriamente dita dos métodos de KDD. Neste momento podem e devem ser experimentados de forma coerente diversos valores nos parâmetros dos algoritmos envolvidos.

Finalmente, a abordagem proposta é concluída pela etapa de “*Avaliação de Resultados*”. Essa etapa corresponde à “análise do que foi feito”. Neste momento, as características do modelo de conhecimento gerado devem ser confrontadas com as expectativas quanto ao modelo formuladas na etapa de “*Definição de Objetivos*”.

A metodologia proposta sugere um processo iterativo e interativo no qual, dependendo dos resultados obtidos, os analistas de KDD podem retornar a

qualquer etapa realizada anteriormente em busca de melhores resultados. Para tanto, a metodologia requer uma documentação detalhada das ações realizadas e dos resultados produzidos.

Considerando o aspecto interativo da metodologia proposta são utilizados dois conceitos de controle em sua aplicação: sessão de KDD e ciclo de KDD.

Cada sessão de KDD representa uma linha de raciocínio e de condução do processo. Cada um dos objetivos estabelecidos no início do processo possui uma sessão de KDD associada. Toda sessão de KDD possui um, e somente um, objetivo a ser alcançado.

Cada sessão de KDD pode ser realizada em um ou mais ciclos. Cada ciclo de KDD envolve as etapas de execução de um plano de ação e pertence a exatamente uma sessão de KDD.

A Figura 7.1 apresenta uma sugestão de formulário para documentação das ações e resultados obtidos. As seções seguintes ilustram o preenchimento do formulário proposto.

<b>Aplicação:</b>		
Sessão: Objetivo: Tarefa de KDD:	Resumo:	
Expectativas quanto ao Modelo de Conhecimento:		
<b>Plano de Ação:</b>		
Ciclo nº:		
Métodos	Obs. sobre Parâmetros	Resultados

**Figura 7.1.** Formulário para Documentação de Ações e Resultados do Processo de KDD.

## Levantamento Inicial

Existem algumas considerações técnicas que devem ser verificadas no início de um processo de KDD:

- Identificar as pessoas e áreas envolvidas no processo de KDD. Os especialistas no domínio da aplicação, a equipe de tecnologia da informação e os grupos de decisão que deverão aplicar os resultados devem ser submetidos, sempre que necessário, a um treinamento em KDD que nivela o conhecimento técnico na área.
- Fazer um levantamento do hardware e software existentes. Recomenda-se que o processo de KDD seja realizado em plataforma com arquitetura expansível, que suporte grande volumes de dados, boa capacidade de processamento e acesso a bases de dados heterogêneas.
- Fazer um inventário das bases de dados disponíveis. Incluem-se neste item tanto bases de dados internas quanto externas. Potenciais bases externas são aquelas relacionadas ao domínio da aplicação e que possam ser utilizadas no enriquecimento dos dados.
- Verificar a existência de Data Warehouses. A utilização de Data Warehouses preexistentes pode poupar muitos esforços de pré-processamento dos dados.
- Compreender o significado e perceber a relevância dos atributos disponíveis. Essa análise tem papel significativo na etapa seguinte de “Definição de Objetivos”. Metadados acerca das bases de dados e seus atributos devem ser documentados.
- Esboçar uma lista de necessidades e expectativas dos usuários quanto ao propósito do KDD. Procurar identificar que critérios podem ser adotados para mensurar o sucesso do processo. Nesse momento, o especialista em KDD começa a refletir sobre os tipos de decisão que são necessárias aos usuários e que tipo de modelo de conhecimento pode satisfazer às expectativas.
- Avaliar a qualidade dos dados disponíveis. Deve-se procurar identificar o propósito para o qual os dados foram coletados, assim como a caracterização do nível de ruído envolvido. Bases de dados poluídas requerem o uso de ferramentas adequadas ao processo de limpeza. Em geral, os recursos de limpeza são fortemente dependentes do domínio da aplicação e podem ser otimizados com a utilização de conhecimentos preexistentes.
- Verificar se os dados estão disponíveis em quantidade suficiente para o processo de KDD. Bases de dados pequenas ou que contenham dados pouco representativos das condições normais do domínio da aplicação podem inviabilizar o processo de KDD.

- Identificar e documentar todo o conhecimento prévio existente e disponível acerca do domínio da aplicação. Esse conhecimento pode ser útil no decorrer do processo de KDD.

Embora nem sempre possível, a participação de especialistas em KDD no projeto de bases de dados que venham a ser utilizadas em processos de KDD é extremamente recomendável. Muitas informações podem ser captadas e armazenadas de forma a viabilizar com vantagens a realização de processo de KDD. Na prática, no entanto, o especialista em KDD recebe uma base de dados já populada com dados da aplicação.

Nesta etapa, o formulário da Figura 7.1 deve ser utilizado para registro do título da aplicação, assim como de um resumo descritivo do projeto.

## Definição de Objetivos

A definição dos objetivos em qualquer aplicação de KDD requer, primeiramente, um entendimento claro da situação vigente no ambiente onde será realizado o processo. Esse entendimento começa a ser formado desde a etapa de levantamento inicial de informações.

Uma análise da natureza dos dados pode fornecer alguns indicadores de possíveis tarefas de Mineração de Dados, que devem ser consideradas na formulação dos objetivos do processo de KDD. Mais detalhes podem ser obtidos em (Goldschmidt, 2003). Um detalhamento deste tipo de análise ficaria fora dos propósitos deste livro.

A etapa de “Definição de Objetivos” requer uma forte interação entre o analista de KDD e os especialistas do domínio da aplicação. Diversas entrevistas com esses especialistas devem ser realizadas.

O analista de KDD deve ser capaz de identificar as expectativas existentes no domínio da aplicação e associá-las a tarefas de Mineração de Dados que, uma vez executadas, possam viabilizar a construção de modelos de conhecimentos com potencial para atender tais expectativas.

O analista de KDD deve, primeiramente, listar todas as expectativas identificadas. Em seguida, deve validá-las junto aos especialistas do domínio da aplicação. Uma vez validadas, o analista de KDD deve procurar agrupá-las em função de sua natureza e de forma que expectativas em um mesmo grupo possam ser atendidas por um mesmo modelo de conhecimento.

Com as expectativas identificadas e agrupadas, o analista de KDD deve analisar que tipo de tarefa de Mineração de Dados, primária ou composta, deve ser aplicada de forma a procurar obter modelos de conhecimento que atendam a cada grupo específico de expectativas. Por exemplo, uma expectativa de previsão de comportamento de cliente, muito comum em marketing, pode ser mapeada em tarefas tais como classificação de cliente ou regras de associação, ou ainda em mineração de seqüências. A identificação do tipo de ta-

referência de Mineração de Dados é fundamental nesta metodologia, pois constitui o núcleo do processo de KDD.

Cada grupo de expectativas pode ser interpretado como um objetivo do processo de KDD conforme ilustra a Figura 7.2. Para cada novo objetivo, devem ser complementadas as informações de seção, objetivo e tarefa de KDD. Vide exemplo de preenchimento na Figura 7.2.

Aplicação: Marketing Direto ao Consumidor	
Sessão: 1	Resumo: Esta seção tem como objetivo extrair
Objetivo: Previsão de comportamento	modelos que, baseados no comportamento de
Tarefa de KDD: Descoberta de Seqüências	compra dos clientes, consigam prever potenciais compras futuras destes clientes
Expectativas quanto ao Modelo de Conhecimento:	
<ul style="list-style-type: none"> <li>• Transparência do modelo</li> <li>• Representação em regras de produção</li> </ul>	

**Figura 7.2.** Exemplo de preenchimento do formulário na etapa “Definição de Objetivos”.

Convém ressaltar que o identificador de sessão é um seqüencial que deve ser incrementado a cada novo objetivo. Um novo formulário deve ser preenchido para cada nova sessão criada.

Uma sessão de KDD deve estar associada a uma única tarefa de KDD. Uma tarefa de KDD é uma especificação lógica de um conjunto de métodos que podem ser utilizados com a mesma finalidade. Um método, no escopo desta metodologia, refere-se a uma implementação específica de uma tarefa de KDD.

Neste momento, o analista de KDD deve procurar especificar as principais expectativas quanto ao modelo de conhecimento que poderá ser produzido. Transparência do modelo e representação na forma de regras de produção são exemplos de expectativas quanto ao modelo de conhecimento esperado.

## Planejamento de Atividades

Para cada sessão de KDD identificada na etapa de “Definição de Objetivos”, o analista de KDD deve elaborar um planejamento de atividades que possibilite o cumprimento do objetivo correspondente.

Nesse momento, o analista de KDD deve identificar dentre os métodos disponíveis, aqueles que implementam a tarefa desejada. Estes são denominados métodos candidatos.

A relação de métodos candidatos deve ser restrita aos métodos que sejam compatíveis com as expectativas formuladas para o modelo de conhecimento

desejado. Por exemplo, o método Back-Propagation deve ser excluído da relação de método candidatos à classificação sempre que, entre as expectativas quanto ao modelo de conhecimento, conste um item que indique como requisito a transparência de modelo de conhecimento gerado.

A filtragem de métodos candidatos requer conhecimento sobre as características e os efeitos de cada método.

Em geral, a escolha dos métodos a serem aplicados depende da preferência pessoal do analista de KDD. Recomenda-se, no entanto, que todos os métodos candidatos não eliminados na filtragem sejam considerados.

Para seleção do método ou dos métodos a serem efetivamente aplicados, recomenda-se que o analista de KDD elabore uma lista ordenada que procure expressar o potencial dos métodos em produzir bons resultados. Para tanto, o analista de KDD deve utilizar-se de sua experiência prévia com tais algoritmos em base de dados similares à base de dados a ser processada. Em (Goldschmidt, 2004) encontra-se descrito um procedimento automatizado para ordenação de métodos candidatos.

Para cada método candidato, o analista de KDD deve formular alternativas de pré-processamento a serem realizadas de forma a adequar a base de dados aos requisitos do método selecionado.

Neste momento, o analista de KDD deve planejar quais métodos de pré-processamento devem ser utilizados, incluindo a ordem de aplicação. Por exemplo, suponha que a base de dados possua dados não normalizados e que o método candidato tenha como precondição que todos os dados estejam normalizados. Suponha ainda que a base de dados contenha dados categóricos. O algoritmo K-NN requer que a base de dados possua apenas atributos numéricos e normalizados. A Figura 7.3 mostra algumas alternativas de pré-processamento, também denominadas planos de ação.

#### Codificação Categórica-Numérica

Codificação Categórica-Numérica → Normalização linear → K-NN

Codificação Categórica-Numérica → Normalização pelo máximo → K-NN

Codificação Categórica-Numérica → Normalização por score → K-NN

**Figura 7.3.** Alternativas de Planos de Ação.

Cabe ressaltar que a geração de planos de ação requer que o analista de KDD possua conhecimento sobre as precondições (requisitos) e os efeitos dos métodos disponíveis. Cada plano de ação definido pelo analista de KDD deve ser documentado, conforme ilustra a Figura 7.4.

Aplicação: Marketing Direto ao Consumidor	
Sessão: 1	Resumo: Esta seção tem como objetivo extrair modelos que, baseados no comportamento de compra dos clientes, consigam prever potenciais compras futuras destes clientes.
Objetivo: Previsão de comportamento	
Tarefa de KDD: Descoberta de Seqüências	
Expectativas quanto ao Modelo de Conhecimento:	
<ul style="list-style-type: none"> <li>• Transparência do modelo</li> <li>• Representação em regras de produção</li> </ul>	
Plano de Ação:	
Preenchimento de valores ausentes → codificação numérica–categórica → partição BD em treino e teste → C4. 5	

**Figura 7.4.** Exemplo de preenchimento do formulário na etapa de “Planejamento de Atividades”.

## Execução dos Planos de Ação

Cada plano de ação definido na etapa anterior deve ser experimentado e ter seus resultados avaliados. Recomenda-se que o analista de KDD responsável pela coordenação do processo distribua os diversos planos de ação pelo pessoal técnico especializado. Dessa forma, os planos podem ser executados simultaneamente. Recomenda-se também que o coordenador centralize a análise dos resultados obtidos pelos diversos planos. Com uma visão global do processo, o coordenador pode perceber detalhes que permitam reavaliar e até redirecionar a estratégia adotada.

A execução de um plano de ação compreende a execução ordenada dos métodos que compõem o plano.

A execução de um plano de ação deve ser em ciclos. A cada ciclo, o analista de KDD deve executar total ou parcialmente o plano de ação, procurando obter melhores resultados. Em geral, esta etapa ocorre de forma integrada à etapa de “Avaliação de Resultados”.

Uma dificuldade freqüente em um processo de KDD é a escolha da parametrização adequada a um algoritmo diante de cada nova situação (Adegboye et al., 2002).

Esse problema pode facilmente aumentar o número de iterações do processo de KDD, na medida em que diversos algoritmos com diferentes parametrizações podem conduzir a diferentes situações na busca de resultados interessantes.

Recomenda-se fortemente que o analista de KDD responsável pela execução de um plano de ação registre as parametrizações adotadas e os resultados obtidos a cada ciclo. A Figura 7.5 ilustra este procedimento.

Aplicação: Marketing Direto ao Consumidor						
Sessão: 1	Resumo: Esta seção tem como objetivo extrair modelos que, baseados no comportamento de compra dos clientes, consigam prever potenciais compras futuras destes clientes.					
Objetivo: Previsão de comportamento						
Tarefa de KDD: Descoberta de Seqüências						
Expectativas quanto ao Modelo de Conhecimento:						
<ul style="list-style-type: none"> <li>• Transparência do modelo</li> <li>• Representação em regras de produção</li> </ul>						
Plano de Ação:						
Preenchimento de valores ausentes → codificação numérica–categórica → partição BD em treino e teste → C4. 5						
Ciclo nº 1						
Métodos	Obs sobre Parâmetros	Resultados				
Preenchimento valores ausentes	Utilizou-se a média das classes para atributos numéricos e a moda das classes para atributos categóricos.	300 registros tiveram valores preenchidos.				
Codificação numérica – categórica. (Discretização)	10 intervalos de igual comprimento em todos os atributos.	Atributos A B C	Comprimento: 10 unidades 1000 unidades 0,1 unidade.			
Partição do BD em treino e teste	80% treino 20%Teste	OK				
C 4.5	Sem Parametrização		Acurácia			
		BD treino	95%			
		BD teste	85%			

**Figura 7.5.** Exemplo de preenchimento do formulário na etapa de “Execução de Planos de Ação”.

Considerando as especificidades de cada método de KDD, foge do escopo deste livro fornecer orientações sobre elas, além das apresentadas no Capítulo 5.

## Avaliação de Resultados

A rigor, esta etapa deve ser realizada ao final do processamento de cada método do plano de KDD. Em muitas situações de pré-processamento não é possível avaliar o resultado produzido de forma isolada.

Em geral a avaliação de resultados pode ser realizada de forma mais efetiva após a execução dos métodos de Mineração de Dados. Neste momento, as características do modelo de conhecimento gerado devem ser confrontadas com as expectativas quanto ao modelo de conhecimento formulado na etapa de “De-

finição de Objetivos”. Algumas medidas podem ser comparadas diretamente. Este é o caso, por exemplo, da acurácia de modelos de previsão em relação à acurácia mínima aceitável. Outras características são, em essência, mais subjetivas. Este é o caso, também a título de exemplo, da utilidade de uma regra de associação. O quanto novo e útil é o conhecimento extraído? Pode-se então facilmente perceber a importância do especialista no domínio da aplicação no momento da avaliação dos resultados para responder a questões desta natureza.

Alguns resultados, embora apurados após a execução dos métodos de Mineração de Dados, podem ser utilizados para definir ajustes na parametrização dos próprios métodos de mineração ou mesmo em métodos de preprocessamento. No exemplo da Figura 7.5, o plano de KDD poderia ser reexecutado variando-se o número de intervalos utilizado para cada atributo. Novos resultados são obtidos e devem ser reavaliados.

Idealmente, cada execução de um plano de KDD deve ser documentada de forma a proporcionar ao analista de KDD um mínimo de controle sobre o processo.

## Considerações Complementares

O leitor pode perceber pelo exposto nas seções anteriores a dificuldade em manter a documentação da metodologia proposta. A aplicação dessa metodologia requer extrema organização e um bom trabalho em equipe, sendo altamente desejável a utilização de ferramentas computacionais que facilitem tal controle. A máquina de KDD, descrita no Capítulo 6, provê mecanismos de controle compatíveis com a metodologia proposta.

São infinitas as possibilidades de ação em problemas de KDD. Diante deste cenário, este capítulo teve como principal objetivo descrever uma forma de trabalho que procure organizar esforços e resultados em um processo tão vasto quanto à Descoberta de Conhecimento em Base de Dados.

Vale a pena reforçar que a metodologia descrita não esgota todas as alternativas de ação em KDD e deve ser adaptada pelo leitor conforme sua experiência e necessidades específicas na área.

# Exemplos de Aplicações

## Considerações Iniciais

Conforme comentado nos capítulos iniciais, é espantoso o crescimento da demanda por aplicações em KDD nas mais diversas áreas do conhecimento humano (Carvalho, 2001). Potencialmente, técnicas de KDD podem ser utilizadas em qualquer domínio de aplicação que contenha razoáveis volumes de dados históricos sobre algum assunto.

Assim, este capítulo tem como objetivo descrever resumidamente algumas aplicações de KDD para ilustrar o potencial desta tecnologia. Não será difícil para o leitor perceber uma formidável diversidade de outras aplicações.

Todas as aplicações descritas da seção “Telefonia” até a seção “Área Financeira” são alguns dos principais exemplos de aplicações reais de KDD vivenciadas por nós, autores. Por motivo de sigilo junto aos clientes, maiores detalhes sobre as aplicações não puderam ser divulgados. A seção “Outros Exemplos” apresenta mais alguns exemplos com boa aderência à utilização da tecnologia de KDD.

## Telefonia

Este projeto teve como objetivo realizar a classificação de clientes de uma grande empresa do ramo de telecomunicações de acordo com seu potencial de compra de serviços. Para tanto, as seguintes etapas foram realizadas: a seleção de uma amostra do banco de dados de clientes. Os clientes selecionados preencheram questionários fornecidos pela empresa. Com tais informações, o banco de dados foi enriquecido. Foi realizado um processo de clusterização sobre a referida amostra de clientes. Após o processo de clusterização, classes de clientes foram definidas. A partir desta definição, um classificador foi gerado ainda considerando tal amostra de clientes. O classificador gerado foi aplicado à base de dados completa, caracterizando o potencial de compras de todos os clientes. Uma vez caracterizado o potencial de compra dos clientes, ações de marketing específicas por cliente puderam ser realizadas.

## Franquia de *Fast-food*

Outro exemplo de aplicação utilizou uma base de dados que continha informações das transações de venda de itens realizadas durante um determinado período de captação de dados a partir do ponto de venda de uma loja do ramo de “fast-food”. Esse período de captação, embora tenha sido localizado, foi definido de forma a refletir o comportamento comercial da loja em dias normais de venda. A base de dados foi submetida à etapa de pré-processamento, sendo em seguida aplicada à tarefa de Descoberta de Regras de Associação. Algumas associações entre produtos foram consideradas interessantes e promoções para estimular a venda combinada destes produtos foram realizadas com êxito (Godoy et al., 2003).

## Ação Social

Este projeto foi desenvolvido junto ao PRODERJ – Órgão de Tecnologia da Informação do Estado do Rio de Janeiro – e teve como objetivo auxiliar no processo de reintegração das pessoas de rua do estado. O governo do estado acolhe pessoas de rua em diversos centros de reintegração. Essas pessoas são submetidas a programas de reintegração social. Em função do perfil de cada pessoa, pode ser determinado o programa mais adequado à sua reintegração. Dessa maneira, foram utilizadas bases de dados contendo diversas informações das pessoas submetidas aos programas. Tais bases incluem o resultado do programa: se a pessoa foi ou não reintegrada à sociedade. A aplicação de KDD teve como meta caracterizar o perfil dessas pessoas de forma a viabilizar um processo de triagem, que procurasse definir o perfil de novas pessoas a priori, direcionando-as a programas de reintegração mais adequados.

## Educação

Todas as escolas do estado do Rio de Janeiro responderam a questionários contendo mais de 600 perguntas referentes à sua gestão durante o ano de 2001. Uma grande base de dados gerada com as respostas desse questionário foi utilizada como fonte para um outro projeto de KDD, também realizado junto ao PRODERJ. Esse projeto teve como objetivos buscar caracterizar perfis de escolas, de forma a descobrir, dentre várias questões, por que determinadas escolas têm uma procura maior que outras, por que algumas escolas têm alto índice de evasão, e assim por diante. Uma vez respondidas tais perguntas, foi possível para o governo do estado estabelecer medidas efetivas contra alguns dos problemas detectados.

## Área Médica

Atualmente encontra-se em desenvolvimento pelo NUPAC (Núcleo de Projetos e Pesquisas em Aplicações Computacionais da UniverCidade – Centro Universi-

tário da Cidade do Rio de Janeiro) um projeto de Mineração de Dados na área da Citopatologia. Participam deste projeto, pesquisadores da própria instituição e pesquisadores externos, além de alunos de doutorado, mestrado e iniciação científica.

Mais especificamente, o Projeto de Citopatologia procura extrair conhecimentos que auxiliem na detecção e na prevenção do câncer de colo de útero. As bases de dados em análise contêm informações sobre diversas pacientes de vários municípios do Rio Grande do Sul, incluindo imagens das amostras de material (células) coletado para exame citopatológico. Considerando o grande volume de amostras a ser analisado, duas linhas de estudo estão sendo realizadas:

- A primeira tem como objetivo gerar classificadores que permitam auxiliar no diagnóstico de doenças a partir das imagens e dos dados das amostras coletadas em novas pacientes. Pretende-se utilizar o conhecimento obtido de duas formas: a) em sistemas de ensino continuado que sejam utilizados no treinamento e aperfeiçoamento de pessoal técnico especializado na análise e diagnóstico dos exames; b) no apoio à decisão no processo diário de análise e diagnóstico, procurando reduzir ao máximo a ocorrência eventual de falhas neste processo.
- A segunda procura validar algumas suspeitas quanto à alta incidência de casos de doenças provenientes de determinadas regiões do estado. Espera-se realizar um estudo de epidemiologia que: a) identifique associações recorrentes entre municípios e doenças; b) explique tais associações com base nas características regionais e sociais dos municípios envolvidos.

## Área Financeira

O objetivo deste projeto foi gerar um classificador para caracterizar clientes que pagam em dia, clientes que pagam em atraso e clientes que não pagam seus créditos. Para tanto, considerou-se o histórico de pagamento de clientes de uma financeira que haviam recebido crédito durante um período definido. O classificador gerado foi incorporado a um Sistema Especialista que funciona como sistema de apoio à decisão na análise de novas solicitações de crédito recebidas na central de atendimento da referida financeira.

Um projeto similar a esse resumido acima teve como objetivo descobrir uma função que mapeasse as características de clientes (pessoas físicas) em limites de cartão de crédito oferecido a estes clientes.

## Outros Exemplos

Um exemplo interessante de aplicação de KDD situa-se na área de arrecadação de impostos. Nesta aplicação, o objetivo do processo de KDD consiste em gerar

um modelo de conhecimento que possa, baseado no comportamento histórico de pagamentos de impostos, identificar potenciais fraudes em novos pagamentos. De forma similar, mecanismos de detecção de fraudes em compras de cartão de crédito podem ser abstraídos a partir do comportamento de compra prévio de cada cliente individualmente.

Pelo menos dois tipos de aplicações de KDD podem ser realizados na área de seguros. Em um deles, o objetivo seria obter um modelo de conhecimento que pudesse, baseado nas características do solicitante de uma apólice de seguro, sugerir um valor de apólice compatível. A detecção de fraudes em sinistros também consiste em um bom exemplo de aplicação de KDD na área de seguros.

Um projeto importante na área da produção consiste na aplicação de técnicas de KDD para geração de modelos que façam a previsão de demanda de energia elétrica para determinados períodos. Nesse caso, a tarefa de previsão de séries temporais deve ser realizada, utilizando-se, para tanto, de registros de consumos de energia elétrica ao longo de períodos anteriores. Aplicação similar pode ser realizada na produção de insumos industrializados, baseado em históricos de volumes de vendas anteriores. Um modelo de conhecimento que permita previsões deste tipo pode ser incorporado a um sistema de planejamento da produção. Em um exemplo análogo, previsões de séries temporais podem ser realizadas a partir do histórico de cotações de ações da bolsa de valores.

# Considerações Finais

## Resumo

Considerando os avanços e a relevância que a área da Descoberta de Conhecimento em Bases de Dados (*Knowledge Discovery in Databases – KDD*) vem ganhando no contexto mundial, este livro tem como principal objetivo fornecer uma introdução sobre o assunto. Para tanto, foram apresentados alguns conceitos básicos, técnicas, orientações e exemplos práticos de aplicações em KDD e Mineração de Dados.

Procurando suprir uma lacuna na literatura nacional sobre KDD, este livro enfoca a Descoberta de Conhecimento em Bases de Dados como um processo, apresentando, discutindo e ilustrando cada uma das etapas e funções envolvidas.

De uma maneira geral, a complexidade do processo de KDD está na dificuldade em perceber e interpretar adequadamente inúmeros fatos observáveis durante o processo e na dificuldade em conjugar dinamicamente tais interpretações de forma a decidir que ações devem ser realizadas em cada caso.

Conforme exposto anteriormente, a participação humana na condução do processo de KDD é de fundamental importância para o sucesso das aplicações. Por outro lado, a formação de pessoal especializado em KDD é uma tarefa árdua, longa e exaustiva. Requer um amplo estudo e o domínio sobre a fundamentação teórica (envolvendo conceitos, técnicas e algoritmos) aliada a uma intensa vivência prática de diversas experiências reais. Somente a prática proporciona os meios para desenvolver a intuição muitas vezes necessária à coordenação e à execução de aplicações de KDD.

A fim de permitir ao leitor vivenciar várias das práticas mencionadas ao longo do texto, está disponível em CD uma versão de demonstração do Bramining, software de Mineração de Dados nacional desenvolvido pelos autores. Algumas bases de dados de exemplo também foram incluídas no CD. Diversas destas bases, no entanto, podem ser obtidas na Internet, em sites específicos (vide seção “Sites Interessantes”).

## Tendências e Perspectivas

A área de Mineração de Textos é uma área em franca expansão na atualidade, que vem despertando interesse de diversas áreas. Propositalmente esse assunto não foi abordado ao longo do texto. Nós, autores, entendemos que este tema, embora relacionado com a área de KDD, merece um tratamento especial, devendo ser objeto de uma publicação específica.

Uma outra área pouco explorada ao longo do livro, mas não menos importante, é a de Mineração de Dados Paralela e Distribuída. Tal importância pode ser facilmente percebida na medida em que as bases de dados aumentam exponencialmente seu volume, algoritmos mais robustos e escaláveis se tornam essenciais. Diversas pesquisas de relevância mundial vêm sendo desenvolvidas nesta área nos últimos anos. São intensas as expectativas de contribuição e crescimento futuro dos estudos em Mineração de Dados Paralela e Distribuída. De forma análoga à Mineração de Textos, nós também acreditamos que os temas tratados nesta área merecem tratamento diferenciado em publicações específicas.

A atual escassez de profissionais com conhecimento para atuar na área de KDD, combinada com a velocidade com que as bases de dados vêm aumentando de tamanho e se multiplicando em âmbito mundial, tem contribuído decisivamente para uma intensificação da demanda por aplicações nesta área. A maioria das grandes bases de dados históricas sobre algum assunto possui um forte potencial para a aplicação da tecnologia de KDD. São inúmeras as alternativas de atuação em KDD: desde a pesquisa voltada ao desenvolvimento tecnológico até a realização de aplicações práticas e a utilização de resultados. Assim, diante deste cenário de oportunidades, a formação de pessoal especializado em KDD vem se tornando uma opção de grande procura e interesse na atualidade.

# Noções Introdutórias em Data Warehouses

## Motivação e Conceitos Básicos

Com o advento da globalização, a competitividade entre as empresas no mundo dos negócios vem aumentando intensamente nos últimos anos. As empresas contemporâneas, cientes da necessidade de adaptação a esse cenário, têm investido na captação, no armazenamento, no tratamento e na aplicação da informação como diferencial estratégico e competitivo na condução dos negócios. Recursos da área da Tecnologia da Informação têm sido fundamentais nesse processo. Em particular, muitos sistemas de informação vêm sendo desenvolvidos e utilizados em diversas aplicações.

A maioria dos sistemas de informação opera sobre bancos de dados chamados transacionais. Esses bancos de dados contêm informações detalhadas que permitem às empresas acompanhar e controlar processos operacionais. Por outro lado, existe uma demanda cada vez maior por sistemas de informação que auxiliem no processo de tomada de decisão. Gerentes e executivos necessitam de recursos computacionais que forneçam subsídios para apoio ao processo decisório, sobretudo nos níveis tático e estratégico das empresas.

Conceptualmente, um Data Warehouse é um conjunto de dados baseado em assuntos, integrado, não-volátil, variável em relação ao tempo, e destinado a auxiliar em decisões de negócios. A orientação a assunto, aliada ao aspecto de integração, permite reunir dados corporativos em um mesmo ambiente de forma a consolidar e apresentar informações sobre um determinado tema. Os dados são não voláteis, pois uma vez carregados no Data Warehouse, estes não podem mais sofrer alterações. Cada conjunto de dados, ao ser carregado em um Data Warehouse, fica vinculado a um rótulo temporal que o identifica dentre os demais. Cada rótulo temporal fica associado, portanto, a uma visão instantânea e sumarizada dos dados operacionais que corresponde ao momento de carga do Data Warehouse. Desta forma, na medida em que o Data Warehouse vai sendo carregado com tais visões, pode-se realizar análises de tendências a partir dos dados.

Convém reforçar as diferenças entre Data Warehouses e bases de dados operacionais. Uma base de dados operacional é um banco de dados clássico que contém informações detalhadas a respeito do negócio em nível transacional.

Nas bases de dados tradicionais, normalmente, os dados encontram-se voltados para a representação de detalhes operacionais corporativos. Em Data Warehouses, os dados encontram-se consolidados de forma a prestar informações para os níveis gerencial e estratégico das empresas. Em geral, os Data Warehouses devem disponibilizar dados sobre a história da empresa de forma a viabilizar consultas, descoberta de tendências e análises estratégicas a partir dos dados. O exemplo abaixo procura ilustrar, de forma simples, o exposto acima.

Considere uma base de dados transacional relacional contendo os detalhes de cada venda realizada por seus vendedores durante um mês. A cardinalidade desta relação representa o número de vendas realizadas durante o mês. Imagine, a título de exemplo, que tenham ocorrido 1.000 vendas durante o mês.

#### *Venda(Seqüencial, Data, Hora, Vendedor, Valor)*

Um Data Warehouse sobre Venda, gerado a partir do exemplo acima poderia conter um resumo sobre as vendas realizadas:

#### *Venda\_Mensal(Mês, Ano, Vendedor, Total)*

Convém perceber que nesta estrutura existiria apenas uma tupla referente a cada vendedor no mês, indicando o total vendido.

O nível de consolidação de informação varia em função da necessidade de cada aplicação e deve ser definido durante o processo de modelagem e construção do Data Warehouse.

A granularidade de uma informação corresponde ao grau de consolidação de informação envolvido. Por exemplo, a relação *Venda* acima apresenta uma maior granularidade do que a relação *Venda\_Mensal*. O detalhamento dos dados na primeira relação é maior do que na segunda.

Sistemas de Informação que utilizam bases de dados transacionais são muitas vezes denominados de aplicações OLTP (*On-Line Transactional Processing*). Por outro lado, Sistemas de Informação que acessam Data Warehouses são usualmente chamados de aplicações OLAP (*On-Line Analytical Processing*). Em geral, estes últimos permitem visualização e navegação pelos dados sob diversas perspectivas e níveis de detalhe.

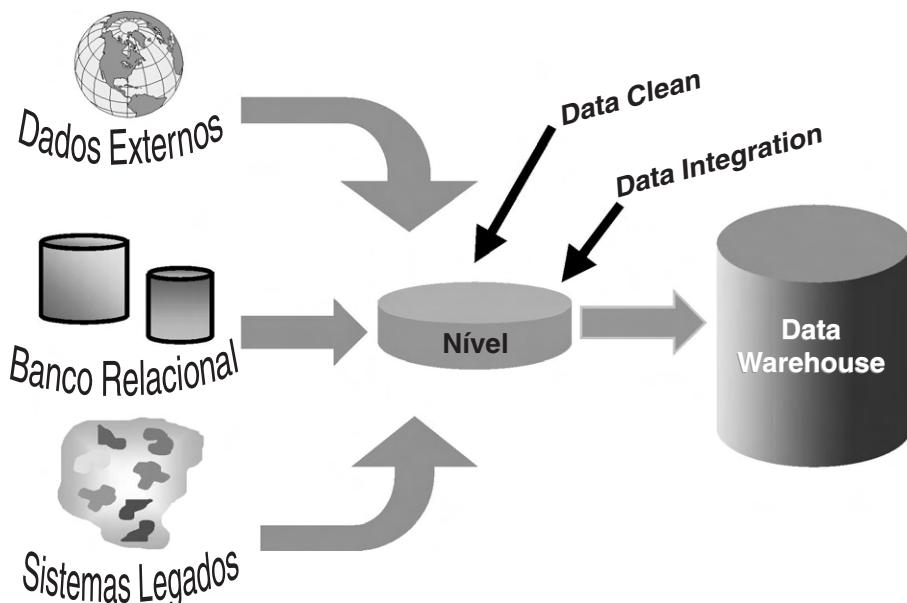
A seguir estão relacionados alguns exemplos de aplicações em Data Warehouses:

- Pesquisa de fraudes;
- Análise de crédito;

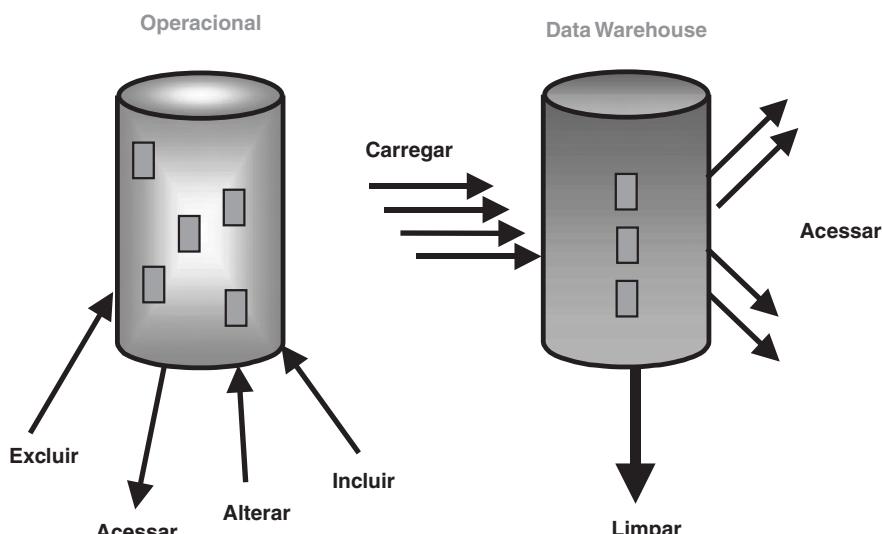
- Análise de sazonalidade da produção
- Análise de risco
- Integração de Informações de Clientes
- Rentabilidade de Clientes e Produtos
- Análises de Resultados de Vendas
- Análises de Ações de Marketing

Em geral, a passagem de dados de um ambiente operacional legado para um Data Warehouse não é tão simples quanto uma mera extração e carga de dados. Muitas vezes há a necessidade de transformação e consolidação dos dados. Existem ferramentas que permitem o chamado processo de ETC (Extração, Transformação e Carga) dos dados, muito úteis para auxiliar no processo de criação de Data Warehouses.

O processo de extração dos dados oriundos de diversas fontes de dados operacionais, transformação e carga em Data Warehouses normalmente prevê um estágio intermediário de preparação dos dados, sincronização e integração dos dados. Durante este estágio, os dados permanecem em uma área de armazenamento intermediária entre as bases de dados operacionais e o Data Warehouse. Ações de limpeza e integração dos dados são realizadas neste momento, conforme ilustra a figura a seguir.



Em bases de dados operacionais, em contrapartida aos Data Warehouses, são permitidas atualizações constantes sobre os dados. O conteúdo de um Data Warehouse somente sofre alterações no momento de carga de novo conjunto de dados, associado a um novo rótulo temporal.



Como a construção e a manutenção de Data Warehouses envolvem um contínuo pré-processamento dos dados (limpeza, transformações, integração, dentre outras ações), estes repositórios são fontes potenciais de informação para serem submetidas ao processo de descoberta de conhecimento em bases de dados.

Um dos maiores desafios na construção e na manutenção de Data Warehouses reside na busca pela qualidade dos dados. Diversos são os tipos de problemas existentes nas bases de dados operacionais e que devem ser tratados no projeto de Data Warehouses. Alguns deles estão citados a seguir:

- Ausência de informação
- Valores inválidos
- Ausência de integridade referencial
- Violações de regras de negócios
- Cálculos inválidos
- Formatos não-padronizados
- Duplicação de informação e inconsistência
- Falhas na modelagem das bases de dados operacionais

Um Data Mart é uma porção *física* ou *lógica* de um Data Warehouse para atender a uma área específica da empresa.

Trata-se de um subconjunto do Data Warehouse. Muitas vezes data marts são criados de forma a oferecer simplicidade, menor custo e agilidade ao processo de construção e manutenção de Data Warehouses.

Uma estratégia comum na construção de Data Warehouses envolve a construção paulatina destes por meio de data marts.

O cubo de dados (ou hipercubo de dados) é um recurso que permite o cruzamento e a visualização dos dados em aplicações OLAP. A figura a seguir contém um exemplo de cubo de dados com três dimensões (local, produto e período), onde cada célula mostra o total vendido de um determinado produto em uma determinada semana em um determinado local.

		Mercado					
		ASIA			EUR.		
		EUA					
<u>Mercado</u>		Prod 1	\$ 120	\$ 115	\$ 123		
<u>Produto</u>		Prod 2	\$ 60	\$ 75	\$ 73		
		Prod 3	\$ 92	\$ 87	\$ 106		
			Sem1	Sem2	Sem3		
						<u>Tempo</u>	

Os metadados assumem um papel de grande relevância na manutenção e na expansão de Data Warehouses. Devem conter informações sobre diversos aspectos, dentre os quais, podem ser destacados:

- A fonte original dos dados e como ter acesso.
- Os responsáveis pela informação original.
- Como são criados os relatórios e para que servem.
- As consultas disponíveis para acesso a determinadas informações.
- Como as definições de negócio e terminologia mudaram ao longo dos anos.
- Quais premissas de negócio foram assumidas na modelagem do Data Warehouse.

## Modelagem Multidimensional

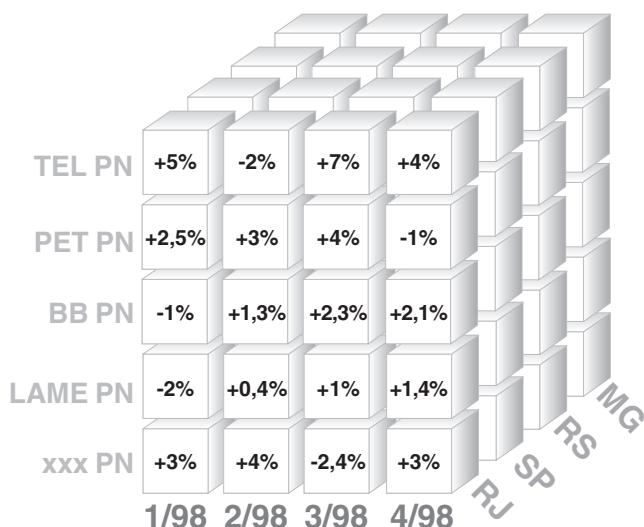
A modelagem multidimensional é uma forma de Modelagem de Dados voltada para concepção e visualização de conjuntos de medidas que descrevem aspectos comuns de um determinado assunto. É utilizada especialmente para sumarizar e reestruturar dados, apresentando-os em visões que suportem a análise dos valores envolvidos.

Enquanto a modelagem de dados tradicional assegura o cumprimento de restrições e evita redundância de informação, a modelagem multidimensional facilita a realização de consultas por usuários não técnicos, acelerando o desempenho destas consultas e admitindo redundância de informação.

O exemplo abaixo ilustra uma modelagem multidimensional onde, a partir das informações de ação, bolsa e mês, expressa-se a lucratividade da ação:

Ação	Bolsa	Mês	Lucratividade
Tel PN	Rio de Janeiro	Janeiro	+5%
Tel PN	Rio de Janeiro	Fevereiro	-2%
Tel PN	Rio de Janeiro	Março	+7%
Tel PN	São Paulo	Janeiro	+4%
Tel PN	São Paulo	Fevereiro	-1%
Tel PN	São Paulo	Março	+4%
Pet PN	São Paulo	Janeiro	+2,5%
BB PN	Rio de Janeiro	Janeiro	-1%

O cubo de dados possui recursos adequados para visualização das informações modeladas em um formato multidimensional. A figura a seguir apresenta o cubo de dados (com apenas três dimensões) do exemplo mencionado:



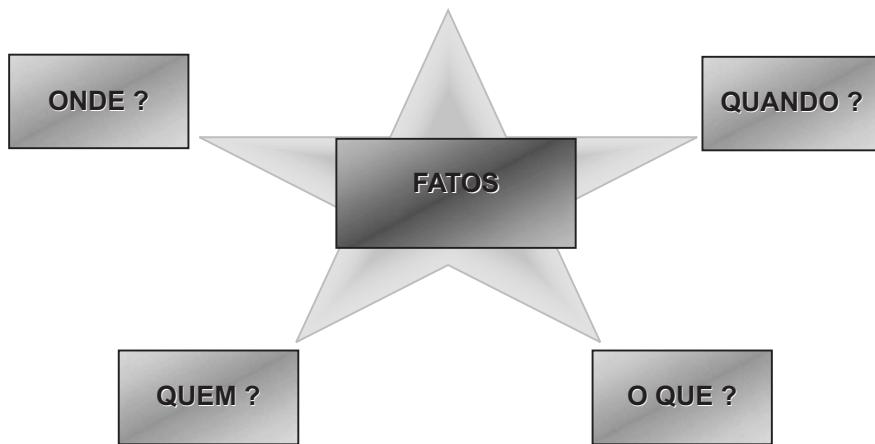
Um modelo multidimensional possui três componentes básicos:

- Fatos – Um fato é uma coleção de itens de dados, composta de dados de medida e de contexto. Representa um item, ou uma transação ou um evento associado ao tema da modelagem. Exemplo: uma tupla da relação acima.
- Dimensões – Uma dimensão é um tipo de informação que participa da definição de um fato. No exemplo: ação, local, mês. As dimensões determinam o contexto do assunto. Normalmente são descritivas ou classificatórias. Em geral, as perguntas “O quê? Quem? Onde? Quando?” ajudam a identificar as dimensões de um assunto.
- Medidas – Uma medida é um atributo ou variável numérica que representa um fato. Exemplos: valor da ação, número de evasões escolares, quantidade de produtos vendidos, valor total de venda etc.

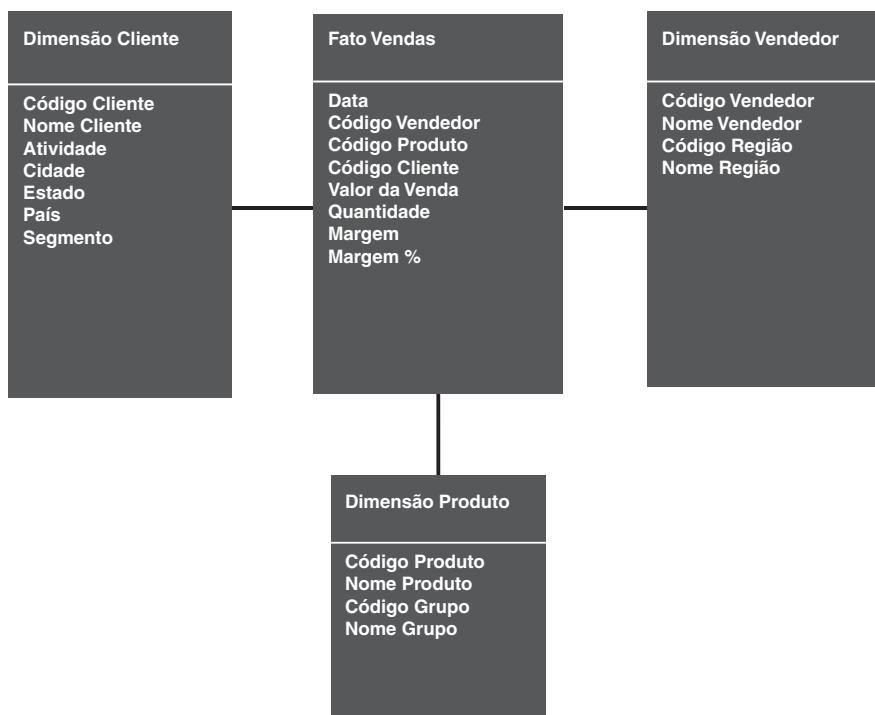
Existem diversos operadores OLAP que permitem acessar os dados em modelos multidimensionais. A seguir encontram-se indicados alguns deles:

- Drill up/down – Utilizado para aumentar ou reduzir o nível de detalhe da informação acessada. Exemplo: Vendas por país, Vendas por UF.
- Slicing – Utilizado para selecionar as dimensões a serem consideradas na consulta. Exemplo: Visualizar as vendas, separadas por país e por mês.
- Dicing – Utilizado para limitar o conjunto de valores a ser mostrado, fixando-se algumas dimensões. Exemplo: Vendas no estado de Minas, de um determinado produto em um determinado ano.
- Pivoting – Utilizado para inverter as dimensões entre linhas e colunas. Exemplo: Ao visualizar vendas por produto e por estado, aplicar o operador para visualizar as vendas por estado e por produto.
- Data Surfing – Executar uma mesma análise em outro conjunto de dados. Exemplo: Ao visualizar as vendas no Brasil, aplicar o operador para realizar a mesma consulta na Inglaterra.

Existem diversas formas de modelagem física de um Data Warehouse. Uma das mais populares é a esquema estrela. Nesse esquema, uma relação central de fatos é cercada por relações que correspondem às dimensões do problema. As dimensões no esquema estrela são usualmente denominadas pontos cardeais, conforme mostra a figura a seguir (Neri, 2002).

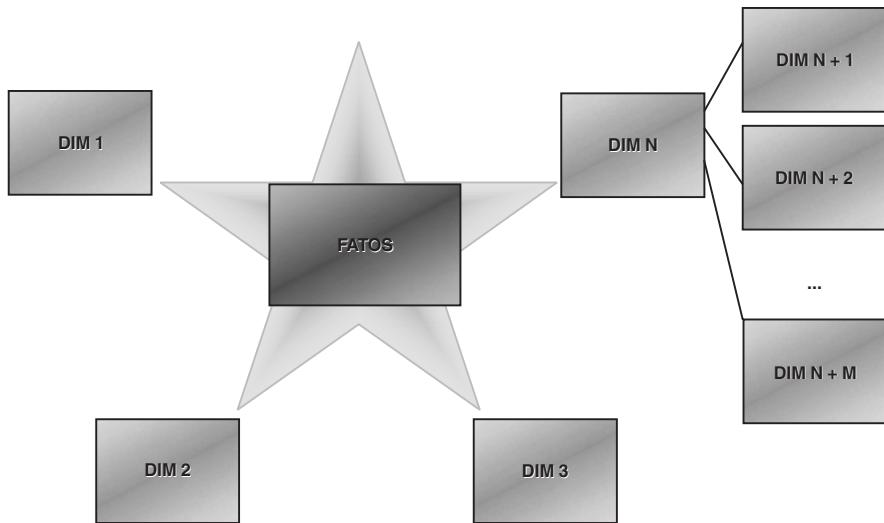


A seguir, encontra-se um exemplo de Data Warehouse modelado segundo o esquema estrela:



Os fatos estão na tabela Vendas. As dimensões estão representadas pelas tabelas Produto, Cliente e Vendedor. As medidas neste exemplo são: valor da venda, quantidade, margem e margem %.

O esquema estrela pode ser estendido de forma a compor o esquema “Floco de Neve” (SnowFlake). Nesse esquema, as dimensões podem ser associadas a novas dimensões, conforme ilustra a figura a seguir.



Para maiores detalhes sobre Data Warehouses, sugerimos (Néri, 2002) e como uma excelente introdução, com muitos exemplos práticos.

# Noções Introdutórias em Redes Neurais

## Considerações Iniciais

Em termos intuitivos, redes neurais artificiais (RNAs) são modelos matemáticos inspirados nos princípios de funcionamento dos neurônios biológicos e na estrutura do cérebro. Esses modelos têm capacidade de adquirir, armazenar e utilizar conhecimento experimental e buscam simular computacionalmente habilidades humanas tais como aprendizado, generalização, associação e abstração.

A tabela abaixo resume a relação entre características e comportamentos de RNAs com elementos da natureza.

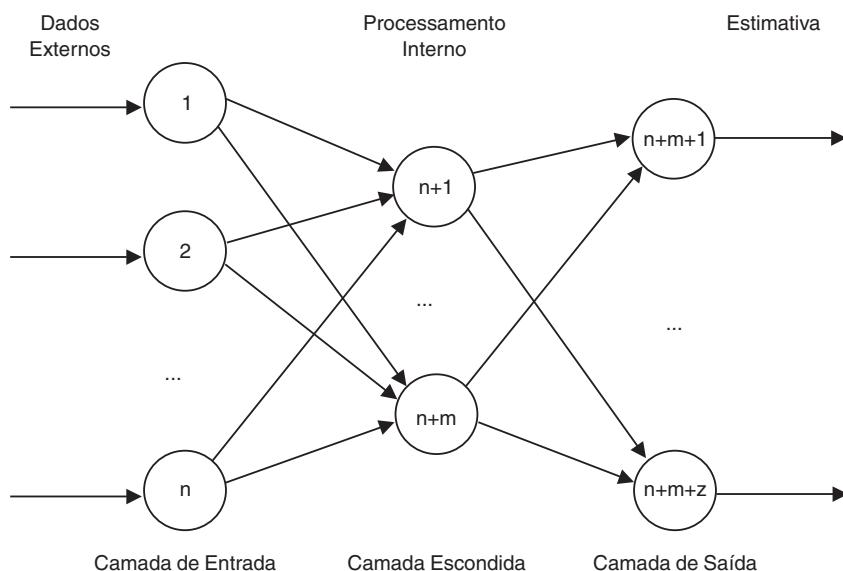
Modelo Natural	Modelo Artificial
Cérebro	RNA
Neurônio Biológico	Neurônio artificial/ elementos processadores
Rede de Neurônios	Estrutura em camadas
10 bilhões de Neurônios	Centenas/ milhares de neurônios

Devido à sua similaridade com a estrutura do cérebro, as RNAs apresentam algumas características similares às do comportamento humano, tais como:

- Busca paralela e endereçamento pelo conteúdo – O cérebro não possui endereço de memória. Analogamente, nas RNAs o conhecimento fica distribuído pela estrutura das redes, de forma que a procura pela informação ocorre de forma paralela e não seqüencial.
- Aprendizado por Experiência – As RNAs tentam aprender padrões diretamente a partir dos dados. Para isso, utilizam um processo de repetidas apresentações dos dados à rede que busca abstrair modelos de conhecimento de forma automática. Esse processo é denominado *aprendizado*, e é implementado por um *algoritmo de aprendizado*.

- Generalização – As RNAs são capazes de generalizar seu conhecimento a partir de exemplos anteriores. A capacidade de generalização permite que RNAs lidem com ruídos e distorções nos dados, respondendo corretamente a novos padrões.
- Associação – As RNAs são capazes de estabelecer relações entre padrões de natureza distinta. Por exemplo, identificar pessoas a partir de características da voz destas pessoas.
- Abstração – Abstração é a capacidade das RNAs em identificar a essência de um conjunto de dados de entrada. Isto significa que as RNAs são capazes de perceber quais as características relevantes em um conjunto de entradas. Assim, a partir de padrões ruidosos as RNAs podem extrair as informações dos padrões sem ruído.
- Robustez e Degradação Gradual – Como a informação fica distribuída em uma RNA, a perda de um conjunto de neurônios artificiais não causa necessariamente o mau funcionamento desta rede. Na realidade, o desempenho de uma RNA tende a diminuir gradativamente na medida em que aumenta a quantidade de neurônios artificiais inoperantes.

Semelhante ao sistema biológico, uma RNA possui, simplificadamente, um sistema de neurônios, e conexões ponderadas por valores reais denominados pesos. Numa RNA os neurônios são arrumados em camadas, com conexões entre elas. A figura abaixo ilustra graficamente a arquitetura de uma RNA simples. Os círculos representam os neurônios e as linhas representam os pesos das cone-

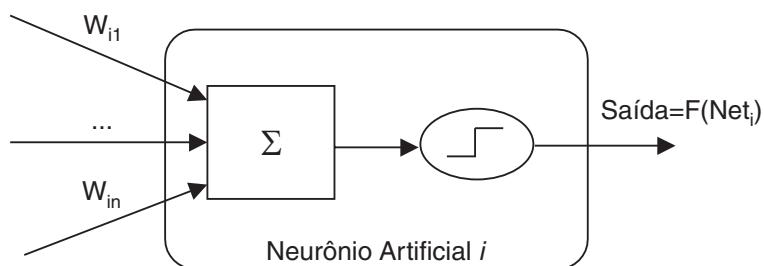


xões. Por convenção, a camada que recebe os dados é chamada camada de entrada e a camada que mostra o resultado é chamada camada de saída. A camada interna, onde ocorre o processamento interno da rede, é tradicionalmente chamada de camada escondida. Uma RNA pode conter uma ou várias camadas escondidas, de acordo com a complexidade do problema.

Em geral o processamento das RNAs ocorre da esquerda para a direita (*RNAs feed forward*). Para fins computacionais os neurônios são rotulados com uma numeração seqüencial de cima para baixo, da esquerda para a direita, conforme ilustrado na figura anterior.

A transmissão do sinal de uma célula para outra é um complexo processo químico, no qual substâncias específicas são liberadas pelo neurônio transmissor. O efeito é um aumento ou uma queda no potencial elétrico no corpo da célula receptora. Se este potencial alcançar o limite de ativação da célula, um pulso ou uma ação de potência e duração fixa é enviado para outros neurônios. Diz-se então que o neurônio está ativo.

O neurônio artificial foi projetado para imitar algumas das principais características de um neurônio biológico. Essencialmente, as entradas são aplicadas a um neurônio artificial, cada uma representando a saída de outros neurônios conectados a ele (vide figura a seguir). Cada entrada é multiplicada por um peso correspondente ( $W_{ij}$ ), gerando entradas ponderadas, de forma análoga à força das *sinapses*. Em seguida todas estas entradas ponderadas são somadas, obtendo-se um valor *NET* (potencial de ativação do neurônio artificial) que será comparado com o valor limite para ativação do neurônio (*F*). Caso esse valor alcance o valor limite de ativação do neurônio, ele será ativado. Caso contrário, ele permanecerá inativo. A figura a seguir mostra o modelo que implementa essa idéia.



## Modelagem

Conforme pode ser observado na figura anterior, em essência, os elementos básicos de um neurônio artificial são:

- Conexões entre os Processadores – A cada conexão de entrada de um neurônio artificial, existe um valor real, denominado peso sináptico, que determina o efeito dessa entrada sobre o neurônio em questão. Quanto maior/menor esse valor, maior a influência positiva/negativa do neurônio de onde sai a referida conexão. De forma análoga, a cada conexão de saída há um peso sináptico que determina a influência desse neurônio sobre o neurônio de chegada dessa conexão.
- Regra de Propagação – É a forma como as entradas provenientes de outros neurônios artificiais são combinadas aos pesos sinápticos correspondentes de um determinado neurônio ( $j$ ). A regra de propagação estabelece o potencial de ativação do neurônio ( $net_j$ ). Uma regra de propagação muito utilizada é o produto escalar entre o vetor de entrada e o vetor de pesos  $\sum_{i=1}^n O_i W_{ji}$ .
- Função de Ativação – Determina o novo valor do estado de ativação de um neurônio artificial, a partir de seu potencial de ativação  $Net_j$ . Determina saída efetiva do neurônio artificial. Esta função pode ter várias formas: uma função linear; uma função limiar (função degrau); ou ainda uma função que simule mais precisamente as características não lineares do neurônio biológico. No caso de  $F$  ser uma função linear, a saída do neurônio é expressa por:

$$F(Net_j) = O_j = K * Net_j$$

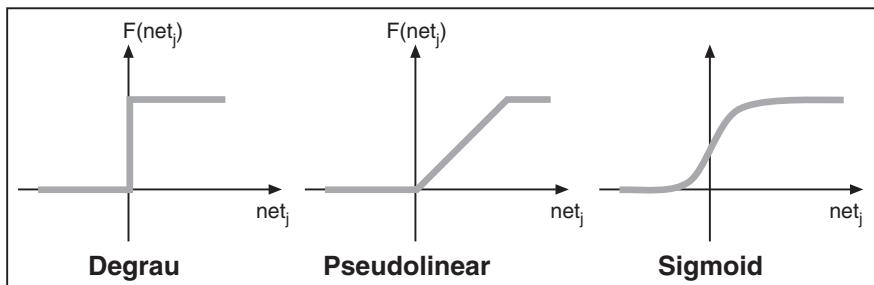
onde  $K$  é uma constante.

Caso  $F$  seja uma função degrau, a saída seria:

$$O_j = \begin{cases} 1, & \text{se } ..net_j \geq T \\ 0, & \text{se } ..net_j < T \end{cases}$$

onde  $T$  é o valor de limiar do neurônio artificial (constante).

No caso de se querer simular com maior precisão os neurônios biológicos, podem ser utilizadas funções não lineares. Elas se caracterizam por não deixarem a saída do neurônio exceder certos valores limites, os quais, em geral, são menores que os valores de  $Net_j$ . A função não linear mais utilizada é a função sigmóide, a qual é representada matematicamente por  $F(x) = 1/(1 + e^{-x})$ . Adequando-a ao neurônio artificial tem-se:  $O_j = 1/(1 + e^{-net_j})$ . A figura a seguir apresenta graficamente algumas das principais funções de ativação utilizadas em RNAs.



É interessante ressaltar que o modelo simples de neurônio artificial apresentado ignora diversas características do neurônio natural, tais como a não consideração dos atrasos de tempo que afetam a dinâmica do sistema – as entradas produzem saídas imediatas – e a não inclusão dos efeitos de sincronismo ou de modulação de freqüência – característica que alguns pesquisadores acham de fundamental importância. Apesar dessas limitações, as RNAs formadas por simples neurônios artificiais possuem atributos semelhantes aos do sistema biológico, como a capacidade de aprendizado e generalização, podendo-se dizer que a essência do funcionamento do neurônio natural tenha sido absorvida.

As várias topologias de redes neurais estão divididas, basicamente, em duas classes:

- Não-recorrentes
- Recorrentes

As RNAs *não-recorrentes* são aquelas que não possuem realimentação de suas saídas para suas entradas e por isso são também ditas “sem memória”. A estrutura das RNAs ditas não-recorrentes é em camadas, podendo estas RNAs serem formadas por uma (*RNA de camada única*) ou mais camadas (*RNA multicamada*). Redes neurais multicamadas contêm um conjunto de neurônios de entrada, uma camada de saída e uma ou mais camadas escondidas. A camada de entrada da rede apenas distribui os padrões. A camada de saída apresenta o resultado final do processamento da rede. As RNAs de uma camada são também chamadas de “*perceptrons*”, e possuem um espectro de representações limitado. As RNAs multicamadas, por suprirem as deficiências das redes de uma única camada, possuem uma gama maior de aplicações bem-sucedidas.

## Processamento

É importante destacar que existem dois tipos de processamento em RNAs. O primeiro tipo é chamado de aprendizado ou etapa de treinamento. Em linhas gerais, o treinamento de uma RNA é o processo de atualização dos pesos sinápticos para a aquisição de conhecimento a partir dos dados. O segundo tipo de

processamento é chamado de teste ou recuperação da informação. O teste de uma RNA é o processo de cálculo da saída da rede a partir da apresentação de padrões de entrada.

De todas as propriedades interessantes das redes neurais artificiais, nenhuma captura tão bem a característica humana como a habilidade de aprender. Em vez de especificar todos os detalhes de uma computação tem-se a possibilidade de treinar uma rede para fazer esta computação. Isso significa que as redes neurais podem tratar problemas em que regras apropriadas são muito difíceis de se conhecer previamente, sendo essas regras abstraídas a partir dos dados.

O objetivo do treinamento de uma RNA é fazer com que a aplicação de um conjunto de entradas produza um conjunto de saídas desejado ou, no mínimo, um conjunto de saídas consistentes. Cada conjunto de entrada ou saída é chamado de vetor. O treinamento é realizado pela aplicação seqüencial dos vetores de entradas (e em alguns casos também os de saída), enquanto os pesos da rede são ajustados de acordo com um procedimento de treinamento predeterminado (algoritmo de treinamento). Durante o treinamento de uma rede, espera-se que os pesos gradualmente converjam para determinados valores, tal que a aplicação dos vetores de entrada produza as saídas desejadas.

Os procedimentos de treinamento que levam as RNAs a aprender determinadas tarefas podem ser classificados em duas classes de treinamento:

- Supervisionado
- Não-supervisionado

O treinamento *supervisionado* necessita de um par de vetores composto do vetor de entrada e do vetor alvo que se deseja como saída. Juntos, esses vetores são chamados de par de treinamento ou vetor de treinamento, sendo interessante ressaltar que geralmente a rede é treinada com vários vetores de treinamento.

O procedimento de treinamento funciona da seguinte forma: o vetor de entrada é aplicado. A saída da rede é calculada e comparada com o correspondente vetor alvo. O erro encontrado é então realimentado através da rede e os pesos são atualizados de acordo com um algoritmo determinado a fim de minimizar este erro. Esse processo de treinamento é repetido até que o erro para os vetores de treinamento tenha alcançado níveis bem baixos.

O treinamento *não-supervisionado*, por sua vez, não requer vetor alvo para as saídas e, obviamente, não faz comparações para determinar a resposta ideal. O treinamento não-supervisionado modifica os pesos da rede de forma a produzir saídas que sejam consistentes, isto é, tanto a apresentação de um dos vetores de treinamento, como a apresentação de um vetor que é suficientemente similar a um dos vetores de treinamento, irá produzir o mesmo padrão nas saídas. O processo de treinamento não-supervisionado extrai as propriedades estatísticas do conjunto de treinamento e agrupa os vetores similares em classes.

## Aplicação

São diversos os exemplos de aplicações práticas reais das RNAs. A seguir encontram-se relacionados alguns deles:

- Reconhecimento de padrões
- Classificação de padrões
- Correção de padrões
- Previsão de séries temporais
- Suporte à decisão
- Mineração de Dados
- E muitos outros...

Caso o leitor deseje, mais detalhes sobre Redes Neurais podem ser obtidos na literatura especializada. Eis algumas sugestões: (Haykin, 1999), (Rezende, 2003) e (Lopes et al., 2000).

# Noções Introdutórias em Lógica Nebulosa

## Considerações Iniciais

A Lógica Nebulosa (Fuzzy Logic, em inglês) é uma teoria matemática que tem como principal objetivo permitir a modelagem do modo aproximado de raciocínio, imitando a habilidade humana de tomar decisões em ambientes de incerteza e imprecisão. Com conceitos e recursos da Lógica Nebulosa, pode-se construir sistemas inteligentes de controle e suporte à decisão que lidem com informações imprecisas e subjetivas, tais como:

- Investimento de alto risco
- Pressão média
- Fluxo muito intenso
- Temperatura alta
- Muito jovem

São todas expressões lingüísticas cuja interpretação pode variar de um indivíduo para outro, sendo portanto, expressões nebulosas.

Como exemplos de aplicações industriais e comerciais da Lógica Nebulosa, podem ser citados:

- Aparelhos de Refrigeração
- Filmadoras
- Freios Antiderrapantes
- Sistema de Análise de Crédito
- Detecção de Fraude em Seguradoras
- Sistema de Análise de Investimentos
- Dentre muitos outros...

Para uma melhor compreensão dos fundamentos e das aplicações da Lógica Nebulosa, os principais conceitos serão definidos e comparados com conceitos da Lógica Clássica e da tradicional Teoria de Conjuntos.

## Conjuntos Nebulosos

Existem três formas de se definir conjuntos na Teoria de Conjuntos:

- Representação Explícita – Enumerando todos os elementos que pertencem ao conjunto.

Por exemplo:  $A = \{-2, -1, 0, 1, 2\}$

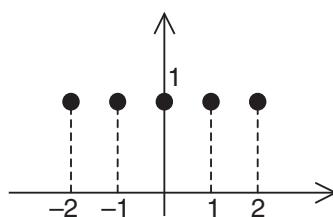
- Representação Implícita – Descrevendo uma lei de formação do conjunto, ou seja, indicando as propriedades dos elementos que pertencem ao conjunto.

Por exemplo,  $A = \{X \in Z / |x| \leq 2\}$

- Representação pela Função Característica – A função característica de um conjunto  $A$ ,  $X_A : U \rightarrow \{0,1\}$ , é uma função que associa a cada elemento do universo de discurso  $U$  um valor binário. Esse valor indica se o elemento pertence (1) ou não pertence (0) ao conjunto  $A$ .

$$X_A(x) = \begin{cases} 0, & \text{se } X \notin A \\ 1, & \text{se } X \in A \end{cases}$$

O gráfico de função característica associada ao conjunto  $A$  dos exemplos anteriores seria:



Segundo a Lógica Clássica uma sentença só pode assumir um dentre os valores verdade: Verdadeiro ou Falso. Analogamente, um elemento pertence ou não pertence a um conjunto. Não existe situação intermediária. Nesta teoria, os conjuntos e as regras são rígidos.

Consideremos os seguintes conjuntos como exemplos:

Muito Jovem =  $\{x \text{ é pessoa/ idade } (x) < 10\}$

Jovem =  $\{x \text{ é pessoa/ } 10 \leq \text{idade } (x) < 40\}$

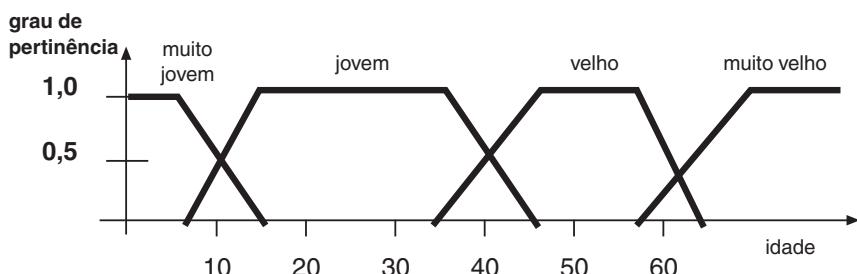
$$\text{Velho} = \{x \text{ é pessoa} / 40 \leq \text{idade}(x) < 60\}$$

$$\text{Muito Velho} = \{x \text{ é pessoa} / \text{idade}(x) \geq 60\}$$

José e João têm respectivamente 39 e 40 anos. Segundo os conceitos definidos pelos conjuntos acima, José é considerado jovem e João, velho. Há uma mudança muito abrupta de um conceito (jovem) para o outro (velho).

A Lógica Nebulosa, assim como a Teoria dos Conjuntos Nebulosos, busca introduzir mecanismos que tornem mais suave a transição entre conceitos. Um deles é a função de pertinência  $\mu_A: x \rightarrow [0,1]$  que expressa o quanto um elemento “x” pertence ao conjunto “A”.  $\mu_A(x_0)$  é o grau de pertinência do elemento  $x_0$  ao conjunto A. O grau de pertinência  $\mu_A(x_0)$  indica o quanto o valor  $x_0$  é compatível com o conceito representado pelo conjunto A. Quanto maior  $\mu_A(x_0)$  for em relação a 1, maior será a compatibilidade de  $x_0$  ao conceito A. Por outro lado, quanto menos compatível  $x_0$  for em relação ao conceito A, menor será  $\mu_A(x_0)$ . A é um conjunto nebuloso.

A figura a seguir mostra quatro exemplos de conjunto nebuloso construídos em função da variável “Idade”. Todos os conjuntos apresentam o formato de trapézios.



Pode-se perceber pelo exemplo que uma pessoa que possua 40 anos, por exemplo, pertence ao mesmo tempo aos conjuntos jovem e velho, com graus de pertinência 0,65 e 0,45, respectivamente.

$$\mu_{\text{velho}}(40) = 0,45$$

$$\mu_{\text{muito jovem}}(40) = 0$$

$$\mu_{\text{jovem}}(40) = 0,65$$

$$\mu_{\text{muito velho}}(40) = 0$$

A pessoa em questão é simultaneamente jovem e velha. Pelos graus de pertinência, observa-se que ela não é tão jovem e nem tão velha.

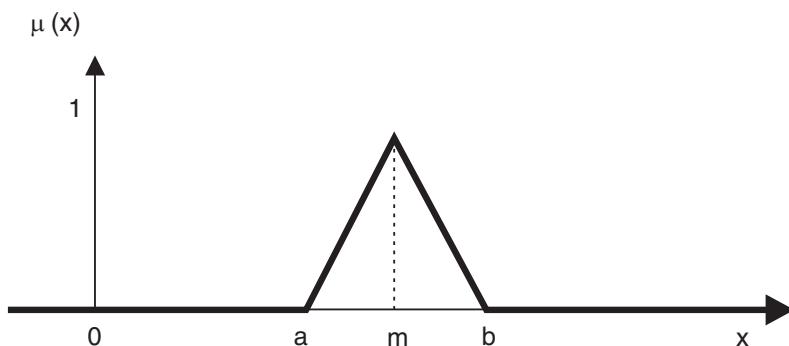
Existem diversos formatos para a representação de conjuntos nebulosos. A escolha de um determinado formato deve ser norteada pela compatibilidade do formato com o conceito que se deseja representar.

Fogem do escopo deste material a apresentação e análise destes formatos. Merecem destaque, no entanto, os formatos triangular e trapezoidal, que são muito utilizados em diversas aplicações da Lógica Nebulosa.

Uma função de pertinência triangular é expressa pela fórmula:

$$\mu(x) = \begin{cases} 0, & \text{se } x \leq a \\ (x-a)/(m-a), & \text{se } x \in ]a, m[ \\ (b-x)/(b-m), & \text{se } x \in ]m, b[ \\ 0, & \text{se } x \geq b \end{cases}$$

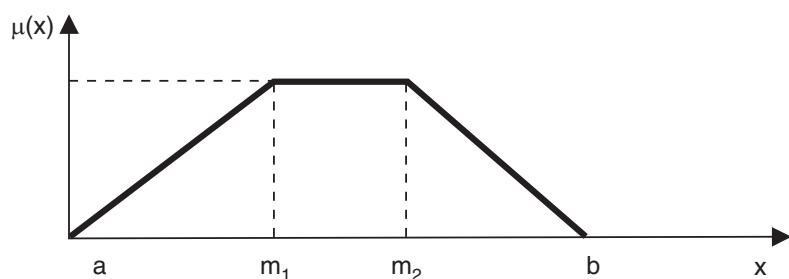
Onde:



Uma função de pertinência trapezoidal é expressa por:

$$\mu(x) = \begin{cases} 0, & \text{se } x \leq a \\ (x-a)/(m_1-a), & \text{se } x \in ]a, m_1[ \\ 1, & \text{se } x \in [m_1, m_2] \\ (b-x)/(b-m_2), & \text{se } x \in ]m_2, b[ \\ 0, & \text{se } x \geq b \end{cases}$$

Onde:



## Operadores Nebulosos

Nesta seção são apresentadas algumas operações sobre conjuntos nebulosos. Todas possuem correspondentes na clássica Teoria de Conjuntos.

Sejam A e B dois conjuntos nebulosos:

- a) A função de pertinência do operador nebuloso de interseção, denominado T-Norm, pode ser definida de diversas maneiras. Abaixo estão dois dos exemplos mais usuais:

- $\mu_{A \cap B}(x) = \min \{\mu_A(x), \mu_B(x)\}$  (Mínimo)
- $\mu_{A \cap B}(x) = \mu_A(x) * \mu_B(x)$  (Produto)

Assim como na Lógica Clássica, a operação de interseção corresponde ao operador lógico de conjunção “E”.

- b) A função de pertinência do operador nebuloso de união, denominado T-Conorm (ou S – Norm), pode ser definida, por exemplo, como:

- $\mu_{A \cup B}(x) = \max \{\mu_A(x), \mu_B(x)\}$  (Máximo)
- $\mu_{A \cup B}(x) = \min \{1, \mu_A(x) + \mu_B(x)\}$  (Soma Limitada)

Assim como na Lógica Clássica, a operação de união corresponde ao operador lógico de disjunção “OU”.

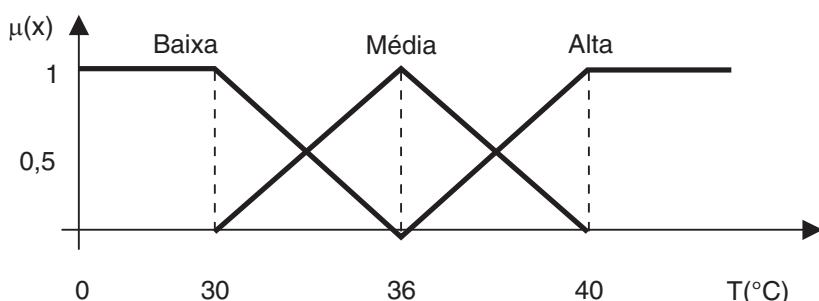
- c) A função do operador nebuloso de complemento pode ser definida por:

- $\mu_A(x) = 1 - \mu_{\bar{A}}(x)$

A operação de complemento corresponde ao operador lógico de negação “Não”.

Os valores de uma variável lingüística definem uma estrutura de conhecimento denominada de partição nebulosa desta variável.

A figura a seguir apresenta uma partição nebulosa possível para a variável temperatura.



## Variáveis Lingüísticas

Uma variável lingüística é um objeto utilizado para representar de modo impreciso um conceito em um determinado problema. Seus valores são expressões lingüísticas subjetivas tais como quente, alto, jovem. Variáveis lingüísticas diferem das variáveis numéricas, que admitem apenas valores precisos. Os valores de uma variável lingüística são conjuntos nebulosos.

Exemplo: Variável Numérica : Temperatura = 40°  
 Variável Lingüística: Temperatura Alta

## Regras Nebulosas

Uma das formas de representação nebulosa do conhecimento mais comum é a representação por meio de regras de produção nebulosas, ou simplesmente regras nebulosas. A estrutura geral de uma regra nebulosa é expressa por uma implicação do tipo:

Se < antecedente nebuloso> Então <conseqüente nebuloso>

O antecedente nebuloso é formado por condições nebulosas que, quando satisfeitas, determinam o processamento do conseqüente por um mecanismo de inferência nebulosa.

O conseqüente nebuloso, por sua vez, é composto por um conjunto de ações nebulosas ou diagnósticos nebulosos que são gerados a partir do disparo da regra.

Uma condição nebulosa, assim como uma ação ou diagnóstico nebuloso, é um predicado que envolve uma variável lingüística e um conjunto nebuloso.

Exemplos de regras nebulosas:

SE idade é *meia-idade* E pressão é *baixa* ENTÃO seguro é *baixo*

SE idade é *jovem* E pressão é *alta* ENTÃO seguro é *alto*

## Arquitetura Funcional de um Sistema de Inferência Nebuloso

Embora existam vários modelos de Inferência Nebulosa, por limitações de espaço será descrito aqui apenas o modelo Mamdani que foi, durante muitos anos, um padrão para a aplicação dos conceitos de Lógica Nebulosa em processamento de conhecimento.

Abaixo encontra-se a forma geral de uma regra típica do modelo Mamdani:

Se  $X_1 = A_1 \wedge \dots \wedge X_n = A_n$

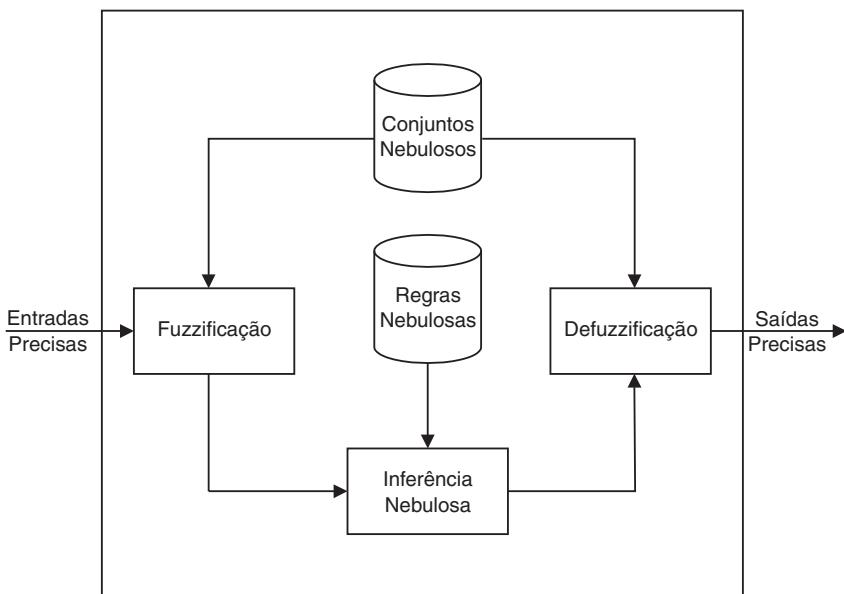
Então  $Y_1 = B_1 \wedge \dots \wedge Y_m = B_m$

A figura a seguir apresenta a arquitetura funcional genérica de um sistema de inferência nebuloso.

Para melhor compreensão do funcionamento dessa arquitetura, considere o seguinte exemplo de um sistema nebuloso para definição do valor de apólice de seguro de vida de clientes de uma seguradora.

- Variáveis lingüísticas e respectivos conjuntos nebulosos:

Variáveis	Tipo	Conjuntos Nebulosos
Idade	Entrada	Meia-idade / Jovem
Pressão	Entrada	Alta / Baixa
Seguro	Saída	Alta / Baixa



**Figura.** Arquitetura Funcional Genérica de um Sistema Nebuloso.

- Conjuntos nebulosos (com representação tabular)

Idade	20	25	30	35	40	45	50	55	50	65
Meia-idade	0,3	0,4	0,6	0,8	0,9	1,0	0,8	0,6	0,3	0,1
Jovem	0,9	0,8	0,7	0,6	0,4	0,3	0,1	0,0	0,0	0,0

Pressão Máx.	95	100	110	120	130	140	150	160	170	175
Pressão Mín.	50	55	60	65	70	75	80	85	90	100
Alta	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
Baixa	1,0	0,9	0,8	0,7	0,6	0,5	0,4	0,3	0,2	0,1

Seguro	300	500	700	800	900	1000	1200
Alto	0,1	0,3	0,4	0,5	0,8	0,9	1,0
Baixo	1,0	0,9	0,6	0,5	0,3	0,1	0,1

- Regras Nebulosas (apenas duas para simplificar o exemplo)

**SE** idade é *meia-idade* E pressão é *baixa* **ENTÃO** seguro é *baixo*

**SE** idade é *jovem* E pressão é *alta* **ENTÃO** seguro é *alto*

Deseja-se neste exemplo, determinar qual o valor da apólice de seguro a ser pago pelo cliente a partir dos valores de idade e de pressão deste cliente.

As entradas e as saídas do sistema devem ser valores escalares, precisos. Considere que desejamos saber qual o valor da apólice de seguro a ser paga pelo cliente João de 35 anos e pressão (130,70).

O primeiro módulo da arquitetura (fuzzificação) é responsável por identificar os graus de pertinência dos valores de entrada em relação aos conjuntos nebulosos correspondentes.

No exemplo:

#### Idade

$$\mu_{\text{meia-idade}}(35) = 0,8$$

$$\mu_{\text{jovem}}(35) = 0,6$$

#### Pressão

$$\mu_{\text{Alta}}(130,70) = 0,5$$

$$\mu_{\text{Baixa}}(130,70) = 0,6$$

Em seguida, o módulo de inferência nebulosa, baseado nos graus de pertinência identificados anteriormente, processa as regras nebulosas existentes na base de conhecimento.

O processamento de cada regra envolve o cálculo da interseção entre os conjuntos indicados no antecedente da regra. Esse processo gera um grau de pertinência de disparo para cada regra.

No exemplo:

#### **Regra 1:**

**Se** idade é meia-idade (0,8) **E** pressão é baixa (0,6) **Então** seguro é baixo (grau de disparo da regra = Min {0,8;0,6} = 0,6)

Assim, por esta regra o seguro é baixo com grau de pertinência 0,6

#### **Regra 2:**

**Se** idade é jovem (0,6) **E** pressão é alta (0,5) **Então** seguro é alto (grau de disparo da regra = Min{0,6, 0,5}=0,5)

Assim, por esta regra, o seguro é alto com grau de pertinência 0,5.

Finalizando, o processo de “defuzzificação” transforma os conjuntos acima em uma saída precisa. Para tanto, primeiro identifica o valor de seguro associado ao grau de disparo de cada regra:

$$\begin{array}{ll} \text{Sendo: } \mu_{\text{alto}}(X) = 0,5 & \text{e} \\ \text{Portanto: } X = 800 & \text{e} \end{array} \quad \begin{array}{ll} \mu_{\text{baixo}}(Y) = 0,6 & \\ & Y = 700 \end{array}$$

Em seguida, aplica-se a média ponderada aos valores de seguro indicados por cada regra e os respectivos graus de disparo:

$$\text{Seguro} = \frac{(0,5 \times 800) + (0,6 \times 700)}{0,5 + 0,6} = \frac{400 + 420}{1,1} = \frac{820}{1,1} = 745,45$$

Assim sendo, pelo sistema nebuloso exemplificado acima, um cliente de 35 anos com pressão 130x70 deve pagar uma apólice de seguro de R\$ 745,45 (saída precisa).

Caso o leitor deseje, mais detalhes sobre Lógica Nebulosa e Sistemas Nebulosos podem ser obtidos nas seguintes referências recomendadas: (Bojadziev & Bojadziev, 1997), (Pedrycz & Gomide, 1998) e (Rezende, 2003).

# Noções Introdutórias em Algoritmos Genéticos

## Introdução

Algoritmos Genéticos são modelos computacionais de busca e otimização de soluções em problemas complexos, inspirados em princípios da teoria da evolução natural de Charles Darwin e da reprodução genética.

Segundo o princípio básico da evolução natural de Darwin, indivíduos mais aptos possuem maiores chances de sobrevivência e, consequentemente, mais oportunidades de gerarem descendentes e perpetuarem seus códigos genéticos pelas gerações seguintes. A identificação de cada indivíduo é expressa pelo seu código genético que fica representado nos cromossomas deste indivíduo.

Resumidamente, Algoritmos Genéticos são técnicas que procuram obter boas soluções para problemas complexos por meio da evolução de populações de soluções codificadas em cromossomas artificiais. Todo problema de difícil modelagem matemática ou com um número muito grande, possivelmente infinito, de soluções é considerado um problema complexo. A tabela abaixo apresenta a analogia entre Algoritmos Genéticos e a Evolução Natural.

Algoritmos Genéticos empregam um processo adaptativo e paralelo de busca de soluções em problemas complexos. O processo é adaptativo, pois as soluções existentes a cada instante influenciam a busca por futuras soluções. O paralelismo do processo é decorrência natural do fato de que diversas soluções são consideradas a cada momento pelos Algoritmos Genéticos.

Exemplos de aplicações de Algoritmos Genéticos:

- Otimização de Funções Matemáticas
- Otimização Combinatorial
- Otimização de Planejamento
- Problema do Caixeiro Viajante
- Problema de Otimização de Rota de Veículos

- Otimização de Distribuição e Logística
- Seleção de Variáveis em Mineração de Dados
- Dentre muitos outros...

Nos Algoritmos Genéticos, um cromossoma é uma estrutura de dados que representa uma das possíveis soluções do espaço de busca de um problema. Cromossomas são então submetidos a um processo evolucionário que avalia, seleciona e combina soluções ao longo de diversos ciclos, procurando obter indivíduos mais aptos (melhores soluções).

#### *Analogia entre Algoritmos Genéticos e a Natureza*

Evolução Natural	Algoritmos Genéticos
Meio Ambiente	Problema
Indivíduo	Solução
Cromossoma	Representação (palavra binária, vetor etc.)
Gene	Característica do Problema
Alelo	Valor da Característica
Reprodução Sexual	Operador de Cruzamento
Mutação	Operador de Mutação
População	Conjunto de Soluções
Gerações	Ciclos

Em geral, um Algoritmo Genético é caracterizado pelos seguintes componentes:

- Problema
- Representação das Soluções do Problema
- Decodificação do Cromossoma
- Avaliação
- Seleção
- Operadores Genéticos
- Técnicas e Parâmetros

Abaixo seguem alguns comentários relevantes acerca de cada um destes componentes.

## Problema

Algoritmos Genéticos são particularmente úteis na solução de problemas complexos que envolvem otimização. Estes são problemas com diversos parâmetros ou características que precisam ser combinadas em busca de soluções melhores. Possuem, em geral, muitas restrições aos valores que os parâmetros podem ter, além de grandes espaços de busca.

Considere o problema de encontrar o valor máximo da função  $f(x)=x^2$ , sendo  $x$  um valor inteiro no intervalo  $[0; 63]$ . Evidentemente esse não é um problema complexo, pois o espaço de busca compreende apenas 64 pontos inteiros a serem avaliados. Além disso, um conhecimento prévio em Análise Matemática, nos permite facilmente concluir que o ponto máximo desta função é  $(63; 3969)$ . No entanto, esse exemplo será utilizado para fins didáticos na ilustração dos demais conceitos envolvidos na utilização de Algoritmos Genéticos.

## Representação das Soluções do Problema

A menor unidade de um Algoritmo Genético é chamada gene. Um gene representa uma unidade de informação do domínio do problema. Uma série de genes forma um cromossoma, que representa uma possível solução completa para o problema.

A representação de possíveis soluções do espaço de busca de um problema define a estrutura do cromossoma a ser manipulado pelo algoritmo. Qualquer que seja a forma de representação de um problema, esta deve ser capaz de representar todas as soluções do espaço de busca que se deseja investigar. A representação binária é a mais empregada, devido à sua simplicidade e facilidade de manipulação.

Para o problema anterior, é necessária uma representação para números inteiros. Podem ser utilizados binários representando números inteiros. Neste caso, seriam necessários cromossomas de 6 bits para representar números entre 0 e 63. Exemplos:

C1: 001001 representa  $x=9$

C2: 000100 representa  $x=4$

Há problemas em que valores binários podem ser utilizados para representar números reais com uma precisão de  $p$  casas decimais. Supondo que os números reais pertençam a um intervalo  $[X_{min}, X_{max}]$ , são necessários  $k$  bits de forma que  $k$  atenda à seguinte relação (Pacheco, 2000):

$$2^k \cdot (X_{max} - X_{min}) * 10^p$$

Nem sempre a representação binária pode ser empregada. Em muitos casos, o problema requer um alfabeto de representações com mais símbolos.

## Decodificação do Cromossoma

O processo de decodificação de um cromossoma consiste na construção da solução a ser avaliada pelo problema.

No exemplo acima, a decodificação consiste em transformar a string binária em um número inteiro.

## Avaliação

Para que um cromossoma seja avaliado é necessário convertê-lo em uma solução para o problema, isto é, decodificá-lo.

A avaliação de cada solução em um Algoritmo Genético é um processo realizado por uma função que tem por objetivo fornecer uma medida da qualidade de tal solução (aptidão) frente ao problema.

As funções de avaliação variam de um problema para outro. No exemplo acima, a função  $f(x)=x^2$  expressa a qualidade de cada indivíduo como solução para o problema. C1 é um indivíduo melhor (mais apto) que C2, pois  $9^2 > 4^2$ .

## Seleção

Durante o processo de seleção, o Algoritmo Genético escolhe os cromossomas, privilegiando aqueles com maiores aptidões para permanecer e se multiplicar na população. Uma forma comum de seleção é aquela em que cada cromossoma tem a probabilidade de permanecer na próxima geração proporcional à sua aptidão (método da roleta). Cromossomas com maiores aptidões possuem mais espaço na roleta e consequentemente possuem maiores chances de serem escolhidos para o processo de reprodução. Assim, se  $f_i$  é a aptidão do  $i$ -ésimo indivíduo na população corrente, a probabilidade  $p_i$  (aptidão relativa) deste indivíduo ser selecionado é obtida por:

$$p_i = \frac{f_i}{\sum_{i=1}^n f_i}$$

Onde  $n$  é o número de indivíduos na população.

No exemplo anterior, supondo que existam apenas os indivíduos C1 e C2:

Indivíduo	Aptidão Relativa
C1	$p_1=81/(81+16)=0,84$
C2	$p_2=16/(81+16)=0,16$

No método da roleta, os indivíduos são organizados em ordem crescente de aptidão, sendo sorteado um número entre 0 e 1. O indivíduo selecionado é o primeiro indivíduo cuja aptidão acumulada seja superior ao número sorteado. No exemplo:

Indivíduo	Aptidão Relativa Acumulada
C1	0,84
C2	1,00 (0,84 + 0,16)

Para números sorteados até 0,84, o indivíduo selecionado seria C1. Para valores acima de 0,84, o indivíduo selecionado seria C2.

## Operadores Genéticos

Uma vez selecionados os indivíduos, o Algoritmo Genético cria novas soluções por meio da combinação e refinamento das informações dos cromossomas usando duas operações genéticas básicas: *crossover* e *mutação*. Essas operações produzem novas soluções que formam uma nova população. Cada nova população é chamada de geração.

Durante o *crossover*, dois cromossomas trocam algumas de suas informações contidas em determinados genes. Novos indivíduos são criados a partir da troca de material genético entre seus pais. Os descendentes possuem características genéticas de ambos os genitores. Por exemplo, sendo C1 e C2, os genitores e 3 o ponto de corte (ou de cruzamento) aleatoriamente selecionado (operador de *crossover* de um ponto), D1 e D2 seriam os descendentes nesta situação.

C1: 001001

C2: 000100

D1: 001100

D2: 000001

D1 é um cromossoma mais apto que seus genitores. No entanto, D2 tornou-se um indivíduo bem menos apto.

Os cromossomas criados a partir do operador de *crossover* são submetidos à operação de mutação. A mutação permite explorar o potencial de novos indivíduos, aumentando a diversidade de indivíduos na população. O operador de mutação altera o conteúdo de uma posição do cromossoma, com uma probabilidade, em geral, bem baixa (<1%). Exemplo do indivíduo C2 antes e após sofrer mutação no último bit:

C2: 000100, antes da mutação

C2: 000101, após a mutação

Embora os operadores genéticos costumem variar em função do problema, três dos tipos de operadores de *crossover* mais comuns são:

- *Crossover* de um ponto – O *crossover* de um ponto recombina duas soluções a partir de um único ponto de corte aleatório. Esse foi o operador utilizado no exemplo acima.
- *Crossover* de dois pontos – No *crossover* de dois são dois os pontos aleatórios escolhidos como referência para a troca de valor dos atributos (genes).

Operação de crossover com dois pontos de corte

Cromossoma	Palavra			Nova Palavra
A	10	0010	10	1001110
C	11	0111	11	11001011

- *Crossover* uniforme – Esse tipo de *crossover*, por sua vez, é capaz de recombinar quaisquer posições entre dois genitores. Para tanto, utiliza uma palavra binária escolhida aleatoriamente para designar os bits selecionados em cada genitor na criação dos descendentes. Exemplo:

G1:	1100101
G2:	0111110
Padrão:	0110100
D1:	0101110
D2:	1110101

## Técnicas e Parâmetros

Em geral, as técnicas, os parâmetros e os tipos de operadores genéticos afetam significativamente o desempenho dos Algoritmos Genéticos. A escolha de técnicas, parâmetros e operadores é empírica, porém em sintonia com o tipo de problema envolvido.

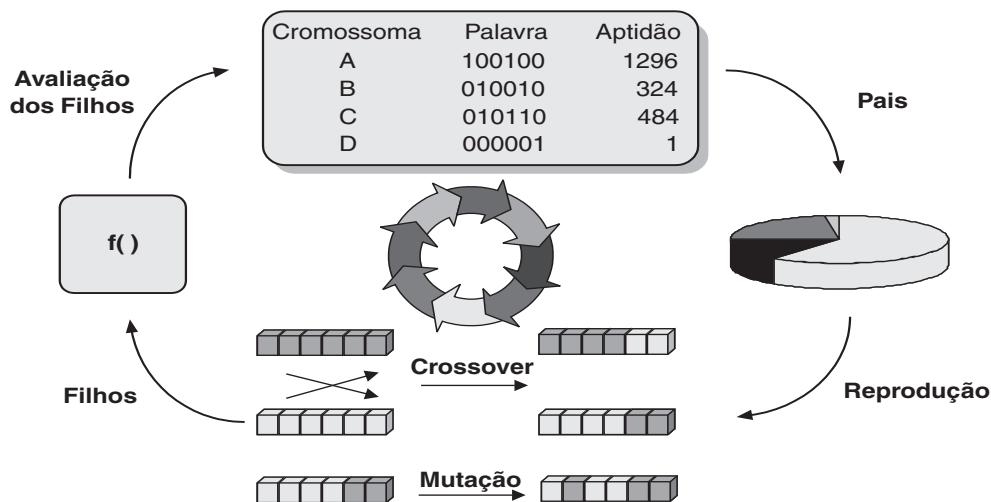
A inicialização da população determina o processo de criação dos indivíduos para o primeiro ciclo do algoritmo, sendo comum a formação da população inicial a partir de indivíduos aleatoriamente criados.

Em um Algoritmo Genético, vários parâmetros controlam o processo evolucionário:

- **Tamanho da população** – Números de pontos no espaço de busca sendo considerados em paralelo.
- **Taxa de Crossover** – Probabilidade de um indivíduo ser recombinado geneticamente com outro.
- **Taxa de Mutação** – Probabilidade do conteúdo de cada gene do cromossoma ser alterado.
- **Número de Gerações** – Quantidade ciclos do Algoritmo Genético. Pode ser deduzido caso se tenha a informação do tamanho da população e total de indivíduos (total de indivíduos / tamanho da população).
- **Total de Indivíduos** – Total de soluções a serem geradas e avaliadas pelo Algoritmo Genético (tamanho da população x número de gerações).

Os últimos dois parâmetros são normalmente empregados como critério de parada de um Algoritmo Genético, isto é, quando o processo evolucionário deve ser interrompido.

Um Algoritmo Genético pode ser descrito como um processo contínuo que repete ciclos de evolução controlados por um critério de parada, conforme ilustrado pela figura a seguir:



As técnicas de reprodução determinam o critério de substituição dos indivíduos de uma população para a geração seguinte. A seguir encontram-se citados alguns exemplos:

- **Troca de toda a População** – A cada ciclo,  $N$  novos indivíduos são criados, substituindo a população anterior. Para tanto,  $N/2$  pares são selecionados para acasalamento, gerando  $N$  descendentes.

- *Elitismo* – Todos os cromossomas são substituídos, sendo o cromossoma mais apto da população corrente copiado para a população seguinte.
- *Steady State* – Gera  $M$  indivíduos ( $M < N$ ), que substituem os piores indivíduos da população corrente. Pode haver duplicação de indivíduos na população resultante.
- *Steady State sem Duplicados* – Similar à técnica de Steady State, não permitindo a presença de indivíduos duplicados na população.

As técnicas de aptidão influenciam na forma com que as aptidões dos cromossomas em uma população são numericamente atribuídas. A seguir encontram-se citados alguns exemplos:

- Método Clássico – Atribuir como aptidão de um cromossoma o valor numérico do resultado da avaliação. Esse método, embora utilizado em muitos problemas, pode apresentar duas situações que precisam ser tratadas:
  - a) Competição próxima – Indivíduos cujas aptidões são numericamente muito próximas, dificultando a distinção entre eles quanto à qualidade da solução proporcionada.
  - b) Superindivíduos – São indivíduos com avaliação muito superior à média, que podem dominar o processo de seleção, impedindo que o Algoritmo Genético obtenha novas soluções, potencialmente melhores do que as representadas pelos superindivíduos.
- Normalização Linear – Consiste em ordenar os cromossomas em ordem crescente de avaliação e atribuir linearmente (equação a seguir) valores de aptidões aos cromossomas entre um intervalo [valor mínimo, valor máximo], distanciados de um valor fixo entre eles.

$$A_i = \min + \left[ \frac{\max - \min}{N - 1} \right] * (i - 1)$$

As técnicas de interpolação de parâmetros de um Algoritmo Genético têm como objetivo buscar o valor ideal de um determinado parâmetro para cada ciclo, durante toda a evolução.

Pode-se intuitivamente perceber que a taxa de aplicação do operador de *crossover* deve ser maior nas primeiras gerações, quando a população se apresenta dispersa pelo espaço de busca. Após várias gerações, os indivíduos tendem a apresentar, na sua maioria, características muito similares. Um incremento da taxa de mutação nesta fase da evolução propicia uma pequena dispersão da população, trazendo novo material genético para a formação de melhores indivíduos. Pelo exposto anteriormente, pode-se perceber ainda a necessidade de que parâmetros do Algoritmo Genético sejam interpolados ao longo

do processo evolutivo. Em linhas gerais, a interpolação de parâmetros pode ser linear ou adaptativa.

Na interpolação linear, uma taxa ou um parâmetro é variado entre um valor inicial e um valor final, por meio de incrementos/decrementos fixos a cada  $k$  gerações. A interpolação adaptativa, normalmente utilizada para ajuste da taxa de aplicação de operadores genéticos, considera o desempenho destes operadores nos ciclos anteriores. Esse desempenho é medido em função do sucesso dos referidos operadores na criação de melhores indivíduos.

## **Estrutura Geral de um Algoritmo Genético**

A seguir se encontra a estrutura geral básica de um Algoritmo Genético. Essa estrutura está descrita em pseudocódigo, pois tem como objetivo apresentar em nível lógico o processo evolutivo em Algoritmos Genéticos. Algumas alterações nessa estrutura são admissíveis em função do problema analisado. Por exemplo: podem ser incluídas funções para implementar técnicas de reprodução ou ainda técnicas de aptidão. Considerando a importância da função de avaliação, optou-se por apresentar em seguida a sua descrição básica. As demais funções não foram descritas, mas podem ser obtidas na maioria das aplicações de Algoritmos Genéticos disponíveis na Internet. Em particular, os autores sugerem uma consulta à biblioteca de funções de Algoritmos Genéticos, escrita em MATLAB, disponível no site: [www.ica.ele.puc-rio.br](http://www.ica.ele.puc-rio.br).

Diferentes critérios de parada podem ser utilizados para terminar a execução de um Algoritmo Genético. Exemplos: a) após um determinado número de iterações (ciclos ou gerações); b) quando a aptidão média ou do melhor indivíduo não melhorar mais; c) quando as aptidões dos indivíduos de uma população se tornarem muito parecidas.

## **Estrutura Geral de um Algoritmo Genético**

T=0

Gerar População Inicial P(0)

Avaliar P(0)

**Enquanto** Critério de Parada não for satisfeito faça:

- a. T=T+1
- b. Selecionar População P(T) a partir de P(T-1)
- c. Aplicar Operadores de Cruzamento sobre P(T)
- d. Aplicar Operadores de Mutação sobre P(T)
- e. Avaliar P(T)

**Fim Enquanto**

## Estrutura Geral de uma Função de Avaliação (Avaliar P(T))

Para todo indivíduo  $i$  da População Atual P(T) faça

Avaliar o indivíduo  $i$ , obtendo sua aptidão.

**Fim Para**

Para maiores detalhes sobre Algoritmos Genéticos, o leitor poderá recorrer às seguintes referências: (Davis, 1990), (Koza, 1992), (Michalewicz, 1994), (Pacheco, 2000), entre outras.

# Como Utilizar a Versão de Demonstração do *Bramining*

## Considerações Iniciais

Conforme mencionado no Capítulo 6, o *Bramining* é uma ferramenta operacional de KDD produzida pelos autores e seus alunos em diversas dissertações de mestrado no IME e na PUC-Rio.

Com o CD que acompanha o livro, o leitor poderá instalar uma versão de demonstração do *Bramining* para uso acadêmico. Esta versão possui todas as funcionalidades da versão oficial da ferramenta, mas permite apenas a utilização em bases de dados com até mil registros. Dessa forma, o leitor poderá experimentar diversas das técnicas apresentadas ao longo do texto, auxiliando seus estudos.

O *Bramining* é um ambiente para a realização operacional de processos de KDD e, conceitualmente, possui três níveis, denominados níveis funcionais. São eles: (a) Nível dos Métodos; (b) Nível das Operações; (c) Nível das Etapas.

O Nível dos Métodos é o nível funcional mais baixo e contém os métodos que se encontram disponíveis no *Bramining* para utilização durante a realização de aplicações de KDD. Os métodos são classificados em operações. Uma operação de KDD é uma especificação lógica de um grupo de métodos que têm a mesma finalidade. As operações disponíveis no *Bramining* compõem o nível funcional intermediário da ferramenta, denominado Nível das Operações. O Nível das Etapas é o nível funcional mais elevado. Nele as operações de KDD são agrupadas nas etapas do processo de KDD: pré-processamento, mineração de dados e pós-processamento.

Na próxima seção, o leitor encontrará um “manual do usuário” simplificado, explicando em linhas gerais como utilizar o *Bramining*. São apresentadas as principais interfaces da ferramenta além de indicações resumidas sobre como utilizá-las. Abaixo de cada interface, estão indicadas as principais funções disponíveis que podem ser acessadas no momento. A fim de facilitar o entendimento quanto à utilização do sistema, as interfaces serão apresentadas de forma enca-deada, utilizando exemplos.

De uma forma geral, as interfaces são auto-explicativas e seguem um mesmo padrão de apresentação, facilitando o rápido aprendizado por parte de seu usuário. Informações técnicas sobre operações e métodos implementados no *Bramining* podem ser obtidas ao longo dos capítulos deste livro.

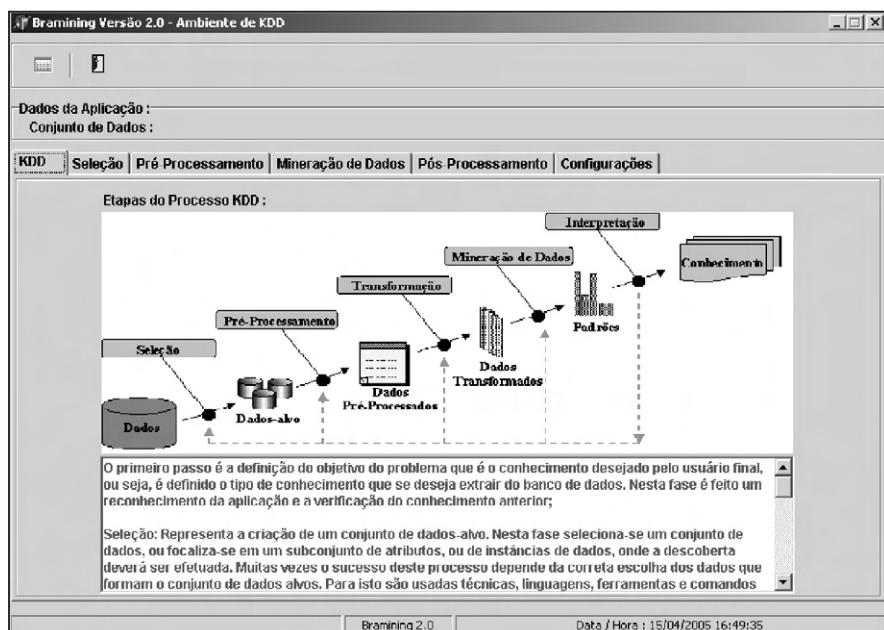
Esta versão do *Bramining* foi implementada na linguagem *Delphi* e acessa o banco de dados *Interbase*. O processo de instalação pode ser iniciado a partir do acionamento do programa *Setup* disponível no CD.

### ATENÇÃO! REQUISITO

Para usar o *Bramining* é necessário ter instalado o *Interbase*.

## Manual do Usuário

### Tela Principal. KDD

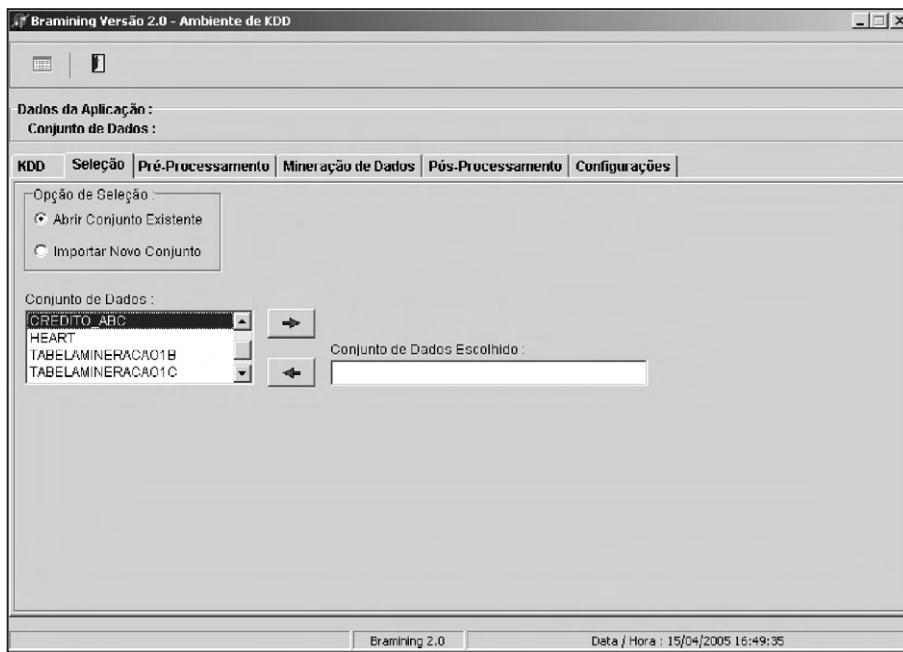


Uma vez instalado e acionado o *Bramining*, o usuário tem acesso à tela principal da ferramenta, na opção KDD. Nesta opção, há uma figura e um texto explicativo sobre o processo de KDD.

## Principais Funções Disponíveis

- Botão para fechamento do sistema.

## Tela Principal. Seleção. Abrir Conjunto Existente

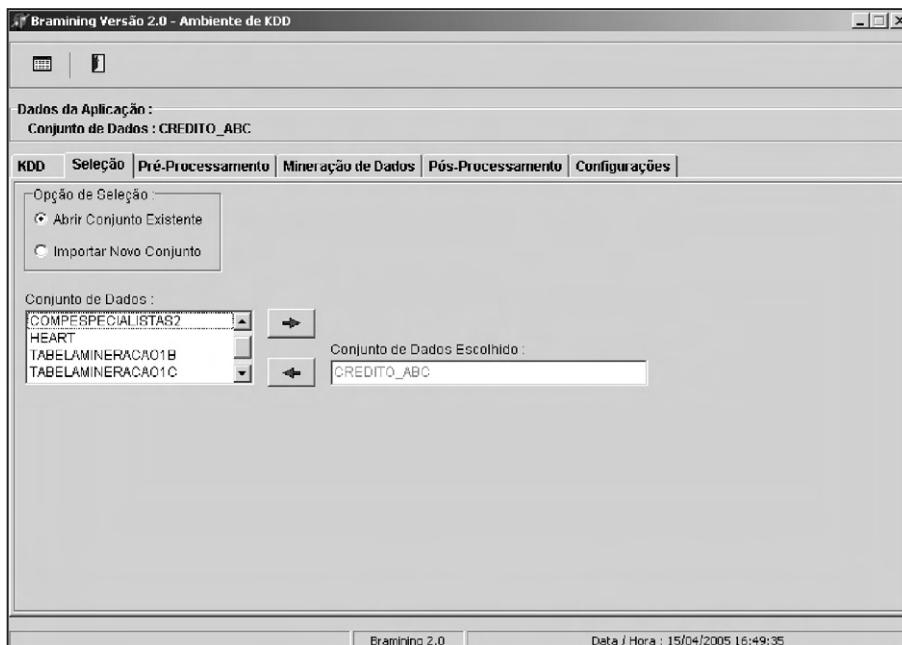


Nesta tela, o usuário deve selecionar o conjunto de dados desejado. Pode, para tanto, abrir um conjunto de dados existente já importado para o ambiente do Bramining. Considere, como exemplo, que o conjunto selecionado tenha sido o “CRÉDITO\_ABC”, conforme mostra a tela a seguir.

### Principais Funções Disponíveis

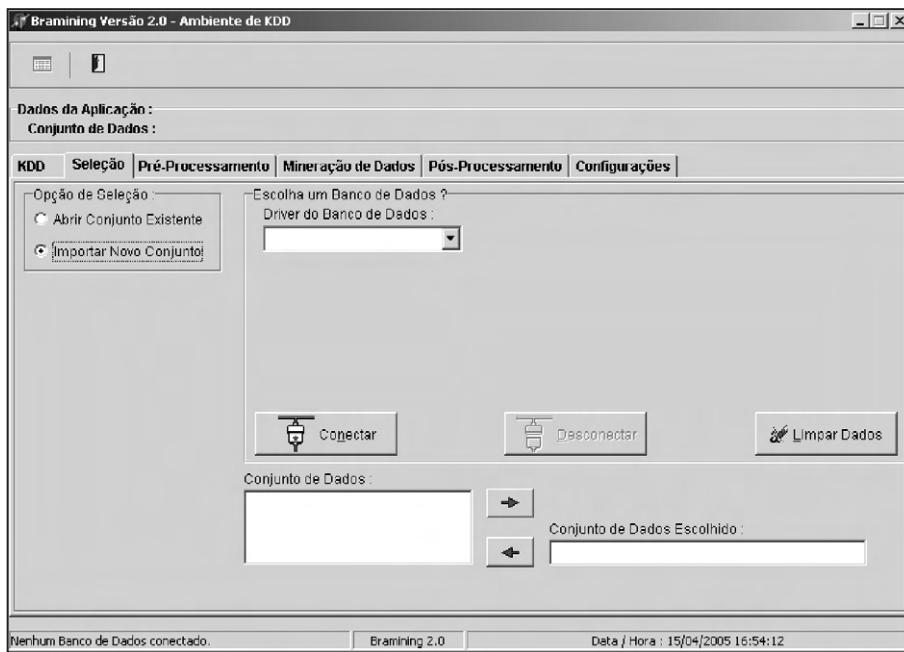
- É utilizado para escolher o conjunto de dados assinalado.
- É utilizado para liberar o conjunto de dados selecionado, podendo ser selecionado um outro conjunto.

## **Tela Principal. Seleção. Abrir Conjunto Existente**



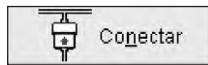
Tela que apresenta os conjuntos de dados existentes na base de dados do Bramining para serem selecionados. A seleção de um conjunto de dados pode ser feita de duas formas: a) com um duplo clique sobre o conjunto desejado; ou b) com um clique sobre o conjunto desejado, pressionando, em seguida, a seta correspondente. O nome do conjunto selecionado deverá aparecer ao lado, no campo “Conjunto de Dados Escolhido:” e no cabeçalho “Dados da Aplicação: Conjunto de Dados:”. Note que o conjunto “CRÉDITO\_ABC” foi selecionado como exemplo. Para selecionar outro conjunto é preciso liberar antes o conjunto que esteja selecionado. Para liberar um conjunto de dados, clique na seta correspondente.

## **Tela Principal. Seleção. Importar Novo Conjunto**

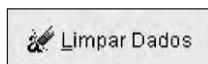


Nesta tela o usuário deve selecionar o tipo de banco de dados a ser importado para o ambiente do *Bramining*. Para isso, clique na caixa “Driver do Banco de Dados”: e selecione uma das opções.

### **Principais Funções Disponíveis**

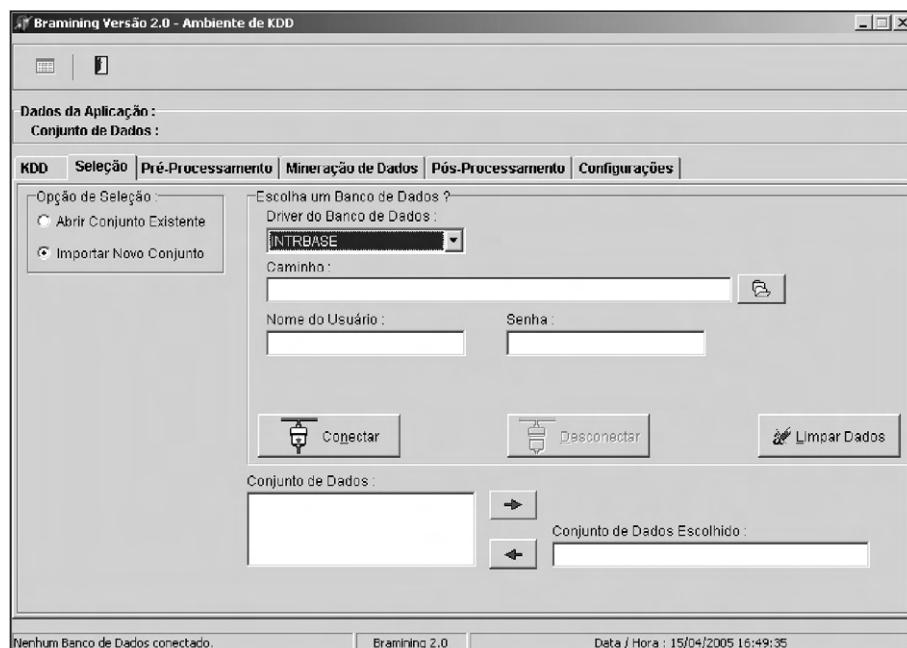


É utilizado para estabelecer a conexão com o tipo de BD selecionado.



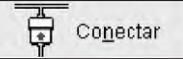
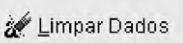
É utilizado para limpar os dados da tela.

## **Tela Principal. Seleção. Importar Novo Conjunto (INTERBASE e MSACCESS)**

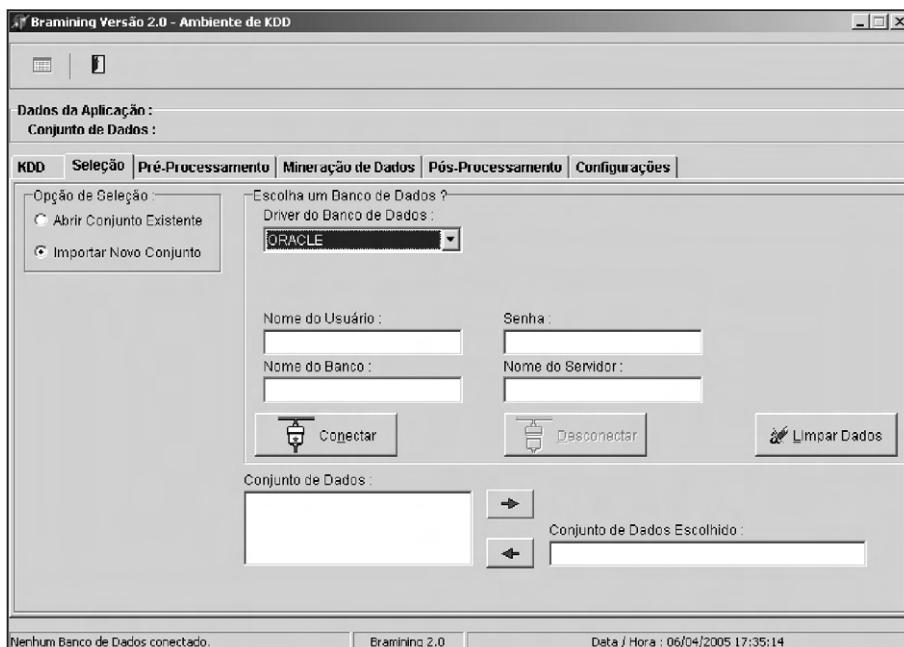


Após escolher o tipo de banco de dados desejado, o usuário deve especificar o caminho onde se encontra o banco de dados, o nome do usuário e a senha para que seja estabelecida a conexão com o banco.

### **Principais Funções Disponíveis**

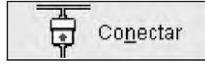
-  É utilizado para escolher arquivo.
-  É utilizado para estabelecer a conexão com o tipo de BD selecionado.
-  É utilizado para limpar os dados da tela.

## **Tela Principal. Seleção. Importar Novo Conjunto (MSSQL e ORACLE)**

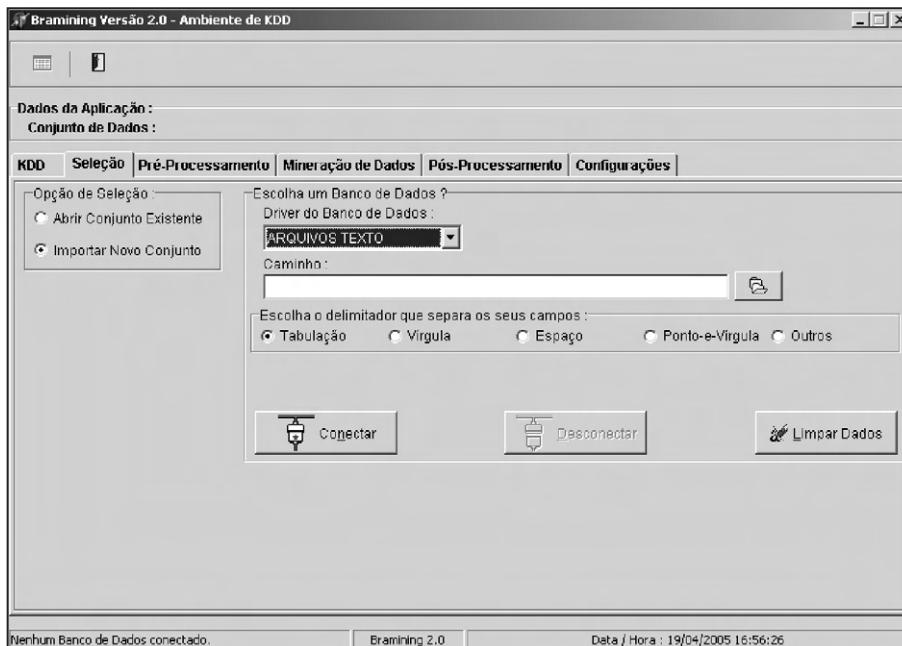


Após escolher o tipo de banco de dados desejado, especifique o nome do usuário, a senha, a identificação do banco e do servidor onde o referido banco se encontra localizado para que seja estabelecida a conexão com o banco de dados.

### **Principais Funções Disponíveis**

-  É utilizado para escolher arquivo.
-  É utilizado para estabelecer a conexão com o tipo de BD selecionado.
-  É utilizado para limpar os dados da tela.

## **Tela Principal. Seleção. Importar Novo Conjunto (ARQUIVOS TEXTO)**

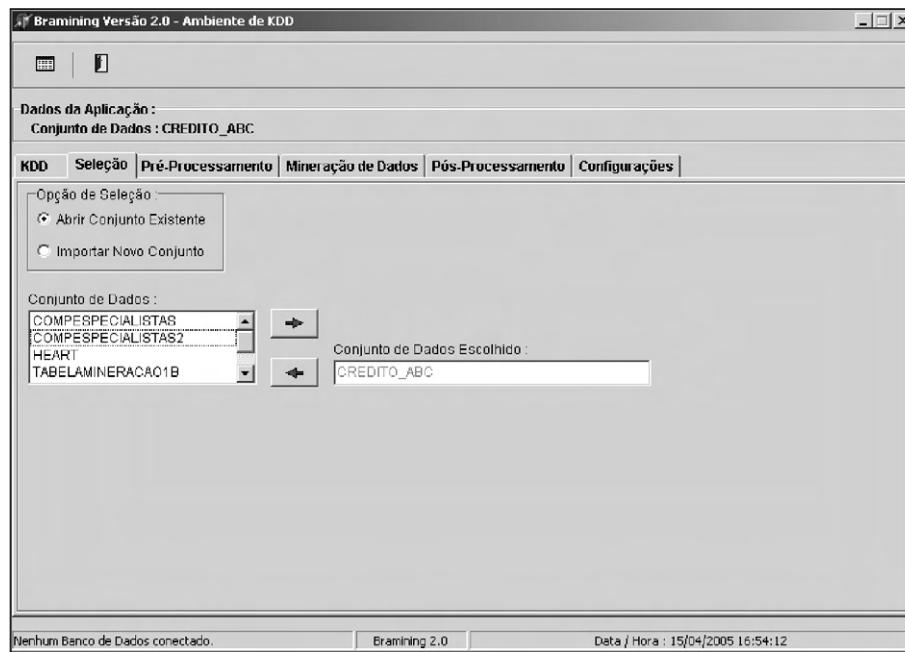


Após escolher o tipo de banco de dados desejado, o usuário deve especificar o caminho onde se encontra o arquivo com o texto desejado e o tipo de delimitador que separa os campos para que seja realizada a importação dos dados.

### **Principais Funções Disponíveis**

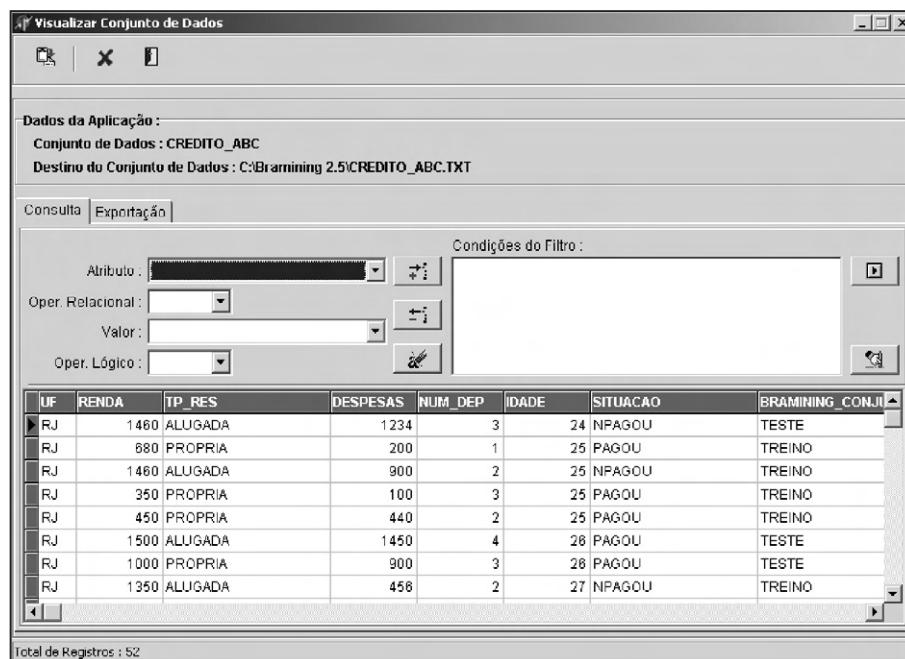
- É utilizado para escolher arquivo.
- É utilizado para estabelecer a conexão com o tipo de BD selecionado.
- É utilizado para limpar os dados da tela.

## Tela Principal. Seleção



Após especificar o conjunto de dados, o usuário pode visualizar o conteúdo deste conjunto, clicando no ícone  , na parte superior, à esquerda da tela.

## Tela Principal. Visualizar Conjunto de Dados (Consulta)



Tela que apresenta o conteúdo do conjunto de dados.

### Principais Funções Disponíveis



É utilizado para adicionar uma condição ao filtro de consulta aos dados do conjunto.



É utilizado para retirar uma condição do filtro de consulta aos dados do conjunto.



É utilizado para cancelar a condição que esteja sendo especificada para inclusão no filtro.



É utilizado para executar o filtro criado pelo usuário.



É utilizado para limpar o filtro criado pelo usuário.



É utilizado para limpar todo o conteúdo da tela, voltando a apresentar todos os dados do conjunto de dados.

## Tela Principal. Visualizar Conjunto de Dados (Consulta)

**Visualizar Conjunto de Dados**

Dados da Aplicação :

Conjunto de Dados : CREDITO\_ABC  
Destino do Conjunto de Dados : C:\Bramining 2.5\CREDITO\_ABC.TXT

Consulta | Exportação |

Condições do Filtro :

Atributo :	SITUACAO	Oper. Relacional :	=	Valor :	NPAGOU	Oper. Lógico :	
------------	----------	--------------------	---	---------	--------	----------------	--

Total de Registros : 52

UF	RENDA	TP_RES	DESPESAS	NUM_DEP	IDADE	SITUACAO	BRAMINING_CONJ
RJ	1460	ALUGADA	1234	3	24	NPAGOU	TESTE
RJ	680	PROPRIA	200	1	25	PAGOU	TREINO
RJ	1460	ALUGADA	900	2	25	NPAGOU	TREINO
RJ	350	PROPRIA	100	3	25	PAGOU	TREINO
RJ	450	PROPRIA	440	2	25	PAGOU	TREINO
RJ	1500	ALUGADA	1450	4	28	PAGOU	TESTE
RJ	1000	PROPRIA	900	3	28	PAGOU	TESTE
RJ	1350	ALUGADA	456	2	27	NPAGOU	TREINO

Nesta tela, o usuário seleciona o atributo, o operador relacional, o valor a ser utilizado e operador lógico para incluir no filtro do conjunto de dados. Suponha que o atributo escolhido seja “SITUAÇÃO”, o valor seja “NPAGOU”, o operador relacional seja “=” e não haja operador lógico.

## **Tela Principal. Visualizar Conjunto de Dados (Após especificação da consulta)**

**Visualizar Conjunto de Dados**

Dados da Aplicação :

Conjunto de Dados : CREDITO\_ABC  
Destino do Conjunto de Dados : C:\Bramining 2.5\CREDITO\_ABC.TXT

Consulta Exportação

Condições do Filtro :

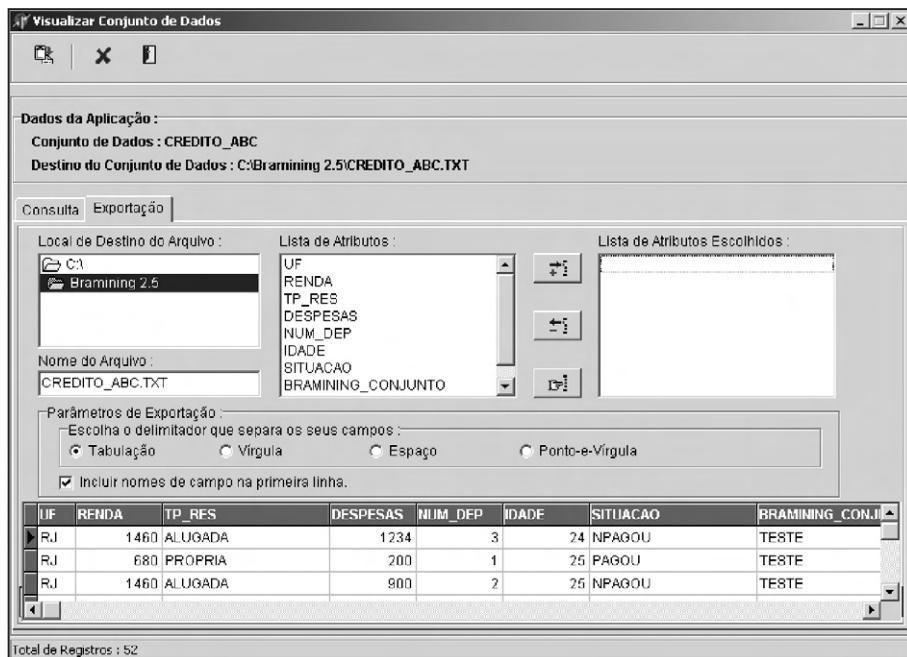
Atributo : UF      Oper. Relacional : =      Valor : SITUACAO = 'NPAGOU'

UF	RENDA	TP_RES	DESPESAS	NUM_DEP	IDADE	SITUACAO	BRAMINING_CONJ
RJ	1460	ALUGADA	1234	3	24	NPAGOU	TESTE
RJ	1460	ALUGADA	900	2	25	NPAGOU	TREINO
RJ	1350	ALUGADA	456	2	27	NPAGOU	TREINO
RJ	1350	ALUGADA	938	1	28	NPAGOU	TESTE
RJ	1200	ALUGADA	100	4	32	NPAGOU	TESTE
RJ	1460	ALUGADA	1200	1	32	NPAGOU	TREINO
RJ	1200	ALUGADA	100	1	32	NPAGOU	TESTE
RJ	1460	ALUGADA	500	0	35	NPAGOU	TREINO

Total de Registros : 12

Após a seleção, o usuário deve adicionar a condição no filtro, para, em seguida, executá-lo. É apresentado como exemplo o conjunto de dados restrito à condição especificada no filtro.

## Tela Principal. Visualizar Conjunto de Dados (Exportação)



Nesta tela, o usuário pode exportar os dados do conjunto de dados para um arquivo texto. Para tanto, o usuário deve escolher o tipo de separador de dados a ser utilizado, os campos desejados, o nome do arquivo a ser gerado e a especificação se os nomes dos campos devem ser incluídos na primeira linha do arquivo.

## **Tela Principal. Pré-Processamento (Após a seleção do conjunto Crédito\_ABC)**

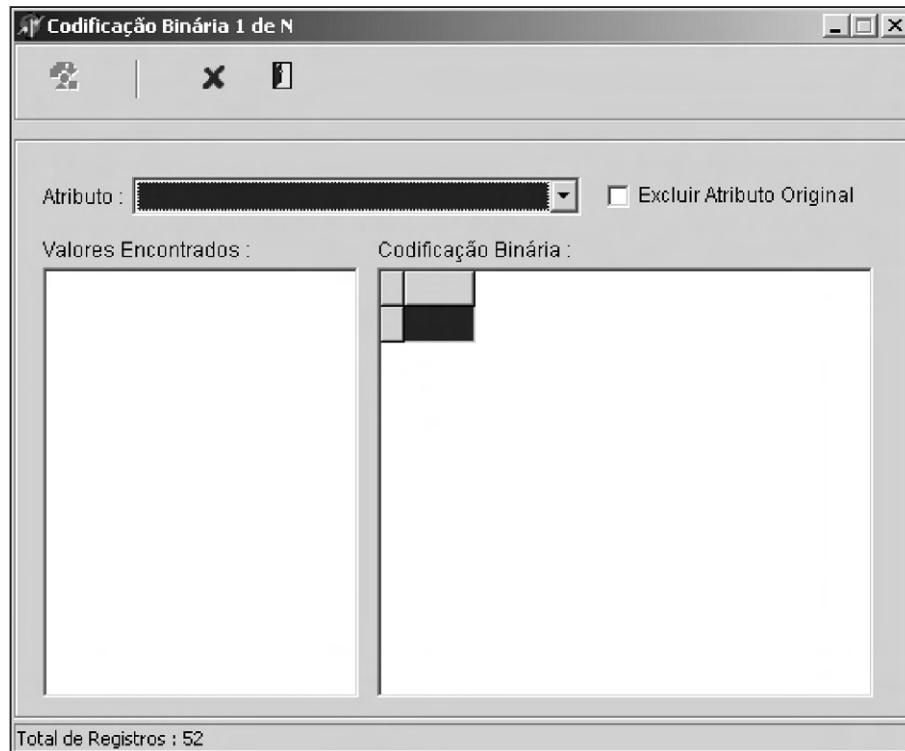


Nesta interface, o usuário pode selecionar a operação de pré-processamento desejada. Ao fazê-lo, são apresentados os métodos disponíveis no *Bramining* que implementam a operação selecionada. Suponha que o método “Codificação Binária 1 de N” tenha sido selecionado. O procedimento é análogo para os demais métodos.

### **Principais Funções Disponíveis**

- |   |  |
|---|--|
|                  | É utilizado para visualizar o conjunto de dados selecionado. |
|  <b>Executar</b> | É utilizado para executar o método selecionado.              |

## Tela Codificação Binária 1 de N

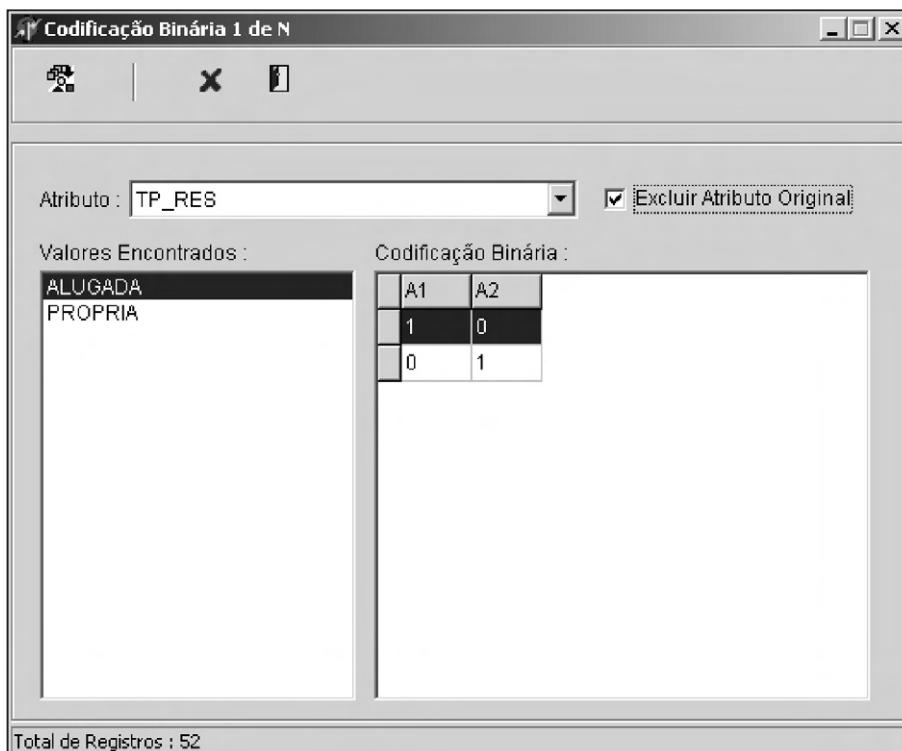


Nesta tela, o usuário escolhe o atributo que deverá sofrer o processo de conversão. Suponha, a título de exemplo, que o atributo escolhido seja “TP\_RES”.

### Principais Funções Disponíveis

- Limpa os dados da interface.
- Fecha a tela, retornando à Tela Principal. Pré-processamento.

### **Tela Codificação Binária 1 de N (Após a seleção do atributo TP\_RES)**



Após a seleção do atributo desejado, para cada valor deste atributo (coluna da esquerda), a codificação binária correspondente (coluna da direita) é apresentada.

#### **Principais Funções Disponíveis**

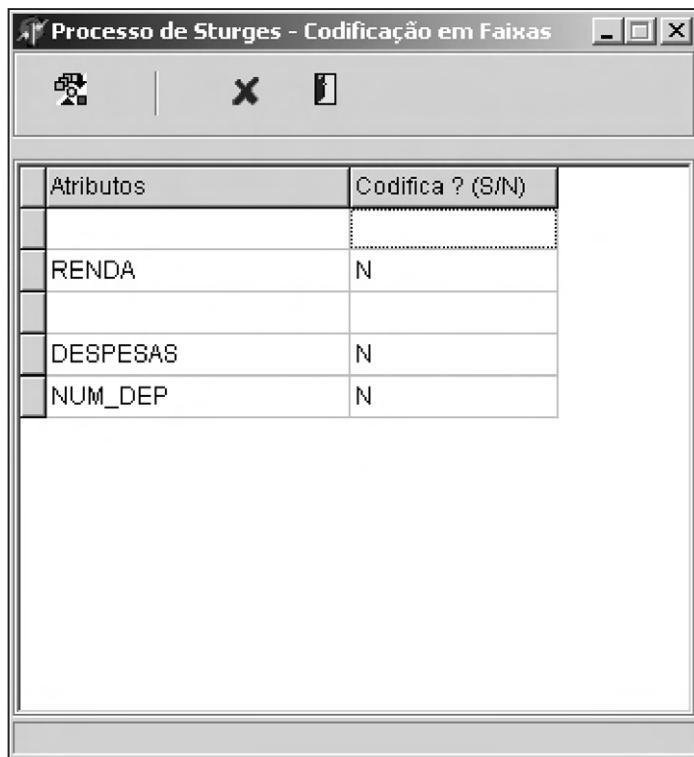
- Aciona o processo de codificação binária 1 de N.
- Limpa os dados da interface.
- Fecha a tela, retornando à Tela Principal. Pré-processamento.

## Tela Principal. Pré-processamento



Suponha que o usuário selecione o método de Discretização de Sturges, na operação de conversão de codificação numérica-categórica. Ao pressionar o botão Executar, a tela próxima tela é apresentada.

## Tela Processo de Discretização de Sturges



Nesta tela, o usuário identifica quais atributos devem ser submetidos ao processo de discretização de Sturges. Para tanto, o usuário deve clicar duas vezes consecutivas sobre o atributo desejado. Neste momento, o valor “N” é alterado para “S”. Para desmarcar um atributo basta repetir o processo sobre o mesmo atributo e o valor “S” é alterado de volta para “N”.

### Principais Funções Disponíveis

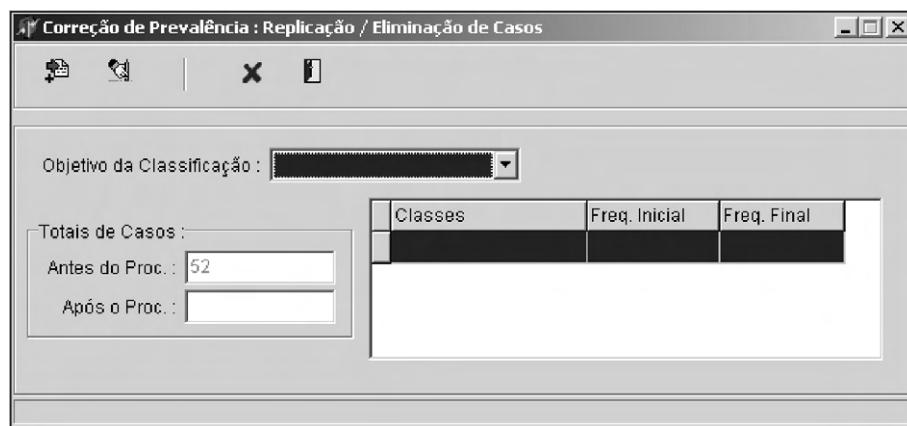
- Aciona o processo de discretização.
- Todos os atributos voltam a ficar assinalados com “N”.
- Fecha a tela, retornando à Tela Principal. Pré-processamento.

## Tela Principal. Pré-processamento



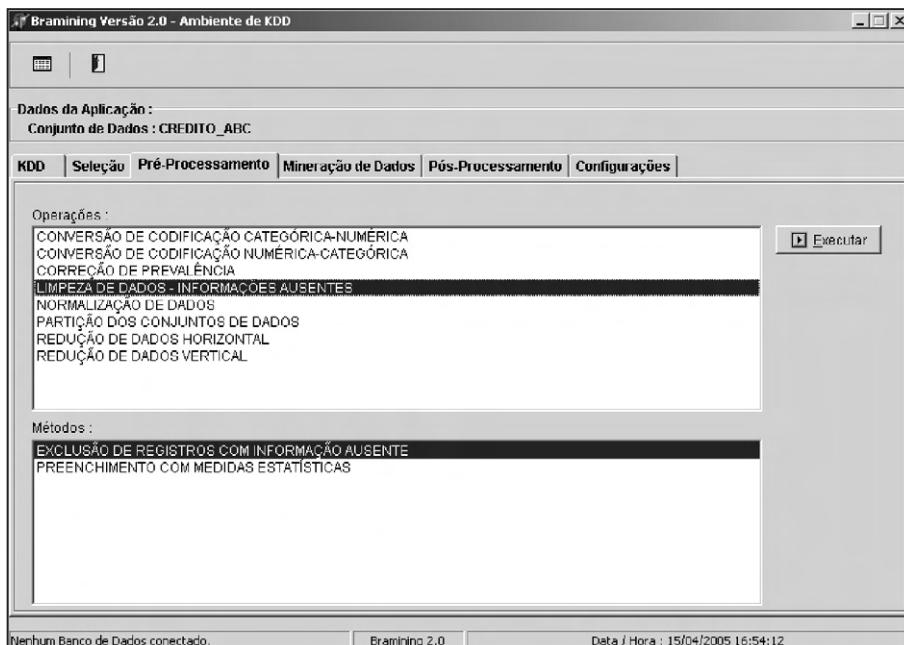
Suponha que o usuário selecione o método “Replicação/Eliminação de Casos”, para a operação de correção de prevalência. Ao pressionar o botão Executar, a tela próxima tela é apresentada.

## Tela Correção de Prevalência: Replicação/Eliminação de Casos



Nesta tela, o usuário deve indicar qual o atributo do conjunto de dados deve ser considerado como objetivo do processo de classificação. Podem ser acionadas duas opções: replicação de casos ou a eliminação de casos. Em ambos os casos, ao final do processamento, o Bramining apresenta o total de casos resultante, bem como os totais de registros em cada classe.

## Tela Principal. Pré-processamento



Suponha que o usuário selecione o método “Exclusão de Registros com Informação Ausente”, para limpeza de dados – informações ausentes. Ao pressionar o botão Executar, a tela próxima tela é apresentada.

## Principais Funções Disponíveis

- É utilizado para visualizar o conjunto de dados selecionado.
- Executar** É utilizado para executar o método selecionado.

## Tela Exclusão de Registros com Informações Ausentes



Nesta tela, o usuário identifica quais atributos devem ser submetidos ao processo de exclusão de registros e sua respectiva quantidade. Para tanto, o usuário deve clicar duas vezes consecutivas sobre o atributo desejado. Neste momento, o valor “N” é alterado para “S”. Para desmarcar um atributo basta repetir o processo sobre o mesmo atributo, o valor “S” é alterado de volta para “N”.

### Principais Funções Disponíveis



É utilizado para excluir registros com informações ausentes.



Todos os atributos voltam a ficar assinalados com “N”.



Fechá a tela, retornando à Tela Principal. Pré-processamento.

## **Tela Preenchimento de Informações Ausentes com Medida Estatística**

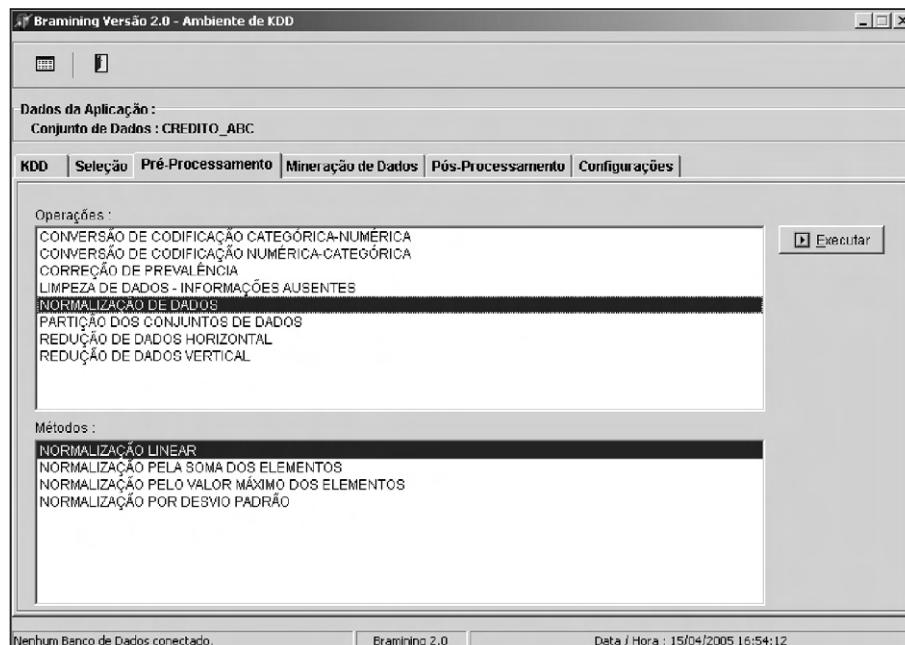


Nesta tela, o usuário identifica quais atributos devem ser submetidos ao processo de preenchimento de informações ausentes e a medida estatística a ser utilizada. Para tanto, o usuário deve clicar duas vezes consecutivas sobre o atributo desejado. Neste momento, o valor “N” é alterado para “S”. Para desmarcar um atributo, basta repetir o processo sobre o mesmo atributo, e o valor “S” é alterado de volta para “N”.

### **Principais Funções Disponíveis**

- [?]** É utilizado para substituir valores ausentes.
- [X]** Todos os atributos voltam a ficar assinalados com “N”.
- [ ]** Fecha a tela, retornando à Tela Principal. Pré-processamento.

## Tela Principal. Pré-processamento

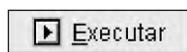


Suponha que o usuário selecione o método “Normalização Linear”, para a operação normalização de dados. Ao pressionar o botão Executar, a tela próxima tela é apresentada.

### Principais Funções Disponíveis

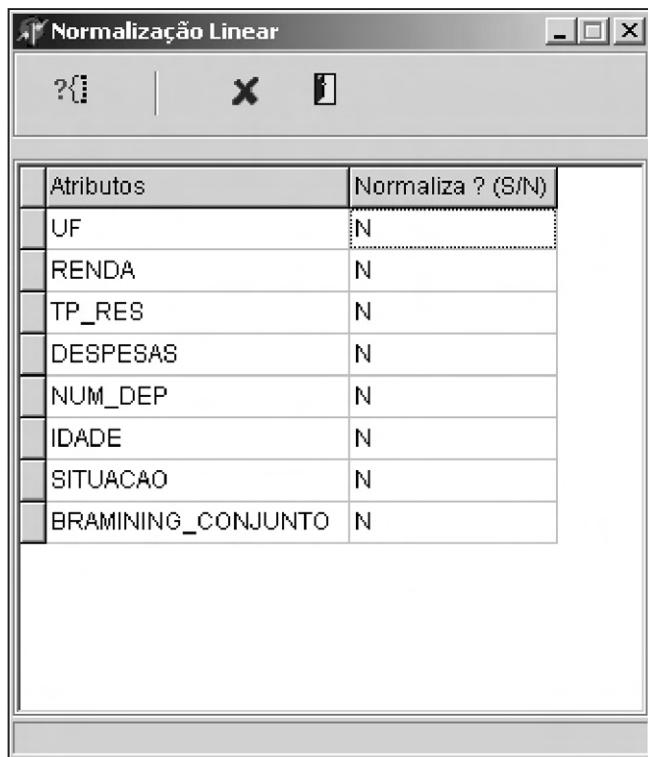


É utilizado para visualizar um conjunto de dados selecionado.



É utilizado para executar o método selecionado.

## Tela de Normalização Linear

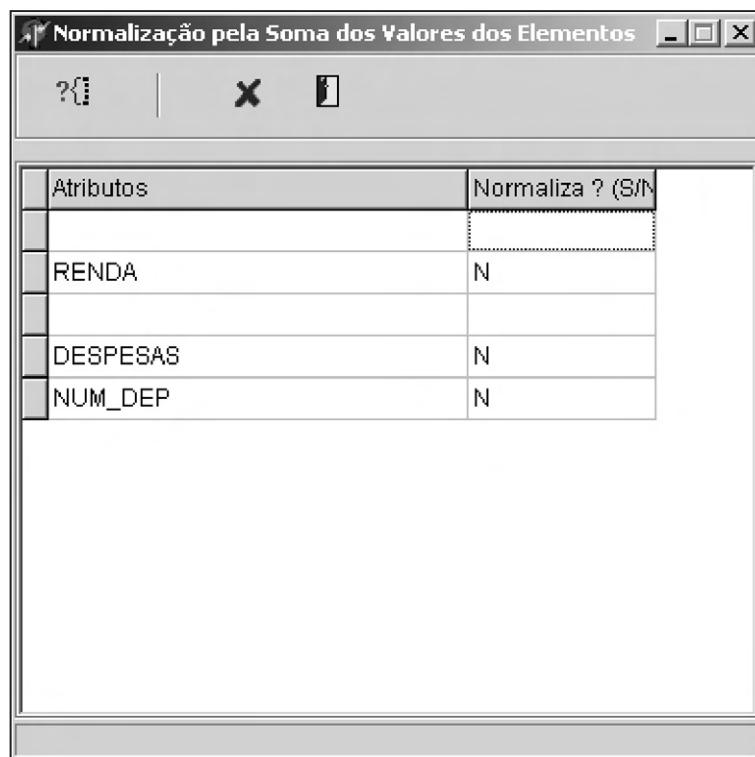


Nesta tela, o usuário identifica quais atributos devem ser submetidos ao processo de normalização linear. Para tanto, o usuário deve clicar duas vezes consecutivas sobre o atributo desejado. Neste momento, o valor “N” é alterado para “S”. Para desmarcar um atributo, basta repetir o processo sobre o mesmo atributo, e o valor “S” é alterado de volta para “N”.

### Principais Funções Disponíveis

- É utilizado para normalizar os dados do conjunto de dados.
- Todos os atributos voltam a ficar assinalados com “N”.
- Fecha a tela, retornando à Tela Principal. Pré-processamento.

## Tela Normalização pela Soma dos Valores dos Elementos

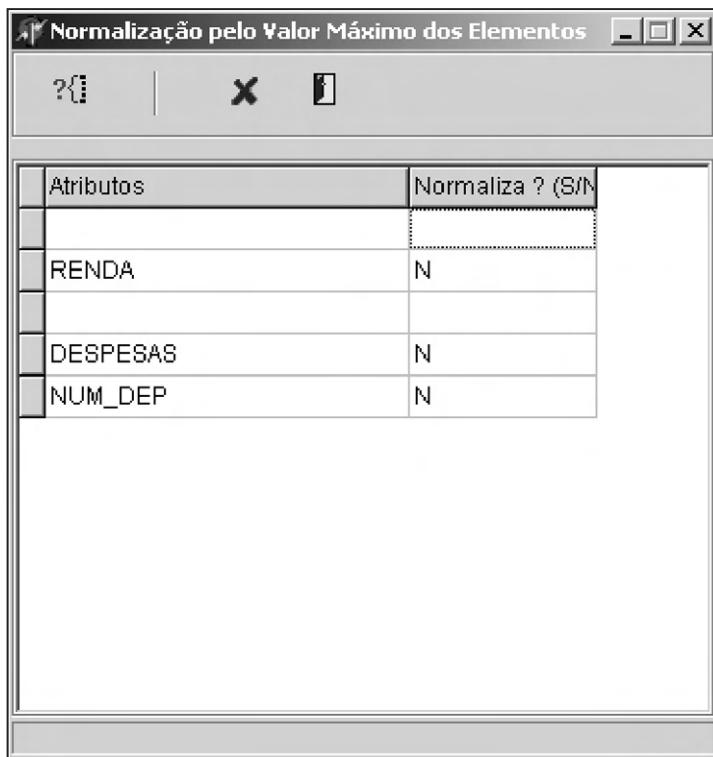


Nesta tela, o usuário identifica quais atributos devem ser submetidos ao processo de normalização pela soma dos valores dos elementos. Para tanto, o usuário deve clicar duas vezes consecutivas sobre o atributo desejado. Neste momento, o valor “N” é alterado para “S”. Para desmarcar um atributo, basta repetir o processo sobre o mesmo atributo, e o valor “S” é alterado de volta para “N”.

### Principais Funções Disponíveis

- É utilizado para normalizar os dados do conjunto de dados.
- Todos os atributos voltam a ficar assinalados com “N”.
- Fecha a tela, retornando à Tela Principal. Pré-processamento.

## Tela Normalização pelo Valor Máximo dos Elementos

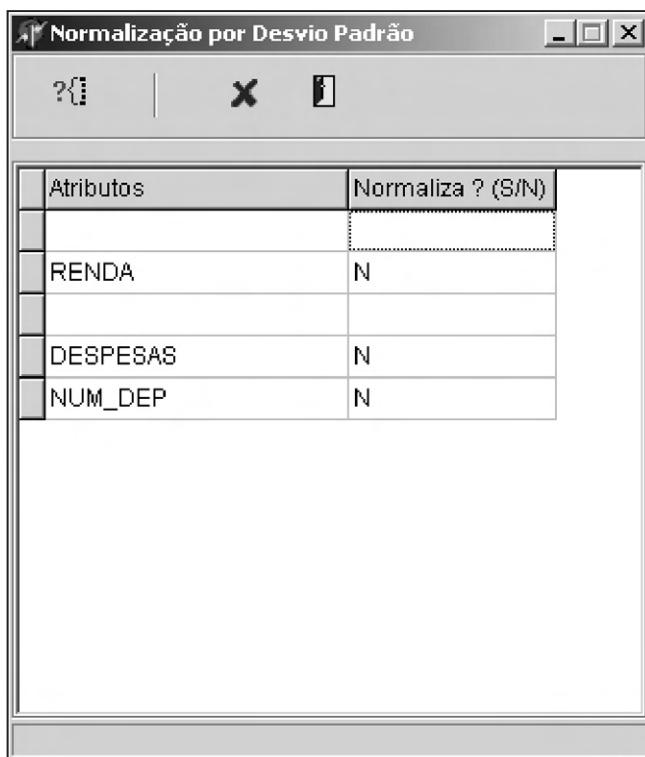


Nesta tela, o usuário identifica quais atributos devem ser submetidos ao processo de normalização pelo valor máximo dos elementos. Para tanto, o usuário deve clicar duas vezes consecutivas sobre o atributo desejado. Neste momento, o valor “N” é alterado para “S”. Para desmarcar um atributo, basta repetir o processo sobre o mesmo atributo, e o valor “S” é alterado de volta para “N”.

### Principais Funções Disponíveis

- É utilizado para normalizar os dados do conjunto de dados.
- Todos os atributos voltam a ficar assinalados com “N”.
- Fecha a tela, retornando à Tela Principal. Pré-processamento.

## Tela Normalização por Desvio Padrão

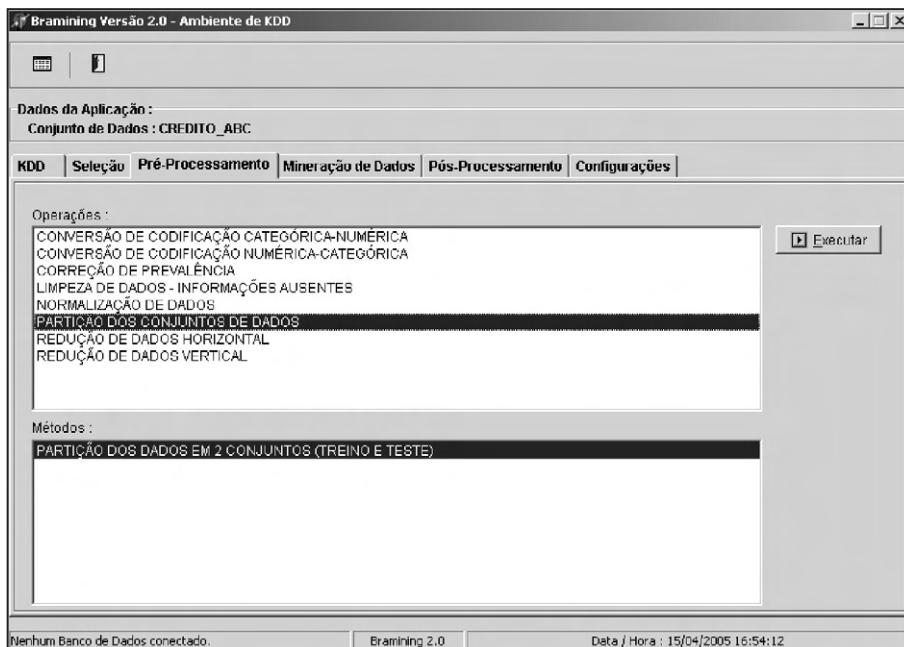


Nesta tela, o usuário identifica quais atributos devem ser submetidos ao processo de normalização por desvio padrão. Para tanto, o usuário deve clicar duas vezes consecutivas sobre o atributo desejado. Neste momento, o valor “N” é alterado para “S”. Para desmarcar um atributo, basta repetir o processo sobre o mesmo atributo, e o valor “S” é alterado de volta para “N”.

### Principais Funções Disponíveis

- É utilizado para normalizar os dados do conjunto de dados.
- Todos os atributos voltam a ficar assinalados com “N”.
- Fecha a tela, retornando à Tela Principal. Pré-processamento.

## Tela Principal. Pré-processamento

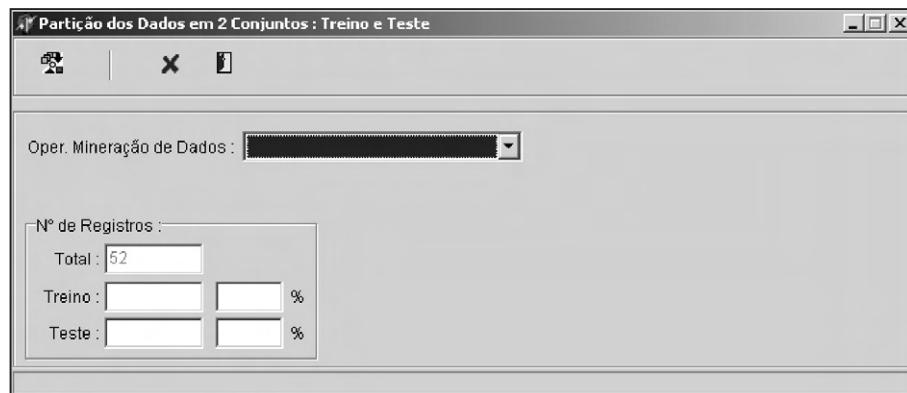


Suponha que o usuário selecione o método “Partição dos Dados em 2 Conjuntos (Treino e Teste)”, na operação de partição dos conjuntos de dados. Ao pressionar o botão Executar, a tela próxima tela é apresentada.

## Principais Funções Disponíveis

-  É utilizado para visualizar um conjunto de dados selecionado.
-  **Executar** É utilizado para executar o método selecionado.

## Tela Partição dos Dados em 2 Conjuntos: Treino e Teste



Nesta tela, o usuário escolhe a operação de Mineração de Dados a ser utilizada e especifica a quantidade ou percentagem dos registros para treino e teste. Suponha, a título de exemplo, que a operação escolhida seja “Classificação”.

### Principais Funções Disponíveis



É utilizado para dividir os dados do conjunto de dados.

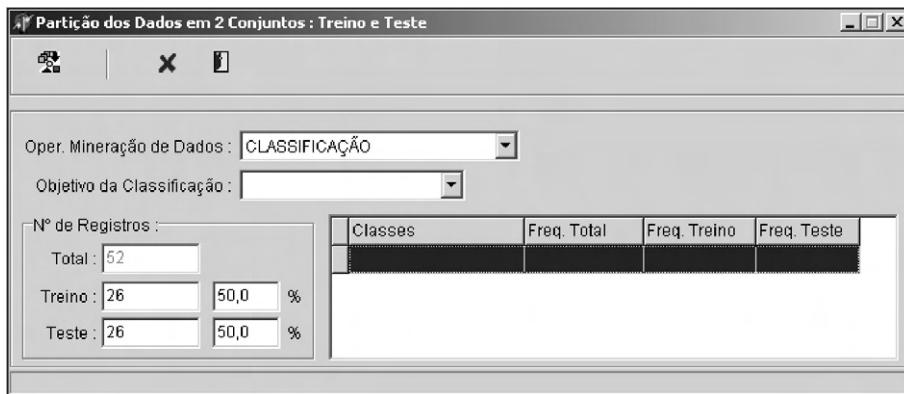


Limpa os valores especificados para os parâmetros do método.



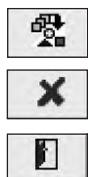
Fechá a tela, retornando à Tela Principal. Pré-processamento.

## **Tela Partição dos Dados em 2 Conjuntos: Treino e Teste (Após a seleção do atributo classificação)**



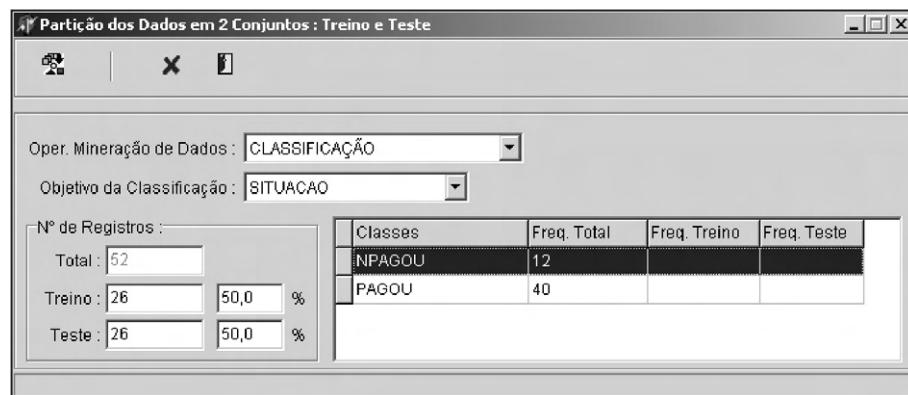
Após ter escolhido o tipo de operação de Mineração de Dados desejada, o usuário deve especificar o objetivo da classificação e o número de registros para treino e para teste. Suponha, a título de exemplo, que o objetivo da classificação escolhido seja “Situação” e que o número de registros para treino e teste seja 26(50%).

### **Principais Funções Disponíveis**



- É utilizado para partitionar os dados do conjunto de dados.
- Cancela a partição dos dados do conjunto escolhido.
- Fecha a tela, retornando à Tela Principal. Pré-processamento.

## **Tela Partição dos Dados em 2 Conjuntos: Treino e Teste (Após a seleção do objetivo da classificação)**



Tela que apresenta os conjuntos de dados referentes ao objetivo da classificação.

### **Principais Funções Disponíveis**



É utilizado para particionar os dados do conjunto de dados.



Cancela a partição dos dados do conjunto escolhido.



Fecha a tela, retornando à Tela Principal. Pré-processamento.

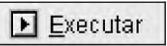
## Tela Principal. Pré-processamento



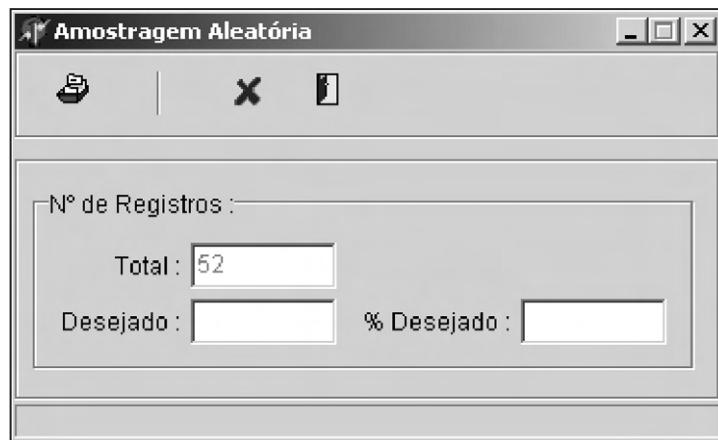
Suponha que o usuário selecione o método “Amostragem Aleatória”, na operação de redução de dados horizontal. Ao pressionar o botão Executar, a tela próxima tela é apresentada.

## Principais Funções Disponíveis

 É utilizado para visualizar um conjunto de dados selecionado.

 **Executar** É utilizado para executar o método selecionado.

## Tela Amostragem Aleatória



Nesta tela, o usuário especifica a quantidade ou percentagem dos registros desejados para amostragem aleatória.

### Principais Funções Disponíveis



É utilizado para processar a amostragem aleatória do conjunto de dados.

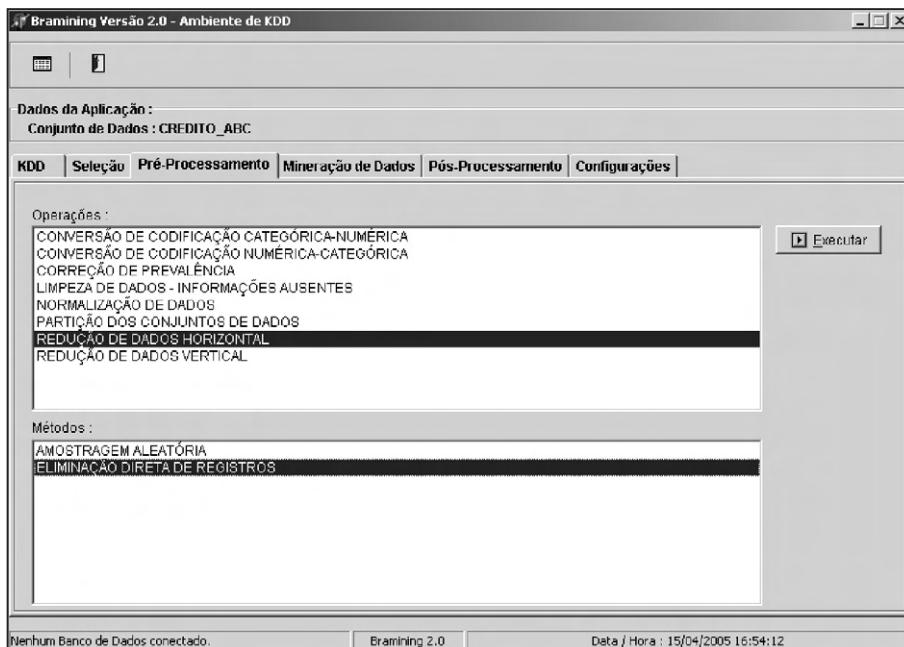


Limpa os parâmetros especificados (valor e porcentagem).



Fecha a tela, retornando à Tela Principal. Pré-processamento.

## Tela Principal. Pré-processamento



Suponha que o usuário selecione o método “Eliminação Direta de Registros”, na operação de redução de dados horizontal. Ao pressionar o botão Executar, a próxima tela é apresentada.

## Principais Funções Disponíveis

É utilizado para visualizar um conjunto de dados selecionado.

**Executar** É utilizado para executar o método selecionado.

## Tela Eliminação Direta de Registros

UF	RENDA	TP_RES	DESPESAS	NUM_DEP	IDADE	SITUACAO	BRAMINING_CON
RJ	1460	ALUGADA	1234	3	24	NPAGOU	TESTE
RJ	680	PROPRIA	200	1	25	PAGOU	TREINO
RJ	1460	ALUGADA	900	2	25	NPAGOU	TREINO
RJ	350	PROPRIA	100	3	25	PAGOU	TREINO
RJ	450	PROPRIA	440	2	25	PAGOU	TREINO
RJ	1500	ALUGADA	1450	4	28	PAGOU	TESTE
RJ	1000	PROPRIA	900	3	26	PAGOU	TESTE
RJ	1350	ALUGADA	456	2	27	NPAGOU	TREINO

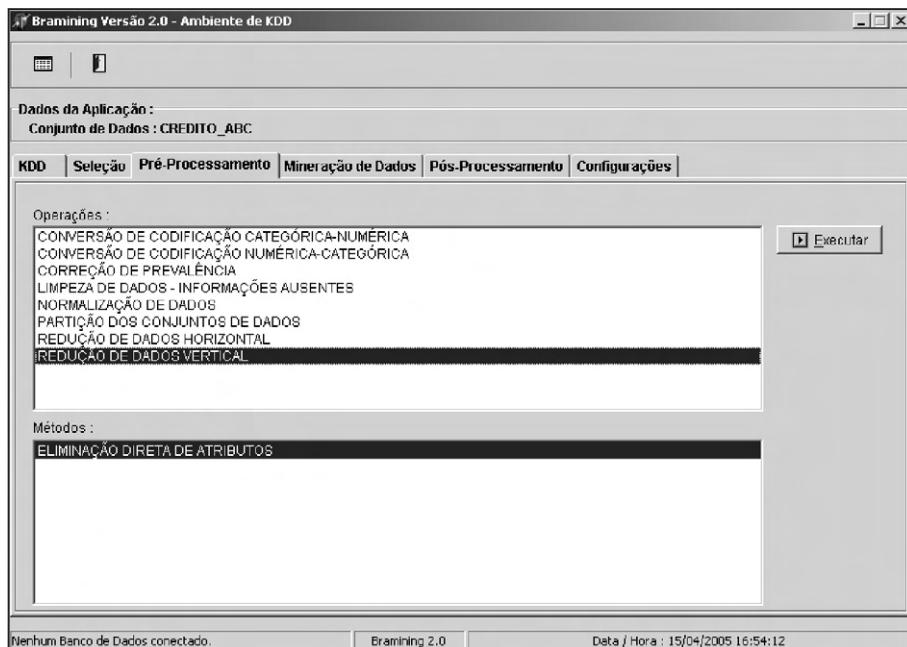
Total de Registros : 52

Tela com funcionamento análogo à Tela Principal. Visualizar Conjunto de Dados (Consulta), apresentada anteriormente.

### Principais Funções Disponíveis

- É utilizado para adicionar uma condição ao filtro de consulta aos dados do conjunto.
- É utilizado para retirar uma condição do filtro de consulta aos dados do conjunto.
- É utilizado para cancelar a condição que esteja sendo especificada para inclusão no filtro.
- É utilizado para executar o filtro criado pelo usuário.
- É utilizado para limpar o filtro criado pelo usuário.
- É utilizado para limpar todo o conteúdo da tela, voltando a apresentar todos os dados do conjunto de dados.

## Tela Principal. Pré-processamento

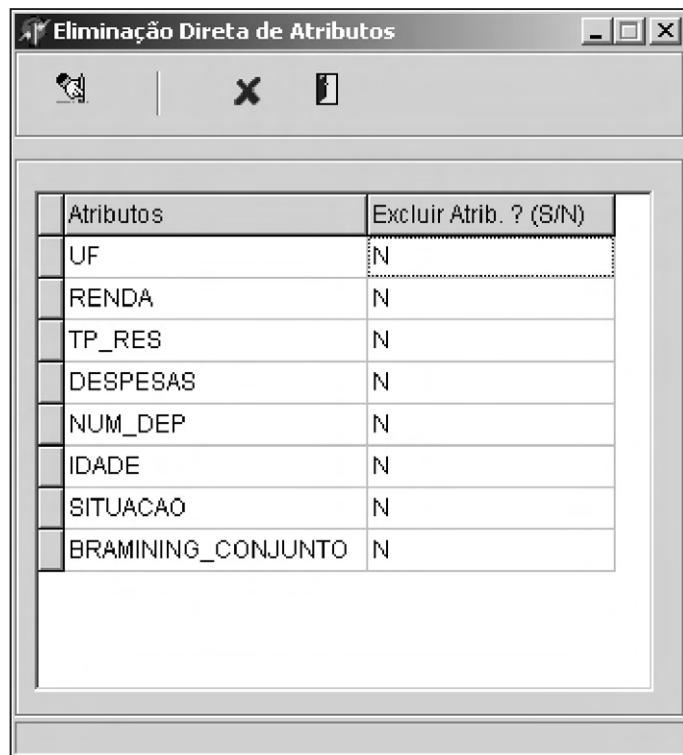


Suponha que o usuário selecione o método “Eliminação Direta de Atributos”, na operação de redução de dados vertical. Ao pressionar o botão Executar, a próxima tela é apresentada.

## Principais Funções Disponíveis

- É utilizado para visualizar um conjunto de dados selecionado.
- Executar** É utilizado para executar o método selecionado.

## Tela Eliminação Direta de Atributos



Nesta tela, o usuário identifica quais atributos devem ser eliminados do conjunto de dados. Para tanto, o usuário deve clicar duas vezes consecutivas sobre o atributo desejado. Neste momento, o valor “N” é alterado para “S”. Para desmarcar um atributo, basta repetir o processo sobre o mesmo atributo, e o valor “S” é alterado de volta para “N”.

### Principais Funções Disponíveis



É utilizado para excluir os atributos do conjunto de dados.

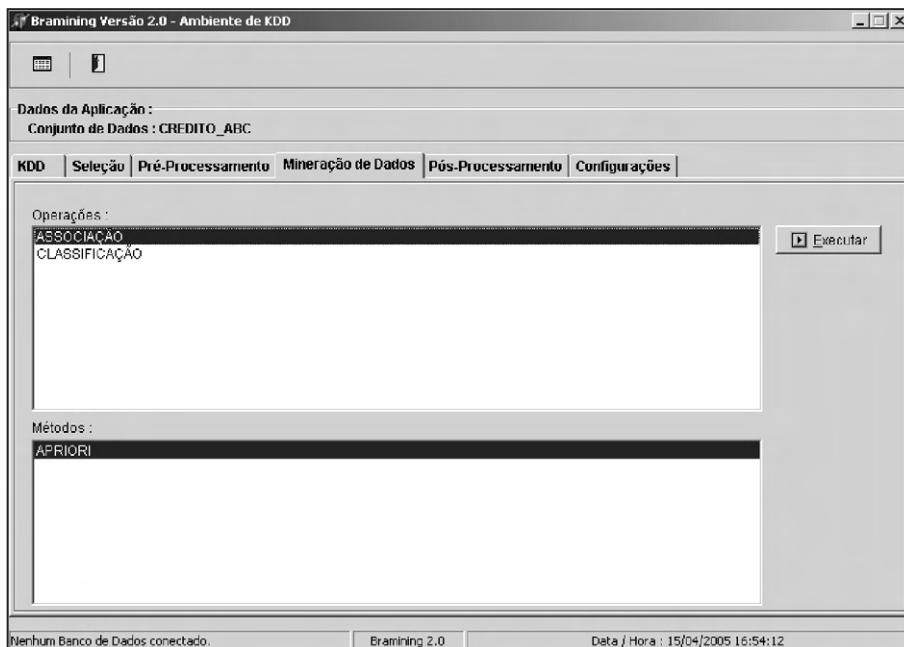


Todos os atributos voltam a ficar assinalados com “N”.



Fechá a tela, retornando à Tela Principal. Pré-processamento.

## Tela Principal. Mineração de Dados

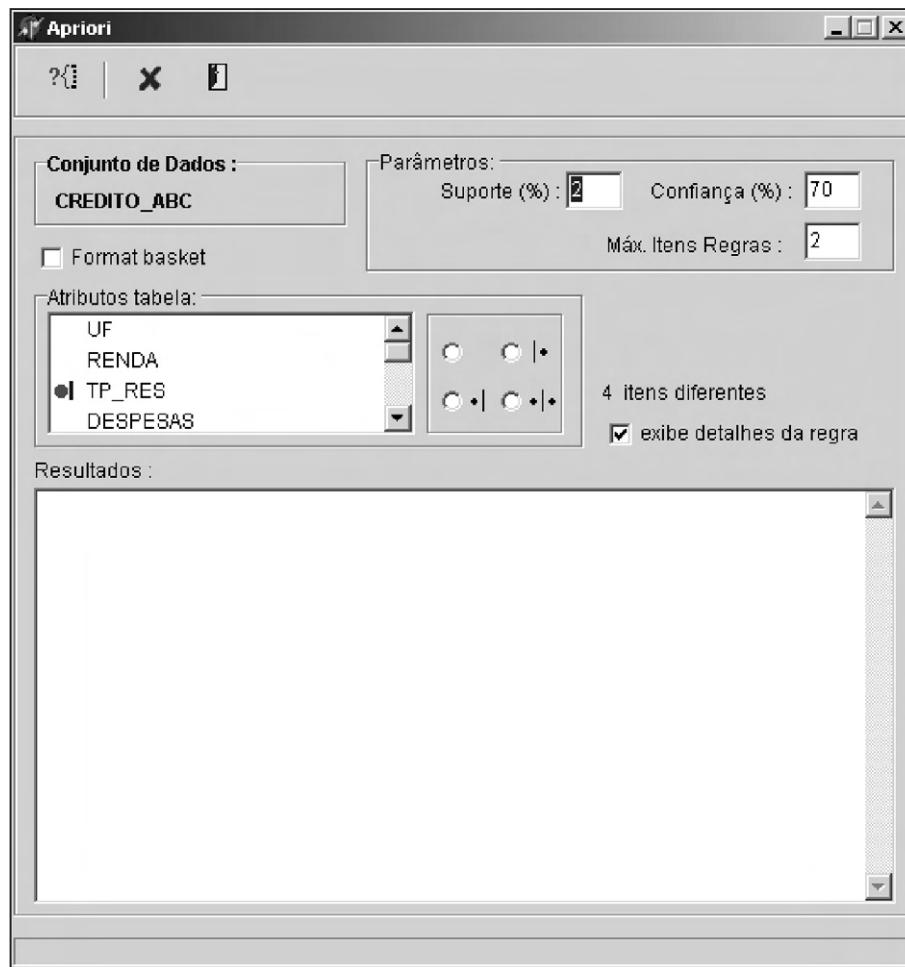


Nesta interface, o usuário pode selecionar a operação de Mineração de Dados. Ao fazê-lo, são apresentados os métodos disponíveis no *Bramining* que implementam a operação selecionada. Suponha que o método “Apriori” tenha sido selecionado. O procedimento é análogo para os demais métodos.

### Principais Funções Disponíveis

-  É utilizado para visualizar um conjunto de dados selecionado.
-  **Executar** É utilizado para executar o método selecionado.

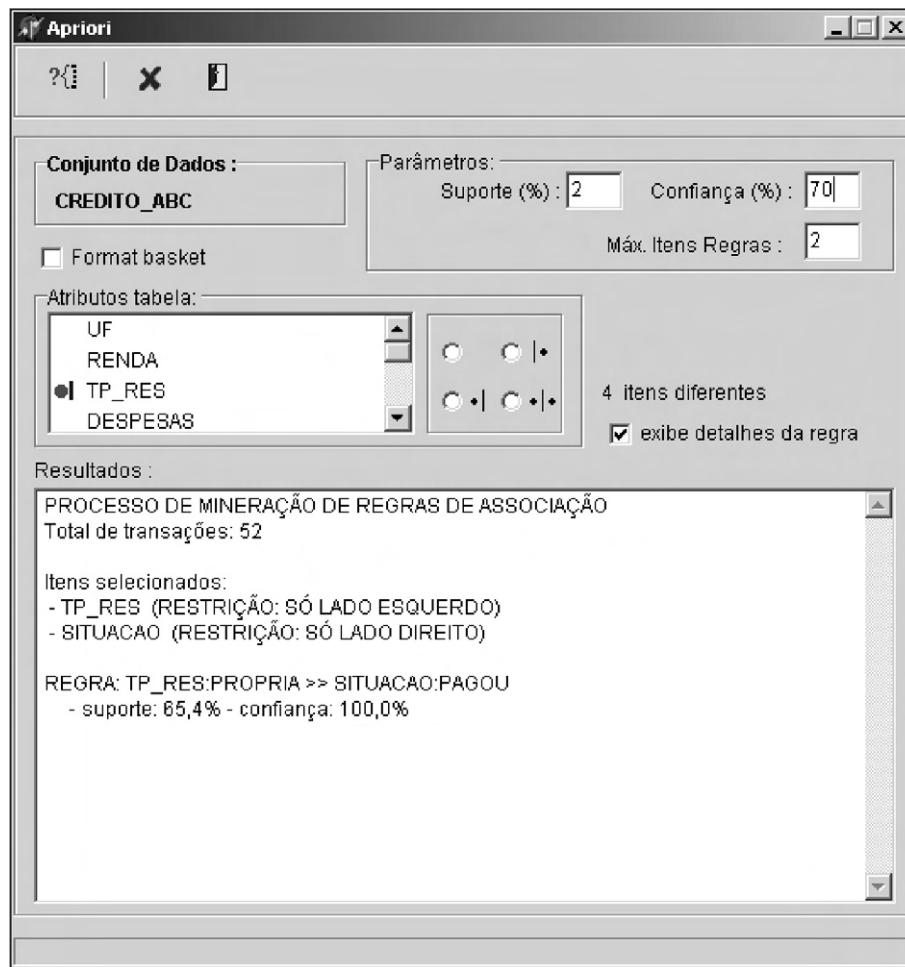
## Tela Principal. Mineração de Dados. Apriori



Nesta tela, o usuário seleciona os parâmetros (vide detalhes sobre o algoritmo Apriori no Capítulo 5), especificando o número máximo de itens.

O usuário seleciona os atributos, especificando em que lado da regra cada atributo da regra deve aparecer: antecedente e/ou conseqüente. Suponha, a título de exemplo, que o atributo escolhido para o antecedente seja “Tp\_Res” e para o conseqüente da regra seja “Situação”.

## **Tela Principal. Mineração de Dados. Apriori (Após a seleção dos atributos)**



Após a execução do algoritmo, o resultado é apresentado.

### **Principais Funções Disponíveis**

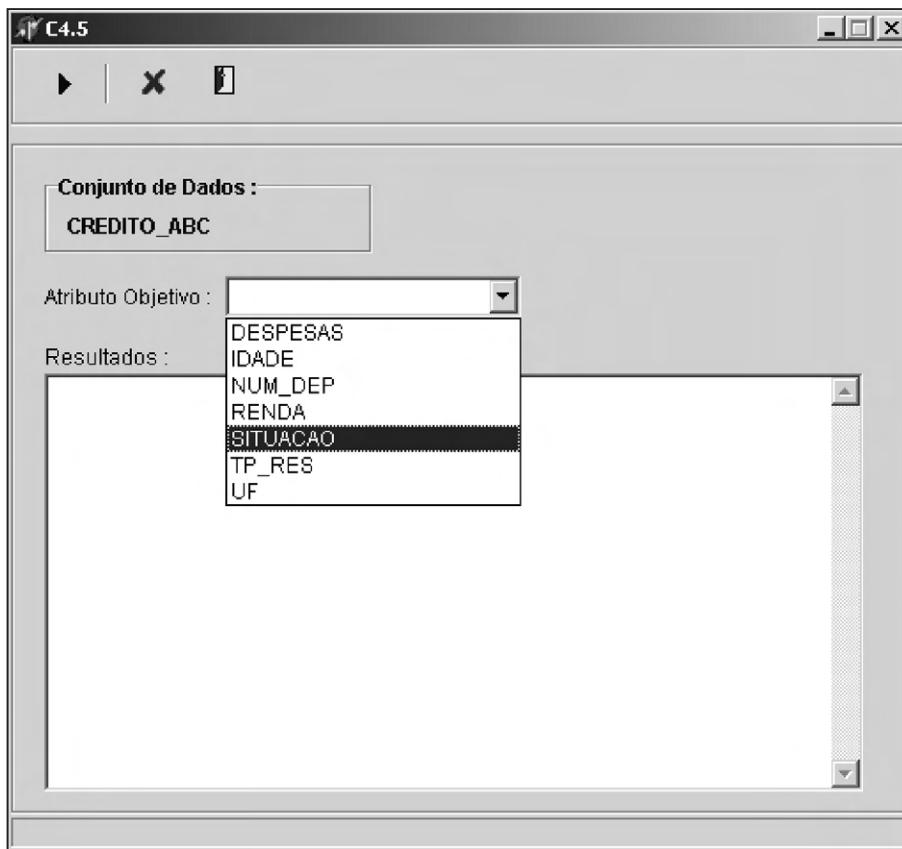
- É utilizado para acionar o algoritmo Apriori.
- Limpa os valores especificados para os parâmetros do algoritmo.
- Fecha a tela, retornando à Tela Principal. Mineração de Dados.

## Tela Principal. Mineração de Dados



Nesta interface, o usuário pode selecionar a operação de Mineração de Dados. Ao fazê-lo, são apresentados os métodos disponíveis no *Bramining* que implementam a operação selecionada. Suponha que o método “C4.5” tenha sido selecionado. O procedimento é análogo para os demais métodos.

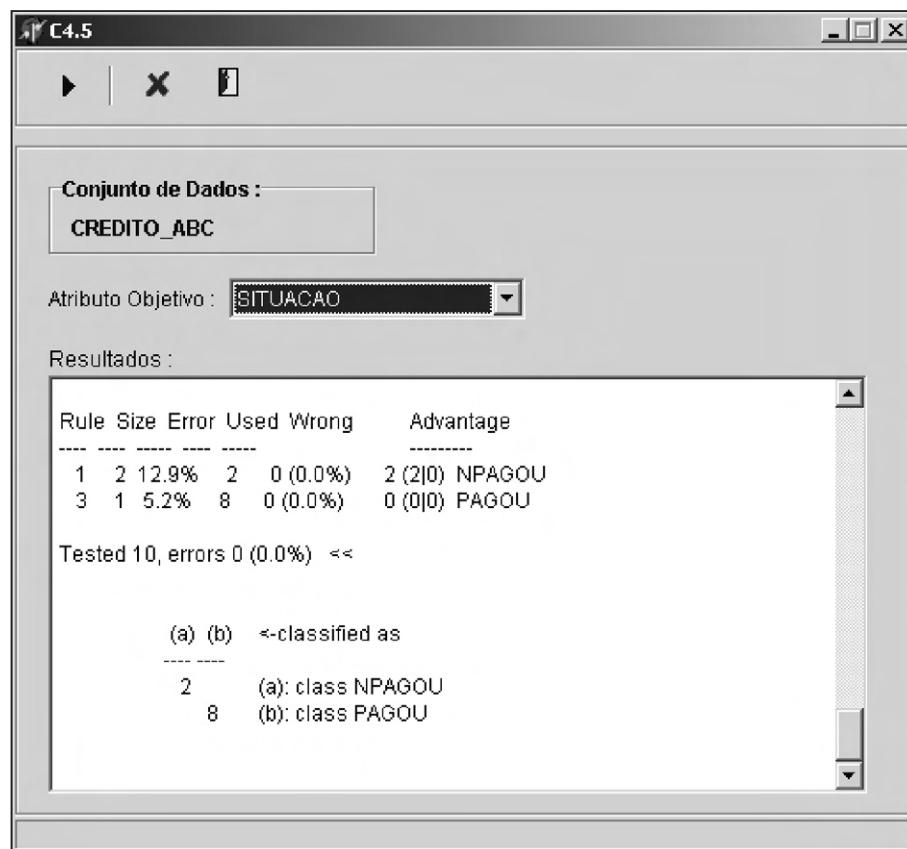
### Tela Principal. Mineração de Dados. Tela C4.5



Nesta tela, o usuário seleciona o atributo objetivo ser utilizado. Suponha, a título de exemplo, que o atributo escolhido seja “Situação”.

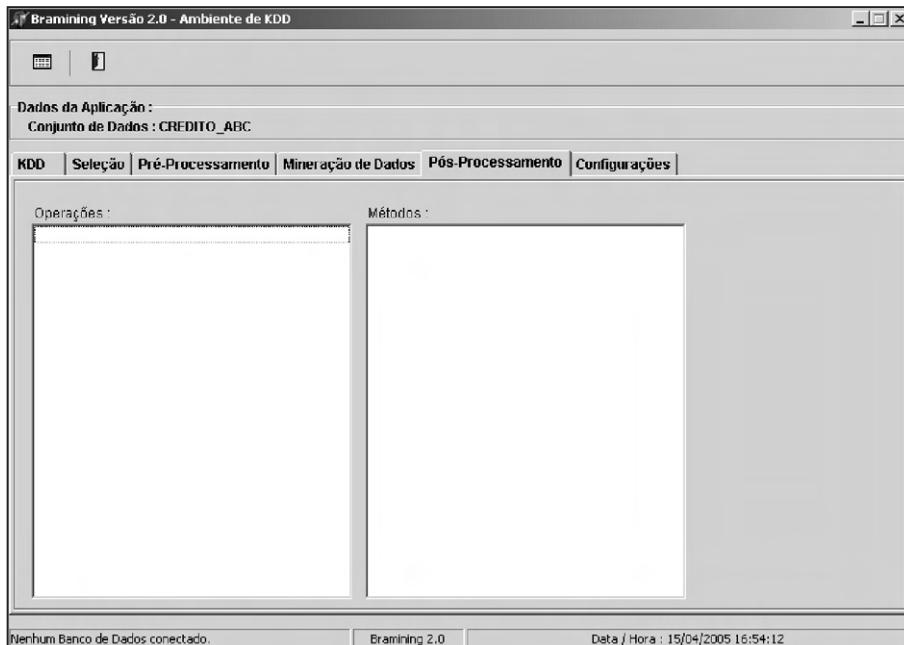
#### Principais Funções Disponíveis

- É utilizado para executar o método C4.5.
- Limpa os valores especificados para os parâmetros do algoritmo.
- Fecha a tela, retornando à Tela Principal. Mineração de Dados.

**Tela Principal. Mineração de Dados. C4.5 (Após a seleção do atributo objetivo "Situação")**

Após a execução do método C4.5, os resultados são apresentados.

## Tela Principal. Pós-processamento



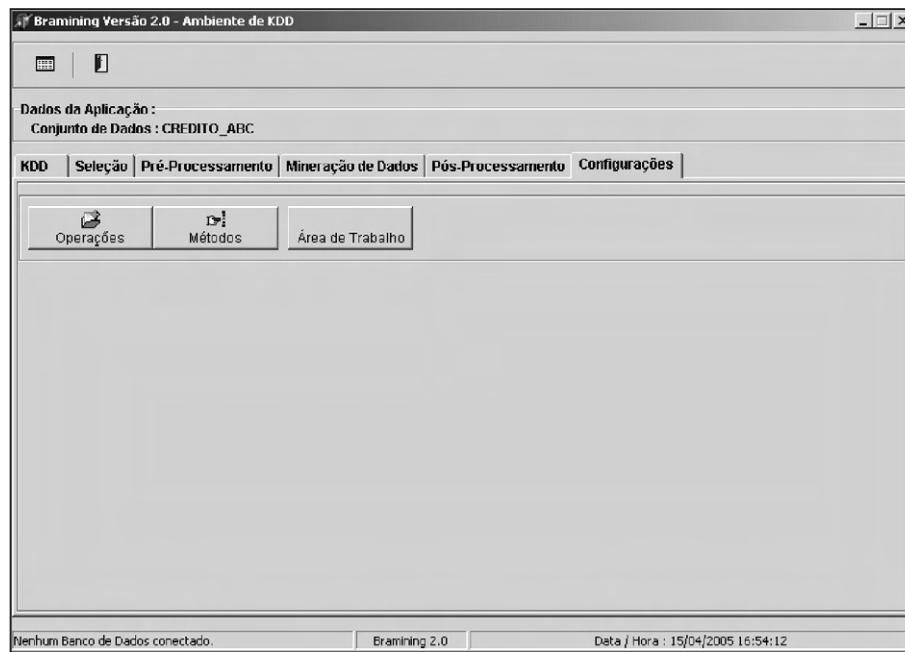
Nesta versão de demonstração do Bramining não foram incluídos métodos de pós-processamento.

## Principais Funções Disponíveis



É utilizado para visualizar um conjunto de dados.

## Tela Principal. Configurações



Nesta interface, o usuário pode configurar o cadastro de operações e o cadastro de métodos incorporados ao Bramining.

### Principais Funções Disponíveis



É utilizado para visualizar um conjunto de dados.



**Operações** É utilizado para acionar a tela de manutenção do cadastro de operações de KDD incorporadas ao Bramining.



**Métodos** É utilizado para acionar a tela de manutenção do cadastro de métodos de KDD incorporados ao Bramining.

**Área de Trabalho** Não disponível nesta versão.

## Tela Principal. Configurações. Cadastro de Operações

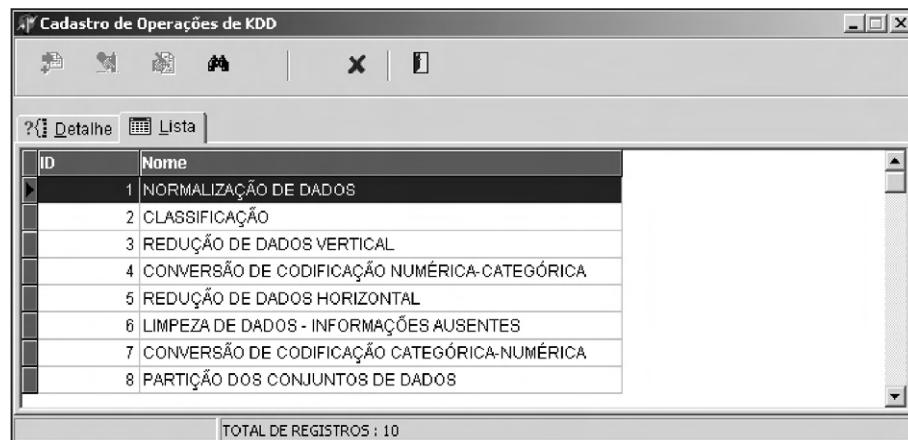


Nesta interface, o usuário pode incluir, excluir, atualizar e consultar os dados das operações disponíveis no Bramining. Para incluir uma nova operação, o usuário deve selecionar a etapa de KDD e preencher os campos de cadastro.

### Principais Funções Disponíveis

-  É utilizado para inserir uma nova operação no sistema.
-  É utilizado para eliminar a operação selecionada.
-  É utilizado para alterar os dados da operação selecionada.
-  É utilizado para consultar os dados das operações do sistema.
-  É utilizado para limpar a tela.
-  É utilizado para fechar a tela corrente.

## **Tela Principal. Configurações. Cadastro de Operações (Após inserção de registro)**

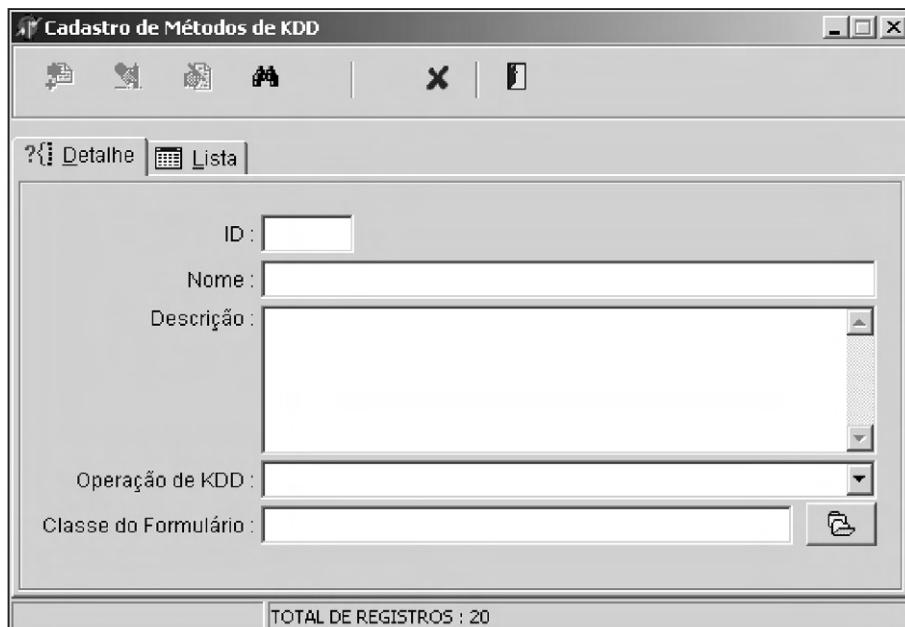


Apresenta a lista de operações disponíveis no Bramining.

### **Principais Funções Disponíveis**

- É utilizado para inserir uma nova operação no sistema.
- É utilizado para eliminar a operação selecionada.
- É utilizado para alterar os dados da operação selecionada.
- É utilizado para consultar os dados das operações do sistema.
- É utilizado para limpar a tela.
- É utilizado para fechar a tela corrente.

## **Tela Principal. Configurações. Cadastro de Métodos de KDD**



Nesta interface, o usuário pode incluir, excluir, atualizar e consultar os dados dos métodos disponíveis no Bramining. Para incluir um novo método, o usuário deve selecionar a operação de KDD à qual o método pertence e preencher os campos de cadastro.

### **Principais Funções Disponíveis**

- É utilizado para inserir um novo método no sistema.
- É utilizado para eliminar o método selecionado.
- É utilizado para alterar os dados do método selecionado.
- É utilizado para consultar os dados dos métodos do sistema.
- É utilizado para escolher o formulário do método a ser incorporado ao sistema.
- É utilizado para fechar a tela corrente.

## **Tela Principal. Configurações. Cadastro de Métodos de KDD (Após a inserção de registro)**



Apresenta a lista de métodos disponíveis no Bramining.

### **Principais Funções Disponíveis**

- É utilizado para inserir um novo método no sistema.
- É utilizado para eliminar o método selecionado.
- É utilizado para alterar os dados do método selecionado.
- É utilizado para consultar os dados dos métodos do sistema.
- É utilizado para limpar a tela.
- É utilizado para fechar a tela corrente.

# Referências Bibliográficas

- ADEGBOYE, A.; COELHO, C. F.; CHAVES, A. L. *Assistentes Inteligentes na Utilização de Algoritmos de Mineração de Dados*. Trabalho de Graduação, Universidade Gama Filho. Rio de Janeiro, 2002.
- AGRAWAL, R.; IMIELINSKI, T.; SWAMI, A. *Mining Association Rules Between Sets of Items in Large Databases*. ACM SIGMOD Conference Management of Data, 1993.
- AHA, D. *Generalizing from Case Studies: A Case Study*. Proceedings of the Nieth International Workshop on Machine Learning. São Francisco: Morgan Kaufmann, 1992.
- AMANT, R. S.; COHEN, P. R. *Evaluation of a Semi-Autonomous Assistant for Exploratory Data Analysis*. Proceedings of the First International Conference on Autonomous Agents. Marina Del Rey: ACM Press, 1997b.
- AMANT, R. S.; COHEN, P. R. Intelligent Support for Exploratory Data Analysis. *Journal of Computational and Graphical Statistics*, v. 7, n. 4, 1998.
- AMANT, R. S.; COHEN, P. R. *Interaction with a Mixed-Initiative System for Exploratory Data Analysis*. Proceedings of the Third International Conference on Intelligent User Interfaces. Orlando: ACM Press, 1997a.
- BENSUSAN, H. *Automatic Bias Learning: an inquiry into inductive basis of induction*. 1999. Ph.D. Thesis – School of Cognitive and Computing Sciences, University of Sussex.

- BENSUSAN, H.; GIRAUD-CARRIER, C.; KENNEDY, C. J. *A High-order Approach to Meta-learning*. Eleventh European Conference on Machine Learning, Workshop on Meta-Learning, Building Automatic Advice Strategies for Model Selection and Method Combination. Barcelona, 2000a.
- BENSUSAN, H.; GIRAUD-CARRIER, C.; PFAHRINGER, B.; SOARES, C.; BRAZDIL, P. *What Works Well tells us What Works Better*. Proceedings of ICML'2000 workshop on What Works Well Where, ICML'2000, 2000b. Disponível em: 03 dez. 2003.
- BERNSTEIN, A.; HILL, S.; PROVOST, F. *Intelligent Assistance for the Data Mining Process*. CeDER Working Paper IS-02-02, Center for Digital Economy Research, Stern School of Business, New York University, 2002.
- BOJADZIEV, G. BOJADZIEV, M.; *Fuzzy Logic for Business, Finance, and Management*. World Scientific, 1997.
- BRACHMAN, R. J.; ANAND, T. *The Process of Knowledge Discovery in Databases*. The KDD Process for Extracting Useful Knowledge from Volumes of Data, p. 37-57, 1996.
- BRAZDIL P. *Data Transformation and Model Selection by Experimentation and Meta-Learning*. Proceedings of ECML-98 Workshop on Upgrading Learning to the Meta-Level: Model Selection and Data Transformation, 1998.
- BRAZDIL P.; SOARES, C. *A Comparison of Ranking Methods for Classification Algorithm Selection*. Proceedings of the Eleventh European Conference on Machine Learning, 2000.
- BRAZDIL, P.; SOARES, C.; COSTA, J. Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results. *Machine Learning*, Kluwer, v. 50, n. 3, 2003.
- BRODLEY, C. *Addressing the Selective Superiority Problem: Automatic Algorithm/Model Class Selection*. Proceedings of the Tenth International Conference on Machine Learning. São Francisco: Morgan Kaufmann, 1993.
- BUCHANAN, B. *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Jose, 2000.
- CARLANTONIO, L. M. *Novas Metodologias para Clusterização de Dados*. Dissertação de Mestrado, Engenharia Civil, COPPE, Universidade Federal do Rio de Janeiro, 2001.
- CARVALHO, L. A. V. *Data Mining: A Mineração de Dados no Marketing, Medicina, Economia, Engenharia e Administração*. 2. ed. São Paulo: Érica, 2001.

- CURRAN, D.; O'RIORDAN, C. *Applying Evolutionary Computation to Designing Neural Networks: a Study of the State of the Art*. Disponível em . Acesso em 19 out. 2002.
- DATE, C. J. *Introdução aos Sistemas de Bancos de Dados*. Campus, 2000.
- DAVIS, L. *Handbook of Genetic Algorithms*. VNR Comp. Library, 1990.
- ENGELS, R. *Planning tasks for knowledge discovery in databases: Performing Task-Oriented User-Guidance*. Proceedings of the International Conference on Knowledge Discovery and Data Mining. Portland: AAAI Press, 1996.
- ENGELS, R.; LINDNER, G.; STUDER, R. *A Guided Tour Through the Data Mining Jungle*. Proceedings of the Third International Conference on Knowledge Discovery in Databases. Newport Beach, 1997.
- ENGELS, R.; THEUSINGER, C. *Using a Data Metric for Preprocessing Advice for Data Mining Applications*. Proceedings of the Thirteenth European Conference on Artificial Intelligence, John Wiley & Sons, 1998.
- ELMASRI, R.; NAVATHE, S. B. *Fundamentals of Database Systems*. Benjamin – Cummins, 1989.
- FAYYAD, U. M.; PIATETSKY-SHAPIRO, G.; SMYTH, P. *From Data Mining to Knowledge Discovery: An Overview*. Knowledge Discovery and Data Mining, Menlo Park: AAAI Press, 1996a.
- FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*, v. 39, 1996b.
- FREITAS, A. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Natural Computing Series, Springer-Verlag Berlin Heidelberg, 2002.
- FRUNKRANZ, J.; PETRAK, J. *An Evaluation of Landmarking Variants*. Proceedings of the Workshop on Integrated Aspects of Data Mining, Decision Support, and Meta-Learning, p. 57-68, 2001.
- GAMA, J.; BRAZDIL, P. *Characterization of Classification Algorithms*. Proceedings of the Seventh Portuguese Conference on Artificial Intelligence, p. 189-200, 1995.
- GODOY, R. *Aplicação de Landmarkers no Processo de Descoberta de Conhecimento em Bases de Dados*. Trabalho de Conclusão de Curso de Graduação, Ciência da Computação, Centro Universitário da Cidade do Rio de Janeiro, 2004. Disponível em [www.univercidade.edu/nupac](http://www.univercidade.edu/nupac).

- GODOY, R.; GOLDSCHMIDT, R.; PASSOS, E. *Mineração de Dados: Aplicação Prática em Pequenas e Médias Empresas*. KM-Rio, 2003.
- GOLDBERG, D. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.
- GOLDSCHMIDT, R. *Assistência Inteligente à Orientação do Processo de Descoberta de Conhecimento em Bases de Dados*. Rio de Janeiro, Tese de Doutorado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro, 2003.
- GOLDSCHMIDT, R.; PASSOS, E. *Utilização de Recursos de Banco de Dados Relacionais em Tarefas de Mineração de Regras Associativas*. I Congresso de Lógica Aplicada à Tecnologia. São Paulo: SENAC, 2000.
- GOLDSCHMIDT, R.; NOGUEIRA, D.; PASSOS, E.; VELLASCO, M. *Bramining: Um Ambiente Integrado para Descoberta de Conhecimento em Bases de Dados*. III Congresso de Lógica Aplicada à Tecnologia. São Paulo: SENAC, 2002a.
- GOLDSCHMIDT, R.; PASSOS, E.; VELLASCO, M. *Assistance in KDD Goal Definition Process*. Proceedings of the International Conference on Control and Automation. Xiamen, 2002b.
- GOLDSCHMIDT, R.; PASSOS, E.; VELLASCO, M. *A Goal Definition Assistant Applied to KDD Process*. Proceedings of the International Conference on Artificial Intelligence. Las Vegas, 2002c.
- GOLDSCHMIDT, R.; PASSOS, E.; VELLASCO, M. *An Action Plan Definition Assistant in KDD Process*. Proceedings of the Second International Conference on Artificial Intelligence and Applications. Malaga, 2002d.
- GOLDSCHMIDT, R.; PASSOS, E.; VELLASCO, M.; PACHECO, M. *Task Definition Assistance in KDD Applications*. CLEI'03 – XXIX Conferência Latino Americana de Informática. La Paz, 2003.
- GONÇALVES, L. B. *Modelos Neuro-Fuzzy Hierárquicos BSP para Classificação de Padrões e Extração de Regras Fuzzy em Bancos de Dados*. Rio de Janeiro, 2001. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.
- GONÇALVES, L. B.; PACHECO, M. *Clusterização de Bancos de Dados Benchmark Utilizando Algoritmos Genéticos*. Relatório Técnico de Estudo Dirigido. Rio de Janeiro, 2000.
- HAN, J. *Data Mining: Concepts and Techniques*. Simon Fraser University, 1996.

- HAN, J.; KEMBER, M. *Data Mining: Concepts and Techniques*. San Francisco: Morgan Kaufmann Publishers, 2001.
- HAYES-ROTH, B. *An Architecture for Adaptive Intelligent Systems*. Artificial Intelligence: Special Issues on Agents and Interactivity, v. 72, 1995.
- HAYKIN, S. *Neural Networks: a Comprehensive Foundation*. Prentice Hall, 1999.
- HELLERSTEIN, J. M.; AVNUR, R.; CHOU, A.; HIDBER, C.; OLSTON, C.; RAMAN, V.; ROTH, T.; HAAS, P.J. Interactive Data Analysis: The Control Project. *IEEE Transactions on Computer Society*, v. 18, p. 51-59, 1999.
- JENSEN, D.; DONG, Y.; Lerner, B. S.; MCCALL, E. K.; OSTERWEIL, L. J.; SUTTON, S. M.; WISE, A. *Coordinating Agent Activities in Knowledge Discovery Processes*. Proceedings of the International Joint Conference on Work Activities Coordination and Collaboration. San Francisco, 1999.
- KALOUSIS, A.; THEOHARIS, T. NOEMON: Design, Implementation and Performance Results of an Intelligent Assistant for Classifier Selection. *Intelligent Data Analysis*, v. 3, n. 5, 1999.
- KALOUSIS, A.; HILARIO, M. *A Comparison of Inducer Selection via Instance-based and Boosted Decision tree Meta-Learning*. Proceedings of the Fifth International Workshop on Multistrategy Learning, 2000.
- KELLER, J.; PATERSON, I.; BERRER, H. *An Integrated Concept for Multi-Criteria ranking of data mining algorithms*. Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination, 2000.
- KERBER, R.; BECK, H.; ANAND, T.; SMART, B. *Active Templates: Comprehensive Support for the Knowledge Discovery Process*. Proceedings of the International Conference on Knowledge Discovery and Data Mining, p. 244-248, 1998.
- KOZA, J. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.
- LARSON, J. A.; NAVATHE, S. B.; ELMASRI, R. A Theory of Attribute Equivalence in Databases with Application to Schema Integration. *IEEE Transactions on Software Engineering*, v. 15, n. 4, 1989.
- LENAT, D.; BROWN, J. Why AM and Eurisko Appear to Work. *Artificial Intelligence*, n. 23, p. 269-294, 1984.
- LIPSCHULTZ, S. *Topologia Geral*. Coleção Schaum. São Paulo: McGraw-Hill, 1979.

- LIVINGSTON, G. *A Framework for Autonomous Knowledge Discovery from Databases*. 2001. Ph.D. Dissertation – University of Pittsburgh.
- LIVINGSTON, G.; ROSENBERG, J. M.; BUCHANAN, B. G. *A Framework for Autonomous Performing Knowledge Discovery in Databases*. Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Jose, 2000.
- LIVINGSTON, G.; ROSENBERG, J. M.; BUCHANAN, B. G. *Closing the Loop: an Agenda- and Justification-Based Framework for Selecting the Next Discovery Task to Perform*. Proceedings of the 2001 IEEE International Conference on Data Mining. San Jose, 2001.
- LOPES, C. H. P. *Classificação de Registros em Banco de Dados por Evolução de Regras de Associação Utilizando Algoritmos Genéticos*. Rio de Janeiro, 1999. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.
- LOPES, C. H. P.; VELLASCO, M.; PACHECO, M. *Descoberta de Conhecimento e Mineração de Dados*. Apostila, Rio de Janeiro, 2000. Disponível em [www.ica.ele.puc-rio.br](http://www.ica.ele.puc-rio.br).
- MAES, P. Agents that Reduce Work and Information Overload. *Communications of the ACM*, v. 37, n. 7, 1994.
- MICHALEWICZ, Z. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, 1994.
- MICHIE, D.; SPIEGELHALTER, D.; TAYLOR, C. *Machine Learning, Neural and Statistical Classifications*. Ellis Horwood, 1994.
- MORIK, K. *The Representation Race – Preprocessing for Handling Time Phenomena*. Proceedings of the European Conference on Machine Learning 2000, Lecture Notes in Artificial Intelligence 1810. Berlin: Springer Verlag, 2000.
- NAKHAEIZADEH, G.; SCHNABL, A. *Development of Multi-Criteria Metrics for Evaluation of Data Mining Algorithms*. Proceedings of the Fourth International Conference on Knowledge Discovery in Databases and Data Mining. Menlo Park: AAAI Press, p. 37-42, 1998.
- NALIATO, F. C. *Data Mining: Técnicas e Aplicações para o Marketing Direto*. Editora Berkeley, 2001.
- NERI, F. *Projeto de Data Warehouse*. Érica, 2002.
- PACHECO, M. *Algoritmos Genéticos: Princípios e Aplicações*. Apostila, Rio de Janeiro, 2000. Disponível em [www.ica.ele.puc-rio.br](http://www.ica.ele.puc-rio.br).

- PEDRYCZ, W.; GOMIDE, F.. *An Introduction to Fuzzy Sets Analysis and Design*. MIT, 1998.
- PFAHRINGER, B.; BENSUSAN, H. *Meta-Learning by Landmarking Various Learning Algorithms*. Proceedings of the Seventeenth International Conference on Machine Learning, 2001.
- PIATETSKY-SHAPIRO, G. The Data-Mining Industry Coming of Age. *IEEE Intelligent Systems*, p. 32-34, 1999.
- PROVOST, F.; BUCHANAN, B. Inductive Policy: The Pragmatics of Bias Selection. *Machine Learning*, v. 20, n. 1, p. 35-61, 1995.
- PYLE, D. *Data Preparation for Data Mining*. San Francisco: Morgan Kaufmann Publishers, 1999.
- QUINLAN, J. R. *Discovering rules by induction from large collection of examples*. Expert Systems in the Micro Electronic Age. Edinburgh, UK: Edinburgh University Press, 1979.
- QUINLAN, J. R. *C4.5: Programs for Machine Learning*. São Francisco: Morgan Kaufmann, 1993.
- RAINHO, P. S. *Mineração de Dados: Conceitos, Técnicas e Aplicações*. Trabalho de Graduação, UGF. Rio de Janeiro, 2001.
- REZENDE, S.O. *Sistemas Inteligentes: Fundamentos e Aplicações*. São Paulo: Manole, 2003.
- RUSSELL, S.; NORVIG, P. *Artificial Intelligence: A Modern Approach*. New Jersey: Prentice-Hall, 1995.
- SHEN, W.; LENG, B. A Metapattern-Based Automated Discovery Loop for Integrated Data Mining – Unsupervised Learning of Relational Patterns. *IEEE Transactions on Knowledge Data and Engineering*, v. 8, n. 6, p. 898-910, 1996.
- SOARES, C.; COSTA, J.; BRAZDIL, P. *Improved Statistical Support to Matchmaking: Rank Correlation Taking Rank Importance Into Account*. JOCLAD 2001: VII Jornada de Classificação e Análise de Dados, 2001.
- SOUZA, F. J. *Modelos Neuro-Fuzzy Hierárquicos*. Rio de Janeiro, 1999. Tese de Doutorado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.
- SPILIOPOULOU, M.; KALOUSIS, A.; FAULSTICH, L.; THEOHARIS, T. NOEMON: *An Intelligent Assistant for Classifier Selection*. FGML98, v.11, Berlin, p. 90-97, 1998. Disponível em [/spiliopoulou98noemon.html](http://spiliopoulou98noemon.html). Acesso em: 03 dez. 2003.

- SRIKANT, R.; VU, Q.; AGRAWAL, R. *Mining Association Rules with Item Constraints*. Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, 1997.
- SUYAMA, A.; YAMAGUCHI, T. *Specifying and Learning Inductive Learning Systems using Ontologies*. The Methodology of Applying Machine Learning, Menlo Park: AAAI Press, 1998.
- UTGOFF, P. *Shift of Bias for Inductive Concept Learning*. Machine Learning: an Artificial Intelligence Approach, v. 3, São Francisco: Morgan Kaufmann, 1986.
- VERDENIUS, F.; ENGELS R. *A Process Model for Developing Inductive Applications*. Proceedings of the Seventh Belgian-Dutch Conference on Machine Learning. Tilburg, p. 119-128, 1997.
- WEISS, S.; INDURKHYA, N. *Predictive Data Mining: a practical guide*. San Francisco: Morgan Kaufmann Publishers, 1998.
- WIRTH, R.; SHEARER, C.; GRIMMER, U.; REINARTZ, T.; SCHLOSSER, J.; BREITNER, C.; ENGELS, R.; LINDNER, G. *Towards Process-Oriented Tool Support for Knowledge Discovery in Databases*. Proceedings of the First European Symposium on Principles of Data Mining and Knowledge Discovery. Trondheim, 1997.
- WOLPERT, D. The Lack of a Priori Distinctions Between Learning Algorithms and the Existence of a Priori Distinctions Between Learning Algorithms. *Neural Computations*, v. 8, 1996.
- WOLPERT, D.; MACREADY, W. *No Free Lunch Theorems for Search*. Technical Report SFI-TR-95-02-010, The Santa Fe Institute, 1996. Disponível em. Acesso em: 05 jan. 2001.
- WOOLDRIDGE, M.; JENNINGS, N. R. *Agent Theories, Architectures, and Languages: a Survey*. Intelligent Agents, Berlin: Springer-Verlag, 1995.
- ZAKI, M. J. *Parallel and Distributed Data Mining: An Introduction*. Large-Scale Parallel Data Mining. Berlin: Springer-Verlag, 2000.

## Sites Interessantes

A seguir encontram-se listados alguns *sites* relacionados à Mineração de Dados e/ou à Inteligência Computacional, que podem ser úteis ao leitor tanto em seus estudos quanto em seus trabalhos profissionais.

*http://www.kdnuggets.com*  
*http://www.ica.ele.puc-rio.br*  
*http://www.mlnet.org*  
*http://www.ics.uci.edu/~mlearn*  
*http://www.the-data-mine.com*  
*http://www.acm.org/sigkdd*  
*http://www.acm.org/sigmod*  
*http://www.dmg.org*  
*http://www.graal-corp.com.br*