

ERIVALDO DA CUNHA CAVALCANTI JÚNIOR

**UTILIZANDO DATA MINING E JAVA EM SISTEMA DE
DECISÃO**

Monografia apresentada ao Curso de Pós-Graduação de Desenvolvimento de Sistemas Corporativo da FARN - Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte como requisito obrigatório para obter o título de especialista em Desenvolvimento de Sistemas Corporativos.

Orientador: David Pereira, M.sc.

**Natal
2007**

ERIVALDO DA CUNHA CAVALCANTI JÚNIOR

**UTILIZANDO DATA MINING E JAVA EM SISTEMA DE
DECISÃO**

Monografia apresentada ao Curso de especialização em Desenvolvimento de Sistemas Corporativos da Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte, aprovada em 01 de março de 2008, pela banca composta pelo seguinte membro:

David Rivaldo da Silva Pereira

David Pereira, M.sc.

Catálogo na Publicação – Biblioteca da FARN
Setor de Processos Técnicos

Cavalcanti, Erivaldo da Cunha Cavalcanti Júnior
Utilizando Data Mining e Java em Sistema de Decisão/ Erivaldo da
Cunha Cavalcanti Júnior. – Natal: FARN, 2008.
73 f.

Orientador: David Pereira (M. Sc.)
Monografia (Especialização em Desenvolvimento de Sistemas
Corporativos) – Faculdade Natalense para o Desenvolvimento do Rio
Grande do Norte.

1. DataMining – Monografia. 2. KDD – Monografia. 3. JavaDataMing –
Monografia. 4. Utilizando DataMining e Java em Sistemas de Decisão–
Monografia. 1. Título.

RN/FARN/BC

CDU 004.4

RB. 30706
FARN
BIBLIOTECA
Nº Reg M000897
16/04/08

RESUMO

Historicamente as organizações têm gastado muito tempo e dinheiro para reunir dados, gerando uma quantidade enorme de informações. A evolução da Tecnologia da Informação e o desenvolvimento de técnicas de aprendizado de máquina aumentaram o potencial das organizações para investir no processo de descoberta de conhecimento (KDD). Neste contexto está inserido a mineração de dados (data mining) como técnica para explorar os dados com o objetivo de descobrir relações e padrões existentes.

Palavras-chave: Data Mining. Data Warehouse. ERP. Java Datamining.

ABSTRACT

Through the years, companies have spent a lot of time and money to gather data generating a huge amount of information. The Information Technology evolution and the development of machine-learning techniques increased the company's potential to invest in the Knowledge Discovery in Database (KDD). Data mining is within this context as a technique for data exploring aiming at finding out the existing relations and patterns.

KeyWorks: Data Mining. Data Warehouse. ERP. Java Datamining.

LISTA DE FIGURAS

Figura 1 – Hierarquia entre Dados, Informação e conhecimento.

Figura 2 – Etapas Operacionais do Processo do KDD

Figura 3 – Base de fatos de uma financeira Hiptética

Figura 4 – Taxonomia de Atividades na Área de KDD.

Figura 5 – Um típico sistema de ERP inserindo dados em um Data Warehouse.

Figura 6 – Ambiente de Criação de Relatório

Figura 7 – Ambiente da criação de relatório

Figura 8 – Origem de dados de vendas

Figura 9 – Origem de dados de vendas

Figura 10 – Híbrido Seqüencial

Figura 11– Modelo Híbrido Auxiliar

Figura 12 – Modelo Híbrido Incorporado

Figura 13 – Algoritmo K-MEANS

Figuras 14 – Passos Subseqüentes do Algoritmo

Figuras 15 – Componentes de um sistema de Data Mining.

Figura 16 – Passos da Mineração de Dados

SUMÁRIO

1	INTRODUÇÃO	7
2	OBJETIVOS.....	17
2.1	OBJETIVO GERAL	17
2.2	OBJETIVO ESPECÍFICO	17
3	SISTEMAS DE ERPS	18
3.1	Visão geral de um ERP	18
3.2	Exemplos de fornecedores de ERPS	23
4	DATA WAREHOUSE.....	25
5	O PROCESSO DE KDD	31
6	MINERAÇÃO DE DADOS	40
7	ALGUNS ALGORITMOS.....	44
8	CLUSTERIZAÇÃO.....	45
8.1	VISAO GERAL.....	45
8.2	CLUSTERIZAÇÃO → CLASSIFICAÇÃO	51
8.3	CLUSTERIZAÇÃO → SUMARIZAÇÃO	52
9	ALGORITMOS DE CLUSTERIZAÇÃO	54
9.1	K-MEANS.....	54
9.2	K-MODES	55
9.3	K-PROPOTYPES.....	55
9.4	K-MEDOIDS.....	56
10	JAVA DATA MINING API.....	58
10.1	MINERE SEUS DADOS COM A JDM, EXPLORE A API JAVA DATA MINING 58	
10.2	Transformando Dados em Conhecimento	59
10.3	Usando a API JDM.....	63
11	CONCLUSÃO	69
12	REFERÊNCIAS.....	70

1 INTRODUÇÃO

O grande crescimento na área da Tecnologia da Informação tem gerado um gigante crescimento no volume de dados. Dados esses gerados por sistemas de informações.

Tecnologias como Internet, ERPs, SGBDs e Sistemas de Informação em geral, são alguns dos exemplos de recursos que viabilizam a proliferação de inúmeras bases de dados, de perfil comercial, administrativo, governamental e científico.

Atualmente, dados científicos em projeto de pesquisa, tais como missões da NASA e o Projeto Genoma Humano, têm alcançado proporções gigantescas. Empresas com FedEx, Wal-Mart, UPS, Banco do Brasil, Caixa Econômica Federal e Sendas possuem bases de dados da ordem de centenas de terabytes de informações (GOLDSCHMIT, 2000).

Diante desse fato, é evidente surgir algumas questões, por exemplo: *"O que farei com todos esses dados armazenados?"*, *"Como utilizar esse patrimônio digital para benefício da instituição?"*, *"Como analisar e utilizar forma útil essa grande massa de dados disponível?"*.

Não existem condições para o homem analisar e dar uma resposta correta para uma grande quantidade de dados, sem o auxílio de ferramentas computacionais apropriadas.

Por isso é imprescindível que o homem esteja de posse de uma ferramenta dessas para auxiliá-lo, de forma automática e inteligente, na tarefa de analisar, interpretar e relacionar esses dados, para que possa desenvolver e selecionar estratégias de ação em cada contexto da aplicação.

Para entender esse contexto surge uma nova área denominada, Descoberta de Conhecimento em Bases de Dados (Knowledge Discovery in Databases – KDD).

Esta área vem despertando grandes interesses junto às comunidades científicas, industrial e comercial. A expressão mineração de dados (Data Mining),

mais conhecida, é na realidade uma das etapas da Descoberta de Conhecimento em Bases de dados.

Neste momento, para proporcionar melhor entendimento do problema é importante destacar as diferenças e a hierarquia entre dado, informação e conhecimento, conforme ilustra a figura 1 (GOLDSCHMIT, 2000).

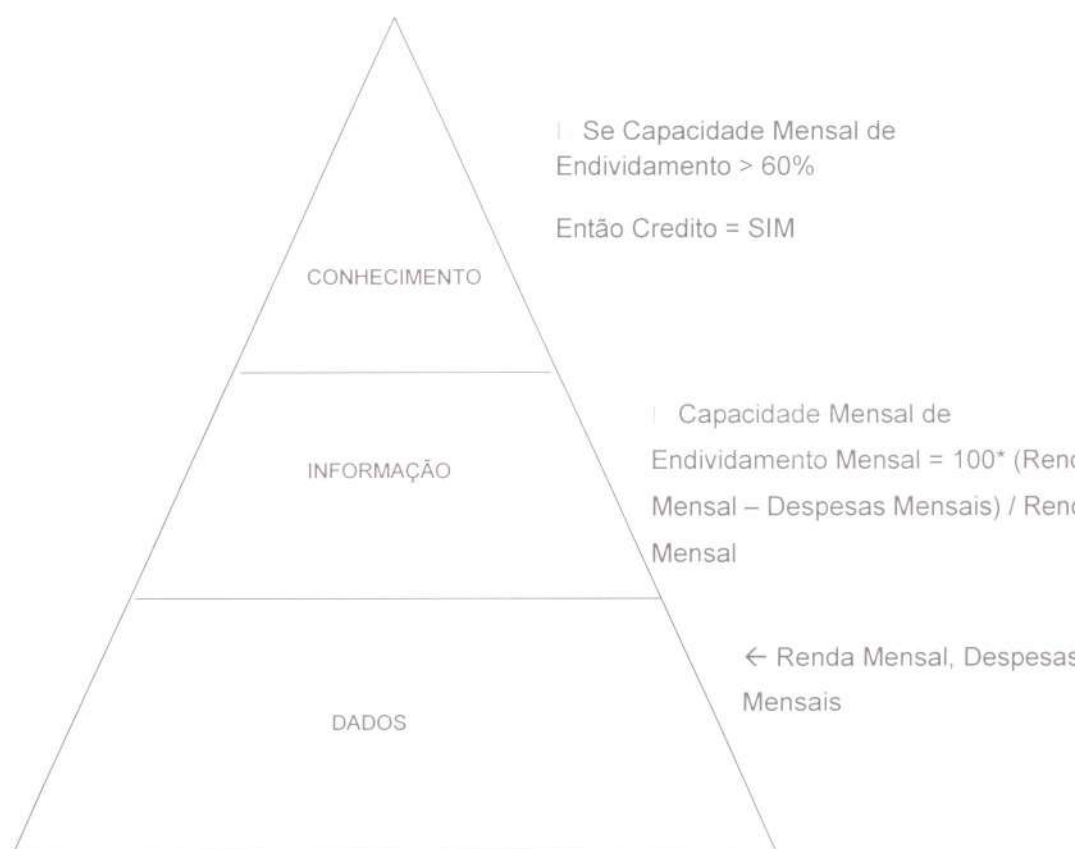


Figura 1: Hierarquia entre Dados, Informação e conhecimento

Os dados na base da pirâmide, podem ser interpretados como itens elementares, captados e armazenados por recursos da Tecnologia da Informação.

No exemplo apresentado, foi considerado uma base de dados de uma financeira que armazena a renda mensal e as despesas de seus clientes.

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

Um exemplo de uma regra de produção tem a seguinte forma SE <condicao> ENTÃO <conclusões>.

No sentido geral o conhecimento não pode ser extraído das bases de dados por meio dos recursos tradicionais da tecnologia da informação. Esses recursos tradicionais seria consultas em bancos de dados, a qual não passaria por uma sequência de processamento com objetivo de gerar de conhecimento. Para que esse conhecimento seja abstraído é utilizado o KDD.

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., 1996): "KDD é um processo, de várias etapas, não trivial, interativo e iterativo para identificação de padrões compreensíveis, válidos, novos e potencialmente úteis a partir de grandes conjuntos de dados".

O processo de descoberta de conhecimento em bases de dados segue um processo composto por várias etapas operacionais. A figura 2 apresenta em etapas um resumo de como é executado o processo KDD.

Nesse resumo o primeiro processo a ser executado é o pré-processamento, que compreende as funções relacionadas à captação, à organização e ao tratamento de dados. A etapa de pré-processamento tem como objetivo a preparação dos dados para os algoritmos das etapas operacionais executadas em processos 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., 1996). Abaixo a figura 2 mostra uma representação de como seria esse processo.



Figura 2: Etapas Operacionais do Processo do 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 de uso (GOLDSCHMIT, 2000). Cabe ao analista humano a árdua tarefa de orientar a execução do processo KDD.

Na definição formal da KDD o termo *iterativo* indica que o homem é o responsável pelo processo. O homem utiliza os recursos computacionais disponíveis para executar a análise e 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.

Ainda considerando a definição de KDD, um padrão deve ser interpretado como um conhecimento restando segundo as normas sintáticas de alguma

linguagem formal (FAYYAD et al., 1996). Ou seja, um padrão compreensível refere-se, portanto, a um padrão representado em alguma forma de representação do conhecimento que possa ser interpretada pelo homem.

O termo *padrão válido* indica que o conhecimento tem que ser verdadeira e adequada ao contexto do KDD.

Um *novo padrão* terá que adicionar novos conhecimentos aos conhecimentos existentes no contexto da aplicação do KDD. E o *conhecimento útil* é aquele aplicado da forma a proporcionar benefícios ao contexto da aplicação do KDD. Para fins ilustrativos o exemplo na área de concessão de crédito da Figura 3 (NALIATO, 2001).

Cada ponto do plano cartesiano representa um dos clientes da financeira. O conjunto de pontos forma a base de fatos (banco de dados). Com objetivo de facilitar a visualização, foram utilizados apenas três atributos: renda, despesas e comportamento do cliente com relação ao crédito recebido da financeira. Renda e despesas estão representadas pelos eixos dos planos. O comportamento do cliente está representada por duas classes: A **X** – clientes negligentes (não pagaram o crédito) e a **O** – clientes pontuais (pagaram o crédito recebido). Genericamente falando, o problema pode envolver n atributos sendo representado em um hiperespaço n -dimensional. Esse exemplo a fim de ter um melhor entendimento 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.

Há diversas formas de separar os dois conjuntos de clientes. Um exemplo seria.

SE renda menor = R\$t
ENTAO Clientes = Não negligente
SENAO Clientes = Negligente

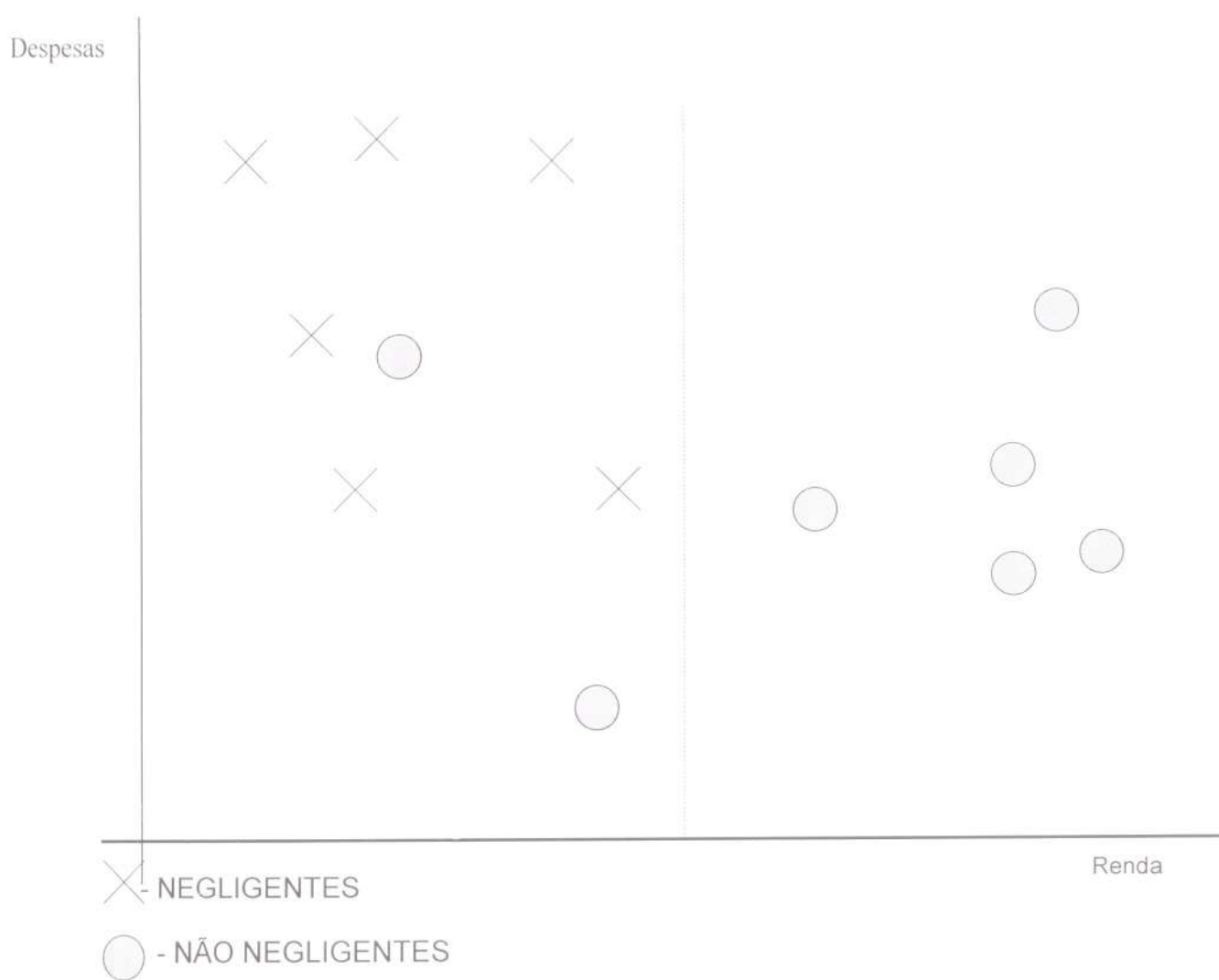


Figura 3: Base de fatos de uma financeira Hiptética

A regra que foi apresentada acima é um padrão compreensível ao homem. Embora ocorra casos que a mesma não satisfaçam, essa regra é válida para a maioria dos registros da base. Um das medidas normalmente utilizada na avaliação de qualidade de uma regra é a *acurácia*, também conhecida como *confiança* ou *precisão da regra*. Refere-se a proporção dos casos que satisfazem ao antecedente e ao consequente da regra em relação ao número de casos que satisfazem somente ao antecedente da regra. Neste exemplo, a separação dos conjuntos de clientes representada pela regra é linear. Ou seja, esse problema não é linearmente separável, ou seja, os dois conjuntos não podem ser separados em uma linha reta.

A Descoberta de Conhecimento em Bases de Dados é multidisciplinar e, historicamente, originou-se de diversas áreas, dentre elas podem se destacar as seguintes.

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

A figura 4 tem como objetivo de situar a área KDD ela apresenta uma taxonomia das atividades na área de Descoberta de Conhecimento em Bases de Dados (GOLDSCHMIT, 2003).

As atividades na área de KDD pode ser dividida em três grandes grupos: atividades voltadas ao desenvolvimento tecnológico, atividades de execução de processos do KDD, e atividades envolvendo a aplicação de resultados obtidos em processos de KDD.

- **Desenvolvimento Tecnológico**

Esse item abrange todas as iniciativas de concepção, aprimoramento e desenvolvimento de algoritmos, ferramenta 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 KDD.

- **Aplicação de Resultados**

Finalmente, uma vez obtidos modelos de conhecimentos 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 estratégicas de negócios que tenham como objetivos procurar tirar proveito do conhecimento obtido. Tais alterações podem variar desde o posicionamento de produtos nas gôndulas de um mercado até políticas estratégicas corporativas (AGRAWAL et al., 1993; GOLDSCHMIT & PASSOS, 2000; GODY et al., 2003). O desenvolvimento de sistemas que utilizam conhecimentos extraídos de bases de dados tem proiciado valiosas ferramentas de apoio à decisão (WEISS & INDURKHYA, 1998).

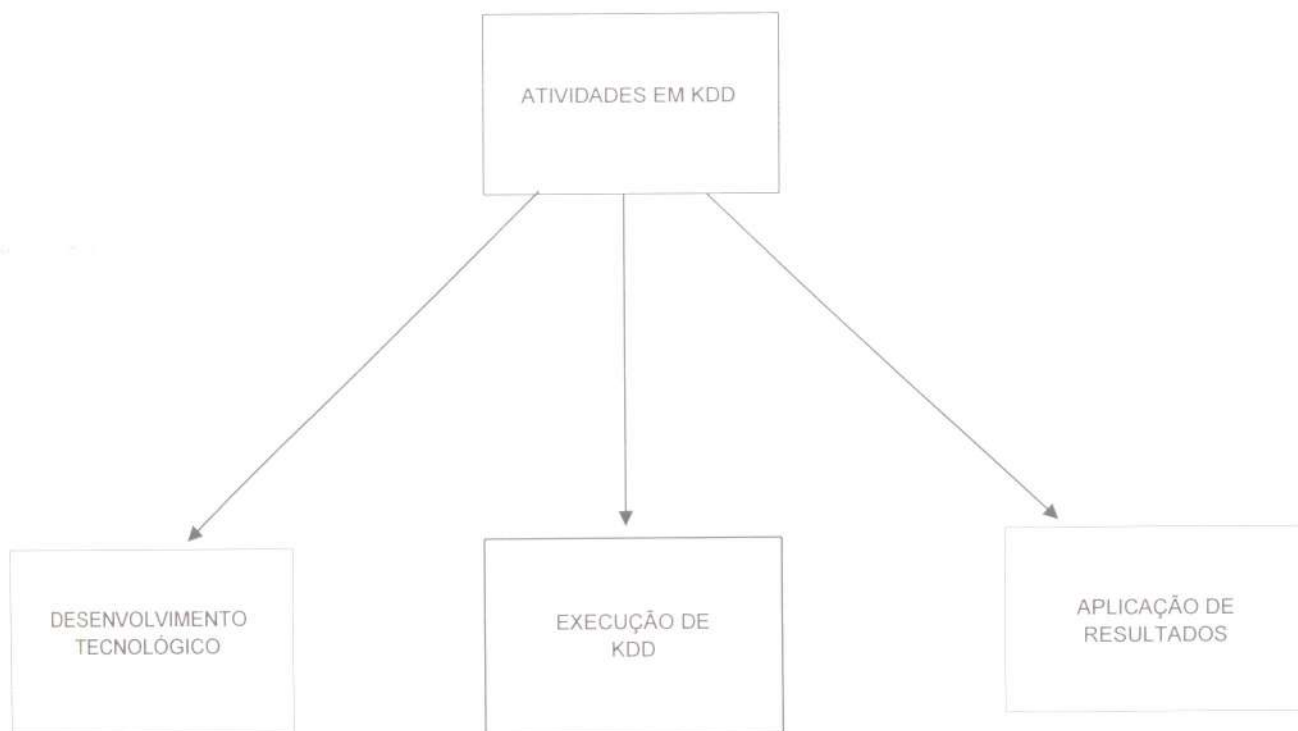


Figura 4: Taxonomia de Atividades na Área de KDD.

Pode ainda em termos históricos fazer uma divisão sobre a Mineração de Dados referenciando a Descoberta de Conhecimento em Bases de Dados em quatro gerações.

A primeira geração de Mineração de Dados (PIATESKY-SHAPIRO, 1999) apareceu nos anos 80 e consistia em ferramentas de análise voltadas a uma única tarefa, sem suporte as demais etapas do processo. Essas tarefas incluíam, em geral, a construção de classificadores usando ferramentas de Introdução 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 (PIATESKY-SHAPIRO, 1999) apareceu em 1995 com o desenvolvimento de ferramentas chamadas "suítes". Essas ferramentas eram dirigidas ao fato de que o processo de descoberta do conhecimento requer múltiplos tipos de análise dos dados. As "suítes" 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.

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 auxílio de especialista em análise de dados. Assim, surgiu a necessidade da terceira de Mineração de Dados (PIATESKY-SHAPIO, 1999) no final dos anos 90. Essas soluções são orientadas para 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 *Softwares Falcão* (Ratinho, 2001) para detecção de fraudes em cartão de crédito é um exemplo desse tipo de sistema.

A quarta geração de Mineração de Dados compreende o desenvolvimento e aplicação de técnicas e ferramentas que auxiliem o homem na própria condução do complexo processo KDD (GOLDSCHMIT, 2000).

2 OBJETIVOS

2.1 OBJETIVO GERAL

O objetivo desse trabalho é dar uma visão geral no processo KDD (Knowledge Discovery in Databases) utilizando a tecnologia Java, essas tecnologias têm como objetivo adquirir conhecimentos para gerar decisões, em diversos sistemas como, por exemplo, sistemas de vendas, sistemas de financeira etc.

2.2 OBJETIVO ESPECÍFICO

O objetivo específico desse trabalho é apresentar um exemplo qual será um sistema inteligente o mesmo irá trazer as decisões corretas baseadas no conhecimento adquirido. Esse trabalho mostrará como a tecnologia JAVA pode ser inserida nesse contexto de mineração de dados.

3 SISTEMAS DE ERPS

3.1 Visão geral de um ERP

Enterprise Resource Planning (ERP), Sistemas Integrados de Gestão Empresarial, no Brasil (SIGE) são sistemas de informações que integram todos os dados e processos de uma organização em um único sistema (Laudon, Padoveze). A integração pode ser vista sob a perspectiva funcional (sistemas de: finanças, contabilidade, recursos humanos, fabricação, marketing e vendas, etc.) e sob a perspectiva sistêmica (sistema de processamento de transações, sistemas de informações gerenciais, sistemas de apoio à decisão, etc.).

Os Erps em termos gerais são uma plataforma de software desenvolvida para integrar os diversos departamentos de uma empresa, possibilitando a automação e armazenamento de todas as informações de negócios.

No final da década de 50, quando os conceitos modernos de controle tecnológico e gestão corporativa tiveram seu início, a tecnologia vigente na época era baseada nos gigantescos mainframes que rodavam os primeiros sistemas de controle de estoques – atividade pioneira da interseção entre gestão e tecnologia. A automatização era cara, lenta – mas já demandavam menos tempo que os processos manuais – e para poucos.

No início da década de 70, a expansão econômica e a maior disseminação computacional geraram o avô dos ERPs, os MRPs (Material Requirement Planning ou planejamento das requisições de materiais). Eles surgiram já na forma de conjuntos de sistemas, também chamados de pacotes, que conversavam entre si e que possibilitavam o planejamento do uso dos insumos e a administração das mais diversas etapas dos processos produtivos.

Seguindo a linha evolutiva, a década de 80 marcou o início das redes de computadores ligadas a servidores – mais baratos e fáceis de usar que os mainframes – e a revolução nas atividades de gerenciamento de produção e logística. O MRP se transformou em MRP II (que significava Manufacturing

Resource Planning ou planejamento dos recursos de manufatura), que agora também controlava outras atividades como mão de obra e maquinário.

Na prática, o MRP II já poderia ser chamado de ERP pela abrangência de controles e gerenciamento. Porém, não se sabe ao certo quando o conjunto de sistemas ganhou essa denominação. Uma datação interessante é 1975, ano no qual surgiu a empresa alemã – um símbolo do setor – SAP (Systemanalyse and Programmentwicklung, na tradução literal Análise de Sistemas e Desenvolvimento de Programas). Com o lançamento do software R/2, ela entrou para a história da área de ERP e ainda hoje é seu maior motor de inovação.

O próximo passo, já na década de 80, serviu tanto para agilizar os processos quanto para estabelecer comunicação entre essas “ilhas” departamentais. Foram então agregados ao ERP novos sistemas, também conhecidos como módulos do pacote de gestão. As áreas contempladas seriam as de finanças, compras e vendas e recursos humanos, entre outras, ou seja, setores com uma conotação administrativa e de apoio à produção ingressaram na era da automação.

A nomenclatura ERP ganharia muita força na década de 90, entre outras razões pela evolução das redes de comunicação entre computadores e a disseminação da arquitetura cliente/servidor – microcomputadores ligados a servidores, com preços mais competitivos – e não mais mainframes. E também por ser uma ferramenta importante na filosofia de controle e gestão dos setores corporativos, que ganhou aspectos mais próximos da que conhecemos atualmente.

As promessas eram tantas e tão abrangentes que a segunda metade daquela década seria caracterizada pelo boom nas vendas dos pacotes de gestão. E, junto com os fabricantes internacionais, surgiram diversos fornecedores brasileiros, empresas que lucraram com a venda do ERP como um substituto dos sistemas que poderiam falhar com o bug do ano 2000 – o problema na data de dois dígitos nos sistemas dos computadores.

Entre as mudanças mais palpáveis que um sistema de ERP propicia a uma corporação, sem dúvida, está a maior confiabilidade dos dados, agora monitorados em tempo real, e a diminuição do retrabalho. Algo que é conseguido com o auxílio e

o comprometimento dos funcionários, responsáveis por fazer a atualização sistemática dos dados que alimentam toda a cadeia de módulos do ERP e que, em última instância, fazem com que a empresa possa interagir. Assim, as informações trafegam pelos módulos em tempo real, ou seja, uma ordem de vendas dispara o processo de fabricação com o envio da informação para múltiplas bases, do estoque de insumos às logísticas do produto. Tudo realizado com dados orgânicos, integrados e não redundantes.

Para entender melhor como isto funciona, o ERP pode ser visto como um grande banco de dados com informações que interagem e se realimentam. Assim, o dado inicial sofre uma mutação de acordo com seu status, como a ordem de vendas que se transforma no produto final alocado no estoque da companhia.

Ao desfazer a complexidade do acompanhamento de todo o processo de produção, venda e faturamento, a empresa tem mais subsídios para se planejar, diminuir gastos e repensar a cadeia de produção. Um bom exemplo de como o ERP revoluciona uma companhia é que com uma melhor administração da produção, um investimento, como uma nova infra-estrutura logística, pode ser repensado ou simplesmente abandonado. Neste caso, ao controlar e entender melhor todas as etapas que levam a um produto final, a companhia pode chegar a ponto de produzir de forma mais inteligente, rápida e melhor, o que, em outras palavras, reduz o tempo que o produto fica parado no estoque.

A tomada de decisões também ganha uma outra dinâmica. Imagine uma empresa que por alguma razão, talvez uma mudança nas normas de segurança, precisa modificar aspectos da fabricação de um de seus produtos. Com o ERP, todas as áreas corporativas são informadas e se preparam de forma integrada para o evento, das compras à produção, passando pelo almoxarifado e chegando até mesmo à área de marketing, que pode assim ter informações para mudar algo nas campanhas publicitárias de seus produtos. E tudo realizado em muito menos tempo do que seria possível sem a presença do sistema.

Entre os avanços palpáveis, podemos citar o caso de uma indústria média norte-americana de autopeças, situada no estado de Illionois, que conseguiu reduzir o tempo entre o pedido e a entrega de seis para duas semanas, aumentando a

eficiência na data prometida para envio do produto de 60% para 95% e reduzindo as reservas de insumos em 60%. Outra diferença notável: a troca de documentos entre departamentos que demorava horas ou mesmo dias caiu para minutos e até segundos.

Esse é apenas um exemplo. Porém, de acordo com a empresa, é possível direcionar ou adaptar o ERP para outros objetivos, estabelecendo prioridades que podem tanto estar na cadeia de produção quanto no apoio ao departamento de vendas como na distribuição, entre outras. Com a capacidade de integração dos módulos, é possível diagnosticar as áreas mais e menos eficientes e focar em processos que possam ter o desempenho melhorado com a ajuda do saco de sistemas.

Algumas das vantagens da implementação de um ERP numa empresa são:

- Eliminar o uso de interfaces manuais.
- Redução de custos.
- Otimizar o fluxo da informação e a qualidade da mesma dentro da organização (eficiência).
- Aperfeiçoar o processo de tomada de decisão.
- Eliminar a redundância de atividades.
- Reduzir os limites de tempo de resposta ao mercado.
- Reduz as incertezas do *lead-time*.

Algumas das desvantagens da implementação de um ERP numa empresa são:

1. A utilização do ERP por si só não torna uma empresa verdadeiramente integrada.
2. Altos custos que muitas vezes não comprovam o custo/benefício.
3. Dependência do fornecedor do pacote.
4. Adoção de best practices aumenta o grau de imitação e padronização entre as empresas de um segmento.
5. Cortes de pessoal, que gera problema social.

6. Torna os módulos dependentes uns dos outros, pois cada departamento depende das informações do módulo anterior, por exemplo. Logo, as informações têm que ser constantemente atualizadas, uma vez que as informações são em tempo real (on line), ocasionando maior trabalho.

7. Excesso de controle sobre as pessoas, o que aumenta a resistência à mudança e pode gerar desmotivação por parte dos funcionários.

Segundo uma pesquisa Chaos e Unfinished Voyages (1995) os principais fatores críticos de sucesso para um projeto de implantação de um ERP são:

- Envolvimento do Usuário.
- Apoio da direção.
- Definição clara de necessidades.
- Planejamento adequado.
- Expectativas realistas.
- Marcos intermediário.
- Equipe competente.
- Comprometimento.
- Visão e objetivos claros.
- Equipe dedicada.

Um dos pontos fortes do sistema é a integração entre os módulos. Os monitores, as operações e a navegação dentro do sistema são de tais formas padronizadas que, na medida em que o utilizador se familiariza com um módulo, aprende mais facilmente e rapidamente os demais. Os acessos às informações

são realizados através de menus com utilização de senhas, permitindo que se controlem quais os utilizadores que têm acesso a que tipo de informações. Além disso, os menus são customizados de forma que cada utilizador visualize e tenha acesso somente às operações que atendam aos objetivos específicos de seu interesse.

Os sistemas ERP são compostos por uma base de dados única e por módulos que suportam diversas atividades das empresas. A figura 3.1 abaixo apresenta uma estrutura típica de funcionamento de um sistema ERP. Os dados utilizados por um módulo são armazenados na base de dados centrais para serem manipulados por outros módulos. Abaixo uma figura dos principais módulos de ERP.

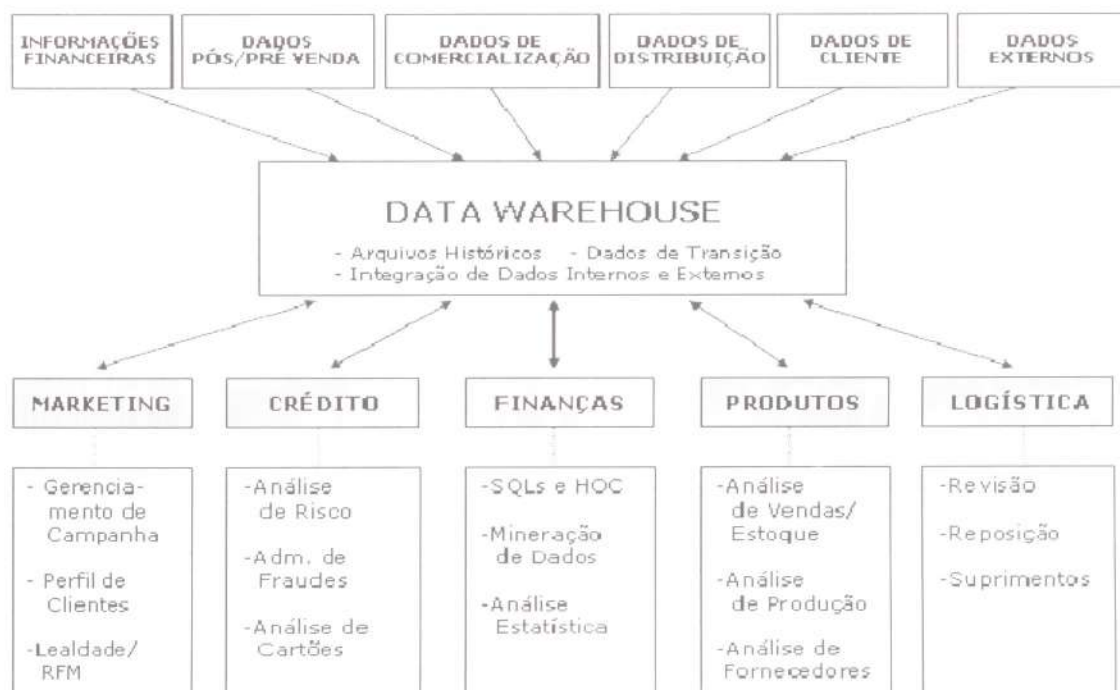


Figura 5: Um típico sistema de ERP inserindo dados em um Data Warehouse

3.2 Exemplos de fornecedores de ERPS

Esses sites são fabricantes dos principais ERPS que existe no mercado mundial.

- Baan – <http://www.baan.com>
- Datasul - <http://www2.datasul.com.br/>
- JD Edwards - <HTTP://www.jdedwards.com/>
- Microsiga - <http://www.microsiga.com.br/>
- MKGroup (Computer Associates) - <HTTP://www.mkgroup.com/>
- Oracle - <http://www.oracle.com/applications/>
- Peoplesoft - <HTTP://www.peoplesoft.com/>
- SAP - <http://www.sap.com/>
- SENIOR SISTEMAS - <http://www.senior.com.br/>
- Microsoft – <http://www.microsoft.com/>

4 DATA WAREHOUSE

Um *data warehouse* é um banco de dados reunido a partir de um ou muitos sistemas e destinado a oferecer suporte à produção de relatórios gerências e à tomada de decisão. De uma forma geral, contudo, quando as pessoas falam a respeito de armazenamento de dados, na verdade elas estão se referindo aos armazenamentos de dados e às ferramentas que, juntos, formam os modernos ambientes de produção de relatórios.

A área do armazenamento e seus produtos continuam a amadurecer, à medida que as empresas ganham experiência com novas tecnologias e adotam as melhores arquiteturas para suas necessidades. As antigas batalhas entre os modelos de dados “normalizados” e “desnormalizados”, embora não tenham sido completamente resolvidas, parecem muito mais próximas de uma solução. A exploração de dados – uma forma de análise destinada a criar modelos previsíveis, que estava no auge há dois anos – está se tornando cada vez uma tecnologia consagrada.

Apesar disso, os objetivos básicos do armazenamento de dados permanecem os mesmos.

- Fornecer uma origem confiável, única e integrada de informações corporativas importantes.
- Proporcionar, aos usuários finais, acesso aos seus dados, sem depender de relatórios produzidos pelo departamento de Informática.
- Permitir aos analistas examinarem os dados corporativos e até produzirem modelos do tipo de previsão “e se”, a partir desses dados.

Os *data warehouse* são mesmo um componente importante das arquiteturas de inteligência empresarial bem fundamentadas.

Um ambiente de produção de relatório moderno proporcionará ao usuário acesso aos seus dados, mas isso não resolve todos os seus problemas. Apenas porque os usuários têm acesso aos dados não garante a integridade desses dados. Isso também não garante que os tempos de resposta do sistema sejam adequados. Isso não garante que seus sistemas não expurgarão dados antigos, antes que o ávido útil esteja terminado. Na verdade, fornecer aos usuários acesso aos dados pouco diz a respeito desses dados.

Para tratar de muito desses problemas de dados, Bill Inmon fala do data warehouse. Na verdade, Inmon é freqüentemente chamado o "pai do armazenamento de dados". Em seu livro, *Building the Data Warehouse* (Jonh Wiley & Sonns Inc., 1996), Inmon descreve o warehouse como um "conjunto de dados orientado para o assunto, integrado, não-volátil, variante no tempo, no apoio de decisões gerencias".

Em temos amigáveis para o usuário, um data warehouse é um conjunto de instantâneos de dados, extraídos de sistemas de processamento de transação, em determinados intervalos. Os data warehouses são bancos de dados usados unicamente para a produção de relatórios. Isso contrasta com os bancos de dados subjacentes aos sistemas operacionais / captura de dados e sistemas de processamento de transação on-line (OLTP) tradicionais. Os exemplos clássicos de sistemas operacionais incluem os seguintes:

- Livro razão geral
- Contas a pagar
- Gerenciamento financeiro
- Processamento de pedidos
- Entrada de pedidos
- Inventário

Os datas warehouses são preenchidas com dados de duas origens principais. Mais freqüentemente, eles são preenchidos com migrações periódicas de dados de sistemas operacionais. A segunda origem é constituída de bancos de dados externos, freqüentemente adquiridos. Exemplos desses dados incluíram listas de informações sobre lucros demográficas. Se puderem vincular essas informações adquiridas aos dados internos a respeito dos clientes, o data warehouse será uma ferramenta excelente para desenvolver um perfil de um bom cliente.

Com esse perfil a equipe de marketing poderia buscar ferramentas que objetivassem prospectos correspondentes a esse perfil.

A figura 7 mostra, em alto nível, como o ambiente de produção de relatório moderno usa data warehouse para integrar dados de uma variedade de origens importantes e tornar esses dados disponíveis para os usuários ávidos de informação.

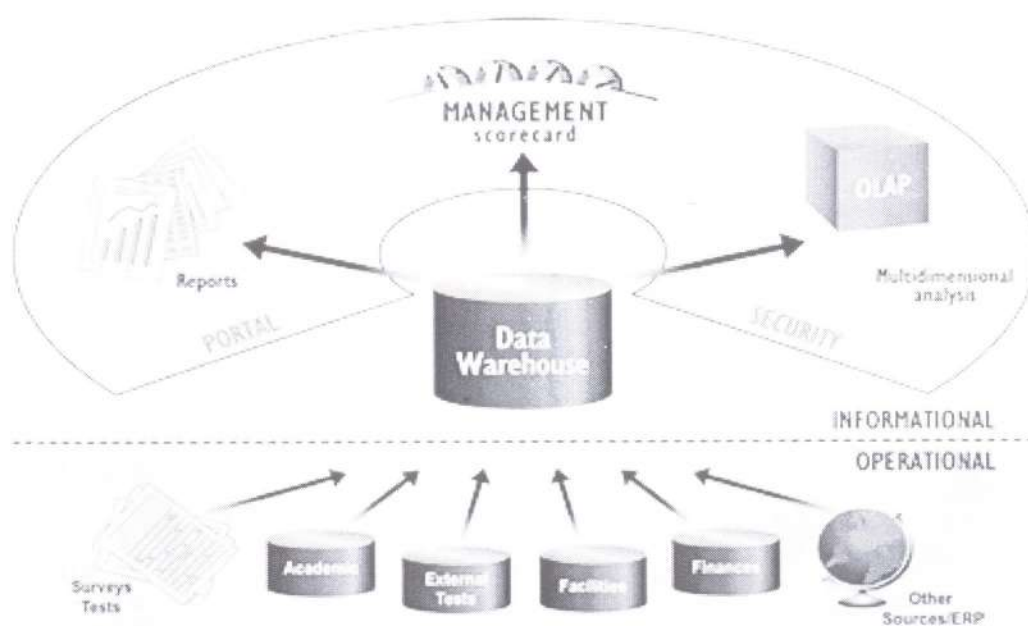


Figura 6: Ambiente da criação de relatório

Os bancos OLTP (On-line Transaction Processing ou Processamento de transações em tempo real) normalmente são destinados a conter informações a respeito de pequenos subconjuntos da empresa. Por exemplo, um varejista poderia ter sistemas de entrada de pedidos separados e um banco de dados para vendas no varejo, por catálogo e no atacado. Cada sistema oferecerá suporte para consultas a respeito das informações que ele captura. Mas, suponha que um usuário queira fazer uma consulta sobre todas as vendas e não simplesmente as vendas capturadas por um sistema particular, independentemente do canal responsável por essa venda. Para tratar desse tipo de situação, o banco de dados data warehouse deve ser orientado para o assunto: organização em áreas de assunto, como vendas, em vez de em torno de origens de dados OLTP. A figura 4.2 mostra como as origens de dados de vendas se transformam na área de assunto vendas do warehouse.

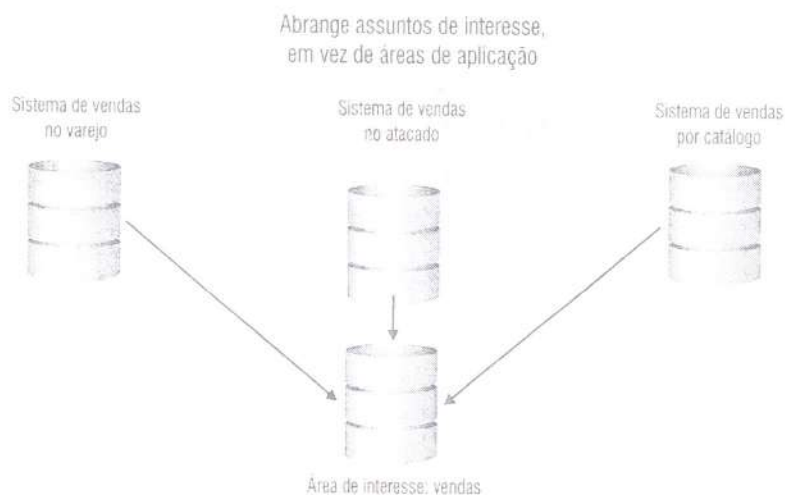


Figura 7: Origem de dados de vendas

Agora, os dados desses sistemas separados podem ser acessados a partir de um único lugar e consultados em um único relatório. Por exemplo, quando uma equipe trabalhou para um grande fornecedor de computadores no início da carreira, os funcionários do fornecedor não podiam responder a esta simples pergunta "Quem são nossos dez principais clientes?". Eles podiam dizer quem eram os dez principais clientes de computadores de grande porte. Podiam dizer quem eram os dez principais clientes de microcomputadores. Podiam dizer quem eram os dez principais clientes de PC. Mas essas respostas por si só não eram inteiramente precisas; quando o fornecedor se atinha a analisar o volume total de vendas de todos os clientes, em todas as linhas de produto.

Imagine quanto tempo era gasto, sempre que a diretoria aparecia com uma nova pergunta. Como não existia nenhum repositório de dados central, cada pedido era um processo penoso. E como se tratava de um processo manual, também havia a possibilidade de erros manuais. Apesar disso, essas eram as melhores disponíveis na ocasião; muitas decisões. Se essa empresa tivesse construído um data warehouse com uma área de assunto de vendas, a gerência teria conseguido informações muito mais precisas. Além disso, as informações a respeito dos

clientes, reunidas a partir de vários sistemas, teriam se tornados acessíveis e evidentes.

Não volátil, significa que o warehouse é read-only (apenas leitura) os usuários não podem gravar nele. Ao contrário dos bancos de dados operacionais, os warehouses oferecem suporte principalmente à produção de relatórios e não captura de dados. O warehouse é um registro histórico a figura 4.3 ilustra essa diferença fundamental entre OLTP e data warehouse.

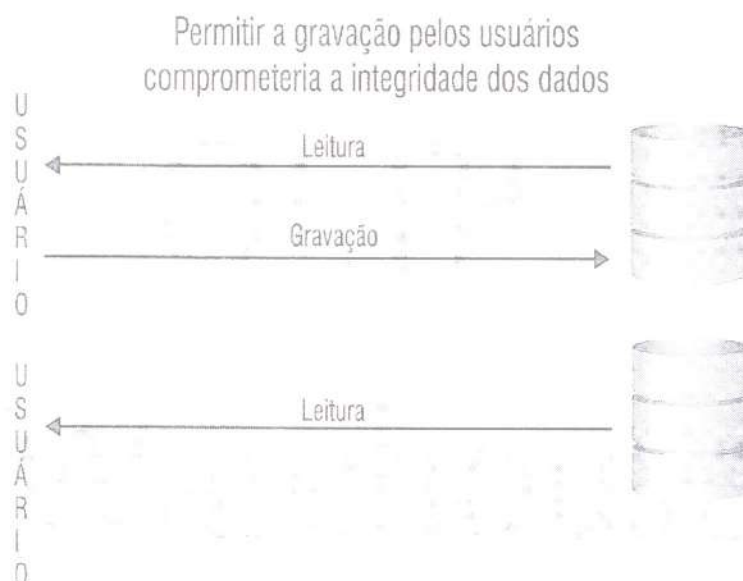


Figura 8 Diferença entre OLTP e um Warehouse

Sem um componente tempo, a maior parte das informações é inútil. Para dizer que as vendas foram R\$ 5.000,000, 00 não é muito importante. Por quê? Por que isso não indica o período de tempo durante o qual as vendas foram R\$ 5.000,000, 00. Esse valor é relativo a 1999, 1998, janeiro de 2007 ou ao ano corrente até o presente momento? Claramente, o tempo é um componente muito importante da produção de relatório e, portanto dos data warehouse.

Além disso, considere a ideia da análise da tendência exige o acesso aos dados históricos. Geralmente, mais de um ano de histórico é necessário,

particularmente quando o negócio é sazonal. A maioria dos sistemas operacionais, por outro lado, não armazena um histórico muito grande: grandes volumes de dados são os inimigos dos tempos de resposta. Como os sistemas operacionais são projetados para respostas imediatas, freqüentemente eles expurgam dados dentro grande volumes de histórico. Essa é uma das principais razões pelas quais os warehouses freqüentemente são bancos de dados enormes. Lembre-se, contudo, de que nem todo histórico deve ser mantido no mesmo nível de detalhes. Por exemplo, um antigo cliente operava no negócio de venda de alimento por atacado. O warehouse desse cliente controlava os detalhes de cada transação de vendas para um mês. Quando os dados tinham um mês de idade, eles eram reunidos em valores de resumo semanais.

Após dois anos. Os dados eram mantidos apenas como resumos mensais. Isso não somente ajuda no desempenho das consultas aos data warehouses, mas também faz sentido no negócio. Os detalhes de toda transação individual geralmente se tornam menos importantes, à medida que avançamos no tempo em relação a essa transação. Daqui a três anos, que importância terá saber que cliente Y comprou um saco de batata, no dia 26 de janeiro? Por outro lado, se estivermos tentando fazer uma estimativa de demanda de inverno para batatas, poderia ser importante saber o quanto desse produto foi vendido no total, durante o mês de agosto.

5 O 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 KDD é composta por três tipos de componentes: o problema em que será aplicado o processo do 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. Segue o detalhe e comentário sobre cada um desses componentes.

- O *problema* a ser submetido ao processo do 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 intencional e extensional (DATE, 2000). O intencional refere-se à estrutura ou 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. No geral, o processo de KDD deduz que os dados serão organizados em uma estrutura tabular bidimensional contendo casos e características do problema a ser analisado. Para o processo de KDD, não requer que os dados que serão analisados estejam em um Data Warehouses, contudo o tratamento e a consolidação dos dados nesse tipo de ambiente é extremamente útil.
 - O *especialista no domínio da aplicação* representa a pessoa ou o grupo de pessoa que conhece o assunto e o ambiente em que deverá ser realizada. Geralmente eles detêm o conhecimento prévio sobre o problema ("*background knowledge*").
 - Os *objetivos da aplicação* compreendem as características esperadas do modelo de conhecimento a ser produzido ao final do processo. Em geral, os objetivos de uma aplicação são definidos em função da opinião dos especialistas no domínio desta aplicação são definidos destas aplicações de KDD em que os

objetivos não estejam bem definidos no início do processo (ENGELS, 1996; ENGELS et al., 1997). Tal objetivo deve ser refinado ao longo do processo de KDD em função dos resultados intermediários obtidos.

- Os *recursos disponíveis* para a solução do problema mencionado.
 - O *especialista* em KDD representa a pessoa ou grupo de pessoa que possui experiência na execução de processo KDD.
 - 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, o mesmo indica para recursos computacionais de *hardware*. Quanto maior a capacidade de processamento e memória da plataforma computacional, maior será a agilidade para obter resultados.
- Os *resultados obtidos* a partir da aplicação dos recursos no problema. Compreender, 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 descreve algum conjunto de dados” (FAYYAD et al., 1996). Todo o modelo de conhecimento deverá ser avaliado com relação ao cumprimento das expectativas definidas nos objetivos da aplicação. É muito comum que, durante o processo KDD, sejam realizadas comparações entre os modelos de conhecimento obtidos.

- Os *históricos* sobre como os modelos de conhecimentos foram gerados também e se enquadram como resultados do processo, pois permitem uma análise crítica e uma revisão das ações realizadas.

Conforme já foi mencionado na introdução a Descoberta de Conhecimento em Bases de Dados é caracterizada por um processo que é dividido em três partes: **Pré-processamento, Mineração de Dados e Pós-processamento.**

A etapa de Pré-processamento abrange as funções relacionadas à captação, à organização e o tratamento de dados. Essa etapa tem como objetivo a preparação para a mineração de dados.

- Seleção de Dados – Essa função, também conhecida como Redução de Dados compreende, em essência a identificação de quais informações, dentre as bases de dados existentes, devem ser efetivamente consideradas durante o processo KDD.
- Limpeza dos Dados – Abrange qualquer tratamento realizado selecionado de forma assegurar a qualidade dos fatos por eles representados. Um exemplo simples de limpeza de dados seria a definição de um intervalo de possíveis valores para um determinado atributo.
- Codificação dos Dados – Essa função tem como objetivo de deixar codificado para ficarem numa forma que possam ser usados como entrada dos algoritmos de Mineração de Dados.
- Enriquecimento dos Dados – Essa função 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.

Na etapa de Mineração de Dados é realizada a busca efetiva por conhecimento útil no contexto da aplicação. É a principal etapa do KDD, alguns autores referem à Mineração de Dados e à Descoberta de Conhecimento como sinônimos.

Na mineração de dados são executados 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), Clusterização são exemplos de técnicas que podem ser utilizadas na etapa de Mineração de Dados.

- *Descoberta de Associação:* Abrange a busca por itens que freqüentemente ocorram de forma simultânea em transações do banco de dados. Um exemplo 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 de televisão.
- *Classificação:* Consiste em descobrir uma função que mapeie um conjunto de registro 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. Com 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 que desejam 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 tecnologia que podem ser aplicadas na tarefa de classificação (MICHEI et al., 1994).

- *Regressão*: É 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 atributo numérico. Com exemplo de aplicação 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 de limite do cartão de crédito para cada cliente em um banco; dentre outros. Estatísticas Redes Neurais, dentre outras áreas, oferecem ferramentas para implementação da tarefa de regressão.

- *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 a propriedade comuns que os distingam de elementos em outros clusters. O objetivo nessa tarefa é maximizar similaridade intercluster. Diferente da tarefa de classificação, que tem rótulos predefinidos, a clusterização precisa automaticamente identificar os grupos de dados aos qual o usuário deverá atribuir rótulos (FAYYAD et al., 1996a).

- *Sumarização*: Essa tarefa, muito comum em KDD, consiste em procurar identificar características comuns entre conjuntos de dados (WEISS & INDURKHYA, 1998).

- *Detecção de Desvios*: Essa tarefa consiste em procurar identificar registros do banco de dados com as características não atendam aos padrões considerados normais no contexto (Weiss & Indurkhyia, 1998).

- *Descobertas de Seqüências*: 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 compradas ao longo do tempo.

A expressão *Operação de KDD* se refere a qualquer função das etapas operacionais do KDD. Como operações de KDD, podem ser citadas: Seleção de Dados, Limpeza de Dados, Codificação, Descoberta de Associação, Regressão, Simplificação de Modelos de Conhecimentos, dentre inúmeras outras.

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

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 tarefa de Classificação.
- O Método *NFHB-Class* (GONÇALVES 2001) baseia-se na combinação de princípios da Teoria da Lógica Nebulosa e Redes Neurais e pode ser aplicado em operações de classificação.
- O Método *Rule Envolver* (LOPES et al., 1999) foi concebido a partir da Teoria de *Algoritmos Genéticos* (DAVIS 1999) e pode ser aplicado em operações de Classificação e Sumarização (FAYYAD et al., 1996a).

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

- *Técnicas Tradicionais*: São tecnologias, que existem independentes do contexto da Mineração de Dados. Em geral, produzem bons resultados também em aplicações de KDD. Como exemplo de tecnologias tradicionais pode ser citado

- *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 redes tentam aprender padrões diretamente a partir dos dados através de um processo de repetidas apresentações dos dados à rede, ou seja, por experiência.
 - *Lógica Nebulosa (Foz Lógica)*: 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 nas avaliações de conceitos.
 - *Algoritmos Genéticos (AG)*: São modelos de otimização, inspirados na evolução natural e na genética aplicados a problemas complexos de otimização.
 - *Estatísticas*: Fornece diversos modelos e técnicas tradicionais para análise e interpretação de dado. Por exemplo: Redes Bayesianas, Análise Discriminante, Análise Exploratória de dados, entre outros (MICHIE et al, 1994).
- *Técnicas Específicas*: São técnicas desenvolvidas especificamente para aplicação em tarefas 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, ParMaxExclat, dentre outros, foram originados a partir do Apriori.
 - *Técnicas Híbridas*: Convém mencionar que técnicas podem ser combinadas de forma a gerar os chamados híbridos. Há várias formas de se definir sistemas híbridos. De uma forma simples, sistemas híbridos são aqueles que utilizam mais e 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.

- Híbrido Seqüencial – Neste modelo, um sistema com paradigma 1 atua como entrada de outro sistema com paradigma 2, como ilustrado na Figura 5.1. 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.

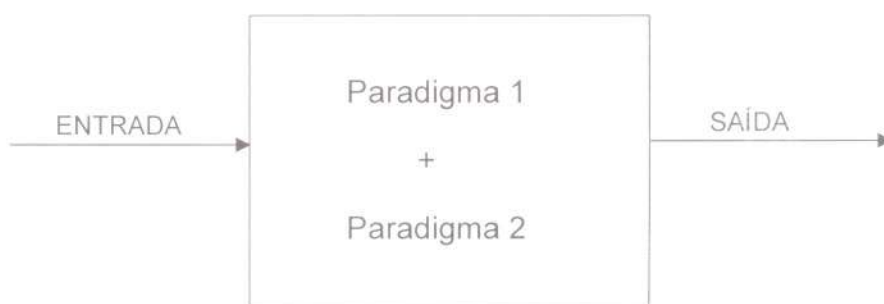


Figura 9: Híbrido Seqüencial

- Híbrido Auxiliar – Neste 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 5.2 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. Neste caso, tem-se um maior grau de hibridização em comparação com o híbrido seqüencial.

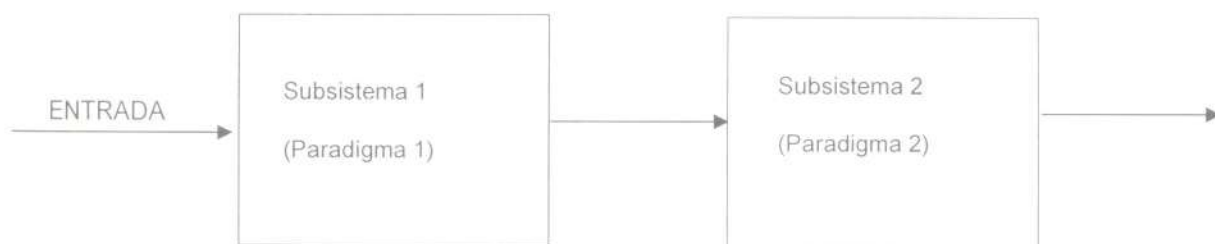


Figura 10. Modelo Híbrido Auxiliar

- 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 5.3 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.

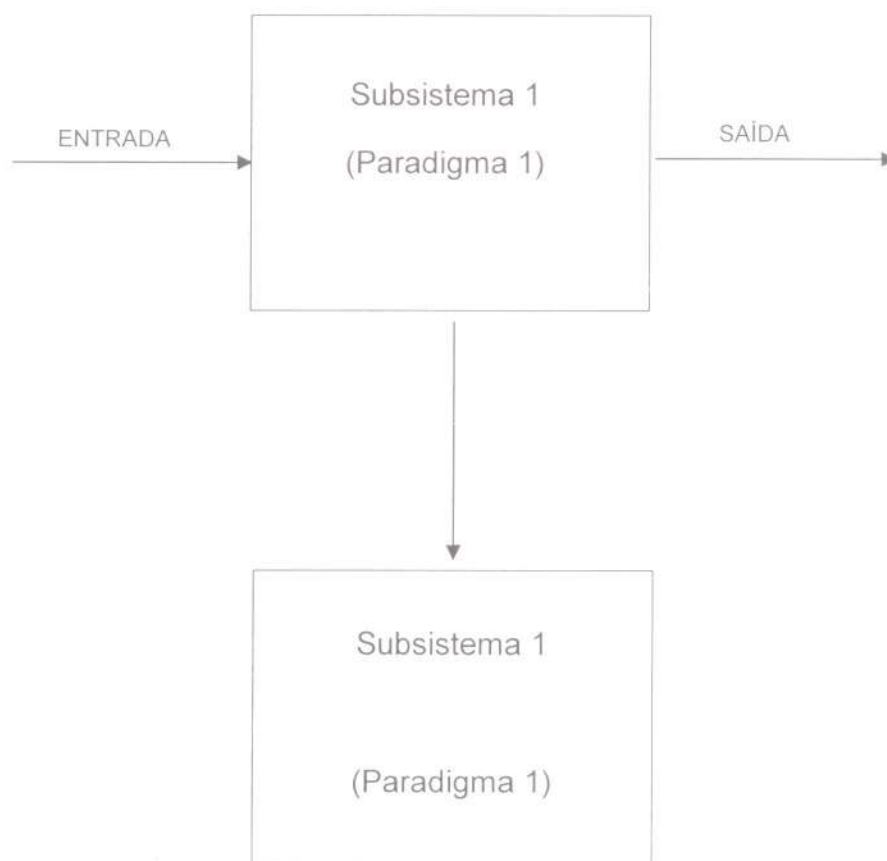


Figura 11: Modelo Híbrido Incorporado

6 MINERAÇÃO DE DADOS

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

A execução da etapa de Mineração de Dados compreende a aplicação de algoritmos sobre os dados procurando abstrair conhecimento. Conforme já foi informando estes algoritmos são fundamentados em técnicas que procuram, segundo determinados paradigmas explorar os dados de forma a produzir modelos de conhecimento.

Todo *conjunto de dados* no processo de KDD corresponde a uma *base de fatos* ocorridos que devem ser interpretados como um *conjunto de pontos* e um *hiperespaco de dimensão K*. A dimensão da base de fatos é determinada pelo número de atributos do conjunto de dados em análise. A figura 6.1 mostra o exemplo no contexto da análise de crédito na quais três informações estão representadas em um plano cartesiano. Os eixos correspondem aos atributos *Renda* e *Despesas*. Cada *ponto representa um caso*. O símbolo associado a cada caso ("círculos" e "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 KDD* compreendem a definição da *tarefa de KDD* a ser executada e da *expectativa* que os conhecedores 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.

O tipo de padrão desejado em nosso exemplo é uma função que separe da melhor forma possível à classe dos clientes negligentes da classe dos clientes não negligentes. São finitas as possibilidades de funções, também denominadas *hipóteses*, que tenha como objetivo separar os dois conjuntos. Tais hipóteses pertencem a um *espaço de padrões*, normalmente infinito, que deve ser percorrido pelo algoritmo de Mineração de Dados em busca de elementos que atendam a determinadas 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 de 5% dos casos analisados.

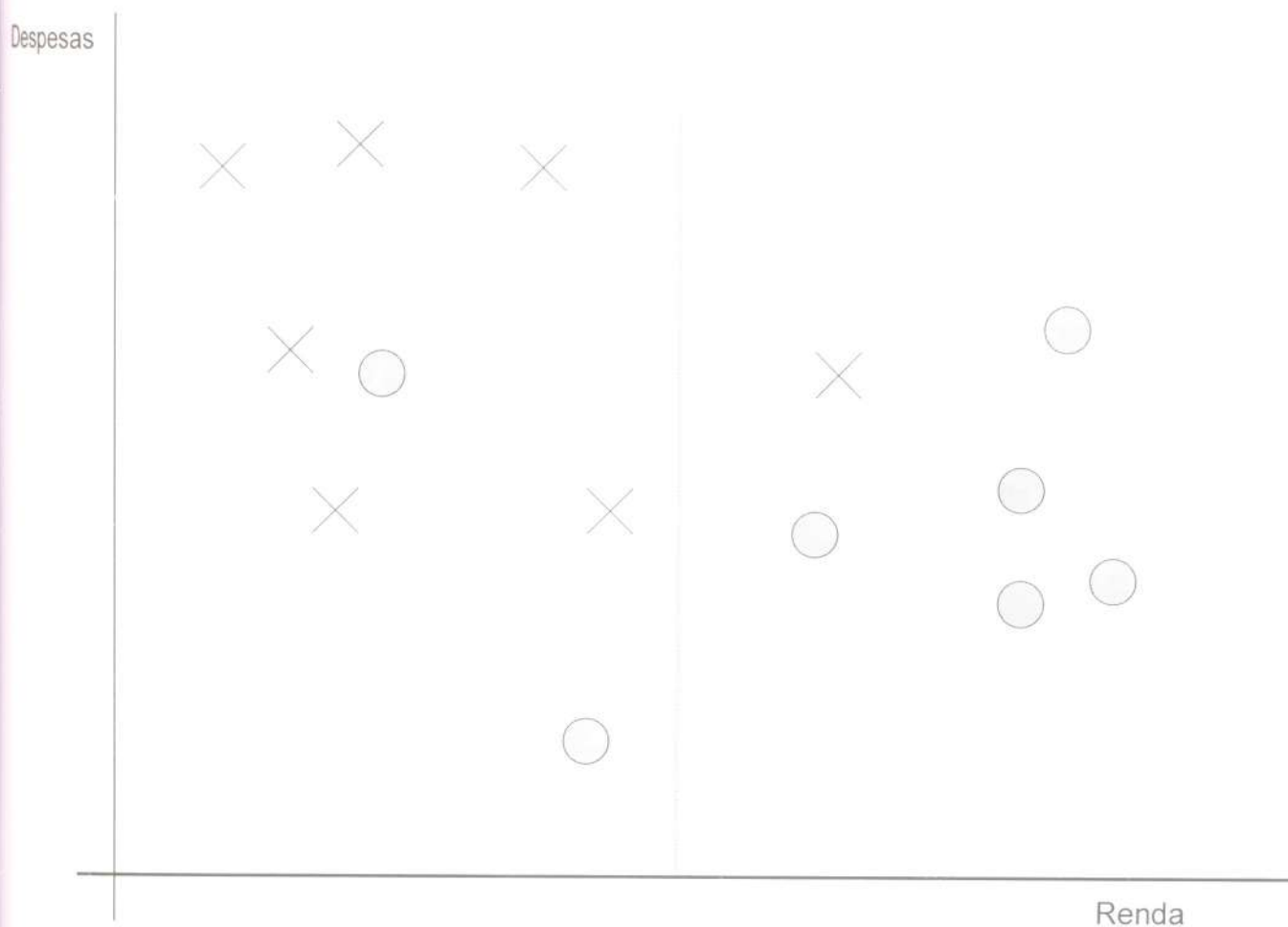


Figura 12: Base de fatos de uma financeira Hiptética

O conceito de medida de interesse é essencial ao processo de KDD por dois motivos principais:

a) Medidas de Interesse pode ser usada após a etapa de Mineração de Dados (etapa de pós-processamento) a fim de ordenar ou filtra 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: *medida de interesse objetiva* e *medidas de interesse subjetivas*.

A medida de interesse objetiva 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.

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 (conhecimento não esperado), contradições (conhecimento 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 confirme suspeitas dos especialistas em determinados temas. A avaliação envolve esse 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 como *Back-Propagation* e *K-NN* devem ser descartados diante desse tipo de expectativa.

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

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

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

7 ALGUNS ALGORITMOS

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.

Um outro conceito muito importante envolvido no processo de KDD é mais especificamente na etapa de 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 compreende 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.

No aprendizado não supervisionado não existe a informação da saída desejada. Os algoritmos partem dos dados, procurando estabelecer relacionamento entre eles. Com exemplos clássicos de algoritmos que utilizam aprendizado supervisionado estão o *K-Means* e o *Apriori*.

8 CLUSTERIZAÇÃO

8.1 VISAO GERAL

A tarefa de *Clusterização*, também chamada de *Agrupamento*, é usada para particionar os registros de uma base de dados em subconjuntos ou clusters, de tal forma que elementos em que um cluster compartilhe um conjunto de propriedades comuns que os distingam dos elementos dos outros clusters. O objetivo desta tarefa é máxima 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 básica 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 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 cluster. 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 objetivos 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 uma matriz $n \times p$ a matriz abaixo.

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

- Matriz de similaridade – Cada elemento da matriz representa uma distância entre pares de objetos. Visto que a 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, obtêm-se uma matriz quadrada de tamanho $n \times n$ como a que segue.

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

O ponto $d(i, j)$ representa a distância ou similaridade entre o objetivo 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 citado: K-Means, Fuzzy K-Means, K-Modes e K-Meodid.

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

- Descobrir clusters com forma arbitrária – A forma de clusters, considerando o espaço euclidiano, pode ser esférica, linear, alongada, elíptica, cilíndrica, espiralada etc. Os métodos de clusterização baseados nas medidas de distâncias Euclidiana 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 compreensíveis.
- 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 de 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, compreensíveis e utilizáveis. Assim, é importante que os algoritmos utilizem representações simples.
- Ser robusto na presença de ruídos – As maiorias da base de dados do mundo real contêm 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 frequentemente, 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 as 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 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 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 de 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 encontra mais próximo ao centro da 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 distancia total calculada.

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

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

1. 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 distancia entre cada par de clusters. Essas distancias são, geralmente, armazenadas em uma matriz de similaridade. Então, são escolhidos dois clusters com 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 de hierarquia), ou até que uma condição de término ocorra (por exemplo, o número de clusters desejado tenha sido alcançado).

2. 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.

8.2 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:

1. 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.
2. 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 modelos de conhecimentos 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 manipulações dos conjuntos de dados original e final.

8.3 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:

1. 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 forme os clusters gerados.
2. Cada novo conjunto de dados é então apresentado a um algoritmo de sumarização que descreve as principais características dos registros envolvidos naquele conjunto.

9 ALGORITMOS DE CLUSTERIZAÇÃO

9.1 K-MEANS

O algoritmo *k-means* é um método popular da tarefa de Clusterização. Tomam-se, 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 de 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 devido um conjunto de n objetos em k cluster 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, baseando na distância entre o objeto e a média do cluster. A partir de então, o algoritmo computa as novas médias de um cluster. Este processo se repete até que uma condição de parada seja atingida.

O funcionamento do algoritmo *k-means* encontra-se resumido na figura 9.1. O algoritmo tenta determinar k partições que minimizem a função do erro quadrado. “Apresentar 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 cluster, 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 do cluster.

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

O algoritmo *k-means* é inicializado com os centros (médias) colocados em posições aleatórios. 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.

9.2 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 frequência) dos objetos, usando medidas de similaridade para tratar objetos categóricos, e usando método baseados em frequência para atualizar as modas dos clusters.

9.3 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.

9.4 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* 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.

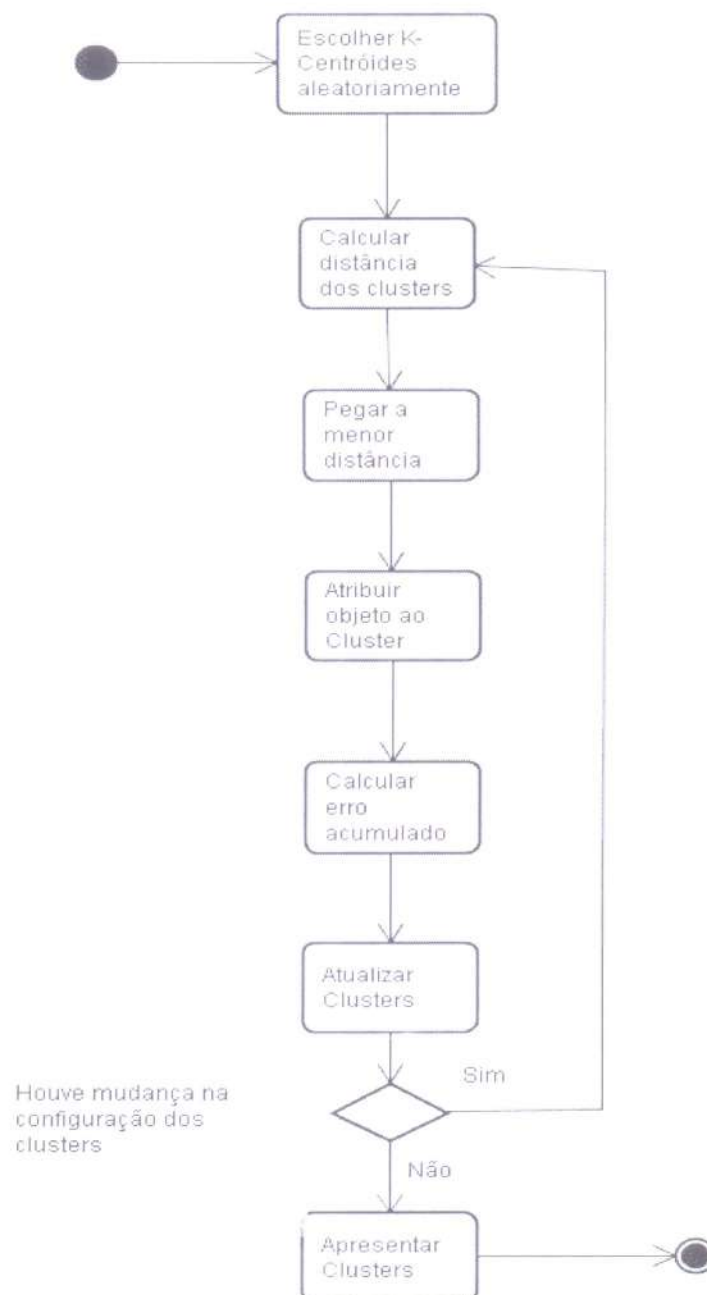


Figura 13: Algoritmo K-MEANS

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.

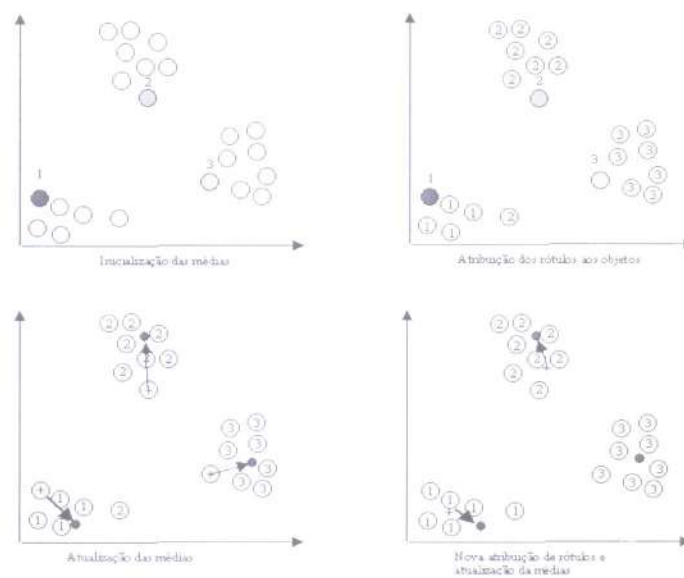


Figura 14: Passos Subseqüentes do Algoritmo

10 JAVA DATA MINING API

10.1 MINERE SEUS DADOS COM A JDM, EXPLORE A API JAVA DATA MINING

A API Java Data Mining (JDM) é a primeira tentativa de criação de um padrão *Standard* que permita às aplicações usarem ferramentas de forma independente do motor do datamining. A JDM promete trazer para a Mineração de Dados o que o JDBC trouxe para o acesso a Base de Dados e fazer da Mineração de Dados uma ferramenta útil para as empresas.

Consultas e navegação representam as principais maneiras de interagir com os dados hoje. Uma aplicação de consulta que utiliza expressão de linguagem de consulta como SQL ou XPATH, executando em um conjunto de dados, irá produzir um subconjunto de dados.

Data mining *"é o processo para encontrar padrões e relacionamentos nos dados, e usar esse conhecimento para classificar novos dados ou ganhar introspecções"*, diz Mark Hornick, Oracle Senior Manager para Data Mining Technologies e SPEC para o Java Data Mining API ou JDM (JSRs 73 and 247).

Até recentemente os dados de mineração esteve sobre o domínio de especialistas altamente qualificados. Ferramentas de Data Mining eram tradicionalmente construídos em empresas ou em produtos fornecidos com interfaces específicas. Com resultado aplicações Java, não tinha qualquer maneira de interagir com softwares de Data Mining. A situação foi semelhante para acessar base de dados com JDBC padrão: Cada aplicação teve que confiar na API dos produtos específicos para o acesso dos dados. A API Java Data Mining (JDM) define como irá interagir com sistemas de Mineração de Dados. *"JDM é compreendido melhor como o JDBC para mineração de dados"* o *"JDBC é uma API muito popular e nós gostaríamos de ver o JDM similarmente igual a ele, utilizando para executar mineração de dados"* diz Hornick da Oracle.

Por que a mineração de dados é um jogo de atividades complexas a especificação inicial da JDM surgiu apenas em 2004 e depois dessa surgiu uma nova

versão que define algumas características que foram deixadas de fora da especificação anterior. JDM 2.0(JSR 247) foi aceito para a JCP em Junho de 2007, e que visa definir muitos recursos adicionais, ficaram de fora da especificação 1.0.

Será explicado neste capítulo o conceito do Data Mining na JDM, e será um código com exemplo, explicando o uso à utilização da JDM.

10.2 Transformando Dados em Conhecimento

Um dos principais objetivos da mineração de dados é/ou para ajudar a entender o passado, ou tentar prever o futuro com base nos dados passados. Para esses fins, as técnicas da mineração de dados ajuda a identificar padrões de um grande armazém de dados e com isso constrói modelos de dados concisos. Tais modelos capturam as características essências dos dados subjacentes, ajudando os seres humanos a ganhar novas perspectivas no conhecimento baseado nesses dados. A mineração de dados difere de outros mecanismos de acesso a dados, tanto em processo e técnica. *"Quando você formular teorias e as testam isso é o raciocínio dedutivo. O Data mining é o contrario ele é indutivo"* Hornick da Oracle. *"É freqüente ouvir pessoas dizerem Eu minerei os dados. Mas eles estão fazendo apenas buscas. Mesmo uma complexa consulta SQL executando em uma grande base de dados é apenas uma extração de dados um resumo detalhado. OLAP (Processo Analítico Online) permite-lhe fazer cortes em cubos de dados, mas isso ainda não é mineração de dados. Dados minerados dão a capacidade de olhar para o futuro, e fazer previsões, para extrair informações de um indivíduo que não teria como discernir por causa da complexidade da quantidade de dados"* Hornick da Oracle.

Um sistema de mineração de dados consiste em um motor de mineração de dados e em um repositório que persiste a mineração de dados em artefatos, tais como os modelos criados no processo. Uma das principais vantagens na API JDM é que ele abstrai para fora dos componentes, das tarefas, e até mesmo algoritmos de um sistema de mineração de dados.

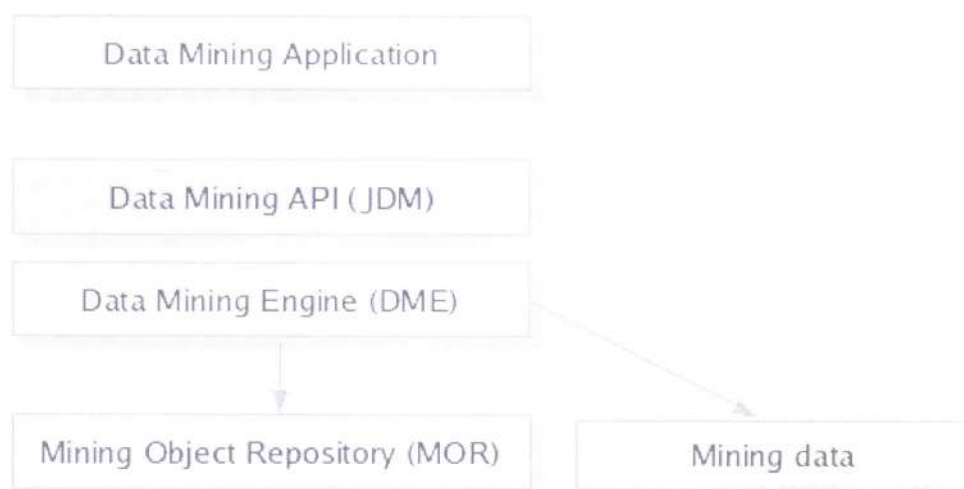


Figura 15: Componentes de um sistema de Data Mining.

Construir um modelo de mineração de dados começa tipicamente com a identificação de padrões recorrentes nos dados, e em seguida, sintetizar os padrões de uma forma que lhes permite comunicar aos seres humanos ou outras máquinas. Modelos podem assumir a forma de uma representação gráfica, um conjunto equações, uma rede neural, ou mesmo um conjunto de regras. Os modelos podem ser aplicados aos novos dados, ou avaliado ou aperfeiçoado na presença de cada vez maiores conjuntos de dados. O processo pode ser resumido da seguinte forma:

Decida o que você vai querer aprender. A primeira, e mais importante passo é decidir que tipo de novos conhecimentos ou introspecção que quiser ganhar com os dados. Quanto mais específico for, maior será a probabilidade de ter um bem sucedido processo de mineração de dados. Por exemplo, querer descobrir quais as promoções irá dispor para aqueles feriados, com isso poderá criar estratégias de vendas para futuras vendas. Ou, pode-se querer prever que os clientes são susceptíveis para comprar um produto que está preste a introduzir a venda. Em outro exemplo, pode querer identificar ponteciais galáxias de um céu.

Selecione e prepare seus dados. Assim que estiver decidido sobre o objetivo, é necessário identificar os dados que poderá ajudar a alcançar esses objetivos. Por exemplo, os registros das vendas e as listas de anúncios podem ser relevantes a decidir-se o que os fatores contribuíram para o sucesso das vendas. Conjuntos de dados relevantes são raramente no formato adequado para a mineração de dados. Frequentemente tem que transformar os dados, possivelmente limpeza de registros incompletos.

Escolha e configure as tarefas que serão mineradas. Em seguida, deve decidir sobre como será a mineração de dados e executar essa tarefa. Após ter selecionado terá, em seguida, de configurar essa tarefa com parâmetros adequados. No JDM API, essa configuração é especificada com configurações.

Selecione e configure os algoritmos de mineração. Definições permitem-lhe selecionar algoritmos para uma tarefa de mineração. Muitos algoritmos de mineração de dados estão disponíveis para uma determinada tarefa. Algoritmos diferem não só na precisão dos seus finais de produto, mas também nos recursos computacionais que necessitam. Muitas ferramentas de mineração de dados pode combinar automaticamente algoritmos a um objetivo desejado.

Crie seu modelo de Mineração de Dados. A saída de execução de uma mineração de dados é a tarefa do modelo de mineração de dados: Esse modelo de preferência, é uma representação de seus dados adequados a seu objetivo. Por exemplo, o modelo poderá ser uma rede neural, uma decisão de árvore, ou mesmo um conjunto de regras compreensível pelos humanos.

Teste e refine seus modelos. Pode criar diversos modelos, avaliar a exatidão de cada modelo com dados passados, e escolher o "melhor"

modelo para esse fim. Uma forma de avaliar os modelos deve aplicar a introspecção recentemente ganha aos dados passados e compara com os resultados que seriam obtidos sem auxílio da visão, por exemplo, por amostragem aleatória. Idealmente sua introspecção recentemente ganha deve produzir melhores resultados.

Relatório de Conclusão ou Previsões para o Futuro. Finalmente, você poderá relatar ou utilizar a mineração de dados para prever modelos do futuro.

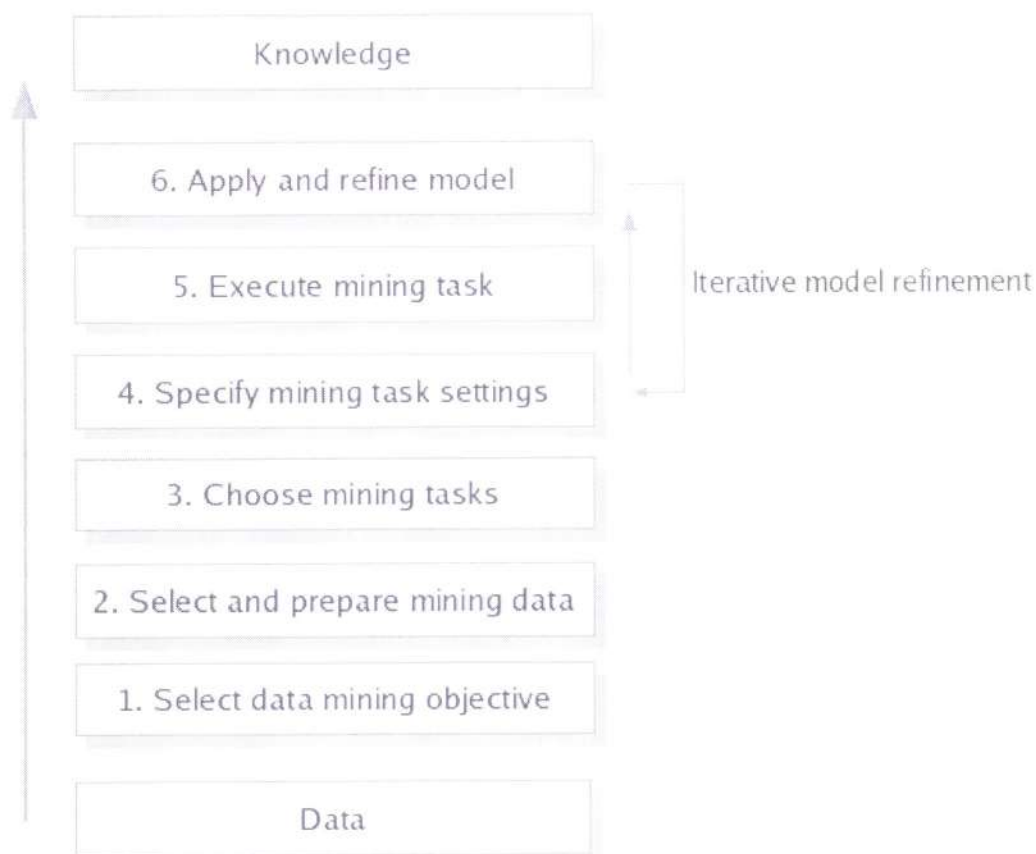


Figura 16: Passos da Mineração de Dados

A tendência atual é para automatizar tanto quanto possível deste processo. *“Mesmo aqueles que não são especialistas em mineração de dados possam colher pela tecnologia de mineração de dados”* Hornick da Oracle. *“Na norma da JDM os usuários podem determinar ajustes automáticos”*.

10.3 Usando a API JDM

Um típico projeto de mineração de dados poderia começar com um analista de dados, alguém muito familiarizado com o problema dominando JDM usando a explorando o melhor modelo a ser usado. O analista poderia persistir os dados em um repositório minerando os dados (MOR) que depois passarão a fazer parte das especificações que o analista poderá usar para implantar uma solução de mineração de dados. As especificações podem incluir outros artefatos, como a pontuação ou algoritmos a usar.

Para criar um modelo mineração de dados, um analista ou desenvolvedor iria seguir algumas etapas simples para que o processo descrito nesse capítulo para JDM interações.

- Identificar os dados que queira usar para construir seu modelo de sua construção dados com uma URL que aponta para os dados.
- Especificar o tipo de modelo que deseja construir, e os parâmetros para construir o processo. Esses parâmetros são denominados, construir configurações no JDM. O mais importante é construir definindo a definição da tarefa de mineração de dados, tais como clustering, classificação, associação ou regras. Além disso, também pode especificar os algoritmos para utilização em tais tarefas. Se não se importam com um determinado algoritmo, a mineração de dados motor pode selecionar um nome, com base nas tarefas desejadas.
- Opcionalmente, pode desejar criar uma representação lógica dos dados. Isso permite, por exemplo, selecionar determinados atributos dos dados físicos e, em seguida, esses atributos mapeiam para valores lógicos.

- Outra etapa opcional é permitir especificar os parâmetros para a sua mineração de dados bem como algoritmos.
- Em seguida, você cria uma tarefa da configuração, e aplica a essa tarefa as referências de dados físicas e os ajustes da configuração.
- O JDM API permite a você verificar as configurações antes de executar uma tarefa. Isso permite pegar erros precoces, uma vez que tarefas podem ser publicadas por um longo tempo.
- Finalmente, executa-se a tarefa. O resultado é que a execução do modelo de dados. Esse modelo terá uma assinatura de um tipo de interface, que descreve a possível entrada de atributos para posterior aplicação do modelo de dados adicionais.

Depois de você ter criado o modelo, você poderá testar esse modelo, e em seguida e então aplicar aos dados adicionais. Construção, testes e aplicação ao modelo de dados adicionais é um processo iterativo. Estes modelos podem então ser salvos no MOR, e utilizado tanto para explicar os dados, ou para prever o resultado de novos dados em relação ao seu objetivo.

O seguinte código de exemplo, com base no JDM 1,0 especificação, cria um modelo que prevê que os clientes teriam compras de um determinado produto. Pode-se construir um modelo com aqueles dados para ajudar a explicar como certos clientes ou atributos da compra impacta a probabilidade de um cliente nas compras dos produtos. Poderia usar depois que o modelo para prever a probabilidade de novas compras.

Suponha um atributo do cliente, *compre*, indique se um cliente comprou o produto, com os dois valores possíveis: Y e N. Atribuir que correspondem a tais valores discretos são denominados atributos categóricos.

Dizem que de um determinado conjunto de categorias um item pertence a (Compradores e não-compradores, neste caso).

Para os clientes futuros, nosso modelo irá prever o valor do atributo da compra. Ou seja, o objetivo dessa mineração dos dados é criar um modelo que classifique os clientes baseados no valor do atributo da compra. A tarefa da classificação tem um alvo, que é o atributo que tentamos prever (compra, neste caso). Todos atributos restantes do cliente serão usados pela classificação como preditos desse alvo.

Os dados minerados para esta tarefa podem ser situados em um arquivo, ou em uma tabela da base de dados. Em ambos os casos, o sistema de mineração de dados necessita traçar os atributos dos dados físicos aos atributos lógicos. Para o exemplo, o atributo da compra pode ser etiquetado como "hasPurchased" na série de dados de física real

Você pode especificar tais mapeamentos através das definições fornecidas à tarefa de classificação. O código abaixo mostra o mapeamento físico de dados lógico, valores, e também como especificar mineração de dados tarefa configurações.

O exemplo assume que você já tenha obtido uma ligação a um motor de mineração de dados, talvez através de uma chamada JNDI. Uma conexão de JDM é representada pela variável do motor, que é do tipo `javax.datamining.resource.connection`. As conexões de JDM são muito similares às conexões de JDBC. Após ter obtido a ligação ao motor de mineração de dados, o próximo passo é definir os dados físicos que serão minerados. A construção de dados é referenciada via um objeto `PhysicalDataSet`, o que, por sua vez, carrega os dados de um arquivo ou uma tabela de banco de dados, com uma URL referenciado. As especificações da JDM definiram os tipos de dados aceitáveis e formato do arquivo de entrada. Você poderia usar um modelo `PhysicalDataSet` para ambos os edifícios e posteriormente para testar e avaliar o seu modelo:

A fonte desse exemplo foi retirada do site e o mesmo encontra-se nas referências.

```
PhysicalDataSetFactory dataSetFactory
    = (PhysicalDataSetFactory) engine.getFactory("javax.datamining.data.PhysicalDataSet");
PhysicalDataSet dataSet = pdsFactory.create("file:///export/data/textFileData.data", true);
engine.saveObject("buildData", dataSet, false);
```

Baseado nos dados físicos pode-se definir um modelo lógico dos dados. Neste exemplo, especifica-se que a compra é um atributo categórico dos dados:

```
LogicalDataFactory logicalFactory
    = (LogicalDataFactory) engine.getFactory("javax.datamining.data.LogicalData");
LogicalData logicalData = logicalFactory.create(dataSet);
LogicalAttributeFactory logicalAttributeFactory = (LogicalAttributeFactory)
    engine.getFactory("javax.datamining.data.LogicalAttribute");
LogicalAttribute purchase = logicalData.getAttribute("purchase");
purchase.setAttributeType(AttributeType.categorical);
engine.saveObject("logicalData", logicalData, false);
```

Em seguida, prossegue-se para especificar as definições para a construção do modelo. Desde a mineração a tarefa que se pretende realizar é a classificação, e criar as definições para classificações, incluindo especificando Bayesian ingênuo como o algoritmo. Este exemplo foi o único atributo alvo de aquisição. Note-se que o próprio algoritmo aceita configurações através de um algoritmo de configurações específico objeto:

```
ClassificationSettingsFactory settingsFactory = (ClassificationSettingsFactory)
    engine.getFactory("javax.datamining.supervised.classification.ClassificationSettings");
ClassificationSettings settings = settingsFactory.create();
settings.setTargetAttributeName("purchase");
NaiveBayesSettingsFactory bayesianFactory = (NaiveBayesSettingsFactory)
    engine.getFactory("javax.datamining.algorithm.naivebayes.NaiveBayesSettings");
NaiveBayesSettings bayesSettings = bayesianFactory.create();
bayesSettings.setSingletonThreshold(.01);
bayesSettings.setPairwiseThreshold(.01);
settings.setAlgorithmSettings(bayesSettings);
engine.saveObject("bayesianSettings", settings, false);
```

Após ter especificado as definições, criamos uma tarefa construir. Como um passo opcional, o JDM API permite a você verificar as configurações antes de começar a construir o modelo. Não iremos tratar verificação erros neste exemplo:

```
BuildTaskFactory buildTaskFactory =
    (BuildTaskFactory) engine.getFactory
    ("javax.datamining.task.BuildTask");
BuildTask buildTask = buildTaskFactory.create("buildData", "bayesianSettings", "model");
VerificationReport report = buildTask.verify();
if( report != null ) {
    ReportType reportType = report.getReportType();
    //MANIPULA ERROS AQUI
}

//SE NÃO TIVER ERROS SALVA A TAREFA
engine.saveObject("buildTask", buildTask, false);

//EXECUTE O CONSTRUTOR DA TAREFA
ExecutionHandle handle = engine.execute("buildTask");

//ESPERAR PELA CONCLUSÃO
handle.waitForCompletion(Integer.MAX_VALUE);
```

Finalmente, podemos acessar o modelo resultante:

```
ExecutionStatus status = handle.getLatestStatus();
if (ExecutionState.success.equals(status.getState())) {
    ClassificationModel model =
    (ClassificationModel) engine.retrieveObject( "model", NamedObject.model );
}
```

As características novas propostas para JDM 2.0 incluem potencialidades de mineração de tempo das séries de dados, o que é útil na previsão e detecção anomalia, como em segurança.

Enquanto o JDM 1,0 API tem sido uma norma aprovada por quase um ano, apenas um punhado de produtos implementar porções da norma no momento da redação deste texto. *"Oracle tem um produto baseado no JDM 1,0. Existem outros vendedores também.. Comercial Muitos vendedores estão à espera de encontrar à margem da procura de mineração de dados, e então implementar [JDM] nos seus próprios produtos", diz Hornick.*

11 CONCLUSÃO

Considerando os avanços e a relevância que a área da Descoberta de Conhecimento em Bases de Dados (Knowledge Discovery in DataBases - KDD) vem ganhando campo no contexto mundial. 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 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 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.

12 REFERÊNCIAS

ACM Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD), <http://www.acm.org/sigs/sigkdd/>

AGRAWAL, R.; IMIELINSKI, T., SWAMI, Mining Association Rules Between Sets of Items in Learning. São Francisco: Morgan Kaufmann, 1992.

AGENTES Inteligentes e KDD, 2007. Disponível em: <<http://blog.dimensaozero.com/?tag=mineracao-de-dados>>. Acessado: 02/12/2007

CALANTONIO, L. M. Novas Metodologias para Clusterização de Dados. Dissertação de Mestrado, Engenharia Civil, COPPE, Universidade Federal do Rio de Janeiro, 2001.

CORRÊA, H.L.; GIANESI, I.G.N.; CAON, M. (1997). Planejamento, programação e controle da produção: MRP II / ERP: conceitos, uso e implantação. São Paulo: Atlas. (Disponível na biblioteca da EP - USP)

DATE, C. J. Introdução aos Sistemas de Banco de Dados. Campus 2000.

DAVENPORT, T.H. (1998). Putting the enterprise into the enterprise system. Harvard Business Review. Julho-Agosto, p.121-131. (t: 827).

DAVIS, L. Handbook of Genetic Algorithms. VNR Comp. Library, 1990.

ENGELS, R. Planning task for knowledge Discovery in database: Performing Task-Oriented User-Guidanc. Proceedings of the Third International Conference on Knowledge Discovery in Databases. Newport Beach, 1997.

FAYYAD U.M; PIATETSKY-SHAPIO, G; SMITH, From Data Mining to Knowledge Discovery: An Overview. Knowledge Discovery and Data Mining, Menlo Park: AAAI Press, 1996.

FONÇALVES, L. B.; PACHECO, M. Clusterização de Banco de Dados Benchmark Utilizando Algoritmos Genéticos. Relatório Técnicos de Estudo Dirigido. Rio de Janeiro, 2000.

GOLDSCHMIT, R.; PASSOS, E. Utilização de Recursos de Banco de Dados Relacionais em Tarefa de Mineração de Regras Associativa. I Congresso de Lógica Aplicada a Tecnologia. São Paulo: Senac, 2002.

GOLDSCHMIT, R.; PASSOS, E. Data Mining Um Guia Prático. EDITORA CAMPUS. Rio de Janeiro, 2006

HAN, J.; KEMBER, M. Data Mining: Concepts and Techniques. San Francisco: Morgan Kaufmann Publishers, 2001.

HAYKIN, S. Neural Network: a Comprehensive Foundation. Prentice Hall, 1999.

MINE Your Own Data with the JDM API, 2005. Disponível em: <http://www.artima.com/lejava/articles/data_mining3.html>. Acessado em 01/02;2008

JAVA DATA MINING API 1.0, JSR 73, <http://www.jcp.org/en/jsr/detail?id=73>

JAVA DATA MINING API 2.0, JSR 247, <http://www.jcp.org/en/jsr/detail?id=247>

KDnuggets (data mining portal), <http://www.kdnuggets.com/>

KELLER, G.; TEUFEL, T. (1998). SAP R/3 process oriented implementation: Iterative Process Prototyping. Addison-Wesley.

KIRCHMER, M. (1998). Business process oriented implementation of standard software. Springer-Verlag.

LAUDO, KENNETH C, (2004). Sistemas de Informações gerenciais : administrando a empresa digital. São Paulo: Prentice Hall, p61.

LAUDON, K. C.; LAUDON, J. P. (1998). Management information systems. Upper Saddle River: Prentice Hall. (Disponível na biblioteca da FEA - USP).

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

Micheal Lesk, How Much Data is There in the World?, <http://www.lesk.com/mlesk/ksg97/ksg.html>

MICHIE, D.; SPIEGELHALTER, D; TAYLOR, C. Machine Learning, Neural and Statistical Classifications. Ellis Horwood, 1994.

MINERAÇÃO de Dados, Disponível < http://pt.wikipedia.org/wiki/Minera%C3%A7%C3%A3o_de_dados> .

MINERAÇÃO de Dados e culpados por associação, 2007. Disponível em: < <http://dispositivodevisibilidade.blogspot.com/2007/11/minerao-de-dados-e-culpados-por.html>>

MyLifeBits project, <http://www.research.microsoft.com/barc/MediaPresence/MyLifeBits.aspx>

NALIATO, F. C. Data Mining: Tecnicas e Aplicações para Marketing Direto. Editora Berkeley, 2001.

PADOVOZE, CLÓVIS LUIS (2004). Sistemas de informações contábeis: fundamentos e análise. São Paulo: Atlas, p68.

PIATETSKZ-SHAPIRO, G. The Data-Mining Industry Coming of Age. IEEE Intelligent System 1999.

Predictive Data Mining: A practical guide, Sholom M. Weiss, Nitin Indurkha, Morgan Kaufmann, 1997. <http://www.amazon.com/exec/obidos/ASIN/1558604030/>

REDES de Produtos (Mineração de Dados Econômicos), 2000, Disponível em: <<http://jcspl.wordpress.com/2007/08/18/redes-de-produtos-mineracao-de-dados-economicos/>>

TÉCNICAS de Data Mining, 2005. Disponível em < <http://geocities.yahoo.com.br/dugimenes/tecnicas.htm>>.

The Basel II banking regulations, <http://www.bis.org/publ/bcbsca.htm>

Vannevar Bush, As We May Think,
<http://ccat.sas.upenn.edu/~jod/texts/vannevar.bush.html>

WEISS, S., INDURKHZA, N. Predictive Data Mining a practical guide. San Francisco: Morgan Kaufmann Publishers, 1998.

What is OLAP? From the OlapReport, <http://www.olapreport.com/fasmi.htm>