SUMÁRIO

|  |  |  |
| --- | --- | --- |
| **LISTA DE ABREVIATURA**......................................................................... | | ix |
| **LISTA DE FIGURAS**................................................................................... | | x |
| **LISTA DE TABELAS**.................................................................................. | | xi |
|  | |  |
| **RESUMO....................................................................................................** | | xii |
|  | |  |
| **ABSTRACT................................................................................................** | | xiii |
|  | |  |
| 1. | **INTRODUÇÃO** | 15 |
| 1.1. | OBJETIVOS | 16 |
| 1.2. | MOTIVAÇÃO | 18 |
| 1.3. | JUSTIFICATIVA | 19 |
| 1.4 | ESTRUTURAÇÃO DA TESE |  |
|  |  |  |
| 2. | **FUNDAMENTAÇÃO TEÓRICA** | 21 |
| 2.1. | BUSINESS INTELLIGENCE | 21 |
| 2.1.1. | Arquitetura de Business Intelligence | 21 |
| 2.2. | DATA WAREHOUSE | 23 |
| 2.2.1. | Arquitetura de Data Warehouse | 25 |
| 2.2.2. | Data Warehouse Empresarial | 26 |
| 2.2.3. | Data Mart | 26 |
| 2.2.4. | Virtual Data Warehouse | 27 |
| 2.3. | PROCESSAMENTO ANÁLITICO ON-LINE | 27 |
| 2.3.1. | Técnicas de Análise de Dados Multidimensionais | 27 |
| 2.3.2. | Suporte Avançado de Banco de Dados | 28 |
| 2.3.3. | Interface fácil de utilizar para os usuários finais | 28 |
| 2.3.4. | Arquitetura Cliente/Servidor | 28 |
| 2.3.5. | Arquitetura OLAP | 28 |
| 2.3.5.1. | Processamento analítico on-line relaciona (ROLAP) | 30 |
| 2.3.5.2. | Processamento analítico on-line multidimensional (MOLAP) | 30 |
| 2.4. | MODELAGEM MULTIDIMENSIONAL | 33 |
| 2.4.1. | Esquema estrela | 33 |
| 2.4.1.1. | Fatos | 33 |
| 2.4.1.2. | Dimensões | 34 |
| 2.4.1.3. | Medidas | 35 |
| 2.4.1.4. | Atributos | 35 |
| 2.4.1.5. | Hierarquias de Atributos | 36 |
| 2.4.2. | Esquema floco de neve | 39 |
| 2.5. | CONCLUSÕES | 42 |
|  |  |  |
| 3. | **O Processo de KDD** | 43 |
| 3.1. | Introdução | 43 |
| 3.2. | Caracterização do processo de KDD | 45 |
| 3.3. | Pré-processamento | 45 |
| 3.3.1. | Seleção de dados | 47 |
| 3.3.1.1. | Redução de dados horizontal | 47 |
| 3.3.1.2. | Segmentação do Banco de Dados | 47 |
| 3.3.1.3. | Eliminação direta de casos | 48 |
| 3.3.1.4. | Amostragem aleatória | 48 |
| 3.3.1.5. | Agregação de informações | 49 |
| 3.3.1.6. | Redução de dados vertical | 49 |
| 3.3.1.7. | Eliminação direta de atributos | 50 |
| 3.3.1.8. | Redução de valores | 50 |
| 3.3.1.8.1. | Redução de valores nominais | 51 |
| 3.3.1.8.2. | Redução de valores contínuos ou (Discretos) | 51 |
| 3.3.2. | Limpeza dos dados | 53 |
| 3.3.2.1. | Limpeza de informações ausentes | 53 |
| 3.3.2.2. | Limpeza de inconsistências | 54 |
| 3.3.2.3. | Limpeza de valores não pertencentes ao domínio | 55 |
| 3.3.3. | Transformação dos dados | 55 |
| 3.3.3.1. | Codificação: Numérica – Categórica | 55 |
| 3.3.3.2. | Codificação: Categórica – Numérica | 56 |
| 3.3.4. | Enriquecimento dos dados | 57 |
| 3.3.5. | Partição dos dados | 58 |
| 3.3.6. | Integração dos dados | 59 |
| 3.4. | Mineração de Dados | 59 |
| 3.4.1. | Definição | 59 |
| 3.4.2. | Tarefas de Mineração de Dados | 61 |
| 3.4.2.1. | Descoberta de Associações | 61 |
| 3.4.2.2. | Classificação | 62 |
| 3.4.2.3. | Agrupamento (Clustering) | 64 |
| 3.4.3. | Métodos de Mineração de Dados | 68 |
| 3.4.3.1. | Métodos baseados em Redes Neurais | 69 |
| 3.4.3.2. | Métodos Estatísticos | 70 |
| 3.4.3.3. | Método Específico – Apriori | 72 |
| 3.4.3.4. | Métodos baseados em Indução de árvores de Decisão | 72 |
| 3.4.3.5. | Métodos baseados em Lógica Nebulosa | 73 |
| 3.4.3.6. | Métodos Hierárquicos | 73 |
| 3.4.3.7. | Métodos baseados em Densidade | 75 |
| 3.4.3.8. | Métodos baseados em Grade | 79 |
| 3.5. | Pós-processamento | 82 |
| 3.6. | Tecnologias de suporte a Mineração de Dados | 85 |
| 3.6.1. | Estatísticas | 85 |
| 3.6.2. | Aprendizado de Máquina | 86 |
| 3.7. | Aplicações de Mineração de Dados | 88 |
| 3.8. | Visualização de Dados | 89 |
| 3.8.1. | Técnicas de Visualização Orientadas a Pixels | 89 |
| 3.8.2. | Técnicas de Projeção Geométrica | 90 |
| 3.8.3. | Visualização Iconográfica | 93 |
| 3.8.3.1. | Chernoff Faces | 93 |
| 3.8.4. | Stick Figures | 94 |
| 3.8.5. | Técnicas de Visualização Hierárquica | 95 |
| 3.9 | Conclusões | 96 |
|  |  |  |
| 4. | **BIG DATA** | 97 |
| 4.1. | Introdução | 97 |
| 4.2. | O Uso do Big Data | 98 |
| 4.3. | Map-Reduce | 99 |
| 4.4. | Sistema Distribuído | 100 |
| 4.4.1. | Organização Física de Nós de Computação | 100 |
| 4.4.2. | Organização de Sistemas de Arquivos em Grande Escala | 101 |
| 4.5. | As Tarefas de Mapeamento | 101 |
| 4.5.1. | Agrupamento e Agregação | 102 |
| 4.6. | As Tarefas de Redução | 102 |
| 4.7. | Detalhes de Execução de Map-Reduce | 103 |
| 4.8. | Lidando com Falhas de Nós | 104 |
| 4.9. | Conclusões | 105 |
|  |  |  |
| 5. | **RACIOCINIO BASEADO EM CASOS** | 106 |
| 5.1. | Introdução | 106 |
| 5.2. | Gestão do Conhecimento | 106 |
| 5.2.1. | Definição de Gestão do Conhecimento | 106 |
| 5.2.2. | O que é Conhecimento | 107 |
| 5.2.3. | Diferença entre Conhecimento, Informação e Dado | 108 |
| 5.2.4. | Atividades da Gestão do Conhecimento | 109 |
| 5.2.5. | Uma Metodologia para Gestão do Conhecimento | 110 |
| 5.3. | Raciocínio Baseado em Casos | 111 |
| 5.3.1. | Definição | 112 |
| 5.3.2. | Representação de Casos | 112 |
| 5.3.3. | Indexação | 114 |
| 5.3.4. | Aquisição (Storage) | 116 |
| 5.3.4.1. | Modelos de Memória Dinâmica | 116 |
| 5.3.4.2. | Modelos de Categorias Exemplares | 116 |
| 5.3.5. | Recuperação | 117 |
| 5.3.5.1. | Algoritmo de Vizinhança | 117 |
| 5.3.5.2. | Algoritmo de Indução | 118 |
| 5.3.6. | Adaptação | 118 |
| 5.4. | Conclusões | 119 |
|  |  |  |
| 6. | **REDES NEURAIS** | 121 |
| 6.1. | Introdução | 121 |
| 6.2. | Definição | 121 |
| 6.3. | O Neurônio | 124 |
| 6.4. | Classificação e Propriedades | 126 |
| 6.4.1. | Aprendizado RNA | 127 |
| 6.4.2. | Tipos de Unidades | 128 |
| 6.4.3. | Tipos de Arquiteturas de Conexões de Redes | 130 |
| 6.5. | Tipos de Aplicações para Redes Neurais | 132 |
| 6.6. | Vantagens das Redes Neurais | 132 |
| 6.7. | Inconvenientes das Redes Neurais | 133 |
| 6.8 | Conclusões | 134 |
|  |  |  |
| 7. | **PROJETO** | 135 |
|  |  |  |
| 8. | **DATA MINING APLICADA AO ESTUDO DE CASO** | 136 |
|  |  |  |
| 9. | **CONCLUSÕES** | 137 |
|  |  |  |
| 10. | **REFERENCIAL BIBLIOGRÁFICO** | 138 |

**LISTA DE ABREVIIATURAS**

|  |  |
| --- | --- |
| KDD | KNOWLEDGE DISCOVERY IN DATABASES |
| BI | BUSINESS INTELLIGENCE |
| OLAP | ONLINE ANALYTICAL PROCESSING |
| RBC | RACICIONIO BASEADO EM CASOS |
| ETL | EXTRACTION, TRANSFORMATION AND LOAD |
| DW | DATA WAREHOUSE |
| SGBD | SISTEMA GERENCIADOR DE BANCO DE DADOS |
| ROLAP | RELACIONATIONAL OLAP |
| MOLAP | MULTIDIMETIONAL OLAP |
| ER | ENTIDADE RELACIONAMENTO |
| 3FN | 3ª FORMA NORMAL |
| SQL | STRUCTURED QUERY LANGUAGE |
| DM | DATA MINING |
| ATM | AUTOMATIC TELLER MACHINE |
| SOM | SELF-ORGANIZED MAP |
| DBASCAN | DENSITY-BASED SPATIAL CLUSTERING OF APPLICATION WITH NOISE |
| OPTICS | ORDERING POINTS TO IDENTIFY THECLUSTERING STRUCTURE |
| DENCLUE | DENSITY-BASED CLUSTERING |
| CLIQUE | CLUSTERING IN QUEST |
| STING | STATISTICAL INFORMATION GRID-BASED |
| IA | INTELIGÊNCIA ARTIFICIAL |
| RFID | RADIO-FREQUENCY IDENTIFICATION |
| DFS | DISTRIBUTED FILE SYSTEM |
| CBR | CASE-BASED REASONING |
| RNA | REDE NEURAL ARTIFICIAL |
| MLP | MULTI-LAYER PERCEPTRON |
| OCR | RECONHECIMENTO DE CARACTERES |

**LISTA DE FIGURAS**

|  |  |
| --- | --- |
| Figura 2.1 | 22 |
| Figura 2.2 | 24 |
| Figura 2.3 | 25 |
| Figura 2.4 | 29 |
| Figura 2.5 | 31 |
| Figura 2.6 | 32 |
| Figura 2.7 | 34 |
| Figura 2.8 | 35 |
| Figura 2.9 | 36 |
| Figura 2.10 | 37 |
| Figura 2.11 | 38 |
| Figura 2.12 | 39 |
| Figura 2.13 | 40 |
| Figura 2.14 | 42 |
| Figura 3.1 | 46 |
| Figura 3.2 | 77 |
| Figura 3.3 | 80 |
| Figura 3.4 | 81 |
| Figura 3.5 | 89 |
| Figura 3.6 | 90 |
| Figura 3.7 | 91 |
| Figura 3.8 | 91 |
| Figura 3.9 | 92 |
| Figura 3.10 | 93 |
| Figura 3.11 | 94 |
| Figura 3.12 | 95 |
| Figura 3.13 | 96 |
| Figura 4.1 | 99 |
| Figura 4.2 | 104 |
| Figura 5.1 | 107 |
| Figura 5.2 | 109 |
| Figura 5.3 | 110 |
| Figura 5.4 | 110 |
| Figura 5.5 | 113 |
| Figura 5.6 | 115 |
| Figura 6.1 | 122 |
| Figura 6.2 | 123 |
| Figura 6.3 | 124 |
| Figura 6.4 | 125 |
| Figura 6.5 | 129 |
| Figura 6.6 | 130 |
| Figura 6.7 | 131 |

**LISTA DE TABELAS**

|  |  |
| --- | --- |
| Tabela 2.1 | 32 |
| Tabela 3.1 | 56 |
| Tabela 3.2 | 56 |
| Tabela 3.3 | 57 |
| Tabela 3.4 | 57 |
| Tabela 3.5 | 84 |
| Tabela 6.1 | 123 |

**RESUMO**

As instituições de ensino, de um modo geral, têm muitos dados acumulados: rastreamento de nota, atendimentos: biblioteca, refeitório, programas de bolsas; resultados de teste, dados sócios econômicos, formulários de inscrições e assim por diante. Mas pouco tem sido feito com estes dados armazenados – seja devido a questão políticas ou por desconhecimento – para melhorar a aprendizagem dos estudantes. No máximo o que se faz é tirar alguns relatórios estatísticos desses dados.

No entanto, com a adoção cada vez maior das tecnologias nas escolas e um maior acesso aos dados, inclusive governamentais, há claramente uma série de oportunidades para uma melhor coleta de dados e análise desses dados para promover uma melhora na aprendizagem no ensino.

Neste contexto, este trabalho tem como objetivo, desenvolver uma ferramenta que auxilie, tanto alunos como gestores na melhoria da aprendizagem no ensino. Para tanto, essa ferramenta deve usar recursos avançados de Mineração de dados e Inteligência Artificial, para diagnosticar problemas relacionados a evasão e reprovação na escola, bem como diagnosticar os problemas relacionados ao mau desempenho dos alunos nas disciplinas. Essa ferramenta ao diagnosticar problemas de desempenho do aluno em determinada disciplina, irá sugerir, como desafios, algumas tarefas para o mesmo resolver, com o intuito de melhor o seu desempenho naquela disciplina.

**Palavras-chave**: Business Intellingence, Mineração de Dados, Descoberta de Conhecimento em Bases de Dados, Raciocínio Baseado em Casos, Redes Neurais, Big Data.

**ABSTRACT**

Educational institutions, in general, have many accumulated data: tracking note, calls: library, cafeteria, scholarship programs, test results, economic data members, registration forms, and so on. But little has been done with the data stored - whether due to political reasons or through ignorance - to improve student learning. At most what you do is take some statistical reports such data.

However, with the increased adoption of technology in schools and greater access to data, including government, there is clearly a lot of opportunities for better data collection and data analysis to promote improved learning in teaching.

In this context, this work aims to develop a tool to assist both students and managers on improving learning in teaching. To this end, the tool must use advanced features of Data Mining and Artificial Intelligence, to diagnose problems related to evasion and failure in school, as well as diagnose problems related to poor performance of students in the disciplines. This tool to diagnose performance problems in the student's particular discipline, will suggest, as challenges, some tasks to the same address, in order to better your performance in that discipline.

**Keywords**: Business Intellingence, Data Mining, Knowledge Discovery in Databases, Case-Based Racionary, Neural Network, Big Data.

Parte I

Capítulo

1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **INTRODUÇÃO**

As escola tem armazenado grande massa de dados, obtidas através do rastreamentos de notas, frequência, grade escolar, compras de livros texto, dados sócio econômicos dos pais, preocupação com a saúde (física e mental) dos alunos, a escolaridade dos pais, dados governamentais e similares. Mas pouco tem sido feito com essas informações - devido a questões de privacidade ou capacidade técnica - para melhorar a aprendizagem dos alunos.

No entanto, com a adoção de mais tecnologias nas escolas e com o acesso mais fácil aos dados governamentais, há claramente uma maior oportunidade para uma melhor coleta de dados e análises desses dados em prol da educação. Infelizmente, muitas escolas e universidades fazem muito pouco com essa riqueza de dados, uma ou outra que possivelmente produz um relatório anual de seus perfis. Mesmo uma simples análise dos dados institucionais poderia levantar perfis dos alunos para um potencial atendimento padronizado ou indicar um atendimento individual ao aluno.

Segundo Rezende (2003), os avanços em hardware e software tem permitido que os computadores tenham aplicações em áreas não convencionais. Como por exemplo, os Sistemas Inteligentes, utilizam a tecnologia da informação para manipular conhecimentos especializados com benefícios qualitativos e quantitativos. Permitindo desta forma que, um maior número de pessoas tenha acesso ao conhecimento a partir da aquisição, sistematização, representação e processamento desse conhecimento.

Os Sistemas Inteligentes podem manipular símbolos que representam entidades do mundo real, e dessa forma, são capazes de trabalhar eficazmente com conhecimento.

A Inteligência Artificial, tem em suas pesquisas, o objetivo de capacitar o computador a executar funções que os seres humanos desempenham usando conhecimento e raciocínio. Então, para que possamos aspirar à ação inteligente, é preciso analisar todos os aspectos relativos ao desenvolvimento e uso da inteligência (Rezende, 2003). Dentro deste contexto, torna-se evidente que a incorporação de conhecimento é um requisito fundamental para a construção de sistemas computacionais inteligentes

Neste trabalho pretende-se desenvolver uma solução tecnológica que permita ao professor fazer o acompanhamento e a avaliação do aluno a partir das interações destes com um ambiente de ensino. O foco principal do trabalho é usar o processo de descoberta de conhecimento em bases de dados, também conhecido *Knowledge Discovery in Databases* (**KDD**), afim de observar a viabilidade e aplicabilidade de um caso real de apoio a decisão. O estudo inclui também o uso de tecnologias de análise e recuperação de dados úteis ao processo decisório, conhecidas como OLAP e, da aplicação de técnicas e algoritmos de Data Mining para descoberta de novos conhecimentos e padrões nos dados. Este conhecimento adquirido poderá então ser usado na melhoria da educação.

1. **OBJETIVOS**
   1. **Objetivos Gerais**

Nesta pesquisa pretende-se desenvolver uma ferramenta para auxiliar o professor ou os gestores a diagnosticar problemas na aprendizagem de alunos, em relação ao ensino aprendizagem, tais como: alto índices de desistências e reprovações, bem como, identificar o mau desempenho do aluno nas disciplinas e sugerir soluções para os problemas encontrados. Para isto, será desenvolver uma solução tecnológica (*Dashboard*) que permita ao professor fazer o acompanhamento e a avaliação do aluno a partir dos dados coletados de diversas fontes, tais como sistema acadêmico e de formulários de pesquisas. O foco principal do trabalho é utilizar a mineração de dados, para descobrir padrões de comportamento dos alunos dentro do sistema, analisando como os alunos adquiriram conhecimento. Os padrões descobertos poderão então ser usado na melhoria do desempenho desses alunos. Portanto, o objetivo principal desse trabalho de investigação é responder as seguintes questões: **(1)** Quais são causas que implicam no alto índice de desistências e de reprovações dos alunos, **(2)** Porque existe hoje um certo desinteresse por parte dos alunos em alguns cursos ministrados pelo IFRN.

* 1. **Objetivos Específicos**
* Coletar os dados dos diversos sistema de informação (acadêmico, formulários de pesquisas do aluno, do professor e da instituição);
* Modelar uma estrutura multidimensional utilizando o modelo estrela ou floco de neve;
* Utilizar um aplicativo para extração, transformação e carga dos dados para o sistema multidimensional;
* Criar um Data Warehouse para organizar os dados para os algoritmos de Mineração de Dados;
* Definir os critérios e indicadores de avaliação do desempenho;
* Construir o processo de avaliação de desempenho da aprendizagem na educação (Data Mining);
* Interpretar os resultados obtidos;
* Desenvolver um Visualizador (protótipo) de dados que será utilizado pelo usuário final.
* Desenvolver a solução completa do sistema.

1. **MOTIVAÇÃO**

O Instituto Federal de Educação, Ciências e Tecnologia do Rio Grande do Norte ao longo de sua história, sempre foi considerado uma escola de muito prestigio nacional, por vários fatores, tais como excelente quadro de professores e servidores, ótimo desempenho de seus alunos no mercado de trabalho, ou ótimo desempenho dos alunos na academia, quando os mesmos procuram fazer ciência ou engenharia. Mas nos últimos tempos, tem-se observado um índice alto na evasão, reprovação e até mesmo um certo grau de desinteresse dos alunos pelos cursos principalmente o curso técnico.

Nos Institutos Federais têm-se muitos dados acumulados obtidos através do rastreamento de nota, atendimentos ao aluno (biblioteca, refeitório, assistência a bolsas de estudos), resultados de teste, dados sócios econômicos, condições dos pais e assim por diante. Mas pouco tem sido feito com estes dados armazenados – seja devido a questão políticas ou por desconhecimento – para melhorar a aprendizagem dos estudantes.

Com a adoção cada vez maior das tecnologias nas escolas e um maior acesso aos dados, inclusive governamentais, há claramente uma série de oportunidades para uma melhor coleta de dados e análise desses dados para promover uma melhora na aprendizagem.

No setor coorporativo, os sistemas inteligentes (*Business Intellingence* - **BI**) têm o papel "na tomada de decisão". Na educação, eu acredito que a análise da aprendizagem terá esse papel. Uma vez que entendendo melhor o processo de aprendizagem - as entradas, as saídas, os fatores que contribuem para o sucesso dos alunos - então podemos começar a tomar decisões informadas que são suportadas por evidência.

A motivação deste trabalho é desenvolver uma ferramenta que permita alunos, professores e administradores de uma instituição de ensino aprendizagem, possam ter acesso para avaliar o processo de ensino aprendizagem. Principalmente, para os gestores poderem analisar o por quer de tanta desistência e reprovação de alunos. Esta ferramenta terá também o papel de ao identificar que um aluno está com desempenho escolar insatisfatório, ele vai sugerir desafios para esse aluno, para que o mesmo possa melhorar seu desempenho escolar. Dessa forma, eu acredito que a análise pode ser uma ferramenta poderosa na motivação do aluno.

1. **METODOLOGIA**

Esta sessão tem como finalidade descrever o modo como será desenvolvida esta pesquisa, em relação aos objetivos definidos. Primeiramente será feita uma pesquisa do referencial bibliográfico, que dará um embasamento teórico para o desenvolvimento do projeto (estudo de caso). Fazendo vir à tona os conceitos mais relevantes sobre Banco de Dados, Modelagem Multidimensional, **OLAP**, *Data Warehouse* e Descoberta de Conhecimento em Bases de Dados, mas conhecido como *Knowledge Discovery Data* (**KDD**).

Após o levantamento bibliográfico, será feita a coleta de dados nas diversas fontes de dados do Instituto Federal, desde a base de dados acadêmica até os relatórios estatísticos disponíveis pelas coordenações pedagógicas e administrativas do Instituto Federal, após isso, será dado início ao processo de descoberta de conhecimento em bases de dados.

Feita a coleta dos dados, então será criado o modelo multidimensionais do sistema, com base nos objetivos a serem alcançados, para em seguida, utilizar ferramentas de extração, transformação e carga de dados, para preparar e carregar os dados no modelo multidimensional.

Com o modelo multidimensional pronto, será criado o *Data Warehouse*, para se ter uma base de dados otimizada para se aplicar o(s) algoritmo(s) de Mineração de Dados para descoberta de novos conhecimentos.

Com o *Data Warehouse* criado, deve-se definir que técnicas e algoritmos de Data Mining devem ser utilizados e em seguida definir também as tecnologias a serem utilizadas no processo de descoberta de conhecimentos.

Considerando a complexidade normalmente inerente a processo de descoberta de conhecimento em bases de dado, a metodologia proposta utiliza como base princípios de planejamento de atividades. Dessa forma, em função dos objetivos de cada aplicação de KDD, os passos do processo de descoberta de conhecimento deverão ser planejados ante do início de sua execução. A aplicação da metodologia será dividida em quatro momentos.

A primeira etapa envolve a definição sobre “o que fazer” diante da base de dados apresentada. Nesta etapa, devem ser executadas as tarefas de “Levantamento Inicial” e de “Definição de Objetivos”.

O levantamento inicial compreende um exame preliminar da base de dados, procurando obter informações sobre a natureza dos dados a serem analisados.

Na definição de objetivos, devem ser identificadas quais as tarefas de Mineração de dados são viáveis, para atender as expectativas e às necessidades do usuário do domínio da aplicação. Nesta fase, devem ser formulados alguns requisitos quanto ao modelo de conhecimento a ser produzido.

A partir da escolha de um objetivo, a abordagem é direcionada para a definição sobre “como fazer”, que corresponde a etapa de “Planejamento de Atividades”. Nesta etapa devem ser definidas as alternativas de plano de ação associados ao objetivo escolhido. Os planos de ação devem ser constituído a partir de cada método de mineração de dados aplicável à tarefa de KDD associado ao objetivo selecionado.

Finalmente, a abordagem proposta é concluída pela etapa de “Avaliação de Resultados”. Essa etapa corresponde à “analise do que foi feito”. Neste momento, as características do modelo de conhecimento gerado devem ser confrontados com as expectativas quanto ao modelo formulados na etapa “Definição de Objetivos”.

Este processo deve ser iterativo e interativo, de forma que, dependendo dos resultados obtidos, os analistas de KDD possam retornar a qualquer etapa realizada anteriormente em busca de melhores resultados. Para que isso seja possível, a metodologia requer uma documentação detalhada das ações realizadas e dos resultados produzidos.

Todo esse processo será englobado em uma ferramenta, onde o usuário possa manipular facilmente, para selecionar as informações utilizadas no processo e também visualizar os resultados obtidos.

1. **ESTRUTURAÇÃO DA TESE**

Esta pesquisa está estruturada em nove capítulos, divididos em duas partes. A parte I compreende a fundamentação teórica, a qual é composta por seis capítulos e a parte II que composta por quatro capítulos, que serão descriminados a seguir.

O primeiro capítulo trata de uma introdução, onde é dada uma visão ampla da pesquisa, os objetivos que se almeja alcançar, a motivação que deu incentivo ao desenvolvimento dessa pesquisa e a metodologia a ser aplicada no desenvolvimento da pesquisa.

No segundo capítulo é feito um estudo do estado da arte do *Business Intelligence* e modelagem multidimensional. Este capítulo é essencial, pois ele dará o embasamento teórico para construção dos modelos multidimensionais e consultas OLAP.

No terceiro capítulo é feito um estudo do estado da arte do Processo de Descoberta de Conhecimento em bases de dados. Capítulo chave, uma vez que mostra o conhecimento teórico envolvido em um processo de descoberta de conhecimento.

No capítulo quarto é dada uma visão geral sobre Big Data, que um termo muito utilizado hoje na mídia internacional, para conceituar conjuntos de dados muitos extremamente grandes, que por esse motivo precisam de ferramentas tecnologias especiais para se fazer analise dados sobre eles. A finalidade, desse capítulo, é dá um embasamento teórico sobre Big Data, uma vez que, será apresentado trabalhos futuros, para resolver determinadas aplicações da pesquisa, usando-se o Big Data.

No capítulo cinco e seis, apresentam o estado da arte sobre Raciocínio Baseados em Casos (**RBC**) e Redes Neurais. A pesquisa terá como uma das metas, a procura por comportamento de alunos que apresentem mau desempenho, e a partir do momento que a ferramenta identifique o problema, o sistema deverá apresentar uma solução para o mesmo. Então, diante deste cenário, pretende-se utilizar os **RBC**s para solucionar esta questão, ou seja, apresentar a solução para o problema encontrado, como base em soluções armazenadas em bases de conhecimento. As Redes Neurais, entram aqui para que se possa fazer uma analogia entre elas e os **RBC**s.

No capítulo 7 será definido o projeto, ou seja, o estudo de caso. Aqui será desenvolvida as etapas do desenvolvimento da ferramenta (protótipo) proposta na pesquisa, ou seja, nos objetivos gerais.

No capítulo 8, será feito todo planejamento do processo de descoberta de conhecimento em bases de dados. Levantamento inicial, definição de objetivos e análise dos resultados obtidos.

No capítulo 9 serão apresentadas as conclusões e analises dos resultados obtidos.

Finalmente, no capítulo 10, temos o referencial bibliográfico e bibliografia consultada.

Parte I

Capítulo

2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Fundamentação Teórica

2.1 **BUSINESS INTELLIGENCE**

Segundo Rob (2011), o termo *Business intelligence* (**BI**) é utilizado para descrever um conjunto amplo, coeso e integrado de ferramentas e processos utilizados para captar, coletar, integrar, armazenar e analisar dados para a geração e a apresentação de informações que deem suporte à tomada de decisões de negócio. Como o próprio nome diz, **BI** trata da criação de inteligência sobre o negócio. Portanto, o **BI**, é um modelo que permite à empresa transformar dado em informação, informação em conhecimento e conhecimento em sabedoria.

O **BI** não é, por si só, um produto, mas um modelo de conceitos, práticas, ferramentas e tecnologias (*data warehouse*, *data mart*, **OLAP** e/ou ferramentas de mineração de dados) que auxiliam uma empresa a compreender melhor seus recursos centrais e identificam oportunidades fundamentais para criar competitividade (Rob, 2011). Em geral, o **BI** envolve as seguintes etapas:

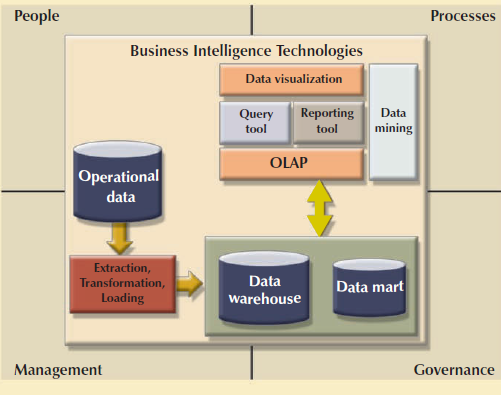
* Coleta e armazenamento de dados operacionais.
* Agregação de dados operacionais em dados de suporte a decisões.
* Análise de dados de suporte a decisões para gerar informações.
* Apresentação dessas informações ao usuário final para dar suporte a decisões de negócios.
* Tomada de decisões de negócio, o que, por sua vez, gera mais dados que são coletados, armazenados etc. (reiniciando o processo).
* Monitoramento para avaliar os resultados das decisões de negócio (Rob, 2011).

Para implementar todas essa etapas, o **BI** utiliza diversos componentes e tecnologias, o que será apresentado nas seções a seguir.

**2.1.1 Arquitetura de Business Intelligence**

Segundo Rob (2011), o **BI** utiliza-se de tecnologias e aplicações para o gerenciamento de todo o ciclo de vida dos dados, da aquisição ao armazenamento, transformação, integração, análise, monitoramento e apresentação. Não existe uma arquitetura única de **BI**, no entanto, há alguns tipos gerais de recursos, que são compartilhados por todas as implementações de **BI**.

Como poderia deixar de ser, uma boa infraestrutura de *TI*, sempre é composta de dados, pessoas, processos, tecnologias e gerenciamento desses componentes e, para uma arquitetura de **BI**, não pode ser diferente. A Figura 2.1 ilustra um modelo de BI.



**Figura 2.1**. Modelo de *business intelligence*. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

Para compreender a arquitetura de **BI**, será feita uma descrição dos componentes básicos que fazem parte de sua infraestrutura. Alguns desses componentes, possuem recursos adicionais. Porém, há quatro componentes básicos que todos os ambientes de **BI** devem fornecer, descritos as seguir (Rob, 2011):

* **Ferramentas de extração, transformação e carregamento (ETL) de dados**: esse componente é encarregado de coletar, filtrar, integrar e agregar dados operacionais a serem salvos em um armazém de dados otimizado para o suporte a decisões.
* **Armazenamento de dados**: o armazém de dados é otimizado para o suporte a decisões e costuma ser representado por um *data warehouse* ou data mart. Ele contém dados de negócios extraídos de bancos de dados operacionais e de fontes externas. Esses dados são armazenados em estruturas otimizadas, com foco na velocidade de análise e consulta.
* **Ferramentas de consulta e análise de dados**: esse componente executa as tarefas de recuperação, análise e mineração, utilizando os dados no armazém de dados e os modelos de análise de dados de negócio. Tal componente é utilizado pelo analista de dados para criar as consultas que acessam o banco de dados. Essa ferramenta orienta o usuário sobre quais dados selecionar e como construir um modelo de dados confiáveis. Tal componente costuma aparecer na forma de uma ferramenta **OLAP**.
* **Ferramentas de apresentação e visualização de dados**: esse componente é encarregado de apresentar os dados ao usuário final de várias formas. É utilizado pelo analista de dados para organizar e apesentar os dados. Essa ferramenta ajuda o usuário final a selecionar o formato de apresentação mais adequado, como relatório resumido, mapa ou gráfico.

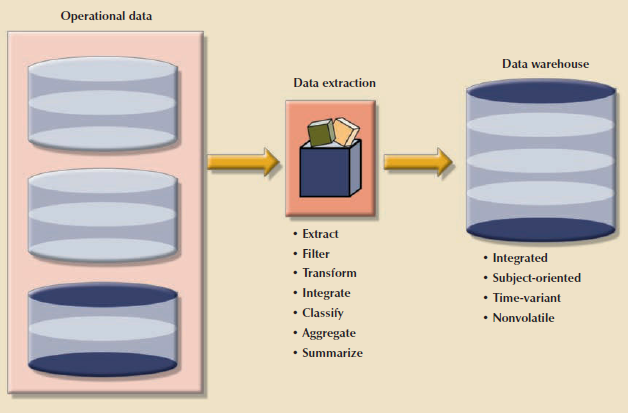
Deve-se ficar atento para o fato de que, os bancos de dados de suporte a decisões tendem a ser muito grandes. Muitos chegam à faixa dos gigabytes ou terabytes. Esses bancos de dados, demandam por análise sofisticada de dados e, incentivaram a criação de um novo tipo de armazém de dados. Esse armazém de dados, contém dados em formatos que facilitam sua extração, análise e a tomada de decisões. É conhecido como *data warehouse* e se tornou o fundamento de uma nova geração de sistemas de tomada de decisões (Rob, 2011).

**2.2 DATA WAREHOUSE**

Segundo Inmon (1994), o termo ***Data Warehouse (DW)*** é “um conjunto de dados integrado, orientado por assunto, variável no tempo e não volátil, que fornece suporte a tomada de decisões”. A seguir, será detalhado cada um desse componentes (Inmon, 1994):

* **Integrado**. O *data warehouse* é um banco de dados consolidado e centralizado, que integra dados proveniente de toda a organização e de várias fontes, com diversos formatos.
* **Orientado por assunto**. Os dados do *data warehouse* são dispostos e otimizados de modo a fornecerem respostas a perguntas provenientes de diversas áreas funcionais da empresa. São organizados e resumidos por temas, contendo assuntos de interesse especifico – produtos, clientes, departamentos, regiões, promoções, e assim por diante.
* **Variável no tempo**. Os sistemas transacionais focam nas transações correntes, enquanto os sistemas de *data warehouse*, representam o fluxo de dados através do tempo. Ou seja, os dados são carregados periodicamente no *data warehouse*, e quando isso acontece, todas as agregações dependentes do tempo (ouse dependentes dessa carga de dados), são recalculadas. Por exemplo, se os dados de vendas da semana, são carregados no *data warehouse*, serão atualizadas todas as agregações dependentes dessa carga, ou seja, os agregados semanais, mensais, anuais e de qualquer outras periodicidade que seja dependente dessa carga. Cada conjunto de dados, ao ser carregado em um *data warehouse*, fica vinculado a um rótulo temporal que o identifica dentre os demais. Cada rótulo temporal, fica portanto, associado a uma visão instantânea e sumarizada dos dados operacionais que corresponde ao momento de carga do *data warehouse*. Dessa forma, na medida que o data warehouse vai sendo carregado com tais visões, pode-se realizar análise de tendências a partir dos dados.
* **Não volátil**. Uma vez inserido um dado no *data warehouse*, ele nunca será removido. Uma vez que ele representa o histórico da empresa. Por este fato, o *data warehouse* está sempre crescendo. Portanto, o Sistema Gerenciador de Bancos de Dados (**SGBD**), que dá suporte a ele, deve ser capaz de suportar vários gigabytes de dados, ou até mesmo tera-bytes, operando com hardware com diversos processadores.

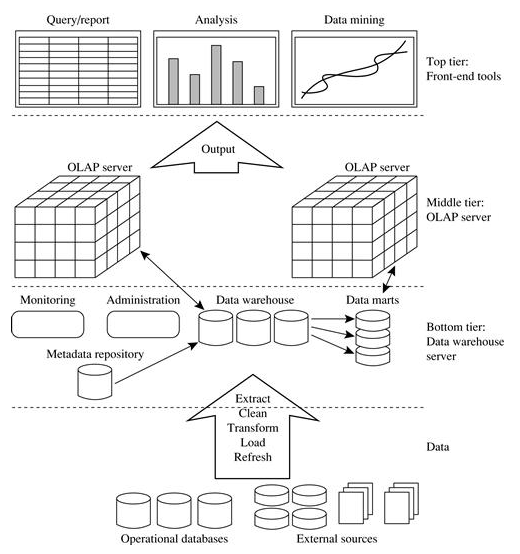
Resumindo, o *data warehouse* é um repositório de dados semanticamente consistente, que serve como uma implementação física de um modelo de dados de apoio a decisões. Ele armazena as informações que uma empresa necessita para tomar decisões (Han & Kamber, 2011). Normalmente é um banco de dados apenas de leitura, otimizado para processamento de análises e consultas. Em geral, os dados são extraídos de diversas fontes e, em seguida, transformados e integrados, antes de serem carregados no *data warehouse* (Inmon, 1994). A Figura 2.2 ilustra como o data warehouse é criado a partir dos dados contidos em um banco operacional.



**Figura 2.2**. Criação de um **data warehouse**. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

**2.2.1 Arquitetura de Data Warehouse**

Segundo Han & Kamber (2011), um **data warehouse** adota uma arquitetura em três camadas, como ilustra a Figura 2.3.



**Figura 2.3**. Uma arquitetura de data warehouse em três camadas. Fonte: Han & Kamber. Data Mining: Concepts and Techniques. 3nd, 2011.

1. A camada inferior (*bottom*) é um servidor de *data warehouse*, que quase sempre, é um sistema de banco de dados relacional. Segundo Han & Kamber (2011), são usadas ferramentas de back-end e utilitários para extrair dados dessa camada e alimentar a camada superior. Ainda segundo Han & Kamber (2011), os dados são extraídos usando interface de programação de aplicativo, conhecidos como ***gateways***. Um *gateway* permite que clientes gerem código SQL, para ser executado no servidor. Pode-se citar como exemplos de *gateways* ODBC (*Open Database Connection*) e OLEDB (*Object Linking and Embedding Database*) da Microsoft e JDBC (*Java Database Connection*). Essa camada também contém um repositório metadata, o qual armazena informações sobre o *data warehouse* e seus conteúdos.
2. A camada intermediária (*middle tier*), segundo Han & Kamber (2011), é um servidor OLAP que geralmente é implementado usando (1) um modelo relacional **OLAP** (**ROLAP**) (fornece recursos de OLAP utilizando bancos de dados relacionais e ferramentas familiares de consulta relacional para armazenar dados multidimensionais; ou (2) um modelo multidimensional **OLAP** (**MOLAP**) (amplia os recursos de OLAP para sistemas de gerenciamento de banco de dados multidimensionais (**SGBDM**s). O SGBDM utiliza técnicas especiais para armazenar dados em matrizes de n dimensões. O pressuposto do MOLAP é que os bancos de dados multidimensionais são os mais adequados para gerenciar, armazenar e analisar dados multidimensionais (Rob, 2011).
3. A terceira camada (top), segundo Han & Kamber (2011), é o fronte-end do cliente, a qual contém as ferramentas de consulta, de relatório, de análise e mineração de dados (por exemplo, análise de tendência, previsão, e assim por diante.).

Segundo Han & Kamber, do ponto de vista da arquitetura, há três modelos de data warehouse: o warehouse empresarial, o data mart e warehouse virtual.

**2.2.2 Data Warehouse Empresarial**

Um data warehouse empresarial, segundo Han & Kamber (2011), coleta todas as informações sobre todos os assuntos da organização. Ele fornece informações de dados de toda empresa, geralmente, oriundas de um ou mais sistemas operacionais ou informações obtidas externamente. Ele contém dados detalhados e/ou sumarizados, e pode variar em tamanho de poucos gigabytes para centenas de gigabytes, terabytes ou superiores. A sua implementação pode ser em um mainframe, supercomputador, em uma plataforma de arquitetura paralela. Requer uma modelagem comercial extensiva e pode levar muitos anos para ser projetado e construído (Han & Kamber, 2011).

**2.2.3 Data Mart**

De acordo com Inmon (1994), embora o *data warehouse*, seja uma proposta muito atraente, que traga muitos benefícios, os gerentes podem relutar em adotar essa estratégia, pelo fato de que, a criação de um *data warehouse* exige tempo, dinheiro e considerável esforço gerencial. Estes fatos, fazem com que muitas empresas iniciem na criação de *data warehouse*, focando em conjuntos de dados gerenciais, orientados a atender pequenas áreas de negócio, dentro da empresa. Esses armazenamentos menores são chamados de ***data marts***. Um ***data mart*** é portanto, segundo Inmon (1994), um pequeno subconjunto de um *data warehouse*, sobre um único assunto, que fornece suporte às decisões de um pequeno grupo de pessoas. No entanto, pode-se criar um *data mart* a partir de dados extraídos de um *data warehouse*, com a finalidade especifica de dar suporte a um acesso mais rápido a determinado grupo ou função. Dessa forma, os *data marts* e o *data warehouse* podem coexistir em um ambiente de business *intelligence* (Inmon, 1994).

**2.2.4 Virtual Data Warehouse**

De acordo com Han & Kamber (2011), um warehouse virtual é um conjunto de visões sobre bases de dados operacionais. Você pode materializar algumas visões operacionais, para obter um processamento de consultas eficientes. O warehouse virtual é o estado de visibilidade global de recursos, com base na aquisição e processamento de dados operacionais em tempo real. Informações disponíveis no armazém virtual tem o potencial de reduzir custos e melhorar o serviço ao cliente. A infraestrutura já está disponível para captura de dados em tempo real, e o custo de aquisição de dados continuará a reduzir.

**2.3 PROCESSAMENTO ANÁLITICO ON-LINE**

De acordo com Rob (2011), a necessidade de suporte a decisões mais intensivo, levou à introdução de uma nova geração de ferramentas. Tais ferramentas, foram denominadas de **processamento analítico on-line** (**OLAP** – *Online Analytical Processing*). Essa nova ferramenta cria um ambiente avançado de análise de dados que dá suporte à tomada de decisões, modelagem comercial e pesquisa operacional. Ainda segundo Rob (2011), esses sistemas comportam quatro características principais:

* Utilizam técnicas de análise de dados multidimensionais.
* Proporcionam suporte avançado a bancos de dados.
* Fornecem interface fácil de utilizar para o usuário final.
* Dão suporte a arquitetura cliente/servidor.

**2.3.1 Técnicas de análise de dados multidimensionais**

De acordo com Rob (2011), a característica mais evidente das modernas ferramentas **OLAP**, é a capacidade de análise multidimensional, onde, os dados são processados e visualizados como parte de uma estrutura multidimensional. Essas técnicas de análise de dados multidimensionais, utilizam as seguintes funções:

* Funções avançadas de apresentação de dados. Gráficos 3D, pivô, tabulações cruzadas, rotação de dados e cubos tridimensionais.
* Funções avançadas de agregação, consolidação e classificação de dados. Permitem que o analista de dados crie vários níveis de agregação, detalhamento de dados, *drill down* e *roll up* de dados em diferentes dimensões e níveis.
* Funções computacionais avançadas. Incluem variáveis orientadas para negócio (tais como participação de mercado, margem de vendas, etc.), relações financeiras e contábeis (tais como lucratividade, despesas gerais, alocação de custos e retorno), funções estatísticas e de previsão.
* Funções avançadas de modelagem de dados. Dão suporte para cenários de simulação, avaliação de variáveis, contribuições de variáveis para o resultado, programação linear, dentre outras, ferramentas de modelagem.

**2.3.2 Suporte Avançado de Banco de Dados**

As ferramentas OLAP, para apresentar suporte eficiente a decisões, deve ter recursos avançados de acesso a dados. Tais recursos incluem (Rob, 2011):

* Acesso a vários tipos de SGBDs, arquivos fora do banco de dados (flat-file) e fontes de dados internas e externas.
* Acesso a dados agregados de *data warehouse*.
* Recursos avançados de navegação de dados, como *drill down* e *roll up*.
* Tempo rápido e consistente de resposta a consultas.
* Capacidade de mapear solicitações de usuários finais, para a fonte adequada de dados e, em seguida, para a linguagem adequada de acesso aos dados (normalmente SQL). Segundo Rob (2011), as ferramentas OLAP mapeiam os elementos de dados do *data warehouse* e do banco de dados operacional em seus próprios dicionários. Esses metadados são utilizados para traduzir as solicitações de análise de dados do usuário final em código de consulta (otimizados) adequados que são, em seguida, direcionados para a(s) fonte(s) de dados correta(s).
* Suporte a bancos de dados muito grandes. Pois como já foi falado, o *data warehouse* pode crescer rapidamente e atingir facilmente gigabytes ou até mesmo terabytes.

**2.3.3 Interface fácil de utilizar para os usuários finais**

De acordo com Rob (2011), os recursos avançados OLAP são mais úteis quando o acesso a eles são simples. Então, os fornecedores conhecedores dessas necessidades, supriram suas ferramentas OLAP, com interfaces gráficas fáceis de utilizar. Muitos desses recursos foram herdados de ferramentas de análise de dados anteriores, já familiares aos usuários finais.

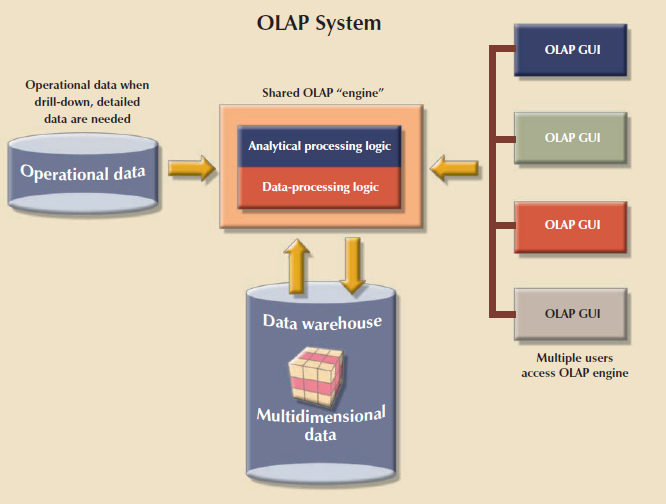
**2.3.4 Arquitetura Cliente/Servidor**

Um ambiente cliente/servidor possibilita que um sistema OLAP seja dividido em vários componentes que definem sua arquitetura. Esses componentes podem, então, ser colocados no mesmo computador ou distribuídos entre diversas máquinas. Assim, segundo Rob (2011), o OLAP é projetado para atender a exigências de facilidades de utilização, ao mesmo tempo em que mantém a flexibilidade do sistema.

**2.3.5 Arquitetura OLAP**

Segundo Rob (2011), os sistemas OLAP são projetados para utilizar tanto dados operacionais como de **data warehouse**. Para isso, existem várias arquiteturas de instalação de sistema OLAP, baseados nas regras de negócio de cada empresa. Por exemplo, existe situação onde o sistema OLAP é instalado em um único computador, o problema nessa arquitetura, é que, o computador deve ser potente para armazenar o sistema OLAP e executar localmente todos os processamentos de dados. Uma outra arquitetura é, aquela em que o GUI do OLAP executa em estações de trabalho clientes, enquanto o mecanismo OLAP, ou servidor, composto da lógica de processamento analítico e de processamento de dados, é executado em computador compartilhado. Nesse caso, o servidor será um *front end* para todos os dados de suporte a decisões do *data warehouse*.

O fato é, segundo Rob (2011), que um sistema OLAP pode acessar ambos tipos de armazenamento de dados (operacional ou *data warehouse*) ou apenas um, dependendo da implementação que se deseje configurar. Em todo caso, a análise multidimensional de dados exige algum tipo de representação de dados multidimensionais, o que normalmente é fornecido pelo mecanismo OLAP. A Figura 2.4 ilustra uma arquitetura OLAP com organização de armazenagem de dados multidimensionais.



**Figura 2.4** Servidor OLAP com organização de armazenagem de dados multidimensionais. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

De acordo com Rob (2011), há diversas formas de gerenciar e armazenar os dados em um sistema OLAP, como já foi visto anteriormente, o OLAP relacional (**ROLAP**) e o OLAP multidimensional (**MOLAP**).

**2.3.5.1** **O processamento analítico on-line relacional** (**ROLAP**, sigla em inglês para *Relational Online Analytical Processing*).

Segundo Rob (2011), essa abordagem se estrutura a partir de tecnologias relacionais existentes e representa uma extensão natural para todas as empresas que já utilizem sistemas de gerenciamento de banco de dados relacionais. O ROLAP utiliza uma técnica especial de projeto que permite à tecnologia SGBDR dar suporte a representações de dados multidimensionais, conhecida como “Esquema estrela”. O esquema estrela é essencial para o sistema ROLAP, uma vez que, a tecnologia relacional utiliza a normalização como metodologia de projetos de banco de dados e, isso é visto como um obstáculo em sistemas OLAP. Como se sabe, a normalização divide as entidades de negócio em partes menores para produzir tabelas normalizadas. O motivo de se usar tabelas normalizadas é reduzir as redundâncias, eliminando as anomalias. No entanto, para sistemas de suporte a decisões, é mais fácil compreender os dados, quando os mesmos, são vistos em relação a outros dados.

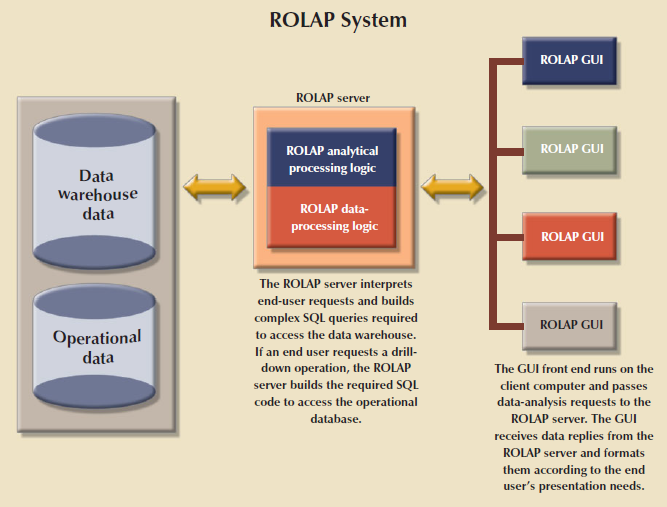
Um outro fator dos sistemas relacionais é que a SQL não é adequada para executar análise avançada de dados. Segundo Rob (2011), a maioria das solicitações de dados de suporte a decisões, exige o uso de consultas de SQL *multiple pass* (capaz de fazer várias passagens de processamento) ou vários comandos de SQL integrada. Portanto, segundo Rob (2011), para resolver este problema, as tecnologias ROLAP estende a SQL de modo que ela possa diferenciar entre exigências de acesso a dados de data warehouse (baseado em esquema estrela) e dados operacionais (tabelas normalizadas). Desse modo, o sistema ROLAP é capaz de gerar o código SQL necessário para acessar dados de esquema estrela.

As ferramentas de ROLAP são produtos Cliente/Servidor em que a interface do usuário final, o processamento analítico e o processamento de dados ocorrem em computadores diferentes. A Figura 2.5 ilustra a interação dos componentes de ROLAP cliente/servidor.

2.3.5.2 **O processamento analítico on-line multidimensional** (**MOLAP**, sigla em inglês para Multidimensional *Online Analytical Processing*).

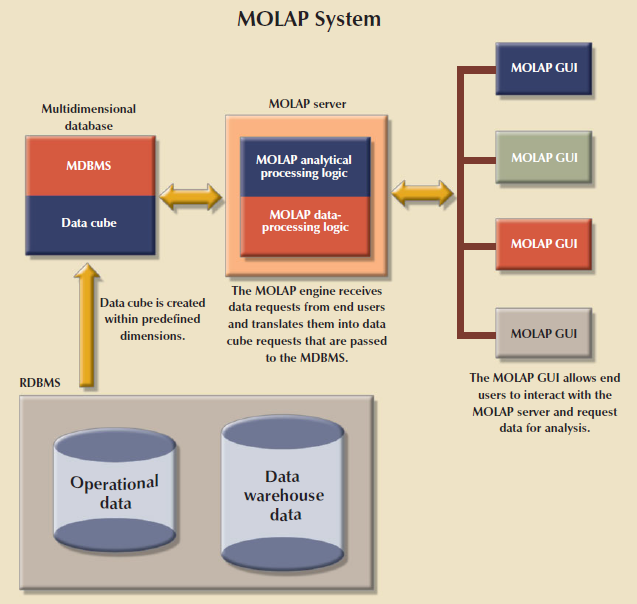
De acordo com Rob (2011), o MOLAP amplia os recursos de OLAP para sistemas de gerenciamento de banco de dados multidimensionais (**SGBDMs**). O pressuposto do MOLAP é que os bancos de dados multidimensionais são os mais adequados para gerenciar, armazenar e analisar dados multidimensionais.

Conceitualmente, os usuários finais de SGBDM visualizam os dados armazenados como um **cubo de dados**. Em um cubo de dados, a localização de cada valor de dado, é uma função dos eixos x, y e z em um espaço tridimensional. Os eixos x, y e z representam as dimensões do valor do dados. Os cubos podem crescer até um número de n dimensões, tornando-se, assim, *hipercubos*. De acordo com Rob (2011), uma característica importante dos cubos é que são estáticos, ou seja, não estão sujeitos a alterações e devem ser criados antes de sua utilização. Um outro fato é, que eles não podem ser criados por consultas ad hoc. Em vez disso, a consulta é feita em cubos pré-criados com eixos definidos. Por exemplo, um cubo de vendas terá as dimensões de produto, localização e tempo, permitindo a consulta apenas dessas dimensões. A Figura 2.6 ilustra uma arquitetura cliente/servidor de MOLAP.



**Figura 2.5** Arquitetura ROLAP de cliente/servidor. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

**Figura 2.6** Arquitetura MOLAP de cliente/servidor. **Fonte** Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.



**Tabela 2.1** Resume algumas das principais diferenças entre OLAP e MOLAP.

|  |  |  |
| --- | --- | --- |
| CARACTERISTICA | ROLAP | MOLAP |
| Esquema | Utiliza esquema estrela  É possível acrescentar novas dimensões dinamicamente. | Utiliza cubos de dados  Dimensões adicionais exigem a recriação do cubo de dados. |
| Tamanho do banco de dados | Médio a grande | Pequeno a médio |
| Arquitetura | Cliente/Servidor com base em padrões aberto | Cliente/Servidor  Proprietário |
| Acesso | Suporte a solicitações ad hoc  Dimensões ilimitadas | Limitado a dimensões predefinidas |
| Recursos | Altos | Muito altos |
| Flexibilidade | Alta | Baixa |
| Escalabilidade | Alta | Baixa |
| Velocidade | Boa com pequenos Data Sets;  Razoável para data sets médios ou grandes | Mais rápido com data sets pequenos ou médios; |

**2.4 MODELAGEM MULTIDIMENSIONAL**

A modelagem multidimensional é uma forma de Modelagem de Dados voltada para concepção e visualização de conjunto de medidas que descrevem aspectos comuns de um determinado assunto. É utilizada especialmente para sumarizar e reestruturar dados, apresentando-os em visões que suportem a análise dos dados envolvidos (Passos & Goldschimdt, 2005).

De acordo com Han & Kamber (2011), o *data warehouse* e as ferramentas **OLAP** são baseadas em um **modelo de dados multidimensional**. Nesse modelos, os dados são visto na forma de um cubo de dados. Um modelo multidimensional possui três componentes básicos: Fatos (***facts tables***), Dimensões (***dimensions***) e Medidas (***measures***). E existem diversas formas de modelagem física de um data warehouse, incluindo esquema estrela (***star schema***), esquema floco de neves (***snowflake***) e constelação de fatos (***fact constellation***). A seguir serão discutidas cada uma desses conceitos.

**2.4.1 Esquema Estrela**

O esquema estrela, segundo Rob (2011), é uma técnica de modelagem de dados multidimensionais de suporte a decisões em um banco de dados relacional. Ainda segundo, Rob (2011), o esquema estrela foi desenvolvido, pois as técnicas de modelagem relacional, entidade relacionamento (**ER**) e normalização existentes não produziam uma estrutura que atendesse às necessidades de análise avançada de dados.

O modelo estrela é fácil de implementar e, ao mesmo tempo em que preserva as estruturas relacionais, em que o banco operacional criado. O esquema estrela básico possui quatro componentes: fatos, dimensões, atributos e hierarquias de atributos.

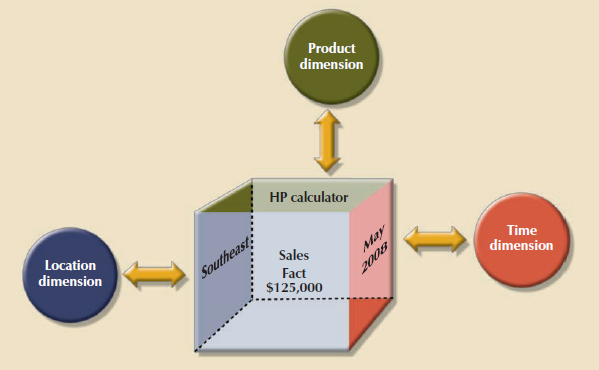
**2.4.1.1 Fatos**

Um **fato** é uma coleção de itens de dados, composta de dados de medidas e de contexto. Representa um item, ou uma transação ou um evento associado ao tema da modelagem. São medidas numéricas (valores) que representam um aspecto ou atividade específica dos negócios. Os fatos normalmente utilizados em análise de dados comerciais são unidades, custos, preços e receitas. Os fatos são armazenados em tabelas de fatos que constituem o centro do esquema estrela. A **tabela de fatos** (***fact table***) contém fatos vinculados por meio de suas dimensões (Kimball, 2002, Passos & Goldschimdt, 2005).

Segundo Rob (2011), os fatos também podem ser computados ou derivados no momento da execução. Esses as vezes são chamados de métricas para diferenciá-los dos fatos armazenados.

**2.4.1.2 Dimensões**

Uma dimensão é um tipo de informação que participa da definição de um fato. As dimensões determinam o contexto do assunto. As **dimensões** são características de qualificação que fornecem perspectivas adicionais a um determinado fato. Os dados de suporte a decisões são quase sempre vistos relacionados a outros dados, por isso, as dimensões são interessantes. Por exemplo, pode-se, em um sistema de suporte a decisões, querer comprar as vendas de certos produtos, entre regiões e entre períodos. Nesse exemplo, teríamos as vendas, as dimensões produto, localização e tempo. Segundo Rob (2011), as dimensões ampliam a visão dos fatos. Essa dimensões são armazenadas em **tabelas de dimensões**. A Figura 2.7 ilustra um esquema estrela para vendas com as dimensões de produto, localização e tempo.



**Figura 2.7** **Esquema estrela**. **Fonte:** Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

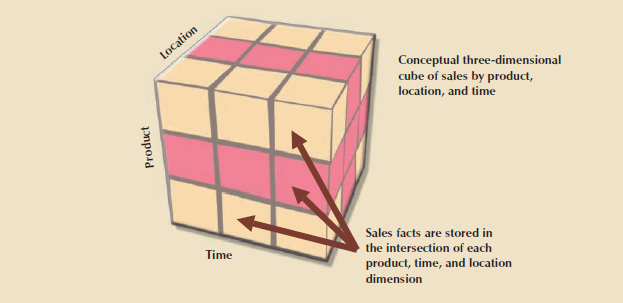
**2.4.1.3 Medidas**

Uma medida é um atributo ou variável numérica que representa um fato. Exemplos: valor da ação, número de evasões escolares, quantidade de produtos vendidos, valor total de venda, etc.

**2.4.1.4 Atributos**

De acordo com Kimball (2002), cada tabela de dimensão contém atributos. Os atributos costumam ser utilizados para buscar, filtrar e classificar fatos. As dimensões fornecem características descritivas sobre os fatos por meio de seus atributos.

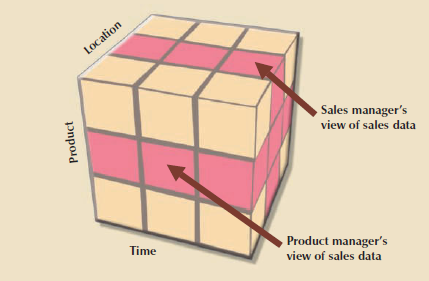
Segundo Rob (2011), conceitualmente, o modelo de dados multidimensional do exemplo de vendas, é melhor representado por um cubo tridimensional. Isso, não significa que haja um limite para o número de dimensões, que podem ser associadas a uma tabela de fatos. Não há limite matemático para o número de dimensões utilizadas. Usar um modelo tridimensional, torna mais fácil a visualização do problema. A Figura 2.8 ilustra uma visão das vendas dimensionadas por produto, localização e tempo, na terminologia de análise de dados multidimensional.



**Figura 2.8 Visão tridimensional de vendas**. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

Observe que cada valor de venda armazenado no cubo da Figura 2.8 está associado às dimensões de localização, produto e tempo. Lembre-se que isso é apenas uma representação conceitual de dados multidimensional e não mostra como os dados estão fisicamente armazenados no *data warehouse*. Os mecanismos **ROLAP** e **MOLAP** têm suas próprias tecnologias para simular esse cubo multidimensional (Rob, 2011).

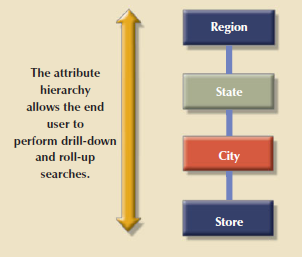
Um dos principais recursos da análise multidimensional, segundo Rob (2011), é a sua capacidade de focar em “fatias” específicas dos cubo. Em termos, multidimensionais, essa capacidade do cubo em executar uma análise mais detalhada é conhecida com detalhamento. A Figura 2.9 ilustra esse conceito. Observando-se a Figura 2.9, ver-se que cada corte através do cubo produz uma fatia. A interseção de fatias produz pequenos cubos de dados.



**Figura 2.9** **Visão de detalhamento de vendas**. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

**2.4.1.5 Hierarquias de Atributos**

De acordo com Kimball (2002), os atributos no interior de dimensões podem ser ordenados em hierarquias bem definidas. A hierarquia de atributos fornecem uma organização vertical utilizada para duas finalidades principais: agregação e análise de dados por ***drill down*** e ***roll up***. A Figura 2.10 ilustra como os atributos da dimensão de localização podem ser organizados em uma hierarquia por região, estado, cidade e loja.

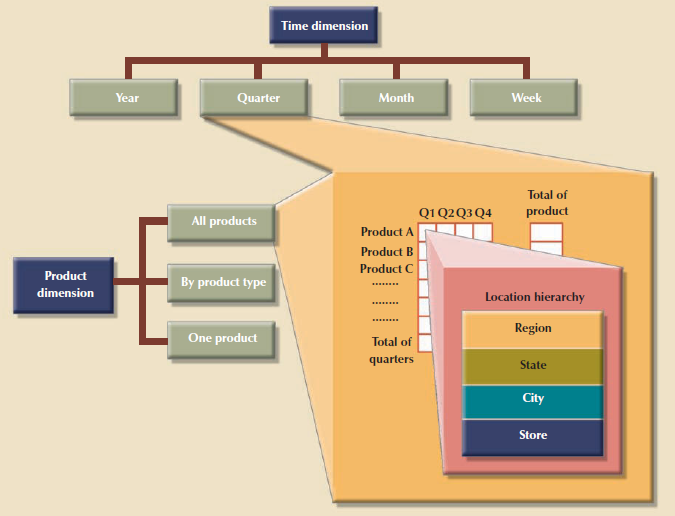


**Figura 2.10** Hierarquia de atributo de localização. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

De acordo com Rob (2011), a hierarquia de atributos fornece a possibilidade de executar buscas de *drill down* e *roll up* no *data warehouse*. A Figura 2.11 ilustra um cenário em que o analista de dados estuda fatos de vendas utilizando as dimensões de produto, tempo e localização. Nesse exemplo, a dimensão de produto é estabelecida como “Todos os produtos”, o que significa que o analista de dados verá todos os produtos no eixo y. a dimensão de tempo (eixo x) é estabelecida com “Trimestre”, indicando que os dados estão agregados por trimestre (por exemplo, vendas totais dos produtos A, B e C em T1, T2, T3 e T4). Finalmente, a dimensão de localização é inicialmente estabelecida para “Região”, garantindo, assim, que cada célula contenha as vendas regionais totais de determinado produto em determinado trimestre.

A tabela de fatos é relacionada com cada tabela de dimensões em um relacionamento de “muitos para um” (M:1). Dito de outra forma, várias linhas de fatos se relacionam a cada linha de dimensão. Deve-se ser lembrado que, os fatos e as dimensões são normalmente representadas por tabelas físicas no banco de dados do *data warehouse*. A Figura 2.12 ilustra os relacionamentos entre as tabelas de fatos de vendas e as tabelas das dimensões de produto, localização e tempo. Além dos mais, foi adicionada uma dimensão de cliente ao modelo, para mostrar como o esquema estrela pode ser expandido facilmente.

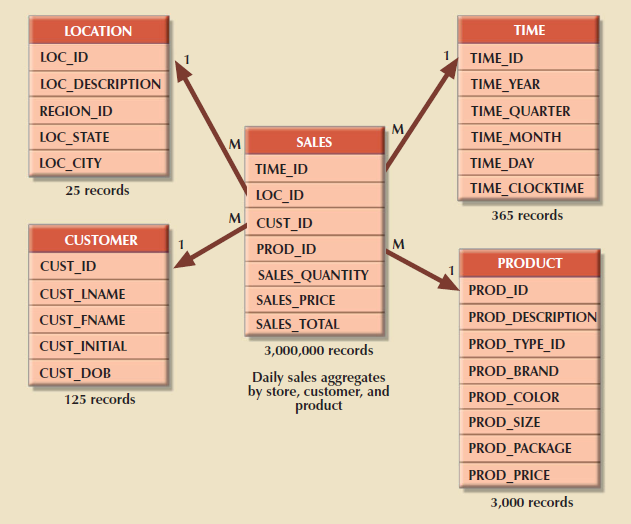
**Figura 2.11** Hierarquias de atributos em análise multidimensional. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.



De acordo com Kimball (2002), a criação de um banco de dados que forneça respostas rápidas e precisas a consultas de análise de dados, é o principal objetivo do *data warehouse*. Dessa forma, existem medidas de aprimoramento de desempenho que objetivam a velocidade das consultas, facilitando o código SQL e melhorando a representação semântica das dimensões de negócio. Segundo Kimball (2002), costuma-se utilizar quatro técnicas para otimizar o projeto de *data warehouse*:

* Normalização de tabelas dimensionais.
* Manutenção de várias tabelas de fatos para representar diferentes níveis de agregação.
* Desnormalização de tabelas de fatos.
* Particionamento e replicação de tabelas.

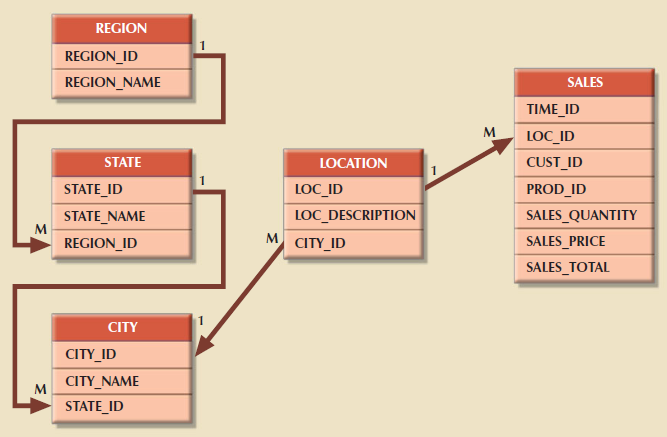
**Figura 2.**12 Esquema estrela para Vendas. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.



**2.4.2 Esquema Floco de Neves**

Segundo Rob (2011), para facilitar a navegação do usuário final, utiliza-se a técnica de normalização das tabelas dimensionais. Esse esquema normalizado é conhecido como esquema floco de neves, que nada mais é do que, um tipo de esquema estrela, no qual as tabelas de dimensões podem ter suas próprias tabelas de dimensões. Resumindo, o esquema floco de neve resulta normalmente da normalização de tabelas de dimensão. Por exemplo, se a tabela da dimensão de localização contém dependência transitivas entre região, estado e cidade, é possível normalizar esta tabela dimensão para a 3FN (terceira forma normal). A Figura 2.13 ilustra essa normalização.

Essa normalização, simplifica as operações de filtragem de dados relacionados a dimensão. No entanto, há um preço a pagar por ela, pois aumenta-se a complexidade das consultas **SQL**(Structurered Query Language). Por exemplo, caso se deseje agregar os dados por região, deve-se utilizar uma junção de quatro tabelas.



**Figura 2.13** **T**abelas de dimensões normalizadas. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

De acordo com KIMBALL (2002), também é possível acelerar as operações de consultas criando e mantendo várias tabelas de fatos relacionadas a cada nível de agregação. KIMBALL (2002), define isso como uma **constelação de fatos**. Por exemplo, uma solução usando essa técnica, para o problema apresentado na Figura 2.13, seria criar uma tabela de fato para Vendas\_Região, Vendas\_Estado, Vendas\_Cidade e Vendas\_Localização. Essas tabelas agregadas são computadas previamente na fase de carregamento de dados, e não no momento da execução (Rob, 2011). A finalidade dessa técnica, segundo Rob (2011), é poupar ciclo do processador durante a execução, acelerando, assim, a análise de dados.

De acordo com Passos e Goldschmidt (2005), existem diversos operadores OLAP que permitem acessar os dados em modelos multidimensionais. A seguir encontra-se indicados alguns deles:

* **Drill up/down** – Utilizado para amentar ou reduzir o nível de detalhe da informação acessada. Exemplo: Vendas por país, Vendas por estado, etc.
* **Slicing** – Utilizado para selecionar as dimensões a serem consideradas na consulta. Exemplo: Visualizar as vendas, separadas por país e por mês.
* **Dicing** – Utilizado para limitar o conjunto de valores a ser mostrado, fixando-se algumas dimensões. Exemplo: Vendas de um determinado estado, de um determinado produto em um determinado ano.
* **Pivoting** – Utilizado para inverter as dimensões entre linhas e colunas. Exemplo: Ao visualizar vendas por produto e por estado, aplicar o operador para visualizar as vendas por estado e por produto.
* **Data Surfing** – Executar uma mesma análise em outro conjunto de dados. Exemplo: Ao avaliar as vendas no Brasil, aplicar o operador para realizar a mesma consulta em Portugal.

Ainda de acordo com Rob (2011), quando um sistema de **BI** é implementado em áreas geograficamente dispersas, as técnicas de particionamento e replicação são especialmente importantes. O **particionamento** separa a tabela em subconjunto de linhas ou colunas e coloca esses subconjuntos próximos ao computador cliente, melhorando, dessa forma, o tempo de acesso, por outro lado, a **replicação** faz uma cópia da tabela e a coloca em uma localização diferente, também com a finalidade de aprimorar o tempo de acesso.

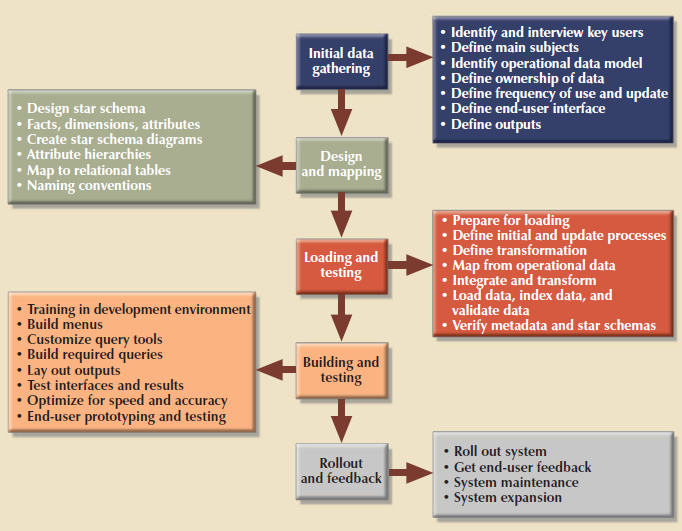
Resumindo, projetar um *data warehouse* significa receber a oportunidade de ajudar a desenvolver um modelo integrado que capture os dados considerados essenciais para a organização, tanto da perspectiva do usuário final, como da perspectiva dos negócios. Para tanto, um projeto de *data warehouse*, deve satisfazer:

* Critérios de integração e carregamento de dados.
* Recursos de análises de dados com desempenho aceitável de consulta.
* Necessidades de análises de dados do usuário final

Segundo Rob (2011), a preocupação técnica mais evidente na implementação de um *data warehouse* é fornecer ao usuário final suporte a decisões com recursos avançados de análise de dados – no momento certo, no formato certo, com os dados certos e ao custo certo. A Figura 2.14 ilustra um processo simplificado de implementação de *data warehouse*.

**2.5 Conclusões**

Neste capítulo foi apresentada uma visão geral sobre *Business Intellingence* (**BI**), onde deu para perceber que, o BI é um conjunto amplo, coeso e integrado de ferramentas e processos utilizados para captar, coletar, integrar, armazenar e analisar dados para a geração e a apresentação de informações que deem suporte à tomada de decisões de negócio.



**Figura 2.14** Mapa do processo de projeto e implementação de data warehouse. **Fonte**: Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

Parte I

Capítulo

3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Fundamentação Teórica

**3. O PROCESSO DE KDD – Knowledge Discovery in Databases**

**3.1 Introdução**

Os constantes avanços na área da tecnologia da informação têm viabilizado o armazenamento de grandes e múltiplas bases de dados. Tecnologias como a Internet, sistemas gerenciadores de banco de dados, dispositivos de armazenamento de dados de maior capacidade e de menor custo e sistemas de informação em geral são alguns dos exemplos que têm viabilizado a proliferação de inúmeras bases de dados de natureza comercial, administrativa, governamental e cientifica (Han & Kamber & Pei, 2011).

Atualmente, dados científicos em projetos de pesquisa, tais como missões espaciais da NASA, monitoramento temporal, em redes sociais, tais como Google, Facebook, etc., de um modo geral, o universo digital, têm alcançado proporções gigantescas, na ordem de zetabytes e ou terabytes de informações.

Diante desse cenário, a análise de grandes quantidades de dados pelo homem é inviável sem o auxílio de ferramentas computacionais apropriadas. Portanto, torna-se imprescindível o desenvolvimento de ferramentas que auxiliem o homem, de forma automática e inteligente, na tarefa de analisar, interpretar e relacionar esses dados para que se possa desenvolver e selecionar estratégias de ação em cada contexto de aplicação (Han; Kamber & Pei, 2011).

Então, para atender a este novo contexto, surge segundo Han; Kamber & Pei (2011), uma nova área denominada ***Data Mining***, que vem despertando grande interesse junto às comunidades cientificas e industrial.

Conforme Han; Kember & Pei (2011), o *Data mining (****DM****),* é um campo que tem conduzido a substanciais pesquisas em relação a mineração em vários tipos de dados, incluindo dados relacionais, dados de data *warehouse*, dados transacionais, dados de séries temporais, dados espaciais, dados textos, multimídia, dados da Web e de arquivos planos (*flat files*). A mineração de dados multidimensionais (também conhecida como *exploratory multidimensional data mining*, **online analytical mining**, ou **OLAM**) integra OLAP com mineração de dados para descobrir conhecimento em bases de dados multidimensionais. Dentre os muitos paradigmas e arquiteturas de sistemas de mineração de dados, a mineração de dados multidimensional, é particularmente importante, pelas seguintes razões:

* **Alta qualidade dos dados no data warehouse**. O data warehouse é uma valiosa fonte de dados de alta qualidade para o OLAP bem como para a mineração de dados.
* **Infraestrutura disponíveis em torno da mineração de dados**. A infraestrutura construída sistematicamente em torno do *data warehouse*, incluindo acesso, integração, consolidação e transformação de múltiplos bancos de dados heterogêneos, conexões ODBC/OLEDB, acesso Web, relatórios e ferramentas OLAP de análise.
* **Exploração de dados multidimensionais baseado em OLAP**. Mineração de dados multidimensionais fornecem facilidades para minerar diferentes subconjuntos de dados em vários níveis de abstração – por cortar e girar o cubo de dados em vários níveis (*drilling*, *dicing*, *slicing*) além de filtrar e girar o cubo os dados em vários ângulos.
* **Seleção online de funções de mineração de dados.** Nem sempre o usuário sabe o que está procurando. Então, por integrar OLAP com várias funções de mineração de dados, a mineração de dados multidimensional fornece usuário flexibilidade para selecionar as funções e tarefas de mineração de dados desejadas dinamicamente.

Não deve-se esquecer que, a finalidade da análise de dados é descobrir previamente características, relacionamentos, dependências ou tendências desconhecidas dos dados. Então, essas descobertas, segundo Rob (2011), tornam-se parte do modelo de informação a partir do qual as decisões são construídas. Portanto, uma típica ferramenta de análise de dados depende de os usuários finais definirem o problema, selecionarem os dados e iniciarem as análises adequadas, de modo a gerar informações que auxiliem na modelagem e resolução dos problemas descobertos por esses usuários.

Embora as ferramentas ***OLAP*** suportem a análise multidimensional e tomada de decisão, ainda segundo Han; Kember & Pei (2011), são necessárias as ferramentas de análise de dados para análises em profundidade, tais como: classificação de dados, associação, sumarização, árvores de decisão, caracterização, e agrupamento de dados (*clusting*), dentre outras.

Portanto, a grande quantidade de dados gerados, coletados ou armazenados, obtidos por operações diárias ou pesquisas cientificas, requer um processo automatizado para descobrir padrões, exceções, tendências ou correlações entre eles. Daí, surge a necessidade da Mineração de dados (Data Mining). Para Han; Kamber & Pei (2011), os desafios nessa área não param de crescer, e que ainda teremos a necessidade de um desenvolvimento ainda maior e que a Mineração de Dados será apenas um capítulo em uma área mais amplas: Ciência dos Dados (*Data Science* – como se registra na literatura Internacional).

O processo de KDD não se resume apenas a mineração de dados, mas esta é considerado, segundo alguns autores, a apenas uma das fases no processo para geração do conhecimento a partir de Bases de Dados. Na sessão seguinte será apresentado todas as fases do processo de KDD.

**3.2 Caracterização do Processo de KDD**

Basicamente, uma aplicação de KDD é composta por três tipos de componentes: o problema em que será aplicado o processo de KDD, os recursos disponíveis para a solução do problema e os resultados obtidos a partir da aplicação dos recursos disponíveis em busca da solução do problema (Passos & Goldschmidt, 2005).

1. **O problema a ser submetido ao processo de KDD** - Este componente pode ser caracterizado por três elementos: conjunto de dados, o especialista do domínio da aplicação e objetivos da aplicação.
2. **Os recursos disponíveis para solução do problema em questão** - Entre eles podem ser destacados: o especialista em KDD, as ferramentas de KDD e plataforma computacional disponível (Passos & Goldschmidt, 2005).
3. **Os resultados obtidos a partir da aplicação dos recursos no problema** - Compreende, fundamentalmente, os modelos de conhecimento descobertos ao longo da aplicação de KDD e o histórico das ações realizadas (Passos & Goldschmidt, 2005).

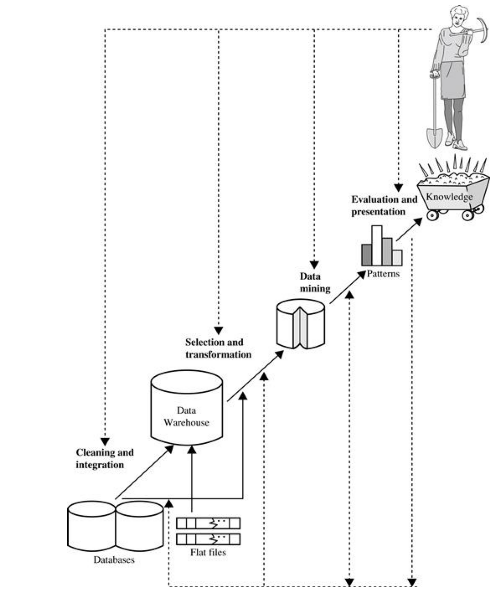
Segundo Han et al. (2011); Witten & Frank (2005); Elmasri (2005), o processo de KDD é mostrado na Figura 3.1 e consiste da sequência iterativa dos seguintes passos: **Seleção de dados, Limpeza dos dados, Enriquecimento, Transformação de dados, Data Mining** e **Avaliação dos padrões e representação do conhecimento**.

Do passo 1 até o passo 4, são as diferentes maneiras de pré-processamento dos dados, onde os dados são preparados para mineração. A etapa de *data mining*, pode interagir com o usuário ou com a base de conhecimento. O conhecimento encontrado, são apresentado ao usuário e pode ser armazenado como um novo conhecimento na base de conhecimento. A **Figura** 3.1, mostra o data mining, como uma etapa no processo de descoberta de conhecimento (*Knowledge Discovery Process*).

Estas etapas ainda podem, segundo alguns autores, serem subdividas em Pré-processamento, Mineração de Dados e Pós-processamento.

**3.3 Pré-processamento**

A etapa de pré-processamento compreende todas as funções relacionadas à captação, à organização e ao tratamento dos dados. Essa etapa tem como objetivo a preparação dos dados para o algoritmo da etapa da Mineração de Dados. As principais funções de pré-processamento são: **Seleção de dados**, **Limpeza dos dados**, **Transformação dos dados**, **Enriquecimento dos dados**, **Partição do Conjunto de Dados** e a **Integração dos Dados**.



**Figura 3.1** Mineração de Dados como um passo no processo de Descoberta de Conhecimento. **Fonte**: Han; Kamber & Pei (2011).

Segundo Witten & Frank (2005), associados a essa etapa, tem-se o conceito de variáveis do problema, também denominado de atributos, que podem ser classificados sob dois aspectos distintos: quanto à representação de seus valores (tipo de dados) e quanto à natureza da informação (tipo de variável). Os tipos de dados indicam a forma em que eles estão armazenados. Os tipos de variáveis expressam a natureza com que a informação deve ser interpretada. Dessa forma, as variáveis podem ser classificadas em (Witten, 2005):

* **Nominais ou Categóricas** – são variáveis utilizadas para nomear ou atribuir rótulos a objetos. Podem assumir valores pertencentes a um conjunto finito e pequeno de estado possíveis. Por exemplo, o gênero de uma pessoa: masculino e feminino. Nesse tipo de variável não há uma ordenação de seus valores ((Passos & Goldschmidt, 2005; Witten, 2005).
* **Discretas** – são similares às variáveis nominais, no entanto, os valores (estados) que elas assumem possuem um ordenamento, e este possui algum significado. O dia da semana é um bom exemplo deste tipo de variável, onde a “segunda-feira” vem após o “domingo” e antes da “terça-feira”, e assim sucessivamente. Podem também, ser intervalos em um espectro contínuo de valores. Por exemplo, faixa de renda e faixa etária, entre outros. De forma similar às variáveis nominais, os valores de variáveis discretas podem ser representadas por tipos de dados alfanuméricos (Passos & Goldschmidt, 2005; Witten, 2005).
* **Contínuas** – são variáveis quantitativas cujos valores possuem uma relação de ordem entre eles. O conjunto de valores de uma variável contínua pode ser finito ou infinito. Como exemplo, pode-se citar renda e idade como variáveis contínuas. Por sua natureza, normalmente os valores das variáveis contínuas são representadas por um tipo de dados numérico (Passos & Goldschmidt, 2005; Witten, 2005).
  + 1. **Seleção de dados**

Essa função, também denominada Redução de Dados, compreende, em essência, a identificação de quais informações, dentre as bases de dados existentes, devem ser efetivamente consideradas durante o processo de KDD. A seleção dos dados pode ter dois enfoques distintos: a escolha de atributos ou a escolha de registros que devem ser considerados no processo de KDD (Passos & Goldschmidt, 2005).

Deve-se mencionar que, em algumas situações, no início do processo de KDD, o especialista de KDD, recebe o conjunto de dados a ser analisado, não estruturados adequadamente para o processo de descoberta de conhecimento. Portanto, é função do especialista de KDD fazer esta estruturação nos dados. Para isto existem algumas técnicas, as quais serão discutas a seguir:

* + - 1. **Redução de Dados Horizontal**

A seleção por redução horizontal é caracterizada pela escolha de casos. Entre as operações de redução de dados horizontal podem ser citadas: amostragem aleatória, eliminação direta de casos, segmentação do banco de dados e agregação de informações (Passos & Goldschmidt, 2005).

* + - 1. **Segmentação do Banco de Dados**

Na segmentação deve-se escolher um ou mais atributos para nortear o processo de segmentação. Suponha que deseje-se apenas analisar os clientes que moram em residência própria. Tal operação poderia ser implementada por uma instrução de seleção em SQL do tipo: “Select \* FROM Cliente WHERE TP\_RES = “P””.

Dessa forma, o conjunto de dados resultantes desta consulta se tornaria o conjunto a ser efetivamente considerado deste ponto em diante no processo de KDD (Passos & Goldschmidt, 2005; Witten, 2005).

* + - 1. **Eliminação Direta de Casos**

Esta operação pode ser interpretada como uma variação da segmentação, e nela são especificados os casos a serem eliminados e não os casos que devem permanecer na análise. Considerando o mesmo exemplo anterior, acima citado, tal operação poderia ser implementada por uma instrução de exclusão em SQL do tipo “DELETE FROM Cliente WHERE TP\_RES <> “P””.

* + - 1. **Amostragem Aleatória**

Esta operação consiste de sortear da base de dados um número preestabelecido de registros de forma que o conjunto resultante possua menos registros do que o conjunto original. Existem várias estratégias de amostragem aleatória. A seguir será ilustrada algumas delas. Suponha que tem-se uma grande massa de dados, contendo N *tuplas*, e que n seja o número de amostras desejadas (n < N) (Passos & Goldschmidt, 2005).

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

Esta operação consiste em reunir (agregar) os dados de forma a reduzir o conjunto de dados original. Na agregação de informações, dados em um nível maior de detalhamento são consolidados em novas informações com menor detalhe. Por exemplo, somar os valores de todas as compras de cada cliente, obtendo o total de despesas por ele realizadas durante um determinado período (Passos & Goldschmidt, 2005).

* + - 1. **Redução de Dados Vertical**

Segundo Passos & Goldschmidt (2005); Han; Kamber & Pei (2011), esta é uma operação de pré-processamento muito importante no processo de KDD. Formalmente, sendo **S** um conjunto de dados com atributos **A1**, **A2**, **A3**, ..., **An**, o problema da redução de dados vertical consiste em identificar qual das 2n combinações entre tais atributos deve ser considerada no processo de descoberta de conhecimento. Ou seja, a redução de dados vertical, também conhecida como redução de dimensão, é implementada pela eliminação ou pela substituição dos atributos de um conjunto de dados. Tem como objetivo procurar encontrar um conjunto mínimo de atributos de tal forma que, a informação original seja preservada. Obviamente, quanto maior for o valor de n, maior será o desafio na escolha dos atributos: o número de possibilidades de subconjuntos de atributos cresce exponencialmente na medida em que n aumenta (Passos & Goldschmidt, 2005).

Ainda segundo Passos & Goldschmidt (2005); Han & Kamber (2011), entre as principais motivações para a aplicação da redução de dados verticais podem ser citadas: (1) um conjunto de atributos bem selecionados pode conduzir a modelos de conhecimento mais conciso e com maior precisão; (2) se o método de seleção dos atributos for rápido, o tempo de processamento necessário para utilizá-lo e, em seguida, aplicar o algoritmo de mineração de dados em um subconjunto dos atributos, pode ser inferior ao tempo de processamento para aplicar o algoritmo de mineração de dados sobre todo o conjunto de atributos; (3) a eliminação de atributo é muito mais significativa em termos de redução do tamanho de um conjunto de dados, do que a exclusão de registros.

Segundo Freitas (2002), existem duas abordagens para a redução de dados vertical, usualmente utilizadas em problemas de Classificação:

* **Abordagem Independente de Modelos** (*Filter*) – nessa abordagem, a seleção de atributos é realizada sem considerar o algoritmo de Mineração de Dados que será aplicado aos atributos selecionados.
* **Abordagem Dependente de Modelo** (*Wrapper*) – essa abordagem consiste em experimentar o algoritmo de Mineração de Dados para cada conjunto de atributos e avaliar os resultados obtidos.

Segundo Han; Kamber & Pei (1999), em quaisquer das abordagens mencionadas acima, há três estratégias clássicas e simples para a escolha do conjunto de atributos, que podem ser utilizadas:

* **Seleção Sequencial para Frente** (*Forward Selection*) – Essa seleção começa com o subconjunto de atributos candidatos vazio. O processo é iterativo. Cada atributo por vez é adicionado ao subconjunto de atributos candidatos, que é avaliado segundo alguma medida de qualidade. Ao final de cada iteração, é incluído no subconjunto de atributos candidatos, aquele atributo que tenha maximizado a medida de qualidade considerada.
* **Seleção Sequencial para Trás** (*Backward Selection*) – É o processo contrário da Seleção Sequencial para Frente: o subconjunto de atributos candidatos começa completo, com todos os atributos do problema. Então, cada atributo é retirado subconjunto, que é avaliado segundo alguma medida de qualidade. Ao final de cada iteração, é excluído do subconjunto de atributos candidatos, aquele atributo que tenha minimizado a medida de qualidade considerada.
* **Combinação das Estratégias Anteriores** – A seleção para frente e a seleção para trás são combinadas de tal forma que, a cada passo do algoritmo, o algoritmo seleciona o melhor atributo (incluindo-o o subconjunto de atributos candidatos) e remove o pior atributo dentre os remanescentes do conjunto de atributos.

De acordo com Han; Kamber & Pei (2011); Freitas (2002), existem estratégias mais interessantes para a escolha do conjunto de atributos a ser utilizados. Como por exemplo, os algoritmos Genéticos, podem ser utilizados no processo de otimização do conjunto de atributos. Outro exemplo, seria os Algoritmos para indução de Árvores de Decisão, tais como **ID3** e **C4.5**, que também podem ser aplicados para selecionar atributos em problemas de classificação. Ainda segundo os autores citados, uma árvore de decisão é construída a partir de dados, de tal forma que todos os atributos que não aparecem na árvore são considerados irrelevantes para o problema.

* + - 1. **Eliminação Direta de Atributos**

Segundo Passos & Goldschmidt (2005), essa tarefa se refere à eliminação de atributos cujo conteúdo não seja relevante ao processo de KDD. A sua utilização depende do conhecimento prévio existente acerca do problema. Essa operação pode ser implementada por uma instrução SQL.

Ainda segundo Passos & Goldschmidt (2005), duas heurísticas podem ser utilizadas para indicar que essa operação deve ser usada:

1. Eliminar todos os atributos que representem valores constantes em todos os registros da base de dados. Uma vez que, atributos nesta situação não contribuem para distinguir os registros uns dos outros, sendo, portanto, dispensáveis do processo de KDD.
2. Eliminar os atributos que sejam identificadores na base de dados em análise. Tal fato, justifica-se, uma vez que essas informações são usadas para identificar unicamente cada registro da base de dados.
   * + 1. **Redução de Valores**

A operação de redução de valores, segundo Passos & Goldschmidt (2005), é uma alternativa interessante à operação de corte de atributos oferecida pela redução de dados vertical. Essa operação consiste em reduzir o número de valores distintos em determinados atributos. Com menos valores, menos comparações são feitas, reduzindo o tempo de processamento de diversos algoritmos de Mineração de Dados. Existem métodos de redução de valores contínuos e métodos de redução de valores nominais.

* + - * 1. **Redução de Valores Nominais**

Essa operação é aplicável somente a variáveis nominais, onde o número de valores é finito, sem ordenação entre os mesmo. A seguir serão apresentados dois métodos (Passos & Goldschmidt, 2005):

* **Identificação de Hierarquia entre Atributos** – Neste caso, o especialista no domínio da aplicação deve apresentar a hierarquia (abstração) existente entre os atributos. Como exemplo, considere um endereço composto pelas seguintes informações: logradouro, bairro, cidade, estado. O especialista da aplicação por meio de uma ordenação, pode definir a hierarquia entre esses atributos como: logradouro ⊂ bairro ⊂ cidade ⊂ estado. A partir desta especificação, pode-se estabelecer um nível de corte do detalhamento da informação.
* **Identificação de Hierarquia entre Valores** – Aqui, o especialista no domínio da aplicação deve apresentar possíveis generalizações para os valores de cada atributo. Como exemplo, considere um conjunto de dados que contenham informações sobre os produtos vendidos: tênis, sapato, sandália, bermudas, calça, camisa, paletó. Então, o especialista pode definir hierarquias para indicar generalizações de conceito com relação aos valores existentes: tênis ⊂ calçado, sapato ⊂ calçado, sandália ⊂ calçado, bermuda ⊂ roupa, calca ⊂ roupa e paletó ⊂ roupa. Em outras palavras, as três primeiras generalizações indicam que tênis, sapato e sandália são casos particulares de calçado. De forma análoga, bermuda, calça e paletó são casos particulares de roupa. A partir desta especialização, os valores originais podem ser substituídos pelas respectivas generalizações, reduzindo um domínio de 7 valores distintos para apenas 2.
  + - * 1. **Redução de Valores Contínuos (ou Discretos)**

De acordo com Passos & Goldschmidt (2005), esse tipo de operação pressupõe aplicação somente a variáveis contínuas ou discretas. Variáveis contínuas ou discretas possuem uma relação de ordenação entre seus valores. A seguir serão apresentados alguns métodos para esse fim.

* **Particionamento em Células (*Bins*) de mesma Cardinalidade (“*Equidepth Bins*”)** – Nesse método, os valores são agrupados em células com o mesmo número de elementos em cada uma delas.
* **Redução de Valores pelas Medianas das Células (“Bin Medians”)** – O mesmo procedimento descrito para o método anterior é realizado neste método. Em seguida, é calculada a mediana de cada uma das células. Os valores originais são substituídos pela mediana associada a cada célula, gerando um novo conjunto de dados.
* **Redução de Valores pelas Médias das Células (“Bin Means”)** - O procedimento deste método é análogo ao descrito para o método anterior. Nesse caso, é calculada a média dos valores em cada uma das células. Os valores originais são substituídos pela média associada a cada célula, gerando um novo conjunto de dados.
* **Redução de Valores pelos Limites das Células (“Bin Boundaries”)** – O mesmo procedimento descrito para o primeiro método é realizado neste método. Em seguida, os valores nos extemos das células são considerados. O procedimento calcula a distância de cada valor em relação aos extremos de cada célula. O valor original é substituído pelo valor do extremo mais próximo, gerando um novo conjunto de dados.
* **Arredondamento de Valores** – O arredondamento de valores, também chamado de aproximação de valores, é uma função comum em nosso dia-a-dia. A seguir é apresentada uma alternativa para arredondamento de valores em números inteiros. A variável x indica o valor original a ser arredondado. A variável y recebe o resultado intermediário do procedimento de arredondamento. O parâmetro k é o número de casas decimais à direita a ser considerado no arredondamento.

**(3.1)**

* **Agrupamento de Valores (“Agrupamento”)** – Consiste em agrupara os valores de um atributo em clusters (grupos) levando em consideração a similaridade existente entre tais valores. O processo de Agrupamento, que será detalhado em sessões seguintes, consiste em procurar minimizar a similaridade dos dados pertencentes ao mesmo cluster e maximizar a similaridade dos dados em clusters distintos. Uma vez concluído o processo de agrupamento, cada cluster pode passar a ser representado pelo média dos valores a ele atribuído. Deve-se ressaltar que o problema da redução de valores pode ser interpretado como um problema de otimização.

De acordo com Passos & Goldschmidt (2005), duas heurísticas podem ser utilizadas para auxiliar na escolha de qual, dentre os métodos acima, deve ser utilizado:

* Quando o número de células for de moderado a grande, recomenda-se a utilização dos métodos de redução de valores pela mediana ou pela média.
* Quando o número de células for pequeno, recomenda-se a utilização do método de redução de valores pelos limites das células.
  + 1. **Limpeza dos Dados**

Abrange qualquer tratamento realizado sobre os dados selecionados de forma a assegurar a qualidade (completude, veracidade e integridade) dos fatos por eles representados. Informações ausentes ou inconsistentes nas bases de dados devem ser corrigidas de forma a não comprometer a qualidade dos modelos de conhecimento a serem extraídos ao final do processo de KDD (Passos & Goldschmidt, 2005).

Em aplicações reais, é comum que os dados sobre os quais se deseja extrair algum conhecimento estejam incompletos, ruidosos ou inconsistentes. Os dados são considerados incompletos se há informações ausente para determinados atributos ou ainda se há dados pouco detalhados. Por outro lado, dados ruidosos são dados errados ou que contenham valores considerados divergentes, denominados de *outliers*, do padrão normal esperado. Dados inconsistentes são aqueles que contêm algum tipo de discrepância semântica entre si (Passos & Goldschmidt, 2005).

É importante perceber que a qualidade dos dados possui grande influência na qualidade dos modelos de conhecimento a serem abstraídos a partir destes dados. Quanto pior for a qualidade dos dados informados ao processo de KDD, pior será a qualidade dos modelos de conhecimento gerados (GIGO, Garbage in, Garbage out) (Passos & Goldschmidt, 2005).

Para se fazer a limpeza dos dados, existem algumas funções. A seguir serão discutidas as algumas delas:

* + - 1. **Limpeza de Informações Ausentes**

Esta função compreende a eliminação de valores ausentes em conjunto de dados. Alguns métodos para preenchimento de valores ausentes estão descritos a seguir:

* **Exclusão de Casos** – Consiste de excluir do conjunto de dados as *tuplas* que possuam pelos menos um atributo não preenchido.
* **Preenchimento Manual de Valores** – em geral, essa abordagem demanda alto consumo de tempo e recursos, sendo muitas vezes inviável na prática. Esse método pode ser implementado por meio de pesquisas junto às fontes de dados originais que procurem captar as informações ausentes.
* **Preenchimento com Valores Globais Constantes** – esse método consiste em substituir todos os valores ausentes de um atributo por um valor padrão tal como “desconhecido” ou “null”. Esse valor padrão pode e deve ser especificado pelo especialista no domínio da aplicação.
* **Preenchimento com Medidas Estatísticas** – medidas estatísticas podem ser empregadas como alternativa à utilização de constantes padrão no processo de preenchimento de valores ausentes. Como exemplo de medidas estatísticas para preenchimento de informações ausentes pode ser citados: média para atributos numéricos e moda para atributos categóricos.
* **Preenchimento com Métodos de Mineração de Dados** – nesse caso, modelos preditivos podem ser construídos de forma a sugerir os valores mais prováveis a serem utilizados no preenchimento de valores ausentes. Algoritmos de Mineração de Dados tais como Redes Neurais, Estatística (modelos Bayesiano) e Árvores de Decisão são alternativas na construção destes modelos.
  + - 1. **Limpeza de Inconsistências**

De acordo com Passos & Goldschmidt (2005), essa função compreende a identificação e a eliminação de valores inconsistentes em conjunto de dados. Uma inconsistência pode envolver uma única tupla, ou um conjunto de *tuplas*. Inconsistência em uma única *tupla* ocorre quando houver divergência entre os valores desta *tupla*. Nesta situação, a participação do especialista no domínio da aplicação é fundamental na identificação de inconsistências. Além do mais, esse processo demanda conhecimento prévio acerca do problema. Existem alguns métodos para limpeza de inconsistências, como descrito a seguir:

* **Exclusão de Casos** – Consiste em excluir do conjunto de dados original, as *tuplas* que possuem pelo menos uma inconsistência. A identificação dos casos com inconsistência pode ser obtida por meio de instruções SQL, cujas restrições especifiquem o tipo de inconsistência a ser verificada.
* **Correção de Erros** – Esse método consiste em substituir valores errôneos ou inconsistentes identificados no conjunto de dados. Pode envolver desde a correção manual até a atualização desses valores em um lote predeterminado de registros.
  + - 1. **Limpeza de Valores não Pertencentes ao Domínio**

Segundo Passos & Goldschmidt (2005), essa função compreende a identificação e a eliminação de valores que não pertençam ao domínio dos atributos do problema. Ela pode ser considerada um caso particular da operação anterior e demanda um conhecimento prévio do domínio de cada atributo. A seguir temos alguns dos métodos utilizados para limpeza de valores não pertencentes ao domínio:

* **Exclusão de Casos** – Consiste em excluir do conjunto de dados original, as tuplas que possuam pelo menos um valor fora do conjunto de valores válidos de cada atributo.
* **Correção de Erros** – Consiste em substituir os valores inválidos identificados no conjunto de valores. Pode envolver desde a correção manual até a atualização destes valores em um lote predeterminado de registros utilizando instruções SQL.
  + 1. **Transformação dos Dados**

Segundo Passos & Goldschmidt (2005), codificação de dados é a operação de pré-processamento responsável pela forma como os dados serão representados durante o processo de KDD.

Deve-se salientar que os dados devem ser codificados de forma a atender às necessidades específicas dos algoritmos de Mineração de Dados. Por exemplo, uma rede neural requer que os dados estejam em uma representação numérica. Dessa forma, se a base de dados a ser processada apresente valores nominais, estes devem ser codificados antes de serem submetidos a rede (Passos & Goldschmidt, 2005).

A maneira como a informação é codificada tem forte influência sobre o tipo de conhecimento a ser encontrado. Em essência, a codificação pode ser: Numérica – Categórica, que transforma valores reais em categorias ou intervalos; ou Categórica – Numérica, que representa numericamente valores de atributos categóricos (Passos & Goldschmidt, 2005).

* + - 1. **Codificação: Numérica – Categórica**
* **Mapeamento Direto** – Consiste na simples substituição dos valores numéricos por valores categóricos. Por exemplo, sexo: 1 🡪 M, 0 🡪 F.
* **Mapeamento em Intervalos** – também chamado Discretização, a representação em intervalos pode ser obtida a partir de métodos que dividam o domínio de uma variável numérica em intervalos. Existem diversos métodos para ser fazer sito, como por exemplo, dividir em intervalos com comprimentos definidos pelos usuário, dividir em intervalos de comprimentos iguais ou até mesmo dividir em intervalos por meio de agrupamentos (clusters) (Passos & Goldschmidt, 2005).
  + - 1. **Codificação: Categórica – Numérica**
* **Representação Binária Padrão** – Nesta representação, cada valor categórico é associado a um valor de 1 até N e é representado por uma cadeia de dígitos binários. Por exemplo, se temos 5 possíveis valores, pode-se representa-los com cadeias binárias de comprimento 3, como mostrado na Tabela 3.1:

**Tabela 3.1** Representação Binária Padrão

|  |  |
| --- | --- |
| Valores originais | Representação binária padrão |
| Casado | 001 |
| Solteiro | 010 |
| Viúvo | 100 |
| Divorciado | 011 |
| Outro | 110 |

* **Representação Binária 1-de-N** – Nessa representação, o código 1-N tem um comprimento igual ao número de categorias discretas permitidas para a variável, onde cada elemento na cadeia de bits é 0, exceto para um único elemento: aquele que representa o valor da categoria em questão, como ilustrado na Tabela 3.2.

**Tabela 3.2** Representação Binária 1-de-N

|  |  |
| --- | --- |
| Valores originais | Representação binária 1-de-N |
| Casado | 00001 |
| Solteiro | 00010 |
| Viúvo | 00100 |
| Divorciado | 01000 |
| Outro | 10000 |

* **Representação Binária por Temperatura** – Essa representação é utilizada mais frequentemente quando os valores discretos estão relacionados de algum modo. Por exemplo, uma variável discreta pode ter um dos seguintes valores: fraco, regular, bom e ótimo. Nesse caso, existe uma graduação entre os valores. Onde o valor ótimo é o melhor valor e fraco o pior valor. Dessa forma, a diferença entre os conceitos fraco e ótimo deve ser a maior possível entre os valores. Pode-se considerar que cada valor corresponde a um acréscimo de bit igual a 1 na representação, conforme mostra a tabela abaixo, como mostra a Tabela 3.3.

**Tabela 3.3** Representação Binária por Temperatura

|  |  |
| --- | --- |
| Valores originais | Representação binária por temperatura |
| Fraco | 0001 |
| Regular | 0011 |
| Bom | 0111 |
| Ótimo | 1111 |

De acordo com Passos & Goldschmidt (2005), uma medida de distância normalmente utilizada conjuntamente a esta representação é a distância de ***Hamming*** (**DH**), também conhecida como *City-Block*. Essa distância expressa a diferença entre duas cadeias de bits, adicionando uma unidade sempre que bits de mesma posição possuem valores distintos. A tabela a seguir mostra a distância de *Hamming* entre os conceitos.

**Tabela 3.4** Distância Hamming entre os conceitos.

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

* + 1. **Enriquecimento dos dados**

De acordo com Passos & Goldschmidt (2005), a função de enriquecimento consiste em conseguir de alguma forma mais informações que possa ser empregada aos registros existentes, enriquecendo os dados, para que esses forneçam mais informações para o processo de descoberta de conhecimento. A seguir será comentada algumas das operações mais usualmente utilizadas no processo de enriquecimento das bases de dados.

* **Pesquisas** – essa operação inclui todas as iniciativas de enriquecimento que envolve a captação de novas informações junto às fontes originais. Normalmente requerem a inclusão de novos atributos ou mesmo de novas tabelas. Busca-se, no enriquecimento, agregar novas informações.
* **Consultas a Bases de Dados Externas** – o processo de enriquecimento pode ser realizado mediante a incorporação de informações fornecidas por outros sistemas. É muito comum a importação de informações advindas de outras bases de dados.
  + 1. **Partição do Conjunto de Dados**

A Mineração de Dados, conforme será detalhada na próxima seção, é a etapa do processo de KDD responsável pela abstração de modelos de conhecimentos a partir dos dados existentes (Passos & Goldschmidt, 2005). A qualidade desses modelos precisam ser avaliada. A avaliação de um modelo de conhecimento requer a confrontação deste com dados visando à mensuração de algumas medidas que expressem a qualidade deste modelo. Para que essa avaliação seja isenta, os dados utilizados na construção do modelo não devem ser os mesmos utilizados na avaliação desse modelo. Portanto, pelo menos dois conjuntos de dados devem ser utilizados no processo de KDD: um conjunto de treinamento e um conjunto de teste. Como, em geral, tem-se um conjunto de dados, a operação de Partição do conjunto de dados em treinamento e teste assume grande importância. A seguir estão indicados alguns métodos utilizados na partição do conjunto de dados, com vistas à posterior avaliação dos modelos de conhecimento gerados.

* **Holdout** – esse método divide aleatoriamente os registros em uma percentagem fixa p para treinamento e (1 – p) para teste, considerando normalmente p > ½. Segundo Rezende (2002), embora não existam fundamentação teórica sobre esta percentagem, valores tipicamente utilizados são: p = 2/3 e (1 – p) = 1/3. Utilizado quando deseja-se produzir um único modelo de conhecimento.
* **Validação Cruzada com K Conjuntos** (*K-Fold CrossValidation*) – Esse método consiste em dividir aleatoriamente o conjunto de dados com N elementos em K subconjuntos disjuntos (folds), com aproximadamente o mesmo número de elementos (N/K). Neste processo, cada um dos K subconjuntos é utilizado como conjunto de teste e os (K – 1) demais subconjuntos são utilizados como treinamento. Assim, o processo é repetido K vezes, sendo gerados e avaliados K modelos de conhecimento.
* **Validação Cruzada com K Conjunto Estratificada** (*Stratifield K-Fold CrossValidation*) – Aplicável em problemas de classificação, este método é similar à Validação Cruzada com K Conjuntos, sendo que ao gerar os subconjuntos mutuamente exclusivos, a proporção de exemplos em cada uma das classes é considerada durante a amostragem. Isso significa que, se o conjunto de dados original possui duas classes com distribuição de 20% e 80%, cada subconjunto também deverá conter aproximadamente esta mesma proporção de classes.
* **Leave-One-Out** – Esse é método é caso particular da Validação Cruzada com K conjuntos, em que cada um dos K subconjuntos possui um único registro. É computacionalmente dispendioso e frequentemente usado em pequenas amostras.
* **Bootstrap** – O conjunto de treinamento é gerado a partir de N sorteios aleatórios e com reposição a partir do conjunto de dados original (contendo N registros). O conjunto de teste é composto pelos registros do conjunto de dados original não sorteados para o conjunto de treinamento. Esse método consiste em gerar os conjuntos, abstrair e avaliar o modelo de conhecimento um número repetido de vezes, a fim de estimar uma média de desempenho do algoritmo de Mineração de dados.
  + 1. **Integração dos Dados**

De acordo com Han; Kamber & Pei (2011); Witten (2005), os dados para o processo de KDD, são selecionados de diversas fontes distintas, como já foi mencionado anteriormente. Então nesta fase, faz-se a integração de várias fontes de dados, mantendo a consistência e a coerência dos dados integrados.

* 1. **Mineração de Dados**

**3.4.1 Definição**

Existem, segundo Han; Kamber & Pei (2011), várias definições para Data Mining, por ser um assunto verdadeiramente interdisciplinar, ela envolve um extenso campo de pesquisa, que associa técnicas e conceitos de diversas áreas como sistemas de banco de dados, sistemas baseados em conhecimento, inteligência artificial, aprendizado de máquina, aquisição do conhecimento, estatística, bancos de dados espaciais e visualização de dados. As várias tarefas desenvolvidas em Mineração de Dados têm como objetivos primário a predição e/ou a descrição. A predição usa atributos para predizer valores futuros de uma ou mais variáveis (atributos) de interesse. A descrição contempla o que foi descoberto nos dados de vista da interpretação humana.

Durante a etapa de Mineração de Dados é realizada a busca efetiva por conhecimentos úteis no contexto da aplicação de KDD. É por tanto, na Mineração de Dados onde são definidas as técnicas e os algoritmos a serem utilizados no problema em questão. A escolha da técnica depende, muitas vezes, do tipo de tarefa de KDD a ser realizada. São muitas as tarefas de Mineração de dados, mas nesse trabalho será dados atenção especial as tarefas **Associação**, **Classificação**, **Árvores de Decisão** e **Agrupamento**, pois essas as pretendidas a serem usadas nessa pesquisa para desenvolvimento do Caso de Uso.

De acordo com Han; Kamber & Pei (2011); Passos & Goldschmidt (2005), a execução da etapa de Mineração de Dados (Data Mining) compreende a aplicação de algoritmos sobre os dados procurando abstrair conhecimento. Estes algoritmos são fundamentados em técnicas que procuram, segundo determinados paradigmas, explorar os dados de forma a produzir modelos de conhecimento.

Mas antes de apresentar as tarefas de KDD, será apresentado um conceito muito importante no processo de Mineração de dados, que é a noção de **similaridade**. Suponha que o conjunto de dados possa ser interpretado como 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 (Passos & Goldschmidt, 2005).

O conceito de distância é formalizado como uma função D : E x E 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) ≤ D(x,z) + D(z,y)

Alguns exemplos de distância mais geralmente usados:

* **Distância Euclidiana**:

**(3.2)**

* **Distância de Hamming (City-Block)**:

**(3.3)**

* **Distância de Minkowski**:

**(3.4)**

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 dados no conjunto de dados.

* + 1. **Tarefas de Mineração de Dados**

As metas primárias que podem ser alcançadas através da Mineração de Dados, são as seguintes (Fayyad, et al, 1996):

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

Segundo Fayyad, et al (1996), a mineração preditiva consiste na generalização de exemplos ou experiências passadas com respostas conhecidas ou regras de negócios estabelecidas por especialistas. Enquanto, a mineração descritiva, consiste na identificação de comportamentos intrínsecos do conjunto de dados, sendo que estes dados não possuem uma classe especifica.

* + - 1. **Descoberta de Associações**

De uma forma geral, a tarefa clássica de busca por regras de associação, também denominada regras associativas, foi introduzida em (Agrawal et al., 1993). Intuitivamente essa tarefa consiste em encontrar conjunto de registros de itens que ocorram simultaneamente e de forma frequente em um banco de dados.

Uma **regra de associação** é uma implicação da forma X => Y, onde X = {x1, x2, ..., xn} e Y = {y1, y2, ..., ym} são conjunto de itens, com xi e yj sendo itens distintos para todo i e todo j. Essa associação estabelece que se um cliente comprar X, ele também estará propenso a comprar Y. Para que uma regra de associação seja de interesse para a mineração de dados, a regra precisa satisfazer algumas medidas. Duas medidas de interesse comuns fornecem suporte e confiança. Segue a seguir, as formulas para cálculo do fator de suporte e de confiança (Agrawal, et al., 1996; Elmasri, 2005).

**(3.5)**

**, onde N é o número total de tuplas.**

**(3.6)**

O fator de **suporte** pode ser descrito como a probabilidade de uma transação qualquer satisfazer tanto X como Y, ao passo que o fator de **confiança** é a probabilidade de que uma transação satisfaça Y, dado que ela satisfaça X. a tarefa de descobrir regras de associação consiste em extrair do banco de dados todas as regras com “Fs” e “Fc” maiores ou iguais a um “Fs” e “Fc” especificado pelo analista de dados (Agrawal, et al., 1996).

Uma associação é considerada frequente se o número de vezes em que a união de conjuntos de itens (x ocorrer em relação ao número total de transações do banco de dados for superior a uma frequência mínima (denominada suporte mínimo) que é estabelecida em cada aplicação. Busca-se por meio do suporte, identificar que associações surgem em uma quantidade expressiva a ponto de ser destacada das demais existentes (Agrawal, et al., 1996).

Ainda segundo Agrawal et al. (1996), uma associação é considerada válida se o número de vezes em que X Y ocorrer em relação ao número de vezes que X ocorrer for superior a um valor denominado confiança mínima, que também é estabelecida em cada aplicação. A medida de confiança procura expressar a qualidade de uma regra, indicando o quanto a ocorrência do antecedente da regra pode assegurar a ocorrência do consequente desta regra.

Desta forma, a tarefa de Descoberta de Associações ou Descoberta de Regras de Associações, pode ser definida formalmente como a busca por **regras de associação frequentes e válidas** em um banco de dados, a partir da especificação dos parâmetros de suporte e confiança mínima (Agrawal et al., 1996).

Os valores dos parâmetros de suporte e confiança mínima devem ser especificados pelo especialista em KDD em conjunto com o especialista no domínio da aplicação (Agrawal et al., 1996).

Segundo Agrawal et al. (1996), existem diversos algoritmos desenvolvidos especificamente para aplicação na tarefa de descoberta de associações, dentre eles: Apriori, DHP (Direct Hashing and Pruning), Parttion, DIC (Dynamic ItemSet Counting), Eclat, MaxEclat, Clique, MaxClique, etc. além dos mais, existem versões destes algoritmos para sistemas distribuídos.

* + - 1. **Classificação**

Essa tarefa pode ser compreendida como a busca por uma função que permita associar corretamente cada entrada **X**i de um conjuntos de dados de entrada, a um único rótulo **Y**i, denominado classe. Uma vez identificada, essa função pode ser aplicada a novas entradas de forma a prever a classe em que tais dados se enquadram. Dados podem ser associados a classes ou a conceitos através de um processo de discriminação ou caracterização (Han; Kamber & Pei, 2011).

De acordo com Han; Kamber & Pei (2011), a tarefa de classificação é um processo de dois passos. No primeiro passo, constrói-se um modelo com base nos dados. No segundo passo, determina se a acurácia desse modelo é aceitável, se assim for, usa-se esse modelo para classificar novos dados.

Para formalizar a tarefa de classificação, consideremos um par ordenado da forma **(x, ƒ(x))**, onde **x** é um vetor de entradas n-dimensional e **ƒ(x)** a saída de uma função **ƒ**, desconhecida, aplicada a **x**. A tarefa de inferência indutiva consiste em, dada uma coleção de exemplos de **ƒ*,*** obter uma função ***h*** que se aproxime de **ƒ**. A função ***h*** é chamada de hipótese ou modelo de ***ƒ*** (Passos, Goldschmidt, 2005).

Nos casos em que a imagem de **ƒ** é formada por rótulos de classes, a tarefa de inferência indutiva é denominada classificação e toda hipótese ***h*** chamada de classificador. A identificação da função ***h*** consiste em um processo de busca no espaço de hipóteses **H**, pela função que mais se aproxime da função original **ƒ**. Esse processo é denominado **aprendizado** (Russel e Norving, 1955, citado por Passos & Goldschmidth, 2005). Todo algoritmo que possa ser utilizado nesse processo é denominado de **algoritmo de aprendizado**. O conjunto de todas as hipótese que podem ser obtidas a partir de um algoritmo de aprendizado **L** é representado por **H**L. Cada hipótese pertencente ao **H**L é representado por ***h***L.

A acurácia de um classificador em um dado conjunto de teste é a percentagem do conjunto de tuplas de testes que são corretamente classificadas pelo classificador Han; Kamber & Pei (2011), ou seja, a acurácia da hipótese ***h*** em mapear corretamente cada vetor de entradas x em **ƒ(x)**. O conjunto de pares **(x, ƒ(x))** utilizado na identificação da função ***h*** é denominada conjunto de treinamento. Por outro lado, o conjunto de pares **(x, ƒ(x))** utilizados para avaliar a acurácia de ***h*** é denominado conjunto de testes. Dessa forma, o algoritmo **L** pode ser interpretado como uma função tal que:

, onde T é o espaço composto por todos os conjuntos de treinamento possíveis para **L**.

Segundo Utgolff (1996), cada algoritmo possui *bias* indutivo que direciona o processo de construção dos classificadores. O *bias* indutivo de um algoritmo pode ser definido como o conjunto de fatores que coletivamente influenciam na seleção de hipótese.

Em termos práticos, o *bias* de um algoritmo de aprendizado **L** afeta o processo de aprendizado de duas formas: restringem o tamanho de espaço de hipóteses **HL**, e impõem uma ordem de preferência sobres as hipóteses em **HL** (Bensusan, 1999, citado por Utgolff, 1996).

Conforme mencionado anteriormente, uma medida de desempenho de um classificador comumente utilizada é a acurácia (Acc(*h*)), também conhecida como precisão do classificador.

**(3.7)**

Acc(*h*) = 1 – Err(*h*), onde:

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

**(3.8)**

Onde:

O operador ||E|| retorna 1 se a expressão E for verdadeira e 0, caso contrário;

**n** é o número de exemplos (registros da base de dados, por exemplo);

**yi** é a classe real associada ai i-ésimo exemplo;

***h*(i)** é a classe indicada pelo classificador para o i-ésimo exemplo.

O modelo derivado pode ser apresentado de várias formas, tais como regras de classificação (***IF-THEN***), árvores de decisão, fórmulas matemáticas, ou redes neurais. Existem muitos outros métodos de construção de modelos de classificação, tais como classificação ***naïve Bayesian***, máquina de vetor de suporte (***support vector machines***), e classificação do vizinho mais próximo (***k-nearest neighbor***), Backpropagation, classificação usando Padrões frequentes, etc (Han; Kamber & Pei, 2011).

**3.4.2.3 Agrupamento (Clustering)**

Técnicas de agrupamento e classificação objetivam realizar uma separação ótima entre objetos de uma coleção, permitindo a descoberta de novos padrões, previamente desconhecidos. O resultado da segmentação, independentemente da ferramenta utilizada, pode ser interpretado eficientemente por um especialista na área de origem dos dados sob análise. A facilidade de visualização resultante do agrupamento, favorece a análise (Han; Kamber & Pei, 2011).

A tarefa de agrupamento, é usada para segmentar os dados de entrada em subconjuntos ou clusters, de tal forma que elementos de um cluster compartilhem um conjunto de propriedades comuns que os distingam dos elementos de outros clusters. O objetivo desta tarefa é maximizar similaridades intracluster e minimizar similaridades intercluster. Diferente da classificação que tem rótulos predefinidos, o agrupamento precisa automaticamente identificar os rótulos. Por essa razão, o agrupamento também é chamada de indução não supervisionada (Passos & Goldschmidt, 2005).

Segundo Han; Kamber & Pei (2011), em geral, a classe não é representada nos dados de treinamento, simplesmente porque elas não são conhecidas no começo. Entretanto, os agrupamentos (*clusters*) podem ser usados para gerar tais rótulos. Os objetos são agrupados baseados no princípio de máxima similaridade intra-classe e mínima similaridade inter-classe. Isto é, o agrupamento de objetos são formados de modo que os objetos dentro do agrupamento tenham alta similaridade em comparação com um outro objeto, mas são muitos diferentes dos objetos em outro agrupamento. Cada agrupamento formado pode ser visto como uma classe de objetos, da qual pode derivar regras. Agrupamento também facilita a formação de taxonomia (**taxonomy formation**), isto é, a organização das observações em uma hierarquia de classes que agrupem eventos similares juntos.

Formalmente para o processo de agrupamento, supõe-se a existência de **n** pontos de dados **x1**, **x2**, ..., **xn** tais que cada ponto pertença a um espaço d dimensional **Rd**. A tarefa de agrupamento desses pontos de dados, separando-os em **k** clusters consiste em encontrar **k** pontos **mj** em **Rd** de tal forma que a expressão (Passos & Goldschmidt, 2005):

**(3.13)**

Seja minimizada, onde **d2(xi, mj)** denota uma distância entre **xi** e **mj**. Os pontos **mj** são denominados centroides ou médias dos clusters.

De forma resumida, o problema descrito acima consiste em encontrar k centroides de clusters de tal maneira que a distância entre cada ponto de dado e o centroide do cluster mais próximo seja minimizada (Passos & Goldschmidt, 2005).

De acordo com (Passos & Goldschmidt, 2005), para que os algoritmos de agrupamento possam efetuar sua tarefa é necessário que sejam utilizadas estruturas de dados capazes de armazenar os objetos a serem processados. Algoritmos de agrupamento que trabalham com dados armazenados na memória principal, normalmente, utilizam uma das seguintes estruturas de dados no seu processamento:

* **Matrizes de dados** – As linhas representam cada um dos objetos a serem agrupados e as colunas, os atributos ou características de cada objeto. Considerando **n** objetos, cada qual com p atributos, obtém-se uma matriz **n x p** como ilustrado abaixo:
* **Matriz de similaridade** – Cada elemento da matriz representa a distância entre pares de objetos. Visto que a distância entre o objeto **i** e o objeto **j** é igual a distância entre o objeto **j** e o objeto **i**, não é necessário armazenar as distâncias entre os objetos. Considerando **n** objetos a serem agrupados, obtém-se uma matriz quadrada de tamanho **n x n** como a ilustrada a seguir:

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

Segundo Han; Kamber & Pei (2011), quando um algoritmo que trabalha com matrizes de similaridade recebe um matriz de dados, ele primeiro a transforma em uma matriz de similaridade antes de iniciar o processo de agrupamento.

Agrupamento ou Clustering é também chamado de ***data segmentation*** em algumas aplicações, porque segmenta grande conjunto de dados em grupos de acordo com suas similaridades. O Agrupamento também pode ser usado em detecção de desvio (outlier detection), onde desvios (valores “distantes” de qualquer cluster) pode ser mais interessante do que os casos comuns. Aplicações de detecção de desvio inclui a detecção de fraudes em cartões de crédito e o monitoramento de atividades criminais em comercio eletrônico (Han; Kamber & Pei, 2011).

Segundo Han; Kamber & Pei (2011), o agrupamento é um campo de pesquisa desafiador e que, para sua utilização no Data Mining são necessários alguns requisitos, tais como:

* **Escalabilidade**: muitos algoritmos de agrupamento trabalham bem com pequenos data sets, contendo pouco mais de 100 objetos de dados; embora, um grande banco de dados, possa conter milhões ou bilhões de objetos, particularmente no cenário da Web. Fazer agrupamento em uma amostra de um grande *DataSet* pode levar a resultado tendencioso. Portanto, os algoritmos de agrupamento necessitam de alta escalabilidade.
* **Habilidade para lidar com diferentes tipos de atributos**: muitos algoritmos são projetados para cluster de dado numérico (baseado em intervalo). Embora, as aplicações possam requerer outros tipos de dados, tais como binário, nominal (categórico), e dados ordinal, ou uma mistura desses tipos. Recentemente, mais e mais aplicações necessitam de técnicas de agrupamento para tipos de dados complexos tais como gráficos, imagens, sequencias e documentos.
* **Descobrir clusters de forma arbitrária:** muitos algoritmos de agrupamento determinam os clusters baseados em medidas de distância Euclidiana ou Manhattan. Algoritmos baseados em tais medidas de distância tendem a encontrar clusters esféricos com tamanho e densidade similares. É importante desenvolver algoritmos que possa detectar clusters de forma arbitrária.
* **Requisitos para determinar os parâmetros de entrada para o domínio do conhecimento**: muitos algoritmos de agrupamento exigem que os usuários forneçam o conhecimento na forma de parâmetros de entrada, como o número desejado de clusters. Consequentemente, o resultado pode ser que os clusters sejam sensíveis aos parâmetros. Parâmetros são difíceis de serem determinados.
* **Habilidade para lidar com dados ruidoso**: muitos conjuntos de dados do mundo real contêm dados errados, ausência de dados, os dados desconhecidos. Como por exemplo, um sensor de leitura de dados, onde em algumas leituras pode ser imprecisas.
* **Clustering incremental e sensitivo a ordem de entrada dos dados**: em muitas aplicações, pode haver atualizações a qualquer momento. Alguns algoritmos de agrupamento não incorporam a capacidade de atualização incremental de clusters, tendo de recalcular tudo do zero. Algoritmos de agrupamento também podem ser sensíveis a ordem de entrada de dados. Isto é, dado um conjunto de objetos, o algoritmo de agrupamento pode retornar diferentes clusters dependendo da ordem em que os objetos são apresentados. Portanto, são necessários algoritmos que sejam incrementais e sensíveis a ordem de entrada dos dados.
* **Capacidade de clustering de alta dimensionalidade:** um conjunto de dados pode conter muitas dimensões ou atributos. Quando agrupamos documentos, por exemplo, cada palavra-chave pode ser considerada uma dimensão, e frequentemente existem milhares de palavras-chave. Muitos algoritmos de agrupamento são bons em manipular conjunto de dados de poucas dimensões, tais como conjuntos envolvendo duas ou três dimensões.
* **Clustering com base em restrições:** aplicações do mundo real pode necessitar de realizar cluster em vários tipos de restrições. Suponha que em seu trabalho, você necessite determinar a localização para novas *automatic teller machine* (**ATM**) em uma cidade. Para decidir sobre isso, você pode agrupar considerando restrições como grupos familiares, redes rodoviárias e número de pessoa por cluster.
* **Interpretabilidade e usabilidade:**  os usuários esperam que o resultado do agrupamento seja interpretável, compreensível, e usável. Isto é, o agrupamento, pode necessitar ser atrelado uma interpretação e semântica especifica e aplicação. Isto é importante para estudar como uma meta da aplicação pode influenciar a seleção das características de agrupamento e métodos de agrupamento.

De acordo com Han; Kamber & Pei (2011), existem várias técnicas e métodos de agrupamento. Entre os principais algoritmos de agrupamento podem ser citados: K-Means, Fuzzy K-Means, K-Modes e K-Medoid. Entretanto, os métodos de agrupamento mais conhecidos e utilizados são os métodos particionamento, os métodos hierárquicos, os métodos baseados em densidade e os métodos baseados em Grade (*Grid*). Na sessão seguinte, serão discutidos alguns desses métodos.

**3.4.3 Métodos de Mineração de Dados**

Nesta sessão trata-se dos métodos de Mineração de Dados, mais é importante destacar que muitos deles são utilizados em outros tipos aplicações.

De acordo com Passos & Goldschmidt (2005), cada método de Mineração de Dados requer diferentes necessidades de pré-processamento. Tais necessidades variam em função do aspecto extensional da base de dados em que o método será utilizado. Em decorrência da diversidade de métodos de pré-processamento de dados, são muitas as alternativas possíveis de combinações entre métodos. A escolha dentre estas alternativas pode influenciar na qualidade do resultado do processo de KDD (Morik, 2000; Engels, 1996; Engels et al., 1997).

De acordo com Passos & Goldschmidt (2005), os métodos de KDD, sendo os métodos de Mineração de Dados um caso particular, podem ser considerados operadores definidos a partir de precondições e efeitos. Uma precondição de um método de KDD é um predicado que estabelece um requisito que deve ser cumprido antes da execução do método. Um efeito de um método de KDD também é um predicado que descreve uma situação gerada após a aplicação do método. Um plano de ação de KDD válido é toda sequência de métodos de KDD onde as precondições para execução de cada um dos métodos da sequência sejam devidamente atendidas.

**3.4.3.1 Métodos Baseados em Redes Neurais**

Segundo Han; Kember & Pei (2011); Passos & Goldschmidt (2005); Rezende (2003), diversos modelos de Redes Neurais podem ser utilizados na implementação de métodos de Mineração de Dados. Classificação, Regressão, Previsão de Séries Temporais e Agrupamento (Clusters) são exemplos de tarefas de Mineração de Dados que podem ser implementadas por métodos de Redes Neurais. Além do mais, alguns modelos de Redes Neurais podem ser aplicados em mais de um tipo de tarefa de Mineração de dados.

A topologia da rede neural varia em função do problema e da representação adotada para os dados. De uma forma geral, em aplicações de Mineração de Dados, a camada de entrada do modelo neural recebe os dados pré-processados de cada registro de um banco de dados. A rede processa esses dados produzindo uma saída cuja natureza também varia em função da aplicação (Passos & Goldschmidt, 2005).

Ainda de acordo com Passos & Goldschmidt (2005), em redes neurais com aprendizado supervisionado, a entrada corresponde aos atributos preditivos enquanto a saída do modelo corresponde ao atributo objetivo do problema. Então, o algoritmo de aprendizado pode estimar o erro, ou distância, entre a saída produzida pela rede e a saída desejada. Dessa forma, o algoritmo, em função do erro calculado, ajusta os pesos das conexões da rede a fim de tornar a saída real tão próxima quanto seja possível da saída desejada. Estes tipos de modelos neurais são muito úteis em geral para reconhecimento de padrões e para tarefas de Mineração de Dados que envolvam predição.

As Redes Neurais com aprendizado não supervisionado são adequados para tarefas que envolvam descrição de dados, como por exemplo, a tarefa de Agrupamento. Nestes casos, não há saída desejada (Passos & Goldschmidt, 2005).

Alguns algoritmos de aprendizado e indicadas a tarefa de Mineração de dados, são:

* **Back-Propagation** – O algoritmo Back-Propagation, também conhecido como algoritmo de retro-propagação do erro, é um algoritmo de aprendizado supervisionado, cuja aplicação é adequada a tarefa de Mineração de Dados tais como Classificação, Regressão ou Previsão. Esse algoritmo tem como objetivo minimizar a função de erro entre a saída gerada pela rede neural e a saída real desejada, utilizando o método do gradiente descendente.
* **Kohonen** – O mapa Kohonen pertence à classe das Redes Neurais Auto organizáveis. Em uma Rede Neural Auto organizável o treinamento é não supervisionado, geralmente baseado em uma forma de competição entre os elementos processados. Entre as principais aplicações das Redes Auto organizáveis estão:
  + **Tarefas de Clusterização –** Tarefa na qual os dados de entrada devem ser agrupados em conjuntos que agregam padrões semelhantes.
  + **Detecção de Regualaridades** – Modelo em que o sistema deve extrair as características relevantes dos padrões de entrada.

Segundo Passos & Goldschmidt (2005), o método de aprendizagem mais comum nas Redes Neurais Auto Organizáveis é denominado aprendizado por competição (*Competitive Learning*). O qual, consiste em uma forma de aprendizado que divide o conjunto de padrões de entrada em grupos inerentes aos dados. Para isto, esse método considera, em sua abordagem mais simples, que os neurônios de saída da rede competem entre si, resultando em apenas um neurônio vencedor.

**3.4.3.2 Métodos Estatísticos**

De acordo com Passos & Goldschmidt (2005); Han; Kamber & Pei (2011), diversos algoritmos de Mineração de Dados são fundamentados em princípios e teorias da Estatística. A seguir serão apresentados alguns deles:

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

Formalmente, sejam **X(A1, A2, ..., An, C)** um conjunto de dados, **C1**, **C2**, ..., **Cn**, as classes do problema (valores possíveis do atributo **C**) e R um novo registro que deve ser classificado. Sejam ainda **a1**, **a2**, ..., **an** na os valores que **R** assume em **X**. O calculador Bayesiano possui dois passos:

1. Calcular a probabilidade **P(C=Ci/R)**, **i**=1,2, ..., k.
2. Indicar com saída do algoritmo a classe **Cj** tal que **P(C=Cj/R)** seja máxima.

* **K-Means** – O algoritmo k-means é um método popular da tarefa de agrupamento. Toma-se, randomicamente, k pontos de dados (dados numéricos) como sendo os centroides (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 centroide de cada cluster é a menor dentre todas as distâncias calculadas. Um novo centroide para cada cluster é computado pela média dos pontos do cluster, caracterizando a configuração dos clusters para a iteração seguinte. O processo termina quando os centroides dos clusters param de se modificar, ou após um número limitado de iterações que tenham sido especificado pelo usuário (Han, Kamber & Pei, 2011; Passos & Goldschmidt, 2005).

Formalmente, segundo Han; Kamber & Pei (2011), suponha um conjunto de dados **D**, contendo **n** objetos em um espaço Euclidiano. O método de particionamento, distribui os objetos em **D** entre k clusters, **C1**, ..., **Ck**, isto é, e para (. A execução do algoritmo ***k-means*** toma um parâmetro de entrada k, e divide um conjunto de n objetos em k clusters tal que a similaridade intracluster resultante seja alta, mas a similaridade intercluster seja baixa. A similaridade em um cluster é medida em respeito ao valor médio dos objetos neste cluster (centro de gravidade do cluster). O método *k-means* é inicializado com os centros (médias) colocados em posições aleatórias. A busca pelo centro comum se faz de forma iterativa. Após essa inicialização, os objetos restantes são agrupados conforme a distância em que se encontram das médias. A diferença entre um objeto , e **ci,** a representatividade do cluster, é medida por *dist* (**p**, **c**), onde *dist* (**x**, **y**) é a distância Euclidiana entre dois pontos **x** e **y**. A qualidade do cluster Ci pode ser medido pela variação intra-cluster, a qual é definida pela soma

**(3.14)**

Onde **E** é a soma do quadrado dos erros para todos os objetos no conjunto de dados; **p** é o ponto no espaço Euclidiano representando um dado objeto; e **ci** é o centroide do cluster (ambos **p** e **ci** são multidimensionais). Em outras palavras, para cada objeto em cada cluster, a distância do objeto ao centro do cluster é quadrada, e as distâncias são somadas (Han; Kamber & Pei, 2011).

* **K-Modes** – O algoritmo k-modes é uma variação do método k-means, só que utilizado para agrupamento de dados categóricos (nominais). Em geral, no lugar do cálculo da média, calcula-se a moda dos objetos, usando medidas de similaridades para tratar objetos categóricos, e usando métodos baseados em frequência para atualizar as modas dos clusters.
* **K-Prototypes** – O método *k-prototypes* é a integração dos métodos *k-means* e *k-modes*. Esse método pode ser aplicado a bases de dados que contenham tanto atributos numéricos quanto atributos categóricos.
* **K-Medoids** - O algoritmo k-medoids baseia-se, primeiramente, em encontrar os medoids (objetos mais centralmente localizado em um cluster). Os objetos restantes são então agrupados com o medoid ao qual ele é mais similar. Há então uma troca iterativa, de um medoid por um não medoid, visando à melhoria do agrupamento. O método então, é realizado baseado no princípio de minimizar a soma das dissimilaridades entre cada objeto **p** e seu correspondente objeto representativo. Isto é, um critério absoluto de erro (***absolute-error criterion***), definido como

**(3.15)**

Onde **E** é a soma dos erros absolutos para todos objetos **p** no conjunto de dados; e **Oi** é o objeto representativo de **Ci**. Isto é o básico para o método ***k-medoids***, os quais os grupos de **n** objetos em **k** clusters, minimizando o erro absoluto.

Os algoritmos **PAM**, **CLARA** e **CLARANS** são os principais algoritmos da classe *k-medoids*. O primeiro (**PAM**) é bem ineficiente, o último (**CLARANS**) é bem mais eficiente, mais ainda tem o problema da dificuldade de detectar clusters de formatos arbitrários.

**3.4.3.3 Método Específico - Apriori**

O *Apriori* é um algoritmo clássico de Mineração de Regras de Associação (Agrawal, 1993). Diversos algoritmos tais como GSP, DHP, Partition, DIC, etc., foram inspirados no funcionamento do *Apriori* e se baseiam no princípio da antimonotonicidade do suporte. Segundo este princípio, “Um *k-itemset* somente pode ser frequente se todos os seus (k-1)-*itemsets* forem frequentes”. Assim sendo, a combinação de *itemsets* para gerar um novo *itemset* somente ocorre quando estes são frequentes.

Segundo Passos & Goldschmidt (2005), tais algoritmos podem ser decompostos basicamente em duas etapas:

1. Encontrar todos os conjuntos d itens frequentes (que satisfaçam à condição suporte mínimo).
2. A partir do conjunto de itens frequentes, gerar as regras de associação (que satisfazem à condição de confiança mínima).

**3.4.3.4 Métodos Baseados em Indução de Árvores de Decisão**

Segundo Passos & Goldschmidt (2005), alguns dos principais métodos de Mineração de Dados são baseados na construção de árvores de decisão a partir da base de dados. Em geral a construção de uma árvore de decisão é realizada segundo alguma abordagem recursiva de particionamento da base de dados. Um exemplo clássico de método baseado na indução de árvores de decisão é o algoritmo C4.5.

* **C4.5** – O C4.5 procura abstrair árvores de decisão a partir de uma abordagem recursiva de particionamento das bases de dados. Utiliza, para tanto, conceitos e medidas da Teoria da Informação.

Uma árvore de decisão é um modelo de conhecimento em que cada nó interno da árvore representa uma decisão sobre um atributo que determina como os dados estão particionados pelos seus nós filhos. Inicialmente, a raiz da árvore contém toda a base de dados com exemplos misturados das várias classes. Um predicado, denominado ponto de separação, é escolhido como sendo a condição que melhor separa ou discrimina a classe. Tal predicado envolve exatamente um dos atributos do problema e divide a base de dados em dois ou mais conjuntos, que são associados cada um a um nó filho. Cada novo nó abrange, portanto, uma partição da base de dados que é recursivamente separada até que conjunto associado a cada nó folha consista inteiramente ou predominantemente de registros de uma mesma classe Passos & Goldschmidt (2005).

De acordo com Passos & Goldschmidt (2005), a maioria dos algoritmos baseados na abstração de árvores de decisão consiste de duas fases: Construção da Árvore de Decisão e Simplificação da Árvore de Decisão.

**3.4.3.5 Métodos Baseados em Lógica Nebulosa**

Diversos métodos de Mineração de dados foram adaptados de forma a incorporar a flexibilidade proporcionada pela Lógica Nebulosa. Entre eles pode-se citar as versões nebulosas K-Means e do C4.5. Nessas versões, os registros da base de dados podem pertencer a diversos clusters e classe simultaneamente, com diferentes graus de pertinência. Um exemplo de algoritmo nebuloso é o algoritmo de Wang-Mendel, concebido para aplicação na tarefa de Previsão de Séries Temporais (Passos & Goldschmidt, 2005).

**3.4.3.6 Métodos Hierárquicos**

Agrupamento hierárquico produz uma fusão sequencial aninhada das variáveis em estudo, baseado em algumas métricas de correlação ou semelhança, como a correlação de Pearson ou a distância Euclidiana. A fusão aninhada é representada por um “dendrograma". No nível mais baixo do dendrograma, cada variável é um membro de um cluster (grupo) individual. No primeiro passo, as variáveis mais semelhantes são fundidas em um cluster. Em seguida, as próximas duas variáveis mais similares são unidas em outro cluster. Em cada passo, os dois grupos mais semelhantes (incluindo conjuntos unitários) são unidos para formar um grande grupo. No nível mais alto do dendrograma, existe um conjunto contendo todas as variáveis (Han; Kamber & Pei, 2011).

A correlação de Pearson entre dois experimentos X e Y é dada por:

**(3.16)**

Onde N é o número de variáveis, Xavg é a média das variáveis no experimento X, e Yavg é a média das variáveis no experimento Y. A correlação de Pearson é uma medida de similaridade comumente usada entre dois vetores, portanto, um menos a correlação é usada como distância métrica.

A distância Euclidiana entre dois vetores é dada por:

**(3.17)**

Onde xi são as variáveis do experimento X e yi são as variáveis do experimento Y.

Além destas métricas, podemos citar ainda a métrica de Minkowski, que tem a distância Euclidiana como um caso particular. Outras versões populares da métrica de Minkowski são as distâncias de Manhattan e de Chebyshev. Outra métrica de distância cujo cálculo é extremamente simples é a distância Camberra que calcula a soma das diferenças fracionárias entre as coordenadas de pares de objetos (Linden,2009).

Uma outra técnica usual de agrupamento é o mapa auto-organizável (*self-organized map* - **SOM**) ou mapa de Kohonen, que é um algoritmo de agrupamento em redes neurais (Amaratunga; Cabreira, 2004). Este permite uma melhor visualização e identificação de agrupamentos similares como também da correlação entre as amostras.

O **SOM** transforma dados de alta dimensão em imagens de uma ou duas dimensões, onde o agrupamento pode ser identificado. É um algoritmo versátil e a visualização dos resultados pode ser feita de diferentes maneiras.

O K-means é também uma heurística de agrupamento não hierárquico que busca minimizar a distância dos elementos a um conjunto de forma iterativa. Podemos citar ainda os Grafos que são modelos matemáticos que representam relações entre objetos. Um grafo é um conjunto dado por G=(V, E), onde V é um conjunto finito de pontos, normalmente denominados de nós ou vértices e E é uma relação entre vértices, ou seja, um conjunto de pares em VxV (Linden, 2009).

De acordo com Han; Kamber & Pei (2011), seja qual for o método a ser usado, a necessidade principal é medir a distância entre dois clusters, onde cada cluster é geralmente, um conjunto de objetos.

Existem outros algoritmos hierárquicos tais como **BIRCH**: ***Multiphase Hierarchical Clustering*** usando recurso de clusters em árvores, o **Chameleon**: ***Multiphase Hierarchical Clustering*** usando modelagem dinâmica e ***Probabilistic Hierarchical Clustersing***. Esses algoritmos não serão discutidos nesse trabalho.

**3.4.3.7 Métodos Baseados em Densidade**

Os métodos baseados em densidade permitem descobrir grupos de formatos arbitrários. Estes métodos consideram grupos como sendo regiões densas de objetos no espaço de dados separados por regiões de baixa densidade, que geralmente representam ruídos, onde esses ruídos estão incluídos nos clusters (Han; Kamber & Pei, 2011). Pode-se citar como exemplos de algoritmos baseados em densidade **DBSCAN**, **OPTICS** e **DENCLUE**, que serão discutidos em seguida.

* **DBASCAN** - O algoritmo **DBSCAN** (*Density-based Spatial Clustering of Applications with Noise*) cria grupos de objetos em regiões de alta densidade e é capaz de descobrir grupos de formatos arbitrários em bases de dados espaciais e com ruído. Para o **DBSCAN** um cluster é definido como um conjunto máximo de pontos densamente conectados. Ele encontra clusters com formatos (***shape***) arbitrários em bancos de dados espaciais, contendo ruídos (*outliers*) (Han; Kamber & Pei, 2011).

O algoritmo DBSCAN distingue três tipos de objetos em um conjunto de dados:

1. **Pontos centrais:** são pontos que estão no interior de uma região densa, onde existe pelos menos ***η*** pontos no raio ***ε*** desse objeto. A cardinalidade desses pontos em relação ao parâmetro de ***ε*-vizinhança** deve ser de no mínimo ***η*** pontos;
2. **Pontos de borda (border point):** estão na fronteira de uma região densa, ou seja, sã pontos que estão na **ε-vizinhança** de algum ponto central, porém não são pontos centrais, pois a cardinalidade desses pontos em relação ao raio *ε* não excede *η*;
3. **Outliers (noise point):** esses pontos não são centrais e nem de borda e assim não são conectados por densidade a nenhum outro ponto, não pertencendo a nenhum *cluster*.

Caso um ponto for classificado como ***outlier***pelo algoritmo, posteriormente ele pode estar na *ε*-vizinhança de outro ponto não visitado ainda pelo DBSCAN. Sendo assim, essa classificação pode ser removida caso o objeto seja diretamente alcançável por densidade a partir de um ponto central ainda não visitado (Han; Kamber & Pei, 2011).

Segundo Han; Kamber & Pei, se for usado um índice espacial, a complexidade computacional do **DBSAC** é O(***n.* log *n***), onde ***n*** é o número de objetos de dados. Caso contrário, a complexidade é O(**n**2). Para configurações dos parâmetros apropriados pelo usuário, o algoritmo é efetivo para encontrar clusters de forma arbitrária.

* **OPTICS (***Ordering Points to Identify the Clustering Structure***)** - Embora o **DBSACN** possa encontrar clusters dado os parâmetros de entrada tais como **ε-vizinhança** (o raio máximo de uma vizinhança) e MintPts (o número de pontos requeridos na vizinhança de um *core objects*), isso transfere para os usuários a responsabilidade de selecionar os valores dos parâmetros que irão levar a descoberta de clusters aceitáveis. Isto é um problema, que também estar associados a muitos outros algoritmos de clusters. A configuração de tais parâmetros são usualmente definidos empiricamente, especialmente para parâmetros do mundo real, conjunto de dados de alta dimensionalidade. Além do mais, muitos algoritmos são sensíveis para esses valores de parâmetros: configurações levemente diferentes pode levar a muitos diferentes agrupamento de dados (Han; Kamber & Pei, 2011). Para superar a dificuldade em usar os parâmetros na análise de clusters, foi proposto um método chamado **OPTICS**.

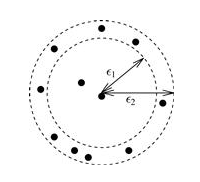
O método **OPTICS** não produz explicitamente um conjunto de clusters de dados. Em vez disso, ele exibe ***cluster ordering***. Isto é, uma lista linear de todos os objetos que estão sobre análise e representa a estrutura de clusters baseado na densidade (*density-based clustering structure*) dos dados. Isto equivale a agrupamento obtida a partir de uma ampla e variada configuração de parâmetros. A ordenação de clusters pode ser usada para extrair informações básicas sobre os clusters (por exemplo, o centro do cluster, ou a forma arbitrária dos clusters), derivar a estrutura intrínseca do cluster, bem como proporcionar a visualização do cluster (Han; Kamber & Pei, 2011).

De acordo com Han; Kamber & pei (2011), para construir diferentes clusters simultaneamente, os objetos são processados em uma ordem especifica. Os objetos selecionados são aqueles que estão em uma distância menor do que **ε,** assim os clusters com alta densidade irão ser construídos primeiro.

A estrutura do algoritmo OPTICS é muito similar ao DBSCAN. Consequentemente, os dois algoritmos têm a mesma complexidade. A complexidade é O(*n.* log *n*) se é usado um índex espacial, e O(n2) caso contrário, onde **n** é o número de objetos.

* **DENCLUE** (*DENsity-based CLUstEring*) - É um método de agrupamento baseado num conjunto de funções de distribuição de densidade. Primeiro é dado uma estimativa da densidade, e então descreve o algoritmo DENCLUE.

Nos métodos DBSCAN e OPTICS, a densidade é calculada contando o número de objetos em uma vizinhança definida pelo parâmetro raio ε. Tal estimativa de densidade pode ser altamente sensível ao valor do raio. Por exemplo, na Figura 3.2, a densidade muda significativamente quando o raio aumenta por um pequeno valor.



**Figura 3.2** Estimando a densidade no DBSAN e OPTPICS. Aumentando o raio de ε1 para ε2, resulta em uma densidade muito maior. Fonte: (Han; Kamber & Pei, 2011).

Segundo Han; Kamber & Pei (2011), para superar este problema, pode ser usado estimativa de densidade de núcleo (***kernel density estimation***), que é uma abordagem da estatística de estimativa de densidade não-paramétrica. A ideia geral por trás da estimativa de densidade de núcleo é simples. Trata-se um objeto observado com um indicador de densidade de alta probabilidade na região circundante. A densidade probabilística de um ponto depende da distância a partir desse ponto para o objeto observado.

Formalmente, seja **x1**, ..., **xn** seja uma amostra independente e identicamente distribuída de uma variável aleatória ƒ. A função de densidade de núcleo de probabilidade estimativa é

**(3.20)**

Onde K() é o núcleo e h especifica o grau de suavização de um objeto no espaço de parâmetros. O núcleo pode ser considerado como uma função que modela a influência de um ponto dentro de uma amostra. Tecnicamente, um núcleo K() é uma função integrável não negativa que satisfaz dois requisitos: e K(-u) = K(u) para todos os valores de u. Uma função núcleo frequentemente utilizada é a função de núcleo Gaussiano com uma média 0 e uma variância de 1:

**(3.21)**

O DENCLUE usa um núcleo Gaussiano para estimar a densidade baseada no dado conjunto de objetos a serem agrupados. Um ponto **x**\* é chamado um ***density attractor*** se ele é máximo local da função de densidade estimada (Han; Kamber & Pei, 2011). Para evitar pontos máximo locais triviais, DENCLUE usa um limite de ruído ξ, e somente considera os atratores de densidade **x**\* tais que ) ≥ ξ. Estes não triviais atratores de densidade são centro de clusters.

Os objetos analisados são associados para o cluster através do atrator de densidade usando um procedimento de subida. Para um objeto x, o procedimento de subida inicia de x e é guiada pelo gradiente da função de estimativa de densidade. O atrator de densidade para x é calculado como

**(3.22)**

Onde é um parâmetro para controlar os passos da convergência, e

**(3.23)**

O procedimento de subida para k > 0 se (), e associar x para o atrator de densidade **x**\* = **x**k. Um objeto é um ruído se ele converge no procedimento de subida para um máximo local x\* com ) ≥ ξ.

De acordo com Han; Kamber & Pei (2011), um cluster é um conjunto de atrator de densidade X e um conjunto de objetos input C tais que cada objeto em C é associado para o atrator de densidade em X, e existe um caminho entre cada par de atrator de densidade onde a densidade é acima de ξ. Por usar múltiplos atratores de densidade conectados por caminhos, **DENCLUE** pode encontrar clusters de forma arbitrária.

Para Han; Kamber & Pei (2011), **DENCLUE** tem várias vantagens. Ele pode ser considerado com uma generalização de vários bem-conhecidos métodos de agrupamento, tais como *single-linkage* e DBSACN. Além disso, DENCLUE é invariante contra ruídos. A função de estimativa de densidade de núcleo pode efetivamente reduzir a influência de ruído por distribuir uniformemente o ruído nos dados de entrada (input).

**3.4.3.8 Métodos Baseados em Grade**

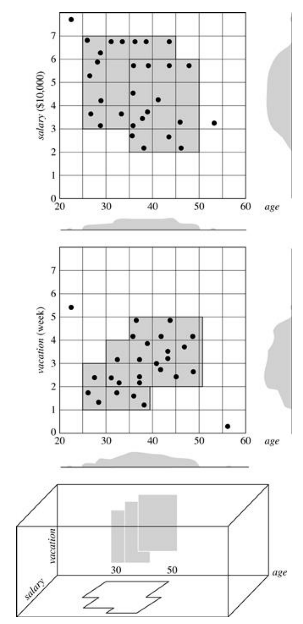
Os métodos baseados em grade (***Grid***) dividem os objetos em um número finito de células que formam uma estrutura de grade multidimensional na qual todas as operações de agrupamento são realizadas. A principal vantagem do método é que as operações independem do número de objetos da base de dados, e sim do número de células da estrutura da grade, o que melhora o desempenho dos algoritmos baseados nesta heurística. Como exemplos de algoritmo baseado em grade pode-se citar o algoritmo CLIQUE e STING (Han; Kamber & Pei, 2011).

* **CLIQUE** - O algoritmo CLIQUE (Clustering in Quest), baseado em grade e em densidade, segmenta o conjunto de dados em subespaços (grade de células) para encontrar agrupamentos suficientemente densos. Cada grade organiza um conjunto de dados, separando os valores contínuos de cada atributo em um número de intervalos discretos. Por fim, cada objeto é atribuído a uma célula a qual seu intervalo contém o valor original do objeto. Os agrupamentos são formados a partir da junção de células densas adjacentes (Han; Kamber & Pei, 2011; Oliveira, 2007).

Segundo Han; Kamber & Pei (2011), o método CLIQUE particiona cada dimensão em intervalos não sobrepostos, dividindo, assim, todo espaço de incorporação dos objetos de dados em células.

A principal estratégia por trás do método CLIQUE é encontrar todas as células nos espaços unidimensionais correspondentes a cada atributo produzido pelas projeções em um dado número de partes iguais. São mantidas somente células que contenham número de elementos acima de um patamar (*MinPts*), denominados de células densa (Han; Kamber & Pei, 2011).

No caso geral, o algoritmo faz a identificação dos clusters em cada um dos atributos. Os atributos que não tem clusters, ou seja, que não atendem ao critério (MinPts), são eliminados. Os atributos restantes são combinados entre si, verificando-se a existência de clusters nesses espaços de mais alta dimensão. O processo continua até que não haja mais formação de clusters. Como resultado, os grupos são formados pela união de hiper-retângulos entre subespaços projetados atendendo fundamentalmente ao critério de densidade (MinPts). Considerando a Figura 3.3, onde o espaço de dado incorporado contém três dimensões: idade (*age*), salário (*salary*) e férias (*vacation*). Uma célula de 2-D, que está no subespaço formado por idade e salário, contém L pontos somente se a projeção desta célula em todas as dimensões, ou seja, idade e salário, respectivamente, contém pelo menos l pontos (Han; Kamber & Pei, 2011).



**Figura 3.3** Unidades densas encontradas em relação as dimensões age, salary e vacation, interceptadas para fornecer um espaço de busca candidato para unidade s densas de alta dimensionalidade. **Fonte**: Han; Kamber & Pei (2011).

* **STING** - STING (Statistical INformation Grid-based) é uma técnica de agrupamento no qual a área espacial dos objetos de entrada é dividida em células retangulares. O algoritmo inicia dividindo a área espacial em níveis de células retangulares, correspondendo a níveis de resolução, formando uma estrutura hierárquica. As células de níveis mais alto são compostas a partir de células de nível mais baixo. Isso gera uma estrutura hierárquica das células produzidas pela subdivisão consecutiva do espaço. A Figura 3.4 mostra uma estrutura hierárquica para agrupamento STING. Os parâmetros estatísticos das células de mais alto nível podem ser facilmente calculadas a partir dos parâmetros das células de mais baixo nível. Os parâmetros são os seguintes: o parâmetro independente, *count*; e os parâmetros dependentes, média (*mean*), *stdev* (desvio padrão), *min* (mínimo), *max* (máximo), e os tipos de distribuição que os valores das células, como segue normal, uniforme e exponencial, ou nenhum (se a distribuição não é conhecida). Quando os dados são carregados do banco de dados, os parâmetros count, mean, stdev, min e max são diretamente calculados.



**Figura 3.4** Estrutura hierárquica para agrupamento STING. **Fonte**:(Han; Kamber & Pei, 2011).

A Figura 3.4, ilustra o processo da subdivisão de espaços e da consequente formação da hierarquia das células efetuado pelo algoritmo STING. A estrutura hierárquica de células representa a informação em diferentes níveis de agrupamento. Em cada nível, são registradas informações estatísticas básicas sobre os atributos de cada célula da grade como: frequência, média, desvio padrão, valor máximo e valor mínimo.

O algoritmo STING tem a propriedade de gerar bons resultados de agrupamento em um curto espaço de tempo de execução, entretanto existem duas grandes dificuldades com este algoritmo. Em primeiro lugar, o desempenho de STING depende da granularidade do nível mais baixo da estrutura de grade. Em segundo lugar, os grupos resultantes são somente limitados horizontalmente ou verticalmente, não existindo qualquer limitação na diagonal. Esta lacuna pode afetar significativamente a qualidade dos grupos (Han, Kamber & Pei, 2011).

De acordo com Han, Kamber & Pei (2011), uma interessante propriedade do STING é que ele aproxima-se do resultado de agrupamento DBSCAN se a granularidade é baixa.

Resumindo, a etapa de Extração de Padrões é direcionada ao cumprimento dos objetivos definidos na Identificação do problema. Nessa etapa é realizada a escolha, a configuração e execução de um ou mais algoritmos para extração de conhecimento. Como esta etapa é um processo iterativo, pode ser necessário que seja executada diversas vezes para ajustar o conjunto de parâmetros visando à obtenção de resultados mais adequados aos objetivos preestabelecidos (Rezende, 2005).

A disponibilização do conjunto de padrões extraídos nesta etapa ao usuário ou a sua incorporação a um Sistema Inteligente ocorre após a análise e/ou o processamento dos padrões na etapa de Pós-processamento.

* 1. **Pós-processamento**

Segundo Liu & Hsu (1996), a obtenção do conhecimento não é o passo final do processo de Extração de Conhecimento de Bases de dados. O conhecimento extraído pode ser utilizado na resolução de problemas da vida real, seja por meio de um Sistema Inteligente ou de um ser humano como apoio a algum processo de tomada de decisão. Para isso é importante que algumas questões sejam respondidas aos usuários:

* O conhecimento extraído representa o conhecimento do especialista?
* De que maneira o conhecimento do especialista difere do conhecimento extraído?
* Em que parte o conhecimento do especialista está correto?

Em geral, os algoritmos de Extração de Padrões podem gerar uma quantidade enorme de padrões, muitos dos quais podem não ser importantes, relevantes ou interessantes para o usuário. Então, fornecer ao usuário uma grande quantidade de padrões descobertos não é produtivo pois, normalmente, ele procura uma pequena lista de padrões interessantes. Portanto, é de vital importância desenvolver algumas técnicas de apoio no sentido de fornecer os usuários apenas os padrões mais interessantes (Goldschmidt, 2003).

Segundo Rezende (2005), diversas medidas para avaliação de conhecimento têm sido pesquisadas com a finalidade de auxiliar o usuário no entendimento e na utilização do conhecimento adquirido.

O Processo de Extração de Conhecimento, tem como um dos principais objetivos, é que o usuário possa compreender e utilizar o conhecimento descoberto. Entretanto, podem ocorrer casos em que os modelos são muito complexos ou não fazem sentido para os especialistas (Pazzani, 2000). Assim, a compreensibilidade do conhecimento extraído é um aspecto bastante importante para o processo de Extração de Conhecimento.

A compreensibilidade de um dado conjunto de regras está relacionada com a facilidade de interpretação dessas regras por um ser humano. Segundo Pazzini (2000), a compreensibilidade de um modelo pode ser estimada, por exemplo, pelo número de regras e o número de condições por regra. Nesse caso, quanto menor a quantidade de regras de um dado modelo e menor o número de condições por regra, maior será a compreensibilidade das regras descobertas.

A maneira de avaliar a qualidade tentando estimar o quanto de conhecimento interessante (ou inesperado) existe, deve combinar fatores numa medida que reflita como o especialista julga o padrão (Piatetsky-Shapiro & Matheus, 1994). As medidas de interesse estão baseadas em vários aspectos, principalmente na utilidade que as regras representam para o usuário final do processo de Extração de Conhecimento (Dong & Li, 1998). Estas medidas podem ser divididas em objetivas e subjetivas (Silberschatz & Tuzhilin, 1995; Piatetsky-Shapiro & Matheus, 1994).

Segundo Horst (1999), medidas objetivas são aquelas que estão relacionadas somente com a estrutura de padrões e do conjunto de dados de teste. Elas não levam em consideração fatores específicos do usuário nem do conhecimento do domínio para avaliar um padrão. Algumas medidas de interesse são: modelos de regras, cobertura de regras mínimas, custo da classificação incorreta.

Como no Processo de Extração de Conhecimento, pode ter diferentes usuários que, podem ter diferentes graus de interesse para um determinado padrão, medidas subjetivas são necessárias. Estas medidas consistem em fatores específicos do conhecimento do domínio e de interesse do usuário, que devem ser tratadas ao selecionar um conjunto de regras interessantes ao usuário final. Algumas regras subjetivas são inesperabilidade e utilidade (Silberschatz & Tuzhilin, 1995).

Segundo Rezende (2005), em um ambiente para avaliação de conhecimento, aspectos objetivos de interesse podem ser utilizados como um primeiro filtro para selecionar regras potencialmente interessantes ao usuário. Os aspectos subjetivos pode ser utilizados como um filtro final para selecionar regras realmente interessantes.

Após a análise do conhecimento, caso este não seja de interesse do usuário final ou não cumpra com os objetivos propostos, o processo de extração pode ser repetido ajustando-se os parâmetros ou melhorando o processo de escolha dos dados para a obtenção de resultados melhores numa próxima iteração (Rezende, 2005).

Existem várias estratégias propostas na literatura para pós-processar o conhecimento descoberto, entre elas a atribuição de medidas de potencial grau de interesse as quais são organizadas em dois grupos, ditas user-driven e data-driven (Silberschatz & Tuzhilin, 1996), (Freitas, 1998).

Outra estratégia é proposta por Hussain et al. (2000) que constitui um método que identifica, a partir de um conjunto de padrões descobertos, um subconjunto de regras que representam regras de exceção e, além disso, atribui uma medida de interesse para cada regra. A Tabela 3.5 mostra a estrutura geral das regras de exceção. Nesta tabela A, B e C são conjuntos não-vazios de itens de dados associados e o símbolo “¬” denota a negação lógica. É importante observar que uma regra de exceção é uma especialização de uma regra geral e uma regra de exceção associa a um item de dados que nega aquele identificado pela regra geral. Este método assume que regras de senso comum representam padrões conhecidos pelo usuário, tendo em vista que aquelas regras têm uma grande cobertura, ao contrário das regras de exceção, que em geral são desconhecidas, uma vez que elas têm baixa cobertura. Sendo assim, as regras de exceção tendem a ser surpreendentes, dado o fato de representarem uma contradição em relação à regra de senso comum. É importante observar que a regra de referência auxilia na explicação da causa da regra de exceção.

**Tabela 3.5 Estrutura de Regras de Exceção**.

|  |
| --- |
| A 🡪 C regra geral (alta cobertura e alta confiança) |
| A, B 🡪 ¬ C regra de exceção (baixa cobertura, alta confiança) |
| B 🡪 ¬ C regra de referência (baixa cobertura e/ou baixa confiança) |

Formalmente, a medida proposta por Hussain et al. (2000) é definida da seguinte forma:

**(3.24)**

Quanto maior o valor da medida de interesse, maior é a chance de a regra ser surpreendente.

Outra estratégia para o Pós-processamento é o filtro de regras de associação, que objetiva selecionar aquelas que associem alguns elementos previamente selecionados (ou descartados) pelo especialista. Esta estratégia, a partir da redução do conjunto de regras, além de facilitar a análise elo especialista também melhora significativamente o desempenho de algoritmos que a partir deste conjunto reduzido venha executar outras estratégias de Pós-processamento. A parametrização do processo de filtro é a partir dos identificadores dos itens de dados que compõem as regras, conforme exemplo:

Regra 1: A 🡪 C

Regra 2: A, B 🡪 D

Regra 3: C 🡪 E

Regra 4: A, D 🡪 C

Supondo que o especialista não esteja interessando em regras em que apresentem o item “D”, independentemente se este consta do antecedente ou do consequente, apenas as Regras 1 e 3 serão selecionadas. Vale destacar novamente que essa estratégia somente é interessante quando o especialista tem conhecimento prévio do que deseja que seja contemplado ou eliminado, caso contrário, padrões interessantes podem ser eliminados. Uma alternativa a esta funcionalidade seria eliminar os itens de dados da base, porém esta alternativa onera computacionalmente a etapa de Pré-processamento, exigindo que um novo conjunto de dados seja construído a cada experimento (Milani & Carvalho, 2013).

Ainda segundo (Milani & Carvalho, 2013), existem ainda outras estratégias para a eliminação de regras extraídas que não agregam novos conhecimentos ao especialista, como por exemplo, a eliminação de redundância. Por exemplo, a partir do conjunto das Regras 1, 2, 3 e 4 percebe-se uma redundância da Regra 4 em relação a Regra 1, ou seja, a Regra 4 já está contemplada pela Regra 1, desta forma a Regra 4 pode ser eliminada do conjunto.

No que se refere aos padrões descobertos pela tarefa de classificação também existem diversas formas para pós-processar os padrões extraídos, entre elas: transcrição da árvore de decisão em regras, eliminação de redundância e atribuição de medidas de interesse, por exemplo, a partir de generalizações sucessivas.

* 1. **Tecnologia de Suporte a Mineração de Dados**

O processo de KDD é realizado incorporando-se várias técnicas de diferentes áreas como Aprendizado de Máquina, Data Warehousing, Banco de Dados, Estatísticas, Visualização de Dados, dentre outras. A etapa de Extração de Conhecimento, por exemplo, utiliza muitos recursos da área de Aprendizado de Máquina, e as demais são consideradas como áreas de apoio ao processo de KDD.

**3.6.1 Estatísticas**

A estatística estuda as coleções, análise, interpretação ou explicação, e apresentação dos dados. A mineração de dados tem conexão própria com a estatísticas (Han; Kamber & Pei, 2011).

Segundo Han; Kamber & Pei (2011), um modelo estatístico, é um conjunto de funções matemáticas, que descrevem as ações dos objetos, em uma classe alvo, em termos de suas variáveis aleatórias e sua distribuição de probabilidades associadas. Ainda segundo autores, os modelos estatísticos são muito usados para modelar dados e classes de dados. Por exemplo, pode-se utilizar os modelos estatísticos para caracterização e classificação de dados em tarefas de Mineração de Dados. Em outras palavras, tais modelos estatísticos podem ser o resultado de uma tarefas de Mineração de Dados. Alternativamente, tarefas de Mineração de Dados podem ser construídas em cima dos modelos estatísticos.

As áreas de Mineração de Dados e Estatística estão fortemente ligadas. Ambas as disciplinas têm como objetivo encontrar padrões e regularidade nos dados. Em geral, a Mineração de Dados enfatiza não só a precisão, mas também a facilidade de entendimento do conhecimento adquirido (Rezende, 2005).

Assim, a Mineração de Dados usa diversos mecanismos disponíveis na Estatística para realizar a descoberta de padrões, calcular aproximações, médias, taxas de erros e desvios. Para a Mineração de Dados é necessário o suporte de várias outras técnicas incluindo organização e estrutura de dados (Rezende, 2005).

Vale ressaltar que técnicas estatísticas também são utilizadas para avaliação de hipóteses e do conhecimento obtido no processo de Mineração de Dados. Além disso, no processo de extração de padrões o aprendizado Bayesiano exerce um papel importante. Entre esses algoritmos utilizados em Mineração de Dados estão o *Naive Bayes* e o *AutoClass* (Rezende, 2005).

Existem muitas outras áreas da Estatísticas que tem sido fundamentais para a Mineração de Dados. No entanto, não é o objetivo desse trabalho analisar todas, pois trata-se de um assunto bastante abrangente.

* + 1. **Aprendizado de Máquina**

De acordo com Han; Kamber & Pei (2011), aprendizagem de máquina, investiga como os computadores pode aprender, com base em dados.

Segundo Rezende (2003), aprendizado de máquina, é uma área de Inteligência Artificial (**IA**), cujo objetivo, é o desenvolvimento de técnicas computacionais sobre o aprendizado, bem como a construção de sistemas capazes de adquirir conhecimento de forma automática.

Para Han; Kamber & Pei (2011), a principal área de pesquisa está voltada para programas de computador, aprenderem automaticamente reconhecer padrões complexos, com base em dados, e a partir daí, tomar decisões inteligentes. Por exemplo, um problema típico de aprendizado de máquina, é programar um computador para que ele possa reconhecer automaticamente os códigos postais escritos, depois de aprender a partir com um conjunto de exemplos.

De acordo com Mitchell (1997), aprendizado de máquina, é uma sub-área da Inteligência Artificial **(IA**), cujo objetivo é desenvolver métodos, técnicas e ferramentas para construir máquinas inteligentes, que se modificam para realizar cada vez melhor sua(s) tarefa(s).

Para aprender, o sistema, bem como os seres humanos, podem se valer de estratégias de aprendizado. A seguir, será apresentado, algumas dessas estratégicas clássicas de aprendizagem de máquina relacionado a mineração de dados (*data mining*) (Han; Kamber & Pei, 2011):

* **Aprendizado supervisionado**: é basicamente um sinônimo para classificação. No aprendizado supervisionado o objetivo é induzir conceitos a partir de exemplos que estão pré-classificados, ou seja, exemplos que estão rotulados com uma classe conhecida. Se as classes possuírem valores discretos, o problema é caracterizado como classificação. Caso as classes possuam valores contínuos, o problema é caracterizado, como regressão (Han; kamber & Pei, 2011). Em geral, de acordo com Rezende (2003), cada exemplo é descrito por um vetor de valores de características, ou atributos, e pelo rótulo da classe associada. O objetivo do algoritmo de indução, é construir um classificador que possa determinar corretamente a classe de novos exemplos ainda não rotulados, ou seja, exemplos que não tenham rótulo da classe. Um algoritmo de aprendizado supervisionado pode ser utilizado quando, em um banco de dados, se tem tanto as perguntas com as respostas. Usado para a realização de treinamento de redes neurais na obtenção de classificação, funções de aproximação ou modelagem e previsões baseadas no tempo (Bigus, 1999).
* **Aprendizado não-supervisionado**: é essencialmente, um sinônimo para agrupamento (*cluster*). No aprendizado não-supervisionado, o indutor analisa os exemplos fornecidos e tenta determinar se alguns deles podem ser agrupados de alguma maneira, formando agrupamentos ou *clusters*. A tarefa do algoritmo é agrupara exemplos não rotulados, i.e., exemplos que não possuem o atributo classe especificado. Nesse caso, é possível utilizar algoritmos de aprendizado para descobrir padrões nos dados a partir de alguma caracterização de regularidade, sendo esses padrões denominados *clusters* (Decker & Focardi, 1995; McCallum, Nigan, & Ungar, 200). Exemplos contidos em um mesmo cluster são mais similares, segundo alguma medida de similaridade, do que aqueles contidos em clusters diferentes. O processo de formação dos clusters é geralmente conhecido por *clustering*. Tipicamente, pode-se usar agrupamento para descobrir classes dentro de dados (Han; Kamber & Pei, 2011).
* **Aprendizado semi**-**supervisionado**: quando existem exemplos rotulados, pode-se utilizar o aprendizado supervisionado para induzir classificadores a partir destes exemplos. Caso contrário, quando os exemplos não estão rotulados, pode-se utilizar o aprendizado não-supervisionado com o objetivo de encontrar os *cluster*s. Já o aprendizado semi-supervisionado, consiste em utilizar algoritmos que aprendam apartir de exemplos rotulados e não rotulados. Ou seja, o aprendizado semi-supervisionado, pode ser aplicável tanto em tarefas de classificação quanto em tarefas de agrupamento (Han; kamber & Pei, 2011).
* **Aprendizado ativo**: é uma abordagem de aprendizado de máquina, que permite que o usuário execute um papel ativo no processo de aprendizagem. Uma abordagem de aprendizado ativo, pode pedir ao usuário (i.e., o especialista do domínio) para rotular um exemplo, que pode ser a partir de um conjunto de exemplos não-rotulados, ou sintetizado pelo programa de aprendizagem. O objetivo é otimizar a qualidade do modelo, através de aquisição de conhecimento ativamente do usuário humano, a partir da quantidade exemplos a serem rotulados (Han; Kamber & Pei, 2011).

Pode-se ver que existem muitas similaridades, entre mineração de dados e aprendizado de máquina. Para as tarefas de classificação e agrupamento, as pesquisas em aprendizado de máquina, frequentemente focam, na precisão do modelo. Em adicional a precisão, as pesquisas em mineração de dados, dão muita ênfase na eficiência e escalabilidade de métodos de mineração em grandes massas de dados, bem como na maneira de lidar com tipos de dados complexos e, explorar novos métodos alternativos (Han; Kamber & Pei, 2011).

* 1. **Aplicações de Mineração de Dados**

As tecnologias de Mineração de Dados podem ser aplicadas em grande variedade de contextos de tomada de decisão. Em particular, áreas de significativo retorno de investimento esperado incluem:

* ***Marketing*** – aplicações como análise de comportamento do consumidor baseados em padrões de consumo; a definição de estratégias de *marketing* incluem propaganda, localização de lojas e mala direta direcionada; segmentação de clientes, layouts de lojas e campanhas de publicidades.
* **Finanças** – aplicações incluem análise de crédito de clientes, segmentação de contas a receber, análise de performance de investimentos financeiros como ações, bonds e fundos mútuos e detecção de fraudes.
* **Produção**- aplicações envolvem otimização de recursos como máquinas, força de trabalho e materiais; projeto ótimo de processos de fabricação e projeto de produção de automóveis baseados nos requisitos dos clientes.
* **Saúde** – aplicações incluem descoberta de padrões em imagens radiológicas, análise de dados experimentais em microarray (gene-chip) para relação com doenças, análises de efeitos colaterais de remédios e efetividade de certos tratamentos, etc.
  1. **Visualização de Dados**

As técnicas e ferramentas para Visualização de Dados são indispensáveis ao processo de Mineração de dados. Elas podem ser usadas durante a execução das etapas do processo de extração de conhecimento melhorando a compreensão dos resultados obtidos e a comunicação entre os usuários (Rezende et al., 1998).

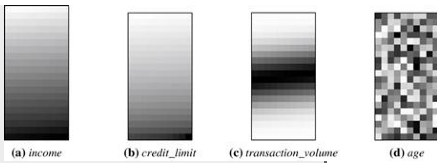
As técnicas de Visualização de Dados estimulam naturalmente a percepção e a inteligência humana, aumentando a capacidade de entendimento e associação de novos padrões. Logo, a Visualização de Dados utiliza a percepção humana como um primeiro método para descobrir valores. Poderosas ferramentas de visualização que consigam gerar diversas formas de visualização (árvores, regras, gráficos 3D/2D, espectro) combinadas com técnicas de Mineração de Dados podem melhorar o processo de Mineração de dados (Fayyad, Grinstein, & Wierse, 2002).

Segundo Han, Kamber & Pei (2011), a visualização de dados, utiliza várias abordagens, incluindo técnicas orientada a pixel, técnicas de projeção geométrica, técnicas baseada em ícone, hierárquica e técnicas baseada em gráfico.

* + 1. **Técnicas de Visualização Orientada a Pixel**

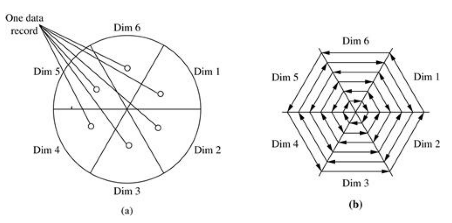
**U**ma maneira simples de visualizar os valores de uma dimensão é usar pixels, onde a cor do pixel reflete os valores da dimensão. Para um conjunto de dados de *m* dimensões, as técnicas ***pixel-oriented***, criam *m* janelas na tela, uma para cada dimensão. Os *m* valores das *m* dimensões de um registro, são mapeados para *m* pixels para uma posição correspondente na janela. As cores dos pixels refletem os valores correspondentes (Han, Kamber & Pei, 2011).

De acordo com Han, Kamber & Pei (2011), dentro de uma janela, os dados são organizados em um ordem globalmente compartilhada por todas as janelas. A ordem global, pode ser obtida, por classificar todos os registros de dados, de uma forma significativa para a tarefa em questão. A **Figura 3.5**, a seguir, mostra um gráfico orientado a pixel, onde, são observados quatro atributos dos clientes: renda (*income*), limite de crédito (*credit\_limit*), volume de transações (*transaction\_volume*) e idade (*age*).



**Figura 3.5**. Adotada de Haa, Kamber & Pei (2011). Visualização de dados de quatro atributos, ordenados pela renda do cliente (ascendente).

Segundo Han, Kamber & Pei (2011), a representação dos registros de dados de forma linear, pode não funcionar bem, quando o número de atributos é muito grande. Isso se deve, segundo Branco (2002), ao fato da dependência direta em relação a resolução da tela, pois quanto maior a dimensionalidade dos dados, maior será o número de janelas e, consequentemente, menor será o número de atributos que poderão ser vistos simultaneamente. Para resolver esse problema, usa-se a técnica Segmentos Circulares (*Circle Segment*s), essa técnica, mapeia os dados em pixels coloridos em segmentos circulares. A **Figura 3.6**, ilustra exemplos dessa técnica (Han, Kamber & Pei, 2011).

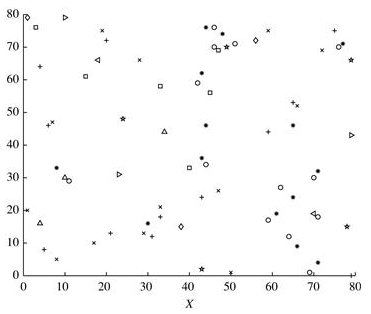


**Figura 3.6**. Adotada de Han, Kamber & Pei (2011). A técnica Segmento Circular. Representando um registro de dados em segmentos circulares.

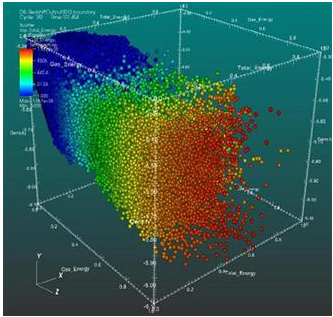
* + 1. **Técnica de Projeção Geométrica**

De acordo com Han, Kamber & Pei (2011), as técnicas de projeções geométricas, projetam os dados multidimensionais em um espaço bidimensional, buscando apresentar projeções interessantes do conjunto de dados. Ainda, segundo Han & kember (2011), em particular, uma técnica bastante utilizada desta categoria, é a técnica gráfico de dispersão (scatter plot), que plota os pontos de dados, usando coordenadas cartesianas. Pode-se adicionar uma terceira dimensão, usando cores ou formas diferentes, para representar pontos de dados. A **Figura 3.7**, mostra um exemplo, onde X e Y são dois atributos espaciais e a terceira dimensão é representada por diferentes formas.

Um gráfico em 3-D, usando gráfico de dispersão, usa três eixos em um sistema de coordenadas cartesianas. No entanto, se ele usar cores, ele pode representar pontos de dados em 4-D, como mostra a **Figura 3.8**.



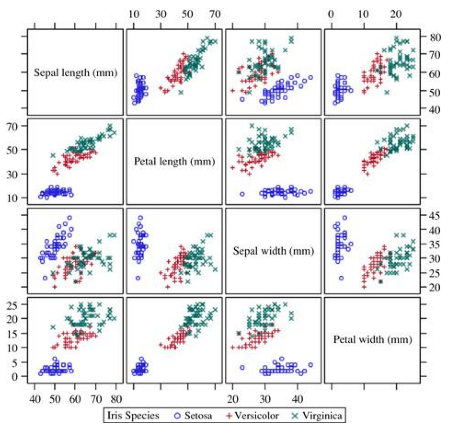
**Figura 3.7**. Adotada de Han, Kamber & Pei (2011). Visualização de um conjunto de dados em 2-D, usando gráfico de dispersão (*scatter plot*).



**Figura 3.8**. Adotada de Han, Kamber & Pei (2011). Visualização de um conjunto de dados em 3-D, usando gráfico de dispersão (*scatter plot*).

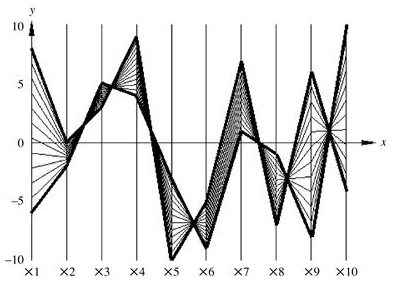
Para conjunto de dados com mais de quatro dimensões, os gráficos de dispersão, são usualmente ineficientes, segundo Han, Kamber & Pei (2011). A técnica matriz de dispersão (*scatter-plot matrix*), é uma generalização para o gráfico de dispersão (*scatter plot*). Para um conjunto de dados n-dimensional, a matriz de dispersão, é uma grade n x n de gráficos de dispersão 2-D, que fornecem uma visualização de cada dimensão com todas as outras dimensões.

A **Figura 3.9**, mostra um exemplo, no qual visualiza o conjunto de dados de íris. O conjunto de dados, consiste de 450 exemplos de cada três espécies de flores de íris. Há cinco dimensões no conjunto de dados: comprimento (*length*) e tamanho (*width*) da pétala, e espécie.



**Figura 3.9**. Adotada de Han, Kamber & Pei (2011). Visualização de um conjunto de dados de íris, usando uma matriz de dispersão (*scatter-plot matrix*).

A técnica de matriz de dispersão, torna-se menos efetiva, à medida que aumenta a dimensionalidade. Uma outra técnica, chamada Coordenadas Paralelas, pode manipular alta dimensionalidade. É uma técnica de visualização, onde as dimensões são representadas como uma série de eixos paralelos, uns aos outros, e com igual espaçamento entre eles, nos quais os valores estão representados (Inselberg; Avidan, 1999). Um registro de dados é representado por uma linha poligonal, que intercepta cada eixo para um correspondente ponto ao valor associado da dimensão (Han, Kamber & Pei, 2011). Veja a **Figura 3.10**.



**Figura 3.10**. Adotada de Han, Kamber & Pei (2011). Visualização que usa Coordenadas Paralelas.

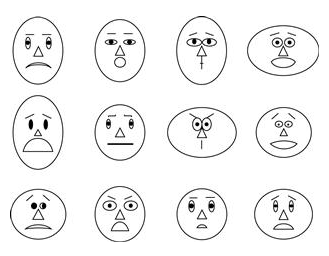
* + 1. **Visualização Iconográfica**

Essa técnica usa pequenos ícones para representar dados multidimensionais. Ou seja, essa técnica, tem como objetivo, mapear os atributos em características particulares de ícones. Cada característica do ícone, representa um atributo dos dados multidimensional (Han, Kamber & Pei, 2011]. Popularmente, essa técnica, trabalha com dois ícones: ***Chernoff faces*** and ***Stick figures***.

* + - 1. **Chernoff faces**

Esta técnica foi introduzido em 1973 pelo estatístico Herman Chernoff. Ele mapeou dados multidimensionais acima de 18 variáveis (ou dimensões) usando características faciais humanas (**Figura 3.10**). Essa técnica ajuda a revelar tendências nos dados. Componentes faciais, tais como os olhos, a boca, as orelhas, e nariz, representam valores atuais das dimensões pelo sua forma, tamanho, posicionamento e localização. Por exemplo, as dimensões podem ser mapeadas para seguir as características: tamanho dos olhos, espaço entre os olhos, comprimento do nariz, curvatura da boca, tamanho da boca, abertura da boca, tamanho da pupila, inclinação da sobrancelha, excentricidade dos olhos e cabeça.

As ***Chernoff*** faces explora a capacidade humana de reconhecer e analisar faces, para perceber pequenas diferenças em características faciais e assimilar muitas características faciais de uma só vez. Desta forma, elas facilitam a visualização de regularidades e irregularidades, presentes nos dados, embora seja limitado para relacionamentos múltiplos (Han & Kamber & Pei, 2011]. Ainda segundo os mesmos autores, uma outra limitação é que, os dados, propriamente dito, não são mostrados, uma vez que aquelas figuras, não transmitem qualquer informação sobre os reais valores ou relacionamentos. Veja a Figura 3.11.

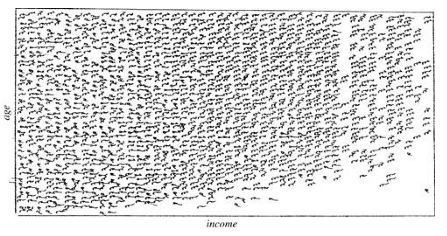


**Figura 3.11**. Adotada de Han & Kamber & Pei (2011). Chernoff faces. Cada face representando um ponto de dado n-dimensional (n 18).

***3.8.4 Stick figures***

A técnica de visualização Stick figures, utiliza duas dimensões da tela para mapear duas dimensões dos dados e as demais dimensões são mapeadas para os ângulos e/ou comprimento dos segmentos de um ícone formado por múltiplos segmentos de retas (Han & Kamber & Pei, 2011). A Figura 3.12, mostra os dados do censo, onde idade e renda são mapeados para os eixos e as demais dimensões (sex, educação, e assim por diante) são mapeadas para *stick figues*.

Segundo Han & kamber & Pei (2011), quando mapeados na tela, os ícones (um para cada item de dado) formam texturas que variam de acordo com as características dos dados, permitindo identificar padrões na imagem que podem indicar dependência funcional entre os atributos visualizados.



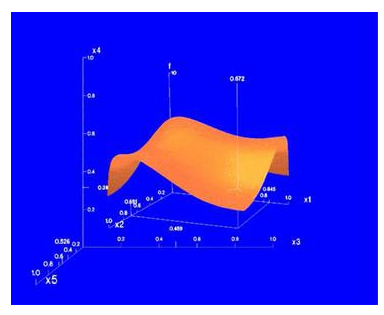
**Figura 3.12**. Adotada de Han & Kamber & Pei (2011). Representação de Censo de dados, usando *Stick Figures*.

* + 1. **Técnica de Visualização Hierárquica**

De acordo com Han, kamber & Pei (2011), as técnicas hierárquicas, geralmente são aplicadas em dados, cuja própria natureza, apresenta uma correlação explicita entre níveis e subconjuntos. Sendo assim, o espaço n-dimensional dos dados (não necessariamente hierárquico) é então dividido em subespaços que são organizados uns dentro do outro de forma hierárquica.

“Worlds-within-Worlds”, também conhecido como n-Vision, é uma representação do método de visualização hierárquico. Suponha que se deseje visualizar um DateSet 6-D, onde as dimensões são **F**, **X1**, ..., **X5**. Deseja-se observar como a dimensão **F** muda em relação as outras dimensões. Se primeiro, fixarmos os valores das dimensões **X3**, **X4**, **X5** para alguns valores selecionados, **C3**, **C4**, **C5**. Então, pode-se visualizar **F**, **X1**, **X2** usando um plot 3-D, chamado um World, como ilustrado na Figura 3.13.

**Figura** 3.13 “Worlds-Within-Worlds” (também conhecindo como n-Vision). Adaptada de Han, Kamber & Pei (2011).



A posição de origem no interior do gráfico é localizada pelo ponto (C1, C2, C3). O usuário pode interagir, alterar os valores desse ponto, e visualizar as mudanças no interior do gráfico.

* 1. **Conclusões**

Neste capítulo foi apresentada a disciplina de Mineração de Dados, que utiliza tecnologias para descobrir conhecimento adicional ou padrões nos dados. Foram discutidas várias técnicas, priorizando os detalhes das regras de associação, a classificação e o agrupamento. Apresentou-se também, alguns algoritmos para algumas dessas áreas.

Como a Mineração de Dados é uma área em constante desenvolvimento, diversas dificuldades e desafios estão surgindo continuamente, as quais precisam ser atacadas. Portanto, nessa sessão foram apresentadas algumas das principais tecnologia que dão suporte ao processo de Mineração de dados, sem a pretensão de esgotá-las.

Parte I

Capítulo

4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Big Data

**4.1 Introdução**

Big Data é um termo que vem chamando a atenção pela acelerada escalada em que volumes cada vez maiores de dados são criados pela sociedade. Fala-se comumente em petabytes de dados gerados a cada dia, e zetabytes começa a ser uma escala real e não mais futurista. A uma década atrás terabyres era uma quantidade futurista, agora temos em nosso próprios computadores. Muito tem sido escrito sobre Big Data e como ele pode servir como base para a inovação, diferenciação e crescimento da análise de dados em grandes massa de dados (Kolb, 2013).

De acordo com Raj (2013), as tecnologias que sustentam o Big Data, podem ser analisadas sob duas óticas: as envolvidas com análise de dados, tendo Hadoop e Map-Reduce como as principais e as tecnologias de infraestrutura, que armazenam e processam os dados. Neste aspecto, destacam-se os banco de dados NoSQL (Not Only SQL).

O termo Big Date está diretamente ligado a questões como volume, variedade, velocidade, complexidade e valor (Mayer-Schönberger, 2013).

* **Volume** – o volume está claro. Geramos petabytes de dados a cada dia. E estima-se que este volume dobre a cada 18 meses.
* **Variedade** - Variedade também, pois estes dados vêm de sistemas estruturados (hoje minoria) e não estruturados (a imensa maioria), gerados por e-mails, mídias sociais (Facebook, Twitter, YouTube e outros), documentos eletrônicos, apresentações estilo Powerpoint, mensagens instantâneas, sensores, etiquetas **RFID** (*Radio-frequency Identification*), câmeras de vídeo, etc.
* **Velocidade** - De acordo com o Gartner, velocidade significa tanto o quão rápido os dados estão sendo produzidos quanto o quão rápido os dados devem ser tratados para atender a demanda. Etiquetas RFID e contadores inteligentes estão impulsionando uma necessidade crescente de lidar com torrentes de dados em tempo quase real. Reagir rápido o suficiente para lidar com a velocidade é um desafio para a maioria das organizações.
* **Valor** - E valor porque é absolutamente necessário que a organização que implementa projetos de Big Data obtenha retorno destes investimentos. Um exemplo poderia ser a área de seguros, onde a análise de fraudes poderia ser imensamente melhorada, minimizando-se os riscos, utilizando-se, por exemplo, de análise de dados que estão fora das bases estruturadas das seguradoras, como os dados que estão circulando diariamente nas mídias sociais.
* **Complexidade** - Quando você lida com grandes volumes de dados, eles vêm de diversas fontes. É um grande desafio vincular, correlacionar, limpar e transformar os dados de um sistema. No entanto, é necessário conectar e correlacionar interações, hierarquias e vínculos múltiplos de informação ou então os dados podem rapidamente sair de controle. Governança de dados pode ajudar a determinar como os dados díspares se relacionam com definições comuns e como integrar sistematicamente os ativos de dados estruturados e não estruturados para produzir informações de alta qualidade, uteis, adequadas e atualizadas.

Em última análise, independentemente dos fatores envolvidos, acreditamos que o termo Big Data é relativo e se aplica (por avaliação do Gartner) sempre que a capacidade da organização de gerenciar, armazenar e analisar os dados exceder sua capacidade atual.

**4.2 O Uso do Big Data**

Os modelos relacionais, quando proposto por Edgar F. Codd, atenderam muito bem, a demanda era acessar dados estruturados, de acordo com (Elmasri & Navathe (2005), gerados pelos sistemas internos das corporações. Estes modelos não foram desenhados para tratar dados não estruturados e nem para volumes de dados na casa dos petabytes de dados.

Para tratar dados na escala de volume, variedade e velocidade do Big Data precisamos de outros modelos. Surgem os softwares de banco de dados NoSQL, desenhados para tratar imensos volumes de dados estruturados e não estruturados. Existem diversos modelos como sistemas colunares como o [*Big Table*](http://static.googleusercontent.com/external_content/untrusted_dlcp/research.google.com/pt-BR/archive/bigtable-osdi06.pdf)(Usado internamente pelo Google),o modelo Key/value como [*DynamoDB da Amazon*](http://aws.amazon.com/pt/dynamodb/), o modelo “*document database*” baseado no conceito proposto pelo Lotus Notes da IBM e aplicado em softwares como MongoDB, e o modelo baseado em grafos como o [Neo4j](http://neo4j.org/), etc (Kolb, 2013).

Aplicações modernas de mineração de dados, frequentemente chamada "Big-Data Analytics", exigir-nos gerenciar grande quantidade de dados rapidamente e em muitas dessas aplicações, exige-se um amplo paralelismo (Kolb, 2013).

Para lidar com aplicações tais como essas, novos tipos de software tem surgido. Estes sistema de programação são projetados para obter o máximo do paralelismo. O novo tipo de software começa com uma nova forma de sistema de arquivos, chamada "Sistema de arquivos distribuídos", que contam com unidades muito maiores do que os blocos de disco dos sistemas operacionais convencionais. Além do mais, os sistemas distribuídos também fornecem replicação de dados ou redundância para proteger os dados, contra falhas frequentes de mídias, que ocorrem quando o dado é distribuído para milhões de nós de computadores (Kolb, 2013).

No topo destes sistemas de arquivos, diversos sistema de alto nível de programação foram desenvolvidos. No centro do novo software está o sistema de programação chamada ***Map-Reduce***. Implementações de ***Map-Reduce*** permite que os cálculos sob os dados em grande escala, sejam executados em clusters de computação de forma eficiente e tolerante a falhas de hardware (kolb, 2013).

**4.3 Map-Reduce**

Map-reduce não é um produto ou um software especifico, mas sim uma tecnologia desenvolvida pelo Google para lidar com grande quantidade de dados, cortando-os e combinando-os no final (Kolb, 2013).

A ideia básica é que os dados que precisam ser processados, entram no sistema, e é cortado em pedaços chamados de chunks. Essas peças de software que é responsável em fazer esses cortes é chamado de “***Mapper***”. Os chunks são então enviados para outra peças de software para fazer o processamento requerido sobre eles, e então eles são ainda enviados para outra peça de software chamado “***Reducers***” que combina o resultado final para a saída. Isto tudo está ilustrado na Figura 4.1 (Kolb, 2013).

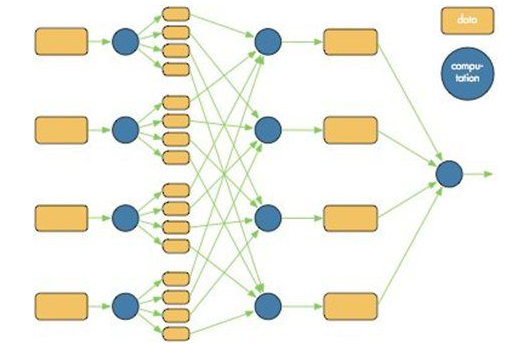
Input (chuncks)

Map (tasks)

Group by keys

Reduce (tasks)

CombinedOutput



**Figura 4.1 Um esquema da função Map-Reduce**. **Fonte**: The Big Data Revolution, Kolb, 2013.

Todas essas peças de software - mapeador, processador, e redutor – tipicamente rodam no mesmo servidor de uma só vez. Dessa forma, a carga de processamento dos dados mapeados podem ser espalhados por todo servir disponível, e mais servidores podem ser adicionado em tempo real, se for necessário um resultado mais rápido (Kolb, 2013).

O importante aqui, é você entender que, a tecnologia de Map-Reduce é capaz de pegar uma grande quantidade de dados, que seria muito dispendioso rodar em apenas um servidor, e poder distribui-lo por vários servidores. Este é um novo paradigma de programação. Existem algumas ferramentas que implementam esse novo paradigma, entre elas cito, o **Hadoop** do *Apache Foundation* (Kolb, 2013).

**4.4 Sistemas Distribuídos**

A maioria dos computadores são feitos com um único processador, com sua memória principal, cache, e disco local (um nó de computação). No passado, grandes cálculos científicos, eram realizadas em aplicações chamada de processamento paralelo, feitos com propósito especial em computadores de uso paralelo com muitos processadores e hardware especializado. No entanto, a prevalência de serviços Web de grande escala tem levado a computação a ser realizada em instalações com milhares de nós de computação operando mais ou menos de forma independente. Nestas instalações, os nós de computação são hardware de commodity, que reduz os custos comparado com as máquinas paralelas (Stonebraker et Al., 2010).

Esta nova arquitetura de computação tem dado origem a uma nova geração de sistemas de programação. Estes sistemas tiram máximo proveito do paralelismo e são tolerantes a falhas, quer seja de software ou de hardware (Stonebraker et al., 201o).

**4.4.1 Organização Física de Nós de Computação**

A nova arquitetura de computação paralela, algumas vezes chamada de computação em cluster, é organizada como segue. Os nós de computação são armazenados em racks, talvez de 8-64 nós em um rack. Os nós em um simples rack são conectados por uma rede, tipicamente uma Ethernet gigabits. Pode haver muitos racks de nós de computador, e racks são interconectados outros níveis de rede ou a um switch (Dean & Ghemawat, 2008).

**4.4.2 Organização de Sistemas de Arquivos em Grande Escala**

Para explorar computação em cluster, o arquivo deve parecer e se comportar um pouco diferente de sistemas de arquivos convencionais encontrados em computadores simples. Este novo sistema de arquivos, geralmente chamado de *distributed file system* ou **DFS** (embora este termos tenha tido outro significado no passado), é tipicamente usado como segue (Dean & Ghemawat, 2008):

* Os arquivos são divididos em chunks, que são tipicamente de 64 megabytes de tamanho. Esses chunks são replicados, talvez três vezes, para três diferentes nós de computação. Além disso, os nós que armazenam um chunk deve estar localizado em racks diferentes, dessa forma não se perde todas as cópias devido uma falha de rack. Normalmente, o tamanho do chunk e grau de replicação podem ser decidido pelo usuário.
* Para encontrar os chunks de um arquivo, existe um outro pequeno arquivo chamado nó **master** ou nó de nome para o arquivo. O nó master é ele mesmo replicado, e o diretório do sistema de arquivos como um todo sabe onde encontrar suas cópias. O diretório pode ser replicado e todos os participantes usando o DFS sabem onde estão suas cópias no diretório.

**4.5 As Tarefas de Mapeamento**

De acordo com Kolb (2013), o arquivo de entrada para uma tarefa Mapeamento, consiste de elementos, que podem ser de qualquer tipo: uma tupla ou um documento, por exemplo. Um chunk é uma coleção de elementos, e nenhum elemento é armazenado em dois chunks. Tecnicamente, todas as entradas para as tarefas de mapeamento (The Map Tasks) e saídas para as tarefas Redução (The Reduce Tasks) são os pares na forma chave-valor (key-value), geradas por uma função hash. Essa forma de entradas e saídas são motivadas pelo desejo de permitir a composição de vários processos Map-Reduce (Kolb, 2013).

A função de mapeamento (Map) recebe um elemento com seus argumentos e produz zero ou mais pares chave-valor. Os tipos de chaves e valores são arbitrários. Mais, as chaves não são "chaves" no sentido usual; elas não precisam ser únicas. Mais uma tarefa de mapeamento pode produzir vários pares chave-valor com a mesma chave, mesmo a partir do mesmo elemento (Kolb, 2013).

**Exemplo 4.1**: Suponha que deseja-se contar o número de ocorrências para cada palavra em uma coleção de documentos. Neste exemplo, o arquivo de entrada é um repositório de documentos, e cada documento é um elemento. A função de mapeamento para este exemplo usa chaves que são do tipo String (a palavra) e valores que são inteiros. A tarefa de mapeamento lê um documento e quebra ele em uma sequência de palavras w1, w2, w3, ..., wn. Ela então emite um sequência de pares de chave-valor onde o valor é sempre 1. Isto é, a saída da tarefa de mapeamento para este documento é a sequência de pares chave-valor:

(w1,1),(w2,1),....,(wn,1)

Note que uma simples tarefa de mapeamento irá processar muitos documentos - todos os documentos em um ou mais chunks. Assim, a saída produzida será mais do que a sequência para o documento sugerida acima. Note também que se uma palavra w aparece m vezes entre todos os documentos atribuídos a esse processo, então haverá m pares chave-valor (w,1) entre sua saída. Uma opção para resolver esse problema é usar agrupamento e agregação, que é combinar esses m pares em um simples par (w, m), isso só é possível porque as tarefas Redução, aplica uma operação associativa e comutativa, para os valores.

**4.5.1 Agrupamento e Agregação**

O processo controlador mestre sabe quantas tarefas Reduce haverá, digamos r tarefas, pois o usuário normalmente informa ao sistema map-reduce quais são as r tarefas. Então o controlador mestre aplica uma função hash e produz uma tabela de chaves de números (códigos) de 0 até r-1. Cada chave produzida pela tarefa Map é um hash e seus pares chave-valor são colocados em um arquivo local. Cada arquivo é destinado para um das tarefas Reduce (Kolb, 2013).

Após todas as tarefas Map terem completadas com sucesso, o controlador mestre junta os arquivos de cada tarefa Map que são destinados para uma particular tarefa e alimenta o arquivo resultante com uma lista de pares chave-valor. Isto é, para chave k, a entrada para a tarefa Reduce que manipula a chave k é um par da forma (k, [v1, v2, ..., vn]), onde (k, v1), (k, v2), ..., (k, vn) são todos pares chave-valor e k, vindo de todas as tarefas Map.

**4.6 As Tarefas de Redução**

Os argumentos da função Reduce é um par consistindo de uma chave e sua lista de valores associados. A saída da função Reduce é uma sequência de zero ou mais pares chave-valor. Esses pares chave-valor pode ser de tipo diferente daqueles enviados das tarefas map para as tarefas Reduce, mais normalmente elas são do mesmo tipo. Referimo-nos a aplicação da função Reduce que reduz para uma simples chave e seus valores associados de redutor (Kolb, 2013).

Uma tarefa reduce recebe uma ou mais chaves e sua lista de valores associados. Isto é, uma tarefa reduce executa um ou mais redutores. As saídas de todas as tarefas reduce são juntas em um simples arquivo. Redutores podem ser divididos em tarefas reduce menores e a função hash associa cada chave com um dos códigos da tabela hash (Kolb, 2013).

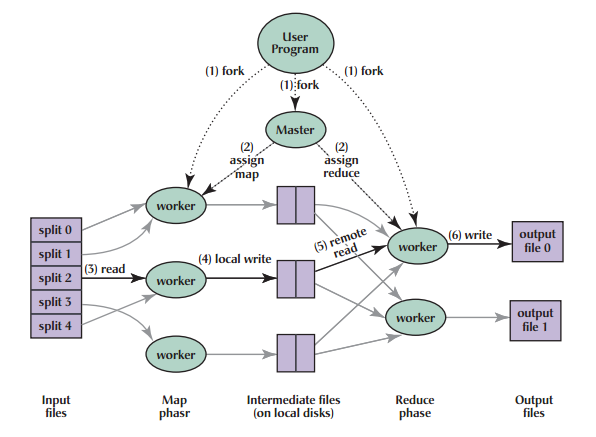
**Exemplo 4.2**: Vamos continuar com o exemplo conta palavras do Exemplo 4.1. A função Reduce simplesmente agrega todos os valores. A saída de um redutor consiste da palavra e da soma. Isto é, a saída de todos as tarefas Reduce é uma sequência de pares (w,m), onde *w* é uma palavra que aparece pelo menos uma vez entre todos os documentos e *m* é o total de ocorrências de *w* em todos os documentos.

**4.7 Detalhes de Execução de Map-Reduce**

A Figura 4.2 oferece um esboço de como processo, tarefas, e arquivos interagem. Aproveitando uma biblioteca fornecida por um sistema map-reduce tal como Hadoop, o programa do usuário bifurca o processo controlador mestre e alguns dos processos Worker para diferentes nós de computação. Normalmente, um Worker manipula suas tarefas Map (um Map worker) ou tarefas Reduce (um Reduce worker), mas não ambos (Kolb, 2013).

O mestre tem muitas responsabilidades. Uma é criar um certo número de tarefas Map e algumas tarefas Reduce, este número sendo selecionado pelo programa do usuário. Estas tarefas serão atribuídas para o processo Worker pelo Mestre. É razoável criar uma tarefa Map para cada chunk de arquivo de entrada, mas pode-se desejar criar poucas tarefas Reduce. A razão para limitar o número de tarefas Reduce é que é necessário para cada tarefa Map criar um arquivo intermediário para cada tarefa Reduce, e se existe muitas tarefas Reduce o número de arquivos intermediários aumenta bastante (Kolb, 2013).

O Mestre (Master) se matem informado do estado de cada tarefa Map e Reduce (ocioso, executando um particular worker, ou concluído). Um processo Worker relata para o Mestre quando ele termina uma tarefa, e uma nova tarefa é agendada pelo Mestre para esse processo Worker (Kolb, 2013).



**Figura 4.2** **Esboço de Iteração de um Processo Map-Reduce. Fonte:** Dean & Ghemawat (2008).

**4.8 Lindando com Falha de Nós**

A pior coisa que pode acontecer é quando o nó de computação que está executando o Mestre falha. Neste caso, todo carga de entrada do map-reduce deve ser reiniciado. Mas somente este nó pode derrubar um processo inteiro; outras falhas iram ser gerenciadas pelo Mestre, e o trabalho do map-reduce irá completar eventualmente (Kolb, 2013).

Suponha que o nó de computação no qual reside o Map worker falha. Esta falha irá ser detectada pelo Mestre, porque ele periodicamente pings o processo Worker. Todas as tarefas Map que foram atribuídas para este Worker terão que ser refeitas, mesmo que tivessem concluído. A razão para refazer completamente as tarefas Map é que sua saída destinada as tarefas Reduce residem no nó de computação, e devido falha, está indisponível para as tarefas Reduce. O Mestre configura o estado de cada uma das tarefas Map para ociosa e reprograma-o para um Worker quando se tornar disponível. O Mestre também informa cada tarefa Reduce que a localização de suas entrada, que são as tarefas Map, mudaram (Kolb, 2013).

Lidar com uma falha para um Reduce Worker é simples. O Mestre, simplesmente, configura os estados de suas correntes tarefas Reduce em execução, para ociosa. Estas serão agendadas para outro reduce worker mais tarde (Kolb, 2013).

4.9 **Conclusões**

Concluindo, Map-Reduce é um modelo de programação, e framework introduzido pelo Google para suportar computações paralelas em grandes coleções de dados em clusters de computadores. Agora Map-Reduce é considerado um novo modelo computacional distribuído, inspirado pelas funções map e reduce usadas comumente em programação funcional. Map-Reduce é um “Data-Oriented” que processa dados em duas fases primárias: Map e Reduce. A filosofia por trás do Map-Reduce é: Diferentemente de data-stores centrais, como um banco de dados, você não pode assumir que todos os dados residem em um lugar central portanto você não pode executar uma query e esperar obter os resultados em uma operação síncrona. Em vez disso, você precisa executar a query em cada fonte de dados simultaneamente. O processo de mapear a requisição do originador para o data source é chamado de ‘Map’, e o processo de agregação do resultado em um resultado consolidado é chamado de ‘Reduce’.

Hoje existem diversas implementações de Map-Reduce, como: Hadoop, Disco, Skynet, FileMap e Greenplum. Hadoop é a implementação mais famosa.

A tecnologia de big data não apenas suporta a habilidade de coletar grandes volumes de dados como também provê a habilidade de compreendê-los e tirar proveito de seu valor.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Capítulo

5

Case-Based Reasonable

**5.1 Introdução**

Este capítulo apresenta uma introdução dos **Sistemas de Raciocínio Baseados em Casos** (**CBR** – *Case-based Reasoning*). No entanto, segundo (Watson, 2003), os **CBR**s e a Gestão do Conhecimento estão relacionados. Portanto, antes de discutir os sistema de Raciocínio Baseados em Caso, discute-se o que é Gestão do Conhecimento.

**5.2 Gestão do Conhecimento**

A função da Gestão do Conhecimento, segundo (Watson, 2003), é permitir que as organizações possam alavancar suas informações e conhecimento, através de experiências. Conhecimento, e consequentemente, sua gestão, está sendo alardeado como base da futura competitividade econômica, por exemplo.

De acordo com Watson (2003), o conhecimento, agora, passa a ser vista como um ativo, a criação e o compartilhamento tem se tornado um importante fator dentro e entre as organizações. Embora, muitos autores levantam a questão em relação ao “paradoxo valor” quando considera a natureza do conhecimento, em particular sua intangibilidade e inadequação como um ativo e a dificuldade de avaliar e proteger seu valor.

**5.2.1 Definição de Gestão do Conhecimento**

Muitos autores abordam o assunto de diferentes perspectivas. Eles, portanto, têm diferentes definições. A maioria da literatura sobre gestão do conhecimento, tratam o conhecimento de forma ampla, e usa-o para cobrir tudo o que a empresa necessita para realizar suas funções. Isto pode envolver o conhecimento formalizado, patentes, leis, programas, e procedimentos, bem como o mais intangível know-how, habilidades, e experiências das pessoas. Ele também inclui a maneira como as organizações funcionam, comunica-se, analisa situações, desenvolve novas soluções para os problemas, e desenvolve novas formas de fazer negócio. Mais ainda, pode envolver questões culturais, costumes, e valores, bem como os relacionamentos entre fornecedores e clientes (Watson, 2003).

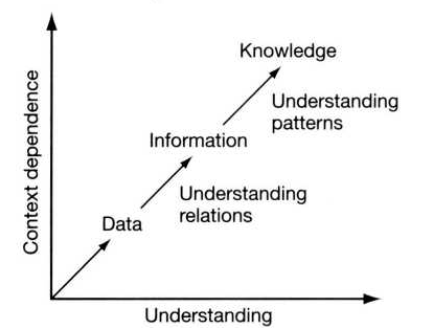
Por outro lado, Gestão inclui todas as maneiras que o conhecimento ativo de uma organização é colhido, armazenado, transmitido, aplicado, atualizado, ou gerado (Watson, 2003). Neste trabalho, centra-se na gestão do conhecimento, por meio da aplicação de uma metodologia para implementar a solução da gestão do conhecimento, nomeadamente, *case-based reasoning* (**CBR**).

Portanto, a definição adotada é a apresentada por Watson (2003), que é “Gestão do Conhecimento envolve a aquisição, armazenamento, recuperação, aplicação, geração, e revisão do conhecimento ativo de uma organização de uma maneira controlada”.

**5.2.2 O que é Conhecimento?**

De acordo com Watson (2003), o conhecimento não existe isolado. Não é algo que pode ser pego e trancado em um cofre de uma empresa. Na verdade, alguns filósofos acreditam que o conhecimento é uma construção humana que não existe fora da mente das pessoas. Vale a pena considerar o relacionamento entre dado e informação. Os computadores têm por décadas sempre manipulado dados (em sistemas de banco de dados.

Dados, informações e conhecimento pode ser considerado, não como entidades discretas, mas como algo continuo, como ilustrado na Figura 5.1. Eles exibem uma relação com seu contexto e quantidade de entendimento quer é requerido ou impactar (Watson, 2003).



**Figura 5.1 A relação de contexto para a compreensão. Fonte:** Appling Knowledge Management: Techniques for Building Corporate Memories (Watson, 2003).

Uma noção importante a se observar na Figura 5.1 é que o conhecimento envolve o reconhecimento ou a compreensão de padrões. Isto envolve a criação de modelos mentais, exemplares, ou arquétipos (Watson, 2003).

De acordo com Watson (2003), quando existe padrão entre informação, o padrão tem o potencial de representado como conhecimento. Embora, os padrões representando conhecimento, devem ter um contexto. O contexto do padrão, proporciona um grau de previsibilidade de onde o padrão pode ser aplicado.

Existe o conhecimento explícito, que é aquele que você sabe expressá-lo ou codifica-lo. Mas, nem todo conhecimento é explícito; alguns são tácitos. Ele pode ser sentido e compreendido, mas não expresso. Mas, os sistemas de gestão do conhecimento devem lidar com ambos conhecimento explícito e tácito. Para muitos da comunidade da gestão do conhecimento, é errado tentar codificar todo conhecimento (isto é, torna-lo explícito), e a tentativa de fazê-lo, resultará em muito conhecimento tácito sendo perdido (Watson, 1997, 2003).

Dessa forma, a representação do conhecimento pelos sistemas de gestão de conhecimento deve ser flexível. O formalismo rígido de sistema especialistas baseados em regras (*rule-based expert systems*) dos anos 80 são restritivos para manipular conhecimento tácito. A representação mais discursiva de uma biblioteca de casos, tais como as usadas pelos sistema de raciocínio baseados em casos, pode ser mais capaz de lidar com o conhecimento tácito; mas você deve reconhecer que não existe formalismo que possa capturar adequadamente todo conhecimento tácito (Watson,1997, 2003).

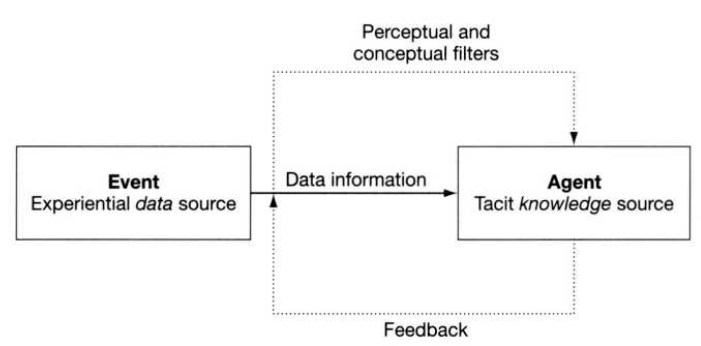
**5.2.3 Diferença entre Conhecimento, Informação e Dado**

Na literatura nem sempre fica claro o que é “gestão do conhecimento” e “Gestão da informação”. Informação e conhecimento são mais apropriadamente vistos em termos de uma dinâmica e interativa relação. A informação facilita o desenvolvimento do conhecimento, que cria mais informações que aprofundam o conhecimento, *ad infinitum*. A natureza dinâmica desta relação é ilustrada na Figura 5.2.

Olhando para a informação puramente em termos de como tenha sido processada, ou seja, olhando os dados, informação e conhecimento de forma continua, não se percebe a complexidade que existe entre os três valores. Note que o elemento feedback dentro da Figura 5.2, que ilustra a dinâmica e o relacionamento interativo entre informação e conhecimento, é um laço (Watson, 2003).

Dado é a discriminação de estado-por exemplo, preto, branco, pesado, leve-que pode ou não transmitir informação para uma pessoa, dependendo da prioridade da pessoa ou do contexto.

**Figura 5.2** **Dado, Informação e Conhecimento**. **Fonte**: Appling Knowledge Management: Techniques for Building Corporate Memories (Watson, 2003).



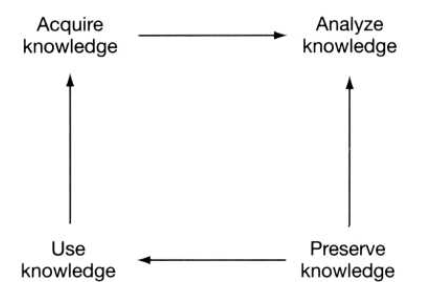
Pode-se caracterizar dado com uma propriedade de coisas e conhecimento com uma propriedade de pessoas, que o predispõe a agir em circunstancias particulares. Informação é um subconjunto de dados que residem em ciosas que levam uma pessoa a agir; que filtrada a partir dos dados por aparato perceptual ou conceitual da pessoa (Watson, 2003).

**5.2.4 Atividades da Gestão do Conhecimento**

De acordo com Watson (2003), o ato de gerenciar conhecimento pode ser caracterizado por quatro atividades:

1. Adquirir conhecimento (aprendendo, criando ou identificando);
2. Analisando conhecimento (avaliar, validar ou valorar);
3. Preservar conhecimento (organizar, representar ou manter); e
4. Usar conhecimento (aplicar, transferir ou compartilhar).

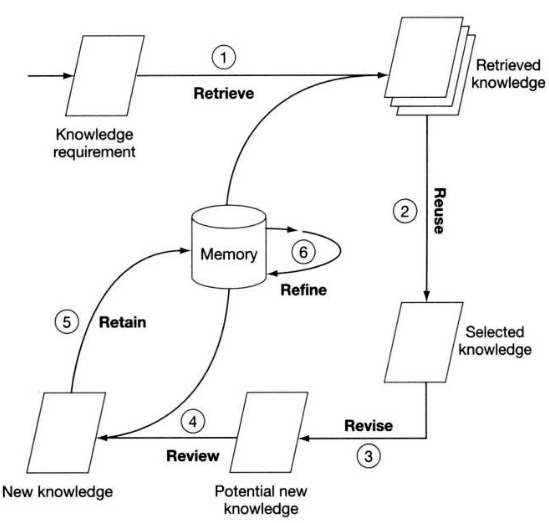
Estas atividades não existem em isolado. Em vez disso, pode-se considera-las como um ciclo, como mostra a Figura 5.3. Você pode ver esta gestão do conhecimento ciclo (***KM-cycle***) com uma simplificação do mais detalhado Case-based reasoning-cycly (CBR-cycle) que será discutido em sessões seguintes (Watson, 2003). O elemento que liga o ciclo é o uso do conhecimento, uma vez que é provável que quando o conhecimento é usado, uma nova visão sobre o conhecimento pode ser criado. Este novo conhecimento, por sua vez, deve ser adquirido, analisado e preservado para uso futuro (Watson, 2003).



**Figura 5.3 O KM-cycle**. **Fonte**: Appling Knowledge Management: Techniques for Building Corporate Memories (Watson, 2003).

* + 1. **Uma Metodologia para o Conhecimento**

Em recente workshop realizado na Universidade de Cambridge, na Inglaterra, um grupo de pessoas ativas gestão do conhecimento e Inteligência Artificial (IA) identificou as principais atividades necessárias por um conhecimento (Watson, 2003). Essa atividades estão ilustradas na Figura 5.4.



**Figura 5.4 O funcionamento do Raciocínio Baseado em Casos** (*CBR-cycle*). **Fonte**: Appling Knowledge Management: Techniques for Building Corporate Memories (Watson, 2003).

1. O processo de recuperação, reuso, e revisão suporta a aquisição de conhecimento.
2. O processo de revisão e refinamento suporta a análise do conhecimento.
3. A proporia memória (juntamente com a recuperação e o refinamento) suportam a preservação do conhecimento.
4. Finalmente, a recuperação, reuso, e revisão suportam o uso do conhecimento.

Em sessões seguintes este processo será mais detalhado e ilustrado com estudo de casos.

Os pontos chaves aqui discutidos foram (Watson, 2003):

* O conhecimento não é estático; isto é, um sistema de gestão do conhecimento deve poder suportar a aquisição, análise, preservação, e reuso do conhecimento como um processo continuo e cíclico.
* O conhecimento existe em duas formas: conhecimento explícito, que pode ser codificado e o conhecimento tácito, que nem sempre pode ser codificado. Se a representação do conhecimento for muito formalizado, muito conhecimento tácito pode ser perdido. Assim, a representação do conhecimento em sistemas de gestão do conhecimento, deve ser flexível e discursiva.
  1. **Raciocínio Baseado em Casos (***Case-Based Reasoning* **– CBR)**

Na sessão anterior foi introduzido o ***CBR-cycle*** e como ele satisfaz os requisitos de um sistema de gestão de conhecimento. Nesta sessão será detalhado cada processo do ***CBR-cycle***.

De acordo com Watson (2003), o **CBR** usa o conceito de similaridade para recuperar coisas (casos) de uma biblioteca (uma base de casos). Casos são usados em muitas situações; por exemplo, para fornecer informações de produtos para um cliente, resolver problemas em uma central de informações ao clientes, configurar equipamentos de manufatura, ou resolver problemas financeiros complexos.

De acordo com Watson (2003), nós resolvemos problemas usando experiências adquiridas e que podemos aprender novas experiências. O Raciocínio baseado em Casos pode ser descrito por seis atividades ocorrendo em ciclo, como discutido na sessão anterior.

Este ciclo é constituído de seis processos:

1. Recuperar
2. Reusar
3. Revisar
4. Avaliar
5. Manter
6. Refinar

**5.3.1 Definição**

Para Watson (2003), a ideia básica em um sistema CBR é que, para um domínio particular, os problemas a serem resolvidos tendem a ser recorrentes e repetir-se com pequenas alterações em relação a sua versão original. Dessa forma, soluções anteriores podem ser reutilizadas também com pequenas alterações.

Riesbeck e Schank (1996), definem **CBR** como “Um sistema de CBR resolve problemas por adaptar soluções que foram utilizadas para resolver problemas anteriores”.

Em seguida será detalhado cada um desses processos, mas primeiro é necessário entender o que é recuperar, reusar, e revisar, e assim, casos.

**5.3.2 Representação de Casos**

De acordo com Watson (2003), casos são registros de experiências que contém conhecimento, que pode ser ambos explicito e tácito. Por exemplo, ele pode ser casos de históricos de pacientes no sentido médico, detalhes de empréstimos bancários, ou descrição de situações de erros de equipamentos. Cada um desses registros de casos compreende:

* Uma descrição
* O respectivo resultado ou solução

Assim, um caso tipicamente compreende um par problema e solução. Uma coleção de casos é chamado de uma base de casos, justamente como uma base de registros é chamado de banco de dados (Watson, 2003).

Uma forma de visualizar é em termos de espeço do problema e espaço de solução. Na Figura 5.5 vê-se que um caso individual é composto de dois componentes: uma descrição do problema e o armazenamento da solução. Estes residem respectivamente no espaço do problema e no espeço de solução. A descrição do problema a ser resolvido é colocado no espaço de problema. Recupera-se o caso mais similar a descrição do problema, e sua solução, armazenada encontrada. Se necessário, ocorrem adaptações, e uma nova solução é armazenada. Este modelo conceitual de CBR, assume que há um mapeamento direto de um-para-um entre o problema e os espaços de solução. (Watson, 1999).

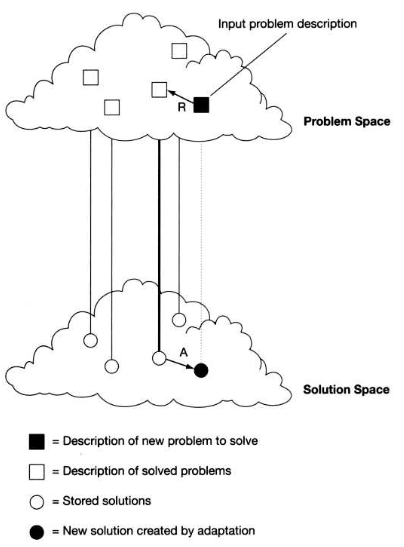
Bases de Casos divide-se em duas grandes categorias (Watson, 2003):

* **Em bases de Casos homogêneas**: todos os casos compartilham os mesmos dados ou estrutura de registros; isto é, casos têm os mesmos atributos mas variando os valores.
* **Em Bases de Casos Heterogêneas**: casos têm estrutura de registros variados; isto é, casos podem ter diferentes atributos e valores variados.

Um exemplo de caso homogêneo pode ser o caso de venda de casa, onde na base de casos de casas tem os mesmos atributos que são suficientes para descrever uma casa. Então, um corretor, que tenha acesso a essa base de casos, pode assumir que já tenha todas as informações para realizar a transação. Embora, se o corretor ainda não tiver esta propriedade na base de casos, ele pode facilmente criar um registro nessa base de casos.

Um exemplo de base de casos heterogênea poderia ser uma base de casos de diagnósticos de pacientes. Registros de pacientes contêm um lote de informações em comum, tais como idade, tipo sanguíneo, pressão sanguínea, mas também muitas informações que são únicas para cada paciente, por exemplo, histórico médico, tratamento, e prognósticos.

Quando desenvolve-se uma base de casos heterogênea, os desenvolvedores nunca pode assegurar que ele tenha um conjunto completo de características (Watson, 2003). Por exemplo, uma base de dados de diagnósticos de pacientes, os desenvolvedores poderia não listar todas as possíveis condições médicas, sintomas, e testes que uma pessoa poderia ter.



**Figura 5.5 Os espaços de problema e de solução**. **Fonte**: Appling Knowledge Management: Techniques for Building Corporate Memories (Watson, 2003).

Dentro de um **Caso** pode-se armazenar muitos tipos de dados, tais como nomes, identificação de produto, valores como custo, temperatura, e notas textuais. Algumas ferramentas de CBR também suportam dados com características de multimídia, tais com imagens, sons e vídeo (Watson, 1999, 2003).

Não há um consenso por parte da comunidade de CBR, que informações exatamente, poderia ser um Caso. Embora, duas medidas pragmáticas poderia ser tomadas para se decidir o que poderia ser representada em Casos: a funcionalidade da informação e a facilidade de aquisição da informação (Watson, 1999).

**5.3.3 Indexação**

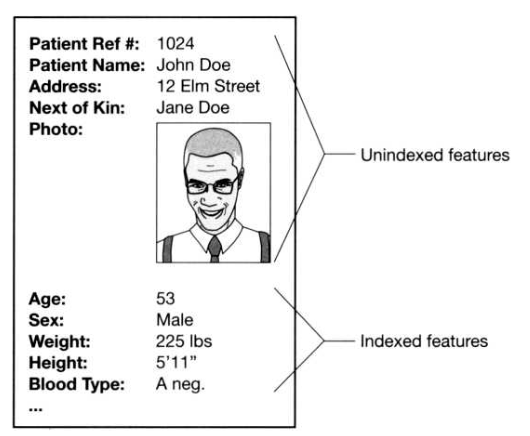
Muitos sistemas de Banco de Dados utilizam-se de índices para agilizar a recuperação de dados. Um índice é computacionalmente, uma estrutura de dados que pode ser realizada em memória, tornando a localização da informação, muito rápida, sem ter que fazer a busca do(s) registro(s) no disco. O CBR também faz uso de índice para agilizar a recuperação de Casos. A informação dentro de um Caso, é de dois tipos (Watson, 1999):

1. Informação Indexadas, que á usada para recuperar um Caso.
2. Informação não indexada, que fornecem informações contextuais para o usuário, mas que não são usadas diretamente para recuperação de Caso.

Por exemplo, em um sistema médico, pode-se usar as informações do paciente, tais como idade, sexo, altura, e peso como características a serem indexadas, que pode ser usada para recuperação do Caso e, outras informações, tais como nome, endereço e fotografia como informações contextuais, ou seja, não indexadas, que não podem ser usadas para recuperação de Casos. A Figura 5.6 ilustra este exemplo.

Como diretrizes, os índices devem:

* Ser preditivo.
* Indicar o propósito em que o Caso será usado.
* Ser abstrato o suficiente para permitir ampliar a base de casos e seu uso no futuro.
* Ser concreto o suficiente para ser reconhecido em futuras situações.



**Figura 5.6 Informações indexadas e não-indexadas**. **Fonte**: Appling Knowledge Management: Techniques for Building Corporate Memories (Watson, 2003).

Se tomarmos como exemplo um sistema bancário. Informações dos clientes, tais como nome e telefone, são claramente não preditivas, você não decidir emprestar dinheiro a um cliente como base em seu nome e telefone. No entanto, informações tais como renda, e seus compromissos financeiros, tais como empréstimos habitacionais, pagamentos de carros, e seguro de vida, e assim por diante, são claramente preditivos. Dessa forma, informações como renda e compromissos financeiros podem ser escolhidos como índices e nome e telefone como informações contextuais (Watson, 1999).

A escolha do índice, tanto pode ser manual com automatizada. Escolher um índice manualmente, envolve decidir o propósito do Caso em respeito ao objetivo do sistema e decidir em que circunstancias o Caso vai ser útil (Watson, 1999).

Existem um número crescente de métodos de indexação automática na literatura, incluindo: MEDIATOR, CHEF e CYRUS, etc.

Várias ferramentas de CBR presentes no mercado suportam a identificação de índices de casos automaticamente, para aplicações práticas, índices pode ser escolhido automaticamente, manualmente ou ambas técnicas (Watson, 1999, 2003).

**5.3.4 Aquisição (Storage)**

A representação do Caso é um importante aspecto no projeto de sistemas CBR, no que se refere a visão conceitual do que é representado no Caso e levando-se em conta os índices que caracterizam os casos. A base de Casos poderia ser organizada em uma estrutura gerenciável que suporte pesquisas e métodos de recuperação eficientes. Deve-se ser encontrado um equilíbrio entre os métodos de armazenamento que preserve a riqueza de casos e seus índices e métodos que simplifique o acesso e recuperação de casos relevantes (Watson, 1999). Este métodos são usualmente chamados de modelos de memória de casos (*case-memory models*). Os dois modelos de memória de casos mais influentes na academia são o modelo de memória dinâmica de Schank e Kolodner, e o modelo categoria exemplares de Porter e Bareiss. Estas técnicas ainda são bastantes utilizadas pela comunidade de Ciência Cognitiva, mas nenhuma ferramenta comercial de CBR usam essas técnicas. Ao invés disso, muitas base de casos utilizam estruturas simples de arquivos planos (flat files), ou estruturas de bancos de dados relacionais e, usa índices para se referir aos casos (Watson, 1999).

**5.3.4.1 Modelos de Memória Dinâmica**

O modelo de memória dinâmica é composto principalmente de pacotes de organização de memória (**MOP**s), que são frames que compõem uma unidade básica da memória dinâmica. Eles representam conhecimento sobre classes de eventos de duas maneiras:

* Instâncias, que representam casos, eventos ou objetos.
* Abstrações, que representam versões generalizadas de instâncias ou de outras abstrações.

O desenvolvimento da teoria mais geral de Schank levou aos pacotes de organização de memória episódicos (**E-MOP**s), implementados no sistema **CYRUS** (Kolodner, 1993). A ideia básica é organizar casos específicos que partilham propriedades similares sob uma estrutura mais geral, ou seja, um **E-MOP**. Uma **E-MOP** contém os casos, as propriedades comuns entre eles e as características que os diferenciam.

**5.3.4.2 Modelos de Categoria Exemplares**

Esse modelo considera que os casos no mundo real podem ser vistos como exemplares de acontecimentos. Neste caso, uma memória de casos é uma rede semântica de categorias de casos ligados por relações semânticas de hierarquias, de semelhança ou de diferenças. Cada caso é associado a uma categoria e suas características têm importância distintas para enquadrá-lo ou não na categoria. Características similares de um caso apontam para as de outro caso ou categoria. Dessa forma, compõe-se uma rede de conhecimento genérico do domínio que permite alguma recuperação do raciocínio do sistema para gerar explicações. Para gerar um novo caso, é buscado um caso semelhante no banco de casos. Se houver pequenas diferenças entre os dois, apenas um deles é retido, ou é armazenado uma única combinação dos dois (Watson, 2003).

**5.3.5 Recuperação**

Dada uma descrição de um problema, um algoritmo de recuperação deveria encontrar os casos mais similares à situação atual, utilizando-se dos índices da memória de casos. Os algoritmo baseiam-se nos índices e na organização de memória para guiar a busca dos casos potencialmente úteis.

De acordo com Watson (2003), a recuperação de casos está diretamente relacionado e dependente ao método de indexação usado. Em geral, duas técnicas são correntemente usadas pelas ferramentas de **CBR** comerciais: algoritmo de vizinhança (***Nearest-Neighbor***) e Indutivo.

**5.3.5.1 Algoritmo de Vizinhança**

Esse método, segundo Watson (2003), baseia-se na comparação entre um novo caso e aqueles armazenados na base de casos, utilizando uma soma ponderada das suas características. Para isso é necessário atribuir um peso a cada uma das características que descrevem o caso e que serão utilizadas na recuperação.

Na prática, a similaridade (isto é, a proximidade) do caso destino para o caso fonte para cada atributo é determinado. Esta medida é multiplicado por um fator peso. Então a soma da similaridade de todos os atributos é calculada. Esta pode ser representada por uma equação relativamente simples

**(5.1)**

Onde

T é o caso destino

S é o caso fonte

n é o número de atributos em cada caso

i é um atributo individual de 1 até n

é a função de similaridade para atributo i nos casos T e S

w é o peso do atributo i

Algoritmos de similares a este são usados por muitas ferramentas CBR para realizar recuperação do caso mais similar. Similaridade são normalmente para cair dentro da faixa de 0 para 1 (onde o significa totalmente dissimilar e 1 exatamente similar) ou usando um percentual, onde 100% é totalmente similar (Watson, 1999; 2003).

**5.3.5.2 Algoritmo de Indução**

Indução é uma técnica desenvolvida por pesquisadores de Aprendizado de Máquinas para extrair regras ou construir de dados passados. Em sistema CBR, a base de casos é analisada por algoritmo de indução para produzir uma árvore de decisão que classifica (ou indexa) os casos. O algoritmo de indução foi amplamente usado pela ferramenta CBR chamada ID3.

**5.3.6 Adaptação**

A tarefa final do Sistema CBR é adaptar a solução associada a um caso recuperado para as necessidades do problema corrente. Quando uma situação é fornecida, o algoritmo de recuperação traz o melhor caso que ele encontrar para a memória. Normalmente, o caso selecionado não atende perfeitamente coma descrição do problema do usuário. Ou seja, existem diferenças entre o problema do usuário e o caso contido no banco de casos que devem ser levadas em conta. Então, o processo de adaptação procura por diferenças salientes entre as duas descrições e aplica regras de forma a compensá-las. Em geral, existem dois tipos de adaptação em CBR (Watson, 2003):

* **Adaptação Estrutural** - as regras de adaptação são aplicadas sobre a solução armazenada junto aos casos.
* **Adaptação Derivacional** – o algoritmo reusa os algoritmos, métodos ou regras que geraram a solução que consta no banco de casos para gerar uma nova solução para o problema corrente. Neste método, a sequência que construiu a solução original deve ser armazenada juntamente com o caso na memória de casos. O algoritmo de adaptação derivacional exige uma perfeita compreensão dos casos armazenados e da forma como as soluções foram geradas.

Segundo Watson (2003), várias técnicas tem sido usadas em sistema CBR. Incluindo as seguintes:

* **Adaptação nula** – ele simplesmente aplica a solução recuperada ao problema corrente sem modificação. Adaptação nula é útil para problemas envolvendo raciocínio complexo mais com solução simples. Por exemplo, em um sistema para concessão de crédito, embora seja necessário coletar muitas informações do cliente, a solução final de conceder ou rejeitar o crédito é direta.
* **Ajuste por parâmetros** – é uma técnica de adaptação estrutural que compara parâmetros específicos entre o caso recuperado e o novo para modificar a solução armazenada na direção apropriada. Esta técnica foi usada no sistema CBR chamado JUDGE, que recomenda sentenças mais curtas para crimes menos violentos.
* **Reinstanciação** – instancia uma nova solução para um caso recuperado do banco de casos com novas características adequadas ao problema do usuário. Por exemplo, o sistema CBR CHEF, que a partir de uma receita existente, criar uma nova receita.
* **Substituição derivacional** – repete o método, ou parte do método que gerou uma solução armazenada em um caso similar de forma a obter a solução para o novo caso, substituindo os atributos distintos. Como no sistema BOGART que reaplica os planos de geração de projetos para novos problemas.
* **Repara guiado por modelos** – utiliza um modelo casual para adaptar as soluções armazenadas ao problemas do usuário. O sistema CBR, CELIA, utiliza-o para aprendizado e diagnóstico de problemas mecânicos de automóveis.

Finalizando, de acordo com Watson (2003), a adaptação é útil em muitas situações. Mais não significa que seja essencial. Muitos dos sistema CBR comerciais não usam adaptação para tudo. Eles simplesmente reusam a solução sugerida para o melhor caso correspondente (i.e., adaptação nula) ou eles deixam a adaptação para as pessoas.

* 1. **Conclusões**

Pode-se concluir que, O **Raciocínio Baseado em Casos** é um método em que problemas novos são resolvidos através de soluções adaptadas que foram usadas para resolver problemas mais antigos.

Um **Caso** é uma pedaço contextualizado de conhecimento que representa uma experiência. Ao se analisar cada caso, se tem a descrição do problema e a solução armazenada. Caso já exista um problema semelhante já anteriormente armazenado no banco de dados, a solução será recuperada. Porém, se não existir um caso similar, a descrição desse novo problema será enviado ao espaço de problemas, recuperando o caso com o problema mais similar possível, criando uma nova solução (Watson,1997).

Devido a essas características, o **RBC** é indicado para tarefas de segmentação e categorização.

A recuperação de casos é profundamente relacionada e dependente do método de categorização utilizado. As duas técnicas utilizadas atualmente são a de **recuperação por vizinho mais próximo** e **recuperação indutiva**. Na recuperação por vizinho mais próximo, deverão ser definidos os índices dos casos e os seus respectivos pesos, dependendo do problema a ser resolvido. Esses índices deverão ser previsíveis, identificar o propósito em que os casos serão utilizados, serem abstratos o bastante para permitir o uso posterior da base de dados, e serem concretos o bastante para serem reconhecidos no futuro (Watson, 1997). Como explicitado anteriormente, o caso similar será utilizado como solução do problema. No caso de recuperação por indução, uma árvore de decisão é utilizada, classificando os casos.

Parte I

Capítulo

6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Redes Neurais Artificiais

**6.1 Introdução**

As Redes Neurais Artificiais (RNA), também conhecidas como métodos conexionistas, são modelos matemáticos que se assemelham às estruturas biológicas e que têm capacidade computacional adquirida por meio de aprendizado e generalização (Haykin, 2001; Negnevitsky, 2005).

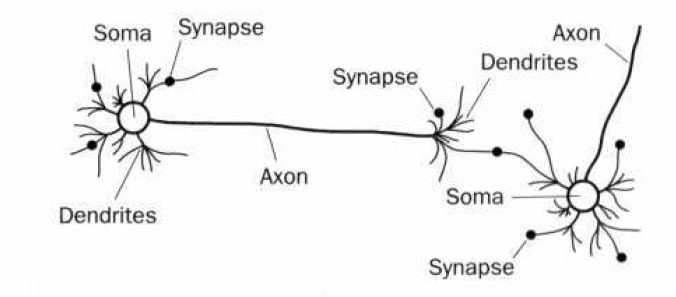
Inicialmente será discutido neste capítulo a representação de conhecimento utilizada pelas Redes Neurais Artificiais, para depois tentarmos analisar a parte referente ao aprendizado. A princípio, é importante salientar que existem diferentes tipos de Redes Neurais Artificiais (**RNA**s) e que cada uma delas tem características próprias em relação a sua representação e aquisição de conhecimento (ou aprendizado) (Osório, 1999).

**6.2 Definição**

Segundo Negnevitsky (2005), uma Rede Neural pode ser definida como um modelo de raciocínio baseado no cérebro humano. O cérebro é composto por um conjunto densamente interligado de células nervosas, ou unidades básicas de processamento, chamada de neurônios. O cérebro humano incorpora cerca de 10 bilhões de **neurônios** e 60 trilhões de conexões, **sinapses**, entre eles (Shepherd e Koch, 1990; citado por Negnevitsky, 2005). Por usar múltiplos neurônios simultaneamente, o cérebro pode realizar suas funções muito mais rapidamente do que o computador mais rápido existente hoje.

Embora cada neurônio tenha uma estrutura muito simples, um exército desses elementos possui um tremendo poder de processamento. Um neurônio consiste de um corpo celular, **soma**, um número de fibras chamadas de **dendritos**, e uma fibra longa chamada de **axônio**. Enquanto os dendritos formam uma rede em torno do corpo celular (soma), o axônio se estende para os dendritos e soma de outros neurônios. A Figura 6.1 é um esquema de uma rede neural (Negnevistsky, 2005).

Os sinais são propagados de um neurônio para outro através de complexas reações eletroquímicas. As substancias químicas liberadas das sinapses causam mudanças no potencial elétrico do corpo celular. Quando o potencial atinge seu limite, um pulso elétrico, é enviado através do axônio. O pulso se espalha e eventualmente atinge as sinapses, causando o aumento ou diminuição de seu potencial. No entanto, a descoberta mais interessante é que a rede neural exibe plasticidade (Negnevistky, 2005).

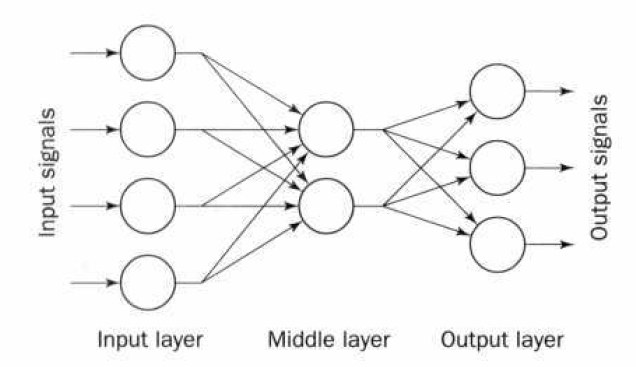


**Figura 6.1** Rede Neural Biológica. **Fonte**: Artificial intelligence: a guide to intelligence systems/Michael Negnevitsky, 2005).

Um neurônio em “desenvolvimento” é sinônimo de um cérebro plástico: a plasticidade permite que o sistema nervoso em desenvolvimento se adapte ao meio ambiente. Assim com a plasticidade parece ser essencial para o funcionamento dos neurônios como unidades de processamento de informações do cérebro humano, também ela o é em relação às redes neurais construídas com neurônios artificiais (Haykin, 2001).

Portanto, de acordo com Negnevitsky (2005), pode-se definir as redes conexionistas ou Rede Neuronal Artificial (RNAs), como sendo formadas por um conjunto de unidades elementares de processamento de informações fortemente conectadas, que denomina-se neurônio artificial. Uma RNA é constituída por um grafo orientado e ponderado. Os nós do grafo são autômatos simples, os chamados neurônios artificiais, que formam através de suas conexões um autômato mais complexo, a rede neural. A Figura 6.2 ilustra uma rede neural artificial.

Segundo Negnevitsky (2005), cada unidade da rede é dotada de um estado interno, que geralmente é denominado de estado de ativação. As unidades podem propagar seu estado de ativação para outras unidades do grafo, passando pelos arcos ponderados, que é chamado de conexões, ligações sinápticas, ou simplesmente de pesos sinápticos. A regra que determina a ativação de um neurônio em função da influência transmitidas de suas entradas, ponderadas pelos seus respectivos pesos, se chama regra de ativação ou função de ativação. E a Tabela 6.1 faz uma analogia entre uma rede biológica e uma rede conexionista.



**Figura 6.2** Arquitetura de uma típica Rede Neural Artificial. **Fonte**: Artificial intelligence: a guide to intelligence systems/Michael Negnevitsky, 2005).

**Tabela 6.1** Analogia entre Rede Neural biológica e Artificial

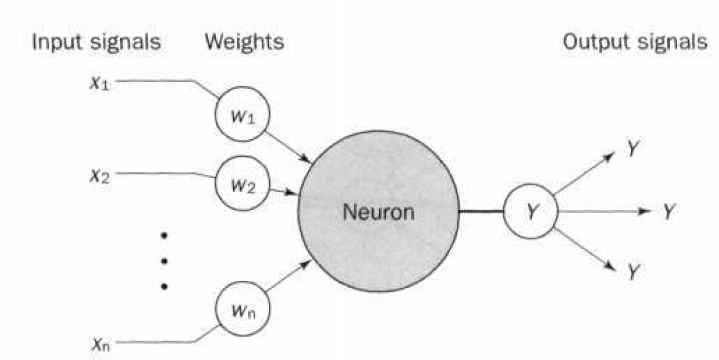
|  |  |
| --- | --- |
| Rede neural biológica | Rede neural artificial |
| Soma | Neuron |
| Dendrite | Input |
| Axon | Output |
| Synapse | weight |

O processamento da informação em RNAs é realizado por meio de estruturas neurais artificiais em que o armazenamento e o processamento da informação são realizados de maneira paralela e distribuída por elementos processadores relativamente simples. Cada elemento processador corresponde a um neurônio artificial (Negnevistsky, 2005).

Segundo Negnevistsky (2005) para se construir uma RNA, deve-se decidir primeiro como os neurônios serão usados e como eles serão conectados na rede. Em outras palavras, deve-se escolher a arquitetura da rede. Então, decide-se que algoritmo de aprendizado usar. E finalmente, treina-se a rede, isto é, inicializa-se os pesos da rede e atualiza-se os pesos para o conjunto de treinamento.

**6.3 O Neurônio**

Um neurônio recebe vários sinais através de seus links de entrada (input), calcula um novo nível de ativação, e envia-o como sinal de saída através dos links de saída. O sinal de entrada pode ser uma linha de dados ou a saída de outro neurônio. O sinal de saída pode ser a solução para o problema ou uma entrada para outro neurônio. A Figura 6.3 mostra um típico neurônio.



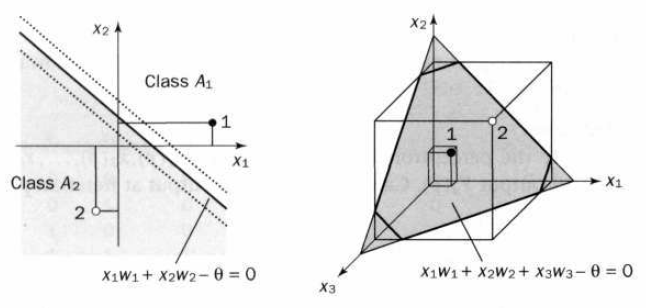
**Figura 6.3** Diagrama de um Neurônio. **Fonte**: Artificial intelligence: a guide to intelligence systems/Michael Negnevitsky, 2005).

As mudanças realizadas nos valores dos pesos sinápticos ou na estrutura de interconexão das unidades de uma rede, são responsáveis pelas alterações no comportamento de ativação desta rede. Estas alterações nas conexões e na estrutura da rede é o que nos permite realizar o aprendizado de um novo comportamento. Desta maneira, pode-se modificar o estado de ativação da saída da rede em resposta a uma certa configuração de entradas. Dessa forma, a rede é capaz de estabelecer associações de entrada-saída (estímulos e respostas) a fim de se adaptar a uma situação proposta. O método utilizado para modificar o comportamento de uma rede é denominado de *regra de aprendizado* (Osório, 1999).

O primeiro modelo matemático do neurônio foi o modelo de proposto por McCulloch e Pitts, em 1943. Mais tarde Rosenblatt (1957) criou o modelo do perceptron. O modelo consiste de um modelador linear seguido de limitador. Um perceptron modela um neurônio tomando uma soma ponderada de suas entradas e compara essa soma com o limiar, o que produz uma saída +1 se suas entradas são positivas e -1 se elas são negativas. O objetivo do *perceptron* é classificar as entradas, ou em outras palavras, aplicar externamente x1, x2, ..., xn estímulos entre umas das classes, ditas A1 e A2. Assim, no caso de um *perceptron* elementar, o espaço n-dimensional é dividido por um hiperplano em duas regiões. O hiperplano é definido por função linearmente separáveis.

**(6.1)**

Para o caso de duas entradas x1 e x2, o limite de decisão leva a forma de uma linha reta como ilustrada na Figura 6.4(a). O ponto, o qual se encontra acima da linha limite, pertence a classe A1; e o ponto 2, o qual se encontra abaixo da linha limite, pertence à classe A2.



1. (b)

**Figura 6.4** Separabilidade linear no perceptrons: (a) duas entradas; (b) três entradas. **Fonte**: Artificial intelligence: a guide to intelligence systems/Michael Negnevitsky, 2005).

O neurônio de McCulloch e Pitts, usa a seguinte transferência ou função de ativação:

**(6.2)**

Onde X é a soma ponderada das entradas do neurônio, e xi é o valor da entrada **i**, wi é o peso da entrada **i**, **n** é o número de neurônios de entrada e **y** é o número de neurônios de saída.

Este tipo de função de ativação é chamada de função sinal (*sign function*).

Assim, a saída do neurônio para uma função sinal, pode ser representado como

**(6.3)**

Segundo Negnevistsky (2005) muitas funções de ativação têm sido testadas, mais somente umas poucas tem tido aplicações práticas. Quatro delas são a função, step, sign, linear e sigmoide.

* As funções de ativação step e sign – também chamadas de ***hard limit functions***, são frequentemente usadas em neurônios para tomada de decisões para tarefas classificação e reconhecimento de padrões.
* A função sigmoide – transforma as entradas, que pode ter qualquer valor entre mais ou menos infinito, em um valor razoável na faixa de 0 e 1.
* A função linear – neurônios com função de ativação linear são frequentemente usadas para aproximações lineares.

**6.4 Classificação e Propriedades**

A grande variedade de modelos existentes nos leva a um estudo ou análise das principais propriedades das redes neurais, que nos permita compreender melhor as vantagens e/ou inconveniências da escolha de um modelo em detrimento de outro. Considere-se que para essa análise sejam avaliados um grupo de atributos tais como: tipo de aprendizado, arquitetura de interconexões, forma interna de representação das informações, tipo de aplicação da rede, etc (Rosa, 1999).

**6.4.1 Aprendizado RNA**

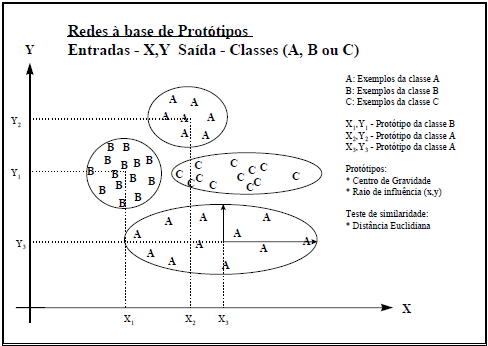
Uma RNA aprende geralmente de forma gradual, onde os pesos são modificados várias vezes, paulatinamente, seguindo-se uma regra de aprendizado que estabelece a forma como estes pesos são alterados. Utiliza-se no aprendizado um conjunto de dados de aprendizado disponível (base de exemplos). Cada iteração deste processo gradativo de adaptação dos pesos de uma RNA, é chamada de época de aprendizado. Os métodos de aprendizado neural podem ser divididos em três grandes classe, segundo a grau de controle dado ao usuário (Rosa, 1999; Rezende, 2003):

* **Aprendizado supervisionado**: o usuário dispõe de um comportamento de referência preciso que ele deseja ensinar a rede. Sendo assim, a rede deve ser capaz de medir a diferença entre seu comportamento atual e o comportamento de referência, e então corrigir os pesos de maneira a reduzir este erro (desvio de comportamento em relação aos exemplos de referência).
* **Aprendizado semi-supervisionado**: o usuário possui apenas indicações imprecisas (por exemplo: sucesso/insucesso da rede) sobre o comportamento final desejado. As técnicas de aprendizado semi-supervisionado são chamadas também de aprendizado por reforço (reinforcement learning) [Sutton 98].
* **Aprendizado não-supervisionado**: os pesos da rede são modificados em função de critérios internos, tais como, por exemplo, a repetição de padrões de ativação em paralelo de vários neurônios. O comportamento resultante deste tipo de aprendizado é usualmente comparado com técnicas de análise de dados empregadas na estatística (*clustering*).
* **Aprendizado instantâneo**: o conjunto de dados de aprendizado é analisado uma única vez e com isto o conjunto de pesos da rede é determinado de maneira imediata em uma única passagem da base de exemplos. Este modo de aprendizado também é conhecido como: *one single epoch learning* / *one shot learning*.
* **Aprendizado por pacotes:** o conjunto de dados de aprendizado é apresentado à rede várias vezes, de modo que possamos otimizar a resposta da rede, reduzindo os erros da rede e minimizando o erro obtido na saída desta. Este modo de aprendizado é caracterizado por trabalhar com uma alteração dos pesos para cada época, ou seja, para cada passagem completa de todos os exemplos base de aprendizado. O algoritmo de aprendizado deve reduzir pouco à pouco o erro de saída, o que é feito ao final de cada passagem (análise) da base de exemplos de aprendizado.
* **Aprendizado continuo:** o algoritmo de aprendizado leva em consideração continuamente os exemplos que lhe são repassados. Se o conjunto de dados é bem delimitado, chamamos este método de aprendizado on-line, e caso o conjunto de dados possa ir aumentando (sendo adicionados novos exemplos no decorrer do tempo), então chamamos este método de aprendizado incremental. O aprendizado on-line se opõe ao aprendizado por pacotes, pois ao contrário deste, para cada novo exemplo analisado já se realiza uma adaptação dos pesos da rede, com o objetivo de convergir na direção da solução do problema. O principal problema do aprendizado contínuo é a dificuldade de achar um bom compromisso entre a plasticidade e a estabilidade da rede. Uma rede com uma grande facilidade de adaptação pode “esquecer” rapidamente os conhecimentos anteriormente adquiridos e uma rede com uma grande estabilidade pode ser incapaz de incorporar novos conhecimentos.
* **Aprendizado ativo:** este modo de aprendizado assume que o algoritmo de adaptação da rede pode passar de uma posição passiva (apenas recebendo os dados do jeito como lhe são passados), para uma posição ativa. Sendo assim, assumimos que este algoritmo poderá vir a intervir sobre a forma como os dados lhe são repassados. Neste caso, a rede pode intervir e determinar assim quais dados que serão considerados e/ou desconsiderados, além também de determinar a ordem em que estes dados deverão ser considerados. A rede pode também vir a solicitar novos dados que julgue necessários para o bom aprendizado do problema proposto.

**6.4.2 Tipos de Unidades**

Segundo Rosa (1999), as unidades de uma rede – os neurônios artificiais – podem ser de diferentes tipos, de acordo com a função interna utilizada para calcular o seu estado de ativação, ou seja, qual a função de ativação utilizada linear, gaussiana, sigmoide, assimétrica, etc. Um outro elemento que pode diferenciar uma unidade, diz respeito a forma como os neurônio armazenam as informações: unidades baseadas em protótipos, unidades do tipo Perceptron.

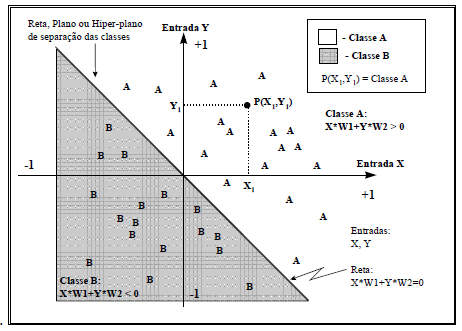
* **Redes à base de protótipos**: este tipo de rede utiliza neurônios que servem para representar protótipos dos exemplos aprendidos – as unidades tem uma representação interna que agrupa as características comuns e típicas de um grupo de exemplos (Orsier 95). As redes baseadas em protótipos tem normalmente um aprendizado não supervisionado (com um ou mais protótipos associados à cada classe). Uma das vantagens deste tipo de redes é a possibilidade de fazer um aprendizado contínuo e incremental, uma vez que não é muito difícil de conceber um algoritmo capaz de aumentar a rede neural através da adição de novos protótipos. Os protótipos são também denominados de clusters, onde apresentamos um exemplo de rede a base de protótipos na Figura 6.5.



**Figura 6.5** Protótipo de uma rede neural com duas entradas. **Fonte**: Osório, Fernando. Redes Neurais – Aprendizado Artificial. Forum de I.A. “99 – pg.13”.

* **Redes à base de Perceptrons**: as unidades do tipo “*Perceptron*” foram criadas por Frank Rosenblatt em 1950. Este é um dos modelos de neurônios mais utilizados na atualidade. Ele é a base de diversos tipos de RNA com aprendizado supervisionado utilizando uma adaptação por correção de erros (usualmente baseada na descida da superfície de erro usando o gradiente). O modelo do Perceptron de múltiplas camadas (**MLP** – *Multi-Layer Perceptron*) tornou-se muito conhecido e aplicado, sendo na maior parte das vezes associado a regra de aprendizado do *Back-Propagation* (Jodoin 94, Widrow 90, Rumelhart 86). A Figura 6.6 apresenta um esquema da representação de conhecimentos nas redes baseadas em *Perceptrons*, e como este tipo de redes é capaz de classificar padrões, gerando planos (ou hiperplanos) de divisão do espaço em que se situam os exemplos (Negnevitsky, 2005).

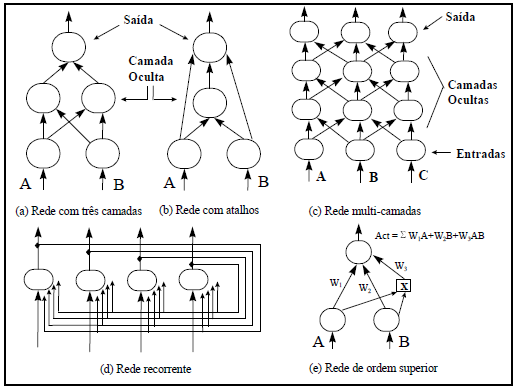
**Figura 6.6** Separação de classes (classificação) através do uso de um Perceptron. **Fonte**: Osório, Fernando. Redes Neurais – Aprendizado Artificial. Forum de I.A. “99 – pg.13”.



**6.4.3 Tipos de Arquiteturas de Conexões de Redes**

As unidades de uma rede neural podem se conectar de diferentes modos, resultando em diferentes arquiteturas de interconexão de neurônios. A Figura 6.7 apresenta alguns exemplos de possíveis maneiras de conectar os componentes de uma RNA. As arquiteturas mais importantes são (Osório, 1999):

* **Redes com uma única camada**: as unidades estão todas em um mesmo nível. Neste tipo de arquitetura, as unidades são conectadas diretamente às entradas externas e estas unidades servem também de saídas finais da rede. As redes de uma única camada possuem normalmente conexões laterais (entre os neurônios de uma mesma camada). Um exemplo deste tipo de arquitetura de redes são as redes do tipo “*Self-Organizing Feature Maps*” (Kohonen,1987).
* **Redes com camadas unidirecional**: as unidades são organizadas em vários níveis bem definidos, que são chamados de camadas ou *layers*. Cada unidade de uma camada recebe suas entradas vindas à partir de uma camada precedente, e envia seus sinais de saídas em direção a camada seguinte. Estas redes são conhecidas como redes *feed-forward*. A Figura 6.7(a) mostra um exemplo de uma rede de três camadas unidirecionais. Esta arquitetura de três camadas (entrada, camada oculta e saída) é muito usada em aplicações práticas das redes neurais.
* **Redes recorrentes**: as redes recorrentes podem ter uma ou mais camadas, mas a sua particularidade reside no fato de que temos conexões que partem da saída de uma unidade em direção a uma outra unidade da mesma camada ou de uma camada anterior à esta. Este tipo de conexões permitem a criação de modelos que levam em consideração aspectos temporais e comportamentos dinâmicos, onde a saída de uma unidade depende de seu estado em um tempo anterior. Os laços internos ao mesmo tempo que dão características interessantes de memória e temporalidade as redes, tornam este tipo de redes muito instáveis, o que nos obriga a usar algoritmos específicos (e usualmente mais complexos) para o aprendizado destas redes. Um tipo particular de redes recorrentes são as redes totalmente conectadas, e um exemplo de modelo recorrente de uma única camada e totalmente conectado são as redes de Hopfield, representadas na figura 6.7(d).
* **Redes de ordem superior:** as unidades deste tipo de rede permitem a conexão direta entre duas ou mais de suas entradas, antes mesmo de aplicar a função de cálculo da ativação da unidade (Fiesler, 1994a). Este tipo de rede serve para modelar “sinapses de modulação”, ou seja, quando uma entrada pode modular (agir sobre) o sinal que vem de uma outra entrada. Um modelo particular de rede de ordem superior são as redes tipo *Sigma-Pi* que foram apresentadas no livro PDP – *Parallel Distributed Processing* (Rumelhart 86), e que são representadas na figura 6.7(e).



**Figura 6.7** Arquiteturas de interconexão de neurônios em redes. **Fonte:** Osório, Fernando. Redes Neurais – Aprendizado Artificial. Forum de I.A. “99 – pg.13”.

A arquitetura de uma rede também pode ser classificada de acordo com a evolução desta no decorrer de sua utilização e desenvolvimento do aprendizado. Em relação a este critério pode-se ter os seguintes tipos (Osório, 99):

* **Redes com estruturas estáticas**: a rede tem a sua estrutura definida antes do início do aprendizado. A quantidade de neurônios, assim como a sua estrutura de interconexões, não sofrem alterações durante a adaptação da rede. As únicas mudanças se realizam à nível dos pesos sinápticos, que são modificados durante o processo de aprendizado (Osório, 98).
* **Redes com estruturas dinâmicas**: as redes que possuem uma estrutura dinâmica são redes onde o número de unidades e conexões pode variar no decorrer do tempo. Estas redes são também chamadas de *ontogênicas* (Fiesler, 1994). As modificações na estrutura da rede podem ser do tipo generativo (incremental) ou do tipo destrutivo (redutor por eliminação/simplificação). A escolha entre estes dois tipos de métodos é bastante polêmica: devemos começar com uma rede pequena e ir aumentando ela, ou devemos começar com uma rede bastante grande e ir reduzindo o seu tamanho posteriormente? Alguns autores defendem a ideia de uma criação construtiva de conhecimentos (Elman 1993, Osório 1999).

**6.5 Tipos de Aplicações para Redes Neurais**

De acordo com diversos autores, as RNAs podem ser aplicadas a diversos tipos de tarefas, tais como: o reconhecimento de padrões (e.g. reconhecimento de faces humanas), a classificação (e.g. reconhecimento de caracteres – **OCR**), a transformação de dados (e.g. compressão de informações), a predição (e.g. previsão de séries temporais, como as cotações da bolsa de valores, ou o uso para diagnósticos médicos), o controle de processos e a aproximações de funções (e.g. aplicações para área de robótica). Todas essas tarefas podem ser agrupadas em dois grandes grupos (Osório, 1999): **Redes para aproximações de funções**, **Redes para classificação de padrões**.

**6.6 Vantagens das Redes Neurais**

De acordo com Osório (1999), as redes conexionistas, em particular aquelas comumente aplicadas na construção de sistemas inteligentes, apresentam as seguintes vantagens:

* **Conhecimento empírico**: em geral as redes aprendem mais fácil do que outros métodos de aquisição de conhecimento, pois o aprendizado acontece a partir de exemplos de maneira simples e permite um aquisição de conhecimento de forma automática.
* **Degradação progressiva**: apesar das redes serem menos sensíveis as perturbações, do que os sistemas simbólicos. As respostas dadas por uma rede se degrada progressivamente na presença de perturbações e distorções dos dados de entrada.
* **Manipulação de dados quantitativos:** as redes trabalham com a representação numérica dos conhecimentos e, isso implica que as redes são melhor adaptadas para a manipulação de dados quantitativos (valores contínuos). Isso pode ser considerado uma vantagem, uma vez que grande parte dos problemas do mundo real, manipulam valores contínuos.
* **Paralelismos em larga escala:** as redes neurais são compostas de um conjunto de unidade de processamento de informações que podem trabalhar em paralelo. Apesar da maioria das implementações de RNAs serem feitas através de simulações em máquinas sequenciais, é possível de se implementar (softwares e hardwares) que possam explorar esta possibilidade de ativação simultânea das unidades de uma rede. A maior parte das implementações de redes neurais simuladas em máquinas sequenciais pode ser facilmente adaptada em uma versão paralela deste sistema.

**6.7 Inconvenientes das Redes Neurais**

As redes apresentam alguns inconvenientes, do mesmo modo que outros tipos de métodos de aprendizado. As redes apresentam os seguintes inconvenientes (Osório, 1999):

* **Arquitetura e parâmetros**: a evolução do processo de aprendizado é bastante influenciado por estes dois parâmetros. Como não existe métodos totalmente automatizados para escolha correta da arquitetura para um problema, fica muito difícil de se encontrar uma boa topologia para a rede, bem como, bons parâmetros de regulagem para o algoritmo de aprendizado. O sucesso da rede depende bastante desses dois elementos, que variam muito de um problema para outro.
* **Inicialização e codificação**: uma má escolha dos pesos iniciais da rede, do método de codificação dos dados de entrada, ou mesmo da ordem de apresentação destes, pode levar ao bloqueio do processo de aprendizado, ou pode dificultar o processo de convergência da rede na direção de uma boa solução. Uma vez que, os algoritmos de aprendizado conexionistas são em geral muito dependentes do estado inicial da rede e da codificação dos dados da base de aprendizado.
* **Caixa preta**: as redes conexionistas são “caixas preta” onde os conhecimentos ficam codificados de tal forma que estes são inteligíveis para o utilizador ou até mesmo para um especialista. Isto pelo fato, de que, os conhecimentos adquiridos por uma rede estão codificados no conjunto de valores dos pesos sinápticos, e também pela maneira pela a qual as unidades se conectam.
* **Conhecimento teórico**: como as árvores de decisão, as redes neurais são orientadas para a aquisição de conhecimentos empíricos (baseados em exemplos). Um modo simplista de se aproveitar algum conhecimento teórico pré-existente, consiste em se converter regras em exemplos (“protótipos” representativos destas regras). Entretanto, este tipo de método não nos garante que a rede será capaz de aprender corretamente estes exemplos, sendo assim, não podemos garantir que ao final do aprendizado todos os conhecimentos teóricos disponíveis estarão bem representados internamente na rede.

Nessa sessão foram apresentadas algumas das vantagens e desvantagens das redes conexionistas, sem a pretensão de cobrir exaustivamente todas elas, mas apenas para se ter uma ideia das principais características deste tipo de sistema.

**6.8 Conclusões**

Neste capítulo apresentou-se uma visão geral sobre os sistemas de I.A. e a necessidade do aprendizado para que um sistema inteligente possa ser considerado como tal. Dando ênfase ao aprendizado neural como sendo uma forma de aquisição de conhecimentos, que dadas as suas peculiaridades, possui um interesse particular na área de inteligência Artificial.

Considerando-se suas principais características: a representação de conhecimentos, o paralelismo inerente as unidades da rede, a sua capacidade de adaptação, entre outros aspectos. No entanto, observa-se que as redes neurais possuem ainda alguns pontos fracos a serem estudados, principalmente no que diz respeito a explicitação dos conhecimentos adquiridos e na dificuldade de convergência em relação a uma solução ótima.

Parte II

Capítulo

7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Projeto

Parte II

Capítulo

8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Data Mining Aplicada ao Estudo de Caso

Parte II

Capítulo

9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Conclusões e Trabalhos Futuros

Capítulo

10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Referencial Bibliográfico

Agrawal, R.; IMIELINSKI, T.; SWAMI, A. Mining Association Rules Between Sets of Itens in Large Databases. ACM SIGMOD Conference Management of Data, 1993.

Dean, Jeffrey; Ghemawat, Sanjay. MapReduce: Simplified Data Processing On Large Cluster. CUMMUNICATIONS OF THE ACM, janeiro de 2008.

Dong, G. & J. LI. Interestingness of discovered association rules in terms of neighbordhood-based unexpectedness. Lecture Notes in Artificial Intelligence, pp. 72-86, 1998.

Elman, Jeffrey L. *Learning and Development in Neural Networks: The Importance of Starting Small*. Cognition, 48(1993), pp.71-99. 1993. Web: http://crl.ucsd.edu/~elman/

Ftp: <ftp://crl.ucsd.edu/pub/neuralnets/cognition.ps.Z>

Elmasri, Ramez; Navathe, Shamkant B. Sistemas de Banco de Dados. São Paulo: Addison Wesley, 2005.

Engels. R. Planning tasks for knowledge discovery in databases: Performing Task-Oriented User-Guidance. Proceeding of the International Conference on Knowledge Discovery and Data Mining. Portland: AAAI Press, 1996.

Engels, R.; LINDNER, G.; STUDER, R. A Guided Tour Through the Data Mining Jungle. Proceeding of the Third International Conference on Knowledge Discovery in Databases. Newport Beach, 1997.

Fayyad, Usama; PIATETSKI-SHAPIRO, Gregory; SMYTH, Padhraic (1996). The KDD Process for Extracting Useful Knowledge from Volumes of Data. In: Communications of the ACM, pp.27-34, Nov.1996.

Fiesler, E. *Neural Networks Formalization and Classification.* Computer Standard & Interfaces, Special Issue on Neural Networks Standards, John Fulcher (Ed.). V.16, N.3. Elsevier Sciences Publishers, Amsterdam, June, 1994. Web: http://www.idiap.ch/idiap-networks.html.

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

Freitas A. A. On rule interestingness measures. Knowledge-Based Systems 12(5-6), 309-315, 1999.

Goldschmidt, R.; Passos, E.; Vellasco, M.; Pacheco, M. Task Definition Assistence in KDD Applications. CLEI’03 – XXIX Conferência Latino Americana de Informática. La Paz, 2003.

Han, Jiawei; Kamber, Micheline. Data Mining: Concepts and Techniques. Second Edition. Elsevier. San Francisco, CA, 2006.

Han, Jiawei; Kamber, Micheline; Pei, Jian. Data Mining: Concepts and Techniques. Third Edition. Elsevier. San Francisco, CA, 2011.

Haykin, Simon. Redes neurais: princípios e práticas/Simon Haykin; trad. Paulo Martins Engel. – 2.ed. – Porto Alegre: Bookman, 2001

Hussain F.; Liu H.; Suzuki E.; Lu H. EXCEPTION RULE MINING WITH

RELATIVE INTERESTINGNESS MEASURE. PAKDD, 2000; pg 86-97.

Inmon, Bill & Chuck Kelly. The Twelve Rules of Data Warehouse for a Client/Server World, Data Management Review, 1994. (\*\*\*\*-)

Kimball, Ralph. Data Warehouse toolkit: o guia completo para modelagem multidimensional /Ralph Kimball, Margy Ross; tradução Ana Beatriz Tavares, Daniela Lacerda. Rio de Janeiro: Campus, 2002.

Kolb, Jason; KOLB. Jeremy. The Big Data Revolution. The Tricks Tour Competitors Don’t Want You To Know By Jason Kolb and Jeremy Kolb. AppliedData Labs. Plainfield, IL, 2013.

Kohonen, Teuvo. *Self-Organization and Associative Memory*. Springer-Verlag Series in

Information Science. 1987.

Kolodner, J. L. Proceedings of the DARPA Case-Based Reasoning Workshop. San Francisco: Morgan Kaufmann Publishers, 1998

Liu, B. & W. Hsu. Post-analysis of learned rules. AAAI 1, 828-834, 1996.

Mayer-Schönberger, Viktor; Cukier, Kenneth. Big Data. A Revolution That Will Transform How We Live, Work and Things. First published in Greta Britain. John Murray (Publishers) an Hachette UK Compnay, 2013.

Morik, K. The Representation Race- Preprocessing for Handling Time Phenomena. Proceedings of the European Conference on Machine Learning 2000, Lecture Notes in Artificial Intelligence 1810. Berlin: Springer Verlag, 2000

Negnevitsky, Michael. Artificial Intelligence: a guide to intelligent Systems/Michael Negvitsky. Pearson Education Limited. Edinburgh Gate, 2005.

Oliveira, C., EDACLUSTER: Um Algoritmo Evolucionário para Análise de agrupamentos Baseados em Densidade e Grade, Dissertação (Mestrado em Engenharia Elétrica), Universidade Federal do Pará, 2007.

Osório, Fernando. Redes Neurais – Aprendizado Artificial. Forum de I.A. “99 – pg.13”. Rosa, João Luís Garcia. Fundamentos da Inteligência artificial /João Luís Garcia Rosa. Rio de Janeiro: LTC, 2011.

Passos, Emanuel; GOLDSCHMIDT, Ronaldo. Data Mining: Um guia prático. Editora Campos. Rio de Janeiro, 2005.

Pazzini, M. J. Knowledge discovery from data? IEEE Intelligent Systems, 10-13, 2000.

Piatetsky-Shapiro, G & C. J. Matheus. The Interestingness of deviations. Em Proceedings of the International Conference on Knowledge Discovery and Data Mining, pp. 23-36, 1994.

Raj, Subu. BIG DATA – AN INTRODUCTION. Kindle Ver 1.1, 2013.

Rezende, Solange Oliveira. Sistemas inteligentes: fundamentos e aplicações. Editora Manole Ltda. Barueri, SP. 2003.

Riesbeck, C. K., and Schank, R. Inside Case-Based Reasoning. Northvale, NJ: Lawrence Erlbaum Associates, 1996.

Rob, Peter; Coronel, Carlos. Database Systems: Design, Implementation, and Management by Peter Rob and Carlos Coronel 8th Edtion. Thomson Place, Boston, Massachusetts, 2009.

Robt, Peter. Sistemas de Banco de Dados: Projeto, implantação e gerenciamento / Peter Rob, Carlos Coronel. São Paulo: Cengage Learning, 2011.

Silberschatz, A. & Tuzhilin. On subjective measures of interestingness in knowledge discovery. Proceeding of the First International Conference on Knowledge Discovery and Data Mining 1, 275-281, 1995.

Stonebraker, Michael, Abadi; Daniel, DeWitt; David J.; Madden, Sam; Paulson, Erik; Pavlo, Andrew; Rasin, Alexander.MapReduce complements DBMSs since databases are not designed for extracttransform-load tasks, a MapReduce specialty. COMMUNICATIONS OF THE ACM, pp 71. Publicado em Janeiro de 2101.

Utgolff, P. Shift of Bias for Inductive Concept Learning. Machine Learning: an Artificial Intelligence Approach, v.3, São Francisco: Morgan Kaufmann, 1996.

Watson, Ian D. Applying case-based reasoning: techniques for enterprise systems. San Francisco, CA: Morgan Kaufmann Publishers, Inc, 1997.

Watson, Ian D. Applying Knowledge Management: Techniques for Building Corpoarte Memory. San Francisco, CA: Morgan Kaufmann Publishers, Inc, 2003.

Witten, Ian H.; Frank, Eibe; Hall, Mark A. Data Mining. Practical Machine Learning Tools and Techiniques. 2nd ed. Morgan Kaufmann Publishers is an imprint of Elsevier, 2005.

Witten, Ian H.; Frank, Eibe; Hall, Mark A. Data Mining. Practical Machine Learning Tools and Techiniques. Third Edition. Morgan Kaufmann Publishers is an imprint of Elsevier, 2011.

**Leituras Complementares**

Boente, A.N.P. Descoberta de Conhecimento em Bases de Dados. Iowa, Tese de Doutorado – Departamento de Informática, AWU – American World University, 2006.

Horst, P. S. Avaliação do conhecimento adquirido por algoritmos de aprendizado de máquina utilizando exemplos. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação. São Paulo, SP – Brasil, 1999.

Larson, Brain; Davis, Mark; English, Dan; Purigton, Paul. Visualizing Data with Microsoft Power View. McGraw Hill companies, 2012.

Loukides, Mike. What is Data Science? The future belongs to the companies and people that turn data into products. O’Reilly, 2011

Mayer-Schönbeger, Victor; Curier Kenneth. A Revolution That Will Transform How We Live, Work and Think. First published in Great Britain in 2013.

Milani, Cristian Simioni, Carvalho, Deborah Ribeiro. PÓS-PROCESSAMENTO EM KDD. Revista de Engenharia e Tecnologia. PUCPR, ISSN 2176-7270, <http://www.revistaret.com.br/ojs-2.2.3/index.php/ret/article/viewFile/170/182>, pp. 153-155, 2013.

Noren, Allen. Big Data Now. Current Perspectives from O’Reilly Radar. O’Reilly Strata Making Data Work, 2011.

Özu, M. Tamer. Princípio de sistemas de banco de dados distribuídos/M. Tamer Özu, Patrick Valduriez; tradução [da 2. Ed. Americana] Vandenberg D. de Souza – Rio de Janeiro: Campus, 2001

Payandeh, Fari. BI vs. Big Data vs. Data Analytics By Example. <http://bigdatastudio.com/2013/08/24/bi-vs-big-data-vs-data-analytics-by-xample/>. Acessado em 27 de setembro de 2013.

Patil, DJ. The skills, Tools, and Perspectives Behind Great Data Science Groups. Building Data Science Teams. O’Reilly Strata Makin Data Work, 2011.

Rhoton, John; Haukioja, Risto. Cloud Computing Architected. Solution Design HandBook. Recursive Press, 2011.

Schank, R. Dynamic Memory: A Theory of Learning in Computers and People. New York: Cambridge University Press, 1982.

Schroeder, Christine da Silva. Critérios e indicadores de desempenho para sistemas de treinamento corporativo virtual: um modelo para medir resultados. Dissertação de mestrado – Universidade Federal do Rio Grande do Sul, Escola de Administração, Programa de Pós-Graduação em Administração, 2005.

Silberschatz, Abraham; Korth, Henry F. e Sudarshan, S. Sistema de Banco de Dados. São Paulo: Makron Books, 1999.

Sosinsky, Barrie. Cloud Computing Bible. Wiley Publishing, Inc, 2011.

Velte, Anthony T.; Velte, Toby J.; Elsenpeter, Robert. Cloud Computing: Computação em Nuvem: Uma Abordagem Prática. ALTA BOOKS. Rio de Janeiro, 2012.

Zikopoulos, Paul C.; Eaton, Chris; deRoos, Dirk; Deutsch, Thomas; Lapis, George. Understanding Big Data.Anaylsis for Enterprise Class. Hadoop and Streaming Data. McGraw-Hill, 2012.