UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

SANDRO DA SILVA CAMARGO

Um Modelo Neural de Aprimoramento Progressivo para Redução de Dimensionalidade

Tese apresentada como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação

Prof. Dr. Paulo Martins Engel
Orientador

Porto Alegre, junho de 2010.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Camargo, Sandro da Silva

Um Modelo Neural de Aprimoramento Progressivo para Redução de Dimensionalidade / Sandro da Silva Camargo – Porto Alegre: Programa de Pós-Graduação em Computação, 2010.

107 f.:il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação. Porto Alegre, BR – RS, 2010. Orientador: Paulo Martins Engel.

1. Heurística 2. Wrapper 3. Redução de dimensionalidade 4. Seleção de características 5. Modelagem neural.. I. Engel, Paulo Martins. III. Título.

AGRADECIMENTOS

Este trabalho deve muito a algumas pessoas e instituições que eu gostaria de agradecer especialmente:

Ao meu orientador, Prof. Dr. Paulo Martins Engel, por todo o estímulo, compreensão e auxílio à realização deste trabalho.

Ao professores do Instituto de Informática por terem contribuído com a minha formação.

Aos professores do Centro de Biotecnologia que, acima de tudo, me mostraram a ciência sob uma perspectiva diferente e fascinante.

Aos grandes amigos que fiz em todo o período de minha formação no instituto de informática.

À minha família, por todo apoio, carinho, amor e por suportarem pacientemente minha luta durante todos estes últimos anos.

Ao Exército Brasileiro, especialmente ao 1º CTA, que teve papel fundamental no suporte financeiro à realização deste meu sonho.

Obrigado.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

ACP	Análise de Componentes Principais
ACI	Análise de Componentes Independentes
ADL	Análise de Discriminantes Lineares
AG	Algoritmos Genéticos
BP	*Backpropagation*
CBR	*Case Based Reasoning*, ou Raciocínio Baseado em Casos
CCS	Contagem de Células Somáticas
DCBD	Descoberta de Conhecimento em Banco de Dados
EB	Exabyte
DAM	Desvio Absoluto Médio
EQM	Erro Quadrado Médio
GLS	*Generalized Least Squares*
IA	Inteligência Artificial
kNN	*k Nearest Neighbor*, ou *k*-ésimo vizinho mais próximo
MB	Megabyte
MD	Mineração de Dados
MLP	*Multi Layer Perceptron*
OLS	*Ordinary Least Squares*
PLI	Programação em Lógica Indutiva
RDD	Redução de Dimensionalidade dos Dados
RGS	*Regression Gradient guided feature Selection*
RNAs	Redes Neurais Artificiais
ROC	*Receiver Operating Characteristic*
SSC	Seleção de Subconjunto de Características
TLFN	*Time Lagged Feed-Forward Network*

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Nas últimas décadas, avanços em tecnologias de geração, coleta e armazenamento de dados têm contribuído para aumentar o tamanho dos bancos de dados nas diversas áreas de conhecimento humano. Este aumento verifica-se não somente em relação à quantidade de amostras de dados, mas principalmente em relação à quantidade de características descrevendo cada amostra. A adição de características causa acréscimo de dimensões no espaço matemático, conduzindo ao crescimento exponencial do hipervolume dos dados, problema denominado "maldição da dimensionalidade". A maldição da dimensionalidade tem sido um problema rotineiro para cientistas que, a fim de compreender e explicar determinados fenômenos, têm se deparado com a necessidade de encontrar estruturas significativas ocultas, de baixa dimensão, dentro de dados de alta dimensão. Este processo denomina-se redução de dimensionalidade dos dados (RDD). Do ponto de vista computacional, a conseqüência natural da RDD é uma diminuição do espaço de busca de hipóteses, melhorando o desempenho e simplificando os resultados da modelagem de conhecimento em sistemas autônomos de aprendizado.

Dentre as técnicas utilizadas atualmente em sistemas autônomos de aprendizado, as redes neurais artificiais (RNAs) têm se tornado particularmente atrativas para modelagem de sistemas complexos, principalmente quando a modelagem é difícil ou quando a dinâmica do sistema não permite o controle *on-line*. Apesar de serem uma poderosa técnica, as RNAs têm seu desempenho afetado pela maldição da dimensionalidade. Quando a dimensão do espaço de entradas é alta, as RNAs podem utilizar boa parte de seus recursos para representar porções irrelevantes do espaço de busca, dificultando o aprendizado. Embora as RNAs, assim como outras técnicas de aprendizado de máquina, consigam identificar características mais informativas para um processo de modelagem, a utilização de técnicas de RDD frequentemente melhora os resultados do processo de aprendizado.

Este trabalho propõe um *wrapper* que implementa um modelo neural de aprimoramento progressivo para RDD em sistemas autônomos de aprendizado supervisionado visando otimizar o processo de modelagem. Para validar o modelo neural de aprimoramento progressivo, foram realizados experimentos com bancos de dados privados e de repositórios públicos de diferentes domínios de conhecimento. A capacidade de generalização dos modelos criados é avaliada por meio de técnicas de validação cruzada. Os resultados obtidos demonstram que o modelo neural de aprimoramento progressivo consegue identificar características mais informativas, permitindo a RDD, e tornando possível criar modelos mais simples e mais precisos. A implementação da abordagem e os experimentos foram realizados no ambiente Matlab, utilizando o *toolbox* de RNAs.

Palavras-Chave: Heurística, *wrapper*, redução de dimensionalidade, seleção de características, modelagem neural, aprimoramento progressivo.

A Progressive Enhancement Neural Model for dimensionality reduction

ABSTRACT

In recent decades, advances on data generation, collection and storing technologies have contributed to increase databases size in different knowledge areas. This increase is seen not only regarding samples amount, but mainly regarding dimensionality, i.e. the amount of features describing each sample. Features adding causes dimension increasing in mathematical space, leading to an exponential growth of data hypervolume. This problem is called "the curse of dimensionality". The curse of dimensionality has been a routine problem for scientists, that in order to understand and explain some phenomena, have faced with the demand to find meaningful low dimensional structures hidden in high dimensional search spaces. This process is called data dimensionality reduction (DDR). From computational viewpoint, DDR natural consequence is a reduction of hypothesis search space, improving performance and simplifying the knowledge modeling results in autonomous learning systems.

Among currently used techniques in autonomous learning systems, artificial neural networks (ANNs) have becoming particularly attractive to model complex systems, when modeling is hard or when system dynamics does not allow on-line control. Despite ANN being a powerful tool, their performance is affected by the curse of dimensionality. When input space dimension is high, ANNs can use a significant part of their resources to represent irrelevant parts of input space making learning process harder. Although ANNs, and other machine learning techniques, can identify more informative features for a modeling process, DDR techniques often improve learning results.

This thesis proposes a wrapper which implements a Progressive Enhancement Neural Model to DDR in supervised autonomous learning systems in order to optimize the modeling process. To validate the proposed approach, experiments were performed with private and public databases, from different knowledge domains. The generalization ability of developed models is evaluated by means of cross validation techniques. Obtained results demonstrate that the proposed approach can identify more informative features, allowing DDR, and becoming possible to create simpler and more accurate models. The implementation of the proposed approach and related experiments were performed in Matlab Environment, using ANNs toolbox.

Keywords: Heuristics, wrapper, dimensionality reduction, feature selection, neural modeling.

1 INTRODUÇÃO

Nas últimas décadas, o desenvolvimento de novas tecnologias de geração e aquisição de dados e a facilidade de obtenção de dados através de simulação, aliadas à redução do custo de armazenamento, têm contribuído para uma sobrecarga de informação nas mais diversas áreas de conhecimento humano. A quantificação desta sobrecarga foi foco de uma pesquisa realizada por Lyman e Varian (2003), onde foi evidenciado que a quantidade de informação digital produzida e armazenada ao redor do mundo dobrou entre 1999 e 2002, tendo crescido 30% a cada ano. Lyman e Varian também concluem que o mundo produz entre 1 e 2 EB de informação não redundante por ano, o que representaria aproximadamente 250MB para cada ser humano.

Além da grande quantidade de dados, também tem se tornado comum a alta dimensionalidade, ou seja, a análise de cada amostra em relação a um grande número de características, que podem atingir quantidades de milhares ou até de milhões. Como conseqüência disso, diversos domínios da ciência, tais como: bioinformática, telecomunicações, astronomia, climatologia, computação, economia, geologia e medicina estão frente a frente com um enorme desafio: aprender a nadar em um imenso mar de dados, ao invés de afogar-se nele. A busca de algum sentido e de compreensão dos dados armazenados torna-se uma necessidade premente, sendo esta busca uma tarefa relativamente trivial para cientistas que estejam muito bem familiarizados com o domínio do problema. Por outro lado, as áreas clássicas do conhecimento humano já estão saturadas, havendo pouco a ser descoberto, fazendo com que os cientistas busquem explorar novas fronteiras do conhecimento em territórios desconhecidos, onde ainda não existem especialistas do domínio. Nestes novos territórios, a busca por conhecimento em grandes bancos de dados é geralmente longa e árdua, sendo notória a carência por ferramentas automáticas eficientes para a exploração destes dados.

Além da carência de especialistas do domínio, à medida que aumenta a quantidade de dados, a dificuldade de compreensão dos dados também é incrementada. Isto ocorre porque o aumento linear da quantidade de características conduz a um crescimento exponencial do hipervolume dos dados. Bellman (1961) criou o termo “maldição da dimensionalidade” para referenciar o problema do crescimento exponencial do hipervolume como função da dimensionalidade dos dados.

Para superar o desafio de compreender sistemas a partir dos dados gerados por eles, cientistas têm utilizado em larga escala técnicas de modelagem. Os modelos matemáticos criados são usados para controle de processos contínuos, investigação de propriedades dinâmicas de processos, otimização de processos, ou para cálculo de condições ótimas de funcionamento de processos (MIKLES E FIKAR, 2007). A utilização de técnicas de modelagem matemática consiste atualmente em um dos pilares da evolução científica.

A maldição da dimensionalidade é um fator desafiador na modelagem matemática visto que, para um hiperplano cartesiano com d dimensões de entrada onde cada dimensão de entrada é particionada em s células, o número total de células seria de s^d (BELLMAN, 1961). Como conseqüência disso, a criação de modelos destes dados necessita considerar espaços de busca inerentemente esparsos (LAROSE, 2006). Desta forma, os cientistas constantemente têm se deparado com a necessidade de encontrar estruturas significativas ocultas, de baixa dimensão, dentro de dados de alta dimensão, sendo tal técnica denominada de redução de dimensionalidade dos dados (RDD). Analogamente, o cérebro humano se confronta com o mesmo problema em suas percepções diárias, extraindo, de forma eficiente, um pequeno número de estímulos relevantes a partir de aproximadamente 30.000 fibras nervosas sensoriais (TENENBAUM et al., 2000). Dada a capacidade limitada do cérebro humano de lidar com a complexidade, esta RDD consiste em um fator chave para permitir a generalização de conceitos transformando as experiências diárias em conhecimento e idéias (TSIEN, 2007). Adicionalmente, a quantidade de exemplos necessários para adaptar um modelo multivariado cresce exponencialmente em relação à quantidade de características que representam cada amostra.

Sob o ponto de vista computacional, a RDD é um processo utilizado a fim de conduzir a uma redução do espaço de busca de hipóteses, permitindo melhorar o desempenho e simplificar os resultados do processo de modelagem (WANG e XIUJU, 2005). Além disso, o uso de muitas variáveis no modelo preditivo pode dificultar a interpretação da análise e viola o princípio da parcimônia, podendo também facilmente conduzir a uma superadaptação (LAROSE, 2006). Embora os algoritmos de mineração de dados já apliquem internamente a seleção das características mais informativas, ignorando as menos informativas, a aplicação de técnicas de RDD geralmente melhora o desempenho destes algoritmos (WITTEN e FRANK, 2005).

Existem diversas técnicas para modelagem matemática em sistemas autônomos de aprendizado (ALPAYDIN, 2010). Dentre estas técnicas, as RNAs foram desenvolvidas como uma generalização de um modelo matemático da cognição humana. RNAs têm se tornado particularmente atrativas para modelagem de sistemas complexos quando a modelagem é difícil ou quando a dinâmica do sistema não permite o controle *on-line*. RNAs criam modelos que representam um mapeamento de um espaço de entradas para um espaço de saídas, servindo como aproximador universal de funções contínuas. Apesar de serem uma poderosa ferramenta, as RNAs têm seu desempenho afetado pela maldição da dimensionalidade.

De acordo com Bishop (1995), em técnicas de modelagem neural a maldição da dimensionalidade manifesta-se de duas formas:

- A existência de muitas características irrelevantes, fator característico de dados com alta dimensionalidade, faz com que a rede utilize quase todos seus recursos para representar porções irrelevantes do espaço de busca.
- Mesmo que a rede consiga focar em características importantes, uma maior quantidade de amostras será necessária para identificar que características são mais ou menos importantes.

Para minimizar o problema da maldição da dimensionalidade, podem ser adotadas duas técnicas: utilização de informação *a priori* ou RDD. Em muitos casos, pela indisponibilidade de informação que possa ser utilizada *a priori*, a RDD é a única alternativa viável.

A utilização eficiente de técnicas de RDD na modelagem neural, possivelmente, irá diminuir a dimensionalidade do espaço de entradas. A conseqüência disso é que uma rede com menos entradas tem menos parâmetros adaptativos a serem determinados, e estes são mais suscetíveis a serem propriamente determinados por um conjunto de dados de tamanho limitado. Isto conduziria a uma rede com melhores propriedades de generalização. Adicionalmente, uma rede com menor quantidade de pesos pode ser mais rápida de treinar.

Entretanto, na maioria das situações, a RDD do vetor de entradas poderá resultar em perda de informação. O grande desafio no projeto de uma boa estratégia de RDD é assegurar que o máximo de informação relevante seja retida. Se muita informação é perdida então a redução resultante no desempenho é maior que qualquer melhora obtida com a RDD.

Conforme Ye (2003), a RDD pode ser dividida em três categorias: seleção de subconjunto de características, extração de características e construção de características. Esta proposta está focada especificamente na categoria de seleção de subconjunto de características, em virtude de esta categoria permitir a criação de modelos mais facilmente explicáveis.

1.1 Objetivos e Escopo da Proposta

O objetivo geral deste trabalho é apresentar e validar um modelo neural de aprimoramento progressivo para redução de dimensionalidade a fim de permitir a construção de modelos preditivos mais precisos e simples de forma mais rápida.

A fim de atingir este objetivo, são apresentados conceitos básicos sobre os processos de descoberta de conhecimento em bancos de dados, pré-processamento, mineração de dados e avaliação de modelos. Serão tratados os fundamentos básicos das RNAs e o algoritmo *backpropagation* para tarefas de aprendizado supervisionado. Adicionalmente, são abordados alguns problemas gerados pela alta dimensionalidade, os fundamentos das técnicas de RDD e especialmente a seleção de subconjunto de características. Também são relatados alguns experimentos que comprovam empiricamente a eficiência da abordagem proposta em diferentes bases de dados.

1.2 Aplicações da abordagem proposta

A abordagem aqui proposta já foi aplicada em problemas de diversos domínios de conhecimento, durante o desenvolvimento desta pesquisa:

a) Predição de séries temporais: a abordagem proposta foi aplicada a dois bancos de dados de séries temporais utilizados para comparação de desempenho em uma competição de redes neurais. Os resultados obtidos nos experimentos levaram o nosso trabalho a ficar entre os três selecionados para apresentação no evento (CAMARGO e ENGEL, 2005).

b) Regressão: a abordagem proposta foi aplicada a problemas de regressão em bases de dados da área de petrologia visando à criação de modelos preditivos de qualidade de reservatórios de hidrocarbonetos. Esta aplicação está descrita em um relatório de pesquisa (CAMARGO, 2005) e três artigos (CAMARGO e ENGEL, 2009, 2010-a, 2010-b) e deu origem a um projeto de pesquisa aprovado para financiamento pelo CNPq (ENGEL, 2005). Outra aplicação desta

abordagem foi realizada em uma base de dados metabólicos de bovinos de leite, visando a criação de modelos preditivos de qualidade do leite. Estes resultados foram apresentados em Campos et al. (2006).

c) Classificação: também na área de veterinária, foram realizados experimentos de classificação visando identificar fatores que possam contribuir para a existência de mastite em bovinos de leite. Tais resultados ainda não foram publicados.

1.3 Organização da Proposta

Esta proposta está organizada da seguinte forma. No capítulo 2 são apresentados os fundamentos básicos sobre aprendizado de máquina, sobre o processo de descoberta de conhecimento em banco de dados e suas fases, abordando pré-processamento e mineração de dados, assim como as formas de validar e comparar os modelos criados.

No capítulo 3 os fundamentos básicos de redes neurais são apresentados. São abordados conceitos dos neurônios naturais, neurônio artificial, funções de ativação, arquiteturas de redes neurais e o algoritmo *backpropagation*.

No capítulo 4 são apresentados os maiores problemas e os conceitos fundamentais a respeito de RDD. São apresentadas algumas técnicas de RDD, sendo abordadas extração, construção e seleção de características. Uma ênfase especial é dada sobre as técnicas de Seleção de Subconjunto de Características (SSC). É feita uma análise do estado da arte em técnicas de RDD.

No capítulo 5 é apresentada a proposta de um modelo neural de aprimoramento progressivo para redução de dimensionalidade. São apresentadas também as evidências experimentais do funcionamento da abordagem e os resultados de experimentos sobre bancos de dados reais para validar a proposta. Estes experimentos foram executados em bancos de dados privados e de repositórios públicos visando descobrir diferentes tipos de conhecimento. Os experimentos foram divididos em três classes de problemas: séries temporais, classificação e regressão.

O capítulo 6 apresenta as conclusões da tese e aponta as direções a serem exploradas nos trabalhos futuros.

2 DESCOBERTA DE CONHECIMENTO EM BANCO DE DADOS

Este capítulo aborda as principais características do processo de descoberta de conhecimento em banco de dados. Inicialmente é apresentada uma visão geral sobre o panorama atual do excesso de informação e, logo após, é discutida a hierarquia de conteúdo da mente humana, a fim de delimitar alguns dos principais termos discutidos no trabalho. A seguir é delineada uma breve contextualização sobre as áreas da inteligência artificial e aprendizado de máquina. Posteriormente, é abordada a área de descoberta de conhecimento em banco de dados e suas fases, descrevendo as tarefas específicas que podem ser realizadas no pré-processamento e no pós-processamento. Os principais tipos de conhecimento que podem ser buscados na fase de mineração de dados também são comentados.

2.1 Excesso de informação: Panorama atual

Nas últimas décadas, a constante evolução das tecnologias de geração e coleta de dados, aliada à progressiva redução do custo de armazenamento e ampla utilização de tecnologias de comunicação, tem contribuído para que a quantidade de dados armazenados de forma eletrônica cresça exponencialmente, conduzindo a uma sobrecarga de informação na maioria das áreas de conhecimento humano.

Lyman e Varian (2003), a fim de estimar a quantidade de informação gerada e armazenada no mundo, pesquisaram as mídias mais comuns para disseminação de informação. Neste trabalho foi evidenciado que a quantidade de informação digital produzida e armazenada ao redor do mundo dobrou entre 1999 e 2002, tendo crescido 30% a cada ano. Eles também concluíram que o mundo produz entre 1 e 2 EB de informação não redundante por ano, o que representaria aproximadamente 250MB para cada ser humano.

Outros trabalhos, como o realizado por Resta (2002), estudaram esta nova sociedade global baseada em conhecimento. Entre os fatos mais marcantes evidenciados, o autor salienta que:

- O conhecimento do mundo dobra a cada 2 ou 3 anos.
- 7000 artigos técnicos e científicos são publicados a cada dia.
- Dados enviados de satélites que orbitam o planeta transmitem dados suficientes para preencher 19 milhões de volumes, de 650MB, a cada duas semanas.

- Estudantes de escolas de ensino médio em países industrializados são expostos a mais informação que seus avós foram durante toda a vida.
- Haverá mais mudanças nas próximas 3 décadas que nos últimos 3 séculos.

Adicionalmente, a grande maioria das pesquisas focadas em discutir a sobrecarga de informação chega ao consenso que na atual era do desenvolvimento humano, processos como criação, distribuição, difusão, uso e manipulação da informação tornaram-se uma importante atividade econômica, política e cultural. Tal sobrecarga de informação levou a criação de rótulos como "sociedade da informação" e "sociedade do conhecimento", para referir-se a esta era que vivenciamos, onde a geração e o acúmulo de dados têm se tornado quase uma obstinação. Pelo fato da informação e do conhecimento terem um papel central na maioria das atividades humanas, tem-se a percepção de que o conhecimento é um fator chave para o sucesso em várias áreas, sejam elas comerciais, governamentais ou científicas.

Apesar da evidente importância do conhecimento, existe uma clara distância entre a capacidade de geração e armazenamento de dados e a capacidade de analisar estes dados a fim de se obter o conhecimento. Além disso, também é comum não existir uma compreensão clara do que efetivamente é o conceito de conhecimento, e de que forma ele se diferencia dos conceitos de dados e informação. Na seção 2.2 é abordada a hierarquia do conteúdo da mente humana, esboçando o limite destes conceitos.

2.2 A hierarquia de conteúdo da mente humana

Segundo Ackoff (1989), o conteúdo da mente humana pode ser dividido em cinco categorias: dados, informação, conhecimento, compreensão e sabedoria, conforme apresentado na figura 2.1. Estas categorias são discutidas a seguir, em ordem crescente de complexidade.

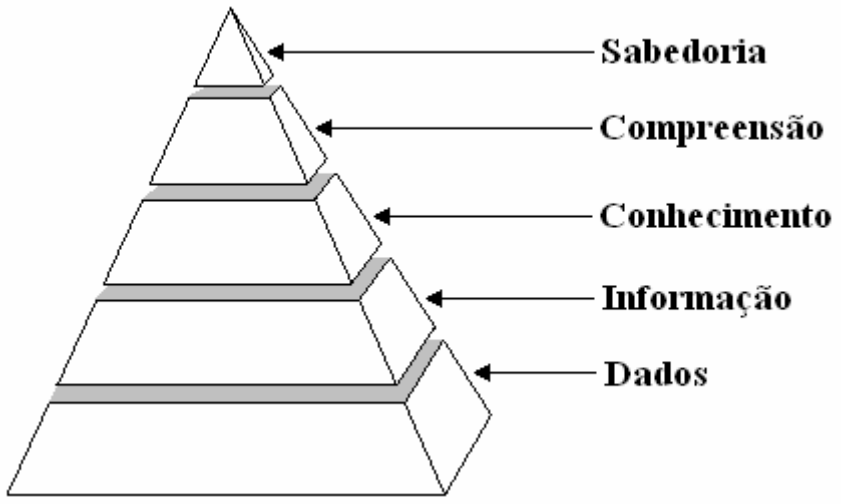

Figura 2.1: Hierarquia de conteúdo da mente humana

Dados: são símbolos que representam as propriedades de objetos, eventos, fenômenos ou do ambiente em um determinado contexto. São entidades em uma forma bruta, originadas de simples observação, desprovidas de significado e sem relação com outras entidades.

Informação: são os dados processados de forma que seja possível agregar-lhes algum significado. A informação é descritiva e permite relacionar passado e presente através da identificação de alguma maneira de relacionamento. Após o processamento, a informação gerada deve fornecer respostas a questões do tipo: “Quem”, “O que”, “Onde”, “Quando” e “Quantos”; de forma a permitir que a informação possa ter alguma utilidade prática.

Conhecimento: é obtido através da coleta de informações apropriadas com o objetivo de serem úteis. O conhecimento representa um padrão que fornece uma base para predição do futuro com certo grau de confiança, baseado na informação sobre o passado e o presente. O conhecimento também pode ser imaginado como um conjunto de conceitos obtidos a partir de dados e informações a fim de responder a questões que começam com “Como”. A aquisição de conhecimento é feita através de um processo denominado aprendizado. O processo de aprender como é o funcionamento de um sistema, a fim de criar um modelo que simule tal sistema, é um dos pilares da evolução científica nos tempos atuais.

Compreensão: permite a geração de conhecimento novo a partir do conhecimento prévio. A compreensão permite responder questões que começam com “Por que”. A existência de uma categoria de compreensão não é um consenso entre os pesquisadores da área. Muitos autores consideram que a compreensão é um processo que permite a transição entre cada categoria e a categoria imediatamente superior na hierarquia. A figura 2.2 apresenta graficamente esta outra concepção.

Sabedoria: A sabedoria é considerada o entendimento da dinâmica do sistema como um todo e de seus princípios de funcionamento. Enquanto a inteligência é a habilidade de aumentar a eficiência, o entendimento sistêmico obtido através da sabedoria permite aumentar a efetividade.

Embora haja algumas críticas sobre esta hierarquia de conhecimento da mente humana, ela ainda é amplamente aceita e difundida entre muitos teóricos da área de computação (FRICKÉ, 2009).

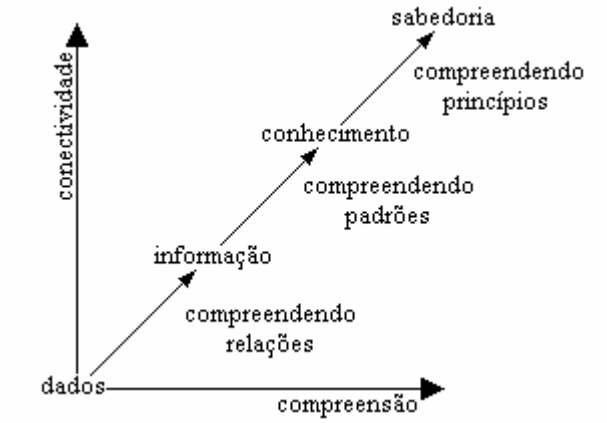

Figura 2.2: Relação dos dados com a compreensão e a conectividade

A figura 2.2 permite supor que existe um processo para transformar um nível no nível superior. Já a partir da pirâmide apresentada na figura 2.1, que apresenta a organização hierárquica, é possível supor que um nível não existe sem o nível inferior. Esta estrutura em forma piramidal mostra que à medida que aumenta a quantidade de dados e informação, existe uma tendência que a quantidade de conhecimento também aumente. Tal conhecimento pode aumentar além de um limiar gerenciável e

manipulável, de forma a perderem-se seus relacionamentos estruturais, o que implica na redução de sua utilidade.

Dentro desta hierarquia de conteúdo apresentada, o presente trabalho está inserido entre o segundo e o terceiro nível, ou seja, na transformação de informação em conhecimento através da compreensão de padrões expressos nos dados.

2.3 Inteligência artificial e aprendizado de máquina

Desde que os computadores foram criados, nutriu-se o desejo de fazê-los exibir um comportamento inteligente. Logo após 1950, com o advento dos computadores programáveis, foram desenvolvidos os primeiros programas com a intenção de imitar o processo de pensamento humano. O crescente interesse por esta área culminou, em 1956, na criação do termo inteligência artificial para denotar um novo tópico de pesquisa na área de computação preocupado em simular a inteligência humana (RUSSELL e NORVIG, 1995).

Segundo Luger e Stubblefield (1998), a inteligência artificial (IA) pode ser conceituada como o ramo da ciência da computação que se preocupa com a automatização do comportamento inteligente. Atualmente, a área de IA engloba uma ampla variedade de sub-campos, dentre eles, uma das mais férteis áreas de pesquisa é o aprendizado de máquina, que se preocupa com a construção de sistemas de alto desempenho capazes de aprender através da experiência e obter conhecimento a partir de dados.

Segundo Konar (2000), há quatro diferentes classes de aprendizado de máquina: aprendizado supervisionado, aprendizado não supervisionado, aprendizado por reforço e aprendizado por programação em lógica indutiva.

O aprendizado supervisionado refere-se a uma classe de algoritmos que visam aprender um relacionamento entre entradas e saída. Este relacionamento geralmente descreve uma dependência ou função $f_o(x)$ presente de forma implícita em um conjunto de treinamento $D = \{[x(i), y(i)] \in \Re \text{ x } \Re, i = 1, ..., l\}$ consistindo de l pares (x_1, y_1), (x_2, y_2), ..., (x_l, y_l). As entradas x consistem em um vetor n-dimensional onde $x \in \Re$, e as saídas y consistem em um vetor 1-dimensional onde $y \in \Re$. Dependendo do valor a ser predito o aprendizado supervisionado pode ser de dois tipos: regressão quando os valores de saída são contínuos, e classificação quando os valores de saída são discretos (KECMAN, 2001). Durante o processo de treinamento, as amostras são sucessivamente submetidas ao algoritmo de aprendizado. Para cada amostra de entrada, o algoritmo tenta predizer a saída. A saída predita pelo algoritmo é comparada com a saída real, a diferença entre elas é utilizada para reajustar os parâmetros do modelo. Desta forma, o algoritmo iterativamente ajusta seus parâmetros para criar um modelo que faça um mapeamento das entradas para a saída. Após o processo de aprendizado supervisionado, é criado um modelo que pode ser utilizado para simular o conhecimento do especialista do domínio.

Já no aprendizado não supervisionado, o objetivo é agrupar l amostras em k grupos, e k é determinado pelos dados e geralmente não é conhecido antes da aplicação do algoritmo. O conjunto de treinamento é formado por $D = \{[x(i)] \in \Re, i = 1, ..., l\}$ consistindo de l amostras (x_1), (x_2), ..., (x_l). As entradas x consistem em um vetor n-dimensional onde $x \in \Re$. Os algoritmos são treinados para descobrir características estatisticamente salientes das amostras e aprender a alocá-los nos diferentes grupos de

acordo com sua similaridade. Desta forma, a idéia é agrupar as amostras de forma a minimizar a distância intra-grupos e maximizar a distância inter-grupos.

No aprendizado por reforço, o sistema de aprendizado não sabe qual a saída desejada para cada conjunto de entradas, sabendo apenas se a saída obtida está correta ou não. Porém, o sistema recebe apenas uma punição ou recompensa do ambiente para cada saída predita. Pelo fato do algoritmo receber uma resposta para cada uma de suas ações, alguns autores consideram o aprendizado por reforço um caso especial de aprendizado supervisionado (MITRA e ACHARYA, 2003).

No aprendizado por programação em lógica indutiva (PLI) há uma combinação do aprendizado de máquina indutivo com a programação em lógica. De maneira formal, a PLI pode ser definida da seguinte forma: tem-se C como um conhecimento prévio do domínio expresso na forma de um conjunto de predicados, exemplos positivos E^+ e exemplos negativos E^-. O objetivo da PLI é encontrar uma forma de predicado lógico H, tal que todos os exemplos em E^+ possam ser logicamente derivados de $C \wedge H$, e nenhum exemplo em E^- possa ser logicamente derivados de $C \wedge H$. A diferença marcante entre a PLI e o aprendizado indutivo convencional é a utilização do conhecimento prévio do domínio.

Dentre estas abordagens de aprendizado apresentadas, o foco deste trabalho foi definido sobre o aprendizado supervisionado.

2.3.1 Aplicações de aprendizado de máquina

Devido aos avanços da tecnologia, que têm conduzido a uma constante evolução das tecnologias de geração, coleta e armazenamento de dados, vivenciamos uma sobrecarga de informação na maioria das áreas de conhecimento humano. Há diversos domínios do conhecimento humano em que grandes volumes de dados são coletados e armazenados. Alguns exemplos incluem: bioinformática, telecomunicações, astronomia, climatologia, computação, economia e geologia (MITRA e ACHARYA, 2003). Pelo fato de as técnicas de análise destes dados não evoluírem tão rapidamente quanto as técnicas de coleta e armazenamento, surge uma importante demanda por ferramentas automáticas para analisar estes dados em tempo aceitável. Tais ferramentas são o foco de pesquisa da área de aprendizado de máquina.

Além das aplicações na área científica, a exploração de grandes bancos de dados comerciais também representa um claro interesse econômico. Isto ocorre principalmente porque a maioria das empresas utiliza computadores para interagir com seus clientes. A redução constante do custo de armazenamento contribuiu para que as empresas passassem a armazenar em banco de dados um histórico das interações com seus clientes, criando-se bancos de dados cada vez maiores com um histórico da atividade da empresa. Este histórico torna-se uma "mina" com valiosas informações sobre a atividade da empresa e pode ser explorado a fim de servir como um poderoso suporte ao processo de tomada de decisão, permitindo a descoberta de padrões de perfis e tendências escondidas no banco de dados (CAMARGO e ENGEL, 2002). Quanto mais a empresa aprende sobre seus dados, maior é o valor agregado deste aprendizado, e maiores são as possibilidades de converter o aprendizado em lucro. Esta idéia é representada na figura 2.3.

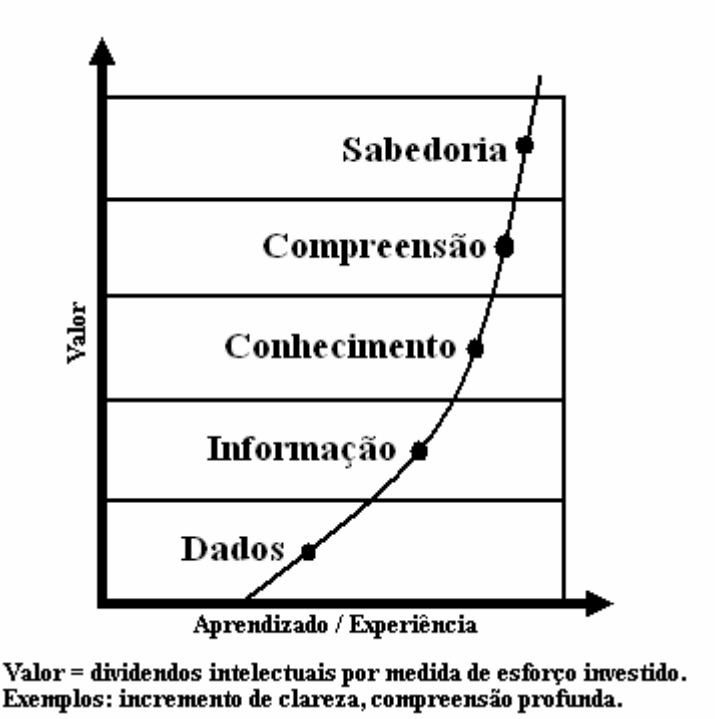

Figura 2.3: Relação entre aprendizado e seu valor

A necessidade da indústria da informação de aplicar o aprendizado de máquina em grandes bancos de dados para obter informação e conhecimento criou uma nova área de pesquisa. Esta nova área, que recebeu o nome de Descoberta de Conhecimento em Bancos de Dados, é discutida a seguir.

2.4 Descoberta de Conhecimento em Banco de Dados

O termo Descoberta de Conhecimento em Bancos de Dados (DCBD) foi introduzido no final da década de 1980 para se referir ao amplo processo de encontrar conhecimento a partir de dados e enfatizar o mais alto nível de aplicações particulares de mineração de dados (FAYYAD et al., 1996). Adicionalmente, outros conceitos de DCBD já fazem uma referência explícita ao tamanho dos bancos de dados. Conforme Sarker et al. (2002), DCBD é o processo de modelar abstrações de grandes bancos de dados através da pesquisa por padrões válidos, novos e não triviais sobre um modelo abstrato.

A necessidade de conhecimento multidisciplinar é evidente no complexo processo de DCBD. A fim de atingir seus objetivos, o campo de DCBD reúne pesquisadores de diversas áreas de pesquisa, tais como: banco de dados, aprendizado de máquina, reconhecimento de padrões, estatística, teoria da informação, inteligência artificial, raciocínio sobre incerteza, aquisição de conhecimento para sistemas especialistas, visualização de dados e computação de alto desempenho.

A literatura apresenta vários modelos que definem os passos básicos para o processo de DCBD sob diversos níveis de abstração diferentes, variando entre modelos altamente abstratos e altamente detalhados. A figura 2.4 apresenta o modelo clássico de DCBD que, sob um mais alto nível de abstração, pode ser dividido em três etapas distintas (FAYYAD et al., 1996):

- Pré-processamento: que inclui atividades como seleção, limpeza e transformação dos dados para torná-los aptos a serem utilizados na etapa de mineração de dados.
- Mineração de dados (MD): que é o núcleo do processo de DCBD, onde efetivamente são aplicados os algoritmos para extração de padrões.
- Pós-processamento: que envolve a interpretação e avaliação dos padrões visando a obtenção de conhecimento.

O termo mineração de dados é frequentemente usado como sinônimo para o processo de DCBD ainda que ele se refirae apenas a um passo dentro do amplo processo de DCBD. MD refere-se efetivamente ao processo de aplicação do algoritmo de descoberta nos dados (SARKER et al., 2002).

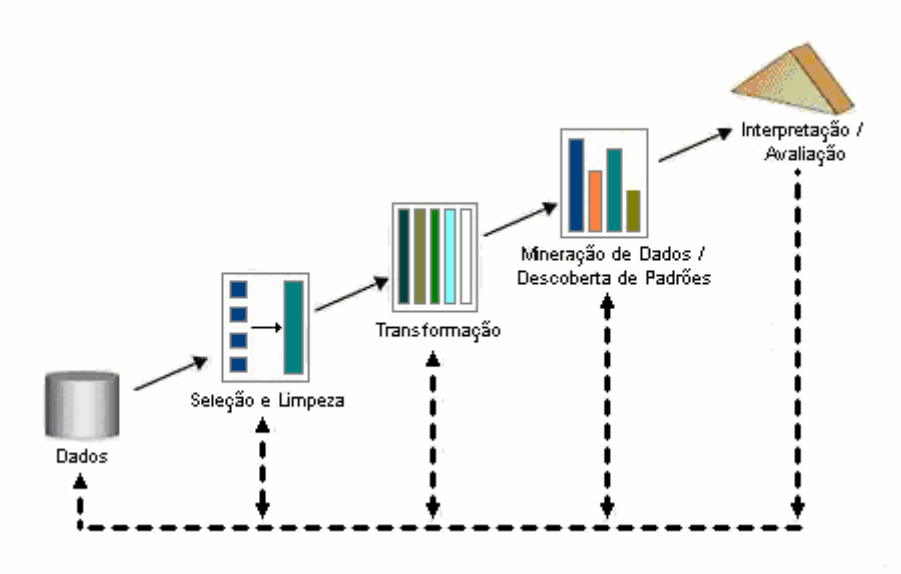

Figura 2.4: O modelo clássico do processo de DCBD

A DCBD é um processo de melhoria incremental, onde são realizados os seguintes passos: selecionar os dados, preparar os dados, construir o modelo, avaliar os resultados, preparar novamente os dados de forma a melhorar o modelo, e assim sucessivamente até que sejam obtidos resultados relevantes.

A seguir passam a ser detalhadas as três etapas do processo de DCBD.

2.4.1 Pré-Processamento

A fase de pré-processamento corresponde à preparação dos dados para o processo de MD. O pré-processamento consome entre 60 e 90% do tempo total do processo de DCBD. Uma fase de pré-processamento bem realizada contribui entre 75 e 90% do sucesso de um projeto de mineração, enquanto a não realização desta fase pode ser 100% responsável pelo insucesso do projeto (YE, 2003).

Isto ocorre porque a maioria das técnicas de mineração de dados requer que os dados estejam consolidados em uma única tabela, limpos, consistentes e completos. Porém, esta situação raramente ocorre em bancos de dados reais. Desta forma, os dados devem ser pré-processados para melhorar a eficiência dos algoritmos de MD e, consequentemente, de todo o processo de DCBD. A forma de coleta e preparação dos

dados, assim como as decisões tomadas nesta etapa são críticas para a qualidade dos resultados obtidos nas fases subseqüentes (MYATT, 2007).

Tarefas básicas na fase de pré-processamento incluem seleção de dados, integração, limpeza, redução de ruído, detecção de *outliers*, transformação, e redução de dimensionalidade. O algoritmo que será utilizado na fase de MD influi fortemente nas tarefas que devem ser executadas no pré-processamento. Como exemplo, pode ser citado que a maioria das redes neurais necessita que todos os dados sejam numéricos, sendo a tarefa de transformação responsável por converter dados não numéricos em numéricos. Por outro lado, árvores de decisão usualmente necessitam que todos os dados sejam categóricos, de forma que valores numéricos devem ser representados de maneira categórica.

2.4.1.1 Seleção

A primeira tarefa a ser realizada na fase de pré-processamento é a seleção dos dados. Supõe-se que os dados já foram coletados de alguma forma e geralmente o responsável pelo processo de DCBD não tem nenhuma influência sobre esta coleta.

Havendo uma quantidade muito grande de dados disponíveis, faz-se necessária a seleção de quais características e quais registros deverão ser utilizados no processo de MD.

Dentre as atividades realizadas nesta fase estão (YE, 2003):

- Seleção de características de entrada relevantes: nesta atividade devem ser selecionadas as características que, segundo o conhecimento do especialista do domínio, contém dados potenciais para o processo de MD. Quando houver poucas características disponíveis, esta tarefa tende a ser irrelevante, Porém, quando a quantidade de características for muito alta, esta tarefa é de importância crucial para a criação de bons modelos. Algumas vezes, não há especialista do domínio ou esta pessoa não tem um conhecimento profundo do problema a ponto de conseguir saber quais características são relevantes ou não. Neste caso, faz-se necessária a utilização de técnicas de redução de dimensionalidade, que são abordadas posteriormente neste capítulo, na seção 2.4.1.6.
- Evitar a seleção de características de entrada redundantes: deve-se evitar a utilização de características que tenham um alto índice de correlação entre si. Isto tende a dificultar o aprendizado durante a fase de MD. O acréscimo de características redundantes somente é recomendado se o nível de ruído destas características for alto, de forma que a redundância poderá compensar o ruído.
- Selecionar registros aleatoriamente: a seleção de registros deve ser feita de maneira aleatória de forma a evitar que os registros selecionados representem alguma tendência pontual em detrimento de tendências globais. Outra forma de seleção de registros também é a seleção estratificada de registros de acordo com os princípios estatísticos.
- Assegurar que os registros representam a realidade: poucos registros, que representam uma parte específica do todo, podem simplesmente representar uma tendência local expressa através dos valores de suas características, e não uma tendência global. A solução para este problema é a utilização de

mais dados. Além disso, deve ter-se sempre em mente que a maior fonte de ruído é a insuficiência de dados. Os problemas gerados por uma quantidade pequena de dados em relação à uma grande quantidade de características são discutidos em Bellman (1961).

2.4.1.2 Integração

É possível antes da execução do processo de MD seja necessária a integração de dados oriundos de várias fontes diferentes em um único arquivo. Uma grande quantidade de problemas pode surgir durante esta integração. Faz-se necessária a integração dos esquemas das diferentes fontes onde podem surgir problemas como:

- Identificação de entidades.
- Redundâncias entre características.
- Duplicação de características.
- Inconsistências na forma de conflitos de valores de dados para uma mesma característica.

Estes problemas vão além do escopo deste trabalho e são foco de uma importante e ampla área de pesquisa chamada de integração de esquemas de banco de dados.

A execução bem sucedida do processo de integração tem influência determinante nos resultados do processo de DCBD.

2.4.1.3 Limpeza

Dados do mundo real tendem a ser incompletos, ruidosos e inconsistentes. A limpeza dos dados visa reparar estes problemas de qualidade. Este objetivo é atingido por meio do preenchimento de valores inexistentes e correção de inconsistências nos dados.

Completar valores omitidos consiste em um sério problema, pois qualquer inclusão, alteração ou exclusão de dados estará modificando, talvez erroneamente, os dados de entrada e, consequentemente, alterando o resultado do processo de MD. Porém esta pode ser a única alternativa para algoritmos que não conseguem lidar com valores omitidos. Dentre as técnicas mais populares para reparar dados incompletos podem ser citadas (SOUMEN, 2009):

- Ignorar o registro completo.
- Preencher os valores omitidos manualmente.
- Usar uma constante global, tal como “?”, para preencher os valores omitidos.
- Usar o valor médio do atributo para preencher o valor omitido.
- Usar o valor médio do atributo, considerando somente os registros pertencentes à mesma classe, para preencher o valor omitido.
- Usar o valor mais provável para preencher o valor omitido.

2.4.1.4 Redução de ruído e detecção de outliers

Ruído é um erro ou variância aleatória na medição de uma variável. As razões mais comuns para existência de ruído são: problemas ocorridos durante as fases de coleta,

entrada ou transmissão de dados; falhas em instrumentos e limitações de tecnologia; inconsistências nas convenções de nomenclatura de características e existência de registros duplicados (SYMEONIDIS e MITKAS, 2005). Para realizar a remoção de ruído geralmente são utilizadas técnicas de suavização. Tais técnicas incluem *binning* e regressão.

- *Binning*: O processo de *binning* é executado a partir de um conjunto ordenado dos valores assumidos por uma variável. Estes valores são divididos em grupos com o mesmo número de elementos. A partir daí os valores originais são substituídos pela média, mediana ou valores mínimo e máximo de cada grupo.
- Regressão linear: por meio desta técnica os valores de uma característica são suavizados a partir de uma combinação linear dos valores de outra característica. Também pode ser utilizada a regressão linear múltipla para suavizar o valor de uma característica a partir do valor de diversas outras características.

Outliers são valores extremos que estão fora dos limites de um intervalo de dados ou estão destoando da tendência dos valores de um determinado atributo. Os *outliers* podem ser originados de erros no processo de entrada dos dados, sendo chamados de *outliers* inválidos, ou também podem representar dados válidos, sendo chamados de *outliers* válidos. Para *outliers* inválidos, deve ser feito um esforço para descobrir seu valor correto. Caso isto não seja possível, este valor pode ser tratado como um valor omitido. *Outliers* válidos não devem ser descartados, pois representam o comportamento real do sistema. A existência de *outliers*, sejam eles válidos ou inválidos, pode ser um fator que prejudica o desempenho dos algoritmos de mineração de dados tornando os resultados instáveis.

Histogramas ou *scatter plots* bidimensionais são técnicas utilizadas para detecção de *outliers*, porém a abordagem mais simples é a definição de limites aceitáveis para o valor da característica. Além destas técnicas, a normalização, que é abordada na seção 2.4.1.5, também pode diminuir os problemas causados pela existência de *outliers*.

2.4.1.5 Transformação

Os dados que serão minerados geralmente não estão em uma forma adequada para maximizar o desempenho dos algoritmos de mineração. Faz-se necessária então a realização do processo de transformação, onde os dados são transformados ou consolidados para as fases seguintes. Dentre as técnicas mais populares utilizadas nesta fase estão (HAN e KAMBER, 2001):

- Agregação: em alguns casos, pode ser necessária a utilização de alguma variável que não está explicitamente representada, mas que pode ser derivada a partir de outras variáveis a partir de qualquer operação matemática. Esta técnica é tipicamente utilizada quando o processo de mineração for executado em múltiplas granularidades. Outro exemplo pode ser o campo de data, cujo valor absoluto pode ser completamente sem utilidade, porém a utilização do dia da semana, que pode ser obtida através do campo data, pode ser de grande utilidade.

- Generalização: os valores originais dos dados são substituídos por conceitos com um significado dentro de uma hierarquia de conceitos. Esta técnica permite a mineração em vários níveis de abstração.
- Normalização: onde os valores do atributo são normalizados para ficarem dentro de um intervalo específico de valores, tal como de -1 e 1, ou de 0 e 1.

A normalização pode ser executada através de três técnicas distintas :

- Normalização min-max ou escalonamento: executa uma transformação linear sobre os dados originais, com base nos valores máximo e mínimo de um dado atributo *A*. Esta normalização mapeia um valor *v* de *A* para um valor *v*' no intervalo [$novo\_min_A$, $novo\_max_A$] através da seguinte fórmula:

  $$v' = ((v - min_A) / (max_A - min_A))(novo\_max_A - novo\_min_A) + novo\_min_A$$

  onde min_A e max_A são respectivamente os valores mínimo e máximo do atributo *A*. Exemplo: supondo-se que valores mínimos e máximos para a característica salário são respectivamente R$1.000 e R$9.000. Pretende-se mapear esta característica para o intervalo [0,1]. Pela normalização, um salário de R$6.200 é transformado da seguinte forma:

  $$\frac{6.200 - 1.000}{9.000 - 1.000}(1 - 0) + 0 = 0.65$$

- Normalização z-*score*: é também conhecida por normalização de média zero. Nesta técnica, os valores de um dado atributo *A*, são normalizados com base em sua média e desvio padrão. Um valor *v* de *A* é normalizado através da seguinte fórmula:

  $$v' = (v - mean_A) / std\_dev_A$$

  onde $mean_A$ e std_dev_A são respectivamente a média e o desvio padrão dos valores do atributo *A*. Esta técnica, além de ser uma elegante forma de tratamento de *outliers***,** também permite que valores omitidos sejam simplesmente preenchidos com "0" atribuindo a estas omissões a média dos valores do atributo. Quando os valores máximo e mínimo do atributo *A* forem desconhecidos esta técnica é muito útil. Exemplo: supondo-se que a média e o desvio padrão dos valores da característica salário são respectivamente R$4.200 e R$1.000. Com a normalização z-score, o salário de R$ 6.200 é transformado da seguinte forma:

  $$\frac{6.200 - 4.200}{1.000} = 1.25$$

- Normalização em escala decimal: os dados são transformados movendo-se o ponto decimal para todos os valores do atributo. Um valor *v* é normalizado para *v*' através da seguinte fórmula:

  $$v' = v / 10^j$$

  onde *j* é o menor número inteiro tal que $Max(|v'|) < 1$. Exemplo: supondo-se que os valores máximo e mínimo da característica salário são respectivamente R$ 9.000 e R$1.000. Para normalizar por uma escala decimal, poderia se dividir

estes valores por 10.000, ou seja, j = 4. Com a normalização em escala decimal, um salário de R$6.200 seria transformado da seguinte forma:

$$\frac{6.200}{10^4} = 0,62$$

Além das atividades abordadas anteriormente, também há outra atividade de extrema importância realizada nesta fase que é a conversão de tipos de dados. São poucos os algoritmos que podem manipular tanto dados categóricos quanto dados numéricos. O caso mais comum é que o algoritmo tenha habilidade de manipular somente um destes tipos de dado. Algoritmos orientados a números necessitam que os dados categóricos sejam transformados para numéricos, em contrapartida algoritmos orientados a categorias necessitam que dados numéricos sejam transformados em categóricos. Apesar da perda de informação implícita à realização desta atividade, ela possibilita a utilização, no processo de MD, de dados que seriam descartados por serem de um tipo que o algoritmo não consegue tratar.

A transformação de dados numéricos para categorias geralmente é realizada através de técnicas de *binning*, sendo que o valor resultante é considerado de forma categórica. Técnicas de *binning* já foram abordadas anteriormente neste capítulo. Já a transformação de dados categóricos para numéricos pode ser feita de duas formas:

- Codificação direta: faz-se mediante a atribuição de um valor numérico para cada categoria. Quando houver uma relação de ordenação entre as categorias, os valores numéricos devem também representar esta ordenação, de forma que o valor 0 represente a primeira categoria, o valor 1 represente a última, e as categorias intermediária assumam valores dentro deste intervalo.
- Codificação 1 para *n* códigos: se não houver uma relação de ordenação entre as categorias, supondo-se que a característica possua *n* categorias possíveis, é mais usual criar *n* características, uma relativa a cada categoria, atribuindo o valor 0 quando a amostra não pertence a *n*-ésima categoria ou 1, caso contrário.

2.4.1.6 Redução de Dimensionalidade

Dados utilizados para MD podem conter centenas ou até milhares de características, sendo muitas delas irrelevantes ou redundantes. Apesar da maioria dos algoritmos de MD conseguirem identificar as características mais e menos relevantes durante o processo de aprendizado, a execução da redução de dimensionalidade durante a fase de pré-processamento geralmente melhora os resultados obtidos pelos algoritmos de MD. Além disso, uma menor quantidade de características irrelevantes ou redundantes conduz a um menor gasto de tempo pelo algoritmo. Uma discussão mais aprofundada sobre redução de dimensionalidade é realizada no capítulo 4.

2.4.2 Mineração de Dados

A entrada na fase de mineração de dados pressupõe que o pré-processamento dos dados foi realizado com sucesso e tem-se razoável nível de confiabilidade em relação à qualidade dos dados pré-processados. Este pressuposto é um fator fundamental para se atingir um bom desempenho durante a fase de mineração de dados.

A fase de mineração de dados, também chamada de fase de modelagem, pode ser definida como o processo de descobrir novas correlações, padrões e tendências

significativos através da mineração de grandes quantidades de dados usando técnicas estatísticas, de aprendizado de máquina, de inteligência artificial, de bancos de dados e de visualização de dados (SUMATHI e SIVANANDAM, 2006). As técnicas aplicadas nesta fase geralmente estão implementadas na forma de algoritmos bem conhecidos. Um algoritmo de mineração de dados é um procedimento bem definido que a partir de dados de entrada produz saídas na forma de modelos ou padrões (HAND et al., 2001).

Para atingir seus objetivos, um algoritmo de mineração de dados geralmente possui quatro componentes básicos:

- Estrutura de modelos ou de padrões que determina o esqueleto básico ou as formas funcionais que são procuradas nos dados.
- Uma função de avaliação que irá julgar a qualidade do modelo criado pelo algoritmo.
- Métodos de otimização e pesquisa cujos objetivos são otimizar a função de avaliação e pesquisar diferentes estruturas de modelos ou de padrões.
- Uma estratégia de gerenciamento de dados para permitir uma manipulação eficiente dos dados durante a busca ou otimização.

Conforme Berry e Linoff (2004), as técnicas de mineração de dados podem descobrir diferentes tipos de conhecimento a partir da execução de um conjunto limitado de tarefas, que podem ser divididas em seis classes distintas: classificação, regressão, predição, regras de associação, agrupamento por similaridade e descrição.

Para descobrir estas diversas formas de conhecimento podem ser aplicadas diferentes técnicas para execução das tarefas de mineração, tais como: árvores de decisão, redes neurais, raciocínio baseado em memória e algoritmos genéticos.

2.4.2.1 Classificação

A classificação é uma tarefa de aprendizado supervisionado muito comum em mineração de dados, além de ser uma tarefa característica da inteligência humana. Esta tarefa pode ser dividida em dois passos. No primeiro passo, uma parte do conjunto de dados, chamado conjunto de treinamento, é utilizado para construir um modelo que mapeie os dados de treinamento em um conjunto de classes previamente definido pelo especialista do domínio. Justamente por ser uma tarefa de aprendizado supervisionado, supõe-se que para cada amostra do conjunto de treinamento, a sua respectiva classe é conhecida. No segundo passo, o modelo é usado para analisar dados ainda não conhecidos, que podem constituir um conjunto de teste, e alocá-los a uma das classes.

Os modelos criados podem representar o conhecimento obtido de várias formas, entre elas: regras na forma SE-ENTÃO, árvores de decisão ou outros formalismos matemáticos. De uma maneira mais formal, tem-se um conjunto de objetos $O = \{o_1, o_2, ..., o_n\}$ e um conjunto de classes $C = \{c_1, c_2, ..., c_m\}$. Um modelo de classificação tem como objetivo aproximar uma função f, tal que $f(o_i) = c_j$.

Árvores de decisão, redes neurais, redes bayesianas e raciocínio baseado em memória são técnicas que se ajustam muito bem à classificação (SYMEONIDIS e MITKAS, 2005).

2.4.2.2 Regressão

O processo de regressão é semelhante ao processo de classificação, a principal diferença entre ambos é que a classificação lida com valores discretos enquanto a regressão, com valores contínuos. Como conseqüência disso temos que, através do processo de regressão, é possível ordenar registros individualmente. Por exemplo, se pelo processo de classificação classificamos registros como 0 ou 1, pelo processo de regressão é possível classificarmos registros com qualquer valor real entre 0 e 1. Redes neurais se ajustam muito bem a tarefas de regressão.

2.4.2.3 Predição

O processo de predição também é semelhante aos processos anteriores exceto pelo fato de que os registros possuem dados temporais e são classificados de acordo com alguma predição de comportamento futuro ou predição de valor futuro. Tanto classificação como regressão podem ser adaptadas para uso em predição através dos exemplos de treinamento onde os valores passados das variáveis a serem preditas são conhecidos, de acordo com os dados históricos para estes exemplos. Os dados históricos são usados para construir um modelo que explica o comportamento corrente observado. A técnica de análise da cesta de compras, usada para descobrir que itens provavelmente serão comprados juntos, pode ser adaptada ao modelo de que compras futuras ou ações tendem a ser tomadas de acordo com os dados correntes. As técnicas de análise da cesta de compras, raciocínio baseado em memória, árvores de decisão e redes neurais podem ser utilizadas no processo de predição.

2.4.2.4 Regras de associação

A extração de regras de associação é o processo de encontrar padrões, associações, correlações ou estruturas causais freqüentes entre conjuntos de itens ou objetos em bancos de dados. Esta tarefa é frequentemente aplicada a bancos de dados de transações onde se deseja extrair regras denotando que a ocorrência de um subconjunto de itens implica a ocorrência de outro subconjunto, disjunto do primeiro, na mesma transação.

De uma maneira formal, tem-se $I = \{i_1, i_2, ..., i_n\}$ sendo um conjunto de objetos chamados itens. Tem-se $D = \{T_1, T_2, ..., T_m\}$ sendo um conjunto de transações, onde cada transação T é uma coleção de itens, com $T \subseteq I$. Tem-se I_a e I_b sendo conjuntos de itens. Uma regra de associação é um relacionamento na forma $I_a \Rightarrow I_b$, onde $I_a \subseteq I$, $I_b \subseteq I$ e $I_a \cap I_b = \varnothing$. A regra de associação r tem um suporte s, se s_r é o percentual de transações em D que contém $I_a \cup I_b$, ou seja, s_r é a probabilidade $P(I_a \cup I_b)$. Uma regra tem confiança c sobre D, se c é o percentual de transações em D que contém I_a e I_b, ou seja, c é a probabilidade condicional $P(I_a \mid I_b)$.

Para a geração de regras de associação o algoritmo mais utilizado é o *Apriori*, porém já foram propostas diversas outras abordagens derivadas deste algoritmo para executar esta tarefa (CAMARGO e ENGEL, 2002).

Esta tarefa também é frequentemente referenciada na literatura como análise de cesta de compras e agrupamento por afinidade.

2.4.2.5 Agrupamento por similaridade ou clusterização

O processo de agrupamento por similaridade consiste em dividir uma população heterogênea em grupos de objetos similares. Um grupo é um conjunto de elementos

desta população com alto nível de similaridade entre si, e baixo nível de similaridade com elementos de outros grupos. Desta forma, o objetivo principal desta tarefa de mineração de dados é atingir duas métricas: maximizar a similaridade entre elementos intra-grupo e minimizar de similaridade entre elementos inter-grupos. Estes grupos não são pré-definidos e também não há exemplos assim como ocorre no processo de classificação. Agrupamento por similaridade pode muitas vezes ser utilizado como preparação para alguma outra forma de mineração de dados.

De uma maneira mais formal, a tarefa de agrupamento consiste em, dado um número inteiro k, encontrar uma forma de particionar os dados em k grupos c_1, c_2, ..., c_k que otimize um dado critério de particionamento.

Para a tarefa de agrupamento, um dos algoritmos mais utilizados é o *k-means* (XU e WUNSCH, 2009).

2.4.2.6 Descrição

O processo de descrição tem como propósito simplesmente descrever os padrões e tendências implícitas a algum conjunto de dados a fim de aumentar a nossa compreensão sobre sistemas, fenômenos ou processos. Um bom processo de descrição de um padrão ou tendência freqüentemente irá sugerir uma explicação para tal padrão ou tendência.

2.4.3 Avaliação da fase de mineração de dados

Após a aplicação dos algoritmos de mineração de dados sobre os dados de treinamento, o passo seguinte é a avaliação do modelo criado a fim de verificar sua qualidade.

A avaliação do modelo é uma atividade complexa que exige formas sistemáticas de trabalho. Os algoritmos de mineração de dados frequentemente exigem a configuração de um conjunto de parâmetros, os quais exercem uma influência determinante nos resultados obtidos. Diferentes valores dos parâmetros geram diferentes modelos. Além disso, é necessária a aplicação de técnicas que possam avaliar o desempenho preditivo do modelo em dados que não foram previamente vistos (OLSON e DELEN, 2008).

2.4.3.1 Particionamento dos dados

Para avaliar como os modelos irão se comportar na predição de dados não vistos, geralmente o conjunto de dados disponível é dividido em duas partes, sendo uma para treinar o modelo e outra para avaliá-lo. Dentre as formas de particionamento, as principais são as seguintes (BISHOP, 1995):

Holdout

Quando há uma grande quantidade de dados disponível para o processo de mineração a avaliação é teoricamente simples. Neste caso, geralmente é utilizada a técnica chamada *holdout* onde os dados são divididos aleatoriamente em duas partições independentes e sem sobreposição: uma de treinamento e outra de teste. A partição de treinamento é usada para construir o modelo, e a partição de teste é utilizada para avaliar a capacidade de generalização do modelo. Em relação ao tamanho das partições, geralmente a partição de treinamento contém 75% dos dados; e a de teste, 25%. Uma variação da técnica *holdout* é a subamostragem aleatória, onde os conjuntos de treinamento e teste são particionados de maneira aleatória, sendo o procedimento

repetido *k* vezes. A exatidão do método é estimada pela média da exatidão obtida em todas das *k* repetições.

Validação Cruzada

Quando há uma quantidade limitada de dados, são mais recomendados os métodos de validação cruzada. Dada a limitação da quantidade de dados, todas as amostras são utilizadas para teste e para treinamento, mas não ao mesmo tempo. As técnicas mais utilizadas são *n-fold* e *leave-one-out* (HASTIE et al., 2001).

Na técnica *n-fold*, os dados são divididos em *n* partições de tamanhos iguais ou similares e o procedimento é repetido *n* vezes. Em cada repetição, a partição *n* é utilizada para teste e as demais partições são utilizadas para treinamento. A divisão dos dados em 10 partições tem se tornado um procedimento padrão visto que, testes em vários bancos de dados, com diferentes técnicas de mineração, tem mostrado que 10 seria um número ideal para obtenção de uma melhor estimativa de erro (WITTEN e FRANK, 2005).

A técnica chamada *leave-one-out* ou *jackknifing* pode ser considerada um caso específico da *n-fold*, onde o valor de *n* é igual ao número de amostras do banco de dados. Desta forma, em cada iteração, são utilizadas *n*-1 amostras para treinamento e 1 amostra para teste. A exatidão do modelo é calculada medindo a exatidão na predição da amostra de teste. A exatidão final do modelo é dada pela média da exatidão de todos os *n* experimentos. O ponto negativo desta técnica de avaliação é seu custo computacional, visto que o processo de treinamento será realizado *n* vezes, cada uma delas utilizando *n*-1 amostras. Porém, este procedimento tem grande utilidade para pequenos bancos de dados.

Outra técnica de validação cruzada é o *bootstrap*, que é baseada no procedimento estatístico de amostragem com substituição. Segundo esta técnica, as amostras do banco são selecionadas por amostragem para fazerem parte ou do conjunto de treinamento ou de teste. Este processo é repetido várias vezes. Muitos especialistas consideram esta a melhor técnica de avaliação de modelos, apesar de seu alto custo computacional (RUD, 2001).

Adicionalmente, algumas vezes, pode ser necessária uma terceira partição de dados que auxiliaria o refinamento dos modelos criados. Esta partição se chama de partição de validação e seria utilizada em um passo intermediário entre o treinamento e o teste do modelo. Dados presentes na partição de validação também são independentes e sem sobreposição em relação às outras partições.

2.4.3.2 Medição do erro

Com a aplicação das técnicas de particionamento dos dados passa a ser possível medir o erro de predição dos modelos criados. Existem diversas métricas para avaliar a qualidade preditiva de um modelo, que passam a ser descritas a seguir.

Erro de regressão

A tarefa de regressão visa predizer um valor numérico contínuo para uma variável dependente. Para avaliar o erro de regressão, deve ser calculada a diferença entre o valor predito pelo modelo e o valor real desta variável para cada uma das amostras. O erro médio do modelo é então calculado pela média de erro para todas as amostras (THEODORIDIS e KOUTROUMBAS, 2003). As duas medidas mais utilizadas para

avaliar o erro de regressão são: o erro quadrado médio (EQM) e o desvio absoluto médio (DAM).

Tendo-se x_i é a i-ésima entrada, $p(x_i)$ é o valor predito para a amostra i, y_i é o valor real de saída e n é a quantidade de amostras, o EQM é definido pela seguinte equação:

$$EQM = \frac{1}{n} \sum_{i=1}^{n} (p(x_i) - y_i)^2$$

O DAM, por outro lado, simplesmente é dado pelos valores absolutos dos erros individuais. O DAM é obtido através da seguinte equação:

$$DAM = \frac{1}{n} \sum_{i=1}^{n} | p(x_i) - y_i |$$

Erro de classificação

A tarefa de classificação visa predizer valores categóricos de uma variável dependente. Uma amostra é classificada incorretamente se o valor predito pelo modelo é diferente do valor real da variável. Por outro lado, se o valor predito é igual ao valor real, a amostra foi classificada corretamente. O desempenho de um modelo preditivo é calculado através do número de erros e do número total de amostras (YE, 2003).

Tendo-se que E_m é o erro de classificação do modelo, e é a quantidade de amostras classificadas incorretamente e n é a quantidade total de amostras, o erro de classificação do modelo é dado pela seguinte equação:

$$E_m = \frac{e}{n}$$

Exatidão

A exatidão de um classificador é uma medida complementar ao erro de classificação. Desta forma, a exatidão do modelo é dada pela quantidade de amostras classificadas corretamente divididas pela quantidade total de amostras.

Falsos positivos, falsos negativos e matriz de classificação binária

A classificação de um conjunto de amostras em duas classes é a aplicação mais comum quando se trata de classificação, embora seja possível aplicar esta técnica quando houver um maior conjunto de classes.

Quando a classificação é realizada a um problema com duas classes distintas, o desempenho preditivo do modelo pode ser descrito através de uma matriz quadrada de ordem 2.

A construção desta matriz parte dos seguintes pressupostos: tem-se o rótulo de classe real C_r+ sendo verdadeiro, e o rótulo de classe real C_r- sendo falso. Tem-se C_p+ como a classe predita verdadeira, e C_p- como a classe predita falsa. Há quatro combinações possíveis, estando as combinações corretas na diagonal principal da matriz e as combinações incorretas na diagonal secundária. Os valores de verdadeiros positivos (VP) e verdadeiros negativos (VN) correspondem as respostas corretas e os valores de falsos positivos (FP) e falsos negativos (FN) correspondem as respostas incorretas. A tabela 2.1 apresenta uma tabela de erro de classificação binária (HAND et al., 2001).

Tabela 2.1: Erro de classificação binária

	C_p+	C_p-
C_r+	VP	FN
C_r-	FP	VN

Precisão, Revocação e medida F

Ainda no caso de classificação binária, em muitas aplicações, pode haver uma grande quantidade de exemplos negativos e uma pequena quantidade de exemplos positivos. Nestes casos, um modelo poderia alcançar uma exatidão muito alta simplesmente classificando todos os dados como negativos. Da mesma forma, também poderia ser obtido um baixo erro de classificação.

Para evitar esta armadilha, podem ser utilizadas outras três métricas: precisão, revocação e medida *F*. A precisão denota a proporção entre a quantidade de exemplos que foram corretamente classificados e a quantidade de exemplos classificados como positivos, sejam eles corretos ou não. Já a revocação denota a proporção entre a quantidade de exemplos que foram corretamente classificados como positivos e a quantidade de exemplos que deveriam ter sido classificados como positivos. A medida *F* é obtida através da média harmônica de precisão e revocação (YE, 2003).

$$precisão = \frac{VP}{VP + FP}$$

$$revocação = \frac{VP}{VP + FN}$$

$$MedidaF = \frac{2}{\frac{1}{precisão} + \frac{1}{revocação}}$$

Sensibilidade e especificidade

São duas métricas amplamente utilizadas para avaliação de diagnósticos em medicina. A sensibilidade é idêntica a revocação e reflete o quanto bom é o modelo na identificação de exemplos positivos (HAND et al., 2001). Já a especificidade reflete o quanto bom é o modelo na identificação de exemplos negativos. Sensibilidade e especificidade são dadas pelas seguintes equações:

$$sensibilidade = \frac{VP}{VP + FN}$$

$$especificidade = \frac{VN}{VN + FP}$$

Teoricamente, as métricas de sensibilidade e especificidade são independentes e ambas podem atingir 100% em um mesmo modelo. Porém, na maioria dos problemas práticos, este patamar é difícil de ser atingido.

Matriz de confusão

Como grande parte dos problemas de classificação envolve apenas duas classes, a tabela 2.1 cobre a maioria dos casos (HAND et al., 2001). Porém, em outros casos, a quantidade de classes pode ser superior a duas. Desta forma, podem ser criadas matrizes com ordem maior que 2, de modo que a ordem da matriz seja igual à quantidade de classes do problema. Esta matriz é chamada de matriz de confusão ou tabela de contingência.

A tabela 2.2 ilustra uma matriz de confusão para um modelo preditivo de quatro classes, onde C_{px} representa a classe predita *x*, e C_{ry} representa a classe real *y*. O valor de cada célula é dado por $Z_{i,j}$, onde *i* denota a classe real do exemplo e *j* denota a classe predita pelo modelo. Para todo $Z_{i,j}$ onde $i = j$, o exemplo foi corretamente predito.

Tabela 2.2: Matriz de confusão

	C_{p1}	C_{p2}	C_{p3}	C_{p4}
C_{r1}	$Z_{1,1}$	$Z_{1,2}$	$Z_{1,3}$	$Z_{1,4}$
C_{r2}	$Z_{2,1}$	$Z_{2,2}$	$Z_{2,3}$	$Z_{2,4}$
C_{r3}	$Z_{3,1}$	$Z_{3,2}$	$Z_{3,3}$	$Z_{3,4}$
C_{r4}	$Z_{4,1}$	$Z_{4,2}$	$Z_{4,3}$	$Z_{4,4}$

Curvas ROC

A teoria das curvas ROC (*Receiver Operating Characteristic*) originou-se na teoria de detecção de sinais (HAND et al., 2001). A curva ROC representa a sensibilidade e o complemento da especificidade em um gráfico para um sistema de classificação binário cujo limiar de distinção entre as duas classes é variável. A análise deste gráfico permite a identificação dos modelos provavelmente ótimos e dos modelos sub-ótimos.

O espaço ROC, que é definido pelo complemento da especificidade em função da sensibilidade, apresenta a relação custo (especificidade) x benefício (sensibilidade) dos modelos à medida que o limiar é alterado. A figura 2.5 apresenta o espaço ROC e a representação de quatro modelos distintos: A, B, C e C'.

O modelo ideal estaria representado na coordenada (0,1) do espaço ROC, indicando que todos os exemplos positivos foram encontrados e que nenhum exemplo negativo foi predito como positivo. A incerteza máxima está representada no espaço ROC pela linha diagonal secundária. Pontos acima da diagonal secundária indicam que o modelo consegue representar algum nível de conhecimento a partir dos exemplos, enquanto pontos abaixo desta diagonal indicam que o modelo é ruim.

O modelo representado pelo ponto A mostra os melhores resultados em comparação com os pontos B e C. O ponto B, que está sobre a diagonal secundária, indica que o modelo atinge um nível de 50% de acerto. Já o ponto C representa o pior modelo, pois a maioria de suas decisões é incorreta. Porém, se todas suas decisões forem tomadas ao contrário, seria criado um novo modelo C' que seria o melhor modelo entre os quatro modelos gerados.

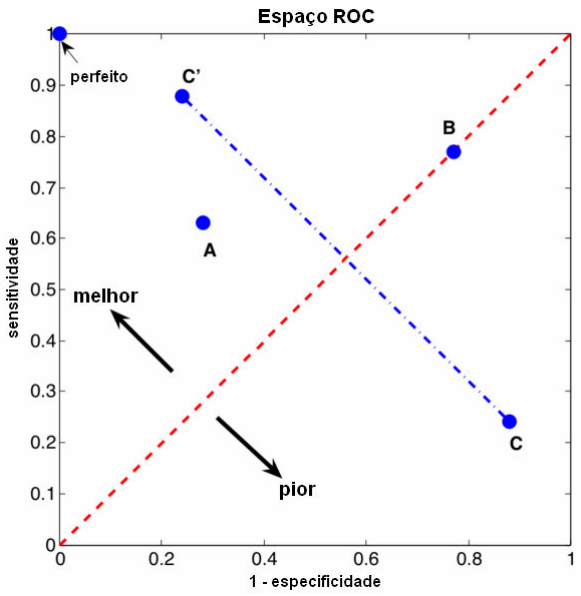

Figura 2.5: O espaço ROC

Curva lift

Curvas *lift* são uma abordagem gráfica para avaliar e comparar a utilidade de diferentes modelos de classificação. O caso mais comum da aplicação de uma curva *lift* é para comparar as respostas entre o modelo criado e a resposta que seria obtida sem a utilização de um modelo. Desta forma, o *lift* permite quantificar a proporção entre os casos positivos encontrados pelo modelo e os casos positivos existentes dentre todos os exemplos (LAROSE, 2005).

A figura 2.6 apresenta um exemplo de curva *lift*. Neste caso, a figura mostra que 10% dos exemplos pertencem à classe em estudo, fato mostrado pela linha base. O gráfico também apresenta o comportamento com a utilização de um modelo preditivo *X*.

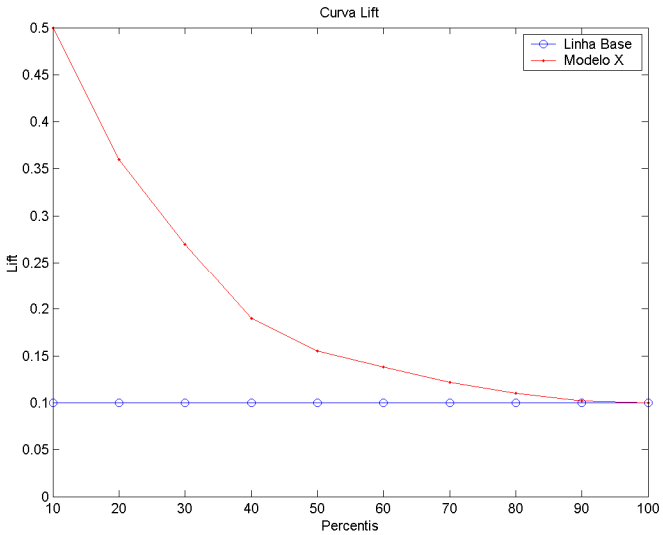

Figura 2.6: Curva *Lift*

2.4.3.3 Outras métricas de desempenho

Apesar do objetivo principal do processo de MD ser a criação de modelos que possam explicar os dados com maior precisão possível, algumas outras métricas também podem ser analisadas no momento de tomar a decisão sobre qual a melhor abordagem para modelar um problema (YE, 2003).

Tempo de Treinamento

O tempo de treinamento refere-se ao tempo necessário para construir o modelo a partir dos dados de treinamento. Segundo Witten e Frank (2005), quando a quantidade de dados é muito grande, duas dimensões distintas tornam-se críticas: espaço e tempo. O espaço torna-se crítico porque algumas abordagens necessitam que todos os dados estejam na memória principal durante o treinamento e/ou teste. Já o tempo torna-se crítico porque muitos dos algoritmos de treinamento não têm escalabilidade linear com o aumento da quantidade de exemplos de treinamento. Já outros algoritmos têm dificuldade em lidar com uma grande quantidade de características de entrada.

O tempo de treinamento de redes neurais, por exemplo, tipicamente é bem maior que o tempo de treinamento de algoritmos de árvores de decisão.

Além do tempo de treinamento, também pode ser levado em consideração o tempo de re-treinamento, em ambientes onde o aprendizado não possa ser incremental. Algumas técnicas de MD têm a necessidade de realizar novamente todo o processo de treinamento se uma nova amostra de treinamento é apresentada.

Tempo de aplicação

O tempo de aplicação refere-se ao tempo necessário para usar o modelo a fim fazer alguma predição sobre um exemplo previamente desconhecido.

Enquanto algumas técnicas têm um tempo de treinamento alto e um tempo de aplicação baixo, outras técnicas não têm tempo de treinamento, mas sua aplicação em quantidades muito grandes de dados pode ser impossível devido ao consumo dos críticos recursos de memória e tempo. Aplicações de DCBD devem levar em conta tanto o tempo de treinamento quanto o tempo de aplicação. Quando a quantidade de dados utilizada nos processos de treinamento e aplicação for pequena, os recursos de espaço e tempo deixam de ser críticos.

2.4.4 Pós-Processamento

Esta fase tem por objetivos traduzir o conhecimento obtido como resultado da aplicação dos algoritmos para uma linguagem passível de compreensão e assegurar a qualidade deste conhecimento descoberto.

Interessabilidade

A interessabilidade do modelo refere-se a sua capacidade de gerar conhecimento que seja interessante para o usuário. As medidas de interesse podem ser subjetivas, objetivas ou imparciais. O interesse subjetivo leva em conta explicitamente as necessidades específicas do usuário e conhecimento prévio. O interesse objetivo mede a relevância de um padrão a partir de sua estrutura e dos dados usados no processo de descoberta de conhecimento porém, ainda requer certo nível de intervenção do usuário. O interesse imparcial refere-se a medidas que podem ser aplicadas autonomamente sobre o resultado do algoritmo a fim de reduzir a quantidade de regras não interessantes,

independentemente do domínio do problema, da tarefa ou dos usuários (MAIMON e ROKACH, 2005).

Interpretabilidade

A interpretabilidade do modelo refere-se a sua capacidade de poder ser traduzido para uma linguagem compreensível pelo ser humano. A interpretabilidade também pode consistir em um importante fator a ser considerado na seleção do melhor modelo, pois permitiria as pessoas envolvidas como o processo de MD obterem algum conhecimento sobre os dados e corrigir alguma eventual falha deste processo.

Modelos baseados em regras são facilmente interpretáveis, já modelos de "caixa preta", tais como as redes neurais, têm grande dificuldade de interpretação. Porém, especificamente no caso das redes neurais, já há diversas propostas na literatura para extração de regras a partir de modelos neurais, embora nenhuma delas seja amplamente aceita e utilizada.

Avaliação do especialista

Em alguns domínios, os modelos criados pelo processo de DCBD têm aplicabilidade prática, podendo ser aplicados sobre dados reais e obtendo resultados que podem ser facilmente avaliados. Porém, em áreas onde o especialista em DCBD tem pouco conhecimento, o envolvimento de um especialista do domínio é um fator que pode contribuir decisivamente para o sucesso do trabalho.

Em áreas críticas, tais como as que lidam diretamente com a saúde humana, também há uma grande dependência do especialista do domínio. Além disso, nessas áreas, o especialista do domínio certamente não basearia suas crenças em um sistema "caixa preta" para tomar sua decisão e a utilização do modelo certamente estaria ligada a existência de alguma forma de consulta aos fatos utilizados pelo modelo para fundamentar sua decisão.

Divergências entre os resultados do modelo e as decisões tomadas pelo especialista do domínio, ainda que raras, podem existir. O modelo criado simplesmente irá representar o conhecimento expresso nos dados de treinamento. A existência de algum viés nestes dados de treinamento irá implicar na possibilidade de decisões incorretas. A existência de algum viés nas decisões do especialista do domínio é pouco provável, pois, possivelmente, sua experiência esteja baseada em um conjunto muito maior de exemplos do que o restrito conjunto de treinamento utilizado pelo processo de DCBD.

Teste de campo

Após a criação de modelos através do processo de DCBD a grande incógnita é sobre o desempenho futuro deste modelo na predição de dados reais. Sem dúvida, a avaliação final do modelo se dará através de sua aplicação sobre dados reais e previamente desconhecidos. Dados reais consistirão em uma valiosa forma de testar a robustez do modelo. Isto ocorre porque dados do mundo real podem ter peculiaridades adicionais em relação aos dados pré-processados e utilizados previamente para treinamento e teste que podem fazer o modelo comportar-se de modo inesperado.

Neste capítulo foram abordados os principais conceitos de descoberta de conhecimento em banco de dados e as fases deste processo. No próximo capítulo, serão apresentados os principais conceitos sobre as RNAs.

3 REDES NEURAIS ARTIFICIAIS

Do ponto de vista físico, um computador moderno pode ser considerado como um artefato constituído de um conjunto de componentes eletrônicos com a capacidade de executar tarefas algorítmicas. Porém, a dificuldade do computador realizar tarefas de natureza não-algorítmica é notória. Já o cérebro dos seres vivos, principalmente dos seres humanos, tem uma peculiar capacidade para solucionar problemas de natureza não algorítmica. Nada mais natural que se desenvolvesse um modelo para solução de problemas não algorítmicos com inspiração no funcionamento do cérebro humano. Desta forma, surgiu o paradigma das Redes Neurais Artificiais (RNAs).

As RNAs consistem um uma poderosa abordagem para aprendizado de valores reais, discretos ou vetoriais. Esta abordagem implementa aspectos importantes de sistemas de reconhecimento de padrões, tais como robustez, adaptatividade, velocidade e aprendizado. O aprendizado é realizado através de exemplos discriminando as características entre os vários padrões de entrada. A partir destes exemplos, a RNA iterativamente reduz o erro e automaticamente descobre os relacionamentos inerentes aos dados (MITRA e ACHARYA, 2003). Estudos realizados sobre as redes neurais humanas constataram que populações neurais envolvidas na codificação de memórias também extraem uma espécie de conceitos generalizados que nos permitem transformar nossas experiências diárias em conhecimento e idéias (TSIEN, 2007). Diversos trabalhos disponíveis na literatura científica relatam o sucesso da aplicação das RNAs nos mais diversos problemas (MITCHELL, 1997). Dentre as tarefas de mineração de dados que podem ser executadas pelas redes neurais estão: classificação de padrões, agrupamento, aproximação de funções, regressão e controle (MITRA e ACHARYA, 2003).

Este capítulo aborda os conceitos fundamentais a respeito das RNAs, a inspiração biológica, o neurônio artificial, arquiteturas básicas das RNAs e as principais formas de treinamento.

3.1 Inspiração biológica

O sistema nervoso central humano consiste de unidades celulares básicas chamadas de neurônios, os quais compõem o cérebro, a retina e a medula espinhal. Os neurônios são altamente estimuláveis, sendo capazes de captar mínimas variações elétricas que ocorrem ao seu redor, processá-las e gerar sinais para outros neurônios vizinhos. Estas variações elétricas são chamadas de impulsos nervosos.

O funcionamento de um neurônio biológico, que decide a natureza de seu sinal de saída como uma função de seus sinais de entrada, ainda não é plenamente conhecida (KONAR, 2000). Apesar disso, existe um consenso entre os pesquisadores da área biológica que o neurônio, após receber um conjunto de sinais de entrada, estima uma média ponderada destes sinais e limita a amplitude resultante do sinal processado através de uma função de inibição não linear. A razão da não linearidade é a concentração de íons de potássio dentro da célula e de sódio fora da célula, que causam uma diferença de potencial elétrico através da membrana celular. Esta diferença de potencial pode assumir valores diferentes para cada um dos neurônios vizinhos devido às diferentes concentrações iônicas locais. Quando um neurônio recebe sinais de seus vizinhos, cada um destes sinais é atenuado diferentemente pelas diferenças de potencial locais. Conseqüentemente, estas diferenças de concentração iônica agem como ponderadores na determinação da média dos estímulos que será feita pelo neurônio em questão.

Para executar suas funções, um neurônio biológico tem três componentes estruturais básicos: os dendritos, o corpo da célula e o axônio. Os dendritos são um conjunto de ramificações que partem do corpo do neurônio e agem como receptores, recebendo os sinais dos neurônios vizinhos e transmitindo-os para o corpo do neurônio. O corpo do neurônio, também chamado de soma, recebe os sinais coletados, processa-os e envia o sinal resultante por uma longa fibra chamada axônio. Na extremidade do axônio estão as terminações sinápticas, onde agem os inibidores. As terminações sinápticas, ou sinapses, controlam o fluxo dos impulsos nervosos do neurônio atual para os dendritos dos neurônios vizinhos.

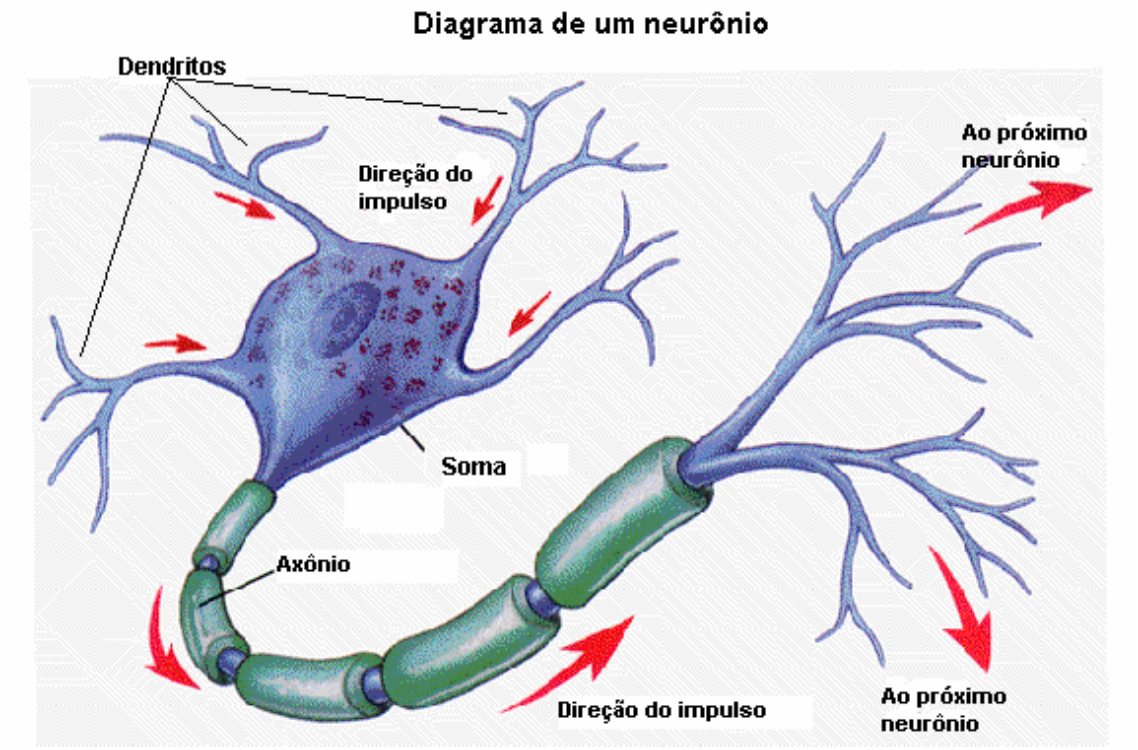

Figura 3.1: O neurônio biológico

O cérebro humano é um sistema de processamento paralelo composto por aproximadamente 10^{11} neurônios, sendo que cada um deles pode receber estímulos de

em torno de 10^3 a 10^4 dendritos e, após o processamento, gera somente uma única saída. O fluxo dos impulsos nervosos se dá no sentido das setas, conforme figura 3.1.

3.2 O neurônio artificial

O neurônio artificial consiste na unidade básica de processamento de informação de uma RNA. A figura abaixo apresenta o modelo do neurônio artificial, que é a unidade fundamental de uma RNA. Tal como seu análogo biológico, o neurônio artificial tem três elementos básicos (FREEMAN e SKAPURA, 1991):

- Um conjunto de sinapses, sendo cada uma delas caracterizada por um peso. Tendo-se uma sinapse de entrada j conectada ao neurônio k, seu valor de entrada x_j é multiplicado pelo seu peso sináptico w_{kj}.
- Uma função de propagação, tipicamente representada por um somatório do produto dos sinais de entrada pelos seus respectivos pesos.
- Uma função de ativação que limita a amplitude do valor de saída do neurônio.

A figura 3.2 apresenta a estrutura básica de um neurônio artificial.

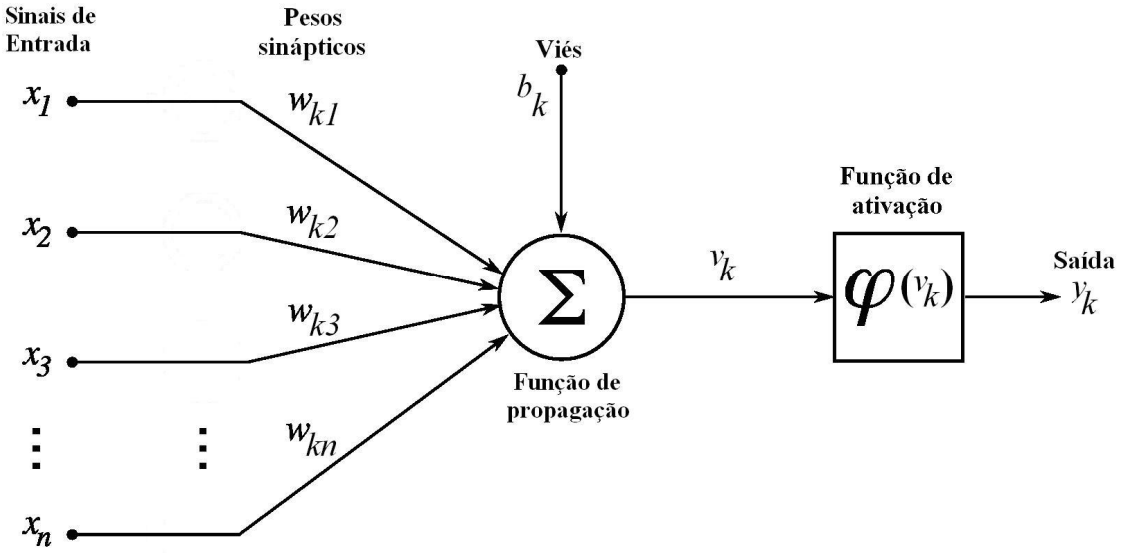

Figura 3.2: O neurônio artificial

A partir da analogia com o neurônio biológico, a descrição matemática do neurônio artificial baseia-se em um modelo com valores das n características de entrada representando os dendritos, sendo elas: $\{x_1, x_2, x_3, ..., x_n\}$. Tipicamente os valores de x variam nos intervalos entre [0,1] ou [-1,1]. Existe também um valor de saída representando o axônio, sendo ele: y_k. Com a finalidade de simular o comportamento das sinapses, as características de entrada possuem pesos sinápticos acoplados, sendo eles: $\{w_{k1}, w_{k2}, w_{k3}, ..., w_{kn}\}$. Os pesos sinápticos podem assumir valores positivos para sinapses excitatórias, ou negativos, para sinapses inibitórias.

O modelo neural também inclui um viés aplicado externamente, denotado por b_k, cuja finalidade é aumentar ou diminuir a entrada da rede da função de ativação, e y_k é o sinal de saída do neurônio. De uma maneira mais formal, um neurônio k pode ser descrito pelo seguinte conjunto de equações:

$$u_k = \sum_{j=1}^{n} w_{kj} x_j$$

e

$$y_k = \varphi(u_k + b_k)$$

Onde x_1, x_2, x_3, ..., x_n são os sinais de entrada; w_{k1}, w_{k2}, w_{k3}, ..., w_{kn} são os pesos sinápticos do neurônio k; v_k é a saída do combinador linear gerada pelos sinais de entrada; b_k é o viés; φ é a função de ativação; e y_k é o sinal de saída do neurônio (HAYKIN, 1999).

3.2.1 Funções de ativação

Cada neurônio propaga seu resultado para outros neurônios conectados a sua saída. Porém, este resultado, antes de ser repassado para outros neurônios, tem seu valor influenciado pela chamada função de ativação.

Enquanto a função de propagação de um neurônio artificial, que representa corpo da célula neural biológica, é modelado por uma função linear, a função de ativação, que representa a sua sinapse, pode ser de natureza linear ou não linear. O tipo de função de ativação depende do problema que o neurônio está tentando resolver. Em problemas lineares são utilizadas funções de ativação lineares.

Em problemas que possuam não linearidade, a sinapse do neurônio pode ser modelada por uma função de inibição não linear a fim de limitar a amplitude do sinal processado pela função de propagação. Desta forma, a utilização de funções de ativação nos neurônios ocultos de uma rede neural artificial é necessária para inserir a não linearidade na rede.

A função de ativação, denotada por $\varphi(y_k)$, na figura 3.2 define o valor de saída y_k de um neurônio.

Os tipos básicos de funções de ativação são:

- Função de limiar: esta função define a saída do neurônio para os valores 0 ou 1, sendo usada para classificar entradas em duas categorias distintas. A função de limiar, apresentada na figura 3.3 a), é definida da seguinte forma:

$$\varphi(v) = \begin{cases} 1 & se \quad v \geq 0 \\ 0 & se \quad v < 0 \end{cases}$$

- Função linear: a saída de uma função de ativação linear é igual a sua entrada, conforme apresentado na figura 3.3 b), e sua função de saída é dada por:

$$\varphi(v) = v$$

- Função de saturação: esta função, apresentada na figura 3.3 c), é definida da seguinte forma:

$$\varphi(v) = \begin{cases} 0 & se \quad v < 0 \\ v & se \quad 0 \leq v \leq 1 \\ 1 & se \quad v > 1 \end{cases}$$

- Função sigmóide logarítmica: esta função possui a propriedade de diferenciabilidade contínua. Sua desvantagem é a de restringir sua saída a somente valores positivos. A função sigmóide logarítmica é apresentada na figura 3.3 d), sendo definida por:

$$\varphi(v) = \frac{1}{1+\exp^{(-v)}}$$

- Função tangente hiperbólica: esta função é similar a sigmóide, mas tem como vantagem o fato de sua saída gerar valores positivos e negativos. Esta função, apresentada na figura 3.3 e), é definida por:

$$\varphi(v) = \frac{2}{(1+\exp^{(-2\times v)})} - 1$$

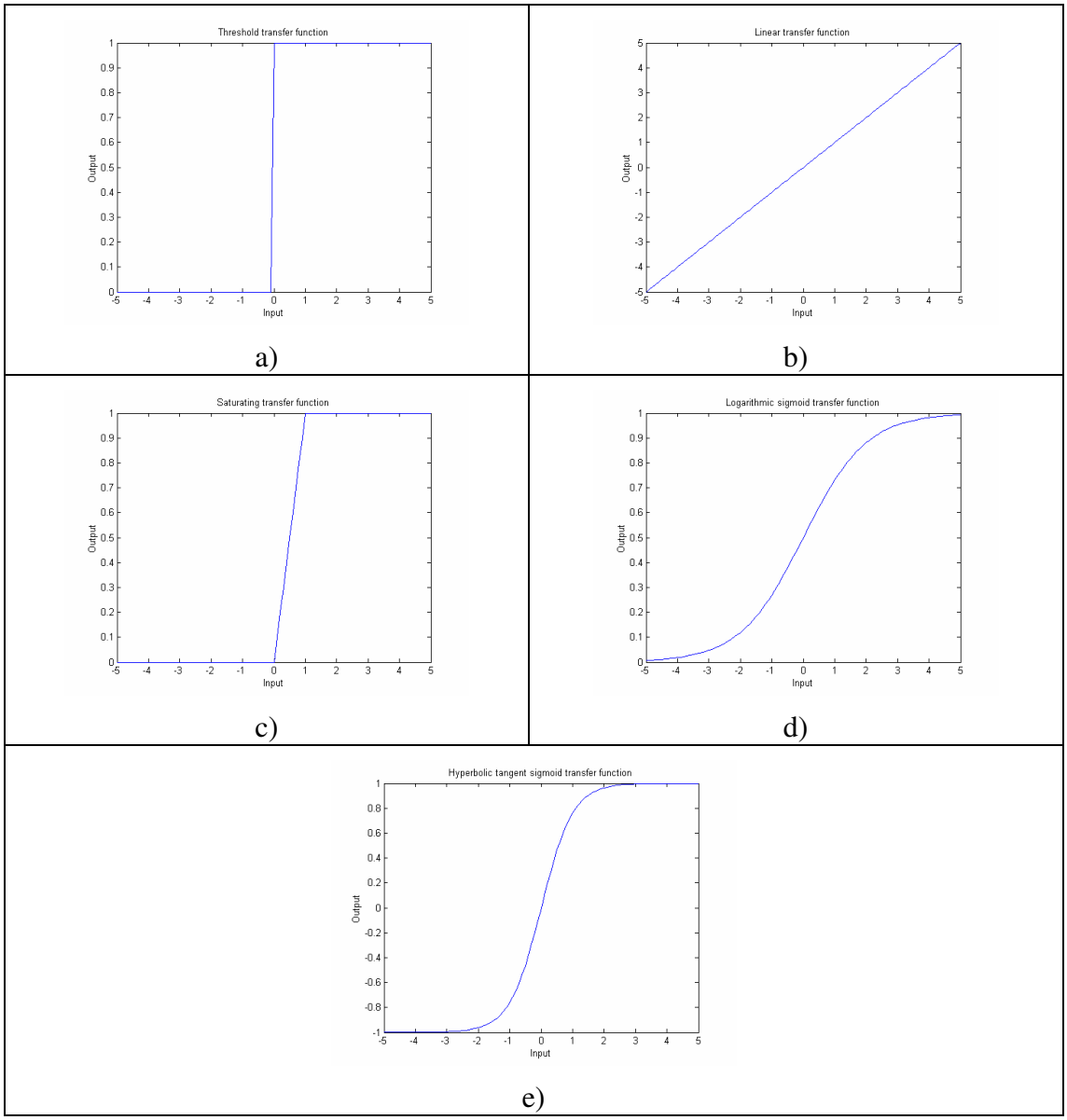

Figura 3.3: Funções de ativação

Além dos tipos básicos apresentados, há inúmeras outras funções de ativação, utilizadas em casos mais específicos (HAGAN et al., 1995), que não foram aqui mencionadas.

3.3 A Rede Neural Artificial

Uma rede neural artificial é um modelo computacional abstrato do cérebro humano. Assim como o cérebro, uma RNA é composta por um conjunto de neurônios artificiais, ou nodos, que são unidades de processamento dotadas de parâmetros adaptativos. Estes nodos são interconectados através de ligações direcionais, que refletem uma relação causal entre os nodos de suas extremidades. Os neurônios estão distribuídos em camadas sendo que os neurônios pertencentes à mesma camada funcionam de forma paralela. De uma maneira mais formal, uma rede neural artificial pode ser vista como um grafo dirigido com pesos. Neste grafo os neurônios artificiais são os nodos e as arestas dirigidas são as conexões entre os neurônios (HAYKIN, 1999). A forma na qual a RNA está estruturada é discutida na seção 3.4.1.

Um organismo é dito como inteligente se ele consegue aprimorar seu comportamento à medida que aumenta sua experiência. Um comportamento aprimorado irá fazer com que o organismo inteligente melhore os resultados de suas ações com o passar do tempo. A expressão da inteligência de uma RNA pode se dar por dois motivos: alteração na estrutura da rede ou alteração nos pesos sinápticos. Quase que a totalidade dos algoritmos de aprendizado de RNAs agem somente a nível de alteração de pesos sinápticos.

3.3.1 Arquiteturas de rede

A arquitetura de uma RNA é definida pelas características de um nodo e pelas características de conectividade dos nodos da rede. As arquiteturas são escolhidas de acordo com as características do problema a ser tratado. Além disso, a forma pela qual os neurônios estão distribuídos também tem uma estreita relação com o algoritmo de treinamento que será utilizado. Dentre as propriedades que caracterizam as diferentes arquiteturas estão:

- Quantidade de camadas:
    - Camada única: em redes de camada única existe somente um nodo entre qualquer entrada e qualquer saída da rede neural.
    - Múltiplas camadas: em redes de múltiplas camadas existe mais de um nodo entre qualquer entrada e qualquer saída da rede neural.
- Conexões dos nodos:
    - Alimentadas a diante: a saída de um nodo da *i*-ésima camada da rede não pode ser usada como entrada de outro nodo da *j*-ésima camada, tal que $j <= i$.
    - Retroalimentadas: a saída de um nodo da *i*-ésima camada da rede é utilizada como entrada de outro nodo da *j*-ésima camada, tal que $j <= i$.
- Conectividade da rede:

- Fracamente conectada: nem todos os nodos da *i*-ésima camada estão conectado com os nodos da *j*-ésima camada, tal que $i = j+1$.
- Totalmente conectada: todos os nodos da *i*-ésima camada estão conectado com os nodos da *j*-ésima camada, tal que $i = j+1$.

A partir da combinação destas propriedades básicas, algumas arquiteturas são mais comuns sendo aplicadas na grande maioria dos problemas. São elas:

- Redes de camada única.
- Redes multicamadas, alimentadas adiante, totalmente conectadas.
- Redes retro-alimentadas.

Em termos da arquitetura de rede, as RNAs são muito diferentes do cérebro humano principalmente pelo fato das RNAs terem uma estrutura organizada e hierárquica. Em termos de escala, o cérebro humano atinge uma escala muito maior que qualquer RNAs já projetada. Porém, mantidas as devidas proporções, as RNAs conseguem paralelizar o processamento da informação de uma forma pretensamente similar ao cérebro humano. Esta paralelização permite que as RNAs consigam executar tarefas muito específicas que requerem inteligência.

O presente trabalho tem como foco de aplicação as redes multicamadas. Embora também seja possível a aplicação da técnica proposta em outras arquiteturas, não foram feitos experimentos que pudessem comprovar tal possibilidade.

3.3.1.1 Redes Multicamadas

Como mencionado anteriormente, as redes multicamadas caracterizam-se pela presença de mais de um neurônio entre suas entradas e suas saídas. As camadas de uma rede multicamadas são classificadas em três grupos:

- Camada de entrada: onde os padrões são apresentados à rede.
- Camadas ocultas: onde é realizada a maior parte do processamento da rede. Pode haver uma ou mais camadas ocultas.
- Camada de saída: onde o resultado final é obtido e apresentado.

As camadas ocultas e de saída são compostas por neurônios, o que significa que elas têm capacidade de processamento. A camada de entrada é composta por elementos que somente repassam para a camada seguinte o seu estímulo de entrada, sem realizar nenhum processamento com sua entrada. Além disso, as redes multicamadas são obrigatoriamente alimentadas adiante, e podem ser tanto fracamente conectadas quanto totalmente conectadas.

A quantidade de neurônios nas camadas ocultas e de saída, assim como a quantidade de camadas ocultas, variam de acordo com a natureza do problema a ser aprendido e devem ser definidas durante o projeto da rede. A definição da quantidade de neurônios na camada de saída é trivial. Em tarefas de regressão geralmente é utilizado somente um neurônio na camada de saída. Em tarefas de classificação geralmente são utilizados tantos neurônios quantas forem as classes a serem preditas. Já as decisões referentes à camada oculta são as mais difíceis, não existindo regras plenamente aceitas para isso, porém existe um consenso que dificilmente devem ser necessárias mais de duas camadas ocultas (MUNAKATA, 2008). A decisão da quantidade de neurônios na

primeira camada oculta e na segunda, caso ela seja necessária, geralmente é tomada após a análise de diversas configurações de valores distintas. Desta forma, são criadas diversas RNAs, cada uma delas com diferentes configurações de camadas ocultas. A configuração que obtiver melhor resultado na modelagem do problema será utilizada. Existem também algumas abordagens híbridas que utilizam algoritmos genéticos para definir estas configurações (TAYLOR, 2006).

A figura 3.4 apresenta um exemplo típico de uma RNA multicamada, alimentada adiante, totalmente conectada, com 7 entradas, 1 camada oculta com 10 neurônios, e 3 neurônios na camada de saída.

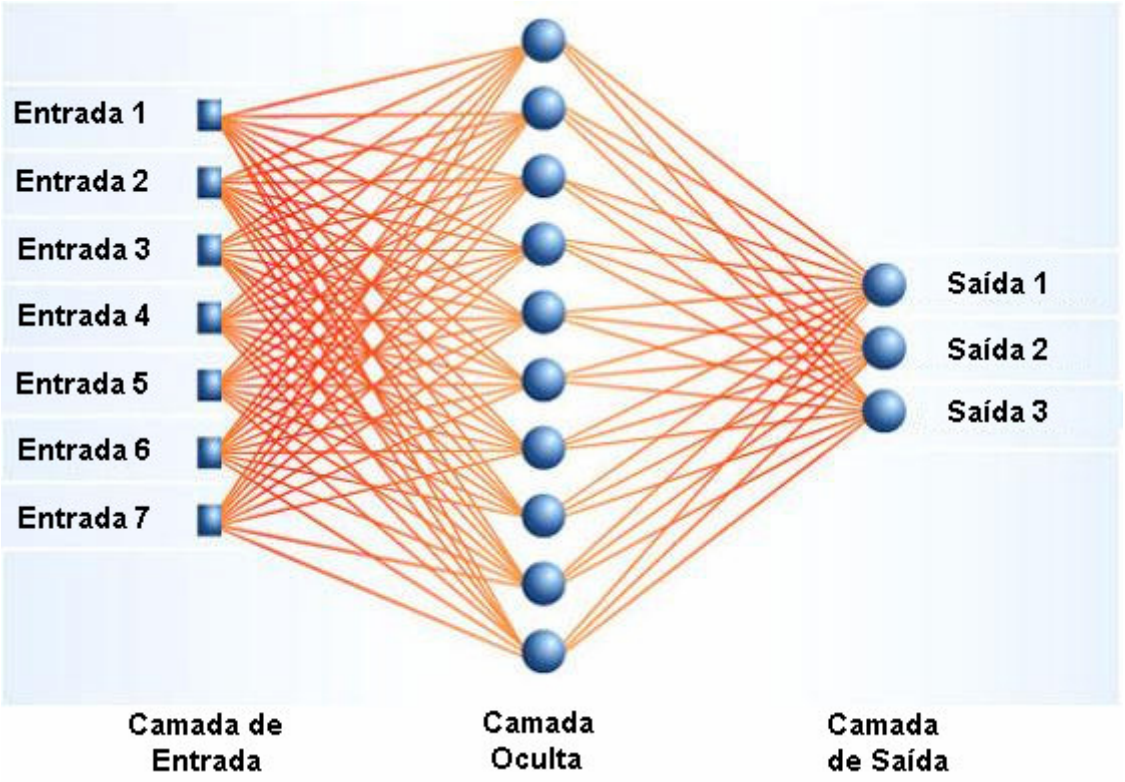

Figura 3.4: Exemplo típico de uma RNA multicamada

3.3.2 Algoritmos de treinamento

O objetivo dos algoritmos de treinamento é fazer a RNA aprender os conceitos expressos nos dados de treinamento. O processo de aprendizado pode ser definido como a pesquisa pelo modelo mais adequado, ou hipótese, descrevendo um conjunto de dados (GUYON, 2006). A definição do modelo mais adequado pode ser vista como uma função na forma:

$$f: \quad X \rightarrow Y$$

onde X representa o conjunto de exemplos de treinamento, e Y representa os valores alvo da função, e f representa o modelo que faz o mapeamento. Tal modelo geralmente depende de parâmetros adaptativos, e o processo de aprendizado consiste em pesquisar os valores ótimos para estes parâmetros adaptativos. Os algoritmos de treinamento das redes neurais consistem em abordagens heurísticas para vasculhar grandes espaços de pesquisa a fim de definir os valores destes parâmetros adaptativos.

Existem diferentes algoritmos de treinamento para RNAs. Estes algoritmos têm uma estreita relação com o tipo de problema que a RNA irá tratar. As RNAs podem ser aplicadas a três tipos de problemas de aprendizado: supervisionado, não supervisionado ou por reforço. Neste trabalho o foco é voltado a problemas de aprendizado supervisionado. Nestes casos, o algoritmo mais utilizado é o *backpropagation*. O algoritmo *backpropagation* requer uma rede com topologia multicamada alimentada a diante totalmente conectada. Embora não existam pesquisas que comprovem este fato, acredita-se que em torno de 90% das aplicações comerciais e industriais de redes neurais utilizem o algoritmo *backpropagation* ou suas variantes (MUNAKATA, 2008).

Dado que o treinamento será supervisionado, todas as saídas são conhecidas para cada entrada. Partindo-se deste princípio, para um determinado padrão de entrada, o valor de saída é estimado através de uma propagação do vetor que representa esta entrada para frente na rede. Ao final desta propagação, o vetor de erro na camada de saída é estimado a partir da diferença entre a saída obtida e a saída desejada. A função de erro dos nodos da camada de saída é então retro-propagada através da rede para cada camada ajustando os pesos na camada. A política de adaptação de pesos no algoritmo *backpropagation* é derivada da abordagem do gradiente descendente de encontrar o mínimo de uma função multi-valorada. Durante a aplicação do algoritmo *backpropagation* é possível então identificar dois tipos distintos de sinais na rede: os sinais funcionais, que são propagados no sentido da entrada para a saída; e os sinais de erro, que são propagados no sentido da saída para a entrada.

Didaticamente, o funcionamento do algoritmo *backpropagation* pode ser dividido em três fases: iniciação, treinamento e critério de parada.

Na fase de iniciação são definidos a estrutura da RNA e os valores de alguns parâmetros básicos para o processo de treinamento:

1. Definir a estrutura da RNA, em termos de quantidade de camadas, de quantidade de neurônios em cada camada, e definir a função de ativação que será utilizada em cada camada;
2. Definir o número máximo de épocas de treinamento;
3. Definir a forma de iniciação dos valores dos pesos sinápticos, que por padrão, assumem valores aleatórios entre -1 e 1;
4. Definir os valores do momento e da taxa de aprendizado;
5. Definir um valor alvo para o EQM;

Muitas vezes somente a execução dos passos 1 e 2, descritos anteriormente, é necessária, já que os parâmetros definidos nos passos de 4 a 5 podem utilizar valores padrão.

Na fase de treinamento efetivamente ocorre o aprendizado da RNA, ou seja, ocorre a adaptação dos pesos. Nesta fase podem ser enfatizados os seguintes passos do algoritmo:

1. Apresentar o vetor de entradas do conjunto de treinamento à rede.
2. Para cada exemplo de entrada deve ser executado um ciclo completo de propagação da entrada e retropropagação do erro;
3. Quando todos os exemplos de entrada do conjunto de treinamento tiverem sido apresentados à rede, está finalizada uma época de treinamento.

4. Inicia-se uma nova época de treinamento, apresentando todo o vetor de entradas à rede até que o critério de parada seja atingido.

O critério de parada pode ser atingido de várias formas distintas:

1. O EQM atingiu um valor suficientemente baixo. Este EQM deve ser calculado com base no erro de todos os exemplos do conjunto de treinamento.
2. O EQM atingiu um valor abaixo do limiar definido na fase de iniciação.
3. A variação do EQM atingiu um valor suficientemente baixo, ou seja, a cada nova época aprende-se muito pouco em relação à época anterior.
4. A quantidade máxima de épocas de treinamento foi atingida, caso este valor tenha sido definido na fase de iniciação.

O passo 2 da fase de treinamento ainda pode ser mais bem detalhado, sendo executados os seguintes sub-processos:

1. Apresentar as entradas do próximo exemplo de treinamento para a camada de entrada.
2. Passar estes valores para a camada seguinte.
3. Realizar o somatório ponderado pelos pesos e calcular as ativações.
4. Apresentar as ativações para a próxima camada, repetindo os passos 3 e 4 até atingir a camada de saída.
5. Ao atingir a camada de saída da rede, calcular o erro comparando a saída da rede com o valor desejado para o padrão.
6. Propagar o erro para a camada anterior, ajustando os pesos, até atingir a camada de entrada.
7. Repetir os passos de 1 a 6, até que todos os exemplos do conjunto de treinamento tenham sido vistos.

O sub-processo 6, descrito logo acima, é o que efetivamente ajusta os pesos da RNA. Este ajuste dos pesos obedece à regra delta generalizada, que foi proposta por Rumelhart e McClelland (1986). Segundo esta regra, dado um vetor d(k) com as respostas de saída desejadas para um determinado conjunto k de treinamento, onde $d(k) = [d_1(k), d_2(k), ..., d_m(k)]^T$, e um vetor $y(k)$ com as respostas de saída obtidas pela RNA para este conjunto de treinamento, onde $y(k) = [y_1(k), y_2(k), ..., y_m(k)]^T$, a regra delta generalizada executa um processo de otimização tal que cada erro de saída seja minimizado, assumindo-se que o erro é definido pela diferença entre os vetores $d(k)$ e $y(k)$.

Uma função de erro instantâneo para a rede é dada pela soma dos quadrados dos erros de saída para todas as unidades de saída através da seguinte equação:

$$E = \frac{1}{2}\sum_{j=1}^{m}[d_j(k) - y_j(k)]^2 = \frac{1}{2}\sum_{j=1}^{m} e_j^2(k)$$

onde o erro de saída e_j descreve o erro entre a j-ésima resposta desejada e a j-ésima saída da rede, e é dado por:

$$e_j = (d_j - y_j)$$

e a constante 1/2 foi introduzida para conveniência no cálculo das derivadas.

Finalmente, se após a execução do algoritmo *backpropagation* não tiverem sido atingidos resultados razoáveis, é conveniente a reformulação das decisões tomadas no passo 1 da fase de iniciação, que é extremamente dependente de cada problema a ser modelado. Além disso, supõe-se que o conjunto de treinamento é representativo em relação ao conjunto de teste.

3.3.3 Codificação de entradas e saídas

Também é importante salientar que as RNAs somente tem capacidade de tratar características numéricas. Tanto os valores de entrada fornecidos quanto os valores de saída obtidos de uma rede neural devem ser transformados de forma a ficarem nos intervalos [0,1] ou [-1,1], dependendo da função de ativação utilizada.

Para variáveis contínuas, devem ser utilizadas as técnicas de normalização. Para variáveis discretas ordenadas devem ser utilizadas as técnicas de *binning*. Para variáveis categóricas devem ser utilizadas as técnicas de conversão de tipos de dados. Estas técnicas são mostradas na seção 2.4.1.5.

Apesar da grande flexibilidade das redes neurais, que permitem a modelagem de uma grande variedade de problemas tanto de natureza linear quanto não linear, esta técnica sofre pesadas críticas pela dificuldade de compreensão dos modelos gerados, o que leva a uma dificuldade de interpretação de seus resultados (LAROSE, 2005). Ainda não existe uma abordagem amplamente aceita que possa extrair o conhecimento armazenado em uma rede neural, embora existam numerosas pesquisas neste sentido.

Neste capítulo foram abordados os principais conceitos sobre as RNAs. No próximo capítulo será abordado o processo da redução de dimensionalidade dos dados, com ênfase na apresentação das técnicas para execução deste processo.

4 REDUÇÃO DE DIMENSIONALIDADE DOS DADOS

Este capítulo apresenta os conceitos básicos a respeito da redução de dimensionalidade dos dados. O problema da maldição da dimensionalidade e o fenômeno do pico, que são as principais justificativas para utilização de técnicas de redução de dimensionalidade, são comentados. Também são apresentadas as técnicas de redução de dimensionalidade e suas respectivas classificações, dando ênfase especial à seleção de subconjunto de características (SSC), que é o foco principal deste trabalho.

4.1 Panorama atual

Nos dias atuais, durante a investigação de fenômenos ou processos, os cientistas constantemente têm se deparado com a necessidade de encontrar estruturas significativas ocultas, de baixa dimensão, dentro de dados de alta dimensão, sendo tal técnica denominada de redução de dimensionalidade dos dados (RDD). Analogamente, o cérebro humano se confronta com o mesmo problema em suas percepções diárias, extraindo, de forma eficiente, um pequeno número de estímulos relevantes a partir de aproximadamente 30.000 fibras nervosas sensoriais (TENENBAUM et al., 2000).

Na maioria das vezes em que estão lidando com dados de alta dimensão, os cientistas têm buscado auxílio em técnicas autônomas de modelagem, tais como as redes neurais. A alta dimensão dos dados manifesta-se através de uma grande quantidade de exemplos e de características descrevendo cada exemplo. À medida que a quantidade de características incrementa, as técnicas de modelagem tornam-se menos precisas e mais lentas. O tempo do processo freqüentemente aumenta em escala exponencial ou polinomial em relação ao incremento da quantidade de características. Um dos grandes desafios para a aplicação da modelagem neural e de outras técnicas de modelagem é maldição da dimensionalidade, que será abordada na próxima seção.

4.2 Maldição da dimensionalidade e o fenômeno do pico

A maldição da dimensionalidade, também conhecida por problema da dimensionalidade ou comportamento de curva em U, foi um problema descoberto por Bellman (1961). Tal problema têm sido frequentemente observado na literatura, sendo observado que o acréscimo de características geralmente degrada o desempenho de um classificador ou regressor se a quantidade de exemplos de treinamento for pequena em relação à quantidade de características.

Adicionalmente, o uso de muitas variáveis de entrada para modelar um conjunto de dados pode desnecessariamente complicar a interpretação dos modelos criados e viola o princípio da parcimônia. De acordo com este princípio, sempre que possível, deve ser considerado um menor número de variáveis no modelo, de forma que ele possa ser mais

facilmente interpretado (LAROSE, 2006). O princípio da parcimônia também é amplamente conhecido como navalha de Occam.

O efeito da curva em U apresenta três regiões com comportamentos distintos do erro em relação à dimensionalidade dos dados de entrada, são elas:

- Região inicial (RI): onde o incremento de características implica uma redução na taxa de erro. Isto ocorre porque os conjuntos de características muito pequenos geralmente não possuem a informação suficiente para a distinção dos padrões de entrada. Cada característica adicionada dá muita informação relevante ao classificador ou regressor, permitindo a diminuição da taxa de erro;
- Região média (RM): onde a taxa de erro atinge um nível de estabilidade mesmo com o incremento de características. As características com muita informação relevante já foram inseridas na região anterior, já as características inseridas neste ponto têm pouca informação relevante para a distinção dos padrões. A inclusão destas características então tende a alterar sutilmente a taxa de erro.
- Região final (RF): é a região onde se manifesta o problema da dimensionalidade, onde o incremento de características provoca um incremento também na taxa de erro. Cada característica adicionada aumenta a quantidade de parâmetros a serem adaptados. Se a característica não possui nenhuma informação relevante para a distinção dos padrões e ocorre o aumento dos parâmetros a serem adaptados, a tendência é uma piora na capacidade preditiva do modelo.

A figura 4.1 representa a curva em U.

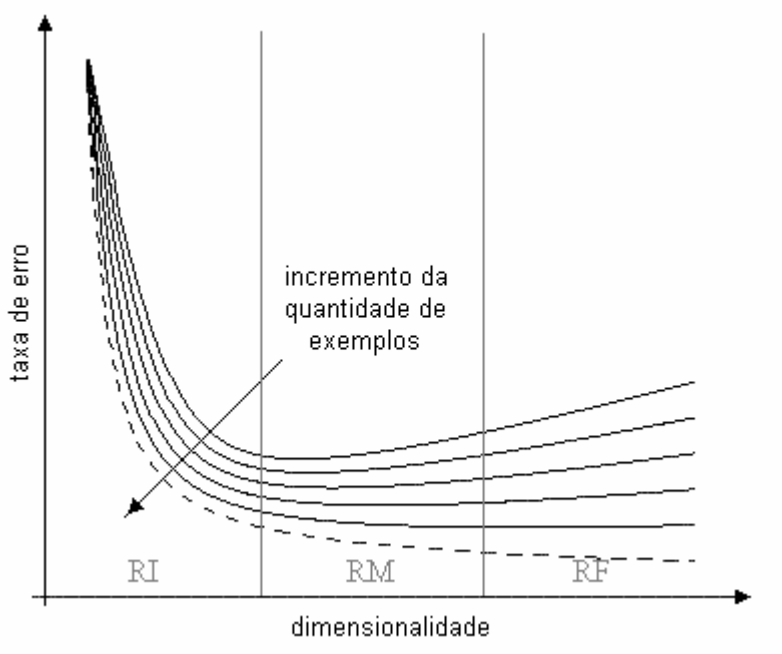

Figura 4.1: Taxa de erro em função da dimensionalidade

A maldição da dimensionalidade pode influenciar todos os classificadores e regressores mais comumente usados. Adicionalmente, o desempenho do classificador ou regressor depende não somente da quantidade de características que descrevem cada exemplo, mas também da quantidade de exemplos, da quantidade de padrões de entrada, e da quantidade de parâmetros do classificador ou regressor a serem adaptados. Infelizmente é muito difícil estabelecer a relação entre a taxa de erro, a quantidade de

exemplos de treinamento, a quantidade de características e a quantidade de parâmetros adaptativos do classificador ou regressor (JAIN et al., 2000). Porém, a fim de evitar os problemas inerentes à maldição da dimensionalidade, é recomendada a utilização de 10 a 20 exemplos de treinamento para cada característica do exemplo (BELLMAN, 1961).

Por outro lado, o fenômeno do pico é observado quando a taxa de erro atinge o valor máximo para um determinado número de características, mas decrementa com o acréscimo de características. Este problema denota que a quantidade de exemplos de treinamento pode crescer exponencialmente em relação à quantidade de características descrevendo cada exemplo (HUA et al., 2005). Conseqüentemente, para um determinado problema, sempre existe uma quantidade ideal de características para um determinado número de exemplos onde a taxa de erro é a menor possível. A figura 4.2 mostra o fenômeno do pico, e a linha preta representa os pontos com menor taxa de erro para uma determinada combinação de características e exemplos.

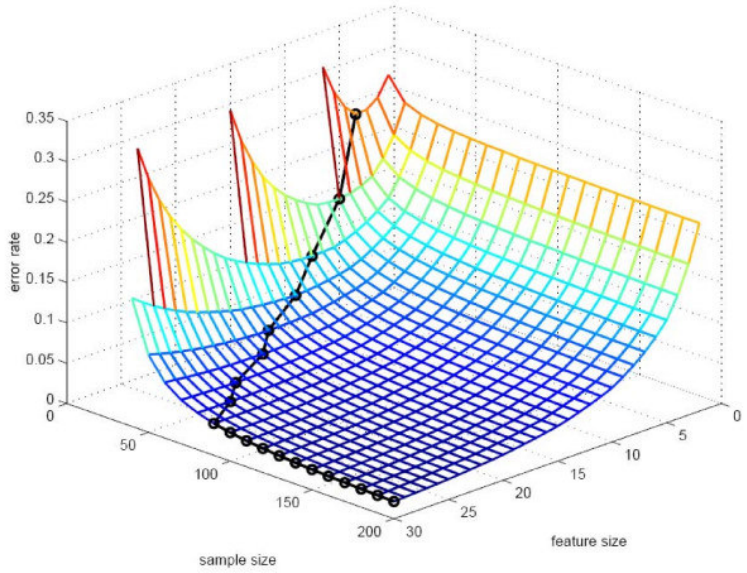

Figura 4.2: Fenômeno do Pico

Além disso, principalmente em espaços de entradas de alta dimensionalidade ou em problemas onde a relação entre a quantidade de exemplos e a quantidade de características não seja a recomendada, a investigação da dimensionalidade ideal geralmente é um fator muito importante na otimização do classificador ou regressor. A maneira de fazer esta investigação seria treinar e testar diversos tamanhos de subconjuntos de características a fim de identificar qual tamanho minimiza o erro do classificador ou regressor. Esta atividade deve ser realizada utilizando as técnicas de RDD, que são descritas em mais detalhes na seção a seguir.

4.3 Classificação das técnicas de RDD

A redução de dimensionalidade dos dados (RDD) é um processo que visa encontrar uma estrutura mais compacta de representação dos dados através do mapeamento de cada exemplo para um vetor de menor dimensão. Além disso, a RDD não deve resultar em perda de informação relevante em relação aos dados originais, ou pelo menos, os

benefícios obtidos com a RDD devem ser maiores que o prejuízo da perda de informação.

Sob um alto nível de abstração, as técnicas de RDD geralmente são aplicadas com algum dos seguintes objetivos:

- Visualização de dados de alta dimensão: a necessidade de visualização de dados de alta dimensão é uma grande necessidade atualmente em diversas áreas do conhecimento humano. Neste caso a redução da quantidade de dimensões dos dados permite projetá-los em espaços bi ou tridimensionais de forma a visualizá-los na tela do computador.
- Compressão de dados: reduzir a quantidade de dimensões dos dados implica em uma menor necessidade de espaço de armazenamento e transmissão mais rápida ou com menor largura de banda.
- Pré-processamento para mineração de dados: a aplicação de técnicas de RDD antes da análise de dados geralmente aumenta a eficiência de processos de classificação e regressão automáticos. Além disso, o desempenho de tais processos também é normalmente incrementado devido ao problema chamado de maldição da dimensionalidade.

A análise de dados reais pode conter centenas de características, sendo que muitas delas são irrelevantes para a mineração de dados (HAN e KAMBER, 2001). Apesar de ser possível o especialista do domínio selecionar as características que ele julga mais informativas, esta tarefa geralmente demanda um grande consumo de tempo. principalmente no caso dos dados não serem conhecidos. Por outro lado, se a área de pesquisa é inovadora, provavelmente não haja especialistas do domínio e nenhum conhecimento prévio poderá ser explorado a fim de selecionar as características mais informativas.

Conforme Cios et al. (2007), especificamente na área de descoberta de conhecimento em bancos de dados, as técnicas de RDD realizam principalmente as seguintes atividades:

- Remover redundâncias nos dados.
- Obter padrões transformados e reduzidos contendo apenas conjuntos relevantes de características que ajudam a projetar classificadores com melhores capacidades de generalização.
- Descobrir variáveis intrínsecas dos dados que ajudem o projeto de um modelo dos dados, e melhorar a compreensão do fenômeno que gera os padrões.
- Projetar dados com alta dimensão, preservando a topologia intrínseca aos dados, em um espaço de baixa dimensão, a fim de facilitar a descoberta de relacionamentos nos dados.

Neste escopo, o resultado prático da aplicação de técnicas de RDD é uma redução do espaço de busca de hipóteses, com a consequente melhora do desempenho e simplificação dos resultados do processo de mineração de dados (WANG e XIUJU, 2005).

A RDD é especialmente útil quando há uma grande quantidade de características descrevendo cada exemplo no banco de dados, fato peculiar aos bancos de dados

científicos. Nestes casos, a quantidade de exemplos necessários para adaptar um modelo multivariado cresce exponencialmente em relação à quantidade de características. Porém, muitas vezes, a obtenção de mais exemplos é difícil devido à grande dificuldade ou ao grande custo deste processo. Além disso, o uso de muitas variáveis no modelo preditivo pode dificultar a interpretação da análise e viola o princípio da parcimônia. Outro fator importante é que muitas variáveis podem mais facilmente conduzir a uma superadaptação do modelo preditivo (LAROSE, 2006).

Embora os algoritmos de mineração de dados já implementem internamente técnicas de RDD, eles geralmente pecam no quesito escalabilidade (YE, 2003). Desta forma, a aplicação de técnicas específicas de RDD em combinação com os algoritmos de mineração geralmente conduz a melhores resultados.

As técnicas de RDD pode ser divididas em três categorias: extração de características, construção de características e seleção de subconjunto de características. Apesar da divisão didática, tanto a extração de características quanto a construção de características geralmente são sucedidas pela seleção. Isto ocorre porque tanto a extração quanto a construção criam novas características.

4.3.1 Extração de características

O processo de extração de características visa extrair um conjunto de características novas a partir do conjunto de características originais através de algum mapeamento funcional (YE, 2003). De uma maneira mais formal, a extração de características pode ser definida da seguinte forma: tem-se um conjunto C de n características originais tal que $C = \{c_1, c_2, ..., c_n\}$, e após o processo de extração de características, será gerado um novo conjunto de características D, com m características, tal que $D = \{d_1, d_2, ..., d_m\}$ e $m < n$. Também tem-se que $d_i = F_i(c_j, c_k, ..., c_l)$, onde F é uma função de mapeamento, d_i é a nova característica extraída, e c_j, c_k, c_l fazem parte do conjunto de características originais. O mapeamento funcional é realizado mediante uma transformação linear ou não linear sobre as características originais.

O objetivo principal da extração de características é encontrar um conjunto mínimo de novas características que obedeça alguma medida de desempenho. Para atingir este objetivo geralmente faz-se necessária uma busca intensiva, que naturalmente será demorada e com alto custo computacional. Além disso, a definição de uma medida de desempenho também é uma atividade muito complexa. A finalidade da medida de desempenho é avaliar se uma característica construída é boa ou não.

Dentre os problemas oriundos da aplicação de tais técnicas podem ser salientados:

- Este processo consome muito tempo pela necessidade de pesquisar novas características que satisfaçam o critério de desempenho. Desta forma deve ser analisada a relação custo x benefício entre o tempo gasto e a otimização obtida no processo de classificação ou regressão.
- As características originais devem ser mantidas, o que significa que a aplicação desta técnica conduz a um aumento de dimensionalidade dos dados. Tal problema faz com que a aplicação posterior de alguma técnica de seleção de características seja necessária, de forma que seja efetivamente reduzida a dimensionalidade dos dados de entrada.
- O modelo de classificação ou regressão gerado a partir de características extraídas é de mais difícil compreensão. A dificuldade de compreensão dá-se

pelo fato de que o processo que deu origem a estas novas características extraídas, pode não ser bem conhecido.

Os algoritmos de extração de características podem ser classificados em termos de tipo de transformação: linear e não linear; e em termos do tipo de aprendizado: supervisionado ou não supervisionado. A tabela 4.1 apresenta algumas técnicas de extração de características e suas respectivas classificações.

Tabela 4.1: Exemplos de técnicas de extração de características

		Tipo de transformação	
		Linear	Não Linear
Natureza do Aprendizado	Não Supervisionado	- Análise de Componentes Principais (ACP) -Análise de Componentes Independentes (ACI)	- Análise de Componentes Principais Não Linear
	Supervisionado	- Análise de Discriminantes Lineares (ADL) - RNAs de camada única	- RNAs multicamadas

Em problemas de natureza não linear, a extração de características frequentemente envolve a aplicação de transformações não lineares. Estes métodos de transformação não lineares são eficientes na aproximação de funções e robustos no tratamento de problemas reais não lineares. A extração de características, por criar um novo conjunto de características, dificulta a compreensão dos resultados obtidos (WANG e XIUJU, 2005).

4.3.2 Construção de características

A construção de características é um processo que visa descobrir informação omitida sobre os relacionamentos entre as características originais e aumentar o espaço de características através da inferência ou criação de características adicionais (YE, 2003). De uma maneira mais formal, a construção de características pode ser definida da seguinte forma: tem-se um conjunto C de n características originais tal que $C = \{c_1, c_2, ..., c_n\}$, e após o processo de construção de características, poderá ser gerado um novo conjunto de m características adicionais $c_{n+1}, c_{n+2}, ..., c_{n+m}$.

De uma forma geral, a construção de características visa descobrir novas características que simplifiquem ao máximo o modelo gerado. Alternativamente, também é possível aplicar a construção de características para criar modelos que tenham uma maior precisão, ao invés de uma maior simplicidade.

As várias abordagens para construção de características podem ser divididas em quatro classes: orientada a dados, orientada a hipóteses, baseadas em conhecimento e abordagens híbridas. A abordagem orientada a dados constrói novas características através da análise das características já existentes e da aplicação de operadores. A abordagem orientada a hipóteses constrói novas características através de hipóteses

geradas previamente. Estas hipóteses podem ser geradas por alguma outra técnica de aprendizado indutivo, tal como árvores de indução ou regras de associação. Abordagens baseadas em conhecimento constroem novas características através da aplicação de conhecimento já existente sobre o problema, geralmente obtido através do especialista do domínio. Abordagens híbridas utilizam uma combinação das abordagens previamente citadas.

Os operadores, citados previamente, assumem um papel fundamental não somente na abordagem orientada a dados, mas também nas demais abordagens de construção de características. Existe uma quantidade muito grande de operadores, e eles são classificados de acordo com o tipo de dado a que serão aplicados. Os operadores mais comuns aplicados a características nominais são: conjunção, disjunção, negação, condicional (se-então) e bicondicional (se-e-somente-se). Já os operadores mais comuns para características numéricas são os operadores algébricos básicos, tais como: adição, subtração, multiplicação, divisão; os operadores relacionais, tais como: igual, diferente, maior, menor; e as funções de agregação, tais como: máximo, mínimo, soma e média.

Existe uma grande quantidade de operadores que podem ser utilizados na construção de atributos. Aliado a isso, pode haver também uma grande quantidade de características de entrada. A explosão combinatorial causada pelas possíveis combinações entre características e operadores torna a construção de características uma tarefa extremamente difícil. Isto faz com que a busca exaustiva pelo espaço de características construtíveis provavelmente torne-se proibitiva. O desenvolvimento de abordagens que possam explorar este espaço de forma inteligente e eficiente é uma necessidade premente, já que a carência por tais abordagens é notória.

Dada a mecânica do processo de construção de características, comentadas no parágrafo anterior, é possível neste processo a geração de uma grande quantidade de novas características. Porém, dentre as características construídas, algumas devem melhorar o desempenho do modelo de classificação ou regressão, e outras não. Por este motivo, existe a necessidade da identificação de quais características construídas devem efetivamente ser adicionadas ao modelo. Esta identificação deve ser realizada com base em alguma métrica de avaliação das novas características. Esta métrica deve estar relacionada com a finalidade do modelo, seja ele de regressão ou de classificação.

Dentre as técnicas atualmente utilizadas para construção de características podem ser citados os algoritmos genéticos, como exemplo de uma abordagem orientada a dados, e o uso de árvores de decisão e regras de associação, como abordagens orientadas a hipóteses. Por outro lado a aplicação de abordagens baseadas em conhecimento, que utilizam algum conhecimento prévio do domínio, provavelmente apresente melhores resultados. Porém esta abordagem nem sempre é passível de utilização.

A construção de características, assim como ocorre com a extração, também deve manter as características originais, fazendo com que ocorra um aumento da dimensionalidade dos dados de entrada. Assim sendo, faz-se necessária a aplicação posterior de alguma técnica de seleção de características, de forma que seja efetivamente reduzida a dimensionalidade dos dados de entrada.

A seleção de subconjunto de características, que é a terceira categoria de técnicas de redução de dimensionalidade, é comentada na próxima seção.

4.4 Seleção de características: fundamentos e estado da arte

O problema de seleção de características pode ser definido como o processo de encontrar um conjunto relevante de M características dentre as N características originais, onde $M \leq N$, para definir os dados a fim de maximizar a exatidão preditiva do modelo (LIU e SETIONO, 1996). A seleção das características que apresentam uma maior diferença entre as classes afeta decisivamente o desempenho do classificador. Da mesma forma, no caso de regressores, a seleção das variáveis mais representativas também conduziria a um melhor desempenho. Tais fatos fazem com que a seleção de características seja um problema chave no processo de reconhecimento de padrões (FUKUNAGA, 1990).

Se toda a informação necessária para a criação do modelo é fornecida, pode parecer que a escolha de um subconjunto ótimo de características de entrada não é uma tarefa crítica. Porém, uma correta adequação da dimensão dos dados de entrada, com a conseqüente redução da quantidade de características de entrada, pode conduzir a uma significante melhora na qualidade do modelo e no tempo de treinamento (SARKER et al., 2002). Embora os algoritmos de mineração de dados já apliquem internamente a seleção das características mais informativas, ignorando as menos informativas, a utilização de técnicas específicas para seleção de características além de melhorarem o desempenho destes algoritmos, também permite uma melhor escalabilidade (WITTEN e FRANK, 2005).

Segundo Ye (2003), os objetivos da SSC em aprendizado de máquina são: 1) Reduzir a dimensionalidade do espaço de características; 2) Acelerar o aprendizado dos algoritmos de mineração de dados; 3) Melhorar a capacidade preditiva dos algoritmos; e 4) Melhorar a compreensibilidade dos resultados obtidos.

4.4.1 Os sub-processos da SSC

Sob um alto nível de abstração, o processo de SSC pode ser visualizado como uma busca em um espaço de estados. O processo de SSC pode ser resumido em 4 sub-processos bem definidos, sendo eles:

- Seleção do ponto de partida.
- Seleção da função de avaliação.
- Seleção da estratégia de busca.
- Seleção do critério de parada.

Para cada um destes 4 sub-processos, existem diversas alternativas possíveis de solução. A combinação destas diversas alternativas gera uma enorme gama de abordagens para realizar a seleção de subconjunto de características.

4.4.2 Seleção de ponto de partida

Para dados de entrada descritos por n características, há potencialmente 2^n possibilidades de pontos de partida para o processo de seleção de características. Há três alternativas mais comumente usadas como ponto de partida, são elas: conjunto com todas as características, conjunto vazio, ou um conjunto selecionado aleatoriamente. Além destas alternativas, também pode ser selecionado algum conjunto que atenda alguma restrição específica, por exemplo, o conjunto com as k características mais

relevantes. Porém, neste caso, ainda seria necessário definir o parâmetro *k* e definir a função de avaliação de relevância.

4.4.3 Seleção da Função de Avaliação

Várias formas de avaliar a evolução do processo de seleção de características são propostas na literatura. De acordo com seu foco de aplicação, as funções de avaliação são divididas em dois grandes grupos: as funções de critérios independentes e as funções de critérios dependentes.

4.4.3.1 Critérios independentes

Os critérios independentes visam avaliar a qualidade preditiva individual das características de entrada em relação à característica de saída. Estes algoritmos frequentemente geram como saída uma lista ordenada das características de entrada, sem preocupação em definir o conjunto mínimo de características a ser utilizado. Em razão de sua forma de funcionamento, estes algoritmos também são chamados de algoritmos de ordenamento de características.

Adicionalmente, os critérios independentes podem ser sub-divididos em: métricas de distância, métricas de teoria da informação, métricas de dependência e métricas de consistência.

Métricas de Distância

As métricas de distância são também referenciadas como métricas de separabilidade, de divergência ou de discriminação. Entre estas métricas podem ser citadas: distância euclidiana, distância euclidiana com pesos, chebyshev, city block ou manhattan e mahalanobis (YAMPOLSKIY e GOVINDARAJU, 2006).

- Distância Euclidiana: é uma das métricas de distância mais populares. A distância euclidiana entre dois vetores pode assumir valores a partir de 0. O valor 0 indica que os vetores são idênticos. A distância euclidiana é definida pela raiz quadrada do somatório das diferenças entre dois vetores *X* e *Y*, onde *n* é o tamanho dos vetores.

$$d_E = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- Distância Euclidiana com pesos: é uma métrica que aproveita o conhecimento de um especialista do domínio para melhorar o resultado da distância euclidiana padrão. Esta métrica permite ao especialista do domínio determinar pesos a cada uma das características de entrada, de forma que características com mais informação discriminatória possam ter pesos maiores. Como aspecto negativo, esta métrica tem uma alta dependência da qualidade das decisões, muitas vezes empíricas, do especialista do domínio.

- Distância de Chebyshev: é uma métrica que define a distância entre dois vetores como sendo a maior distância entre os pares de elementos dos vetores. Esta métrica é definida por:

$$D_C = \max_i \left(|x_i - y_i| \right)$$

- Distância Manhattan: é uma métrica baseada na soma dos tamanhos das projeções do segmento de linha entre os pontos no eixo das coordenadas. Desta forma, a métrica retorna a soma das diferenças absolutas de dois vetores. A distância Manhattan é calculada da seguinte forma:

$$d_M = \sum_{i=1}^{n} |x_i - y_i|$$

- Distância de Mahalanobis: é uma métrica baseada na correlação entre vetores, pela qual padrões podem ser identificados e analisados. Esta métrica pode ser definida como distância de dissimilaridade entre dois vetores aleatórios *X* e *Y*, da mesma distribuição, com a matriz de covariância *S*

$$d_M(x) = \sqrt{(x-u)^T S^{-1} (x-u)}$$

Métricas de teoria da informação

As métricas de teoria da informação determinam o ganho de informação de uma característica. Existem diversas métricas de ordenação de importância de características que utilizam conceitos de teoria da informação. Muitas destas características são baseadas em estimativas empíricas da informação mútua entre cada uma das características de entrada e a característica de saída. As formas de cálculo da informação mútua podem ser de duas classes distintas: funções baseadas em distribuição de probabilidade, aplicáveis a características com valores discretos, e funções baseadas em densidade de probabilidade, aplicáveis a características com valores contínuos. As funções baseadas em distribuição de probabilidade são bem mais difundidas e mais simples de serem aplicadas. Já as funções baseadas em densidade de probabilidade, por sua complexidade, não são tão utilizadas. Quando as características são contínuas, comumente elas passam por um processo de discretização, sendo posteriormente aplicadas as funções baseadas em distribuição de probabilidade (COVER e THOMAS, 2006).

Entre as métricas baseadas em distribuição de probabilidade podem ser citadas: entropia, entropia conjunta, entropia condicional, informação mútua e ganho de informação, também chamado de entropia relativa ou divergência Kullback-Leibler (MACKEY, 2003).

- Entropia: dada uma característica X, que assuma valores aleatórios, a sua entropia irá quantificar a incerteza intrínseca aos valores assumidos por esta característica. A entropia de uma característica X é dada por:

$$H(X) = -\sum_{x} p_x \log_2(p_x)$$

- Entropia conjunta, que pode ser utilizada para calcular quanta entropia existe entre duas características *X* e *Y*, cujos valores sejam discretos. A entropia conjunta é dada por:

$$H(X,Y) = -\sum_{x,y} p_{x,y} \log_2(p_{x,y})$$

- Entropia condicional, que quantifica a entropia de uma característica de saída *Y*, dada uma característica de entrada *X*.

$$H(Y \mid X) = H(Y,X) - H(X)$$

- Informação mútua: que permite medir a quantidade de informação que pode ser obtida sobre uma característica, com base na observação de outra característica. A informação mútua é dada por:

$$MI(X,Y) = H(X) + H(Y) - H(X \mid Y)$$

- Ganho de informação: esta métrica permite calcular a medida de divergência entre duas características, sejam elas discretas ou contínuas. O ganho de informação é dado por:

$$D_{KL}(p(X) \parallel q(X)) = \sum_{x \in x} p(x) \log \frac{p(x)}{q(x)}$$

Entre as métricas baseadas em densidade de probabilidade pode ser citada a entropia diferencial. A entropia diferencial é dada por:

$$h(X) = -\int_x f(x) \log f(x) dx$$

Métricas de dependência ou correlação:

Este conjunto de métricas permite quantificar o quanto a variação do valor de uma característica pode ser predito através do valor de outra característica. Existem diversas métricas de dependência ou correlação, estando elas divididas em dois grandes grupos: métricas lineares e não lineares.

A métrica de correlação linear mais comum é o coeficiente de correlação de Pearson. Este coeficiente pode ser considerado a mais simples abordagem para a filtragem de características relevantes, sendo amplamente difundido principalmente na área da estatística (DALGAARD, 2002). Considerando-se a predição de uma característica *y*, em função de uma característica de entrada *x*, o coeficiente de correlação de Pearson é definido como:

$$R(i) = \frac{\mathrm{cov}(X,Y)}{\sqrt{\mathrm{var}(X)\,\mathrm{var}(Y)}}$$

onde *X* é o vetor de entrada, *Y* é o vetor com os valores de saída, *cov* significa a covariância e *var* significa a variância das características. O coeficiente de correlação é definido apenas se o desvio padrão de ambas as características é finito e diferente de zero. O valor de $R(i)$ pode assumir valores no intervalo [-1, 1]. O valor 1 indica que as variáveis têm uma correlação direta, e o valor -1 indica que as variáveis têm uma correlação inversa. Valor 0 indica que as variáveis são totalmente independentes. Valores de $R(i)$ entre estes extremos indicam o grau de relacionamento entre as variáveis, ou seja, o quando da variância total da característica *y* que é explicada pela relação entre *x* e *y*. Além disso, um valor absoluto de correlação pode ser fornecido pelo uso de $R(i)^2$, de forma a permitir a utilização de uma ordenação da importância das características de entrada na predição do valor da característica de saída.

Além desta métrica linear, na literatura também são propostas algumas outras que são extensões do coeficiente de correlação de Pearson para o caso específico de tarefas de classificação onde existam apenas duas classes. Dentre estas extensões pode ser citados o critério de Fischer.

O critério de Fisher, ou discriminante linear de Fisher, é um método de redução de dimensionalidade que projeta dados de alta dimensão em um espaço unidimensional.

Este processo de projeção maximiza a distância entre as médias das duas classes e minimiza a variância dentro cada classe (DALGAARD, 2002). O critério de Fisher pode ser definido como:

$$J(w) = \frac{|m_1 - m_2|^2}{s_1^2 + s_2^2}$$

Onde J representa o processo de projeção, m_1 *e* m_2 representam as médias das classes 1 e 2, e s_1^2 e s_2^2 representam as variâncias das classes 1 e 2.

Já o coeficiente de *Gini* é uma medida de dispersão estatística, muito difundida para calcular desigualdades de distribuição de renda ou de riqueza. O coeficiente de *Gini* pode assumir valores no intervalo entre 0 e 1. O valor 0 indica a igualdade perfeita entre as distribuições, já o valor 1 indica a desigualdade perfeita.

Tendo-se que n é o número de elementos e μ é o tamanho médio dos conjuntos, o coeficiente de *Gini* é dado por:

$$G = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} | x_i - x_j |}{2n^2\mu}$$

Como mencionado anteriormente, os critérios de correlação linear podem detectar apenas dependências lineares entre as características. Uma das formas mais simples de diminuir esta restrição é a realização de uma adaptação não linear das variáveis antes da utilização do critério de correlação. Entre as possíveis adaptações não lineares podem ser citadas: potenciação, radiciação, transformações logarítmicas e transformação inversa (GUYON, 2003).

Quando o problema a ser tratado é reconhecidamente não linear, ou quando as técnicas lineares não têm sucesso em identificar as características mais relevantes, podem ser utilizadas as métricas não lineares, tais como: o chi-quadrado, e os coeficientes de Spearman, Kendall e Goodman-Kruskal.

O coeficiente de correlação de Spearman é uma medida de correlação não paramétrica que não faz suposições sobre a distribuição de freqüência das variáveis. diferentemente do coeficiente de Pearson, não requer que a relação entre as variáveis seja linear (DALGAARD, 2002).

Supondo-se que d_i é a diferença entre cada valor correspondente de x e y, e n é o número de valores dos vetores, o coeficiente de Spearman é dado por:

$$\rho = 1 - \frac{\sum d_i^2}{n(n^2 - 1)}$$

O critério de correlação de Kendall é um método não paramétrico usado para medir o grau de correspondência entre duas listas ordenadas e avaliar o grau de significância desta correspondência (DALGAARD, 2002). Tendo-se que n_c é o número de pares concordantes e n_d é o número de pares discordantes, o coeficiente de kendall pode ser definido por:

$$\tau = \frac{n_c - n_d}{\frac{1}{2} n(n-1)}$$

O coeficiente de Kendall pode assumir valores no intervalo [-1,1]. O valor -1 é obtido pelo total desacordo entre os vetores e o valor 1 é obtido pela similaridade total entre os vetores. O valor 0 é obtido pela independência completa entre os vetores. Valores entre estes extremos indicam um maior ou menor desacordo entre os vetores.

Assim como o coeficiente de Kendall, o coeficiente de Goodman/Kruskal também é uma métrica de correlação estatística que visa definir o grau de correspondência entre duas listas ordenadas. O cálculo do coeficiente também é realizado pela análise dos pares em vetores e verificação se eles são congruentes ou discordantes. Este coeficiente é dado pela diferença entre a probabilidade de obter-se um par concordante e de obter-se um par discordante. O coeficiente de Goodman/Kruskal é dado por:

$$\lambda = \frac{n_c - n_d}{n_c + n_d}$$

Métricas de consistência

As métricas de consistência têm características diferentes das métricas tratadas anteriormente. Dentre elas pode ser citada a métrica *Min-Feature bias*, utilizada pelo algoritmo *Focus* (FOUNTAIN et al., 1991). Este algoritmo realiza uma busca exaustiva no espaço de estados de características a fim de encontrar um conjunto mínimo de atributos que seja suficiente para descrever a classe de todos os exemplos de treinamento. Como restrição a utilização deste algoritmo, tem-se o fato dele ter sido proposto para domínios booleanos sem ruído.

Outro algoritmo que se enquadra na categoria de métricas de consistência é o algoritmo *Relief*. Este algoritmo foi desenvolvido por Kira e Rendell (1992) e possui uma função de avaliação de características mais complexa que o algoritmo *Focus*. *Relief* é eficiente para a estimação da qualidade dos atributos a partir de dependências encontradas entre eles.

O algoritmo *Relief* original pode tratar características tanto discretas quanto contínuas. Por outro lado, ele pode ser aplicado apenas a problemas de classificação com somente duas classes. Quando aplicado a características discretas, o algoritmo retorna 1 se os valores são diferentes, ou 0, se os valores são iguais. Quando aplicado a características contínuas, o algoritmo retorna a diferença normalizada no intervalo [0,1].

O algoritmo *Relief* pode ter seu desempenho fortemente afetado por dados redundantes e ruidosos, tornando seus resultados pouco confiáveis. Para superar esta restrição, foi proposta uma extensão deste algoritmo, chamada *Relief-A* que consegue tratar dados com ruído e dados omitidos.

A extensão *Relief-D* permite a utilização desta abordagem em problemas de classificação com mais de duas classes.

Outra extensão do algoritmo original é a *Relief-F*, proposta por Kononenko (1994). *Relief-F* pode tratar problemas de regressão e também permite o tratamento de problemas com valores omitidos.

4.4.3.2 Critérios dependentes

Os critérios dependentes visam avaliar a qualidade preditiva de um conjunto de características de entrada em relação à característica de saída. Algoritmos inseridos neste grupo buscam gerar como resultado o subconjunto mínimo de características de entrada, sem ter nenhuma preocupação sobre a relevância individual das características. Como resultado prático, as características que estão no subconjunto mínimo são consideradas relevantes, e todas as demais características, irrelevantes (KANTARDZIC, 2002).

Porém, antes de avaliar a qualidade do subconjunto de características, outra decisão de extrema importância é a seleção da estratégia de busca que será utilizada para explorar o espaço de busca de subconjuntos. Este assunto é discutido na próxima seção.

4.4.4 Seleção da estratégia de busca

No caso da função de avaliação ser aplicada a um conjunto de características e não a todas as características individualmente, a seleção da estratégia de busca torna-se uma decisão necessária. Existem diversas alternativas para estratégias de busca, que são divididas em três grupos distintos: completa, heurística e não determinística.

A tarefa de seleção de subconjunto de características pode ser vista sob o ponto de vista de uma busca no super-conjunto das possíveis soluções para o problema. Dado um conjunto de dados com n características de entrada, o super-conjunto das possíveis soluções para o problema seria composto de todas as combinações possíveis de atributos, ou seja, 2^n - 1 possibilidades.

A busca pode ser realizada de três formas distintas: I) para frente: partindo de um conjunto mínimo, acrescentando-se características a cada passo; II) para trás: partindo de um conjunto máximo, eliminando-se características a cada passo; III) bidirecional: partindo-se de um conjunto de tamanho médio, acrescentando ou eliminando-se características a cada passo.

4.4.4.1 Estratégias de busca completa

Estratégias enquadradas nesta categoria são consideradas completas pois garantem encontrar uma solução para o problema, caso esta solução exista. Também são consideradas ótimas pois garantem encontrar a melhor solução, quando há diversas soluções diferentes. Como inconveniente, estas estratégias geralmente necessitam pesquisar todo o espaço de busca a fim de encontrar o melhor subconjunto de características. Assim, se há n características, devem ser gerados 2^n subconjuntos. Porém, a complexidade de tempo e de espaço para encontrar a melhor solução pode tornar a aplicação desta estratégia proibitiva caso a quantidade de características seja muito grande. Caso esta estratégia encontre várias soluções possíveis, ou seja, vários subconjuntos com a melhor avaliação, o subconjunto escolhido será aquele com menor quantidade de características (LUGER e STUBBLEFIELD, 1998).

Estratégias completas não necessariamente são exaustivas. Se o critério de avaliação possuir determinadas propriedades, tais como monotonicidade, é possível encontrar o melhor subconjunto de características sem avaliar todo o espaço de busca.

4.4.4.2 Estratégias de busca heurística

Como, muitas vezes, vasculhar todo o espaço de busca pode ser inviável, podem ser aplicadas algumas formas alternativas para pesquisar seletivamente o espaço de busca. Estas formas são conhecidas como heurísticas. Pretensamente, uma heurística irá guiar a busca, segundo algumas restrições, e terá uma alta probabilidade de sucesso na busca de melhor solução para o problema (LUGER e STUBBLEFIELD, 1998).

Dentre as estratégias de busca mais utilizadas para seleção de subconjuntos de características a estratégia heurística é a mais utilizada. Estas estratégias utilizam alguma abordagem para vasculhar o espaço de características e encontrar o melhor subconjunto.

A estratégia da heurística geralmente pode estar implementada na forma de um algoritmo de aprendizado, tais como algoritmos genéticos ou redes neurais, ou pode ser uma regra que auxilie a busca pelo melhor subconjunto de características, tais como entropia ou análise de componentes principais.

As três primeiras técnicas abordadas acima mencionam as expressões "melhor característica" e "pior característica". A determinação de quais são a melhor e a pior característica em um determinado passo do algoritmo geralmente é determinada pela análise da sensitividade das características. A análise de sensitividade permite definir uma ordenação das características de entrada de acordo com sua importância em relação à predição da característica de saída. Para realizar a análise de sensitividade, deve-se criar um modelo com todas as características de entrada e calcular a sua precisão de acordo com a métrica desejada. A partir daí, deve ser eliminada uma característica de entrada e calculada novamente a precisão do modelo. A diferença de precisão entre o modelo com todas as características e entrada e o modelo com uma característica de entrada a menos, irá determinar a importância da característica eliminada. Este processo deve ser repetido com todas as características de entrada, e a partir daí será possível criar a ordenação da importância das características.

4.4.4.3 Estratégias não-determinísticas

As estratégias não-determinísticas são aplicáveis a problemas de otimização. O ponto positivo destas estratégias é sua capacidade de superar as limitações dos métodos determinísticos em muitos problemas de otimização, principalmente quando o objetivo admite um grande número de soluções sub-ótimas. O principal ponto negativo dos métodos não-determinísticos é que eles são computacionalmente caros e, em conseqüência, mais lentos que métodos clássicos de otimização.

Os métodos não-determinísticos geram conjuntos de soluções candidatas para o problema e seu propósito é convergir probabilisticamente a candidatos que maximizem a função objetivo.

Dentre as estratégias de busca não-determinísticas podem ser citados: subida de encosta, recozimento simulado e algoritmos genéticos.

Subida de Encosta (Hill Climbing)

A abordagem de subida de encosta é uma técnica de otimização de busca local que utiliza um procedimento de melhora iterativa.

O algoritmo inicia selecionando uma solução aleatória no espaço de busca. Tal solução inicial geralmente é ruim. Seguindo o princípio da perturbação mínima, durante

cada iteração do algoritmo, um novo ponto, vizinho do ponto atual, é selecionado. Se o novo ponto constitui-se em uma solução melhor para o problema que o ponto atual, o novo ponto torna-se o atual. Por outro lado, se o novo ponto constitui-se em uma solução pior que o ponto atual, o novo ponto é ignorado e é selecionado aleatoriamente um outro ponto vizinho do ponto atual. O método termina sua execução ao passar um determinado número de iterações sem obter-se nenhuma melhora na solução. Assim, o ponto atual é retornado como a melhor solução para o problema (MICHALEWICZ e FOGEL, 2000).

Esta classe de algoritmos é considerada de busca local porque permite que sejam encontrados valores localmente ótimos, além de o algoritmo ser altamente dependente da qualidade da solução inicial e do posicionamento da solução inicial em relação ao ótimo global. A existência de muitos ótimos locais dificulta que seja encontrado o ótimo global. Uma forma de diminuir o impacto destes problemas é fazer com que o algoritmo seja executado diversas vezes, com pontos iniciais em diferentes posições do espaço de busca. Espera-se que ao menos uma das soluções iniciais conduza ao ótimo global.

Há diversas variações do algoritmo original de subida de encosta. Estas variações diferem principalmente no modo pelo qual um novo ponto é selecionado para comparação com o ponto corrente. Dentre as variações do algoritmo original a mais eficiente é a subida de encosta pela trilha mais íngreme. Esta variação examina cada um dos possíveis vizinhos do ponto atual. O ponto adjacente que possuir a melhor avaliação é selecionado e comparado com o ponto atual. Caso o melhor ponto adjacente tenha melhor avaliação que o ponto atual, o adjacente torna-se o ponto atual. Caso contrário, o ponto atual é retornado como a solução do problema.

É esperado que a aplicação de algoritmos de subida de encosta encontre uma solução muito próxima da solução ótima, porém não é correto supor que este algoritmo irá encontrar a solução ótima.

Recozimento Simulado (Simulated Annealing)

O recozimento simulado constitui-se em uma classe de algoritmos que utilizam uma metaheurística para otimização baseada na metáfora de um processo térmico, utilizado na área de metalurgia, a fim de obter-se estados de baixa energia em sólidos. O recozimento simulado é dividido em duas etapas. Na primeira etapa, a temperatura do sólido é aumentada para um valor máximo na qual o sólido irá se fundir. Na segunda etapa, a temperatura do material é reduzida lentamente até que o material se solidifique. Durante a segunda fase, os átomos que compõe o sólido organizam-se em uma estrutura uniforme com energia mínima (LEE e EL-SHARKAWI, 2008).

A metaheurística proposta pela abordagem de recozimento simulado é muito parecida com o reinício aleatório aplicado à abordagem de subida de encosta. Partindo do princípio que algoritmos de busca locais geralmente irão retornar algum ótimo local, o recozimento simulado melhora estes algoritmos pela inserção de um reinício em alguma outra posição aleatória no espaço de busca. Desta forma, o recozimento simulado é implementado como um passo externo a qualquer algoritmo de busca local, permitindo um melhor desempenho desta classe de algoritmos.

O recozimento simulado obriga a definição prévia de um número máximo de reinícios como forma de impedir que o algoritmo continue a ser executado indefinidamente. A solução final do algoritmo é a solução com a melhor avaliação dentre as n soluções, dado que n é o número máximo de reinícios.

Algoritmos genéticos

Os algoritmos genéticos (AG) consistem em uma classe de algoritmos de otimização estocásticos inspirados nos princípios biológicos de genética e de seleção natural (HAUPT e HAUPT, 2004). Tais princípios fundamentam uma forma robusta de evolução bem sucedida de organismos, definindo uma heurística que permite a uma população, composta de muitos indivíduos, evoluir através da aplicação regras de seleção específicas. Esta evolução se dá para um estado que maximize uma função de adaptação. Tal função de adaptação, dado um indivíduo, retorna um valor contínuo que permite avaliar o nível de adaptação deste indivíduo ao ambiente no qual ele está inserido.

Segundo as regras de seleção natural, os organismos menos adaptados ao ambiente morrem, enquanto os que estão mais bem adaptados ao ambiente irão viver e reproduzir-se, transferindo suas características para seus descendentes através da herança genética. Cada nova geração estaria mais bem adaptada ao ambiente que a geração anterior. Ocasionalmente, mutações aleatórias podem ocorrer durante a reprodução, o que geralmente conduz a morte dos indivíduos mutados, mas também pode conduzir a novas espécies melhor adaptadas. Também pode ocorrer a recombinação, ou *crossover*, que faz com que durante o processo de reprodução, dois cromossomos sejam cortados em alguma posição randômica e suas partes cortadas sejam trocadas.

Outra possibilidade importante é a criação de soluções híbridas combinando métodos não-determinísticos e métodos tradicionais, tais como a perturbação aleatória. Estas abordagens combinam as vantagens de métodos determinísticos e não-determinísticos e aceleram a convergência dos algoritmos estocásticos.

4.4.4.4 Busca sequencial

Qualquer uma das estratégias de busca mencionadas anteriormente pode aplicar uma das seguintes técnicas de busca seqüencial:

- Seleção passo a passo para frente, ou incremento seqüencial: Procedimentos que aplicam esta técnica partem de um conjunto de características selecionadas (CCS) que estará vazio, e um conjunto de características originais (CCO), que conterá todas as características de entrada disponíveis. A partir daí, a melhor característica do CCO é adicionada ao CCS. A cada passo, a melhor das características restantes no CCO é adicionada ao CCS. O procedimento acaba quando o conjunto CCO estiver vazio, ou quando for atingido um critério de parada pré-estabelecido.
- Eliminação passo a passo para trás, ou poda seqüencial: O procedimento inicia com o CCS contendo todas as características de entrada disponíveis. A cada passo, a pior característica é removida do CCS. O procedimento acaba quando o CCS for vazio, ou for atingido o critério de parada pré-estabelecido.
- Combinação de seleção para frente e eliminação para trás: O procedimento combina as técnicas anteriores, iniciando a partir do CCS vazio, e do CCS com todas as características de entrada disponíveis. A cada passo do algoritmo a melhor característica do CCO é adicionada ao CCS, e a pior

característica do CCO é eliminada. O procedimento acaba quando o conjunto CCO estiver vazio ou quando for atingido o critério de parada.

4.4.5 Formas de funcionamento

Em relação à forma de funcionamento, os algoritmos de SSC geralmente são classificados, de acordo com a abordagem que utilizam, como filtros, *wrappers* ou embutidos (GUYON, 2006). Abordagens de filtro são utilizadas na fase de pré-processamento. Abordagens de *wrapper* funcionam como uma espécie de invólucro ao redor do algoritmo de indução. Diferentemente dos filtros, os *wrappers* funcionam intimamente ligados com alguma classe específica de algoritmos de aprendizado. O *wrapper* irá procurar bons subconjuntos de características e submete-los ao próprio algoritmo de aprendizado, que será utilizado como função de avaliação destes subconjuntos. O melhor subconjunto será aquele em que o algoritmo de aprendizado atingir a menor taxa de erro. Já as abordagens embutidas, modificam o algoritmo de aprendizado de forma a otimizar suas funções de SSC.

Apesar de terem uma classificação clara, estas diversas classes de algoritmos podem ser utilizadas em combinação. A seguir são detalhadas cada uma das formas de funcionamento dos algoritmos de SSC.

4.4.5.1 Abordagens de filtro

Antes da seleção de qual filtro poderá ser utilizado em um determinado problema, deve-se analisar a natureza das variáveis envolvidas. Com aplicabilidade em aprendizado supervisionado, que é o foco deste trabalho, as abordagens de filtro geralmente são divididas em três grandes grupos distintos: métricas de correlação, métricas baseadas em conceitos de tecnologia da informação e métricas bayesianas. Também existem algumas outras abordagens pontuais, que não estariam incluídas em nenhum destes grupos, tais como o *bootstrap* e algoritmo do vizinho mais próximo.

Abordagens de filtro fazem a SSC somente baseadas nas informações contidas nos dados, tais como a separabilidade interclasse. Ou seja, toda e qualquer métrica utilizada para avaliar a relevância de uma característica é calculada somente com base nos dados originais submetidos ao filtro. Em processos de mineração de dados, os filtros são utilizados na fase de pré-processamento, e por serem completamente independentes do algoritmo de mineração de dados que será utilizado, não recebem deste algoritmo nenhuma retro-alimentação sobre a qualidade das características que foram selecionadas no processo de filtragem. Basicamente, a função dos filtros é eliminar características que potencialmente terão pouca relevância na fase de mineração de dados (GUYON, 2006).

Justamente por serem aplicadas no pré-processamento, e consequentemente não incorporarem nenhuma relação com a tarefa de aprendizado, os filtros tornam-se menos custosos computacionalmente que algoritmos aplicando as outras abordagens. A independência que os filtros têm do algoritmo de mineração tem dois aspectos importantes. Se por um lado os filtros raramente permitem a obtenção dos melhores resultados por não explorarem as melhores capacidades dos algoritmos, por outro, um mesmo filtro pode ser utilizado em combinação com diversos algoritmos de mineração diferentes.

De uma maneira mais formal, um filtro pode ser definido da seguinte forma: tem-se um conjunto C de n características originais tal que $C = \{c_1, c_2, ..., c_n\}$, e tem-se um

conjunto E, com m exemplos, tal que $E = \{e_1, e_2, ..., e_m\}$. Um filtro pode ser definido como uma função f que retorna um valor de relevância $J(C_i|E)$ que estima, com base no conjunto de exemplos E, o nível de relevância de uma dada característica c_i. Tal tarefa geralmente será de classificação ou regressão. As m características que obtiverem um maior valor de relevância serão passadas ao algoritmo de mineração na forma de um conjunto $X_{opt} = \{x_1, x_2, ..., x_m\}$, onde $X \supseteq C$. Desta forma, o filtro gera como saída o conjunto de características X_{opt} ordenado pelas suas respectivas relevâncias da seguinte forma: $J(x_1) \leq J(x_2) ... \leq J(x_m)$. Já as características com menor valor de relevância serão filtradas e não serão repassadas ao algoritmo de mineração. Com a realização da filtragem, o algoritmo de treinamento não mais receberia o conjunto inicial de exemplos E, mas sim o conjunto E_{Xopt} que seria o conjunto original com a dimensionalidade reduzida de acordo com as características selecionadas pelo filtro. Por também realizarem uma ordenação das características originais, segundo alguma métrica de relevância, diz-se que alguns filtros realizam o processo de *feature ranking*.

A definição de um limiar para separar as características relevantes das não relevantes não é uma tarefa trivial, de forma que ainda pode ser necessária a utilização de um *wrapper* para definir este limiar de acordo com o algoritmo de mineração que será utilizado. A utilização de uma abordagem híbrida de filtro seria realizada de forma que fossem gerados n diferentes subconjuntos de características, cada um deles contendo as n características mais informativas, de forma que o primeiro subconjunto conteria apenas a característica mais informativa, o segundo subconjunto conteria as duas características mais informativas, e assim sucessivamente. A função do *wrapper* seria testar qual destes n subconjuntos gerados é o melhor.

Adicionalmente, os filtros ainda podem ser classificados em locais e globais. Os filtros globais avaliam as características levando em conta todos os dados disponíveis, independentemente de seu contexto. Neste caso, supondo uma tarefa de classificação, todos os dados de entrada seriam tratados igualmente, independentemente de suas respectivas classes. Já os filtros locais, no mesmo caso previamente citado, seriam aplicados diversas vezes a cada uma das classes do problema, analisando somente os exemplos pertencentes àquela classe específica. No caso da aplicação de filtros locais em tarefas de regressão, poderiam ser aplicados vários filtros diferentes, sendo um deles aplicado a uma faixa específica de valores de saída.

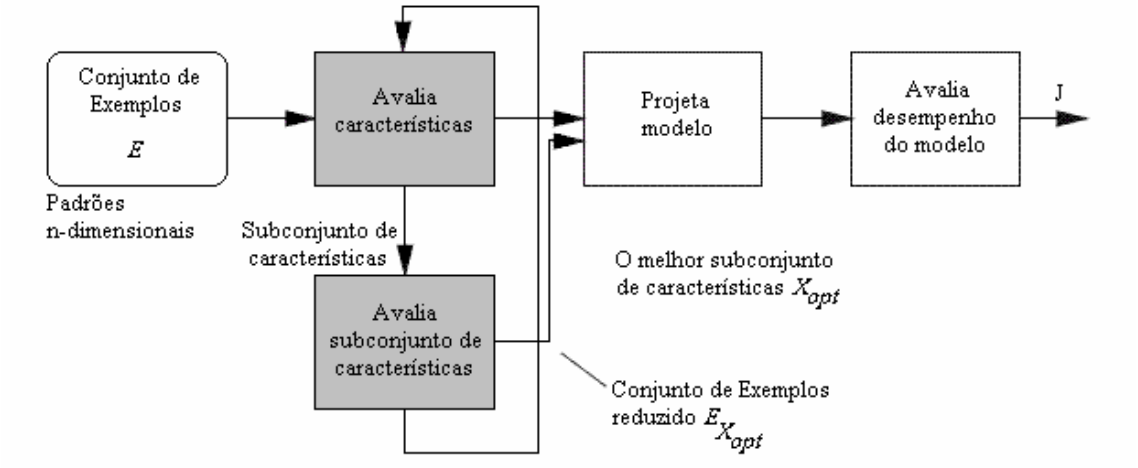

Figura 4.3: Abordagem de filtros

Adaptado de Cios et al. (2007)

A figura 4.3 representa a abstração de um filtro e sua interação com os demais processos típicos de um processo de mineração de dados. As caixas com fundo cinza representam os processos realizados por um filtro.

4.4.5.2 Abordagens de wrapper

A idéia de *wrapper* foi proposta originalmente por Kohavi e John (1997). Abordagens de *wrappers* determinam o quão bom é um subconjunto de características através da efetiva avaliação deste subconjunto pelo algoritmo de aprendizado. Desta forma, a partir de conjunto de dados de entrada podem ser gerados diversos subconjuntos. Cada um destes subconjuntos deve ser submetido ao algoritmo de mineração de dados, sendo executado um ou mais ciclos completos de treinamento/teste. O melhor subconjunto de características será aquele no qual o algoritmo de mineração obtiver uma maior exatidão preditiva. O método de avaliação dos subconjuntos geralmente é a validação cruzada. Alternativamente, para selecionar o melhor subconjunto de características, pode ser utilizada alguma outra métrica de desempenho, tais como as abordadas na seção 2.4.3.3.

A figura 4.4 representa a abstração de um *wrapper*, sendo que as caixas com fundo cinza representam o *wrapper*, as demais caixas representam as atividades convencionais de um processo de mineração de dados.

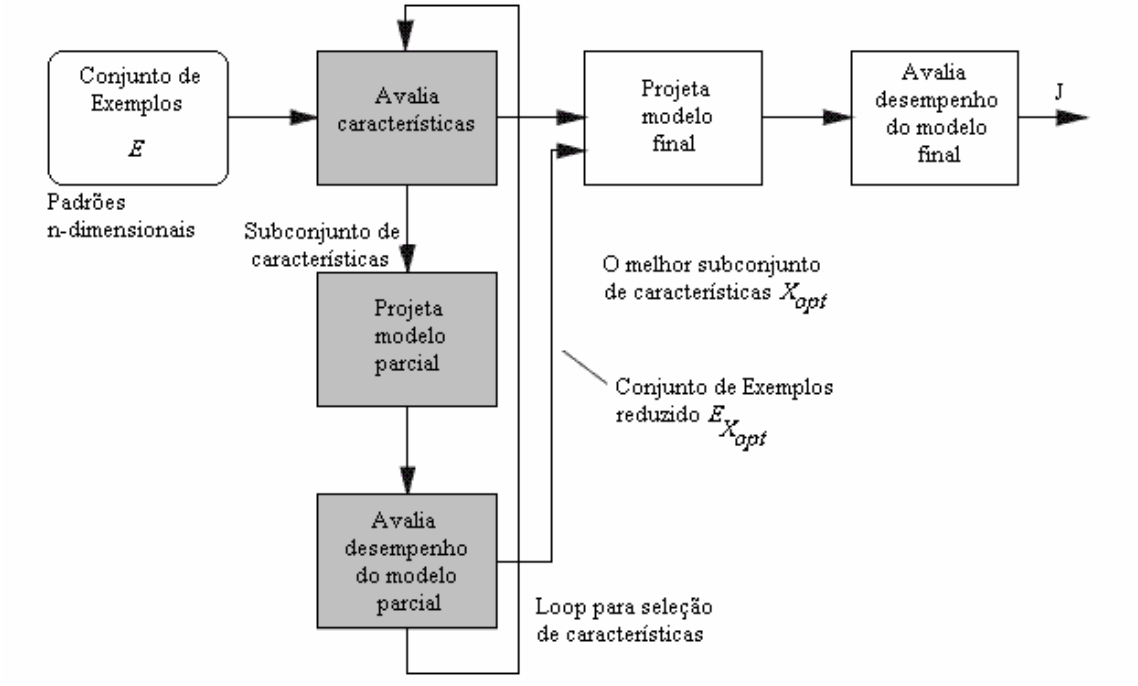

Figura 4.4: Abordagem de *wrappers*

Adaptado de Cios et al. (2007)

O cerne das abordagens de *wrappers* está na forma pela qual os subconjuntos são gerados e avaliados. Esta forma é definida através da estratégia de busca utilizada pelo *wrapper*. As estratégias de buscas utilizadas em abordagens *wrapper* podem ser classificadas em estratégias ótimas, estratégias de seleção seqüencial e estratégias estocásticas.

4.4.5.3 Abordagens embutidas

Abordagens embutidas de SSC estão inseridas dentro do algoritmo de aprendizado, ou seja, tem total interação com o aprendizado, diferentemente das abordagens de filtros e *wrappers*. Em virtude desta característica, não há a necessidade de criar novos processos dentro da estrutura de mineração de dados.

A figura 4.5 apresenta a estrutura básica da abordagem embutida.

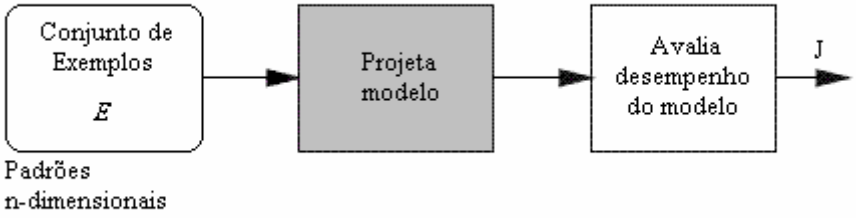

Figura 4.5: Abordagem embutida.

Algoritmos que utilizam abordagem embutida podem ser divididos em duas classes: algoritmos gulosos (*greedy*) e algoritmos preguiçosos (*lazy*). Os algoritmos gulosos substituem gulosamente os exemplos de treinamento pelo conceito que foi aprendido. Posteriormente, somente o conceito aprendido é utilizado para classificar novos exemplos. São exemplos métodos gulosos de abordagens embutidas os algoritmos ID3 (QUINLAN, 1986) e C4.5 (QUINLAN, 1993). Os algoritmos ID3 e C4.5 geram árvores de decisão. Estes algoritmos classificam instâncias ordenando-as da raiz da árvore em direção a suas folhas. As características mais relevantes posicionam-se mais perto da raiz da árvore. À medida que a relevância da característica diminui, esta característica é posicionada mais longe da raiz e mais próxima às folhas. Após o processo de aprendizado, pode ser executado algum processo de poda da árvore, que irá eliminar as características menos informativas que compõe a árvore (MITCHELL, 1997).

Por outro lado, algoritmos preguiçosos usam o conjunto de treinamento para predizer o comportamento de novas instâncias. Estes algoritmos são assim chamados porque atrasam o processamento até que uma nova instância necessite ser classificada. Os algoritmos preguiçosos possuem as seguintes características básicas (MITCHELL, 1997):

- O processo de aprendizado consiste simplesmente no armazenamento dos exemplos de treinamento. O processo de generalização é adiado, até que haja a necessidade de predição do comportamento de uma nova instância.
- São dirigidos pela demanda, ou seja, cada vez que uma nova instância é submetida, seu relacionamento com as instâncias de treinamento é analisado.
- Não armazenam a consulta construída e não armazenam resultados intermediários.

São exemplos de métodos preguiçosos de abordagens embutidas o algoritmo do *k*-ésimo vizinho mais próximo (kNN), proposto por Cover e Hart (1967) e o algoritmo de

raciocínio baseado em casos (CBR), proposto por Kolodner (1993), assim como suas respectivas variações.

Navot et al (2005) apresentam o algoritmo RGS (*Regression, Gradient guided, feature Selection*). Este algoritmo realiza a seleção de características de entrada baseado na técnica do *k*-ésimo vizinho mais próximo. O RGS pode ser utilizado como um filtro para outros algoritmos de regressão, ou como um *wrapper* para estimação pelo algoritmo kNN. O algoritmo utiliza uma versão do algoritmo do *k*-ésimo vizinho mais próximo que atribui pesos as características de entrada. O método captura dependências complexas da função alvo em relação a suas entradas e usa o erro *leave-one-out* como uma regularização natural. É não linear. Tem implementação e funcionamento relativamente simples.

Por outro lado, o algoritmo RGS tem algumas limitações, entre elas podem ser citadas as seguintes:

- Sua utilização não é apropriada para tarefas de classificação.
- Há a necessidade de definição de alguns parâmetros para o funcionamento do algoritmo, são eles:
  - k: número de vizinhos;
  - β: Fator de decaimento gaussiano;
  - T: número de iterações;
  - $\{\eta t\}_{t=1}^{T}$: esquema de decaimento do tamanho do passo.

  A utilização de diferentes valores para estes parâmetros conduz a diferentes resultados no processo de seleção de características. A definição destes valores pode ser feita empiricamente, ou através da realização de um conjunto de experimentos para identificar quais os valores seriam os mais apropriados para o problema sendo tratado.
- A alta dimensão do espaço de entradas é outro fator que deteriora o desempenho do algoritmo. Sua aplicação em ambientes com baixa dimensionalidade geralmente é bem sucedida (GERTHEISS e TUTZ, 2008).

4.4.6 Seleção do Critério de Parada

O critério de parada define quando o processo de SSC deve ser finalizado e deve retornar a melhor solução encontrada. O critério de parada é uma decisão crítica, pois, caso ele seja definido erroneamente, podem ocorrer dois problemas extremos. No primeiro, se o critério de parada for muito restritivo, uma porção muito grande do espaço de busca seria analisada, o que resultaria em uma grande quantidade de tempo para alcançar uma solução ótima. No segundo, caso o critério de parada seja pouco restritivo, o tempo da busca seria pequeno, porém haveria grande probabilidade que a solução encontrada não fosse satisfatória.

Dentre os critérios de parada mais utilizados podem ser citados:

- Parar de incluir ou excluir características quando este processo não traz nenhuma melhora ao desempenho do modelo.
- Parar de incluir ou excluir características quando uma quantidade previamente definida de características for atingida.

- Gerar uma determinada quantidade de subconjuntos de características, e selecionar o melhor subconjunto dentre os gerados.

Neste capítulo foram abordados os principais conceitos a respeito da RDD, abrangendo desde a justificativa para a sua execução, até as principais técnicas utilizadas para realização deste processo. No próximo capítulo será apresentado o Modelo Neural de Aprimoramento Progressivo e alguns resultados de sua aplicação.

5 O MODELO NEURAL DE APRIMORAMENTO PROGRESSIVO

Neste capítulo é abordado o modelo neural de aprimoramento progressivo. São apresentadas a fundamentação teórica e a estrutura do modelo proposto. Também são descritos os experimentos realizados sobre bancos de dados sintéticos e reais seguidos da avaliação e discussão dos resultados obtidos.

5.1 Fundamentação teórica e estrutura do modelo

O foco principal deste trabalho é a proposta de um modelo neural de aprimoramento progressivo para redução de dimensionalidade em problemas de aprendizado supervisionado, baseado na ordenação da importância das características de entrada. A importância das características é dada por um escore baseado nos pesos da camada oculta de uma rede previamente treinada. Este modelo é aplicável em RNAs, do tipo MLP com uma camada oculta, treinadas através do algoritmo *Backpropagation*. Dentro deste escopo, a presente proposta visa elucidar a seguinte hipótese de pesquisa: "Os pesos das sinapses que ligam a camada de entrada à primeira camada oculta teriam relação direta com a importância que cada característica de entrada possui para a predição da característica de saída".

Esta hipótese foi formulada a partir da análise de 3 problemas evidenciados na literatura:

1) Embora as RNAs, assim como os demais algoritmos de mineração de dados, já apliquem internamente a redução de dimensionalidade, ignorando as características menos informativas, a utilização de abordagens de redução de dimensionalidade geralmente melhora o desempenho destes algoritmos (WITTEN e FRANK, 2005).

2) De acordo com Bishop (1995) a existência de muitas características de entrada irrelevantes faz com que a rede utilize quase todos seus recursos para representar porções irrelevantes do espaço de busca. Por outro lado, mesmo que a rede consiga focar em características importantes, uma maior quantidade de amostras será necessária para identificar que características são mais ou menos importantes.

3) Em pesquisas experimentais sobre o sistema nervoso, que são realizadas em cobaias, a ligação entre estímulo e resposta pode ser estudada de duas formas distintas. Uma delas é através da codificação neural, que estuda como os estímulos são codificados em potenciais de ativação neurais. A outra forma é a decodificação neural que estuda como a resposta a um estímulo é gerada a partir destes potenciais de ativação

(PANINSKI et al., 2007). Funcionalmente, em uma rede MLP, também podem ser reconhecidas estas duas áreas distintas: a área codificadora e a área decodificadora. Quando uma RNA é treinada com os dados relativos a algum problema, os padrões expressos nestes dados ficam representados nos pesos sinápticos. Os pesos sinápticos entre a camada de entrada e a primeira camada oculta agem como codificadores dos estímulos recebidos, que expressam os padrões encontrados nos dados de entrada. Já os pesos sinápticos entre a última camada oculta e a camada de saída agem como decodificadores, reconstruindo um valor de saída a partir dos padrões extraídos dos dados de entrada pela RNA (ALPAYDIN, 2010).

Assim, dada esta realidade, propõe-se que a definição da importância de cada característica seja dada por um escore que se baseia nos pesos sinápticos da região codificadora da rede, ou seja, os pesos sinápticos que ligam a camada de entrada à primeira camada oculta. A partir da definição da importância que cada característica de entrada tem na predição do valor da saída da rede neural é então proposta uma abordagem de redução de dimensionalidade para otimizar a criação dos modelos neurais.

Considere-se uma RNA do tipo MLP, com *N* entradas, *L* unidades ocultas e uma única saída, conforme figura 5.1, treinada para uma tarefa de regressão pelo algoritmo *Backpropagation* (BP) com um conjunto de *P* dados de treinamento. Para tanto, a função de ativação das unidades da camada oculta é a tangente hiperbólica, e a da saída é linear. Considera-se ainda que as entradas foram normalizadas, de modo que tenham média zero.

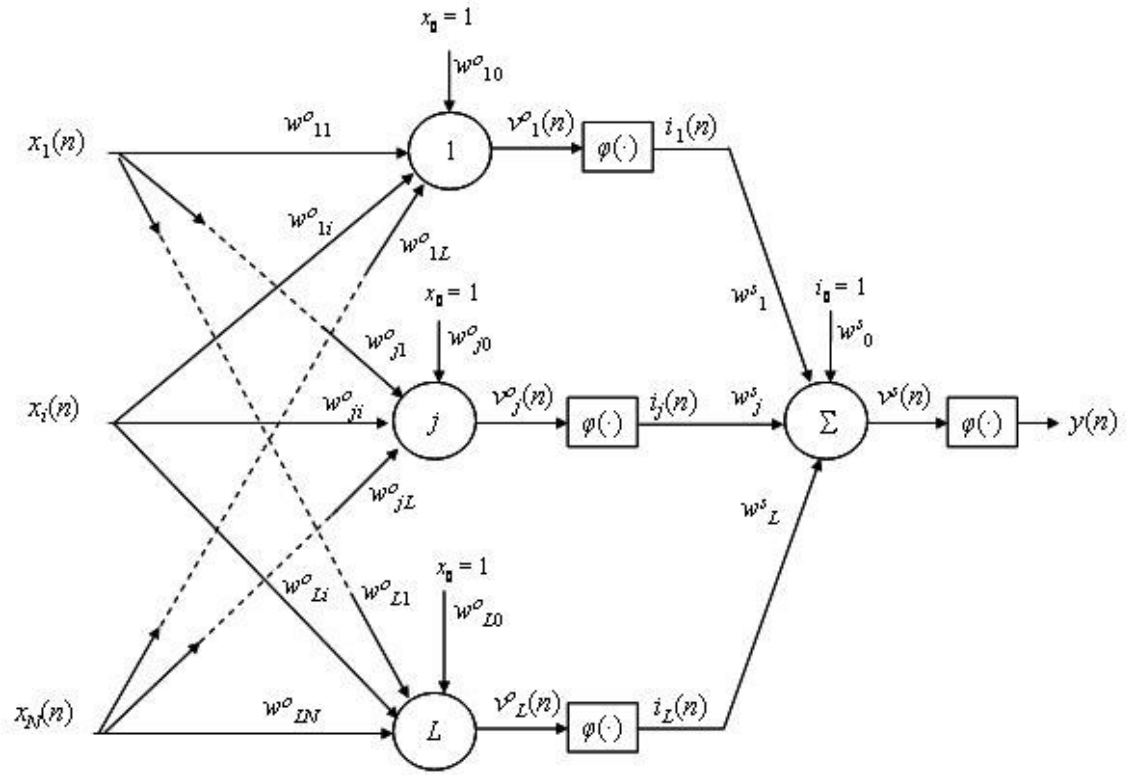

Figura 5.1: RNA do tipo MLP

Dentro deste escopo, deseja-se ordenar as entradas pela sua importância em relação à predição da saída da rede. Para isso, propomos utilizar apenas a informação dos valores dos pesos das unidades da camada oculta ajustados pelo algoritmo BP para a tarefa de interesse.

Os pesos da camada oculta são os parâmetros usados na transformação não linear do espaço original de entrada para o espaço intermediário definido pelas unidades ocultas. As saídas da camada oculta formam um vetor de características que serve de base para a regressão linear efetuada pela unidade de saída, onde os pesos são os parâmetros do regressor linear de saída. Durante o treinamento, o algoritmo BP ajusta os pesos da camada oculta de modo a formar características intermediárias ótimas para o problema de regressão, que é realizado pela camada de saída. Como os pesos da camada de saída são compartilhados por todas as unidades da camada oculta, a nossa suposição é que os pesos da camada oculta fornecem a informação necessária para a ordenação da importância das entradas no problema de regressão. A partir destas considerações, derivamos a seguir a expressão para o cálculo do escore utilizado na ordenação das entradas.

O cálculo do escore é dado pela seguinte fórmula:

$$s_i = \frac{1}{L}\sum_{j=1}^{L}\left|w^{o}_{ji}\right|$$

Tendo-se que:

- O escore da característica *i* é dado por s_i.
- Há *L* neurônios na primeira camada oculta.
- w^{o}_{ji} é o peso da sinapse entre o *i*-ésimo neurônio da camada de entrada e o *j*-ésimo neurônio da primeira camada oculta.

A escolha da função de média para definir o escore das características de entrada é inspirada na neurociência. Na literatura são descritos diversos esquemas de codificação de informação, dentre eles, o mais difundido é o *rate coding*. Este esquema de codificação assume que quase toda, senão toda, informação sobre os estímulos está contida na taxa de ativação dos neurônios e, o cálculo da taxa de ativação é dado através de uma função de média aritmética (KANDEL, SCHWARTZ e JESSELL, 2000).

Então, presume-se que a partir do cálculo do escore seria possível identificar a importância de cada uma das características de entrada, o que fornece o subsídio para a aplicação do modelo neural de aprimoramento progressivo.

Como base teórica para demonstrar a validade da nossa hipótese de pesquisa, mostraremos a seguir a relação do escore proposto com a sensibilidade média (s_i) para todas as saídas da camada oculta em relação à entrada x_i. Postulamos que a importância de uma entrada x_i está relacionada com a média das sensibilidades de cada saída da camada oculta em relação a esta entrada, definida como s_{ji}. O valor de s_{ji} pode ser calculado pela propagação de uma pequena variação Δx_i sobre o valor médio (nulo) da entrada, até a saída da camada oculta, mantendo as outras entradas em zero. Inicialmente a propagação de Δx_i pelo neurônio *j* da camada oculta produz uma variação do seu potencial de ativação dada por:

$$\Delta v_j^o = \Delta x_i w_{ji}^o$$

Como a função de ativação tangente hiperbólica próximo a zero tem ganho unitário, segue que a variação na saída do neurônio *j* é dada por:

$$\Delta i_j = \Delta v_j^o = \Delta x_i w_{ji}^o$$

Com isso, a sensibilidade s_{ji}, da saída da camada oculta i_j em relação à entrada x_i é dada diretamente pelo peso desta conexão:

$$s_{ji} = \frac{\Delta i_j}{\Delta x_i} = w_{ji}^o$$

Para calcularmos a sensibilidade média s_i para todas as saídas da camada oculta em relação à entrada x_i, não podemos simplesmente somar todas as contribuições individuais de cada saída, pois elas possuem sinal. Sendo assim, optou-se por definir a sensibilidade s_i, como a média dos valores absolutos dos s_{ji}, ou seja:

$$s_i = \frac{1}{L}\sum_{j=1}^{L} s_{ji} = \frac{1}{L}\sum_{j=1}^{L}\left|w_{ji}^o\right|$$

As métricas de redução de dimensionalidade propostas na literatura e abordadas na seção 4.4.3 ou analisam individualmente a relevância de cada característica de entrada em relação à característica de saída, ou necessitam da configuração de parâmetros adicionais para serem utilizadas. A abordagem proposta neste trabalho utiliza uma métrica de escore que:

1) permite identificar as dependências entre diversas características de entrada em relação à predição da saída;

2) não necessita de configuração de parâmetros adicionais; e

3) pode ser integrada ao aprendizado através de redes neurais de maneira direta e pouco custosa em termos de implementação.

A estrutura do modelo

A estrutura do modelo neural de aprimoramento progressivo é bastante similar à estrutura teórica de um *wrapper*, que foi apresentada na figura 4.4. A figura 5.2 apresenta à estrutura da abordagem proposta. Esta estrutura de aprimoramento progressivo é similar a estrutura proposta por Effroymson, (1960). A maior diferença entre as duas abordagens consiste na métrica de ordenação dos atributos pois, enquanto

a abordagem de Effroymson é baseada em regressão múltipla, nossa abordagem é baseada no escore neural aqui proposto.

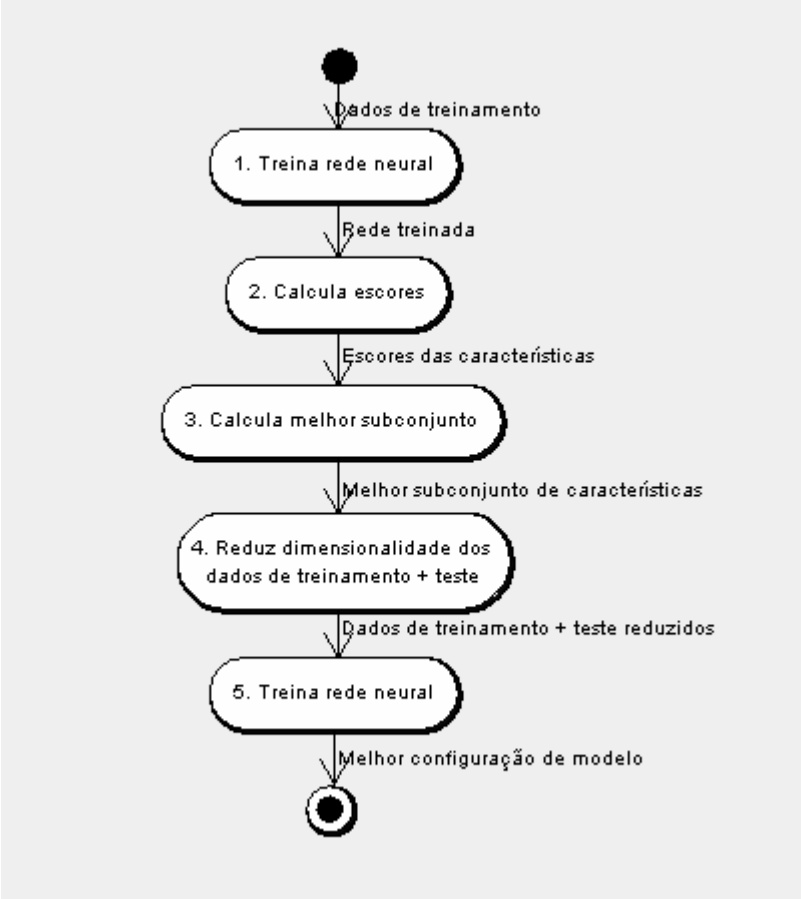

Figura 5.2: Seqüência de atividades do modelo neural de aprimoramento progressivo

Conforme apresentado na figura 5.2, a abordagem é baseada em cinco passos distintos: 1) Treinar a rede neural com os dados de treinamento incluindo todas as características disponíveis; 2) Calcular os escores de cada uma das características de entrada; 3) Definir o melhor subconjunto de características; 4) Reduzir a dimensionalidade dos dados de treinamento + teste com base no melhor subconjunto de características encontrado; 5) Treinar novamente a rede neural com os dados de treinamento + teste com dimensionalidade reduzida para gerar a melhor configuração de modelo. Os três primeiros passos são executados sobre os dados de treinamento, enquanto os demais são executados sobre os dados de treinamento + teste.

Na atividade 3 é realizada a avaliação dos subconjuntos de atributos através da criação de modelos incrementais, a partir do modelo mais simples com o atributo de maior escore, de forma que cada novo modelo contenha um atributo a mais que o modelo anterior. A cada nova característica acrescentada ao modelo, é realizada novamente a redução de dimensionalidade dos dados de treinamento originais, a rede neural é treinada, e seu resultado é avaliado. Este processo é realizado iterativamente até que seja atingido o critério de parada. A figura 5.3 apresenta o detalhamento da atividade 3.

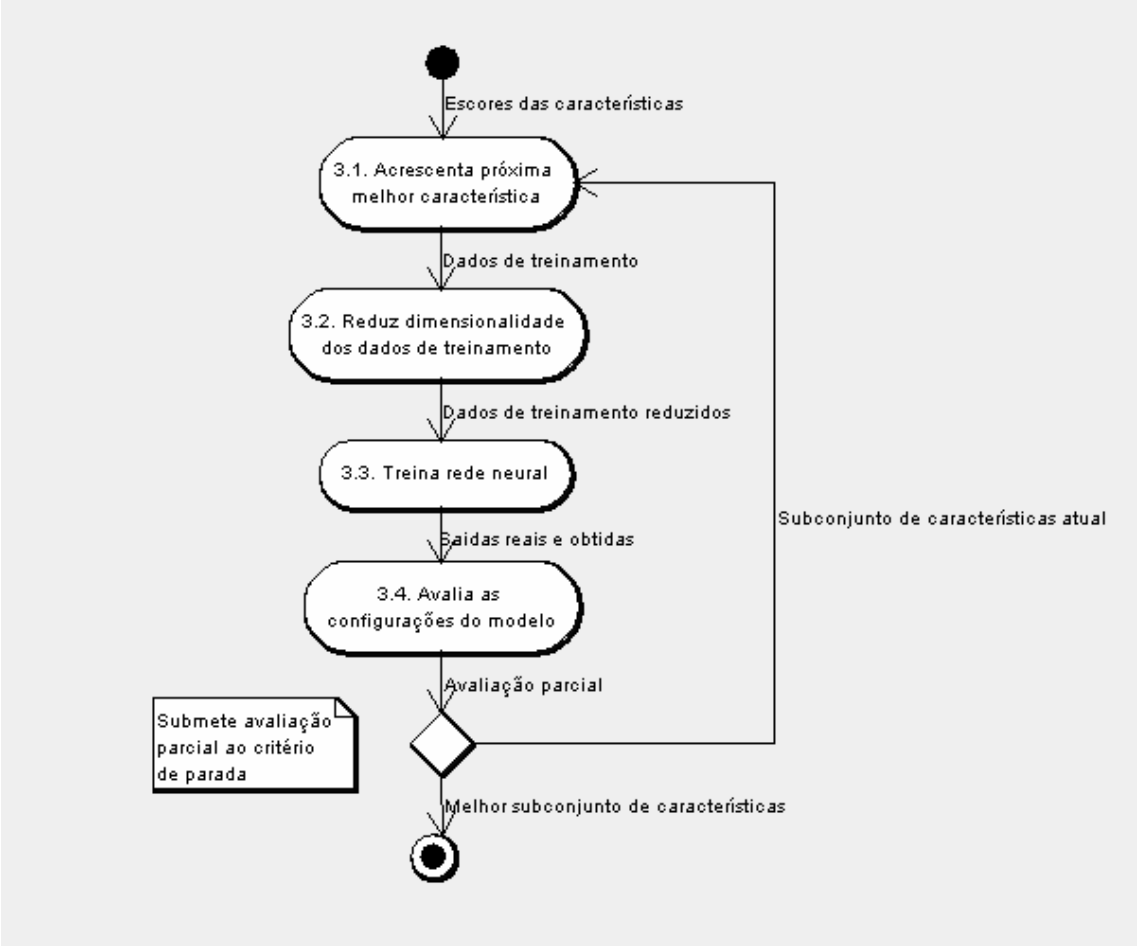

Figura 5.3: Seqüência das atividades que compõem o cálculo do melhor subconjunto de características

Tendo-se que os dados originais são representados em uma matriz m x n, onde m é a quantidade de amostras e n é a quantidade de variáveis descrevendo cada amostra, a abordagem proposta visa transformar os dados em uma nova matriz m x o, onde $o < n$. Esta nova matriz é denotada por E_{Xopt}. O conjunto X_{opt} que contém as o características selecionadas para este modelo são aqueles com maiores escores, e a intenção é que o modelo com o características de entrada atinja um maior nível de exatidão preditiva que o modelo com n características.

Adicionalmente, salienta-se que a abordagem proposta pode ser aplicada tanto em tarefas de classificação quanto de regressão, conforme evidenciado na avaliação de desempenho, e funciona com alto nível de eficiência tanto em problemas lineares quanto em problemas não lineares. Uma relevante restrição à aplicação da abordagem proposta é que a RNA treinada com todas as características de entrada deverá produzir um modelo com desempenho no mínimo equivalente a um "aprendiz fraco" (HAYKIN, 1999). Caso o modelo inicial não consiga aprender absolutamente nada sobre os dados, então não haverá nenhum conhecimento relevante expresso nos pesos sinápticos e as saídas geradas pela rede serão, em média, iguais à incerteza máxima. Se esta restrição não for obedecida, a aplicação do modelo neural de aprimoramento progressivo não irá gerar nenhum benefício adicional em relação ao modelo inicial.

5.2 Avaliação da proposta sobre dados sintéticos

Para realizar a avaliação de desempenho de nossa abordagem foram realizados alguns experimentos de classificação e regressão sobre ambientes plenamente controlados, sendo gerados conjuntos de dados fictícios com relações não lineares bem conhecidas entre entradas e saídas. Uma forma sistemática de avaliação de desempenho é importante para que sejam obtidos resultados confiáveis permitindo a comparação e, principalmente, a reprodução dos experimentos por outros pesquisadores.

Para modelar estes problemas foi utilizada uma RNA treinada com o algoritmo Resilient Backpropagation proposto por Riedmiller e Braun (1993). Para obter-se uma significância estatística, os experimentos foram repetidos 20 vezes sobre condições idênticas de execução. O processo de validação cruzada foi realizado sendo utilizada a técnica *10-fold*. Todos os experimentos demonstrados nos resultados preliminares foram realizados na ferramenta Matlab®, utilizando o *toolbox* de redes neurais.

5.2.1 Conjunto de dados sintético XOR

Para este experimento foi gerado um conjunto de dados no qual estava expresso o problema do XOR, que é um problema clássico na área de redes neurais (HAYKIN, 1999). O problema do XOR é um problema de natureza não linear, no qual a aplicação de técnicas lineares não é eficiente. Além disso, no conjunto de entradas foram incluídas diversas características irrelevantes para a solução do problema. O objetivo deste experimento é comprovar a capacidade da abordagem proposta para identificar as características mais informativas e melhorar os resultados obtidos pela rede neural que utiliza todas as características de entrada.

5.2.1.1 Banco de Dados

Foi gerado um banco de dados fictício com 20 características e 100 amostras. O valor de cada característica para uma dada amostra pode assumir aleatoriamente os valores 0 ou 1. O conjunto de entradas *input_data* é dado por:

$$input\_data = round(rand(features,samples))$$

onde *features* é o número de características para cada amostra, definido como 20; e *samples* é o número de amostras do conjunto de dados, definido como 100. Foram gerados diversos conjuntos de dados com diferentes valores de *features* e *samples*. Nos experimentos realizados foram definidos os valores 20 e 100 pois permitiam um bom balanceamento entre quantidade de amostras e dificuldade do problema; além de ser obtido um problema de razoável dificuldade.

Os valores de saída para este conjunto de amostras são dados por uma função XOR dos valores da primeira e da segunda coluna:

$$output\_data = xor(input\_data(1,:),input\_data(2,:))$$

Adicionalmente, tanto os valores de entrada quando os de saída são escalonados de forma que o valor mínimo seja -1 e o valor máximo seja 1, em virtude deste ser um dos requisitos das redes neurais utilizadas. A função de escalonamento é dada pela seguinte fórmula:

$$pe = 2*(p-minp)/(maxp-minp) - 1$$

onde *minp* é o valor mínimo assumido pela característica em todas as amostras, *maxp* é o valor máximo assumido pela característica em todas as amostras, *p* é o valor da característica na amostra atual, e *pe* é o valor *p* escalonado.

5.2.1.2 Resultados

Após a conclusão do processo de treinamento da rede, foram calculados os escores para cada uma das características de entrada. A figura 5.4 mostra estes escores, onde percebe-se que a abordagem conseguiu indentificar as características de entrada mais relevantes para a predição do valor de saída da rede.

Enquanto o modelo neural criado com todas as 20 características de entrada obteve 28% de taxa de erro, o modelo criado após a utilização do modelo neural de aprimoramento progressivo obteve 2,85% de taxa de erro, conseguindo ser mais exato com uma menor quantidade de características.

Tabela 5.1: Distância euclidiana entre os valores ideais de escore e os valores obtidos com cada uma das abordagens utilizadas no problema do XOR

Abordagem	**Distância**
Escore proposto	0,28812
Mahalanobis	1,69662
GLS	2,12132
OLS	2,12132
Internal Product	2,12132
Covariance	2,12132
Kendall	2,12314
Spearman	2,12314
Correlation coefficient	2,12314
T Test Regression	2,80792
Regression	2,80792
Welch Test	2,88818
T Test	2,88818
Wilcoxon	2,94615
U Test	2,95901
Kruskal Wallis	2,95901
Sign	2,99669
Chi-square	3,01954
Entropy	3,35720
Var Test	3,70111
Bartlett	4,01253

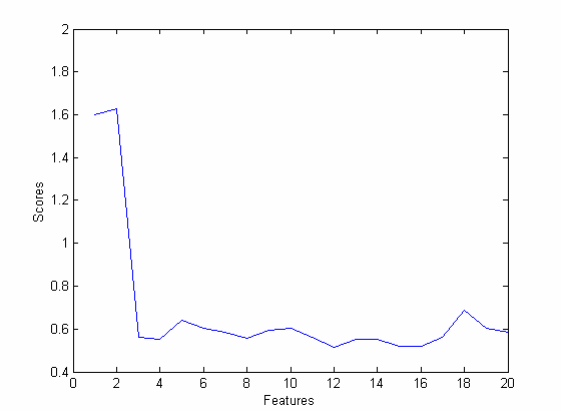

Figura 5.4: Escores para cada uma das 20 características de entrada

Além do cálculo dos escores de cada uma das 20 características pela abordagem proposta, outras outras abordagens descritas na literatura foram testadas (YAMPOLSKIY e GOVINDARAJU, 2006). Dado que somente 2 das 20 características são entradas relevantes para a função XOR, os escores ideais seriam: escore máximo "1" para as duas características relevantes; e escore mínimo "0" para as demais características. Os escores finais das características, após aplicação das diversas abordagens, foram escalonados no intervalo [0,1]. A eficiência de cada uma das abordagens foi baseada na distância euclidiana entre os escores obtidos e os escores ideais. A tabela 5.1 mostra que o escore proposto conseguiu obter valores mais próximos dos pesos ideais que as demais abordagens utilizadas.

5.2.2 Conjunto de dados sintético SENO

Para este experimento foi gerado um conjunto de dados sintético no qual estava expresso um problema proposto envolvendo operações com a função SENO. Da mesma forma que no experimento anterior, no conjunto de entradas foram incluídas diversas características irrelevantes para a solução do problema. Novamente, o objetivo do experimento é comprovar a capacidade da abordagem proposta de identificar as características mais informativas e melhorar os resultados obtidos pela rede neural que utiliza todas as características de entrada, desta vez em um problema não linear de regressão.

5.2.2.1 Banco de Dados

Foi gerado um banco de dados fictício com 50 características e 200 amostras. O valor de cada característica para uma dada amostra pode assumir aleatoriamente valores contínuos entre -1 e 1. O conjunto de entradas *input_data* é dado por:

$$input\_data = rand(features,samples)$$

onde *features* é o número de características para cada amostra, definido como 50; e *samples* é o número de amostras do conjunto de dados, definido como 200. Foram gerados diversos conjuntos de dados com diferentes valores de *features* e *samples*. Nos experimentos realizados foram definidos os valores 50 e 200 pois permitiam um bom balanceamento entre quantidade de amostras e dificuldade do problema; gerando um problema de razoável dificuldade.

Os valores de saída para este conjunto de amostras são dados por uma função SIN dos valores da primeira e da segunda coluna:

output_data =

sin(input_data(10,:))-sin(input_data(20,:))+sin(input_data(30,:))+sin(input_data(40,:))

Adicionalmente, tanto os valores de entrada quando os de saída são escalonados de forma que o valor mínimo seja -1 e o valor máximo seja 1. A função de escalonamento é dada pela seguinte fórmula:

$$pe = 2*(p-minp)/(maxp-minp) - 1$$

onde *minp* é o valor mínimo assumido pela característica em todas as amostras, *maxp* é o valor máximo assumido pela característica em todas as amostras, *p* é o valor da característica na amostra atual, e *pe* é o valor *p* escalonado.

5.2.2.2 Resultados

A figura 5.5 mostra os escores obtidos para cada uma das 50 características de entrada. Através da figura, percebe-se que os escores dos atributos 10, 20, 30 e 40 têm valores bem mais elevados que os escores dos demais atributos. Isto demonstra que o escore proposto torna possível quantificar, no problema sob análise, a relevância destes atributos em relação à saída do problema, exatamente conforme definido nas fórmulas de criação do banco de dados descritas no item anterior. Os demais escores têm valores menores, dado que eles possuem somente ruído, conforme expresso no banco de dados criado.

Enquanto o modelo neural criado com todas as 50 características de entrada obteve 0,0286 como o melhor valor absoluto de erro de teste, o modelo criado após a aplicação da abordagem proposta obteve 0,0102 de taxa de erro, conseguindo ser mais exato com uma menor quantidade de características. Neste experimento, a redução do erro foi de 65%.

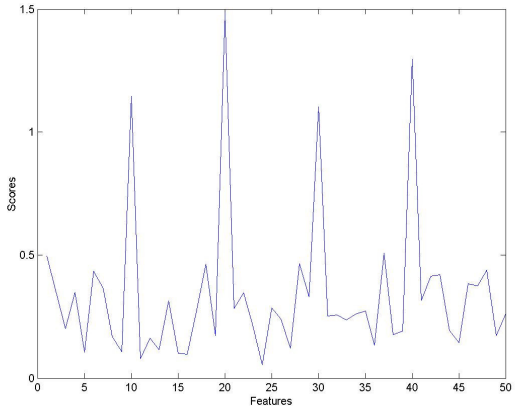

Figura 5.5: Escores para cada uma das 50 características de entrada

Adicionalmente, foram calculados os escores de cada uma das 50 características pela abordagem proposta e por outras abordagens descritas na literatura. Dado que somente 4 das 50 características são entradas relevantes para a função SENO, os escores ideais seriam: escore máximo “1” para as duas características relevantes; e escore mínimo “0” para as demais características. Os escores finais das características, após aplicação das diversas abordagens, foram escalonados no intervalo [0,1]. A eficiência de cada uma das abordagens foi baseada na distância euclidiana entre os escores obtidos e os escores

ideais. A tabela 5.2 mostra que a abordagem proposta conseguiu obter resultados mais próximos aos escores ideais do que as demais abordagens utilizadas.

Tabela 5.2: Distância euclidiana entre os valores ideais de escore e os valores obtidos com cada uma das abordagens utilizadas no problema do SIN

Abordagem	**Distância**
Escore proposto	0,92331
Covariance	1,00363
Correlation coefficient	1,00377
Kendall	1,01867
Spearman	1,01945
GLS	1,54805
OLS	1,54805
Internal Product	1,54805
Sign	1,73190
Bartlett	2,19305
Var Test	2,19311
Chi-square	2,21705
Mahalanobis	3,91332
T Test Regression	4,29966
Regression	4,29966
Entropy	6,58780

No experimento 5.2.1, a distância euclidiana entre os valores ideais de escore a abordagem proposta foi sensivelmente menor que as demais abordagens. Isto demonstra que o escore é sensivelmente mais eficiente que as demais abordagens no problema proposto de natureza não linear. Por outro lado, no experimento 5.2.2, os escores obtidos com a abordagem proposta são pouco melhores que outras abordagens, reconhecidamente eficientes neste problema de natureza linear.

5.3 Avaliação da proposta sobre dados reais

Além dos testes sobre dados sintéticos que geraram as evidências experimentais, a abordagem proposta foi aplicada a problemas de diversas naturezas diferentes. Todos os experimentos demonstrados nos resultados preliminares foram realizados na ferramenta Matlab®, utilizando o *toolbox* de redes neurais. Para realizar os experimentos foram utilizadas RNAs do tipo MLP com as seguintes características comuns:

- Tipo *FeedForward*.
- 3 camadas.
- Soma como função de propagação para a camada oculta.

- Função de transferência sigmóide tangente hiperbólica para a camada oculta.
- Soma como função de propagação para a camada de saída.
- Função de transferência linear para a camada de saída.
- Pesos sinápticos randomicamente iniciados no intervalo [-1,1].
- 5 reinicializações / retreinamentos.
- *10-fold cross-validation*.

A quantidade de neurônios em cada uma das camadas varia de acordo com o experimento, sendo estas quantidades apresentadas posteriormente.

Foram realizados testes sobre problemas de séries temporais, regressão e classificação. Em todas as aplicações apresentadas aqui a abordagem proposta demonstrou ser eficiente. Os resultados das aplicação são descritos a seguir.

5.3.1 Séries Temporais

Métodos clássicos de reconhecimento de padrões, que estão entre os maiores focos de aplicação das redes neurais, geralmente envolvem as tarefas de classificação e regressão. Porém, outra aplicação potencial das redes neurais são as séries temporais, que consistem no estudo da variação de um sinal durante o passar do tempo. Nesta classe de problemas, a modelagem do aspecto temporal passa a ser um fator crítico para a solução do problema.

Tabela 5.3: Matriz de regressão criada com o vetor de entrada

Entradas					Saídas
x_1	x_2	...	x_{m-1}	x_m	x_{m+1}
x_2	x_3	...	x_m	x_{m+1}	x_{m+2}
x_3	x_4	...	x_{m+1}	x_{m+2}	x_{m+3}
...	...	...	...	...	...
...	...	...	...	...	...
x_{n-m}	x_{n-m+1}	...	x_{n-2}	x_{n-1}	x_n

O tempo pode ser modelado em uma rede neural de forma implícita ou explícita (HAYKIN, 1999). No experimento a seguir descrito o tempo foi representado de maneira implícita, na forma de memórias de curta duração, e foi utilizada uma rede neural estática do tipo MLP. Este tipo de rede é denominada *Time Lagged Feedforward Network* (TLFN). A memória de curta duração implementada em uma TLFN consiste na apresentação de um sinal x_n e dos m valores anteriores $x_{n-1}, x_{n-2},..., x_{n-m}$. A fim de atender os requisitos desta memória de curto prazo, houve a necessidade de realizar uma transformação sobre os dados de entrada, que estavam em uma forma vetorial, e foram transformados para uma forma de matriz de regressão. Supondo-se um vetor de entrada

X, com *n* elementos, e uma rede neural com *m* entradas, a matriz de regressão é gerada de acordo com a tabela 5.3.

5.3.1.1 Banco de Dados

Para realizar os experimentos foi utilizado um conhecido banco de dados com a quantidade de passageiros de linhas aéreas nos Estados Unidos, originalmente publicado por (BOX et al., 1976). Estes dados consistem na quantidade mensal medida durante 12 anos consecutivos, entre 1949 e 1960, totalizando 144 amostras. O foco da mineração é a predição do número de passageiros para os 4 anos subseqüentes, de 1961 a 1964, ou seja, as próximas 48 amostras.

Este banco de dados foi foco de uma competição de predição de séries temporais no 25th *International Symposium on Forecasting*, ocorrido em 2005.

5.3.1.2 Experimentos

Foram realizados experimentos com diversas configurações de redes neurais diferentes. Além disso, outra decisão crítica foi em relação à geração da matriz de regressão. O dilema enfrentado foi a determinação da quantidade de entradas a ser utilizada para a rede neural. Esta decisão é crítica devido à pouca quantidade de amostras. À medida que o valor de *m* é aumentado, a quantidade de exemplos para treinamento/teste diminui. A relação entre a quantidade de amostras e entradas é dada da seguinte forma:

$$Q_e = n - m$$

Onde Q_e é a quantidade de amostras disponíveis para treinamento.

De acordo com as diversas configurações de matriz de regressão utilizadas, a melhor configuração possível foi obtida com o valor de $m = 48$. Esta configuração resulta na existência de 96 exemplos para treinamento/teste. Tal cenário denota um problema de alta dimensionalidade, dado que a relação entradas/características é 2.

Para realizar o experimento de predição desta série temporal, foi utilizada uma rede neural com as seguintes características:

- *m* neurônios na camada de entrada.
- 2 neurônios na camada oculta.
- 1 neurônio na camada de saída.

Os resultados obtidos com esta configuração de rede neural são descritos na figura 5.6. Tal resultado mostra claramente que a rede neural utilizada não conseguiu aprender corretamente a tendência crescente da série temporal.

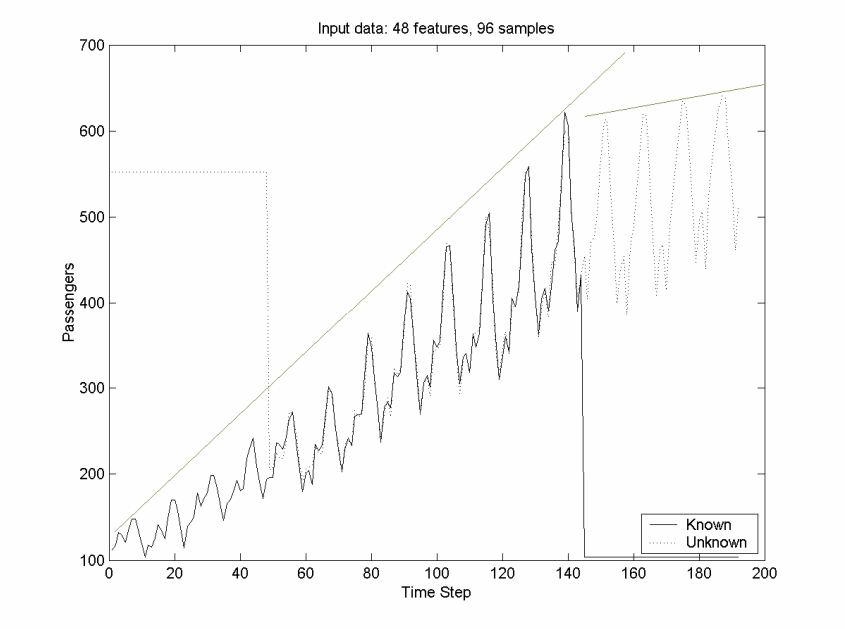

Figura 5.6: Predição dos valores da série temporal usando 48 características de entrada

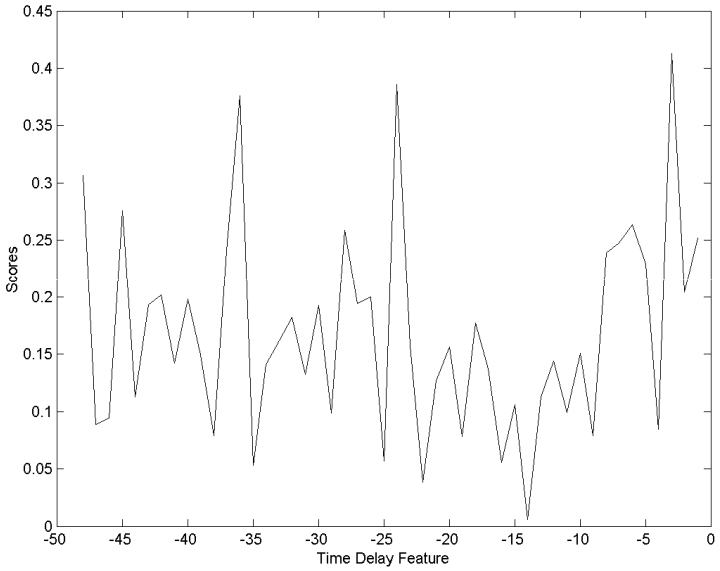

Figura 5.7: Escores das características de entrada

A partir do treinamento realizado, foram obtidos os escores para as características de entrada. Tais escores são apresentados na figura 5.7. A escala negativa no eixo x do gráfico representa a quantidade de atrasos de tempo de cada característica em relação à característica sendo predita. O maior escore apresentado no gráfico, 0.4, é projetado no eixo x no valor -3. Isto significa que a característica mais informativa para predizer a amostra x_t é a característica x_{t-3}. A segunda característica mais informativa é a x_{t-24}, e assim sucessivamente.

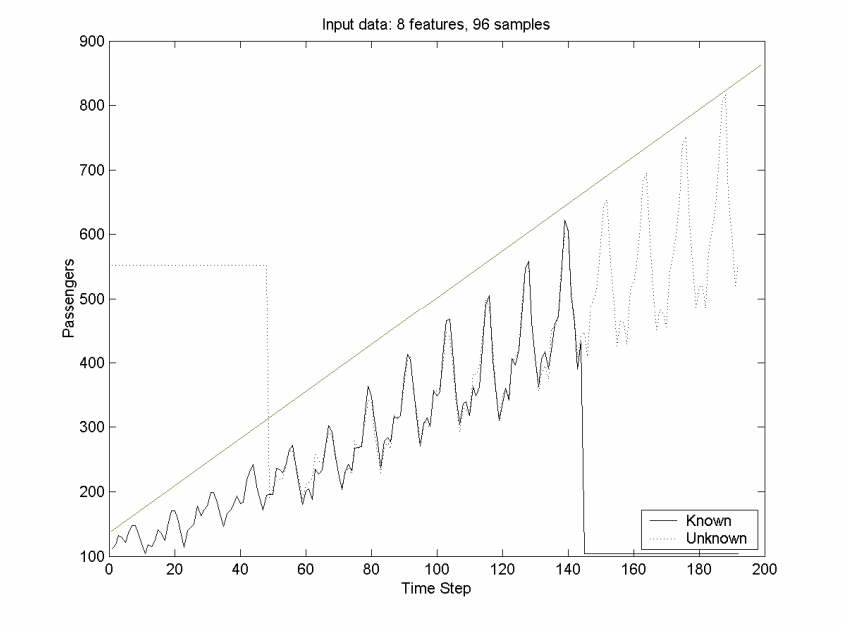

Figura 5.8: Predição dos valores da série temporal usando o conjunto reduzido de características de entrada

A figura 5.8 mostra a predição da série temporal para as próximas 48 entradas, somente com as 8 características de escore mais alto, de acordo com os preceitos da abordagem proposta. Relativamente ao resultado descrito na figura 5.6, percebe-se que a figura 5.8 mostra um resultado mais coerente, onde as amostras preditas mantêm a mesma tendência de crescimento das amostras conhecidas.

Os resultados obtidos com a aplicação da abordagem proposta foram submetidos à competição de predição do 25th *International Symposium on Forecasting* e ficaram entre os 3 melhores trabalhos (CAMARGO e ENGEL, 2005).

5.3.2 Regressão

Para realizar os experimentos de regressão foi escolhido um problema real na área de exploração de petróleo. Tal aplicação possui uma enorme justificativa econômica. A importância desta aplicação dá-se pelo fato de o petróleo ser um bem vital para diversos tipos de indústrias, além de consistir em uma preocupação crítica para diversas nações. Em algumas regiões do planeta, o petróleo chega a ser responsável pela geração de mais de 50% da energia utilizada. Toda a cadeia produtiva do petróleo, incluindo as fases de produção, distribuição, refino e venda, representa a maior indústria do planeta em termos financeiros.

Uma das fases primárias no processo de exploração de petróleo é a perfuração. Dentro desta fase, deve ser realizada uma estimativa do valor da reserva a ser perfurada. A fim de permitir uma estimativa mais exata deste valor, a utilização de modelos preditivos da qualidade dos reservatórios seria de fundamental importância. O foco dos modelos preditivos está concentrado nos principais fatores de qualidade, que são: macroporosidade, porosidade petrofísica e permeabilidade petrofísica.

O objetivo principal dos experimentos de regressão realizados é desenvolver modelos de qualidade de reservatórios de petróleo a partir de dados e interpretações produzidas no estudo dos arenitos da Formação de Uerê, Devoniano da Bacia do Solimões, por Lima e De Ros (2003). Os resultados parciais obtidos nestes

experimentos foram incluídos em um projeto de pesquisa submetido ao CNPq e aprovado para financiamento (ENGEL, 2005).

5.3.2.1 Banco de Dados

O banco de dados utilizado para realização dos experimentos é composto por 96 características e 58 amostras. A tarefa de regressão foi executada a fim de predizer 3 características distintas: macroporosidade, porosidade petrofísica e permeabilidade petrofísica. Para executar estas predições, nem todas as 93 características disponíveis foram utilizadas.

As características, representando parâmetros petrográficos e petrofísicos são divididas em dois tipos: características atômicas e características totalizadoras. As características atômicas são agrupadas em classes. Todas as características totalizadoras, que são classes, podem ser obtidas através da soma de *n* características atômicas. Também é sabido que a soma de todas as características totalizadoras é igual a 100. As características macroporosidade, porosidade petrofísica, e permeabilidade petrofísica, focos das próximas predições, são exemplos de características totalizadoras.

Das 58 amostras disponíveis, foram excluídas 10 amostras por serem consideradas *outliers*. Estas amostras serão consideradas nos trabalhos futuros, pois, de acordo com o especialista do domínio, podem ser consideradas uma classe de amostras distinta das demais.

5.3.2.2 Experimentos

Predição de macroporosidade

Para predição de macroporosidade foram executados os experimentos descritos a seguir.

Em adição às características apresentadas na introdução da seção 5.3, a rede neural tinha as seguintes características particulares:

- 60 neurônios na camada de entrada.
- 4 neurônios na camada oculta.
- 1 neurônio na camada de saída.

Nos experimentos realizados foi obtido um erro médio de predição da macroporosidade de 2,1403, o que representa 20,61% de taxa de erro. A figura 5.9 apresenta os valores obtidos e desejados para cada exemplo predito.

A figura 5.10 apresenta o erro absoluto para cada um dos exemplos, em uma das cinco repetições do experimento, sendo que a média dos erros foi 2,006 e o desvio padrão 1,957.

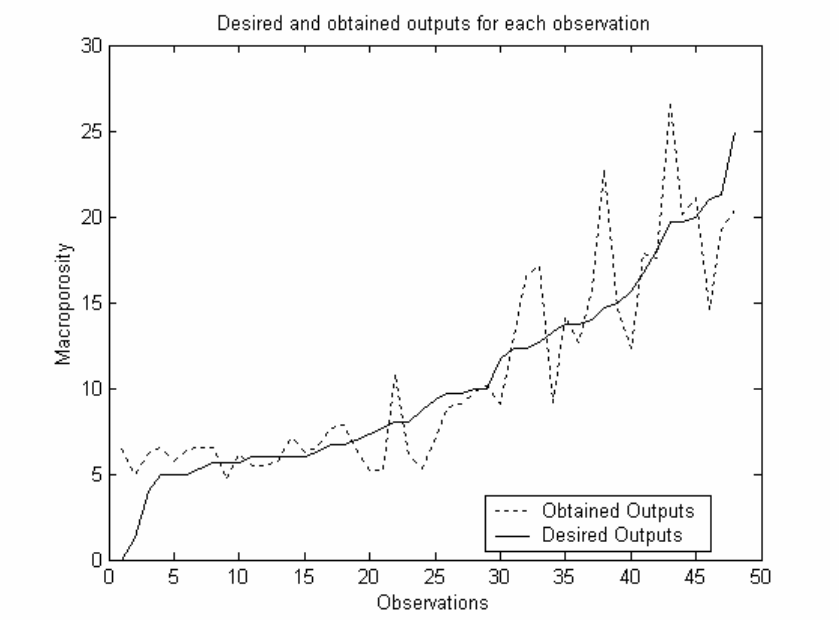

Figura 5.9: Resultado desejado x resultado obtido de macroporosidade para cada uma das 48 amostras por meio da regressão com 60 entradas na rede.

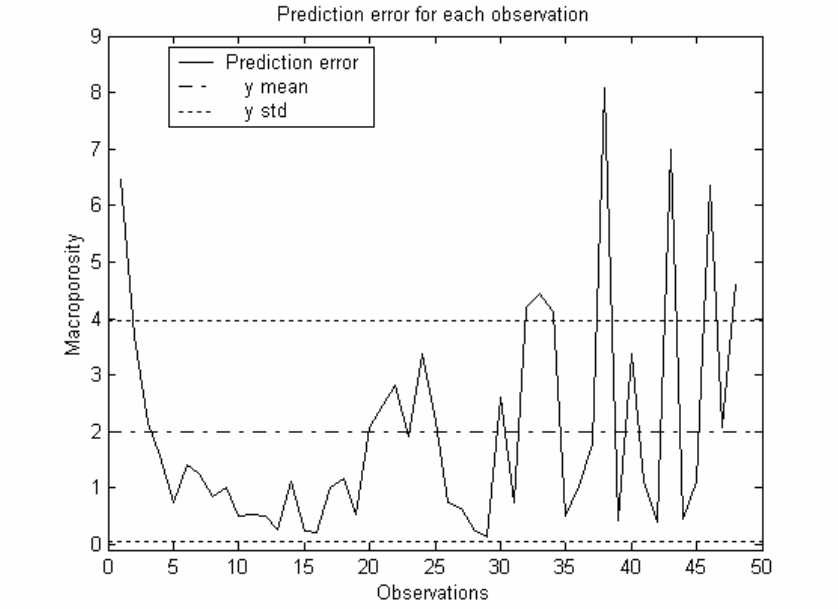

Figura 5.10: Erro de predição de macroporosidade para cada uma das 48 amostras usando 60 características como entrada da rede.

A figura 5.11 apresenta os pesos de cada uma das sinapses das características de entrada. A média dos pesos foi 0,2556 e o desvio padrão 0,3704. Baseados nos pesos das sinapses foram identificadas as características mais relevantes para a tarefa de regressão do valor da macroporosidade. As 10 características com maiores pesos são apresentados na tabela 5.4 em ordem decrescente de importância. As características de 1 a 3 têm pesos maiores que a média mais um desvio padrão, indicando a sua grande importância para predição da macroporosidade. As características 4 e 5 têm pesos maiores que a média, o que também mostra sua importância. As demais características identificam os maiores pesos, menores que a média.

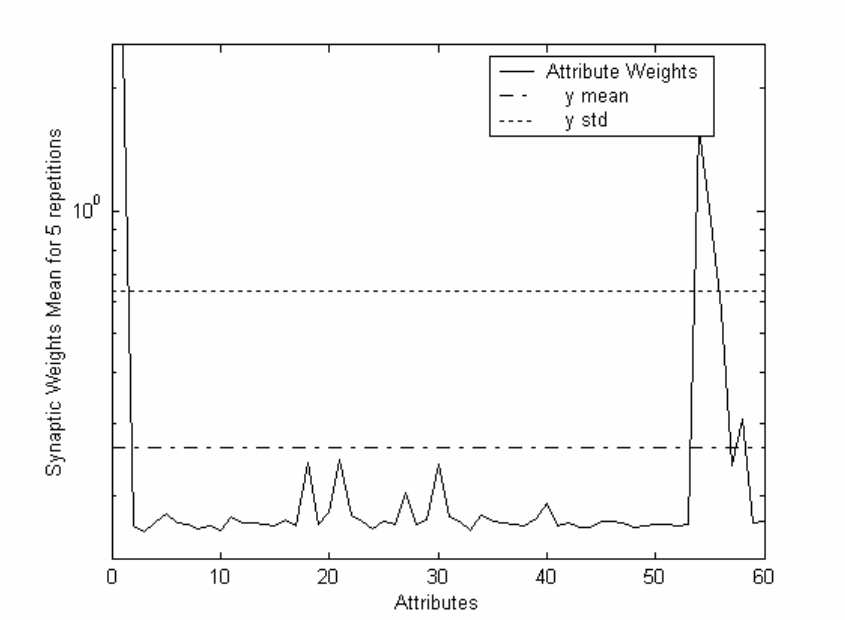

Figura 5.11: Pesos sinápticos de cada uma das 60 características da camada de entrada usados para predição de macroporosidade.

Tabela 5.4: Características mais importantes para a predição da macroporosidade

Entrada	Atributo	Descrição	Peso
1	1	Quartz Monocrystalline	2,5538
2	54	Intergranular Volume	1,5992
3	55	Cement Total	1,0115
4	56	Carbonate Total	0,5677
5	58	Grain Replacement Total	0,3078
6	21	Quartz Overgrowth	0,2446
7	18	Clay Ooid	0,2396
8	30	Silicified Secondary Matrix	0,2379
9	57	Silica Total	0,2375
10	27	Microquartz Rims	0,2030

A partir da identificação das características mais importantes para a predição da macroporosidade, apresentadas na tabela 5.4, foi realizado o processo de aprimoramento progressivo.

Tendo-se que a média do valor da macroporosidade é 10,38 é possível inferir-se uma idéia do percentual de erro representado pelo erro absoluto com diferentes números de características utilizadas como entrada na rede.

Um fator que pode ser comprovado experimentalmente é que o erro de predição de macroporosidade com apenas três características é menor que o erro com todas as sessenta características disponíveis. A tabela 5.5 apresenta os valores de erro obtidos em cada um dos experimentos com diferentes quantidades de características de entrada. É notório que a rede possui um conjunto ótimo de características de entrada, com o qual o erro é mínimo, que pode ser um subconjunto das características disponíveis para o aprendizado.

Tabela 5.5: Variação da taxa de erro em função do número de características de entrada

Características de Entrada	Erro Absoluto	Erro Percentual
1	2,7173	26,17%
2	2,7110	26,11%
3	1,8637	17,97%
4	1,9578	18,86%
5	2,0533	19,78%
60	2.1403	20,61%

A figura 5.12 apresenta os valores obtidos e desejados para cada amostra predita utilizando somente 3 características de entrada. A figura 5.13 apresenta os valores de erro absolutos utilizando 3 e 60 características de entrada, sendo perceptível que o erro de predição com 3 características é quase sempre menor que com 60 características, o que mostra indiscutivelmente a importância das características 1, 2 e 3 da tabela 5.4.

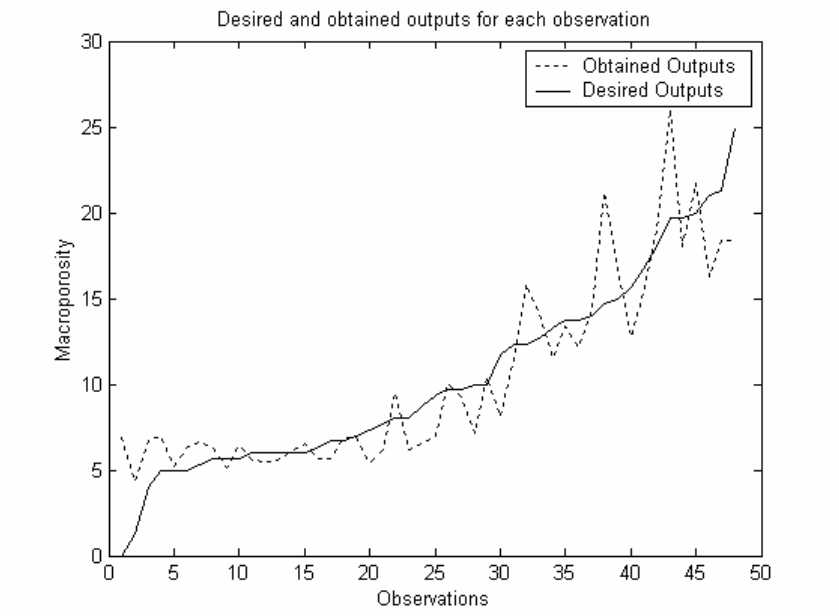

Figura 5.12: Resultado desejado x resultado obtido de macroporosidade por meio da regressão com 3 entradas na rede.

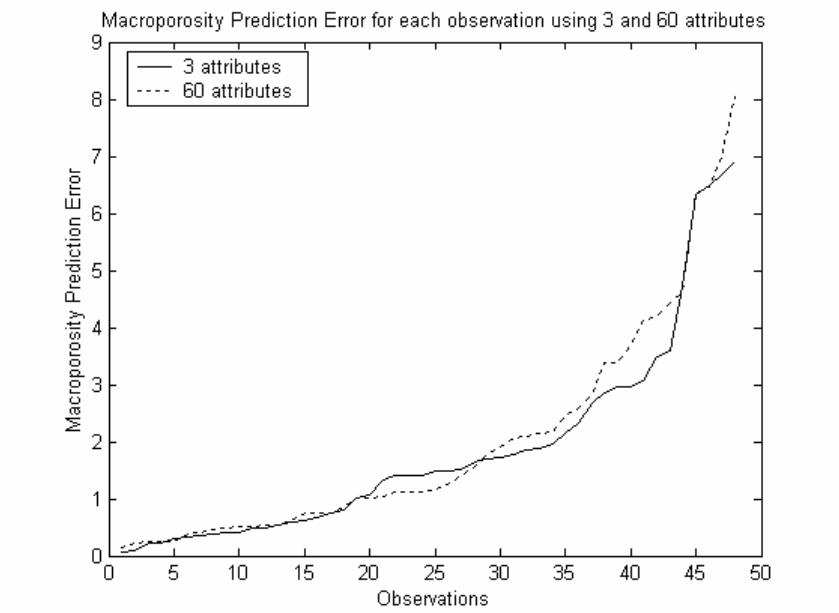

Figura 5.13: Erros de predição de macroporosidade com 3 e 60 características de entrada.

Os resultados destes experimentos foram comparados com a abordagem de regressão multivariada, que é largamente utilizada neste problema, e os resultados são discutidos em mais detalhes em Camargo e Engel (2009).

Predição de porosidade petrofísica

Para predição de porosidade petrofísica foram executados os experimentos descritos a seguir.

Foi utilizada uma rede neural com as seguintes características particulares:

- 70 neurônios na camada de entrada.
- 4 neurônios na camada oculta.
- 1 neurônio na camada de saída.

Nos experimentos realizados foi obtido um erro médio de predição da porosidade petrofísica de 2,2367, o que representa 16,31% de taxa de erro. A figura 5.14 apresenta os valores obtidos e desejados para cada amostra predita.

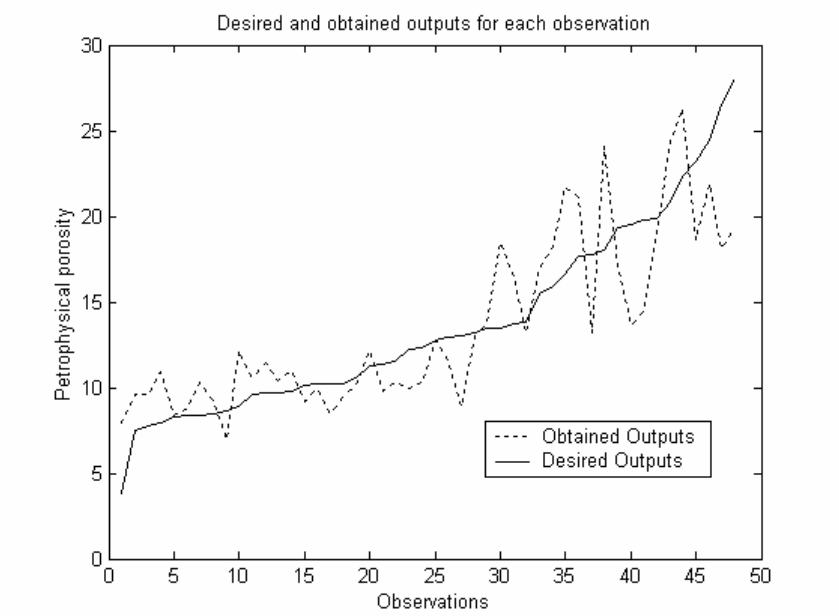

Figura 5.14: Resultado desejado x resultado obtido de porosidade petrofísica por meio da regressão com 70 entradas na rede.

A figura 5.15 apresenta o erro absoluto, em uma das cinco repetições do experimento, sendo que a média dos erros foi 2,47, e o desvio padrão do erro foi 2,096.

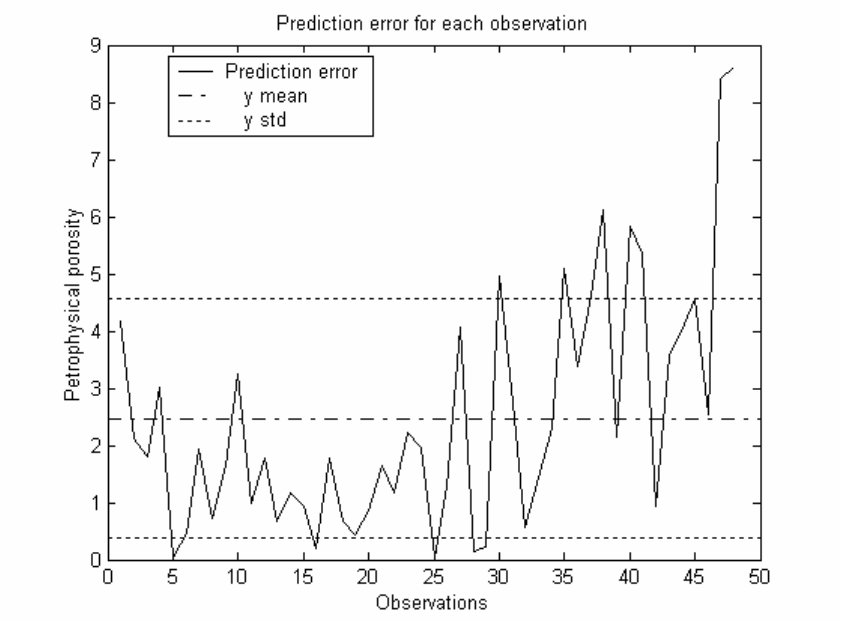

Figura 5.15: Erro de predição de porosidade petrofísica para cada um das 48 amostras usando 70 características como entrada da rede.

A figura 5.16 apresenta os pesos de cada uma das sinapses das características de entrada. A média dos pesos foi 0,2043 e o desvio padrão 0,2373. Baseados nos pesos das sinapses foram identificadas as características mais relevantes para a tarefa de regressão do valor da porosidade petrofísica. As 10 características com maiores pesos são apresentados na tabela 5.6 em ordem decrescente de importância. As características de 1 e 2 têm pesos maiores que a média mais um desvio padrão, indicando a sua grande importância para predição da porosidade petrofísica. As características 3 a 8 têm pesos maiores que a média, o que também mostra sua importância. As demais características identificam os maiores pesos, menores que a média.

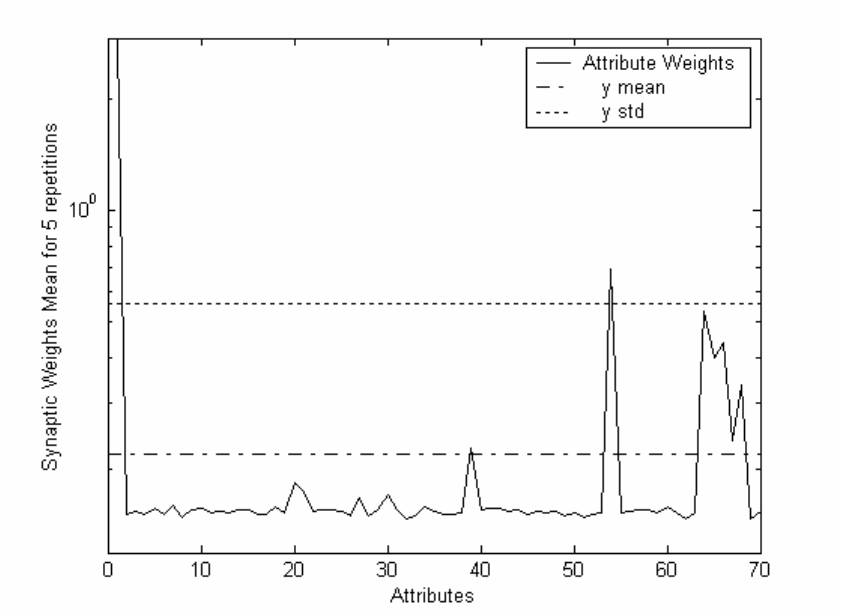

Figura 5.16: Pesos sinápticos de cada uma das 70 características da camada de entrada usadas para predição de porosidade petrofísica.

Tabela 5.6: Características mais importantes para a predição da porosidade petrofísica

Entrada	Atributo	Descrição	Peso
1	1	Quartz Monocrystalline	2,9278
2	54	Intergranular Porosity	0,6925
3	64	Intergranular Volume	0,5350
4	66	Carbonate Total	0,4382
5	65	Cement Total	0,3989
6	68	Grain Replacement Total	0,3381
7	67	Sílica Total	0,2391
8	39	Illite Intergranular Fibrous	0,2267
9	20	Mud Pseudomatrix + Bioturbation Matrix	0,1833
10	21	Quartz Overgrowth	0,1723

A partir da identificação das características mais importantes para a predição da porosidade petrofísica foram feitos diversos outros experimentos, com diversas quantidades de características de entrada. Para cada experimento com *n* características de entrada, estas *n* características eram os com maior peso das suas sinapses de entrada, de acordo com a tabela 5.6.

Tendo-se que a média do valor da porosidade petrofísica é 13,71 é possível inferir-se uma idéia do percentual de erro representado pelo erro absoluto com diferentes números de características utilizadas como entrada na rede.

Um fator que pode ser comprovado experimentalmente é que o erro de predição de porosidade petrofísica com apenas duas características é menor que o erro com todas as setenta características disponíveis. A tabela 5.7 apresenta os valores de erro obtidos em cada um dos experimentos com diferentes quantidades de características de entrada. Novamente é notório o fato de que a rede possui um conjunto ótimo de características de entrada, com o qual o erro é mínimo, que pode ser um subconjunto das características disponíveis para o aprendizado.

A figura 5.17 apresenta os valores obtidos e desejados para cada amostra predita utilizando somente 2 características de entrada. A figura 5.18 apresenta os valores de erro absolutos utilizando 2 e 70 características de entrada, sendo perceptível que o erro de predição com 2 características é quase sempre menor que com 70 características, o que mostra indiscutivelmente a importância das características 1 e 2 da tabela 5.6.

Tabela 5.7: Variação da taxa de erro em função do número de características de entrada

Características de Entrada	Erro Absoluto	Erro Percentual
1	2,8676	20,91%
2	2,0747	15,13%
3	2,0902	15,24%
4	2,1626	15,77%
5	2,1012	15,32%
70	2,2367	16,31%

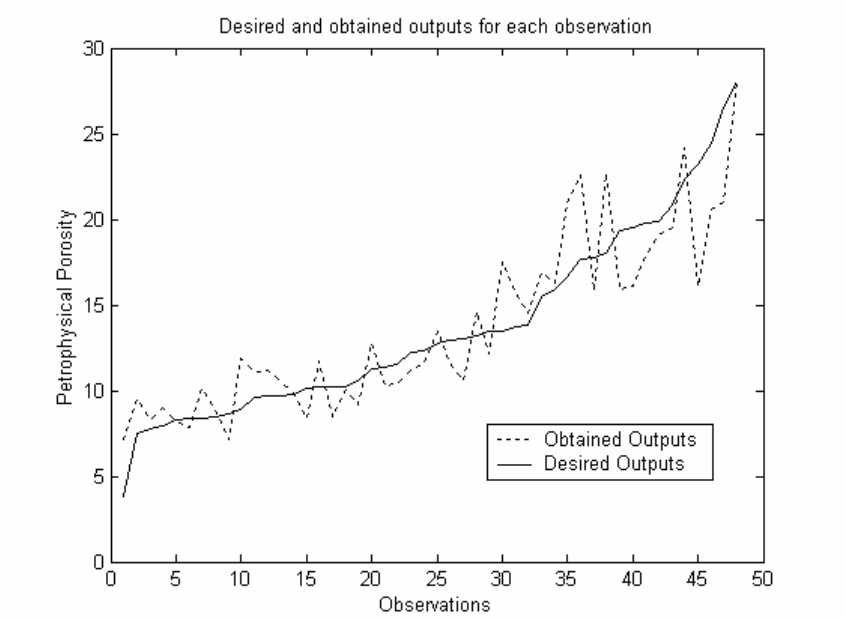

Figura 5.17: Resultado desejado x resultado obtido de porosidade petrofísica por meio da regressão com 2 entradas na rede.

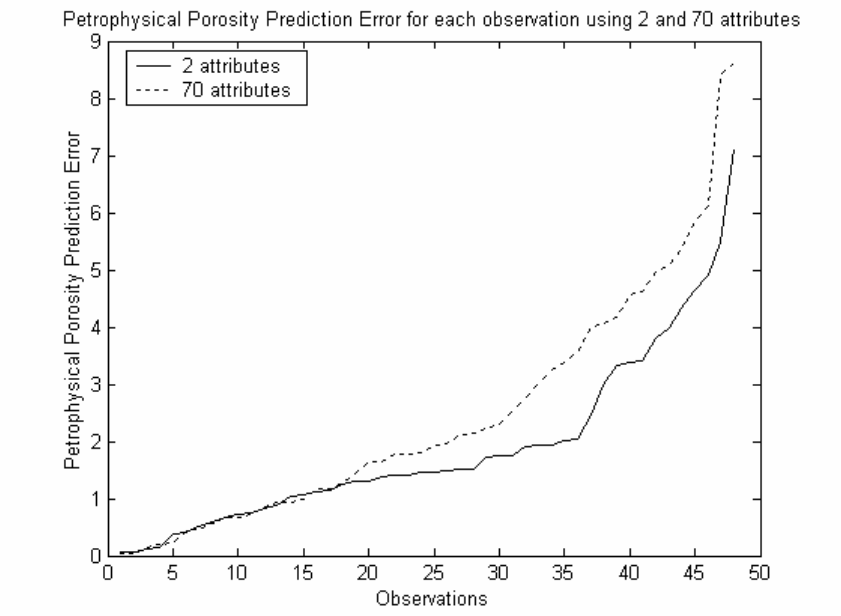

Figura 5.18: Erros de predição de porosidade petrofísica com 2 e 70 características de entrada.

5.3.3 Classificação

Os experimentos de classificação foram realizados a partir de um banco de dados sobre metabolismo de vacas de um rebanho leiteiro, visando identificar a propensão de um determinado indivíduo do rebanho a ter mastite. A mastite é uma inflamação da glândula mamária. Esta doença é a mais preocupante em rebanhos leiteiros em todo o mundo devido à alta incidência de casos clínicos, alta incidência de infecções não perceptíveis a olho nú e aos prejuízos econômicos que acarreta. As lesões no tecido mamário causadas pela mastite tornam as células excretoras menos eficientes, com menor capacidade de produzir e secretar leite. Especificamente no Brasil, pesquisas demonstram que tecidos com mastite produzem entre 25 e 42% menos de leite (GAONA, 2005).

Em Campos et al. (2006) os resultados apresentados por esta abordagem foram analisados e validados pelo especialista do domínio. Para dar uma maior confiabilidade nos resultados obtidos, além da validação do especialista do domínio, foram utilizadas métricas que avaliam a significância estatística dos resultados obtidos, tais como: exatidão, sensibilidade, especificidade e precisão. Adicionalmente, para avaliar a capacidade do modelo predizer dados não vistos, foram utilizadas técnicas de validação cruzada.

5.3.3.1 Banco de Dados

Para realizar os experimentos foi utilizado um banco de dados de bovinos de um rebanho leiteiro. As características eram indicadores do metabolismo: energético, protéico, mineral, endócrino e do funcionamento hepático, de bovinos leiteiros de alta produção sob condições de manejo controlado. O foco da mineração é a identificação de relacionamentos dos indicadores metabólicos com a Contagem de Células Somáticas (CCS), fator que determina a presença ou ausência de mastite. O banco de dados era composto de 107 amostras, sendo 84 negativas e 23 positivas, cada uma descrita por 40 características, sendo uma destas características o alvo da predição.

5.3.3.2 Experimentos

O primeiro experimento executado leva em consideração todas as características disponíveis, sendo criado um modelo com 39 variáveis. A figura 5.19 apresenta os escores da camada de entrada referentes a estas 39 variáveis. O segundo passo é ordenar as variáveis de acordo com os seus respectivos escores. O terceiro passo é criar diversos modelos, partindo do modelo com um único atributo, que tem o maior escore, e inserindo-se gradativamente as próximas características com maior escore. No experimento atual, foi utilizada a técnica de validação cruzada *leave-one-out*.

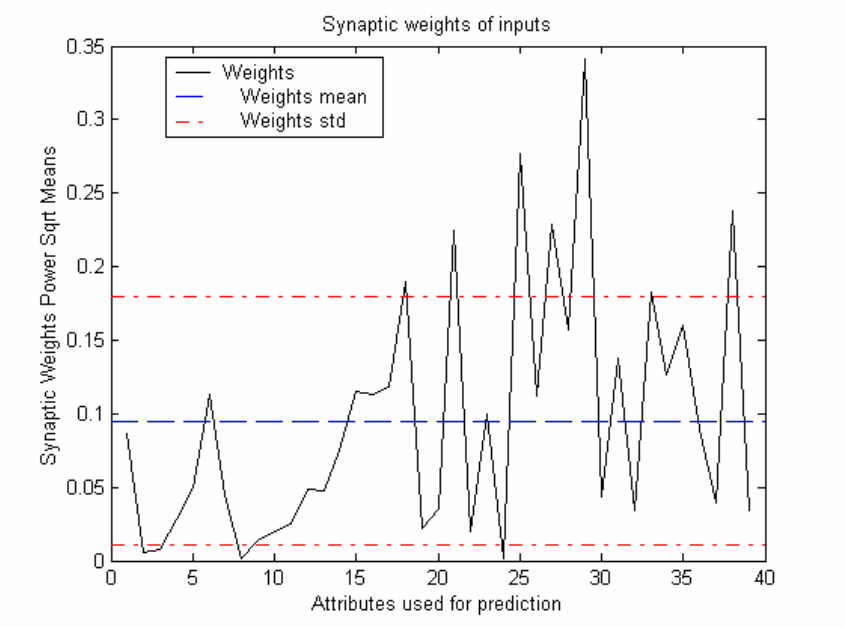

Figura 5.19: Escores das 39 características de entrada

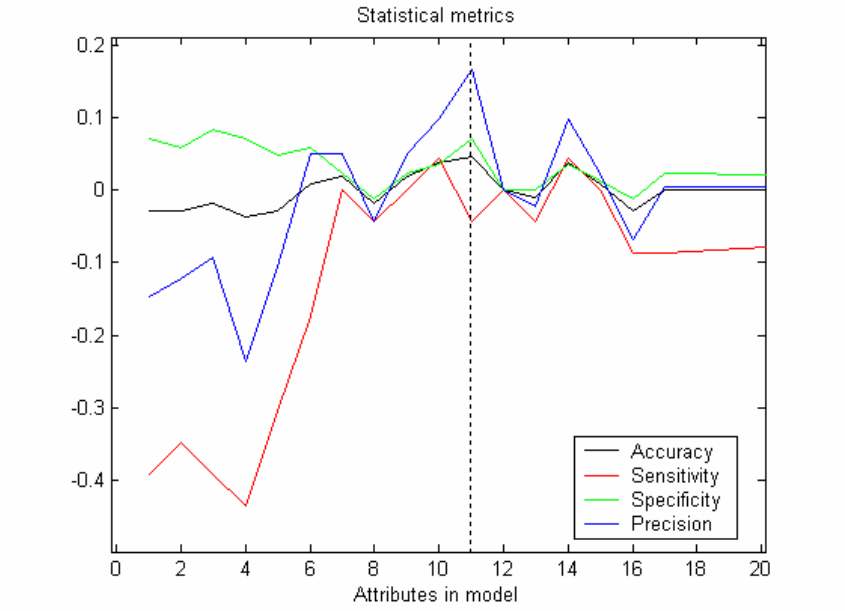

Figura 5.20: Desempenho relativo dos modelos com diferentes quantidades de características

A tabela 5.8 apresenta a comparação entre dois modelos gerados com a abordagem proposta, com 10 e 11 características, e um modelo com todas as características presentes nos dados originais. A comparação é baseada em métricas estatísticas e mostra os melhores resultados obtidos com modelos mais simples. Nota-se que o modelo com 10 características é mais eficiente que o modelo com 39 características de acordo com todas as métricas analisadas. Já a partir da inclusão do 11º atributo, o modelo melhorou sensivelmente em relação às métricas de especificidade e precisão, todavia diminuiu sua sensibilidade, apresentando uma maior dificuldade de predição dos casos positivos.

A figura 5.20 apresenta o desempenho relativo dos modelos com 1 até 17 características em relação ao modelo original com 39 características. O valor 0 do eixo y representa o valor das métricas em relação ao modelo original.

Tabela 5.8: Comparação de 2 modelos gerados com a abordagem proposta e modelo original com todas as características.

Métrica	10 Características	11 Características	39 Características
Exatidão	0,83178	0,84112	0,79439
Sensibilidade	0,56522	0,47826	0,52174
Especificidade	0,90476	0,94048	0,86905
Precisão	0,61905	0,68750	0,52174

Os resultados obtidos com estes experimentos estão detalhadamente descritos em Campos et al. (2006). Além deste experimento de classificação previamente descrito, também foram realizados, sobre este banco de dados, outros experimentos de regressão visando identificar as características que mais contribuem para a obtenção de leite de alta qualidade. Tais experimentos de regressão estão descritos em detalhes em Gaona (2005).

6 CONCLUSÕES E TRABALHOS FUTUROS

Nesta tese foi abordado o problema de redução de dimensionalidade em problemas de aprendizado neural supervisionado, utilizando MLP. Foi demonstrado que o problema da alta dimensionalidade influi negativamente na qualidade dos modelos gerados através da aplicação da técnica de RNA. Embora as RNA consigam identificar as características de entrada mais relevantes em relação às características de saídas que estão sendo preditas, o aumento da quantidade de características de entrada irrelevantes vai gradativamente deteriorando a qualidade do modelo preditivo. Dentro desta realidade, em dados de alta dimensionalidade, a utilização de modelos neurais pode ser dificultada.

A necessidade de descobrir conhecimento em dados de alta dimensionalidade tem se tornado cada vez mais comum em diversos ramos da ciência, principalmente devido à evolução e surgimento de novas tecnologias de obtenção e geração de dados. Muitas vezes também, estes dados referem-se a áreas que representam a exploração de novas fronteiras da ciência, onde ainda não existe a figura do especialista do domínio. Desta forma, a utilização de conhecimento prévio do especialista no processo de descoberta de conhecimento é impossível.

Durante o processo convencional de aprendizado neural supervisionado, as redes neurais, assim como as demais técnicas de mineração de dados, demonstram a capacidade de identificar as características de entrada mais ou menos informativas. Porém, se as características de entrada pouco informativas continuarem fazendo parte do modelo, elas passam a gerar ruído que tende a diminuir a precisão do modelo. Adicionalmente, um modelo com mais características de entrada tem seu processo de treinamento mais lento, e pelo fato do modelo não obedecer ao princípio da navalha de Occam, um modelo com mais características de entrada é mais complexo de ser explicado.

Dentro desta realidade, a aplicação prévia de técnicas de redução de dimensionalidade geralmente melhora o desempenho dos algoritmos de mineração de dados. Porém, um dos fatores que dificulta a utilização destas técnicas é que durante o já complexo processo de descoberta de conhecimento, surge a necessidade de conhecer e avaliar as abordagens de redução de dimensionalidade a fim de realizar-se uma escolha da técnica mais indicada para o algoritmo de mineração sendo utilizado.

Neste escopo, a seguinte tese apresenta as seguintes contribuições:

1) A abordagem é intuitiva e de fácil aplicação, podendo ser integrada ao processo de aprendizado neural de maneira transparente e sem a necessidade de configuração de parâmetros adicionais.

2) Pode ser aplicada de maneira idêntica tanto em problemas de regressão quanto de classificação e séries temporais.

3) Pode ser aplicada de maneira idêntica tanto em problemas de natureza linear quanto não linear. As contribuições 1 e 2 já consistem em restrições para aplicação de muitas outras abordagens similares propostas na literatura.

4) A aplicação da abordagem demonstrou, nos estudos de caso apresentados, a capacidade de gerar modelos mais precisos, mais rápidos e mais simples do que os modelos neurais convencionais. Adicionalmente, os modelos criados são mais facilmente explicáveis devido à menor quantidade de características utilizadas na construção do modelo.

Além destes aspectos citados anteriormente, a maior contribuição deste trabalho está na proposição de uma arquitetura de redução de dimensionalidade única, aplicável a estes diversos tipos de problemas.

Como restrições à utilização da abordagem proposta, podem ser considerados relevantes os seguintes aspectos:

1) Deve ser realizado o escalonamento das entradas e remoção da média. A não realização deste processo fará com que haja um viés nos escores, originado no processo de treinamento do algoritmo *backpropagation*. Este viés tenderá a determinar um maior escore para aquelas características de entrada cuja média absoluta tende a ser mais próxima a 1. Já as características de entrada cuja média absoluta for mais próxima a 0, tenderão a ter um escore também próximo a 0.

2) O treinamento inicial da RNA com todas as características de entrada deverá produzir um modelo com desempenho no mínimo equivalente a um “aprendiz fraco”. Caso o treinamento inicial não consiga aprender absolutamente nada sobre os dados, então não haverá nenhum conhecimento relevante expresso nos pesos sinápticos e as saídas geradas pela rede serão, em média, iguais à incerteza máxima. Dentro deste escopo, a aplicação do modelo neural de aprimoramento progressivo não irá gerar nenhum benefício adicional em relação aos modelos convencionais de RNAs.

Com isso, conforme demonstrado no capítulo 5, o escore utilizado reflete a sensibilidade média das saídas da camada oculta em relação à característica considerada. Logo, obedecidas as restrições mencionadas anteriormente, garante-se a validade da hipótese de pesquisa.

Como trabalhos futuros, similarmente ao que é evidente na área de neurociência, a área de redes neural apresenta uma área de estudo muito interessante que é a codificação neural. Assim, a tradução de dados para pesos sinápticos e a respectiva operação inversa consistem em uma área extremamente interessante e pouco explorada. A grande restrição de muitos pesquisadores à utilização de redes neurais é a criação de modelos do tipo caixa-preta. A partir do momento que os pesos sinápticos, que representam o conhecimento aprendido pela rede, passarem a ser mais explorados poderão ser um poderoso substrato para abrir a caixa-preta e contribuir para a interpretabilidade dos modelos neurais.

Adicionalmente, pode ser explorada a aplicação da abordagem proposta a outros algoritmos de treinamento de redes neurais MLP, a fim de demonstrar a generalidade da proposta. A aplicação da abordagem sobre redes neurais recorrentes também poderia constituir-se em uma nova fronteira a ser explorada.

Por fim, deve-se salientar que a redução de dimensionalidade demonstra uma realidade expressa nos dados em um determinado instante do tempo. Como o mundo real é dinâmico, consequentemente modelos que visam descrever o mundo real também são. Assim o processo de redução de dimensionalidade deve ser realizado sempre que houver uma mudança de realidade a fim de verificar o quanto a mudança impactou no resultado do processo de descoberta de conhecimento.

REFERÊNCIAS

ACKOFF, R. L. **From data to wisdom**. Journal of Applied Systems Analysis. Vol. 16, 1989. p. 3-9.

ALPAYDIN, E. **Introduction to Machine Learning**. 2 ed. Cambridge: MIT Press, 2010.

BELLMAN, R. **Adaptive Control Processes: A Guided Tour**. Princeton: Princeton University Press, 1961.

BERRY, M. J. A.; LINOFF, G. S. **Data mining techniques for marketing, sales, and customer relationship management**. 2. ed. Indianapolis: Wiley Publishing Inc, 2004.

BISHOP, C. M. **Neural networks for pattern recognition**. New York: Oxford University Press, 1995.

BOX, G. E. P.; JENKINS, G. M.; REINSEL, G. C. **Time Series Analysis, Forecasting and Control**. 3. ed. Holden-Day. Series G, 1976.

CAMARGO, S. S. **Mineração de dados: um estudo de caso sobre parâmetros petrográficos e petrofísicos dos arenitos da formação de Uerê**. 2005. 48 f. Relatório de Pesquisa – Instituto de Informática, UFRGS, Porto Alegre.

CAMARGO, S. S.; ENGEL, P. M. A Heuristic Approach for Dimensionality Reduction in Neural Modeling. In: IV International Symposium on Mathematical and Computational Biology, Biomat, 2007.

CAMARGO, S. S.; ENGEL, P. M. MiRABIT: A new algorithm for mining association rules. In: INTERNATIONAL CONFERENCE OF THE CHILEAN COMPUTER SCIENCE SOCIETY, SCCC, 22, 2002, **Proceedings...** Copiapó: IEEE Press, 2002.

CAMARGO, S. S.; ENGEL, P. M. Time Series Prediction with Focused Time Lagged Feed-Forward Networks. In: INTERNATIONAL SYMPOSIUM ON FORECASTING, ISF, 25, 2005, San Antonio, Texas, 2005. p. 123.

CAMARGO, S. S. ; ENGEL, P. M. Uma nova métrica para redução de dimensionalidade em modelos de aprendizado neural. In: CONGRESO ARGENTINO DE CIÊNCIAS DE LA COMPUTACIÓN, CACIC, XV, 2009, **Anales…** San Salvador de Jujuy, 2009.

CAMARGO, S. S. ; ENGEL, P. M. A Progressive Enhancement Neural Model to Predict Reservoir Quality in Sandstones. In: Third Southern Conference on

Computational Modeling, 2010, Rio Grande, Brasil. 2010 Third Southern Conference on Computational Modeling, 2010-a. IEEE Press. (aceito para publicação)

CAMARGO, S. S. ; ENGEL, P. M.. A Progressive Enhancement Neural Model to Predict Reservoir Quality in Sandstones. Vetor (FURG), 2010-b. (aceito para publicação)

CAMPOS, R.; CAMARGO, S. S.; ENGEL, P. M.; SILVA, S. C.; GONZALEZ, F. H. D. Use of metabolic indicators to predict milk quality using an artificial neural network based model. In: CONGRESS OF THE INTERNATIONAL SOCIETY OF ANIMAL CLINICAL BIOCHEMISTRY, ISACB, 12, 2006, Istanbul – Turquia, 2006.

CIOS, K. J. et al. **Data Mining: A knowledge discovery approach**. New York: Springer, 2007.

COVER, T.; HART, P. Nearest Neighbor Pattern Classification. **IEEE Transactions on Information Theory**. 13, 1967. p. 21-27.

COVER, T. M.; THOMAS, J. A. **Elements of Information Theory**, 2. ed. New Jersey: John Wiley and Sons, 2006.

DALGAARD, P. **Introductory Statistics with R**. New York: Springer, 2002.

EFFROYMSON, M. A. Multiple regression analysis, In: A. Ralston, and H. S. Wilf (Eds), **Mathematical Methods for Digital Computers**, Wiley, New York, 1960. p.191-203.

ENGEL, P. M. **Criação de Modelos da Qualidade de Reservatórios pela Aplicação de Técnicas de Descoberta de Conhecimento sobre Parâmetros Petrográficos e Petrofísicos de Arenitos – DC3PA**, 2005. 9 f. Projeto de Pesquisa – Instituto de Informática, UFRGS, Porto Alegre.

FAYYAD, U. M. et al. From data mining to knowledge discovery: an overview. In: Fayyad, U. M. et al. Advances in Knowledge discovery and data mining. Menlo Park: MIT Press, 1996. p. 37-54.

FREEMAN, J. A.; SKAPURA, D. M. **Neural networks: algorithms, applications and programming techniques**. New York: Addison-Wesley, 1991.

FRICKÉ, M. **The Knowledge pyramid: a critique of the DIKW hierarchy**. Journal of Information Science. Vol. 35, N. 2. 2009. p. 131-142.

FOUNTAIN, T.; ALMUALLIM, H.; DIETTERICH, T. G. Learning with many irrelevant features. Technical Report, UMI Order Number: 91-30-04, Oregon State University, 1991.

FUKUNAGA, K. **Introduction to Statistical Pattern Recognition**. 2. ed. New York: Academic Press, 1990.

GAONA, R. C. **Modelagem da composição química do leite através de indicadores metabólicos em vacas leiteiras de alta produção**. 2005. 114 f. Tese de doutorado – Faculdade de Veterinária, UFRGS, Porto Alegre.

GERTHEISS, J.; TUTZ, G. **Feature Selection and Weighting by Nearest Neighbor Ensembles**. 2008. 26 f. Technical Report – Department of Statistics, University of Munich, Munich.

GUYON, I.; ELISSEEFF, A. An Introduction to Variable and Feature Selection. In: **Journal of Machine Learning Research**. v. 3, 2003. p. 1157-1182.

GUYON, I. et al. **Feature Extraction: Foundations and Applications**. New York: Springer, 2006.

HAGAN, M. T.; DEMUTH, H.B.; BEALE, M. **Neural Network Design**. Thomson Learning, 1995.

HAN, J.; KAMBER, M. **Data mining: concepts and techniques**. San Francisco: Morgan Kauffman, 2001.

HAND, D.; MANILLA, H.; SMYTH, D. **Principles of Data Mining**. Cambridge: MIT Press, 2001.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The elements of statistical learning: Data mining, inference and prediction**. New York: Springer, 2001.

HAUPT, R. L.; HAUPT, S. E. **Practical Genetic Algorithms**. 2nd Edition. New Jersey: John Wiley & Sons, 2004.

HAYKIN, S. **Neural Networks: A Comprehensive Foundation**. 2nd Edition. Delhi: Prentice-Hall, 1999.

HUA, J. et al. Optimal number of features as a function of sample size for various classification rules. **Bioinformatics**, v. 21, n. 8, 2005. p. 1509-1515.

JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical Pattern Recognition: A Review. In: **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 22, n. 1, 2000. p. 4-37.

KANDEL, E. R.; SCHWARTZ, J. H.; JESSELL, T. M. **Principles of Neural Science**. 4th edition. New York: McGraw-Hill Medical, 2000.

KANTARDZIC, M. **Data Mining: Concepts, Models, Methods, and Algorithms**. New York: John Willey & Sons, 2002.

KECMAN, V. **Learning and soft computing: support vector machines, neural networks, and fuzzy logic models**. Cambridge: MIT Press, 2001.

KIRA, K; RENDELL, L. A. The Feature Selection Problem: Traditional Methods and a New Algorithm. In: **Proc. 10th National Conf. on Artificial Intelligence**, MIT Press, 1992. p. 129-134.

KOHAVI, R.; JOHN, G. H. Wrappers for feature subset selection. **Artificial Intelligence**, v. 97, n. 1-2, 1997. p. 273-324.

KOLODNER, J. L. **Case-Based Reasoning**. San Francisco: Morgan Kaufmann, 1993.

KONAR, A. **Artificial intelligence and soft computing: behavioral and cognitive modeling of the human brain**. Boca Raton: CRC Press, 2000.

KONONENKO, I. Estimating attributes: analysis and extensions of RELIEF. **Proc. 1994 European Conf. Machine Learning**, LNAI 784, 171-182, 1994.

LAROSE, D. T. **Discovering knowledge in data: an introduction to data mining**. [S.l.]: John Wiley & Sons, 2005.

LAROSE, D. T. **Data Mining Methods and Models**. New Jersey: John Wiley & Sons, 2006.

LEE, K. Y.; EL-SHARKAWI, M. A. **Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems**. New Jersey: John Wiley & Sons, 2008.

LIMA R. D.; DE ROS, L. F. The role of depositional setting and diagenesis on the reservoir quality of Devonian sandstones from the Solimões Basin, Brazilian Amazonia, **Marine and Petroleum Geology**, 19, 2002. p. 1047-1071.

LIU, H.; SETIONO, R. Feature Selection and Classification: a probabilistic wrapper approach. In: IEA-AIE, AAII, 17., 1996. **Proceedings...** Menlo Park, CA: Press: The MIT Press, 1996.

LUGER, G. F.; STUBBLEFIELD, W. A. **Artificial Intelligence: Structures and Strategies for Complex Problem Solving**. 3. ed. [S.l.]: Addison Wesley Longman, 1998.

LYMAN, P.; VARIAN, H. R. **How Much Information**. Berkeley, [s.n.]. Out. 2003. Disponível em: <http://www.sims.berkeley.edu/how-much-info-2003>. Acesso em: Mai. 2007.

MACKEY, D. J. C. **Information Theory, Inference and Learning Algorithms**. Cambridge: Cambridge University Press, 2003.

MAIMON, O.; ROKACH, L. (Editores) **Data Mining and Knowledge Discovery Handbook**. New York: Springer, 2005.

MICHALEWICZ, Z.; FOGEL, D. B. **How to Solve It: Modern Heuristics**. New York: Springer, 2000.

MIKLES, J.; FIKAR, M. **Process Modeling, Identification and Control**. New York: Springer, 2007.

MITCHELL, T. M. **Machine learning**. New York: McGraw-Hill, 1997.

MITRA, S; ACHARYA, T. **Data Mining: multimedia, soft computing and bioinformatics**. New Jersey: John Willey & Sons, 2003.

MUNAKATA, T. **Fundamentals of the New Artificial Intelligence: Neural, Evolutionary, Fuzzy and More**. London: Springer-Verlag, 2008.

MYATT, G. J. **Making Sense of Data: a practical guide to exploratory data analysis and data mining**. New Jersey: John Willey & Sons, 2007.

NAVOT, A. et al. Nearest neighbor based feature selection for regression and its application to neural activity. In: **Advances in Neural Information Processing Systems** V. 18, 2006. p. 995-1002, MIT Press.

OLSON, D. L.; DELEN, D. **Advanced Data Mining Techniques**. Berlin: Springer Verlag, 2008.

PANINSKI, L.; PILLOW, J.; LEWI, J. Statistical models for neural encoding, decoding, and optimal stimulus design. In: **Progress in Brain Research** V. 165, 2007. p. 493-507, Elsevier.

QUINLAN, J. R. Induction of Decision Trees. **Machine Learning**. v. 1, n. 1, 1986. p. 81-106.

QUINLAN, J. R. **C4.5: Programs for Machine Learning.** San Mateo, CA: Morgan Kaufmann, 1993.

RESTA, P. **Information and Communication Technologies in Teacher Education: A Planning Guide**. Paris: UNESCO, 2002.

RIEDMILLER, M.; BRAUN, H. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In **Proc. of the IEEE Intl. Conf. on Neural Networks**, 1993. p. 586-591, San Francisco.

RUD, O. P. **Data mining cookbook: modeling data for marketing, risk and customer relationship management**. New York: John Wiley & Sons, 2001.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. New Jersey: Prentice-Hall, 1995.

SARKER, R. A.; ABBASS, H. A.; NEWTON, C. **Heuristic & Optimization for Knowledge Discovery**. London: Idea Group Publishing, 2002.

SOUMEN, C. **Data Mining: Know it all**. Burlington: Elsevier, 2009.

SUMATHI, S.; SIVANANDAM, S. N. **Introduction to Data Mining and its applications**. Berlin: Springer-Verlag, 2006.

SYMEONIDIS, A. L.; MITKAS, P. A. **Agent intelligence through data mining**. New York: Springer, 2005.

TAYLOR, B. J. (Editor) **Methods and Procedures for the verification and validation of artificial neural networks**. New Jersey: Springer, 2006.

TENENBAUM, J. B., DE SILVA, V.; LANGFORD, J. C. A global geometric framework for nonlinear dimensionality reduction. In: *Science Magazine*, V.290 N.5500, 2000. p. 2319-2323.

THEODORIDIS, S.; KOUTROUMBAS, K. **Pattern Recognition**. 2. ed. London: Academic Press, 2003.

TSIEN, J. S. The Memory Code. **Scientific American**, New York, v. 297, n.1, 2007. p. 52-59.

WANG, L.; XIUJU, F. **Data mining with computational intelligence**. Berlin: Springer-Verlag, 2005.

WEBB, A. R. **Statistical Pattern Recognition**. Malvern: John Wiley & Sons, 2002.

WITTEN, A. A.; FRANK, E. **Data Mining: Practical Machine Learning Tools and Techniques**. San Francisco: Morgan Kaufmann Publishers, 2005.

YAMPOLSKIY, R. V.; GOVINDARAJU, V. Similarity Measure Functions for Strategy-Based Biometrics. Proceedings of World Academy of Science, Engineering and Technology, V. 18, 2006.

YE, N. **Handbook of Data Mining**. London: Lawrence Erlbaum Associates Publishers, 2003.

XU, R.; WUNSCH, D. C. **Clustering**. New Jersey: John Wiley & Sons, 2009.