UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Maurício Braga de Paula

Indução automática de árvores de decisão

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

João Bosco da Mota Alves, Dr.

Florianópolis, novembro de 2002

Indução automática de árvores de decisão

Maurício Braga de Paula

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação (Sistemas de Conhecimento) e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

_____

Fernando Álvaro Ostuni Gauthier, Dr.

Banca Examinadora

_____

João Bosco da Mota Alves, Dr.

_____

Luiz Fernando Jacintho Maia, Dr.

_____

Ilson Wilmar Rodrigues Filho, Dr.

AGRADECIMENTOS

Ao Prof. João Bosco da Mota Alves e ao Prof. Luiz Fernando Jacintho Maia pela orientação, confiança, incentivo e por terem acreditado na realização deste trabalho.

À Universidade Federal de Santa Catarina, em especial ao Curso de Pós-Graduação em Ciência da Computação.

Aos professores do PGCC, com os quais muito aprendi.

Aos professores membros da banca examinadora pela avaliação e contribuição para o aperfeiçoamento deste trabalho.

Aos amigos professores João Artur de Souza e Gertrudes Aparecida Dandolini pelos ensinamentos, apoio e contribuição para o enriquecimento deste trabalho.

Ao amigo Luiz Rodrigues Maia Neto pelos ensinamentos sobre o BSD.

A todos os meus colegas do Laboratório de Experimentação Remota (RExLab) e da Universidade do Sul de Santa Catarina.

Aos meus pais Ivanor e Neusa, que me ensinaram os verdadeiros caminhos a serem trilhados, revestindo minha existência de amor, carinho e dedicação.

À minha irmã Jaqueline, que sempre me apoiou.

À minha avó Diná (*in memorium*), pelo incentivo e acompanhamento nesta nova etapa de minha vida.

À minha namorada Angelisa Caramão de Araújo, pela paciência e compreensão.

A todos aqueles que de alguma forma contribuíram para a realização deste trabalho.

A Deus que torna tudo possível...

SUMÁRIO

LISTA DE FIGURAS ..... **v**

LISTA DE TABELAS ..... **vi**

RESUMO ..... **vii**

ABSTRACT ..... **viii**

1 Introdução **9**

1.1 Objetivos 11
1.1.1 Objetivo Geral 11
1.1.2 Objetivos Específicos 11
1.2 Metodologia de desenvolvimento da pesquisa 12
1.3 Estrutura do trabalho 12

2 Materiais e Métodos **14**

2.1 Aprendizado 14
2.2 Aprendizado de Máquina 15
2.3 Classificação 17
2.3.1 Definição 18
2.3.2 Aplicações 18
2.3.3 Algoritmos de Classificação 19

3 Classificação Estatística **21**

3.1 Discriminadores Lineares 23

4 Redes Neurais Artificiais **26**

4.1 Aplicações 27
4.2 O Neurônio Artificial 29
4.3 Arquiteturas 33
4.4 Aprendizado 35
4.4.1 Supervisionado 35
4.4.2 Não supervisionado 37
4.5 Redes Perceptron 38
4.5.1 Limitações: O problema do OU-EXCLUSIVO 40
4.6 Redes Multilayer Perceptron 41
4.7 Ensaio computacional: classificação de padrões 43

5 Aprendizagem Automática **47**

5.1 Árvores de Decisão 47
5.1.1 Processo geral de construção de árvores decisão 48
5.1.2 Técnicas para a seleção de atributos 49
5.2 Exemplo de indução de árvores de decisão 50
5.3 CLS 53
5.4 ID3 54
5.5 C4.5 66

6 Conclusões e Recomendações **81**

7 Referências Bibliográficas **83**

LISTA DE FIGURAS

Figura 1.1 – Metodologia de desenvolvimento 12
Figura 2.1 – Sistema de classificação 17
Figura 3.1 – Processo de inferência estatística (Martins, 2001) 22
Figura 3.2 – Separação de duas classe por uma reta 23
Figura 3.3 – Classificação (base íris) realizada pelo discriminador linear de Fisher (MICHIE et al, 1994) 25
Figura 4.1 - O neurônio de McCulloch e implementações de algumas funções booleanas (kovács, 1996) 30
Figura 4.2 - O Neurônio artificial (Tafner et al, 1996) 31
Figura 4.3 - Funções de transferência (Kovács, 1996) 32
Figura 4.4 - Rede neural artificial 32
Figura 4.5 - RNA de uma única camada 33
Figura 4.6 - RNA multicamada 34
Figura 4.7 - RNA feedforward ou acíclica 34
Figura 4.8 - RNA feedback ou cíclica 35
Figura 4.9 - O perceptron elementar de Rosemblatt (Bishop, 1995) 38
Figura 4.10 - A unidade de processamento do perceptron 39
Figura 4.11 - Plano que representa as combinações possíveis do XOR 41
Figura 4.12 – Interface de gerenciamento do SNNS 44
Figura 4.13 – Grafo da arquitetura da rede neural para a base íris 45
Figura 4.14 – Curva de aprendizagem (Íris): erro médio quadrado – número de épocas45
Figura 4.15 – Distribuição gráfica dos exemplares da base íris 46
Figura 5.1 - Exemplo de uma árvore de decisão binária (Weiss, 1991) 47
Figura 5.2 - Exemplo de uma árvore de decisão não binária (Weiss, 1991) 48
Figura 5.3 – Busca de estados do problema do presente 51
Figura 5.4 – Construção da árvore em nível parcial 61
Figura 5.5 – Árvore de decisão final 65
Figura 5.6 – Árvore de decisão da base íris gerada pelo MLC++ 66
Figura 5.7 - Árvore de decisão gerada pelo C4.5 da Tabela 5.6 69

LISTA DE TABELAS

Tabela 4.1 - Tabela verdade do ou-exclusivo 40

Tabela 5.1 - Tabela de decisão para o problema do presente 51

Tabela 5.2 – Conjunto de treinamento 55

Tabela 5.3 – Subconjunto (janela) T_1: céu igual a sol 63

Tabela 5.4 – Subconjunto (janela) T_2: céu igual a nublado 63

Tabela 5.5 – Subconjunto (janela) T_3: céu igual a chuva 64

Tabela 5.6 – Tabela de registros que informam as condições climáticas para a realização de um jogo de golfe com a utilização de atributos discretos e contínuos 68

Tabela 5.7 – Subconjunto (janela) T_1: atributos contínuos 69

Tabela 5.8 – Subconjunto (janela) T_1 após a aplicação da função de permutação 70

Tabela 5.9 – Relação de retirada de caso patológico por número de elementos mal classificados (erro) 79

RESUMO

Com o considerável aumento da quantidade de informações disponíveis, as capacidades de aquisição automática de conhecimento têm se tornado muito importante.

A capacidade de aprendizagem e de aplicação do conhecimento é uma das características da inteligência humana e uma das principais áreas de análise da Inteligência Artificial. As atividades humanas mais comuns exibem a aplicação do conhecimento adquirido pelo homem, podendo ser consideradas tarefas de classificação; termo no qual decorre da necessidade de uma tomada de decisão ou da realização de uma previsão com base em informações disponíveis.

O Aprendizado de Máquina (AM), um dos nichos da Inteligência Artificial, é uma das eficazes maneiras de adquirir inteligência de qualquer sistema computacional.

Este trabalho consiste na construção de um procedimento de indução automática de árvores de decisão simbólica a partir de amostras de dados com atributos não binários. O objetivo deste, é extrair informações de um conjunto de treinamento de instâncias possivelmente conhecidas do problema e subseqüentemente classificar novas instâncias em suas respectivas classes.

ABSTRACT

With the considerable increase of the amount of available information, the capacities of automatic acquisition of knowledge have become very important.

The learning capacity and of application of the knowledge it is one of the characteristics of the human intelligence and one of the main areas of analysis of the Artificial Intelligence. The more common human activities exhibit the application of the acquired knowledge for the man, could be considered classification tasks; term in which elapses of the need a decision or of the accomplishment of a forecast based on available information.

The Machine Learning (ML), one of the niches of the Artificial Intelligence, is one in the effective ways of acquiring intelligence of any computational system.

This work consists of the construction of a procedure of automatic induction of trees of symbolic decision starting from samples of data with non binary attributes. The objective of this, is to extract information of a group of training instances possibly known of the problem and subsequently to classify new instances in your respective classes.

1 Introdução

Com o considerável aumento da quantidade de informações disponíveis, as capacidades de aquisição automática de conhecimento têm se tornado muito importante. Os inúmeros fatores envolvidos na descoberta do conhecimento tem tornado inviável a análise humana.

Para que os sistemas computacionais possam exibir comportamento inteligente, o conhecimento do domínio do problema deve ser representado na forma na qual os sistemas de informação trabalham. Esta aquisição pode ser feita de forma explícita ou implícita (PILA, 2001): a aquisição explícita do conhecimento está relacionada ao contato do especialista do domínio com o engenheiro de conhecimento, o qual usufrui alguns artifícios e técnicas como questionários, entrevistas para a aquisição do conhecimento (REZENDE, PUGLIESI, 1998 apud PILA, 2001). A aquisição do conhecimento implícito é mais complexa, haja vista, o não fornecimento de informações pelo especialista no domínio do problema em questão.

A Inteligência Artificial, um dos nichos da ciência da computação, procura compreender os princípios que tornam possíveis a modelagem de sistemas inteligentes de acordo com as características da inteligência apresentada no comportamento humano.

As atividades humanas mais comuns exibem a aplicação do conhecimento adquirido pelo homem podendo ser consideradas tarefas de classificação; termo no qual decorre da necessidade de uma tomada de decisão ou da realização de uma previsão com base em informações disponíveis.

No domínio desta dissertação, o termo classificação consiste na construção de um procedimento capaz de prever a classe de um exemplo com base num conjunto de atributos que o descrevem.

A tarefa de classificação de informações é exercida por um programa que realiza o aprendizado conforme um conjunto de exemplos que lhe é fornecido. Nesta base de observações a classe em que cada exemplar pertence é conhecida.

No decorrer dos anos, inúmeros métodos e modelos de classificação têm surgido. Estes podem ser classificados em três áreas distintas: a *Classificação Estatística*, a *Aprendizagem Automática* e as *Redes Neurais*.

Apesar destes métodos estarem enquadrados em áreas diferentes, todos tentam buscar metodologias capazes de tomar decisões; e ainda serem suficientemente genéricos para serem aplicados com determinado sucesso numa gama de problemas práticos (MICHIE et al, 1994).

Embora diversos modelos matemáticos que tem surgido nas diferentes áreas do conhecimento relacionadas à classificação, as árvores de decisão têm sido consideradas como os modelos mais adequados para a extração do conhecimento para a Aprendizagem Automática, pois são simples e podem ser facilmente compreendidos pelos humanos; e podem ser expressos numa sub-rotina em qualquer linguagem de programação (MICHIE et al, 1994, QUINLAN, 1992).

O Aprendizado de Máquina é uma das eficazes maneiras de adquirir inteligência de qualquer sistema computacional (WANG et al, 2000).

As árvores de decisão, subárea de aprendizado de máquina, pesquisa métodos computacionais relacionados à aquisição automática de novos conhecimentos, novas habilidades e novas formas de organizar a informação (MITCHEL, 1997 apud PILA, 2001). Segundo QUINLAN (1979), a indução de árvores de decisão é uma maneira eficiente de aprendizado por exemplos. As árvores de decisão são uma das mais populares escolhas para o aprendizado e raciocínio de sistemas que trabalham com aprendizado supervisionado.

Os algoritmos de árvores de decisão proporcionam uma das melhores abordagens metodológicas à aquisição de conhecimento simbólico. O objetivo do aprendizado é extrair informações de um conjunto de treinamento de instâncias possivelmente conhecidas do problema; subseqüentemente classificar novas instâncias em suas respectivas classes.

Uma árvore de decisão pode ser vista como uma hierarquia de nodos, os quais são representados por um nodo raiz e por outros nodos que foram distribuídos conforme as suas instâncias de classificação. O nodo raiz e os outros nodos internos são chamados de nodos de teste, pois aponta (arco de ramificação) para um outro nodo de decisão, cuja

hierarquia é menor. A expansão da árvore corresponde à avaliação de uma determinada instância conforme suas características ou atributos.

Quando todos os elementos que compõe um conjunto bem definido de informações são classificados de acordo com algum determinado critério, não há domínio de conhecimento disponível que possa demarcar os limites do mesmo; e o objetivo para tal classificação é adquirir conhecimento de forma a descrever a qual classe o exemplar pertence.

Como o objetivo é inferir novos conhecimentos, a indução é uma das técnicas empregadas para tal propósito.

Segundo DURKIN (1994), a indução é o processo de raciocínio sobre um dado conjunto de informações para concluir princípios gerais ou regras. A habitual tarefa de inferência consiste em realizar a predição discreta sobre um determinado objeto, dados alguns detalhes (atributos ou características) referente ao mesmo.

1.1 Objetivos

1.1.1 Objetivo Geral

Desenvolver um modelo de indução automática de árvores de classificação simbólica a partir de amostras de dados com atributos não binários.

1.1.2 Objetivos Específicos

Como objetivos específicos este trabalho apresenta:

a) Selecionar atributos relevantes para a construção de árvores de decisão.

b) Realizar a extração de conhecimento de dados e aprendizagem automática.

c) Desenvolver uma função para determinar o ganho de informação em atributos contínuos.

d) Analisar casos (exemplares) patológicos que degradem a construção da árvore de decisão.

e) Realizar a análise de sub árvores com o intuito de obter o corte de ramos supérfluos.

1.2 Metodologia de desenvolvimento da pesquisa

Para alcançar os objetivos propostos, o trabalho foi dividido em etapas, que juntas, integram o trabalho desenvolvido.

A primeira etapa consiste do levantamento bibliográfico (através de livros, revistas, internet, artigos, dentre outros) acerca do problema de concepção da indução automática de árvores de decisão; sobre o conjunto de processos que formam o conhecimento: desenvolvimento motor, a habilidade cognitiva, e a organização do conhecimento; sobre as capacidades de aprendizagem e da aplicação do conhecimento; sobre reconhecimento de padrões (extração de características, classificação) e inteligência artificial.

Uma vez conhecidos os conceitos e processos da construção da árvore de decisão, buscam-se formas alternativas de abordar o problema. Desenvolve-se o método que seleciona atributos (discretos e contínuos) relevantes para a construção de árvores de decisão.

Tendo desenvolvido o modelo, o próximo passo é a sua avaliação. Nesta etapa implementa-se o sistema. O sistema é utilizado para construir uma árvore de decisão, ou seja, um classificador automático para amostras de dados. Nesta fase são feitos os ajustes dos parâmetros.

A última etapa consiste das considerações finais sobre a pesquisa.

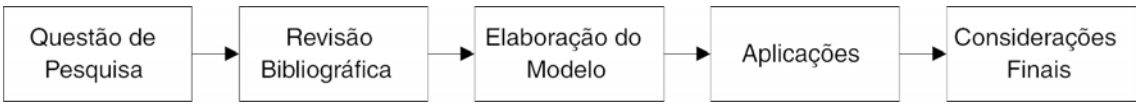


Figura 1.1 – Metodologia de desenvolvimento

1.3 Estrutura do trabalho

Este trabalho está estruturado em 6 capítulos, seguidos da referência bibliográfica.

O capítulo 1 – Introdução – apresenta os aspectos preliminares do trabalho, bem como a apresentação do tema, sua importância, os objetivos do trabalho e sua respectiva estruturação.

No capítulo 2 – Materiais e Métodos – tem-se uma visão breve e teórica sobre a ação ou efeito de aprender dos humanos e das máquinas. Também são apresentados alguns conceitos e definições fundamentais na área da classificação.

No capítulo 3 – Classificação Estatística – apresenta uma breve revisão teórica sobre a contribuição dos métodos estatísticos no processo geral e aplicado dos classificadores.

O capítulo 4 – Redes Neurais – tem-se uma descrição geral das Redes Neurais Artificiais, abordando suas principais características e funções aplicadas ao processo de classificação.

No capítulo 5 – Aprendizagem automática – apresenta os conceitos relacionados à aprendizagem automática, abordando as árvores de decisão. Ainda é realizada uma descrição detalhada do processo de construção das árvores de decisão dos principais algoritmos.

Finalmente o capítulo 6 traz as conclusões e as considerações finais sobre a pesquisa desenvolvida.

2 Materiais e Métodos

2.1 Aprendizado

A ação ou efeito de aprender compreende a aquisição de novas formas de conhecimento: o desenvolvimento motor e a habilidade cognitiva, a organização do novo conhecimento e as descobertas de novos fatos e teorias através da observação e experimentação.

O ser humano possui a capacidade de aprender habilidades motoras e cognitivas. A visão é um dos principais meios para adquirir habilidades motoras; e mesmo se não tivermos a visão, ainda somos capazes de aprender as mesmas habilidades motoras. A linguagem é muito importante na aquisição de habilidades cognitivas. A fala é uma característica particular do homem, apesar de não ser inata: a fala tem que ser aprendida. O aprendizado é a chave da superioridade da inteligência humana (MONARD et al, 1997).

Desde os primórdios da era computacional, pesquisas têm sido realizadas para que as capacidades de aprendizado humano possam ser implantadas em computadores; e modelar este propósito é um dos maiores desafios para os pesquisadores no campo da Inteligência Artificial. O estudo e a modelagem de processos de aprendizagem em computadores e suas múltiplas manifestações constituem o objetivo principal do estudo de aprendizado de máquinas (POZO, 2002).

Segundo POZO (2002), o campo de aprendizado de máquinas está organizado em três principais nichos de pesquisa:

- Estudos orientados a trabalho: o desenvolvimento e a análise de sistemas aprendizes para a melhoria da performance de determinados grupos de trabalho.
- Simulação cognitiva: a investigação e simulação computacional de processos de aprendizado humano.
- Análises teóricas: a exploração teórica do espaço de possibilidades metódicas de aprendizagem e algoritmos independentes do domínio.

O aprendizado humano continua a ser objeto de estudo para a comunidade científica; questiona-se os mecanismos de aprendizado, a inata habilidade de adquirir fatos, habilidades e outros conceitos abstratos (WEISS, 1991). O homem é um modelo de observação para o estudo das capacidades cognitivas para o campo de aprendizado de máquinas, visto o vasto espaço de possibilidades de métodos de aprendizagem que este apresenta.

2.2 Aprendizado de Máquina

Desde a concepção dos computadores tem-se questionado quando estas máquinas poderão ser construídas para aprender. O impacto resultante de tal façanha seria fascinante e ao mesmo tempo inspiraria cuidado. Uma máquina seria capaz de reproduzir o comportamento humano, tal como a fala. Entretanto a tarefa de se dotar comportamento inteligente nestas máquinas não é uma tarefa trivial. Muitos dos sistemas computacionais são desenvolvidos com o intuito de aprender sobre um domínio específico.

Um sistema de aprendizado ou aprendizado de máquina é um programa de computador que realiza decisões com base na experiência contida em casos resolvidos com sucesso. Ao contrário de um sistema especialista, o qual resolve problemas de uma maneira semelhante a um especialista humano, um sistema de aprendizado puro pode usufruir diferentes técnicas para explorar o poder computacional de um computador, considerando seu relacionamento com o processo de cognição humana (WEISS, 1991). Estas técnicas incluem métodos puramente matemáticos, bem como outros que realizam a busca sistemática numa vasta gama de possibilidades. Os objetivos dos sistemas de aprendizado não são diferentes dos comumente citados em sistemas especialistas. Estes são: lidar com a tomada de decisão de problemas reais complexos, e resolver estes problemas.

Pessoas estão constantemente enfrentando tomadas de decisão perante aos fatos da vida - sejam estes problemas profissionais ou particulares. Na escolha de como tomar uma determinada decisão, elas confiam esta decisão em atitudes e experiências prévias. Alguns indivíduos podem ser considerados especialistas em uma determinada área ou campo devido ao considerável acúmulo de experiência; e são conhecidos como

profissionais do conhecimento por realizarem decisões precisas, explanar e manter suas conclusões.

A quantidade de especialistas, em diversos nichos de pesquisa, está escassa e a codificação do conhecimento tende a ser bastante limitada e árdua na prática. O conhecimento extraído de um determinado problema e a documentação na forma de casos já resolvidos é a única fonte de conhecimento.

Da perspectiva de desenvolvimento de um sistema, há várias razões que demonstram o considerável aumento de interesse por sistemas de aprendizado. Novos métodos formais e novas técnicas de confecção tem sido desenvolvidas e o poder computacional dos computadores atuais proporcionam que simulações sejam realizadas.

O auxilio do computador na tomada de decisão está dentre os primeiros programas de pesquisa em análise médica, análise de sinais, análise de imagem, dentre outras aplicações em reconhecimento de padrões.

O processo de aprendizagem automática sustenta a idéia de incorporar conhecimento em um sistema sem a necessidade de um engenheiro do conhecimento. Um argumento a favor da construção de sistemas de aprendizado é que estes apresentam potencial para exceder a performance dos especialistas no domínio do assunto e ainda descobrir novas relações no meio de conceitos e hipóteses, examinando um conjunto de exemplares resolvidos com sucesso.

O conceito de aprendizado de máquina é bastante vasto, e este trabalho abordará uma das tarefas mais comuns de aprendizado, a classificação. Para problemas de classificação, um sistema de aprendizado pode ser visto como um método que por si só constrói um sistema de tomada de decisão, chamado classificador. Uma simples maneira de representar um classificador é uma "caixa preta" que produz uma decisão para cada padrão de dados admissível apresentado a ele. A Figura 2.1 ilustra a estrutura de um sistema de classificação. Este aceita um padrão de dados como entrada e produz uma decisão como saída.

Um sistema de aprendizado possui disponível um conjunto finito de exemplos de casos previamente resolvidos. Os dados para cada caso consistem de um modelo padrão de observações e sua correspondente classificação. O objetivo destes sistemas de aprendizado é personalizar a estrutura do classificador para um problema específico,

através de um modo comum de relacionar qualquer padrão particular de observações a uma classe específica.

A representação básica de um problema de classificação é de certa forma simples. Cada amostra de um problema resolvido consiste de observações e da correspondente pertinência da classe correta. Normalmente, uma simples conclusão ou classe é resultante de um dado padrão de observações.

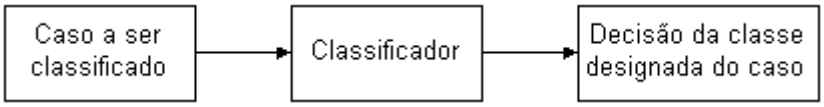


Figura 2.1 – Sistema de classificação

Um conjunto de amostras contém os dados que o sistema usará para achar as regras de decisão genéricas para o classificador.

2.3 Classificação

A classificação é encontrada numa considerável gama de atividades desenvolvidas pelos seres humanos. E dá-se quando há a necessidade de se tomar alguma decisão baseada em alguma informação prévia, vivenciada por este, e disponível para a decisão.

Para a distribuição de classes, é analisado o problema de construir um algoritmo capaz de prever a classe de um exemplar baseado num conjunto de características e/ou atributos desse mesmo exemplar. Este processo é normalmente obtido através de um programa de aprendizagem que estabelece a árvore de decisão a partir de um conjunto de treinamento para qual a verdadeira classe já é conhecida. Este procedimento virá a identificar a classe de novas observações.

A capacidade de aprendizagem e de aplicação do conhecimento é uma das características da inteligência humana e uma das principais áreas de análise da esfera de estudo da Inteligência Artificial.

Embora haja diversas metodologias e modelos propostos para realizar o processo de classificação, as árvores de decisão evidenciam-se pela maneira na qual extraem o conhecimento e realizam a aprendizagem automática. Uma árvore de decisão é uma estrutura em árvore capaz de prever a classe de um exemplar baseado em decisões realizadas sobre os atributos que descrevem o mesmo.

Dentre os vários modelos de indução de árvores de decisão, destacam-se o ID3 e o C4.5, desenvolvido por QUINLAN (1992). Estes algoritmos são adotados como ponto de referência para estudo e desenvolvimento de novos métodos de classificação e indução de árvores de decisão.

Nesta dissertação são expostos os conceitos fundamentais referentes à classificação. Também são apresentados alguns métodos utilizados na classificação de acordo com as três principais áreas segundo a cronologia histórica: a *Classificação Estatística*, a *Aprendizagem Automática* e as *Redes Neurais*.

2.3.1 Definição

A classificação possui dois significados distintos. Dado um conjunto de observações, o objetivo é encontrar um conjunto de classes ou grupos (*cluster*). Conhecendo-se o ponto de partida a existência de um determinado número de classes, o foco é encontrar uma regra que possa classificar um novo exemplar em uma das classes previamente definidas (MICHIE et al, 1994). A primeira acepção é normalmente conhecida como aprendizagem não supervisionada e a segunda, como supervisionada. Neste trabalho será utilizada a segunda definição.

2.3.2 Aplicações

Há inúmeros fatores para se realizar a classificação de forma automática. Michie et al (1994) cita alguns exemplos da utilização destes algoritmos:

- O processo de classificação mecânico pode ser realizado de forma mais rápida, tal como a leitura automática de códigos postais, de forma a ordenar parte ou toda a correspondência de uma agência de correios.
- A concessão de crédito a um determinado cliente.
- No campo da medicina, utiliza-se tal procedimento para realizar um diagnóstico médico baseado em sintomas apresentados pelo paciente.
- Aplicações na área da meteorologia, onde a verdadeira classe de observação será somente conhecida no futuro. Predição do estado do tempo para a semana, de acordo com certos parâmetros de medidas meteorológicas.

2.3.3 Algoritmos de Classificação

A tarefa de classificação ocorre numa grande variedade de atividades humanas. A classificação acontece sempre que é necessário tomar uma decisão ou realizar uma predição baseada na informação disponível. Uma tarefa de classificação pode ser considerada como a aplicação formal de um método sempre que são tomadas decisões face novas situações (MICHIE et al, 1994).

Neste trabalho, o conceito de classificação é abordado de forma mais restrita. O problema consiste em desenvolver uma função capaz de atribuir uma classe, de um conjunto predefinido de classes, a cada exemplar de uma do conjunto de observações. A atribuição é realizada com base num conjunto de atributos que descrevem cada um dos exemplares pertencentes ao conjunto de treinamento.

O método de construção de um algoritmo de classificação é normalmente conhecido como reconhecimento de padrões, discriminação, ou aprendizagem supervisionada (MICHIE et al, 1994).

O conjunto de treinamento ou de exemplares utilizados para a construção de um algoritmo de classificação pode ser referenciado como o conjunto de observações, ou conjunto de dados de entrada. O termo conjunto de dados aplica-se normalmente ao conjunto de todos os dados de um determinado domínio que se encontra disponível. O conjunto de treinamento é normalmente formado por um subconjunto do conjunto de dados. O termo domínio refere-se ao contexto dos dados possam vir a ser utilizados para a tarefa de classificação.

A elaboração deste algoritmo de classificação e a predição da classe de um conjunto de treinamento são feitas a partir do conjunto de atributos ou características que descrevem estas observações.

No conjunto dos dados a serem estudados há a distinção entre atributos contínuos, cujos valores são números reais, e discretos, cujos valores encontram-se restritos ao um número finito de valores possíveis.

Embora as áreas em questão têm envolvido pesquisadores de diferentes vertentes assim como têm sido dirigidas para questões e propósitos diferentes, todos os grupos

procuram objetivos comuns; de acordo com MICHIE et al (1994), inúmeros métodos de classificação apresentam procedimentos com algumas peculiaridades, tal como:

- Igualar, sem superar, as decisões tomadas pelos humanos, com a vantagem de demonstrar consistência com determinado grau e clareza;
- Capaz de manipular uma grande variedade de problemas e com informações suficientes para serem generalistas o suficiente;
- Serem utilizados em aplicações práticas com sucesso.

3 Classificação Estatística

Os procedimentos, técnicas e métodos estatísticos são fundamentais para a compreensão de informações numéricas. A estatística é a ciência dos dados; ela envolve a coleta, classificação, sumarização, organização, análise e interpretação de dados (MARTINS, 2001).

A organização, sumarização e descrição de um conjunto de dados é chamada estatística descritiva. Através de técnicas como a construção de gráficos, tabelas, medidas de tendência/dispersão/assimetria, pode-se compreender o comportamento de uma variável expressa no conjunto de dados sob análise. Através da estatística descritiva são apresentados métodos numéricos para a determinação de medidas. Estas proporcionam o entendimento do conjunto de dados quantitativos (populações e amostras), oriundos de variáveis que são objeto de estudo.

Entende-se por população ou universo a totalidade de itens, e amostra por uma parte da população que é selecionada para análise. O objetivo principal da investigação é descrever características peculiares da população, ou seja, conhecer os parâmetros que a descrevem.

Para a estimação de características de uma população baseados nos resultados amostrais, utiliza-se métodos de inferência estatística. As bases da estatística inferencial foram estabelecidas pelos matemáticos Bernoulli, Gauss, Demoivre, entre outros. Os métodos sobre a estatística inferencial foram modelados e desenvolvidos somente no século XX por estatísticos como Fisher, Gosseti, entre outros (MARTINS, 2001).

Além das técnicas de inferência estatística e da estatística descritiva, há ainda as investigações que envolvem a coleta e análise de dados com propósitos de previsão.

Dentre as diversas maneiras para se coletar dados, a amostragem é a mais freqüente. No campo da estatística, normalmente, as pesquisas são realizadas por meio de estudo dos elementos que compõe uma amostra extraída da população na qual se pretende analisar. A amostragem é um campo da estatística que estuda a teoria e a prática da aquisição de um conjunto de dados (AMADO, 2001); é também considerado um método estatístico do qual se obtém conclusões sobre as características de um

conjunto de dados mediante o exame de um grupo parcial ou subconjunto do conjunto de dados.

Para se analisar uma população, a estatística utiliza informações da amostra para inferir sobre a população. De uma maneira genérica, tem-se uma população e almeja-se, por meio de uma amostra, conhecer alguma característica da mesma.

Segundo Martins (2001), o processo de inferência ou indução estatística exemplifica-se do seguinte modo: X é uma variável da população que se deseja estudar; seja O uma característica de X que se deseja conhecer; o parâmetro O é desconhecido; deste modo, necessita-se construir um estimador $\hat{O}$ que, mediante os elementos da amostra, ofereça um valor mais aproximado de O (Figura 3.1).

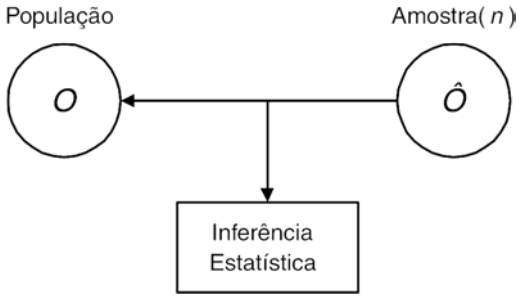


Figura 3.1 – Processo de inferência estatística (Martins, 2001)

Como o objetivo da estatística é conhecer populações por meio das informações amostrais; como as populações são caracterizadas por medidas numéricas descritivas (parâmetros), a estatística busca a realização de inferências sobre esses parâmetros populacionais desconhecidos.

De acordo com Martins (2001), os métodos para realizar inferências a respeito destes parâmetros pertencem a duas categorias:

- estimação: consiste em determinar estimativas dos parâmetros populacionais, ou seja, calcular a estimativa do parâmetro por ponto ou intervalo;
- teste de hipóteses: consiste na tomada de decisão relativa ao valor de um parâmetro populacional; neste caso, admite-se um valor hipotético para um parâmetro populacional, e com base nas informações da amostra, realiza-se um teste estatístico para aceitar ou rejeitar o valor hipotético.

A busca de associação entre variáveis é freqüentemente um dos propósitos das pesquisas empíricas. A possível existência de relação entre variáveis orienta análises e conclusões. No campo da estatística o foco é dirigido para as observações empíricas, não apenas como estas podem ser sintetizadas, mas como realizar o processo de inferências e decisões a partir de dados. A estatística reúne uma grande variedade de áreas, sendo algumas relevantes para o objetivo de criar classificadores eficientes a partir de um conjunto de observações (DUDA e HART, 1973 apud AMADO, 2001).

A análise discriminante pode ser caracterizada por apresentar um modelo probabilístico, o qual fornece a probabilidade de uma dada observação pertencer a uma determinada classe.

Outra área de suma importância na estatística é a análise de regressão linear simples e múltipla, as quais possuem como propósito a construção de modelos preditivos.

3.1 Discriminadores Lineares

Os discriminadores lineares são uma das formas mais comum de um classificador. O nome discriminador linear indica uma combinação linear de uma dada evidência que será utilizada para separar ou discriminar as classes e selecionar uma classe para um novo caso.

Para um problema, onde cada dado possui *d* atributos, significa geometricamente que as superfícies de separação entre os exemplos será de *d-1* hiperplanos. Se por exemplo, tivermos três atributos um plano será suficiente para separar as classes; para apenas dois atributos uma reta é suficiente. A Figura 3.2 representa a separação de duas classes, C_1 e C_2, com dois atributos a_1 e a_2, através de uma reta.

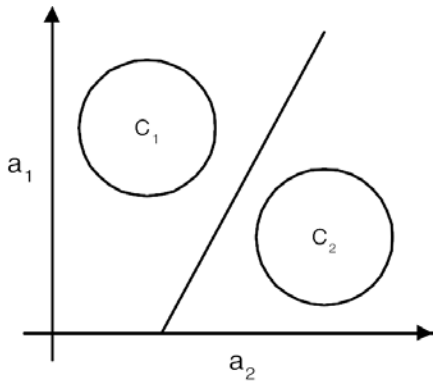


Figura 3.2 – Separação de duas classe por uma reta

Em determinadas situações, as classes são justapostas e conseqüentemente não podem ser completamente separadas por um plano. Um exemplo clássico deste problema é o OU exclusivo (XOR).

Uma forma geral para qualquer classificador linear é dada na equação abaixo, onde $\{a_1, a_2, ..., a_d\}$ é o vetor de atributos, d é o número de atributos, e w_i são constantes que devem ser estimadas (WEISS, 1991).

$$w_1 a_1 + w_2 a_2 \cdots w_d a_d - w_0 = 0 \qquad (3.1)$$

Em se tratando de discriminar várias classes uma de outra, pode-se definir um discriminador linear para cada classe, combinando-os com o propósito de separar as superfícies. No problema de discriminação de duas classes, um simples hiperplano serve para separa-las. Para várias classes deve-se combinar discriminantes para cada par de classes ou modificar o problema de decisão de forma que estes estejam dispostos numa seqüência de decisões binárias de cada classe versus todas as outras classes.

Um discriminador linear simplesmente implementa uma soma ponderada dos valores observados. Uma questão pertinente acerca dos valores para as constantes é como determinar os seus valores. Há, potencialmente, inúmeros números que poderiam ser testados. Busca-se reduzir o número de possibilidades de busca de valores satisfatórios para um determinado problema.

Um dos modelos mais antigos é o discriminador linear de Fisher, o qual está presente na maioria dos pacotes de softwares estatísticos. A idéia deste é dividir o espaço de amostras por uma série de linhas em duas dimensões, planos em três dimensões, e hiperplanos dimensões superiores.

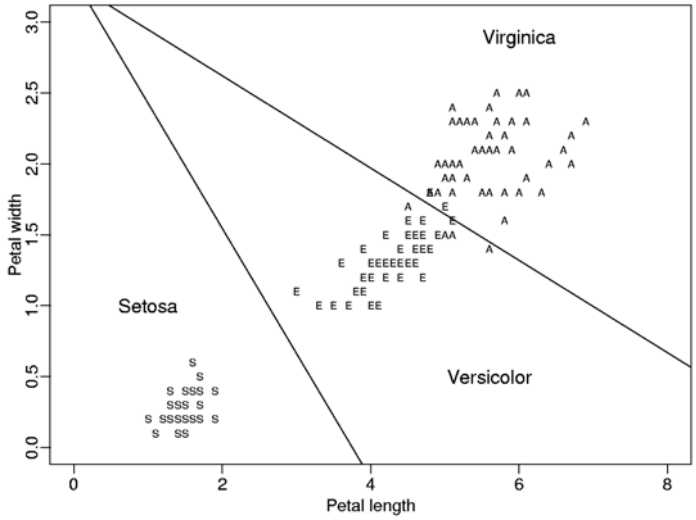


Figura 3.3 – Classificação (base íris) realizada pelo discriminador linear de Fisher (MICHIE et al, 1994)

4 Redes Neurais Artificiais

A tecnologia das Redes Neurais Artificiais (RNA's) visa solucionar problemas de reconhecimento de padrões que geralmente são baseados em um conjunto de informações previamente conhecidas. Geralmente os conjuntos de dados são divididos em conjunto de treinamento e conjunto de teste. Atualmente, pesquisadores em RNA's estão buscando uma compreensão das capacidades da natureza humana, as quais possibilitam que as pessoas construam soluções para problemas que não sejam resolvidos através de métodos tradicionais.

As redes neurais artificiais visam na sua maioria solucionar problemas de inteligência artificial, modelando sistemas através de circuitos (conexões) que possam simular o sistema nervoso humano, abrangendo a capacidade que o mesmo possui de aprender e agir perante as mais adversas situações apresentadas, bem como adquirir conhecimento através da experiência e da observação.

Segundo o pesquisador da Universidade de Helsinki (TAFNER et al, 1996), Teuvo Kohonen, uma rede neural artificial tem a seguinte definição: "uma rede massivamente paralela de elementos interconectados e suas organizações hierárquicas que estão preparadas para iterar com objetos do mundo real do mesmo modo que um sistema nervoso biológico faz".

A complexidade das estruturas elementares das Redes Neurais Biológicas é muito maior do que a dos modelos matemáticos usados nas Redes Neurais Artificiais, demonstrando as dificuldades encontradas para se tentar imitar o funcionamento do sistema nervoso humano. O sistema nervoso é formado por bilhões de células nervosas, enquanto que uma rede neural artificial possui de dezenas a no máximo milhares de unidades de processamento (neurônios).

Uma rede neural artificial pode ser vista como um conjunto de várias unidades interconectadas (similar à estrutura do cérebro), denominadas de neurônios artificiais, cada qual contendo uma pequena porção local de memória. Estes conceitos foram baseados e fundamentados nos estudos realizados nas células nervosas naturais. Portanto, busca-se aproximar ao máximo o funcionamento das redes neurais artificiais

das redes neurais biológicas, na tentativa de buscar a desenvoltura com que o cérebro humano desempenha suas funções.

Alguns modelos de redes neurais artificiais possuem muitos neurônios conectados numa estrutura de pesos de conexão e com facilidade de adaptação, proporcionando uma estrutura paralela. A estrutura paralela é desejável pois se algum(s) neurônio(s) falhar (em), os efeitos na rede como um todo não será significante para o desempenho do sistema se outro caminho de conexão entre os neurônios puder burlar a falha, surgindo então a tolerância à falhas.

A princípio, as redes neurais podem calcular qualquer função computável que é realizada em um computador digital, ou seja, possuem a capacidade de modelar relações lineares e não lineares. Principais características das RNA's (BARONE, 1999):

- capacidade de "aprender" através de exemplos e de generalizar este aprendizado de forma a reconhecer elementos similares, que não foram apresentados no conjunto de exemplos (treinamento);
- bom desempenho em tarefas pouco ou mal definidas, onde falta o conhecimento explícito de como resolvê-las, o aprendizado se dá através de exemplos;
- robustez à presença de informações falsas ou ausentes, escolha dos elementos no próprio conjunto de treinamento (integridade do conjunto de treinamento);
- no contexto de classificação de padrões, uma rede neural pode fornecer informações sobre quais padrões selecionar em função do grau de confiança apresentado (confiabilidade do conjunto de treinamento);
- tolerância à falhas.

4.1 Aplicações

Um dos principais objetivos da pesquisa sobre redes neurais artificiais na computação é desenvolver modelos matemáticos das estruturas neurais, não necessariamente baseadas na biologia, que podem efetuar diversas funções. Na maior parte dos casos, os modelos neurais são compostos por conjuntos de elementos não

lineares que operam em paralelo e que são classificados de acordo com modelos/padrões relacionados à biologia. Quando um método é criado visando utilizar aspectos de redes neurais artificiais, começam com o desenvolvimento de um neurônio artificial ou computacional baseado no entendimento de estruturas neurais biológicas, seguidas do aprendizado de mecanismos voltados para um determinado conjunto de aplicações e o treinamento do suposto sistema. Segue-se mais detalhadamente as seguintes fases:

- estudo do problema;
- desenvolvimento de modelos neurais motivados por neurônios biológicos;
- modelos de estruturas e conexões sinápticas;
- escolha de um algoritmo de aprendizado (um método de ajuste de pesos ou forças de conexões internodais);
- construção de um conjunto de treinamento;
- o treinamento propriamente dito;
- fase de testes;
- utilização da rede.

As diferenças entre as aplicações, os algoritmos de aprendizagem e as estruturas de interconexões entre os neurônios levam os pesquisadores a desenvolver diferentes modelos (arquiteturas) de redes neurais. Do ponto de vista estrutural, a arquitetura de redes neurais pode ser classificada como estática, dinâmica ou fuzzy, podendo ter uma ou múltiplas camadas. Além disso, diferenças computacionais surgem devido a forma como são feitas as conexões entres os neurônios. Estas conexões podem ser *feed forward*, *backward*, lateralmente conectadas, topologicamente ordenadas ou híbridas. As aplicações de redes neurais podem ser classificadas em diversas classes como:

- reconhecimento e classificação de padrões;
- processamento de imagem;
- visão computacional;
- identificação e controle de sistemas;

- processamento de sinais;
- robótica;
- filtros contra ruídos eletrônicos;
- análise do mercado financeiro;
- controle de processos.

Cabe ressaltar que em uma determinada aplicação de um sistema, que faz o uso das redes neurais artificiais, não precisa necessariamente ser classificada em apenas uma das citadas acima.

4.2 O Neurônio Artificial

O primeiro modelo matemático para uma rede neural, proposto por McCulloch e Pitts, era simples diante das informações disponíveis naquela época sobre o funcionamento elétrico de uma célula nervosa (Figura 4.1). Era um dispositivo binário, sendo que a saída do neurônio poderia ser pulso ou não pulso (ativo ou não), e as várias entradas tinham um ganho arbitrário, podendo ser excitatórias ou inibitórias. Para se determinar à saída do neurônio, calculava-se a soma ponderada das entradas com os respectivos ganhos como fatores de ponderação, excitatórios ou inibitórios. Se o resultado atingisse um certo limiar, a saída do neurônio era pulso (ativo), caso contrário, não pulso (não ativo). Assim como o neurônio biológico, o neurônio artificial possui um ou mais sinais de entrada e apenas um sinal de saída. As informações podem ser recebidas através de sensores ou de outros neurônios artificiais que fazem parte da Rede Neural Artificial (RNA). Estes sinais são processados e enviados para a saída. Os sinais de entrada (estímulos) devem chegar até o neurônio simultaneamente, isto é, todas as informações devem chegar ao núcleo do neurônio artificial ao mesmo tempo.

O processamento paralelo em computadores seqüenciais (por exemplo, os microcomputadores atuais) pode ser paradoxal, mas não o é, ocorre de fato. A simulação de um ambiente paralelo é possível, e é desta forma que ocorre esse tipo de processamento para as redes neurais. O modelo matemático simula o paralelismo da rede neural através de um algoritmo (TAFNER, 1996).

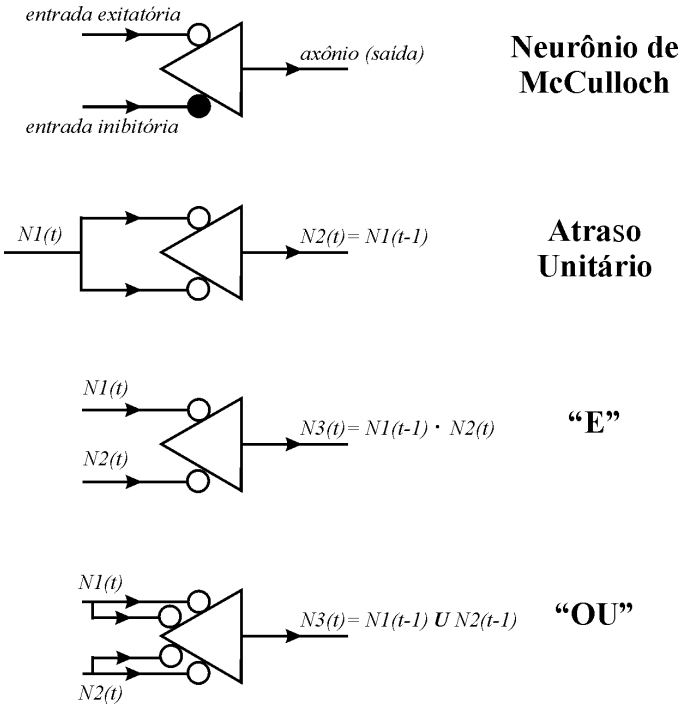


Figura 4.1 - O neurônio de McCulloch e implementações de algumas funções booleanas (kovács, 1996)

Um dos atributos de grande importância do neurônio artificial é o peso. Os pesos, também conhecidos por pesos sinápticos, são representados pela letra **w** (*weight*) e representam o grau de importância que determinada entrada possui em relação àquele determinado neurônio.

O valor do peso é alterado em função da intensidade do sinal de entrada, e dessa forma, o peso muda o seu valor representativo para a rede (processo de aprendizagem). Deduz-se que, quanto mais estimulada for uma entrada, mais estimulado será o peso correspondente, e quanto mais for estimulado um peso, mais significante e influente o mesmo será para o resultado do sinal de saída do respectivo neurônio.

Matematicamente, os pesos são vistos como um vetor de valores $[w_1, w_2, ..., w_n]$ para um neurônio, ou uma matriz de pesos, coleção de vetores, para um conjunto de neurônios.

O sinal de excitação do neurônio é resultante do somatório do produto dos sinais de entrada, representados por um vetor $[x_1, x_2, ..., x_n]$, pelo vetor de pesos do neurônio ($\sum_{i=0}^{n} x_i w_i$ - o valor correspondente a $x_0 w_0$ será explicado adiante e corresponde ao viés,

representando um estímulo inicial a rede). Após esta operação, os sinais de entrada passam a ser chamados de entradas ponderadas.

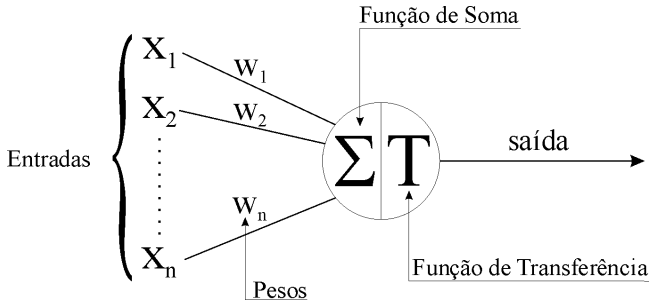


Figura 4.2 - O Neurônio artificial (Tafner et al, 1996)

A próxima tarefa a ser tomada pelo neurônio, é a de verificar se o valor resultante da soma entre o produto dos sinais de entrada pelos respectivos pesos atingiu ou não um valor predeterminado, chamado de limiar. Se o valor calculado atingiu o limiar, o mesmo é repassado adiante através da saída. Caso contrário, se o valor não atingiu o limiar, o sinal não será transferido. Esse processo de verificação é chamado de função de transferência, que também é conhecido como limiar lógico.

A resposta final da rede ou das camadas subjacentes está diretamente ligada com o resultado obtido pela função de transferência. Por isso, deve-se dar a devida atenção a este processo. A lógica neural expõe, que a intensidade dos sinais de entrada, dispara, ou não, o sinal do neurônio, fazendo com que este estimule o neurônio seguinte (TAFNER et al, 1996).

Além da função de transferência, há a função de ativação, a qual antecede a mesma e tem como função, suceder um nível de ativação dentro do próprio neurônio, ou seja, o neurônio, através desta função, decidirá o que fazer com o resultado da soma ponderada das entradas (ativar ou não). Essa decisão tem efeito somente ao respectivo neurônio artificial.

Em alguns modelos simples de redes neurais artificiais, a função de ativação pode ser a própria função de soma das entradas ponderadas do neurônio. Já em modelos mais elaborados, a função de ativação pode possuir um processamento atribuído, o qual pode ser, por exemplo, o uso de um valor prévio de saída como uma entrada para o próprio neurônio, servindo de auto-excitação para o mesmo (TAFNER et al, 1996).

O valor de saída do neurônio será produzido após a chamada da função de ativação, seguido pela função de transferência.

Em alguns casos, o neurônio artificial pode não ter efeito no neurônio seguinte se o valor de ativação não ultrapassar um certo valor mínimo. Este fator é resultante das características sigma ou ríspidas que a função de transferência tem como propriedade. Devido a esse fator, há vários tipos de funções de transferência (Figura 4.3).

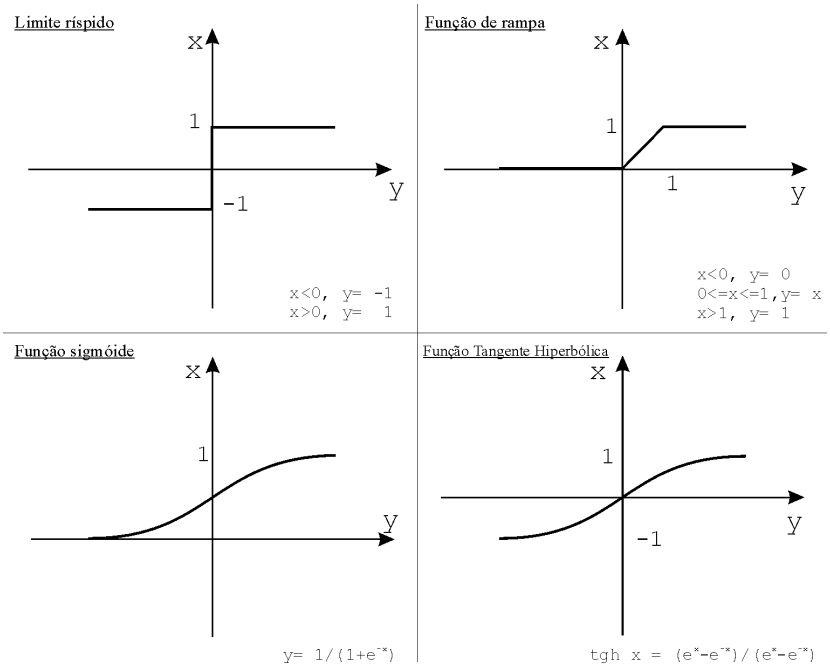


Figura 4.3 - Funções de transferência (Kovács, 1996)

Assim como nas redes neurais biológicas, o conjunto de vários neurônios artificiais interconectados, formam as redes neurais artificiais.

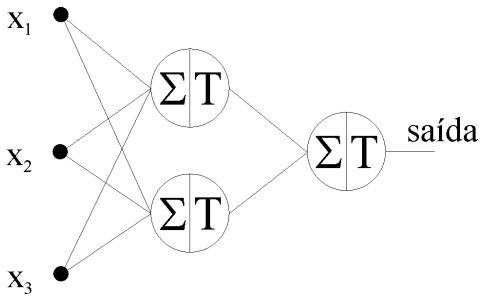


Figura 4.4 - Rede neural artificial

4.3 Arquiteturas

Um outro detalhe importante a ser considerado é a maneira como os neurônios artificiais podem ser agrupados. Este agrupamento se sucede no cérebro humano de maneira que as informações possam ser processadas de forma dinâmica ou interativa. Biologicamente, as redes neurais são organizadas e construídas de forma tridimensional por componentes microscópicos.

Uma rede neural pode ter uma ou várias camadas. As redes que possuem uma única camada são as redes que possuem um nó entre uma entrada e uma saída da rede (Figura 4.5). Esse tipo de rede é indicado para a solução de problemas linearmente separáveis. Já as redes multicamadas possuem mais de uma camada entre as já existentes camadas de entrada e saída (Figura 4.6).

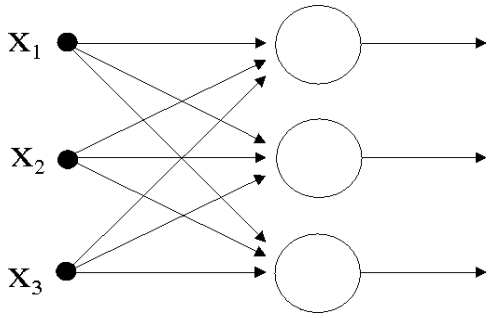


Figura 4.5 - RNA de uma única camada

As redes neurais artificiais multicamadas possuem as chamadas camadas escondidas (hidden), que também são chamadas de intermediárias ou ocultas. Esse número de camadas pode ser indeterminado, e estão situadas entre a camada de entrada e a camada de saída da rede neural (CARVALHO, 1998).

As camadas ocultas são constituídas por neurônios artificiais, da mesma forma com que as camadas externas (entrada e saída) são compostas, e tendo como característica diferenciada o não contato com o mundo externo (Figura 4.6). Os sinais são passados para os outros neurônios obedecendo às funções de transferência que cada neurônio possui (NASCIMENTO, 1994).

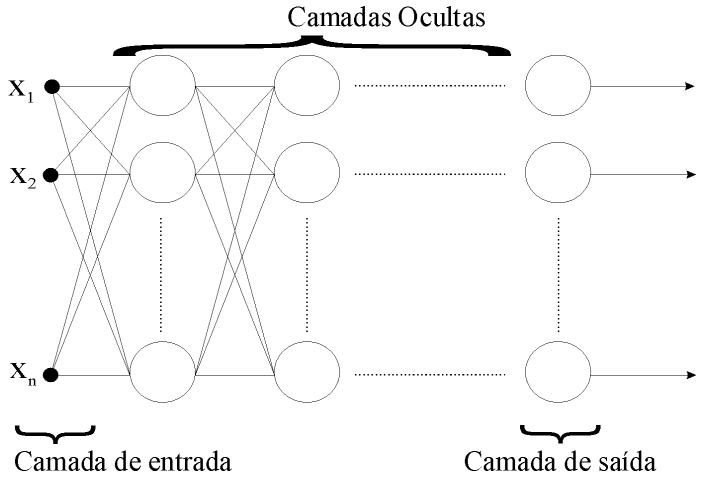


Figura 4.6 - RNA multicamada

Os nodos que compõe a rede neural artificial podem ter conexões do tipo:

- *feedforward* ou acíclicas (Figura 4.7) – a saída de um neurônio na *i*-ésima camada da rede não pode ser usada como entrada de nodos em camadas de índice menor ou igual a *i* (CARVALHO, 1998). Uma aplicação típica para as redes neurais artificiais *feedfoward* é de desenvolver modelos não-lineares que também são usados para o reconhecimento e classificação de padrões. Uma rede *feedforward* pode ser vista como uma ferramenta que realiza a análise de regressão não linear (NASCIMENTO, 1994).

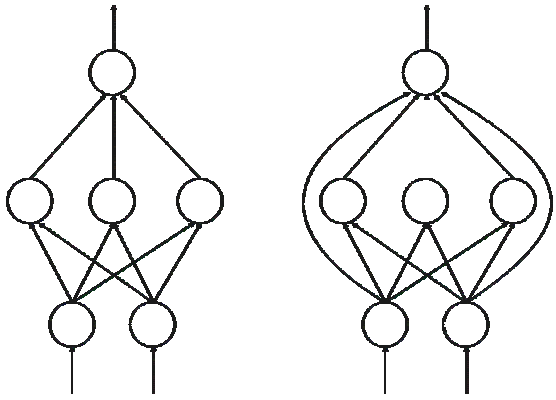


Figura 4.7 - RNA feedforward ou acíclica

- conexões feedback ou cíclica (Figura 4.8) – a saída de algum neurônio na *i*-ésima camada da rede é usada como entrada de nodos em camadas de índice menor ou igual a *i*. Se todas as ligações entre os neurônios forem cíclicas, a

rede é chamada auto associativa; estas redes associam um padrão de entrada com ele mesmo, e são particularmente úteis para a recuperação ou regeneração de um padrão de entrada (CARVALHO, 1998).

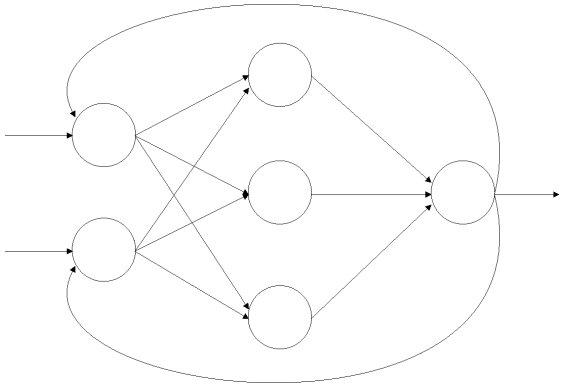


Figura 4.8 - RNA feedback ou cíclica

4.4 Aprendizado

Para o aprendizado das redes neurais, foram propostos diversos métodos de treinamento, sendo estes subdivididos em dois paradigmas principais: o aprendizado supervisionado e o não supervisionado. Para estes modelos existem vantagens e desvantagens que serão expostas a seguir. As RNA's possuem a capacidade de aprender por exemplos, determinando a magnitude das conexões entre os neurônios pertencentes à rede. Logo, um conjunto de procedimentos definidos para ajustar os parâmetros de uma RNA, a fim que a mesma possa aprender uma determinada função, é chamado de algoritmo de aprendizado. A designação de uma RNA, na resolução de um determinado problema, passa inicialmente por um processo de aprendizagem, onde a rede procura extrair informações relevantes de padrões de informação apresentados a ela, modelando uma representação própria.

4.4.1 Supervisionado

A vasta maioria das redes neurais artificiais tem utilizado o treinamento supervisionado. Deste modo, a saída atual da rede neural é comparada com a saída desejada. Os pesos terão os seus valores iniciais inicializados aleatoriamente, e serão ajustados, através do algoritmo de aprendizagem, pela rede na próxima iteração ou ciclo.

O ajuste sináptico é dependente do valor esperado e do sinal atual de saída. Desta maneira, o método de aprendizado tenta minimizar o fluxo corrente de erros de todos os elementos em processamento. Esta redução global de erros trabalha modificando continuamente os pesos até que a rede alcance uma certa precisão.

Com o aprendizado supervisionado, as redes neurais artificiais devem ser treinadas antes de serem usadas. O treinamento consiste da apresentação dos sinais de entrada e saída à rede. Estes dados são freqüentemente referenciados ao conjunto de treinamento. A fase de treinamento pode consumir uma grande fatia de tempo. Em alguns sistemas protótipos, com um inadequado poder de processamento, o aprendizado pode levar semanas. O treinamento é considerado completo quando a rede neural alcança um certo nível de performance. Este nível significa que a rede alcançou uma precisão estatística conforme as produções de saída necessárias para uma dada seqüência de entradas. Quando não há mais a necessidade de aprendizado, os pesos são praticamente "congelados" para a aplicação. Alguns tipos de redes neurais permitem um treinamento contínuo, com uma taxa muito baixa de aprendizado, enquanto a mesma está em operação. Este processo ajuda a rede a adaptar-se gradualmente as condições de mudança.

O conjunto de treinamento precisa ser suficientemente grande para conter as informações necessárias para que a rede aprenda os moldes e as relações importantes. Se a rede é treinada somente com um exemplo em um determinado tempo, todos os pesos serão ajustados meticulosamente para este fato, os quais poderiam sofrer alterações drásticas no aprendizado de um próximo fato. Conforme um resultado, o sistema precisa aprender com todos os fatos em conjunto, provendo posteriormente o melhor ajuste dos pesos para todo o conjunto de fatos.

A maneira com que os sinais de entrada são representados, ou codificados, determina o maior componente constituinte para o sucesso de instrução da rede. Normalmente, as redes neurais artificiais somente manipulam, ou trabalham, com dados numéricos como entrada. Por este motivo, os dados do mundo exterior, devem ser tratados e convertidos para que se possa alimentar a rede. Esta captura de estímulos do mundo real pode ser feita através de vários tipos de dispositivos, tais como: câmeras de vídeo, diversos tipos de sensores, microfones, etc.

Várias técnicas de condicionamento já estão disponíveis para serem aplicadas a implementações de redes neurais artificiais, viabilizando e principalmente facilitando para que o desenvolvedor da rede encontre o melhor formato para os dados, e uma arquitetura adequada para a rede objetivando uma determinada aplicação.

Após o treinamento supervisionado, é importante analisar o que a rede pode realizar com os dados que ainda não foram apresentados à mesma. Se o resultado de saída do sistema não for razoável para este novo conjunto de dados (chamado conjunto de teste), presume-se que o treinamento da rede ainda não foi suficiente.

Esta avaliação é crítica para assegurar que a rede simplesmente não memorizou um dado conjunto de dados, mas sim aprendeu os modelos/padrões gerais envolvidos na aplicação (generalização). É importante ressaltar que às vezes o problema da generalização é devido à má qualidade dos dados usados para o treinamento e não um problema da rede.

4.4.2 Não supervisionado

O aprendizado não supervisionado implica no aprendizado da rede sem a necessidade de um conjunto de treinamento.

Estas redes não suportam influências externas para ajustar os seus pesos sinápticos, pois há um monitoramento de desempenho interno da mesma, analisando as regularidades e/ou tendências dos sinais de entrada, e conseqüentemente adaptando-se automaticamente as necessidades da rede.

Possuindo características de autonivelação, sem um suposto auxílio para determinar se o aprendizado converge ou não para o caminho certo, a rede possui mecanismos, mais precisamente, informações, de como se organizar. Esta propriedade e percepção da rede devem-se a topologia e as regras de aprendizado adotado pela rede neural artificial.

Uma rede com o algoritmo de aprendizado não supervisionado deve ter enfatizado a cooperação entre as camadas de unidades de processamento. A competição entre estas unidades é a base de aprendizado da rede. Normalmente, quando a competição pelo aprendizado ocorre de fato, somente os pesos pertencentes à unidade de processamento vencedora são ajustados.

4.5 Redes Perceptron

As redes neurais artificiais com função de ativação foram inicialmente estudadas por Rosenblatt em meados de 1958, as quais foram chamadas por ele de Perceptrons. O entusiasmo de Rosenblatt levou-o a construir suas redes em hardware, inclusive usando um algoritmo de aprendizado.

Estas redes foram aplicadas para a classificação de problemas que geralmente possuíam como fonte de alimentação imagens binárias de caracteres ou simplesmente moldes de informações (BISHOP, 1995). O perceptron em sua origem era uma simulação computacional da retina, a qual demonstrou como o sistema nervoso visual reconhece padrões (TAFNER, 1996).

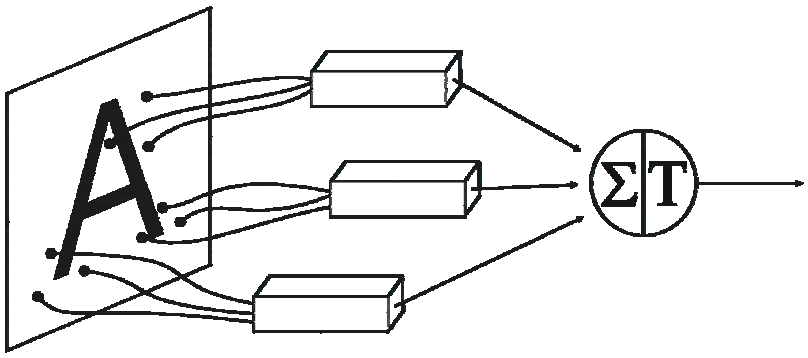


Figura 4.9 - O perceptron elementar de Rosemblatt (Bishop, 1995)

Enquanto Rosenblatt estava desenvolvendo o perceptron, Widrow e seus colegas, estavam trabalhando em uma linha de pesquisa similar a de Rosemblatt; mais conhecida como ADALINE. Como já exposto, o termo **ADA**ptive **LIN**ear **E**lement refere-se a uma única unidade de processamento com um limiar não linear.

Como as redes neurais artificiais de uma única camada possuem uma certa limitação, Rosenblatt resolveu então usar um número fixo de neurônios para transformar e tratar os dados provindos do mundo exterior. Estas unidades de processamento podem ser chamadas de função base de um discriminador limiar (BISHOP, 1995).

Rosenblatt propunha resolver problemas como a implementação das funções booleanas **E** e **OU** de duas variáveis, sendo que a escolha dos ganhos para este caso parecia ser trivial. Entretanto, para a implementação de uma função discriminatória arbitrária, a escolha não é tão simples e muito menos trivial, e dependendo do número

de variáveis envolvidas, sem a existência de algum método, beira o impossível (KOVÁCS, 1996).

Inspirado também pelas idéias de McCulloch, Rosenblatt compôs a rede perceptron por uma camada de entrada, onde cada elemento pertencente à camada de entrada fazia a distribuição do sinal que ele recebia para todas as unidades de processamento. Os neurônios eram essencialmente compostos por unidades de processamento (sigma) e de funções de transferência, sendo que estas, eram responsáveis pela soma ponderada dos sinais oriundos das conexões com os dados de entrada. Foi adicionada a camada de entrada um elemento especial chamado viés, o qual possui um sinal de valor sempre um. A conexão entre o viés e a unidade sigma tem peso w_0, que por sua vez é ajustado da mesma maneira com que os demais pesos o são.

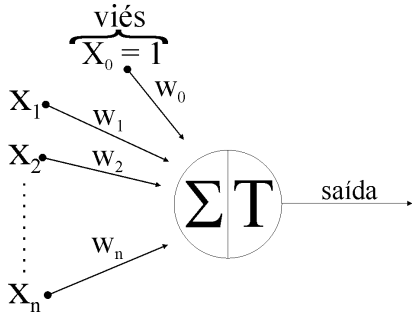


Figura 4.10 - A unidade de processamento do perceptron

O nível de ativação de uma rede perceptron é dado pela soma ponderada dos pesos sinápticos com os valores de entrada, $\sum x_i \cdot w_i$.

Estas redes usam uma função de transferência do tipo *hard-limiter* (limite ríspido), onde a ativação do limiar resulta num valor de saída 1, ou, –1 caso contrário. Dados os valores de entrada x_i, os pesos w_i, e um limiar t, o perceptron computa os valores de saída da seguinte maneira :

$$\begin{cases} \mathbf{1} & \textbf{se} \quad \sum x_i w_i \geq t \\ -\mathbf{1} & \textbf{se} \quad \sum x_i w_i < t \end{cases} \qquad 4.1$$

As redes perceptron usam como configuração, o treinamento supervisionado. O perceptron altera os seus pesos, visando reduzir o erro.

4.5.1 Limitações: O problema do OU-EXCLUSIVO

Um dos problemas que o perceptron não seria capaz de resolver era o do ou-exclusivo. Foi baseado neste exemplo que Minsky e Papert mostraram à comunidade científica que o modelo de Rosenblatt não era tão eficiente e promissor.

Tabela 4.1 - Tabela verdade do ou-exclusivo

x_1	x_2	saída
1	1	0
1	0	1
0	1	1
0	0	0

Considerando uma rede perceptron com duas entradas [x_1, x_2], dois pesos [w_1, w_2], e um limiar t, a rede, para aprender com estes fatos, deveria encontrar os pesos designados para satisfazer a tabela verdade e as seguintes premissas (LUGER, 1998):

- para a linha 1 da tabela verdade: $w_1 \cdot 1 + w_2 \cdot 1 < t$
- para a linha 2 da tabela verdade: $w_1 \cdot 1 + 0 > t$
- para a linha 3 da tabela verdade: $0 + w_2 \cdot 1 > t$
- para a linha 4 da tabela verdade: $0 + 0 < t$

As premissas apresentadas, baseadas nos pesos [w_1, w_2] e no limiar t, não possuem solução. Logo, o perceptron de uma única camada é incapaz de resolver este tipo de problema.

O motivo pelo qual torna o problema do ou-exclusivo impossível para as redes do tipo perceptron é que as duas classes que precisam ser distinguidas não são linearmente separáveis.

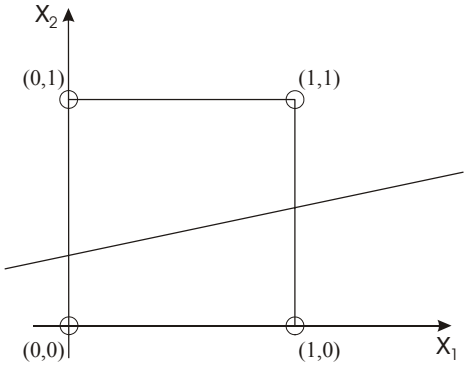


Figura 4.11 - Plano que representa as combinações possíveis do XOR

Percebe-se que é impossível plotar uma linha reta que separe em duas dimensões os pontos {(0,0) , (1,1)} de {(0,1) , (1,0)}.

Cada parâmetro dos dados de entrada corresponde a uma dimensão, com cada valor de entrada definindo um ponto no espaço (LUGER, 1998).

4.6 Redes Multilayer Perceptron

Os problemas não linearmente separáveis podem ser resolvidos através das redes com uma ou mais camadas intermediárias. A alteração da arquitetura da rede, como a inserção de camadas ocultas e/ou o número de neurônios, a princípio, não parece ser problema, pois um dos principais agravantes passa a ser o algoritmo de treinamento para as redes multicamadas. Fator este que, devido à inexistência ou desconhecimento, causou uma atenuação nas pesquisas em redes neurais artificiais em meados da década de 70. Uma das alternativas adotadas é dividir a rede em um conjunto de sub redes, sendo uma sub rede para cada camada, com um treinamento independente. Este método de subdivisão, muitas vezes, ou não é possível ou é muito complicado. Outra possibilidade seria realizar um treinamento completo, isto é, de uma só vez. O problema encontrado para este segundo método está em como realizar o treinamento dos nodos que pertencem à camada intermediária, visto que é extremamente complicado determinar que tipo de resposta desejada estes teriam, ou seja, como determinar o erro. A aplicabilidade deste método está restrita a definição do erro nos nodos pertencentes às camadas intermediárias da rede. Se for utilizada uma função do tipo limiar, a avaliação do erro será complexa, visto que, os nodos das camadas intermediárias e de saída não terão como saber a margem de erro ou a diferença entre as respostas de seus nodos com relação às respostas desejadas. Uma das soluções para o problema apresentado seria a

utilização de uma função de ativação não linear, a qual resolve o mesmo em parte, visto que a utilização deste tipo de função em redes multicamada resultaria na equivalência de uma rede de uma única camada (CARVALHO, 1998).

Adotou-se então treinar as redes com mais de uma camada através de métodos baseados no gradiente descendente. Métodos baseados no gradiente descendente precisam ter a função de ativação contínua, diferençável e não decrescente. A função adotada precisa informar os erros que a rede cometeu para as camadas anteriores, com uma boa precisão. Logo a função que mais se adapta a estas características é a função do tipo sigmóide (CARVALHO, 1998).

O processamento atribuído a cada neurônio pertencente à rede é resultante da combinação do processamento realizado pelos neurônios da camada anterior, que por sua vez estão atribuídos a este nodo da próxima camada. A medida com que cada camada intermediária da rede se aproxima da camada de saída há uma delimitação do espaço de decisão dos dados que está recebendo. Para uma rede com duas camadas intermediárias, teríamos a primeira camada oculta, delimitando o espaço de padrões de treinamento através das "retas traçadas" pelos neurônios. A segunda camada forma regiões convexas, onde o número de lados que compõe tal região é determinado pela quantidade de unidades conectadas a este neurônio, que por sua vez combina as retas que surgiram da camada anterior. Cada neurônio da camada de saída forma regiões, provenientes das combinações das regiões convexas (CARVALHO, 1998). Conclui-se que cada neurônio que compõe uma rede Multilayer Perceptron contribui para a detecção de características dos dados apresentados.

A determinação do número de camadas a ser utilizada influi de forma crucial no aprendizado da rede. O uso de um grande número de camadas intermediárias não é recomendado, visto que o erro ocorrido em uma camada é propagado a camadas anteriores da rede. A determinação do número de neurônios que pertence a camadas intermediárias é definida de forma empírica, e normalmente depende da distribuição dos padrões de treinamento e validação da rede. Um uso excessivo de neurônios levará a rede a decorar o conjunto de treinamento, ao invés de extrair as características gerais (generalizar). Ao processo de memorização do conjunto de treinamento, dá-se o nome de *overfitting*. Um número razoavelmente pequeno de neurônios levará a rede a

aumentar o tempo de treinamento, dificultando a determinação da representação ótima do problema proposto. Neste caso, alguns neurônios poderão ficar sobrecarregados, pois estes precisam lidar com um número elevado de restrições a serem analisadas.

4.7 Ensaio computacional: classificação de padrões

Nesta seção realiza-se um experimento computacional para ilustrar o comportamento da aprendizagem de uma rede perceptron de múltiplas camadas usada como classificador de dados. O experimento computacional envolve um problema de classificação, sendo este retirado do repositório de bases de dados para aprendizado de máquina da UCI – Universidade da Califórnia de Irvine (BLAKE, 1998). As bases de dados contidas na UCI são utilizadas pela comunidade de científica de aprendizado de máquina para realizar análises empíricas de algoritmos de AM.

Para a concepção e modelagem da Rede Neural, utilizou-se o simulador para redes neurais da Universidade de Stuttgart. O SNNS (*Stuttgart Neural Network Simulator*) é um software simulador para redes neurais para estações de trabalho Unix e está disponível em: <http://www-ra.informatik.uni-tuebingen.de/SNNS/>. O SNNS foi desenvolvido no Instituto de Sistemas Paralelos e Distribuídos de alto desempenho (IPVR - *Institute for Parallel and Distributed High Performance Systems*) na Universidade de Stuttgart.

A objetivo do projeto SNNS é criar um ambiente eficiente e flexível de simulação para pesquisa em aplicações de redes neurais. O simulador SNNS consiste em dois componentes principais:

1) núcleo(*kernel*) do simulador escrito na linguagem C

2) interface gráfica para o X (servidor de interface gráfica)

O núcleo do simulador opera nas estruturas de dados internos da rede neural, e executa todas as operações de aprendizado e teste. Este também pode ser usado em outras partes de um programa em C – embutido em aplicações customizadas. O simulador também suporta topologias arbitrárias de rede. O SNNS também pode ser estendido pelo usuário; este pode definir funções de ativação, funções de saída e ainda algoritmos de aprendizado, os quais deverão ser escritos em C e ligados ao núcleo do simulador.

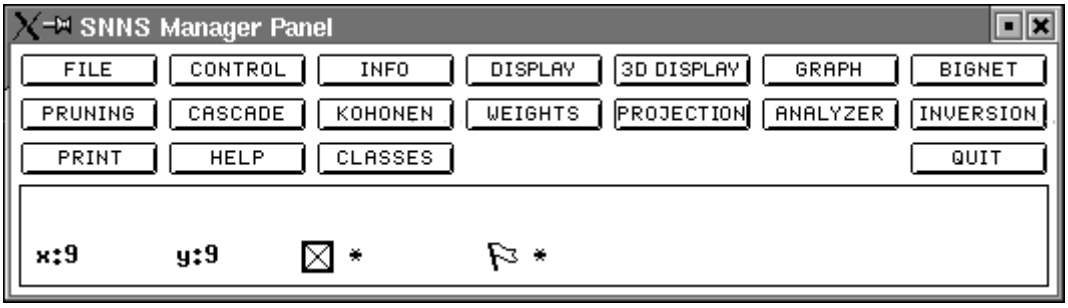


Figura 4.12 – Interface de gerenciamento do SNNS

A interface gráfica de usuário, chamada de XGUI, é desenvolvida numa camada acima do núcleo, e apresenta a rede neural em 2D e 3D. Ainda é possível através desta controlar o núcleo do simulador durante uma simulação. Além da interface gráfica o SNNS possui um editor de rede integrado, o qual pode ser usado diretamente para criar, manipular e visualizar a rede de neural de vários modos.

A primeira base de dados a ser utilizada é a íris, popularizada por R. A. Fisher em 1936, ao ilustrar os princípios da análise discriminante (BLAKE, 1998). A base consiste de cento e cinqüenta amostras de flores de espécies setosa, versicolor e virgínica. Para cada espécie há cinqüenta observações. Cada instância do conjunto de dados é caracterizada por quatro atributos: o comprimento da sépala, a largura da sépala, o comprimento da pétala e a largura da pétala. Uma classe é linearmente separável das outras duas; e as outras duas não o são uma da outra. A Figura 4.13 mostra o grafo arquitetural de um perceptron de múltiplas camadas, totalmente conectada, para o problema de classificação da íris utilizando o simulador SNNS. Este grafo apresenta uma camada de entrada com quatro neurônios – um neurônio para cada atributo, uma camada oculta com cinco neurônios e uma camada de saída com três neurônios. Cada neurônio na camada de saída corresponde a uma das espécies do problema. O fluxo do sinal através da rede progride para frente, da esquerda para a direita e de camada em camada.

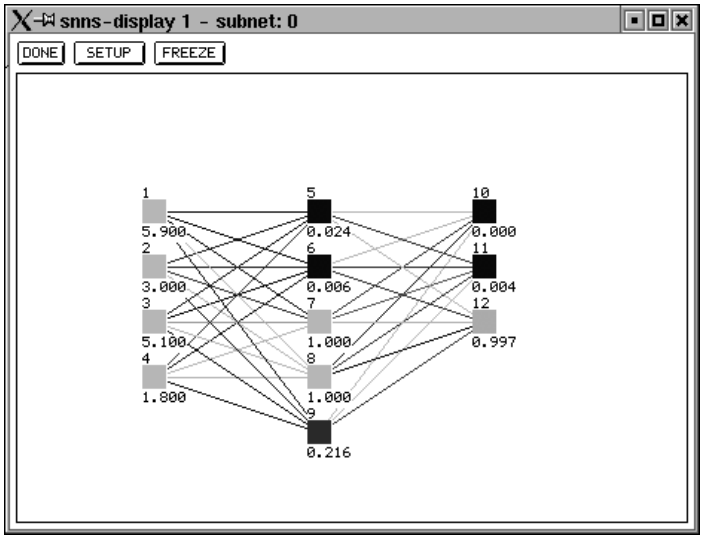


Figura 4.13 – Grafo da arquitetura da rede neural para a base íris

Na Figura 4.14 é apresentada as curvas de aprendizagem experimentais.

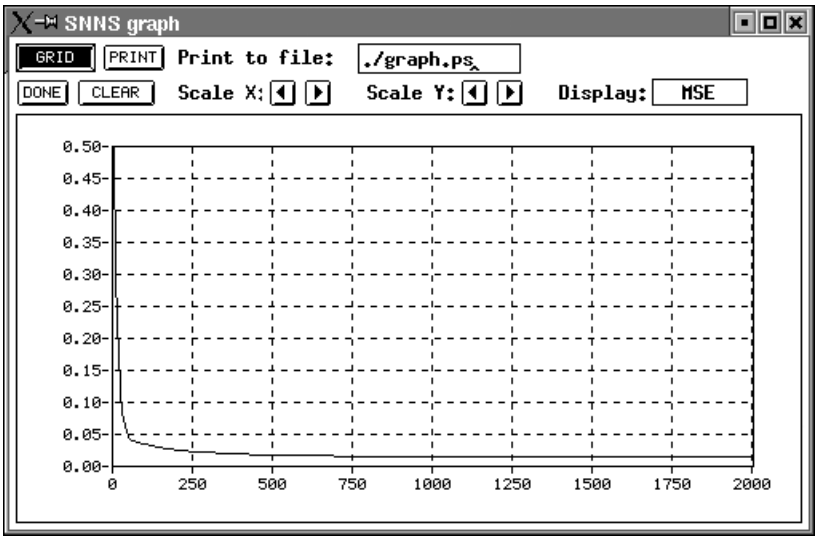


Figura 4.14 – Curva de aprendizagem (Íris): erro médio quadrado – número de épocas

O algoritmo de aprendizagem utilizado para o treinamento da rede é o *Resilient Backpropagation* (Riedmiller, 1994). O modelo de cada neurônio da rede inclui uma função de ativação logística, conforme apresentado na Figura 4.3. Após duas mil (2000) épocas, a rede apresentou uma percentual de acerto de aproximadamente 97%. O conjunto de teste é o mesmo conjunto de treinamento; ambos com cento e cinqüenta (150) exemplares. Destes 150, três (71, 84 e 134) não foram corretamente classificados. A Figura 4.15 apresenta os exemplares da base íris na forma gráfica, combinando os atributos dois a dois {[1 2] [1 3] [1 4] [2 3] [2 4] [3 4]}. Percebe-se que as espécies

versicolor (×) e virgínica (•) não são mutuamente exclusivas. Já a espécie (classe) setosa (∗) é linearmente separável das outras.

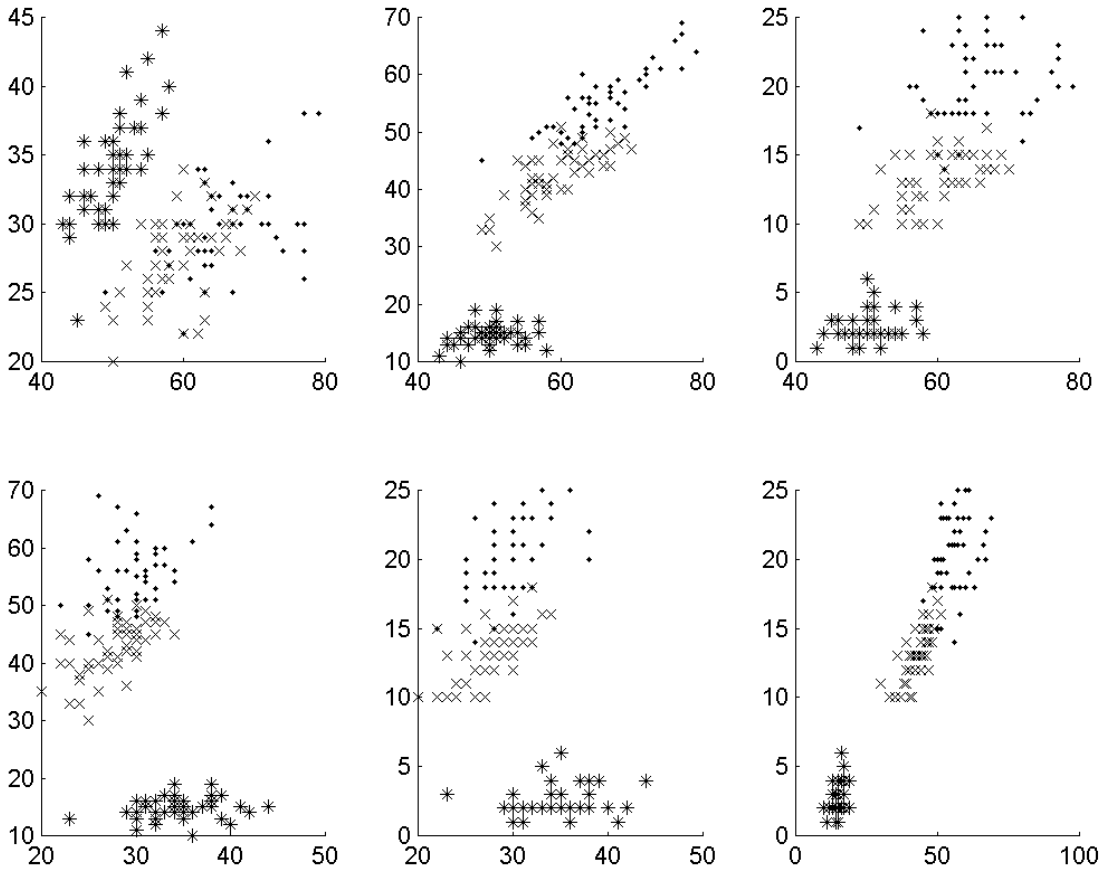


Figura 4.15 – Distribuição gráfica dos exemplares da base íris

5 Aprendizagem Automática

5.1 Árvores de Decisão

As árvores de decisão são atualmente uma das técnicas mais utilizadas para o particionamento de exemplos em conjuntos de regras de decisão. A Figura 5.1 é um exemplo de uma árvore de decisão binária. Uma árvore de decisão consiste de nodos e ramos. Cada nodo representa um simples teste ou decisão. No caso de uma árvore binária, a decisão pode ser verdadeira ou falsa. O nodo inicial é comumente referido como nodo raiz. Dependendo do resultado do teste, a árvore poderá se ramificar à esquerda ou à direita em direção a outro nodo. Por fim, o nodo terminal, conhecido também como nodo folha, é alcançado, e uma decisão é realizada a uma classe designada.

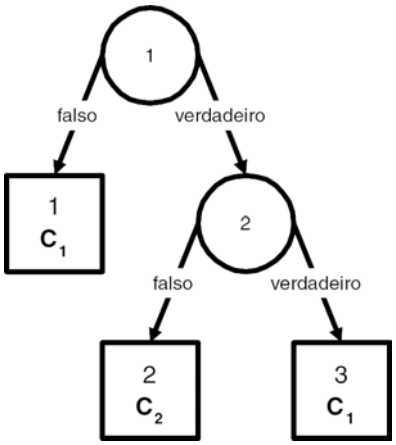


Figura 5.1 - Exemplo de uma árvore de decisão binária (Weiss, 1991)

Na Figura 5.1, o nodo 1 é o nodo raiz e o nodo 2 é o único nodo não terminal. Os nodos terminais são os nodos quadrados. Quando um novo caso é apresentado, o teste representado pelo nodo 1 é considerado primeiro.

Há um considerável número de características gerais de uma árvore de decisão binária. Somente dois ramos deixam cada nodo, e somente um ramo entra em qualquer nodo. Em qualquer árvore binária, há *n* nodos terminais e *n-1* nodos não terminais. Na construção de árvores de decisão binárias, é normalmente utilizada a convenção de realizar decisões verdadeiras em ramos da direita e decisões falsas no ramo da esquerda.

Árvores de decisão não binárias são largamente utilizadas. Nestes tipos de árvores, mais do que dois ramos deixam um nodo. Assim como as árvores binárias, somente um ramo entra num nodo. A Figura 5.2 ilustra um árvore de decisão não binária: o nodo 2 possui três ramos. Neste tipo de árvore, um teste realizado em um nodo resulta na divisão de dois ou mais conjuntos disjuntos que cobrem todas as possibilidades, isto é, todo novo caso deve pertencer a um dos subconjuntos disjuntos.

Na Figura 5.2, os valores do nodo 2 são divididos em três conjuntos disjuntos.

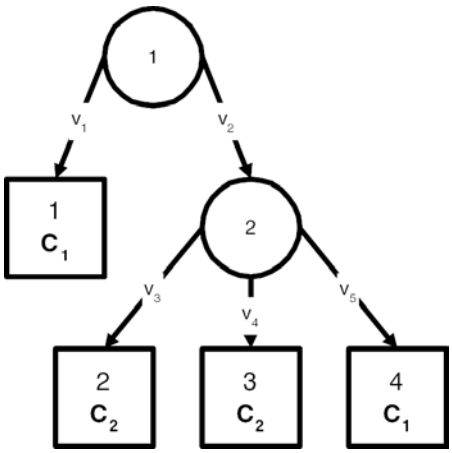


Figura 5.2 - Exemplo de uma árvore de decisão não binária (Weiss, 1991)

Para qualquer árvore, todos os caminhos conduzem a um nodo terminal, correspondendo a uma regra de decisão, a qual é uma conjunção de vários testes. Se há múltiplos caminhos para uma dada classe, então os caminhos representam disjunções. Todos os caminhos numa árvore de decisão são mutuamente exclusivos. Para cada novo caso, um e somente um caminho na árvore deverá ser satisfeito.

5.1.1 Processo geral de construção de árvores decisão

O processo de aprendizagem da estrutura de uma árvore de decisão ou de regras equivalentes de dados é conhecido com indução de árvores ou regras(WEISS, 1991). O tipo de estrutura apresentada na Figura 5.2 sugere um método simples de indução de árvores. Nesta estrutura, todos os nodos são divididos em conjuntos disjuntos de valores distintos, representados pelo nodo de teste.

Uma árvore pode ser induzida pela seleção de algum atributo ou teste, dividindo esse atributo em conjuntos disjuntos; e repetindo o processo para todos os nodos

subseqüentes. Os nodos tornam-se terminais quando todos os exemplares restantes pertencerem a uma única classe.

De forma alternativa, os nodos podem se tornar terminais quando o número de casos restantes no grupo for menor que um limiar mínimo estabelecido; tal como cinco casos, e o nodo é associado à classe que possuir a maior freqüência naquele nodo.

Na Figura 5.2, o nodo 1 foi escolhido como nodo raiz. Os valores de teste representados pelo nodo 1 possuem apenas dois valores possíveis, v_1 e v_2. Atributos binários são comuns, mas em qualquer evento, a divisão é encontrada pela análise dos dados e na determinação de valores distintos para o teste. Se todos os valores dos exemplares de v_1 são membros da classe C_1, esse nodo torna-se um caminho terminal.

Os casos envolvendo v_2 incluem casos de ambas as classes C_1 e C_2. Portanto, uma divisão adiante deverá ocorrer. Algum teste é selecionado para o nodo 2. Este teste pode assumir três valores distintos, v_3, v_4, e v_5. Um ramo para cada um destes valores é criado para o nodo 2. Se os casos de cada grupo (v_2 e v_3, v_2 e v_4, v_2 e v_5) pertencem a uma única classe, todos se tornam nodos terminais.

5.1.2 Técnicas para a seleção de atributos

Uma árvore pode ser induzida pela sucessiva seleção e subdivisão de atributos. Estes atributos podem ser escolhidos de forma randômica, e eventualmente uma árvore pode ser formada de nodos terminais a partir do momento em que cada nodo possui membros de apenas uma única classe. Assim, a taxa de erro aparente é minimizada e normalmente é zero.

A seleção randômica de atributos levará a árvores extensas. Em amostras reais, muitos atributos contem ruídos. Este tipo de processo de seleção randômica de atributos despende muito tempo na escolha de atributos irrelevantes.

O modo de seleção de um atributo é determinado pelo exame de cada atributo, avaliando-se a qualidade ou representatividade do mesmo; de forma a aumentar a performance de decisão da árvore.

Experiências com problemas reais mostram que diversas funções de avaliação tendem a uma boa escolha de atributos úteis, gerando inclusive árvores pequenas (WEISS, 1991).

O conceito subjacente de qualquer avaliação no processo de divisão é selecionar um atributo que produzirá a melhor árvore. Porém, a função de avaliação prediz somente um único nodo. Deste modo, a função de avaliação precisa ser uma heurística que procura realizar boas decisões com informações incompletas. A função de avaliação mais utilizada para a escolha do atributo trabalha pela redução do grau de incerteza no nodo corrente.

A função de avaliação mais popular é função de *entropia* (SHANNON, 1948, ICS, 2002) e a função *gini* (WEISS, 1991, ICS, 2002). O objetivo global é reduzir a impureza e casualidade de classes no nodo corrente e em futuros nodos.

A equação 5.1 apresenta a função de *entropia* e a equação 5.2 a função *gini*; onde p_j é a probabilidade da classe j.

$$-\sum_j p_j \log p_j \qquad 5.1$$

$$1-\sum_j p_j^2 \qquad 5.2$$

5.2 Exemplo de indução de árvores de decisão

Segundo DURKIN (1994), indução é o processo de raciocínio sobre um dado conjunto de fatos para princípios gerais ou regras. Toma-se como exemplo a seguinte linha de pensamento: se eu dissesse a alguém que eu gosto de futebol, vôlei e natação, esta pessoa provavelmente concluiria por indução que eu gosto de esportes. A indução busca padrões em informações disponíveis com o propósito de inferir conclusões racionais.

O aprendizado indutivo tem sido uma importante área de pesquisa em Inteligência Artificial e pode ser visto como uma busca de soluções no espaço de estados do problema.

Para ilustrar o aprendizado indutivo, será apresentado um problema que procura determinar qual presente comprar para uma determinada pessoa perante algumas características (DURKIN, 1994). O problema em questão aborda alguns pontos como: dinheiro, a idade da pessoa que receberá o presente e os tipos de presente. O problema procura achar um presente apropriado com base no dinheiro disponível e a idade da

pessoa. Num primeiro momento, para facilitar a elaboração e resolução do problema, assume-se que os atributos dinheiro e idade podem assumir valores tais como: dinheiro {muito, pouco}, e idade {criança, adulto}. O conhecimento a respeito deste problema é dado por um conjunto de exemplos obtidos por um especialista no assunto; e é representado na Tabela 5.1. Os conceitos de dinheiro e idade representam os fatores de decisão ou os atributos do problema. Dos exemplos mostrados na Tabela 5.1, pode-se induzir uma busca no espaço do problema conforme a Figura 5.3.

Tabela 5.1 - Tabela de decisão para o problema do presente

Fatores de decisão		Resultado
Dinheiro	**Idade**	**Presente**
muito	adulto	carro
muito	criança	computador
pouco	adulto	torradeira
pouco	criança	calculadora

Fonte: DURKIN, 1994.

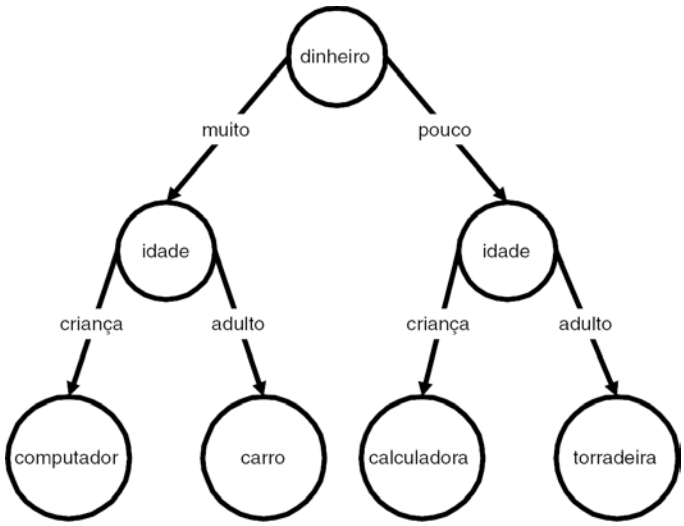


Figura 5.3 – Busca de estados do problema do presente

Por indução, pode-se criar uma árvore de decisão a partir do conjunto de exemplares para que se possa realizar o processo de busca, ou pode-se ainda criar um conjunto de regras para se utilizar em um sistema especialista. Para o exemplo apresentado na Tabela 5.1, as regras são:

SE Há muito dinheiro
E Se estiver comprando um presente para um adulto
ENTÃO Comprar um carro

SE Há muito dinheiro
E Se estiver comprando um presente para uma criança
ENTÃO Comprar um computador

SE Há pouco dinheiro
E Se estiver comprando um presente para um adulto
ENTÃO Comprar uma torradeira

SE Há pouco dinheiro
E Se estiver comprando um presente para uma criança
ENTÃO Comprar uma calculadora

A tarefa da indução é desenvolver regras de classificação que podem determinar a classe de um objeto através dos valores de seus atributos. Os objetos são descritos em termos de uma coleção de atributos (QUINLAN, 1983). Cada atributo mede alguma característica importante do objeto; e cada objeto, no domínio da aplicação, pertence a um conjunto de classes mutuamente exclusivas onde a classe deste objeto é conhecida.

Segundo Oliveira (2001), se o conjunto de treinamento contém dois objetos que têm valores idênticos para cada atributo e pertencerem a classes diferentes, é claramente impossível à diferenciação entre estes objetos com referência somente em seus dados atributos. E neste caso, os atributos serão considerados inadequados para o conjunto de treinamento e conseqüentemente para a tarefa de indução.

Um conflito ocorre quando dois exemplos contêm valores idênticos para todos os seus atributos, mas valores de classes diferentes. Um conflito normalmente significa que os atributos escolhidos são inadequados para a tarefa de classificação. Pode-se remover este problema introduzindo atributos adicionais, o que é uma tarefa para o especialista do domínio (THOMPSON, 1986 apud OLIVEIRA, 2001).

As folhas da árvore de decisão são os nomes da classe, os nós[1] representam testes baseados nos atributos com ramos rotulados, com os possíveis valores do atributo, para um resultado de classificação. Para classificar um objeto, começa-se da raiz da árvore, avalia-se o teste, ou seja, o nó da árvore é comparado com o respectivo atributo do objeto em questão, partindo pelo ramo determinado pelo valor do atributo do objeto que se pretende classificar, e o processo continua até que uma folha seja encontrada, na qual o objeto é afirmado a pertencer à classe nomeada pela folha.

O principal propósito da indução é construir árvores de decisão que possam classificar de forma correta os exemplares do conjunto de treinamento, mas também o restante dos exemplares que compõe o conjunto de dados.

5.3 CLS

O *Concept Learning System* (CLS), desenvolvido por Earl Hunt em 1966 (GESTWICKI, 1997), é um método algorítmico para resolver tarefas de aprendizado simples, utilizando-se dos conceitos aprendidos para classificar novos exemplos. Devido à natureza do CLS, este requer um pequeno número de possíveis valores discretos para os vetores de características.

O CLS pode encontrar uma regra de classificação ou uma árvore de decisão para uma determinada coleção de exemplos pertencente a duas classes de decisão; e utiliza esta regra para classificar um novo exemplo em uma destas classes. Um exemplo é classificado iniciando-se pela raiz da árvore, realizando testes, seguindo os ramos até que um determinado nodo seja alcançado. Este indicará SIM ou NÃO e o exemplo estará contido em uma das duas classes respectivas (DURKIN, 1994).

O algoritmo começa com uma árvore de decisão vazia e interativamente constrói a árvore adicionando nodos de decisão até que a árvore possa classificar de forma correta todos os exemplos de treinamento de um determinado conjunto *C*. Segue abaixo o procedimento do algoritmo CLS (COHEN e FEIGNBAUM, 1982 apud DURKIN, 1994):

1. Se todos os exemplos em *T* são positivos, então criar um nodo SIM e parar.

[1] Ponto de emergência de um ramo.

Se todos os exemplos em T são negativos, então criar um nodo NÃO e parar.

Caso contrário, seleciona-se (utilizando-se de algum critério heurístico) um atributo A com os valores v_1, v_2,..., v_n e cria-se um nodo de decisão baseado em A.

2. Dividir o conjunto de treinamento T em subconjuntos, T_1, T_2,..., T_k, agrupando os elementos de mesma característica v.
3. Aplicar o algoritmo de forma recursiva para cada conjunto T_j.

5.4 ID3

Em meados da década de 70, um pesquisador em inteligência artificial, chamado J. Ross Quinlan utilizou o modelo de formação de conceitos para desenvolver um programa chamado ID3 (*Itemized Dichotomizer 3*). O algoritmo desenvolvido por Quinlan "aprendeu", através de um pequeno conjunto de treinamento, como organizar e processar um amplo conjunto de dados. Ele utilizou a estratégia de dividir para conquistar, combinado com a lógica dicotômica para produzir resultados impressíveis em relação ao baixo tempo de processamento (GESTWICKI, 1997).

O ID3 é um algoritmo que procura usar a lógica e a matemática para processar, organizar e simplificar grandes conjuntos de dados (HOLSHEIMER e SIEBES, 1991). O algoritmo ID3 é descendente do *Concept Learning System* (CLS) e constrói árvores de decisão de maneira *top down*.

De acordo com Oliveira (2001), para a formação da árvore de decisão o conjunto de treinamento T é dividido em subconjuntos T_i onde esta divisão é feita de acordo com o atributo A, escolhido para raiz da árvore, e seus possíveis valores distintos do atributo; cada T_i conterá apenas os objetos de T com valores v_i de A e conseqüentemente cada T_i será menor que T. Sendo que, a partir do atributo raiz A partem os ramos rotulados com os possíveis valores distintos de A para cada subconjunto T_i. A idéia básica é "dividir e conquistar", isto é, proceder com esta escolha da raiz e conseqüentemente com a divisão do conjunto em subconjuntos para cada T_i, um por vez (estratégia de busca *top down* e *hill climber*), até encontrar uma folha para este. O resultado final será uma árvore para T, pois esta estratégia renderá subconjuntos até que satisfaçam a exigência de uma

classe para uma folha, ou seja, até que nestes subconjuntos tenham objetos pertencentes apenas a uma única classe, logo uma folha é encontrada.

Dado um conjunto de exemplares, cada exemplar possui a mesma estrutura, consistindo em um número pré-determinado de atributos. Um destes atributos representa a categoria do registro/exemplar. O problema é determinar a árvore de decisão que, com base nas respostas e perguntas com relação aos atributos não categóricos prediz corretamente o valor do atributo categórico. Normalmente o atributo categórico assume apenas valores {verdadeiro, falso}, ou {sucesso, falha}, ou algo equivalente.

Abaixo, na Tabela 5.2, há registros que informam as condições climáticas para a realização de um jogo de golfe. O atributo categórico especifica se deverá haver ou não o jogo. Os atributos não categóricos e seus possíveis valores são apresentados no Quadro I.

Quadro I – Atributos não categóricos

Atributo	**Possíveis valores**
céu	sol, nublado, chuva
temperatura	alta, baixa, suave
umidade	alta, normal
vento	sim, não

e o conjunto de treinamento é:

Tabela 5.2 – Conjunto de treinamento

Nº exemplar	**Céu**	**Temperatura**	**Umidade**	**Vento**	**Classe**
1	sol	alta	alta	não	não joga
2	sol	alta	alta	sim	não joga
3	nublado	alta	alta	não	joga
4	chuva	alta	alta	não	joga
5	chuva	baixa	normal	não	joga
6	chuva	baixa	normal	sim	não joga
7	nublado	baixa	normal	sim	joga

8	sol	suave	alta	não	não joga
9	sol	baixa	normal	não	joga
10	chuva	suave	normal	não	joga
11	sol	suave	normal	sim	joga
12	nublado	suave	alta	sim	joga
13	nublado	alta	normal	não	joga
14	chuva	suave	alta	sim	não joga

Para ilustrar os conceitos de conteúdo da informação e entropia, observa-se o seguinte exemplo:

M representa o conjunto {1, 2}. O número necessário de bits para representar um elemento de *M* é um, e o conteúdo da informação de cada bit é um item de dado.

A entropia pode ser definida matematicamente da seguinte maneira: *m* denota o número de elementos em *M* e n_a denota o número de instâncias de elementos *a* em *M*. Portanto, a probabilidade p_a de escolha de *a* em *M* é definida por:

$$p_a = \frac{n_a}{m} \tag{5.3}$$

Para um sistema com *n* classes, a entropia geral do sistema é definida por:

$$Entropia = \sum_{i=1}^{n} - p_i \log_2 p_i \tag{5.4}$$

Por exemplo, se o conjunto de dados *M* possui dois valores distintos, onde há probabilidade de se encontrar cada um dos valores é igual, a entropia é calculada da seguinte forma:

$$\begin{aligned} Entropia(M) &= -\left(\frac{1}{2}\right) * \log_2\left(\frac{1}{2}\right) - \left(\frac{1}{2}\right) * \log_2\left(\frac{1}{2}\right) \\ Entropia(M) &= 1 \end{aligned} \tag{5.5}$$

O algoritmo ID3 estabelece a árvore de decisão através da estratégia de dividir para conquistar. No princípio, apenas o modo raiz está presente na árvore. Para cada nodo da árvore a estratégia é aplicada, com o propósito de explorar a melhor escolha local.

O objetivo matemático principal do ID3 é reorganizar os dados de forma a criar uma eficiente árvore de decisão. Para cada nó da árvore são aplicados os seguintes passos ao conjunto de exemplos *T* (DURKIN, 1994, GESTWICKI, 1997, SALVATORE, 2000, AMADO, 2001):

Criar_Árvore(*T*)

(1) Determinar a distribuição dos exemplos pelas classes em *T*.

(2) Se todos os exemplos em *T* pertencem à mesma classe C_i, então retornar um nó folha que representa a classe C_i.

(3) Se os exemplos em *T* pertencem a mais do que uma classe:

(3.1) Encontrar a melhor divisão para o conjunto de exemplos *T*.

(3.2) Dividir o conjunto de exemplos *T* e construir um nó de decisão N.

(3.3) Para cada subconjunto T_j criado pela divisão:

(3.3.1) Filho *j* de *N* = Criar_Árvore(*T*)

(3.4) Retorna o nó de decisão N

No início da concepção de um nó da árvore de decisão, o método procura encontrar a distribuição de freqüência dos exemplos pelas classes para o conjunto de treinamento *T* associado ao nó (passo 1).

Para a determinação das distribuições, o algoritmo percorre o conjunto de exemplares associados ao nó para determinar o número de exemplos que pertence a cada uma das classes.

Segundo o passo 2, se todos os exemplos pertencem à mesma classe ou se o número for inferior a um determinado valor, então um nó ou nodo folha é associado a classe C_i (classe mais freqüente).

Se *T* possui casos que pertencem a duas ou mais classes, então *T* é dividido com base nos valores de um único atributo (passo 3.2). O nó é um nó de decisão que representa o teste realizado ao atributo escolhido; após a divisão do conjunto de exemplos os mesmos passos são aplicados de forma recursiva para cada um dos subconjuntos resultantes da divisão (passo 3.3.1).

O atributo que dá origem à divisão do conjunto de exemplos é determinado no passo 3.1. O tamanho e a precisão das árvores de decisão depende da escolha dos atributos para a divisão e representação dos exemplos. Para a determinação do melhor atributo, o algoritmo avalia o ganho de informação de cada atributo, ou seja, cada uma das possíveis divisões é avaliada. Será escolhido o atributo que possuir maior ganho de informação; este será o nó ou nodo de teste.

Ganho de informação

O ganho de informação de um atributo *A* para um conjunto de exemplares *T* é calculado pela equação 5.6. Se *A* é um atributo discreto e T_1, T_2, ...,T_j são subconjuntos de *T*, consistindo de exemplares com valores distintos do atributo *A*, então:

$$ganho = \text{info}(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} * \text{info}(T_i) \qquad 5.6$$

onde:

$$\text{info}(T) = -\sum_{j=1}^{k} \frac{freq(C_j, T)}{|T|} * \log_2 \left(\frac{freq(C_j, T)}{|T|} \right) \qquad 5.7$$

sendo:

$freq(C_j,T)$	o número de exemplos em *T* que pertencem à classe C_j
$\|T\|$	a cardinalidade de *T*, ou seja, o número total de exemplos
k	o número de classes
m	a divisão de *T* em *m* subconjuntos

A equação 5.7, quantidade *info*(*T*), é também conhecida como entropia, a qual mede a aleatoriedade de uma variável ou atributo

O processo de construção da árvore de decisão utilizando o método ID3 para a Tabela 5.2 dá-se da seguinte maneira:

Entropia do sistema

O objetivo do cálculo da entropia está na classificação *booleana* (jogar golfe versus não jogar golfe), em que há quatorze exemplos, nove positivos e cinco negativos; ou seja, *T* = [9+, 5-].

$$\begin{aligned}\text{info}(T) &= -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \ldots - p_n \log_2 p_n \\ &= -\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right) - \left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right) \\ &= 0{,}940\end{aligned}$$

Após calcular a entropia do sistema, busca-se qual o atributo possui melhor ganho de informação (equação 5.6):

Céu

O atributo céu pode assumir três valores, conforme o Quadro I.

T_{sol} = [2+, 3-], T_{nublado} = [4+, 0-] e T_{chuva} = [3+, 2-]

$$\begin{aligned}\text{info}(sol) &= -\left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) - \left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) \\ &= 0{,}52877 + 0{,}44217 \\ &= 0{,}97094\end{aligned}$$

$$\begin{aligned}\text{info}(nublado) &= -\left(\frac{4}{4}\right)\log_2\left(\frac{4}{4}\right) \\ &= 0\end{aligned}$$

$$\begin{aligned}\text{info}(chuva) &= -\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) \\ &= 0{,}44217 + 0{,}52877 \\ &= 0{,}97094\end{aligned}$$

Logo,

$$\begin{aligned}Ganho(\text{info(T)}, céu) &= 0{,}940 - \left(\frac{5}{14}\right)\cdot \text{info}(sol) - \left(\frac{4}{14}\right)\cdot \text{info}(nublado) - \left(\frac{5}{14}\right)\cdot \text{info}(chuva) \\ &= 0{,}940 - \left(\frac{5}{14}\right)\cdot 0{,}97094 - \left(\frac{4}{14}\right)\cdot 0 - \left(\frac{5}{14}\right)\cdot 0{,}97094 \\ &= 0{,}2464\end{aligned}$$

Temperatura

O atributo temperatura pode assumir três valores.

T_{alta} = [3+, 2-], T_{suave} = [3+, 1-] e T_{baixa} = [3+, 2-]

$$\text{info}(alta) = -\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) = 0{,}97094$$

$$\text{info}(suave) = -\left(\frac{3}{4}\right)\log_2\left(\frac{3}{4}\right) - \left(\frac{1}{4}\right)\log_2\left(\frac{1}{4}\right) = 0{,}811$$

$$\text{info}(baixa) = -\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) = 0{,}97094$$

Logo,

$$\begin{aligned} Ganho(\text{info}(T), temperatura) &= 0{,}940 - \left(\frac{5}{14}\right)\cdot \text{info}(alta) - \left(\frac{4}{14}\right)\cdot \text{info}(suave) - \left(\frac{5}{14}\right)\cdot \text{info}(baixa) \\ &= 0{,}940 - \left(\frac{5}{14}\right)\cdot 0{,}97094 - \left(\frac{4}{14}\right)\cdot 0{,}811 - \left(\frac{5}{14}\right)\cdot 0{,}97094 \\ &= 0{,}015 \end{aligned}$$

Umidade

O atributo umidade pode assumir dois valores.

T_{alta} = [3+, 4-] e T_{baixa} = [6+, 1-]

$$\text{info}(alta) = -\left(\frac{3}{7}\right)\log_2\left(\frac{3}{7}\right) - \left(\frac{4}{7}\right)\log_2\left(\frac{4}{7}\right) = 0{,}985228$$

$$\text{info}(baixa) = -\left(\frac{6}{7}\right)\log_2\left(\frac{6}{7}\right) - \left(\frac{1}{7}\right)\log_2\left(\frac{1}{7}\right) = 0{,}591672$$

Logo,

$$\begin{aligned} Ganho(\text{info}(T), umidade) &= 0{,}940 - \left(\frac{7}{14}\right)\cdot \text{info}(alta) - \left(\frac{7}{14}\right)\cdot \text{info}(baixa) \\ &= 0{,}940 - \left(\frac{7}{14}\right)\cdot 0{,}985228 - \left(\frac{7}{14}\right)\cdot 0{,}591672 \\ &= 0{,}151 \end{aligned}$$

Vento

O atributo vento pode assumir dois valores.

T_{sim} = [3+, 3-], $T_{\text{não}}$ = [6+, 2-]

$$\text{info}(sim) = -\left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) - \left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) = 1$$

$$\text{info}(não) = -\left(\frac{6}{8}\right)\log_2\left(\frac{6}{8}\right) - \left(\frac{2}{8}\right)\log_2\left(\frac{2}{8}\right) = 0{,}811278$$

Logo,

$$Ganho(\text{info}(T), vento) = 0{,}940 - \left(\frac{8}{14}\right)\cdot \text{info}(sim) - \left(\frac{6}{14}\right)\cdot \text{info}(não)$$
$$= 0{,}940 - \left(\frac{6}{14}\right)\cdot 1 - \left(\frac{8}{14}\right)\cdot 0{,}811278$$
$$= 0{,}047841$$

Desta maneira, o ID3 na concepção da árvore de decisão escolhe o atributo (céu) de maior ganho de informação para ser o nodo raiz da árvore (Figura 5.4).

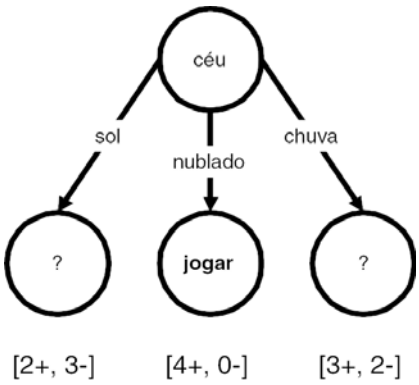


Figura 5.4 – Construção da árvore em nível parcial

Analisando-se a Figura 5.4, verifica-se que os ramos sol e chuva ainda estão indefinidos, e o processo deve continuar no próximo nível da árvore. Os exemplares do conjunto de treinamento *T* são divididos em subconjuntos de acordo com os valores do atributo céu, derivando em três subconjuntos.

O subconjunto T_1 possui elementos {1, 2, 8, 9, 11} pertencentes a duas classes, logo repete-se o processo de indução para este ramo da árvore.

Temperatura

T_{alta} = [2+, 0-], T_{suave} = [1+, 1-] e T_{baixa} = [0+, 1-]

$$\text{info}(alta) = -\left(\frac{2}{2}\right)\log_2\left(\frac{2}{2}\right) = 0$$

$$\text{info}(suave) = -\left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) - \left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) = 1$$

$$\text{info}(baixa) = -\left(\frac{1}{1}\right)\log_2\left(\frac{1}{1}\right) - \left(\frac{1}{1}\right)\log_2\left(\frac{1}{1}\right) = 0$$

Logo,

$$Ganho(\text{info}(sol), temperatura) = 0{,}97094 - \left(\frac{2}{5}\right)\cdot \text{info}(alta) - \left(\frac{2}{5}\right)\cdot \text{info}(suave) - \left(\frac{1}{5}\right)\cdot \text{info}(baixa)$$
$$= 0{,}97094 - \left(\frac{2}{5}\right)\cdot 0 - \left(\frac{2}{5}\right)\cdot 1 - \left(\frac{1}{5}\right)\cdot 0$$
$$= 0.57094$$

Umidade

T_{alta} = [3+, 0-] e T_{baixa} = [0+, 2-]

$$\text{info}(alta) = -\left(\frac{3}{3}\right)\log_2\left(\frac{3}{3}\right) = 0$$

$$\text{info}(baixa) = -\left(\frac{2}{2}\right)\log_2\left(\frac{2}{2}\right) = 0$$

Logo,

$$Ganho(\text{info}(s), umidade) = 0{,}97094 - \left(\frac{3}{5}\right)\cdot \text{info}(alta) - \left(\frac{2}{5}\right)\cdot \text{info}(baixa)$$
$$= 0{,}97094 - \left(\frac{3}{5}\right)\cdot 0 - \left(\frac{2}{5}\right)\cdot 0$$
$$= 0{,}97094$$

Vento

T_{sim} = [1+, 1-], $T_{\text{não}}$ = [2+, 1-]

$$\text{info}(sim) = -\left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) - \left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) = 1$$

$$\text{info}(não) = -\left(\frac{2}{3}\right)\log_2\left(\frac{2}{3}\right) - \left(\frac{1}{3}\right)\log_2\left(\frac{1}{3}\right) = 0{,}918295$$

Logo,

$$Ganho(\text{info}(sol), vento) = 0{,}97094 - \left(\frac{2}{5}\right)\cdot \text{info}(sim) - \left(\frac{3}{5}\right)\cdot \text{info}(não)$$
$$= 0{,}97094 - \left(\frac{2}{5}\right)\cdot 1 - \left(\frac{3}{5}\right)\cdot 0{,}918295$$
$$= 0.019963$$

Examinando-se os ganhos verifica-se que o atributo com o maior ganho de informação é o atributo umidade, o qual deve ser o nó seguinte na árvore.

Tabela 5.3 – Subconjunto (janela) T_1: céu igual a sol

Nº exemplar	Céu	Temperatura	Umidade	Vento	Classe
1	sol	alta	alta	não	não joga
2	sol	alta	alta	sim	não joga
8	sol	suave	alta	não	não joga
9	sol	baixa	normal	não	joga
11	sol	suave	normal	sim	joga

Já no subconjunto T_2, observa-se que todos os exemplares {3, 7, 12, 13} contidos nesse subconjunto pertencem somente a uma classe (jogar). Neste caso, o processo de indução cessa para este subconjunto e um nodo folha é rotulado com o nome desta classe.

Tabela 5.4 – Subconjunto (janela) T_2: céu igual a nublado

Nº exemplar	Céu	Temperatura	Umidade	Vento	Classe
3	nublado	alta	alta	não	joga
7	nublado	baixa	normal	sim	joga
12	nublado	suave	alta	sim	joga
13	nublado	alta	normal	não	joga

Para o subconjunto T_3, o processo deve continuar, pois este possui elementos {4, 5, 6, 10, 14} pertencentes a duas classes.

Temperatura

T_{alta} = [0+, 1-], T_{suave} = [1+, 1-] e T_{baixa} = [1+, 1-]

$$\text{info}(alta) = -\left(\frac{1}{1}\right)\log_2\left(\frac{1}{1}\right) = 0$$

$$\text{info}(suave) = -\left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) - \left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) = 1$$

$$\text{info}(baixa) = -\left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) - \left(\frac{1}{2}\right)\log_2\left(\frac{1}{2}\right) = 1$$

Logo,

$$Ganho(\text{info}(chuva), temperatura) = 0{,}97094 - \left(\frac{1}{5}\right)\cdot \text{info}(alta) - \left(\frac{2}{5}\right)\cdot \text{info}(suave) - \left(\frac{2}{5}\right)\cdot \text{info}(baixa)$$
$$= 0{,}97094 - \left(\frac{1}{5}\right)\cdot 0 - \left(\frac{2}{5}\right)\cdot 1 - \left(\frac{2}{5}\right)\cdot 1$$
$$= 0.17090$$

Vento

T_{sim} = [2+, 0-], $T_{\text{não}}$ = [0+, 3-]

$$\text{info}(sim) = -\left(\frac{2}{2}\right)\log_2\left(\frac{2}{2}\right) = 0$$

$$\text{info}(não) = -\left(\frac{3}{3}\right)\log_2\left(\frac{3}{3}\right) = 0$$

Logo,

$$Ganho(\text{info}(chuva), vento) = 0{,}97094 - \left(\frac{2}{5}\right)\cdot \text{info}(sim) - \left(\frac{3}{5}\right)\cdot \text{info}(não)$$
$$= 0{,}97094 - \left(\frac{2}{5}\right)\cdot 0 - \left(\frac{3}{5}\right)\cdot 0$$
$$= 0.97094$$

Do mesmo modo realizado para o subconjunto T_1, verifica-se que o atributo com maior ganho de informação é o atributo vento, o qual deve ser o nó seguinte na árvore.

Tabela 5.5 – Subconjunto (janela) T_3: céu igual a chuva

Nº exemplar	**Céu**	**Temperatura**	**Umidade**	**Vento**	**Classe**
4	chuva	alta	alta	não	joga
5	chuva	baixa	normal	não	joga
6	chuva	baixa	normal	sim	não joga
10	chuva	suave	normal	não	joga
14	chuva	suave	alta	sim	não joga

A Figura 5.5 apresenta a árvore de decisão final gerada pelo ID3 para o conjunto de treinamento da Tabela 5.2.

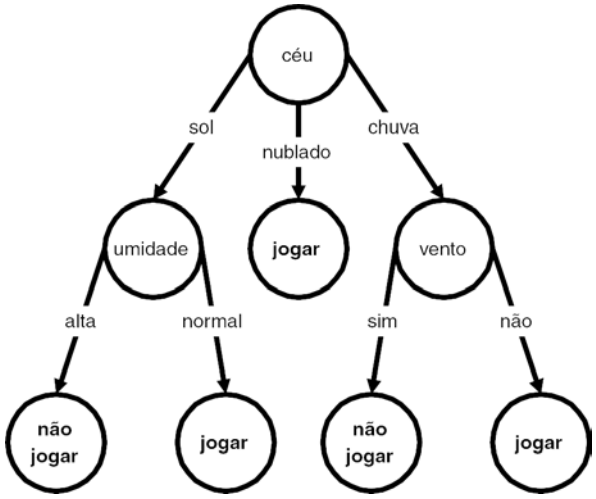


Figura 5.5 – Árvore de decisão final

Observa-se que o atributo temperatura não foi selecionado para fazer parte da árvore, devido ao fato do ID3 o ter considerado irrelevante para tarefa de classificação.

A idéia básica do ID3 é (INGARGIOLA, 1994):

- na árvore de decisão cada nodo corresponde a um atributo não categórico e cada arco a um possível valor respectivo a este. Um nodo folha da árvore especifica o valor da classe esperada do atributo categórico dos registros descritos pelo caminho entre o nodo raiz e o nodo folha.
- na árvore de decisão, cada nodo deve ser associado a um atributo não categórico o qual é mais informativo dentre os atributos não considerados no caminho entre este e a raiz.
- a entropia é usada para medir o quão informativo é um nodo.

Embora o ID3 utilize técnicas que empregam informações aproximadas para selecionar o atributo mais discriminante, este possui inúmeras limitações, tal como (DURKIN, 1994):

- as regras não são probabilísticas
- este não consegue lidar com exemplos contraditórios
- os resultados são excessivamente sensitivos a pequenas alterações no conjunto de treinamento
- o ID3 não manipula atributos contínuos

- admite um único tipo (discreto) de atributo

Há algumas versões do ID3 que realizam o processo de manipulação de atributos contínuos. Na Figura 5.6 é apresentada árvore de decisão gerada pelo software MLC++ da SGI (Silicon Graphics, Inc.). O gráfico foi gerado pelo software *Graphviz* (editor gráfico para *X Window System*) da AT&T e Bell Labs.

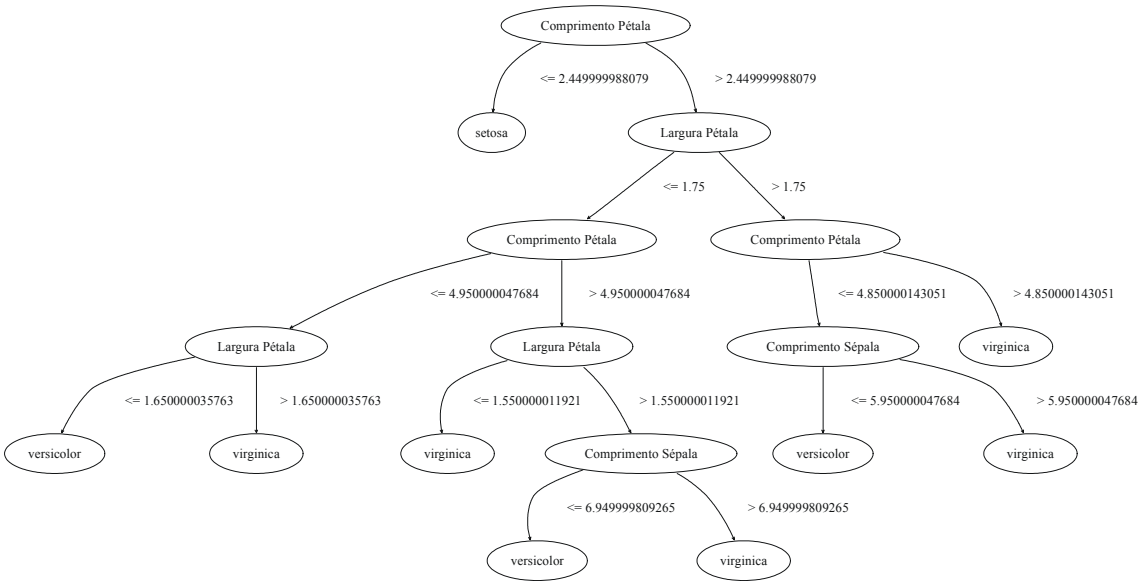


Figura 5.6 – Árvore de decisão da base íris gerada pelo MLC++

5.5 C4.5

Dentre os vários algoritmos de indução de árvores de decisão, o ID3 e seu sucessor c4.5 são os mais populares na comunidade científica. Estes algoritmos e suas variações são tema de inúmeros artigos de pesquisa desde que Quinlan apresentou o ID3.

O c4.5 gera um classificador que é capaz de agir como um especialista, classificando inclusive casos desconhecidos. Este foi desenvolvido com o intuito de tornar o modelo de classificação mais inteligível (POZO, 2002).

Assim como o ID3, o c4.5 constrói a árvore de decisão através do conjunto de treinamento, ou de exemplos, combinando uma estrutura de dados em árvore que pode ser usada para classificar novos exemplos. O c4.5 também emprega os uso da Teoria da

Informação para avaliar a qualidade de um nodo de teste (SEGRE, 1993). O algoritmo de uma forma peculiar extrai a máxima quantidade de informação de um conjunto de exemplares dada a condição de que somente um atributo será utilizado para realização do teste.

Este novo procedimento de indução de árvores de decisão apresenta uma série de extensões com relação ao algoritmo original ID3.

O c4.5 representa o resultado de vários anos de investigação na aprendizagem automática e na extração do conhecimento, sendo tomado como ponto de referência para o desenvolvimento de novos algoritmos; pois apresenta resultados que demonstram que este procedimento de indução de árvores de decisão oferece uma boa precisão na classificação e é considerado um dos mais rápidos (MICHIE et al. 1994, Salvatore 2000, Amado 2001).

Na construção da árvore de decisão o c4.5 lida com conjuntos de treinamentos que possuem exemplares com valores de atributos desconhecidos, avaliando o ganho de um atributo considerando apenas os registros que possuem atributos definidos. Na usabilidade da árvore de decisão, podem ser classificados registros que possuem valores de atributos desconhecidos através da estimativa de vários resultados. O c4.5 também trata o caso de atributos contínuos,cujo domínio pertence ao conjunto dos números reais (INGARGIOLA, 1994).

O processo de construção da árvore de decisão no c4.5 é realizada em dois momentos (AMADO, 2001). No primeiro momento, a árvore é disposta sob a aplicação da regra em que a construção da árvore é finalizada quando todos os exemplares associados a um nó pertencem à mesma classe ou se algum determinado critério de parada for especificado. No segundo momento, é realizada uma simplificação da árvore de decisão; cada subárvore é avaliada e serão reduzidas, a uma fração equivalente, aquelas que forem consideras insignificantes no processo de aumento significativo da precisão do conjunto.

Se os atributos temperatura e umidade da Tabela 5.2 fossem atributos contínuos, o ID3 (algoritmo original) construiria um nodo umidade com quatorze ramos, visto que o método cria um ramo para cada valor deste atributo. A Tabela 5.6 apresenta uma outra visão de como os algoritmo c4.5 montaria esta árvore, conforme a Figura 5.7.

Tabela 5.6 – Tabela de registros que informam as condições climáticas para a realização de um jogo de golfe com a utilização de atributos discretos e contínuos

Nº exemplar	**Céu**	**Temperatura**	**Umidade**	**Vento**	**Classe**
1	sol	85	85	não	não joga
2	sol	80	90	sim	não joga
3	nublado	83	78	não	joga
4	chuva	70	96	não	joga
5	chuva	68	80	não	joga
6	chuva	65	70	sim	não joga
7	nublado	64	65	sim	joga
8	sol	72	95	não	não joga
9	sol	69	70	não	joga
10	chuva	75	80	não	joga
11	sol	75	70	sim	joga
12	nublado	72	90	sim	joga
13	nublado	81	75	não	joga
14	chuva	71	80	sim	não joga

Os atributos temperatura e umidade são contínuos e os valores deste são analisados de forma ordenada. Seja $v=(v_1, v_2,..., v_n)$ o conjunto de valores possíveis para um determinado atributo. A ordenação dispõe os elementos de v em ordem crescente, ou seja, é realizada uma permutação dos elementos de v (equação 5.8), tal que, para quaisquer v_i e v_{i+1} em v, $v_i \leq v_{i+1}$ (DANDOLINI, 2000).

$$Perm(v) \qquad 5.8$$

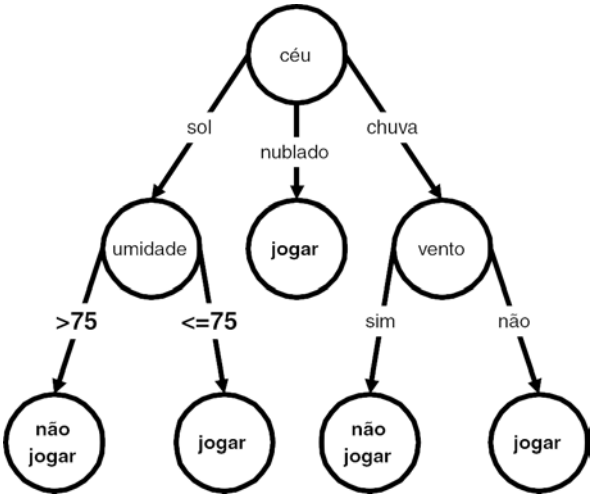


Figura 5.7 - Árvore de decisão gerada pelo C4.5 da Tabela 5.6

Para cada valor de um dos atributos contínuo v_i, $i = 1, 2,...n$, será particionado os exemplares considerando as seguintes condições (SALVATORE, 2000):

para cada $i \in [1, n\text{-}1]$ o valor de teste (ponto de partição ou divisão) será $v_p = \frac{(v_i + v_{i+1})}{2}$ e os valores dos ramos de partição $P_1^v = \{v_j \mid v_j \leq v\}$ e $P_2^v = \{v_j \mid v_j > v\}$.

A avaliação de uma divisão do conjunto de exemplos fundamentada num atributo contínuo analisa as n-1 possíveis divisões segundo o critério de ganho de informação. Será escolhido o valor que possuir maior ganho.

Para elucidar o conceito de divisão do conjunto de exemplos via um atributo contínuo, é apresentado o cálculo do ganho de informação para o atributo contínuo umidade definindo os valores de partição para o mesmo.

Utiliza-se a mesma janela utilizada no exemplo do método ID3; mas neste caso, utilizado-se atributos contínuos. O subconjunto T_1 possui os mesmos elementos {1, 2, 8, 9, 11}, conforme a Tabela 5.7.

Tabela 5.7 – Subconjunto (janela) T_1: atributos contínuos

Nº exemplar	Céu	Temperatura	Umidade	Vento	Classe
1	sol	85	85	não	não joga
2	sol	80	90	sim	não joga
8	sol	72	95	não	não joga
9	sol	69	70	não	joga

11	sol	75	70	sim	joga

Aplicado-se a função de permutação (equação 5.8) tem-se a Tabela 5.8. Após ordenar os exemplos por ordem crescente acha-se os pontos de partição $v_p = \frac{(v_i + v_{i+1})}{2}$.

Tabela 5.8 – Subconjunto (janela) T_1 após a aplicação da função de permutação

Nº exemplar	**Céu**	**Temperatura**	**Umidade**	**Vento**	**Classe**	**Pontos de partição**
9	sol	69	70	não	joga	v_{p1}
11	sol	75	70	sim	joga	
1	sol	85	85	não	não joga	v_{p2}
2	sol	80	90	sim	não joga	v_{p3}
8	sol	72	95	não	não joga	

Cálculo dos pontos de partição

$$v_{p1} = \frac{(70 + 85)}{2} = 77{,}5$$

$$v_{p2} = \frac{(85 + 90)}{2} = 87{,}5$$

$$v_{p3} = \frac{(90 + 95)}{2} = 92{,}5$$

Cálculo do ganho de informação para cada partição

(v_{p1})

$$p(jogar \mid umidade < 77{,}5) = \frac{2}{2} = 1$$

$$p(não\ jogar \mid umidade < 77{,}5) = \frac{0}{2} = 0$$

$$p(jogar \mid umidade > 77{,}5) = \frac{0}{3} = 0$$

$$p(não\ jogar \mid umidade > 77{,}5) = \frac{3}{3} = 1$$

$$\text{info}(umidade < 77{,}5) = -1 * \log_2(1) - 0 * \log_2(0) = 0$$

$$\text{info}(umidade > 77{,}5) = -0 * \log_2(0) - 1 * \log_2(1) = 0$$

$$\text{info}(umidade) = \frac{2}{5} * \text{info}(umidade < 77{,}5) + \frac{3}{5} * \text{info}(umidade > 77{,}5) = 0$$

$$Ganho(\text{inf } o(sol), umidade) = 0{,}97094 - \text{info}(umidade) = 0{,}97094$$

(v_{p2})

$$p(\,jogar \mid umidade < 87{,}5) = \frac{2}{3}$$

$$p(não\ jogar \mid umidade < 77{,}5) = \frac{1}{3}$$

$$p(\,jogar \mid umidade > 87{,}5) = \frac{0}{2} = 0$$

$$p(não\ jogar \mid umidade > 87{,}5) = \frac{2}{2} = 1$$

$$\text{info}(umidade < 87{,}5) = -\left(\frac{2}{3}\right) * \log_2\left(\frac{2}{3}\right) - \left(\frac{1}{3}\right) * \log_2\left(\frac{1}{3}\right) = 0{,}918$$

$$\text{info}(umidade > 87{,}5) = -0 * \log_2 0 - 1 * \log_2(1) = 0$$

$$\text{info}(umidade) = \frac{3}{5} * \text{info}(umidade < 87{,}5) + \frac{2}{5} * \text{info}(umidade > 87{,}5) = 0{,}550$$

$$Ganho(\text{inf } o(sol), umidade) = 0{,}97094 - \text{info}(umidade) = 0{,}420$$

(v_{p3})

$$p(\,jogar \mid umidade < 92{,}5) = \frac{2}{4} = \frac{1}{2}$$

$$p(não\ jogar \mid umidade < 92{,}5) = \frac{2}{4} = \frac{1}{2}$$

$$p(\,jogar \mid umidade > 92{,}5) = \frac{0}{1} = 0$$

$$p(não\ jogar \mid umidade > 92{,}5) = \frac{1}{1} = 1$$

$$\text{info}(umidade < 92{,}5) = -\left(\frac{1}{2}\right) * \log_2\left(\frac{1}{2}\right) - \left(\frac{1}{2}\right) * \log_2\left(\frac{1}{2}\right) = 1$$

$$\text{info}(umidade > 92{,}5) = -0 * \log_2 0 - 1 * \log_2(1) = 0$$

$$\text{info}(umidade) = \frac{4}{5} * \text{info}(umidade < 92{,}5) + \frac{1}{5} * \text{info}(umidade > 92{,}5) = 0{,}8$$

$$Ganho(\text{inf } o(sol), umidade) = 0{,}97094 - \text{info}(umidade) = 0{,}170$$

A partição que possui o maior ganho de informação é v_{p1}. Portanto a mesma será escolhida para o nodo de teste da árvore. O valor de teste nos ramos do atributo umidade pode ser o próprio valor de v_{p1}. Também há a possibilidade de se utilizar um valor que pertença ao conjunto de valores possíveis do atributo contínuo umidade. Este não deve ultrapassar o valor do ponto de partição. Para o caso apresentado, o valor 75 seria selecionado.

Abaixo, será apresentado o resultado dos testes realizados em quatro bases de dados utilizando-se o sistema de bibliotecas do C4.5 (QUINLAN, 1993). Este sistema foi desenvolvido para o Berkeley BSD 4.3 e está disponível na página pessoal de Quinlan. Para executar os testes para este trabalho, portou-se o referente sistema para o FreeBSD 4.4.

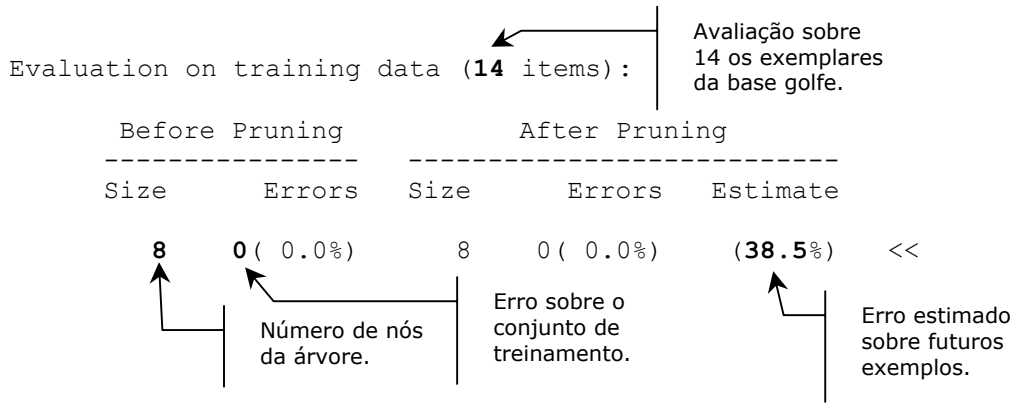
A primeira base de dados utilizada é uma base que reporta as condições necessárias para a realização de um jogo de golfe. Este exemplo foi apresentado no Workshop "Providing and Integrating Educational Resources for Faculty Teaching Artificial Intelligence" na Universidade da Filadélfia (EUA) – UGAI Lectures: Building Classification Models: ID3 and C4.5 (INGARGIOLA, 1994).

O conjunto de treinamento é mesmo da Tabela 5.6, o qual foi utilizado para apresentar o processo de construção da árvore de decisão utilizando-se o algoritmo C4.5. A saída gerada pelo C4.5 para o conjunto de treinamento golfe é apresentada abaixo:

```
C4.5 [release 8] decision tree generator  Tue Nov 12 02:07:12 2002
----------------------------------------

    Options:
      File stem <golf>

Read 14 cases (4 attributes) from golf.data

Decision Tree:

ceu = nublado: jogar (4.0)
ceu = sol:
|   umidade <= 75 : jogar (2.0)
|   umidade > 75 : nao jogar (3.0)
ceu = chuva:
|   vento = sim: nao jogar (2.0)
|   vento = nao: jogar (3.0)


Tree saved
```

Número total de exemplares do conjunto de treinamento.

4 exemplos que pertencem à classe jogar, com nenhum erro; somente se o valor do atributo céu for igual a nublado.



A árvore de decisão gerada para a base golfe apresenta oito (8) nós, e não possui nenhum erro de classificação.

A próxima base de dados a ser testada com o sistema é a íris, a qual já foi apresentada no Ensaio computacional: classificação de padrões – no capítulo que aborda as Redes Neurais Artificiais. Saída gerada pelo sistema:

```
C4.5 [release 8] decision tree generator  Tue Nov 12 02:17:36 2002
---------------------------------------

    Options:
      File stem <iris>

Read 150 cases (4 attributes) from iris.data

Decision Tree:

largura_petala <= 0.6 : setosa (50.0)
largura_petala > 0.6 :
|   largura_petala > 1.7 : virginica (46.0/1.0)
|   largura_petala <= 1.7 :
|   |   comprimento_petala <= 4.9 : versicolor (48.0/1.0)
|   |   comprimento_petala > 4.9 :
|   |   |   largura_petala <= 1.5 : virginica (3.0)
|   |   |   largura_petala > 1.5 : versicolor (3.0/1.0)


Tree saved


Evaluation on training data (150 items):

       Before Pruning           After Pruning
      ----------------   ---------------------------
      Size      Errors   Size      Errors   Estimate

         9    3( 2.0%)       9    3( 2.0%)    ( 6.5%)   <<
```

46 exemplos que pertencem à classe virginica com 1 erro.

Observa-se que neste caso, a árvore de decisão gerada pelo C4.5 apresenta três (3) erros de classificação; um exemplo mal classificado na classe virginica e dois na classe

versicolor. Analisando-se estes três casos patológicos, verificou-se que os exemplares que perturbam o conjunto são os mesmos encontrados no teste realizado com a rede neural artificial. Retirando-se os exemplares 71, 78 e 84, observa-se que os sistema C4.5 gera uma árvore de decisão sem erros, conforme demonstrado abaixo:

```
C4.5 [release 8] decision tree generator  Tue Nov 12 02:23:04 2002
----------------------------------------

    Options:
      File stem <iris>

Read 147 cases (4 attributes) from iris.data

Decision Tree:

largura_petala <= 0.6 : setosa (50.0)
largura_petala > 0.6 :
|   largura_petala > 1.6 : virginica (46.0)
|   largura_petala <= 1.6 :
|   |   comprimento_petala <= 4.9 : versicolor (47.0)
|   |   comprimento_petala > 4.9 : virginica (4.0)


Tree saved


Evaluation on training data (147 items):

       Before Pruning           After Pruning
      ----------------   ---------------------------
      Size      Errors   Size      Errors   Estimate

         7    0( 0.0%)      7    0( 0.0%)    ( 3.6%)   <<
```

Verifica-se ainda que a quantidade de nós necessários para representar o conhecimento passou de nove (9) para sete (7).

O próximo exemplo a ser apresentado é referente ao reconhecimento de classe de três tipos de vinhos. Os dados referentes à base de dados com 178 exemplares é resultante da análise química de 13 constituintes do vinho, cuja uva foi cultivada numa mesma região da Itália, mas derivada de três de cultivadores diferentes. Esta base também foi retirada da UCI Machine Learning Repository (BLAKE, 1998). Todos os 13 atributos são contínuos e a distribuição de classe ocorre da seguinte maneira: classe 1 com 59 exemplares; classe 2 com 71; e a classe 3 com 48. Segue abaixo a árvore de decisão gerada:

```
C4.5 [release 8] decision tree generator  Tue Nov 12 02:45:01 2002
----------------------------------------
```

```
    Options:
      File stem <wine>

Read 178 cases (13 attributes) from wine.data

Decision Tree:

g <= 1.57 :
|   j <= 3.8 : 2 (13.0)
|   j > 3.8 : 3 (49.0/1.0)
g > 1.57 :
|   m <= 720 : 2 (54.0/1.0)
|   m > 720 :
|   |   j <= 3.4 : 2 (4.0)
|   |   j > 3.4 : 1 (58.0)


Tree saved


Evaluation on training data (178 items):

       Before Pruning           After Pruning
      ----------------   --------------------------
      Size      Errors   Size      Errors   Estimate

         9    2( 1.1%)      9    2( 1.1%)    ( 5.1%)   <<
```

A árvore de decisão gerada pelo C4.5 apresenta 9 nós e 2 erros. Verificando-se os dois casos patológicos, constatou-se que os erros apresentados são referentes aos exemplares 44 e 62; que ao serem retirados do conjunto de treinamento, originou a seguinte árvore de decisão:

```
C4.5 [release 8] decision tree generator  Tue Nov 12 02:52:13 2002
---------------------------------------

    Options:
      File stem <wine>

Read 176 cases (13 attributes) from wine.data

Decision Tree:

g <= 1.57 :
|   j <= 3.8 : 2 (13.0)
|   j > 3.8 : 3 (48.0)
g > 1.57 :
|   m <= 720 : 2 (53.0)
|   m > 720 :
|   |   j <= 3.4 : 2 (4.0)
|   |   j > 3.4 : 1 (58.0)


Tree saved
```

```
Evaluation on training data (176 items):

       Before Pruning           After Pruning
      ----------------   --------------------------
      Size      Errors   Size      Errors   Estimate

         9     0( 0.0%)      9    0( 0.0%)    ( 3.7%)   <<
```

Observa-se que não há erros nesta árvore e que o número de nós que constitui a mesma permanece inalterado. Fato este que é contrariado no próximo exemplo.

O último exemplo a ser apresentado, utilizando-se o algoritmo C4.5, é uma base de dados de tipos de vidros. O estudo sobre a classificação dos tipos de vidro foi motivado pela investigação na área criminalista. Na cena do crime o vidro encontrado pode ser utilizado como uma evidência, desde que o tipo seja corretamente identificado.

A base de dados, também retirada da UCI Machine Learning Repository (BLAKE, 1998), contém 214 evidências, com 9 atributos. Abaixo são apresentadas as informações sobre os atributos e os tipos de vidros:

Atributos

RI	índice de refração
Na	Sódio
Mg	Magnésio
Al	Alumínio
Si	Silício
K	Potássio
Ca	Cálcio
Ba	Bário
Fe	Ferro

Tipos de vidro (atributo classe)

1	janelas para edifícios (*float_processed*)
2	janelas para edifícios (*non_float_processed*)

3	vidros automotivos (*float_processed*)
4	vidros automotivos (*non_float_processed*)[2]
5	recipientes
6	utensílios para mesa
7	bulbo de lâmpadas

Árvore gerada pelo C4.5:

```
C4.5 [release 8] decision tree generator  Tue Nov 12 03:02:45 2002
---------------------------------------

    Options:
      File stem <glass>
      Verbosity level 0

Read 214 cases (9 attributes) from glass.data

Decision Tree:

Ba <= 0.27 :
|   Mg <= 2.41 :
|   |   K <= 0.03 :
|   |   |   Na <= 13.75 : 2 (3.0)
|   |   |   Na > 13.75 : 6 (9.0)
|   |   K > 0.03 :
|   |   |   Na > 13.49 : 2 (7.0/1.0)
|   |   |   Na <= 13.49 :
|   |   |   |   RI <= 1.5241 : 5 (13.0/1.0)
|   |   |   |   RI > 1.5241 : 2 (3.0)
|   Mg > 2.41 :
|   |   Al <= 1.41 :
|   |   |   RI <= 1.51707 :
|   |   |   |   RI <= 1.51596 : 1 (3.0)
|   |   |   |   RI > 1.51596 :
|   |   |   |   |   Fe > 0.12 : 2 (2.0)
|   |   |   |   |   Fe <= 0.12 :
|   |   |   |   |   |   Al > 1.27 : 3 (5.0)
|   |   |   |   |   |   Al <= 1.27 :
|   |   |   |   |   |   |   Mg <= 3.47 : 3 (2.0)
|   |   |   |   |   |   |   Mg > 3.47 : 2 (2.0)
|   |   |   RI > 1.51707 :
|   |   |   |   K <= 0.23 :
|   |   |   |   |   Mg <= 3.34 : 2 (2.0)
|   |   |   |   |   Mg > 3.34 :
|   |   |   |   |   |   Si > 72.64 : 3 (3.0)
|   |   |   |   |   |   Si <= 72.64 :
|   |   |   |   |   |   |   Na <= 14.01 : 1 (14.0)
|   |   |   |   |   |   |   Na > 14.01 :
```

[2] Não utilizada nesta base

```
|   |   |   |   |   |   |   |   Al <= 0.51 : 1 (3.0)
|   |   |   |   |   |   |   |   Al > 0.51 :
|   |   |   |   |   |   |   |   |   Si <= 71.36 : 1 (3.0/1.0)
|   |   |   |   |   |   |   |   |   Si > 71.36 : 3 (2.0)
|   |   |   |   K > 0.23 :
|   |   |   |   |   Mg > 3.75 : 2 (10.0)
|   |   |   |   |   Mg <= 3.75 :
|   |   |   |   |   |   Fe <= 0.14 :
|   |   |   |   |   |   |   RI <= 1.52043 : 1 (36.0)
|   |   |   |   |   |   |   RI > 1.52043 : 2 (2.0/1.0)
|   |   |   |   |   |   Fe > 0.14 :
|   |   |   |   |   |   |   Al <= 1.17 : 2 (5.0)
|   |   |   |   |   |   |   Al > 1.17 : 1 (6.0/1.0)
|   |   Al > 1.41 :
|   |   |   Si > 72.49 : 2 (39.0/6.0)
|   |   |   Si <= 72.49 :
|   |   |   |   Ca <= 8.28 : 2 (6.0)
|   |   |   |   Ca > 8.28 : 3 (5.0/1.0)
Ba > 0.27 :
|   Si <= 70.16 : 2 (2.0/1.0)
|   Si > 70.16 : 7 (27.0/1.0)
```

Árvore de decisão simplificada (com poda); eliminação de ramos supérfluos da primeira árvore gerada.

Simplified Decision Tree:

```
Ba <= 0.27 :
|   Mg <= 2.41 :
|   |   K <= 0.03 :
|   |   |   Na <= 13.75 : 2 (3.0/1.1)
|   |   |   Na > 13.75 : 6 (9.0/1.3)
|   |   K > 0.03 :
|   |   |   Na > 13.49 : 2 (7.0/2.4)
|   |   |   Na <= 13.49 :
|   |   |   |   RI <= 1.5241 : 5 (13.0/2.5)
|   |   |   |   RI > 1.5241 : 2 (3.0/1.1)
|   Mg > 2.41 :
|   |   Al <= 1.41 :
|   |   |   RI <= 1.51707 :
|   |   |   |   RI <= 1.51596 : 1 (3.0/1.1)
|   |   |   |   RI > 1.51596 :
|   |   |   |   |   Fe > 0.12 : 2 (2.0/1.0)
|   |   |   |   |   Fe <= 0.12 :
|   |   |   |   |   |   Al > 1.27 : 3 (5.0/1.2)
|   |   |   |   |   |   Al <= 1.27 :
|   |   |   |   |   |   |   Mg <= 3.47 : 3 (2.0/1.0)
|   |   |   |   |   |   |   Mg > 3.47 : 2 (2.0/1.0)
|   |   |   RI > 1.51707 :
|   |   |   |   K <= 0.23 :
|   |   |   |   |   Mg <= 3.34 : 2 (2.0/1.0)
|   |   |   |   |   Mg > 3.34 :
|   |   |   |   |   |   Si <= 72.64 : 1 (22.0/4.8)
|   |   |   |   |   |   Si > 72.64 : 3 (3.0/1.1)
|   |   |   |   K > 0.23 :
|   |   |   |   |   Mg > 3.75 : 2 (10.0/1.3)
|   |   |   |   |   Mg <= 3.75 :
|   |   |   |   |   |   Fe <= 0.14 :
|   |   |   |   |   |   |   RI <= 1.52043 : 1 (36.0/1.4)
|   |   |   |   |   |   |   RI > 1.52043 : 2 (2.0/1.8)
|   |   |   |   |   |   Fe > 0.14 :
```

```
|   |   |   |   |   |   |   Al <= 1.17 : 2 (5.0/1.2)
|   |   |   |   |   |   |   Al > 1.17 : 1 (6.0/2.3)
|   |   Al > 1.41 :
|   |   |   Si > 72.49 : 2 (39.0/8.3)
|   |   |   Si <= 72.49 :
|   |   |   |   Ca <= 8.28 : 2 (6.0/1.2)
|   |   |   |   Ca > 8.28 : 3 (5.0/2.3)
Ba > 0.27 :
|   Si <= 70.16 : 2 (2.0/1.8)
|   Si > 70.16 : 7 (27.0/2.6)


Tree saved


Evaluation on training data (214 items):

       Before Pruning           After Pruning
      ----------------   --------------------------
      Size      Errors   Size      Errors   Estimate

        51   14( 6.5%)     45   16( 7.5%)    (20.9%)   <<
```

A árvore de decisão não simplificada para a base *glass* possui 51 nós com 14 exemplares classificados de forma incorreta. Após a simplificação (poda) de alguns ramos da árvore, a mesma apresenta 45 nós com 16 erros.

Analisando-se alguns casos patológicos nesta base, constatou-se que os exemplares 3, 4, 6, 20, 29, 79, 98, 127, 145, 152, 154 e 156 deturpam a conhecimento representado pela árvore.

Da mesma maneira que nos exemplos citados acima, retirou-se cada um destes exemplares da base e obtiveram-se os seguintes resultados:

Tabela 5.9 – Relação de retirada de caso patológico por número de elementos mal classificados (erro)

Exemplar	Erros na árvore de decisão
3	8
4	13
6	10
20	13
29	9
79	15
98	15

127	15
145	15
152	16
154	15
156	15

Verifica-se que nem sempre a retirada de um elemento do conjunto de treinamento reduzirá o número de erros apresentados na árvore. Fato este comprovado com a evidência 152.

Ainda, em outro teste, retirou-se todos os exemplares citados na Tabela 5.9, e verificou-se que a árvore gerada pelo algoritmo C4.5 ainda apresentava erros – neste caso, oito (8).

6 Conclusões e Recomendações

O poder computacional apresentado pelas redes perceptron de múltiplas camadas dá-se através das características que descrevem as mesmas, juntamente com a habilidade de aprendizado por experiência dado pelo treinamento supervisionado.

Além destas características importantes que fazem da rede neural um classificador profícuo, estas também são responsáveis pelas deficiências no estado atual do conhecimento e sobre o comportamento da rede.

A presença distribuída de não linearidade e a alta conetividade da rede de neurônios ocultos torna o processo de aprendizagem mais difícil de ser visualizado. É neste quesito que as árvores de decisão apresentam um bom modelo de representação do conhecimento adquirido, pois são fáceis de entender.

As árvores de decisão, além de apresentarem o conhecimento de forma explícita, produzem diferentes árvores a partir de um mesmo conjunto de dados; proporcionando diferentes caminhos (abordagens) de se classificar um novo exemplar.

Outra característica de suma importância é a capacidade que as árvores de decisão possuem de induzir conhecimento em grandes bases de dados, organizando os dados de forma compacta para qualquer tipo de dados. Deve-se salientar que este processo nem sempre é obtido com sucesso, pois a árvore tende se tornar complexa.

Outro problema encontrado na construção automática de árvores de decisão é o *overfitting*, semelhante às redes neurais, ou seja, a árvore de decisão não generaliza o conhecimento adquirido, mas sofre uma auto especialização; nós folha com apenas um único exemplar são criados para determinar a qual classe este pertence. Se o número de exemplares únicos em nós folha for suficientemente grande com relação ao número total de exemplares, verifica-se que a árvore não generalizou o conhecimento.

Para recomendações futuras, este trabalho apresenta algumas considerações importantes, tal como a utilização de indução automática de árvores de decisão para definir uma topologia mais adequada para uma rede neural artificial. Na maioria dos processos de concepção de topologias de redes neurais é empírico; principalmente para problemas não lineares, onde há a necessidade de se utilizar neurônios ocultos.

Também, se recomenda como trabalho futuro, explorar outros critérios de seleção de atributos e os valores para particionar a árvore, como alternativa aos processos tradicionais como a entropia.

7 Referências Bibliográficas

AMADO N. M. R. *Algoritmos Paralelos de Indução de Árvores de Decisão* – Faculdade de Engenharia da Universidade do Porto. Universidade do Porto, Porto, Dissertação de Mestrado, 2001.

BARONE D. A. C. *Projeto Revox.* Disponível em: <http://www.ucs.tche.br/revox>, 1999. Acesso em: 3 novembro 2000.

BLAKE C. L. & MERZ C., C.J. (1998). UCI Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science. Disponível em: <http://www.ics.uci.edu/~mlearn/MLRepository.html>. Acesso em: 3 abril 2002.

BISHOP C. M. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

CARVALHO A., LUDEMIR A. *Fundamentos de Redes Neurais Artificiais: 11ª Escola de Computação*. Imprinta Gráfica e Editora Ltda, 1998.

CHURCHLAND P. S., SJENOWSKI T. J. *The computacional Brain*. Cambridge, Mass.: MIT Press, 1992.

DANDOLINI G. A. *Mapa FAN no estagiamento automático do sono*. Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis.Tese (Doutorado em Engenharia de Produção), 2000.

DUDA R. O., HART P. E. *Pattern Classification and Scene Analysis*. New York, Willey and Sons, 1973.

DACS. *Data & Analysis Center for Software. Artificial Neural Networks Technology*. Disponível em: <http://www.dacs.dtic.mil>. Acesso em: 20 setembro 2000.

DURKIN J. *Expert systems*: design and development. New Jersey: Prentice Hall, 1994.

GESTWICKI P. *ID3: History, Implementation, and Applications* - 1997. Disponível em: <http://www.fredonia.edu/students/nixo1903/>. Acesso em: 31 maio 2002.

HAYKIN S. *Neural Networks: A comprehensive Foundation*. New York: Macmillan College Publish Company, 1994.

HOLSHEIMER M., SIEBES A. Data Mining: *The Search for Knowledge in Databases*. Amsterdam, The Netherlands, 1991. Disponível em: <http://citeseer.nj.nec.com/holsheimer91data.html>. Acesso em: 25 setembro 2002.

ICS. *Institute of Information & Computing Sciences – DataMining: cursusjaar 2001/2002*. Disponível em: http://www.cs.uu.nl/docs/vakken/dm/. Acesso em: 15 novembro de 2002.

INGARGIOLA G. *Building Classification Models: ID3 and C4.5*. A lecture on the UGAI Workshop, 1994 – "Providing and Integrating Educational Resources for Faculty Teaching Artificial Intelligence". Disponível em: <http://yoda.cis.temple.edu:8080/UGAIWWW/>. Acesso em: 21 outubro 2002.

KOVÁCS Z. L. *Redes Neurais Artificiais: Fundamentos e Aplicações*. Segunda Edição, Collegium Cognitio, 1996.

LUGER G. F., STUBBLEFIELD W. A., *Artificial Intelligence*. Addison Wesley, 1998.

LUGER G. F., STUBBLEFIELD W. A. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving* – 2nd Edition, The Benjamin/Cummings Publishing Company, Inc., 1993.

MANNILA H. *Data Mining: Machine Learning, Statistics, and Databases*. Eight International Conference on Scientific and Statistical Database Management, Stockholm, June, 1996, p. 1-8. Disponível em: <http://citeseer.nj.nec.com/52294.html>. Acesso em: 25 setembro 2002.

MARTINS G. A. *Estatística geral e aplicada*. São Paulo: Atlas, 2001.

MICHIE D., SPIEGELHALTER D. J., TAYLOR C. C. *Machine learning, Neural and Statistical Classification*. Ellis Horwood, 1994.

MONARD M. C., BATISTA G. E., KAWAMOTO S., PUGLIESI J. B. *Uma Introdução ao Aprendizado Simbólico de Máquina por Exemplos*, 1997. Disponível em: <http://labic.icmc.sc.usp.br/>. Acesso em: 21 outubro 2002.

OLIVEIRA A. F. N. *Uma metodologia de uso de técnicas de indução para criação de regras de sistemas especialistas* – Programa de Pós-Graduação em Engenharia de Produção. Universidade Federal de Santa Catarina. Florianópolis, Dissertação de Mestrado, 2001.

NASCIMENTO C. L. *Artificial Neural Networks in Control and Optimization*. University of Manchester, Tese de Doutorado. Manchester, 1994.

PILA A. D. *Seleção de Atributos Relevantes para Aprendizado de Máquina Utilizando a Abordagem de Rough Sets**. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo – ICMC/USP, 2001.

RIEDMILLER M. *Rprop – Description and Implementation Details*. Technical Report – Institut für Logik, Komplexität und Deduktionssyteme, University of Karlruhe, 1994.

SEGRE A. *Book Review: C4.5: Programs for Machine Learning by J. Ross Quinlan*. Morgan Kaufmann Publishers, Inc., 1993.

SHANNON C. E. *A Mathematical Theory of Communication*. The Bell System Technical Journal, Vol. 27, pp. 379-423, 623-656, July, October, 1948.

PATTERSON D. W. *Artificial Neural Networks: Theory and Applications*. Prentice Hall, 1995.

POZO A. T. R. *Aprendizado de Máquina*. Universidade Federal do Paraná, Departamento de Informática - Centro Politécnico. Disponível em: <http://www.inf.ufpr.br/~aurora/>. Acesso em: 1º outubro 2002.

POZO A. T. R. *Árvores de Decisão*. Universidade Federal do Paraná, Departamento de Informática - Centro Politécnico. Disponível em: <http://www.inf.ufpr.br/~aurora/>. Acesso em: 1º outubro 2002.

QUINLAN J. R. *Discovering Rules by Induction from Large Collection of Examples*, in Expert Systems in Microelectronic Age, D. Michie (Ed.), Edinburgh University Press, Edinburgh, 1979, pp. 169-201.

QUINLAN J. R. *Learning Efficient Classification Procedures and Their Application to ChessEnd Games*, in Machine Learning: An AI Approach, R. Michalski, T. Mitchell, and J. Carbonell (Eds.),Tioga Publishing, Palo Alto, 1, 1983, pp. 463-482.

QUINLAN J. R. *Personal Home Page – AI Group, CSE*. Disponível em: <http://www.cse.unsw.edu.au/~quinlan/>. Acesso em: 30 maio 2002.

SALVATORE R. *Efficient c4.5. Technical report*. IEEE Transactions on Knowledge and Data Engineering, Università di Pisa, 2000.

TAFNER M. A. *Reconhecimento de palavras isoladas usando redes neurais artificiais*. Programa de Pós-Graduação em Engenharia de Produção. Universidade Federal de Santa Catarina. Florianópolis, Dissertação de Mestrado, 1996.

TAFNER M., Xerez M., Rodrigues I. *Redes Neurais Artificiais: Introdução e Princípios de Neurocomputação*. EKO, 1996.

WANG X., Chen B., Qian G. *On the optmization of fuzzy decision trees*. Fuzzy Sets and Systems 112 (2000) 117-125.

WEISS S. M., Kulikowski C. A. Computer systems that learn: Classification and prediction methods from statistics, neural nets, machine learning, and expert systems. Morgan Kaufmann Publishers, Inc.,1991.