UNIVERSIDADE DO EXTREMO SUL CATARINENSE – UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

LIDIANE ROSSO RAIMUNDO

O ALGORITMO CART PELO CRITÉRIO DE GINI NA TAREFA DE CLASSIFICAÇÃO DA *SHELL* ORION *DATA MINING ENGINE*

CRICIÚMA, JULHO DE 2007

LIDIANE ROSSO RAIMUNDO

O ALGORITMO CART PELO CRITÉRIO DE GINI NA TAREFA DE CLASSIFICAÇÃO DA *SHELL* ORION *DATA MINING ENGINE*

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientador: Profª. M.Sc. Merisandra Côrtes de Mattos

CRICIÚMA, JULHO DE 2007

Aos meus pais, Valmir e Albertina, meu querido irmão Marcos e meus amigos pelo apoio concedido.

AGRADECIMENTOS

Agradeço a Deus pela sua presença constante em minha vida, pelo auxílio nas minhas escolhas e por me confortar nas horas difíceis.

Agradeço também:

A minha família, por todo amor, carinho, compreensão e apoio concedidos e também pela sólida formação dada em minha juventude, o que proporcionou continuidade nos estudos até hoje.

Ao meu namorado Geovani por estar sempre ao meu lado me confortando nas horas difíceis, transmitindo seu amor e compreensão, além da disposição para auxiliar quando necessário sem importar-se com dia ou hora solicitada.

A minha grande amiga Daiane pelo apoio, amizade e pelos momentos alegres que compartilhamos ao longo da nossa graduação.

Aos meus colegas Cassiano, Wilson e Dirlei pela ajuda dispensada durante a implementação efetuada neste trabalho de pesquisa e pela compreensão com relação a compensação de horas quando necessário.

A professora Alair pelas horas dispensadas na tentativa de resolução dos formalismos matemáticos envolvidos no trabalho.

Agradeço profundamente a grande incentivadora desta pesquisa que não mediu esforços para auxiliar na conclusão da mesma, minha professora e orientadora Merisandra, contribuindo também, além do conhecimento, com sua amizade.

Enfim, a todos que contribuíram para a execução desse trabalho, seja pela ajuda constante ou por uma simples palavra de amizade.

“Um homem nunca sabe
aquilo de que é capaz
até que o tenta fazer”.

(Charles Dickens)

RESUMO

O avanço computacional no que se refere ao processamento e armazenamento contribuiu para a formação de grandes repositórios de dados, tornando-se necessário o desenvolvimento de tecnologias destinadas à análise de informações e obtenção de novos conhecimentos. Dentre essas tecnologias o *data mining* constitui-se como uma das alternativas, pois é a principal etapa do processo de descoberta de conhecimento em bases de dados, utilizando para isso ferramentas computacionais. Essas ferramentas são em sua maioria comerciais, sendo que no Grupo de Pesquisa em Inteligência Computacional da UNESC encontra-se em desenvolvimento o projeto de uma *shell* gratuita denominada Orion *Data Mining Engine*. Esta pesquisa faz parte deste projeto e fundamentou-se na modelagem matemática e implementação do algoritmo CART para indução de árvores de decisão na tarefa de classificação do processo de *data mining* da *Shell* Orion. Na realização dos testes verificou-se que as árvores de decisão são representações simples do conhecimento e um meio eficiente de construir classificadores que estabelecem classes baseadas nos atributos de um conjunto de dados.

Palavras-Chaves: *Data Mining*, Classificação, Árvores de Decisão, Algoritmo CART, *Shell* Orion.

ABSTRACT

The computer advance in data processing and storage has contributed to the formation of great repositories of data, making necessary the development of technologies focused on the analysis of information and obtainment of new knowledge. Amongst these technologies the date mining consists as one of the alternatives, for, it is the main stage in the process of discovering knowledge in databases, making use of computer tools in order to do so. These tools are in its majority commercial, being that, in the Computational Intelligence Research Group at UNESC, it is found in development, the project of a free shell called Orion Date Mining Engine. This research is part of this project and it is also based on the mathematical modeling and implementation of the algorithm CART for induction of trees of decision in the task of classification of the process of data mining in the Shell Orion . During the tests, it was verified that the decision trees are simple representations of the knowledge and an efficient way to construct classifiers which establish classes based on the attributes of a data set.

Word-Key: Data Mining, Classification, Trees of Decision, Algorithm CART, Shell Orion

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

CART	Classification and Regression Trees
CEPSRM	Centro Estadual de Pesquisas em Sensoriamento Remoto e Meteorologia
CHAID	Chi Square Automatic Interaction Detection
DCBD	Descoberta de Conhecimento em Bases de Dados
DM	*Data Mining*
ID3	Induction of Decision Tree
KDD	Knowledge Discovery in Databases
OMS	Organização Mundial de Saúde
SGBD	Sistema Gerenciador de Banco de Dados
UFMG	Universidade Federal de Minas Gerais
UFRGS	Universidade Federal do Rio Grande do Sul
UML	Unified Modeling Language
UNESC	Universidade do Extremo Sul Catarinense

SUMÁRIO

1 INTRODUÇÃO

O avanço da tecnologia facilitou o armazenamento de informações e devido a isto, atualmente as organizações de médio e grande porte possuem grandes bases de dados. Porém, muitas dessas instituições desconhecem como proceder com elas e como utilizá-las em benefício próprio.

A análise desses dados tem sido efetuada por meio de métodos estatísticos, porém, para descobrir informações desconhecidas em uma base de dados, somente esse método matemático não é suficiente.

A fim de atender essa necessidade, surgiu o conceito de *Data Mining* (*DM*), envolvendo métodos estatísticos e técnicas da Inteligência Artificial, que consiste em uma das etapas do processo de Descoberta de Conhecimento em Bases de Dados (DCBD), originalmente denominado de *Knowledge Discovery in Databases* (KDD). *Data Mining* é composto por tarefas como: classificação, associação, regressão, clusterização e previsão de séries temporais, entre outros. O processo em si de descoberta do conhecimento se dá por meio da aplicação de métodos como: árvores de decisão, redes neurais, algoritmos genéticos, lógica nebulosa e outros.

Essas tarefas e métodos de *DM* para serem aplicados em bases de dados, encontram-se disponíveis em diferentes ferramentas, denominadas de *Shells*, sendo na sua grande maioria comerciais.

Considerando isto, o Grupo de Pesquisa em Inteligência Computacional Aplicada do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense vem desenvolvendo, por meio de seus alunos e responsáveis, uma *Shell* denominada Orion *Data Mining Engine*.

A *Shell* Orion possui implementadas as tarefas de associação e classificação. No módulo de classificação, o método primeiramente disponibilizado para indução de árvores de decisão foi o algoritmo ID3, tendo-se atualmente o desenvolvimento do algotitmo C4.5 que realiza a simplificação da árvore de decisão, característica esta não implementada pelo ID3.

No que se refere ao módulo de classificação da *Shell* Orion, esta pesquisa consistiu na modelagem matemática e implementação do algoritmo CART. Assim, pode-se ampliar o número de algoritmos disponíveis e permitir com isso, que sejam obtidos diferentes resultados por meio da tarefa de classificação nessa *shell*.

1.1 OBJETIVO GERAL

Implementar o algoritmo CART pelo critério de gini para indução das árvores de decisão no módulo de classificação da *Shell* Orion *Data Mining Engine.*

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos desta pesquisa consistem em:

a) entender o processo de *data mining*, a tarefa de classificação e o método de árvores de decisão;
b) compreender o algoritmo CART para indução de árvores de decisão;
c) demonstrar matematicamente o funcionamento do algoritmo CART para a tarefa de classificação;
d) aplicar o algoritmo CART na tarefa de classificação em *data mining*;

e) utilizar uma base de dados a fim de testar o funcionamento do algoritmo CART no módulo de classificação da *Shell* Orion.

1.3 JUSTIFICATIVA

Data Mining consiste em um processo no qual são exploradas grandes quantidades de dados, na busca de padrões consistentes ou relacionamentos entre variáveis para então validá-los e aplicá-los a novos subconjuntos de dados (CARVALHO, 2002).

Esse processo é composto por tarefas que possibilitam a descoberta de novos conhecimentos praticamente impossíveis de serem identificados pelos seres humanos, caso a base de dados a ser explorada possua uma grande quantidade de registros. Sendo assim, com o rápido crescimento das organizações e conseqüentemente aumento do volume de informações a serem armazenadas, torna-se quase que indispensável a utilização de uma ferramenta para auxílio na descoberta de novos conhecimentos e padrões que possam representar informações úteis para as organizações.

A *Shell* Orion *Data Mining Engine,* uma ferramenta para aplicação de tarefas e métodos de *DM*, é um projeto do Grupo de Pesquisa em Inteligência Computacional Aplicada do Curso de Ciência da Computação da UNESC que encontra-se em desenvolvimento. Dessa maneira, esta pesquisa tem como finalidade a implementação do algoritmo CART para a classificação de dados na *Shell* Orion.

A tarefa de classificação é um processo que encontra propriedades comuns entre um conjunto de registros pertencentes a um banco de dados e as classifica em diferentes classes de acordo com um modelo (HAN; KAMBER, 2001).

A Classificação é uma das tarefas mais importante e populares, que pode ser compreendida como a busca por uma função que permita associar cada registro de um banco de dados a uma única classe (GOLDSCHMIDT; PASSOS, 2005). Dentre os métodos disponíveis para a realização desta tarefa, a *Shell* Orion possui implementada até então o de árvores de decisão. Neste método, os resultados são apresentados em forma de árvore, permitindo que sua interpretação seja efetuada de maneira simples e clara, isto é, por meio da observação da árvore desenvolvida, fato este que, segundo Rodrigues (2005), tornou-se uma das principais vantagens de sua utilização.

A construção da árvore de decisão será realizada por meio da implementação do algoritmo *Classification and Regression Trees* (CART) que significa Árvores de Classificação e Regressão. O CART originou-se das áreas de inteligência artificial e estatística, podendo ser aplicado a problemas de classificação e regressão, diferenciando-se de alguns algoritmos pela capacidade de simplificação da árvore (poda).

Uma das principais características do algoritmo CART é a capacidade de pesquisa e relações entre os dados, mesmo quando elas não são evidentes, bem como a produção de resultados sob a forma de árvores de decisão que apresentam simplicidade e legibilidade (FONSECA,1994).

1.4 ESTRUTURA DO TRABALHO

Esta pesquisa é composta por 8 capítulos. No capítulo 1 encontra-se a contextualização ao tema proposto, bem como os objetivos e justificativa para realização deste trabalho.

Os conceitos fundamentais de *Data Mining* são abordados no capítulo 2 e são considerados indispensáveis para o entendimento da pesquisa realizada.

No capítulo 3 é apresentada a *Shell* Orion *Data Mining Engine*, uma ferramenta gratuita para aplicação de tarefas e métodos visando a descoberta de conhecimento em bases de dados.

As características da tarefa de classificação no processo de *data mining*, e o método de árvores de decisão são descritos respectivamente nos capítulos 4 e 5, onde são apresentados alguns exemplos de algoritmos que utilizam uma abordagem recursiva para a construção de árvores de decisão na tarefa de classificação.

No capítulo 6 é apresentado o algoritmo CART, o qual foi implementado pelo critério de Gini na tarefa de classificação da *Shell* Orion *Data Mining Engine*. Neste contexto são conceituados os elementos envolvidos no crescimento de uma árvore, os critérios de divisão disponíveis e a técnica da *poda* por minimização do custo-complexidade utilizada por esse algoritmo, envolvendo um vasto formalismo matemático. Já no capítulo 7 são abordados alguns exemplos de uso do algoritmo de classificação CART em *data mining*.

O trabalho desenvolvido, a demonstração matemática do algoritmo CART e resultados obtidos mediante a aplicação de uma base de dados para testes do seu funcionamento são apresentados no capítulo 8.

Por fim, tem-se a conclusão, onde também são descritas sugestões para trabalhos futuros.

2 DATA MINING

A habilidade de descobrir conhecimento útil e de agir sobre ele vem se tornando cada vez mais importante no mundo competidor em que se vive (KANTARDZIC, 2003).

O método normalmente utilizado para a descoberta desses conhecimentos costuma ser a elaboração de modelos estatísticos, porém estes apresentam limitações se levados em consideração alguns aspectos como (TONI, 2000):

a) a análise se torna trabalhosa quando o número de informações é grande;
b) seguidamente, possuem condições que limitam o número de casos a utilizar, fazendo com que apenas uma pequena parte do universo esteja disponível para análise;
c) poucas organizações dispõem de pessoal preparado para esta tarefa tão especializada, que mesmo com a ajuda de estatísticos, o projeto e a elaboração de modelos podem demorar para serem concluídos.

Considerando-se estes aspectos, surgiu o conceito de *Data Mining* (DM) que é caracterizado como o processo de descoberta de conhecimento em bases de dados que possa auxiliar na tomada de decisões nas organizações.

Data mining é a principal etapa do processo de descoberta de conhecimento[1] em bases de dados, sendo responsável pela busca efetiva do conhecimento, por este motivo alguns autores referem-se aos termos de *data mining* e descoberta de conhecimento como se tratando de um só conceito (GOLDSCHMIDT; PASSOS, 2005).

Han e Kamber (2001) conceituam *data mining* como o processo de descoberta de conhecimento a partir de grandes quantidades de dados armazenados

[1] Denominado de *Knowledge Discovery in Databases* (KDD), consiste no processo total de descoberta de conhecimento útil em bases de dados, sendo que o *Data Mining* representa uma das etapas desse processo (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

tanto em bancos de dados, *data warehouses*[2], ou qualquer outro meio de armazenamento de informações.

O processo de *data mining* consiste na utilização de algoritmos específicos para descobrir padrões nos dados, sendo que a aplicação incorreta pode resultar na descoberta de conhecimentos não relevantes, ou seja, em relações sem nenhum sentido ou inválidas (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

A descoberta de novos conhecimentos é um processo interativo e iterativo. Goldschmidt e Passos (2005) explicam que o mesmo é interativo pois exige a atuação de uma pessoa como sendo responsável pela atividade e iterativo devido a possibilidade de repetições do processo na busca de resultados satisfatórios.

A descoberta de novos conhecimentos em bases de dados envolve algumas etapas que o caracterizam do início ao fim, sendo que uma delas é a de *data mining*.

2.1 ETAPAS DO PROCESSO DE DESCOBERTA DE CONHECIMENTO

A descoberta de conhecimento em bases de dados é um processo composto basicamente por três etapas (Figura 1):

a) **pré-processamento:** nesta primeira etapa os dados são preparados para aplicação dos algoritmos de *data mining*. A captação, organização e tratamento dos dados são efetuados por meio de funções como (GOLDSCHMIDT; PASSOS, 2005):
   - **seleção dos dados**: selecionar os dados que serão utilizados durante o processo, conforme objetivo pré-estabelecido;

[2] Conjunto de banco de dados integrados, utilizado para armazenar grandes volumes de dados de sistemas de suporte à decisão (GOLDSCHMIDT; PASSOS, 2005).

- **limpeza dos dados**: garantir a qualidade dos dados, como por exemplo, a correção de informações errôneas, ausentes ou inconsistentes;
- **codificação dos dados**: codificar os dados a fim de possibilitar sua utilização nos algoritmos de *data mining*;
- **enriquecimento dos dados**: acrescentar informações nos registros existentes na base de dados de modo a aumentar o número de informações envolvidas no processo.

b) ***data mining***: significa a descoberta de conhecimento em bases de dados. Nesta etapa são definidas as tarefas e algoritmos de *data mining* que serão utilizados para atingir um determinado objetivo (OLIVEIRA, 2006).

c) **pós-processamento:** etapa responsável pela avaliação do conhecimento gerado. Segundo Romão (2006) o pós-processamento é utilizado para avaliar o processo de descoberta e efetuar a seleção do conhecimento que seja mais relevante. Goldschmidt e Passos (2005) afirmam que a avaliação, análise e interpretação dos resultados obtidos devem ser realizadas por dois especialistas: um em *data mining* e outro no domínio da aplicação. Além da avaliação dos resultados, são definidas as metas que devem ser alcançadas e as decisões a serem tomadas.

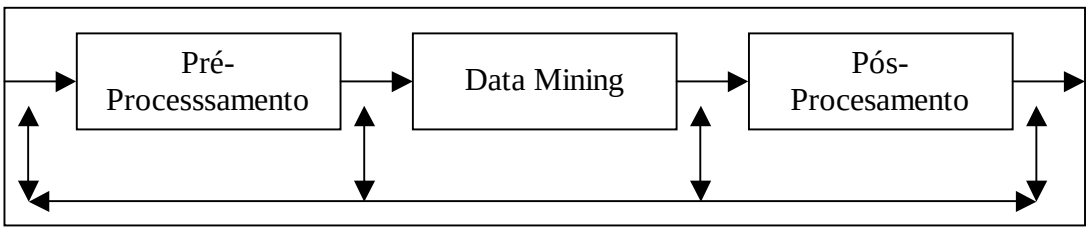

Figura 1. Etapas do processo de descoberta de conhecimento
Fonte: GOLDSCHMIDT, R.; PASSOS, E. (2005).

Dependendo do resultado que se deseja, o processo de *data mining* pode ser utilizado levando em consideração um determinado objetivo.

2.1.1 Objetivos do *Data Mining*

Baseando-se no conhecimento que se deseja obter, o *data mining* pode ser utilizado com os seguintes objetivos (TONI, 2000):

a) **explanatório**: necessidade de explicar algum evento ou medida observada, tal como, porque a venda de *softwares* caiu em determinada região;

b) **confirmatório**: confirmar uma hipótese. Uma determinada clínica, por exemplo, deseja analisar os registros de seus clientes para determinar se pacientes fumante possuem mais probabilidade de contrair doenças relacionadas ao coração do que pacientes não fumantes;

c) **exploratório**: analisar os dados buscando conhecimentos novos e relevantes, como por exemplo, uma empresa que desenvolve software para administração pública pode analisar os registros de seus clientes para determinar se as grandes cidades são as que mais adquirem licenças de seus produtos em relação a cidades menores.

Mediante a definição dos objetivos do processo de *data mining* pode-se utilizá-lo para descobrir, por exemplo, o que um cliente almeja de uma empresa, as relações existentes entre seus produtos, qual o setor que apresenta o maior índice de despesas, o funcionário cujas atividades renderam mais lucros à empresa, entre outros, e por meio das informações obtidas, efetuar uma previsão de vendas ou estabelecer novos rumos e diretrizes de negócios (OLIVEIRA, 2006).

A descoberta destes entre outros conhecimentos é efetuada por meio do processo de *data mining*, que possui algumas metodologias específicas.

2.1.2 Metodologias do *Data Mining*

Conforme descrito por Carvalho (2002) *data mining* pode ser realizado por meio de três diferentes metodologias:

a) **descoberta não supervisionada de relações**: não existe um problema específico a ser solucionado, porém a busca por novas relações;
b) **teste de hipótese**: baseia-se na procura por relações que podem ou não comprovar uma hipótese, obtendo uma validação ou não;
c) **modelagem matemática de dados**: esta metodologia é realizada por meio do conhecimento para a comprovação de que os dados obtidos são válidos.

Essas metodologias são aplicadas no processo de *data mining* independente da tarefa a ser utilizada para descobrir novos conhecimentos. As tarefas de *data mining* podem ser caracterizadas como diferentes abordagens aplicadas para a solução de vários problemas, sejam de escopo empresarial, econômico, entre outros.

2.1.3 Tarefas de *Data Mining*

As tarefas de *data mining* podem ser classificadas em duas categorias (HAN; KAMBER, 2005):

a) **descritiva**: caracteriza as propriedades gerais dos dados na base, de forma a torná-las compreensível pelo ser humano, como por exemplo, associação e clusterização;

b) **preditiva**: caracteriza a previsão de valores de um determinado atributo com base no histórico desses dados, como por exemplo, classificação, regressão e previsão de séries temporais.

A escolha de uma tarefa de *data mining* é feita de acordo com o problema e os objetivos do usuário, sendo as mais comuns: associação, classificação, regressão, clusterização e previsão de séries temporais (GOLDSCHMIDT; PASSOS, 2005).

A seguir têm-se as principais tarefas de data mining, seu objetivo e exemplo de aplicação:

a) **associação:** encontrar conjuntos de itens que possuam ocorrência simultânea e freqüente em um banco de dados. A tarefa de associação também é caracterizada como sendo a busca por regras de associação freqüentes e válidas, mediante a especificação de parâmetros de suporte e confiança mínimos, que são determinados pelos especialistas em *data mining* no domínio de aplicação(GOLDSCHMIDT; PASSOS, 2005). Esta tarefa pode ser aplicada, por exemplo, para identificar produtos que sejam vendidos em conjunto;

b) **classificação:** processo que encontra um conjunto de modelos ou funções que descrevem e distinguem classes ou conceitos de dados. Esses modelos ou funções obtidos são baseados na análise de um conjunto de dados de treinamento (registros cuja classe é conhecida) e podem ser utilizados para prever a classe em que os registros do banco de dados pertencem (HAN; KAMBER, 2001). Um exemplo de utilização

desta tarefa seria um professor classificar um aluno em um dos conceitos: ruim, bom e ótimo, considerando a freqüência, nota e participação deste aluno em sala de aula;

c) **regressão:** é similar a tarefa de classificação, sendo diferenciada apenas pelo tipo de atributo, pois a regressão está restrita apenas a atributos numéricos (BREIMAN et al, 1984). A tarefa de regressão pode ser utilizada, por exemplo, para efetuar a estimativa da probabilidade de um paciente sobreviver mediante o resultado de um conjunto de exames diagnósticos;

d) **clusterização:** objetiva realizar agrupamentos de registros em um conjunto de dados. Na tarefa de clusterização os registros são particionados em subconjuntos ou *clusters*, assim, os elementos de cada *cluster* compartilham propriedades comuns que servem para diferenciá-los dos elementos de outros grupos (GOLDSCHMIDT; PASSOS, 2005). Uma empresa que agrupa seus clientes por região do país, de modo a encontrar a maior concentração da venda de seus produtos pode ser citado como um exemplo de clusterização.

e) **previsão de séries temporais:** avaliar o valor futuro de um determinado índice com base em registros de seu comportamento no passado. Como exemplo de aplicação, pode ser citada a previsão de faturamento e custo de mão de obra para os próximos três meses de uma empresa de desenvolvimento de softwares.

Segundo Romão (2002) para a escolha da tarefa mais adequada é essencial um breve conhecimento a respeito do domínio de aplicação do *data mining*, como por

exemplo: os atributos importantes, os relacionamentos possíveis, a função útil para o usuário, que padrões já são conhecidos, entre outros.

As tarefas de *data mining* para serem realizadas necessitam que um determinado método seja adotado, de acordo com o conhecimento que se deseja extrair da base de dados. Conforme descrito por Scoss (2006) um método pode ser conceituado como um ou mais algoritmos implementados nas ferramentas de *data mining* disponibilizadas, objetivando a descoberta de conhecimento em bases de dados.

Alguns dos métodos bastante conhecidos são:

a) **árvores de decisão:** representada por uma estrutura hierárquica que traduz uma árvore invertida que se desenvolve da raiz para as folhas, onde a cada nível da árvore um problema complexo é decomposto em sub-problemas mais simples (RODRIGUES, 2005);

b) **redes neurais artificiais:** busca simular ou modelar a forma como o cérebro realiza uma determinada tarefa, por meio de seus neurônios. Uma rede neural é composta por unidades de processamento simples, chamadas elementos processadores, nodos ou simplesmente neurônios, que têm o objetivo de calcular determinadas funções (MOTTA, 2004);

c) **algoritmos genéticos**: são técnicas que procuram obter a melhor resposta para problemas complexos por meio da evolução de populações de soluções codificadas em cromossomas artificiais. É um processo adaptativo, pois as soluções existentes a cada instante influenciam na busca por outras e, paralelo devido ser decorrente do fato de que diversas soluções são consideradas a cada momento (GOLDSCHMIDT; PASSOS, 2005).

Esses métodos, bem como as tarefas mencionadas anteriormente, estão disponíveis em diversas ferramentas de *data mining*. Tais ferramentas possuem em sua maioria mais de uma tarefa de *data mining* implementada, sendo necessárias ao processo de descoberta de conhecimento em bases de dados, tornando-o mais rápido e qualificado.

Uma visão geral das ferramentas disponíveis, suas respectivas tarefas e alguns dos métodos implementados, podem ser observados na Tabela 1.

Tabela 1. Ferramentas de *data mining*

Ferramentas	Tarefas	Fabricante	Tipo
SPSS/ Clementine	Classificação, Regressão, Associação, Clusterização, Seqüência, Detecção de Desvios	SPSS Inc. www.spss.com	Comercial
PolyAnalist	Classificação, Regressão, Regras de Associação, Clusterização, Sumarização, Detecção de Desvios	Megaputer Intelligence www.megaputer.com	Comercial
Weka	Classificação, Regressão, Regras de Associação	Universit of Waikato www.cs.waikato.ac.nz	Gratuita
Darwin	Classificação	Thinking Machines Http://en.wikipedia.org/ Wiki/thinking_machines	Comercial
Intelligent Miner	Classificação, Regras de Associação, Clusterização, Seqüência, Sumarização	IBM www.ibm.com	Comercial
WizRule	Sumarização, Classificação, Detecção de Desvios	WizSoft Inc. www.wizsoft.com	Comercial
Bramining	Classificação, Regras de Associação, Regressão, Sumarização	Grall Corp. www.graal-corp.com.br	Comercial
SAS Enterprise Miner	Classificação, Regras de Associação, Regressão, Sumarização	SAS Inc. www.sas.com	Comercial
Oracle *Data Mining*	Classificação, Regressão, Associação, Clusterização, Mineração de Textos	Oracle www.oracle.com	Comercial
Orion *Data Mining* Engine	Classificação, Associação, Clusterização		Gratuita

Fonte: Adaptado de GOLDSCHMIDT, R.; PASSOS, E. (2005)

As ferramentas de *data mining* disponíveis, em sua maioria são licenciadas, resultando em uma carência com relação as gratuitas. Um exemplo de ferramenta gratuita é a *Shell* Orion *Data Mining* Engine, cujo desenvolvimento iniciou-se em 2005. Ela vem sendo desenvolvida por meio da aplicação dos conceitos abordados nos tópicos anteriores, que estão relacionados ao processo de *data mining*.

No que se refere a *Shell* Orion *Data Mining* Engine, esta pesquisa fundamentou a base para a implementação do algoritmo CART para indução de árvores de decisão no módulo de classificação. Portanto, neste contexto, essa ferramenta adquire um espaço dedicado à sua apresentação.

3 A *SHELL* ORION *DATA MINING ENGINE*

A *Shell* Orion é uma ferramenta de *data mining* que vem sendo desenvolvida pelos alunos e responsáveis do Grupo de Pesquisa em Inteligência Computacional Aplicada do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense.

A ferramenta está sendo implementada por meio da linguagem de programação Java, que segundo Pelegrin (2005) foi escolhida devido algumas características como: possibilidade de reutilização do código; ser multiplataforma e pelo fato de suas ferramentas de programação serem gratuitas.

A *Shell* Orion possui atualmente duas tarefas: Associação e Classificação. O módulo de associação, implementado por meio do algoritmo *Apriori,* foi desenvolvido no ano de 2005, como Trabalho de Conclusão de Curso intitulado: O Módulo da Técnica de Associação pelo Algoritmo *Apriori* no Desenvolvimento da *Shell* de *Data Mining* Orion (CASAGRANDE, 2005).

No que se refere ao módulo de classificação, implementado por meio do algoritmo ID3, o mesmo foi desenvolvido também em 2005, como Trabalho de Conclusão de Curso intitulado: A Tarefa de Classificação e o Algoritmo ID3 para Indução de Árvores de Decisão na *Shell* de *Data Mining* Orion (PELEGRINI, 2005).

A evolução no desenvolvimento desta ferramenta pode ser acompanhado na Tabela 2, onde são descritas as etapas e tarefas bem como os algoritmos já implementados e os que se encontram em desenvolvimento.

Tabela 2. Evolução da *Shell* Orion *Data Mining Engine*

Etapas do processo de descoberta de conhecimento

Período	Pré Processamento	Data Mining			Pós processamento
		Associação	Classificação	Clusterização	
2005.1 a 2005.2	-	A priori	ID3	-	-
2006.2 a 2007.1	-	-	CART	Koronen e K-means	-
2007.2	X	-	C4.5	Gustafson-Kessel	-

Na *Shell* Orion, para que o usuário tenha acesso a um dos módulos da ferramenta, é necessário efetuar a conexão com o banco de dados a ser explorado. As opções de conexão, bem como o meio de acesso ao módulo desejado estão disponíveis na interface principal.

3.1 INTERFACE DA *SHELL* ORION

A interface principal da *Shell* Orion (Figura 2) está dividida em quatro blocos, descritos abaixo:

a) **1º bloco**: identifica o protótipo e apresenta o logotipo do projeto;

b) **2º bloco**: apresenta as informações com relação ao banco de dados utilizado, possuindo quatro campos a serem preenchidos:

- **driver JDBC:** deve ser informado o Sistema Gerenciador de Banco de Dados (SGBD) que será utilizado, tendo-se atualmente disponíveis os *drivers*: PostgreSQL, Firebird e MySQL,
- **nome do BD:** é necessário que seja preenchido neste campo o nome do banco de dados a ser minerado,

- **usuário:** fornecer o nome do usuário para conexão com o banco de dados,
- **senha:** informar a senha correspondente ao usuário que deseja acessar o banco de dados;

c) **3º bloco**: após definida a conexão, o terceiro bloco torna-se ativo e possibilita a escolha da tabela por meio do campo Tabela**,** que apresentará seus atributos para visualização;

d) **4º bloco**: com todos os parâmetros indicados, torna-se possível a escolha da tarefa de *data mining*.

Figura 2. Interface principal da *Shell* Orion
Fonte: PELEGRINI, D. (2005); CASAGRANDE; D. (2005)

No que se refere as conexões da *Shell* Orion o objetivo é proporcionar aos usuários várias opções para descoberta de conhecimento dos dados armazenados em diferentes SGBD. Algumas dessas opções de conexão já encontram-se implementadas na Orion.

3.2 CONEXÕES DA *SHELL* ORION

A *Shell* Orion disponibiliza a conexão com os *drivers*: PostgreSQL[3], Firebird[4] e MySQL[5]. Estes *drivers* possuem licença *freeware* e o d*ownload* dos mesmos pode ser efetuado por meio do *site* dos fabricantes. A decisão de implementar a conexão com esses bancos de dados, segundo Casagrande (2005), deu-se em função dos mesmos possuírem licença *freeware*, bom desempenho e confiabilidade.

Quanto maior o número de conexões que a *Shell* Orion proporcionar, sendo com bancos de dados *freeware* (sem licença), como é o caso das conexões já implementadas, ou com bancos de dados *shareware* (com licença), maiores serão as funcionalidades e opções de uso da ferramenta.

A partir do momento em que a conexão com o banco de dados é estabelecida, pode-se definir qual tarefa de *data mining* será utilizada para a descoberta de novos conhecimentos.

3.3 TAREFAS DE *DATA MINING* DA *SHELL* ORION

As tarefas de *data mining* atualmente implementadas na *Shell* Orion são: associação e classificação, que estão dispostas em módulos separados, possuindo etapas

[3] PostgreSQL: (http://www.postgresql.org)
[4] Firebird: (http://www.borland.com)
[5] MySQL (http://www.mysql.com)

onde devem ser transmitidas as informações necessárias para a sua execução e de seu respectivo algoritmo.

O módulo de associação da *Shell* Orion (Figura 3) possui implementado o algoritmo *Apriori*, sendo possível encontrar os relacionamentos entre os itens da base de dados (CASAGRANDE, 2005). Segundo Goldschmidt e Passos (2005) o algoritmo é dividido em duas etapas: na primeira, são encontrados todos os conjuntos de ítens frequentes que satisfazem uma determinada condição; na segunda etapa são geradas regras de associação, a partir do conjunto de ítens encontrado primeiramente.

O algoritmo *Apriori* é considerado um clássico, devido a sua grande utilização para encontrar *itemsets* (grupo de ítens ou de subconjuntos) em grandes bases de dados (ARBEX, 2005).

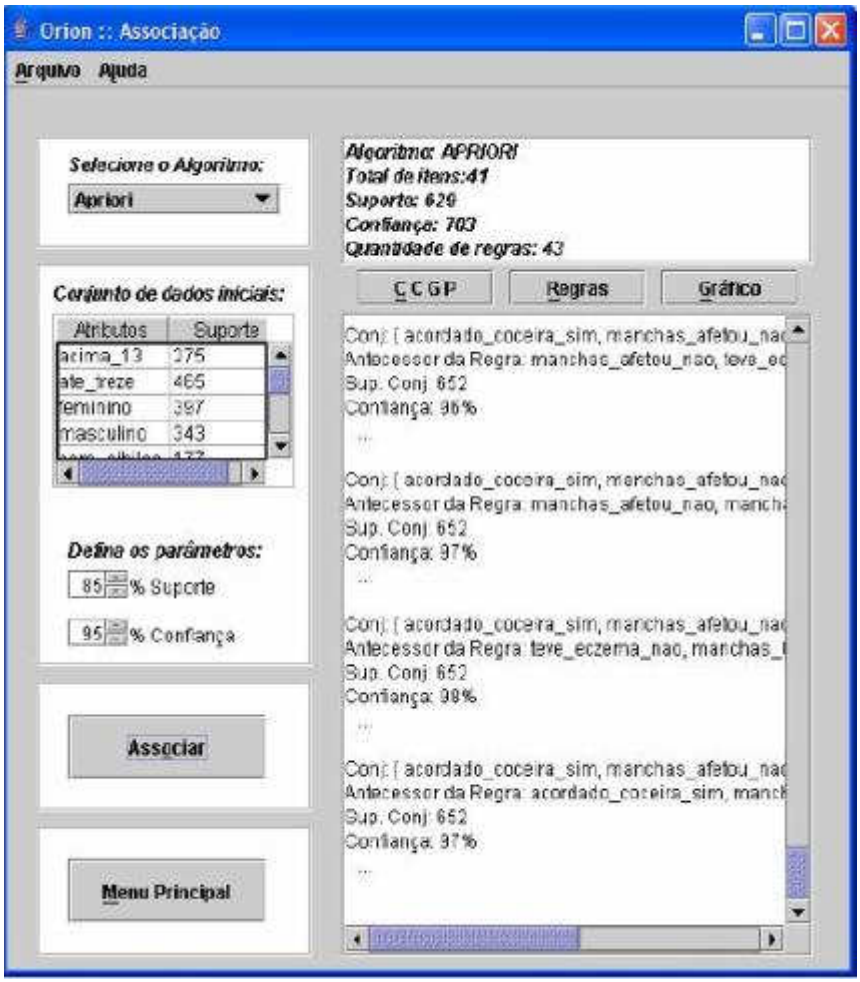

Figura 3. Módulo de associação da *Shell* Orion
Fonte: CASAGRANDE; D. (2005)

No que se refere ao módulo de classificação (Figura 4) a *Shell* Orion possui implementado o algorítmo ID3, que segundo Rodrigues (2005) consiste em um processo de indução de árvores de decisão, desenvolvida de cima para baixo, tendo como principal objetivo a escolha do melhor atributo para cada nó de decisão da árvore.

A representação por meio de árvores de decisão permite ao usuário visualizar, de forma clara, o fator que mais direciona o seu objeto de saída (SERRA, 2002).

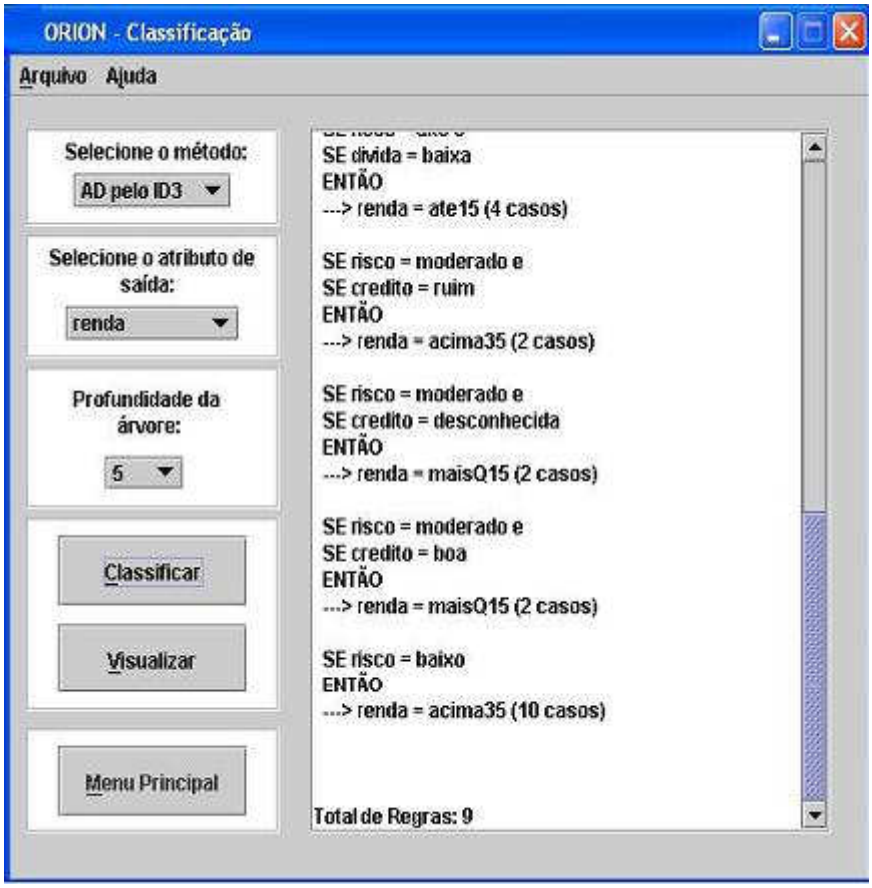

Figura 4. Módulo de classificação da *Shell* Orion
Fonte: PELEGRINI, D. (2005)

A função do módulo de classificação da *Shell* Orion por meio do algoritmo ID3 consiste em produzir regras de classificação, onde o objeto de saída deve ser selecionado pelo usuário, e gerar uma árvore gráfica (Figura 5) possibilitando uma

melhor visualização em relação ao atributo que mais influenciou na construção do modelo de conhecimento obtido (PELEGRIN, 2005).

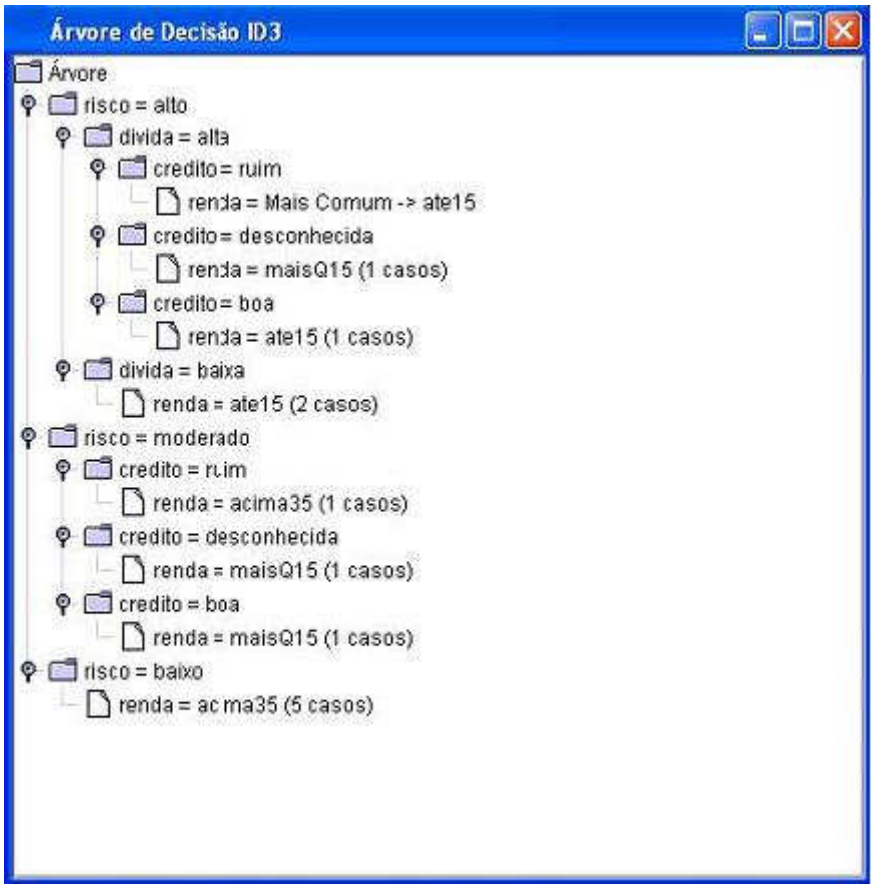

Figura 5. Visualização gráfica da árvore de decisão construída por meio do ID3
Fonte: PELEGRINI, D. (2005)

Além dessas duas tarefas e métodos de *data mining* já implementados, o projeto de desenvolvimento da *Shell* Orion implica na implementação de outros, a fim de ampliar o número de opções para descoberta de conhecimento em bases de dados.

Sendo assim, com base no objetivo de desenvolvimento de uma ferramenta de *data mining* com várias tarefas, métodos e algoritmos implementados, esta pesquisa consiste na modelagem matemática e implementação do algoritmo CART para indução de árvores de decisão na tarefa de classificação. Dessa maneira, pode-se ampliar o número de algoritmos disponíveis e permitir com isso, que diversos métodos sejam

oferecidos, podendo-se inclusive obter diferentes resultados por meio da tarefa de classificação.

A tarefa de classificação é apresentada no próximo capítulo, com o intuito de proporcionar a fundamentação teórica básica para o entendimento da mesma no processo de descoberta de conhecimento em bases de dados.

4 A TAREFA DE CLASSIFICAÇÃO NO PROCESSO DE DATA MINING

Considerada uma das mais conhecidas e utilizadas tarefas de data *mining, a* classificação é um processo que encontra propriedades comuns entre um conjunto de registros pertencentes a um banco de dados e as classifica em diferentes classes de acordo com um modelo (GOLDSCHMIDT; PASSOS, 2005).

Nessa tarefa, os registros pertencentes ao banco de dados a ser explorado são divididos em dois conjuntos (RUSSEL; NORVIG, 2004):

a) **dados de treinamento**: composto pelos registros que serão utilizados na elaboração de um modelo de classificação;

b) **dados de teste**: possui os registros a serem utilizados na avaliação do modelo de classificação gerado.

Os dados do treinamento são analisados por um algoritmo de classificação e os dados de teste são utilizados para estimar a exatidão das regras de classificação. Caso a exatidão seja aceitável, as regras podem ser aplicadas à classificação de novos dados (HAN; KAMBER, 2001).

A descoberta de modelos de classificação em bases de dados é composta por duas fases (GONÇALVES, 2006):

a) **aprendizagem**: aplicação de um algoritmo classificador sobre o conjunto dos dados de treinamento, obtendo-se como resultado o classificador propriamente dito;

b) **teste**: avaliação da acurácia[6] por meio da utilização do conjunto de dados de teste. Caso a acurácia seja alta, o modelo gerado é considerado eficiente, podendo ser utilizado na classificação de novos dados.

[6] Medida de desempenho de um classificador que utiliza a taxa de classificação incorreta do mesmo (GOLDSCHMIDT; PASSOS, 2005).

Os conjuntos de dados são as variáveis, também denominadas de atributos, envolvidas no problema a ser solucionado, e podem ser classificadas em (GOLDSCHMIDT; PASSOS, 2005):

a) **nominais ou categóricas**: utilizadas para nomear rótulos a objetos. São representadas por tipos de dados alfanuméricos e não existe um ordenamento de seus valores. Exemplos: solteiro, casado, viúvo, divorciado;
b) **discretas**: semelhante as variáveis nominais, sendo também representada por tipos de dados alfanuméricos. O que as diferencia é o fato de que as discretas podem assumir um ordenamento. Exemplos: segunda-feira (logo após o domingo), faixa de renda e faixa etária;
c) **contínuas**: representam variáveis quantitativas (dados numéricos) com relação de ordem entre si. Exemplos: renda e idade.

As variáveis de entrada podem assumir valores categóricos ou contínuos, sendo que na tarefa de classificação o objeto de saída deve ser categórico. Considerando isso, o processo de classificar pode ser entendido como a busca por uma função que possibilite a correta associação de cada registro de um conjunto de informações X, a um único registro de um conjunto Y, que corresponde a classe. Após a definição de tal função, a mesma pode ser utilizada para prever a classe de novos registros. Este exemplo pode ser observado por meio da Figura 6 (GOLDSCHMIDT; PASSOS, 2005).

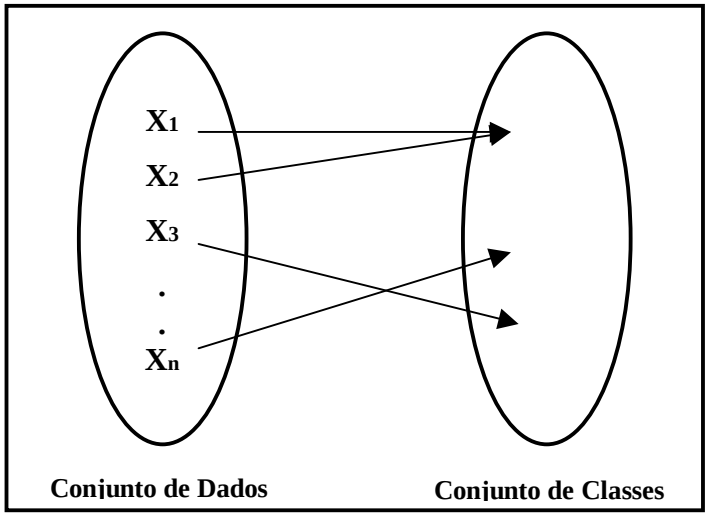

Figura 6. Associação entre registros de dados e classes
Fonte: GOLDSCHMIDT, R.; PASSOS, E. (2005)

De modo a exemplificar o processo de classificação mencionado acima, considere os dados contidos na Tabela 2.

Tabela 3. Base de Dados (Conjunto de Dados de Treinamento)

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

Fonte: GOLDSCHMIDT, R.; PASSOS, E. (2005)

O conjunto de dados de treinamento contido na Tabela 2 é composto por atributos que são selecionados na base de dados a ser analisada. Cada registro, representado pelos atributos *Aparência, Temperatura, Umidade, Vento e Jogar Golfe*, contém os dados necessários para oferecer uma recomendação quanto a prática de golfe. O atributo *Jogar Golfe*, que pode obter como saída somente os valores *Sim* e *Não*, representa o rótulo categórico Y (Figura 6). Este atributo representa a classe e os demais (*Aparência*, *Temperatura*, *Umidade* e *Vento*) são os preditivos, ou seja, utilizados para desenvolver o classificador.

Além deste exemplo de utilização, a tarefa de classificação pode ser aplicada em: diagnóstico médico, risco de crédito, detecção de intrusão, análise de risco de seguro, entre outras.

Independente da área aplicada deve-se utilizar um critério para a avaliação da qualidade das regras descobertas na tarefa de classificação. Os três mais utilizados são (CARVALHO, 2005):

a) **previsão preditiva**: é o resultado da divisão entre o número de exemplos de teste classificados corretamente e o número total de exemplos de teste;

b) **compreensibilidade**: geralmente é medida pela simplicidade, sendo obtida em função do número de regras descobertas e da quantidade de condições por regra. Quanto maiores estes números, menos compreensível é o conjunto de regras;

c) grau de interesse do conhecimento descoberto: pode ser obtido por meio da avaliação do conhecimento descoberto do ponto de vista do usuário**.**

A aplicação da tarefa de classificação, assim como das demais tarefas, é efetuada por meio de métodos. Diversos métodos podem ser utilizados para a classificação como: árvores de decisão, redes neurais, algoritmos genéticos, entre outros (GONÇALVES, 2006).

Dentre esses métodos, o de árvores de decisão foi implementado na tarefa de classificação da *Shell* Orion por meio algoritmo CART. Assim, os resultados são apresentados em forma de árvore, permitindo que sua interpretação seja efetuada de forma simples e clara.

5 O MÉTODO DE ÁRVORES DE DECISÃO NA TAREFA DE CLASSIFICAÇÃO

O método de árvores de decisão é uma abordagem de aprendizado supervisionada que, por ser um método eficiente, é bastante utilizado na construção de um classificador de dados (KANTARDZIC, 2003).

Uma abordagem supervisionada compreende a abstração de um modelo de conhecimento a partir dos dados apresentados na forma de pares ordenados (entrada, saída desejada). Esse modelo de conhecimento, caracterizado como classificador, é gerado a partir dos dados de treinamento e avaliado por meio do conjunto de testes (GOLDSCHMIDT; PASSOS, 2005).

No processo de geração de um classificador, a árvore de decisão possui como entrada dados contidos em um conjunto de atributos e retorna uma decisão, que representa o valor de saída previsto, de acordo com a entrada (RUSSEL; NORVIG, 2004).

O princípio de uma árvore de decisão é o de dividir para conquistar[7], onde um problema complexo de se classificar é decomposto em problemas mais simples a cada nível da árvore, o que caracteriza a geração de nós descendentes (RODRIGUES, 2005).

A construção da árvore, cuja estrutura pode ser observada na Figura 7, é efetuada mediante uma abordagem recursiva de particionamento de dados. Para isso, o conjunto de dados de treinamento é dividido em duas ou mais partições utilizando-se restrições sobre o conjunto de valores de cada atributo. Este processo é repetido

[7] Sucessiva divisão do problema em vários subproblemas de menores dimensões, até que uma solução para cada um dos problemas mais simples possa ser encontrada (MORAES, 2001).

recursivamente, sendo que o mesmo é encerrado quando todos, ou pelo menos a maioria dos exemplos em cada uma das partições pertençam a uma classe (GOLDSCHMIDT; PASSOS, 2005).

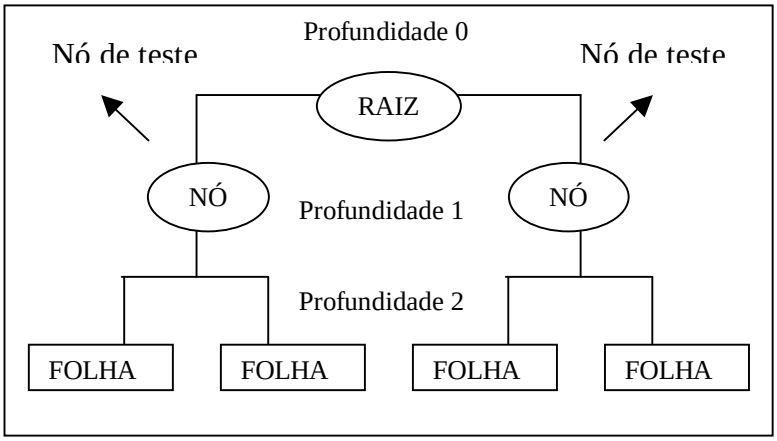

Figura 7. Estrutura de uma árvore de decisão
Fonte: ROSA, J. (2005)

A decisão de uma árvore é alcançada por meio da execução de vários testes, onde um nó corresponde ao valor de uma das propriedades, um nó-filho especifica o valor possível do teste e uma folha representa o valor que será retornado caso esta folha seja alcançada (RUSSEL; NORVIG, 2004).

Baseando-se no conjunto de dados de treinamento apresentado na Tabela 2, foi gerada a árvore de decisão da Figura 8.

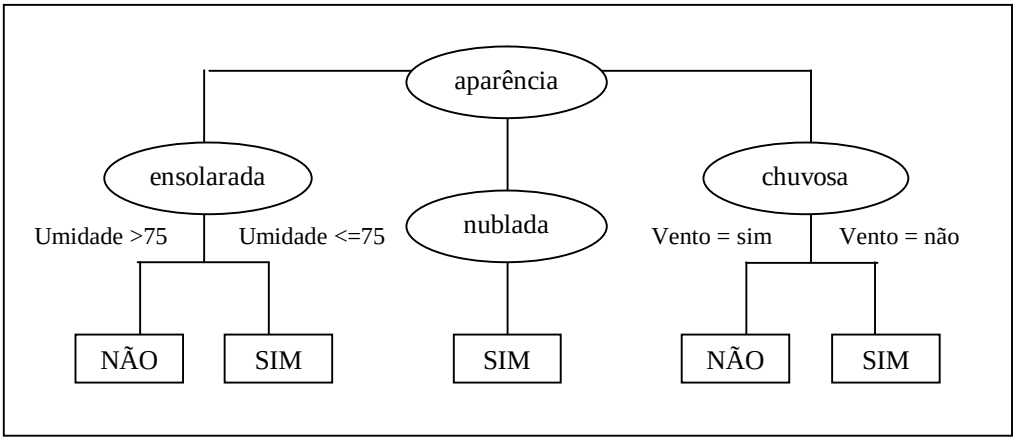

Figura 8. Árvore de decisão construída com base na Tabela 2
Fonte: GONÇALVES, E.C. (2006)

Esta árvore identifica uma recomendação quanto a prática de golfe baseando-se em seus dados preditivos, representados na árvore pelos nós. Uma ramificação (partindo de um nó interno) constitui em um resultado para o teste (Ex: aparência = nublada). Uma folha da árvore é um rótulo de classe (Ex: Jogar Golfe = Sim, Jogar Golfe = Não). Em cada nó da árvore deve ser escolhido um atributo, gerando a divisão dos dados do conjunto de treinamento, sendo que uma nova classificação é efetuada percorrendo a árvore da raiz para a folha (GONÇALVES, 2006).

Existem situações em que uma árvore construída pode possuir anomalias devido a ruídos presentes nos dados de treinamento. De modo a evitar este tipo de problema existe uma técnica chamada *Poda* que utiliza métodos estatísticos para remoção dos nós da árvore que não são relevantes, resultando em uma classificação mais rápida e na melhoria da habilidade de classificar corretamente os dados (HAN; KAMBER, 2001).

O processo efetuado pela técnica da *poda* pode ser observado na Figura 9, onde o t4, t5, t8, t9, t10 e o t11 são todos os descendentes de t2, similarmente, t4, t2, e o tl são antepassados do t9, porém o T3 não é um antepassado do t9 (BREIMAN et al, 1984):

a) um ramo da árvore T_t de T sendo que t Є T da raiz consiste em um nó t e em todos os descendentes de t em T. O ramo T_{t2} é ilustrada na Figura 9 (b);

b) podar um ramo T_t de uma árvore T consiste em eliminar de T todos os descendentes de t, isto é, eliminando todo o T_t exceto seu nó da raiz. Uma árvore podada desta forma é representada por: $T - T_t$ e pode ser verificada na Figura 9 (c).

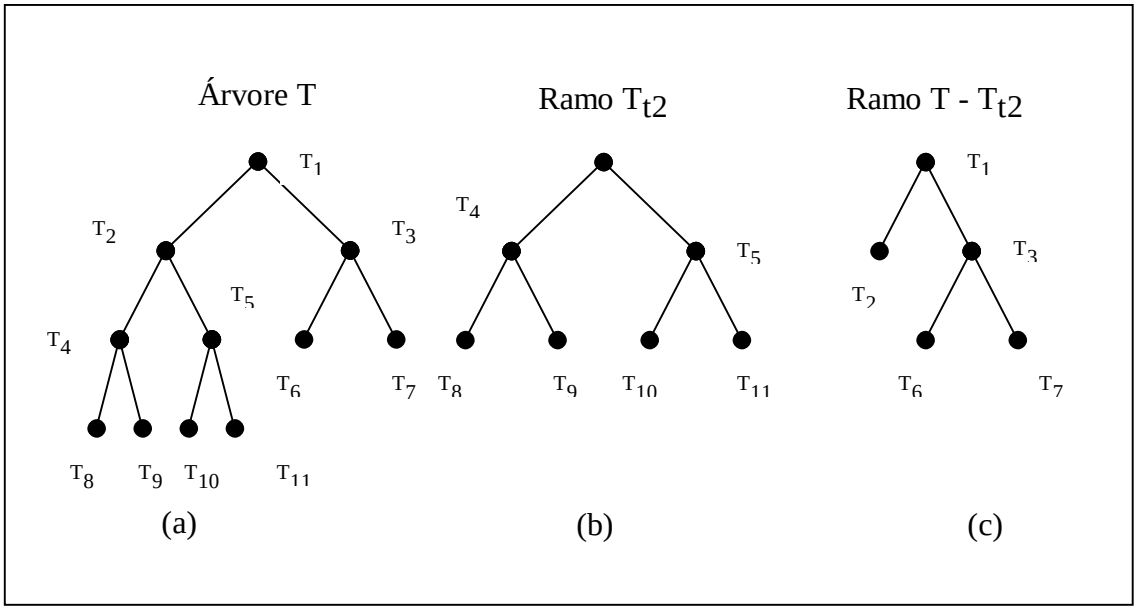

Figura 9. Demonstração da técnica da *poda*
Fonte: BREIMAN, L. et al (1984)

A aplicação da técnica da *poda* em árvores de decisão pode ser efetuada por meio de duas maneiras (KANTARDZIC, 2003):

a) **pré-poda**: decidir por não dividir um determinado nó utilizando um critério de parada, como por exemplo: se não houver nenhuma diferença significativa antes ou depois da divisão do nó, este deve ser transformado em uma folha. O nome pré-poda é devido a decisão ser efetuada antes da divisão do nó;

b) **pós-poda**: remover retrospectivamente da árvore os nós que não são relevantes para a solução do problema. Cada nó podado será transformado em folha. Na pós-poda a decisão é efetuada após a árvore estar construída.

Independente da técnica de *poda* utilizada, o conhecimento gerado por meio da árvore de decisão construída pode ser representado em forma de regras de classificação. A definição de uma regra deve ser efetuada percorrendo-se a árvore, partindo da raiz para a folha.

Uma regra de classificação é uma expressão da forma:

SE <antecedente> ENTÃO <conseqüente>

O antecedente deve ser formado por um ou mais atributos preditivos, enquanto o conseqüente é o objeto de saída. Uma regra do tipo SE/ENTÃO indica que a classe conseqüente pode ser determinada pelos atributos preditivos indicados no antecedente (GONÇALVES, 2005).

As regras de classificação facilitam a compreensão do modelo de conhecimento gerado, pois mesmo que as árvores sejam podadas em determinados casos elas podem ter um tamanho grande, tornando seu entendimento complexo (KANTARDZIC, 2003).

Baseando-se na Figura 8, pode-se gerar as seguintes regras de classificação:

1. SE (aparência = ensolarada) E (umidade <= 75) ENTÃO (jogar = sim)

2. SE (aparência = ensolarada) E (umidade > 75) ENTÃO (jogar = não)

3. SE (aparência = nublada) ENTÃO (jogar = sim)

4. SE (aparência = chuvosa) E (vento = sim) ENTÃO (jogar = não)

5. SE (aparência = chuvosa) E (vento = não) ENTÃO (jogar = sim)

A construção de uma árvore de decisão é realizada mediante a utilização de algoritmos que utilizam uma abordagem recursiva.

5.1 ALGORITMOS DE CLASSIFICAÇÃO PARA INDUÇÃO DE ÁRVORES DE DECISÃO

Segundo Han e Kamber (2001) os algoritmos de classificação podem ser avaliados e comparados utilizando-se os seguintes critérios:

a) **acurácia**: é a capacidade de classificar corretamente novos elementos;

b) **rapidez**: refere-se aos custos computacionais na construção e na utilização do modelo;

c) **escalabilidade**: possibilidade de construir classificadores na presença de grandes bases de dados;

d) **robustez**: é a capacidade de realizar decisões corretas na presença de dados com ruído ou incompletos;

e) **interpretabilidade**: refere-se ao nível de entendimento do modelo construído.

Os principais algoritmos para indução de árvores de decisão são CHAID , CART, ID3 e C4.5. Segundo Rodrigues (2005) esses algoritmos particionam, de maneira recursiva, os dados de treinamento em subconjuntos construindo uma árvore da raiz para as folhas.

5.1.1 ALGORITMO CHAID

O algoritmo *Chi Square Automatic Interaction Detection* (CHAID) foi um dos primeiros métodos a serem desenvolvidos. Criado em 1978 por Jay Magdson, constitui um método eficiente para o crescimento de uma árvore de decisão (RODRIGUES, 2005).

O propósito do método é dividir um conjunto de dados de tal forma que os subconjuntos sejam significativamente diferentes com relação a um determinado critério, que é a variável binária dependente ou de resposta. O CHAID suporta atributos categóricos e numéricos (variáveis explicativas[8]), porém a variável de resposta deve ser categórica. Os segmentos (categorias) derivados para cada variável são exclusivos e

[8] São os atributos envolvidos no processo de avaliação dos pontos de divisão da árvore (GOLDSCHMIDT; PASSOS, 2005).

exaustivos, ou seja, cada resposta está contida em somente uma categoria e todas as possibilidades de resposta encontradas na amostra para cada variável pertencem a alguma categoria resultante no CHAID (VASCONCELLOS, 2002).

Uma das vantagens do algoritmo CHAID é o fato de parar o crescimento da árvore antes de ocorrer o problema de *overfitting*[9], ou seja, não utiliza o método da poda. Porém, ele requer uma grande quantidade de dados, para que seja possível assegurar que a quantidade de observações dos nós-folhas seja significativa. Esse fato caracteriza uma desvantagem do algoritmo (RODRIGUES, 2005).

5.1.2 ALGORITMO ID3

O algoritmo *Induction of Decision Tree* (ID3) foi desenvolvido na Austrália nos anos 80 pelo professor Ross Quinlan, da Universidade de Sydney (SERRA, 2002).

Consiste em um método de árvore de decisão onde a sua construção é realizada de cima para baixo com o objetivo de escolher o melhor atributo para a divisão de cada nó.

O ID3 suporta apenas atributos categóricos, seja para variáveis explicativas ou de resposta, iniciando sua execução com todas as amostras de treinamento no nó da raiz da árvore, onde um atributo é selecionado para dividir estas amostras. Para cada valor do atributo uma filial é criada, e o subconjunto correspondente das amostras que têm o valor do atributo especificado pela filial é movido para o nó recentemente criado (nó filho).

[9] Ajuste excessivo ao conjunto de treinamento, ou seja, caso este conjunto não seja suficientemente representativo, o classificador pode ter um bom desempenho com dados de treinamento, porém não com os dados de teste (GOLDSCHMIDT; PASSOS, 2005).

O algoritmo é então aplicado recursivamente a cada nó filho até que todas as amostras estejam em uma classe e, cada caminho percorrido da raiz à folha na árvore de decisão representa uma regra de classificação (KANTARDZIC, 2003).

O método é recursivo, pois após ter encontrado um atributo de divisão para cada nó, iniciando pela raiz, o mesmo algoritmo é aplicado aos seus descendentes, até que critérios de parada sejam atingidos (HAN; KAMBER, 2005).

A escolha do atributo de partição no ID3 é efetuada mediante a utilização da medida do ganho de informação, um método estatístico utilizado para a construção de árvores de decisão que tem por objetivo reduzir a incerteza sobre o valor do objeto de saída e minimizar o número de testes necessários para classificar uma amostra de treinamento (HAN; KAMBER, 2005).

Uma vantagem do ID3 é a sua simplicidade, caracterizando o processo de construção da árvore relativamente simples, o que facilita a compreensão do seu funcionamento. A árvore produzida por ele, não pode ser reutilizada sem que esta tenha sido novamente construída, fato este caracterizado como uma desvantagem do algoritmo (RODRIGUES, 2005).

5.1.3 ALGORITMO C4.5

O algoritmo C4.5, desenvolvido em 1993 por Ross Quinlan , é uma extensão do ID3, ampliando o domínio da classificação dos atributos categóricos aos numéricos. (KANTARDZIC, 2003).

A diferença entre os dois algoritmos pode ser observada por meio de algumas características presentes no C4.5 (ESPÍNDOLA, 2004):

a) tratamento de atributos numéricos ou com valores ausentes;

b) uso do critério de ramificação chamado razão de ganho de informação;

c) poda da árvore após a sua indução.

O C4.5 envolve duas fases: construção e simplificação da árvore de decisão. A primeira caracteriza a avaliação dos pontos de divisão de cada nó interno, bem como a identificação do melhor ponto e a criação de partições a partir deste. A segunda utiliza a técnica da poda (GOLDSCHMIDT; PASSOS, 2005).

A poda da árvore é realizada para identificar e excluir ramos menos seguros. Esta técnica utilizada pelo algoritmo é definida como uma vantagem do C4.5 se comparado ao ID3, pois resulta em uma classificação mais rápida e na melhoria da habilidade de classificar amostras de teste corretamente (MOTTA, 2004).

Outra vantagem desse algoritmo é a capacidade de *validação cruzada*[10] com dois ou mais grupos, melhorando assim a estimativa do erro cometido pelo classificador (RODRIGUES, 2005).

Em relação as desvantagens do C4.5 Quinlan (1993) considera que o fato deste algoritmo trabalhar com dados numéricos não é simples, pois estes são estados mais de uma vez no mesmo percurso o que pode dificultar o entendimento da árvore.

5.1.4 ALGORITMO CART

O *Classification And Regression Trees* (CART) foi desenvolvido em 1984 por quatro estatísticos: Leo Breiman, Jerome Friedman, Richard Oslen e Charles Stone (BREIMAN et al, 1984).

Uma das suas características principais é a grande capacidade de pesquisa de relações entre os dados, mesmo quando elas não são evidentes, bem como a produção de resultados sob a forma de árvores de decisão (FONSECA,1994).

[10] Do inglês c*ross-validation.* Este método divide os dados do treinamento em um subconjunto para construir a árvore e estima então a taxa de não classificação no subconjunto restante (HAND et al, 2001).

O CART envolve as fases de construção e simplificação da árvore de decisão, escolhendo a melhor variável para divisão dos dados em dois nós, onde o procedimento de divisão é aplicado recursivamente aos dados em cada um dos nós-filhos, e assim por diante (HAND et al, 2001).

As principais vantagens do algoritmo CART são (RODRIGUES, 2005):

a) permite a utilização de diferentes tipos de variáveis independentes (discretas, contínuas e nominais);
b) não necessita efetuar transformações das variáveis iniciais independentes, como logaritmação ou normalização, pois o método pode ser aplicado a diferentes tipos de dados;
c) uma mesma variável pode ser utilizada em diferentes estágios o que permite que sejam observados os efeitos que esta produz sobre as outras;
d) não necessita satisfazer nenhuma condição de aplicabilidade do modelo;
e) utiliza a técnica da poda da árvore, caracterizada pelo corte dos nós não relevantes para a resolução do problema, produzindo árvores menores.

A Tabela 3 apresenta um comparativo dos algoritmos para indução das árvores de decisão quanto aos tipos de variáveis envolvidas no processo de classificação.

Tabela 4. Algoritmos para indução de árvores de decisão

Algoritmos	**Variável Explicativa**	**Variável de Resposta**	**Simplificação da Árvore de Decisão**
CHAID	Categórica e numérica	Categórica	Não
CART	Categórica, numérica e valores ausentes	Categórica e numérica	Sim
ID3	Categórica	Categórica	Não
C4.5	Categórica, numérica e valores ausentes	Categórica	Sim

Como o objetivo geral desta pesquisa é a implementação do algoritmo CART para indução de árvores de decisão no módulo de classificação da *Shell* Orion, o capítulo a seguir dedica-se exclusivamente ao entendimento desse algoritmo.

6 O ALGORITMO CART PARA INDUÇÃO DE ÁRVORES DE DECISÃO

Os conceitos e o funcionamento do algoritmo CART apresentados neste capítulo são exclusivamente extraídos do livro *Classification and Regression Trees*, apresentado em 1984 pelos desenvolvedores do algoritmo, os quatro estatísticos: Leo Breiman, Jerome Friedman, Richard Oslen e Charles Stone.

O objetivo dos autores foi o desenvolvimento de um método para identificar o elevado risco de morte dos pacientes que chegavam ao Centro Médico de San Diego, na Califórnia. Conforme os autores, quando um paciente entrava no centro médico apresentando sintomas de ataque cardíaco, eram medidas 19 variáveis durante as primeiras 24 horas. Estas variáveis incluíam pressão sanguínea e idade, bem como os sintomas considerados indicadores importantes para diagnosticar a condição do paciente. A Figura 10 mostra a árvore de classificação e as regras geradas a partir do estudo realizado.

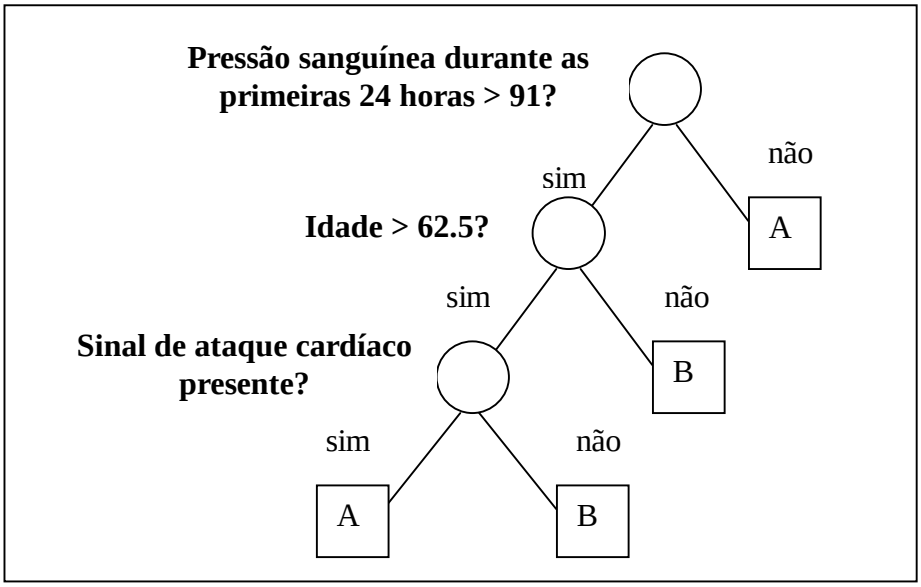

Figura 10. Árvore de classificação
Fonte: BREIMAN, L. et al (1984)

As regras identificadas na árvore classificaram pacientes em: A (risco alto) ou B (risco baixo), de acordo com as respostas, *sim* ou *não*, para as três perguntas selecionadas.

A metodologia utilizada para construir tais regras é o principal assunto discutido no livro, onde são apresentadas duas estruturas de árvore: *Tree Structured Classifiers* e *Tree Structured Regression.* Nelas as árvores originadas são sempre binárias[11], sendo que a diferença entre ambas é a representação do conteúdo de suas folhas que, no caso da classificação é a classe, enquanto na regressão, é o valor da previsão.

O algoritmo CART por meio de sua representação em forma de árvores permite a obtenção de resultados, em geral, bastante superiores aos obtidos pelas técnicas estatísticas clássicas, sendo superado apenas num restrito número de casos, e quando superado, a diferença nos resultados é mínima.

No que se refere ao tratamento de variáveis dependentes, o CART suporta categóricas (classificação) ou numéricas (regressão), e o procedimento para o crescimento da árvore pode ser entendido como um método de divisão recursiva e binária. Caracteriza-se por binária porque os dados são subdivididos sempre em duas partições (um nó pai é subdividido em dois nós filhos), e recursiva, pois o processo é repetido para cada subdivisão dos dados (cada nó filho posteriormente será considerado como um pai). Esse processo é realizado até que não seja mais possível efetuar divisões ou esta for limitada por algum critério ajustado pelo analista. O resultado deste algoritmo é sempre uma árvore binária que pode ser percorrida da raiz às folhas respondendo apenas a questões simples do tipo *sim* ou *não*.

[11] São árvores que ou são nulas ou são constituídas por um nó raiz e duas subárvores binárias, a subárvore esquerda e a subárvore direita. Por sua vez cada uma destas subárvores ou são nulas ou constituídas por um nó raiz e duas subárvores binárias, a subárvore esquerda e a subárvore direita (FONSECA, 1994).

De modo a demonstrar os processos envolvidos no crescimento da árvore de decisão por meio do CART, o conteúdo apresentado nas próximas seções está relacionado diretamente as árvores estruturadas por classificadores, pois o objetivo da pesquisa é a implementação deste algoritmo na tarefa de classificação da *Shell* Orion.

Na execução da tarefa de classificação por meio do algoritmo CART existem quatro elementos envolvidos no crescimento de uma árvore:

a) um conjunto *Q* de perguntas binárias no formato $X \in A?$, $A \subset \chi$, onde χ representa o conjunto de todas as medidas possíveis para a divisão do nó;

b) um bom critério de divisão $\phi(s, t)$ que possa ser avaliado para toda divisão *s* de qualquer nó t;

c) uma regra de parada;

d) uma regra para atribuir cada nó terminal a uma classe.

Cada um desses elementos envolvidos no crescimento de uma árvore de decisão por meio do CART será conceituado e exemplificado, visando uma melhor compreensão dos mesmos.

6.1 CONJUNTO DE PERGUNTAS BINÁRIAS

O conjunto *Q* de perguntas binárias é utilizado para efetuar a divisão de cada nó *t*. Nos casos em que a resposta for *sim*, segue-se para o nó esquerdo t_L e nos que a resposta for *não*, para o nó direito t_R, conforme mostra a Figura 11.

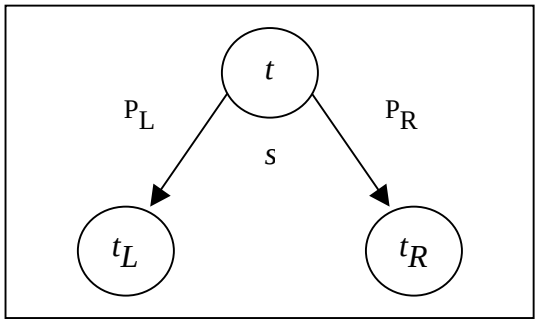

Figura 11. Divisão da árvore
Fonte: BREIMAN, L. et al (1984)

Em todo o nó *t*, supõe-se que existe uma divisão *s* do nó que o divide em t_L e em t_R tal que, uma probabilidade p_L para o nó *t* segue para t_L e uma probabilidade p_R segue para t_R.

A divisão de cada nó da árvore é realizada mediante a procura do melhor ponto de divisão. Todas as variáveis (também denominadas de atributos) que pertencem ao conjunto de dados são avaliadas e, para cada uma é encontrado o seu melhor ponto. Depois de realizado esse processo, a variável escolhida para a divisão do nó é aquela que possuir o melhor ponto, sendo que para isso o algoritmo CART dispõe de alguns critérios.

6.2 CRITÉRIOS DE DIVISÃO

O CART constrói uma árvore de decisão dividindo o conjunto de dados de treinamento em cada nó de acordo com a função de um único atributo do conjunto. A primeira tarefa é descobrir qual dos atributos realiza a melhor divisão, isto é, qual consegue particionar o conjunto em subconjuntos homogêneos. Na escolha do melhor divisor, cada atributo é levado em consideração, sendo testado como possível divisor. Aquele que conseguir dividir os dados em dois conjuntos, com a maior homogeneidade interna e o maior grau de diferença entre conjuntos, é o escolhido.

No que se refere ao número de divisões possíveis, consideram-se os seguintes critérios de acordo com o tipo de variável:

a) se a variável explicativa possui k valores ordenados, ou seja, os atributos são numéricos, são consideradas k –1 divisões;

b) caso a variável explicativa seja categórica, são consideradas 2^{k-1}-1 divisões.

A fim de tornar compreensível o funcionamento das equações que serão apresentadas posteriormente, faz-se necessário compreender o cálculo de probabilidade. Em um nó *t*, o total de casos é representado por *N*(*t*) e o número de casos que pertencem a classe *j* do nó *t* por $N_j(t)$. A proporção dos casos da classe *j* que pertencem ao nó *t* é $N_j(t)$ / *N*(*t*). Dado um conjunto de prioridades, $\pi(j)$ é interpretado como a probabilidade da classe *j* estar presente na árvore. Com isso, tem-se a equação (1) como sendo a estimativa de resubstituição para a probabilidade dos casos das classes *j* no nó *t*:

$$p(i, j) = \pi(j) N_j(t) / N_j \tag{1}$$

Enquanto, o cálculo dessa estimativa para todas as classes *j* é dado por:

$$p(j) = \sum_j p(i, j) \tag{2}$$

Com o resultado das equações (1) e (2), pode-se obter a estimativa de resubstituição da probabilidade dos casos de uma determinada classe *j* estarem presentes no nó *t*, dado por :

$$p(j / i) = p(j, i) / p(j) \tag{3}$$

que satisfaz:

$$\sum_j p(j / t) = 1 \tag{4}$$

Quando $\pi(j) = \{N_j / N\}$, então $p(j / t) = N_j(t) / N_j$, assim $\{p(j / t)\}$ são as proporções de casos da classe *j* no nó t.

O cálculo da probabilidade é utilizado, por exemplo, na etapa de seleção do atributo que realizará a divisão dos dados na árvore, onde se procura diminuir a impureza dos nós, sendo que essa impureza é definida como segue:

a) as probabilidades das J classes de um nó *t* são definidas por *p*(*j*/*t*), onde *j*=1,...J, é dada pela proporção de cada classe nos casos abrangidos por esse nó, tal que:

$$\sum_{i=1}^{J} p(j/t) = 1 \tag{5}$$

b) seja *i(s,t)* a medida de impureza de um nó *t* definida por uma função ϕ que obedeça os requisitos:

$$\phi\left\{\frac{1}{J},...\frac{1}{J}\right\} = valormáximo \tag{6}$$

$$\phi(1,0,...,0) = 0 \quad \phi(0,1,...,0) = 0 \quad \phi(0,0,...,1) = 0 \tag{7}$$

De acordo com as equações (6) e (7) a impureza de um nó é máxima quando todas as classes possuem prioridade igual e mínima quando existe apenas uma classe. Dada uma função de impureza ϕ, define-se a medida de impureza i(s,t)de todo nó *t* com:

$$i(s,t) = \phi(p(1/t), p(2/t),..., p(J/t)) \tag{8}$$

Dessa maneira, pode-se definir o ganho de uma divisão como sendo a diferença entre a impureza do nó pai e a soma desta para os nós filhos. Se a divisão *s* separar o nó *t* em dois subconjuntos t_L e t_R com as proporções p_L e p_R, o ganho da impureza será dado por:

$$\Delta i(s,t) = i(t) - p_L i(t_L) - p_R i(t_R) \tag{9}$$

Essa diferença é apresentada na árvore como *improvement*[12], sendo que a

[12] Representa a melhoria do nó, ou seja, o melhor ponto para divisão de um nó (BREIMAN et al, 1984).

variável explicativa escolhida para a divisão do nó é a que corresponde ao maior *improvement.*

O algoritmo CART considera três critérios possíveis para selecionar a melhor divisão dos dados: Entropia, Critério de Twoing e Critério de Gini.

6.2.1 Entropia

A entropia, utilizada no algoritmo CART para problemas que apresentam apenas duas classes, é uma medida empregada para calcular a homogeneidade de um conjunto de exemplos, medindo a quantidade de informação necessária para codificar a situação presente no nó. Seu cálculo é dado pela equação:

$$i(s,t) = -\sum_{i=0}^{J} p_i \log_2 p_i \tag{10}$$

Onde: J - corresponde ao número de classes

p_i - probabilidade do nó i

Nos casos que apresentam mais de duas classes, o algoritmo CART utiliza para realizar a divisão dos nós os critérios de Twoing e Gini.

6.2.2 Critério de Twoing

Um critério de divisão alternativo disponível para o CART é o de Twoing, onde é medida a diferença da probabilidade de uma classe pertencer mais ao descendente esquerdo do que ao direito do nó. A função para o critério de Twoing é dada por:

$$i(s,t) = \frac{p_L p_R}{4}\left[\sum_j \left| p(j/t_L) - p(j/t_R) \right|\right]^2 \tag{11}$$

Onde: p_L - probabilidade do nó descendente esquerdo

p_R - probabilidade do nó descendente direito

$p(j/t_L)$ - probabilidade da classe j nó descendente esquerdo

$p(j/t_R)$ - probabilidade da classe j nó descendente direito

O critério de Twoing consiste em separar a variável de resposta em duas grandes classes, e deste modo encontrar a melhor divisão na variável explicativa com base na divisão efetuada em duas classes. O ponto de separação *s* que maximiza esse critério será o valor a ser utilizado na partição do nó. As duas grandes classes, C_1 e C_2, são definidas por:

$$C_1(s) = \{j : p(j/t_L) \geq p(j/t_R)\}$$

$$C_2(s) = C - C_1$$

Onde: C – número de categorias da variável de resposta

6.2.3 Critério de Gini

O critério de Gini mede a heterogeneidade dos dados e pode ser entendido como a probabilidade condicional de erro dado um conjunto de treinamento selecionado de forma aleatória que é dividido em um nó *t*, onde cada classe *j* tem uma probabilidade p(j/t). Segundo o critério de Gini, a impureza de um nó é dada por:

$$i(s,t) = 1 - \sum_j p^2(j/t) \qquad (12)$$

Onde: *p (j / t)* – probabilidade *a priori* da classe j se formar no nó t.

Quando o valor de Gini é igual a zero, o nó é puro. Contudo, quando se aproxima do valor um, o nó é impuro, pois aumenta o número de classes uniformemente

distribuídas neste nó. Com a utilização do critério de Gini, a tendência é isolar em um nó os registros que representam a classe mais freqüente.

Nos testes efetuados pelos autores do algoritmo CART, no que se refere aos critérios de Gini e Twoing, os resultados mostraram que as divisões efetuadas pelo Twoing tendem a produzir nós descendentes com um tamanho mais adequado do que as divisões encontradas pelo Gini. Existem poucas diferenças entre os resultados apresentados pelos dois critérios. No entanto, se comparar os dois aplicando-se em vários conjuntos de dados, onde os resultados diferem, as divisões de Gini são as melhores. Um é claramente superior a outro no sentido de produzir os nós descendentes puros, onde o Twoing seleciona o divisor mais impuro. Por estas razões, geralmente é preferível o uso do critério de Gini.

Mediante essa definição, optou-se pela utilização do critério de Gini na implementação do algoritmo CART na *Shell* Orion *Data Mining Engine* e, devido isso será demonstrado com mais detalhes o funcionamento desse critério.

6.2.3.1 Funcionamento do Critério de Gini

De modo a facilitar o entendimento da equação (9), considera-se uma variável categórica *Y* para *j* = 1,…, J. Cada nó na árvore é identificado por um número único representado pelo símbolo t, e possuirá uma distribuição estimada da probabilidade para os valores da variável de resposta. Em um problema com três classes, por exemplo, sendo J = 3, as probabilidades são escritas por $p(1/t)$, $p(2/t)$ e $p(3/t)$. Assim, se um nó t possuísse um total de 150 casos, onde para uma variável explicativa numérica esses casos fossem distribuídos igualmente nas três classes, ou

seja, 50 para cada uma (50/50/50), as probabilidades das classes, dadas pela equação (3), seriam:

$$p\ (1\ /\ t) = 50/150 = 1/3 = 0.3333;$$

$$p\ (2\ /\ t) = 50/150 = 1/3 = 0.3333;$$

$$p\ (3\ /\ t) = 50/150 = 1/3 = 0.3333.$$

Dada uma distribuição da probabilidade para cada classe, pode-se calcular a impureza deste nó. Neste caso o resultado do critério de Gini, por meio da equação (12), consiste em:

$$i(s,t) = 1 - ((1/3)^2 + (1/3)^2 + (1/3)^2)$$

$$i(s,t) = 1 - (0{,}111 + 0{,}111 + 0{,}111)$$

$$i(s,t) = 1 - 0.333$$

$$i(s,t) = 0.667$$

Neste exemplo, considerando três classes, o índice de Gini é $i(s,t) = 0.667$. Se um nó possuir amostras exclusivamente de uma única classe, uma das probabilidades seria 1, as demais probabilidades seriam zero e o índice da heterogeneidade $i(s,t)$ seria zero. Este conceito faz sentido, pois um nó que contém somente uma classe não tem nenhuma heterogeneidade.

Supondo-se que o cálculo apresentado pertence ao nó pai da árvore, e que este foi subdivido em dois nós filhos, sendo que o nó esquerdo contém 50 casos (50/0/0) e o nó direito 100 casos (0/50/50), o cálculo de Gini para os nós filhos é dado por:

$$i(t_L) = 1 - ((1/1)^2 + (0)^2 + (0)^2)$$

$$i(t_L) = 1 - 1$$

$$i(t_L) = 0$$

$$i(t_R) = 1 - ((0)^2 + (1/2)^2 + (1/2)^2)$$

$$i(t_R) = 1 – (0,25 + 0,25)$$

$$i(t_R) = 0,5$$

Tendo-se o resultado de Gini para os nós pai e filhos, é possível encontrar o ganho da divisão por meio da equação (9):

$$\Delta i(s,t) = 0.667 – ((50/150) * 0 + (100/150) *0.5)$$

$$\Delta i(s,t) = 0.667 – 0.3333$$

$$\Delta i(s,t) = 0.3333$$

Esse procedimento é realizado para todas as variáveis explicativas envolvidas no problema, sendo que a escolhida para a divisão do nó será a que possuir o maior índice de Gini, ou seja, o resultado mais elevado da equação (9) dentre todas as variáveis explicativas.

Depois de encontrado o melhor ponto de divisão dos dados e com isso efetuada a divisão do nó, caso não exista mais ganho em dividi-lo novamente, faz-se a associação de uma determinada classe à folha gerada.

6.3 ASSOCIAR UMA CLASSE A FOLHA

Um nó da árvore é atribuído como sendo um não terminal (folha) mediante a especificação de regras de parada, como:

a) o número de casos do nó é menor que o estabelecido para ser o mínimo.Ex: determinar que um nó deverá conter no mínimo dois casos;
b) o índice de Gini do atributo escolhido como ponto de divisão é igual a zero;
c) a soma das probabilidades de um nó é igual a zero.

Após atribuir um nó como folha é necessário determinar qual classe deverá ficar associada a ela. Neste caso, pode-se escolher um dos critérios existentes: determinação baseada na noção de custos ou atribuição da classe mais provável.

A determinação baseada na noção de custos utiliza os conceitos de matriz de custo[13] e matriz de confusão[14], cujo objetivo será a minimização dos custos provenientes da adoção de uma dada classe. Seu cálculo é dado por:

$$Custo(k) = \sum_{i=1}^{J} p_i C_{i,k} \tag{13}$$

Onde: J – número de classes

p_i – probabilidade da classe *i*

$C_{i,k}$ – valor da linha *i*, coluna *k* da matriz de custos

O critério utilizado para associar uma determinada classe à folha no algoritmo CART, implementado na *Shell* de *Data Mining* Orion, é a atribuição da classe mais provável, exemplificado na Seção 6.3.1.

6.3.1 Atribuição da Classe mais Provável

Este critério visa a minimização do erro de classificação, cuja classe a ser atribuída à folha será a mais provável dentro dos exemplos que se encontram nessa folha. Seu cálculo é dado por:

$$\max_j \left(p_j \right) = \max_j \frac{N_j}{N} \tag{14}$$

Onde: j – 1....*J*, sendo *J* o número de classes

[13] Possui o custo de cada erro possível, tendo N^2 elementos onde N corresponde ao número de classes (FONSECA, 1994).

[14] Demonstra os resultados do teste do classificador em relação ao número de exemplos da classe i aos quais foi atribuída a classe j, com ambos variando entre 1 e N (FONSECA, 1994).

N – número total de exemplos na folha

N_j – número de exemplos da classe j na folha

Supõe-se que um nó *t* tornou-se folha contendo 53 casos com a seguinte distribuição entre três classes: classe 1 = 50, classe 2 = 1 e classe 3 = 2. Aplicando-se a equação (14) para encontrar a classe que será atribuída a essa folha, tem-se:

Classe 1 = 50/53 = 0,9434

Classe 2 = 1/53 = 0,0189

Classe 3 = 2/53 = 0,0377

Observando-se os resultados demonstrados anteriormente, verifica-se que a classe 1 foi a que apresentou a máxima probabilidade, portanto, deve-se atribuí-la ao nó *t*.

Essa verificação deverá ser efetuada sempre que um nó tornar-se folha, ou seja, durante todo o crescimento da árvore. Após o término da construção da árvore, é adotada uma técnica chamada *poda*, cujo objetivo é a remoção ou corte dos ramos e sub-árvores que não são relevantes para a resolução do problema.

6.4 PODA

O desenvolvimento de uma árvore é realizado até o ponto em que não existam mais atributos a serem testados ou somente uma classe esteja presente no nó. Contudo, quando o objetivo é apenas a diminuição do erro estimativo, a tendência é resultar em uma árvore extensa, caso a quantidade de dados envolvidos no problema seja grande.

Uma árvore extensa pode tornar-se pouco eficiente pelo fato de considerar casos não tão relevantes contidos no conjunto de treinamento, prejudicando nos

resultados. Porém, se a mesma for simples está sujeita a desconsiderar informações importantes para a solução do problema.

O processo para obter a árvore com tamanho adequado consiste em três estágios:

a) crescer a árvore baseada em um critério de divisão (conforme apresentado na Seção 6.2);
b) podar esta árvore para produzir uma seqüência de sub-árvores menores;
c) obter estimativas de custo de cada árvore.

De modo a evitar o tamanho excessivo das árvores de decisão desenvolvidas por meio do algoritmo CART, após o término de seu crescimento aplica-se a técnica da poda por minimização do custo de complexidade.

6.4.1 Poda por Minimização do Custo-Complexidade

A utilização da técnica da poda por minimização do custo-complexidade em uma árvore de decisão consiste na execução de três passos: criar uma árvore inicial tão grande quanto necessário, de modo a resultar em um erro estimado baixo; podá-la para obter um conjunto de árvores de dimensões menores; utilizar uma estimação de erro de modo a selecionar a melhor de todas as árvores criadas.

6.4.1.1 Criação da Árvore Inicial

Inicialmente, deve-se construir uma árvore de classificação $T_{\max}$ tão grande quanto necessário para posteriormente realizar a sua redução. A melhor maneira de

crescer uma árvore inicial é continuar dividindo os nós até que cada nó terminal contenha apenas um caso.

6.4.1.2 Criação de um Conjunto de Árvores Menores a partir da Árvore Inicial

Uma árvore inicial, mesmo com pequena dimensão, pode gerar várias sub-árvores a partir da sua redução. De modo a tornar esse processo computacionalmente suportável, a poda implica na adoção de um critério de comparação entre árvores de igual dimensão, utilizando o número de nós terminais (folha). A maneira correta de efetuar essa comparação é por meio do cálculo do erro estimado por resubstituição.

O erro estimado por resubstituição de um nó *t* é dado por:

$$r(t) = 1 - \max_j p(j/t) \tag{15}$$

Sendo que o erro total desse nó será, portanto:

$$R(t) = r(t)p(t) \tag{16}$$

Considerando $\tilde{\mathrm{T}}$ como sendo o número de nós terminais de uma árvore T, o erro estimado por substituição total pode ser calculado como sendo:

$$R(\mathrm{T}) = \sum_{t \in \tilde{\mathrm{T}}} R(t) \tag{17}$$

Assim, consegue-se definir o custo-complexidade de uma árvore T por meio da equação:

$$R_\alpha(\mathrm{T}) = R(\mathrm{T}) + \alpha\left|\tilde{\mathrm{T}}\right| \tag{18}$$

Onde: $R(\mathrm{T})$ – erro estimado por resubstituição da árvore T

$\left|\tilde{\mathrm{T}}\right|$ – número de nós terminais da árvore

α – constante ≥ 0 chamada de parâmetro de complexidade

Dessa maneira, $R_\alpha(\mathrm{T})$ é uma combinação linear entre o custo da árvore e sua complexidade, onde o objetivo é encontrar uma árvore que minimize o valor de R_α fazendo variar o valor de α iniciando por zero até que esteja na presença de uma árvore com apenas um nó.

De modo a explicar o processo de geração da seqüência de árvores, considera-se t_L e t_R como sendo os nós esquerdo e direito resultantes da divisão do nó t. Devendo-se verificar para qualquer árvore:

$$R(t) \geq R(t_L) + R(t_R) \tag{19}$$

Para encontrar a árvore T_1 se deve percorrer todos os nós de T_{max} e, caso $R(t) = R(t_L) + R(t_R)$, elimina-se os nós t_L e t_R. Este procedimento destina-se a eliminar todos os nós inúteis de forma a criar uma primeira árvore a partir da qual inicia-se a verdadeira seqüência de redução, explicada a seguir.

Seja $\{t\}$ o nó que resulta da eliminação do ramo T_t, para todos os ramos da árvore dado que, $R_\alpha(\mathrm{T}_t) = R(\mathrm{T}_t) + \alpha\left|\tilde{\mathrm{T}}_t\right|$ e que $R_\alpha(\{t\}) = R(\{t\}) + \alpha$, deve-se procurar o valor de α para o qual tem-se $R_\alpha(T_t) = R_\alpha(\{t\})$. Deste modo, obtém-se a equação $R(\mathrm{T}_t) + \alpha\left|\tilde{\mathrm{T}}_t\right| = R(\{t\}) + \alpha$, que resolvendo-a por ordem de α, resulta em:

$$\alpha = \frac{R(t) - R(\mathrm{T}_t)}{\left|\tilde{\mathrm{T}}_t\right| - 1} \tag{20}$$

Contudo, deve-se então escolher o ramo que contém um valor de α mais baixo. Este é o valor para o qual será efetuada a primeira redução na árvore T_1 para o que se reduz de T_t. O processo é repetido baseando-se como ponto de partida o valor de

α encontrado nesta interação e a árvore resultante da inicial depois de retirado T_t. Tal processo é realizado até que se encontre uma árvore contendo apenas um nó, onde se tem uma seqüência de árvores e respectivos valores de α tais que:

$$T_1 \succ T_2 \succ T_3 \succ ... \succ \{t\} \qquad \alpha_1 < \alpha_2 < \alpha_3 < ... < \alpha_t$$

O funcionamento da poda por parâmetro de complexidade pode ser observado na Seção 8.2.4. Depois de aplicada a poda e gerada uma seqüência de árvores, deve-se escolher qual delas será o classificador final.

6.4.1.3 Escolha da Melhor Árvore

A escolha da árvore final, dentre todas as geradas na etapa da poda, requer uma estimativa do erro de cada uma delas de modo que possa ser escolhida aquela que apresentar melhor capacidade de classificação. Essa estimativa é encontrada por meio de uma técnica denominada *Cross-Validation* ou Validação Cruzada.

Inicialmente é gerada uma árvore utilizando todo o conjunto de treinamento inicial e posteriormente, calculada a sua seqüência de reduções. O próximo passo é dividir esse conjunto aleatoriamente em V (normalmente utilizando 10) subconjuntos, contendo se possível, o mesmo número de casos em cada um. Dessa maneira serão construídas V árvores diferentes, utilizando para tal (V-1)/V dos casos, sendo que o restante 1/V é utilizado para avaliar o erro.

Seja $T^{(v)}(\alpha)$, onde *v*=1,...,V a árvore que possui o menor custo-complexidade para cada α. Utilizando o subconjunto do conjunto de treinamento inicial que não foi empregado no seu desenvolvimento, define-se $N_{ij}^{(v)}$ como o número de casos da classe *j* que foram classificados incorretamente pela árvore $T^{(v)}(\alpha)$. Desse

modo, o número total de casos da classe *j* mal classificados por todas as árvores de complexidade α é dado por:

$$\mathrm{N}_{i,j} = \sum_{i,j} \mathrm{N}_{ij}^{(v)} \tag{21}$$

Conseqüentemente, a probabilidade do erro por validação cruzada será:

$$R^{(CV)}\left(\mathrm{T}(\alpha)\right) = \frac{1}{\mathrm{N}} \sum_{i \# j} \mathrm{N}_{i,j} \tag{22}$$

Resultando-se assim, no cálculo do erro para uma árvore de complexidade α. Nota-se que as árvores construídas com base no conjunto de treinamento inicial são idênticas a $\mathrm{T_k}$ para valores de α tal que $\alpha_k \leq \alpha < \alpha_{k+1}$. Por fim,

$$\alpha_k^{'} = \sqrt{\alpha_k \alpha_{k+1}} \tag{23}$$

onde $\alpha_k^{'}$ é o centro geométrico entre α_k e α_{k+1}. O erro por validação cruzada da *k*-enésima árvore da seqüência de reduções efetuada sobre a árvore produzida e baseada no conjunto de treinamento total será dada por:

$$R^{(CV)}\left(\mathrm{T}_k\right) = R^{CV}\left(\mathrm{T}\left(\alpha_k^{'}\right)\right) \tag{24}$$

Portanto, será escolhida a árvore que permita minimizar o valor do erro calculado podendo-se utilizar esse valor como estimativa de erro para o classificador final. O funcionamento da validação cruzada pode ser observado de forma mais explicativa na Seção 8.2.4.

Finalizada a explicação a respeito das características e do formalismo matemático acerca do algoritmo CART, no próximo capítulo são apresentados alguns exemplos da sua aplicabilidade em diferentes pesquisas científicas nas mais diversas áreas do conhecimento.

7 ALGUNS EXEMPLOS DE USO DO ALGORITMO DE CLASSIFICAÇÃO CART EM *DATA MINING*

Existem várias aplicações da tarefa de classificação, no processo de *data mining*, utilizando o método de indução de árvores de decisão por meio do algoritmo CART. Algumas destas aplicações podem ser observadas conforme segue.

7.1 USO DE ÁRVORES DE DECISÃO NA CLASSIFICAÇÃO DE IMAGENS DIGITAIS

Este trabalho foi publicado no XI Simpósio Brasileiro de Sensoriamento Remoto em Belo Horizonte em 2003 e desenvolvido por Robin Thomas Clarke e Hélio Radke Bittencourt, sendo que o primeiro pertence ao Centro Estadual de Pesquisas em Sensoriamento Remoto e Meteorologia (CEPSRM) localizado na Universidade Federal do Rio Grande do Sul (UFRGS) e, o segundo, ao Laboratório de Estatística da Universidade Luterana do Brasil (ULBRA).

Nesta pesquisa, o algoritmo CART foi utilizado na classificação de um segmento de uma imagem obtida por meio do satélite Landsat-TM utilizando seis bandas espectrais. Foram consideradas três classes para a classificação da imagem: água, culturas e vegetação. A árvore de decisão, utilizando o algoritmo CART, foi construída por meio do software *Gesntat 6.1*, onde uma amostra de treinamento de 1084 *pixels* foi processada (CLARKE; BITTENCOURT, 2003).

O resultado obtido, segundo os autores foi muito bom, sendo o próprio algoritmo quem elegeu as bandas espectrais com maior poder discriminatório, considerada uma informação muito importante para o pesquisador. Segundo eles, apesar

do CART ter sido desenvolvido há vários anos, existe um consenso de sua importância e valor científico na atualidade.

7.2 AVALIAÇÃO DO DESEMPENHO LOGÍSTICO DA CADEIA BRASILEIRA DE SUPRIMENTOS DE REFRIGERANTES

Este trabalho foi desenvolvido como Dissertação de Mestrado do Curso de Engenharia de Produção da Universidade Federal de Minas Gerais (UFMG) em 2003 (QUINTÃO, 2003).

A pesquisa teve como objetivo a avaliação do desempenho logístico de quatro elos da cadeia brasileira de suprimentos para refrigerantes: fornecedores de embalagem para refrigerantes, indústria de refrigerantes, atacado e supermercado. Além desses, foram avaliados de forma indireta mais dois elos: fornecedor de ingredientes para a fabricação e de matéria-prima para a produção das embalagens de refrigerante (QUINTÃO, 2003).

De modo a obter dados para a avaliação, foi realizada uma pesquisa entre 54 empresas onde as respostas foram analisadas tomando como base o conceito do gerenciamento da cadeia de suprimentos e do efeito de chicoteamento. Segundo Quintão (2003), o efeito de chicoteamento refere-se à propagação dos erros e distorções da demanda do consumidor final ao longo da cadeia de suprimentos. Esse efeito foi avaliado na pesquisa por meio da tarefa de classificação utilizando o algoritmo CART para indução de árvores de decisão.

A utilização do CART teve como objetivo verificar o relacionamento das variáveis descritas pela pesquisa: estratégias de remediação do efeito de chicoteamento

utilizadas pelas empresas e avaliações de desempenho logístico da cadeia de suprimentos. (QUINTÃO, 2005).

7.3 INDUÇÃO DE ÁRVORES DE DECISÃO: HISTCLASS – PROPOSTA DE UM ALGORITMO NÃO-PARAMÉTRICO

Esta Dissertação de Mestrado, desenvolvida em 1994 na Universidade de Nova Lisboa, em Lisboa. Trata-se da implementação do algoritmo HistClass para indução de árvores de decisão. Após o desenvolvimento do mesmo, foram realizados testes de comparação com os algoritmos ID3, CART e C4.5 (FONSECA, 1994).

Diante das comparações efetuadas nesta tese, um fato bastante evidente citado pelo autor é a dimensão extremamente reduzida das árvores produzidas pelo algoritmo CART, onde o mesmo constatou que por meio da poda por redução de custo versus complexidade, é obtido um melhor desempenho.

Conforme o autor, ás arvores geradas pelo algoritmo CART, após serem podadas, são as que possuem as mais reduzidas dimensões, que apesar disso, apresentam um bom desempenho.

7.4 ESTRATÉGIAS AUXILIARES PARA GRADUAÇÃO DOS TUMORES ASTROCÍTICOS SEGUNDO OS CRITÉRIOS HISTOPATOLÓGICOS ESTABELECIDOS PELA OMS

Este estudo foi realizado em 2006 por Mário Henrique Girão Faria, Régia Maria do Socorro Vidal do Patrocínio e Silvia Helena Barem Rabenhorst. O objetivo foi

desenvolver estratégias auxiliares para graduação dos tumores astrocíticos a partir dos critérios histológicos estabelecidos pela Organização Mundial de Saúde (OMS).

O estudo deu-se pela análise clínico-epidemiológica e reavaliação histopatológica[15] de 55 astrocitomas[16] de diferentes gradações e de cinco amostras de tecido cerebral não-tumoral, utilizando o algoritmo CART, implementado no software *CART 5.0*[17] (FARIA; PATROCINIO; RABENHORST, 2006).

Os resultados obtidos com a árvore de decisão desenvolvida pelo CART permitiram concluir que a distribuição por idade, sexo e pela localização tumoral dos portadores dessas neoplasias reproduziu, de um modo geral, as tendências mundiais. A presença de células multinucleadas e de gemistócitos foi associada à malignidade tumoral. Os achados histopatológicos avaliados segundo critérios semiquantitativos e pelo modelo confirmaram a gradação original, reiterando a aplicabilidade e a reprodutibilidade dos mesmos (FARIA; PATROCINIO; RABENHORST, 2006).

7.5 MINERAÇÃO DE DADOS GEOGRÁFICOS PARA MAPEAR AS FITOFISIONOMIAS DO BIOMA CERRADO

A presente pesquisa foi realizada em 2005 por Luciano Teixeira de Oliveira, Luis Marcelo Tavares de Carvalho e Fausto Weimar Acerbi Junior, cujo principal objetivo foi desenvolver uma metodologia para o mapeamento das diversas fitofisionomias do bioma cerrado em regiões antropizadas, envolvendo a comparação da eficiência entre as imagens ETM+ e da exatidão da classificação proveniente de

[15] Estudo microscópico dos tecidos doentes (Dicionário Digital de Termos Médicos, disponível em: http://www.pdamed.com.br).

[16] Tumor cerebral, constituído por células gliais adultas denominadas astrócitos (Dicionário Digital de Termos Médicos, disponível em: http://www.pdamed.com.br).

[17] Desenvolvida por *Salford Sistems*, CART é uma ferramenta de *data mining* licenciada que analisa grandes bases de dados a procura de relacionamentos significativos. A empresa disponibiliza uma versão para avaliação de 30 dias. Maiores informações podem ser obtidas por meio do site http://www.salfordsystems.com.

imagens com 30 e 15 cm de resolução, bem como a avaliação da precisão de mapeamentos em dois níveis de detalhamento.

A partir das imagens foram adquiridos três conjuntos de atributos, sendo eles: época seca, úmida e temporal, dos quais foram extraídas amostras de treinamento. Essas amostras foram utilizadas para gerar e selecionar as melhores árvores de decisão produzidas pela ferramenta *CART 5.0*, utilizando o algoritmo CART por meio do critério de Gini. Estas mesmas amostras foram utilizadas para a classificação pelo método da *máxima verossimilhança*[18] no software *Envi 4.0*[19] (OLIVEIRA; CARVALHO; ACERBI JUNIOR, 2005).

Diante dos resultados obtidos, os autores concluíram que o procedimento por árvore de decisão se mostrou mais eficiente que o da máxima verossimilhança.

Após a exemplificação do uso do algoritmo CART, apresenta-se na próxima seção o trabalho desenvolvido, sendo descrito os passos de todo o processo de desenvolvimento desta pesquisa.

[18] Classificação aplicada no tratamento de dados satélites, onde o usuário determina a significância nos erros de atributos especificados para uma classe em comparação a outras (Guia do Envi, http://www.unb.br/fav/renova/FOTO/ENVI/F-Classificacao.pdf).

[19] Desenvolvido por *SulSoft*, trata-se de um software para processar imagens de Sensoriamento Remoto. Maiores informações podem ser obtidas em http://www.envi.com.br.

8 O ALGORITMO CART NA TAREFA DE CLASSIFICAÇÃO DA *SHELL* ORION *DATA MINING* ENGINE

A *Shell* Orion, por meio de suas tarefas e métodos implementados, consiste em uma forma de apoio ao meio acadêmico, auxiliando na agregação de conhecimentos relacionados a *data mining* .

A implementação do algoritmo CART para a tarefa de classificação por meio da indução de árvores de decisão na *Shell* Orion ampliou o número de algoritmos para esta tarefa, proporcionando que diferentes resultados sejam obtidos por meio da classificação.

A realização de testes do algoritmo deu-se em função da utilização da base de dados Íris, que apresenta características de três tipos de flores e possui cinco atributos, sendo quatro numéricos e um categórico.

8.1 BASE DE DADOS UTILIZADA

A fim de ilustrar o funcionamento do algoritmo CART na tarefa de classificação da *Shell* Orion, foi utilizada uma base de dados composta por 150 exemplos referente a três tipos de plantas da família das Iridáceas: setosa, versicolor e virgínica (Figura 12).

Nesta família de plantas existem mais de 1000 espécies, sobretudo nos países temperados, poucas das quais brasileiras. Muitas delas são ornamentais e comuns nos jardins. Uma espécie brasileira bastante conhecida é a palma-de-santa-rita (ZANUSSO, 2002).

Esta base de dados está disponível nas ferramentas WEKA 3.5.5 e CART 5.0, cujos dados estão distribuídos em 50 exemplos para cada um dos três tipos de flores. Ela é bastante utilizada para representar árvores de decisão e possui informações referentes a cinco características das flores:

a) comp_setala (comprimento da sépala);

b) larg_setala (largura da sépala);

c) comp_petala (comprimento da pétala);

d) larg_pétala (largura da pétala);

e) especies (espécies).

Íris Setosa Íris Versicolor Íris Virgínica

Figura 12. Imagem das iridáceas: setosa, versicolor e virgínica.

8.2 METODOLOGIA

O desenvolvimento do algoritmo CART na *Shell* Orion *Data Mining Engine* fundamentou-se metodologicamente pelas seguintes etapas: levantamento bibliográfico, modelagem do módulo de classificação do CART; demonstração matemática do algoritmo CART; implementação e realização de testes.

8.2.1 Levantamento Bibliográfico

Compreendeu a pesquisa bibliográfica dos temas envolvidos neste trabalho de modo a se entender e descrever o processo de *data mining*, a tarefa de classificação e o funcionamento do algoritmo CART para indução de árvores de decisão.

Durante a pesquisa, os estudos foram dedicados principalmente ao entendimento da tarefa de classificação e principalmente do algoritmo CART, sendo que estes constituem a base desta pesquisa.

No que se refere ao algoritmo CART, os estudos estenderam-se por um longo período, devido a complexidade de suas fórmulas conforme demonstração matemática apresentada na Seção 8.2.3.

8.2.2 Modelagem do Módulo de Classificação do Algoritmo CART

Na modelagem do módulo de classificação do algoritmo CART na *Shell* Orion *Data Mining Engine*, utilizou-se a *Unified Modeling Language* (UML), realizando-se os diagramas de caso de uso, atividades e seqüência por meio da ferramenta JUDE que possui uma versão gratuita disponível no endereço http://jude.change-vision.com/jude-web/index.html.

O diagrama de caso de uso demonstra as atividades a serem realizadas pelo usuário no módulo, bem como a função do sistema (Figura 13):

a) **informar parâmetros de entrada:** o usuário informa o critério de divisão a ser utilizado (Entropia, Gini e Twoing), o atributo de saída desejado e a quantidade de partições a serem utilizadas no *cross-validation*. Além disso, o usuário solicita ao sistema a execução do

algoritmo;

b) **executar o algoritmo:** realizada a solicitação pelo usuário, o sistema executa o algoritmo CART e mostra as regras e a árvore gerada.

Figura 13. Diagrama de caso de uso

A Figura 14 demonstra o diagrama de atividades que possibilita a identificação do fluxo de tarefas executadas pelo usuário e o sistema:

a) **informa parâmetros de entrada:** o usuário informa os dados necessários para o funcionamento do algoritmo;

b) **solicita execução do algoritmo:** o usuário solicita a execução do algoritmo CART na *Shell* Orion;

c) **processa o algoritmo CART:** o sistema executa o algoritmo;

d) **visualiza os resultados:** finalizado o processamento do algoritmo CART os resultados são demonstrados ao usuário por meio de regras e de árvore de decisão.

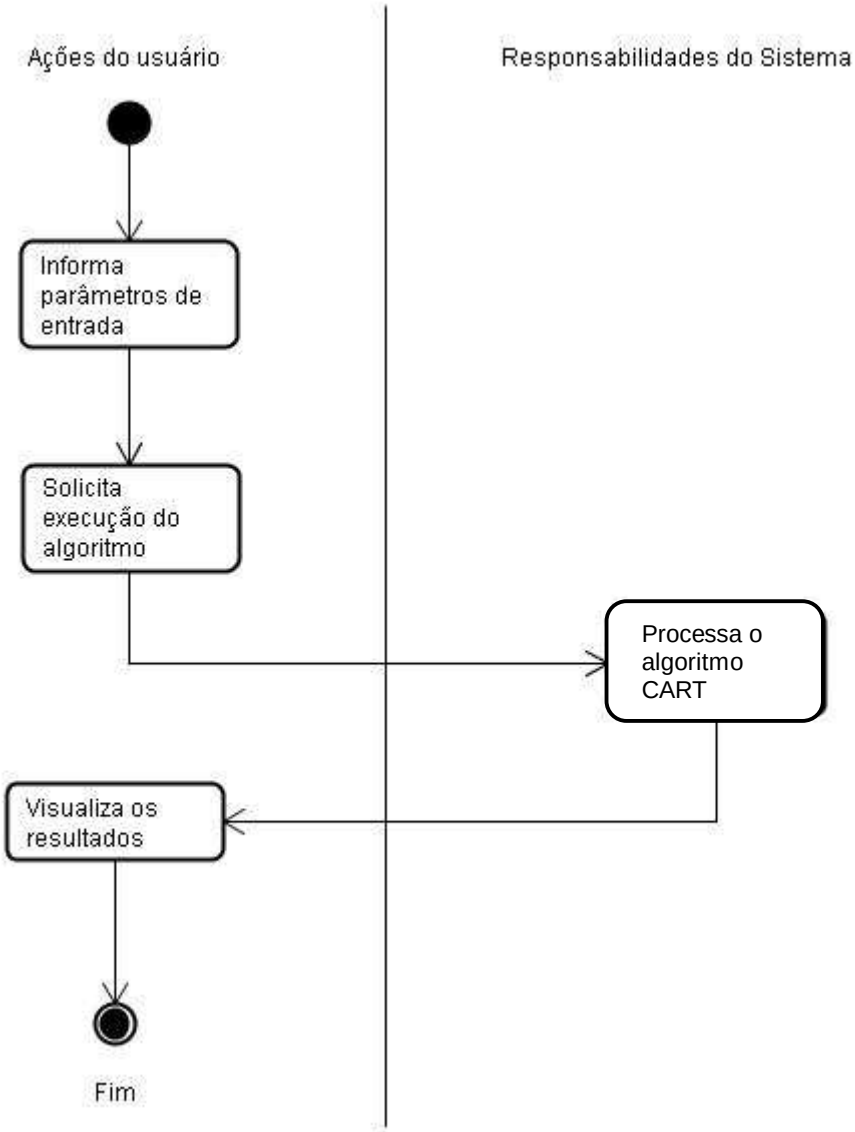

Figura 14. Diagrama de atividades

O diagrama de seqüência apresenta o processo onde o usuário solicita o método CART e após a sua execução os resultados retornam a ele.

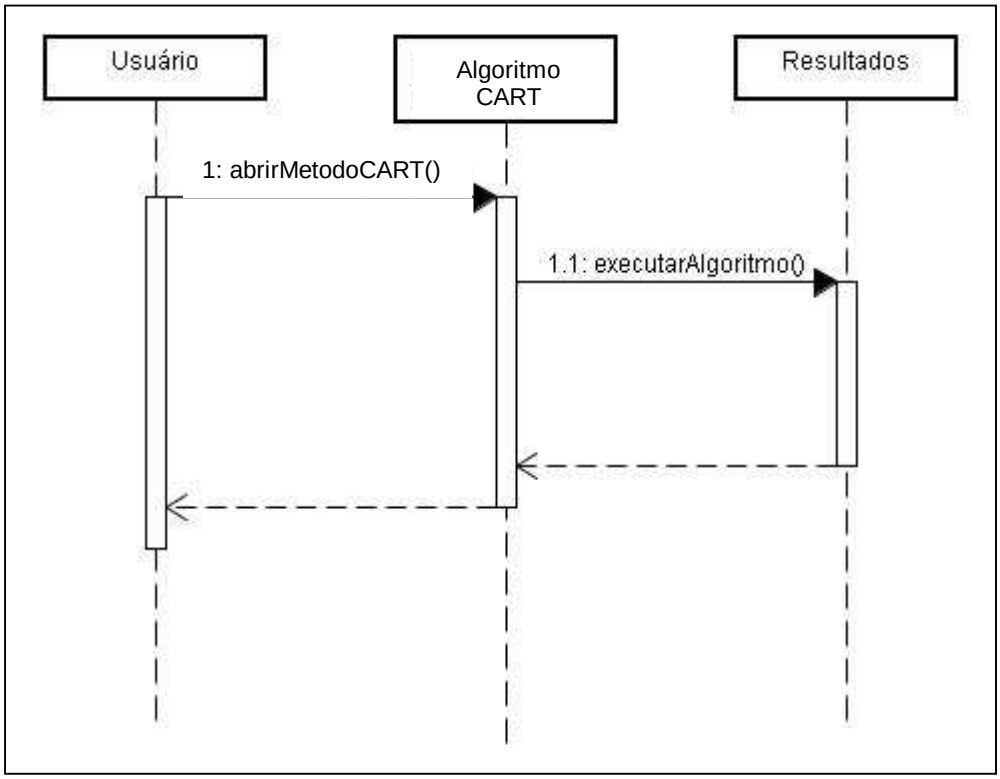

Figura 15. Diagrama de seqüência

8.2.3 Demonstração Matemática do Algoritmo CART

Nesta etapa realizou-se a demonstração matemática dos cálculos referente ao índice de Gini, atribuição da classe mais provável para uma folha, poda por minimização do custo-complexidade e valores de α para selecionar a árvore final.

Os cálculos a serem demonstrados foram aplicados em alguns atributos da base de dados referente as informações das iridáceas, sendo que a tabela utiliza 150 registros e possui quatro atributos numéricos: *comp_sepala, larg_sepala, comp_pelata, larg_petala,* e um atributo categórico: *especies*.

Considerando-se que o objeto de saída escolhido seja *espécies*, deve-se então, identificar quais as classes são geradas por esse atributo, ou seja, os valores que o mesmo pode assumir. Neste caso, as classificações que *espécies* apresenta são três:

a) iris_setosa, com 50 ocorrências;

b) iris_versicolor, com 50 ocorrências;

c) íris_virginica, com 50 ocorrências.

Logo, a impureza do nó pai é calculada a seguir:

$$i(s,t) = i(t) - p_L i(t_L) - p_R i(t_R)$$

$$i(s,t) = 1 - ((1/3)^2 + (1/3)^2 + (1/3)^2)$$

$$i(s,t) = 1 - (0{,}111 + 0{,}111 + 0{,}111)$$

$$i(s,t) = 1 - 0.333$$

$$i(s,t) = 0.667$$

Tendo em vista que os demais atributos são numéricos, para cada caso dos dados de treinamento será inserido um índice identificador, conforme mostra a Tabela 4.

Tabela 5. Exemplo de alguns dados da base com seus identicadores

id	comp_sepala	larg_sepala	comp_petala	larg_petala	especies
0	5.5	3.5	1.4	0.2	iris_setosa
1	4.9	3.0	1.4	0.2	iris_setosa
2	7.0	3.2	4.7	1.4	iris_versicolor
3	6.4	3.2	4.5	1.5	iris_versicolor
4	6.3	3.3	6.0	2.5	iris_virginica
5	5.4	2.7	5.1	1.9	iris_virginica

Inicialmente, para encontrar o melhor ponto de divisão dos atributos, uma cópia dos dados de treinamento é efetuada, sendo que estes devem estar ordenados pelo valor do atributo, conforme a Tabela 5. A ordenação dos dados no algoritmo CART

implementado na *Shell* Orion foi utilizada por meio do algoritmo *quickSort*[20].

Tabela 6. Ordenação dos dados por ordem de valor do atributo

id	comp_sepala	id	larg_sepala	id	comp_petala	id	larg_petala
1	4.9	5	2.7	0	1.4	0	0.2
5	5.4	1	3.0	1	1.4	1	0.2
0	5.5	2	3.2	3	4.5	2	1.4
4	6.3	3	3.2	2	4.7	3	1.5
3	6.4	4	3.3	5	5.1	5	1.9
2	7.0	0	3.5	4	6.0	4	2.5

Após efetuada a ordenação dos dados, calcula-se o índice de Gini para cada posição dos dados, sendo que a condição de particionamento da árvore de decisão será a média entre os valores do atributo para as posições atual e seguinte da tabela. De modo a simplificar o entendimento do cálculo de Gini, são demonstrados abaixo os passos que se deve seguir para encontrar o melhor ponto de divisão em cada atributo:

1. cria-se uma variável

Ex: *melhorGini*;

2. uma variável recebe o primeiro valor do atributo

Ex: *divisaoCorrente* = 4.9;

3. uma variável que corresponderá a quantidade de dados que estão à direita, recebe o valor zero

Ex: *direita* = 0;

[20] Método de ordenação muito rápido e eficiente quando considerado uma grande quantidade de dados (DEITEL, 2005).

4. uma variável que corresponderá a quantidade de dados que estão à esquerda, recebe o total de dados existentes para o atributo

Ex: *esquerda* = 6;

Enquanto existirem dados, executam-se os passos:

5. caso o primeiro dado da lista do atributo for maior que *divisaoCorrente*, encontrar o valor de Gini;

6. calcula-se o índice de Gini para o primeiro dado da lista do atributo;

7. se for o primeiro Gini encontrado, *melhorGini* recebe este valor; se não for, caso o valor de Gini encontrado for maior que o armazenado em *melhorGini*, então *melhorGini* passa a conter o valor de Gini encontrado;

8. *divisaoCorrente* recebe o próximo dado do atributo

Ex: *divisaoCorrente* = 5.4;

9. *direita* recebe o que possui + 1 e *esquerda* o -1;

10. caso executado o passo 6, encontrar o ponto de divisão:

(valor do passo 5 + *divisaoCorrente*) /2.

Aplicando-se esses passos aos dados do atributo *comp_sepala*, constantes no Anexo A, tem-se:

1. *melhorGini*

2. *divisaoCorrente* = 4.3

3. *direita* = 0

4. *esquerda = 150*

<u>1ª iteração:</u>

5. 4.3 > 4.3 (os valores são iguais)

6. portanto, não se calcula o índice de Gini

7. não atribui-se valor a *melhorGini*

8. *divisaoCorrente* = 4.3

9. *direita* = 1*; esquerda* =149

10. não foi executado o passo 6

2ª iteração:

5. 4.4 > 4.3 (os valores não são iguais)

6. *i(s,t)* = 0.6666667 - ((1.0/150.0) * 0.0 - (149.0/150.0 * 0.6666366))

i(s,t) = 0.0044743

7. *melhorGini* = 0.0044743

8. *divisaoCorrente* = 4.4

9. *direita* = 1*; esquerda* =148

10. 4.4 + 4.3 / 2 = 4.35

Repetindo-se este procedimento para todos os dados do atributo *comp_sepala*, encontra-se o melhor ponto de divisão para este atributo:

51ª iteração:

5. 5.5 > 5.4 (os valores não são iguais)

6. *i(s,t)*= 0.6666667 - ((52.0/150.0) * 0.2374260 - (98.0/150.0 * 0.5458142))

i(s,t)= 0.2277603

7. *melhorGini* = 0.2277603

8. *divisaoCorrente* = 5.5

9. *direita* = 53*; esquerda* =97

10. 5.5 + 5.4 / 2 = 5.45

Nas demais iterações, o índice de Gini não foi maior que o encontrado na 51ª iteração, cujo melhor ponto para divisão deste atributo é 5.45. Efetuando-se esse procedimento para todos os atributos, a fim de encontrar aquele que irá efetuar a divisão do nó raiz, chega-se aos resultados apresentados na Tabela 6.

Tabela 7. Índice de Gini encontrado para atributos

Atributo	**Indice de Gini**	**Ponto de Divisão**
comp_sepala	0.2277603	5.45
larg_sepala	0.1203704	3.35
comp_petala	0.3333333	2.45
larg_petala	0.3333333	0.8

Conforme se pode observar, o atributo a ser utilizado para a divisão do nó será o *comp_petala*, pois foi o que atingiu o maior índice de Gini e seu ponto de divisão será 2.45. O atributo *larg_petala* atingiu o mesmo valor, porém nestes casos, o primeiro atributo encontrado é o selecionado.

Utilizando-se então o atributo *comp_petala* para dividir o nó raiz, tem-se a primeira divisão conforme mostra a Figura 16, onde, nos casos em que *comp_petala* for menor ou igual a 2.45, segue para a esquerda do nó, e caso for maior que 2.45, direciona-se para a direita.

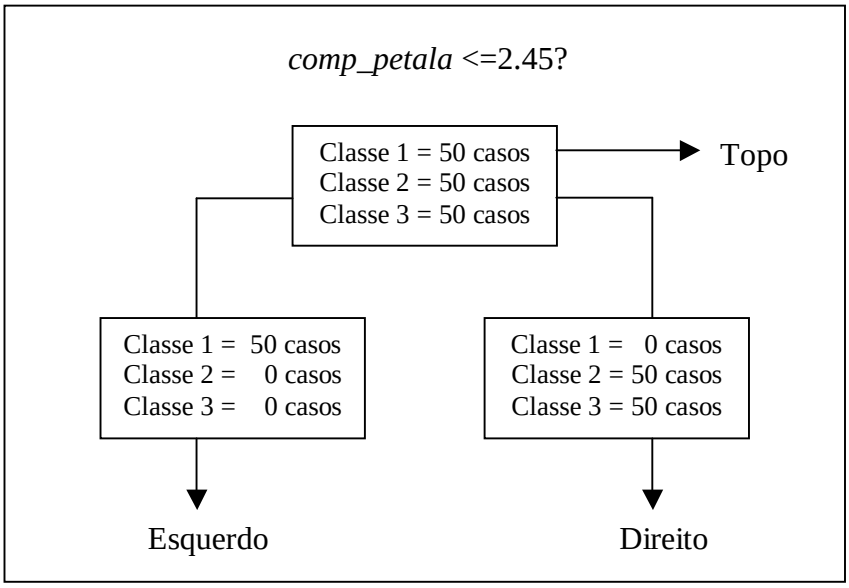

Figura 16. Divisão do primeiro nó da árvore

Aplicando-se todos os procedimentos mencionados anteriormente com o objetivo de efetuar a divisão do nó esquerdo, obtém-se para todos os atributos o índice de Gini igual a zero. Portanto, esse será um nó terminal, ou seja, uma folha da árvore e, quando se encontra uma folha, atribui-se a classe mais provável (maior probabilidade) a ela, que neste caso é a classe 1 conforme demonstrado abaixo:

Classe 1 = 50/50 = 1 Classe 2 = 0/50 = 0 Classe3 = 0/50 = 0

O desenvolvimento da árvore dá-se aplicando todos esses procedimentos. Assim, quando todos os casos dos nós da árvore apresentarem a mesma classe encerra-se o desenvolvimento da mesma, gerando a árvore demonstrada na Figura 17.

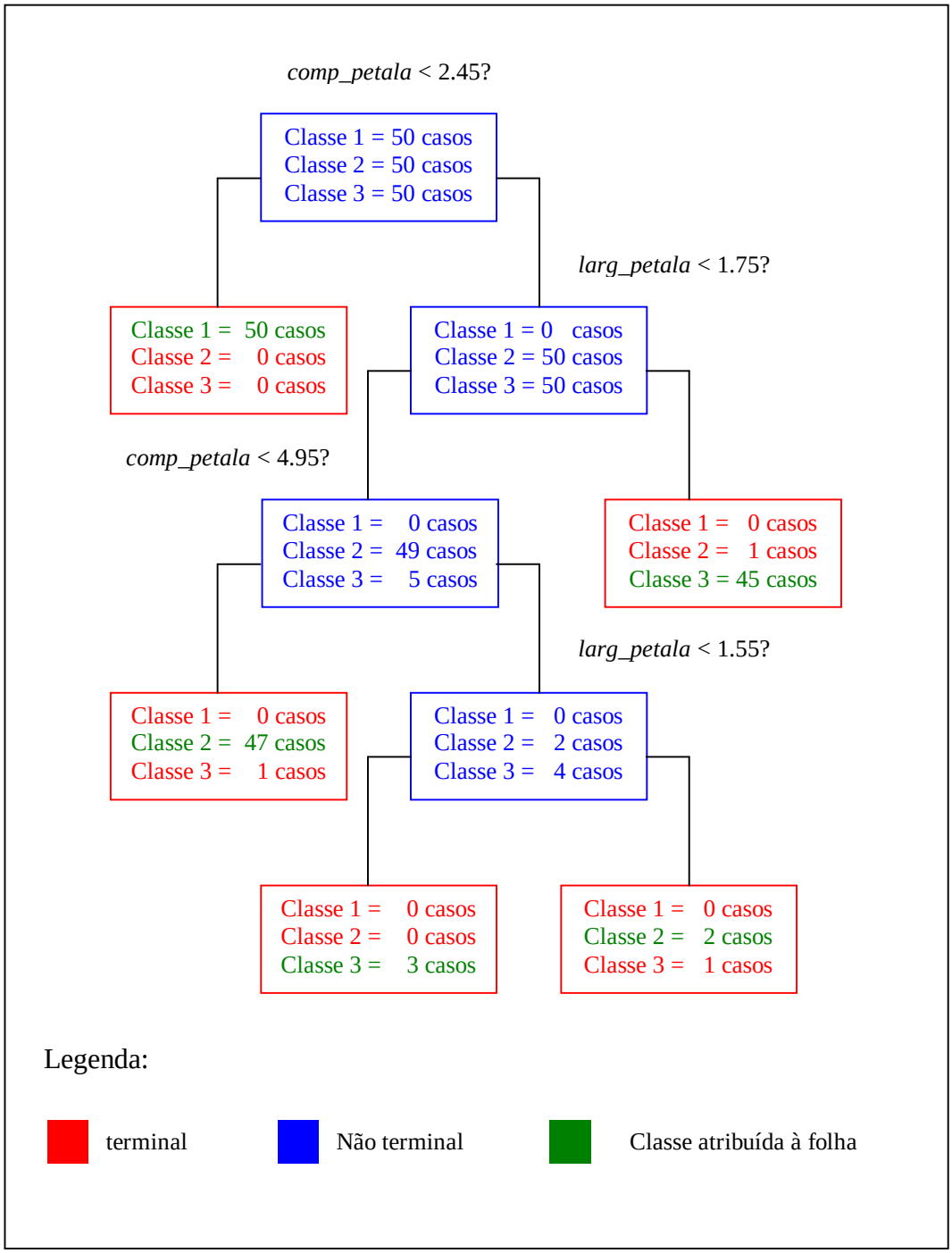

Figura 17. Árvore construída pelo CART

Após desenvolvida a árvore, deve-se podá-la de modo a gerar uma seqüência de sub-árvores, conforme Figura 18.

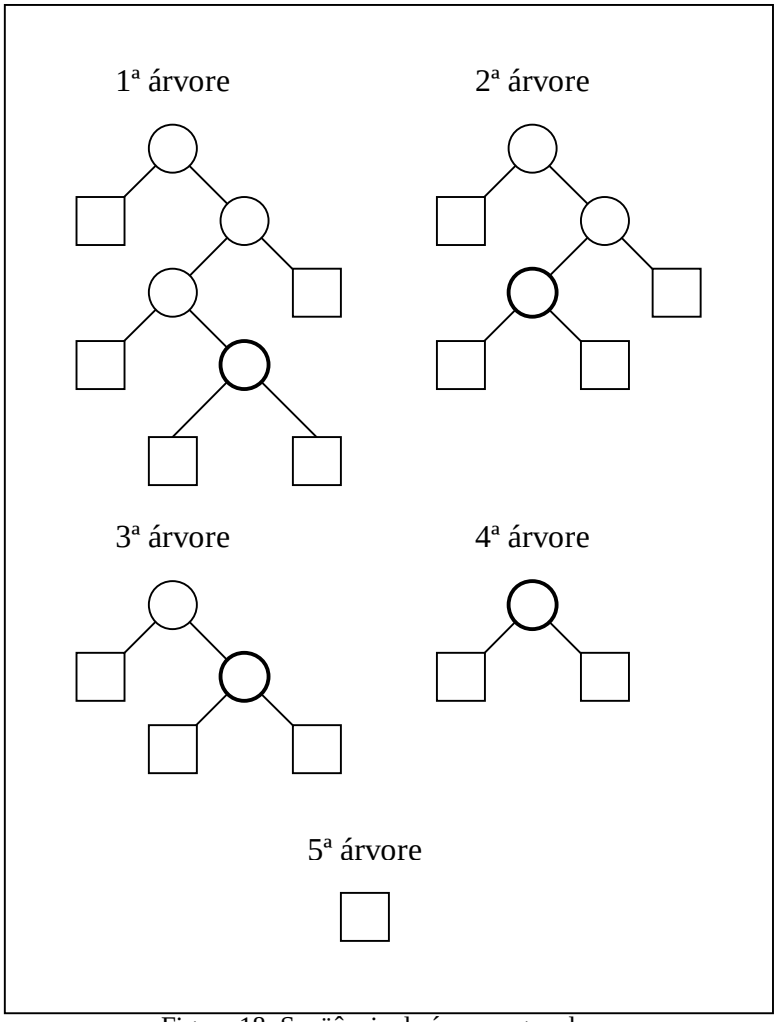

Figura 18. Seqüência de árvores geradas

Observando-se as Figuras 17 e 18, têm-se todos os dados necessários para encontrar os valores de α, demonstrados na Tabela 7, para cada árvore. O cálculo é efetuado da seguinte maneira: percorre-se a árvore partindo-se da raiz; ao encontrar o último nó não-terminal, aplica-se a equação (20) resultando nos valores de α para cada árvore, conforme:

1ª árvore: $\alpha = 0$ (sempre inicia com 0 e segue variando até a última árvore)

2ª árvore: $\alpha = \dfrac{\dfrac{2-1}{150}}{2-1} = 0.0066666667$

3ª árvore: $\alpha = \dfrac{\dfrac{5-3}{150}}{2-1} = 0.0133333333$

4ª árvore: $\alpha = \frac{\frac{50-6}{150}}{2-1} = 0.2933333333$

5ª árvore: $\alpha = \frac{\frac{100-50}{150}}{2-1} = 0.3333333333$

Tabela 8. Parâmetro de complexidade para cada sub-árvore gerada

Árvore	**Nós terminais**	**Parâmetro de Complexidade**
1	5	0.0
2	4	0.0066666667
3	3	0.0133333333
5	2	0.2933333333
6	1	0.3333333333

Esses valores de α são armazenados em uma lista que será utilizada na última etapa do processo para a obtenção do classificador, onde se encontra a melhor árvore por meio da técnica de validação cruzada.

O processo de validação cruzada consiste em dividir o conjunto de treinamento em subconjuntos, de forma que possuam o tamanho mais semelhante possível. A divisão é feita baseando-se na quantidade de partições informadas para a realização do *cross-validation.* De acordo com Breiman et al (1984) o número 10 constitui-se no valor ideal de partições.

Considerando-se isto e a quantidade de dados disponíveis na base Iridáceas, o conjunto de amostras é dividido aleatoriamente em 10 subconjuntos contendo cada um

deles 15 casos. Após, são construídas tantas árvores quanto o número de subconjuntos criados e aplica-se a técnica da poda, dividindo-se cada uma em um conjunto de sub-árvores onde serão encontrados os valores de α (demonstrado anteriormente) e a taxa de erro para cada uma, que consiste em dividir o número de amostras não classificadas pelo total de dados da base.

A árvore contendo todos os dados de treinamento foi desenvolvida anteriormente (Figura 17), portanto, deve-se encontrar a melhor delas aplicando-se a fórmula 21, descrita abaixo:

$$\mathrm{N}_{i,j} = \sum_{i,j} \mathrm{N}_{ij}^{(v)}$$

Onde: para cada validação cruzada efetuada somam-se as taxas de erro encontradas em suas sub-árvores e divide-se pelo número de partições pré-definidas, que neste caso é 10.

O total de erros de cada validação é armazenado em uma lista que será percorrida encontrando-se a posição que possui o menor valor, posteriormente, procura-se por esta posição na lista de α. Encontrando-se este valor o mesmo será utilizado na fórmula (23), que resulta no melhor parâmetro de complexidade em relação ao erro obtido. Então, a partir do momento em que esse valor for menor que o α encontrado para cada sub-árvore (desenvolvida com o total de amostras conforme Tabela 7) ocorre a simplificação da árvore (poda).

Alguns detalhes da implementação efetuada bem como os resultados obtidos mediante testes realizados a fim de confirmar o funcionamento do algoritmo CART na *Shell* Orion são apresentados na seção seguinte.

8.2.4 Implementação e Testes do Módulo de Classificação por meio do Algoritmo CART

A implementação do algoritmo CART para a classificação de dados na *Shell* Orion *Data Mining Engine* foi realizada por meio da tecnologia Java e do ambiente de desenvolvimento Netbeans 5.5 que se encontra, gratuitamente, disponível para *download* em http://www.netbeans.org.

No desenvolvimento deste algoritmo os conceitos, técnicas e formalismos matemáticos foram baseados na bibliografia dos quatro estatísticos criadores do CART, Leo Breiman, Jerome Friedman, Richard Oslen e Charles Stone, intitulada *Classification and Regression Trees*.

A Figura 19 apresenta a interface principal da *Shell* Orion *Data Mining Engine*, o menu da etapa de *data mining* e o submenu com a tarefa de classificação e o método CART.

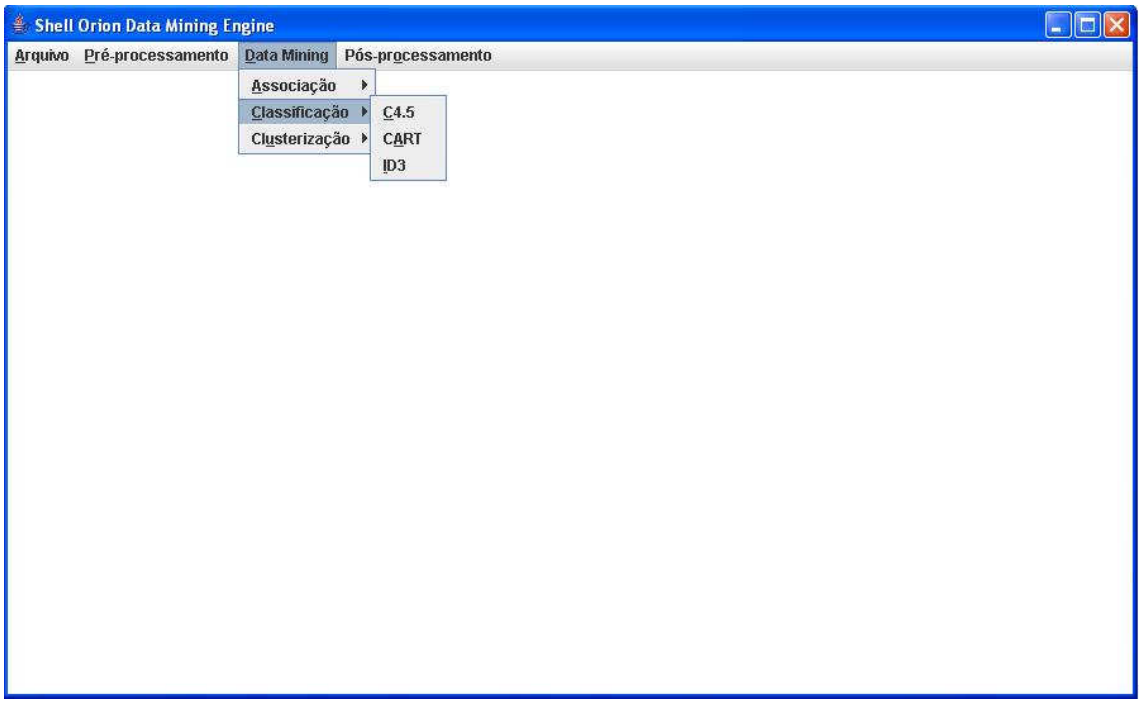

Figura 19. Interface principal

A tarefa de classificação, para identificar uma função que associe corretamente cada registro de uma base de dados a uma classe, por meio do método de indução de árvores de decisão pelo algoritmo CART necessita que o usuário informe alguns parâmetros de entrada (Figura 20):

a) **critério:** o usuário deve selecionar um dos critérios disponíveis para encontrar o melhor ponto de divisão na árvore, sendo que atualmente se encontra implementado o critério Gini;

b) **atributo de saída:** deve-se informar qual será a saída do classificador, ou seja, o conseqüente da regra de classificação a ser gerada;

d) ***cross-validation:*** informa-se neste campo o número de partições que o algoritmo efetuará para executar a simplificação da árvore.

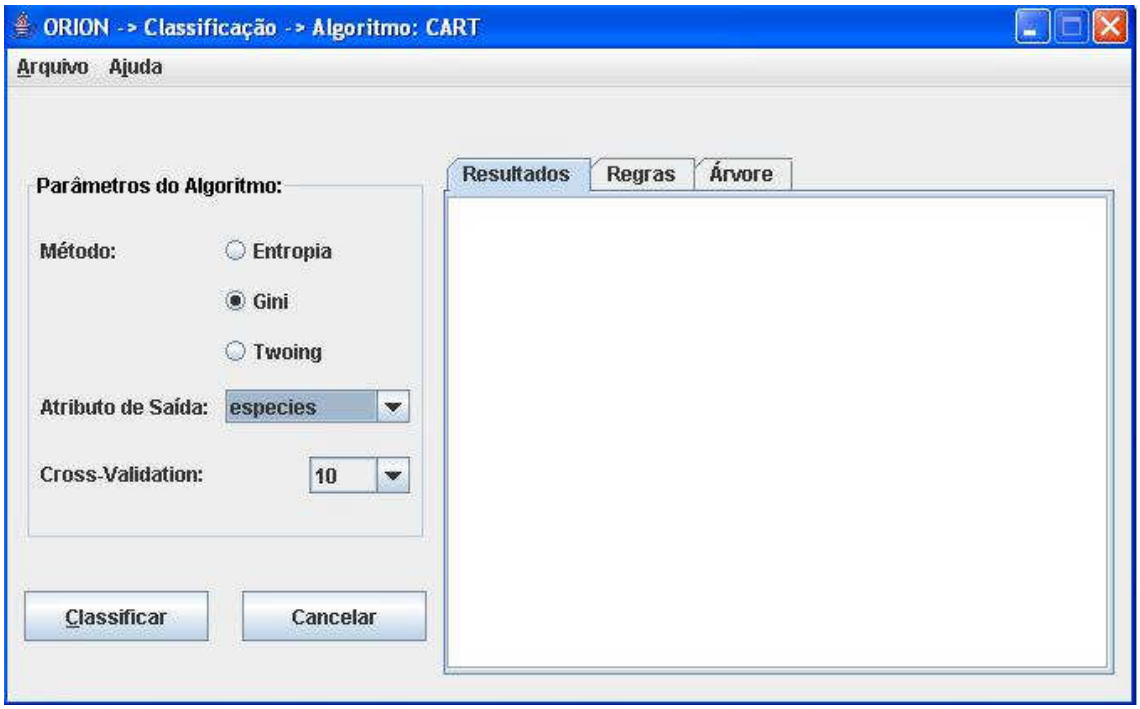

Figura 20. Algoritmo CART da *Shell* Orion *Data Mining Engine*

Após a execução do algoritmo CART, conforme os parâmetros de entrada, percebe-se as regras de classificação geradas (Figura 21), tendo-se também uma forma de visualização gráfica por meio de uma árvore (Figura 22).

Figura 21. Regras geradas pelo algoritmo CART

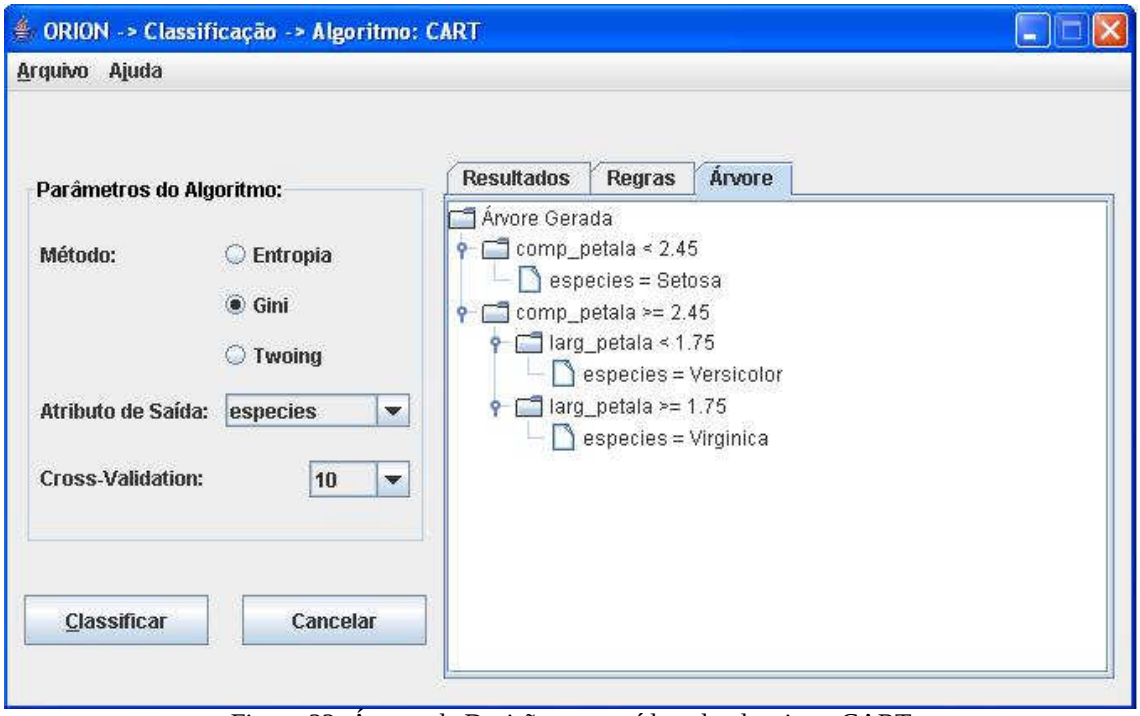

Figura 22. Árvore de Decisão construída pelo algoritmo CART

Finalizando-se a implementação deste algoritmo na tarefa de classificação foram efetuados alguns testes utilizando-se a base de dados das Iridáceas, que é disponibilizada e bastante utilizada por diferentes ferramentas que o implementam, como por exemplo, Weka e CART.

8.3 RESULTADOS OBTIDOS

Os resultados foram obtidos por meio da realização dos testes na tarefa de classificação pelo algoritmo CART na *Shell* Orion *Data Mining Engine* no que se refere ao funcionamento do módulo e a análise das relações e padrões descobertos na base de dados das Iridáceas.

Na realização dos testes do algoritmo CART para a classificação dos dados utilizou-se um microcomputador com sistema operacional Windows XP *Home Edition*, processador Intel 1.8 Ghz e 512 MB de memória RAM.

O algoritmo CART teve seu funcionamento testado por meio de diferentes valores de *cross-validation*, analisando-se que conforme se diminuía a quantidade das partições efetuadas pelo algoritmo para realizar a simplificação da árvore o tamanho da mesma também diminuía, demonstrando que o aumento das partições exige a execução de mais cálculos para que a poda venha a ser realizada e conseqüentemente elevando-se o tempo de processamento.

Considerando-se a construção e simplificação de uma árvore de decisão pelo algoritmo CART, onde se tem como parâmetro de *cross-validation* 10 partições são desenvolvidas um total de 11 árvores, sendo que uma foi construída com os 150 casos da base (total de dados) e as demais com 135 casos cada uma (150 – 150/10). A partir disso, foram realizadas podas nestas árvores a fim de originar as sub-árvores e foram

calculados os erros de classificação para cada uma, a seguir reúnem-se todos estes erros resultando no erro total para cada árvore. Finalmente, seleciona-se o menor erro e o valor de alfa nesta posição a fim de realizar cálculos que identifiquem o classificador final.

Mediante os vários cálculos matemáticos envolvidos na execução do algoritmo CART para a classificação de um registro de uma base de dados a um rótulo (classe) o desempenho não foi satisfatório apresentando um tempo de processamento de 10 segundos para os 150 casos da base Iridáceas.

Realizando-se este processo anteriormente descrito foi possível analisar o conhecimento gerado por meio do algoritmo CART na base de dados das Iridáceas, para isso consideraram-se os quatro atributos (largura e comprimento da pétala e o mesmo para sépala) e os três tipos de flores (setosa, versicolor e virginica) para o atributo de saída.

As regras geradas de modo a classificar os tipos de flores identificaram que quando se tem o comprimento da pétala menor que 2.45 cm todas as amostras são classificadas corretamente no tipo setosa. Caso este atributo possua um valor maior ou igual a 2.45 cm e a largura da pétala for menor que 1.75 cm tem-se a espécie do tipo versicolor. A última regra considera que tendo-se a largura da pétala maior ou igual a 1.75 cm as amostras são classificadas no tipo virgínica. Durante a classificação das flores versicolor e virgínica nem todas as amostras foram classificadas corretamente, diante disso, verifica-se que o classificador final obtém um percentual de erro de 4% (Figura 23).

```
petallen < 2.45: 1(50.0/0.0)
petallen >= 2.45
|  petalwid < 1.75: 2(49.0/5.0)
|  petalwid >= 1.75: 3(45.0/1.0)
 -> Número de nós-folha = 3
 -> Tamanho da Árvore = 5
```

Comparando-se o percentual de erro de 4% com o obtido em outras ferramentas de *data mining* utilizando-se a mesma base de dados, o resultado foi satisfatório, verificando-se dessa forma a correta execução do algoritmo CART na Shell Orion.

CONCLUSÃO

O processo de *data mining* é fundamental não somente para a descoberta de novos conhecimentos, mas também para confirmação dos já existentes podendo proporcionar benefícios significativos às instituições no que se refere a tomada de decisão e vantagem estratégica.

A sua aplicação se dá por meio de tarefas e métodos, sendo que nesta pesquisa os estudos concentraram-se acerca do algoritmo CART para indução de árvores de decisão, possibilitando a classificação correta dos dados e não exigindo para isso transformações a fim de adaptá-los ao algoritmo. Além disso, torna as árvores menores em relação a outros métodos de indução já que implementa a simplificação, o que vem a auxiliar no entendimento das relações descobertas.

Na realização desta pesquisa concluiu-se a importância do conteúdo referente a estrutura de dados, em especial o de árvores binárias, abordado durante o período de graduação. Contudo, encontraram-se algumas dificuldades especialmente no que se refere ao entendimento dos formalismos matemáticos envolvidos no algoritmo CART. Isto ocorreu devido a carência de material didático e explicativo que auxiliasse nesse estudo, bem como da diversidade de cálculos envolvidos. Dificuldades essas superadas, fazendo-se com que os objetivos da pesquisa em relação ao entendimento, demonstração matemática e implementação do algoritmo CART na Shell Orion *Data Mining Engine* fossem atingidos.

O algoritmo CART na *Shell* Orion *Data Mining Engine* apresentou desempenho satisfatório classificando corretamente os dados a que foi submetido e apesar da diversidade de cálculos presentes na sua execução tendo um tempo de processamento considerado aceitável.

Finalizando, tem-se algumas sugestões de trabalhos futuros a fim de dar continuidade ao desenvolvimento da *Shell* Orion, como por exemplo: a implementação de diferentes critérios de divisão, como entropia e twoing, bem como de tipos de dados no algoritmo CART; o desenvolvimento de variações desse algoritmo a fim de otimizá-lo e de outros métodos de classificação, como os que se referem a redes neurais, algoritmos genéticos e classificadores bayesianos; e a elaboração de uma metodologia estatística para avaliação dos algoritmos desenvolvidos na *Shell* Orion.

REFERÊNCIAS

ARBEX, Eduardo Compasso et al. **Iniciação Cientifica**: *Data Mining*. Associação Educacional Dom Bosco em Resende, RJ, 2005. Disponível em: <http://www.inf.aedb.br/datamining/index.html>. Acesso em: 14 Nov 2005.

BREIMAN, Leo et al. **Classification fan Regression Trees**. New York: Chapman e Hall/CRC, 1984.

CLARKE, Robin Thomas, BITTENCOURT, Hélio Radke. 2003. **Uso de Árvores de Decisão na Classificação de Imagens Digitais.** Disponível em: <http://marte.dpi.inpe.br/col/ltid.inpe.br/sbsr/2002/11.13.15.19/doc/15_136.pdf> Acesso em 20 nov 2006.

CARVALHO, Luiz Alfredo Vidal de. **Data mining:** a mineração de dados no marketing, medicina, economia, engenharia e administração. São Paulo: Érica, 2002.

CARVALHO, Déborah Ribeiro. **Árvore de Decisão/Algoritmo Genético para Tratar o Problema de Pequenos Disjuntos em Classificação de Dados**. Tese de Mestrado – Programa de Engenharia Civil, Universidade Federal do Rio de |Janeiro, Rio de Janeiro, 2005.

CASAGRANDE, Diego Paz. **O Módulo da Técnica de Associação pelo Algoritmo Apriori no desenvolvimento da *Shell* de *Data Mining* Orion.** Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina, 2005.

DEITEL, H. M; DEITEL, P. J. **Java: como program**ar, São Paulo: Pearson Education do Brasil, 2005.

ESPÍNDOLA, Rogério Pinto. **Sistema Inteligente para Classificação de Dados.** 2004**.** Tese de Doutorado - Trabalho de Conclusão de Curso – Programa de Pós-Graduação em Engenharia, Universidade Federal do Rio de Janeiro, Rio de Janeiro, BR – SC. Disponível em: <http://www.cos.ufrj.br/~marta/papers/TeseMauroS.pdf> Acesso em 20 set 2006.

FARIA, Mário Henrique Girão; PATROCINIO, Régia Maria do Socorro Vidal do;RABENHIRST, Silvia Helena Barem. **Estratégias Auxiliares para Graduação dos Tumores Astrocíticos segundo os Critérios Histopatológicos Estabelecidos pela OMS.** Jornal Brasileiro de Patologia e Medicina Laboratorial. Rio de Janeiro, BR-SC. Disponível em: http://www.scielo.br/scielo.php?pid=S1676-24442006000500012&script=sci_abstract

FAYYAD, Usama; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. *From Data mining to Knowledge Discovery in Databases*. 1996. in: *The American Association for Artificial Intelligence*. **AI Magazine**. Disponível no link: http://kdnuggets.com/gpspubs/aimag-kdd-overview-1996-Fayyad.pdf. Acessado em: 23 nov 2006.

FONSECA, José Manuel Matos da. **Indução de Árvores de Decisão**. Tese de Mestrado - Departamento de Informática – Universidade Novas de Lisboa, Lisboa 1994.

GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel. **Data Mining:** um guia prático. Rio de Janeiro: Elsevier, 2005.

GONÇALVES, Eduardo Correia. **Extração de Árvores de Decisão com a Ferramenta de Data Mining Weka**. Artigo - Instituto Brasileiro de Geografia e Estatística – IBGE, 2006. Disponível em: <http://www.devmedia.com.br/articles/viewcomp.asp?comp=3388> acesso em 15 fev 2007.

HAN, Jiawei; KAMBER, Micheline. **Data mining** : concepts and techniques. San Francisco: Morgan Kaufmann Publishers, 2001.

HAND, David; MANNILA, Heikki; SMYTH, Padhraic. **Principles of Data Mining.** Massachusetts: The MIT Press, 2001.

KANTARDZIC, Mehmed. **Data Mining: Concepts, Models, Methods, and Algorithms**. John Wiley e Sons. 2003.

MORAES, Celso Roberto. **Estrutura de dados e algoritmos: uma abordagem didática.** São Paulo: Berkeley Brasil, 2001.

MOTTA, Custódio G. L. da. **Sistema Inteligente para Avaliação de Riscos em Vias de Transporte Terrestre.** 2004. 153 f. - Programa de Pós-graduação de Engenharia, Universidade Federal do Rio de Janeiro. Disponível em: <http://www.coc.ufrj.br/teses/mestrado/inter/2004/Teses/MOTTA_CGL_04_t_M_int.pdf.> Acesso em: 12 nov. 2006.

OLIVEIRA, Luciano Teixeira de; CARVALHO, Luis Marcelo Tavares de; ACERBI JUNIOR, Fausto Weimar. **Mineração de Dados Geográficos para Mapear as Fitofisionomias do Bioma Cerrado.** Disponível em: <http://marte.dpi.inpe.br/col/ltid.inpe.br/sbsr/2004/11.23.11.27/doc/4177.pdf > Acesso em 25 jun 2007.

OLIVEIRA, Rodrigo Réus de. **Uso de *Data Mining* para obter perfis de clientes com maior lucratividade**. Monografia (Especialização em Gerenciamento em Banco de Dados) - Universidade do Extremo Sul Catarinense, Criciúma, 2006.

PELEGRIN, Diana Colombo. **A Tarefa de Classificação e o Algoritmo Id3 para Indução de Árvores de Decisão na *Shell* de *Data Mining* Orion.** Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina, 2005.

QUINLAN, J.R.**C4.5:** Programs for Machine Learning. California: Morgan Kaufmann Publishers, 1993.

QUINTAO, Ronan Torres. **Avaliação do Desempenho Logístico da Cadeia Brasileira de Suprimentos de Refrigerantes**. 2003. Dissertação de Mestrado – Curso de Engenharia de Produção, Universidade Federal de Minas Gerais, Minas Gerais, BR-PR. Disponível em:
<http://www.dep.ufmg.br/pos/defesas/diss083.pdf> Acesso em 15 nov 2006.

ROSA, Joel Maurício Correia da. **Modelos de Regressão com Transição Suave Estruturados por Árvores.** 2005. Tese de Doutorado - Departamento de Engenharia Elétrica da PUC-Rio - Programa de Pós–Graduação em Engenharia Elétrica.

RODRIGUES, Marco Antonio dos Santos. **Árvores de Classificação.** 2005. Monografia – Departamento de Matemática – Universidade do Açores, Açores.

ROMÃO, Wesley. **Descoberta de conhecimento relevante em banco de dados sobre ciência e tecnologia.** 2002. 253 f. Tese de Doutorado **–** Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis, BR – SC. Disponível em:< http://teses.eps.ufsc.br/defesa/pdf/3079.pdf> Acesso em: 02 nov. 2006.

RUSSELL, Stuart J.; NORVIG, Peter. **Inteligência artificial**. Rio de Janeiro: Elsevier, 2004.

SCOSS, Anne Marie. **A Clusterização e a Classificação no processo de *Data Mining* para Análise do Perfil do Desempenho Docente, por área, no contexto da Avaliação do Ensino de Graduação.** Monografia (MBA Gerenciamento em Banco de Dados) – Programa de Pós-graduação - Universidade do Extremo Sul Catarinense, 2006.

SERRA, Laércio. **A Essência do Business Intelligence.** São Paulo: Berkeley, 2002.

TONI, José Alexandre de. **Definiçao de um Data Mark em Cooperativas Agropecuárias.** 2000. Tese (Mestrado em Engenharia de Produção) - Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis, BR – SC. Disponível em:< http://teses.eps.ufsc.br/defesa/pdf/4034.pdf > Acesso em: 02 nov. 2006.

VASCONCELLOS, Maurício Sandoval. **Proposta de Método para análise de concessões de Crédito a Pessoas Físicas.** 2002. Tese de Mestrado em Economia – Universidade de São Paulo, São Paulo, BR-SP.

ZANUSSO, Maria Bernadete. **Rede Difusa com T-Normas Diferenciáveis.** 2002. Departamento de Ciências Exatas e Tecnologia. Universidade Federal do Mato Grosso doSul.

BIBLIOGRAFIA COMPLEMENTAR

BITTENCOURT, Guilherme. **Inteligência artificial:** ferramentas e teorias**.** Florianópolis: UFSC, 1998.

BISPO, Carlos Alberto Ferreira. **Uma análise da nova geração de sistemas de apoio à decisão.** 1998. 174 f. Dissertação (Mestrado em Engenharia da Produção) – Escola de Engenharia de São Carlos da Universidade de São Paulo, São Carlos. Disponível em: < http://www.teses.usp.br/teses/disponiveis/18/18140/tde-04042004-152849/publico/dissertacao_carlos.pdf> Acesso em: 03 nov. 2006.

CAMPOS, Maria Luiza; ROCHA FILHO, Arnaldo. Data Warehouse**.** In: Jornada de Atualização em Informática, 16., 1997, Brasília. **Anais**... Disponível em: <http://genesis.nce.ufrj.br/dataware/tutorial/extraind.html#tg_VI3> Acesso em: 02 jun. 2005.

LUGER, George F. **Inteligência Artificial.** Porto Alegre: Bookmann, 2004.

SELINGER, Tarcísio Cardoso. **A Técnica de Clusterização, por Meio Do Algoritmo KMeans, no Processo de *Data Mining* em Saúde Bucal.** 2003. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense,
Criciúma- SC.

SILVA, Marcelino Pereira dos Santos. **SKDQL – Uma Linguagem Declarativa de Especificação de Consultas e Processos para Descoberta de Conhecimento em Banco de Dados e sua Implementação.** 2002. Dissertação - Centro de Informática, Universidade Federal de Pernambuco, Recife - PE.

WITTEN, I. H; FRANK, Eibe. **Data mining** : practical machine learning tools and tecniques with Java implementations. San Francisco: Morgan Kaufmann Publishers, 2000.

_____. Data mining**: practical machine learning tools and tecniques**. **San Francisco: Morgan Kaufmann, 2005.**

ANEXO A – DADOS DA BASE

Neste anexo são apresentados todos os dados dos atributos constantes na base de dados Iridáceas a fim de demonstrá-la anterior e posteriormente a ordenação dos seus índices.

A seguir tem-se o significado de cada um dos atributos da base na tabela, que para efeito de simplificação foi substituído por uma letra:

A – Identificador dos valores da tabela

B – Valor do atributo comp_sepala

C – Identificador dos valores da tabela ordenados

D - Valor do atributo comp_sepala ordenado

E - Valor do atributo larg_sepala

F - Valor do atributo larg_sepala ordenado

G - Valor do atributo comp_petala

H - Valor do atributo comp_petala ordenados

I - Valor do atributo larg_petala

J - Valor do atributo larg_petala ordenados

A	B	C	D	A	E	B	F	A	G	B	H	A	I	B	J
0	5.1	13	4.3	0	3.5	60	2.0	0	1.4	22	1.0	0	0.2	9	0.1
1	4.9	8	4.4	1	3.0	62	2.2	1	1.4	13	1.1	1	0.2	34	0.1
2	4.7	42	4.4	2	3.2	119	2.2	2	1.3	14	1.2	2	0.2	37	0.1
3	4.6	38	4.4	3	3.1	68	2.2	3	1.5	35	1.2	3	0.2	13	0.1
4	5.0	41	4.5	4	3.6	93	2.3	4	1.4	2	1.3	4	0.2	32	0.1
5	5.4	6	4.6	5	3.9	41	2.3	5	1.7	40	1.3	5	0.4	12	0.1
6	4.6	47	4.6	6	3.4	87	2.3	6	1.4	42	1.3	6	0.3	29	0.2
7	5.0	22	4.6	7	3.4	53	2.3	7	1.5	41	1.3	7	0.2	30	0.2
8	4.4	3	4.6	8	2.9	80	2.4	8	1.4	38	1.3	8	0.2	28	0.2
9	4.9	29	4.7	9	3.1	81	2.4	9	1.5	36	1.3	9	0.1	33	0.2
10	5.4	2	4.7	10	3.7	57	2.4	10	1.5	16	1.3	10	0.2	49	0.2
11	4.8	30	4.8	11	3.4	108	2.5	11	1.6	28	1.4	11	0.2	24	0.2
12	4.8	24	4.8	12	3.0	89	2.5	12	1.4	4	1.4	12	0.1	22	0.2
13	4.3	11	4.8	13	3.0	146	2.5	13	1.1	17	1.4	13	0.1	27	0.2
14	5.8	12	4.8	14	4.0	98	2.5	14	1.2	0	1.4	14	0.2	25	0.2
15	5.7	45	4.8	15	4.4	106	2.5	15	1.5	33	1.4	15	0.4	36	0.2
16	5.4	37	4.9	16	3.9	69	2.5	16	1.3	1	1.4	16	0.4	42	0.2
17	5.1	57	4.9	17	3.5	72	2.5	17	1.4	8	1.4	17	0.3	38	0.2
18	5.7	34	4.9	18	3.8	113	2.5	18	1.7	12	1.4	18	0.3	39	0.2
19	5.1	1	4.9	19	3.8	79	2.6	19	1.5	49	1.4	19	0.3	47	0.2
20	5.4	106	4.9	20	3.4	90	2.6	20	1.7	47	1.4	20	0.2	48	0.2
21	5.1	9	4.9	21	3.7	92	2.6	21	1.5	6	1.4	21	0.4	35	0.2
22	4.6	43	5.0	22	3.6	134	2.6	22	1.0	45	1.4	22	0.2	46	0.2
23	5.1	7	5.0	23	3.3	118	2.6	23	1.7	31	1.5	23	0.5	8	0.2
24	4.8	4	5.0	24	3.4	111	2.7	24	1.9	32	1.5	24	0.2	10	0.2
25	5.0	35	5.0	25	3.0	123	2.7	25	1.6	48	1.5	25	0.2	7	0.2
26	5.0	60	5.0	26	3.4	59	2.7	26	1.6	39	1.5	26	0.4	14	0.2
27	5.2	40	5.0	27	3.5	82	2.7	27	1.5	34	1.5	27	0.2	11	0.2
28	5.2	25	5.0	28	3.4	83	2.7	28	1.4	37	1.5	28	0.2	1	0.2
29	4.7	26	5.0	29	3.2	94	2.7	29	1.6	7	1.5	29	0.2	2	0.2
30	4.8	93	5.0	30	3.1	67	2.7	30	1.6	19	1.5	30	0.2	0	0.2
31	5.4	49	5.0	31	3.4	142	2.7	31	1.5	21	1.5	31	0.4	4	0.2
32	5.2	98	5.1	32	4.1	101	2.7	32	1.5	10	1.5	32	0.1	3	0.2
33	5.5	39	5.1	33	4.2	126	2.8	33	1.4	9	1.5	33	0.2	20	0.2
34	4.9	46	5.1	34	3.1	71	2.8	34	1.5	15	1.5	34	0.1	40	0.3
35	5.0	44	5.1	35	3.2	73	2.8	35	1.2	3	1.5	35	0.2	41	0.3
36	5.5	0	5.1	36	3.5	133	2.8	36	1.3	27	1.5	36	0.2	6	0.3
37	4.9	19	5.1	37	3.1	132	2.8	37	1.5	30	1.6	37	0.1	45	0.3
38	4.4	23	5.1	38	3.0	128	2.8	38	1.3	11	1.6	38	0.2	19	0.3
39	5.1	17	5.1	39	3.4	130	2.8	39	1.5	29	1.6	39	0.2	18	0.3
40	5.0	21	5.1	40	3.5	54	2.8	40	1.3	43	1.6	40	0.3	17	0.3
41	4.5	27	5.2	41	2.3	114	2.8	41	1.3	26	1.6	41	0.3	31	0.4
42	4.4	28	5.2	42	3.2	99	2.8	42	1.3	25	1.6	42	0.2	44	0.4
43	5.0	59	5.2	43	3.5	55	2.8	43	1.6	46	1.6	43	0.6	15	0.4
44	5.1	32	5.2	44	3.8	122	2.8	44	1.9	5	1.7	44	0.4	26	0.4
45	4.8	48	5.3	45	3.0	76	2.8	45	1.4	20	1.7	45	0.3	16	0.4
46	5.1	84	5.4	46	3.8	121	2.8	46	1.6	23	1.7	46	0.2	5	0.4
47	4.6	10	5.4	47	3.2	58	2.9	47	1.4	18	1.7	47	0.2	21	0.4
48	5.3	20	5.4	48	3.7	103	2.9	48	1.5	24	1.9	48	0.2	23	0.5

A	B	C	D	A	E	B	F	A	G	B	H	A	I	B	J
49	5.0	5	5.4	49	3.3	107	2.9	49	1.4	44	1.9	49	0.2	43	0.6
50	7.0	31	5.4	50	3.2	97	2.9	50	4.7	98	3.0	50	1.4	57	1.0
51	6.4	16	5.4	51	3.2	96	2.9	51	4.5	57	3.3	51	1.5	60	1.0
52	6.9	90	5.5	52	3.1	64	2.9	52	4.9	93	3.3	52	1.5	93	1.0
53	5.5	81	5.5	53	2.3	63	2.9	53	4.0	79	3.5	53	1.3	67	1.0
54	6.5	89	5.5	54	2.8	78	2.9	54	4.6	60	3.5	54	1.5	81	1.0
55	5.7	80	5.5	55	2.8	74	2.9	55	4.5	64	3.6	55	1.3	62	1.0
56	6.3	53	5.5	56	3.3	8	2.9	56	4.7	81	3.7	56	1.6	79	1.0
57	4.9	33	5.5	57	2.4	104	3.0	57	3.3	80	3.8	57	1.0	80	1.1
58	6.6	36	5.5	58	2.9	102	3.0	58	4.6	59	3.9	58	1.3	69	1.1
59	5.2	64	5.6	59	2.7	112	3.0	59	3.9	69	3.9	59	1.4	98	1.1
60	5.0	88	5.6	60	2.0	13	3.0	60	3.5	82	3.9	60	1.0	95	1.2
61	5.9	69	5.6	61	3.0	105	3.0	61	4.2	92	4.0	61	1.5	82	1.2
62	6.0	66	5.6	62	2.2	84	3.0	62	4.0	71	4.0	62	1.0	73	1.2
63	6.1	94	5.6	63	2.9	88	3.0	63	4.7	89	4.0	63	1.4	90	1.2
64	5.6	121	5.6	64	2.9	77	3.0	64	3.6	53	4.0	64	1.3	92	1.2
65	6.7	95	5.7	65	3.1	25	3.0	65	4.4	62	4.0	65	1.4	74	1.3
66	5.6	79	5.7	66	3.0	95	3.0	66	4.5	67	4.1	66	1.5	96	1.3
67	5.8	113	5.7	67	2.7	91	3.0	67	4.1	99	4.1	67	1.0	94	1.3
68	6.2	99	5.7	68	2.2	1	3.0	68	4.5	88	4.1	68	1.5	99	1.3
69	5.6	96	5.7	69	2.5	145	3.0	69	3.9	96	4.2	69	1.1	97	1.3
70	5.9	18	5.7	70	3.2	138	3.0	70	4.8	94	4.2	70	1.8	88	1.3
71	6.1	15	5.7	71	2.8	149	3.0	71	4.0	61	4.2	71	1.3	87	1.3
72	6.3	55	5.7	72	2.5	147	3.0	72	4.9	95	4.2	72	1.5	89	1.3
73	6.1	92	5.8	73	2.8	12	3.0	73	4.7	74	4.3	73	1.2	53	1.3
74	6.4	101	5.8	74	2.9	127	3.0	74	4.3	97	4.3	74	1.3	58	1.3
75	6.6	67	5.8	75	3.0	116	3.0	75	4.4	65	4.4	75	1.4	64	1.3
76	6.8	142	5.8	76	2.8	135	3.0	76	4.8	90	4.4	76	1.4	55	1.3
77	6.7	114	5.8	77	3.0	129	3.0	77	5.0	87	4.4	77	1.7	71	1.3
78	6.0	82	5.8	78	2.9	61	3.0	78	4.5	75	4.4	78	1.5	63	1.4
79	5.7	14	5.8	79	2.6	45	3.0	79	3.5	66	4.5	79	1.0	65	1.4
80	5.5	61	5.9	80	2.4	66	3.0	80	3.8	84	4.5	80	1.1	75	1.4
81	5.5	70	5.9	81	2.4	38	3.0	81	3.7	85	4.5	81	1.0	76	1.4
82	5.8	149	5.9	82	2.7	75	3.0	82	3.9	51	4.5	82	1.2	59	1.4
83	6.0	62	6.0	83	2.7	65	3.1	83	5.1	55	4.5	83	1.6	134	1.4
84	5.4	119	6.0	84	3.0	52	3.1	84	4.5	106	4.5	84	1.5	50	1.4
85	6.0	138	6.0	85	3.4	140	3.1	85	4.5	78	4.5	85	1.6	91	1.4
86	6.7	85	6.0	86	3.1	139	3.1	86	4.7	68	4.5	86	1.5	133	1.5
87	6.3	83	6.0	87	2.3	141	3.1	87	4.4	58	4.6	87	1.3	51	1.5
88	5.6	78	6.0	88	3.0	3	3.1	88	4.1	91	4.6	88	1.3	119	1.5
89	5.5	91	6.1	89	2.5	9	3.1	89	4.0	54	4.6	89	1.3	54	1.5
90	5.5	127	6.1	90	2.6	137	3.1	90	4.4	86	4.7	90	1.2	52	1.5
91	6.1	134	6.1	91	3.0	30	3.1	91	4.6	56	4.7	91	1.4	66	1.5
92	5.8	73	6.1	92	2.6	86	3.1	92	4.0	50	4.7	92	1.2	68	1.5
93	5.0	71	6.1	93	2.3	34	3.1	93	3.3	63	4.7	93	1.0	84	1.5
94	5.6	63	6.1	94	2.7	37	3.1	94	4.2	73	4.7	94	1.3	78	1.5
95	5.7	148	6.2	95	3.0	47	3.2	95	4.2	76	4.8	95	1.2	72	1.5
96	5.7	97	6.2	96	2.9	110	3.2	96	4.2	126	4.8	96	1.3	86	1.5
97	6.2	126	6.2	97	2.9	70	3.2	97	4.3	138	4.8	97	1.3	61	1.5
98	5.1	68	6.2	98	2.5	2	3.2	98	3.0	70	4.8	98	1.1	83	1.6
99	5.7	123	6.3	99	2.8	120	3.2	99	4.1	121	4.9	99	1.3	129	1.6

A	B	C	D	A	E	B	F	A	G	B	H	A	I	B	J
100	6.3	56	6.3	100	3.3	115	3.2	100	6.0	123	4.9	100	2.5	56	1.6
101	5.8	87	6.3	101	2.7	29	3.2	101	5.1	127	4.9	101	1.9	85	1.6
102	7.1	72	6.3	102	3.0	125	3.2	102	5.9	52	4.9	102	2.1	106	1.7
103	6.3	136	6.3	103	2.9	50	3.2	103	5.6	72	4.9	103	1.8	77	1.7
104	6.5	100	6.3	104	3.0	143	3.2	104	5.8	113	5.0	104	2.2	127	1.8
105	7.6	146	6.3	105	3.0	35	3.2	105	6.6	119	5.0	105	2.1	116	1.8
106	4.9	133	6.3	106	2.5	42	3.2	106	4.5	77	5.0	106	1.7	123	1.8
107	7.3	103	6.3	107	2.9	51	3.2	107	6.3	146	5.0	107	1.8	126	1.8
108	6.7	115	6.4	108	2.5	23	3.3	108	5.8	110	5.1	108	1.8	125	1.8
109	7.2	128	6.4	109	3.6	49	3.3	109	6.1	114	5.1	109	2.5	103	1.8
110	6.5	137	6.4	110	3.2	144	3.3	110	5.1	133	5.1	110	2.0	138	1.8
111	6.4	132	6.4	111	2.7	124	3.3	111	5.3	149	5.1	111	1.9	70	1.8
112	6.8	51	6.4	112	3.0	100	3.3	112	5.5	101	5.1	112	2.1	149	1.8
113	5.7	74	6.4	113	2.5	56	3.3	113	5.0	83	5.1	113	2.0	108	1.8
114	5.8	111	6.4	114	2.8	148	3.4	114	5.1	141	5.1	114	2.4	107	1.8
115	6.4	116	6.5	115	3.2	136	3.4	115	5.3	142	5.1	115	2.3	137	1.8
116	6.5	147	6.5	116	3.0	11	3.4	116	5.5	147	5.2	116	1.8	146	1.9
117	7.7	110	6.5	117	3.8	6	3.4	117	6.7	145	5.2	117	2.2	130	1.9
118	7.7	104	6.5	118	2.6	7	3.4	118	6.9	111	5.3	118	2.3	142	1.9
119	6.0	54	6.5	119	2.2	31	3.4	119	5.0	115	5.3	119	1.5	101	1.9
120	6.9	58	6.6	120	3.2	20	3.4	120	5.7	148	5.4	120	2.3	111	1.9
121	5.6	75	6.6	121	2.8	26	3.4	121	4.9	139	5.4	121	2.0	131	2.0
122	7.7	86	6.7	122	2.8	24	3.4	122	6.7	137	5.5	122	2.0	113	2.0
123	6.3	65	6.7	123	2.7	39	3.4	123	4.9	112	5.5	123	1.8	147	2.0
124	6.7	77	6.7	124	3.3	28	3.4	124	5.7	116	5.5	124	2.1	110	2.0
125	7.2	144	6.7	125	3.2	85	3.4	125	6.0	134	5.6	125	1.8	122	2.0
126	6.2	145	6.7	126	2.8	17	3.5	126	4.8	136	5.6	126	1.8	121	2.0
127	6.1	124	6.7	127	3.0	43	3.5	127	4.9	132	5.6	127	1.8	102	2.1
128	6.4	108	6.7	128	2.8	27	3.5	128	5.6	140	5.6	128	2.1	139	2.1
129	7.2	140	6.7	129	3.0	0	3.5	129	5.8	103	5.6	129	1.6	105	2.1
130	7.4	112	6.8	130	2.8	40	3.5	130	6.1	128	5.6	130	1.9	128	2.1
131	7.9	76	6.8	131	3.8	36	3.5	131	6.4	144	5.7	131	2.0	124	2.1
132	6.4	143	6.8	132	2.8	22	3.6	132	5.6	124	5.7	132	2.2	112	2.1
133	6.3	52	6.9	133	2.8	4	3.6	133	5.1	120	5.7	133	1.5	104	2.2
134	6.1	139	6.9	134	2.6	109	3.6	134	5.6	129	5.8	134	1.4	132	2.2
135	7.7	120	6.9	135	3.0	48	3.7	135	6.1	108	5.8	135	2.3	117	2.2
136	6.3	141	6.9	136	3.4	21	3.7	136	5.6	104	5.8	136	2.4	143	2.3
137	6.4	50	7.0	137	3.1	10	3.7	137	5.5	102	5.9	137	1.8	141	2.3
138	6.0	102	7.1	138	3.0	46	3.8	138	4.8	143	5.9	138	1.8	148	2.3
139	6.9	125	7.2	139	3.1	44	3.8	139	5.4	100	6.0	139	2.1	118	2.3
140	6.7	109	7.2	140	3.1	18	3.8	140	5.6	125	6.0	140	2.4	120	2.3
141	6.9	129	7.2	141	3.1	117	3.8	141	5.1	130	6.1	141	2.3	115	2.3
142	5.8	107	7.3	142	2.7	131	3.8	142	5.1	135	6.1	142	1.9	145	2.3
143	6.8	130	7.4	143	3.2	19	3.8	143	5.9	109	6.1	143	2.3	135	2.3
144	6.7	105	7.6	144	3.3	5	3.9	144	5.7	107	6.3	144	2.5	136	2.4
145	6.7	118	7.7	145	3.0	16	3.9	145	5.2	131	6.4	145	2.3	114	2.4
146	6.3	122	7.7	146	2.5	14	4.0	146	5.0	105	6.6	146	1.9	140	2.4
147	6.5	117	7.7	147	3.0	32	4.1	147	5.2	122	6.7	147	2.0	100	2.5
148	6.2	135	7.7	148	3.4	33	4.2	148	5.4	117	6.7	148	2.3	109	2.5
149	5.9	131	7.9	149	3.0	15	4.4	149	5.1	118	6.9	149	1.8	144	2.5