UNIVERSIDADE FEDERAL DO PARANÁ

Mestrado em Informática

ÁRVORES DE DECISÃO FUZZY NA MINERAÇÃO DE IMAGENS DO SISTEMA FOOTSCANAGE

ANDERSON VIÇOSO DE ARAÚJO

CURITIBA

2006

ANDERSON VIÇOSO DE ARAÚJO

ÁRVORES DE DECISÃO FUZZY NA MINERAÇÃO DE IMAGENS DO SISTEMA FOOTSCANAGE

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientadora: Profª. Drª. Olga R. P. Bellon
Co-orientador: Prof. Dr. Luciano Silva

CURITIBA

2006

ANDERSON VIÇOSO DE ARAÚJO

ÁRVORES DE DECISÃO FUZZY NA MINERAÇÃO DE IMAGENS DO SISTEMA FOOTSCANAGE

Dissertação aprovada como requisito parcial à obtenção do grau de Mestre no Programa de Pós-Graduação em Informática da Universidade Federal do Paraná, pela Comissão formada pelos professores:

Orientadora: Profª. Drª. Olga R. P. Bellon
Departamento de Informática, UFPR

Profª Drª Heimar de Fátima Marin
UNIFESP

Prof. Dr. Nelson Delfino d'Ávila Mascarenhas
UFSCAR

Prof. Dr. Alexandre I. Direne
Departamento de Informática, UFPR

Curitiba, 19 de Abril de 2006

AGRADECIMENTOS

Gostaria muito de agradecer aos meus pais: Carlos e Rose; minhas irmãs: Andrezza e Ana Vitória; e minha tia Sueli, que apesar de distantes, contribuíram com apoio, incentivo e sempre dando forças para que eu conseguisse chegar ao final deste trabalho.

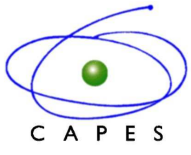
Destaco também a colaboração de amigos, colegas de curso, amigos do grupo IMAGO, destacando: Everton, Emanuel, Leandro, Rubisley e Jorge, que tiveram grande participação neste projeto, principalmente os dois últimos devido a implementação da ferramenta de processamento de imagens. Não esquecendo de agradecer os amigos Daniel, Isaías, Nélson e Dona Diva, que fizeram da minha estadia em Curitiba a melhor possível.

Sou grato também aos professores, Dr. Alexandre I. Direne, pela atenção, ensinamentos e sugestões; e Drª Mônica Cat pelo auxílio na obtenção dos dados e ajuda na interação entre o Departamento de Informática e o Hospital de Clínicas da UFPR.

Obrigado à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), pelo apoio financeiro, possibilitando assim uma dedicação exclusiva ao curso.

É necessário registrar também a importância da Financiadora de Estudos e Projetos (FINEP) e do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPQ) dando suporte ao desenvolvimento de todo Projeto FootScanAge.

Também tenho uma gratidão muito especial pela professora Drª Olga Regina Pereira Bellon pela atenção, orientação, dedicação e pelas oportunidades a mim oferecidas. Ao professor Dr. Luciano Silva, pela co-orientação, disposição e auxílio no direcionamento do trabalho, meus sinceros agradecimentos.



SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

RESUMO

A correta determinação da idade gestacional é um dos principais recursos utilizados pelos especialistas de neonatologia para orientá-los nos diagnósticos e medidas terapêuticas para garantir a sobrevida de recém-nascidos. Neste contexto, o sistema FootScanAge se apresenta como uma alternativa aos métodos atuais por ser um método não-invasivo baseado em características extraídas de imagens da superfície plantar de neonatos.

Neste trabalho é apresentado o sistema FootScanAge e suas funcionalidades, enfatizando o uso de algoritmos de mineração de dados conhecidos como algoritmos de árvores de decisão *fuzzy*, que fornecem uma distribuição de probabilidades mais concisa sobre as idades gestacionais possíveis para cada neonato. Esses algoritmos são aplicados sobre uma base de características extraídas dessas imagens, fornecendo várias informações para auxiliar no cálculo final da idade gestacional, validado por um especialista.

ABSTRACT

The correct gestational age determination is one of the most used measures to guide diagnostics of the specialists in neonatology to guarantee the newborn surveillance. In this context, the FootScanAge system presents a new alternative to current methods, bringing a non-invasive method that is based on features extracted from newborns plantar surface images.

This work presents the FootScanAge tool and its functionalities, emphasizing the use of data mining algorithms known as fuzzy decision tree algorithms. These algorithms are applied on a database of image features, providing results and showing how image mining can help the determination of the final gestational age score, validated by an expert.

CAPÍTULO 1

INTRODUÇÃO

Atualmente é grande a quantidade de instituições, empresas e pessoas que têm realizado grandes investimentos na geração, processamento e publicação de imagens digitais, mas não apenas imagens, como também áudio e vídeo, isto é, dados multimídia. Tal cenário vem gerando um crescimento explosivo nos acervos e bancos de dados multimídia, que utilizam as características da internet para impulsionar ainda mais esse crescimento. Tais agrupamentos superam muito a usual capacidade de interpretar e analisar esses dados. Através de avanços científicos, algumas áreas de pesquisa naturalmente geram ainda mais dados, tais como: Visão Computacional, Medicina, Geoprocessamento, entre outras.

A mineração de dados em bancos de imagens vem ao encontro dessas questões, através de técnicas, algoritmos e sistemas capazes de extrair conhecimento (informações úteis), relacionamentos espaciais e padrões interessantes não explícitos em grandes acervos de imagens. Os investimentos, a disponibilidade e a demanda sobre esses acervos de imagens, aliados à impossibilidade humana de análise manual e individual destes dados, motivam a pesquisa deste tema em direção ao fomento de novas ferramentas para análise automática (ou semi-automática) e inteligente de bancos de dados de imagens.

Focando na área médica, percebe-se que existe um grande acúmulo de textos e imagens dentro dos hospitais e clínicas. Esses dados estão sendo apenas armazenados e não estão sendo analisados utilizando alguma técnica computacional, como a mineração de dados e imagens, por exemplo. Por outro lado, existem vários exemplos de aplicações de diferentes técnicas de processamento de imagens, que beneficiam áreas da medicina, onde as imagens médicas têm papel fundamental [74].

Resumidamente, este trabalho apresenta a implementação de um sistema que utiliza mineração de imagens para extrair informações de um conjunto de imagens. Através dessa mineração, o sistema fornece a resposta correspondente a imagem de entrada.

A resposta gerada pelo sistema, é a idade gestacional (IG) de um recém-nascido e a imagem de entrada é uma imagem digitalizada da superfície plantar do recém-nascido correspondente. A IG é um termo bastante utilizado pelos neonatologistas que corresponde ao tempo de gestação, ou seja, tempo em semanas desde a concepção ao nascimento.

Quando se fala de idade gestacional, percebe-se que mesmo com a evolução da ciência e da medicina, os métodos para sua determinação, não foram capazes de acompanhar essa evolução, mostrando-se em alguns casos, deficientes, imprecisos, exigindo muita manipulação dos neonatos e, além disso, foram baseados em pesquisas sobre populações que envolviam poucos prematuros [14]. Baseado em estudos iniciais que demonstraram a importância das dimensões do pé e de suas sub-regiões, bem como, informações sobre cisuras e poros [9], foi desenvolvido o método FootScanAge. Este método pode ser considerado a evolução dos métodos de determinação de idade-gestacional, mostrando-se não-invasivo e menos subjetivo que os demais métodos ao utilizar como referência apenas as características da superfície plantar de neonatos para estimar uma idade gestacional.

Para servir de ferramenta ao método, foi desenvolvido o sistema FootScanAge, que combina Mineração de Dados e Processamento de Imagens, em uma técnica conhecida como Mineração de Imagens [83]. Trata-se de um sistema de suporte à decisão que permite a manipulação de uma base de características extraídas de imagens da superfície plantar de pés de recém-nascidos que são analisados através de algoritmos de mineração de dados. Em um trabalho anterior, Severich [69] iniciou os estudos sobre o processamento das imagens da superfície plantar de recém-nascidos, desenvolvendo uma ferramenta para o processamento dessas imagens. Existem alguns outros trabalhos que seguem uma abordagem similar, por exemplo, Antonie [4] mostra a classificação de imagens para identificar câncer de mama.

O algoritmo principal a ser apresentado nesse trabalho é baseado na lógica *fuzzy*. Os conjuntos *fuzzy* e a lógica *fuzzy* permitem a modelagem de incertezas relativas ao espaço de hipóteses e por isso, a representação *fuzzy* está se tornando popular ao lidar com problemas com certo grau de ruído.

O objetivo é agrupar a representação *fuzzy*, com sua capacidade racional aproximada

e as árvores de decisão, para manter as vantagens de ambos: tratamento de incertezas e processamento gradual com compreensão, popularidade e facilidade de aplicação. Isto vai aumentar o poder representativo e como conseqüência a aplicabilidade das árvores de decisão, anexando a elas um conhecimento adicional baseado na representação *fuzzy*.

Também é importante deixar claro que todo o sistema desenvolvido foi gerado utilizando ferramentas livres, sendo que, os desenvolvedores do sistema apóiam a criação de sistemas baseados em *software* livre, tanto que o sistema FootScanAge foi desenvolvido como sendo uma ferramenta livre. Destacando também a importância do projeto FootScanAge devido a sua indicação para o prêmio Banco do Brasil de Tecnologia Social em 2005[1].

1.1 Objetivos

O objetivo principal deste trabalho é a implementação de um algoritmo de aprendizado de máquina que utiliza árvores de decisão *fuzzy*, aplicado sobre um conjunto de dados extraídos de imagens. Embora eficiente na tomada de decisões em diversas áreas, as árvores de decisão padrão são inadequadas para expressar incertezas e ambigüidades. Variações superficiais nos valores de um atributo podem resultar em uma classificação diferente e/ou inesperada [62]. Para lidar com esta dificuldade, o emprego de árvores de decisão juntamente com a teoria dos conjuntos *fuzzy* tem sido constantemente realizado. Com esse algoritmo, é possível gerar uma distribuição de probabilidades mais concisa, dando assim um melhor conjunto de possibilidades para auxiliar na decisão final do especialista.

Um objetivo secundário é mostrar e discutir conceitos muito importantes dentro da computação atual, tais como: Mineração de imagens e dados, algoritmos de aprendizado de máquina, técnicas de processamento de imagens e visão computacional, entre outros. Também é apresentada a interligação entre esses conceitos, como e porque eles devem ser utilizados.

Para exemplificar todos os conceitos citados, é focada a descrição do processo de mineração de imagens do sistema FootScanAge, mostrando toda sua arquitetura, os métodos

[1] http://www.fundacaobancodobrasil.org.br

e algoritmos utilizados para chegar aos melhores resultados para auxílio na definição da idade gestacional para um neonato. Mostrando também detalhes da implementação e utilização do sistema.

1.2 Organização

Inicialmente, no capítulo 2, é apresentada uma breve descrição do projeto FootScanAge e dos métodos de descoberta de idade gestacional utilizados atualmente.

No capítulo 3, são introduzidos conceitos utilizados dentro do sistema, tais como: mineração de imagens, processamento de imagens, visão computacional, bancos de dados multimídia, entre outros.

Os principais algoritmos de aprendizado de máquina, principalmente os algoritmos utilizados na implementação do sistema FootScanAge se encontram no capítulo 4.

A arquitetura do sistema de mineração de imagens do sistema será apresentada no capítulo 5, assim como será descrita cada etapa da mineração dentro do projeto, as estratégias de apresentação e avaliação dos resultados para auxiliar no cálculo da idade gestacional final.

Finalmente no capítulo 6, encontra-se a conclusão e os trabalhos futuros relacionados a este trabalho e também ao sistema FootScanAge.

CAPÍTULO 2

O PROJETO FOOTSCANAGE

A neonatologia, especialidade pediátrica que cuida de recém-nascidos, é uma ciência que se baseia, quase que totalmente na estimativa, previsão e antecipação dos eventos para que as tomadas de decisões possam ser imediatas diante do reconhecimento do problema clínico, a fim de que o recém-nascido possa ser adequadamente tratado [3].

A idade gestacional é um aspecto importante dentro da neonatologia, orientando a assistência neonatal adequada ao estado fisiológico de desenvolvimento do recém-nascido, sendo em alguns casos, determinante dos limites de viabilidade.

Nas situações onde não se dispõe de informações precisas relativas aos exames pré-natais, a alternativa é a utilização de métodos pós-natais de estimativa da idade gestacional. Entretanto, esses métodos não foram baseados ou desenvolvidos em populações de recém-nascidos prematuros, dificultando assim sua reprodutibilidade e precisão nessa população. Algumas dificuldades na aplicação dos escores[1] pós-natais de estimativa da idade gestacional têm sido relatados, especialmente entre recém-nascidos prematuros e/ou doentes [14].

Em geral, todos os métodos pós-natais dependem de um certo grau de subjetividade e experiência em neonatologia, especialmente em relação aos itens neurológicos dos escores. Reconhecendo esta importância, deu-se início as pesquisas que constituem o que foi denominado de Projeto FootScanAge para Determinação da Idade Gestacional [14], com o objetivo de determinar a idade gestacional através da morfologia da superfície plantar do recém nascido e do processamento da sua imagem digital, obtida através do borrão (i.e. tinta sobre a impressão plantar pressionada contra papel. A seção 5.3 contém mais detalhes sobre a obtenção das imagens) adquirido rotineiramente por profissionais na sala de parto. A figura 2.1 mostra um exemplo de imagem de superfície plantar extraída, onde

[1]Estimativa de idade gestacional através de algum método.

o pé apresentado e o borrão não são necessariamente os mesmos. Através da cooperação entre os Departamentos de Informática e de Pediatria da Universidade Federal do Paraná, deu-se início ao projeto, com o desenvolvimento de um sistema de processamento de imagens, com o objetivo de analisar, através de visão computacional, as características morfológicas da superfície plantar.

(a) (b)

Figura 2.1: Exemplo de superfícies plantares: (a) Foto da superfície plantar de um recém-nascido; (b) Imagem digitalizada a partir borrão da superfície plantar.

De acordo com os nascimentos ocorridos a partir de agosto de 2000 dentro do Hospital de Clínicas da UFPR, foi iniciada a digitalização das imagens correspondentes a superfície plantar dos recém-nascidos (figura 2.2). Inicialmente, a aquisição das imagens foi difícil, pois necessitava-se de uma certa experiência e padronização na metodologia para a obtenção das imagens. Com o passar do tempo a qualidade e a quantidade de imagens foi aumentando significativamente, sendo assim, depois de algum tempo de coleta de dados, possível a aplicação e validação do método sobre o conjunto de imagens.

Figura 2.2: Exemplos de pés de recém-nascidos [14].

2.1 Métodos para estimativa de idade gestacional

Existem vários métodos e estudos ligados a estimativa da idade gestacional de neonatos. A tabela 2.1 são apresentados alguns desses métodos utilizados para comparação com o método FootScanAge, a ser descrito posteriormente, detalhando o método, a data de criação e o número de critérios envolvidos. A maioria dos métodos envolvem a análise de vários critérios subjetivos, que na maioria das vezes são difíceis de quantificar e analisar.

Ano	**Método**	**Critérios**
1970	Dubowitz [22]	21 critérios (10 físicos e 11 neurológicos)
1970	Ballard [7]	12 critérios (6 físicos e 6 neurológicos)
1976	Parkin [58]	4 critérios
1978	Capurro [12]	6 critérios
1991	New Ballard [6]	12 critérios antigos + 2 novos
2003	FootScanAge [14]	1 critério (imagem da superfície plantar)

Tabela 2.1: Histórico de alguns métodos utilizados para estimativa da idade gestacional.

Parkin [58], Capurro [12] e New Ballard [6] são os métodos mais utilizados atualmente dentro dos hospitais e clínicas com a finalidade de estimar a idade gestacional de neonatos.

2.1.1 Idade gestacional padrão-ouro

O padrão-ouro de estimativa da idade gestacional, atualmente, consiste da associação das informações relacionadas à data da última menstruação fornecida pela gestante; e biometria fetal realizada no 1° trimestre de gestação, durante o atendimento pré-natal. Entretanto, muitas são as vezes onde estas informações não estão disponíveis, resultando no desconhecimento do tempo de gestação.

Para poder classificar corretamente as instâncias, existe a necessidade de se possuir uma idade gestacional padrão-ouro, ou seja, uma idade que corresponda exatamente com a idade gestacional do bebê, para que assim o classificador gerado pelo algoritmo de aprendizado não esteja se baseando em informações incorretas ou não existentes.

O maior problema da idade padrão-ouro é a dificuldade de conseguir todos os requisitos citados, sendo assim, pequeno o número de mulheres com este perfil, conseqüentemente,

sendo possível criar uma base com poucos exemplos, mas isso não seria interessante para a pesquisa e alcançar um número razoável de exemplos tomaria muito tempo.

As instâncias iniciais tomadas como entrada para a fase de extração de características não possuíam todas as características citadas acima, ou seja, não eram consideradas padrão-ouro; isto pode gerar um erro propagado sobre a idade gestacional estimada no final de todo o processo. Contudo, os dados existentes são extraídos de imagens e prontuários reais. Sendo assim, os resultados são correspondentes a resultados válidos e coerentes. Conseqüentemente todo o processo de mineração poderá ser executado novamente e com facilidade, quando as instâncias padrão-ouro estiverem disponíveis para utilização, e assim, tendo resultados ainda mais precisos.

CAPÍTULO 3

CONCEITOS

Dentro deste capítulo serão abordados os conceitos mais importantes relacionados com o projeto FootScanAge e sua implementação. O objetivo deste capítulo é abordar principalmente a descoberta de conhecimento, a mineração de imagens e o seu relacionamento com outras disciplinas, onde cada uma delas também será abordada superficialmente.

3.1 Processamento de imagens

O processamento de imagens é uma área da ciência da computação que agrupa técnicas que tem intenção de analisar imagens para uma certa finalidade e/ou tentam modificá-la até alcançar o resultado desejado. A maioria dessas técnicas usualmente transformam imagens em outras imagens [38]. Realçar, comprimir e corrigir imagens são alguns dos tópicos inclusos dentro deste campo.

A melhoria da informação visual para a interpretação humana e o processamento de dados de cenas para a percepção automática, podem ser consideradas as duas principais áreas do processamento de imagens [31].

O reconhecimento de objetos também tem sido amplamente pesquisado no campo do processamento de imagens. Modelos de objetos fornecidos *a priori* permitem que um sistema de reconhecimento encontre objetos do mundo real em imagens digitais.

A mineração de imagens é um exemplo do relacionamento entre o processamento de imagens e aprendizado de máquina. Outros relacionamentos são vistos mais adiante, assim como, a importância desses relacionamentos.

3.2 Visão computacional

Ao se discutir inteligência artificial juntamente com processamento de imagens, o conceito mais correto a ser utilizado é a visão computacional. Ele envolve várias atividades, desde a captura e processamento das imagens até o reconhecimento de objetos de um cenário, como também a interpretação de uma imagem em conjunto com aspectos psicológicos e fisiológicos da visão. Então, o seu objetivo pode ser definido como em [70]: "O objetivo da visão computacional é tomar decisões úteis sobre objetos e cenas reais baseadas em imagens".

Mesmo com a distância entre o conhecimento a ser modelado, a descrição do mundo real e a representação das imagens, o desafio da compreensão das imagens permanece e tem sido bastante explorado por sistemas de Visão Computacional.

3.3 Banco de dados multimídia

Khoshafian [41] define bancos de dados multimídia como sendo: "Bancos com gerenciadores de bases de dados de alta capacidade e performance que suportam vários tipos de multimídia, como também tipos alfa-numéricos e manipulam grandes volumes de informação".

Dentro do contexto apresentado neste trabalho, percebe-se que uma das diferenças principais entre mineração de dados e aprendizado de máquina, que serão definidos no decorrer do capítulo, é a preocupação que a mineração tem com a quantidade de dados a ser pesquisada. Por isso, a mineração de dados depende dos mecanismos de pesquisa, armazenamento e implementação usados em Sistemas Gerenciadores de Bancos de Dados (SGBDs). Esses sistemas possuem ferramentas de *backup*, linguagens de consulta, controle transacional, controle de integridade, mecanismos de segurança e estruturas de indexação para permitir flexibilidade, robustez e performance no acesso a dados [43]. Em especial, determinados aspectos de um banco de dados são bastante relevantes para técnicas de mineração: paradigmas de banco de dados, linguagens de consulta e indexação. As bases de imagens fazem parte de uma disciplina maior que são os bancos de dados multimídia.

3.4 *Data Warehouse*

Pode-se definir um *Data Warehouse* (DW) como um repositório integrado, orientado para análise, histórico, com dados apenas para leitura, designado para ser utilizado como base para suporte à decisão e descoberta de conhecimento [37, 61].

Um DW funciona como uma base de dados para dar suporte à decisão mantido separadamente das bases de dados operacionais da organização. Geralmente integra dados de diversas origens heterogêneas e por isso necessita de uma estrutura flexível que suporte consultas e geração de relatórios analíticos.

Geralmente dados históricos, variantes no tempo, ficam armazenados em um DW por muitos anos. Esses dados são geralmente organizados de modo a facilitar sua análise por um usuário especializado. Uma organização típica de dados é armazenar informações quantitativas (por exemplo, vendas de produtos) em grandes tabelas, chamadas tabelas de fatos; e dados qualitativos, informação descritiva (por exemplo, atributos do produto) armazenados em pequenas tabelas, chamadas tabelas de dimensão.

A base de dados de um DW serve apenas para leitura, no sentido que um item dessa base é raramente alterado. Em um DW o usuário obtém a informação desejada executando consultas pré-definidas que fazem junções entre as tabelas de fatos e dimensões. Atualizações no DW geralmente consistem na inserção de novos dados (e as vezes da retirada de dados mais antigos) num período pré-determinado de tempo.

3.5 Recuperação de imagens por conteúdo

Na recuperação de imagens por conteúdo [19], o usuário descreve o conteúdo desejado em termos de características visuais e o sistema recupera as imagens que melhor correspondem a descrição. Esses sistemas responsáveis por essa busca por similaridade são chamados *SRICs* (Sistemas de Recuperação de Imagens por Conteúdo).

Os SRICs são sistemas caracterizados pela pesquisa, consulta e recuperação de dados visuais, normalmente utilizando atributos de baixo nível de uma imagem, tais como: cor, textura, forma e informações semânticas (palavras-chave e anotações descritivas).

Os SRICs são usados em conjunto com os bancos de dados multimídia para indexar [13] as imagens existentes no banco de dados a ser pesquisado.

Um dos maiores desafios dos SRICs é conhecido como diferença semântica (*semantic gap*) [32], que é a discrepância entre o que os sistemas provêem e o que é desejado pelo usuário. Esta diferença limita a eficiência dos sistemas de recuperação de imagens por conteúdo.

Smeulders tem realizado estudos bastante aprofundados sobre recuperação de imagens por conteúdo nos últimos anos [73].

3.6 Lógica *fuzzy*

Para resolver problemas em que temos um evento que não é definido precisamente, Lofti Zadeth, propôs em 1965 a Teoria dos Conjuntos Difusos ou Lógica *fuzzy* [80].

O objetivo da lógica *fuzzy* é modelar aproximadamente o modo de raciocínio humano, visando desenvolver sistemas computacionais capazes de tomar decisões racionais em um ambiente de incerteza e imprecisão. A lógica *fuzzy* oferece um mecanismo para manipular informações imprecisas, tais como os conceitos de muito, pouco, pequeno, alto, bom, quente, frio, etc, fornecendo uma resposta aproximada para uma questão baseada em um conhecimento que é inexato, incompleto ou não totalmente confiável.

Conjuntos e elementos são duas noções primitivas da teoria dos conjuntos. Na teoria clássica, ou um elemento pertence a um certo conjunto ou não. Então, o grau de pertinência de um elemento a um conjunto é binário {0,1}. Entretanto, no mundo real isso geralmente não acontece, por causa das medidas imprecisas, ruídos, informações ausentes, subjetividade, etc. Por causa destes problemas, sistemas que usavam atributos simbólicos, iniciaram a inclusão de componentes numéricos [18].

Os Conjuntos *fuzzy* provêem bases para a representação *fuzzy*. A teoria *fuzzy*, define uma função de pertinência (μ), que indica o quanto o elemento u (pertencente a um Universo U) pertence a um conjunto V, numa escala entre 0 e 1: $\mu_u(V) : U \rightarrow [0, 1]$. Juntamente com essa função, são definidos todos os operadores que atuam no conjunto [81], assim como, toda a base lógica que precisa existir para sua utilização prática.

Esta teoria tem grande uso em tarefas de classificação e agrupamento principalmente em problemas na área de medicina e também em algoritmos de segmentação.

Para exemplificar a teoria dos conjuntos difusos, consideremos, por exemplo, a definição de "jovem". Para algumas pessoas alguém de 25 anos é jovem enquanto para outras pessoas 35 anos é ainda jovem. Embora seja difícil estabelecer um limite para o conceito de jovem, alguns consensos podem ser estabelecidos como o fato de alguém de 1 ano ser definitivamente (100%) jovem e alguém de 100 anos não ser nada (0%) jovem. Assim, uma pessoa de 35 anos pode ter um grau de aderência ao conceito "jovem" entre 0 e 100%. Como os conjuntos *fuzzy* estabelecem funções de pertinência onde elementos podem pertencer parcialmente a um ou mais conjuntos, o termo *fuzzy* "jovem" pode ser descrito pelo conjunto *fuzzy* descrito na tabela 3.1.

Idade	Grau de Pertinência
25	1
30	0,8
35	0,6
40	0,4
45	0,2
50	0

Tabela 3.1: Função de pertinência para os conjuntos do exemplo para "jovem".

Através de uma função *fuzzy*, é possível definir conceitos como "pouco jovem" ou "muito jovem" podem ser obtidos aplicando operações matemáticas ao conjunto *fuzzy* "jovem" onde "pouco" e "muito" podem ser definidos por funções matemáticas.

3.7 Descoberta de conhecimento

O descobrimento de conhecimento em bancos de dados, ou em inglês *Knowledge Discovery in Databases* (KDD), pode ser definido como o processo não trivial de identificar padrões válidos, novos, potencialmente úteis e compreensíveis [27]. A descoberta de conhecimento também pode ser definida como um conjunto de técnicas automáticas utilizadas para explorar grandes bases de dados de forma a descobrir relações e padrões existentes nestas informações [33].

O processo consiste em vários passos: seleção, pré-processamento, transformação, mineração, interpretação dos dados e avaliação dos resultados. Essas fases são apresentadas na figura 3.1 em sua seqüência de execução. Todas as fases são melhor descritas a seguir e mais tarde são comparadas com as etapas do processo de mineração de imagens.

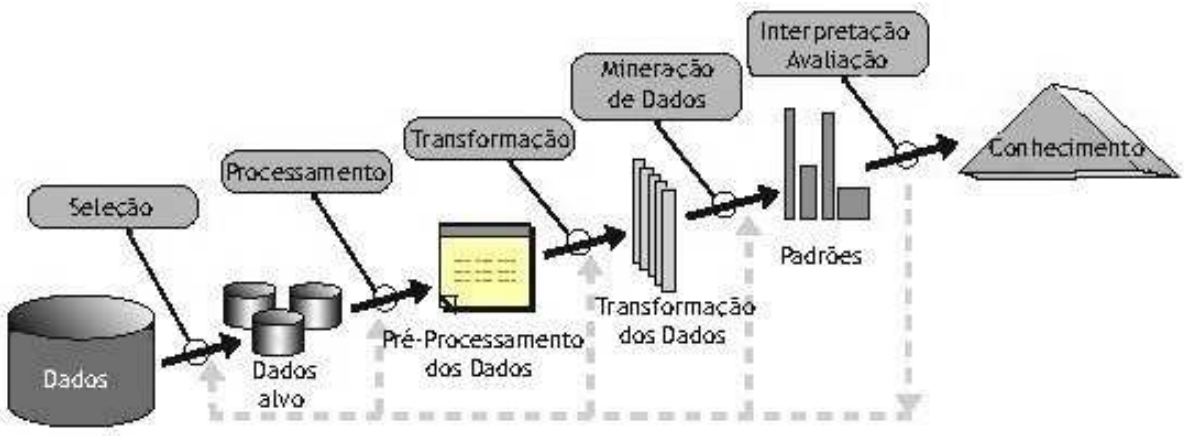


Figura 3.1: Fases do KDD.

Dentro da fase de mineração de dados do KDD são aplicados algoritmos de mineração de dados para encontrar padrões sendo que o conjunto desses padrões também pode ser chamado de "conhecimento".

Embora algoritmos de aprendizado de máquina são a parte central para o processo da mineração de dados, é importante notar que o processo também envolve outros passos importantes, incluindo construir e manter o banco de dados, formatar e limpar os dados, visualização e indexação dos dados, o uso do conhecimento humano para formular as entradas para o algoritmo de aprendizado, avaliar as regularidades empíricas que ele descobre e determinar como distribuir sistematicamente os resultados [52]. Além disso, a mineração de dados aborda várias áreas, incluindo: bancos de dados, interação humano-computador, estatística e algoritmos de aprendizado de máquina. O foco aqui é o papel dos algoritmos de aprendizado de máquina no processo de mineração de dados.

Antes de entrar nas fases do processo de KDD, deve-se iniciar a construção de um *Data Warehouse*, sendo ele um meio efetivo de organizar grandes volumes de dados para sistemas de suporte a decisão e aplicações de KDD.

3.7.1 Fases do KDD

Para iniciar a descoberta de conhecimento é realizada a seleção. Nesta etapa, são definidos os dados realmente importantes para a extração de informações e que tipo de conhecimento deve ser extraído do conjunto de dados, que geralmente está armazenado em um DW (seção 3.4).

Em seguida, é realizado um pré-processamento dos dados, onde se deseja reduzir a complexidade do problema. Essa fase também é responsável pela limpeza dos dados. As inconsistências são identificadas, *outliers* são eliminados e os atributos mais significativos são selecionados, reduzindo assim a dimensionalidade do problema e evitando que variáveis correlatas sejam utilizadas na análise.

Já na fase de transformação dos dados, busca-se codificar e transformar os dados para facilitar o restante do processo de mineração. Em suma, é qualquer processo capaz de melhorar o desempenho do algoritmo minerador. Por exemplo, a discretização de variáveis contínuas em faixas de valores pode ser considerada uma transformação. A normalização dos valores de uma variável contínua, que faz com que os valores dessa variável passem a se encontrar dentro do intervalo $[0, 1]$, também é considerada uma transformação.

A mineração de dados (ou *Data Mining*) é a principal etapa do processo de KDD, pois é a responsável por extrair informações implícitas a partir dos dados.

Alguns autores consideram a mineração de dados um passo do processo de KDD, outros consideram como sendo o processo inteiro. Nesta dissertação, a mineração de dados é encarada como uma fase do processo do KDD, que se utiliza de algoritmos de aprendizado de máquina para descobrir informações contidas no conjunto de dados.

Esse passo, segundo Fayyad [25] consiste em fazer a análise dos dados e aplicar algoritmos de aprendizado de máquina, sob limitações computacionais aceitáveis, para produzir um conjunto de padrões sobre os dados. Esse processo é automático ou semi-automático, ou seja, com ou sem a interação do usuário. Ao aplicar algoritmos de aprendizado de máquina, esses analisam os dados e retornam os resultados encontrados. Os resultados podem ser uma informação nova, uma descoberta ou uma resposta para um certo problema: um número, uma classe, ou apenas, sim ou não.

A mineração de dados é proveniente da técnica de aprendizado de máquina e atualmente diversas técnicas são usadas para realizar essa tarefa, sendo que algumas delas serão mencionadas no próximo capítulo.

As informações resultantes são utilizadas na última fase do processo, na fase de avaliação e interpretação. Nessa fase, deve-se identificar o que e qual a importância do que foi extraído do conjunto de dados e, os resultados finais devem ser usados pelos usuários para análises e principalmente para tomada de decisões estratégicas. Caso necessário, é importante o auxílio de algum especialista na área de estudo, para melhor interpretar o conhecimento extraído, como por exemplo, um biólogo quando analisa resultados obtidos através de um experimento realizado sobre um conjunto de bactérias.

3.8 Mineração de imagens

A mineração de imagens é tratada como sendo a combinação entre mineração de dados e processamento de imagens. Na mineração de imagens o objetivo é descobrir padrões que são significantes em um dado conjunto de imagens. Zhang [83] mostra que a mineração de imagens lida com a extração de conhecimento implícito, relacionamento entre dados de imagens ou outros padrões não explicitamente armazenados nos banco de dados de imagens. O foco da mineração de imagens é determinar como a representação de baixo nível (*pixels*[1]) de uma imagem pode ser eficientemente e efetivamente processada para identificar objetos e relacionamentos.

As fases da mineração de imagens possuem objetivos similares as fases da descoberta de conhecimento, já descritas na figura 3.1. As diferenças são:

- Seleção: Selecionar as imagens a serem utilizadas dentro da mineração;
- Pré-processamento e transformação: Tratar e alterar as imagens;
- Extração de características: Uma nova fase, não existente na descoberta de conhecimento, onde são obtidas informações das imagens;

[1]Uma abreviação para *picture element*. É o menor elemento mostrado na tela, representado como um ponto correspondente a uma certa cor ou nível de intensidade [31].

- Mineração de dados: Utilizar algoritmos de aprendizado de máquina sobre as características adquiridas das imagens;

- Interpretação e avaliação: Possuem objetivos similares à mineração de dados.

A extração de características é uma fase exclusiva da mineração de imagens, ou seja, ela não se encontra na mineração de dados comum. A extração é responsável por adquirir informações das imagens, tais como informações sobre: os objetos contidos na imagem e suas formas, cores, textura, etc. As informações extraídas serão mais tarde utilizadas como entrada na fase de mineração dos dados e tratadas como se fossem dados comuns vindos de qualquer base de dados.

A mineração de imagens é uma área de pesquisa multidisciplinar que possui ligações com várias outras áreas de conhecimento tais como: visão computacional, processamento de imagens, mineração de dados, bancos de dados, aprendizado de máquina, Sistemas de Recuperação de Imagens por Conteúdo, entre outras. Essa integração é mostrada na figura 3.2.

Existem importantes diferenças entre bancos de dados relacionais e bancos de dados de imagens e isso geralmente cria um mal entendido na definição de mineração de imagens, onde é dito que a mineração de imagens não é nada mais do que aplicar algoritmos de mineração de dados em imagens.

Em um sistema de mineração de imagens a abordagem pode ser dirigida de duas maneiras diferentes: à função ou à informação. A dirigida à função é uma abordagem onde o sistema é organizado de acordo com as funções de cada módulo: aquisição, pré-processamento, mineração, etc. Já a abordagem dirigida à informação se baseia nas nos níveis de representação da informação (*pixels*, objetos, relacionamentos, etc). Esses conceitos são melhores descritos no próximo tópico.

Em se falando de ferramentas de mineração de imagens, Zaiane apresenta o "Multi-MediaMiner" em [82]. Essa ferramenta de mineração de imagens foi criada com base na experiência de outras aplicações e possui uma divisão das tarefas dentro do sistema em módulos. Os módulos são: *MM-Characterizer*, *MM-Associator* e *MM-Classifier*.

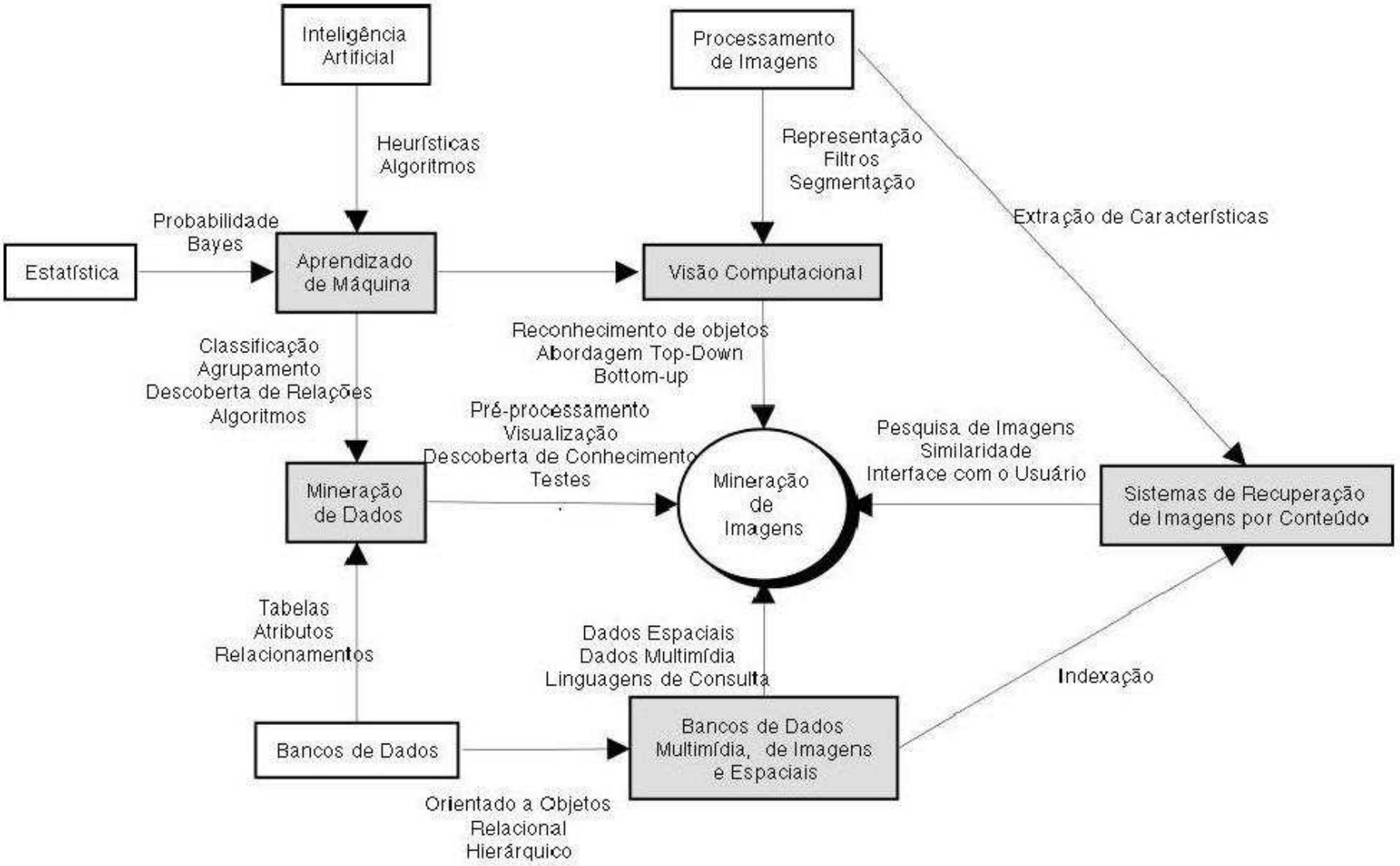


Figura 3.2: Integração entre a mineração de imagens e outras áreas [76, 75].

- *MM-Characterizer*: O módulo descobre um conjunto de características de vários níveis de abstração através de um banco de dados multimídia;
- *MM-Associator*: Encontra um conjunto de regras de associação relevante do conjunto de dados (imagens ou vídeo);
- M*M-Classifier*: Classifica os dados multimídia baseado em alguns rótulos fornecidos. O resultado é a classificação de um grande conjunto de dados multimídia e um descritor de característica para cada classe.

3.8.1 Arquitetura de um sistema de mineração de imagens

A estrutura da arquitetura de um sistema de mineração de imagens pode ser de várias formas diferentes. Os conceitos relativos a abordagem e a modularidade da mineração de imagens, são descritos a seguir.

3.8.1.1 Abordagem: orientada a modelo × orientada a dados

Na abordagem orientada a modelo, ou *top-down*, o modelo orienta a extração de características de baixo nível da imagem e as valida na mesma ou em outras imagens.

Já na abordagem orientada a dados, ou *bottom-up*, se baseia nas características de baixo nível da imagem, onde o processamento começa na própria imagem e termina no reconhecimento do objeto.

3.8.1.2 Modularidade: orientado à função × informação

Em um sistema de mineração de dados orientado à função, as descrições são orientadas a aplicação e a sua estrutura é organizada baseada na funcionalidade dos módulos (aquisição, mineração, interpretação e avaliação). A maioria dos sistemas de mineração de imagens e dados são orientados à função. Um problema desses sistemas é que eles falham ao enfatizar diferentes níveis de representação de informação.

Uma arquitetura orientada à informação, destaca a representação das informações descritas na imagem: *pixels*, objetos, análise semântica e padrões (conhecimento).

3.9 Aprendizado de máquina

Na fase do KDD conhecida como mineração de dados, são utilizados algoritmos de aprendizado de máquina para tentar descobrir conhecimento dentro do banco de dados. Pode-se definir "aprender" como Witten [78]: "Conseguir conhecimento por estudo, experiência ou por ser ensinado; tornar-se ciente por informação, observação ou receber instrução".

No aprendizado de máquina existem várias abordagens e técnicas. A classificação e a regressão são dois exemplos de abordagens diferentes utilizadas dentro do processo de aprendizado. Dentro da classificação, por exemplo, vários métodos de aprendizado podem ser utilizados, como: árvores de decisão, regras de indução, regras de associação, redes neurais, aprendizado de Bayes, aprendizado por agrupamento de classificadores, etc. Cada uma dessas abordagens têm suas características e funcionalidades específicas, onde o desempenho delas é muito dependente do problema e da base de dados correspondente.

As abordagens de aprendizado de máquina são melhor descritas dentro do capítulo 4, assim como alguns algoritmos relacionados.

3.9.1 Treinamento e teste

Em seguida é discutido como são realizados o treinamento e teste do aprendiz (classificador). Para treinamento e teste do algoritmo de aprendizado a ser utilizado, existem várias abordagens, como por exemplo:

- Utilizar uma base para treinamento e a mesma base para teste;
- Utilizar uma base para treinamento e outra base diferente para teste;
- Dividir a base original em duas partes: uma para treinamento e o restante para teste. Geralmente é utilizada 66% da base para treino e o restante para teste;
- Validação cruzada.

A validação cruzada pode ser considerada o método mais importante, por conseguir usufruir melhor da base de dados, criando classificadores diferentes para sub-conjuntos da base e selecionando assim o melhor deles. Esse método é melhor detalhado a seguir.

3.9.1.1 Validação cruzada

Em [67] é descrito o método de validação cruzada (ou *Cross-Validation*), que é um método para particionamento da base de dados, para um melhor aprendizado. O método consiste em decidir o número de conjuntos ou partições dos dados, dividir o conjunto de dados nessas N partes (ou *folds*), então aplicar o algoritmo de aprendizado sobre $N - 1$ partes para treinamento e sobre a parte restante para teste. Assim repetidamente para todas as partes e isto garante que todos os exemplos sejam utilizados para teste exatamente uma vez. Esta técnica também é chamada de *N-fold Cross-Validation*.

Como a validação cruzada é executada N vezes em conjuntos de dados diferentes, ela pode gerar N árvores de decisão diferentes. Daí, o algoritmo de validação cruzada escolhe a árvore que alcançou uma melhor porcentagem de acerto dentre as árvores geradas.

3.9.2 Modelos de mineração

Para aplicar os algoritmos de aprendizado de máquina é preciso treinar o algoritmo com uma base de treinamento, como descrito no tópico anterior. Depois de treinado ele deve ser armazenado, juntamente com todas as informações como: algoritmo utilizado (ou o agrupamento deles), o conjunto de atributos a serem minerados, pesos dos atributos, etc. Esse conjunto de informações é chamado de modelo de mineração ou modelo de aprendizado.

O modelo de mineração é utilizado para minerar as novas instâncias que farão parte da base de dados. Pode-se ter vários modelos armazenados e criar novos modelos de acordo com o crescimento da base, para que assim exista um modelo sempre atualizado de acordo com as informações mais recentes.

A mineração é composta de três etapas:

- Geração do modelo de mineração: trata-se do treinamento do algoritmo de aprendizado, através do método selecionado, como apresentado anteriormente;
- Seleção do modelo: apenas selecionar o modelo a ser utilizado. Podem existir mais de um modelo armazenado, com características e algoritmos diferentes, etc;

- Aplicação do modelo: aplicar o algoritmo sobre a nova instância a ser incluída na base de dados e analisar o resultado.

Geralmente depois de aplicado o modelo de mineração selecionado sobre uma nova instância, o resultado da mineração sobre essa instância também é armazenado para análises posteriores.

CAPÍTULO 4

ESTRATÉGIAS DE APRENDIZADO DE MÁQUINA

Em seguida, serão apresentadas as principais abordagens e técnicas utilizadas pelo processo de aprendizado de máquina que, como já foi dito anteriormente, é usado pela fase de mineração de dados e imagens. Este capítulo mostra como se deseja extrair informações relevantes de um banco de dados. Aqui não são exemplificados todos os algoritmos citados, mas características que indicam a melhor escolha do algoritmo a ser utilizado, o seu funcionamento e o tipo de resposta obtida.

Tanto o aprendizado por classificação, como por regressão, que são descritos a seguir, podem ser realizados utilizando várias outras abordagens que são melhor descritas no decorrer do capítulo.

4.1 Classificação

No aprendizado por classificação [78], o esquema de aprendizado toma um conjunto de exemplos já classificados e através dele se é esperado aprender uma forma de classificar exemplos não vistos. De forma mais simples, ele consiste em associar um conjunto de atributos a um atributo alvo (classe). Geralmente isso é realizado descobrindo-se regras, formas, padrões dentro dos dados e utilizar essas informações descobertas para classificar novos conjuntos de dados.

De maneira mais formal, a tarefa da classificação é: "Descobrir algum tipo de relacionamento entre valores de predição e o valor objetivo, tal que o conhecimento descoberto possa ser usado para prever a classe de uma nova tupla" [28].

Dentro do aprendizado de máquina, uma das abordagens mais utilizadas é a classificação, que também é chamada de aprendizado supervisionado, porque a entrada e a saída desejada são fornecidas previamente por um supervisor externo [24].

Um exemplo simples de classificação seria a partir de um conjunto de dados de carac-

terísticas de um certo grupo de pessoas (como idade, características físicas, etc), tentar classificar cada pessoa em sendo: bebê, criança, adulto ou idoso.

4.2 Regressão

Na regressão [78], também conhecida como predição numérica, o conhecimento a ser descoberto, ao contrário da classificação, não é uma classe discreta e sim um valor numérico, assim como os atributos do conjunto de dados. Esta técnica é bastante conhecida e utilizada dentro da estatística. Dentro da predição numérica, o método mais comum é a regressão linear, onde a idéia é tentar expressar a resposta como uma combinação linear dos atributos, usando pesos para eles:

$$x = w_0 + w_1 a_1 + w_2 a_2 + \ldots + w_k a_k \tag{4.1}$$

onde x representa o resultado, $a_1, a_2, \ldots, a_k$ são os valores para os atributos e $w_1, w_2, \ldots, w_k$ são os pesos. Os pesos podem ser calculados de várias formas através dos dados de treinamento.

A forma principal e mais utilizada para se avaliar a regressão é usar o erro quadrático médio (*Mean-Squared Error* - MSE), por ser considerado uma medida matematicamente mais fácil de manipular e por ser considerada "bem comportada". O MSE pode ser definido como:

$$MSE = \frac{(p_1 - x_1)^2 + \ldots + (p_n - x_n)^2}{n} \tag{4.2}$$

onde n corresponde a quantidade de instâncias, p_k são as respostas preditas para a regressão e x_k é o resultado real, para $k = [1, n]$.

Uma alternativa é usar o erro médio absoluto (*Mean Absolut Error* - MAE), pois o MSE tende a exagerar o efeito das instâncias que possuem valores muito distantes da média (*outliers*). A equação do MAE é definida como:

$$MAE = \frac{|p_1 - x_1| + |\ldots| + |p_n - x_n|}{n} \tag{4.3}$$

onde as variáveis tem a mesma função que na equação 4.2.

A regressão linear é um bom e simples esquema para predição numérica na qual vem sendo utilizado por décadas dentro da estatística. Entretanto, esse tipo de modelo sofre por uma grande desvantagem, a linearidade. Outros métodos que se utilizam da regressão na conjunção com outros métodos, para gerar resultados mais interessantes, serão demonstrados durante este capítulo.

Com mostrado na classificação, um exemplo simples de regressão seria tentar estimar a renda de uma família, baseando-se em características como: número de filhos, número de carros e outras despesas.

4.3 Tabelas de decisão

A mais simples, usualmente mais fácil de entender e também mais rudimentar maneira de representar o resultado de uma máquina de aprendizado é fazê-lo parecido com a entrada - uma tabela de decisão [78]. A tabela representa a função em forma tabular ou matricial; a linha superior da tabela especifica as variáveis ou condições a serem avaliadas e as linhas inferiores especificam a ação correspondente a ser executada quando um teste de avaliação é satisfeito.

A representação chamada de DTM (*Decision Table Majority*) [42], uma das mais simples utilizadas, é a utilização de uma tabela de decisão com um mapeamento de regras padrão para a classe majoritária. Essa representação tem dois componentes: um esquema, que é o conjunto de características que estão incluídos na tabela e um corpo que consiste de instâncias rotuladas no espaço definido pelas características definidas no esquema. Dada uma instância não rotulada, um classificador por tabela de decisão procura pelos casamentos perfeitos dentro da tabela usando apenas as características que estão no esquema (note que pode haver várias instâncias que casam dentro da tabela). Se nenhuma instância é encontrada, a classe majoritária da DTM é retornada, senão a classe majoritária de todas as instâncias encontradas é retornada. Para construir uma DTM, o algoritmo de indução deve decidir quais características incluir no esquema e quais instâncias armazenar no corpo.

4.4 Aprendizado por regras

O objetivo dos algoritmos pertencentes a esta abordagem é gerar um conjunto de regras que demonstrem algum conhecimento ou aprendizado.

Algumas maneiras de aprender utilizando regras são:

- Aprender uma árvore de decisão e depois traduzir essa árvore em um conjunto equivalente de regras, sendo uma regra para cada folha da árvore;
- Utilizar algoritmos genéticos que codificam cada regra em uma cadeia de caracteres (*string*) e que usam operadores de busca genética para explorar o espaço de hipóteses;
- Ou utilizar algoritmos que aprendem um conjunto de regras diretamente utilizando regras de primeira ordem $X \Rightarrow Y$ (se X então Y).

Os algoritmos utilizados para se aprender um conjunto de regras são baseados na estratégia de aprender uma regra, retirar os dados que são cobertos por ela e depois iterar o processo. Esse processo é chamado de cobertura seqüencial, onde aprender uma regra significa utilizar um conjunto de exemplos positivos e negativos; retornar uma única regra que cobre vários exemplos positivos e possivelmente alguns negativos.

4.4.1 Regras de associação

A mineração de regras de associação [2] é uma técnica popular de aprendizado de máquina, que visa descobrir padrões (associações) fortes e interessantes entre itens dentro de um conjunto de dados. A finalidade da mineração de regras de associação é descobrir regras que indiquem a existência de outros atributos. Também são conhecidas como regras do tipo *Se-Então*, onde uma simples regra exemplo seria: **Se** compra_pão **então** compra_leite.

Os algoritmos geralmente são baseados nas medidas: Confiança (equação 4.5) e Suporte (equação 4.4). A função da medida de Suporte é determinar a freqüência que um conjunto de itens ocorre dentre todas as transações da base de dados, é a porcentagem de transações onde este conjunto de dados aparece. A toda regra $X \Rightarrow Y$ é associado um

grau de confiança, onde X é chamado de antecedente ou pré-condição e Y de conseqüente ou conclusão. A Confiança é a medida da força da regra e determina a sua validade, ou melhor, a probabilidade condicional de se encontrar Y, já tendo encontrado X é dada pela confiança.

$$Suporte(X \cup Y) = \frac{|Exemplos(X \cup Y)|}{|Exemplos|} \tag{4.4}$$

$$Confiança(X \Rightarrow Y) = \frac{|Exemplos(X \cup Y)|}{|Exemplos(X)|} \tag{4.5}$$

Olunkule apresentou uma forma interessante de extrair regras de associação de conjuntos de imagens médicas em [55]. Já Ordonez [56] apresenta um algoritmo de mineração de dados, construído sobre um sistema de recuperação de imagens por conteúdo, para encontrar regras de associação em imagens 2D e seus resultados.

A seguir será apresentado o algoritmo Apriori [1], que é o algoritmo mais conhecido para aprendizado de regras de associação e gera todas a regras significativas de associação entre itens na base de dados.

4.4.2 Apriori

O algoritmo Apriori [1] é um dos algoritmos mais conhecidos quando o assunto é mineração de regras de associação em grandes bancos de dados centralizados. O algoritmo encontra todos os conjuntos de itens freqüentes e incorpora técnicas de estimativa e poda.

O módulo principal do algoritmo faz uso de duas funções: a função *AprioriGen*, para gerar os candidatos e eliminar aqueles que não são freqüentes; e a função *GenRules*, utilizada para extrair as regras de associação. O primeiro passo do algoritmo Apriori é realizar a contagem de ocorrências dos itens para determinar os conjuntos de itens freqüentes. Os passos posteriores, consistem de duas fases. Primeiro, os conjuntos de itens freqüentes encontrados no passo anterior são utilizados para gerar os conjuntos de itens potencialmente freqüentes, os conjuntos de itens candidatos. Na seqüência, é realizada uma nova busca no banco de dados, contando-se o suporte de cada candidato.

4.5 Agrupamento

Técnicas de agrupamento, ou também *Clustering*, são geralmente aplicadas quando não existe um atributo classe a ser predito, mas sim quando as instâncias devem ser divididas em grupos de objetos que possuem características similares. Geralmente não se sabe o número de grupos (*clusters*) desejados, então os algoritmos de agrupamento tentam encontrar o melhor agrupamento para os dados existentes [25]. De alguma maneira os grupos refletem o domínio em que as instâncias se encontram.

Algoritmos de agrupamento têm como principal objetivo maximizar a variância inter-classes e minimizar a variância intra-classes, agrupando em uma mesma classe padrões que possuem forte relação entre si.

Quando são aprendidos agrupamentos ao invés de um classificador, a saída toma a forma de um diagrama que mostra como as instâncias estão posicionadas dentro dos agrupamentos. No caso mais simples de agrupamento, é associado um agrupamento para cada instância.

Os agrupamentos encontrados podem ser usados para classificação, deve-se rotular cada agrupamento com uma classe. Alguns dos algoritmos de agrupamento mais utilizados são o *K-Means* [23], o *Fuzzy C-means* [23] e os hierárquicos [40]. Para melhor detalhamento sobre as ténicas de agrupamento pode-se observar Dubes [21] e Celinski [15] para agrupamento de imagens segmentadas.

4.6 Algoritmos genéticos

Algoritmos genéticos (AGs) [8] são baseados em processos da evolução biológica, ou seja, utilizam a teoria Darwiniana como um processo adaptativo de otimização. Estes algoritmos simulam combinação genética (*crossover*), mutação e seleção dos indivíduos melhor adaptados. A idéia principal desta técnica é que com o passar do tempo a evolução selecione as espécies melhor adaptadas [50].

Falando mais formalmente, os AGs são modelos estocásticos e probabilísticos de busca e otimização, inspirados na evolução natural e na genética, aplicados a problemas com-

plexos de otimização. Problemas de otimização tipicamente envolvem três componentes: variáveis, restrições e objetivos. As variáveis descrevem os vários aspectos do problema. As restrições monitoram os valores que as variáveis podem ter. Por último, as funções objetivas são utilizadas para avaliar a solução. As funções objetivas geralmente envolvem a minimização ou a maximização de algum tipo de recurso. São as funções objetivas que medem a qualidade de uma regra gerada em um Algoritmo Genético. As variáveis, as restrições e as funções objetivas, descritas em um problema de otimização definem a geografia básica do espaço de busca e determinam que técnicas podem ser usadas. Técnicas baseadas em heurísticas como Algoritmos Genéticos não podem garantir a solução ótima.

Normalmente os Algoritmos Genéticos são utilizados dentro da mineração de dados para realizar tarefas de classificação de padrões, descrição de registros e seleção de atributos da bases de dados. Na classificação de padrões, os AGs podem buscar a evolução de regras que representem o domínio do problema. Essas regras podem ser, por exemplo, regras *fuzzy* que de uma forma geral são de fácil interpretação, o que incentiva o uso dessa técnica. Os AGs também têm sido aplicados junto de outras técnicas de mineração de dados, tais como: redes neurais, para encontrar os pesos ótimos das ligações; ou a técnica do vizinho mais próximo, para encontrar os pesos a serem aplicados a cada previsor.

4.7 Redes neurais artificiais

Uma Rede Neural Artificial (RNA) é uma técnica computacional que constrói um modelo matemático, emulado por computador, de um sistema neural biológico simplificado, com capacidade de aprendizado, generalização, associação e abstração. As redes neurais tentam construir representações internas de modelos ou padrões achados nos dados, através de um processo de repetidas apresentações dos dados à rede, mas essas representações não são apresentadas para o usuário [78]. Dessa forma, uma RNA procura por relacionamentos, constrói modelos e os corrige de modo a diminuir seu próprio erro.

Estruturalmente, uma rede neural consiste em um número de elementos interconectados (chamados neurônios) organizados em camadas que aprendem pela modificação das conexões (equivalente às sinapses, chamadas de pesos), que conectam as camadas,

semelhante ao sistema biológico. Depois de muitas repetições, uma superfície pode ser internamente definida que se aproxima muito dos pontos dentro do grupo de dados.

A figura 4.1 representa a arquitetura de uma RNA simples. Os círculos representam os neurônios e as linhas representam os pesos das conexões. Por convenção, a camada que recebe os dados é chamada camada de entrada (representada por x_1, x_2 e x_3) e a camada que mostra o resultado é a camada de saída, onde z representa um neurônio dessa camada. A camada interna, onde localiza-se o processamento interno, é tradicionalmente chamada de camada escondida ou intermediária (representada na figura 4.1 por y_1 e y_2). Cada camada tem uma função específica, sendo que a camada de saída recebe os estímulos da camada intermediária e constrói o padrão que será a resposta. As camadas intermediárias funcionam como extratoras de características, seus pesos são uma codificação de características apresentadas nos padrões de entrada e permitem que a rede crie sua própria representação, mais rica e complexa, do problema. Uma RNA pode conter uma ou várias camadas escondidas, de acordo com a complexidade do problema [35].

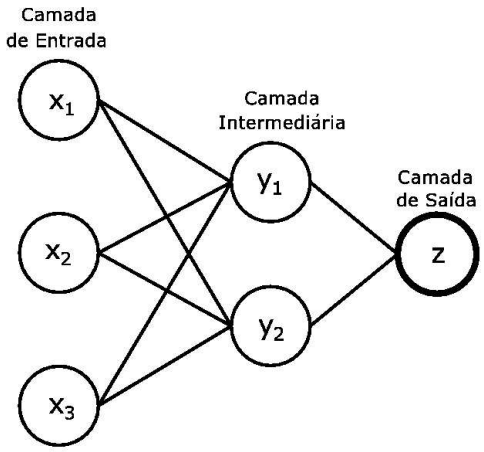


Figura 4.1: Exemplo de topologia de redes neurais.

O perceptron, uma rede neural com apenas um neurônio, é a unidade mais simples de uma rede neural. Tomando um vetor de números reais como entrada, o perceptron calcula uma combinação linear desses atributos e então retorna o resultado. Essa rede possuindo apenas uma camada é capaz de solucionar problemas que sejam linearmente separáveis.

A retropropagação (*Backpropagation*) surgiu para resolver alguns problemas que existiam dentro do treinamento das redes neurais, se tornando assim um dos algoritmos mais populares para esse treinamento, sendo em parte responsável pelo ressurgimento do interesse nesta área.

4.7.1 Retropropagação (*Backpropagation*)

De maneira resumida, durante o treinamento com o algoritmo *backpropagation*, a rede opera em uma seqüência de dois passos. Primeiro, um padrão é apresentado a camada de entrada da rede. A atividade resultante flui através da rede, camada por camada, até que a resposta seja produzida pela camada de saída. No segundo passo, a saída obtida é comparada à saída desejada para esse padrão particular. Se esta não estiver correta, o erro é calculado. O erro é propagado a partir da camada de saída até a camada de entrada e os pesos das conexões das unidades das camadas internas vão sendo modificados conforme o erro é retropropagado.

4.8 Classificadores Bayesianos

O aprendizado Bayes provê uma abordagem probabilística à inferência e é baseado na suposição de que as quantidades de interesse são governadas pela distribuição de probabilidade, que decisões ótimas podem ser feitas através da razão entre essas probabilidades e os dados observados. A regra de bayes estabelece:

$$P(A|B) = P(B|A)\frac{P(A)}{P(B)} \tag{4.6}$$

sendo A, B, e C são subconjuntos do espaço de amostras (universo do problema, ou seja, a base de dados) e $P(A)$ representa a probabilidade do evento A ocorrer, independente de B. De modo análogo, $P(B)$ é a probabilidade de B ocorrer, independente de A. Finalmente, $P(B/A)$ é a probabilidade do evento B ocorrer, dado que A ocorre. A teoria de probabilidade Bayesiana baseada nesses dados, através da equação citada, determina a probabilidade do evento A ocorrer, dado que B ocorre. O princípio básico desse método

está fundamentado na teoria da probabilidade Bayesiana [71], representada pela seguinte fórmula básica:

$$P(AB|C) = P(A|C)P(B|AC) = P(B|C)P(A|BC) \tag{4.7}$$

a notação $P(AB|C)$ significa: a probabilidade dos eventos A e B acontecerem dado que o evento C acontece.

Na tarefa de classificação de padrões, a regra de Bayes pode ser utilizada para determinar as probabilidades dos padrões pertencerem a cada uma das classes em questão, desse modo, sejam $A_1, \ldots, A_k$ atributos, $(a_1, \ldots, a_k)$ uma tupla do banco de dados, e C uma classe a ser prevista. A previsão ótima é uma classe de valor c tal que:

$$P(C = c|A_1 = a_1 \cap A_2 = a_2 \cap A_1 = a_1 \cap \ldots A_k = a_k) \tag{4.8}$$

onde P seja máximo.

Um dos classificadores práticos mais utilizados dentro da classificação Bayesiana, é o chamando *Naive Bayes* [78], apresentado a seguir, seguido pelo conceito de redes Bayesianas.

4.8.1 Classificador *Naive Bayes*

O classificador *Naive Bayes* [78] se aplica para cada tarefa de aprendizado, onde cada instância x é descrita como um conjunção de valores de atributos e a função alvo $f(x)$ pode tomar como argumento qualquer valor (v_j) de um conjunto finito V. O aprendiz é mandado classificar para uma nova instância do conjunto de treinamento com atributos $(a_1, a_2, a_3, \ldots, a_n)$. A abordagem Bayesiana tenta classificar cada nova instância para o valor mais provável (V_{map}), dados os valores dos atributos $(a_1, a_2, a_3, \ldots, a_n)$ que descrevem a instância. Onde V_{map} é:

$$V_{map} = argMax_{v_j \in V} P(a_1, a_2, a_3, \ldots, a_n|v_j)P(v_j) \tag{4.9}$$

esse classificador é baseado na simples suposição de que os valores dos atributos são condicionalmente independentes dado o valor alvo. Em outras palavras, a suposição é a que dado o atributo alvo da instância, a probabilidade de observar a conjunção $(a_1, a_2, a_3, \ldots, a_n)$ é apenas o produto de suas probabilidades para os atributos individuais $P(a_1, a_2, a_3, \ldots, a_n|v_j) = \prod_i P(a_i|vj)$:

$$V_{NB} = argMax_{v_j} P(v_j) \prod_i P(a_i|vj) \tag{4.10}$$

onde V_{NB} denota o valor de saída do classificador, ou seja, a classificação.

4.8.2 Redes Bayesianas

A rede Bayesiana é um método que alia a experiência e o conhecimento de especialistas com as estatísticas dos dados. Este método é realizado em duas etapas. Primeiro, um ou mais especialistas codificam seu conhecimento em redes Bayesianas. Em seguida, utiliza-se o banco de dados para atualizar o conhecimento e criar uma ou mais novas redes Bayesianas, melhorando as informações previamente fornecidas.

As redes Bayesianas são parecidas com as redes neurais artificiais, descritas anteriormente, porém com duas diferenças importantes: possuem maior facilidade de interpretação de seus resultados e fornecem a possibilidade de codificação de conhecimento por parte de um ou mais especialistas. O fato de existir a necessidade de um especialista para codificar o conhecimento torna este método difícil de ser automatizado, indo de encontro ao objetivo principal deste trabalho que é criar um modelo automático.

Heckerman [34] apresenta maiores informações com relação as redes Bayesianas, demonstrando em detalhes como aplicar esse método na descoberta de conhecimento a partir de bancos de dados.

4.9 Aprendizado *Lazy*

O aprendizado *lazy*, também chamado de *Instance-Based Learning* (IBL), é definido por alguns autores [47] como um método que produz descrições intencionais dos conceitos das

descrições dos exemplos. De fato, essa é a diferença principal entre os métodos de indução e aprendizado *lazy*.

Nos métodos *lazy*, o processo de aprendizado e o processo de utilização do conhecimento aprendido não podem ser considerados separadamente. Para resolver um novo problema usando o método de aprendizado *lazy*, a solução de um problema antigo é transferida para o novo problema, ocorrendo uma generalização entre um problema antigo e o problema novo. Os métodos do aprendizado *lazy* são aqueles que são centrados no problema, de maneira que o passo de generalização é feito sob-demanda, quando um novo problema deve ser resolvido.

A base de vários algoritmos de aprendizado *lazy* é o algoritmo k-NN (k-*nearest neighbour* ou k-ésimo vizinho mais próximo). Dado um novo exemplo para classificar, o algoritmo recupera as k instâncias mais similares e prediz a classe para qual o novo exemplo pode pertencer. A qualidade do k-NN depende de como essas instâncias são consideradas similares, ou seja, a função de similaridade.

4.10 Árvores de decisão

Nesta técnica, assim como em outras técnicas de classificação, o usuário escolhe a variável que quer avaliar e o algoritmo procura os atributos mais correlacionados e gera uma árvore de decisão com várias ramificações [78]. Depois de criada a árvore, esta será utilizada para classificar novas instâncias, respeitando os valores dos atributos da nova instância.

Dado um espaço amostral de um problema, uma árvore de decisão [11] procura representar as informações nele contidas. Quanto maior o poder de predição de uma árvore, maior é a sua fidelidade à amostra analisada.

As árvores de decisão nada mais são do que meios de representar resultados de mineração de dados na forma de árvore e elas constituem uma técnica bastante popular na realização da tarefa de classificação devido as seguintes características [44]:

- Velocidade na geração;
- Facilidade de aplicação em domínios numéricos;

- Facilidade de compreensão das decisões finais.

As abordagens de geração de árvores de decisão geralmente utilizam a técnica de divisão-e-conquista [84]. Para essa geração, um atributo é selecionado inicialmente, de modo que maximize alguma medida de predição, chamada de critério de ramificação, que geralmente é escolhido utilizando de fórmulas como ganho de informação e entropia.

Após a seleção do atributo, é definida em quantas partes será dividida a base de dados e então, são identificados em quais valores do domínio do atributo ocorrerá a divisão. Esse constitui-se de testes lógicos, sendo chamado de nó de decisão. O primeiro nó de decisão é um nó especial chamado de nó raiz da árvore. Para cada subconjunto de dados gerado pela raiz, o processo é repetido, recursivamente, até que alguma instrução determine a interrupção do processo, o critério de parada, formando os nós-folhas. Nas árvores de classificação, esses nós folhas possuem a informação das classes de um problema.

Assim como o critério de ramificação, o critério de parada também pode ser definido segundo a vontade do pesquisador. O mais comum é que se interrompa a indução quando não houver mais progresso ao se criar uma nova ramificação ou quando um subconjunto a ser criado for formado por apenas um registro. Nesse segundo caso, a árvore gerada pode se tornar muito complexa (com muitos nós) e, freqüentemente, com alta taxa de erro na predição [64]. Para evitar esta situação ou se interrompe o processo antes que a árvore se torne muito complexa, ou permite-se que a árvore seja criada até o fim para depois remover parte de sua estrutura. No primeiro caso, chamado de parada ou pré-poda, não se perde tempo construindo uma estrutura que não será aproveitada. A grande dificuldade desta estratégia é determinar o momento exato da interrupção pois é possível que se pare antes ou depois do ponto ideal. No segundo caso, chamado de poda ou pós-poda, há um consumo maior de tempo para se gerar toda a árvore, algumas vezes substancial. Quinlan [64] defende que a maior exploração da base de dados freqüentemente compensa esta desvantagem, permitindo o encontro de sub-árvores mais confiáveis, isto é, com maior poder de predição.

A maioria dos algoritmos de aprendizado que utilizam árvores de decisão são variações de um algoritmo básico que aplica uma busca gulosa *top-down* pelo espaço de busca de

possíveis árvores, o ID3 (*Itemized Dichotomizer 3*). Um dos algoritmos mais utilizados atualmente para a classificação utilizando esta abordagem é o algoritmo C4.5 [64]. O C4.5 é derivado do ID3 e tem esse nome de acordo com a versão de implementação (4.5) e linguagem (C) utilizada para essa implementação.

Um exemplo de árvore de decisão é mostrada na figura 4.2, onde X_1 e X_2, são valores para atributos da base de dados que devem ser comparados com os valores 20 e 80, respectivamente, para descobrir para que galho (lado) da árvore a inferência deve seguir; C_1, C_2 e C_3 são as classes que devem ser atribuídas as instâncias que alcançarem aquele nó-folha da árvore.

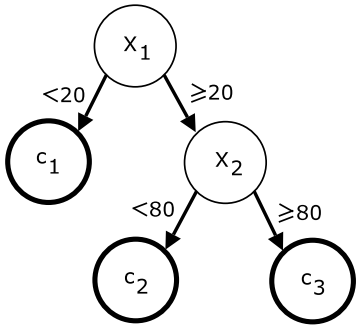


Figura 4.2: Exemplo de árvores de decisão padrão.

Mais tarde, dentro do conceito de árvores de decisão *fuzzy*, as árvores de decisão clássicas serão consideradas árvores de decisão *crispy*, pelo fato de não possuir uma suavização entre os nós de decisão da árvore, diferentemente das árvores de decisão *fuzzy*. Esse processo de discretização suave também será apresentado na seção 4.10.7.

4.10.1 ID3

O algoritmo ID3 é o algoritmo mais conhecido e simples para aprendizado utilizando árvores de decisão [62]. O ID3 utiliza a abordagem de busca gulosa *top-down*, tentando assim descobrir qual o melhor atributo, se baseando na pergunta: "Qual o atributo deve ser testado no nó corrente da árvore?". Ou seja, o algoritmo funciona procurando o melhor atributo para cada posição na árvore, começando pela sua raiz. Basicamente, o ID3 busca o atributo que possui o melhor ganho de informação, para o conjunto de

exemplos de entrada, cria para cada valor desse atributo um nó filho e o processo se repete, mas agora olhando apenas para os exemplos que possuem valores de atributos correspondentes aos valores acima na árvore.

Para o ID3 escolher o melhor atributo entre os atributos existentes, ele busca o atributo que possui o melhor ganho de informação. O ganho de informação é baseado no cálculo da entropia para cada atributo. Os cálculos da entropia e do ganho de informação são mostrados nas equações 4.11 e 4.12, respectivamente:

$$Entropia(S) = \sum_{i=0}^{c} -p_i log_2 p_i \tag{4.11}$$

$$Ganho(S, A) = Entropia(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropia(S_v) \tag{4.12}$$

onde S indica o conjunto de exemplos, p indica cada instância do conjunto de exemplo, c o número de classes, A o atributo corrente e v o valor para cada possibilidade do atributo A.

O algoritmo 1 é a descrição do algoritmo ID3, como apresentado em [51]. Onde os parâmetros são: Exemplos e Atributos. Os Exemplos são as instâncias de treinamento. Os Atributos são uma lista de atributos que devem ser testados e aprendidos pela árvore. O retorno do algoritmo é a árvore de decisão que classifica os Exemplos.

4.10.2 CART

O CART (*Classification And Regression Trees*), é um algoritmo de exploração e previsão de dados que usa métricas de entropia para escolher ramificações ótimas [11].

Este algoritmo trabalha com atributos contínuos e produz árvores binárias, isto é, aquelas em que cada nó de decisão possui apenas dois ramos, definindo dinamicamente de que maneira é realizado o particionamento. Os particionamentos dos atributos e a escolha de um deles para a ramificação da árvore são atividades dependentes entre si. O par atributo/particionamento escolhido é aquele que minimize a impureza nos dados. No CART, um ramo pode escolher várias vezes um mesmo atributo para a ramificação, o que prejudica a sua interpretação. Uma árvore é desenvolvida até que a impureza dos dados

Algoritmo 1 Algoritmo ID3 (Exemplos, Atributos)

```
1: R ← Nó raiz
2: if Todos Exemplos são positivos then
3:    return Raiz com classificação positiva
4: else if Todos Exemplos são negativos then
5:    return Raiz com classificação negativa
6: else if Todos Atributos são nulos then
7:    return Raiz com classificação mais freqüente
8: else
9:    A ← atributo com maior ganho de informação
10:   Atributo de R ← A
11:   for all V_i ∈ A do
12:      Adicionar um novo filho para a raiz, correspondente a A = V_i
13:      ExemplosV_i ← subconjunto de Exemplos em que A = V_i
14:      if ExemplosV_i está nulo then
15:         Adicionar uma nova folha com a classificação mais comum dentro de Exemplos
16:      else
17:         Adicionar a sub-árvore ID3 (Exemplos, Atributos−A)
18:      end if
19:   end for
20:   return R
21: end if
```

não consiga ser reduzida e depois é realizada a poda baseando-se na complexidade da árvore (profundidade e quantidade de nós) e nos erros cometidos durante o treinamento.

Da mesma forma que o algoritmo C4.5, o CART é recursivo e pode lidar com dados ausentes e diversos tipos numéricos (contínuos, categóricos, booleanos, etc.). Entretanto, por realizar particionamentos binários, este algoritmo apresenta dificuldades de trabalhar com atributos que podem assumir mais de duas classes (ex.: baixo, médio e alto). Nesses casos, podem existir dois ou mais nós para representar o mesmo atributo, gerando árvores maiores e mais complexas.

4.10.3 Árvores de regressão e árvores de modelo

As árvores usadas para a predição numérica são como árvores de decisão, só que ao invés de armazenar os valores das classes nas folhas, por exemplo, elas podem armazenar valores médios para as instâncias que chegam até aquela folha, chamadas de árvores de regressão. Uma outra árvore usada para a predição numérica é a árvore de modelo, que ao invés da média, armazena um modelo de regressão linear nas suas folhas.

O algoritmo M5'P [63] usa o conceito de árvores de modelo para gerar árvores mais precisas para solucionar problemas. Já o algoritmo M5'Rules [36] é muito similar ao M5'P, com a diferença de extrair regras das árvores.

4.10.4 M5'P

Enquanto árvores de regressão possuem valores em suas folhas, as folhas construídas pelo M5'P [63] podem ter modelos lineares multivariados. O M5'P aprende eficientemente e pode cuidar de tarefas de grandes dimensões, com centenas de atributos. Uma vantagem do M5'P é que as árvores de modelo são geralmente muito menores do que as árvores de regressão e têm provado maior precisão nos problemas estudados.

O objetivo do algoritmo é construir um modelo para cada folha que relacione os valores alvo dos casos de treinamento que chegam àquela folha com os valores dos atributos das instâncias. O valor do modelo é calculado pela precisão com qual ele prediz os valores alvo dos casos não vistos.

Para a criação de uma árvore de modelo, supomos que temos um conjunto T de casos de treinamento. Cada instância pode possuir tanto atributos contínuos como discretos. As árvores também são construídas através do método dividir-e-conquistar. O conjunto T é associado à uma folha ou algum teste é escolhido para que divida T em subconjuntos que correspondam aos resultados dos testes. Aplicando esse mesmo processo recursivamente aos subconjuntos, até alcançar o critério de parada.

O primeiro passo para construir uma árvore de modelo é computar o desvio padrão dos valores dos casos desejados em T. A menos que T contenha poucos casos ou seus valores variem pouco, T é dividido nos resultados do teste. Cada teste em potencial é avaliado determinando-se subconjuntos de casos associados com cada resultado. Depois de examinar todos os possíveis testes, o M5'P escolhe um que maximize essa redução do erro esperado.

4.10.5 *M5'Rules*

O *M5'Rules* [36] é um novo método para geração de regras a partir de uma árvore de modelo. Ele funciona aplicando um classificador de árvores de decisão para todo o conjunto de treino e uma árvore é gerada de modo similar ao método M5'R descrito anteriormente. A seguir, a melhor folha (de acordo com algumas heurísticas) é transformada em uma regra e o restante da árvore é descartada. Todas as instâncias cobertas por esta regra são removidas do conjunto de dados. Esse processo é aplicado recursivamente para as instâncias restantes e termina quando todos as instâncias estão sendo cobertas por uma ou mais regras. Essa é a estratégia básica de divisão-e-conquista para aprendizado de regras; entretanto, ao invés de construir uma única regra, como é feito usualmente, é construída uma árvore de modelo completa a cada estágio, transformando a sua "melhor" folha em uma regra.

4.10.6 *Logistic Model Trees (LMT)*

As árvores de modelo logístico [45] adaptam a idéia das árvores de modelo para problemas de classificação, com a diferença de usar regressão logística ao invés de regressão linear. É usado um processo de adaptação por etapas para construir os modelos logísticos de regressão nas folhas, redefinindo incrementalmente aqueles construídos em níveis superiores da árvore.

4.10.7 Árvores de decisão *fuzzy*

O emprego de árvores de decisão juntamente com a teoria dos conjuntos *fuzzy* têm sido bastante utilizado. De modo simplificado, árvores de decisão *fuzzy* aplicam a lógica *fuzzy* na formulação dos nós de decisão, nós de decisão *fuzzy*, e das conclusões definidas nos nós-folhas. Além disso, a inferência realizada pela árvore também é *fuzzy*. Diferentemente do que ocorre com as árvores de decisão *crispy*, nas árvores *fuzzy* existem vários caminhos a serem percorridos para se realizar a predição de um elemento. Assim, diversos nós-folhas realizam a predição com diferentes graus de certeza (grau de pertinência).

As árvores de decisão *fuzzy* são construídas de maneira similar ao algoritmo de indução de árvores *crispy* apresentado anteriormente. Entretanto, a criação das árvores de decisão *fuzzy* pode ser feita de diferentes maneiras. Chiang [17] indica alguns métodos de geração de árvores *fuzzy* (*Pre-fuzzification* e *Post-fuzzification*). Enquanto Marsala [49] discute qual a melhor maneira de construí-las.

Existem algumas diferenças, entre as árvores *fuzzy* e *crispy*, provenientes das características da lógica *fuzzy*. Dentre estas diferenças, pode-se destacar:

Quanto à indução da árvore:

- Cada nó interno da árvore que utiliza um atributo contínuo para divisão possui conjuntos *fuzzy* associados a seus ramos referentes às suas sub-árvores. A divisão dos exemplos é realizada segundo a pertinência dos exemplos para cada um dos conjuntos *fuzzy* que o nó possui;

- Devido à possibilidade de existir pertinências parciais aos conjuntos *fuzzy* de cada nó, os exemplos de treinamento podem corresponder a vários nós da árvore;

- As fórmulas do ganho de informação e entropia devem ser alteradas para se adequar a existência de pertinências parciais dos exemplos aos nós;

Quanto à classificação de novos exemplos:

- Devido à existência de pertinências parciais, durante a classificação de um exemplo, este pode alcançar vários nós folha da árvore. Desta maneira, o processo de caminhar na árvore para determinar a classe de um nó deve ser alterado;

- Por poder alcançar vários nós da árvore, um exemplo pode ser classificado em diversas classes diferentes. Desta maneira, a determinação da classe de um exemplo deve ser alterada. Um algoritmo *fuzzy* para classificação pode predizer a classe para a qual o exemplo possui maior grau de pertinência ou fornecer ao usuário as probabilidades do exemplo pertencer a cada classe.

Além disso, um nó-folha pode realizar mais de uma conclusão. Então, é necessário definir uma estratégia de inferência para a árvore observando a diversidade de conclusões

presente em cada nó-folha e a diversidade de ramos que geraram uma resposta. Se a resposta da árvore for um conjunto *fuzzy* e precisar ser "defuzzificada", um procedimento adicional precisará ser definido também.

Apesar das pesquisas sobre árvores de decisão *fuzzy* não serem tão recentes, o primeiro trabalho foi em 1977 [16], a quantidade de trabalhos não é tão volumosa quanto as de outras técnicas híbridas oriundas de sistemas *fuzzy*: Redes Neurais [53], Regras de decisão [5] e Agrupamento [59]. Alguns trabalhos são apresentados a seguir. Em geral, os modelos de construção de árvores *fuzzy* são inspirados nos modelos tradicionais de árvores. Desses, o que provavelmente mais influenciou as pesquisas foi o ID3. Olaru [54] mostra uma técnica completa para utilização de árvores de decisão *fuzzy*, usando o conceito de discretização suave, que é abordado mais adiante na seção 4.10.9.

4.10.8 FID3

O FID3 [39] (*Fuzzy* ID3) é uma árvore de decisão *fuzzy*, ou seja, as decisões tomadas nos nós da árvore seguem critérios *fuzzy*. Dessa forma, este método apresenta uma maior generalização que as tradicionais árvores *crispy*. O FID3 é utilizado por modelos híbridos na extração de regras de classificação. O algoritmo têm características semelhantes às árvores *crispy*, como o aprendizado *top-down* recursivo e a possibilidade de lidar com diversos tipos de atributos, porém tem alguns detalhes próprios, tais como: os padrões estão distribuídos nas partições através de funções de pertinência e os resultados variam em função das funções de pertinência e dos operadores utilizados.

No FID3, o conceito de entropia é "fuzzificado" e um elemento irá pertencer a um ramo se o seu grau de pertinência ao ramo for positivo. Janikow também incorporou o tratamento a valores ausentes nos registros, manipulação de atributos contínuos e propôs diversas estratégias para se obter a classificação de um novo elemento pela árvore *fuzzy*.

4.10.9 SDT

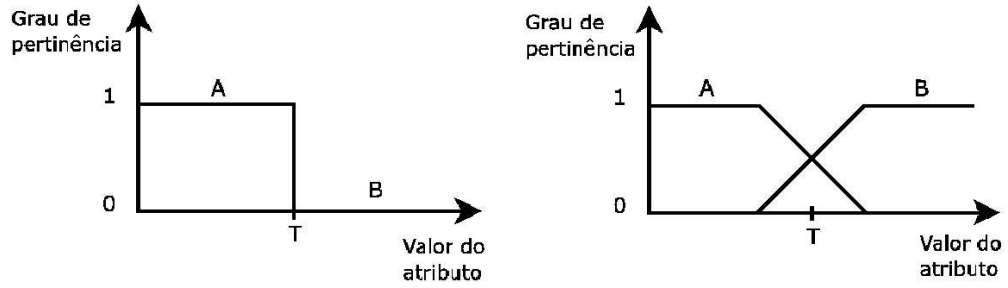
O algoritmo SDT (*Soft Decision Tree*) gera uma árvore de decisão *fuzzy* que utiliza discretização suave dos atributos para transformar nós de decisão em nós de decisão *fuzzy*.

O SDT propõe um critério de indução de árvores de decisão alternativo, com a determinação em tempo de execução dos conjuntos *fuzzy*. Esse critério aplica-se somente a exemplos cujos atributos são contínuos pré-classificados segundo o atributo de classe. A árvore de decisão induzida é equivalente a uma árvore de decisão *crispy*, mas o processo de indução e de classificação utilizam critérios *fuzzy*, como funções de pertinência *fuzzy*.

De acordo com o algoritmo proposto por Peng [60] o domínio dos atributos é particionado em espaços disjuntos através de uma discretização suave, utilizando dois conjuntos *fuzzy* a cada iteração para a construção de um nível da árvore, gerando assim uma árvore binária de decisão *fuzzy*. Os dois conjuntos utilizados compartilham parte do intervalo de valores do domínio que se sobrepõe. Portanto, tanto no processo de indução da árvore quanto na classificação haverá elementos que pertencerão a mais de um conjunto com graus de pertinência diferentes.

O processo de discretização suave envolve a análise do ganho de informação para cada atributo. A discretização suave é definida por três parâmetros: um limiar e os outros dois são funções de pertinência (A_1 e A_2) correspondentes aos conjuntos *fuzzy* 1 e 2, onde $A_1(a_i) + A_2(a_i) = 1$. O limiar ou ponto de corte, é a localização entre valores contínuos de um atributo de uma instância, que divide o conjunto de dados em dois conjuntos *fuzzy*. Esse limiar corresponde a média aritmética dos valores de um atributo A de instâncias com classes diferentes que separa o domínio em duas classes, ou seja, $T = (a_i + a_{i+1})/2$. As funções de pertinência são determinadas de acordo com as características do atributo, tal como a sua incerteza. Usualmente, uma grande sobreposição entre esses dois conjuntos é usada para um atributo que é altamente incerto, por exemplo, pode-se usar a distância média entre os dados como sendo a largura de sobreposição.

Na figura 4.3 pode-se ver os pontos de corte (T) em ambas as abordagens de discretização: a *crispy*, utilizada em árvores de decisão comum; e a suave (*fuzzy*) utilizada em árvores de decisão *fuzzy*.



(a) (b)

Figura 4.3: Pontos de corte para as abordagens de discretização: (a) *Crispy*; (b) Suave (*Fuzzy*).

A figura 4.4 mostra um exemplo simples para um certo limiar T, onde o elemento X pertence 30% a classe B e 70% a classe A, sendo V_X o valor de X para o atributo corrente.

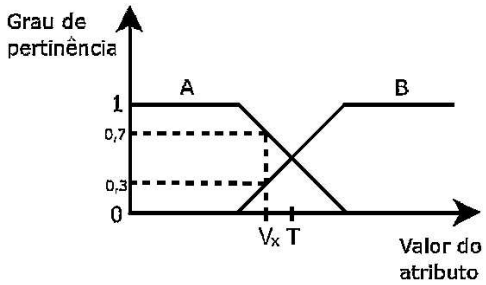


Figura 4.4: Exemplo de discretização *fuzzy*.

A entropia *fuzzy* (E_F) de um conjunto de exemplos S é definida na equação 4.13:

$$E_F(S) = \sum_{j=1}^{k} p(c_j, S) log_2 p(c_j, S) \tag{4.13}$$

$$p(c_j, S) = \sum_{a_i \in c_j} (A_1(a_i) + A_2(a_i)) \tag{4.14}$$

onde k é o número de classes, c_j corresponde a classe j e $p(c_j, S)$ é a proporção *fuzzy* de exemplos em S. Depois da discretização suave, a entropia geral é calculada usando a probabilidade *fuzzy*, como:

$$E_F(A,T;S) = \frac{N_{S_1}}{N_S}E_F(S_1) + \frac{N_{S_2}}{N_S}E_F(S_2) \tag{4.15}$$

onde $N_{S_1} = |S_1|$, $N_{S_2} = |S_2|$ e $N_S = |S|$.

O algoritmo de Peng busca o atributo e o limiar que possua a menor entropia entre todos os possíveis. Para determinar qual atributo será utilizado para subdividir o domínio do espaço, o método ordena as instâncias crescentemente para cada atributo. Então, para cada atributo o algoritmo calcula o ganho de informação entre cada par de exemplos onde existe uma mudança do valor da classificação. O algoritmo não calcula o ganho de informação entre atributos de dois elementos que possuem a mesma classificação, pois busca-se maximizar o ganho de informação. Fayaad [26] demonstra que o ganho de informação obtido pela partição do domínio por dois atributos de mesma classificação não é máximo. A cada iteração do algoritmo, o espaço é subdividido de forma a minimizar a entropia entre os dados. Um aspecto importante é determinar a condição de parada que determine os nós folhas. Existem várias abordagens para determinação da condição de parada que levam em consideração a quantidade de exemplos classificados e as características desses exemplos.

Uma vez determinada a árvore de decisão, busca-se classificar novos exemplos. A cada nó da árvore de decisão *fuzzy* gerada existe uma função de pertinência correspondente, assim uma instância ao percorrer a árvore pode pertencer simultaneamente a mais de uma classe. Uma abordagem recursiva proposta para o problema é que um exemplo será disparado para os ambos os galhos direito e esquerdo da árvore quando os graus de pertinência os lados forem maiores que zero. O processo se repete recursivamente até alcançar os nós folhas. A classificação de exemplos baseada neste modelo, terá como resultado, a pertinência de um exemplo a vários nó folhas que podem representar várias classes diferentes. Surgem então dois problemas: como determinar o grau de pertinência aos nós folhas e como determinar qual classe classificará o exemplo.

Peng propõe que a classificação final de um exemplo seja dada pelo maior valor da função de pertinência de para uma certa classe. Para determinar o grau de pertinência de cada nó folha há várias abordagens. Peng mostra duas medidas que podem ser utilizadas.

A primeira medida sugere a utilização da média ponderada dos valores de todas as funções de pertinência do caminho que leva da raiz ao nó folha. A segunda medida proposta considera calcular o mínimo entre o valor de todas as funções de pertinência do caminho entre a raiz e a folha.

Na implementação do algoritmo é necessária a utilização de 4 parâmetros de entrada:

- Alfa (α): é utilizado como critério para a decisão de criar ou não um nó filho, através do cálculo do grau verdade (η_k):

$$\eta_k = \frac{N_{S_k}}{N_S} \tag{4.16}$$

  sendo k a classe corrente e N_{S_k} o tamanho do conjunto S_k. Se η for menor que alfa então não deve ser criado um nó filho para aquele lado;

- Beta (β): Neste algoritmo a condição de parada ocorre quando máximo entre as porcentagens dos elementos pertencentes as diversas classes é maior que beta. Ou seja, o maior grau verdade entre todas as classes (j), para cada um dos filhos:

$$\mu_{1c_j} = \frac{\sum_{a_i \in c_j} A_1(a_i)}{N_{S_1}} \tag{4.17}$$

$$\mu_{2c_j} = \frac{\sum_{a_i \in c_j} A_2(a_i)}{N_{S_2}} \tag{4.18}$$

- Lambda (λ): É um parâmetro que indica se uma instância deve ou não descer para um lado da árvore no momento de sua geração;

- Delta (Δ): O parâmetro delta dá a inclinação da reta e define o intervalo onde os exemplos pertencerão simultaneamente a dois conjuntos *fuzzy*.

Um alfa pequeno e um beta grande irá gerar uma árvore grande com alta precisão no treinamento. Entretanto, quando os dados são incertos ou com ruídos, um alfa muito pequeno e um beta muito grande pode tornar a árvore de decisão *fuzzy* sobrecarregada(*over-fitted*). O algoritmo 2 é a descrição do algoritmo proposto por Peng.

Algoritmo 2 Algoritmo PengSDT (Exemplos, Atributos)

for all $A \in$ Atributos **do**
  Ordenar Exemplos de acordo com A, gerando $(a_1, a_2, \ldots, a_N)$
  Gerar os candidatos a pontos de corte $T = (a_i + a_{i+1})/2$, usando limites entre classes
  Fuzzificar T para gerar candidatos a discretização suave, usando o par de exemplos, a_1 e a_2, na qual formam uma partição *fuzzy* sobre o ponto de corte
  Avalie cada candidato a discretização suave usando $E_F(S)$
  Selecione o menor valor de $E_F(A, T; S)$
end for
Selecionar A com a menor discretização para gerar dois filhos
Calcular o grau verdade para ambos os filhos (η_1 e η_2)
if (η_1 ou η_2) $> \alpha$ **then**
  Criar dois filhos para o nó corrente
  if $max(\mu_{1c_j, c_j \in C}) > \beta$ **then**
    O filho esquerdo vira folha com classificação c_j
  else
    Gerar S_1 contendo as instâncias onde $A_1 > \lambda$
    Executar PengSDT (S_1, Atributos)
  end if
  if $max(\mu_{2c_j, c_j \in C}) > \beta$ **then**
    O filho direito vira folha com classificação c_j
  else
    Gerar S_2 contendo as instâncias onde $A_2 > \lambda$
    Executar PengSDT (S_2, Atributos)
  end if
end if

4.11 Agrupamento de classificadores

Um agrupamento de classificadores (*ensemble*) é um conjunto de classificadores na qual decisões individuais são combinadas de alguma maneira (tipicamente por votação ponderada ou não) para classificar novos exemplos.

Este agrupamento pode ser avaliado por três características: precisão, diversidade e tamanho. Primeiramente deve-se definir a precisão (taxa de acerto/erro) e diversidade do *ensemble*. Depois, preocupar-se com o tamanho do *ensemble*, sendo qual a influência dele na precisão e diversidade [46]. Um bom *ensemble* é aquele considerado preciso e ao mesmo tempo diverso [20]. Uma condição necessária e suficiente para que um agrupamento de classificadores seja mais preciso do que qualquer um dos seus membros é que os seus membros devem ser precisos e diversos, sendo que um membro pode ser considerado preciso se sua taxa de erro é menor do que apenas classificar aleatoriamente. Dois membros podem

ser considerados diversos se eles tem respostas diferentes para a mesma instância [20].

Existem 3 razões fundamentais para podermos construir um bom agrupamento de classificadores:

- Estatística: Aparece quando a quantidade de dados de treinamento disponível é muito pequena comparada ao tamanho do espaço de hipóteses. Sem dados suficientes, o aprendiz pode encontrar várias hipóteses diferentes no espaço de hipóteses onde todas terão a mesma precisão. Construindo um agrupamento, o algoritmo pode fazer uma média entre os votos dos classificadores e reduzir o risco de escolher o classificador errado;

- Computacional: Vários algoritmos funcionam fazendo algum tipo de busca local, que pode ficar parada no ótimo local. Por exemplo, algoritmos de redes neurais empregam descida do gradiente para minimizar uma função de erro sobre os dados de treinamento, e algoritmos de árvore de decisão empregam um regra gulosa de separação para o crescimento da árvore. Em casos em que existem dados de treinamento suficientes, ainda pode ser muito difícil computacionalmente para o algoritmo de aprendizado encontrar o melhor espaço de hipóteses;

- Representacional: Na maioria das aplicações de aprendizado de máquina, uma função verdade não pode ser representada por qualquer uma das hipóteses do espaço de hipóteses. Fazendo uma soma ponderada das hipóteses retiradas do espaço, pode ser possível expandir o espaço das funções representáveis.

Uma das áreas mais ativas na pesquisa de aprendizado supervisionado tem sido estudar métodos para construir bons agrupamentos de classificadores. A descoberta principal é que os agrupamentos são geralmente mais precisos que os classificadores individuais que os compõem.

A próxima seção mostra alguns métodos de agrupamento de classificadores e como eles gerenciam esse agrupamento.

4.11.1 *Bagging*

O *Bagging* [65] é um método genérico para aumentar a precisão de um algoritmo de aprendizado. É um método utilizado em combinação com algoritmos de aprendizado para a construção de conjuntos de classificadores.

No aprendizado utilizando o *Bagging* é gerada uma amostra de um conjunto de treinamento de tamanho N (com substituição) das instâncias originais, para cada tentativa $t = 1, 2, 3, \ldots, T$. Esse conjunto de treinamento possui o mesmo tamanho dos dados originais, mas algumas instâncias podem não aparecer enquanto outras podem aparecer mais de uma vez. Daí, o sistema de aprendizado gera um classificador C_t da amostra e o classificador final C é formado agregando-se os T classificadores dessas tentativas descritas. Para classificar uma instância x, um voto para a classe k é gravado para cada classificador no qual $C_t(x) = k$ e $C(x)$ é, então, a classe com a maior quantidade de votos (sendo que os empates são resolvidos arbitrariamente).

Segundo Breiman [10]: "O elemento vital é a instabilidade dos métodos de predição. Se a preocupação com os conjuntos de aprendizado podem causar mudanças significativas no preditor construído, então o *Bagging* pode aumentar a eficiência".

4.11.2 *Boosting*

Assim como o *Bagging*, o *Boosting* [68] também é um método para aumentar a precisão de um aprendiz. Ele combina algoritmos de aprendizado para a construção de um *ensemble* utilizando regras extraídas dos dados.

Construir uma regra altamente precisa é uma tarefa difícil. Por outro lado, não é difícil formular regras pequenas/fáceis de manuseio. O *Boosting* é baseado na observação de que encontrar muitas regras brutas de manuseio é mais fácil que encontrar uma única regra de alta precisão [29].

O funcionamento do *Boosting* [65] é simples, ele é iniciado com a execução de um algoritmo para encontrar as regras brutas de manuseio. O algoritmo de *Boosting* chama o algoritmo base repetidamente, cada vez alimentando com um diferente subconjunto de exemplos de treinamento (este conjunto possui pesos para forçar o algoritmo a encontrar

regras). Cada vez que o algoritmo é chamado, são geradas novas regras fracas de predição e depois de muitas rodadas, o algoritmo combina estas regras (fracas/brutas) em uma única regra de predição, que deverá ser muito mais precisa do que qualquer outra regra bruta. Para a formação das regras fracas, são colocados pesos nos exemplos mais comumente não classificados pelas regras fracas precedentes, forçando o algoritmo de aprendizado focar sua atenção nos exemplos mais difíceis. Para combinar as regras brutas em uma, simplesmente escolhe-se as que mais apareceram (tiveram maior destaque/maior peso); ou seja, é realizada uma combinação das regras que classificam mais exemplos.

O classificador final agrega os classificadores aprendidos por votos, mas o voto de cada classificador é uma função de sua eficiência.

O algoritmo de *Boosting* mais estudado é o AdaBoost [30]. Ele é realmente um algoritmo de *Boosting* que pode eficientemente converter um algoritmo de aprendizado fraco real em um algoritmo de aprendizado forte e pode ser utilizado para casos multiclasses.

4.11.3 *Stacking*

O *Stacking* [79] também é um modo de se combinar classificadores. É um pouco diferente do *Bagging* e *Boosting*, pois pode combinar classificadores de diferentes tipos. Contrariamente, a esses dois métodos, o *Stacking* não é muito utilizado pois é difícil de ser analisado teóricamente e porque não existe maneira geral de utilizá-lo.

Stacking introduz o conceito de meta-aprendiz, que substitui o procedimento de voto, para escolher a melhor classificação para uma instância. É de responsabilidade do *Stacking* gerar um classificador, o meta-aprendiz, que descobre melhor como combinar a saída dos aprendizes base.

CAPÍTULO 5

ARQUITETURA DO SISTEMA FOOTSCANAGE

O FootScanAge é um sistema voltado para a estimativa da idade gestacional de recém-nascidos através de suas respectivas imagens da superfície plantar. Em resumo, o sistema FootScanAge processa a imagem de entrada, faz a extração de características da imagem resultante, armazena as características no banco de dados e depois utiliza algoritmos de aprendizado de máquina para estimar a idade gestacional da nova instância sendo processada.

Iniciamente ver-se-á neste capítulo um breve resumo sobre os módulos do sistema e como eles foram construídos. Em seguida, serão apresentadas as fases da mineração de imagens (obtenção, pré-processamento, processamento, mineração e avaliação) e como elas foram implementadas dentro do sistema. Focando no algoritmo de árvores de decisão *fuzzy* e seu funcionamento.

5.1 Módulos do sistema

Para tornar mais robusta e flexível a sua implementação, o sistema foi dividido em 2 módulos integrados, *FootScanAge Web* e *FootScanAge Desktop*, que compartilham um banco de dados comum, cuja modelagem é voltada não apenas para cálculo da IG, como também para outras pesquisas na área de pediatria e neonatologia.

O primeiro módulo, o *FootScanAge Web*, é voltado para a manutenção da base de dados maternos, gestacionais e de neonatos, incluindo diagnósticos e avaliações das fases pré-natal e pós-parto. Esse módulo é disponibilizado como uma aplicação executada em um servidor web utilizando o servidor de páginas Tomcat[1] e implementada em Java[2] com o auxílio da ferramenta JBanana[3].

[1]http://jakarta.apache.org/tomcat/ - Jakarta Apache Tomcat
[2]http://java.sun.com - Java 2 Platform, Standard Edition (J2SE)
[3]http://www.jbanana.org - JBanana Framework

A figura 5.1 mostra a tela de entrada do módulo *FootScanAge Web*, onde no canto superior esquerdo pode ser acessado todo o sistema de cadastro e visualização das informações cadastradas na base de dados.

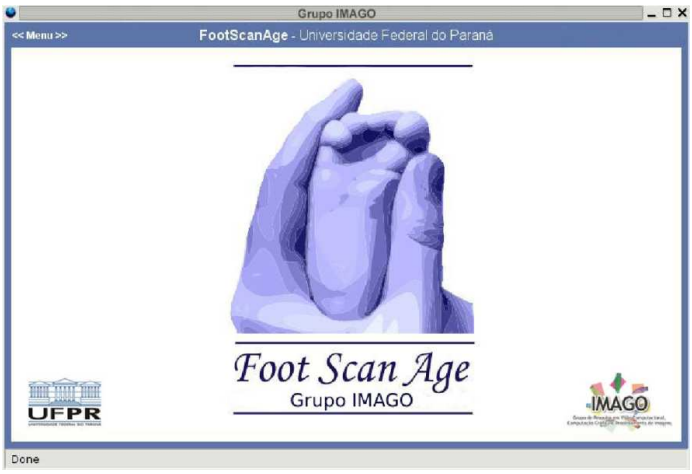


Figura 5.1: Tela inicial do módulo *FootScanAge Web*.

Na tela de cadastro das informações relativas a imagem, visto na figura 5.2, deve ser selecionada a região da impressão digital correspondente à imagem e a data de aquisição da imagem. Através dessa tela é possível fazer o envio da imagem para o servidor, através do *link* ***Selecionar*** e também a visualização da imagem enviada através do *link* ***Visualizar***. Depois de feito o cadastro o usuário deve salvar as informações através do botão **Salvar**. O *FootScanAge Desktop* é ativado através do módulo *FootScanAge Web* ativando o *link* **Analisar imagem**.

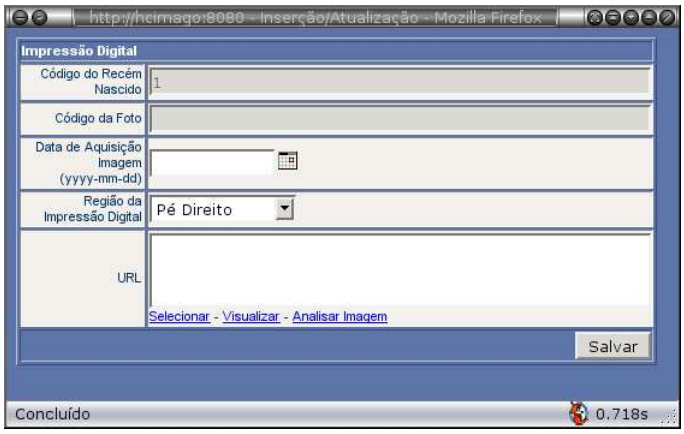


Figura 5.2: Tela de inserção da imagem do Pé no sistema.

O segundo módulo, *FootScanAge Desktop*, também desenvolvido em Java, é responsável pelo processamento das imagens, mineração das imagens e o cálculo da idade gestacional. Para o processamento de imagens foi utilizada a biblioteca JAI[4] (*Java Advanced Imaging*) e os algoritmos de aprendizado de máquina foram obtidos do pacote livre chamado *WEKA* [78].

De agora em diante durante este trabalho quando for necessário referenciar o módulo *FootScanAge Web*, esse será chamado exatamente desta maneira. Quando for citado apenas sistema FootScanAge deseja-se referenciar o módulo *FootScanAge Desktop*, considerado o módulo principal dessa discussão.

5.2 Modelagem dos dados

As tabelas foram modeladas de acordo com a necessidade do armazenamento de informações do prontuário médico e também de características relevantes para a mineração dos dados. O prontuário médico possui várias informações do recém-nascido e de sua mãe, obtidas diretamente com o profissional responsável. Com isso foram geradas várias tabelas dentro do banco de dados, que englobam todas as informações necessárias sobre o recém-nascido, sua mãe e seu parto. Esses dados são preenchidos utilizando o módulo *FootScanAge Web* diretamente pela *internet*.

Para armazenamento dessa base de dados está sendo usado o Sistema Gerenciador de Bancos de Dados (SGBD) PostgreSQL[5], gratuito e disponível livremente.

Para teste dos dados foi criado um banco de dados chamado *FootScanAge*. As tabelas mais importantes para a mineração e o processamento de imagens são descritas a seguir. Também estão detalhados nas tabelas seguintes os atributos existentes nas tabelas e uma breve descrição para cada atributo.

[4]http://java.sun.com/products/java-media/jai/index.jsp
[5]http://www.postgreSQL.org

5.2.1 Tabelas do banco de dados

As tabelas descritas a seguir são as tabelas mais importantes para o processamento de imagens e para a mineração de dados.

Tabela 5.1: Descrição das tabelas utilizadas durante a modelagem.

Tabela	**Descrição**
tbImagemPe	Armazena as características extraídas do pé como um todo.
tbImagemPeRegiao	Armazena características extraídas de cada região.
tbFootScanAge	Armazena as informações importantes para a mineração, como idades gestacionais, modelos de mineração, etc.

Tabela 5.2: Descrição dos atributos da tabela tbImagemPe.

Atributo	**Descrição**
codFoto	Código da imagem
codImagemPe	Código da imagem do pé
anguloCurvaturaTibial	Ângulo da Curvatura interna do pé
anguloCurvaturaFibular	Ângulo da Curvatura externa do pé
alturaCurvaturaTibial	Altura da Curvatura interna do pé
alturaCurvaturaFibular	Altura da Curvatura externa do pé
larguraCurvaturaTibial	Largura da Curvatura interna do pé
larguraCurvaturaFibular	Largura da Curvatura externa do pé
comprimentoHalux	Distância do calcanhar ao halux
momento1	Momento de ordem 1
momento2	Momento de ordem 2
momento3	Momento de ordem 3
momento4	Momento de ordem 4
momento5	Momento de ordem 5
momento6	Momento de ordem 6
momento7	Momento de ordem 7
dataCad	Data de cadastro do pé
limiarBin	Limiar de binarização do pé
urlImagemPre	URL da imagem pré-processada
urlImagemFinal	URL da imagem final (processada)

Tabela 5.3: Descrição dos atributos existentes na tabela tbImagemPeRegiao.

Atributo	**Descrição**
codImagemPe	Código da imagem
codRegiao	Código da região
numRegiao	Número da região $(0-3)$ - Onde 0 indica todo pé
area	Área da região
perimetro	Perímetro da região
largura	Largura da região
altura	Altura da região
mbrTopoX	Ponto superior esquerdo do MRE (eixo X)
mbrTopoY	Ponto superior esquerdo do MRE (eixo Y)
mbrBaseX	Ponto inferior direito do MRE (eixo X)
mbrBaseY	Ponto inferior direito do MRE (eixo Y)
porcentagemOcupacaoPe	% de ocupação da região em relação ao pé
porcentagemOcupacaoMRE	% de ocupação da região em relação ao MRE do pé
porcentagemOcupacaoMRERegiao	% de ocupação da região em relação ao MRE da própria região
alongamento	Relação entre o maior lado do MRE sobre o menor lado
compactação	Relação entre o perímetro ao quadrado e a área
circularidade	Indica o quanto a imagem é circular
porcentagemPregueamento	Número de pixels correspondentes ao pregueamento da superfície plantar

Tabela 5.4: Apresentação dos atributos da tabela tbFootScanAge.

Atributo	**Descrição**
codFSA	Código chave da tabela
codRN	Código do recém-nascido
codFoto	Código da imagem utilizada
codModeloClassificacao	Chave do modelo de mineração de classificação utilizado
codModeloRegressao	Chave do modelo de mineração de regressão utilizado
idadeBallard	IG calculada a partir do método de Ballard
idadeCapurro	IG através do método de Capurro
idadeParkin	IG calculada pelo método de Parkin
idadeClassificacao	IG gerada através do modelo de classificação
idadeRegressao	IG adquirida através do modelo de regressão
idadeFootScanAge	IG calculada através do agrupamento de classificadores
idadeFinal	IG final, definida pelo especialista
dataCad	Data de cadastro das informações na tabela

5.3 Obtenção das imagens

A obtenção das imagens a serem mineradas foi feita em diversas etapas, sendo a coleta das imagens realizada sempre da mesma forma, como visto no capítulo 2 e em [14]. As etapas da obtenção foram:

- Um conjunto de 186 imagens foi obtido no Hospital de Clínicas da UFPR a partir de agosto de 2000, contendo imagens da superfície plantar de vários bebês, sendo alguns prematuros e na sua grande maioria a termo. Essa aquisição inicial foi produto de um esforço para a validação de métodos para obtenção de impressões digitais;
- Ao final dessa etapa, constatou-se que o melhor método de obtenção até o momento é passar tinta no pé do bebê e pressioná-lo contra uma folha de papel *couchet* e digitalizar a impressão plantar através de escaner de mesa. Isso permitiu a obtenção de imagens com maior riqueza de definições.
- Assim o tipo de imagem selecionada para ser utilizada no sistema é uma imagem similar as imagens encontradas na figura 5.3.

(a) (b) (c) (d) (e) (f)

Figura 5.3: Amostra do conjunto de imagens obtido nesta fase.

Desta base inicial foram acrescidos mais 19 imagens de bebês prematuros, totalizando 169 bebês a termo e 36 prematuros. Para auxiliar os testes da ferramenta de processamento e mineração, mais 75 imagens foram adicionadas posteriormente a base de dados.

Cada imagem obtida atualmente é acompanhada de algumas informações do prontuário do bebê como descrito abaixo, sendo as IGs e escores em semanas:

- data de nascimento;
- peso em quilogramas;
- escore Ballard[1];
- escore Parkin;
- idade ecográfica[1];
- idade gestacional ouro[1].

Estes dados presentes em bases de dados e planilhas já existentes, foram convertidas para o novo modelo de dados, descrito nas tabelas 5.2, 5.3 e 5.4.

5.4 Seleção da imagem

Antes de iniciar o processamento e a mineração, deve-se selecionar a imagem a ser utilizada, dentro da aba de obtenção do sistema *FootScanAge Desktop*, mostrada na figura 5.4. A figura citada mostra toda a tela de seleção da imagem, ela é a tela inicial sistema. Na parte superior tem-se um pequeno conjunto de campos responsáveis pela busca da imagem a ser processada, ele contém os campos: código do recém-nascido, registro geral da mãe e nome da mãe. A parte principal da aba de obtenção é composta de uma tabela que apresenta as informações resultantes a busca realizada. Após selecionada a instância correspondente a imagem a ser processada ou minerada, o usuário deve selecionar a tupla desejada da tabela e apertar o botão *OK*.

5.5 Pré-processamento da imagem

Todas as imagens adquiridas foram utilizadas no processamento e extração de características. O pré-processamento da imagem foi realizado utilizando uma seqüência de

[1]Utilizada quando disponível na base de dados.

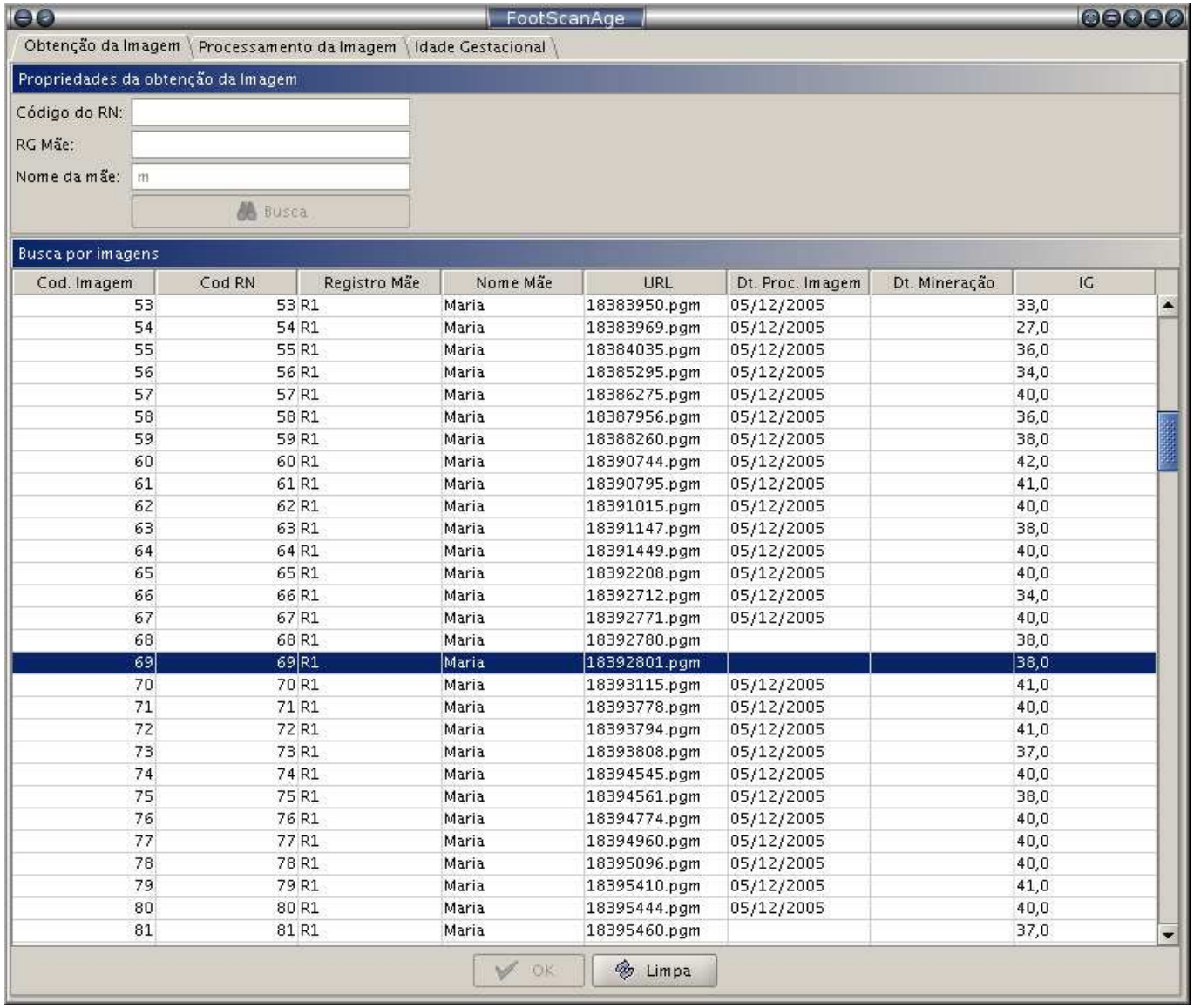


Figura 5.4: Aba de seleção da imagem do sistema FootScanAge Desktop.

passos para limpeza e extração de ruídos. O objetivo específico desta fase é apenas descobrir o menor retângulo envolvente (MRE) do pé sem ruídos, que indica a região de interesse, região de estudo da imagem ou ainda *Region Of Interest* (ROI).

Os passos do pré-processamento são: binarização, filtragem mediana, fechamento, rotação e recorte do MRE, nesta ordem. Um exemplo de pré-processamento pode ser visto na figura 5.5, que mostra as imagens processadas após cada etapa do pré-processamento aplicadas sobre um imagem de entrada.

Dentro do pré-processamento, ocorre na binarização uma transformação da imagem original utilizando como limiar a média dos tons de cinza entre todos os pixels da imagem, tranformando assim a imagem de entrada em uma imagem em preto e branco.

Após a binarização ocorrem cinco iterações (valor ótimo obtido através da aplicação sobre todas as imagens da base) de uma filtragem mediana, com a finalidade de eliminar os pixels externos (eliminar ruídos do tipo *salt-and-pepper* [31]), mas tenta preservar a aparência geral da imagem. A filtragem mediana apenas aplica uma máscara não-linear sobre a imagem, onde para cada posição da imagem, o ponto central é alterado para a média dos valores cobertos pela máscara.

Também é usada uma operação de fechamento seguida de uma abertura, ambas são operações de morfologia matemática [48], que são agrupamentos de outras operações. A abertura, em geral, suaviza o contorno de uma imagem, quebra partes estreitas e elimina pequenas saliências. O fechamento, por sua vez, funde pequenas partes quebradas, alarga as estreitas, elimina pequenos orifícios e preenche pequenos buracos no contorno.

Essas operações de fechamento e abertura são aplicadas para garantir que se tenha apenas o objeto (pé) e que sejam removidas pequenas regiões desconexas.

A rotação é o próximo passo, que tem a finalidade de rotacionar a imagem deixando o pé em posição totalmente vertical e no centro da imagem. Para executar essa rotação, falando resumidamente, é necessária a deteção dos pontos extremos do pé, ponto médio superior e inferior; traçar uma linha entre os pontos detectados e rotacionar o pé até que essa linha se torne paralela ao ângulo de 90°. Esse processo é repetido mais uma vez para que a imagem fique centralizada.

O último passo é a extração do MRE do pé. Para essa extração é necessário o cálculo dos pixels superior esquerdo e inferior direito (retângulo da ROI). O MRE serve para delimitar a região de interesse dentro de toda a imagem. Após a deteção ocorre um corte na imagem original, para que assim seja processada apenas a região de interesse.

Após todas estas fases têm-se a imagem original com a região do pé centralizada, sem ruídos em volta da imagem. A partir da imagem pré-processada dá-se início ao processo de extração de características da imagem.

(a) (b) (c) (d) (e)

Figura 5.5: Fases do pré-processamento: (a) Imagem original; (b) Binarização; (c) Filtragem mediana; (d) Fechamento e (e) Imagem rotacionada.

5.6 Processamento da imagem

O processamento das imagens é a fase mais longa de todo o processo e é responsável por transmitir a imagem original de entrada passando por algumas etapas. Após o processamento de cada etapa, os resultados vão se propagando até a última etapa, terminando assim o processamento daquela imagem.

As fases existentes dentro do processamento são: binarização, rotulação, remoção dos dedos, deteção das regiões e extração de características, sendo esta última a fase principal de todo o processo.

A figura 5.6 mostra o funcionamento da fase de processamento da imagem, iniciando pela imagem original obtida à esquerda, no centro estão todas as fases do processamento e em seguida a imagem processada; e à direita as características extraídas.

Figura 5.6: Resumo da fase de processamento da imagem.

O processamento da imagem é a etapa que provê transformações sobre a imagem para que não existam problemas na extração das características. Então, esta etapa nada mais é do que uma forma de melhorar e alterar a imagem para que seja alcançado o objetivo de aumentar a qualidade da imagem processada e conseqüentemente aumentar a qualidade das informações extraídas.

Algumas técnicas clássicas de processamento de imagens, como: filtragem por mediana, deteção de bordas, limiarização e rotulação de componentes conexas foram estudadas e introduzidas de forma a obter o melhor resultado na extração de características da superfície plantar.

A diferença entre essas e as etapas do pré-processamento, é que naquela fase a imagem é processada automaticamente apenas com o intuito de extrair a ROI da imagem, e nesta, cada etapa do processamento é feita somente sobre a ROI, já determinada no pré-processamento e também pode-se ter a interação do usuário.

A tela de processamento da imagem é apresentada na figura 5.7 e foi implementada de acordo com [72]. No canto inferior esquerdo encontra-se o painel responsável pela visualização da imagem original e a imagem da etapa anterior do processo; na parte inferior central tem-se um painel indicando qual o painel de imagem selecionado; ao lado desse painel, existe o painel de ações, correspondente ao fluxo do processamento da imagem, as ações são: Avançar, Voltar ou Auto; os painéis mais importantes contidos na

imagem são detalhados a seguir.

No canto superior esquerdo da figura 5.7 encontra-se o painel com as etapas do processo, indicando se ela já foi ou não executada, melhor detalhado na figura 5.8; Logo abaixo desse painel, existe um outro painel contendo os parâmetros que podem ser alterados pelo usuário dentro de cada etapa do processamento da imagem, como na figura 5.9.

5.6.1 Binarização

A binarização da imagem é feita através da descoberta de um limiar através de um algoritmo. Dentro do sistema FootScanAge, a binarização pode ser feita utilizando três algoritmos: Otsu [57], Renyi [66] e Média/Mediana.

O método de Otsu é o mais conhecido e utilizado para a binarização de imagens digitais. Ele expõe uma metodologia que encontra um limiar através da minimização da diferença entre variâncias de duas classes: objeto e fundo.

Já Renyi, visa otimizar uma função critério. A binarização usando a Entropia de Renyi usa duas distribuições probabilísticas (objeto e fundo), derivadas da distribuição original dos níveis de cinza de uma imagem e inclui os métodos da soma de Entropia Máxima e da Correlação Entrópica.

O método da média/mediana é uma abordagem simples que determine a média e mediana da distribuição dos valores dos pixels e se faz uma média aritmética entre esses dois valores, chegando assim ao limiar a ser utilizado.

Esta binarização é executada de maneira simples, apenas seleciona os pixels maiores que o limiar descoberto pelo algoritmo escolhido, com o objetivo de transformar uma imagem em tons de cinza em uma imagem binária, ou seja, com apenas pontos pretos e brancos.

5.6.2 Rotulação de componentes conexas

A rotulação de componentes conexas é um método que agrupa regiões da imagem que apresentam alguma similaridade, cor ou tons de cinza, por exemplo. A imagem binária

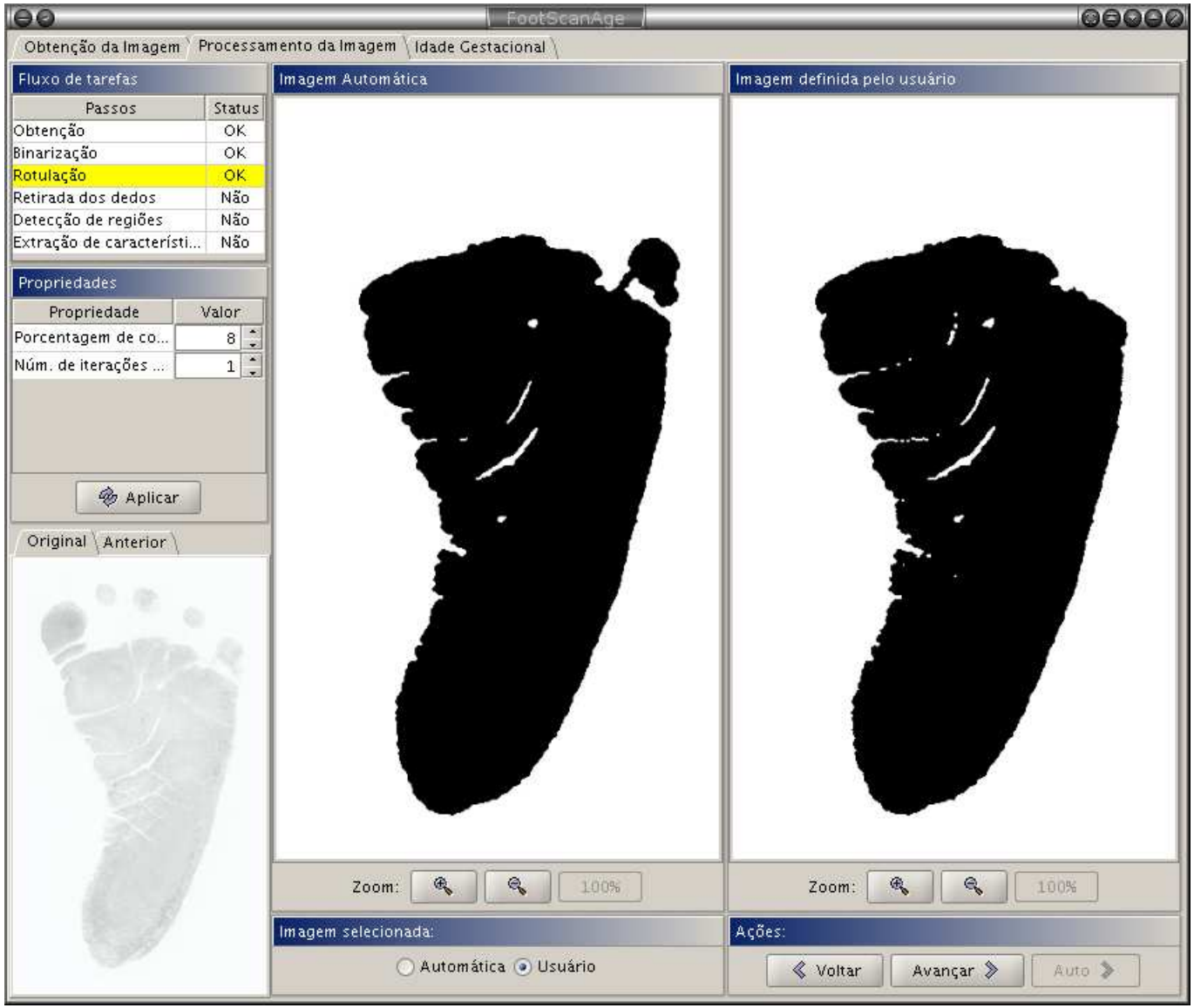


Figura 5.7: Tela de processamento da imagem do sistema FootScanAge.

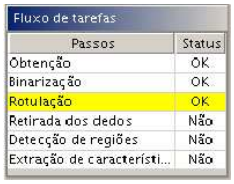


Figura 5.8: Painel de fluxo de tarefas.

Figura 5.9: Painel de parâmetros de entrada.

passa pelo algoritmo iterativo [70] que detecta quais regiões dentro da imagem são conexas, ou seja, estão ligadas entre si e atribui a elas um mesmo rótulo.

Após a deteção das componentes, apenas componentes conexas maiores que uma porcentagem (parâmetro entrado pelo usuário) estarão contidas na imagem resultante dessa fase. Ao eliminar as outras componentes, os dedos podem, ou não, ser retirados totalmente. Isto vai depender da imagem de entrada, caso os dedos estiverem "grudados" ao pé, eles não serão retirados nesta fase, somente na fase seguinte.

5.6.3 Retirada dos dedos

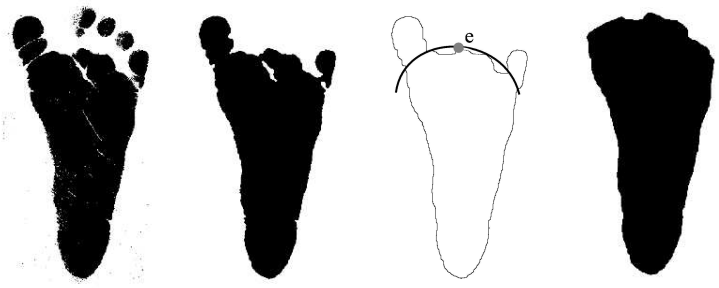
Estudos preliminares do método FootScanAge mostram que a região da superfície planar sem os dedos é mais apropriada para o processo de extração de características.

Esta etapa foi projetada de forma a proporcionar a retirada total dos dedos, já que a etapa anterior não consegue remover os dedos unidos ao pé.

Caso os dedos não tenham sido totalmente retirados na fase anterior, ou seja, alguns (ou todos) dedos se encontram na mesma componente conexa da base do pé, nesta fase faz-se a remoção dos dedos utilizando um algoritmo de corte. Esse algoritmo se utiliza de uma parábola, gerada através de informações do contorno do pé e um parâmetro de entrada, para tentar eliminar os dedos da imagem. A eliminação ocorre de forma simples, sendo que os pixels localizados acima da parábola são removidos, conseqüentemente eliminando

os dedos da imagem. Todos as subfases são apresentadas na figura 5.10, indicando as imagens intermediárias resultantes para cada subfase desta etapa.

Após a eliminação dos pixels, alguns pixels (ruídos) são removidos utilizando um fechamento simples.



(a) (b) (c) (d)

Figura 5.10: Imagens intermediárias da fase de remoção dos dedos: (a) Imagem binária; (b) Remoção das menores componentes conexas; (c) Párabola utilizada para corte; (d) Imagem final do pé sem os dedos.

Caso o usuário não fique satisfeito com a remoção proporcionada pelo algoritmo de corte, ele pode alterar o parâmetro de entrada e tentar uma nova remoção. Este procedimento de repetição pode ser aplicado praticamente em todas as etapas do processamento.

5.6.4 Deteção de regiões

Com base em estudos relativos a superfície plantar do recém-nascido [14], os especialistas chegaram à algumas hipóteses sobre essa região do pé. As hipóteses relativas as características extraídas do pé, mostram que seria interessante estudar a superfície plantar por regiões, sendo assim, no sistema FootScanAge, o comprimento do pé foi dividido em três regiões, para uma concretização dessa teoria.

Já com a imagem do pé sem os dedos, a fase de deteção de regiões determina três regiões de mesma altura dentro da imagem, essas regiões podem ser alteradas de acordo com a interação com o usuário, através de duas barras de movimentação entre as regiões.

Dentro do sistema, as regiões são representadas em diferentes tons de cinza para melhor identificação de cada uma delas.

As regiões são importantes para a próxima fase, a fase de extração de características, na qual as características de cada região são extraídas, juntamente com as características globais do pé.

5.6.5 Extração de características

A fase extração de características é feita ao final de todo processamento e nela são calculados os valores para as características da imagem, descritas nas tabelas 5.2 e 5.3.

A abordagem de extração de características da mineração de imagens do sistema FootScanAge é considerada híbrida: semi-automática, pois necessita da interação com o usuário, e; orientada a dados, pois se baseia nas informações dos pixels da imagem para determinar as características da imagem.

É de responsabilidade desta fase extrair as características, calcular os MREs das regiões e extrair deles informações tais como: largura, altura, porcentagem de ocupação, etc. Toda a extração das características é feita a partir de uma imagem já binarizada, sem os dedos, dilatada e dividida em três regiões.

O sistema compara as características recém processadas com as médias e desvios-padrões das mesmas características de imagens já processadas que pertencem a mesma idade gestacional. Desta maneira, o sistema permite que o usuário valide os dados gerados: se aceitar, o sistema cadastra os dados no banco; caso contrário, o usuário pode refazer todo o processamento ou apenas partes dele.

Por fim, após a validação, essas características são gravadas no banco de dados (nas tabelas 5.2 e 5.3). Mais tarde elas serão utilizadas na fase onde ocorre a mineração propriamente dita.

Algumas características já apresentadas anteriormente têm seus conceitos mais bem detalhados a seguir, também mostrando como elas foram extraídas das imagens.

No final desta etapa, a tela de processamento de imagens sofre uma alteração para apresentar as características extraídas da imagem, substituindo o painel de visualização da

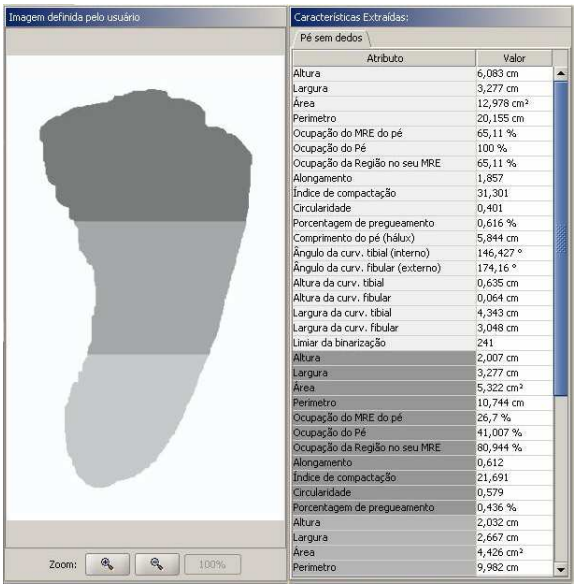


Figura 5.11: Imagem final processada e características extraídas.

imagem da direita por um painel contendo uma tabela relativa as características extraídas, como apresentado na figura 5.11.

5.6.5.1 Perímetro e Área

O perímetro de um objeto ou região corresponde ao número total de pixels que constituem o contorno fechado do objeto ou região em vizinhança 8 [48].

Já a área de um objeto ou região corresponde ao número total de pixels que constituem o objeto ou região em vizinhança 8.

5.6.5.2 Alongamento e Compactação

A razão entre a largura e a altura do menor retângulo envolvente é chamada de alongamento. A compactação é uma característica que segue a fórmula:

$$C = \frac{P^2}{A} \tag{5.1}$$

onde P corresponde ao perímetro e A a área da região analisada.

5.6.5.3 Circularidade

A circularidade de um objeto simples ou de uma região traduz a sua semelhança em relação a uma circunferência. Um objeto é mais circular quanto mais a sua circularidade for próxima de 1, seguindo a fórmula:

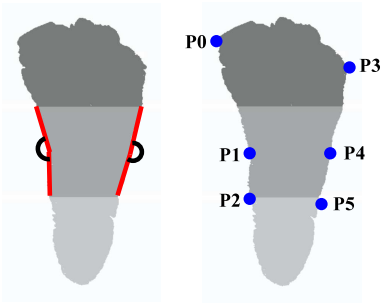
$$Circ = \frac{4\pi A}{P^2} \tag{5.2}$$

onde P e A correspondem ao mesmo que na equação 5.1.

5.6.5.4 Curvatura do Pé

Na realidade a curvatura do pé não corresponde a uma única característica e, sim em três. Então, nesta fase deve-se subdividir a curvatura nessas outras três características mais específicas: ângulo da curvatura, altura da curvatura e largura da curvatura.

A extração das características da curvatura é realizada a partir de três pontos de cada uma das laterais da superfície plantar. A relação entre os pontos laterais formam as três características extraídas, já citadas anteriormente. A figura 5.12 mostra os três pontos selecionados de cada lado do pé (P_0, P_1, P_2, P_3, P_4 e P_5) para a imagem exemplo.



(a) (b)

Figura 5.12: Extração das características da curvatura do pé: (a) Ângulos da curvatura; (b) Pontos selecionados.

5.6.5.5 Momentos centrais e invariantes

A forma dos segmentos de borda pode ser descrita quantitativamente através dos momentos, que são considerados os métodos mais populares [31]. A vantagem dos momentos em relação a outras técnicas é que sua implementação é direta, além deles carregarem uma interpretação "física" da fronteira de uma forma. Os momentos usados aqui são denominados invariantes, devido as suas propriedades de invariância a transformações geométricas.

Os parâmetros de forma baseados nos momentos invariantes têm como objetivo representar o contorno fechado de um objeto através das suas propriedades estatísticas. Normalmente, os momentos invariantes calculados são sete e baseiam-se nos momentos centrais de segunda e terceira ordem. A razão pela qual são calculados estes sete momentos invariantes deve-se ao fato de que a partir de certa ordem não se obtêm mais detalhes do que é necessário e distinguível pelo ser humano. O parâmetro da forma baseado nos momentos poderá incluir todos ou apenas alguns dos momentos acima definidos, dependendo das necessidades da aplicação.

5.6.5.6 Excentricidade

A excentricidade é um parâmetro de forma geométrico inerente para o contorno fechado de um simples objeto e permite representar o objeto como a relação geométrica do seu contorno fechado. Este parâmetro tem a vantagem da invariância a transformações geométricas, tais como rotação, translação, mudança de escala e ponto inicial.

A excentricidade de um objeto simples ou região é definida como a relação existente entre seu R_{max}, e seu R_{min}. Os vários raios são definidos como a distância entre a centróide do objeto e o menor elipse envolvente do objeto. A expressão a seguir define excentricidade (E):

$$E = \frac{R_{max}}{R_{min}} \tag{5.3}$$

5.7 Limpeza dos dados

Ao final do processamento de imagens, as informações são resgatadas do banco de dados, das tabelas já citadas e é aplicado um filtro de limpeza sobre o conjunto resgatado.

Para a limpeza, o sistema executa uma pré-classificação da mesma antes de ser inserida na base de dados e estima uma possível idade gestacional. Para isso, é criado um classificador simples, gerado através do algoritmo M5'P, treinado a partir da própria base de dados, para que assim sejam gerados menos erros, ou seja, somente os *outliers* sejam identificados.

O algoritmo estima uma idade gestacional para cada instância da base de dados. Com base nesta possível idade gestacional, a idade predita, é feita a diferença com a idade gestacional real, caso essa diferença seja maior que um valor pré-definido (nesse caso, igual a 4), então essa instância é considerada um *outlier* e é removida do conjunto de dados. Quando executado sobre a base de dados existente esse filtro retira aproximadamente 5% das instâncias.

5.8 Mineração de dados

Com os dados limpos em mãos, inicia-se realmente a mineração dos dados. Para melhor detalhamento deste tema, será seguida a ordem de geração, seleção e aplicação do modelo de mineração.

Para minerar os dados foi escolhido um atributo que será considerado a classe, ou seja, o atributo que representa as idades gestacionais dos bebês, que nesse caso é a idade gestacional gerada através do método de Parkin. O algoritmo de aprendizado de máquina irá utilizar esse atributo classe para aprender. A idade gestacional ideal seria uma idade gestacional padrão-ouro, como já descrito anteriormente, para que assim seja possível gerar resultados absolutamente confiáveis.

A mineração foi feita sobre o resultado de uma consulta sobre as tabelas tbImagemPe, tbImagemPeRegiao e tbFootScanAge (já descritas na seção 5.2). Existem algumas consultas pré-definidas focando em diferentes características, sendo que esse conjunto de con-

sultas pode ser modificado mais tarde para englobar mais tabelas e mais atributos para a mineração. A consulta utilizada atualmente engloba apenas os atributos selecionados para a mineração de dados.

A implementação da mineração de dados foi feita utilizando o pacote *WEKA*, distribuído gratuitamente na internet. O *WEKA* se trata de uma ferramenta para a mineração de dados que possibilita fazer testes utilizando diferentes tipos de algoritmos e comparando os seus resultados sobre a base de teste. Foram feitos vários testes com a base de dados utilizando diversos algoritmos existentes no pacote *WEKA*, testando o desempenho dos algoritmos existentes no pacote. Para testar cada algoritmo isoladamente foi utilizado o programa *WEKA Explorer* e alguns testes feitos com a base de dados serão mostrados no capítulo de testes.

Dentro da fase de mineração de dados do sistema, os algoritmos podem ser escolhidos e agrupados pelo usuário de várias maneiras. Os resultados podem ser analisados pelo especialista e assim, através dessa gama de possibilidades, a idade gestacional final pode ser identificada.

A figura 5.13 apresenta a tela de determinação da idade gestacional ou aba de mineração dos dados. No canto superior esquerdo encontra-se um painel com as imagens original e processada; Abaixo desse painel, existe a tabela com as características extraídas da imagem selecionada dentro da etapa de processamento da imagem. Os outros painéis contidos na imagem são detalhados dentro das próximas seções.

5.8.1 Geração do modelo de mineração

Podem ser gerados vários modelos de classificação ou regressão. Para criar um modelo, deve-se somente escolher os algoritmos que farão parte do modelo, como pode ser visto na figura 5.14. Tanto para o modelo de classificação e regressão, após a execução dos algoritmos da abordagem sobre o conjunto de treinamento, é gerado o modelo de mineração.

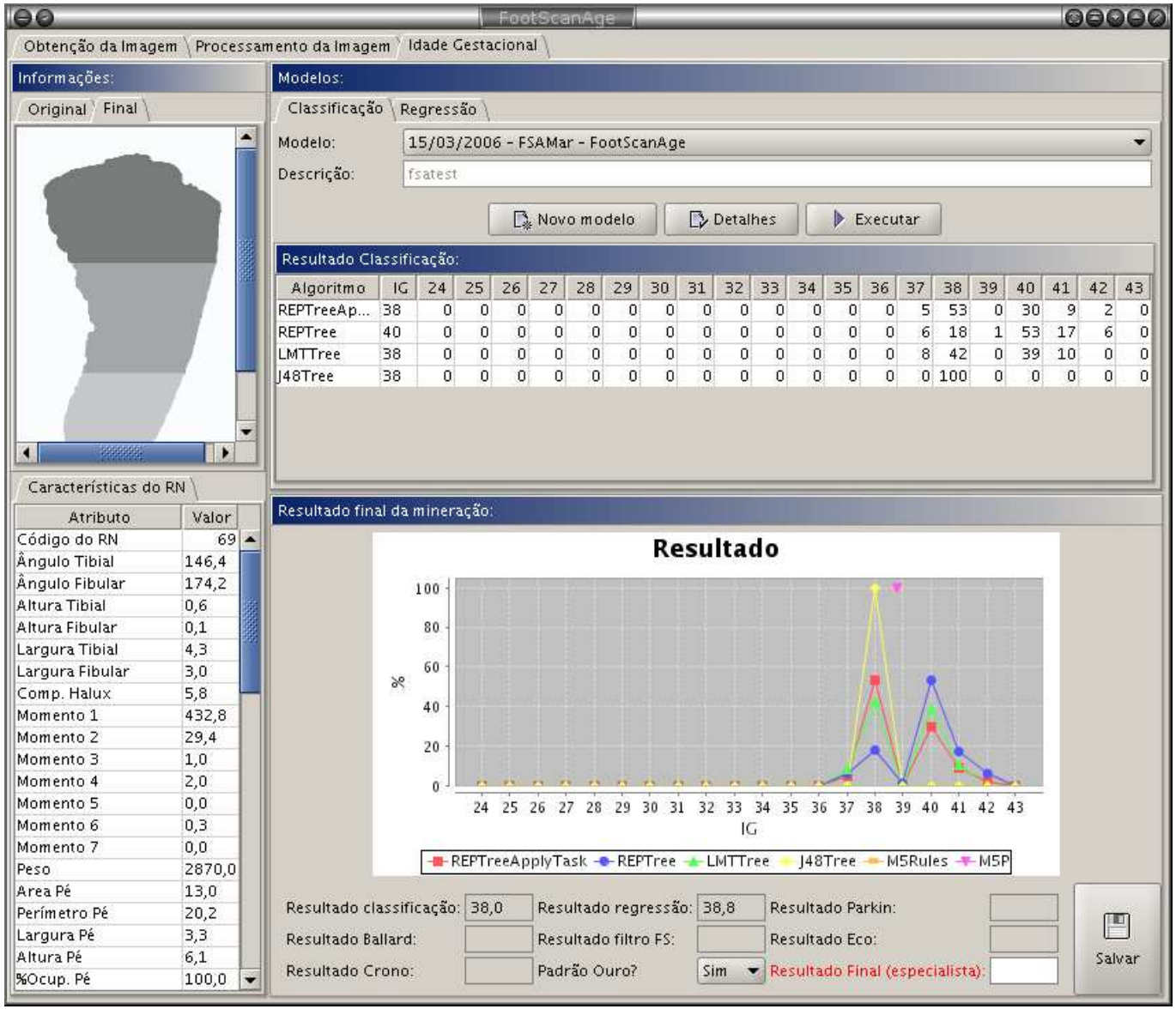


Figura 5.13: Tela de mineração de imagens do sistema FootScanAge.

Figura 5.14: Geração de modelos da Mineração

Para utilizar os algoritmos de classificação, foi necessário discretizar manualmente o atributo classe (IG) em 20 classes (24-43 semanas), foi escolhido esse intervalo pois ele engloba o grupo de bebês que se deseja analisar, incluindo também os bebês prematuros.

Testes iniciais foram feitos sobre o conjunto de dados e mostraram que os melhores algoritmos de classificação disponíveis são: J4.8 [64], REPTree [64] e LMTree [45]. Todos estes algoritmos utilizam de árvores de decisão para classificar as instâncias e estão disponíveis para seleção do usuário. O algoritmo SDT[60], que usa árvores de decisão fuzzy, também está disponível dentro do sistema e foi implementado baseando-se no algoritmo de Peng [60], já citado na seção 4.10.9, com algumas alterações apresentadas na seção 5.8.1.1. Os algoritmos citados acima também podem ser agrupados e assim melhor analisar os seus resultados.

Já para os algoritmos de regressão não é preciso discretizar o atributo classe, porque o algoritmo de regressão aprende tentando acertar exatamente o valor final, sendo assim mais próximos do resultado esperado do que os algoritmos de classificação, possuindo um erro aproximado da idade predita à idade real.

Na regressão, se encontram presentes o algoritmo *M5'P* [63], que combina árvores de decisão e regressão linear [78] nas folhas; e o algoritmo *M5'Rules* [36], que utiliza regras extraídas de árvores de decisão. Ambos algoritmos foram melhor detalhados na seções 4.10.4 e 4.10.5, respectivamente.

O objetivo é que conforme a base de dados cresça, a base de conhecimento também amadureça. Também deve ser observado que o propósito destes modelos não é apenas o

cálculo da idade gestacional, como também servir de base para outras análises, uma vez que o cadastro disponibilizado pelo sistema FootScanAge é bastante amplo.

5.8.1.1 Implementação do algoritmo SDT

A implementação do algoritmo de árvore de decisão *fuzzy* utilizado no sistema FootScanAge baseou-se no algoritmo citado por Peng [60], já referenciado na seção 4.10.9.

As diferenças principais entre o algoritmo implementado e o algoritmo de Peng está na seleção dos atributos, no critério de poda das sub-árvores, na utilização do limiar de corte e do parâmetro Delta.

No algoritmo implementado, foi utilizada a mesma abordagem de escolha de atributos que no C4.5. que é similar a do seu predecessor (ID3), que escolhe o melhor atributo através de medidas de entropia e ganho de informação sobre o conjunto de instâncias de treino. A alteração na forma de seleção de atributos foi implementada pois o algoritmo de Peng se mostrou muito frágil e custosa. É importante dizer que os atributos não podem ser escolhidos para mais de um nó da árvore.

Com descrito na seção 4.10.9, a implementação da classificação de novas instâncias deste algoritmo poderia ser feita de duas maneiras, na implementação descrita aqui nesta seção, optou-se por utilizar a média aritmética entre os valores de todas as funções de pertinência do caminho que leva da raiz ao nó folha, para que assim os nós pertencentes ao caminho percorrido pela instância dentro da árvore sejam considerados e influenciem no resultado final do classificador.

Já o critério de poda implementado foi simples:

- É realizada uma busca na árvore, buscando por nós que possuam folhas com o mesmo atributo classe. Caso são encontradas, elas são podadas e o nó pai se transforma em uma folha com a classificação igual a dos filhos;

- Caso o número de instâncias que chegam a uma certa folha for menor que um número mínimo de objetos (nesse caso é igual a 2) então a folha é cortada. Se apenas um dos filhos do nó corrente tenha essa característica, então o outro nó se torna o nó

corrente.

A utilização do limiar de corte e do parâmetro Delta foi feita de maneira que Delta se tornasse um parâmetro que corresponde a porcentagem do limiar de corte que o nó englobará ($delta * T$). Por exemplo, ao classificar uma nova instância i com valor do atributo A igual a a_i, seja menor que o limiar T, somado com uma porcentagem de T, ou seja, $a_i < T + (delta * T)$, então essa instância possui um certo grau de pertinência ao filho esquerdo da árvore. Similarmente se $a_i > T - (delta * T)$ então também possui um certo grau de pertinência ao lado direito dessa árvore. Isso pode ser melhor visualizado na figura 5.15.

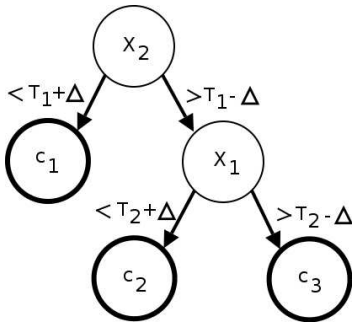


Figura 5.15: Exemplo de árvore de decisão *fuzzy* gerada pelo algoritmo SDT.

Durante a implementação e execução do algoritmo sobre a base de dados, descobriu-se alguns problemas do algoritmo implementado, entre eles:

- Ajuste dos parâmetros de entrada: para cada conjunto de dados novo deve-se fazer o ajuste. Depois de teste empíricos sobre a base de dados, o melhor conjunto de parâmetros descoberto foi: alpha=0.15, beta=0.5, lambda=0.45 e delta=0.45;
- O algoritmo de poda: pois quanto maior a quantidade de dados, maior o tamanho da árvore final, fazendo assim com que a construção da árvore e aplicação do modelo sobre novos exemplos seja mais lenta.

5.8.1.2 Seleção das características

Após um estudo feito com as 58 características extraídas automaticamente das imagens existentes, foi descoberto que apenas 9 dessas características podem ser consideradas com-

pletamente relevantes para a mineração de dados, ou seja, elas sozinhas provêem um bom índice de classificação. Essas características foram descobertas a partir de vários testes de seleção de melhores atributos, utilizando os algoritmos disponíveis no pacote *Weka Explorer* (aba *select attributes*), os resultados foram: *moment1, moment2, moment7, area0, perimeter0, elongatedness0, compactness0, circularity0* e *percentualCisureOccupation0*.

Com base nos estudos citados acima, foi criado um classificador utilizando apenas os 9 atributos mais votados descritos na tabela. Esse classificador criado mostrou resultados melhores do que o classificador que utilizava todas as características extraídas (seção 5.10.1). A exclusão destes atributos também deixou a criação do modelo do classificador mais rápida, pois com isso a dimensionalidade do problema diminuiu.

Esse resultado interessantemente demonstrou que a fase de divisão de regiões pode ser retirada do processamento de imagens, pois as características extraídas das regiões não demonstraram nenhuma evolução dentro da mineração de dados. Dentro dos 9 atributos selecionados, todos são atributos relativos ao pé como um todo e não das regiões.

Os atributos não selecionados para a mineração e a respectiva descrição, foram:

- *heightRegionX*[6]: como as características foram extraídas de forma automática das imagens, as regiões dentro do pé foram divididas em tamanhos iguais, logo, elas estavam trazendo informações irrelevantes para a mineração;
- *haluxLength* (Distância do calcanhar ao hálux): É uma informação proporcionalmente muito similar à informação da altura do pé (*heightRegion0*) e por isso acaba se mostrando uma informação redundante;
- Atributos das regiões: Não trouxeram nenhuma informação adicional que possibilitasse o aumento na porcentagem de acerto do classificador;
- Outros atributos: Não se mostraram consistentes dentro da distribuição dos dados, em alguns momentos interferiram muito na decisão do algoritmo de classificação.

Segue uma breve descrição dos atributos selecionados para a mineração:

[6]Sendo X um valor entre 0 e 3

- *moment1, moment2* e *moment7*: dentre os momentos extraídos, estes 3 mostraram maior importância;

- *area0* e *perimeter0*: são atributos que correspondem a características físicas dos pés, indicando a proporção dos tamanho do pé com a idade gestacional;

- *elongatedness0, compactness0* e *circularity0*: são características correlativas diretamente a forma das imagens. Isso expressa que a forma dos pés também está diretamente ligado a idade gestacional.

- *percentualCisureOccupation0* (Porcentagem de pregueamento): É um atributo que ainda não foi muito bem explorado devido a qualidade das imagens adquiridas, mas mesmo assim foi selecionado para auxiliar na classificação dos pés;

Ainda foi realizado um outro estudo que mostrou que apenas duas das variáveis (características): *heightRegion0*, *widthRegion0*, já são responsáveis por bons resultados, similares aos encontrados utilizando as 9 características citadas.

Não se pode confirmar ainda a irrelevância dos atributos não selecionados para a mineração de dados, pois, depois de adquiridas novas imagens com melhor qualidade estas características podem mostrar certa relevância.

5.8.2 Seleção do modelo de mineração

Após gerado um modelo na fase anterior, ocorre a fase de seleção do modelo. O usuário pode selecionar o modelo que acabou de criar ou selecionar um modelo já existente, tanto de regressão como de classificação.

Podem ser criados modelos em diferentes datas, usando diferentes atributos e algoritmos, como já dito.

Para seleção e criação do modelo de mineração dentro do sistema FootScanAge deve se utilizar o painel mostrado na figura 5.16. Neste painel deve-se selecionar o modelo dentro da caixa de seleção de modelo ou criar um novo modelo apertando o botão **Novo Modelo**; para visualizar detalhes sobre o modelo selecionado através do botão **Detalhes**;

e para a execução do modelo deve-se apenas utilizar o botão **Executar**. Sendo o mesmo funcionamento para ambas as abas de classificação e regressão.

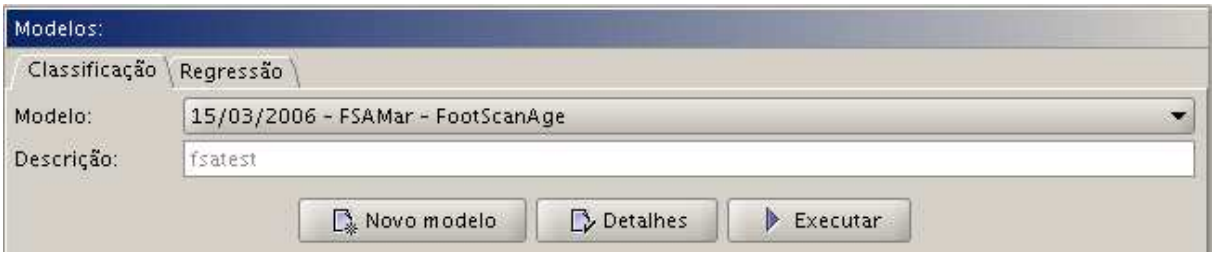


Figura 5.16: Painel de seleção e criação do modelo de mineração a ser aplicado sobre a instância corrente.

5.8.2.1 Aplicação do modelo de mineração

O modelo selecionado é aplicado sobre os dados da nova imagem, como exemplificado na figura 5.17, que resume a fase aplicação do modelo de mineração de dados. Através da figura pode-se visualizar uma nova instância, a aplicação dos modelos de classificação e/ou regressão, a estimativa das IGs para cada algoritmo e o conjunto de resultados para uma estimativa final feita pelo especialista responsável.

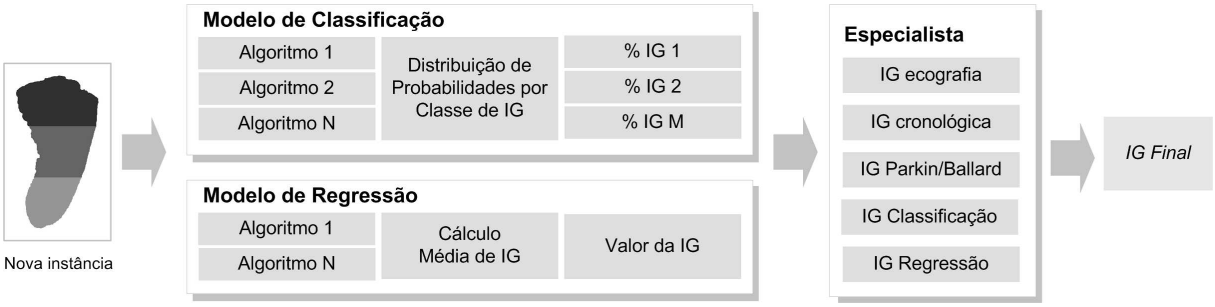


Figura 5.17: Aplicação do modelo de mineração.

Durante a aplicação do modelo, o sistema calcula e propõe um valor para a idade gestacional: no caso do modelo de classificação, consiste na distribuição de probabilidade por idade/classe, mostrando os algoritmos utilizados e a porcentagem de probabilidade para cada classe, como apresentado na figura 5.18; no modelo de regressão o valor obtido é a idade gestacional propriamente dita.

Resultado Classificação:

Algoritmo	IG	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
REPTreeAp...	38	0	0	0	0	0	0	0	0	0	0	0	0	0	5	53	0	30	9	2	0
REPTree	40	0	0	0	0	0	0	0	0	0	0	0	0	0	6	18	1	53	17	6	0
LMTTree	38	0	0	0	0	0	0	0	0	0	0	0	0	0	8	42	0	39	10	0	0
J48Tree	38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0

Figura 5.18: Tabela com o resultado da mineração para o modelo de classificação aplicado.

Podem ser aplicados os dois modelos ao mesmo tempo, para a mesma instância, assim sendo possível comparar os resultados dos modelos.

Os resultados também podem ser vistos através de um gráfico, destacado na figura 5.19. Esse gráfico também é utilizado para melhor visualização das probabilidades para cada algoritmo, cada um representado por uma linha de uma certa cor. Ainda na figura 5.19 é possível visualizar componentes caixas de texto com as IGs estimadas através de outros métodos e também a IG a ser validada pelo especialista. Com base na análise de todos os resultados e dos dados do recém-nascido, o especialista indicará qual a idade gestacional para o neonato, que deve salvar a IG final através do botão **Salvar**.Uma vez confirmada e salva a classificação, o sistema poderá usar essa nova imagem como base para geração de um novo modelo.

Figura 5.19: Gráfico com o resultado da mineração para os modelos aplicados e as estimativas das IGs.

5.9 Refinamento

No decorrer do desenvolvimento do projeto não foi utilizado nenhum pós-processamento ou refinamento dos dados, pois os resultados da mineração são objetivos e são armazenados diretamente no banco de dados, juntamente com a *URL* (*Universal Resource Locator*) das imagens (original e final).

5.10 Testes e resultados

O aprendizado dos algoritmos de mineração depende de um atributo-alvo, que neste caso é a própria idade gestacional. Entretanto, a IG ideal não é necessariamente a indicada pelo especialista, e sim a que é conhecida como idade gestacional padrão-ouro, obtida a partir de gestações que tenham tido um acompanhamento pré-natal consistente.

Para realizar os testes preliminares, foi construída uma base de dados a partir de 280 imagens de recém-nascidos obtidas ao longo dos estudos iniciais do projeto, sendo o atributo-alvo derivado a partir de idade calculada através do método de Parkin. Nesta seção são apresentados os resultados alcançados pela ferramenta desenvolvida.

5.10.1 Apresentação dos resultados

Em todos os testes a serem descritos, a base de dados de referência foi exportada e analisada pelos algoritmos envolvidos no sistema através de testes usando validação cruzada com 10 partições.

Também foram realizados testes utilizando o peso do recém-nascido em kilogramas como sendo uma característica auxiliar no aprendizado juntamente com as características selecionadas, para mais tarde poder analizar a influência dessa característica nos resultados finais dos classificadores.

5.10.1.1 Resultados dos algoritmos de regressão

As tabelas seguintes mostram o resultado dos testes para os algoritmos de regressão do sistema, utilizando os 9 atributos selecionados. Os algoritmos também são testados usando

o *Bagging* como meta-classificador e os algoritmos do sistema como algoritmos base. As colunas das tabelas são: algoritmo testado, tempo de criação do modelo de mineração em milisegundos e a distância entre a IG real e a IG predita pelo classificador, sendo o erro médio absoluto em semanas.

Os primeiros resultados do sistema a serem apresentados não utilizam o filtro de limpeza descrito na seção 5.7. A tabela 5.5 mostra esses resultados.

Algoritmo	**Tempo**	E_{MA}
M5'P	222.0	1.3184
M5'Rules	254.0	1.3216
Bagged M5'P	1879.0	1.3284
Bagged M5'Rules	3100.0	1.3106

Tabela 5.5: Primeiros resultados obtidos pelos algoritmos de regressão.

Já a tabela 5.6 mostra resultados similares aos da tabela anterior, mas agora fazendo uso do filtro de limpeza descrito na seção 5.7.

Algoritmo	**Tempo**	E_{MA}
M5'P	186.0	1.1702
M5'Rules	218.0	1.1919
Bagged M5'P	1632.0	1.1102
Bagged M5'Rules	2756.0	1.1117

Tabela 5.6: Resultados dos algoritmos de regressão, utilizando o filtro de limpeza.

A tabela 5.7 mostra os resultados alcançados usando o peso como característica auxiliar no aprendizado do classificador.

Algoritmo	**Tempo**	E_{MA}
M5'P	182.0	1.1535
M5'Rules	213.0	1.1557
Bagged M5'P	1723.0	1.1217
Bagged M5'Rules	2407.0	1.1330

Tabela 5.7: Resultados dos algoritmos de regressão, utilizando o peso.

Os resultados da tabela 5.8 foram alcançados usando o filtro de limpeza descrito na seção 5.7 e o peso do recém-nascido.

Algoritmo	**Tempo**	E_{MA}
M5'P	185.0	1.1423
M5'Rules	214.0	1.1637
Bagged M5'P	1704.0	1.1027
Bagged M5'Rules	2444.0	1.1070

Tabela 5.8: Resultados dos algoritmos de regressão, utilizando o peso como característica auxiliar.

Não foi gerada uma tabela contendo o tamanho das árvores geradas pelos algoritmos de regressão, pois o tamanho dessas árvores está relacionado com a quantidade de modelos lineares (M5'P) e a quantidade de regras extraídas (M5'Rules), ou seja, o número de folhas. Este número se mostrou relativamente constante, variando apenas entre 2 e 3.

5.10.1.2 Resultados dos algoritmos de classificação

Para utilizar os algoritmos de classificação foi necessário discretizar o atributo-alvo (IG) em 20 classes (24-43 semanas), que representam o espaço de possibilidades das instâncias existentes no mundo real. É importante citar que todos os testes foram feitos usando os 9 atributos selecionados da base de dados, já descritos anteriormente. Aqui também foram testados os algoritmos como sendo base do algoritmo de *Bagging*.

Nas próximas tabelas 5.9, 5.10, 5.11 e 5.12 estão os resultados dos algoritmos de classificação, onde os resultados correspondem à porcentagem de acerto de cada algoritmo, em relação à classe predita pelo algoritmo para a instância e a sua classe real. Na primeira coluna de cada tabela tem-se o algoritmo utilizado e as colunas seguintes mostram o tempo de geração do modelo e o resultado final, ou seja, a porcentagem de acerto na classificação das instâncias.

A tabela 5.9 mostra os resultados dos algoritmos sem utilizar o filtro de limpeza de instâncias.

Algoritmo	**Tempo**	**Resultado**
SDT	809.0	32.1167
J4.8	151.0	31.7518
REPTree	69.0	39.7810
LMT	15058.0	39.4160
Bagged SDT	6457.0	39.7810
Bagged J4.8	929.0	34.3065
Bagged REPTree	581.0	37.5912
Bagged LMT	257757.0	29.9270

Tabela 5.9: Resultados obtidos pelos algoritmos de classificação.

Já tabela 5.10 mostra os resultados usando o filtro de limpeza.

Algoritmo	**Tempo**	**Resultado**
SDT	481.0	37.7431
J4.8	98.0	29.1828
REPTree	58.0	42.0233
LMT	11995.0	41.6342
Bagged SDT	4750.0	42.8015
Bagged J4.8	929.0	35.4085
Bagged REPTree	758.0	38.1322
Bagged LMT	359647.0	34.2412

Tabela 5.10: Resultados dos algoritmos de classificação, utilizando filtro de limpeza.

Ao usar o peso como característica auxiliar, os resultados são os encontrados na tabela 5.11.

Algoritmo	**Tempo**	**Resultado**
SDT	629.0	40.2214
J4.8	84.0	33.5793
REPTree	61.0	42.0664
LMT	9346.0	43.5424
Bagged SDT	6784.0	42.8044
Bagged J4.8	832.0	34.3173
Bagged REPTree	592.0	39.1143
Bagged LMT	279831.0	33.2103

Tabela 5.11: Resultados da classificação, usando o peso como característica auxiliar.

Usando o filtro de limpeza juntamente com o peso dentro do conjunto de características tem-se os resultados encontrados na tabela 5.12.

Algoritmo	**Tempo**	**Resultado**
SDT	504.0	43.9393
J4.8	80.0	33.7121
REPTree	63.0	46.2121
LMT	8031.0	43.1818
Bagged SDT	4774.0	44.6969
Bagged J4.8	799.0	36.3636
Bagged REPTree	549.0	43.1818
Bagged LMT	391740.0	36.3636

Tabela 5.12: Resultados da classificação, usando o peso e filtro de limpeza.

A tabela 5.13, mostra o tamanho da árvore gerada pelos algoritmos, também utilizando o filtro de limpeza em 5 iterações diferentes, e a média de tamanho para cada algoritmo.

Iteração	**J4.8**	**REPTree**	**LMT**	**SDT**
1	99.0	9	1	353.0
2	105.0	7	1	311.0
3	111.0	19	1	425.0
4	91.0	9	1	329.0
5	105.0	15	1	389.0
Média	102.2	11.8	1	361.4

Tabela 5.13: Tamanho das árvores geradas pela classificação.

Em seguida são apresentadas três tabelas com a distribuição de probabilidades para os algoritmos sobre instâncias retiradas aleatóriamente da base de dados usando filtro de limpeza. Nas tabelas seguintes são apresentadas apenas as distribuições das classes 32 a 42, pois este é o intervalo onde se encontram todas as probabilidades para as instâncias selecionadas. Os resultados são apresentados em forma de porcentagem arredondada e a idade gestacional real aparece em destaque na cor vermelha. Uma observação importante é que

em alguns casos o somatório da distribuição de probabilidades para o algorimto LMT não chega a 100% devido a omissão dos resultados para as classes entre 24-31 e a classe 43.

Algoritmo	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
SDT	0	0	0	0	0	4,55	11,24	0	84,21	0	0
J4.8	0	0	0	0	0	0	0	0	100	0	0
REPTree	0	0	0	0	3,22	12,90	32,26	0	38,71	3,22	9,68
LMT	0	0,5	0,7	0	1,16	11,82	23,82	0	47,06	11,90	3,04

Tabela 5.14: Distribuição de probabilidades para os algoritmos, com IG real igual a 40.

Algoritmo	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
SDT	0	0	0	0	0	22,27	34,87	0	42,86	0	0
J4.8	0	0	0	0	0	0	0	0	0	87,5	12,5
REPTree	0	0	0	0	0,8	6,4	16	0,8	51,2	19,2	5,6
LMT	0	1,52	3,62	0	4,02	18,87	23,09	0	36,73	10,24	1,57

Tabela 5.15: Distribuição de probabilidades para os algoritmos, com IG real igual a 38.

Algoritmo	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
SDT	0	0	0	0	0	16,98	41,08	0	41,94	0	0
J4.8	0	0	0	0	0	0	0	0	1.0	0	0
REPTree	0	0	0	0	0	50	25	0	0	25	0
LMT	0	2,27	2,93	0	3,55	7,45	18,26	0	44,92	9,87	2,28

Tabela 5.16: Distribuição de probabilidades para os algoritmos, com IG real igual a 37.

5.10.2 Análise dos resultados

Os algoritmos descritos possuem uma porcentagem de erro bem próximas, sendo que todos são gerados de diferentes maneiras, tentando englobar ainda mais o espaço de hipóteses e mostrar qual traz os melhores resultados para o conjunto de dados.

Os resultados utilizando o filtro de limpeza de *outliers* conseguiu aumentar em média 2% da porcentagem de acerto dos algoritmos classificação e diminuindo em 0.2 o erro médio absoluto para os de regressão. Deve-se dizer também que a velocidade da geração dos modelos diminui com a utilização desse filtro.

A utilização do *Bagging* como meta-classificador também trouxe melhorias para a classificação em ambas as abordagens, para a maioria dos algoritmos, porém acarreta um grande acréscimo no tempo de criação do modelo, que chega a ser aproximadamente 10 vezes maior para todos os algoritmos. O *Bagging* diminuiu a taxa de acerto apenas nos algoritmos LMT e J4.8.

Para os algoritmos de regressão com filtro de limpeza, por exemplo, foram alcançados resultados com **erro médio absoluto aproximado de 1.1 semanas de erro**. Sendo o algoritmo *Bagged M5'P* se mostrando melhor dentro da tarefa de estimar a IG para um recém-nascido.

O desempenho dos algoritmos de classificação devem ser analisados de forma diferenciada, de acordo com o tipo de discretização. Como a discretização utilizada foi bastante granular, observa-se que um resultado médio de 40% de acerto entre 20 classes pode ser considerado ótimo, assim como quando ocorre o erro de classificação, a diferença entre a classe predita e a real é muito pequena, sendo geralmente de 1 classe apenas.

O algoritmo LMT alcançou bons resultados dentro da classificação, mas o seu tempo de execução chega a ser extremamente maior que todos os outros, mas como ele se trata de um algoritmo que extrai modelos de regressão logística, espera-se que ele seja bem menor que as outras árvores geradas e se o modelo gerado for a melhor solução para o conjunto de dados, então o tamanho da árvore gerada é de apenas 1 nó.

Os melhores resultados dentro da classificação ficaram entre os algoritmos: *Bagged* SDT e o REPTree. Ambos utilizando o filtro de limpeza, mostraram grande poder de classificação das instâncias dentre o conjunto das 20 classes descritas, **chegando a alcançar** $44,69\%$ **e** $46,21\%$ **de acerto**, respectivamente.

Os resultados do algoritmo SDT, comparado com os outros algoritmos considerados muito bons dentro da mineração de dados, se mostrou muito bem, sendo que o

objetivo principal deste algoritmo é a distribuição de probabilidades gerada para cada nova instância. Os resultados da distribuição de probabilidades para os algoritmos são mostrados nas tabelas 5.14, 5.15 e 5.16, apresentando uma distribuição bastante concisa comparando-se com os outros resultados. Na tabela 5.14, por exemplo, o SDT mostra a maior probabilidade para a classe correta da instância. A tabela 5.15 mostra um exemplo em que nenhum dos algoritmos consegue classificar corretamente a instância, mas o SDT mostra uma distribuição mais próxima da classificação. Já na tabela 5.16 o SDT não responde com a maior probabilidade para a classe correta, sendo que desta forma o exemplo estudado deve possuir características similares às instâncias com IG igual a 38 ou 41.

O tempo de execução do algoritmo e o tamanho da árvore gerada ainda são os maiores problemas do algoritmo. Ao se confrontar o tempo de execução e o tamanho da árvore com os outros algoritmos utilizados, podemos perceber quão grande é essa diferença. Dentro da seção de trabalhos futuros (seção 6.2) são discutidas melhorias que podem ser feitas na implementação deste algoritmo.

Para poder ter uma idéia dos resultados alcançados aqui, serão citados resultados da estimativa da IG utilizando outros métodos. Os métodos de Dubowitz [22] (que utiliza 11 características externas e 10 neurológicas) e Parkin [58] (que utiliza 4 características externas) foram avaliados em [77] em dois conjuntos de recém-nascidos. Os testes também incluem estudos sobre os pesos dos bebês, através de informações da idade gestacional e peso, classificando os bebês em: grande para sua idade gestacional, pequeno para a sua idade gestacional e adequado para sua idade gestacional.

Os resultados encontrados por Vogt [77] para o método de Dubowitz em relação aos bebês pequenos para sua idade gestacional foi superestimado. Enquanto os bebês adequados para sua idade gestacional tiveram resultados subestimados. Já os grandes para sua idade gestacional se mostraram perto da curva padrão. Em geral os resultados do método de Dubowitz teve **aproximadamente 5 semanas de erro**, enquanto o método de Parkin chegou a **próximo a 6 semanas**.

Os resultados anteriores correspondentes aos métodos de Dubowitz e Parkin, mostram resultados interessantes ao se comparar com os resultados do sistema FootScanAge. É

claro que os resultados não podem ser comparados a fio, pois o conjunto de recém-nascidos é diferente, mas essa comparação serve apenas como base para apresentação dos resultados do sistema descrito neste trabalho.

Está bem descrito também durante este trabalho que o sistema deve manter seus resultados para qualquer conjunto de recém-nascidos, desde que sejam obtidas imagens similares as consideradas neste trabalho.

CAPÍTULO 6

CONCLUSÃO E TRABALHOS FUTUROS

6.1 Conclusão

Analisando os resultados da seção 5.10 se mostram muito promissores ao se falar de métodos para estimativa da idade gestacional. Pois os resultados utilizando o filtro de limpeza e o peso para os algoritmos: *Bagged M5'P*, *Bagged* SDT e REPTree, por exemplo, o FootScanAge consegue alcançar um **erro médio absoluto aproximado de 1.1 semanas de erro,** $44,69\%$ **e** $46,21\%$ **de acerto**, respectivamente. Sendo assim resultados impressionantes, comparando-se superficialmente com os outros métodos, mesmo não citando aqui a invasividade da obtenção das características necessárias para a estimativa da idade gestacional e subjetividade de cada uma das características para os outros métodos de estimativa.

Focando na parte mais importante desse resultado, a distribuição das probabilidades entre as classes, é de suma importância frisar os resultados mostrados pelo algoritmo de árvores de decisão *fuzzy*, o SDT implementado, devido as suas características já descritas, enfatizando a lógica *fuzzy* embutida.

A utilização do peso como uma característica auxiliar, também é um bom caso de estudo, pois em testes preliminares mostrados na seção 5.10, mostraram que a taxa de erro médio absoluto ainda pode ser minimizada utilizando outras características. Mas, a utilização dessa característica pode minimizar praticidade do método, pois a característica principal do método FootScanAge é a utilização de poucas características comparado com os outros métodos, onde na realidade existe uma única característica necessária para a descoberta da idade gestacional, a imagem (e suas características). Então, todas as características podem ser traduzidas em uma única. Por isso a discussão de se utilizar o peso ou não para auxiliar e melhorar a precisão do classificador, ainda deve ser discutida com atenção, mesmo porque os resultados com e sem peso foram muito similares.

A aplicação da lógica *fuzzy* ao campo de indução de árvores de decisão, embora seja uma pesquisa nova, mostra um futuro promissor. Novos algoritmos e idéias tendem a surgir com o tempo e os resultados obtidos são muito interessantes. Os atuais algoritmos *fuzzy* possuem, em geral, mesma precisão que os algoritmos para indução de árvores tradicionais. A implementação de um algoritmo de árvores de decisão *fuzzy* comprovou esse fato para o conjunto de dados testado. Porém, mais testes seriam necessários para comprovar a real eficiência do algoritmo que ainda apresenta limitações como o fato de trabalhar somente com dados contínuos.

Também foram realizados testes utilizando diversos tipos de métodos de agrupamentos de classificadores, através de votação, média, *Bagging*, *Boosting* e *Stacking*. Mas apenas o *Bagging* mostrou resultados interessantes, sendo por isso utilizado dentro dos testes finais.

6.1.1 Projeto FootScanAge

Pode-se ver com facilidade que o método FootScanAge possui uma ótima taxa de erro médio absoluto comparando com outros resultados, mas as comparações mais interessantes entre o FootScanAge e os outros métodos é o número de características necessárias para determinar a idade gestacional e, principalmente, por ser um método não-invasivo.

Com todos os resultados dos algoritmos, dicas e pistas disponibilizadas pelo sistema FootScanAge em mãos e a sua experiência, o especialista responsável pelo voto da idade gestacional final pode, com muito mais confiança, atribuir a idade gestacional que ele acredita ser adequada ao bebê correspondente a imagem da superfície plantar analisada.

O método FootScanAge já é um método consagrado e foi considerado finalista no prêmio Fundação Banco do Brasil de Tecnologia Social - Edição 2005[1]. O FootScanAge conseguiu chegar a este ponto devido as suas características mais importantes:

- Baixo custo: pois é implementado usando *software* livre e pode ser usado em diferentes plataformas;
- *Software* livre: O sistema é uma ferramenta livre e de livre distribuição para qualquer

[1]http://www.fundacaobancodobrasil.org.br

hospital público do país;

- Eficiente prontuário eletrônico: Armazenamento de informações sobre a mãe e o recém-nascido;
- Otimização dos recursos da UTI neonatal;
- Melhoria na taxa de sobrevida dos recém-nascidos.

6.2 Trabalhos futuros

Ao se falar do algoritmo de árvores de decisão *fuzzy* descrito neste trabalho, os seus resultados se mostraram bem interessantes, mas em cima da implementação descrita ainda poderiam ser feitas algumas melhorias no algoritmo como por exemplo:

- Diminuir o tamanho da árvore gerada;
- Melhorar o algoritmo de poda implementado;
- Otimizar a criação do modelo.

Olaru [54] mostra algumas melhorias para algoritmos de árvores de decisão *fuzzy*, tais como: *Backfitting*, *reffiting* e poda.

É certo também que o algoritmo deveria passar por mais testes utilizando diferentes bases de dados, para que sua eficiência possa ser analisada em diferentes espaços de hipóteses.

Já dentro do sistema FootScanAge como um todo, existe ainda alguns desafios e dificuldades a serem enfrentados para tornar o método mais robusto e preciso:

- Necessidade da coleta sistemática de imagens de recém-nascidos, principalmente prematuros, para ampliar a base a ser analisada;
- Melhoria do processo de obtenção de impressões digitais dos neonatos (embora tenha havido uma grande evolução da qualidade dos métodos graças aos estudos iniciais do projeto FootScanAge);

- Tornar o método adaptável a diferentes populações de recém-nascidos;
- Dificuldade de obtenção da idade gestacional padrão-ouro.

Um passo seguinte ao projeto FootScanAge já foi dado, através de um novo projeto chamado *FootScanID*, que se trata da identificação do recém-nascido através da imagem da sua superfície plantar.

BIBLIOGRAFIA

[1] R. Agrawal, T Imielinski, e A. N. Swami. Mining association rules between sets of items in large databases. Peter Buneman e Sushil Jajodia, editors, *Proceedings ACM SIGMOD International Conference on Management of Data*, páginas 207–216, 1993.

[2] R. Agrawal e R. Srikant. Fast algorithms for mining association rules. Jorge B. Bocca, Matthias Jarke, e Carlo Zaniolo, editors, *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, páginas 487–499. Morgan Kaufmann, 1994.

[3] M. Amato, P. Hüppi, e R. Claus. Rapid biometric assessment of gestational age in very low birth weight infants. *Journal of perinatal medicine*, 19(5):367–371, 1991.

[4] M. Antonie, O. Zaane, e A. Coman. Application of data mining techniques for medical image classification. *Proceedings of 2nd Int. Workshop Multimedia Data Mining*, páginas 94–101, 2001.

[5] J. F. Baldwin e Dong (Walter) Xie. Simple fuzzy logic rules based on fuzzy decision tree for classification and prediction problem. Zhongzhi Shi e Qing He, editors, *Intelligent Information Processing (IIP) II*, páginas 175–184. Springer, 2004.

[6] J. L. Ballard, J. C. Khoury, K. Wedig, L. Wang, B. L. Eilers-Walsman, e R. Lipp. New ballard score, expanded to include extremely premature infants. *The Journal of Pediatrics*, 119(3):417–423, 1991.

[7] J. L. Ballard, K. K. Novak, e M. Driver. A simplified score for assessment of fetal maturation of newly born infants. *The Journal of Pediatrics*, 95(1):769–774, 1979.

[8] R. Belew e L. Booker. Genetic algorithms. *Proceedings of the Fourth International Conference*, páginas 175–184. Morgan Kaufmann, 1991.

[9] O. R. P. Bellon, M. Cat, L. Silva, e K. L. Boyer. Using computer vision to help the determination of the gestational age of newborns. *Academic Radiology*, 12(5):544–553, 2005.

[10] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[11] L. Breiman, J. Friedman, R. Olshen, e C. Stone. *Classification and Regression Trees*. Chapman & Hall, 1984.

[12] H. Capurro, S. Konichezky, D. Fonseca, e R. Caldeyro-Barcia. A simplified method for diagnosis of gestational age in the newborn infant. *The Journal of Pediatrics*, 93(1):120–122, 1978.

[13] V. Castelli e L. D. Bergman. *Image databases*. A Wiley-interscience publication. John Wiley & Sons, 2002.

[14] M. Cat. *Método FootScanAge para Determinação da Idade Gestacional, Tese de Doutorado em Medicina (Pediatria)*. Tese de Doutorado, Universidade Federal do Paraná, UFPR, 2003.

[15] T. M. Celinski. Métodos de agrupamento: Uma abordagem comparativa com aplicação em segmentação de imagens de profundidade. Dissertação de Mestrado, Universidade Federal do Paraná - UFPR, 1998.

[16] R. L. P. Chang e T. Pavlidis. Fuzzy decision tree algorithms. *IEEE Transactions on Man, Systems, Cybernetics*, SMC-7(1):28–35, 1977.

[17] I. Chiang e J. Y. Hsu. Fuzzy classification trees for data analysis. *Fuzzy Sets and Systems*, 130(1):87–99, 2002.

[18] P. Clark e T. Niblett. Induction in noisy domains. *Progress in Machine Learning-Proceedings of EWSL 87: 2nd European Working Session on Learning*, páginas 11–30, Bled, Yogoslavia, 1987.

[19] A Del Bimbo. *Visual information retrieval*. Morgan Kaufmann, 1999.

[20] T. G. Dietterich. Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1875:1–15, 2000.

[21] R. C. Dubes e A. K. Jain. *Algorithms for clustering data*. Prentice Hall, 1988.

[22] L. M. Dubowitz, V. Dubowitz, e C. Goldberg. Clinical assessment of gestational age in the newborn infant. *The Journal of Pediatrics*, 77(1):1–10, 1970.

[23] R. O. Duda, P. E. Hart, e D. G. Stork. *Pattern Classification*. A Wiley-interscience publication. John Wiley & Sons, 2001.

[24] L. V. Fausett. *Fundamentals of neural networks: architetures, algorithms, and applications.* Prentice-Hall, 1994.

[25] U. Fayyad e R. Uthurusamy. Data mining and knowledge discovery in databases. *Communications of the ACM*, 39(11):24–26, 1996.

[26] U. M. Fayyad e K. B. Irani. On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, 8(1):87–102, 1992.

[27] U. M. Fayyad, G. Piatetsky-Shapiro, e P. Smyth. Knowledge discovery and data mining towards a unifying framework. *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, páginas 82–88, 1996.

[28] A. A. Freitas e S. H. Lavinston. *Mining very large databases with parallel processing.* Kluwer Academic, 1998.

[29] Y. Freund e R. E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *European Conference on Computational Learning Theory*, páginas 23–37, 1995.

[30] Y. Freund e R. E Schapire. Experiments with a new boosting algorithm. *Proceedings of the International Conference on Machine Learning*, páginas 148–156, 1996.

[31] R. C. Gonzalez e R. E. Woods. *Processamento de imagens digitais*. Edgard Blücher, 1992.

[32] V. N. Gudivada e V. V. Raghavan. Content-based image retrieval systems. *IEEE Computer*, 28(9):18–22, 1995.

[33] J. Han e M. Kamber. *Data Mining: Concepts and Techniques*. Academic Press, 2001.

[34] D. Heckerman. Bayesian networks for knowledge discovery. *Advances in Knowledge Discovery and Data Mining*, 11:273–305, 1996.

[35] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[36] G. Holmes, M. Hall, e E. Frank. Generating rule sets from model trees. *Proceedings of the 12th Australian Joint Conference on Artificial intelligence: Advanced Topics in Artificial intelligence*, volume 1747, páginas 1–12. Ed. Lecture Notes In Computer Science, 1999.

[37] W. H. Inmon. *Building the data warehouse*. QED Information Sciences, Inc., Wellesley, MA, USA, 1992.

[38] R. Jain, R. Kasturi, e B. G. Schunck. *Machine Vision*. McGraw-Hill, 1995.

[39] C. Z. Janikow. Fuzzy decision trees: issues and methods. *IEEE Transactions on Man, Systems, Cybernetics*, 28(1):1–14, 1998.

[40] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 2:241–254, 1967.

[41] S. Khoshafian e A.B. Barei. *Multimedia and image databases*. Morgan Kaufmann, 1996.

[42] R. Kohavi. The power of decision tables. Nada Lavrac e Stefan Wrobel, editors, *Proceedings of the European Conference on Machine Learning*, Lecture Notes in Artificial Intelligence 914, páginas 174–189, Berlin, Heidelberg, New York, 1995. Springer Verlag.

[43] H. Korth e A. Silberschatz. *Sistemas de bancos de dados*. Makron Books, 1995.

[44] R. Kothari e M. Dong. Decision trees for classification: A review and some new results. *In: Pal, S.R., Pal, N.R. (eds.), Lecture Notes in Pattern Recognition*, páginas 241–252. World Scientific Publishing, 2001.

[45] N. Landwehr, Hall M., e E. Frank. Logistic model trees. *Proceedings of the 14th European Conference on Machine Learning*, páginas 241–252, 2003.

[46] H. Liu, A. Mandvikar, e Mody J. An empirical study of building compact ensembles. *Lecture Notes in Computer Science*, 3129:622–627, 2004.

[47] R. L. Mantaras e E. Armengol. Machine learning from examples: Inductive and lazy methods. *Data & Knowledge Engineering*, 25:99–123, 1998.

[48] O. Marques F. e H. Vieira N. *Processamento Digital de Imagens*. BRASPORT - Livros e Multimídia, Cambridge, MA, USA, 1999.

[49] C. Marsala e B. Bouchon-Meunier. Choice of a method for the construction of fuzzy decision trees. *The 12th IEEE International Conference on Fuzzy Systems*, volume 1, páginas 584 – 589, 2003.

[50] M. Mitchell. *An introduction to genetic algorithms*. MIT Press, Cambridge, MA, USA, 1996.

[51] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[52] T. M. Mitchell. Machine learning and data mining. *Communications of the ACM*, 42(1):622–627, 1999.

[53] D. Nauck e R. Kruse. Nefclass: A neuro-fuzzy approach for the classification of data. *Proceedings of the ACM Symposium on Applied Computing*, páginas 461–465, 1995.

[54] C. Olaru e L. Wehenkel. A complete fuzzy decision tree technique. *Fuzzy Sets and Systems*, 138(2):221–254, 2003.

[55] A. Olukunle e S. Ehikioya. A fast algorithm for mining association rules in medical image data. *Proceedings of Canadian Conference on Electrical and Computer Engineering*, volume 2, páginas 12–15, 2002.

[56] C. Ordonez e E. Omiecinski. Discovering association rules based on image content. *Proceedings of the IEEE Advances in Digital Libraries Conference (ADL'99)*, Baltimore, Maryland, 1999.

[57] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[58] J. M. Parkin, E. N. Hey, e J. S. Clowes. Rapid assessment of gestational age at birth. *Archives of Disease Childhood*, 51(4):259–263, 1976.

[59] W. Pedrycz e Z.A. Sosnowski. C-fuzzy decision trees. *IEEE Transactions on Systems, Man and Cybernetics, Part C*, 35(4):498–511, 2005.

[60] Y. Peng e P. Flach Flach. Soft discretization to enhance the continuous decision tree induction. Nada Lavrac I, Christophe Giraud-Carrier e Steve Moyle, editors, *In: Integrating Aspects of Data Mining, Decision Support and Meta-Learning ECML/PKDD'01 workshop notes*, páginas 109–118, 2001.

[61] V. Poe, S. Brobst, e P. Klauer. *Building a Data Warehouse for Decision Support*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1997.

[62] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[63] J. R. Quinlan. Learning with continuous classes. *5th Australian Joint Conference on Artificial Intelligence*, páginas 343–348. Adams And Sterlings, 1992.

[64] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

[65] J. R. Quinlan. Bagging, boosting, and c4.5. *Proceedings of AAAI'96 National Conference on Artificial Intelligence*, páginas 725–730. Adams And Sterlings, 1996.

[66] P. Sahoo, C. Wilkins, e J. Yeager. Threshold selection using renyi's entropy. *Pattern recognition*, 30(1):71–84, 1997.

[67] C. Schaffer. Cross-validation. *Fourth International Workshop on Artificial Intelligence and Statistics*, páginas 725–730. Adams And Sterlings, 1993.

[68] R. Schapire. The boosting approach to machine learning: An overview. *MSRI Workshop on Nonlinear Estimation and Classification*, 2001.

[69] M. Severich. Desenvolvimento de uma ferramenta de processamento de imagens para o sistema footscan. Dissertação de Mestrado, Universidade Federal do Paraná, UFPR, 2002.

[70] L. G. Shapiro e G. C. Stockman. *Computer Vision*. Prentice Hall, 2001.

[71] W. M. Shen. Bayesian probability theory - a general method for machine learning. Relatório técnico, MCCCarnot - Microelectronics and Computer Technology Corporation, Austin, TX, 1993.

[72] L. Silva, O. R. P. Bellon, R. P. Lemes, e M. N. L. Meira, J. A.Cat. An image processing tool to support gestational age determination. *Proceedings of the 19th IEEE International Symposium on Computer-Based Medical Systems*, 2006.

[73] A. W. M. Smeulders, M. Worring, e S. Santini. Content-based image retrieval at the end of the early years. *IEEE Transactions on PAMI*, 22(12):1349–1380, 2000.

[74] M. Sonka e J. M. Fitzpatrick. *Handbook of Medical Imaging: Medical Image Processing and Analysis*, volume 2. SPIE Press, 2000.

[75] E. V. Vieira. Mineração de imagens: Conceitos e aplicação em sistemas de recuperação de imagens por conteúdo. Dissertação de Mestrado, Universidade Federal do Paraná - UFPR, 2002.

[76] E. V. Vieira, O. R. P. Bellon, e L. Silva. Mineração de imagens. *Revista de Informática Teórica Aplicada - RITA*, 9(2):67–96, 2002.

[77] H. Vogt, B. Haneberg, P. H. Finne, e A. Stensberg. Clinical assessment of gestational age in the newborn infant. an evaluation of two methods. *Acta paediatrica Scandinavica*, 70(5):669–672, 1981.

[78] I. H. Witten e E. B. Frank. *Data mining: practical machine learning tools and techniques with Java implementations*. Morgan Kaufmann, 2000.

[79] D. H. Wolpert. Stacked generalization. Relatório Técnico LA-UR-90-3460, Los Alamos, NM, 1990.

[80] L.A. Zadeh. Fuzzy sets. *Information And Control*, 8:338–353, 1965.

[81] L.A. Zadeh. Fuzzy logic and approximate reasoning. *Synthese*, 30:407–428, 1975.

[82] O. R. Zaiane, Han J., Li Z., S. H. Chee, e J. Y. Chiang. Multimediaminer: A system prototype for multimedia data mining. *Proceedings ACM SIGMOD International Conference on Management of Data*, volume 27, páginas 581–583. ACM Press, 1998.

[83] J. Zhang, W. Hsu, e M. L. Lee. Image mining: issues, frameworks and techniques. *Proceedings of the 2nd International Workshop on Multimedia Data Mining*, páginas 13–20, 2001.

[84] N. Ziviani. *Projeto de Algoritmos*. Editora Pioneira, 1994.