UNIVERSIDADE FEEVALE

JOSUÉ VIEZZER

AUXÍLIO A TOMADA DE DECISÃO PARA PROGRAMAÇÃO COMERCIAL DE EMISSORAS DE RÁDIO UTILIZANDO APRENDIZADO DE MÁQUINA

Novo Hamburgo
2011

JOSUÉ VIEZZER

AUXÍLIO A TOMADA DE DECISÃO PARA PROGRAMAÇÃO COMERCIAL DE EMISSORAS DE RÁDIO UTILIZANDO APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso
apresentado como requisito parcial
à obtenção do grau de Bacharel em
Ciência da Computação pela
Universidade Feevale

Orientador: Rodrigo Rafael Villarreal Goulart

Novo Hamburgo
2011

RESUMO

Com o avanço dos computadores e dos softwares, nas últimas duas décadas, o mercado de trabalho exigiu um aumento da produtividade e da qualidade em todos os campos de atuação, e as emissoras de rádio não estão fora desta realidade. Hoje, com a automação, os roteiros comerciais de emissoras de rádio, podem ser criados de maneira rápida e ágil. E para tornar mais eficiente à análise por informações que contribuam para gerar tomada de decisões na criação de roteiros comerciais para emissoras de rádio, este trabalho tem como objetivo realizar experimentos de aprendizado de máquina, utilizando a ferramenta Weka a partir de uma base de dados em ARFF que foi desenvolvida a partir da operação do software SisRadio módulo OPEC.

Palavras-chave: Inteligência Artificial. Aprendizado de Máquina. Weka. Automação de Rádio. Roteiro Comercial de Emissoras de Rádio.

ABSTRACT

With the advancement of computers and software in the past two decades, the labor market required an increase in productivity and quality in all fields, and radio stations are not out of this reality. Today, with automation, the routes of commercial radio stations can be created quickly and agile. And to make it more efficient for the analysis to generate information to assist decision-making in creating scripts for radio commercials, this work aims to conduct experiments in machine learning using the tool Weka from a database in ARFF which was developed from the operation of the software SisRadio module OPEC.

Keywords: Artificial Intelligence. Machine Learning. Weka. Radio Automation. Roadmap Business of Radio.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizado de Máquina
ARFF	Attribute-Relation File Format
CSV	Comma-Separated Values
FITERT	Federação do Trabalho em Rádio e Televisão
FM	Freqüência Modulada
KDD	Knowledge Discovery in Databases
MD	Mineração de Dados
MD’s	Mini-Disc
WEB	World Wide Web
WEKA	Waikato Environmet for Knowledge Analysis
XML	eXtensible Markup Language

SUMÁRIO

INTRODUÇÃO

O rádio é um meio de comunicação baseado na propagação de informação sonora por meio de ondas eletromagnéticas, sua criação se deu devido à invenção da válvula triodo, em 1906, nos Estados Unidos, por Lee De Forest. A válvula triodo permite a ampliação dos sinais elétricos, possibilitando a audição de sons complexos transmitidos por ondas eletromagnéticas. Porém, só na década de 1920 foi que o rádio despontou. No Brasil, mais precisamente, em 7 de setembro de 1922 foi realizada a primeira transmissão radiofônica oficial, durante a comemoração ao Centenário da Independência do Brasil, no Rio de Janeiro. Já na década de 1930, o governo brasileiro deu licença para a veiculação de mensagens comerciais, assinando o Decreto-lei nº 21.111.

A introdução de mensagens comerciais fez a radiodifusão brasileira se popularizar, permitindo assim o surgimento de programas de "variedades", o que transformou o rádio em fenômeno social, tento seu apogeu como veículo de comunicação de massa dos anos 1920 a 1960. Com a chegada da TV, o rádio perdeu prestígio, e somente vinte anos depois, na década de 1980, com o crescimento das emissoras de rádio FM e uma nova linguagem de comunicação, passou a ter novamente números expressivos no mercado publicitário. Já em 1990, com a expansão das transmissões via-satélite e a formação de redes, fez com que as emissoras de rádio obtivessem uma boa fatia das mídias publicitárias de âmbito nacional.

Hoje, emissoras de rádio já possuem estruturas de trabalho totalmente informatizadas e com mídias em formato digital, embora, muitas utilizem ainda de CD Player's, MD's e toca-discos. As novas tecnologias, aliadas a evolução da computação trouxeram mais velocidade e mobilidade para o rádio, processos de edição de áudio foram simplificados, sistemas de automação e administração de redes via-satélite proporcionaram mais qualidade e confiabilidade ao anunciante.

Porém, neste ambiente, surgem dificuldades a serem analisadas, como o preenchimento do roteiro comercial numa emissora de rádio afiliada, que na maioria dos casos o estoque de mídia é menor que o pré-estabelecido pela emissora geradora. Outro aspecto é a alocação das mídias dentro do intervalo comercial, pois a valorização de cada mídia é representada por sua faixa horária e seu posicionamento dentro de cada intervalo comercial. Em vista disso, se busca desenvolver parâmetros de correção para a programação comercial de emissoras de rádio afiliadas. E devido à inexistência de trabalhos relacionados, a fundamentação deste trabalho parte das experiências do especialista, o roteirista de intervalos

comerciais encarregado pela criação e geração do roteiro comercial numa emissora de rádio afiliada.

Portanto, pretende-se com este trabalho de conclusão contribuir para a busca de conhecimento no campo da programação comercial de emissoras de rádio afiliadas a uma rede via-satélite, e para isso apresentada experimentos de aprendizado de máquina, utilizando o software *Weka* (Waikato Environmet for Knowledge Analysis) com a finalidade de identificar problemas na programação comercial automatizada e na extração de conhecimento a partir de programações pré-estabelecidas.

No primeiro capítulo, são descritos os procedimentos para a criação e geração de um roteiro comercial através do uso do software SisRadio. A partir dos dados gerados pela aplicação, é descrita a etapa prática de preparação dos dados que é apresentada no terceiro capítulo.

Já no segundo capítulo, conceitua-se aprendizado supervisionado, que tem sido uma importante aplicação para a aquisição de conhecimento no mundo real. Bem como, são aduzidos os métodos e exemplos que auxiliam nesse processo.

Como citado anteriormente, o terceiro capítulo expõe as metodologias utilizadas para a produção, e também, o pré-processamento da base de dados deste trabalho.

No quarto capítulo, são apresentadas as características do *Weka* e seus parâmetros para geração de um classificador no qual serão extraídas as regras para obtenção de conhecimento, juntamente com os experimentos realizados na ferramenta de mineração.

E por fim, são expostos os resultados e conclusões obtidos no trabalho a partir dos experimentos realizados.

1 ROTEIRO COMERCIAL DE UMA EMISSORA DE RÁDIO AFILIADA

1.1 Roteirista de intervalos comerciais

De acordo com o Manual dos Radialistas, desenvolvido pela FITERT, Federação do Trabalho em Rádio e Televisão, o roteirista de intervalos comerciais, é responsável por elaborar a programação dos intervalos comerciais das emissoras de rádio, distribuindo as mensagens comerciais ou publicitárias de acordo com a direção comercial da emissora. Para se compreender a atividade de um roteirista numa emissora de rádio, é necessário antes conhecer as etapas para a veiculação de uma mídia, mensagem comercial, dentro de uma emissora de rádio afiliada, Figura 1.1.

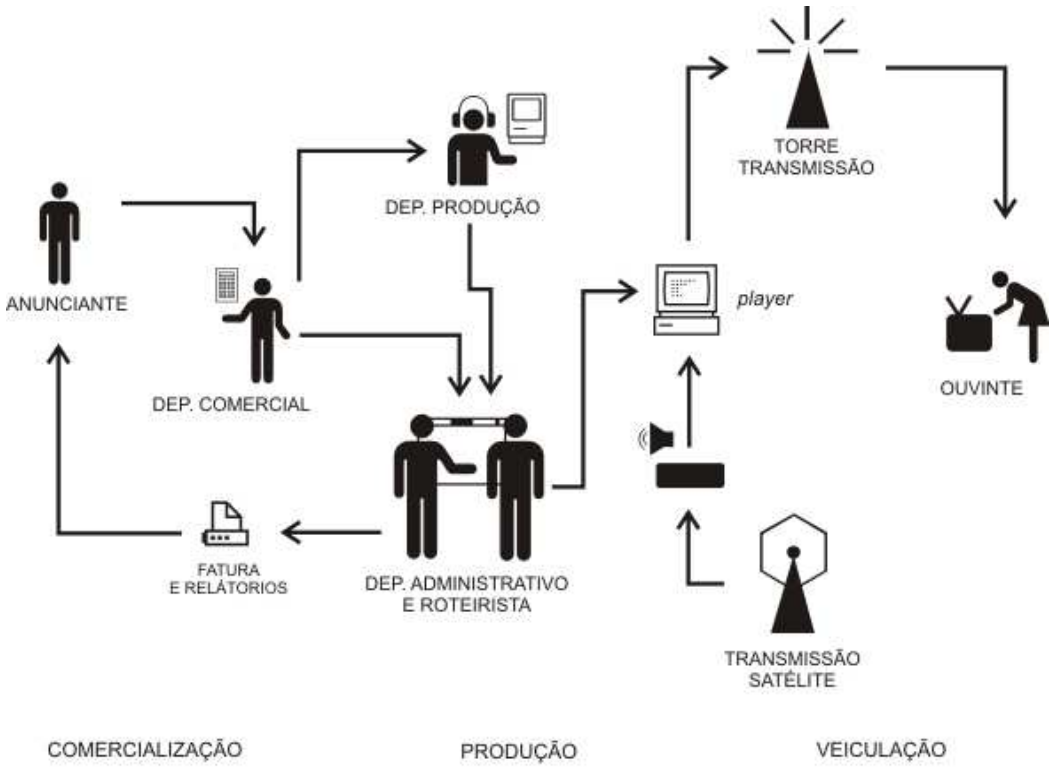

Figura 1.1: Etapas para veiculação de uma mídia numa emissora de rádio afiliada
Fonte: (AUTOR, 2011)

Para se veicular uma mensagem comercial numa emissora de rádio afiliada, é necessário seguir algumas etapas, que são descritas aqui como comercialização, produção e veiculação. Na comercialização, presumindo que o contrato com o anunciante seja acordado, o departamento comercial intermedeia o processo de produção da mídia de áudio passando um *briefing* para o departamento de produção, ao mesmo tempo em que encaminha ao departamento administrativo/roteirista as informações de contrato, como dados gerais do

anunciante, informações de faturamento e aproveitamento. Na produção, a mídia de áudio é gravada e repassada ao roteirista. O roteirista de intervalos comerciais efetua o cadastro da mídia no software de roteiro, obedecendo as informações contidas no contrato, como números de inserções, período do contrato e horários de veiculação. Após o processo de cadastro o roteirista gera o roteiro comercial diário, que é enviando ao *player*, computador que executa as mídias de áudio. Como numa emissora de rádio afiliada a programação musical é gerada via-satélite, o *player* somente executará a programação dos intervalos comerciais no momento exato em que receber um pulso eletrônico, transmitido juntamente com a programação musical. Este pulso, além de funcionar como um *play*, também bloqueia o áudio da transmissão. Tornando possível assim a auscultação do intervalo comercial pelo ouvinte da emissora de rádio afiliada.

Hoje a atividade do roteirista, dentro de uma emissora de rádio, está diretamente ligada a operação do software de roteiro comercial. Para auxiliar na descrição da atividade, na seção 1.2 é apresentado o funcionamento do módulo Comercial do Sistema SisRadio (SIS, 2011) desenvolvido pela empresa Performática Computação, que atua desde 1986 no mercado de desenvolvimento de softwares, localizada na cidade de Porto Alegre, Rio Grande do Sul.

1.2 Módulo Comercial (OPEC) do sistema SisRadio

A automação de emissoras de rádio, SisRadio, da empresa Performática Computação, é desenvolvida em Delphi e utiliza o bando de dados MS-Access, e é dividido em cinco módulos: Comercial, Faturamento, Musical, Pesquisa de Ouvintes e Utilitários, cada módulo pode ser operado em conjunto ou separadamente. Na Figura 1.2, é apresentada a tela de entrada dos módulos Comercial, Musical e Utilitários.

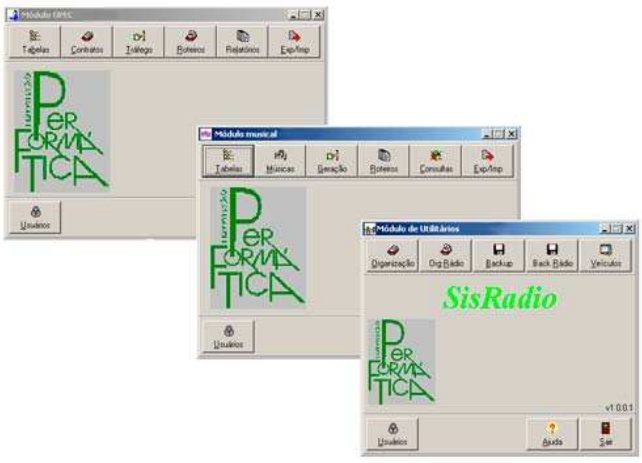

Figura 1.2: Tela de entrada dos módulos Comercial, Musical e Utilitários do SisRadio
Fonte: (AUTOR, 2011)

De acordo com o manual do módulo OPEC, que está disponível junto aos arquivos de instalação, o procedimento básico para operar o software consiste em cadastrar os contratos, já fornecendo informações sobre dados gerais do anunciante, aproveitamento e faturamento. Logo após, cadastrar as mídias, vinculando-as aos contratos, e depois elaborar os roteiros a partir das informações de reserva de execuções provenientes dos contratos e por fim os roteiros são exportados ao *player*, isto é, ao computador do estúdio.

Como o roteiro comercial, gerado a partir das operações do módulo OPEC, é empregado como o princípio do desenvolvimento da base de dados, proposta neste trabalho, a seguir são apresentadas de forma detalhada apenas as etapas essenciais para a elaboração do roteiro.

Na tela de entrada do módulo OPEC, Figura 1.3, no menu superior estão presentes os seguintes botões: *Tabelas*, *Contratos*, *Tráfego*, *Roteiros*, *Relatórios* e *Exp/Imp*. Já no menu inferior, encontram-se os botões: *Usuários*, *Ajuda* e *Sair*.

Figura 1.3: Tela de entrada do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

O botão *Tabelas* do menu superior, apresenta o sub-menu *Selecione a tabela desejada*, entre as tabelas listadas neste sub-menu, as utilizadas para a criação do roteiro são: *Tabela de vendas* e *Tabela de estruturas dos roteiros*. Na *Tabela de vendas*, Figura 1.4, são cadastrados os programas, seus valores de comercialização e o aproveitamento comercial, com isso, se tem a reserva de espaço no roteiro comercial. Selecionado a opção *Tabela de estruturas dos roteiros*, Figura 1.5, é possível cadastrar modelos de estruturas de roteiros. Nestes modelos é que são criados os blocos comerciais, obedecendo horários de veiculação e

tempos de duração. Com a criação destes modelos é possível se dimensionar o estoque de tempo de uma emissora de rádio, na Figura 1.6 é apresentado um relatório com a análise de encaixe, em inserções de 30 segundos. Isto é, quantos materiais (mídias) de 30 segundos podem ser inseridas no roteiro comercial diário, como pode se visualizar a capacidade em mídias de 30 segundos chega ao total de 394 inserções. No caso de emissoras de rádio afiliadas, estes modelos seguem as determinações da emissora geradora.

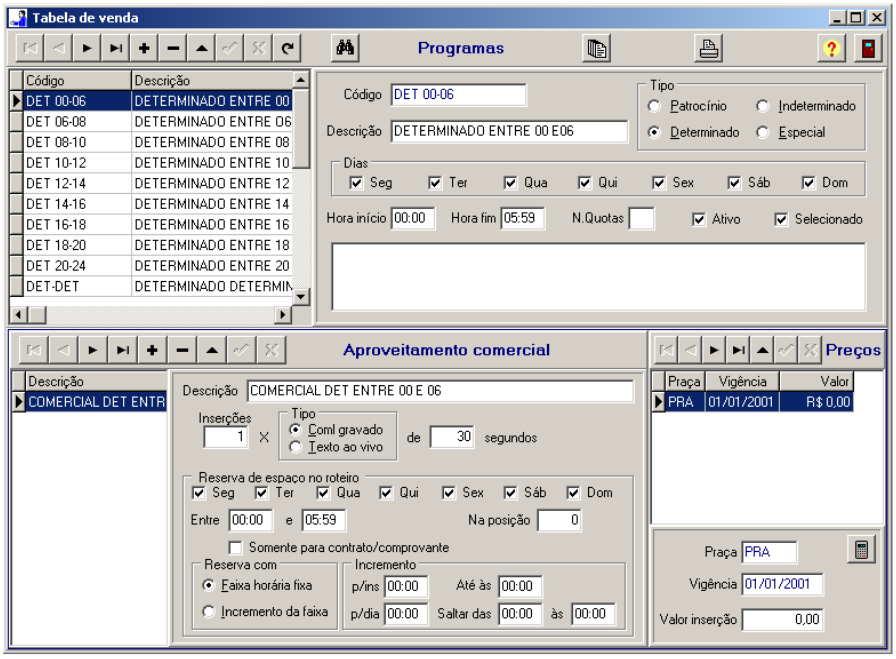

Figura 1.4: Tabela de venda do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

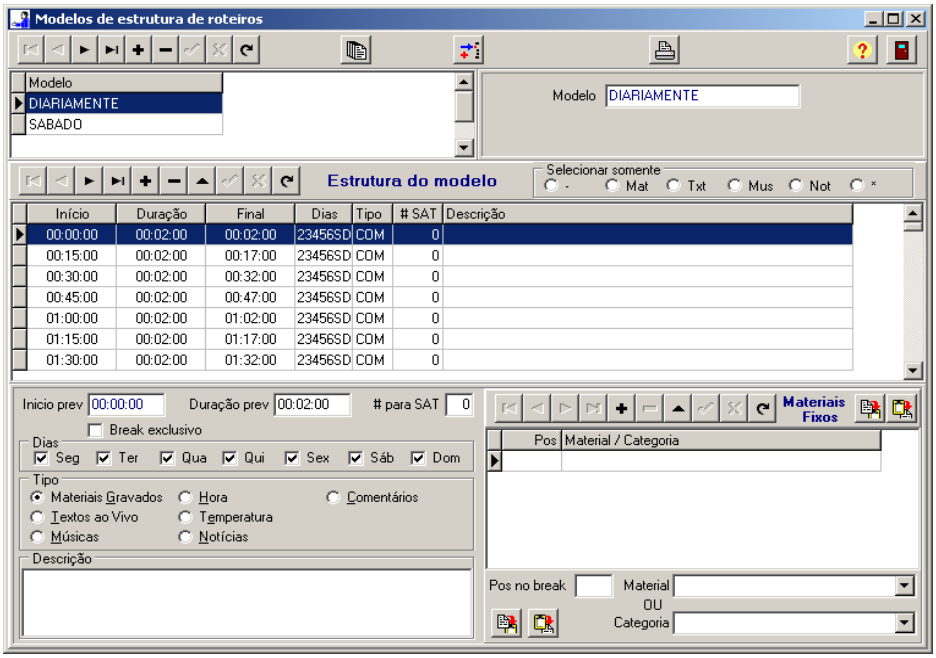

Figura 1.5: Modelos de estrutura de roteiros do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

Análise de encaixe

01/03/2011 a 01/03/2011 - 00:00:00 a 23:59:59

				Materiais avulsos			Materiais contratados								
							Faturados			Não faturados					
Data	Base	UN	Capacidade	Fixos	Avulsos	Total	Reserva	Alocados	Total	Reserva	Alocados	Total	Total	Ocupado	Livre
01/03/2011 - TER	Roteiro	Q30	394	62	117	179	21	216	237	0	0	0	237	416	-22

Datas listadas: 1

UN: Seg = tempos em segundos / Q30 = quantidade de comerciais de 30"

Figura 1.6: Análise de encaixe módulo Comercial do sistema SisRadio
Fonte: (AUTOR, 2011)

Depois que a estrutura do roteiro comercial estiver concluída, o próximo passo é o cadastramento dos anunciantes, para isso, se seleciona a opção *Contrato* no menu superior, da tela de entrada. Na tela apresentada, a aba *Geral* fica selecionada, Figura 1.7, é nela que são informados os dados gerais de cada anunciante, como nome do anunciante, ramo do contrato, campanha, início e término do contrato, e algumas informações de faturamento, que são inseridas na caixa *Faturar* localizada na base inferior da janela.

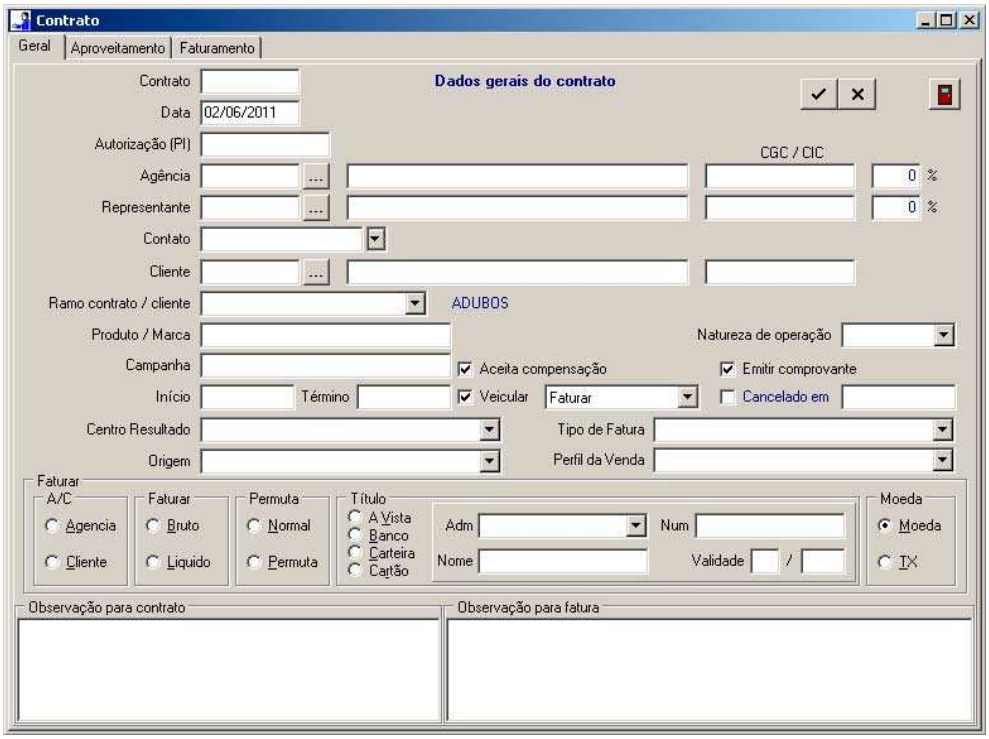

Figura 1.7: Contrato aba Geral do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

Na Figura 1.8 pode se observar que a aba *Aproveitamento* é dividida em três caixas: *Programas do contrato*, *Aproveitamento dos programas* e *Quantidade do aproveitamento por*

dia. Na primeira caixa é onde são associados os programas cadastrados na *Tabela de vendas*, ao inserir um programa, os valores de aproveitamento são apresentados na segunda caixa. É nesta caixa que são vinculadas as mídias de áudio, que podem ser cadastradas nesta caixa através do botão +, como também na opção da tela de entrada *Tráfego*, sub-menu *Cadastro de materiais para veiculação*, Figura 1.9. E na última caixa desta aba, *Quantidade do aproveitamento por dia*, é exibido o mapa de reservas.

E na última aba, *Faturamento* são inseridos os vencimentos e valores do contrato, que podem ser exportados para o módulo Faturamento do software SisRadio, Figura 1.10.

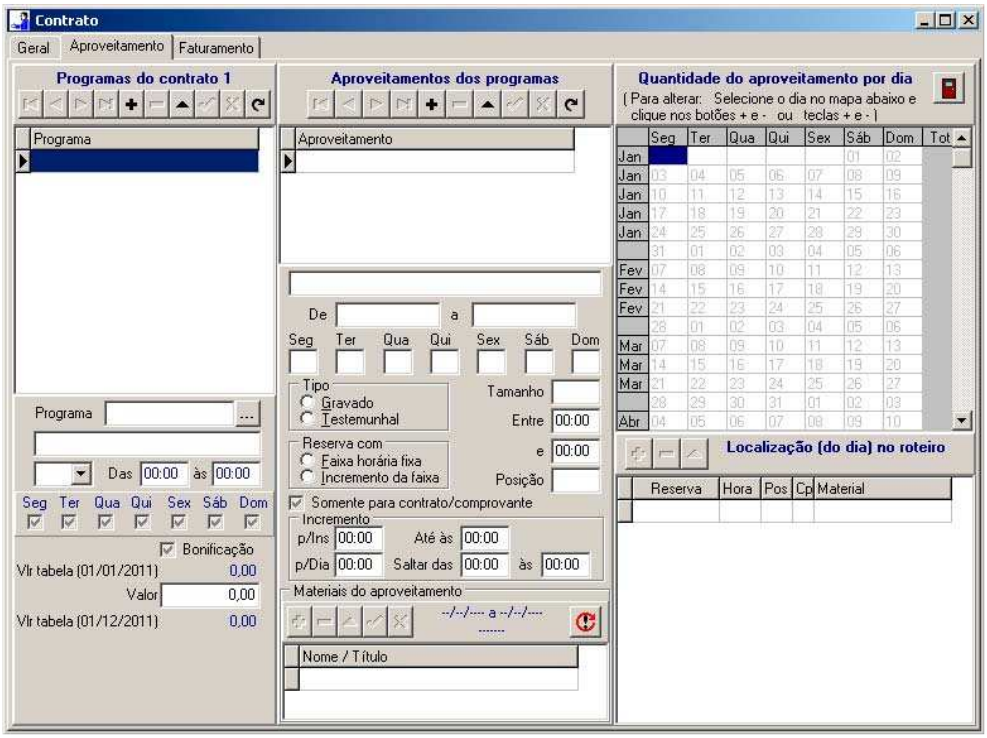

Figura 1.8: Contrato aba Aproveitamento do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

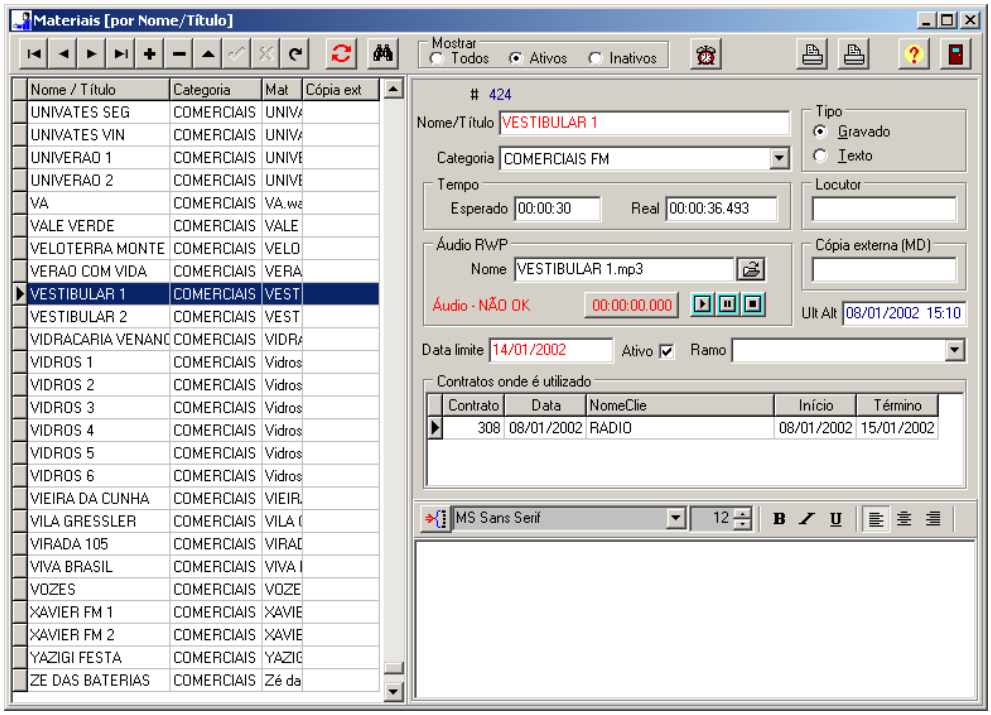

Figura 1.9: Cadastro de materiais do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

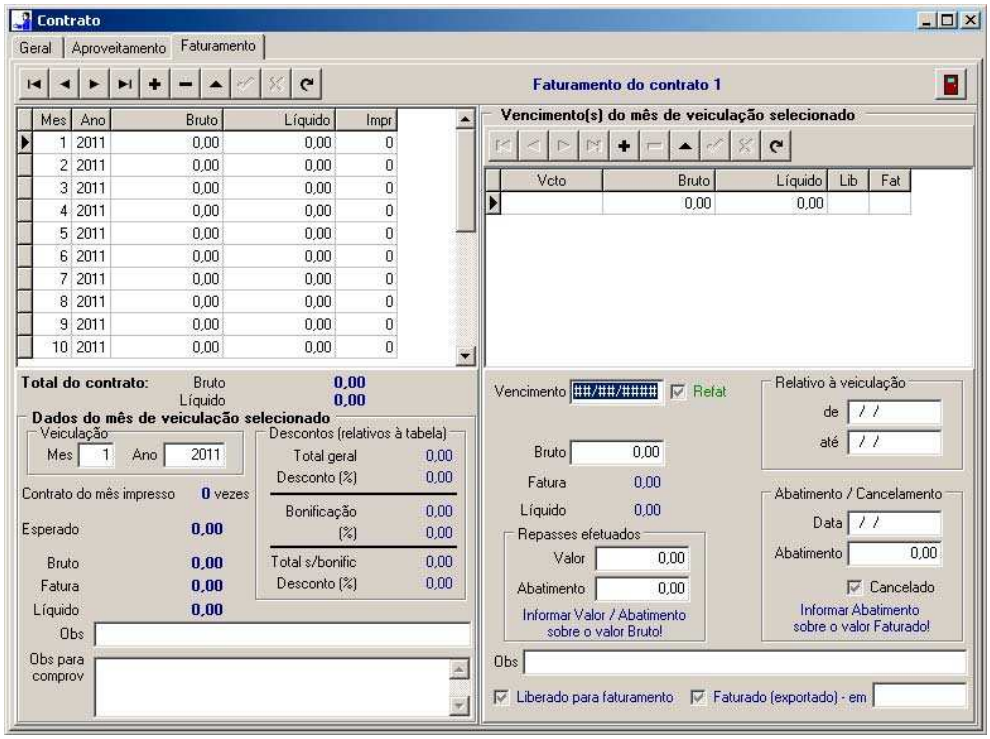

Figura 1.10: Contrato aba Faturamento do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

Inseridos os contratos, pode-se passar para a etapa de criação de roteiros, para isso, deve-se clicar em *Roteiros*, na tela de entrada do módulo OPEC do Sistema SisRadio, Figura 1.3, onde é exibida a janela *Roteiros comerciais*. Nesta etapa, Figura 1.11, é possível, após a

seleção do dia no campo *Data do roteiro*, modificar a estrutura de roteiro, como número de intervalos, duração de cada intervalo e seus horários. Selecionando a opção *Roteiro do dia (modificar)*, segundo botão abaixo do campo *Data do roteiro*, abre-se a janela de manutenção do roteiro, Figura 1.12. Como o próprio nome da opção remete, na janela de manutenção de roteiro, pode se efetuar modificações dentro de cada intervalo comercial, inserindo ou excluindo mídias e alterando os seus posicionamentos dentro de cada intervalo comercial (bloco). Na janela à esquerda, fica a caixa, *Materiais*, na qual apresenta duas abas, *Avulsos* e *Reservas.* Na aba *Reservas* é onde são encontradas as mídias a serem programadas, isto é, as mídias contratadas que devem ser veiculadas neste dia. E na aba *Avulsos*, são listadas todas as mídias cadastradas. Já na caixa ao centro, é mostrado o *Mapa do dia*, e à direta o conteúdo de cada intervalo comercial. Para se fazer as alterações, basta selecionar uma mídia na janela à esquerda e arrastá-la para o *Mapa do dia*, no intervalo desejado. Após inserir a mídia no bloco pode-se alterar seu posicionamento dentro de cada intervalo pela caixa à direta, selecionando a aba *Break*.

Clicando no botão *Alocar reservas*, na caixa *Materiais*, Figura 1.12, se tem a opção de alocar as mídias de forma automática no roteiro de intervalos comerciais. Podendo distribuir as mídias de forma aleatória, para as mídias reservadas, ou copiando as posições do dia anterior, tanto das reservas como das mídias avulsas. O roteiro comercial utilizado neste trabalho foi gerado a partir da cópia das posições do dia anterior, copiando também as mídias avulsas veiculadas.

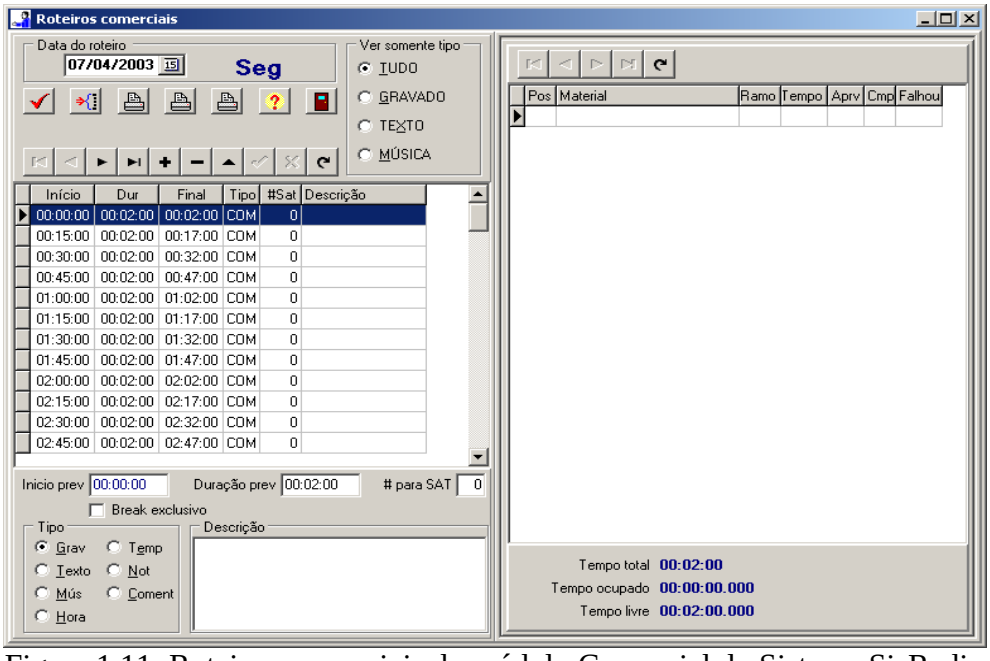

Figura 1.11: Roteiros comerciais do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

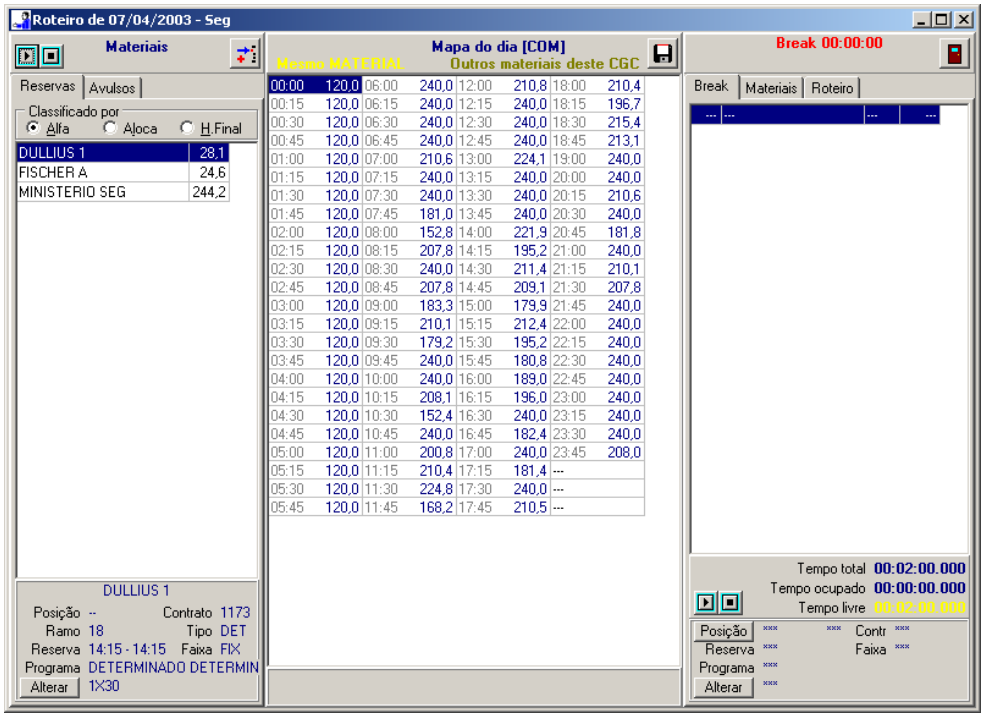

Figura 1.12: Manutenção de roteiro do módulo Comercial do Sistema SisRadio
Fonte: (SIS, 2011)

Ao alocar as mídias reservadas de forma automática, selecionando a opção *Distribuir aleatoriamente*, são exibidas as confirmações apresentadas na Figura 1.13.

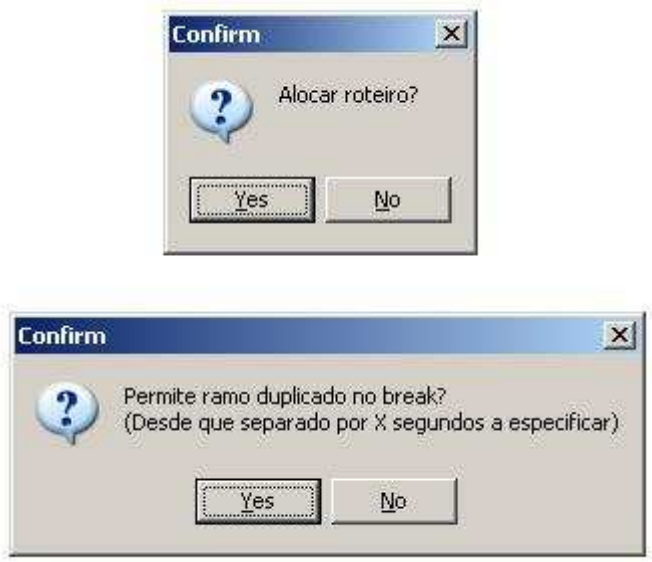

Figura 1.13: Confirmações para distribuir mídias aleatoriamente
Fonte: (AUTOR, 2011)

Confirmando a alocação do roteiro, a segunda janela da Figura 1.13 é exibida, se a opção *Permitir ramo duplicado no break?* for aceita, uma nova janela é apresentada solicitando o tempo em segundos para separar as mídias de mesmo ramo dentro de cada bloco do roteiro comercial.

Depois que as mídias foram inseridas no roteiro comercial, independentemente da forma escolhida pelo roteirista, é necessário ajustar cada intervalo comercial de acordo com o tempo estabelecido na estrutura de roteiros. Concluídas e salvas as alterações do roteiro, se passa para uma nova etapa, que é o envio da grade comercial para o *player*. Para isso, se retorna a tela de entrada do Sistema SisRadio, Figura 1.3, e seleciona-se a opção *Exportar para o estúdio* listada ao clicar no botão *Exp/Imp*. Abrindo a janela *Exportar para o estúdio*, deve se selecionar a opção *Sempre ON*, da caixa *Controle do Satélite*, pois através desta opção que o pulso eletrônico é reconhecido, possibilitando assim o disparo automático do intervalo comercial. Os dados exportados para o *player* são divididos em pastas de acordo com o dia, os intervalos comerciais, correspondem aos arquivos no formato XML (eXtensible Markup Language), como podem ser visualizados na Figura 1.14.

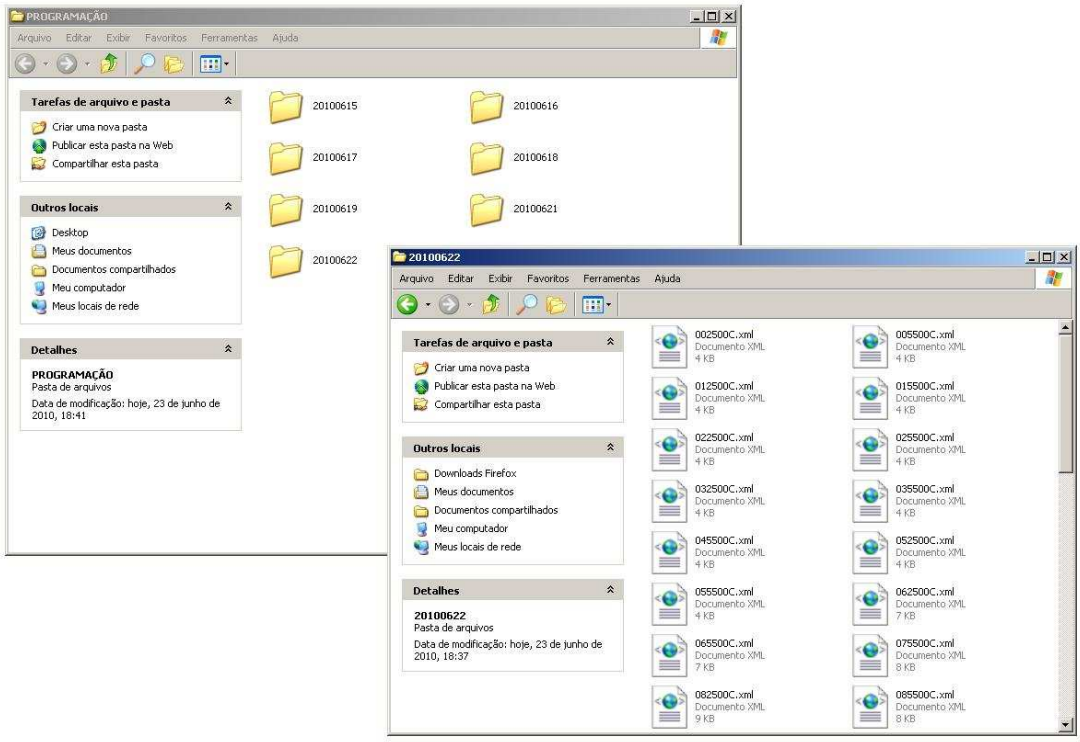

Figura 1.14: Estrutura de pastas do roteiro exportado
Fonte: (AUTOR, 2011)

Já as *tags* dos arquivos XML representam o conteúdo de cada intervalo comercial, isto é, onde estão listadas as mídias (Figura 1.15), e suas informações, como: <path>,

caminho do arquivo; <nome>, nome da mídia; <campanha>, campanha em que a mídia está inserida; <tempo>, tempo da mídia; <formato>, formato da mídia; <cliente>, nome do cliente; <atividade>, atividade do cliente; e <locutor>, nome do locutor. Com as informações de um roteiro gerado pelo módulo OPEC do Sistema SisRadio, foi criada a base de dados utilizada para etapa prática deste trabalho de conclusão, apresentada no terceiro capítulo.

```
- <bloco tp="C" tempo="130269" usr="usuário">
  - <medias>
    + <controle></controle>
    - <media tp="C" ind_tkm="0" ind_tes="0">
        <nro_int/>
        <path>C:\MIDIAS\14256.mp3</path>
        <nro_tkm/>
        <nome>PADARIA DO MANUEL</nome>
        <campanha>OFERTA DO PÃO FRANCÊS</campanha>
        <tempo>34037</tempo>
        <base/>
        <formato>.mp3</formato>
        <testemunhal>0</testemunhal>
        <obs/>
      - <atividade ind_tkm="0">
          <nro_tkm/>
          <nome>PADARIA</nome>
        </atividade>
      - <cliente ind_tkm="0">
          <nro_tkm/>
          <nome>MANUEL OLIVEIRA PANIFICADORA LTDA..</nome>
        </cliente>
      - <locutor ind_tkm="0">
          <nro_tkm/>
          <nome>LOCUTOR 01</nome>
        </locutor>
      </media>
    + <media tp="V"></media>
    + <media tp="C" ind_tkm="0" ind_tes="0"></media>
    + <media tp="C" ind_tkm="0" ind_tes="0"></media>
    + <media tp="V"></media>
    + <media tp="C" ind_tkm="0" ind_tes="0"></media>
    + <media tp="O"></media>
    + <controle></controle>
    </medias>
  </bloco>
```

Figura 1.15: Estrutura do arquivo XML
Fonte: (AUTOR, 2011)

1.3 O problema na construção de um roteiro para emissoras de rádio afiliada

Durante a criação de um roteiro comercial para emissoras de rádio afiliadas o roteirista deve realocar as mídias, de maneira que obedeçam a formatação e a metodologia aplicada pela emissora de rádio. Os softwares de automação comercial de rádio, como apresentado anteriormente, simplificam o desenvolvimento de um roteiro, porém, a revisão, o acerto de tempos, o posicionamento das mídias dentro dos intervalos, as bonificações e a vinhetagem são operações que ficam no encargo do roteirista.

No caso de emissoras de rádio afiliadas, os tempos dos intervalos comerciais devem seguir o padrão estipulado pela emissora geradora. O não cumprimento de tal definição acarretará na formação de “vácuos” ou sobreposição de áudios. Portanto o roteirista preenche os espaços com mídias avulsas, bonificações.

Outro caso que atrai atenção é a alocação de mídias com duplicidade de ramo no intervalo, isto é, como proceder quando há materiais de clientes concorrentes ou até mesmo do próprio cliente num intervalo comercial. No software SisRadio, a ocorrência de duplicidade é sinalizada, o que facilita a visualização para o roteirista, embora, a correção ou a definição da melhor posição na grade é tarefa do roteirista. A Figura 1.16, apresenta uma ocorrência de duplicidade de ramo dentro de um intervalo comercial.

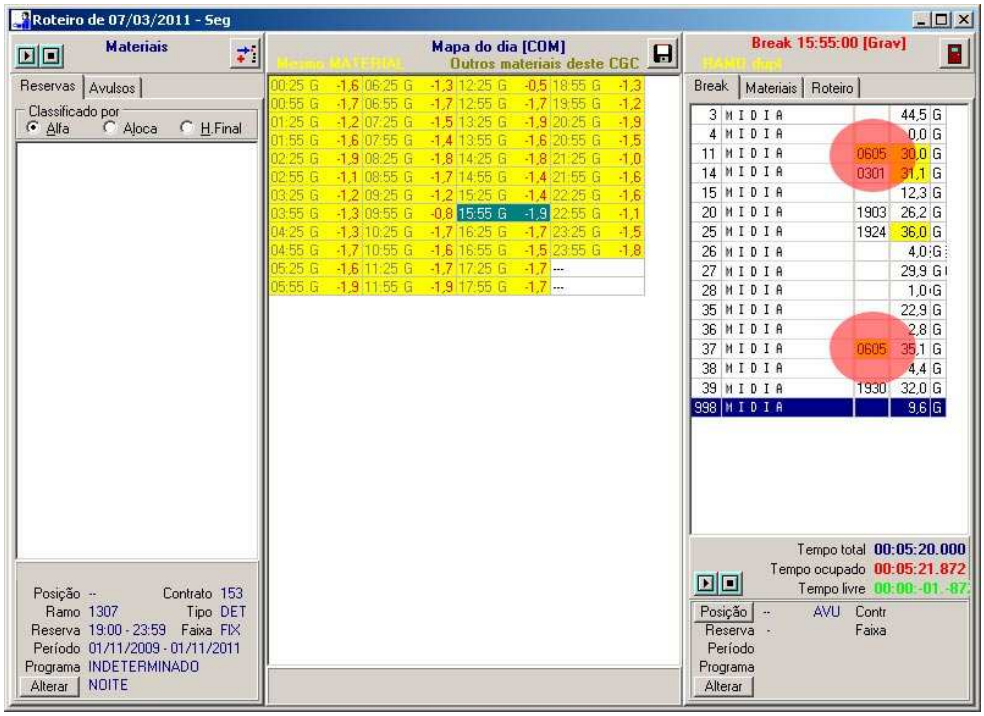

Figura 1.16: Duplicidade de ramo em um intervalo comercial
Fonte: (AUTOR, 2011)

Há também a questão da escolha e da alocação das mídias que devem ser utilizadas para o preenchimento dos espaços dentro dos intervalos comerciais. Outro problema é a alocação dentro do intervalo comercial, numa emissora de rádio, o que faz valorizar uma mídia são as faixas horárias e o posicionamento dentro do intervalo comercial, quanto mais ao topo do bloco, mais chances ele tem de ser ouvido pelo público.

2 APRENDIZADO DE MÁQUINA

Imagine computadores recomendando tratamentos mais eficazes a partir dos dados de prontuários médicos, ou casas que aprendem como otimizar os custos de energia com base na experiência dos padrões de uso de seus moradores. O aprendizado de máquina tem como finalidade, criar sistemas ou métodos que a partir de experiências, consigam aprender, adquirir conhecimento, e assim melhorando a capacidade de tomar decisões.

Aprendizado de máquina, segundo Rezende (2003), é uma área de Inteligência Artificial cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática. Um sistema de AM é um programa que permite obter conclusões através da comparação de um conjunto de exemplos a ele inserido.

De acordo com Liu (2007), o aprendizado de máquina é comparável ao aprendizado humano, que a partir de experiências passadas, adquiri novos conhecimentos, a fim de melhorar a capacidade de realizar suas tarefas no mundo real. No entanto, os computadores não possuem “experiências”, o AM aprende a partir dos dados recolhidos no passado, e desta forma, representa as experiências passadas nas aplicações do mundo real.

O aprendizado de máquina tenta fazer com que os programas de computador “aprendam” com os dados que eles estudam, de tal modo que esses programas tomem decisões diferentes, baseadas nas características dos dados estudados, usando a estatística para os conceitos fundamentais e adicionando heurística avançada da Inteligência Artificial e algoritmos para alcançar os seus objetivos (MARTINHAGO, 2005).

Mas antes de partir para as características, conceitos e definições do aprendizado de máquina, deve-se entender a idéia de aprendizagem. Aprender significa procurar tirar lição, proveito do que se vê ou observa (LUFT, 1988).

Para Russel (2004), a idéia de aprendizagem é que as percepções devem ser usadas não apenas para agir, mas também para melhorar a habilidade do agente[1] para agir no futuro. A aprendizagem ocorre à medida que o agente observa suas interações com o mundo e com seus próprios processos de tomada de decisão.

---

[1] Agente é tudo o que pode ser considerado capaz de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por intermédio de atuadores (RUSSEL, 2004).

Aprendizagem é uma característica exclusivamente humana, e não consiste em somente memorizar exemplos, mas sim, generalizar conclusões como base nestes exemplos.

A indução é a forma de inferência lógica que permite obter conclusões genéricas sobre um conjunto de exemplos. Na indução um conceito é aprendido efetuando-se inferência indutiva sobre exemplos apresentados. A inferência indutiva é um dos principais métodos utilizados para derivar conhecimento novo e predizer eventos futuros (REZENDE, 2003).

O aprendizado indutivo nada mais é do que a obtenção de conclusões genéricas sobre um conjunto particular de exemplos. Como mostra a Figura 2.1, o aprendizado indutivo pode ser dividido em supervisionado, não-supervisionado e por reforço.

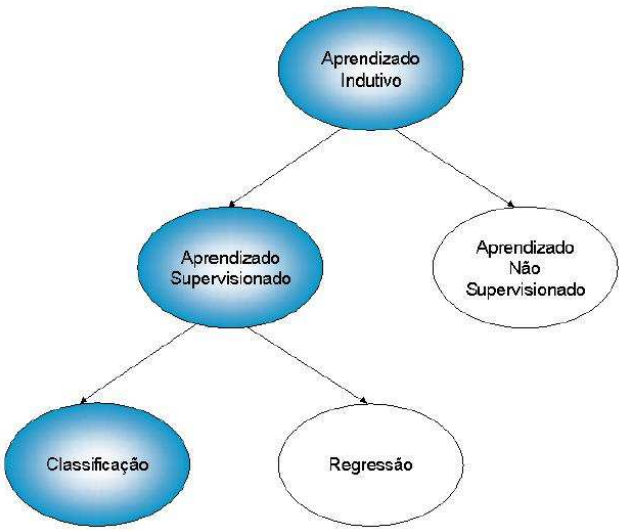

Figura 2.1: Hierarquia do aprendizado
Fonte: (REZENDE, 2003)

No aprendizado supervisionado, o algoritmo de aprendizado (indutor) recebe um conjunto de exemplos de treinamento, os quais são associados a uma classe (saída), de valores discretos ou contínuos, e cada exemplo é descrito por atributos (entrada), que também podem ser de valores discretos ou contínuos. O objetivo do algoritmo é criar um modelo de classificação que ao analisar os novos exemplos, os associe de acordo com os exemplos utilizados no treinamento. O aprendizado supervisionado em função de valores discretos é chamado de classificação, e o aprendizado em função de valores contínuos, regressão.

Já no aprendizado não-supervisionado, o algoritmo analisa os exemplos e tenta de alguma maneira agrupá-los, esse processo tem o nome de *clusters*.

No aprendizado por reforço, o conjunto de exemplos de treinamento é formado por apenas atributos de entrada, e ao invés do aprendizado supervisionado, que é associado a uma

classe (saída), o aprendizado por reforço, informa valores de recompensa ou penalidade. Um exemplo simplificado do seu significado, é o apresentado por Russel (2004), imagine disputar um jogo de xadrez cujas regras você não conhece; depois de aproximadamente uma centena de movimentos, seu oponente anuncia: “Você perdeu.” O método de aprendizagem por reforço, aprende em função de valores de recompensa ou penalidade. Neste trabalho, vamos nos concentrar em descrever o problema da classificação no aprendizado supervisionado.

2.1 Aprendizado supervisionado

O aprendizado supervisionado, segundo Liu (2007), tem sido um grande sucesso nas aplicações do mundo real. Ele é usado em quase todos os domínios, incluindo textos e domínios Web. Porém o problema do aprendizado supervisionado compreende a aprendizagem de uma função a partir de exemplos de entradas e saídas.

Um exemplo ou instância consiste em um par, $(x, f(x))$, onde x é a entrada e $f(x)$ é a saída da função aplicada a x. E a partir de um conjunto de exemplos o indutor, algoritmo de aprendizado, tem a tarefa de induzir uma função h, chamada hipótese, na qual se assemelha de f. O grande desafio do aprendizado é justamente esse, encontrar uma hipótese que suponha corretamente os exemplos ainda não vistos. Um exemplo é formado por uma tupla de valores de atributos. Tais atributos podem ser do tipo: discreto (nominal) ou contínuo, por exemplo, um número real.

No aprendizado supervisionado, todo exemplo tem por propriedade um atributo especial, chamado de classe, através dele, que se deseja aprender para fazer previsões a respeito. Desta forma, um conjunto de exemplos é constituído por exemplos, contendo valores de atributos e uma sua classe associada. Como mostra na Tabela 1.1:

Tabela 1.1: Conjunto de exemplos

E	A_1	A_2	...	A_m	C
E_1	A_{11}	A_{12}	...	A_{1m}	C_1
E_2	A_{21}	A_{22}	...	A_{2m}	C_2
⋮	⋮	⋮	⋱	⋮	
E_n	A_{n1}	An2	...	A_{nm}	C_n

Fonte: (AUTOR, 2010)

Durante o processo de aprendizado, normalmente um conjunto de exemplos é dividido em dois subconjuntos disjuntos: o conjunto de treinamento, utilizado para gerar um

modelo de classificação por meio de um algoritmo de aprendizado, e o conjunto de teste, que é usado para avaliar a precisão do modelo de classificação.

Liu (2007) ilustra as etapas do aprendizado, aqui apresentadas na Figura 2.2. Na etapa 1, um algoritmo de aprendizado utiliza um conjunto de treinamento para gerar um modelo de classificação. Esta etapa é chamada de *training phase* (fase de treinamento). Na etapa dois, o conjunto de teste é usado para testar o modelo de classificação, para se obter uma exatidão da classificação. Se a precisão do modelo sobre o conjunto de teste for satisfatória, o modelo de classificação pode ser utilizado para predizer classes de novos casos no mundo real. Já se a precisão não for satisfatória, é preciso voltar e escolher um algoritmo de aprendizado diferente e/ou fazer algum tratamento posterior nos dados, pois no mundo real, é comum encontrarmos dados imperfeitos (ruído de dados). Essa etapa de avaliação dos dados é chamada de *data pre-processing* (pré-processamento). É importante salientar ainda, que a prática do aprendizado geralmente envolve muitas repetições de etapas antes de se construir um modelo de classificação satisfatório.

Figura 2.2: Etapas do aprendizado
Fonte: (LIU, 2007)

A precisão (*p*) de um modelo de classificação pode ser medida a partir de uma simples equação:

$$p = x/z \quad (1)$$

Na Equação (1), *x* representa o número de classificações corretas e *z* o número total de casos do conjunto de teste. Uma classificação correta significa que o modelo de classificação previu uma classe igual à classe do conjunto de teste. Outra maneira de se medir a precisão de um modelo de classificação, é usando uma matriz de confusão.

De acordo com Rezende (2003), a matriz de confusão de uma hipótese *h* oferece uma medida efetiva do modelo de classificação, ao mostrar o número de classificações corretas *versus* as classificações preditas para cada classe sobre um conjunto de teste. O número de acertos, para cada classe se localiza na diagonal principal da matriz. Se os demais elementos

da matriz forem iguais a zero, significa que o modelo de classificação não cometeu erros. A Tabela 1.2, apresenta uma matriz de confusão de um modelo de classificação ideal, que não cometeu erros, e na Tabela 1.3, uma matriz de confusão da qual o modelo de classificação deve ser revisto.

Tabela 1.2: Matriz de Confusão sem erros

P_1	P_2	P_3	P_4	Classe
5	0	0	0	C_1
0	7	0	0	C_2
0	0	3	0	C_3
0	0	0	1	C_4

Fonte: (AUTOR, 2011)

Tabela 1.3: Matriz de Confusão com erros

P_1	P_2	P_3	P_4	Classe
5	0	0	0	C_1
0	5	0	0	C_2
0	2	3	0	C_3
0	0	0	1	C_4

Fonte: (AUTOR, 2011)

Observando a Tabela 1.2, imagine um conjunto de teste que possua 16 instâncias associadas às classes (C_1, C_2, C_3 e C_4), e estas com um número de instâncias correspondentes (5,7,3,1). Gerado um modelo de classificação, o conjunto de teste é submetido à avaliação. Para que o modelo de classificação seja considerado ideal, as classes preditas (P_1, P_2, P_3 e P_4) devem ter as mesmas classificações do que as classes do conjunto de teste. Portanto, na coluna P_1, linha C_1, deve apresentar cinco exemplos classificados corretamente, assim consecutivamente com nos demais, $E(P_2, C_2) = 7$, $E(P_3, C_3) = 3$ e $E(P_4, C_4) = 1$.

Porém, na Tabela 1.3, existem exemplos que foram classificados de modo errado, por exemplo, $E(P_2, C_3) = 2$, significa que dois exemplos do conjunto de teste deveriam ser classificados com os valores da classe C_2, mas no entanto, o valor predito foi o correspondente a classe C_3.

O erro de precisão de um modelo de classificação pode muitas vezes ser associado a fatores, como, ruído de dados, ou até mesmo o excesso de ajuste do algoritmo ou de seus parâmetros, pelo desenvolvedor. Na literatura este excesso de ajustes tem o nome de *overtuning*. Já quando o modelo de classificação se ajusta em excesso ao conjunto de treinamento, se tem um *overfitting*. Pode ocorrer também um *underfitting*, que é quando o modelo de classificação se ajusta muito pouco em relação ao conjunto de treinamento, fazendo com que o modelo de classificação se torne muito genérico.

Para lidar com problemas de ruído e *overfitting* se utiliza da técnica de poda, que consiste em eliminar exemplos de treinamento que possuam baixa relevância para a geração da hipótese. Conforme Rezende (2003) existe dois métodos de poda:

Pré-poda: que durante a geração da hipótese, alguns exemplos de treinamento são ignorados, de maneira que a hipótese final não classifique todos os exemplos de treinamento corretamente;

Pós-poda: após a geração de uma hipótese, ela é generalizada por meio da eliminação de algumas partes, como a exclusão de ramos em uma árvore de decisão ou de algumas condições nas regras induzidas.

2.1.1 Indução de árvores de decisão

Árvores de decisão, segundo Mitchell (1997), é um dos métodos de ensino que estão entre os mais populares algoritmos de inferência indutiva, e têm sido aplicados com sucesso a uma ampla gama de tarefas de aprendizado, como para diagnosticar casos médicos e para aprender a avaliar o risco de crédito dos requerentes de empréstimo.

Russel (2004) observa que a indução de árvores de decisão é uma das formas mais simples, e ainda assim mais bem-sucedidas, de um algoritmo de aprendizado. Ela serve como uma boa introdução à área de aprendizado indutivo, tem fácil implementação, e o mais importante, é simples de ser compreendida por seres humanos.

Em aprendizado supervisionado, um conjunto de exemplos é submetido a um algoritmo de aprendizado, que por sua vez gera um modelo de classificação. Este modelo de classificação pode ser representado graficamente por uma árvore de decisão.

Entre os algoritmos de aprendizado que induzem modelos de classificação, está a indução de árvore de decisão. Uma árvore de decisão é uma estrutura recursiva definida com (PATR *et al*., 2003)

- um nó folha que corresponde a uma classe, ou
- um nó de decisão que contém um teste sobre algum atributo. Para cada um dos possíveis valores do atributo tem-se um ramo para outra árvore de decisão (subárvore). Cada subárvore contém a mesma estrutura de uma árvore.

Uma árvore de decisão pode ser usada para classificar novos exemplos, começando a partir da raiz, descendo até as folhas onde estão os possíveis resultados. A Figura 2.3 é um exemplo de uma árvore de decisão, cada círculo é um teste de atributo para um determinado conjunto de exemplos, e cada quadrado representa uma classe.

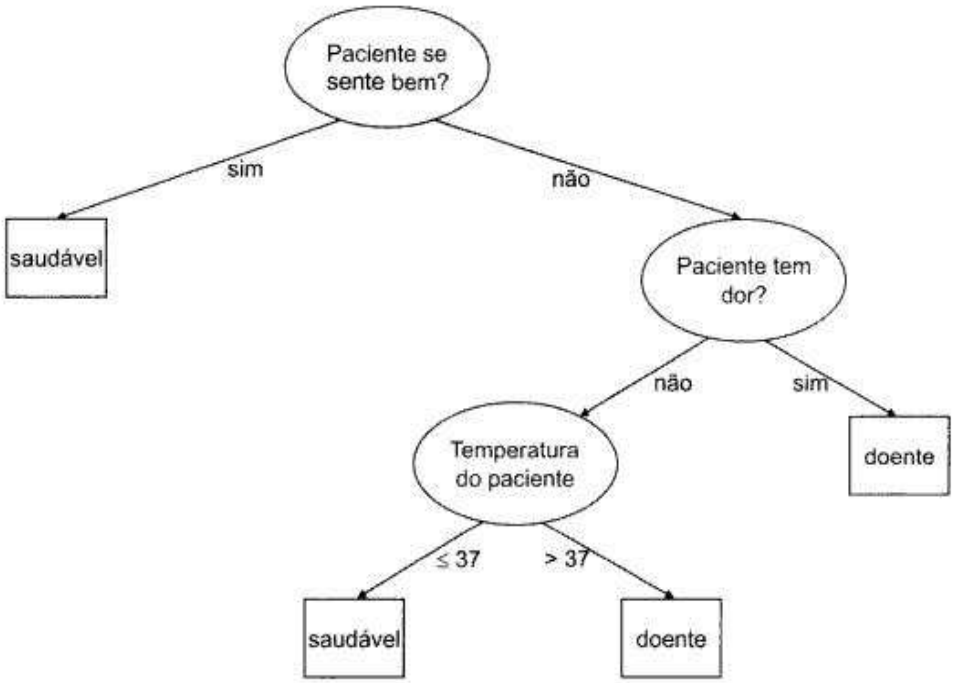

Figura 2.3: Exemplo de uma árvore de decisão
Fonte: (REZENDE, 2003)

2.1.2 Construindo uma árvore de decisão

O método para a construção de uma árvore de decisão a partir de um conjunto de treinamento, segundo Rezende (2003), é surpreendentemente simples. Porém deve se seguir alguns passos: escolher o conjunto de treinamento; definir qual atributo será usado para particionar o conjunto; e por final a poda da árvore de decisão.

Para exemplificar o processo de construção de uma árvore de decisão será usado um conjunto de exemplos para classificação de pedidos de empréstimos. Utilizando técnicas baseadas no sistema C4.5 de Quinlan, como mostra na Tabela 1.4. Cada exemplo possui os seguintes atributos:

- **Idade**: este exemplo possui três valores nominais para o atributo, jovem, médio e velho;
- **Emprego**: indica se o candidato tem um trabalho;
- **Residência**: o terceiro atributo mostra se o candidato possui uma casa;
- **Crédito**: este atributo mostra os três possíveis valores considerados para a avaliação de crédito do candidato, justo, bom e excelente;
- **Classe**: mostra se o pedido de empréstimo foi aprovado ou reprovado, representa o atributo-classe.

Tabela 1.4: Conjunto de exemplos para classificação de pedidos de empréstimos

Exemplo	Idade	Emprego	Residência	Crédito	Classe
1	jovem	não	não	justo	reprovado
2	jovem	não	não	bom	reprovado
3	jovem	sim	não	bom	aprovado
4	jovem	sim	sim	justo	aprovado
5	jovem	não	não	justo	reprovado
6	médio	não	não	justo	reprovado
7	médio	não	não	bom	reprovado
8	médio	sim	sim	bom	aprovado
9	médio	não	sim	excelente	aprovado
10	médio	não	sim	excelente	aprovado
11	velho	não	sim	excelente	aprovado
12	velho	não	sim	bom	aprovado
13	velho	sim	não	bom	aprovado
14	velho	sim	não	excelente	aprovado
15	velho	não	não	justo	reprovado

Fonte: (LIU, 2007)

De acordo com Rezende (2003), a chave para o sucesso de um algoritmo de aprendizado por árvore de decisão é o critério utilizado para escolher o atributo que particiona o conjunto de exemplos. Os critérios para seleção do atributo são:

- **aleatória**: seleciona-se qualquer atributo do conjunto de exemplos;
- **menos valores**: é selecionado o atributo que tem o menor número de valores possíveis;
- **mais valores**: o atributo selecionado é aquele que tem o maior número de valores possíveis;
- **ganho máximo**: o atributo selecionado resultará no menor número de subárvores.

Para este exemplo, será escolhido o atributo *Emprego*, desta forma o teste baseado em um único atributo obedece ao critério de *ganho máximo*. Com isso, o conjunto de exemplos foi dividido em dois subconjuntos, como mostra na Tabela 1.5.

Tabela 1.5: Conjunto de exemplos dividido em dois subconjuntos

Teste	Exemplo	Idade	Emprego	Residência	Crédito	Classe
se emprego = não	1	jovem	não	não	justo	reprovado
	2	jovem	não	não	bom	reprovado
	5	jovem	não	não	justo	reprovado
	6	médio	não	não	justo	reprovado
	7	médio	não	não	bom	reprovado
	9	médio	não	sim	excelente	aprovado
	10	médio	não	sim	excelente	aprovado
	11	velho	não	sim	excelente	aprovado
	12	velho	não	sim	bom	aprovado
	15	velho	não	não	justo	reprovado
se emprego = sim	3	jovem	sim	não	bom	aprovado
	4	jovem	sim	sim	justo	aprovado
	8	médio	sim	sim	bom	aprovado
	13	velho	sim	não	bom	aprovado
	14	velho	sim	não	excelente	aprovado

Fonte: (LIU, 2007)

A partir da divisão do conjunto de exemplo, se tem o nó raiz da árvore, que é o atributo *Emprego*, e os seus resultados formam os ramos da árvore de decisão (Figura 2.4).

Figura 2.4: Nó raiz da árvore de decisão
Fonte: (LIU, 2007)

Como pode ser notado na Tabela 1.5, o primeiro subconjunto ainda contém exemplos com classes diferentes. Por isso é necessário executar outro teste com base em um único atributo. Seguindo o padrão do primeiro teste, será escolhido o atributo *Residência*, pois possui um número menor de valores para seus exemplos, o resultado da escolha do atributo *Residência* está representado na Tabela 1.6. Pode se observar que os exemplos foram divididos em três subconjuntos, cada um deles contendo valores iguais para o atributo-classe.

É importante salientar que uma boa seleção de atributos para o particionamento do conjunto de exemplos, gera uma árvore decisão menor e mais precisa, como mostra na Figura 2.5, onde *Residência* é um nó interno e *Aprovado* e *Reprovado* são os nós folha.

Tabela 1.6: Conjunto de exemplos dividido em três subconjuntos

Teste	Exemplo	Idade	Emprego	Residência	Crédito	Classe
se emprego = não	1	jovem	não	não	justo	reprovado
	2	jovem	não	não	bom	reprovado
	5	jovem	não	não	justo	reprovado
	6	médio	não	não	justo	reprovado
	7	médio	não	não	bom	reprovado
	15	velho	não	não	justo	reprovado
se emprego = não *e* residência = sim	9	médio	não	sim	excelente	aprovado
	10	médio	não	sim	excelente	aprovado
	11	velho	não	sim	excelente	aprovado
	12	velho	não	sim	bom	aprovado
se emprego = sim	3	jovem	sim	não	bom	aprovado
	4	jovem	sim	sim	justo	aprovado
	8	médio	sim	sim	bom	aprovado
	13	velho	sim	não	bom	aprovado
	14	velho	sim	não	excelente	aprovado

Fonte: (LIU, 2007)

Figura 2.5: Árvore de decisão resultante
Fonte: (LIU, 2007)

Conforme Rezende (2003), a poda de árvore de decisão, em geral, pode melhorar o desempenho para exemplos não vistos. Embora a poda descarte algumas informações, ela se faz necessária quando o aprendizado ocorre em exemplos contendo ruído.

Como a árvore de decisão do exemplo foi gerada a partir de um conjunto de exemplos muito pequeno, no qual não apresentava dados imperfeitos (ruídos), não houve a necessidade de poda-lá.

3 MEDOTOLOGIA DE DESCOBERTA DE CONHECIMENTO

3.1 Descoberta de conhecimento em banco de dados e mineração de dados

Com o avanço dos computadores e dos softwares, nas últimas duas décadas, o mercado de trabalho exigiu um aumento da produtividade e da qualidade em todos os campos de atuação. Hoje para se manterem competitivas no mercado, as organizações precisam ter acesso às informações importantes, e ainda, ter meios de utilizá-las no processo de tomada de decisões (MARTINHAGO, 2005). Para isso é necessário técnicas e ferramentas de análise de dados. Uma área de pesquisa que vem crescendo e que atrai muitos pesquisadores é o processo de descoberta de conhecimento em banco de dados (*Knowlege Discovery in Databases*). Conforme Wiederhold (1996), a área de descoberta de conhecimento passou a ser uma das áreas mais estudadas e mais desejadas da computação.

Descoberta de conhecimento em banco de dados, segundo Fayyad *et al*. (1996a), é o processo não trivial para identificar padrões válidos, novos, potencialmente úteis e compreensíveis em dados existentes. A partir de 1989, o termo Descoberta de Conhecimento em Banco de Dados foi utilizado para se referir ao processo total de descoberta de conhecimento em banco de dados, com a aplicação de técnicas de mineração de dados. Já para Liu (2007), mineração de dados também é chamada de descoberta de conhecimento em banco de dados. E é comumente definida como o processo de descobrir padrões úteis ou conhecimento a partir de fonte de dados, como: base de dados, textos, imagens e Internet.

De acordo com Carvalho (2002), muitas vezes os termos são confundidos como sinônimo. Porém, o termo Descoberta de Conhecimento em Banco de Dados é empregado para descrever todo o processo de extração de conhecimento de um conjunto de dados, e Mineração de Dados refere-se a uma fase deste processo, como mostra na Figura 3.1.

Figura 3.1: Relação entre KDD e mineração de dados
Fonte: (CARVALHO, 2002).

O processo de descoberta de conhecimento em banco de dados, segundo Fayyad *et al*. (1996b) é composto de cinco fases que envolvem a coleta dos dados, o pré-processamento de dados, a transformação de dados, a mineração de dados e a avaliação e interpretação dos resultados. A seguir na Figura 3.2 são apresentadas as cinco fases do processo de KDD.

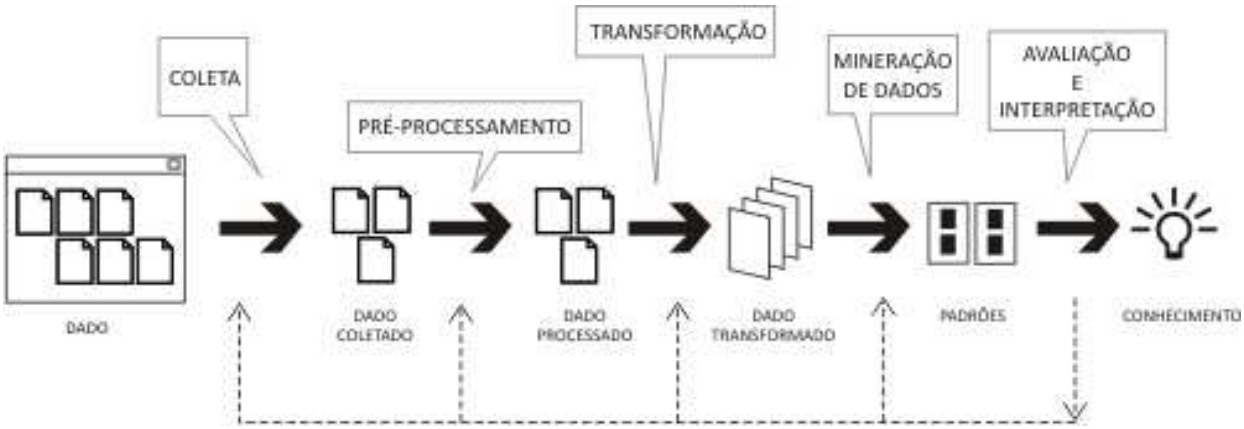

Figura 3.2: Principais fases do processo de KDD
Fonte: (FAYYAD *et al*., 1996b)

Na seção a seguir são traçadas as principais fases do processo de KDD, e na mesma ordem, são apresentadas as atividades desenvolvidas neste trabalho para elaboração de um conjunto de exemplos necessário pra a realização de experimentos de AM exibidos no Capítulo 5.

3.2 Fases do Processo de KDD

3.2.1 Coleta de dados

A primeira fase consiste em coletar os atributos necessários para a tarefa de descoberta de conhecimento. Segundo Batista (2003 apud PROVOST & DANYLUK, 1995) coletar dados é uma atividade crítica porque os dados podem não estar disponíveis em um formato apropriado para serem utilizados no processo de descoberta de conhecimento em banco de dados. Ou, mesmo se disponíveis, os dados podem precisar ser rotulados com o auxílio de um especialista.

Para iniciar a coleta de dados no trabalho, primeiramente foi escolhido o roteiro comercial gerado no dia 1º de Março de 2011 para originar a tabela de atributo-valor. A definição para o uso deste roteiro se deu devido ao modelo de estrutura, que corresponde a estrutura utilizada entre o período de segunda a sexta-feira, no qual, apresenta um número maior de anunciantes no período, o que induz também a uma maior diversidade de exemplos,

para a realização dos testes. Este roteiro foi exportado em formato XML, o que auxiliou nesta fase. Porém foram adicionados outros atributos, que não estão presentes nos arquivos XML. Pelo *Mapa do dia*, onde é realizada a manutenção do roteiro, e através do conhecimento do especialista, o roteirista de intervalos comerciais, como mostra a Figura 3.3.

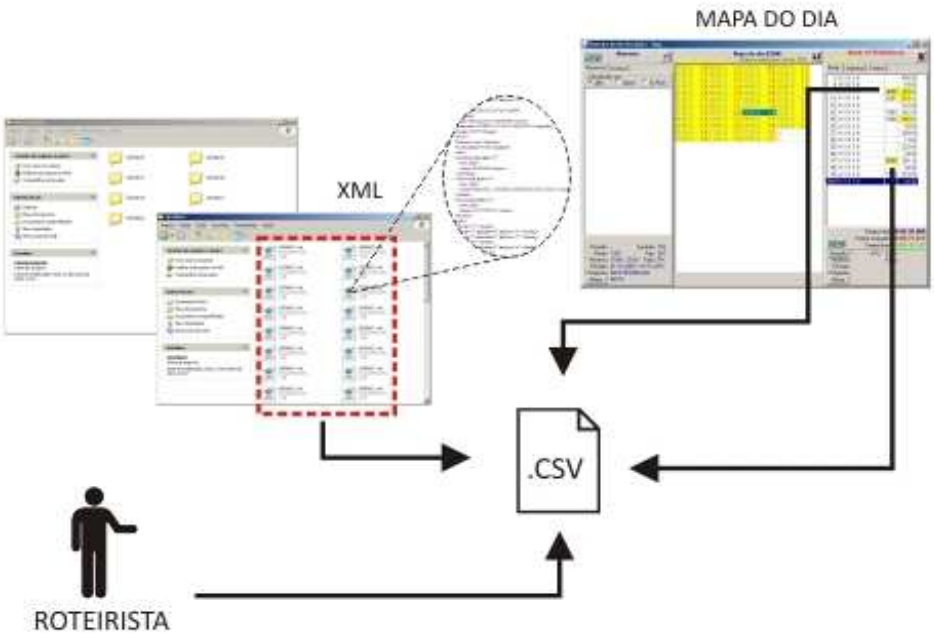

Figura 3.3: Coleta de dados
Fonte: (AUTOR, 2011)

Os dados dos arquivos XML, gerados a partir da operação do módulo OPEC do Sistema SisRadio, foram exportados para um arquivo CSV, após este processo, foram inseridos manualmente as informações do *Mapa do dia* (Figuras 3.4 e 3.5) e as fornecidas pelo especialista. A unificação dos dados conclui a fase de coleta.

3.2.2 O pré-processamento e transformação dos dados

O próximo passo é o pré-processamento, que consiste em verificar a qualidade dos dados coletados, se há presença de ruídos, anomalias, inexistência de dados, entre outros problemas. Conforme Michalski (1998), a tarefa de pré-processamento de dados é umas das mais custosas dentro da análise de dados, pois a qualidade dos dados reflete diretamente na qualidade do conhecimento gerado a partir desses dados. Na análise inicial da base de dados, detectaram-se vários campos em branco, tais campos foram corrigidos manualmente, isto é, eliminados ou preenchidos com um ponto de interrogação ‘?’. Isto porque no software *Weka*, o ponto de interrogação é o sinal que representa um campo nulo. Os atributos *tp*, *usr*,

nro_int_ctr, *txt_nome_ctr*, *txt_param_ctr*, *tempo2*, *ind_tkm*, *ind_test*, *nro_int*, *path*, *nro_tkm*, *campanha*, *base*, *formato*, *testemunhal*, *obs*, *ind_tkm5*, *ind_tkm6*, *ind_tkm8*, *ind_tkm11*, *ind_tkm12*, *nome13* e *texto*, oriundos dos arquivos XML foram eliminados, pois não apresentavam registros ou relevância para o trabalho. Mantendo então somente os atributos *bloco*, *tempo*, *nome*, *temp4*, *nome7* e *nome10* que foram renomeados respectivamente para: *Bloco*, *Tempo_do_bloco*, *Mídia*, *Tempo_da_mídia*, *Anunciante* e *Ramo*, descritos a seguir:

- **Bloco**: corresponde ao bloco em que a mídia está alocada, exemplo, 00:55, 14:55, 16:25 e 21:25;
- **Tempo_do_bloco**: tempo do bloco em segundos;
- **Mídia**: nome da mídia;
- **Tempo_da_mídia**: tempo da mídia em segundos;
- **Anunciante**: nome do anunciante em que a mídia pertence;
- **Ramo**: ramo em que o anunciante está cadastrado, exemplo, restaurante, vestuário, hotel e supermercados.

E pelo *Mapa do dia*, onde é realizada a manutenção do roteiro no software OPEC, foi possível coletar outros atributos, como:

- **Posição_no_bloco**: atributo que determina a posição de cada mídia dentro do intervalo comercial, e é representado pela faixa numérica de 1 a 999;
- **Programa**: corresponde ao nome do programa ou faixa horária em que a mídia está vinculada, exemplo, Hits Manhã e Faixa 24h;
- **Veiculação**: representa a modalidade em que a mídia deve ser veiculada, neste trabalho foi dividida em avulso, determinado, indeterminado, patrocínio, vinheta e citação;
- **Reserva**: indica qual bloco ou faixa de blocos a mídia deve ser veiculada, exemplo, 06:25, 00:25 - 05:55, 11:55 e 00:25 – 23:55;
- **Reserva_correta**: indica se a mídia foi alocada no bloco correto de sua reserva ou não (Figura 3.4);
- **Duplicidade_de_ramo**: verifica a existência de mídias de mesmo ramo no bloco (Figura 3.5);

- **Duplicidade_de_anunciante**: indica a ocorrência de mídias de mesmo anunciante no bloco.

Conforme ilustra a Figura 3.6, os valores de cada atributo proveniente do *Mapa do dia* foram coletados de forma manual, isto é, partindo das observações do *Mapa do dia*, se inseriu os valores de cada atributo no arquivo CSV.

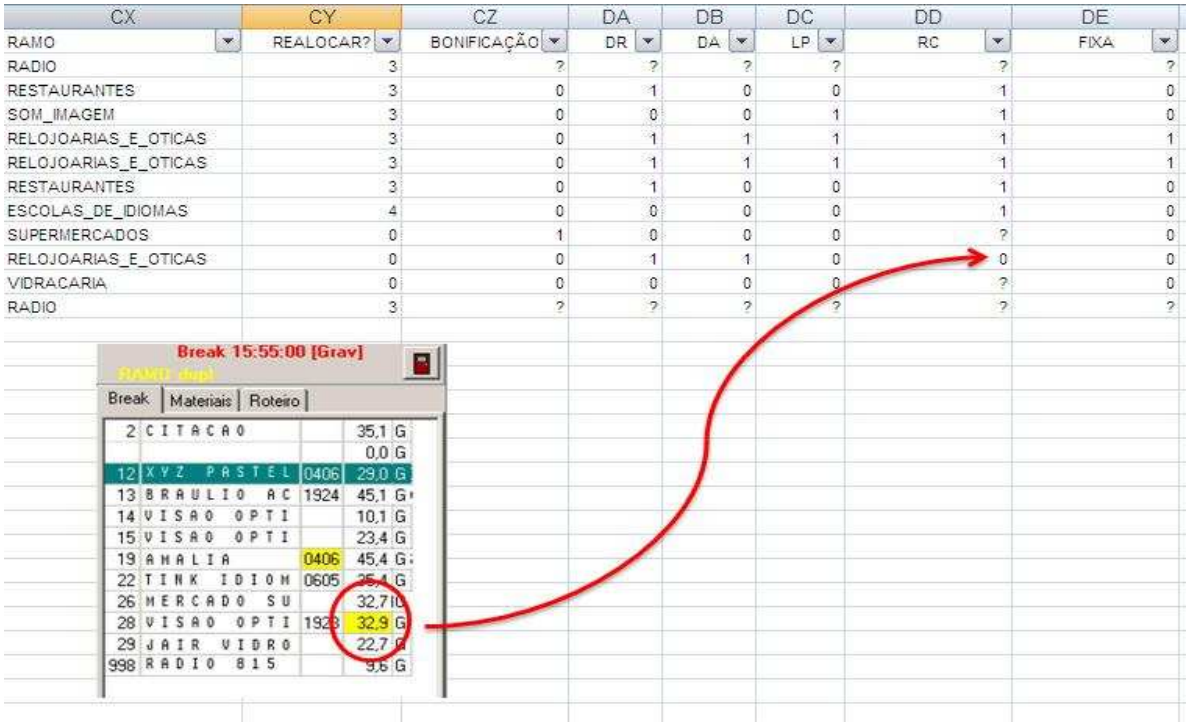

Figura 3.4: Valor do atributo *Reserva_correta* coletado pelo *Mapa do dia*
Fonte: (AUTOR, 2011)

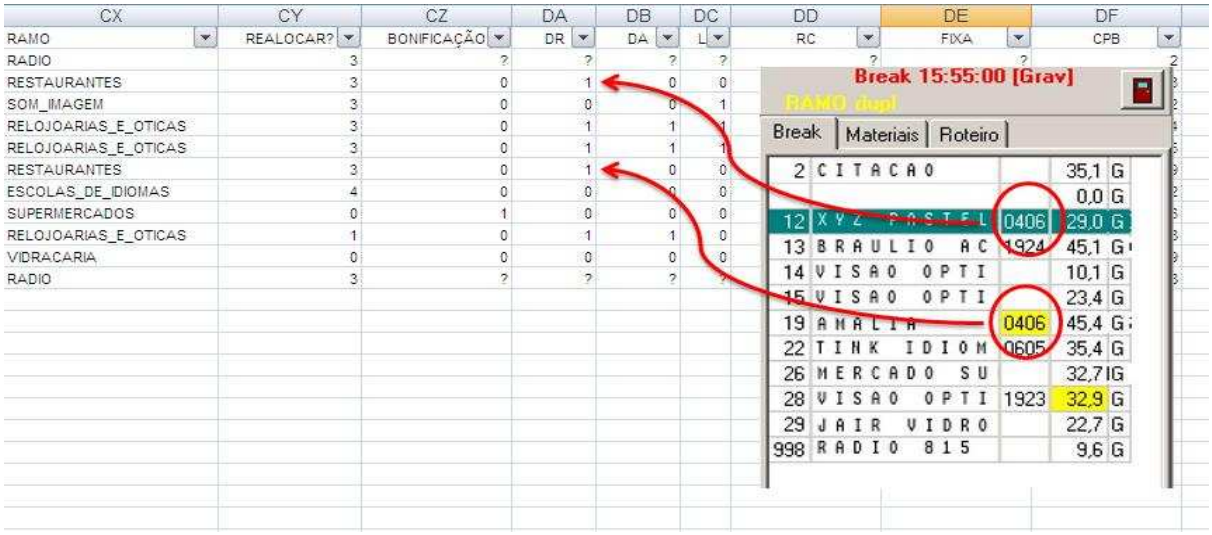

Figura 3.5: Valor do atributo *Duplicidade_de_ramo* coletado pelo *Mapa do dia*
Fonte: (AUTOR, 2011)

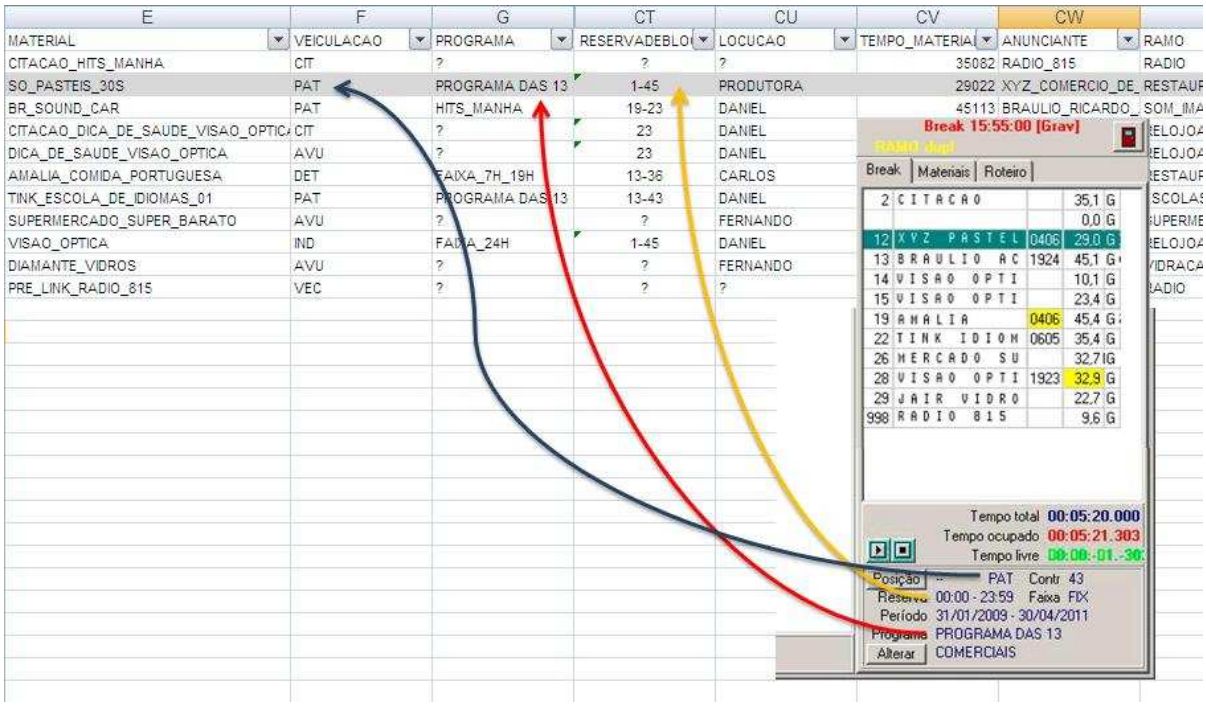

Figura 3.6: Coleta de dados a partir do *Mapa do dia*
Fonte: (AUTOR, 2011)

Já através do conhecimento do especialista, o roteirista de intervalos comerciais, se obteve os atributos:

- **Bonificação**: informa se a mídia é uma bonificação ou não;
- **Locutor**: neste atributo é informado o nome do locutor, isto é, com que voz a mídia foi gravada, exemplo, Daniel, Lucas e Braulio;
- **Bloco_corrigido**: sugestão, informada pelo especialista, para melhor posicionamento da mídia dentro do bloco, tem a mesa representação do atributo *Posição_do_bloco*;
- **Material_fixo_na_estrutura**: informa quando uma mídia tem posição fixa em um determinado bloco, previamente cadastrada na estrutura de roteiros do módulo OPEC;
- **Locuções_aproximadas**: este atributo indica a ocorrência de mídias consecutivas gravadas com a mesma voz;

- **Realocar**: atributo-meta, que indica se a mídia deve ser realocada ou não seguindo uma escala de 0 a 4, onde 0 determina a maior propensão para realocação e 4 menor incidência para a mídia ser transferida de posição ou de bloco.

Desta forma, com a junção dos atributos coletados do *Mapa do dia*, (Figura 3.4) dos arquivos XML e os indicados pelo especialista a base de dados ficou definida com 19 atributos e 554 instâncias.

Depois da fase de pré-processamento pode existir a necessidade de transformar a forma de representação dos dados, para garantir maior eficiência dos algoritmos que serão utilizados para a extração de padrões. Um exemplo citado por Batista (2003) é o caso de atributos do tipo data e hora, na qual, diversas implementações algorítmicas utilizadas em mineração de dados não são capazes de analisá-los. Em vista disso, como a estrutura dos roteiros é dividida em faixas horárias, se optou em transformar tais valores, em atributos do tipo inteiro, como os atributos *Bloco* e *Reserva*. Outra transformação foi a normalização dos valores de alguns atributos para intervalos específicos, como, ‘0,1’ e ‘0,1,2,3,4’ para o atributo-meta.

Após realizar as etapas descritas acima, foi necessário formatar a base de dados em CSV de acordo com o formato de entrada do software *Weka*, isto é, um arquivo no formato ARFF. Um arquivo ARFF consiste de uma lista de todas as instâncias, com os valores de atributos para cada instância separados por vírgula. Para isso, deve se carregar o arquivo da base de dados em CSV em um editor de texto, e efetuar três modificações:

1. Converter o ponto e vírgula presente no arquivo CSV em apenas vírgula, em toda a extensão do arquivo;
2. Inserir o cabeçalho do arquivo ARFF, de modo que contenha os comandos @ relation, @attribute e @data como na Figura 4.2 do Capítulo 4;
3. E por fim, salvar o arquivo texto em .arff.

Modificado o arquivo para o formato ARFF, a base de dados já pode ser analisada pelo software de mineração de dados *Weka*. Caso exista alguma inconsistência no arquivo ARFF, tais como, a ausência de valores de atributos ou a não declaração de valores de atributos no cabeçalho, como mostra Figura 3.7, o arquivo ARFF deve ser revisto e corrigido.

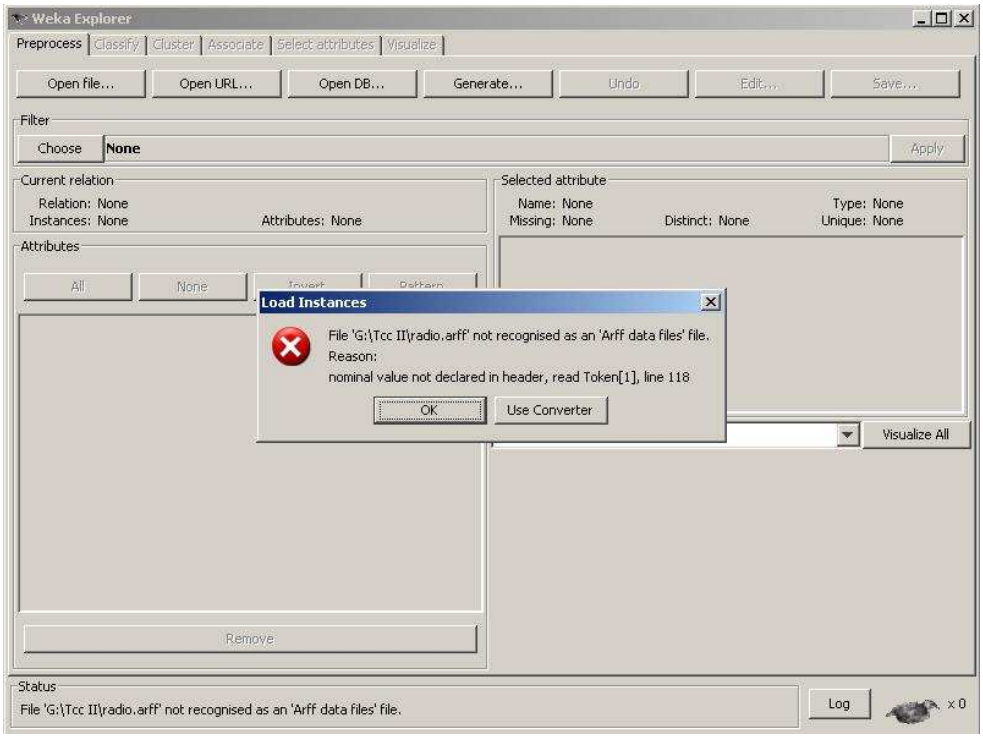

Figura 3.7: Erro ao carregar arquivo ARFF – valor não declarado no cabeçalho
Fonte: (AUTOR, 2011)

3.2.3 Mineração de dados

Com a base de dados concluída[2] e pré-processada, pode-se passar para a próxima fase, a mineração de dados. Para Martinhago (2005), a fase da mineração de dados é considerada como o núcleo do processo de descoberta de conhecimento em banco de dados. E segundo Han e Kamber (2006), é um campo multidisciplinar que inclui as seguintes áreas: banco de dados, inteligência artificial, aprendizado de máquina, redes neurais, estatísticas, reconhecimento de padrões, sistemas baseados em conhecimento, recuperação de informação, computação de alto desempenho e visualização de dados.

A mineração de dados consiste em obter informações através de uma base de dados existente, usando seus atributos para extrair informações que não são óbvias e que precisam ser trabalhadas para serem úteis na tomada de decisão. Para isto são aplicados algoritmos para identificar padrões no conjunto de dados analisado (SILVEIRA, 2003).

Portanto, a fase de mineração de dados implica na escolha do algoritmo no qual serão aplicados os dados, porém antes de efetuá-la, deve se conhecer os objetivos pretendidos para a

[2] Disponível em: <http://cid-ef508a027333b4b5.office.live.com/self.aspx/.Public/TCC/radio.arff>

solução a ser encontrada, pois é sabido que nenhum algoritmo é ótimo para todas as aplicações. Para Cabena *et al*. (1997), uma característica importante de mineração de dados, são seus algoritmos, cada um deles têm específicas entradas e saídas, também técnicas bem definidas que normalmente provêm de áreas como Aprendizado de Máquina, Reconhecimento de Padrões e Estatística. Estas técnicas, muitas vezes, podem ser associadas para se obter resultados melhores, entre as técnicas mais utilizadas podem-se citar regras de associação, classificação, clustering e regressão. Como o objetivo deste trabalho é realizar uma classificação, que consiste na predição de um valor claro, a seguir é descrita a técnica de classificação.

3.2.3.1 Classificação

Classificação consiste em construir um modelo de algum tipo que possa ser aplicado a dados não classificados visando categorizá-los em classes. Um objeto é examinado e classificado de acordo com uma classe definida, isto é, utiliza um conjunto de exemplos pré-classificados para desenvolver um modelo que pode classificar a base de dados utilizada (SILVEIRA, 2003).

De acordo com Cabena *et al*. (1997) a classificação é uma técnica de aprendizado supervisionado, ou seja, os resultados precisam ser analisados por um especialista para fazer a avaliação de relevância. A classificação gera modelos a partir de exemplos de uma base, que são chamados de conjunto de treinamento, que devem ser uma amostra dos registros que serão analisados. Como os resultados precisam ser analisados por um especialista, os modelos gerados, precisam ser de fácil compreensibilidade e entre as técnicas de classificação mais utilizadas está a árvore de decisão, que é uma representação simples do conhecimento, e um meio eficiente de construir classificadores que predizem classes baseadas nos valores de atributos de um conjunto de dados (MARTINHAGO, 2005).

Se optou pelo algoritmo J48 que é uma versão em Java do algoritmo C4.5, criado para a construção de árvores de decisão. O algoritmo C4.5 deriva do algoritmo simples que usa o esquema “ dividir-pra-conquistar”, de Mitchell, porém, estendido para considerar problemas no mundo real como, por exemplo, valores ausentes e valores numéricos (MARTINAGO, 2005).

3.2.4 Avaliação e interpretação de resultados

Concluída a fase de mineração de dados, se chega à última etapa do processo de KDD, a avaliação e interpretação de resultados. Esta fase busca descobrir se o conhecimento extraído pode ser utilizado como apoio a algum processo de tomada de decisão. Para isso, deve se avaliar os resultados a partir de métricas tais como medidas precisão, erro, compreensibilidade e interessabilidade.

A compreensibilidade de um dado conjunto de regras está relacionada com a facilidade de interação dessas regras pelo ser humano. A compreensibilidade de um modelo pode ser estimada, por exemplo, pelo numero de regras e números de condições por regra. Nesse caso, quanto menor a qualidade de regras de um dado modelo e menor o número de condições por regra, maior será a compreensibilidade das regras descobertas (FERTIG *et al*., 1999). Já a interessabilidade mede o valor de um padrão combinado validade, novidade, utilidade e simplicidade (SILBERSCHATZ & TUZHILIN, 1995).

Além de avaliar o resultado através de métricas é muito importante que o especialista e o usuário verifiquem e julguem a aplicabilidade do conhecimento extraído. Após a avaliação, se o conhecimento extraído for considerado relevante é momento então de consolidar o conhecimento gerado e incorporar este dentro dos sistemas, documentar ou então utilizar na tomada de decisões (FAYYAD *et al*., 1996a). E caso não cumpra os objetivos propostos, o processo de extração pode ser repetido ajustando os parâmetros ou melhorando o processo de escolha de dados para a obtenção de resultados melhores numa próxima iteração (REZENDE, 2005).

4 EXPERIMENTOS

Neste capítulo são apresentadas as funcionalidades da ferramenta *Weka* e ainda os experimentos e os resultados obtidos com a aplicação de técnicas de mineração de dados através do *Weka*. Partindo do objetivo de se obter parâmetros de correção para roteiros comerciais de emissoras de rádio, utilizou-se a base de dados desenvolvida neste trabalho, a partir da análise do software SisRadio módulo OPEC, apresentada no Capítulo 3.

4.1 Ferramenta de mineração de dados

O *Weka* é um software de mineração de dados, escrito inteiramente em Java, que contém um conjunto de algoritmos de aprendizado de máquina para realizar tarefas de mineração de dados. Desenvolvido pela Universidade de Waikato da Nova Zelândia, o *Waikato Environmet for Knowledge Analysis* é de domínio público e portável para qualquer sistema operacional. O software possui ferramentas de pré-processamento, classificação, regressão, *clustering*, regras de associação e visualização em duas dimensões de dados.

O software pode ser usado através de uma interface gráfica de usuário, ou por linha de comando, na qual é recomendada para realizar experimentos com grau de complexidade mais elevado, pois usa menos memória e oferece funcionalidades que não estão presentes no modo de interface gráfica de usuário, o *Weka GUI Chooser*.

Na inicialização (Figura 4.1) do *Weka GUI Chooser*, é apresentado o menu e as quatro principais aplicações do *Weka*, que são (FRANK, HOLMES, *et al.*, 2009):

Explorer: ambiente para explorar os dados com o *Weka*;

Experimenter: ambiente para realizar experimentos e testes entre sistemas de aprendizagem;

KnowledgeFlow: ambiente semelhante ao *Explorer*, porém com interface *drag-on-drop*, isto é, arrastar e largar.

Simple CLI: ambiente que possibilita a utilização de uma interface de linha de comando para sistemas operacionais que não fornecem sua própria interface de linha de comando.

Figura 4.1: Weka GUI Chooser – Tela de apresentação do software
Fonte: (AUTOR, 2011)

Para a composição deste trabalho, foi utilizada a aplicação *Explorer* e suas seções *Preprocess* e *Classify* durante a execução do software *Weka*. Ao iniciar o ambiente da aplicação, apenas a seção *Preprocess* está ativada, isso porque é necessário selecionar um arquivo, que contenha os dados a serem pré-processados, antes mesmo de explorá-los. O *Weka* suporta vários formatos de arquivos, como CSV e ARFF, que é o formato padrão do software. Para demonstrar o funcionamento das seções *Preprocess* e *Classify*, será utilizada uma base de dados associada à classificação de animais, disponível[3] na página da Universidade da Califórnia de Irvine. Esta base de dados está dividida em dois arquivos C4.5, *zoo.data*, que contém os valores de cada atributo, e *zoo.names*, com as descrições e informações da base de dados e suas instâncias. Estes arquivos devem ser combinados e transformados em um único arquivo com a extensão *.arff*, obedecendo a estrutura da Figura 4.2.

[3] UCI – MACHINE LEARNINIG REPOSITORY. Disponível em: < http://archive.ics.uci.edu/ml/datasets/Zoo>. Acesso em 20/03/2010.

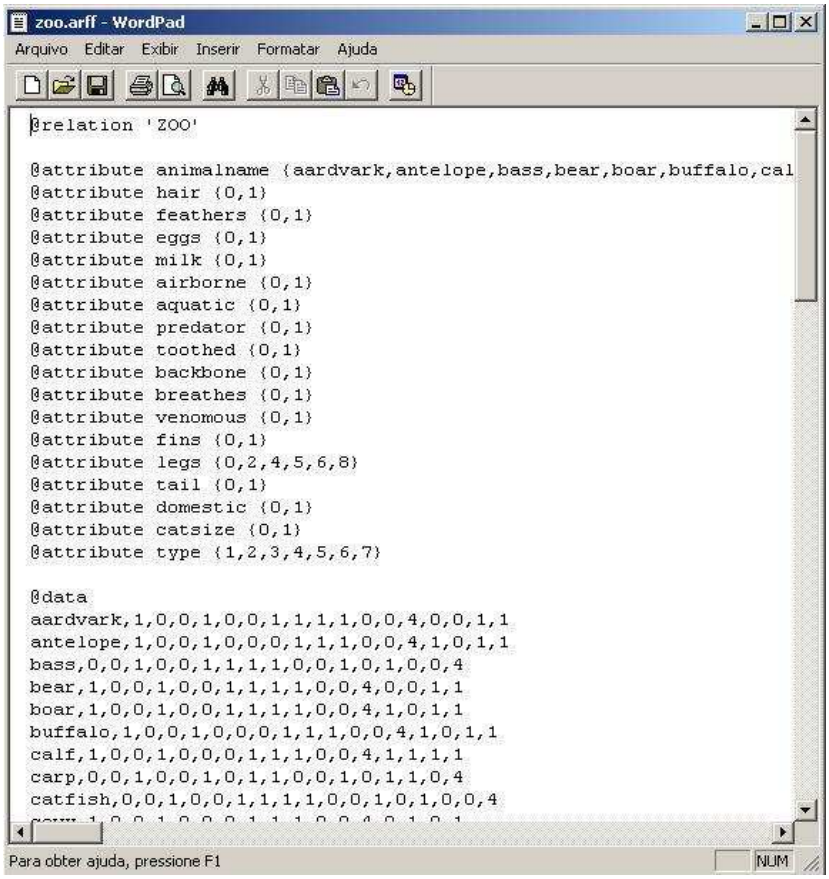

Figura 4.2: Estrutura arquivo ARFF
Fonte: (AUTOR, 2011)

O arquivo ARFF, deve conter o nome da base de dados (*@relation*), a lista de atributos (*@attribute*), que podem ser do tipo: *nominal*, *numeric*, *integer* e *real*, e os dados (*@data*), que devem ser separados por vírgula, uma linha para cada instância.

Uma vez que o arquivo é carregado na aplicação *Explorer*, seção *Preprocess*, informações como nome da base, número de instâncias, e atributos são exibidos na caixa *Current relation*. Abaixo da caixa *Current relation*, está a caixa *Attributes*, que apresenta todos os atributos relacionados, dando a possibilidade de marcar e desmarcar os atributos a partir dos botões que se apresentam no topo da caixa, *All*, *None*, *Invert* e *Pattern*, este último permite ao usuário selecionar atributos de uma base Perl 5. Na parte inferior da caixa *Attributes*, o botão *Remove*, permite excluir um ou mais atributos selecionados.

Já na caixa *Selected Attribute*, informações e estatísticas como nome do atributo (*Name*), tipo do atributo (*Type*), números em porcentagem de instâncias em que o atributo está ausente (*Missing*), número de valores distintos que o atributo assume na base de dados (*Distinct*), e número em porcentagem de instâncias que possuem um valor único para o atributo (*Unique*) são apresentadas quando algum atributo é clicado na caixa *Attributes*.

Abaixo dessas estatísticas, há uma lista que mostra mais informações sobre os valores armazenados em cada atributo, que diferem de acordo com o seu tipo. E mais abaixo, um histograma que por meio de cores apresenta relacionamentos entre atributos e classes.

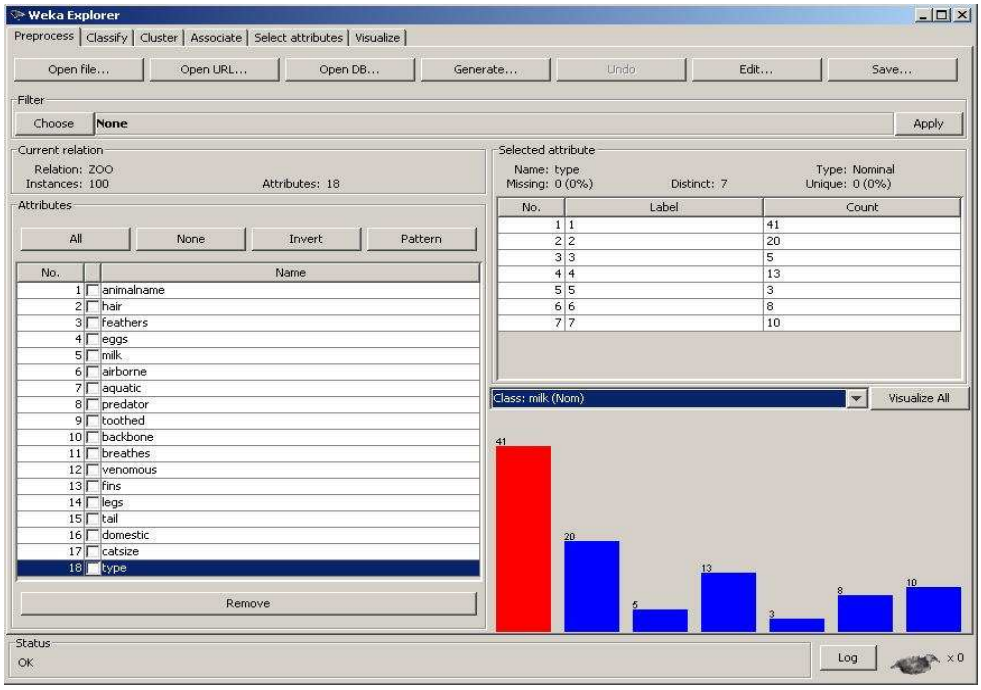

Figura 4.3: Weka Explorer – Relação entre atributo e classe
Fonte: (AUTOR, 2011)

A Figura 4.3 mostra uma relação entre o atributo type (espécie), e a classe milk (mamífero); pode se observar no histograma, que dos 100 exemplos de animais contidos na base de dados, 41 deles são mamíferos e todos pertencem ao type (espécie) 1.

Clicando no botão *Visualize All* disposto acima do histograma, o usuário pode visualizar gráficos para todos os atributos não mostrados, através de uma nova janela, como na Figura 4.4.

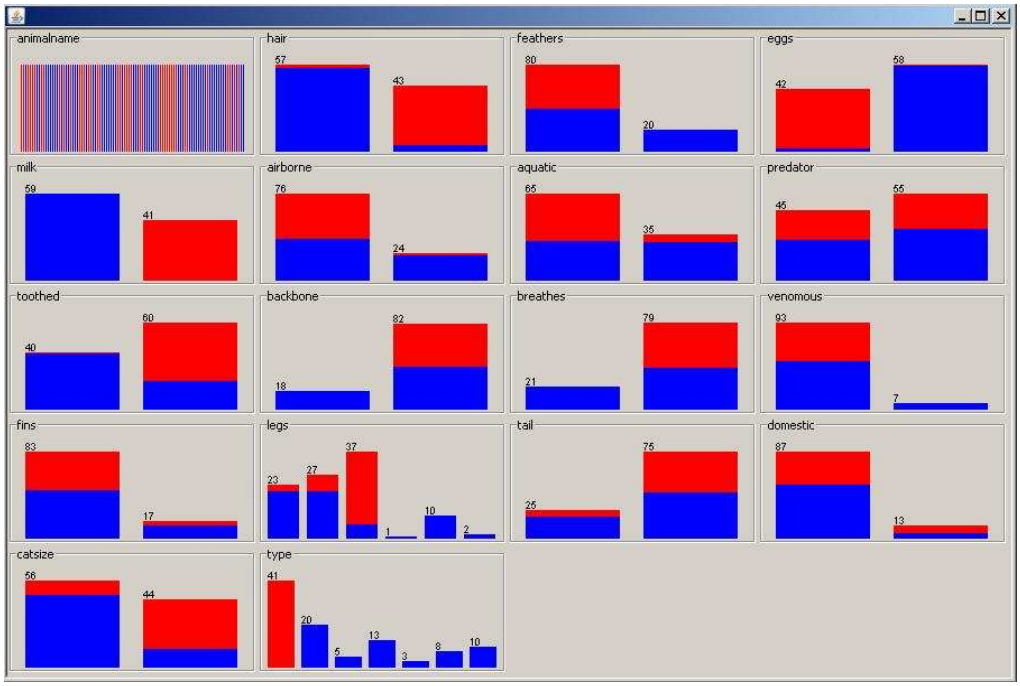

Figura 4.4: Weka Explorer – Visualização de todos os gráficos
Fonte: (AUTOR, 2011)

Ainda na seção *Preprocess*, clicando no botão *Edit*, abrirá uma nova janela, onde é possível editar os dados da relação. Para armazenar as alterações realizadas basta clicar no botão *Save*, qualquer alteração no arquivo pode ser desfeita pelo botão *Undo*.

Já no topo da seção *Classify*, encontra-se a caixa *Classifier* que permite a escolha do classificador. Abaixo da caixa *Classifier*, que permite selecionar um algoritmo para classificação, está a caixa *Test options*, que possibilita a aplicação de quatro modos de testes: *Use training set*, o usuário estabelece todo o conjunto de dados para treinamento; *Supplied test set*, o usuário carrega outro arquivo, que contenha um conjunto de dados para executar somente como teste; *Cross-valitation*, o classificador testa o conjunto de dados, através do método de validação cruzada, utilizando o número de dobras fornecido pelo usuário no campo *Folds*; e *Percentage split*, o usuário estipula a porcentagem do conjunto de dados que será usada para o treinamento e o restante é definido como teste.

Escolhido o classificador, o método de teste, e o atributo a ser usado como classe, o teste pode ser iniciada no botão *Start*. Na caixa abaixo, *Result list*, e no quadro *Classifier output*, à direita, serão apresentados os resultados do processo, como mostra a Figura 4.5.

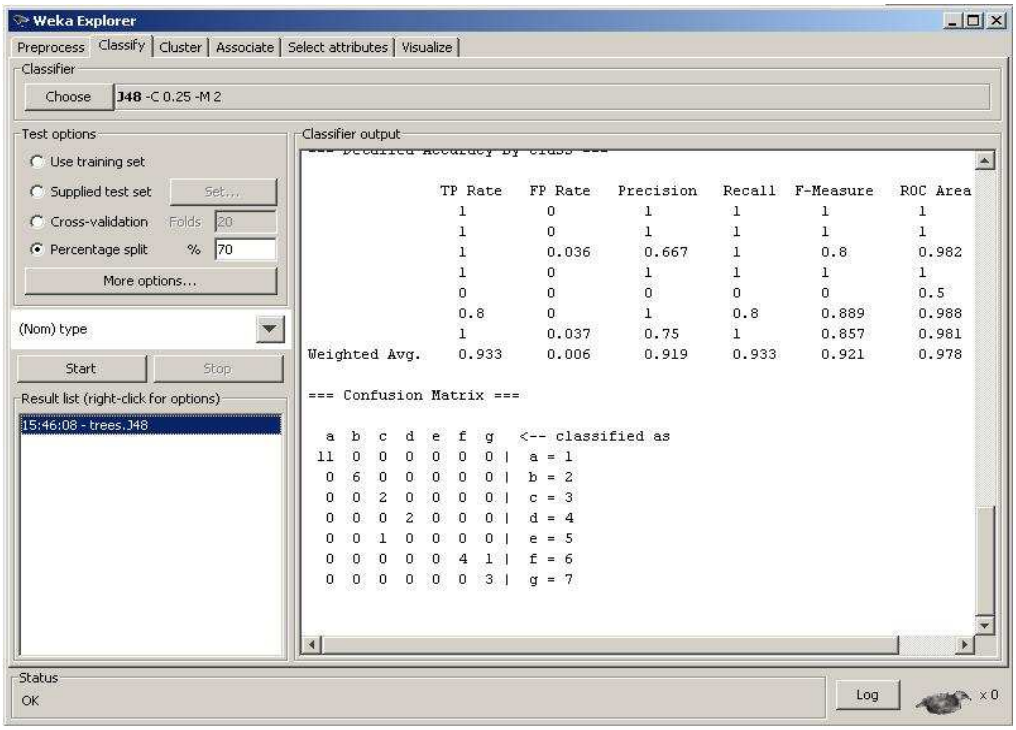

Figura 4.5: Weka Explorer – Resultados da classificação
Fonte: (AUTOR, 2011)

No quadro *Classifier output*, são apresentados os resultados do processo, estas informações são divididas em seções: *Run information*, que contém uma lista de informações como nome da relação, número de instâncias, atributos e modo de teste; *Classifier model*, é uma representação textual do modelo de classificação, que foi gerado a partir do conjunto de dados de treinamento. E ainda, uma lista de informações dos testes, onde a Matriz de Confusão (*Confusion Matrix*) pode ser visualizada, nela, mostra quantas instâncias foram atribuídas a cada classe.

E por último, clicando com o botão direito do mouse, sobre o resultado, na caixa *Result list*, o usuário pode visualizar a árvore de decisão gerada pelo classificador (Figura 4.6), e os erros da classificação; que no gráfico são representados por quadrados. Exemplo na Figura 4.7.

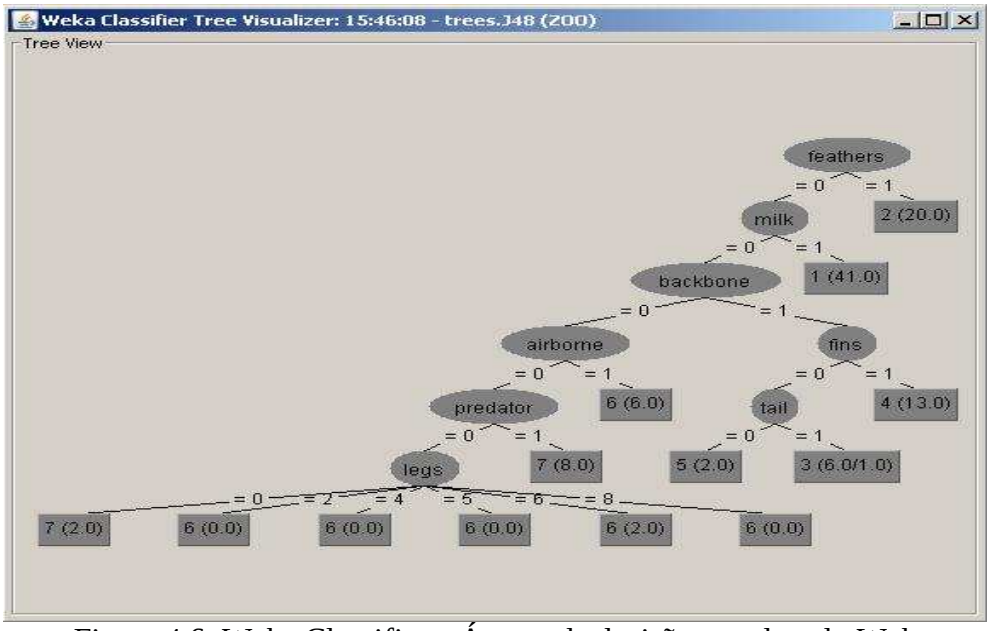

Figura 4.6: Weka Classifier – Árvore de decisão gerada pelo Weka
Fonte: (AUTOR, 2011)

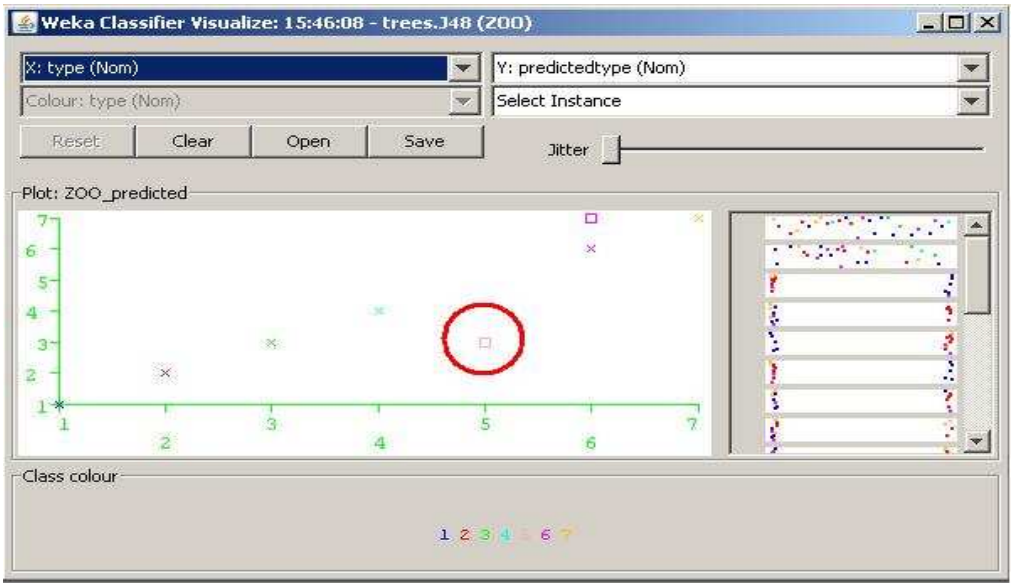

Figura 4.7: Weka Classifier – Visualização dos erros de classificação
Fonte: (AUTOR, 2011)

Clicando sobre cada quadrado, o usuário poderá visualizar as informações da instância que foi classificada de forma errada, como mostra a Figura 4.8.

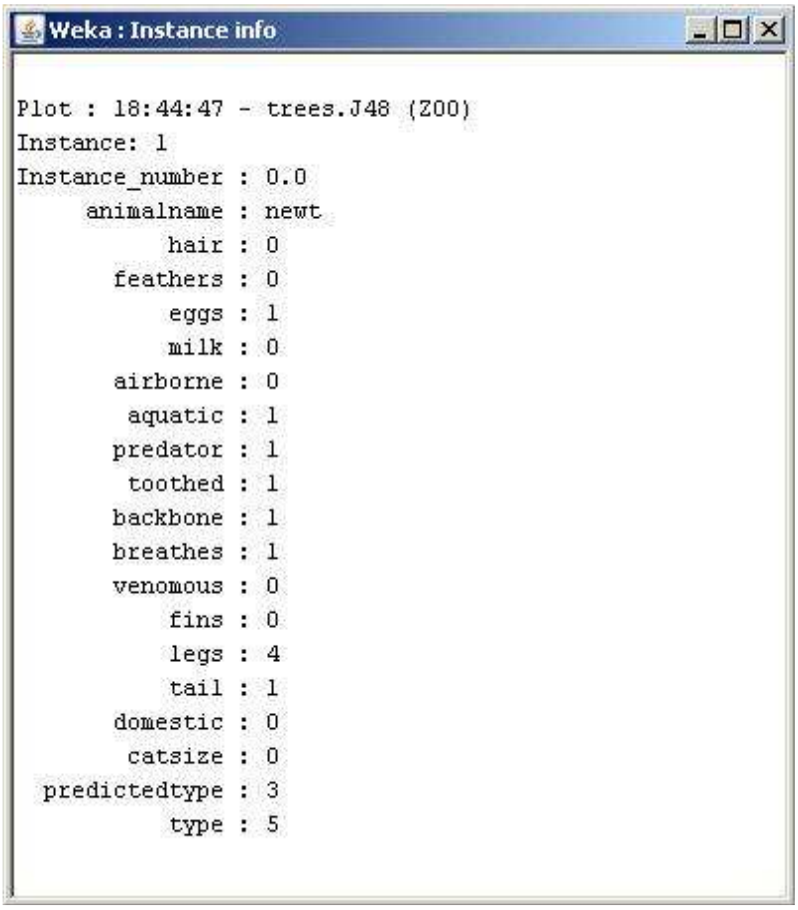

Figura 4.8: Weka – Informações da instância
Fonte: (AUTOR, 2011)

Como já foi descrito no Capítulo 2, erros de precisão no modelo de classificação podem ser causados por inúmeros fatores, como ruído de dados ou até mesmo excesso de ajuste no algoritmo de aprendizado. No *Weka*, estes ajustes podem ser feitos através da caixa *GenericObjectEditor* (Figura 4.9). Para editar as configurações do algoritmo de aprendizado, basta clicar sobre o algoritmo selecionado na caixa *Classifier*, ao lado do botão *Choose*.

Na caixa *GenericObjectEditor* do algoritmo selecionado, é possível configurar algumas parâmetros como: *binarySplits*, que indica a utilização de divisão binária para atributos nominais na construção da árvore de decisão; *confidenceFactor*, valor utilizado para poda (quando menor o valor do campo maior a incidência de poda); *debug*, se selecionado o valor *True*, são apresentadas mais informações de saída sobre o classificador; *minNumObj*, campo que defini o número mínimo de instâncias por folha a ser usado na árvore de decisão; *numFolds*, determina a quantidade de dados que será usado na técnica de poda chamada *Reduced-Error*; já o parâmetro *reducedErrorPruning*, ativa ou desativa esta técnica de poda; *saveInstanceData*, define se o conjunto de treinamento pode ser salvo para visualização; *seed*, quando ativada, durante utilização da técnica de poda *Reduced-Error*, faz com que os dados

sejam misturados para aprimorar a classificação; *subTreeRaising*, substitui um nó interno por um dos nós que estão abaixo, gerando novamente a árvore; *unpruned*, se selecionado o valor *True*, desativa o uso da poda; e *useLaplace*, conta o número de folhas excluídas utilizando o método de transformada de Laplace.

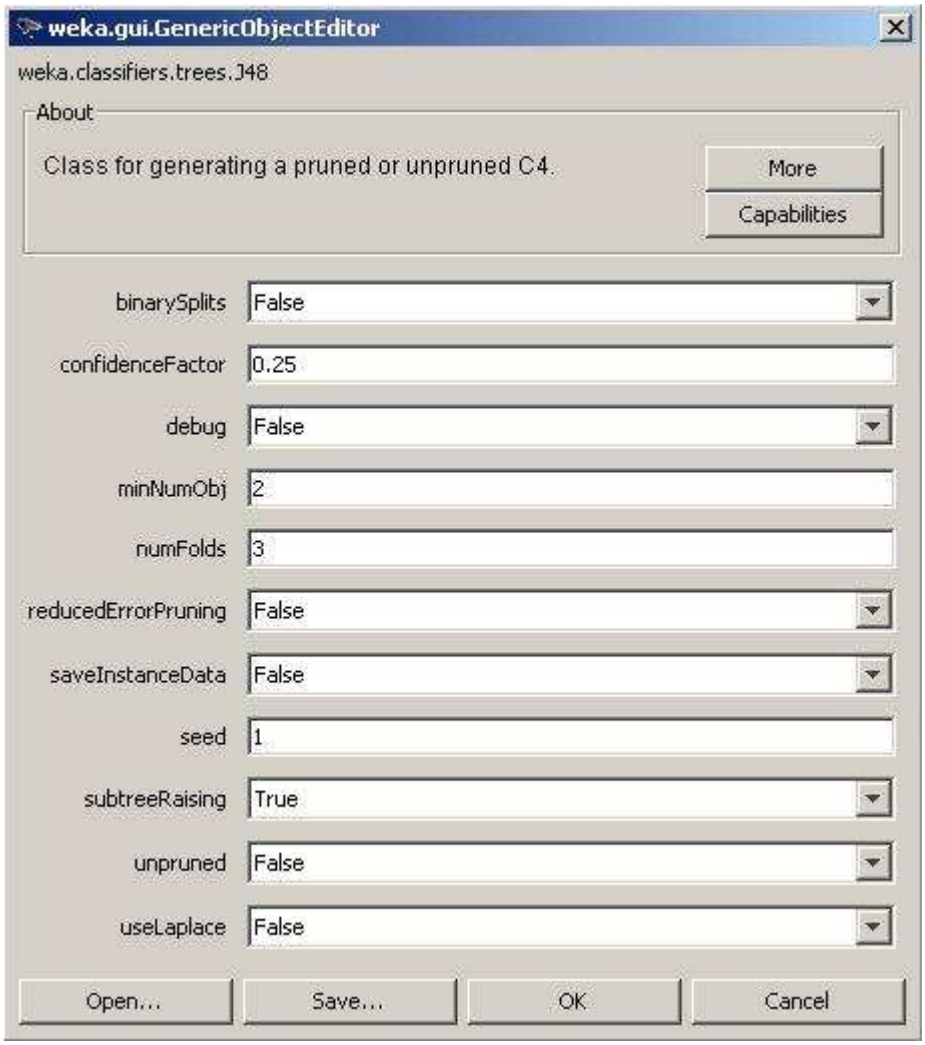

Figura 4.9: Weka – Caixa para configuração do algoritmo de aprendizado
Fonte: (AUTOR, 2011)

4.2 Experimento nº 1

Para a realização do experimento nº 1 optou-se pela utilização do algoritmo J48 e a opção de teste *Percentage split*, definida em 70%, que no software Weka representa a utilização de 70% do conjunto de exemplos para realização do treinamento e restante para os testes. O objetivo do experimento foi desenvolver parâmetros de correção para a programação comercial de emissoras de rádio afiliada, a partir de regras obtidas da predição do atributo *Realocar*. Através da metodologia *bottom-up*, a árvore gerada foi analisada, e da suas análises são extraídos padrões, regras, que representam o conhecimento para o especialista, o roteirista de intervalos comerciais. A seguir a Figura 5.1 apresenta o modelo de classificação.

```
Veiculação = AVU
|  Material_fixo_na_estrutura = 0: 0 (108.0/1.0)
|  Material_fixo_na_estrutura = 1
|  |  Duplicidade_de_ramo = 0: 4 (11.67/0.67)
|  |  Duplicidade_de_ramo = 1: 3 (8.49)
Veiculação = DET
|  Locuções_aproximadas = 0
|  |  Duplicidade_de_ramo = 0: 4 (15.14/2.14)
|  |  Duplicidade_de_ramo = 1: 3 (3.03)
|  Locuções_aproximadas = 1
|  |  Reserva_correta = 0: 1 (2.02/0.02)
|  |  Reserva_correta = 1: 3 (10.09/1.0)
Veiculação = PAT
|  Reserva_correta = 0
|  |  Locuções_aproximadas = 0
|  |  |  Duplicidade_de_ramo = 0: 2 (32.29/0.29)
|  |  |  Duplicidade_de_ramo = 1: 1 (3.03/0.03)
|  |  Locuções_aproximadas = 1
|  |  |  Duplicidade_de_ramo = 0: 1 (11.46/0.46)
|  |  |  Duplicidade_de_ramo = 1: 0 (3.03/0.03)
|  Reserva_correta = 1
|  |  Locuções_aproximadas = 0
|  |  |  Duplicidade_de_ramo = 0: 4 (44.4/0.4)
|  |  |  Duplicidade_de_ramo = 1: 3 (7.06)
|  |  Locuções_aproximadas = 1
|  |  |  Duplicidade_de_ramo = 0: 3 (32.94/0.64)
|  |  |  Duplicidade_de_ramo = 1: 2 (4.04/0.04)
Veiculação = CIT: 4 (39.36/0.36)
Veiculação = IND
|  Reserva_correta = 0: 2 (13.45/4.45)
|  Reserva_correta = 1
|  |  Locuções_aproximadas = 0: 4 (25.91/0.91)
|  |  Locuções_aproximadas = 1: 3 (2.02)
Veiculação = VEC: 4 (176.59/1.59)
```

Figura 5.1: Árvore de decisão experimento nº 1
Fonte: (AUTOR, 2011)

Analisando a árvore de decisão pode-se observar que os atributos que aparecem nos primeiros nós da árvore tem significado importante para a sua similaridade, o atributo *Veiculação* é determinante para indicar com precisão se a mídia dever ser realocada ou não. Outro ponto observado no experimento é a relação entre simplicidade e precisão do

conhecimento extraído. O modelo de classificação gerado pelo software *Weka*, na forma de árvore de decisão, apresentou uma taxa de acerto de 95%, isto é, das 166 instâncias contidas no conjunto de teste 158 foram classificadas corretamente. Já a sua simplicidade é medida através do tamanho da árvore de decisão gerada, como se pode notar o tamanho total da árvore, contando nós e folhas, é de 35.

Observou-se também outra questão importante, as regras extraídas utilizando a hierarquia de declarações do tipo “se-então” (Tabela 1.7) são muito similares ao processo intuitivo do roteirista para a verificação dos roteiros, o que quer se demonstrar com essa analise é que as regras extraídas condizem com a realidade.

Tabela 1.7: Regras para realocação imediata e não realocação de mídias

R	Regras
R_1	**Se** veiculação = AVU **e** Material_fixo_na_estrutura = 0 **então** Realocar = 0
R_2	**Se** Veiculação = PAT **e** Reserva_correta = 0 **e** Locuções_aproximadas = 1 **e** Duplicidade_de_ramo = 1 **então** Realocar = 0
R_3	**Se** Veiculação = AVU **e** Material_fixo_na_estrutura = 1 **e** Duplicidade de ramo = 0 **então** Realocar = 4
R_4	**Se** Veiculação = DET **e** Locuções_aproximadas = 0 **e** Duplicidade_de_ramo = 0 **então** Realocar = 4
R_5	**Se** Veiculação = PAT **e** Reserva_correta = 1 **e** Locuções_aproximadas = 0 **e** Duplicidade_de_ramo = 0 **então** Realocar = 4
R_6	**Se** Veiculação = CIT **então** Realocar = 4
R_7	**Se** Veiculação = IND **e** Reserva_correta = 1 **e** Locuções_aproximadas = 0 **então** Realocar = 4
R_8	**Se** Veiculação = VEC **então** Realocar = 4

Fonte: (AUTOR, 2011)

4.3 Experimento nº 2

O segundo experimento, igualmente ao primeiro, também se optou pelo algoritmo J48 e a opção de teste *Percentage split*, definida em 70%, embora foram realizados dois testes, um com poda e outro sem poda (*unpruned*) o objetivo foi identificar o melhor posicionamento das mídias dentro do intervalo comercial. Para isso, o roteirista de intervalos comerciais reavaliou o roteiro gerado, e indicou uma nova posição para cada mídia dentro do bloco, através do atributo *Bloco_corrigido*. Ao gerar os modelos de classificação na ferramenta *Weka*, para se predizer o atributo *Bloco_corrigido*, a taxa de acerto dos modelos apresentou um valor elevado de 80% e 77% respectivamente, porém ao se comparar os resultados em relação ao conhecimento do especialista se verificou incoerência e excesso de generalização (Figura 5.2) por parte do primeiro teste.

```
Posição_no_bloco = 23: 23 (15.0/1.0)
Posição_no_bloco = 24: 24 (12.0/3.0)
Posição_no_bloco = 25: 25 (16.0/2.0)
Posição_no_bloco = 26: 26 (12.0/2.0)
Posição_no_bloco = 27: 27 (7.0)
Posição_no_bloco = 28
|   Tempo_do_bloco <= 321352: 28 (4.0)
|   Tempo_do_bloco > 321352: 25 (2.0/1.0)
Posição_no_bloco = 29
|   Tempo_da_midia <= 30589: 29 (4.0)
|   Tempo_da_midia > 30589: 11 (2.0/1.0)
Posição_no_bloco = 30: 30 (6.0/2.0)
Posição_no_bloco = 31: 31 (6.0/1.0)
Posição_no_bloco = 32: 32 (9.0/3.0)
Posição_no_bloco = 33: 33 (9.0/3.0)
Posição_no_bloco = 34: 34 (6.0)
Posição_no_bloco = 35: 35 (7.0/2.0)
Posição_no_bloco = 36: 36 (4.0/1.0)
```

Figura 5.2: Trecho árvore de decisão experimento nº 2
Fonte: (AUTOR, 2011)

4.4 Considerações Finais

Embora não tenha se extraído nenhum conhecimento novo, os resultados obtidos no experimento nº 1 são de grande valia, pois confirmam que as etapas anteriores ao processo de extração de conhecimento, como coleta, pré-processamento e transformação dos dados foram desenvolvidas e tratadas com qualidade. E devido a árvore decisão gerada no experimento ser de fácil compreensibilidade ao olhar humano, as suas regras extraídas são de grande importância e podem ser usadas com uma ferramenta de auxílio na tomada de decisão para a criação de roteiros comerciais de rádio, pois representam e confirmam as operações de verificação realizadas pelo roteirista de forma pontual.

O experimento nº 2, não cumpriu com os objetivos desejados, não foi possível identificar padrões coerentes em relação ao conhecimento do especialista e foi descartado.

CONCLUSÕES

Este trabalho teve por objetivo contribuir para a busca de conhecimento no campo da programação comercial de emissoras de rádio afiliadas e para isso apresentou experimentos de mineração de dados e aprendizado de máquina supervisionado, utilizando o software *Weka*.

Devido à inexistência de trabalhos relacionados, a conclusão foi fundamentada a partir das experiências do roteirista comercial de emissora de rádio e percebe-se que embora não tenha se extraído nenhum conhecimento novo, os resultados obtidos nos experimentos de mineração e aprendizado de máquina supervisionado são de grande importância, pois direcionam para um caminho com excelente potencial a ser seguindo, ainda que pouco empregado na radiodifusão. A partir das regras extraídas pela ferramenta *Weka*, se pode observar uma similaridade com processo intuitivo do roteirista para a verificação dos roteiros, o que confirma a funcionalidade das regras extraídas para auxiliar a tomada de decisão no campo da programação comercial de rádio.

Apesar de não ser o objetivo central do trabalho é importante evidenciar uma grande contribuição, o desenvolvimento da base de dados utilizada para a realização dos testes na ferramenta de mineração *Weka*.

Para trabalhos futuros, sugere-se a realização de novos experimentos de mineração e aprendizado de máquina supervisionado relacionados ao campo de programação comercial de rádio, embora, ampliando o processo de coleta dos dados e correlacionando-os ao setor financeiro das emissoras, com o objetivo de buscar padrões para se atingir anunciantes em potencial.

REFERÊNCIAS BIBLIOGRÁFICAS

BATISTA, G. E. P. A. **Pré-processamento de Dados em Aprendizado de Máquina Supervisionado**. Instituto de Ciências Matemáticas e de Computação, ICMC. São Carlos, SP. 2003.

CABENA, P. et al. **Discovering Data Mining from Concept to Implementation.** Upper Saddle River, New Jersey: Prentice Hall, 1997.

CARVALHO, J. V.; SAMPAIO, M. C.; MONGIOVI, G. **Utilizaçao de Técnicas de Data Mining para o Reconhecimento de Carcteres Manuscritos**. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, 14., 1999, Florianópolis. Anais. Florianópolis, 1999. p. 235-249.

HAN, J.; KAMBER, M. **Data mining: concepts and techniques.** 2ª Edição. São Francisco, CA: Morgan Kaufmann, 2006.

FAYYAD, U. et al. **From data mining to knowledge discovery: an overview.** In: Advances in Knowledge Discovery and Data Mining. 1996a.

_____. **Knowledge Discovery and Data Mining: Towards a Unifying Framework.** In: Second International Conference on Knowledge Discovery and Data Mining (KDD-96). 1996b.

FRANK, E. et al. **The WEKA Data Mining Software: An Update**. Departamento de Ciência da Computação, Universidade de Waikato. Hamilton, Nova Zelândia. 2009.

FERTIG, C. S. et al. **A fuzzy beamsearch rule induction algotihm**. In: Proceedings of the Third European Conference (PKDD-99) Lecture Notes in Artificial Intelligence 1704, p. 341 – 347. 1999.

LIU, B. **Web data mining:** exploring hyperlinks, contents, and usage data. Chicago, IL: Springer, 2007.

LUFT, C. P. **Pequeno Dicionário da Língua Portuguesa**. São Paulo, SP: Editora Scipione, 1988.

MARTINHAGO, S. **Descoberta de conhecimento sobre o processo seletivo da UFPR**. Departamento de Matemática, Setor de Ciências Exatas e Departamento de Construção Civil, Setor de Tecnologia da Universidade Federal do Paraná. Curitiba, PR. 2005.

MICHALSKI, R. S.; BRATKO, I.; KUBAT, M. **Machine Learninig and Data Mining**. John Wiley. 1998.

MITCHELL, T. M. **Machine Learning**. [S.l.]: McGraw-Hill, 1997.

PATRI, R. C.; BARANAUSKA, J. A.; MONARD, M. C. **Padronização da Sintaxe e Informações sobre Regras Induzidas a partir de Algoritmos de Aprendizado de Máquina Simbólico**. Instituto de Ciências Matemáticas e Computação, Laboratório de Inteligência Computacional, Universidade de São Paulo. São Paulo, SP. 2003.

REZENDE, S. O. **Mineração de Dados**. In: ENCONTRO NACIONAL DE INTELIGÊNCIA ARTIFICIAL, 5., 2005, São Leopoldo. Anais. São Leopoldo: Unisinos, 2005.

_____. **Sistemas inteligentes:** fundamentos e aplicações. 1ª Edição. Barueri, SP: Editora Monole Ltda., 2003.

RUSSEL, S.; NORVING, P. **Inteligência Artificial**. 2ª Edição. São Paulo, SP: Editora Campus, 2004.

SILBERSCHATZ, A.; TUZHILIN, A. **On subjective measures of interestingness in knowledge discovery.** Proceeding of First International Conference on Knowledge Discovery and Data Mining 1, 275-281.1995.

SILVEIRA, R. F. **Mineração de Dados Aplicada a Definição de Índices em Sitemas de Raciocínio Baseado em Casos.** Curso de Pós-Graduação em Web Sistemas de Informação, Universidade Federal do Rio Grande do Sul. Porto Alegre, RS. 2003.

SIS. **Performática Computação**, 2011. Disponivel em: <http://www.performatica.com.br>. Acesso em: 23 Maio 2011.

WIEDERHOLD, G. On the barries and Future of knowledge discovery. In: FAYYAD et al. **Advances in knowledge discovery and data mining.** Cambridge, Mass:AAAI/MIT Press, 1996. p. VII-XI.