CENTRO UNIVERSITÁRIO FEEVALE

SIDINEI PEREIRA GONCHOROSKI

UTILIZAÇÃO DE TÉCNICAS DE KDD EM UM CALL CENTER ATIVO

Novo Hamburgo, Dezembro de 2007.

SIDINEI PEREIRA GONCHOROSKI

UTILIZAÇÃO DE TÉCNICAS DE KDD EM UM CALL CENTER ATIVO

Centro Universitário Feevale
Instituto de Ciências Exatas e Tecnológicas
Curso de Ciência da Computação
Trabalho de Conclusão de Curso

Professor Orientador: Juliano Varella de Carvalho

Novo Hamburgo, Dezembro de 2007.

AGRADECIMENTOS

Gostaria de agradecer a todos os que, de alguma maneira, contribuíram para a realização desse trabalho de conclusão, em especial:

Aos amigos pelo companheirismo, meus pais pelo apoio incondicional, minha noiva que eu amo muito e meu orientador Juliano Varella de Carvalho pelo incentivo e ensinamentos.

RESUMO

Os *Call Centers* Ativos possuem estrutura para realizar campanhas de *telemarketing* ou vendas através do telefone e são boas alternativas para que as empresas aumentem ou mantenham seu número de clientes. No entanto, as empresas do ramo de atendimento podem não explorar e aproveitar da melhor maneira o conhecimento nas bases de dados. Podem ainda não identificar o melhor perfil de vendas ou ter dificuldade em criar uma estratégia para aumentar o desempenho da produção. Através do processo de descoberta de conhecimento em banco de dados (KDD) é possível identificar regras e padrões válidos aplicando as técnicas e algoritmos de mineração de dados. Sendo assim, esse trabalho apresenta o problema detalhado do *Call Center*, KDD com suas técnicas e algoritmos, dando ênfase principalmente às árvores de classificação e o software Weka. Foi realizado um estudo de caso, quais técnicas são utilizadas, como foi realizada a validação dos resultados e o software desenvolvido para aplicar a mineração de dados. Por fim, a apresentação dos resultados da comparação do desempenho com *Data Mining* e da aplicação no *Call Center* em tempo real.

Palavras-chave: Descoberta de Conhecimento. Técnicas de Mineração de Dados. Weka. Algoritmos de Mineração de Dados. *Call Center* Ativo.

ABSTRACT

The Outbound Call Center companies have structures to carry out sales or telemarketing campaigns through telephone, which are, good alternatives for companies to increase or to keep the number of costumers. However, these companies may not explore and use their databases records in the best way. Also, may not identify the best sales profiles or have difficulty in creating a strategy to increase the production performance. Through the discovery process of knowledge in databases (KDD), it is possible to identify rules and valid patterns applying the techniques and algorithms of data mining. Therefore, this assignment presents the Call Center problem in details, KDD techniques and algorithms, giving emphasis to the classification trees and Weka software. It was done a study case, what techniques are used, how it was the results validation and software developed to apply Data Mining. Finally, the presentation of performance results comparison with Data Mining and real time application at Call Center.

Key words: Knowledge Discovery. Data Mining techniques. Weka. Algorithms of Data Mining. Active Call Center.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE GRÁFICOS

LISTA DE ABREVIATURAS E SIGLAS

AED	Análise Exploratória de Dados
API	*Application Programming Interface*
BD	Banco de Dados
DM	*Data Mining*
ID3	*Iterative Dichotomizer 3*
JVM	*Java Virtual Machine*
JDK	*J2SE Development Kit*
JRE	*J2SE Runtime Environment*
J2SE	*Java 2 Standard Edition*
KDD	*Knowledge Database Discovery*
MD	Mineração de Dados
MER	Modelo Entidade Relacionamento
MIS	*Management Information Systems*
PA	Posição de Atendimento
PVI	Problema com Valores Iniciais
SAC	Serviço de Atendimento ao Consumidor
SGBD	Sistemas Gerenciadores de Banco de Dados
SQL	*Structured Query Language*
XML	*Extensible Markup Language*

SUMÁRIO

INTRODUÇÃO

O mundo dos negócios sempre explorou os meios de comunicação para aumentar seus índices de vendas e divulgar seus produtos a um público maior. Dentre os meios de comunicação existentes, o telefone continua sendo uma opção para se comunicar com as pessoas. As empresas cada vez mais preocupadas em acompanhar o ritmo constante da concorrência e dos desafios do mercado utilizam o telefone para comercializar seus produtos e aumentar seu público. Um *Call Center* Ativo é um tipo de empresa dotada de toda a estrutura física, humana e tecnológica, capaz de realizar atendimentos através do telefone e promover campanhas de *Telemarketing*[1]. É através dos *Call Centers* Ativos que as empresas encontram as soluções mais simples e acessíveis de atingir seus objetivos e explorar um mercado mais abrangente.

Vários setores da sociedade passaram, a partir do crescimento da computação e dos bancos de dados (BD), a armazenar suas operações e produções. Estes dados formaram grandes bases que não possuem um tratamento específico e que não disponibilizam aos profissionais, muitas vezes, informações e conhecimento que possam ser utilizados em seu trabalho de modo eficiente. Através desses dados as empresas puderam criar grandes listas de *prospects*[2], chamadas de *mailing*. O trabalho do *Call Center* Ativo entra em cena no momento que essas empresas decidem realizar uma campanha de *Telemarketing* para recuperar clientes, aumentar vendas ou adquirir novos clientes.

Ao acompanhar as empresas do ramo é possível notar que existem certas limitações e dificuldades de explorar e aproveitar o conhecimento nas bases de dados geradas e atualizadas dentro das empresas de atendimento. Os operadores realizam os contatos corrigindo os dados dos clientes e preenchendo as propostas com os dados ainda não conhecidos.

---

[1] "A utilização planejada de recursos de telecomunicações e informática como forma de se obter lucro direto ou indireto, através da satisfação do mercado consumidor de qualquer bem ou serviço." (DANTAS, 1994, p. 47).
[2] Pessoa que é alvo do atendimento e candidata a adquirir o produto ou serviço oferecido.

As estratégias geradas a partir da experiência dos responsáveis podem falhar, neste caso é preciso realizar uma nova análise até que se descubra, por exemplo, que uma característica não considerada importante antes é o diferencial para o sucesso da venda em questão. Quando não existem indicações sobre quais são as características que classificam um cliente como potencial, realizar a análise pode acarretar perda de tempo e mais demora a se realizar uma venda, já que os clientes serão atendidos aleatoriamente.

A descoberta de conhecimento em banco de dados, do termo em inglês *Knowledge Database Discovery* (KDD) representa, "O processo não-trivial de identificar válidos, novos, potencialmente utilizáveis e por fim, padrões compreensíveis dentro dos dados." (TRADUÇÃO NOSSA) (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996, p. 6). Estas características significam:

Dados: Conjunto de informações que são armazenados em Sistemas Gerenciadores de Banco de Dados (SGBD), em vários registros, com os mesmos atributos e que representam um tipo de coleção. Ex: Todas as vendas de um produto.

Padrão: Conforme (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996), padrão é o grupo de itens que tem maior incidência em um conjunto de dados. Por exemplo, analisando um grupo de pessoas para descobrir o tipo de cliente que adquire capitalizações, tendo posse dos valores de rendimento e da quantidade de crediários que cada um possui, chega-se a conclusão que a segunda não influencia na aquisição e somente os que têm rendimentos superiores a certo limiar acabam as adquirindo, independente dos crediários.

Processo: Se trata do processo envolvido na descoberta do conhecimento, que passa pela preparação dos dados, garimpagem e análise. As etapas do processo de KDD, de acordo com (FELDENS, apud CARVALHO; SAMPAIO; MONGIOVI, 1999), são: seleção, pré-processamento, transformação, garimpagem, análise e assimilação.

Válido*:* São aqueles padrões considerados válidos e interessantes ao objetivo traçado.

Novo: Representa todo conhecimento adquirido e que não estava previsto ou não poderia ser deduzido através de hipótese. Um padrão gerado por hipótese não é considerado novo já que poderia ser comprovado através, por exemplo, de estatística. Um destes exemplos é a Análise Exploratória de Dados (AED) do termo em inglês *Exploratory data analysis*, que é uma técnica da área de estatística lançada por (TUKEY, 1977) que visa obter o máximo de informações ocultas entre os dados (HOAGLIN; MOSTELLER; TUKEY, 1992).

Potencialmente Utilizável: Alguns dos padrões encontrados podem acabar não sendo úteis. Para que a descoberta de conhecimento seja relevante, é preciso que o resultado não represente algo totalmente sem sentido para o negócio. Padrões que sejam muito amplos ou que tenham pouca variação em relação a outros acabam não tendo muita utilidade.

Compreensível: É poder criar padrões que possam ser entendidos pelos seres humanos e acrescentem conhecimento útil para a tomada de decisões.

A tarefa de descobrir conhecimento não é simples. Os dados recolhidos e armazenados não são preparados de forma que a qualquer momento sejam analisados para que mostrem ao usuário os relacionamentos e padrões. Os dados acabam vindo de várias fontes e necessitando de um tratamento, um pré-processamento para se definir informações importantes e corrigir possíveis imperfeições, já que tiveram origem em outros locais não sendo fruto de técnicas de KDD, conforme WIEDERHOLD (1996). Estes dados podem não estar completos.

A mineração de dados é uma área interdisciplinar que agrupa estatística, BD e inteligência artificial munida de vários algoritmos (FREITAS, 1998) e pode ser definida como:

> O termo *data mining* é normalmente utilizado pela comunidade de estatísticos, analistas de dados e os MIS (*Management Information Systems*). [...] a visão de que KDD é todo processo de descoberta de conhecimento útil em dados enquanto *data mining* se refere à aplicação de algoritmos para extração de padrões em dados sem os passos adicionais do processo de KDD. (TRADUÇÃO NOSSA) (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996, p. 3-4).

A KDD utiliza os algoritmos e técnicas de Mineração de Dados para adquirir informações importantes em bases que não estão visíveis ou identificáveis pelos usuários e que podem ser úteis na tomada de decisão. KDD também significa extrair informações de grandes bases de dados sem possuir hipóteses previamente criadas, conforme (CABENA et al., 1997). Através da Mineração de dados e seus algoritmos é possível descobrir regras de associação, classificação e *Clustering* (FAYYAD et al., 1996) (FREITAS, 1998). São exemplos de algoritmos para mineração de dados: *LargeKItemSets* (AGRAWAL; IMIELINSKI; SWAMI, 1993), *Apriori* (AGRAWAL; SRIKANT, 1994), *AprioriTid* (AGRAWAL et al., 1993), *Partition* (SAVASERE; OMIECINSKY; NAVATHE, 1995) e *Multiple Level* (ML-T2L1) (SRIKANT; AGRAWAL, 1996).

Com base nos conceitos citados, a motivação do trabalho foi aplicar KDD e desenvolver uma ferramenta que facilite o tratamento e pré-processamento dos dados, que utilizasse a ferramenta Weka[3], *open source e freeware* para processar a base de dados e que voltado para um *Call Center* ativo, consiga descobrir padrões e regras que pudessem auxiliar na seleção dos *prospects* das listas de *mailing,* a fim de ter um índice de vendas melhor, com um número menor de tentativas com os clientes. Desta maneira, espera-se melhorar o desempenho e aproveitar eficientemente os recursos e informações disponíveis, mas não utilizadas até o momento, por não existir uma ferramenta especializada para tal.

A motivação acadêmico-científica do trabalho é estudar a fundo KDD, suas técnicas e a ferramenta Weka. Analisar o problema no *Call Center*, testar os algoritmos utilizados e árvores geradas servem para planejar e interpretar os resultados, a fim de comprovar a eficiência de KDD e a aplicabilidade no negócio de *telemarketing*. A criação do software contribui com a solução de fácil utilização para a aplicação de descoberta de conhecimento nas bases de dados. A motivação pessoal e profissional para a realização do trabalho é desenvolver uma solução de KDD para *Call Center* que possa ter papel fundamental na criação de estratégias na empresa, com base na mineração dos registros das transações. No futuro deseja-se criar um produto direcionado para este tipo de segmento.

No capítulo inicial são tratados detalhes de como funciona um *Call Center*, dos produtos, as ferramentas disponíveis e os problemas. O próximo capítulo traz o referencial teórico sobre KDD. Em seguida, é apresentada a ferramenta Weka e alguns algoritmos das técnicas: associação, classificação e *clustering*. No capítulo 4, o *software* desenvolvido para facilitar o tratamento dos dados e apresentação dos resultados é exibido e são detalhadas suas principais características e funções. Nesse mesmo capítulo, são apresentados os detalhes da estrutura dos dados, a técnica utilizada e sua validação. Para a apresentação dos testes, o capítulo 5 trata dos testes de comparação dos resultados sem mineração com os resultados previstos se fossem utilizados os resultados de mineração. Após esse item, os resultados dos dois testes da aplicação de *Data Mining* em tempo real estão disponíveis. A conclusão e os trabalhos futuros propostos estão no capítulo final do trabalho.

---

[3] WEKA 3: Data Mining Software in Java. Nova Zelândia. Universidade de Waikato, 2007. Apresenta todas as características do projeto e do software Weka. Disponível em: <http://www.cs.waikato.ac.nz/ml/weka/index.html>. Acesso em: 28 mar. 2007.

1 CALL CENTER ATIVO

Desde o momento da invenção do telefone, por volta da década de 1870, passou a ser um meio de comunicação para aproximar as pessoas, mesmo que esta invenção tenha demorado a romper barreiras e se popularizar. Hoje em dia, uma das formas mais eficientes de se comunicar com as pessoas é utilizando o telefone, móvel ou então residencial.

O foco deste estudo é a atividade dos *Call Center* Ativos que surgiram para explorar essas facilidades na comunicação e localização das pessoas. Os também chamados *Contact Centers* são empresas de atendimento que possuem tecnologia e estrutura telefônica capaz de realizar atendimentos ou campanhas de *telemarketing*. Embora não se tenha registros precisos, em meados da década de 80 é que estes acabaram se espalhando pelo Brasil e hoje já existem muitas empresas especializadas nessa área e que buscam personalizar o relacionamento com seus clientes. Algumas das empresas de *Call Center* do país são Atendebem[4], Meta[5], Atento[6], Ask![7].

Um *Call Center* Ativo é aquele responsável por realizar contatos com *Prospects*[8], ao contrário de Serviço de Atendimento ao Consumidor (SAC) ou central de relacionamento com o cliente que muitas empresas possuem e que é responsável por receber os contatos das pessoas interessadas em produtos ou informações.

Os funcionários destas empresas são chamados de atendentes e cada um possui para realizar suas funções, uma posição de atendimento (PA), que é uma mesa com algum tipo de isolamento lateral (pequenas paredes com cerca de 60 cm) para diminuir o ruído ao seu redor, um computador e um telefone.

---

[4] http://www.atendebem.com.br
[5] http://www.metatmkt.com.br
[6] http://www.atento.com.br
[7] http://www.askcallcenter.com.br
[8] Pessoa alvo do atendimento e candidata a adquirir o produto ou serviço oferecido.

Quanto à estrutura física e de tecnologia das empresas deste segmento, os terminais que cada operador utiliza são ligados em rede para possibilitar a utilização de *softwares* que são responsáveis por disponibilizar consultas e permitir que o atendente faça o registro de seus contatos e de suas vendas. Em relação à telefonia algumas soluções também são utilizadas como gravadores, monitores ou gerenciadores de ligações para controlar todo o fluxo das ligações da empresa.

Para coordenar todos os atendentes, estas empresas mantêm uma estrutura que é composta por supervisores que tem ligação direta com os operadores, coordenadores que são responsáveis por passar as orientações necessárias aos supervisores e ainda acima destes, os gerentes que além de ter contato direto com o contratante, tem responsabilidade sobre os outros níveis desta empresa.

Ainda sobre as pessoas envolvidas nesse trabalho, existem outros setores que não possuem uma denominação específica e que tem como responsabilidade tratar de relatórios, importações e gerenciamento de nomes que os operadores entrarão em contato.

O ambiente de uma empresa de *telemarketing* é preparado para que o operador tenha o máximo de motivação e apoio dos colegas e supervisores para que possa atender sempre melhor um *prospect*. Além de ser um ambiente de trabalho bastante motivacional, os operadores têm que manter uma postura fiel ao contratante e precisam seguir *scripts* de atendimento que são criados e remodelados para que seu atendimento seja o melhor possível e que sua meta seja alcançada através das vendas.

Os atendentes são peças fundamentais em uma empresa de *telemarketing* e para otimizar seus resultados, nos dois últimos anos, conforme ATENDEBEM (2007), as empresas têm investido em treinamento e qualificação para seus operadores. Diversos treinamentos são exemplos de qualificação oferecida ao atendente: reversão de objeções levantadas pelo *prospect*, entonação de voz, aperfeiçoamento e adaptação de *script* e aprofundamento de produto.

Outra característica envolvida em uma empresa de *telemarketing* é a questão das metas de produção. Estas são estipuladas pelas empresas que contratam o serviço e toda a empresa, desde a área de tecnologia até a própria operação – como é chamado o setor onde os operadores e supervisores trabalham – são envolvidos e contribuem para que as metas sejam alcançadas.

A área de tecnologia de um *Call Center* é talvez uma das mais importantes dentro do quadro da empresa. Ela é responsável por manter toda a estrutura de *hardware* e *software* disponível, para que não haja desperdício de tempo, e acabe por contribuir diretamente no sucesso das operações realizadas. Para ilustrar a importância desta área pode-se considerar que se por algum motivo a parte de telefonia for interrompida ou então se a rede não estiver disponível, toda a produção da empresa é comprometida e certamente vai impactar nos resultados.

1.1 Contratante

A contratante é a empresa, órgão ou pessoa que utiliza o serviço de um *Call Center* Ativo para aumentar seu volume de vendas e, por conseguinte, seu público alvo.

Na maioria dos contratos entre a empresa contratante e o *Call Center*, a primeira normalmente é responsável por fornecer as listas com o nome dos clientes e seus dados, chamadas de *mailing*, para quem o *telemarketing* Ativo deverá entrar em contato. Existem outros casos em que a criação dessa base de clientes se dá por indicação de *prospects*, onde cada um ao ser abordado indica outra pessoa para receber o contato.

A empresa contratada fornece toda a estrutura e pessoal para que a contratante possa realizar os devidos treinamentos de produto aos operadores, que a partir do momento do início da campanha. Nesse momento a empresa de *telemarketing* e seus funcionários devem estar preparados para trabalhar com os mais diversos tipos de produtos e serviços. Tanto o setor de tecnologia quanto a operação têm que entender o negócio e produto que são repassados em detalhes. A constante adaptação é uma característica desse tipo de empresa.

Outros deveres da empresa contratante são definir os tipos de produtos e as regras de negócio envolvidas, além de repassar os *layouts* de transmissão dos arquivos de *mailing* e para a transmissão das propostas. Um dos meios mais utilizados para o recebimento dos nomes no *Call Center* e envio das vendas para o contratante é a troca de arquivos. Portanto é o contratante quem define o layout e formato para esses arquivos. É a maneira mais simples de relacionamento e comunicação entre as partes.

As necessidades da empresa contratante muitas vezes são de urgência e por este motivo a estrutura da empresa contratada tem que ser o mais adaptável possível. É comum para quem trabalha nesse meio receber demandas de cancelamentos, importações e alterações em massa, através de remessas com identificadores particulares da empresa contratante.

1.2 Produto

Assim como em toda a transação comercial, o produto é fundamental para o sucesso da atividade. Nem sempre as campanhas em um *Contact Center* são somente de aquisição de bens e serviços, podem ser pesquisas, *marketing* e qualquer outro tipo de contato possível pelo telefone.

No momento em que a campanha realizada trate da venda ao *prospect,* existem algumas dificuldades que podem ser apontadas, por exemplo, o fato de o contato ser realizado por telefone é uma mudança na cultura das pessoas e uma das principais dificuldades encontradas.

Nesse tipo de comércio não há como o *prospect* apresentar documentos ou assiná-los e por este motivo a gravação do áudio do contato do operador se torna a garantia de ambas as partes. Na necessidade de entrega de documentação, correios, fax ou entregadores são peças incluídas dentro do fluxo de informações de uma venda.

É interessante notar que mesmo não tendo sucesso o contato para venda, pode-se considerar que houve uma divulgação das qualidades e características do produto ofertado. Quebrar barreiras entre o *prospect* e o produto são desafios constantes na rotina de um operador de *telemarketing*, já que a resistência aos contatos por telefones está sempre presente.

Os produtos comercializados por meio de *telemarketing* precisam ser adaptados. No caso de serviços, os problemas apontados anteriormente necessitam que dados extras sejam solicitados, para evitar que aconteçam fraudes. Nos casos de aprovação de crédito, a exigência é muito maior, tanto que certas empresas preferem que o *Call Center* realize apenas contatos para clientes que tenham sido pré-aprovados e já tenham feito parte de seu quadro de clientes. Essa atitude ajuda a garantir perdas em relação à inadimplência.

Outra modalidade de atendimento para venda de produtos é aquela em que é feito apenas um pré-cadastro no *Call Center,* nesses casos é feito apenas um contato para captar apenas os dados principais, ficando para a contratante o contato para efetivar a compra. Essa serve para produtos mais complexos ou então para acabar com uma dificuldade que é a captação de novos clientes. É importante lembrar que o *telemarketing* atinge e entra em contato com um número muito maior de pessoas.

A quantidade de informações sobre o *prospect* no recebimento do *mailing* varia conforme o tipo de produto a ser ofertado. Para produtos como cartões de crédito a existência

de dados como a renda e data de nascimento podem facilitar o direcionamento do produto mais adequado ao perfil do *prospect*.

1.3 Ferramentas e dados disponíveis

Para que o operador de *telemarketing* possa atender com maior agilidade os clientes, ele precisa de uma estrutura tecnológica disponível. Cada operador possui um terminal ou um computador ligado em rede para que possa utilizar o *software* que vai disponibilizar para ele a pesquisa de informações e também que permite que todos os seus contatos e vendas sejam registrados. Juntamente, o telefone, a central telefônica e os equipamentos de gravação do áudio dos contatos são necessários para sejam registradas as conversas, se caso forem necessárias no futuro em questões judiciais ou de controle. Todas estas questões são transparentes para que o operador direcione seu foco para a realização das vendas.

As ferramentas disponíveis para que o operador possa gerenciar e realizar seus contatos é centralizado em um *software* que possui uma *interface* simples ao operador. Este *software* é ligado em rede e utiliza um banco de dados para disponibilizar os dados ao atendente. Este possui ao seu alcance funções como transferir e resgatar um *prospect* para uma agenda pessoal, a fim de utilizá-lo em um futuro atendimento, agendar seu contato para que um outro operador qualquer possa atendê-lo em outro momento, realizar a venda com o preenchimento dos dados obrigatórios, rejeitarem clientes e receber um outro nome da base de clientes.

Um *mailing* possui vários atributos sobre a pessoa que está sendo contatada. Dados como seu nome, data de nascimento, endereço e obviamente o telefone são disponibilizados ao operador no momento do atendimento. Em situações especiais outros dados são adicionados como renda, profissão, tempo de conta e qualquer outro dado que seja conhecido do contratante do serviço de *telemarketing*. Na figura 1.1 é mostrado um exemplo dos tipos de atributos presentes em um *mailing*.

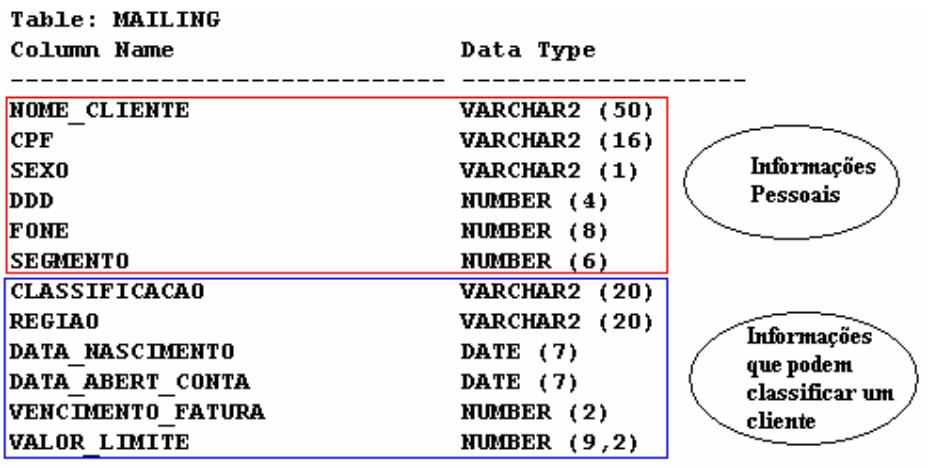

Figura 1.1 – Tipos de atributos de uma tabela *mailing* em um Banco de Dados
Fonte: FIGURA NOSSA

As empresas que realizam campanhas de *telemarketing* com seus clientes, podem ter atributos que classificam estes dentro da empresa. Em uma campanha de venda de títulos de capitalização, os clientes alvo podem ser aqueles que possuem cartão de crédito, desta maneira podem ser classificados como da categoria *internacional*, *gold*, *premium*, etc.

O recebimento dos nomes é totalmente transparente ao atendente e não há como este buscar um cliente que não esteja disponível ou em sua agenda particular. Para tal o *software* precisa de algum módulo que faça a busca destes nomes e entrega até o atendente. Essa busca de nomes pode ser simples através de uma consulta ao banco de dados em todos os computadores ou complexa com a utilização de servidor centralizando as buscas. Simplificando esse processo, o *software* poderia ir buscando na base de dados os nomes na mesma seqüência em que foi inserido cada *prospect* em cada lista. Os nomes vão sendo trabalhados e acabam sendo efetivados como vendas ou são sinalizados como uma rejeição à oferta. Os nomes podem ser também agrupados conforme suas características em listas distintas e estas serem disponibilizadas para trabalho separadamente. Ao fim destes nomes, uma nova carga é necessária para que os atendentes continuem seus trabalhos.

Esta carga de nomes na base é realizada ou pelo coordenador ou pela pessoa que tem como função exclusiva de cuidar da base de nomes. Para gerenciar estes dados, os coordenadores possuem ferramentas para que possam incluir e retirar de trabalho certas listas, além de possuírem relatórios que permitam acompanhar o rendimento de cada operador ou cada campanha.

1.4 Dificuldades

Em uma base de clientes, muitos podem não ter o perfil mínimo estipulado para a campanha e como os dados que o classificam como tal não são classificadores de prioridade de atendimento ou apenas foram captados no momento da ligação obrigam o atendente a fazer a rejeição do contato e um precioso tempo é perdido.

Estatisticamente quanto maiores forem as tentativas, da mesma maneira maiores serão as chances de realizar vendas. No momento em que os nomes das listas são trabalhados na ordem que estão no banco de dados, despreza-se a possibilidade de trabalhar primeiramente os nomes com um perfil igual ao de outras vendas já realizadas. Dessa forma, ao não trabalhar os nomes conforme seu perfil o mesmo desempenho atingido com uma lista completa, poderia ser atingida apenas com uma parte desta lista, se fosse possível ordenar pelo perfil do cliente.

Obrigar que cada estação fosse responsável por buscar um novo nome para atendimento, além de realizar várias consultas ao banco de dados deixa o controle descentralizado. É preciso considerar que quanto maior esta base de dados, maior será o tempo gasto para que seja realizada a busca desse novo nome para trabalho.

Conhecer os atributos que podem classificar ou identificar um possível comprador e não poder utilizar estas informações para focar os contatos também se torna uma das grandes perdas de desempenho.

Com o passar do tempo notou-se que alguns dos operadores de *telemarketing* tinham maior afinidade ao telefone com pessoas de maior idade, outros tinham facilidade com pessoas da região nordeste do país, outros ainda tinham facilidade em falar com pessoas que possuíam uma renda superior. Além desses dados também puderam ser associados determinados tipos de produtos a certo tipo de clientes e outros dados que auxiliavam no aumento das possibilidades de se conseguir contato de sucesso com os clientes.

A descoberta de conhecimento nesse contexto está mais voltada à conquista de experiência. As pessoas responsáveis pelo estudo do desempenho das vendas acabam baseando suas decisões e análises conforme o que aprenderam com o tempo. Não existe nenhum método cientifico utilizado para tal além da estatística. Nessa área perda de tempo é perda de oportunidades e isso certamente influencia nos resultados.

O próprio fato de com o tempo se conhecer os perfis que tem os melhores resultados, mesmo assim ter controle total para direcionar estes nomes para o atendimento deixa muito

impotente o processo da busca de nomes. Apenas dividir tipos de clientes diferentes e agrupá-los em listas com cada perfil é uma saída, mas demanda tempo para fazer a divisão e necessita de processos especiais para fazer a divisão da lista de *mailing* recebida. Ter que tratar os dados antes da carga de nomes de *prospects* no banco de dados vai ocupar tempo e vai gerar várias listas diferentes. Já que muitos são os tipos de cliente gerados pela combinação de atributos são muitos os perfis, por exemplo, o perfil de clientes mais velhos e com tempo de conta maior que dois anos formarão um perfil, da mesma maneira a faixa etária combinada com 5 tipos de faixa de tempo de conta formarão 5 perfis. A quantidade de perfis será igual ao produto da quantidade Q1 de variações de um atributo A pela quantidade Q2 de atributo B (A(Q1) * B(Q2)). Ainda sobre esta questão, a definição de quais atributos é importante é mais uma dificuldade que somente é resolvida através da realização de testes e analise dos acertos e erros.

Centralizar as informações e decisões nas pessoas pode prejudicar o processo de venda em uma empresa de *telemarketing*. É necessário que os melhores caminhos sejam conhecidos ou estarem ao alcance de qualquer um e não apenas a quem esta acompanhando diariamente a operação de *telemarketing*. Na primeira troca de funcionário podem ser perdidas importantes informações sobre o trabalho. Matemática e estatística podem ajudar a definir os melhores perfis, mas existem informações importantes dentro de uma base de dados que não podem ser identificadas matematicamente.

Através da quantidade das vendas e dos totais de cada tipo de cliente é possível encontrar os tipos de cliente que mais compram, mas a relação de mais de uma característica não é tão clara. Por exemplo, pode-se saber que quem tem renda maior compra mais, mas não é explícito que quem tem renda alta somente compra uma capitalização determinada e quem adquire alguma com maior valor são aqueles que possuem uma estabilidade profissional há mais de 2 anos. Nesse exemplo quem tem estabilidade menor apenas adquire capitalizações de valores menores.

1.5 Soluções e problemas não resolvidos

Como uma das primeiras soluções para as dificuldades, a busca dos nomes acabou sendo retirada de cada estação e criou-se um *software* servidor que centralizaria a busca e a distribuição dos nomes disponíveis.

Para resolver o problema para disponibilizar os nomes foram criadas tabelas dentro do banco de dados a fim de que cada atributo dentro do banco de dados possuísse um

identificador e que tivesse um cadastrado para o mesmo. Por exemplo, no caso da data de nascimento foi criado um cadastro de faixas etárias, dessa forma através da data era calculado a que faixa o cliente pertencia. Essa solução não foi nada mais do que fazer a normalização dos dados.

Neste momento já é possível classificar os clientes através dos novos atributos que são chaves estrangeiras para as tabelas do banco de dados que possuem as faixas de atributos e o cadastro dos atributos. Assim, já poderia ser definido que, por exemplo, o melhor cliente era o que possuía a faixa de renda 2, faixa etária 3, etc.

Com os dados normalizados fica possível acrescentar ao sistema novas características. Se necessário, agora pode ser criado uma forma de todos os nomes serem importados juntos em lista única e a distribuição ser feita conforme cada atributo. Se o usuário que é responsável pela gerência definir que um atributo qualquer é responsável por aumentar o rendimento das vendas, os clientes com este atributo deverão ter prioridade em relação a outros e dessa forma o usuário terá um controle maior sobre os nomes que serão disponibilizados.

Nota-se que dessa forma, a divisão dos nomes que chegam até a empresa de *telemarketing* é dispensável, visto que os nomes possuem uma forma de serem identificados de maneira mais eficiente e classificados.

Algumas das dificuldades ainda ficam por conta dos administradores do sistema, as tomadas de decisão ainda são feitas baseadas no conhecimento adquirido com o tempo e a escolha dos atributos mais importante ou do perfil melhor dos vendedores ainda é realizada com base no tempo de experiência dos gerenciadores da operação e por muitas vezes no conhecimento adquirido a cada erro ou insucesso.

Estas soluções já trazem para a campanha novas características e já pode ser mais bem aproveitada classificando os clientes e definindo prioridades no atendimento. O tempo que antes era perdido com os atendimentos a clientes com perfil inferior, agora fica para um segundo momento e aqueles que possuem as características comuns aos mais vendidos tem prioridade. Na busca da meta exigida, os melhores caminhos já estão disponíveis e certamente serão utilizados.

O problema atual está em conseguir adquirir mais conhecimento através do banco de dados. Através dos resultados obtidos no passado, conseguirem definir regras e associar

clientes com atributos diferentes, mas que formam uma classe de clientes com grande propensão de venda para os operadores.

Atualmente com o cenário comentado é preciso que além da utilização do conhecimento das pessoas, que existam maneiras de se definir níveis mínimos de desempenho para que o sistema aponte as principais combinações que permitam encontrar os melhores grupos de clientes.

Muitas informações importantes dentro da base de dados podem ou não estar visíveis ao usuário. Apoiar as decisões em métodos científicos de busca de informações deve ser realizado para que os resultados sejam mais consistentes e que erros ou enganos sejam evitados.

Precisam-se criar diversas opções de ações que podem ser tomadas e que estejam disponíveis para que o usuário possa fazer comparações e optar pelas regras mais eficientes. É interessante que se tenham mais regras para que sempre seja possível ter uma segunda opção.

Não existe como fazer planejamento da utilização das listas de clientes. Embora se saiba que certo atributo é fundamental para definir que o *prospect* pode ser um possível comprador, disponibilizar todos os nomes apenas apoiado nessa informação pode deixar escapar outros dados que em combinação trazem um resultado diferente.

As próprias escolhas de quais atributos são importantes não tem fundamentação alguma e a combinação de atributos é realizada testando durante o período de trabalho. Sendo ineficiente resultará em tempo perdido e muitos clientes em potencial não serão contatados. Essa perda tem causado um custo para empresa e deveria ser evitado ou diminuído.

2 A DESCOBERTA DE CONHECIMENTO EM BANCO DE DADOS (KDD)

Em vários setores de sociedade houve um crescimento dos dados das atividades do comércio, da medicina, dos negócios e da ciência que passaram a ser armazenados. Tanto foi o crescimento que estes dados saíram do controle daqueles que os armazenavam. Até o momento que a quantidade se mantinha em um volume pequeno, ficava fácil fazer uma análise dos dados através da visualização, os próprios SGBD disponibilizavam ferramentas que permitiam a visualização e a manipulação dos dados.

O nível da informação adquirida dentro dessas bases de dados era superficial e gerado com muito esforço. Todo o conhecimento adquirido ou disponível ao usuário era aquele que estava diante dos olhos e da capacidade deles de interpretar grandes quantidades de registros. Entretanto, informações valiosas se perdem em meio aos grandes volumes de dados, e por isto novas tecnologias se mostram necessárias ao processo de recuperação de informação.

A descoberta do conhecimento em banco de dados, conforme introduzido, é definido pelo processo de identificar padrões válidos e novos que possam ser utilizados e compreendidos (FRAWLEY; PIATETSKY-SHAPIRO; MATHEUS, 1992) (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). A busca desses padrões significa encontrar relacionamentos e classificações que tornam possíveis encontrar informações surpreendentes dentro das bases. A área de descoberta de conhecimento passou a ser uma das áreas mais estudadas e mais desejadas da computação, conforme WIEDERHOLD (1996).

Não é difícil encontrar exemplos de situações em que muitos dados são armazenados e acabam não sendo aproveitados. Grandes redes de comércio acumulam milhares de registros de suas vendas. Um exemplo muito citado nesse ramo são as grandes redes que possui números muito elevados de vendas diárias. Na ciência, a área do estudo do corpo humano como os códigos genéticos humanos, gera um banco de dados com *gigabytes* de informações e combinações. A medicina ao recolher informações sobre pacientes com a mesma doença a

fim de encontrar características que possam determinar algo em comum entre os pacientes ou alguma fragilidade (BRACHMANN; ANAND, 1996).

Registros da exploração de minério ou de petróleo são importantes para a continuidade do trabalho, da mesma forma se tornam de difícil análise pelos gestores, pela diversidade e quantidade de dados sobre toda a exploração.

Na área de tecnologia, estudos de interpretação de imagens podem utilizar as ferramentas de descoberta de conhecimento para aperfeiçoar processos de análise.

A grande quantidade de dados proporciona aos profissionais duas faces da situação. Com o auxílio dos bancos de dados muito mais informações podem ser armazenadas, maior é o histórico das transações e são guardados muito mais dados sobre o negócio. Por outro lado, por maior que seja a qualidade dos profissionais, chega um momento que a análise direta e por visualização se torna muito lenta ou então deixa de ser eficiente.

Percebe-se que existem pelo menos duas falhas que devem ser corrigidas. Primeiramente, esses dados devem ser processados de maneira mais rápida, para que os esforços possam ser direcionados a outras tarefas. A segunda melhoria seria incluir um método científico no processo de análise, garantindo, dessa forma, maior precisão e a descoberta de conhecimento contido dentro de milhares de registros.

Conforme WIEDERHOLD (1996), a tarefa de descobrir conhecimento não é simples. Os dados recolhidos e armazenados não são preparados de forma que a qualquer momento sejam analisados para que mostrem ao usuário os relacionamentos entre eles. Técnicas de KDD normalmente não são aplicadas em dados que já são alguns resultados de outro processo de KDD, mas aplicados em dados que podem ser de outros setores e áreas distintas[9]. Esses fatores são cruciais no momento em que é necessário atingir um nível alto de sucesso. Segundo (FREITAS, 1998), o tipo de método utilizado para que seja resolvida uma tarefa é o paradigma da descoberta de Conhecimento e para estes métodos é desejável eficiência, flexibilidade e generalidade.

Informações importantes podem não estar presentes na base ou podem ter sido registradas de forma incompleta. Tomando como exemplo a área comercial, ao se registrar os

[9] Essa origem, em geral não possui dados preparados para a mineração ou não pode ser utilizado sem uma nova transformação.

dados das vendas de um ano inteiro, muitos dos dados pessoais dos clientes podem ter sido alterados e com o passar do tempo isto contribui para que esses dados estejam desatualizados.

Outra situação que pode ocorrer é o caso da área coletora dos dados ter deficiência durante a coleta e por conseqüência, no fim da captação muitos atributos podem ser perdidos ou não identificáveis. A união de mais de uma base de dados pode ser utilizada para permitir que o modelo se torne completo e rico de informações.

Torna-se muito importante ter um bom modelo que possa proporcionar ao usuário a visualização e relacionamento entre todas as variáveis consideradas candidatas, inclusive adicionar aquelas que não estão disponíveis. Dessa maneira é possível definir qual é modelo que será utilizado para auxiliar nos possíveis problemas de falta de informação, que pode prejudicar o resultado, conforme (WALKER, apud WIEDERHOLD, 1996).

A falta de informação dentro da base a ser utilizada se torna um risco para alguns estudos (FRAWLEY; PIATETSKY-SHAPIRO; MATHEUS, 1992), por exemplo, embora se chegue à conclusão que o poder aquisitivo do cliente é fator fundamental para a aquisição de um título de capitalização, ainda pode existir outro fator que não foi considerado e que influencia fortemente a compra dos títulos.

O processo de KDD possui alguns passos que são altamente interativos e “em princípio, de qualquer passo talvez necessite voltar para um passo anterior” (TRADUÇÃO NOSSA), conforme (FREITAS, 1998, p. 41).

Os dados precisam ser tratados a fim de retirar as imperfeições e informações irrelevantes para fazer a busca de padrões e após, serão avaliados, para definir se todo o processo precisa ser refinado para obter maior qualidade. KDD com o auxílio de algoritmos pode gerar alguns padrões que o usuário pode interpretar e utilizar.

Uma característica muito importante para levar em consideração é que a descoberta do conhecimento não se dá exclusivamente por algoritmos e métodos, é preciso que exista a intervenção humana a fim de definir quais são os níveis e interpretar se as respostas geradas são úteis.

Para ilustrar o processo de KDD, segue a figura 2.1, representando as suas várias etapas.

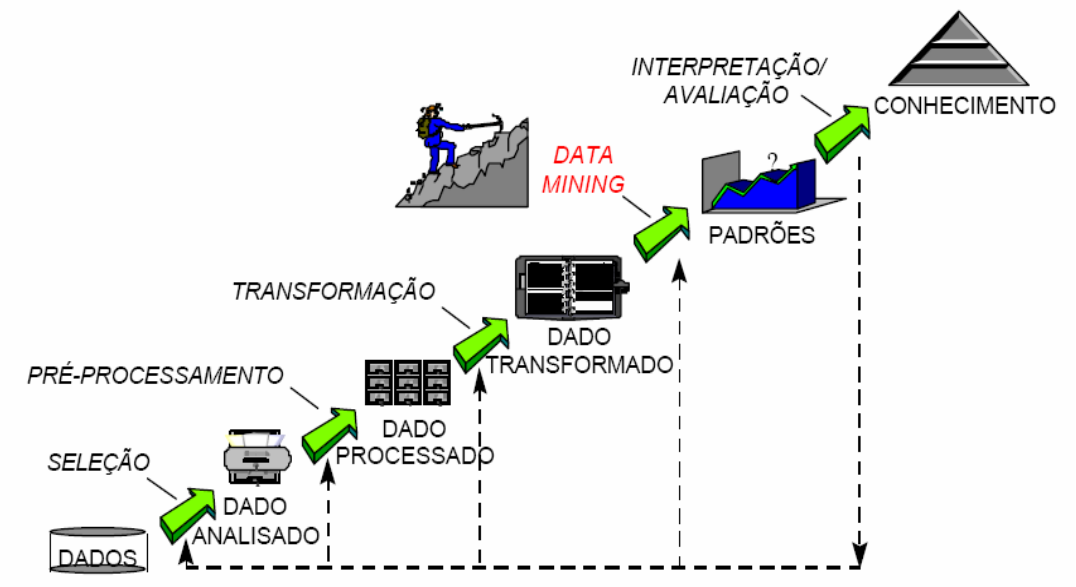

Figura 2.1 – O processo de KDD
Fonte: OLIVEIRA et al., 2002, p. 2

Antes mesmo de se aplicar as técnicas escolhidas, é preciso que a pessoa defina qual será seu foco e que possua algum conhecimento dentro da área onde os dados forem captados para que possa fazer a sua interpretação. O usuário deve eleger as principais variáveis para montar um modelo para testes.

O pré-processamento e limpeza dos dados se tornam essenciais para que seja removido tudo aquilo que não é necessário ou importante. Dificilmente se encontrará alguma base de dados que esteja totalmente preparada para ser processada por alguma técnica de KDD e, por este motivo, que WIEDERHOLD (1998) descreve como maior barreira para a aquisição de conhecimento os próprios dados armazenados.

Na análise de um público investidor em títulos de capitalização, não se pode afirmar que o perfil da base analisada é o mesmo para todos os investidores: mudanças financeiras e de confiança podem interferir. Nessa situação, podem existir outras variáveis que interferem nos números, porém ao analisar são nulas ou indiferentes. Se o número de variáveis envolvidas for muito grande pode ser necessário que seja feita uma redução da quantidade destas (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). Ao se definir um número muito grande de variáveis pode-se criar uma diversidade muito grande de resultados, todos com as características semelhantes e que não trazem informações úteis.

BRACHMANN e ANAND (1996) reforçam a afirmação que o processo de KDD não deve ser automático e deve ser assistido e orientado por um humano, para que os sistemas

de descoberta nos auxiliem a entender melhor como adquirir mais conhecimento em grandes bancos de dados. O ser humano sozinho não possui capacidade e velocidade o bastante para processar tantas informações e gerar conhecimento não explícito. Os sistemas por sua vez resolvem esse problema, mas não possuem ainda o bom senso necessário para interpretar os dados e apenas retornar aquilo que é interessante e que seja significante de fato. O que é importante considerar no resultado de um processo de KDD é que a informação gerada é tão pessoal, pelo fato do usuário determinar quando é relevante e acionável, que o conhecimento deve ser estruturado para que o sistema ou outras pessoas possam utilizar.

2.1 Arquitetura de KDD

Os passos do processo de KDD envolvem preparação dos dados e a mineração dos dados que é a parte fundamental da arquitetura de KDD, onde será feita a escolha dos algoritmos utilizados e qual técnica será aplicada. Após ser realizada a busca dos padrões (aplicação da técnica), chega o momento de fazer a interpretação. Como o processo de KDD é realizado de forma que se não atingir os objetivos propostos pode ser necessário reiniciar a rotina. Pode ser refeito até que a resposta possa ser considerada sólida e agregue algum tipo de conhecimento que não seria possível identificar simplesmente pela visualização dos dados.

Uma peça fundamental do processo de KDD, certamente é a figura humana. Não há uma maneira de medir exatamente se os objetivos foram bem definidos, se as metas são corretas e avaliar se o resultado atingiu o que era esperado. O amadurecimento dos resultados se dá exclusivamente com o intensa busca por melhorias. Em relação ao envolvimento de pessoas durante o processo de KDD, a partir das experiências de trabalho de estudiosos, BRACHMAN e ANAND (1996, p. 39-40) consideram que:

> Descoberta de Conhecimento é uma tarefa de aprendizado intensivo consistindo de complexas interações, protegido todo o tempo, entre um humano e uma grande base de dados, possivelmente auxiliado por uma heterogênea suíte de ferramentas. [...] Para o maior sucesso do desenvolvimento de ferramentas de suporte a descoberta de conhecimento, é necessário entender a exata natureza das relações entre humanos e os dados que levam a descoberta de conhecimento. (TRADUÇÃO NOSSA).

A arquitetura do processo de KDD pode ser composta de cinco fases, conforme figura 2.1, que auxiliarão na difícil tarefa de transformar os dados em informações (FAYYAD et al., 1996). Conforme já comentado, o processo de KDD normalmente inicia com os dados armazenados sem nenhum tipo de tratamento ou modelagem especial para que seja aplicada a descoberta de conhecimento. Estes dados armazenados de maneira "bruta" e representam o início do processo. Nesse momento, o problema é definido e são estipulados os objetivos a

serem alcançados. É muito importante levar em consideração o custo/benefício da análise dos dados, apenas realizar o processo de descoberta de conhecimento, sem um motivo ou sem um foco, traz uma chance muito grande de que apenas se perca tempo e não se consiga nenhum resultado expressivo.

A primeira etapa trata da seleção dos dados, onde são identificados os dados a serem utilizados e que bases de dados estão localizadas. Em relação aos dados, é importante levar em consideração algumas características: nesse ponto do processo de KDD pode ser necessária a junção de mais de uma base de dados e serão definidos quais os dados dessas ou dessa única base serão utilizados. As empresas acabam armazenando informações importantes para seus negócios e que podem ser aproveitadas no processo de KDD, mas isso não livra da necessidade ou priva esta de buscar informações de fontes externas.

Como exemplo de seleção, pode ser considerado a mineração de informações sobre as vendas de produtos importados em um site de compras pela internet quando os clientes adquiriram somente produtos que custaram mais de R$ 100, durante um mês, mas além desses dados deve-se incluir a cotação do dólar comercial de cada dia do período selecionado. Esta informação da cotação do dólar neste exemplo não estava presente junto aos dados e dessa forma foi obtida externamente agregando um outro dado que pode ser muito relevante na descoberta de conhecimento. Segundo BRACHMAN e ANAND (1996, p. 42), é o "analista o responsável pelos objetivos, realizando as consultas na base de dados para extrair dados relevantes ao objetivo" (TRADUÇÃO NOSSA).

Na etapa do pré-processamento dos dados é realizada a adaptação da base para que possa ser aplicada a mineração de dados. Essa etapa se torna necessária quando a base não está preparada ou a integração de duas bases acarrete na falta de consistência dos dados. Se a aplicação da descoberta de conhecimento fosse realizada através do tempo de conta no banco[10], esse dado do cliente possivelmente estaria armazenado em forma de data, sendo necessário que fosse feita a conversão para meses. Outra situação que pode ocorrer é que o formato das datas de duas bases, por exemplo, seja diferente ou uma delas já estar no formato de meses. A limpeza dos dados é uma tarefa que garante qualidade ao processo e nesse ponto cabe a pessoa que está realizando o pré-processamento remover os casos em que falta alguma informação ou que os dados estejam corrompidos.

---

[10] O tempo de conta representa a quantidade de meses que o cliente possui vínculo com a instituição.

A próxima etapa é a transformação, onde os dados de entrada serão recebidos do pré-processamento, de maneira que estarão formatados diferentemente de quando não haviam sido pré-processados. A transformação será exclusivamente para que os dados já formatados sejam organizados de maneira que a ferramenta e/ou técnica escolhida possa realizar a garimpagem. Cada ferramenta de mineração de dados e/ou técnica pode ter uma maneira especial de receber os dados. As etapas de seleção, processamento e transformação formam a preparação dos dados em um processo de KDD.

As próximas etapas serão compostas pela própria garimpagem dos dados, onde serão escolhidos os algoritmos e finalmente pela análise dos resultados a fim de identificar se o conhecimento é relevante (CARVALHO; SAMPAIO; MONGIOVI, 1999). No momento da análise é que vai ser definido se vai ser necessário que o processo seja reiniciado no caso do resultado não esteja dentro dos objetivos definidos.

2.2 Mineração dos dados (MD)

A mineração ou garimpagem de dados, segundo CABENA et al. (1997, p. 12) é o "processo de extrair previamente informação não conhecida, válida e acionável de grandes bases de dados e então utilizar a informação para realizar cruciais decisões no mundo dos negócios" (TRADUÇÃO NOSSA). A mineração de dados (MD), muitas vezes confundida com KDD, na verdade trata apenas de uma das etapas de todo o processo (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

Data Mining (DM) além de ser uma área interdisciplinar (FREITAS, 1998), segundo DWBRASILl (2007) nasceu da estatística, inteligência artificial e outra área que mistura essas duas, chamada de aprendizagem de máquina onde os programas adquirem conhecimento utilizando os conceitos da estatística e os algoritmos e técnicas da inteligência artificial. Deixando claro o objetivo de DM no processo de KDD, "a mineração busca selecionar o algoritmo ou os algoritmos para processar os dados" (TRADUÇÃO NOSSA) (CABENA et al., 1997, p. 55).

Uma característica importante de DM, são seus algoritmos, cada um deles têm específicas entradas e saídas, também conhecidas como suas técnicas (CABENA et al., 1997). Como mais conhecidas e utilizadas pode-se citar a descoberta de regras de associação, classificação, *clustering* e regressão.

2.2.1 Regras de associação

A descoberta de regras de associação, como o próprio nome resume, baseia-se na descoberta de regras que possam associar itens dentro da base e probabilisticamente, através do cálculo da freqüência de um item, definir as melhores regras.

As regras que podem ser consideradas melhores são aquelas onde o percentual de acerto é maior em relação aos outros. Por exemplo, uma regra para identificar as vendas de títulos de capitalização define que o melhor cliente é o que possui 22 anos, renda maior que R$800 e que mora na região sul. Essa regra se aplica a 81% do montante das vendas. Para que outra regra possua maior qualidade deve possuir um percentual maior que 81% dos registros da base.

Segundo AGRAWAL e SRIKANT (1994, p. 487), "o problema de mineração de regras de associação é gerar todas as regras de associação que tenham suporte e confiança maiores que o mínimo suporte especificado pelo usuário [...] e o mínimo de confiança [...]" (TRADUÇÃO NOSSA).

A regra de associação é definida de maneira que se "X então Y" ou "$X \Rightarrow Y$", onde X e Y são conjuntos de itens e $X \cap Y = 0$. Para essas regras X é definido como o antecedente e Y é o conseqüente (CARVALHO; SAMPAIO; MONGIOVI, 1999). Para que uma regra possa ser avaliada, os dois fatores são considerados: Suporte e Confiança. O suporte é o número de transações que contém o conjunto de item, divido pelo total de transações, ou seja, o suporte é a freqüência em determinado conjunto de itens, já que identifica a quantidade dos registros que possuem os atributos analisados em relação a todos os presentes na amostra.

Suporte = nº. registros com X e Y / nº. total de registros

A confiança é o cálculo para a regra "Se X então Y", onde o total de registros X e Y são divididos pelo total de registros X

Confiança = nº. registros com X e Y / nº. registros com X

O índice chamado de confiança representa a freqüência com que um conjunto de itens é comercializado em relação às vendas antecedentes, ou seja, é medida a confiança que se pode depositar em uma análise, de que registros de um conjunto possuirão características associadas a outras, em quantidade maior que o mínimo definido e esperado por quem aplica a técnica.

Para descobrir todas as regras de associação, existem alguns problemas envolvidos que podem ser caracterizados como:

- "Procurar todos os grupos de item (*itemsets*) que tenham suporte da transação acima do suporte mínimo" (TRADUÇÃO NOSSA) (AGRAWAL; IMIELINSKI; SWAMI, apud AGRAWAL; SRIKANT, 1994, p. 488). Conforme já discutido, o suporte é o número de tansações analisadas que possuem os *itemsets*, sendo que os grandes grupos são os que possuem o mínimo, ao contrário são pequenos grupos (AGRAWAL; SRIKANT, 1994).

- "Usar os grandes grupos de itens para gerar as desejáveis regras" (TRADUÇÃO NOSSA) (AGRAWAL; SRIKANT, 1994, p. 488). Para um grande grupo G, são geradas as regras para cada um dos subgrupos S no formato $S \Rightarrow (G - S)$, removendo S quando possuir valor inferior de confiança mínima (AGRAWAL; SRIKANT, 1994).

Como exemplo de algoritmo de regras de associação pode ser citado o algoritmo Apriori, um dos mais conhecidos e utilizados na área. Nesse tipo de técnica é mais comum reunir os registros de venda e selecionar os atributos para identificar as características que formam um perfil de comprador com potencial.

2.2.2 Classificação

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 dentro de uma base, que são chamados de conjunto de treinamento, que deve ser uma amostra dos registros que serão analisados (CABENA et al., 1997). A especialização da classificação apresentada nesse trabalho é a indução de árvores (*tree induction*), técnica que "constrói um modelo preeditivo na forma de árvores de decisão" (TRADUÇÃO NOSSA) (CABENA et al., 1997, p.51).

As árvores de decisão representam uma árvore de forma invertida, onde as raízes passam a ser folhas e essa hierarquia é disposta de forma que ao seguir a estrutura é possível tomar as decisão e executar a tarefa da maneira que foi proposta, já que em cada nível as opções a serem tomadas são os nós do nível seguinte e as decisões são tomadas até que sejam atingidos os nós terminais (MONGIOVI, 1998). Na classificação todos os registros fazem parte de classes que são identificadas pelo atributo objetivo e os registros vão conter esse atributo e um conjunto de atributos "previsores". Através das classes que já são conhecidas no conjunto de teste (relacionamentos encontrados), o objetivo então é descobrir o atributo que é

"meta[11]" naqueles registros que ainda não foram classificados (base a ser analisada) (FREITAS, 1998).

Como exemplo de classificação, podemos separar 100 registros de pessoas que foram contatadas pela operação de *telemarketing* e definir que nosso atributo meta serão os clientes que compraram ou não o título de capitalização. Nosso objetivo com essa classificação é encontrar em outros 2000 clientes ainda não contatados àqueles que através da árvore criada são prováveis compradores. Os métodos automáticos de classificação podem ser indutivos ou de indução neural (também conhecidos como conexionistas por alguns autores). Os algoritmos indutivos são aqueles que geram as árvores de decisão e os conexionistas são as redes neurais (CABENA, 1997).

2.2.3 Clustering

Clustering, ou ainda agrupamento, é uma técnica que visa criar classes e agrupa os registros com atributos semelhantes. É um tipo de aprendizagem não supervisionada, o que quer dizer que o resultado não requer avaliação do usuário. "*Clustering* é uma tarefa descritiva comum onde uma semente identifica um grupo finito de categorias ou *cluster* para descrever os dados" (TRADUÇÃO NOSSA) (JAIN; DUBES, apud FAYYAD et al., 1996, p.14).

Essa técnica pode ser utilizada para realizar uma análise inicial dos dados e assim obter uma visão geral dos agrupamentos existentes na base de dados e então, analisar a fim de definir quais grupos serão utilizados ou levados em consideração. Pode-se, após essa técnica, utilizar outra como a classificação. Na figura 2.2, pode-se observar um gráfico fictício de que mostra um *clustering* aplicado a algum tipo de dado, onde foram identificados 3 clusters.

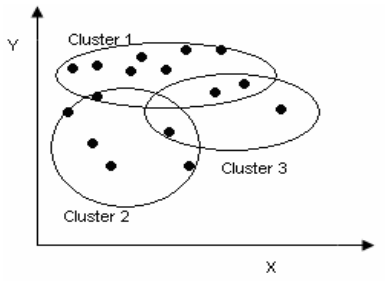

Figura 2.2 – Representação de 3 *Clusters* gerado com a técnica
Fonte: Adaptado de FAYYAD et al., 1996, p. 14

[11] É o que se deseja descobrir dos registros com base no resultado do conjunto de treinamento.

Como característica de *clustering* pode-se destacar que não há um atributo meta e todos possuem a mesma importância. A técnica é utilizada especialmente para a exploração e sumarização dos dados (FREITAS, 1998).

Como exemplos, as características dos clientes que já adquiriram títulos de capitalização podem ser analisadas. A idéia de aplicar *clustering* nesse caso é descobrir grupos ou categorias de clientes. Essas categorias são geradas através dos atributos como idade, renda, residência, tempo de relacionamento com a instituição financeira, sexo, etc. Ao ser feita uma média simples de idade e renda, chega-se a uma média de clientes com 37 anos e renda de 2000 reais, porém existem alguns clientes com idade muito elevada e com renda menor que 2000 reais. Dessa maneira, apenas através de uma média tem-se a impressão de que a maioria dos registros possui realmente essa idade e renda.

Ao aplicar o *clustering*, se deseja conhecer quais são os grupos de clientes conforme características semelhantes existentes na base. Continuando o exemplo, é encontrado um *cluster* de idade próxima 24 anos e renda aproximada R$5000 reais, segundo a quantidade de registros que se enquadram nesse *cluster*, torna-se possível identificar que é grande o bastante, em relação ao total, para ser dado foco aos *prospects* com essas características no momento do atendimento.

A aplicação da técnica dá ao usuário uma visão geral de todos os registros da base, já que vários *clusters* podem ser identificados e conforme sua quantidade é possível definir os mais importantes. Outra característica do resultado da técnica é que fica possível entender a tendência de aquisição do produto conforme as características. Ao colocar as dados em um gráfico, por exemplo, nota-se que a quantidade clientes compradores cresce conforme vai aumentado um atributo X e diminuindo outro Y, ao contrário ou ainda ao aumentar e diminuir ambos.

2.2.4 Regressão

"A regressão é uma função de aprendizado que mapeia um atributo para uma variável preeditiva real-validada" (TRADUÇÃO NOSSA) (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996, p. 13). A regressão serve para estimar a probabilidade de certo fenômeno.

Nessa técnica é gerada uma linha de regressão que representa a relação entre duas variáveis representadas na figura 2.3 em forma de gráfico. A linha desenhada na diagonal no

gráfico representa a função linear onde se classificam os registros. Para os próximos dados a serem analisados, aqueles que estiverem próximos dessa linha tem maior probabilidade de possuírem o mesmo resultado.

Exemplificando, sendo X a renda e Y o valor do título comprado, se obtém a função linear que determina o valor de título adquirido conforme o valor da renda do cliente. Na figura 2.3, é apresentada a representação de uma linha de regressão.

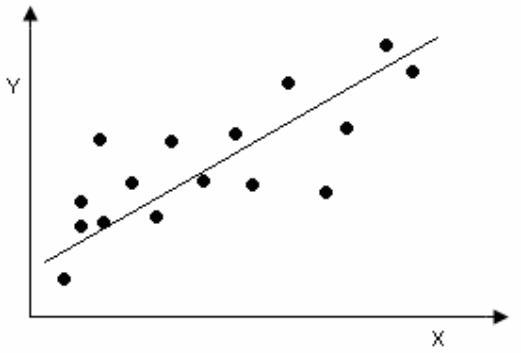

Figura 2.3 – Representação da Linha de Regressão
Fonte: Adaptado de FAYYAD et al., 1996, p. 14

A forma mais simples de regressão é a linear, utilizada na figura 2.3. São utilizadas duas variáveis, onde uma delas é aleatória (y), que é a função linear de outra deste mesmo tipo (x), formando a equação (y = a + bx) (LÓPEZ; HERRERO, 2004). "Nesta equação a variação de *y* se assume que é constante, e *a* e *b* são os coeficientes de regressão que especificam a intersecção com a linha de ordenadas, e a inclinação da reta" (TRADUÇÃO NOSSA) (LÓPEZ; HERRERO, 2004, p. 56). O coeficiente de regressão é medido através do método dos mínimos quadrados, que utilizam as equações abaixo, para diminuir os erros dos dados e da estimativa da linha (PRESS et. al, apud LÓPEZ; HERRERO, 2004).

$$b = \frac{S_{xy}}{S_x^2}$$

$$a = y - bx$$

Ao obter S registros de exemplo com seus pontos (X1, Y1), (X2, Y2) ... (Xs, Ys) é possível obter os coeficiente com essas equações, onde Sxy é a covariância de x e y, enquanto a variância de x é representada por Sx^2.

Na figura 2.4, tem-se um exemplo de regressão linear que representa vendas de capitalização, estudando a relação do valor da renda e do valor dos títulos adquiridos:

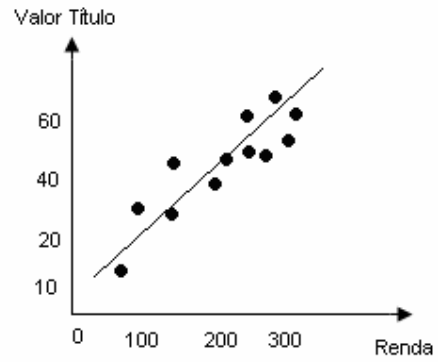

Figura 2.4 – Regressão linear do valor de renda e valor de título adquirido
Fonte: FIGURA NOSSA

No exemplo da figura 2.4, pode-se perceber esta relação, onde a regressão identificou que a aquisição do valor do título está ligada com o valor da renda. O aumento desse total sugere que a aquisição de título de capitalização se dá conforme o aumento do valor da renda.

A regressão linear múltipla segue os mesmo princípios da regressão linear, porém utilizando mais de uma variável preditiva (x) e a variável (y) como uma função linear de um vetor multidimensional ($y = b_0 + b_1x_1 + b_2x_2 + ... + b_nx_n$).

Segundo LÓPEZ e HERRERO (2004, p. 60-61), ainda existem as regressões lineares ponderadas localmente, que "geram modelos durante o processo preeditivo" (TRADUÇÃO NOSSA), dando peso diferente para os exemplos de treinamento e a regressão não linear utiliza uma função polinomial para os dados que o resultado depende do valor das variáveis independentes da função polinomial.

2.3 Interpretação e avaliação dos dados

Esta última etapa, embora seja a mais simples, é uma das mais importantes. Nesse momento do processo de KDD é que o usuário, que deve sempre acompanhar o processo, define se a resposta da mineração é útil.

É algo comum na mineração de dados, que os algoritmos utilizados tragam até o usuário alguma informação que não pode ser considerada como relevante. Por exemplo, ao aplicar uma regra de associação se obtém uma regra indicando que as pessoas com renda muito alta compram títulos de valor maior, logo, para este caso, a informação já era conhecida.

Quando o usuário interpretar o resultado do processo de KDD, ele vai identificar a necessidade ou não de reiniciar o processo e gerar outro tipo de regra ou informação, se as obtidas não forem acionáveis. Após a avaliação, se o conhecimento gerado 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; PIATETSKY-SHAPIRO; SMYTH, 1996).

3 WEKA

A ferramenta Weka[12] é um *software open source* implementado com a linguagem Java, desenvolvido pela Universidade de Waikato (WITTEN; FRANK, 2005) e disponível em (WEKA, 2007). O *software* implementa diversos algoritmos de mineração de dados, possibilitando ao usuário gerar arquivos de texto (.arff) para serem analisados. Na figura 3.1 pode-se verificar o formato do arquivo .arff:

```
%Informações dos registros de venda na base de teste
@relation perfil-vendas

@attribute CLASSIFICACAO        {BASE M2,CAPITALIZACAO,CAPITALIZACAO M2,EXCLUSIVO,INSTITUCIONAL,PLATINUM,UNICO,NONE}
@attribute COD_REGIAO_MAILING   {1,2,3,4,5,6,7,8,9,10,11}
@attribute COD_CLASSIFICACAO    {1,2,3,4,5}
@attribute DIA_VENC_FATURA      {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,24,25,26,27,28,29,30}
@attribute COD_TEMPO_CONTA      {1,2,3,4,5,6,7,8}
@attribute COD_FAIXA_ETARIA     {1,2,3,4,5,6}
@attribute COD_VALOR_LIMITE     {1,2,3,4,5,6,7,8,9,10,11,12,13}
@attribute COD_SEXO_ATB         {1,2}

@data
%
% 12 instances
%
INSTITUCIONAL,4,4,1,1,1,1,3
UNICO,11,2,1,1,1,1,3
PLATINUM,7,3,1,1,1,1,3
PLATINUM,8,3,1,1,1,1,3
EXCLUSIVO,7,5,1,1,1,1,3
PLATINUM,7,3,1,1,1,1,3
EXCLUSIVO,7,5,1,1,1,1,3
UNICO,4,2,1,1,1,1,3
UNICO,8,2,1,1,1,1,3
UNICO,7,2,1,1,1,1,3
UNICO,7,2,1,1,1,1,3
INSTITUCIONAL,6,4,1,1,1,1,3
```

Figura 3.1 – Arquivo .arff do Weka
Fonte: WEKA, 2007

O arquivo arff é separado por *tags* que identificam a estrutura do arquivo. A *tag* @relation, identificada na figura 3.1 por (1) identifica o nome do arquivo. Cada *tag* @attibute (2) identifica no arquivo os atributos presentes e seus tipos. A seqüência das *tags* é seguida conforme a seqüência aqui apresentada. Para marcar o início dos dados a serem analisados é

[12] O *software* foi batizado dessa maneira pelas iniciais de *Waikato Enviroment Knowledge Analysis*

inserido no arquivo @data (3), abaixo desta tag, todos os dados serão analisados. A presença de % no início da linha (4) comenta toda a linha e então não será considerada.

A ferramenta permite ao usuário alterar os parâmetros de entrada dos algoritmos através de uma interface gráfica que facilita a sua utilização. A ferramenta é formada por pacotes, específicos para cada aplicação de técnica de mineração de dados. Os pacotes são: *attribute selection*, *classifiers*, *clustering*, *association rules*, *filters* e *estimators*. O primeiro pacote *weka.atributeSelection* que seleciona os atributos em uma base de dados ou arquivo. Os pacotes *weka.classifiers*, *weka.cluster*, *weka.association* possuem as implementações dos algoritmos de cada técnica, respectivamente. O pacote *weka.estimators* possue subclasses que servem para "computar os diferentes tipos de distribuição de probabilidade" (OLIVEIRA et al., 2002, p. 4). O pacote *weka.filters* permite selecionar um conjunto de atributos ou instância de dados (OLIVEIRA et al., 2002).

O Weka vem sendo utilizado em vários trabalhos de DM, torna fácil sua utilização pela portabilidade e facilidade de implementação. Essa ferramenta foi escolhida por possuir os principais algoritmos e técnicas que serão estudadas e já ter sido utilizada com sucesso por outros trabalhos. Como exemplos de utilização do Weka, podem ser citados trabalhos como (OLIVEIRA et al., 2002), (PINHEIRO, 2006) e (LOPÉZ; HERRERO, 2006).

O algoritmo de associação implementado pelo Weka está localizado no pacote *weka.association* em duas classes *ItemSet* e *Apriori,* que são responsáveis pelo algoritmo *Apriori* que é um dos mais conhecidos para esse tipo de técnica. Entre os algoritmos de classificação presentes no Weka estão incluídos os algoritmos *weka.classifiers.J48.J48* e o *weka.classifiers.J48.PART*. Já o pacote *weka.cluster* implementa, por exemplo, dois algoritmos de aprendizagem não supervisionada: *Cobweb* e *EM*. O Weka também possui algoritmos para regressão que podem ser selecionados dentre os algoritmos de classificação ou através do pacote *weka.classifiers.functions*. A seguir, serão apresentados alguns algoritmos presentes no Weka.

3.1 Algoritmo de regra de associação

O algoritmo Apriori se encarrega de fazer a busca por regras de associação entre os dados contidos dentro da base. É importante ressaltar que, na maioria das vezes, é preciso que se faça um tratamento dos dados antes de serem processados pelo algoritmo. A fim de recuperar mais precisas e melhores informações, a base de dados a ser analisada deve ser possuir somente atributos considerados importantes. Ao retirar os dados que não são

relevantes ou que não devem fazer parte da análise, o algoritmo poderá realizar sua função de modo mais eficiente.

Considerando que este estudo trata de dados que estão normalizados dentro de um SGBD, estes estarão agrupados conforme a necessidade do *software* criado para realizar as vendas com os *prospects*. Junto com informações importantes como idade, residência, valor de renda encontram-se chaves primárias de tabelas, campos de texto com observações e outros dados que são de difícil agrupamento e talvez códigos que são exclusivamente de controle do *software* ou da própria empresa contratante que envia os cadastros de *prospects*.

Para retirar estes dados desnecessários em um primeiro momento, pode ser feito um tratamento e recuperar somente aqueles que são inicialmente classificados importantes pelo usuário. Com a seqüência de experimentos sendo realizados é que se atinge a maturidade em relação aos dados que são realmente relevantes.

O algoritmo *Apriori*, através da análise dos dados busca recuperar conjuntos de itens que acabam sendo freqüentes e a estes conjuntos é dado o nome de *itemsets* freqüentes (L_K). Seu objetivo é encontrar todos estes conjuntos presentes na base analisada. Para aprimorar seus resultados, estes conjuntos são recuperados em uma quantidade mínima estipulada. Isso garante definir o nível de critério da associação

O algoritmo é composto por uma estrutura principal e mais duas estruturas secundárias, que são funções utilizadas pela parte principal. Uma delas realizando a geração de candidatos e eliminando os que não são freqüentes (Apriori-gen) e a outra gerando as regras de associação.

O algoritmo trabalha através de dois passos, um deles é a geração onde são criadas as combinações encontradas no arquivo e o outro passo trata de fazer o corte das combinações que não aparecem na freqüência desejada e pré-estipulada (Suporte e Confiança). Na figura 3.2, é apresentado o algoritmo Apriori.

```
Function Apriori(Banco de Transações D, Smin) Grandes Conjuntos de Itemset (L)

L1 = {grande 1-itemsets};
k = 2;

Enquanto Lk-1 <> 0 faça
        Inicio
                Ck = apriori-gen(Lk-1; {gera os novos candidatos}
                Para toda  transação t Є D faça
                        Inicio
                                Ct = subconj(Ck, t); {candidatos contidos em t}
                                Para todo candidato c Є Ct faça
                                        contador(c) = contador(c) + 1
                        fim;
                Lk = {c Є Ck | contador(c) ≥ Smin};
                k = k + 1
        fim;

Retorna (L = Uk Lk).
```

Figura 3.2 – Algoritmo Apriori
Fonte: Adaptado de MONGIOVI , 1998

O algoritmo Apriori realiza a contagem dos grandes grupos e em seguida (k), são gerados candidatos (C_k) dentro de cada grande grupo (L_{k-1}) existente no (k-1) passo do algoritmo e verificado o suporte dos candidatos (AGRAWAL; SRIKANT, 1994).

Para que fique mais claro, é possível fazer uma simulação do funcionamento do *Apriori*. Para isso pode-se considerar um pequeno exemplo com dados sobre algumas vendas[13] e seus atributos.

Na tabela 3.1, é apresentado um exemplo de dados para aplicar o algoritmo, que serão os *itemsets* (L), a coluna venda identifica o identificador de cada registro e o restante dos dados representa a presença da característica (1) ou então a ausência da mesma (0):

Quadro 3.1 – Quadro de Vendas de Capitalização

Venda	Masculino	Renda > 1000	Reside em SP
1	1	1	1
2	0	1	1
3	1	1	1

[13] Vendas de títulos de capitalização de uma instituição bancária.

Venda	Masculino	Renda > 1000	Reside em SP
4	1	1	1
5	1	0	1
6	1	0	1
7	1	1	1
8	1	1	1
9	1	1	0
10	1	1	0

Fonte: Do autor

No primeiro passo do algoritmo principal, este deve determinar os *itemsets* freqüentes através da contagem das ocorrências e em seguida nos passos (k), realizar outras operações. Pode ser definido como suporte mínimo 0,6. Dessa forma, já é possível separar os grandes conjuntos que são:

Tabela 3.1 – Grande grupo de um elemento

Atributo	Suporte
Masculino	0,9
Renda	0,8
SP	0,8

Fonte: Do autor

Tabela 3.2 – Grande grupo de dois elementos

Atributo	Suporte
Masculino, Renda	0,7
Masculino, SP	0,7
Renda, SP	0,6

Fonte: Do autor

Para calcular o suporte de cada grupo, é aplicada a fórmula do suporte. Para ilustrar, o suporte do atributo “Masculino”, do grupo de um elemento, é feito através do total de registros “Masculino” (9 registros) dividido pelos 10 registros que são o total da amostra, obtendo o suporte 0,9. Para o grupo de dois registros é feito o cálculo, quando os dois atributos são 1, divido pelo total: 7 (Masculino e Renda) / 10 (Total) = 0,7.

O conjunto abaixo acabou sendo cortado pelo fato do suporte não ter atingido o valor mínimo estipulado, que no caso foi 0,6.

Tabela 3.3 – Grupo que não foi selecionado pela simulação

Atributo	Suporte
Masculino, Renda, SP	0,5

Fonte: Do autor

Nestes próximos passos chamados de K, a primeira ação a ser tomada é gerar os chamados *itemsets* candidatos (C_k), conforme tabelas 3.1 e 3.2 e 3.3, que são *itemsets* encontrados em (k-1) e são definidos como possíveis *itemsets* freqüentes. Nesse ponto a função Apriori_gen é utilizada para gerar um conjunto de todos *itemsets* freqüentes, levando-se em conta de que se o suporte mínimo foi alcançado, os conjuntos abaixo deste também alcançarão este mínimo.

Os *itemsets* são comparados e retirados os *sub-itemsets* (c_k) que pertençam a C_k e que não pertençam (k-1) ao *itemset* L_k-1. Após isso, o algoritmo deve fazer uma nova busca entre os dados levando em conta o nível de suporte encontrado em cada candidato definido.

A função Apriori-gen tem a função de unir os elementos de L_{k-1} a cada 2 e reter apenas aqueles em que todos os seus subconjuntos de tamanho k-1 pertençam a L_{k-1}. Na figura 3.3, pode-se verificar a função Apriori-gen:

```
Apriori-gen(Lk-1)

    {Junção}
    Insert into Ck
    From  Lk-1p, Lk-1q {elementos p e q de Lk-1}
    Select p.item1 = q.item1, ...p.itemk-2 = q.itemk-2
           p.itemk-1 < q.itemk-1;

    {Poda}
    Eliminar todo c que pertence a Ck tal que algum (k-1)
    subconjunto de c não pertence a Lk-1
           Ck = {c ∈ Ck | ∀ s ⊂ c, s ∈ Lk-1}

    Para  todo c ∈ Ck faça
           Para todo s ⊂ c e |s| = k-1 faça
                  Se s não pertence a Lk-1 então elimina c de Ck
```

Figura 3.3 – Função Apriori-gen
Fonte: Adaptado de MONGIOVI, 1998

Para ilustrar o Apriori-gen, será definido como nível de confiança 0,8. Aplicando a fórmula que define a confiança citada anteriormente resultará em:

Tabela 3.4 – Regras com sua classificação de confiança

Regra	Fator de Confiança	Além do mínimo?
{Masculino} → Renda	0,77	NÃO
{Renda} → Masculino	0,87	SIM
{Masculino} → SP	0,77	NÃO
{SP} → Masculino	0,87	SIM
{Renda} → SP	0,75	NÃO
{SP} → Renda	0,75	NÃO

Regra	Fator de Confiança	Além do mínimo?
{Masculino, Renda} → SP	0,62	NÃO
{Masculino, SP} → Renda	0,62	NÃO
{Renda, SP} → Masculino	0,55	NÃO

Fonte: Do autor

Por fim, a geração das regras é feita ao percorrer os *itemsets* freqüentes e descobrir os *subsets* que não estão vazios quando o suporte do *itemset* freqüente dividido pelo suporte do *subset* atender ao mínimo de confiança definido no algoritmo.

As primeiras 6 regras são geradas a partir do grande grupo de um elemento (Tabela 3.1). Para calcular a confiança de cada regra é aplicada a fórmula quando o atributo possui o valor igual a 1. Por exemplo, a regra {Masculino} → Renda é calculada, dividindo a quantidade de registros "Masculino" e "Renda" (7 registros) pela quantidade de registros "Masculino" (9 registros), resultando na confiança 0,77. Da mesma maneira são geradas as outras regras (X → Y). As últimas 3 regras são realizadas com o grupo de dois elementos, onde somente são somados os registros de X e Y quando, no caso de {Masculino, SP} → Renda, os registros possuem "Masculino", "SP" e "Renda" igual a 1 (5 registros) dividido pelos registros "Renda" (8 registros), ou seja, confiança 0,62.

Como foi definido nível de confiança 0,8, teremos somente duas regras que atingirão o mínimo de confiança que serão {Renda} → Masculino e {SP} → Masculino, ou seja, das vendas que já foram realizadas, se o cliente se possui renda maior de R$1000, então é do sexo masculino e se é morador de São Paulo, então seu sexo também é masculino.

Se ao analisar os resultados, for definido que estes são aplicáveis e trazem informações importantes, os contatos para novas vendas poderão ser direcionados para os clientes com renda maior de R$1.000 e do sexo masculino ou ainda homens e residentes em São Paulo. Esse exemplo possui poucos registros, mas serve didaticamente para compreender a aplicação da técnica de regras de associação.

3.2 Algoritmos de regra de classificação

A ferramenta Weka possui dois algoritmos de Classificação da família J48, que são o J48.J48 e o J48.PART. O primeiro destes é uma versão desenvolvida na linguagem Java para o Weka do algoritmo C4.5 release 8 de (QUINLAN, 1993) que é a última versão do algoritmo de geração de árvores antes do C5.0 e geram árvores de decisão c4.5 com ou sem poda.

O C4.5 (QUINLAN, 1993) é um algoritmo melhorado em relação do Iterative Dichotomizer 3 (ID3) (QUINLAN, 1990) que consiste em realizar indução de árvores de cima para baixo recursivamente, para tentar escolher sempre o melhor nó de decisão da árvore, que entre as melhoras inclui o combate aos métodos de *overfliting*[14], podando a árvore.

O algoritmo utiliza um tipo de pós-poda, onde um ramo da árvore é podado e transformado em folha. Esse corte é feito de forma estatística, levando em consideração os erros em um nó e seus descendentes, dessa maneira só haverá a poda se o desempenho de toda a árvore não sofrer grande impacto. Mais duas características podem ser destacadas, a validação cruzada de um ou mais grupos que serve para melhorar a estimativa de erro e a outra de gerar regras de decisão a partir de árvores e compará-las entre si (QUINLAN, 1993).

A validação cruzada é a operação onde os "dados de treinamento são misturados e reamostrados para a classificação com a árvore criada" (TRADUÇÃO NOSSA) (SANTOS, 2005, p. 6). Esta experiência é repetida conforme o número de dobras (*folds*) definidas.

O J48.J48, que é a versão Java do C4.5 no *software* Weka, "constrói um modelo de árvore de decisão baseado em um conjunto de dados de treinamento e usa esse modelo para classificar outras instâncias num conjunto de teste" (TRADUÇÃO NOSSA) (OLIVEIRA et al., 2002, p. 4-5).

Este algoritmo possui como parâmetros informações importantes para definir o grau de qualidade que as árvores serão geradas. Como exemplos dos parâmetros podem ser citados a quantidade mínima de instâncias por cada folha e também um parâmetro para identificar se a árvore poderá ser binária.

O outro algoritmo (J48.PART) constrói regras de produção a partir da árvore de decisão e está é a diferença em relação ao J48.J48. Para a criação dessas regras o algoritmo induz regras inicialmente de uma árvore montada e depois segue refinando estas. Para cada uma dessas regras "é estimada a cobertura das instancias da base. Isso ocorre repetidamente até que todas as instâncias sejam cobertas" (TRADUÇÃO NOSSA) (OLIVEIRA et al., 2002, p. 5).

---

14 Overfliting é quando a taxa de acertos no conjunto de treinamento é alta, mas alcança níveis muito baixos nos teste. (MATINHAGO, 2005)

3.2.1 Algoritmo ID3 (Iterative Dichotomizer 3)

O algoritmo ID3 é um algoritmo indutivo, seu conhecimento preliminar do conjunto de treinamento gera informação, que depois vai ser validada. Esse conhecimento conforme já discutido, é gerado no formato de árvores de decisão. O algoritmo ID3 é apresentado na figura 3.4.

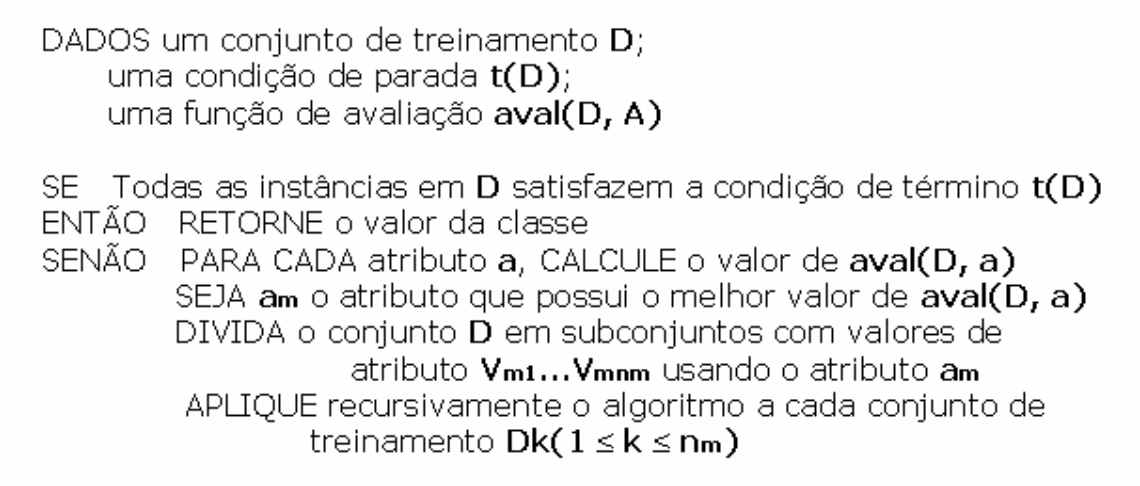

Figura 3.4 – Algoritmo ID3
Fonte: Adaptado de MONGIOVI, 1998

A função de avaliação das regras que foram induzidas é a entropia, quanto menor for este valor mais informativo será o atributo, isso significa também que menor será a árvore gerada. A escolha do melhor nó, já comentada anteriormente é feita através de uma função de avaliação, que utiliza estatística.

No quadro 3.2, pode-se observar um conjunto **S** de exemplos:

Quadro 3.2 – Conjunto S, que representa o conjunto de treinamento

	REGIÃO	MASCULINO	IDADE	GOLD	CLASSE
Ex1	Sul	Sim	> 30	Sim	Venda
Ex2	Sudeste	Sim	> 30	Não	Venda
Ex3	Norte	Não	> 30	Sim	Rejeição
Ex4	Sul	Não	> 30	Não	Venda
Ex5	Sudeste	Não	< 31	Sim	Rejeição
Ex6	Sul	Não	> 30	Não	Rejeição
Ex7	Norte	Sim	< 31	Não	Venda
Ex8	Sudeste	Não	< 31	Sim	Rejeição

Fonte: Do autor

E um conjunto de n classes C={C_1, C_2, C_3...C_n), no caso {Venda, Rejeição}, sendo que a probabilidade (p_i) da classe C_i em S, a entropia deste conjunto de exemplos S é representada na figura 3.5:

$$Entropia = -\sum_{i=1}^{c} p_i \log_2 p_i$$

Figura 3.5 – Função da Entropia
Fonte: Adaptado de MONGIOVI, 1998

Ou seja, a função de avaliação de um atributo **a** é a média ponderada dos grupos de exemplos segundo **a**, onde:

c – número de classe no conjunto de treinamento

p_i – é o número de exemplos em que o atributo **a** possui o valor v_i, dividido pelo número de exemplos no conjunto de treinamento com uma classe. Em outras palavras, é a probabilidade de se ter um número de exemplos (> 30 anos e venda) dividido pelo número de vendas.

Executando um exemplo, sendo que a coluna *Gold* identifica os clientes que possuem cartão *Gold* entre a base de treinamento (note que é feito para cada classe, venda e rejeição), teremos:

Entropia(Gold = Sim) = - 1/4.log2(1/4) – 3/4.log2(3/4) = 0,81

Entropia(Gold = Não) = - 3/4.log2(3/4) – 1/4.log2(1/4) = 0,81

Entropia(Gold) = (4/8).0,81 + (4/8).0,81 = 0,81

Atributo Região:

Entropia(Região = Sul) = - 1/3.log2(1/3) – 2/3.log2(2/3) = 0,881

Entropia(Região = Norte) = - 1/2.log2(1/2) – 1/2.log2(1/2) = 1

Entropia(Região = Sudeste) = - 1/3.log2(1/3) – 2/3.log2(2/3) = 0,881

Entropia(Região) = (3/8).0,881 + (2/8).1 + (3/8).0,881 = 0,785

Atributo Idade:

Entropia(Idade = >30) = - 3/5.log2(3/5) – 2/5.log2(2/5) = 0,937

Entropia(Idade = <31) = - 1/3.log2(1/3) – 2/3.log2(2/3) = 0,881

Entropia(Idade) = (5/8).0,937 + (3/8).0,881 = 0,916

Para o atributo Masculino, que foi selecionado como melhor:

Entropia(Masculino = Sim) = - 3/4.log2(3/4) – 0/4.log2(0/4) = 0,33

Entropia(Masculino = Não) = - 1/4.log2(1/4) – 4/4.log2(4/4) = 0,50

Entropia(Masculino) = (3/8).0,33 + (5/8).0,50 = 0,43

Aplicando o algoritmo chega-se até a árvore da figura 3.6:

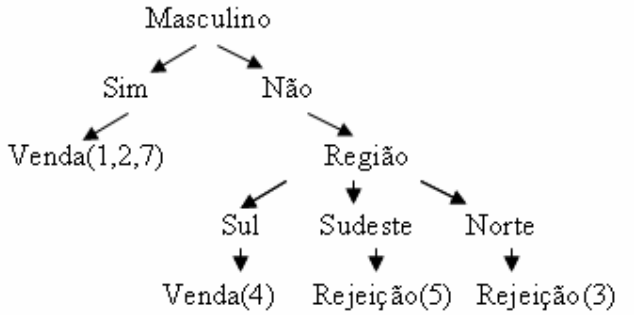

Figura 3.6 – Árvore gerada através do algoritmo ID3
Fonte: FIGURA NOSSA

Para entender a aplicação, serão seguidos os passos. O atributo escolhido "Masculino" será utilizado para criar as classes D_k de D, onde D(sim) = {1, 2, 7} e como todos são "Venda" a execução é interrompida já que a regra diz que todos os masculinos refletem em venda (Masculino = Sim: Venda). O próximo passo do algoritmo é buscar por outro atributo para seguir após "Masculino = Não". A escolha foi "Região" e o algoritmo recursivamente segue testando e escolhendo os melhores nós com a menor entropia.

Ao utilizar classificação podem ocorrer situações onde alguns registros não podem ser classificados pelo fato de não pertencerem à árvore. Para que fique mais claro, ao observar os registros 4 e 6 da tabela 3.6, pode-se notar que ambos possuem os atributos Região = SUL e Masculino = NÃO, porém, o primeiro é uma venda e o outro é uma rejeição.

Para o exemplo ilustrado a escolha foi venda e o registro 6 passa a fazer parte dos registros que não puderam ser classificados. Esse registro possui os mesmo atributos do registro 4, porém, o próximo atributo escolhido é a região e por este motivo, como o registro não é da região sul como as outras vendas não pôde ser classificado. Ao gerar a árvore o algoritmo chegou até a regra que a região sul identifica que o registro é um venda e não uma rejeição.

Os registros com erro são aqueles em que o registro é classificado, porém suas características não o identificam como tal. É o caso do registro 4 que embora seja venda que

não é do sexo masculino, porém faz parte da região das outras vendas (SUL). Dessa maneira ficou classificado, mas classificado como erro.

3.2.2 Algoritmo C4.5

Em 1993, QUINLAN (1993) apresentou uma inovação do Algoritmo ID3, utilizando poda da árvore. Essa poda tem o objetivo de remover ramos da árvore, que é chamado de sobreajustamento, que pode significar que a árvore ficou mais complexa do que deveria ser.

No C4.5 é utilizado a abordagem de "dividir para conquistar" (TRADUÇÃO NOSSA) (MITCHEL, apud MARTINHAGO, 2005, p. 48). Nesse tipo de abordagem o problema original é divido em partes menores semelhantes ao original, recursivamente vão sendo resolvidos e suas soluções formarão uma combinação para o problema inicial (CORMEN et al., 2002). Como passos do algoritmo, conforme MARTINHAGO (2005) devem-se:

- Escolher um atributo;

- Adicionar um ramo para cada atributo;

- Passar os exemplos para as folhas (levando em conta o atributo escolhido);

- Para cada nó folha (se forem da mesma classe) associar a classe ao nó folha, se não, repetir os passos anteriores.

O algoritmo, de forma recursiva, vai dividindo o conjunto de treinamento até que resulte apenas uma classe em cada subconjunto. Ao invés de gerar um ramo que possui mais de uma possibilidade, é feita a poda para diminuir esse ramo em apenas uma folha. Esse é o princípio da poda da árvore, transformar ramos em folhas.

O C4.5 realiza esses cortes baseado em métodos estatísticos com base nos erros do nós e seus descendentes. Para identificar a raiz e seus descendentes são realizados os cálculos da entropia e do ganho de informação. A entropia, já utilizada no ID3, mede o quanto maior é a capacidade de previsão. Dessa maneira, podem ser notadas 2 árvores distintas:

-Árvore sem poda gerada pelo algoritmo ID3;

-Árvore com poda gerada pelo algoritmo C4.5, onde o mesmo decidiu por cortar o ramo do atributo "Masculino" com valor igual a não e transformá-lo em uma folha.

A figura 3.7, apresenta as duas árvores para verificar a utilização da poda:

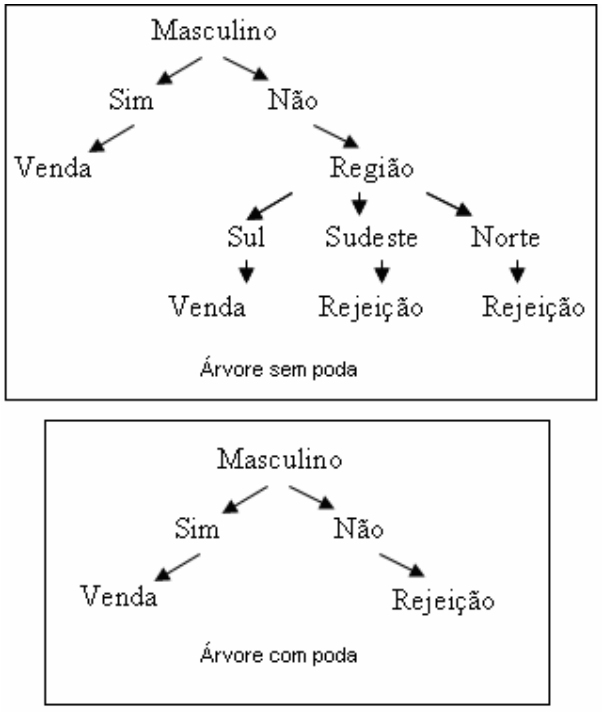

Figura 3.7 – Árvore sem poda e árvore com poda
Fonte: FIGURA NOSSA

O ganho de informação vai medir a redução da entropia nas várias partições dos exemplos de treinamento, de acordo com o valor de um atributo (QUINLAN, 1996) e, dessa maneira, consiga gerar árvores com profundidade e menos nós. A figura 3.8, representa a expressão do ganho de informação.

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

Figura 3.8 – Expressão do Ganho de Informação
Fonte: Adaptado de LÓPEZ; HERRERO, 2004

Nessa expressão, A é o atributo e o subconjunto de S é aquele em que o atributo A tem valor v. Dessa forma, Sv são os casos em cada classe. O ganho de informação mede a eficácia de um atributo nos dados de treinamento, a escolha do atributo que mais reduz a entropia faz com que se gerem árvores com menos nós e ramificações.

O algoritmo C4.5, possui ainda a capacidade de melhorar a estimativa do erro, fazendo a validação cruzada com dois ou mais grupos, chamada de *v-fold*. Além disso, é

possível trabalhar com valores contínuos ou indisponíveis (FERNÁNDEZ, 2004). Na figura 3.9, é apresentado o pseudocódigo do algoritmo C4.5.

```
Função C45
            (R: conjunto de atributos não classificadores,
            C: atributo classificador,
            S: conjunto de treinamento) retorna árvore de descisão;
Inicio
            Se S está vazio,
                  retornar um único nó com valor Falha;
            Se todos os registros de S têm o mesmo valor para o atributo classificador,
                  retornar um único no com esse valor;
            Se R está vazio,
                  retornar um único nó com o valor mais freqüente do atributo
                  classificador dos registros de S;
            Se R não está vazio,
                  D = atributo com maior valor de ganho(D,S) dos atributos de R;
                  Sejam {dj | j=1,2,..., m} os valores do atributo D;
                  Sejam {dj | j=1,2,..., m} os subconjuntos de S correspondentes aos
valores de dj respectivamente;
                  Devolver uma árvore com a raiz nomeada como D, com arcos
nomeados (d1, d2,..., dm) que vão respectivamente às árvores
                  C4.5(R-{D}, C, S1), C4.5(R-{D}, C, S2), C4.5(R-{D}, C, Sm);
Fim
```

Figura 3.9 – Pseudocódigo algoritmo C4.5
Fonte: Baseado em FERNÁNDEZ, 2004, p. 9

O algoritmo parte de um conjunto de treinamento (S) e um atributo classificador (C). Para ilustrar o exemplo, pode-se imaginar que o atributo (C) escolhido no conjunto de treinamento da Tabela 3.6 seja MASCULINO. Se o conjunto de treinamento estiver nulo, não é possível continuar a execução do algoritmo e nesse caso, será retornada uma falha. Se todos os registros possuírem o atributo MASCULINO = SIM, nesse caso não há o que fazer senão retornar um nó MASCULINO = SIM, que foi o caso da figura 3.6, onde todos os atributos deste tipo eram VENDA.

Se não existem atributos que não sejam o classificador, então deve ser retornado um nó com o valor do classificador que mais aparece nos registros. Caso contrário deve ser aplicada a fórmula e gerado o ganho de informação dos atributos não classificadores, o maior valor é retornado como a raiz da árvore, da mesma maneira da escolha do atributo REGIÃO (figura 3.6) e com os arcos SUL, SUDESTE e NORTE, que são seus subconjuntos.

3.2.3 Algoritmo J48.J48

Para que o usuário possa utilizar e gerar as árvores de decisão no *software* Weka, é necessário que utilize o algoritmo J48. O Weka possui uma implementação da última versão pública da família do C4.5, que no caso é a *release* 8. Após esta, foi lançada a versão comercial do C5.0. Essa classe gera as árvores de C4 com ou sem poda.

Os algoritmos dentro do Weka, são organizados em pacotes e no caso do J48, estão localizados em *weka.classifiers.j48.J48*.

O Weka permite que estes algoritmos sejam executados em linha de comando e dessa forma, para executar o algoritmo J48 deve-se utilizar a seguinte linha de comando:

java weka.classfiers.j48.J48 –t arquivo.arff

Essa linha fará a invocação da *Java Virtual Machine* (JVM) e segundo WEKA (2007), através do parâmetro –t é indicado qual conjunto de treinamento será utilizado. A classe J48 na verdade, não possui as rotinas para a geração de árvores, mas inclui as instâncias de outras classes. O pacote j48 é o que possui as classes que executam o J4.8. Na figura 3.10, pode-se observar os parâmetros do algoritmo no Weka.

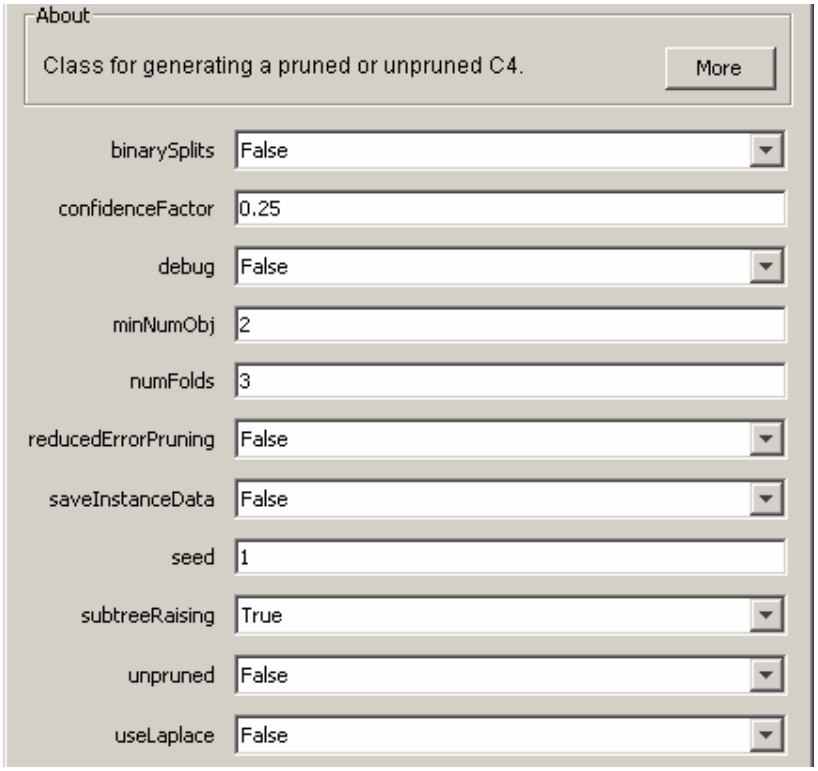

Figura 3.10 – Tela de configuração do J48 no Weka
Fonte: WEKA, 2007

Os parâmetros da figura 3.10 representam:

binarySplits – informa ao algoritmo se é possível a geração de árvores binárias;

confidenceFactor – no caso da utilização de poda, quanto menor o valor de confiança, maior será a poda;

debug – essa opção serve exclusivamente para gerar informações sobre a execução do algoritmo que não são mostradas enquanto estiver marcada como falso;

numMinObj – representa o número mínimo de instâncias por cada folha;

numFolds – está diretamente ligado com a redução de erros na poda. Uma dobra (ou *Fold*) é utilizada para a poda e o restante serve para crescer a árvore (WEKA, 2007). Cada dobra representa a execução do algoritmo. A cada execução o resultado vai sendo refinado, testado e comprovada sua qualidade em qualquer árvore;

reducedErrorPruning – ativa ou não a redução de erros na poda. É realizada a poda através de um conjunto de validação;

saveInstanceData – salva os dados de treinamento para visualização;

seed – esse parâmetro é utilizado, quando ativo, para a redução de erros na geração das árvores, testando-a através de dados gerados aleatoriamente. Cada semente utilizada realiza a mistura dos dados para serem processados. Como a mistura é aleatória, árvores diferentes são testadas, aprimorando e reduzindo erros de classificação;

subTreeRaising – utilizar uma operação de substituir um nó interno da árvore por um dos nós que estão abaixo (a escolha do nó abaixo é feita através da taxa de erros) e logo após classificar novamente a árvores. Esta é uma ação utilizada na poda;

unpruned – não utilizar poda;

useLaplace – conta o número de folhas excluídas baseadas na transformada de Laplace (WEKA, 2007), "que é um método simples para transformar um Problema com Valores Iniciais (PVI), em uma equação algébrica, de modo a obter uma solução deste PVI de uma forma indireta, sem o cálculo de integrais e derivadas para obter a solução geral da Equação Diferencial" (TRADUÇÃO NOSSA) (SODRÉ, 2003, p. 1).

Além dessas configurações, é possível, através do modo gráfico, marcar e escolher um conjunto de teste. Também se pode ativar a validação cruzada e o percentual de separação,

que determina o tamanho do conjunto de registros que serão separados para a geração dos sucessores e criação de novos classificadores (WEKA, 2007), conforme figura 3.11:

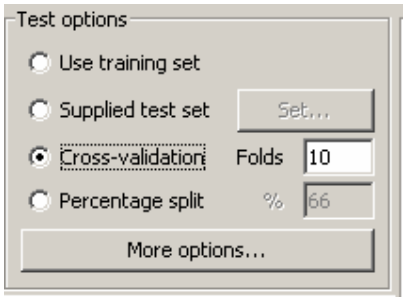

Figura 3.11 – Tela de opções de teste no Weka
Fonte: WEKA, 2007

O Weka permite, para a classificação, observar os erros, visualizar a árvore, entre outras opções utilizando o clique do botão direito na lista de resultados, conforme figura 3.12:

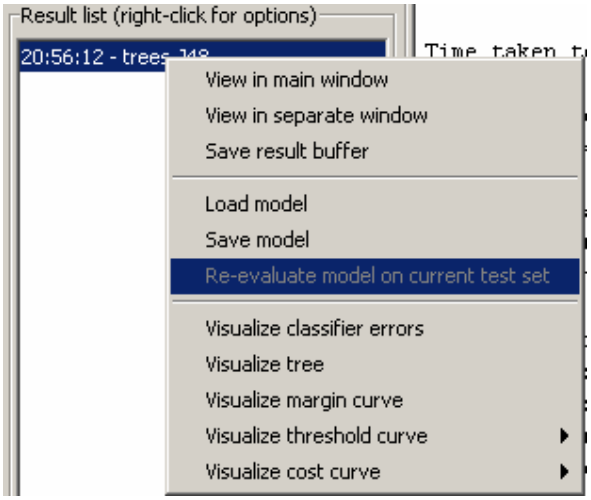

Figura 3.12 – Outras opções de visualização de classificação no Weka
Fonte: WEKA, 2007

Na opção de visualização de árvores, a ferramenta gera a árvore para ser analisada, conforme figura 3.13:

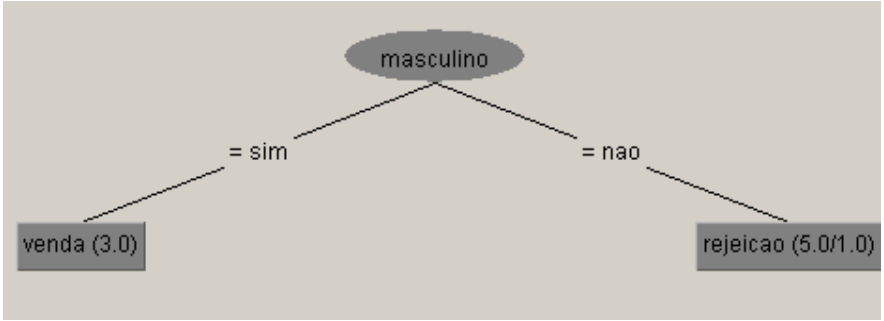

Figura 3.13 – Visualização de árvore
Fonte: WEKA, 2007

Na figura 3.14, pode-se observar o gráfico dos erros de classificação. Nesse exemplo houve um erro que está identificado no gráfico.

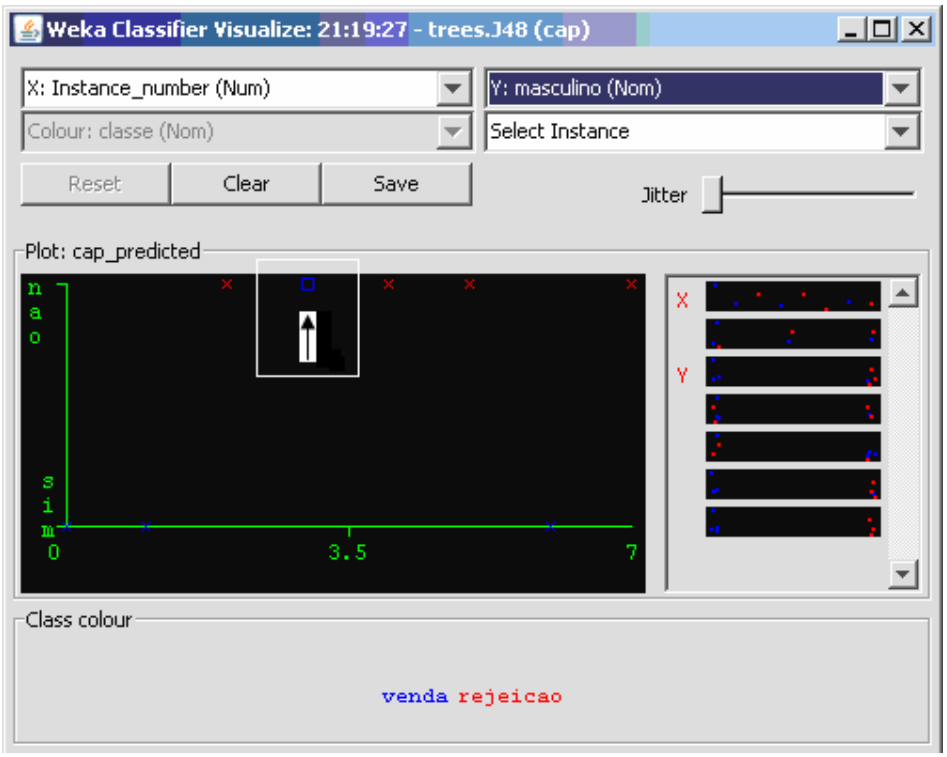

Figura 3.14 – Visualização de erros de classificação
Fonte: WEKA, 2007

Nessa situação, esse registro embora seja de “rejeição” está dentro das características que identificam uma venda. Nesse exemplo, o algoritmo, utilizando poda chegou à conclusão que sempre que MASCULINO = ‘SIM’ então VENDA e no gráfico pode-se visualizar em destaque um caso de VENDA onde MASCULINO = ‘NÃO’, por este motivo o Weka destacou no gráfico essa informação para que o usuário analise.

3.2.4 Algoritmo J48.PART

Conforme já foi introduzido, o algoritmo PART constrói regras de produção a partir da árvore de decisão. Para a geração da lista de decisão, o algoritmo parte de uma árvore já montada e realiza então a indução de regras, que após vão sendo comprovadas ou alteradas. Este algoritmo também atua segundo a abordagem “dividir para conquistar”. Segundo WEKA (2007), a cada iteração é criada uma árvore de forma parcial e transformando a melhor folha (maior ganho de informação) em uma regra.

Para utilizar o algoritmo via linha de comando, J48 deve-se utilizar a seguinte linha:

java weka.classfiers.j48.PART arquivo.arff

Até que todas as instâncias (registros de base de treinamento) possuam a estimativa de quanto representam em relação aos outros registros da base, o PART executa repetidamente a fim de refinar e selecionar as regras com cobertura de folhas mais alta, sempre em relação à quantidade de registros da base. Para ilustrar a execução do PART, será utilizado o mesmo exemplo da tabela 3.6.

Na execução do algoritmo J48, já tratado anteriormente, pode-se observar na figura 3.13 que o J48 gerou uma árvore com a raiz "MASCULINO" e que se este for 'Sim' é uma venda e se for 'Não' caracteriza uma rejeição.

Na figura 3.15, pode-se observar a lista de regras de decisão geradas a partir do exemplo. A lista de decisão do algoritmo PART gerou duas regras:

```
masculino = nao: rejeicao (5.0/1.0)

: venda (3.0)

Number of Rules  :      2
```

Figura 3.15 – Lista de regras geradas pelo J48.PART
Fonte: WEKA, 2007

Nesse exemplo se chegou até duas regras que compõem a lista de decisão. A primeira regra informa que quando o SEXO for 'Masculino' esse cliente será uma rejeição, pode-se ainda observar que dos 5 registros classificados como rejeição, apenas 1 deles não era do sexo masculino. A segunda regra informa ao usuário que em qualquer outro caso seria venda, ou seja, o resultado diz que o atributo sexo é fundamental para classificar esse conjunto de registros.

A característica do PART, que é gerar regras a partir de árvores e transformar o melhor nó em uma regra, agora pode ser observada. Nesse exemplo, o algoritmo J48.PART partiu da árvore gerada como no J48 (até então, ambos os algoritmos são iguais) e selecionou a folha VENDA e a transformou em uma regra, em que todo o registro que não se enquadrar na primeira regra deve ser classificado como venda.

Da mesma maneira do J48.J48, este algoritmo pode ser utilizado através da tela *Weka Explorer*, porém o PART possui menos parâmetros que podem ser vistos na figura 3.16. A utilização dos parâmetros é a mesma do outro algoritmo.

Segue figura 3.16, com os parâmetros do algoritmo J48.PART:

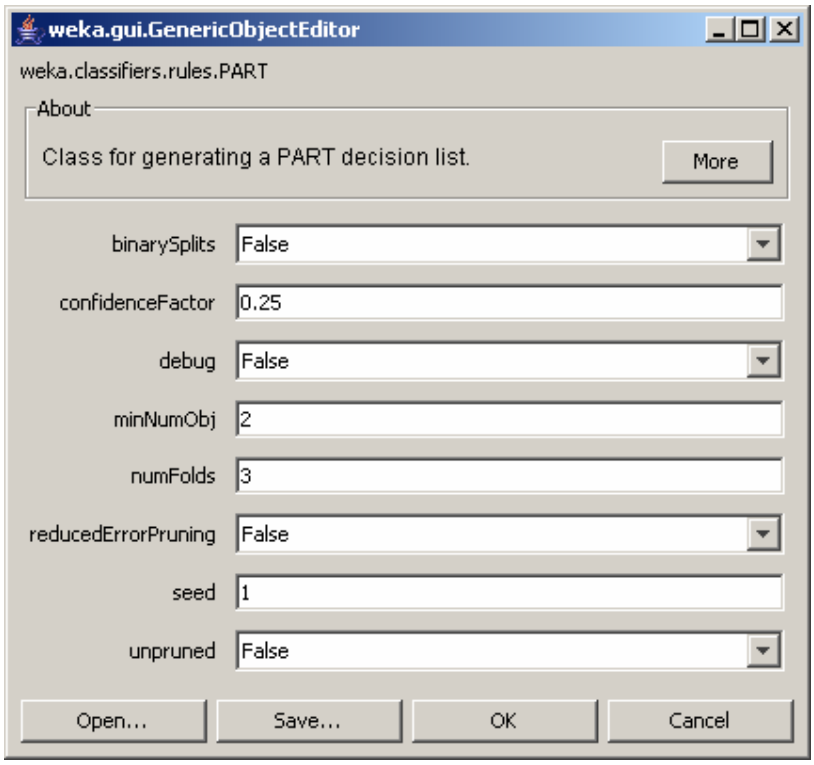

Figura 3.16 – Parâmetros do J48.PART
Fonte: WEKA, 2007

Juntamente com o J48 e J48.PART podem ser utilizados métodos para aumentar desempenho, como os métodos *Bagging* e *Boosting*.

3.2.5 Método Bagging e Boosting

O Weka possui métodos de meta aprendizagem. Estes podem ser utilizados na tarefa de construir conjuntos de classificadores (OLIVEIRA et al., 2002). Todos os métodos de meta aprendizagem podem ser selecionados no Weka através do pacote "*classifiers*" no item "meta", disponíveis na ferramenta junto com os algoritmos de classificação. Essas classes têm a função de aumentar o desempenho e capacidade da geração de regras, por este motivo são incorporados a outros algoritmos de aprendizagem (WEKA, 2007). Para este estudo, foram destacados apenas dois métodos: *Bagging* e *Boosting*, utilizados também em outros trabalhos como OLIVEIRA et al. (2002), por exemplo.

Bagging é o nome dado a um procedimento que constrói classificadores partindo de conjuntos de amostras que são processadas de forma independente e sucessiva, onde se pode definir o número de interações, sendo que o padrão são 10 *Bagging interations*. Em outras

palavras, o *Bagging* se diferencia da utilização do número de sementes do J48 pelo fato que o conjunto de reamostragem é gerado com número de sementes definidas de maneira aleatória, onde as ocorrências de repetição são substituídas (OLIVEIRA et al., 2002) (WEKA, 2007).

Para utilizar *Bagging* no algoritmo J48, por exemplo, pode-se utilizar o Weka selecionando o *Bagging* dentre os métodos "meta" ou então através de linha de comando no sistema operacional, da seguinte maneira:

java weka.classifiers.bagging –W jaws.classifiers.j48.J48...-- -U

Dessa maneira, todas as opções do Bagging e J48 podem ser utilizadas, sendo separadas por "--", "-W" do *Bagging* e "–U" do J48, sem que exista conflito entre as opções de cada um.

No método *Boosting*, cada instância gerada recebe um peso, sendo que inicialmente todas possuem o mesmo. Ao ser feita a primeira indução, os pesos daquelas que foram definidas como erros são alterados. Essa comparação e definição de erros são realizadas com base nos classificadores que já foram construídos (OLIVEIRA et al., 2002). Dessa maneira, aqueles conjuntos de amostras, que acabam sendo identificados com erros, são separados e como resultado se obtém os conjuntos de amostras que conseguiram ter o melhor aproveitamento nas interações.

3.3 Algoritmos de *Clustering*

O Weka possui algoritmos de *Clustering* para procurar grupos de instâncias que sejam similares dentro da base de dados, embora tenha seu foco direcionado à classificação. Dos geradores de *cluster*, presentes na ferramenta podem ser citados: k-Means, EM, Cobweb, X-means, FarthestFirst.

Para está técnica, pode-se destacar o algoritmo SimpleKMeans que assim como os outros algoritmos presentes no Weka, pode ser aplicado sobre um conjunto de treinamento, que serve para que seja avaliado como o algoritmo se comporta nessa amostra da base de dados. Os algoritmos de *Clustering* funcionam "modelando a distribuição de instâncias probabilisticamente" (TRADUÇÃO NOSSA) (WEKA, 2007, p. 34) e é por este motivo que pode ser interessante verificar os resultados na base de teste.

O SimpleKMeans executa 10 vezes (opção definida por padrão e que pode ser configurada) o algoritmo K-Means para trazer um resultado que pode ser considerado

estatisticamente bom para o usuário (PINHEIRO, 2006). Na figura 3.17, é apresentado o K-Means.

```
1.  Eleger k exemplos que atuam como sementes (k número de clusters);
2.  Para cada exemplo, acrescentar exemplo à classe mais similar;
3.  Calcular o centroide de cada  classe, que passam a ser as novas sementes;
4.  Se não se chega a um critério de convergência (por exemplo, duas
interações não mudam as classificações dos exemplos), voltar a 2.
```

Figura 3.17 – Pseudocódigo algoritmo K-Means
Fonte: Adaptado de LÓPEZ; HERRERO, 2004, p.45

O pacote que contém o algoritmo é *weka.clusterers* e a ferramenta, para os algoritmos que fazem a modelagem de instâncias conforme citado, mostra como as muitas instâncias são atribuídas a cada um dos *clusters*, conforme WEKA (2007). Utilizando os mesmo dados do quadro 3.2 e aplicando o *clustering* com percentual de divisão de 66%, o que significa que esse percentual da base de dados é utilizado para treinamento e o restante para teste. Como opções, número de clusters igual a 2 e 10 *seeds*.

O resultado desse teste é apresentado na figura 3.18:

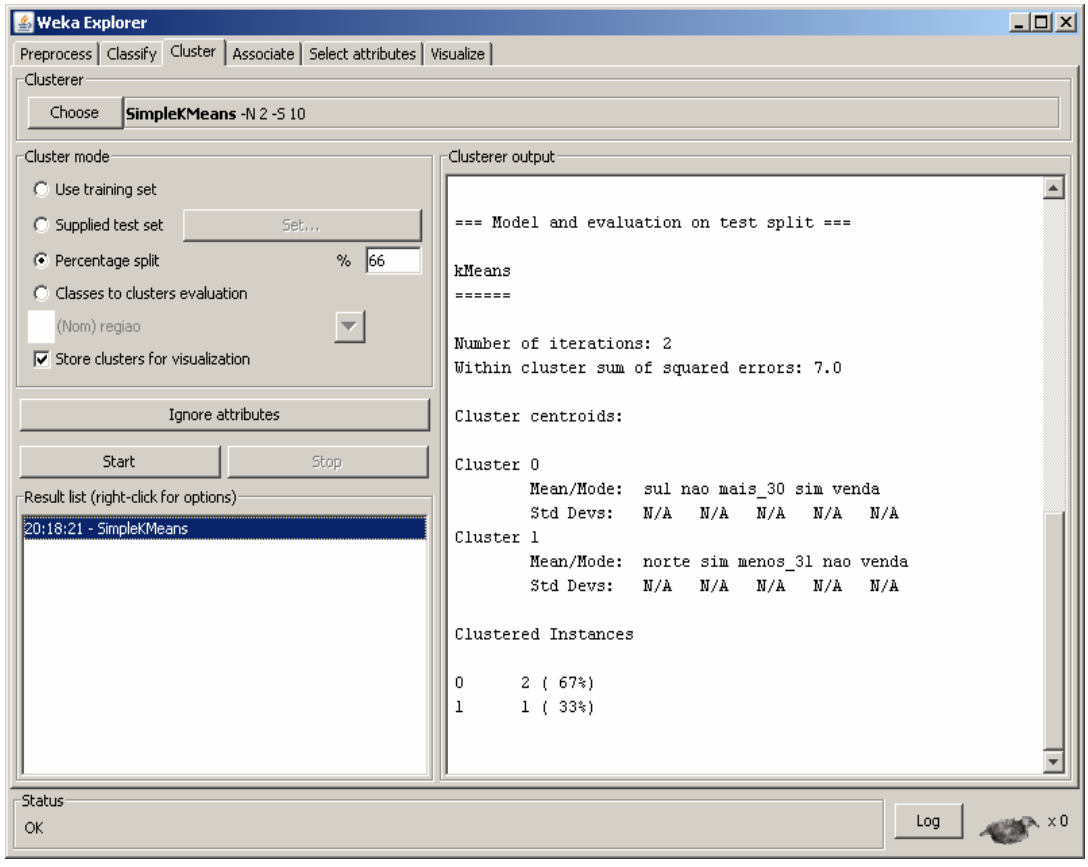

Figura 3.18 – Saída algoritmo SimplekMeans, para o conjunto testado
Fonte: WEKA, 2007

Na figura 3.18, pode-se observar que o algoritmo realizou duas interações, onde a soma dos erros enquadrados totalizou 7.0. São 8 instâncias com 5 atributos:

- REGIAO {sul, sudeste, norte};

- MASCULINO {sim, nao};

- IDADE {mais_30, menos_31};

- GOLD {sim, nao};

- CLASSE {venda, rejeicao};

Dos oito registros, foram encontrados dois *clusters*:

-{sul, não, mais_30, sim, venda}: 2 registros;

-{norte, sim, menos_31, não, venda}: 1 registro;

A partir desse resultado, o usuário pode avaliar o quanto bem o modelo representa os dados. Em um modelo maior, e com a configuração do número de *clusters*, podem-se encontrar as diferentes categorias presentes na base de dados.

4 *SOFTWARE* PARA MINERAÇÃO E ESTUDO DE CASO DO *CALL CENTER*

Para realizar o tratamento e apresentação dos resultados, foi desenvolvido um *software* que faz uma interface entre o usuário, base de dados e o Weka direcionado para o ambiente disponível. Objetivo desse capítulo é estudar o *Call Center*, apresentar a estrutura da empresa e a ferramenta de mineração, quais as suas funções e como utiliza as bibliotecas em Java do Weka.

Ao invés de serem desenvolvidos dois protótipos para o tratamento dos dados e outro para a apresentação dos resultados, estes protótipos foram unidos para disponibilizar apenas uma ferramenta, mais completa e evitar a utilização de duas aplicações para o mesmo fim, a mineração dos dados no *Call Center*. Antes de tratar da aplicação, é importante conhecer a estrutura disponível.

4.1 Estudo de Caso do *Call Center*

Um estudo de caso é uma técnica utilizada para realizar algum tipo de estudo, envolvendo uma pesquisa para conhecer detalhes sobre algo real. A definição para SIMON, apud FELBER (2005, p. 57), especifica que é uma técnica "[...] onde se faz uma pesquisa sobre um caso particular, para tirar conclusões sobre princípios gerais daquele caso específico".

Busca-se estudar o fenômeno da utilização de KDD na área de *telemarketing* e conhecer todo o contexto. De acordo com (YIN, apud FELBER, 2005), é utilizado porque não fica evidente e ainda não se conhece a relação entre contexto (*Call Center*) e fenômeno (KDD), por este motivo que se torna possível utilizar um estudo de caso.

Para utilizar técnicas de descoberta de conhecimento em um *Call Center* ativo, foram utilizados dados reais de uma empresa deste ramo, que atualmente não possui nenhuma ferramenta de garimpagem. Como os dados são confidenciais, estes não contiveram nomes de

clientes, documentos ou qualquer outro dado pessoal. No momento de realizar a validação e comparação de resultados, por não ser autorizado, o *Call Center* não foi identificado. Entretanto, os testes foram realizados de maneira que poderão ser aplicados a qualquer outra empresa de *Call Center*, desde que os dados possam ser disponibilizados de maneira similar e que exista uma forma de priorizar os clientes a serem contatos.

Foi escolhida para este estudo, a venda de títulos de capitalização, e tal escolha se deve ao fato de existir registros de atividades em um período maior de um ano na comercialização deste produto e uma grande quantidade de atributos de cliente, presentes nas bases que são enviadas para a empresa de *telemarketing*. Essa venda de títulos é realizada para clientes que possuem cartão de crédito da empresa contratante do serviço.

4.1.1 Modelagem dos dados

Na lista de *prospects* enviada pela empresa contratante pode-se, antecipadamente, destacar alguns atributos que foram utilizados no processo da descoberta de conhecimento. Na figura 4.1, é apresentada a tabela do banco de dados existente hoje no *Call Center* e onde são armazenados os registros recebidos.

```
Table: MAILING
Column Name                    Data Type
------------------------------ ------------------
NUM_MAILING                    NUMBER (6)
SEQ_MAILING                    NUMBER (6)
NOME_CLIENTE                   VARCHAR2 (50)
CPF                            VARCHAR2 (16)
DDD1                           NUMBER (4)
FONE1                          NUMBER (8)
CLASSIFICACAO                  VARCHAR2 (20)
DATA_NASCIMENTO                DATE (7)
DATA_ABERT_CONTA               DATE (7)
DIA_VENC_FATURA                NUMBER (2)
VALOR_LIMITE                   NUMBER (9,2)
COD_REGIAO_MAILING             NUMBER (2)
COD_CLASSIFICACAO              NUMBER (3)
COD_FAIXA_ETARIA               NUMBER (3)
COD_TEMPO_CONTA                NUMBER (3)
COD_VALOR_LIMITE               NUMBER (3)
```

Figura 4.1 – Resultado do comando DESC de uma tabela de *Mailing*
Fonte: FIGURA NOSSA

Os atributos pré-selecionados foram: região onde o cliente mora, classificação para o contratante (cliente *Gold*, Internacional, etc), seu valor de limite no cartão de crédito, tempo de fidelidade ao contratante, data de vencimento da fatura e faixa etária.

Para se obter uma melhor organização das informações, foram utilizadas as tabelas de cadastro de cada uma das características selecionadas. Dessa maneira pode-se ter controle sobre cada um dos atributos e todos os possíveis valores. Por exemplo, existe uma tabela chamada FAIXAS_ETARIAS para que se cadastrem as faixas de idade que são utilizadas. Os campos da figura 4.1 iniciados por "COD_", identificam que essa coluna é uma chave estrangeira para outra tabela.

Como os dados são tratados de maneira diferente pelo cliente, na tabela de *mailing*, existem colunas para os dados originais e colunas para realizar a inclusão dos códigos, segundo as tabelas. Por exemplo, o contratante envia apenas a data de nascimento de cada *prospect* e por isso deve é feito o cálculo e definida sua faixa etária. Na figura 4.2, é apresentado o Modelo Entidade Relacionamento (MER) para os dados utilizados durante a aplicação de KDD:

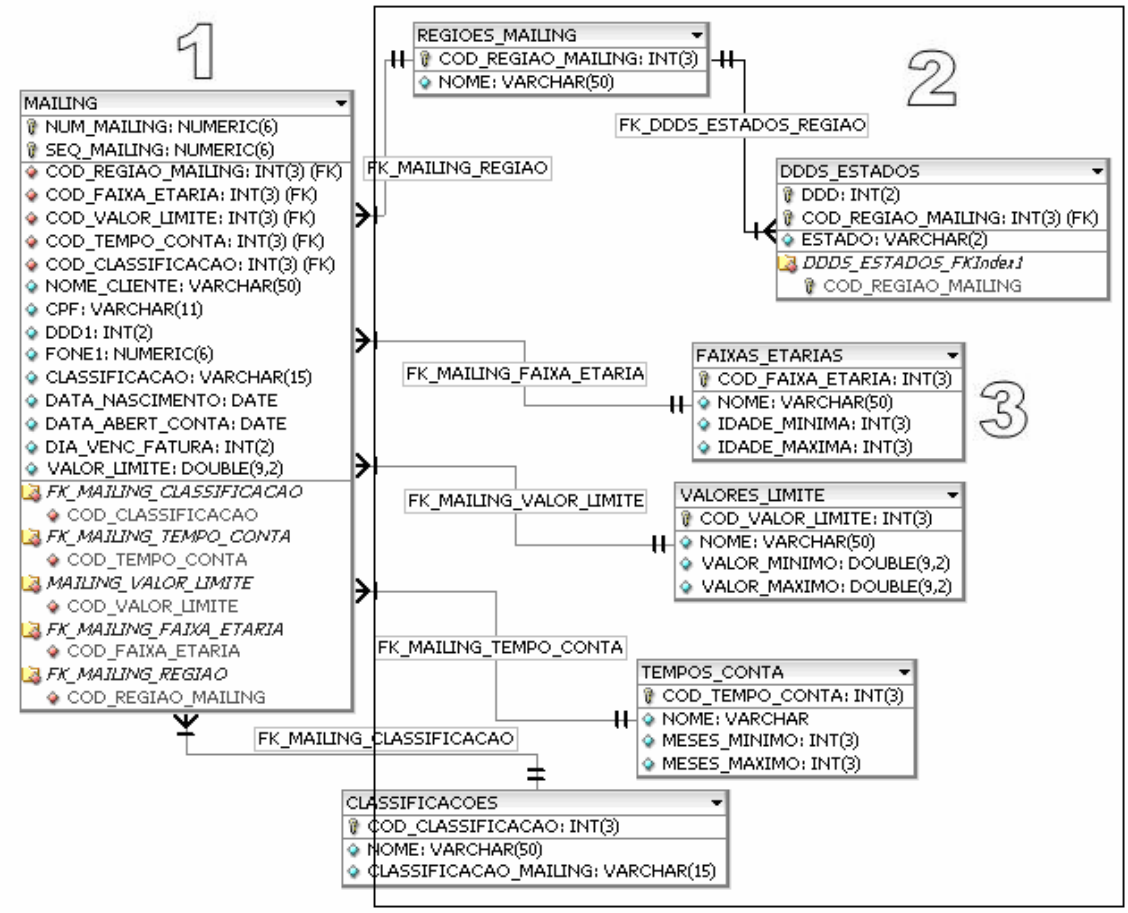

Figura 4.2 – MER dos dados de vendas de títulos de capitalização
Fonte: Adaptado de DBDESIGNER 4 (dbdesigner, 2007)

O objeto MAILING, indicado por (1), possui os dados recebidos pelo *Call Center*. Os objetos CLASSIFICACOES, TEMPOS_CONTA, VALORES_LIMITE, FAIXAS ETARIAS e REGIOES_MAILING (2) possuem os cadastros dos tipos de cada atributo que foram utilizados. Por exemplo, na tabela FAIXAS ETARIAS (3), possuem todas as faixas etárias desejadas. Na figura 4.3 pode-se observar os dados da tabela de faixas etárias:

COD_FAIXA_ETARIA	NOME	IDADE_MINIMA	IDADE_MAXIMA
1	NÃO INFORMADO	0	0
2	18 A 29 ANOS	1	29
3	30 A 39 ANOS	30	39
4	40 A 49 ANOS	40	49
5	50 A 59 ANOS	50	59
6	ACIMA DE 60	60	999

Figura 4.3 – Registros da tabela de faixas etárias
Fonte: FIGURA NOSSA

Por questões de desempenho de consultas ao banco de dados e relacionamento de 1 para N entre as tabelas é armazenada uma chave estrangeira na tabela de *mailing* com a respectiva faixa etária do momento da inclusão de um novo cliente. Cada faixa, tempo de conta, classificação e região poderá estar incluído em 1 ou mais clientes e o respectivo campo na tabela *mailing* não pode ser nulo.

A mesma lógica é utilizada para os outros objetos, com diferença para o tempo de conta que utiliza a data de abertura de conta, valor limite que é definido através do valor de limite, da classificação que é através do campo de texto recebido com a descrição da classificação do cliente e a região que é definia através do DDD do telefone do cliente com o auxílio da tabelas de DDD por cada estado. Ainda é possível utilizar o dia de vencimento da fatura, nesse caso com valores que são entre 1 e 30.

Para a aplicação de KDD na venda de títulos de capitalização, foi utilizado este modelo e para a criação do arquivo arff do weka foram utilizados os seguintes atributos, conforme quadro 4.1:

Quadro 4.1 – Atributos utilizados no arquivo arff

```
@attribute COD_REGIAO_MAILING {CENTRO- OESTE,MG, NORDESTE_I, NORDESTE_II, NORTE, PR,
RJ-ES, RS, SC, SP_CAPITAL, SP_INTERIOR}
@attribute COD_CLASSIFICACAO {EXCLUSIVO, INSTITUCIONAL, NÃO_INFORMADO, PLATINUM,
CLASS, UNICO}
@attribute DIA_VENC_FATURA     real
@attribute COD_TEMPO_CONTA {ACIMA_DE_5_ANOS, DE_0_A_6_MESES, DE_1_A_2_ANOS,
DE_2_A_3_ANOS, DE_3_A_4_ANOS,DE_4_A_5_ANOS,DE_7_A_12_MESES,NÃO_INFORMADO}
@attribute COD_FAIXA_ETARIA     {18_A_29_ANOS, 30_A_39_ANOS, 40_A_49_ANOS,
50_A_59_ANOS, ACIMA_DE_60, NÃO_INFORMADO}
@attribute COD_VALOR_LIMITE {A_PARTIR_5000, DE_1000_À_1999, DE_200_À_299,
DE_2000_À_4999,
DE_300_À_399,DE_400_À_799,DE_5000_A_9999,DE_800_À_999,NÃO_INFORMADO,10000_A_24999,25
000_A_49999}
@attribute COD_SEXO_ATB        {MASCULINO,FEMININO,NÃO_INFORMADO}
@attribute VENDA               {SIM, NAO}
```

Fonte: Do autor

4.1.2 Técnicas de KDD utilizadas

A técnica de KDD utilizada é a classificação que foi escolhida para gerar árvores de decisão, a fim de encontrar os atributos dos clientes que os fazem adquirirem os títulos ou então recusarem a proposta, montando, desta forma, perfis de compra. Além de encontrar as características determinantes para a compra, pode-se evitar que sejam contatados clientes com as características que levam à rejeição do produto.

Para encontrar os melhores resultados de classificação, foram feitos comparativos entre os resultados do J48.J48 e J48.PART. Os métodos de *Bagging* e *Boosting* foram utilizados para aprimorar e refinar as regras, assim como utilizado em (OLIVEIRA et al., 2002).

As regras e informações produzidas podem ser repassadas ao setor responsável pelo gerenciamento das bases de clientes e desta maneira foi realizado o acompanhamento dos resultados durante a aplicação dos testes. Dessa maneira, esses responsáveis podem verificar o conhecimento gerado e testá-lo diretamente na produção da empresa.

4.1.3 Validação

A validação dos resultados foi realizada de duas maneiras, utilizando os registros anteriores à data do conjunto de treinamento, para testar a regras em um conjunto maior de dados e como outra forma de validação aplicar as regras geradas diretamente na empresa, para selecionar os *prospects* que serão disponibilizados para contato para equipe de *telemarketing*.

Pelo fato de existir uma grande base de dados das transações que já foram realizadas, foi possível que fosse feita uma verificação das regras, conferindo se os registros que não faziam parte do conjunto de treinamento também podem ser classificados dentro das mesmas regras. O conjunto de treinamento pode ser aumentado para refinar o resultado.

O setor de análise de informações da empresa onde foi utilizado KDD pôde dar suporte e informações referentes aos perfis de clientes que hoje em dia recebem um foco maior. Na figura de especialistas da área de *telemarketing* podem analisar as regras geradas e definir se alguma informação não é aplicável e assim foi realizado.

Esse mesmo setor possui ferramentas para direcionar para os atendentes os clientes com o perfil definido na regras. No momento que os registros das listas de *mailing* entrarem em produção, já se pode começar a colher informações do desempenho que as regras estão tendo.

Para finalizar a validação foi realizada uma comparação do desempenho da produção com KDD e no método utilizado hoje. Para tornar a validação mais eficiente, foi realizada uma comparação do número de contatos necessários para se chegar a um total de vendas, primeiramente analisando a forma utilizada atualmente para distribuir os nomes, e depois a quantidade de contatos necessários para o mesmo número de venda utilizando KDD. Para obter esses números ao utilizar mineração, somam-se todos os contatos menos àqueles que não fazem parte do perfil gerado pelo processo de garimpagem.

Para analisar os resultados da mineração com cada algoritmo, serão realizadas as seguintes anotações em relação aos resultados:

-Registrar a árvore gerada assim como é gerada no Weka;

-Quantidade e percentual de registros classificados correta e incorretamente;

-Matriz de confusão gerada;

-Opcionalmente, registrar a quantidade de folhas e o tamanho da árvore;

4.2 *Software* para tratamento e apresentação dos resultados

O *software* tem a função de possibilitar que o usuário possa selecionar os dados, os atributos e aplicar classificação de maneira mais simples, direta e de interpretação mais simples do que utilizar o Weka. Dessa maneira será transparente a criação do arquivo (.arff), o

pré-processamento e a geração do resultado da mineração para que o usuário possa focalizar sua atenção na análise e assimilação do conhecimento gerado.

A aplicação foi desenvolvida em linguagem Java, para que seja possível utilizar as bibliotecas do Weka, criadas na mesma linguagem, sem maiores esforços. Dessa maneira, é possível ter acesso a todos os algoritmos de classificação presentes na interface gráfica do Weka, da mesma maneira que seria possível utilizá-los através de linha de comando.

4.2.1 Tecnologia e desenvolvimento da aplicação

O arquivo weka.jar, distribuído juntamente com o *software* Weka, torna possível que o usuário utilize os algoritmos de mineração sem a necessidade de uma aplicação gráfica. Da mesma maneira, possibilita que sejam desenvolvidas aplicações que utilizem a ferramenta Weka.

Para fazer uso da aplicação com Weka, é preciso um ambiente Java J2SE, que é *Java 2 Standard Edition*, com o kit de desenvolvimento (JDK) e não apenas a máquina virtual (JRE) e pode ser utilizado tanto no sistema operacional Windows como Linux. O ambiente de desenvolvimento utilizado foi Eclipse[15], na versão 3.1 para o desenvolvimento das classes utilizadas, juntamente com o Netbeans[16] 5.5, utilizado para o desenvolvimento da interface pela facilidade de utilização.

Ao iniciar um novo projeto no Eclipse, é preciso incluir o Weka nas classes do projeto, basta selecionar a *tab Libraries*, no item de configurações e então adicionar Um arquivo .jar externo, adicionando o weka.jar, do local onde ele estiver armazenado, normalmente onde o Weka foi instalado.

Antes de iniciar o desenvolvimento, se torna importante conhecer algumas classes importantes do Weka (SANTOS, 2007). Essas classes são:

-*FastVector*: que é uma implementação de vetores ou *array* de tamanho que pode ser alterado;

-*Instances*: que é o arquivo arff ou sua estrutura mapeada na memória durante sua execução. Esse mapeamento inclui a relação, atributos e dados;

-*Instance*: possui as informações de uma instância dos dados;

---

[15] http://www.eclipse.org
[16] http://www.netbeans.org

-*Attribute*: possui as informações de um atributo.

Munido dessas informações se torna possível criar a seleção dos dados para gerar o arquivo arff, necessário para executar os algoritmos.

4.2.2 Seleção dos dados

A primeira etapa da mineração é realizar a seleção dos dados. Dentro do *Call Center*, o setor que é responsável por selecionar os clientes que deverão ser disponibilizados aos atendentes é o mesmo setor que possui o perfil para qual essa aplicação foi direcionada. O tipo de usuário para qual o *software* foi desenvolvido possui conhecimentos consolidados na área de *Call Center* e total conhecimento na estrutura utilizada para armazenar os dados no BD.

Relatórios para as empresas contratantes também são gerados e construídos por este setor e por isso possuem acesso para consultadas dentro do BD, bem como conhecimento em SQL para que sejam possíveis consultas rápidas e o desenvolvimento de novos relatórios.

Como esse tipo de usuário é de um nível mais avançado, ele possui condições e conhecimento para realizar a seleção dos dados.

Para que seja possível realizar a seleção de forma que o resultado já esteja ponto para a mineração, o *software* permite que o usuário através de uma consulta em SQL possa escolher quais e quantos dados serão utilizados.

A aplicação permite configurar conexão com o BD através da identificação do *driver* e endereço de conexão no arquivo chamado "config", na pasta raiz onde o *software* estiver instalado. Os bancos de dados disponíveis para conexão são Oracle[17], Mysql[18] e PostgreSQL[19].

Ao iniciar a aplicação, deve ser informado o usuário, a senha e o nome do BD de conexão e depois de conectado, através de botão "Conectar", já é possível executar SQL para realizar a seleção dos dados, conforme indicado na figura 4.4 através de (1).

Para os casos que exista algum problema na conexão, o usuário será notificado para corrigir erros nos dados de conexão ou verificar a disponibilidade do BD, através de uma mensagem na tela informando do problema ocorrido.

---

[17] http://www.oracle.com
[18] http://www.mysql.com
[19] http://www.postgresql.org/

A consulta em SQL deve ser criada dentro do campo de texto correspondente (2) e ao executá-la é gerada uma amostra dos dados, ou seja, o número de linhas selecionado (3).

A figura 4.4 tem o objetivo de apresentar a tela em questão e como são dispostas as informações:

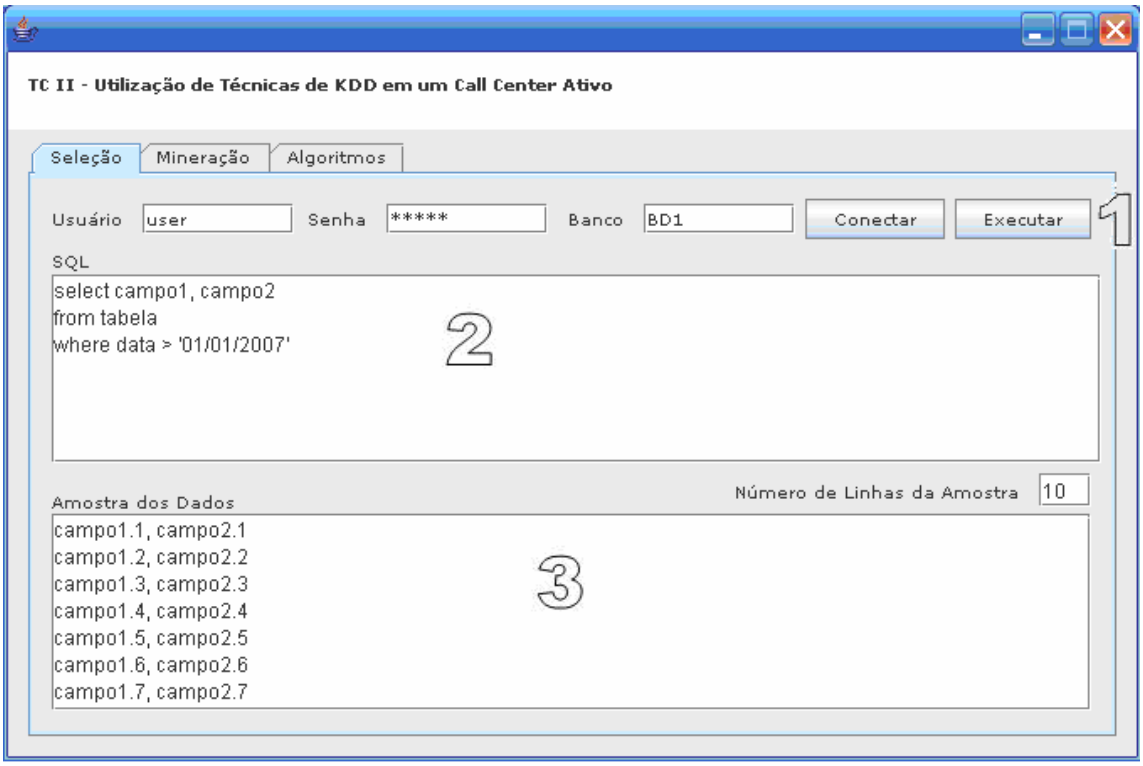

Figura 4.4 – Conexão, seleção e apresentação de amostra da seleção dos dados.
Fonte: Do autor.

A conexão com o BD é feita através de um *pool* de conexão, que é uma técnica de *Connection pooling*, que ao invés de abrir e fechar conexões tantas vezes que sejam necessárias, consumindo recursos e criando várias conexões com o BD, reutiliza as conexões existentes. Dessa maneira, a classe *DBConnectionManager* cuida para que seja verificado se existe uma conexão livre para utilização, caso contrário uma nova conexão é aberta. Para que seu funcionamento seja possível é utilizado um objeto "*instance*", criada de forma estática para que seja instanciada na JVM uma única vez. A figura 4.5, apresenta detalhes da manipulação das instâncias da classe *DBConnectionManager*.

```
package database;

import java.io.*;

public class DBConnectionManager {
    static private DBConnectionManager instance;     // The single instance
    static private int clients;

    private Vector drivers = new Vector();
    private PrintWriter log;
    private Hashtable pools = new Hashtable();

    /**
     * Retorna uma única instância, criando uma se é a
     * primeira vez que o método é invocado
     *
     * @return DBConnectionManager The single instance.
     */
    static synchronized public DBConnectionManager getInstance() {
        if (instance == null) {
            instance = new DBConnectionManager();
        }
        clients++;
        return instance;
    }
```

Figura 4.5 – Objeto instante e método getIntance().
Fonte: Do autor.

O método *getInstance()*, destacado na figura 4.5 é responsável pela verificação se existe outra instância criada, nota-se que este método é estático e como é "*synchronized*", não pode ser invocado mais de uma vez ao mesmo tempo.

A classe *ApplicationManager*, responsável por executar as consultas ao BD, precisa apenas criar um objeto *DBConnectionManager* e criar um bloco estático para buscar um instância da classe de conexão. A figura 4.6, apresenta e chamada do método *getInstance()*.

```
public class ApplicationManager {

    public static DBConnectionManager connMgr;

    //bloco estático
    //quando a classe for caregada pra jvm irá rodar, depois não mais
    static {
        connMgr = DBConnectionManager.getInstance();
    }
```

Figura 4.6 – Classe ApplicationManager e método getIntance().
Fonte: Do autor.

Para executar a *query* do usuário é preciso apenas buscar uma conexão e executá-la, conforme ilustrado na figura 4.7.

```
Connection con = null;
PreparedStatement pstmt = null;
ResultSet rs = null;

con = connMgr.getConnection("Pool");   Buscando a Conexão
pstmt = con.prepareStatement(sql);
rs = pstmt.executeQuery();             Executando a query
```

Figura 4.7 – Execução da *query*, na classe ApplicationManager.
Fonte: Do autor.

Depois dos dados selecionados, é preciso que seja feito o pré-processamento dos dados, onde parte desse processo é feito internamente pelo *software* para apresentar a amostra dos dados.

4.2.3 Pré-processamento

Da mesma maneira que ao utilizar alguma outra ferramenta de mineração, ou até mesmo o Weka, pode se tornar necessário realizar um pré-processamento dos dados. As tarefas de unificação de cada um dos formatos dos dados fica a cargo do usuário que está processando, porém algumas tarefas são realizadas automaticamente.

Os dados são formatados para que estejam todos em fonte maiúscula e que não tenham espaços. Por exemplo, se o dado original é “De 0 a 6 meses“, ele é formatado para “DE_0_a_6_MESES”, para uniformizar o formato e facilitar as próximas etapas de transformação.

Para os registros que possuam algum atributo que não seja informado, o software atribui o valor “NÃO INFORMADO”, a fim de não realizar a limpeza dos dados.

As tarefas de pré-processamento são realizadas de forma transparente ao usuário e tem objetivo de facilitar a criação da SQL da seleção e preparar os dados para a transformação, necessária para utilizar o Weka.

4.2.4 Transformação

O processo de transformação é responsável pela tarefa de modificar o arquivo para que seja possível realizar a mineração. Conforme já apresentado, para que seja possível utilizar o Weka e os algoritmos, é necessário um arquivo arff com toda a sua estrutura. O software desenvolvido, que utiliza as bibliotecas do Weka, necessita do mesmo arquivo.

Para utilizar a interface gráfica do Weka, é necessário que se crie um arquivo arff para ser importado e processado. O objetivo é facilitar o tratamento e a mineração dos dados, para isso, a criação do arquivo é feita internamente e também de modo transparente para evitar o envolvimento da criação de arquivos e importação.

Ao ser executada a SQL, cada um dos campos se torna um atributo. A classe responsável pela criação do arquivo armazena cada um dos possíveis valores presentes de cada atributo, para formar o item "@attribute", obrigatório na estrutura do arquivo. Cada atributo é um objeto *weka.core.Attibute*. Para a criação de atributos numéricos basta criar a linha de código do quadro 4.2:

Quadro 4.2 – Criação de um objeto weka.core.Attribute

```
Attribute x = new Attribute("x");
```

Fonte: Do autor

Para a criação de atributos nominais é preciso utilizar a classe *weka.core.FastVector*. É instanciado um objeto *FastVector*, este objeto possui os métodos *addElement*, com os quais é possível adicionar cada dos valores que o atributo nominal possuirá, por exemplo, no caso do atributo venda, é preciso que sejam adicionados dois elementos, "SIM" e "NAO". Para criar este tipo de objeto serão percorridas todas as linhas selecionadas e os diferentes valores gerados são guardados sem que exista repetição dos mesmos.

Este vetor é convertido em um atributo através de um de seus construtores, que tem como argumentos uma *string* e o próprio objeto *FastVector*, com o código do quadro 4.3:

Quadro 4.3 – Criação de Atributo nominal com Weka

```
FastVector classesNominais = new FastVector(5);
classesNominais.addElement("A");
classesNominais.addElement("B");
classesNominais.addElement("C");
classesNominais.addElement("D");
classesNominais.addElement("E");
Attribute classes = new Attribute("classes", classesNominais);
```

Fonte: Do autor

Cada tupla gerada pela consulta SQL forma uma instância do arquivo arff e cada um dos campos é o valor de cada atributo. O objeto *weka.core.Instances* armazena cada um dos objetos *weka.core.Instance* que possui cada uma das tuplas pré-processadas e transformadas para completar o arquivo arff. Com o arquivo criado, resta iniciar a garimpagem dos dados.

4.2.5 Garimpagem dos dados

O Weka que possui os algoritmos das técnicas de KDD será a ferramenta para gerar as árvores de classificação, utilizadas para serem testadas na empresa de *Call Center*. Para evitar que o usuário tenha que utilizar uma segunda aplicação e para que se tenha acesso às principais funções do Weka, apresentando ao usuário os resultados de uma maneira simples e com as informações necessárias para que possa aplicar os conhecimentos adquiridos, foi desenvolvida esta aplicação.

Os algoritmos e suas opções podem ser cadastrados através de tela própria, dessa maneira além de alterar algum argumento do algoritmo, é possível adicionar outro para ser listado e utilizado pelo usuário. Com acesso ou conhecimento da API do Weka, é possível utilizar qualquer um dos algoritmos de classificação disponíveis. Inicialmente, o software já possui o algoritmo J48, J48 .PART e o método de meta aprendizagem AdaboostM1. Os dados são armazenados em arquivo XML e pode ser atribuída uma descrição ao cadastro. A figura 4.8 apresenta a tela de cadastro de novos algoritmos:

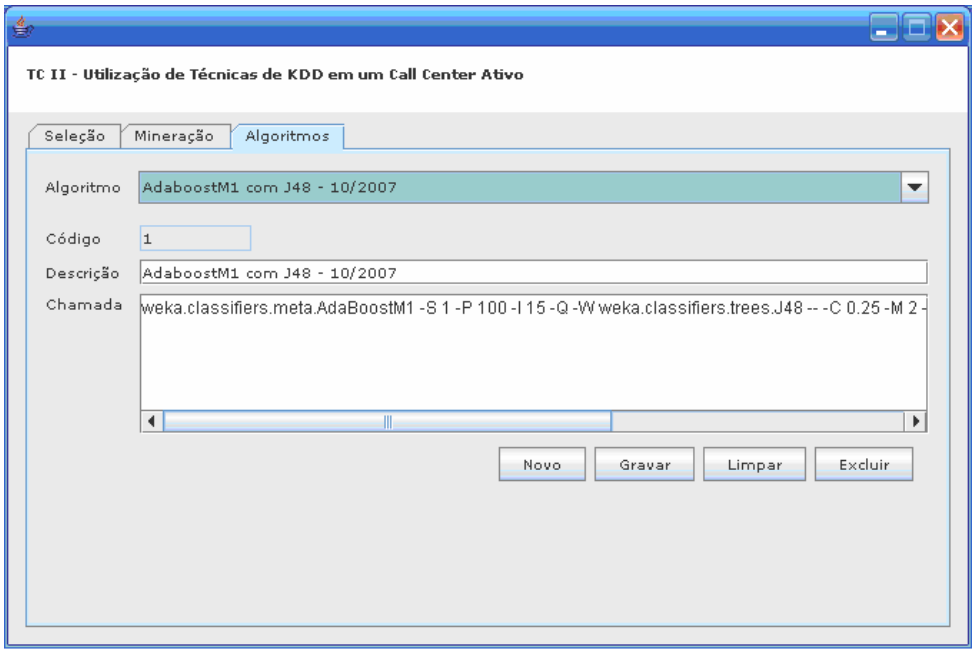

Figura 4.8 – Tela de Cadastro de algoritmos
Fonte: Do autor.

Para a aplicação de um algoritmo é necessário que seja feita a seleção dos dados, após isso, é preciso escolher qual algoritmo da lista será utilizado. Os algoritmos são armazenados em um componente de seleção, marcado na figura 5.3 como (1). A figura 4.9 apresenta a tela de escolha do algoritmo e apresentação dos resultados da mineração:

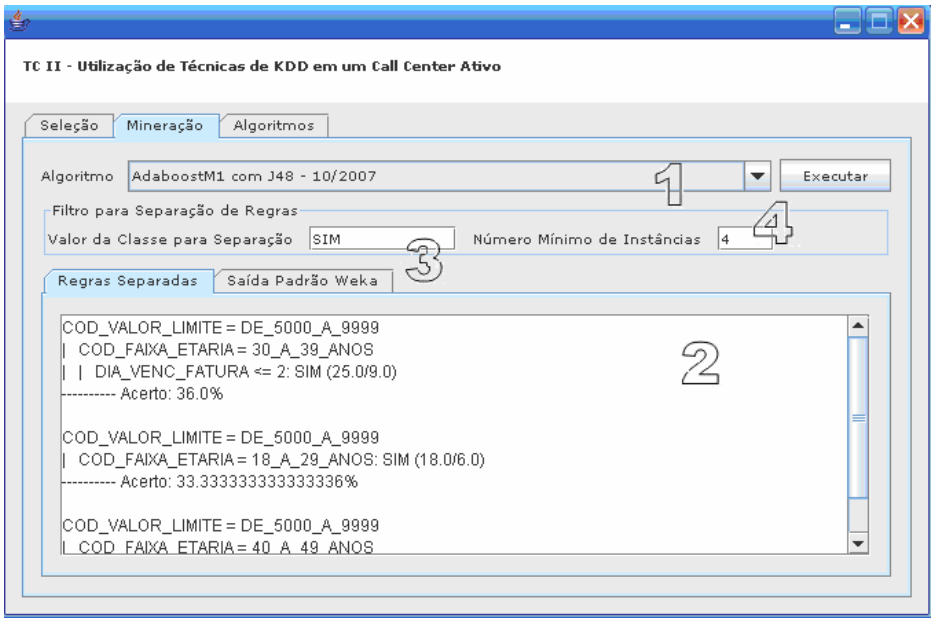

Figura 4.9 – Tela de seleção do algoritmo e apresentação dos resultados
Fonte: Do autor.

Como argumentos da classe responsável por executar o algoritmo de classificação, são necessários: CLASSIFICADOR, FILTRO e ARQUIVO. Da mesma maneira da utilização de linha de comando, o algoritmo J48 pode ser cadastrado através de:

CLASSIFICADOR nome_classe [opções] FILTER nome_classe [opções] DATASET nome_arquivo

O argumento CLASSIFICADOR é utilizado para instanciar a classe *weka.classifiers.Classifier* que se refere ao algoritmo seguido das sua opções. O argumento FILTRO indica a utilização de algum filtro do Weka, presentes no pacote *weka.filters*. O ARQUIVO que é o nome do arquivo, substituído pelo objeto *weka.core.Instances*, criado na etapa de transformação. Para executar o algoritmo, a classe utiliza o método "*buildClassifier*", conforme quadro 4.4:

Quadro 4.4 – Código fonte para executar um classificador do Weka

```
Classifier classificador = null;

Filter filtro = null;

Instances arquivo = null;

classificador = Classifier.forName(nome, opcoes);

Filtro.setInputFormat(arquivo);

Instances arquivo_filtrado = Filter.useFilter(arquivo, filtro);

Classificador.buildClassifier(arquivo_filtrado);
```

Fonte: Do autor

O usuário pode optar em visualizar os resultados no mesmo formato do "*Weka Explorer*" ou então visualizar as regras separadas, marcado na figura 4.6 com (2), conforme os parâmetros da classe pelos quais as regras são separadas (3) e a quantidade mínima de registros classificados (4). Essa visualização é gerada através dos métodos de busca de resultados do classificador e impressão para arquivo ou tela. Na figura 4.10, é apresentado o trecho de código responsável pela impressão da árvore.

```
result.append(m_Classifier.toString() + "\n");
result.append(m_Evaluation.toSummaryString() + "\n");
try {
  result.append(m_Evaluation.toMatrixString() + "\n");
}
catch (Exception e) {
  e.printStackTrace();
}
```

Figura 4.10 – Impressão do resultado da mineração
Fonte: Do autor.

Na figura 4.10, no item marcado com (1), identifica a impressão da árvore gerada com o classificador. No item (2), a impressão do resumo da quantidade de instâncias e no item (3) a matriz de confusão. Para a impressão das informações foi criado um método chamado de "*toString()*" na classe ExecWeka que retorna o objeto "*result*", que posteriormente é transferido para o campo de texto da "Saída Padrão Weka", na aba de Mineração. Para ser feito o filtro das regras, somente a árvore é necessária e por isso o método "toStringArvore()" retorna apenas a árvore gerada na mineração para ser feita a análise e interpretação dos resultados.

4.2.6 Análise e interpretação dos resultados

O usuário precisa fazer a interpretação e a análise dos resultados da mineração, para definir se o resultado pode ser aplicável e se está dentro da qualidade esperada. O resultado filtrado que é o apresentado como primeira opção, separando as regras conforme os parâmetros informados facilitam a análise e a assimilação do conhecimento gerado. Os resultados no formato do Weka é outra opção ao usuário, que pode solicitar sua apresentação na aba correspondente.

O filtro de separação das regras atua diretamente na árvore gerada pelo classificador, para ilustrar essa atuação, na figura 4.11 é apresentada uma árvore de exemplo:

```
COD_VALOR_LIMITE = DE_5000_A_9999
|   COD_FAIXA_ETARIA = 18_A_29_ANOS: SIM (18.0/6.0)
|   COD_FAIXA_ETARIA = 30_A_39_ANOS
|   |   DIA_VENC_FATURA <= 2: SIM (25.0/9.0)
|   |   DIA_VENC_FATURA > 2: NAO (11.0/2.0)
|   COD_FAIXA_ETARIA = 40_A_49_ANOS
|   |   DIA_VENC_FATURA <= 15: NAO (6.0/1.0)
|   |   DIA_VENC_FATURA > 15: SIM (4.0/1.0)
|   COD_FAIXA_ETARIA = 50_A_59_ANOS: NAO (13.0/5.0)
|   COD_FAIXA_ETARIA = ACIMA_DE_60: NAO (8.0/2.0)
|   COD_FAIXA_ETARIA = NÃO_INFORMADO: NAO (0.0)
```

Figura 4.11 – Árvore de exemplo para separação das regras
Fonte: FIGURA NOSSA.

Para fazer a separação das regras é utilizado um vetor para armazenar os níveis que vão sendo lidos enquanto a árvore estiver sendo percorrida, linha a linha.

Cada ocorrência do caractere “|” identifica o nível da árvore que a linha pertence. A falta desse caractere na linha identifica que é a posição 0 do vetor, uma ocorrência identifica que pertence ao nível 1, assim sucessivamente vai sendo preenchido o vetor até que se encontre o valor correspondente ao valor de classe utilizado. Como somente o último valor é armazenado em cada nível, é possível que continue a varredura na árvore.

Na Figura 4.12, é mostrada a separação das regras, sendo marcado em vermelho a primeira regra e em azul a segunda regra.

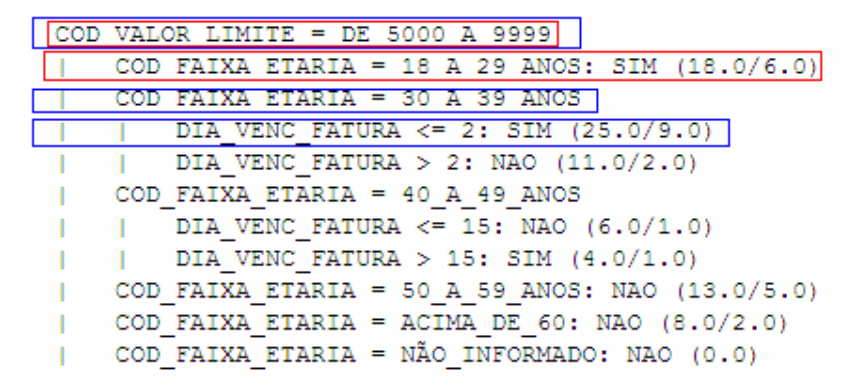

Figura 4.12 – Separação das regras
Fonte: FIGURA NOSSA.

É possível notar que a primeira linha da árvore é utilizada para a primeira e para a segunda regra, enquanto as outras linhas vão sendo armazenadas no índice correspondente, na sua regra. O código responsável por atribuir os valores das linhas percorridas nos índices do vetor é apresentado na figura 4.13. Os índices do vetor que não contém nenhuma informação são desconsiderados. O objeto “linha” representa a linha que está selecionada e “arrArvore” o vetor onde os valores são atribuídos.

```
if (linha.lastIndexOf("|") == -1){
    arrArvore[0] = linha;
}else{
    arrArvore[linha.lastIndexOf("|")+1] = linha;
}
```

Figura 4.13 – Preenchimento do vetor com as linhas selecionadas
Fonte: FIGURA NOSSA.

Ao ser feito o filtro para separar as regras, como a classe utilizada é o atributo VENDA, que possui os valores SIM ou NAO, para separar as regras com o valor de venda SIM e com o número de registros classificados igual a 4, a árvore gerada é filtrada de maneira que só sejam apresentadas ao usuário as folhas correspondentes. A figura 4.14 apresenta as regras separadas pelo *software* desenvolvido:

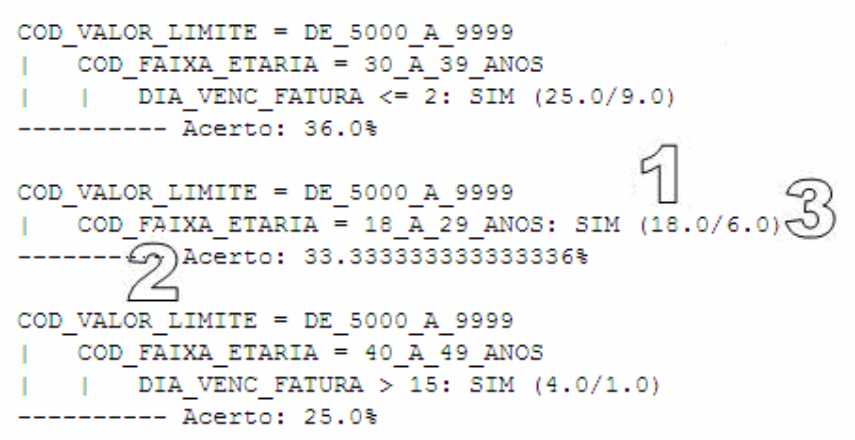

Figura 4.14 – Regras separadas pelo *software* desenvolvido
Fonte: FIGURA NOSSA.

Cada regra é apresentada separadamente, facilitando a interpretação do usuário. Os registros são ordenados pela quantidade de instâncias classificadas dentro da regra, indicado na figura por (1) e o percentual de acerto (2) permite que seja verificado o índice de acerto das instâncias classificadas incorretamente (3) em relação ao total de instâncias classificadas na regra (1).

Dessa maneira, o *software* possui as principais características para facilitar a aplicação da mineração e análise dos resultados, deixando transparente muitas tarefas e realizando a interface entre usuário, banco de dados e Weka.

4.2.7 Contribuições para o projeto Weka

Algumas funções presentes nessa aplicação podem se tornar sugestões para o projeto do Weka, a fim de melhorar a ferramenta e testar se realmente trazem algum benefício para o usuário. Deve-se considerar que estas contribuições são pessoais do autor e não foram discutidas a ponto de serem consideradas unânimes entre o segmento de *Call Center* Ativo ou usuários do Weka.

Como sugestões de contribuições, podem ser sugeridas as funções de seleção, transformação dos dados e criação do arquivo arff dos itens 4.2.2, 4.2.3, 4.2.4. Através dessa proposta, pode-se melhorar a seleção da fonte de dados que hoje é feita de maneira simples, conforme a figura 4.15.

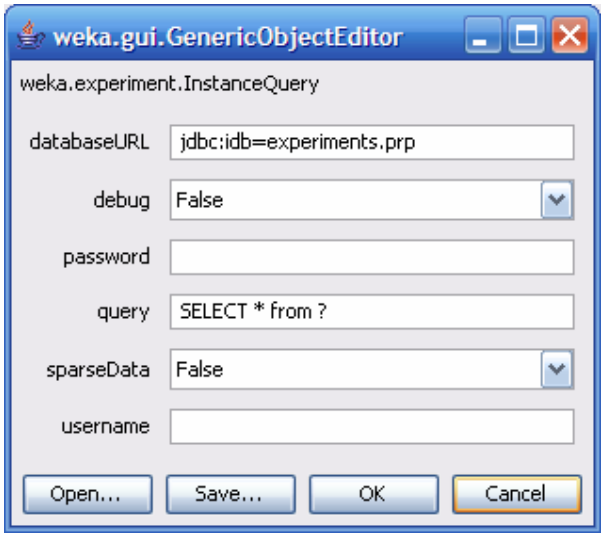

Figura 4.15 – Seleção dos dados de um BD no Weka
Fonte: WEKA, 2007.

Além de possibilitar uma utilização mais intuitiva, a união das duas propostas pode permitir a facilidade de conexão já disponível no Weka com edição da *query* e apresentação das amostras de dados deste trabalho. Ainda pode ser dada uma maior atenção à transformação dos dados, sendo realizado algum tipo de parametrização ou personalização desde o pré-procesamento.

Outra contribuição está na apresentação dos resultados, onde a árvore pode ser filtrada, exibindo ao usuário apenas as informações desejadas, conforme a classe utilizada e o número de instâncias informado. Essa possibilidade de visualizar a saída padrão ou então a árvore filtrada, de certa maneira, pode auxiliar o usuário a definir as categorias de clientes geradas na mineração, economizando o tempo de leitura da árvore. Embora esse ponto seja destacado como sugestão de contribuição, não se deve simplesmente trocar a árvore gerada pela árvore filtrada, pois um equívoco na definição dos parâmetros pode ocultar informações importantes. A visualização da saída padrão do Weka, permite que o usuário possa fazer uma análise do filtro aplicado.

A questão do cadastro dos algoritmos se torna uma maneira de centralizar um histórico e as configurações utilizadas, de maneira que o usuário possa reaproveitar no futuro. Em um ambiente onde está bem definido o tipo de algoritmo e suas opções, torna-se mais simples possuir graficamente um local onde se pode armazenar e utilizar essas configurações, tanto para aplicar em uma nova base como para comparar com as configurações utilizadas em outra base, através da chamada do algoritmo que foi salva no *software*.

Todas são propostas que podem ser discutidas e certamente dependem muito de onde está sendo utilizada a mineração, assim como foi realizado um estudo de caso do ambiente de *Call Center* utilizado para esse *software*.

5 APLICAÇÃO DOS TESTES INICIAIS E DE COMPARAÇÃO DO DESEMPENHO COM MINERAÇÃO

Para a aplicação dos testes, é preciso chegar até um modelo que possa ser utilizado pelo *Call Center* para criar estratégias de vendas e que atinja resultados que possam ser avaliados pelo número de contatos e vendas. O objetivo dos testes deste trabalho é verificar se a aplicação da mineração de dados no *Call Center* ativo consegue melhorar os resultados das vendas.

5.1 Informações importantes para análise dos resultados

Quando em uma árvore de classificação o resultado é do tipo VENDA NÃO (3.0/1.0), significa que dos 3 registros que classificados como NÃO, um deles possuía um outro valor de classe, nesse caso SIM, porque a classe VENDA pode assumir somente os valores SIM ou NÃO.

Ao longo do trabalho serão apresentadas matrizes de confusão. Essas matrizes possuem informações sobre a qualidade dos resultados. Na tabela 5.1 é apresentada um exemplo de matriz de confusão:

Tabela 5.1 – Matriz de Confusão do Weka

	A	B	Classificada como
A	1	2	A = SIM
B	3	4	B = NAO

Fonte: Do autor

Essa representação significa que das três vendas do arquivo (A-1 registro e B-2 registros), duas foram classificadas como VENDA = NÃO. Também pode ser identificado que das sete "não vendas" do arquivo, três delas foram classificadas como VENDA = SIM.

5.2 Estratégias de vendas sem *Data Mining*

O setor da empresa responsável pela análise do *mailing* e por disponibilizar os registros para serem trabalhados já possui uma estratégia. Essa estratégia está centrada na

conversão da base, ou seja, na quantidade de vendas que são realizadas pela quantidade de contato efetivos[20]. Certo período é analisado, o total de registros é contado e os valores de cada atributo são listados, por exemplo, é feita a seleção dos registros do mês inteiro e para cada atributo (Região, Faixa Etária, Tempo de Conta) e para cada valor desse atributo (RS, SC, 18 a 29 anos, 30 a 39, 0 a 6 meses, 1 a 2 anos) é apresentada a quantidade de registros, registros já contatados, taxa de conversão, entre outras informações.

Testando cada um dos atributos, filtrando os registros com o valor desse atributo, é possível chegar ao tipo de atributo com valor de conversão maior. Por exemplo, é testado o atributo "valor limite do cartão de crédito", se ao selecionar somente os registros com valor de limite de 1000 a 1999 se chega ao maior percentual de conversão, esse atributos é considerado na hora de distribuir os nomes para os atendentes entrarem em contato.

Nota-se que essa análise é trabalhosa, pois exige que todas as possibilidades sejam testadas nas 11 regiões, e faixas etárias e 8 faixas de tempos de conta. Nesse caso, alguns testes podem não ser feitos já que se exige que o próprio usuário os faça manualmente e da mesma maneira para testar cada relação de um valor de atributo com outro atributo se torna mais complexo, como por exemplo, a relação da região RS com a faixa etária de 18 a 29 anos e da região RS com o tempo de conta de 0 a 6 meses e com cada uma das faixas etárias.

Na figura 5.1 é apresentada uma parte do *software* utilizado atualmente para conhecer as melhores conversões das bases de clientes. Cada Caixa de seleção (1) é testada marcando-a para incluir na seleção e conforme incluídas vão sendo destacadas (2) se atingirem os valores mínimos configurados internamente pelo próprio setor. Ainda é possível configurar período, lista e informações sobre o turno (3). O total e resumo de toda a seleção feita são apresentados (4) e dessa forma o usuário consegue identificar na comparação visual de cada teste, a categoria de cliente que tem o melhor desempenho. Como se trata de uma ferramenta exclusiva da empresa, não poderá ser apresentado totalmente.

[20] Contato efetivo é o contato foi realizado com o cliente e que se pôde abordá-lo e que foi finalizado como venda ou rejeição.

Segue figura 5.1, com partes do software desenvolvido e utilizado pelo setor responsável pelas estratégias do *Call Center*:

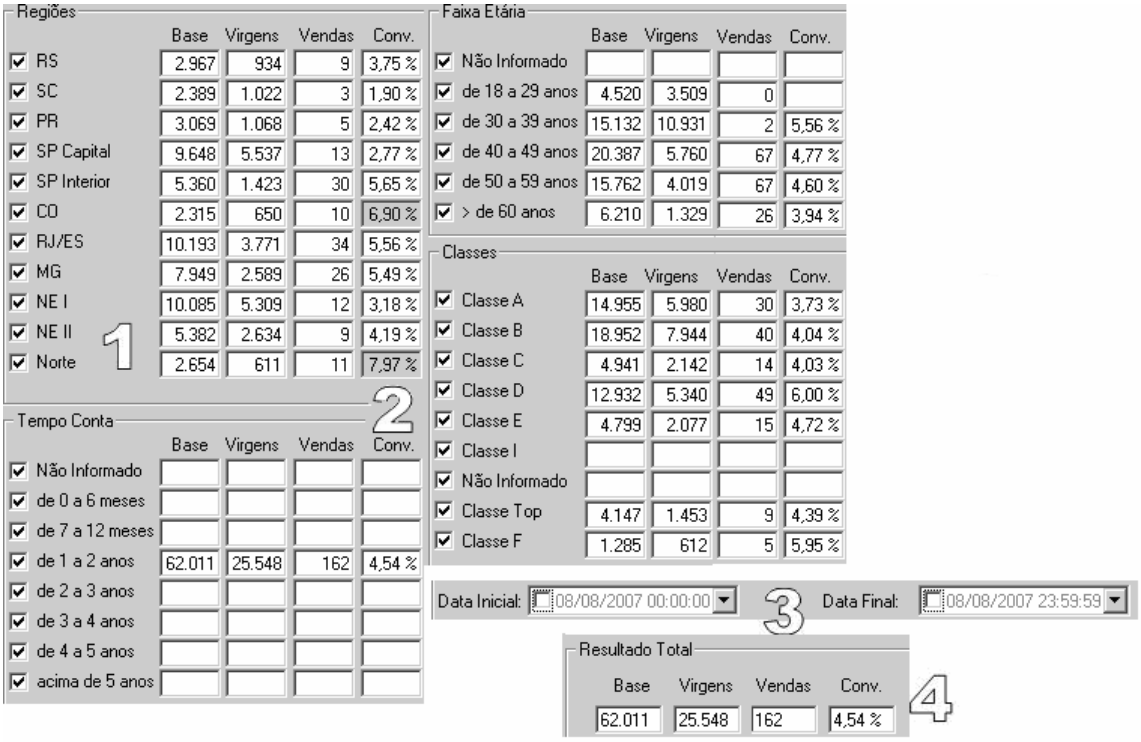

Figura 5.1 – Tela do *software* utilizado atualmente para verificar a conversão das listas de clientes
Fonte: FIGURA NOSSA.

Com base, na experiência do setor e de suas análises, atualmente o *Call Center* tem como melhores clientes aqueles que possuem valor de tempo de conta de 0 a 6 meses e com faixa etária maior de 40 anos. Dessa maneira, os primeiros clientes a serem contatados são os que possuem o tempo de conta nessa faixa e idade maior que aquela conhecida como mais propensa à compra. Caso seja necessário aumentar a quantidade de títulos de certo valor é dado foco nos clientes conforme seu valor de limite, ou seja, se o título que se deseja vender possui valor alto, são direcionados apenas a clientes que possuem limite mais alto.

5.3 Testes iniciais com os dados do *Call Center*

Os testes iniciais têm o objetivo de conhecer melhor os resultados e chegar ao melhor resultado na mineração. Para alcançar esses objetivos são gerados arquivos arff e testados com vários algoritmos e opções de algoritmos.

Foram gerados dois arquivos que possuem os mesmos registros, porém um deles possui atributos numéricos e o outro possui valores nominais. Como o Weka não permite

atributos com valores de mais de 1 *token*, foi preciso gerar novamente os registros e depois realizar um tratamento para substituir os espaços entre as palavras por "*underline*". Devido ao fato de ser muito trabalhosa a retirada dos espaços em cada atributo, foi alterada a SQL utilizada para realizar esse tratamento na geração do arquivo ARFF do Weka.

Foram recolhidos registros de 01/06 até 05/06, sendo que destes, 999 rejeições e 299 vendas. Em um segundo momento foi necessário separar somente as rejeições por parte do cliente, já que nesses dados pode existir qualquer tipo de rejeição, até aquelas por telefone inválido ou outro contato não efetivo com o *prospect*.

O primeiro arquivo a ser testado foi aquele que possui os valores dos atributos nominais. A primeira intenção foi testar se existe diferença entre os resultados quando numérico ou nominal. Como resultado da aplicação do J48 obteve-se o resultado apresentado na figura 5.2:

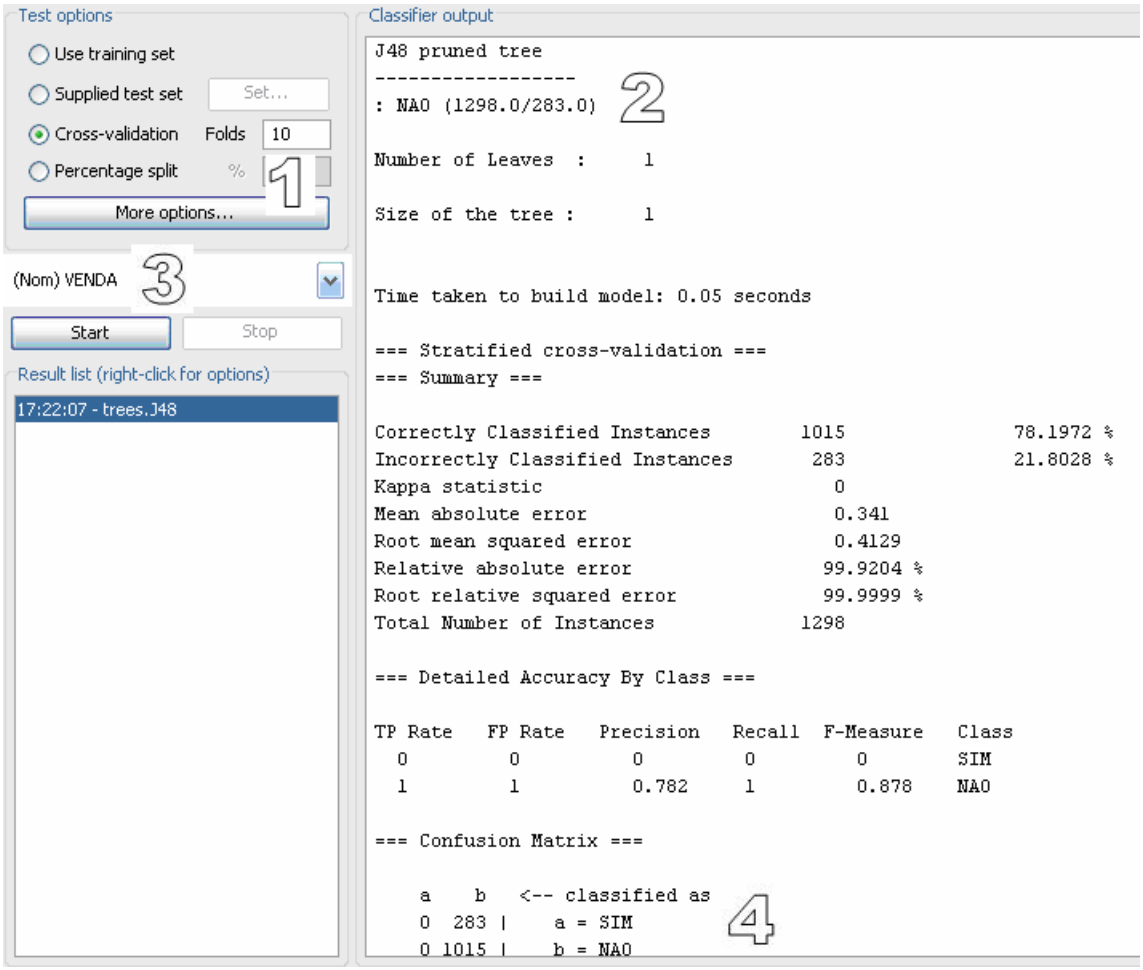

Figura 5.2 – Resultado da mineração do primeiro teste inicial
Fonte: Weka, 2007.

Conforme a figura 5.2 e os itens destacados na mesma foram utilizados as configurações padrões do Weka, com 10 *folds* na configuração de "*cross-validation*" (1). A árvore gerada possui apenas 1 folha (2) que identifica NÃO (referente ao atributo que indica se existe venda ou não). Note que a classe selecionada para gerar a árvore é o atributo VENDA (3). Isso ocorre pelo fato de que existem mais registros de rejeição do que registros de venda dentro dessa base (4). A mesma situação ocorre dentro da operação de *telemarketing*, onde são necessários muitos contatos para encontrar algum cliente que deseja adquirir o produto oferecido. Justamente para reduzir esse problema do setor de *telemarketing*, que esse trabalho é motivado.

A primeira atitude para testar outros resultados é mudar o número de *folds* para não perder regras em cada redução de erros na poda do J48. Como o Weka só permite valores maiores que 1 para *fold*, foi atribuído 2 e não houve mudança alguma no resultado. A próxima tentativa foi verificar as opções do algoritmo, a fim de descobrir algo que influencia no resultado do J48. As configurações padrões do Weka para o J48 são apresentadas na figura 5.3:

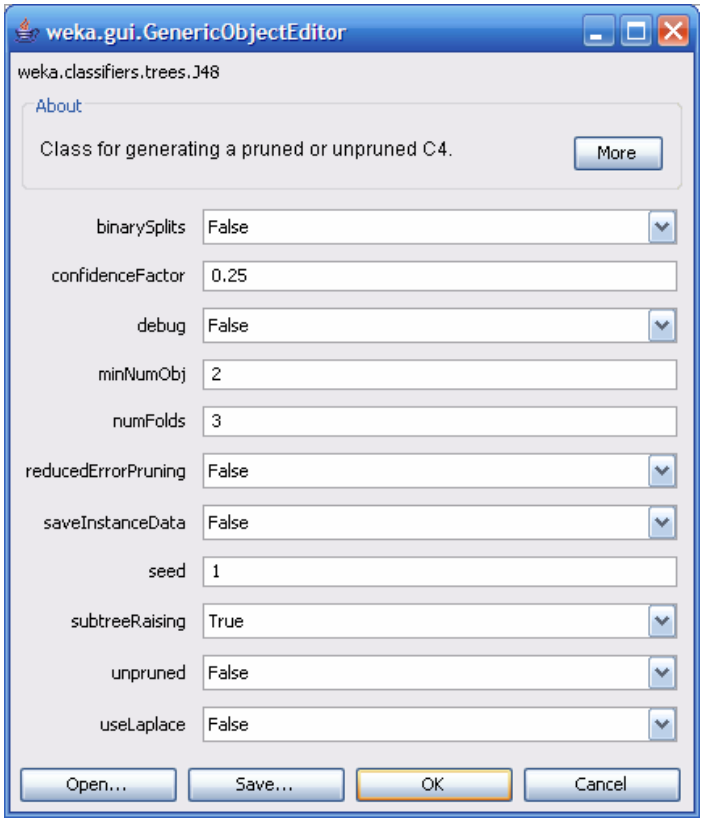

Figura 5.3 – Configuração do algoritmo J48
Fonte: Weka, 2007.

O fator de confiança foi alterado de 0,25 para 0,1 na tentativa de diminuir bastante o grau de precisão esperada pelo algoritmo, mas não houve resultado positivo. Para estes testes a matriz de confusão é apresentada no tabela 5.2:

Tabela 5.2 – Matriz de Confusão do primeiro teste com J48

	A	B	Classificada como
A	0	283	A = SIM
B	0	1015	B = NÃO

Fonte: Do autor

Observando-se a matriz de confusão acima se verifica que todos os registros que deveriam ser do tipo venda igual a SIM foram classificados como NÃO, caracterizando erros de classificação.

A opção *binary splits* serve para ativar a divisão binária quando se utiliza atributos nominais, situação deste teste. O resultado dessa vez foi positivo já que foram geradas mais de uma folha. A matriz de confusão resultante foi alterada para o resultado apresentado na tabela 5.3, porém ainda possui uma quantidade alta de instâncias classificadas incorretamente:

Tabela 5.3 – Matriz de Confusão do J48 gerada com *Binary Splits*

	A	B	Classificada como
A	9	274	A = SIM
B	44	971	B = NÃO

Fonte: Do autor

Para esta situação a árvore gerada possui mais de uma folha e já permite associar alguma informação mesmo que inicial, conforme figura 5.4:

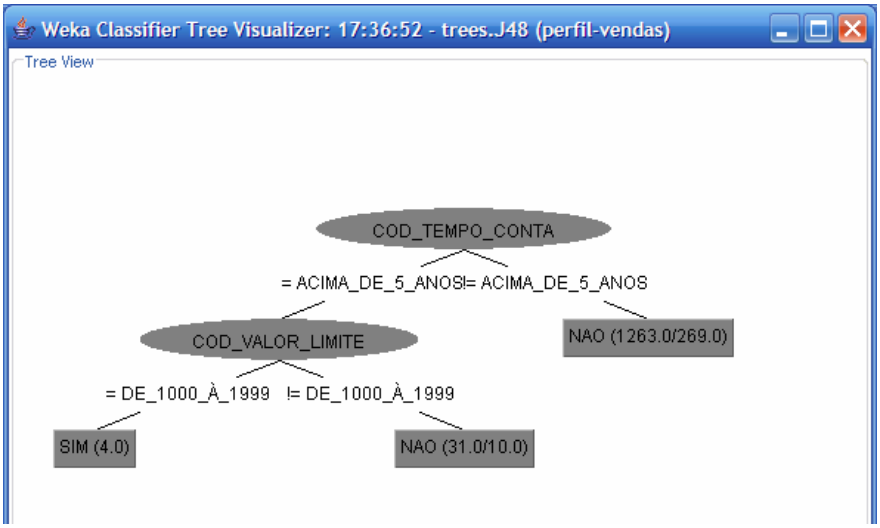

Figura 5.4 – Árvore gerada pelo algoritmo J48
Fonte: Weka, 2007.

Através dessa árvore se obtém a informação que quando os clientes não possuem tempo de conta acima de 5 anos acabam não comprando o produto. Ao contrário, aqueles que além de terem 5 anos como clientes, quando possuem de 1000 a 1999 reais de limite de crédito, adquirem o produto, enquanto os outros que possuem essa mesma faixa de limite são rejeições.

Como continuidade do teste, quanto mais se aumenta o valor de confiança, mais folhas são geradas. Diminuindo a confiança, a árvore tende a ficar com 1 folha apenas e então, pouco informativa. Na figura 5.5, pode-se perceber que somente alterando o fator de confiança para 0.26, já é possível chegar a uma árvore bem maior.

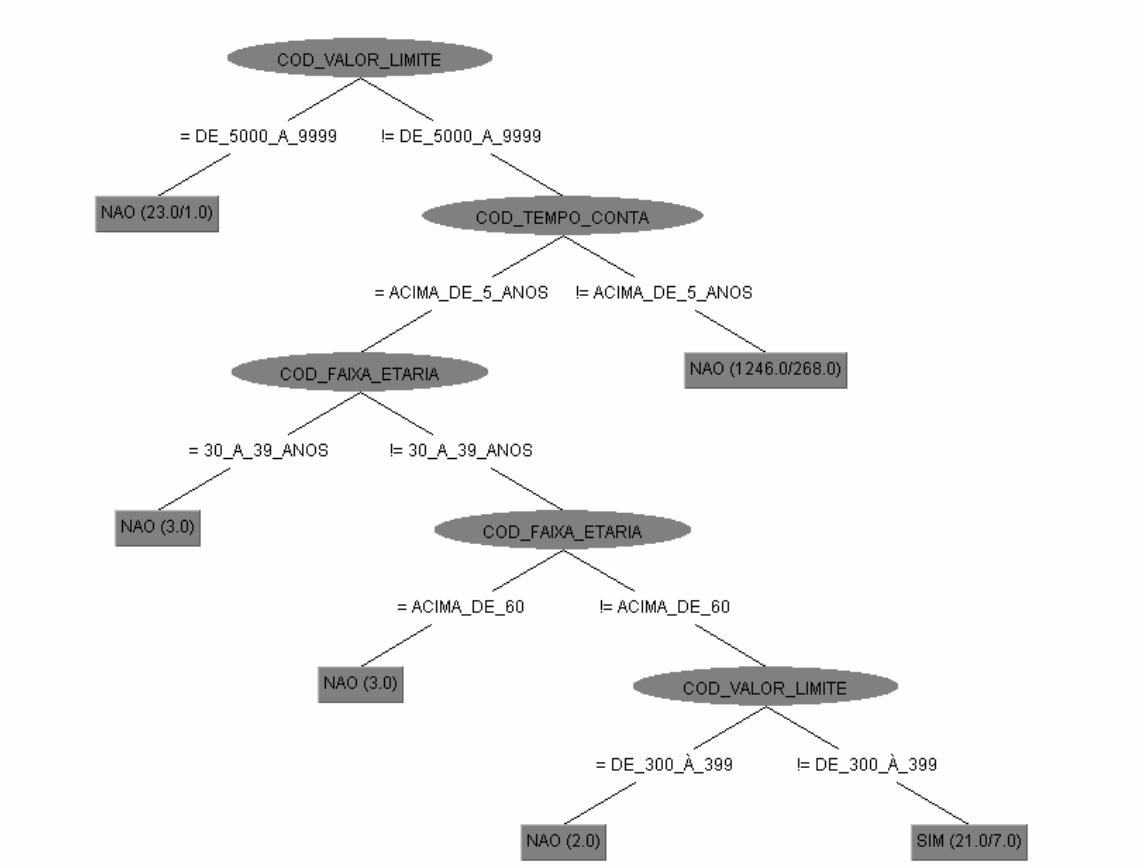

Figura 5.5 – Nova árvore gerada pelo Weka
Fonte: Weka, 2007.

Ativando a opção *reducerErrorPruning* (reduzir erros na poda), a árvore gerada com fator de confiança 0,25 ganha muito mais folhas.

A nova árvore é apresentada no quadro 5.1:

Quadro 5.1 – Árvore de classificação com redução de erros na poda

```
COD_TEMPO_CONTA = ACIMA_DE_5_ANOS
|  COD_VALOR_LIMITE = DE_1000_A_1999: SIM (3.0)
|  COD_VALOR_LIMITE != DE_1000_A_1999: NAO (24.0/9.0)
COD_TEMPO_CONTA != ACIMA_DE_5_ANOS
|  COD_FAIXA_ETARIA = 18_A_29_ANOS: NAO (54.0/20.0)
|  COD_FAIXA_ETARIA != 18_A_29_ANOS
|  |  COD_VALOR_LIMITE = 10000_A_24999: NAO (4.0)
|  |  COD_VALOR_LIMITE != 10000_A_24999
|  |  |  COD_VALOR_LIMITE = DE_800_A_999: NAO (43.0/2.0)
|  |  |  COD_VALOR_LIMITE != DE_800_A_999
|  |  |  |  COD_VALOR_LIMITE = DE_2000_A_4999
|  |  |  |  |  COD_REGIAO_MAILING = RS: NAO (6.0)
|  |  |  |  |  COD_REGIAO_MAILING != RS
|  |  |  |  |  |  COD_REGIAO_MAILING = PR: NAO (5.0)
|  |  |  |  |  |  COD_REGIAO_MAILING != PR
|  |  |  |  |  |  |  COD_REGIAO_MAILING = SC: NAO (2.0)
|  |  |  |  |  |  |  COD_REGIAO_MAILING != SC
|  |  |  |  |  |  |  |  COD_REGIAO_MAILING = NORDESTE_II: SIM (8.0/3.0)
|  |  |  |  |  |  |  |  COD_REGIAO_MAILING != NORDESTE_II
|  |  |  |  |  |  |  |  |  COD_REGIAO_MAILING = MG
|  |  |  |  |  |  |  |  |  |  COD_FAIXA_ETARIA = 30_A_39_ANOS: SIM (2.0)
|  |  |  |  |  |  |  |  |  |  COD_FAIXA_ETARIA != 30_A_39_ANOS
|  |  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA = DE_1_A_2_ANOS: NAO (3.0/1.0)
|  |  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA != DE_1_A_2_ANOS
|  |  |  |  |  |  |  |  |  |  |  |  COD_SEXO_ATB = MASCULINO: SIM (2.0)
|  |  |  |  |  |  |  |  |  |  |  |  COD_SEXO_ATB != MASCULINO: NAO (7.0/3.0)
|  |  |  |  |  |  |  |  |  COD_REGIAO_MAILING != MG
|  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA = DE_0_A_6_MESES: NAO (20.0/2.0)
|  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA != DE_0_A_6_MESES
|  |  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA = DE_2_A_3_ANOS: SIM (8.0/4.0)
|  |  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA != DE_2_A_3_ANOS
|  |  |  |  |  |  |  |  |  |  |  |  COD_REGIAO_MAILING = NORDESTE_I
|  |  |  |  |  |  |  |  |  |  |  |  |  COD_FAIXA_ETARIA = 30_A_39_ANOS: SIM (6.0/3.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  COD_FAIXA_ETARIA != 30_A_39_ANOS: NAO (9.0)
|  |  |  |  |  |  |  |  |  |  |  |  COD_REGIAO_MAILING != NORDESTE_I
|  |  |  |  |  |  |  |  |  |  |  |  |  COD_FAIXA_ETARIA = 50_A_59_ANOS: NAO (29.0/9.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  COD_FAIXA_ETARIA != 50_A_59_ANOS
|  |  |  |  |  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA = DE_1_A_2_ANOS: NAO (7.0/2.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA != DE_1_A_2_ANOS
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  COD_FAIXA_ETARIA = 30_A_39_ANOS: NAO (10.0/4.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  COD_FAIXA_ETARIA != 30_A_39_ANOS
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  COD_REGIAO_MAILING = SP_INTERIOR: NAO (7.0/3.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  COD_REGIAO_MAILING != SP_INTERIOR
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  DIA_VENC_FATURA <= 12.0: SIM (8.0/1.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  DIA_VENC_FATURA > 12.0
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA = DE_3_A_4_ANOS: SIM (3.0/1.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  COD_TEMPO_CONTA != DE_3_A_4_ANOS: NAO (2.0)
|  |  |  |  COD_VALOR_LIMITE != DE_2000_A_4999: NAO (594.0/106.0)

Number of Leaves  :     25

Size of the tree :  49
```

Fonte: Do autor

A utilização de conjunto de treinamento e do percentual de divisão não influenciou na geração da árvore, mas não utilizar a poda, de fato, é algo que dificulta a análise do resultado.

Ao não utiliza-la foi gerada uma árvore com tamanho igual a 391 e com 196 folhas, o que identifica que foram geradas muito mais regras e que talvez muitas delas passam a ser desnecessárias.

A matriz de confusão sem a utilização de poda é apresentada na tabela 5.4:

Tabela 5.4 – Matriz de Confusão do J48 utilizando sem poda

	A	B	Classificada como
A	52	231	A = SIM
B	173	842	B = NÃO

Fonte: Do autor

Algo que se deve levar em consideração é que a quantidade de registros não classificados e a matriz de confusão, como no exemplo da tabela 5.4. Quanto mais próximos de 0 for o valor das células que contém os valores 131 e 173, isso identifica que a classificação chegou a um resultado mais otimizado, ou seja, com menos erros.

Como os resultados obtidos não foram considerados satisfatórios e para que se tenha total controle sobre as configurações do algoritmo J48, que havia sido pré-selecionado, foi criado um arquivo de teste com apenas 10 registros para observar com mais facilidade os avanços e resultados.

Arquivos com essa quantidade de registros e com esses atributos selecionados, não geraram uma árvore de qualidade por causa dos registros utilizados. Na figura 5.6, é apresentado o arquivo manipulado:

```
%Informações dos registros de venda na base de teste
@relation perfil-vendas

@attribute COD_REGIAO_MAILING   {CENTRO-OESTE,MG,NORDESTE_I,NORDESTE_II,NORTE,PR,RJ-ES,RS,SC,SP_CAPITAL,SP_INTERIOR}
@attribute COD_CLASSIFICACAO    {EXCLUSIVO,INSTITUCIONAL,NÃO_INFORMADO,PLATINUM,UNICLASS,UNICO}
@attribute DIA_VENC_FATURA      real
@attribute COD_TEMPO_CONTA      {ACIMA_DE_5_ANOS,DE_0_A_6_MESES,DE_1_A_2_ANOS,DE_2_A_3_ANOS,DE_3_A_4_ANOS,
                                DE_4_A_5_ANOS,DE_7_A_12_MESES,NÃO_INFORMADO}
@attribute COD_FAIXA_ETARIA     {18_A_29_ANOS,30_A_39_ANOS,40_A_49_ANOS,50_A_59_ANOS,ACIMA_DE_60,NÃO_INFORMADO}
@attribute COD_VALOR_LIMITE     {A_PARTIR_5000,DE_1000_À_1999,DE_200_À_299,DE_2000_À_4999,DE_300_À_399,
                                DE_400_À_799,DE_5000_A_9999,DE_800_À_999,NÃO_INFORMADO,10000_A_24999,25000_A_49999}
@attribute COD_SEXO_ATB         {MASCULINO,FEMININO,NÃO_INFORMADO}
@attribute VENDA                {SIM, NAO}

@data
%
% 10 instances
%
NORTE,NÃO_INFORMADO,15,DE_0_A_6_MESES,ACIMA_DE_60,DE_1000_À_1999,MASCULINO,SIM
NORDESTE_II,NÃO_INFORMADO,12,DE_0_A_6_MESES,40_A_49_ANOS,DE_2000_À_4999,MASCULINO,NAO
RS,NÃO_INFORMADO,15,DE_7_A_12_MESES,50_A_59_ANOS,DE_1000_À_1999,FEMININO,NAO
MG,NÃO_INFORMADO,29,DE_1_A_2_ANOS,40_A_49_ANOS,DE_2000_À_4999,MASCULINO,SIM
RS,NÃO_INFORMADO,12,DE_0_A_6_MESES,40_A_49_ANOS,DE_2000_À_4999,FEMININO,NAO
NORDESTE_II,NÃO_INFORMADO,8,DE_0_A_6_MESES,50_A_59_ANOS,DE_1000_À_1999,MASCULINO,NAO
RS,NÃO_INFORMADO,12,DE_7_A_12_MESES,ACIMA_DE_60,DE_1000_À_1999,FEMININO,NAO
NORTE,NÃO_INFORMADO,15,DE_1_A_2_ANOS,30_A_39_ANOS,DE_1000_À_1999,FEMININO,NAO
RS,NÃO_INFORMADO,12,DE_7_A_12_MESES,ACIMA_DE_60,DE_1000_À_1999,MASCULINO,SIM
NORDESTE_I,NÃO_INFORMADO,29,DE_1_A_2_ANOS,50_A_59_ANOS,DE_400_À_799,FEMININO,NAO
```

Figura 5.6 – Arquivo arff com 10 registros
Fonte: FIGURA NOSSA.

Dos 10 registros, 3 eram registros com o atributo venda como SIM. Esses registros de venda acabam sendo classificados como não venda e somam os 30% de instâncias classificadas incorretamente, ou seja, registros que o atributo VENDA é SIM, mas são

classificados como NAO. Esse exemplo se tornou útil para conhecer as configurações e os algoritmos do Weka, embora não tenha gerado resultados mais esclarecedores uma vez que a árvore somente possuía 1 folha, o que representa que todos esses registros somente resultariam em rejeição por parte do cliente, mesmo existindo 3 vendas.

O Weka possui uma ferramenta para visualizar os tipos de dados do arquivo, na aba *Visualize* do *Weka Explorer*, onde pode perceber-se a diversidade de certo atributo. Na figura 5.6, pode ser visualizado o arquivo de 10 registros apresentado anteriormente, com os atributos região e valor limite selecionado. Quanto menor a quantidade de pontos no gráfico, indicado por (1), menor a diversidade de valores do atributo conforme os eixos X e Y.

As cores do gráfico são apresentadas conforme o atributo selecionado (2), no caso do exemplo da figura 5.6, onde foi selecionado o atributo VENDA que possui os valores do tipo SIM representado pela cor azul e NÃO com a cor vermelha. Dessa maneira, é possível além de verificar a diversidade de valores dos registros é possível verificar a distribuição do atributo que é selecionado para ser identificado pela cor, ou seja, quanto mais pontos vermelhos na área destacada na figura (4), maior é a quantidade de regiões que não são vendas. A figura 5.7 apresenta o visualizador de dados do Weka:

Figura 5.7 – Visualizador de dados do Weka do arquivo exemplo de 10 registros
Fonte: Weka, 2007.

5.4 Seleção do algoritmo

Depois de terem sido feitos os testes iniciais é preciso realizar testes para selecionar o algoritmo que obterá o melhor resultado, em se tratando de instâncias classificadas corretamente. O objetivo é aumentar a quantidade de instâncias classificadas corretamente do que no teste do item 5.3.

Em primeiro lugar, foi gerado outro arquivo com dados atualizados de 5 dias de contatos indicando venda ou rejeição. Esse novo arquivo possui 1298 registros para selecionar o melhor algoritmo para o problema do *Call Center*.

O algoritmo J48 com utilização de redução erros na poda e utilizando conjunto de treinamento, chegou a 79,96% de instâncias classificadas corretamente e com a matriz de confusão da tabela 5.5:

Tabela 5.5 – Matriz de Confusão do J48 utilizando poda

	A	B	Classificada como
A	38	245	A = SIM
B	15	1000	B = NAO

Fonte: Do autor

Essa matriz de confusão mostra que a quantidade de acertos se deve mais aos registros que foram corretamente classificados como não venda do que aqueles que foram vendas. Esse resultado não é interessante, visto que não se pode conhecer o melhor perfil de vendas. O perfil de rejeições ainda não é o foco dos testes.

Conforme a experiência do trabalho de OLIVEIRA et al (2002), para tentar alcançar modelos mais precisos, também é interessante realizar testes com o algoritmo J48 PART e utilizando os métodos de meta aprendizagem *Bagging* e *Boosting*.

Aplicando o algoritmo J48 PART, atingiu-se 74,73% de instâncias classificadas corretamente, percentual que aumenta utilizando a redução de erros e conjunto de treinamento chegando a 80,97%. A matriz de confusão ainda não teve grande melhora em relação ao J48. Segue na tabela 5.6, a matriz de confusão gerada:

Tabela 5.6 – Matriz de Confusão do algoritmo J48 PART

	A	B	Classificada como
A	73	210	A = SIM
B	37	978	B = NAO

Fonte: Do autor

O método de meta aprendizagem *Bagging* atingiu melhores resultados com o algoritmo J48, com um percentual de 82,58% dos registros classificados corretamente. Já o método de *Boosting*, chamado AdaboostM1, que é uma classe para *boosting* apenas para classes nominais, aumentou o total para 84,82%. A matriz de confusão gerada, presente na tabela 5.7, foi a seguinte:

Tabela 6.7 – Matriz de Confusão do método de meta aprendizagem AdaboostM1

	A	B	Classificada como
A	121	162	A = SIM
B	35	980	B = NAO

Fonte: Do autor

A opção de *resampling* trata de utilizar reamostragem ao invés de utilizar a alteração de peso, característica que atribui pesos e depois altera os pesos das instâncias classificadas incorretamente.

Com essa reamostragem, chegou-se a 96,76% dos registros classificados de forma correta. A matriz de confusão da tabela 5.8 indica que dos registros de venda classificados corretamente foram 94,69% (268 registros de 283 registros do tipo SIM) e as rejeições foi de 97,33% (988 registros de 1025 registros do tipo NAO).

Tabela 5.8 – Matriz de Confusão de *Boosting* com reamostragem

	A	B	Classificada como
A	268	15	A = SIM
B	27	988	B = NAO

Fonte: Do autor

Dessa forma, com o modelo utilizado, o método AdaboostM1 foi selecionado como aquele que gerou os melhores resultados. Desse ponto em diante, com o algoritmo selecionado, é necessário trabalhar com as regras geradas e realizar os testes de desempenho, comprovando a qualidade e aplicabilidade da mineração dos dados em um *Call Center* ativo.

5.5 Testes de diminuição de contatos e aumento de vendas nos contatos sem mineração

Esse teste se propõe a verificar dentre os registros do trabalho do *Call Center*, qual seria o desempenho se fosse utilizada a mineração. Esses registros/contatos foram realizados sem a utilização da mineração, somente com as estratégias já utilizadas atualmente. Os contatos serão minerados e através do resultado da mineração poderão ser encontradas as características dos clientes que obtiveram melhor índice de vendas. O objetivo é descobrir se

utilizados somente os registros com essas características, se seria possível aumentar as vendas e diminuir os contatos.

O método de *boosting* escolhido será aplicado nessa base e as regras geradas serão separadas. Com base nessas regras, será possível estimar a quantidade de contatos que seriam necessários para atingir a mesma quantidade de vendas ou então se seria possível aumentar a quantidade de vendas por tentativa.

Para realizar esses testes tornou-se necessário adicionar mais uma informação importante. Em certos momentos do trabalho do *Call Center*, é preciso dar um foco maior em certos tipos de títulos de capitalização. O desempenho da empresa, nesse tipo de produto, também é medido pelo faturamento obtido nas vendas, que trata do somatório do valor dos títulos vendidos. Nesse caso, foi adicionado ao modelo proposto outro atributo, que identifica o tipo de título de capitalização adquirido. Através desse tipo de título, é possível identificar seu valor, estes estão nomeados com seu valor no nome (por exemplo: CAP_35_00, CAP_50_00, CAP_100_00)

Foram analisados os contatos de 17/09 a 05/10, sendo que o total das vendas é 2267 dentre os 9092 registros de contatos efetivos com os clientes, já com o novo atributo. Primeiramente, será analisado o total das vendas de título de 50 reais onde foram vendidos 768 títulos na empresa. A mineração desses dados chegou ao seguinte resultado de 1888 instâncias classificadas corretamente que representa 83,28% dos registros e classificadas incorretamente um total de 379 instâncias que totaliza 16,72% das instâncias. A matriz de confusão dessa nova mineração é apresentada na tabela 5.9:

Tabela 5.9 – Matriz de Confusão da mineração com a classe PRODUTO

	A	B	C	D	E	F	G	H	I	Classificada como
A	0	0	0	0	0	0	0	0	0	A = REJEICAO
B	0	152	0	1	39	47	1	6	0	B = CAP_20_00
C	0	0	0	0	0	0	0	0	0	C = CAP_30_00
D	0	1	0	24	1	5	0	6	0	D = CAP_40_00
E	0	7	0	0	576	85	7	25	0	E = CAP_35_00
F	0	10	0	1	34	696	2	25	0	F = CAP_50_00
G	0	0	0	0	7	5	85	0	0	G = CAP_70_00
H	0	5	0	0	28	30	0	355	0	H = CAP_100_00
I	0	0	0	0	1	0	0	0	0	I = CAP_200_00

Fonte: Do autor

Para as regras separadas, o nível exigido, definido empiricamente, foi de 20 registros. O nível referido trata do total de registros classificados na regra gerada, como por exemplo,

(38/18). No caso de resultados insatisfatórios, pode ser alterado. As regras geradas estão presentes no quadro 5.2:

Quadro 5.2 – Regras separadas do teste de diminuição de contatos

```
COD_VALOR_LIMITE = DE_400_A_799
| COD_SEXO_ATB = MASCULINO
| | COD_REGIAO_MAILING = RJ-ES
| | | DIA_VENC_FATURA > 5: CAP_50_00 (38.0/18.0)
---------- Acerto: 47.368421052631578%

COD_VALOR_LIMITE = DE_1000_A_1999
| COD_REGIAO_MAILING = SP_CAPITAL
| | DIA_VENC_FATURA > 22: CAP_50_00 (26.0/12.0)
---------- Acerto: 46.15384615384615%

COD_VALOR_LIMITE = DE_400_A_799
| COD_SEXO_ATB = FEMININO
| | COD_REGIAO_MAILING = SP_CAPITAL
| | | DIA_VENC_FATURA > 12: CAP_50_00 (24.0/11.0)
---------- Acerto: 45.833333333333336%

COD_VALOR_LIMITE = DE_400_A_799
| COD_SEXO_ATB = MASCULINO
| | COD_REGIAO_MAILING = PR
| | | DIA_VENC_FATURA > 12: CAP_50_00 (20.0/8.0)
---------- Acerto: 40.0%

COD_VALOR_LIMITE = DE_400_A_799
| COD_SEXO_ATB = MASCULINO
| | COD_REGIAO_MAILING = SP_CAPITAL: CAP _50_00 (20.0/10.0)
---------- Acerto: 50.0%
```

Fonte: Do autor

Os restantes das regras, não atingiram o total de 20 instâncias e, portanto são somente apresentadas e não incluídas em um primeiro momento.

Para realização a comparação, os registros que possuem apenas as características separadas serão manipulados para verificar se o percentual de conversão será maior. Conforme as melhores regras com mais de 19 registros selecionados, os atributos selecionados foram:

- COD_VALOR_LIMITE = DE_400_À_799

- COD_VALOR_LIMITE = DE_1000_À_1999

- COD_REGIAO_MAILING = RJ-ES

- COD_REGIAO_MAILING = SP_CAPITAL

- COD_REGIAO_MAILING = PR

Essas características, consideradas principais, foram destacadas no quadro 5.2. A idéia é fazer a comparação das vendas, onde foram necessários 9092 contatos efetivos para se conseguir 768 vendas do título de 50 reais.

Na base total, o percentual de conversão era de 8,44% já que eram 768 vendas em 9092 contatos efetivos.

Com a mineração realizada e considerando que então seriam trabalhados apenas os nomes com as características selecionadas, seriam 1939 contatos efetivos realizados e chegar-se-ia a 225 vendas de capitalização de 50,00. Dessa maneira, o percentual de conversão de 11,60 % com 21,32% dos contatos realizados. Dessa maneira, hipoteticamente, seria uma estratégia seria muito mais eficiente. No gráfico 5.1, são apresentados os números alcançados na venda de títulos de 50 reais:

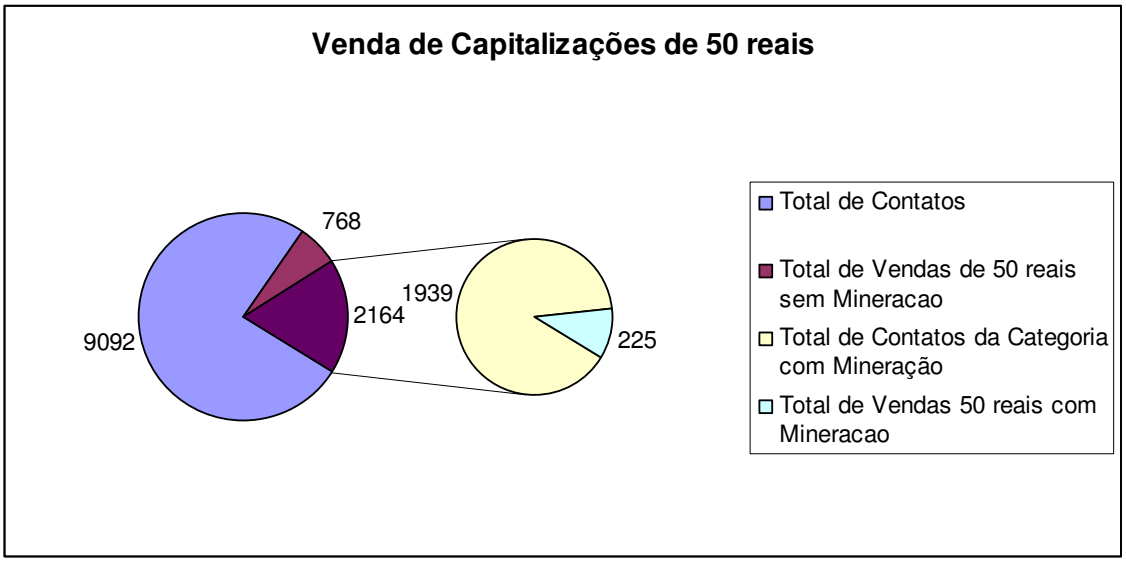

Gráfico 5.1 – Gráfico de Venda de Capitalizações de 50 reais
Fonte: Do autor.

Mesmo a quantidade de vendas sendo menor, a quantidade de contatos também seria reduzida e perder-se-ia menos tempo com contatos, que tem grande chance de serem sem sucesso. Assim, sobraria mais tempo para tentar outras estratégias ou então continuar com a estratégia atual e chegar ao número de 768 vendas em menos tempo.

Para comprovar a teoria, será analisado também o segundo produto mais vendido, o título de R$35,00. Nesse caso, as melhores regras são apresentadas no quadro 5.3:

Quadro 5.3 – Melhores regras para venda de títulos de 35 reais

```
COD_VALOR_LIMITE = DE_300_A_399
| COD_FAIXA_ETARIA = ACIMA_DE_60: CAP_35_00 (24.0/7.0)
---------- Acerto: 29.166666666666668%

COD_VALOR_LIMITE = DE_400_A_799
| COD_SEXO_ATB = FEMININO
| | COD_REGIAO_MAILING = SP_CAPITAL
| | | DIA_VENC_FATURA <= 12: CAP_35_00 (23.0/11.0)
---------- Acerto: 47.82608695652174%

COD_VALOR_LIMITE = DE_800_A_999
| COD_TEMPO_CONTA = DE_0_A_6_MESES
| | COD_REGIAO_MAILING = RJ-ES
| | | DIA_VENC_FATURA > 5: CAP_35_00 (22.0/10.0)
---------- Acerto: 45.45454545454545%

COD_VALOR_LIMITE = DE_400_A_799
| COD_SEXO_ATB = FEMININO
| | COD_REGIAO_MAILING = MG
| | | COD_TEMPO_CONTA = DE_7_A_12_MESES: CAP_35_00 (21.0/10.0)
---------- Acerto: 47.61904761904762%
```

Fonte: Do autor

Os atributos selecionados foram:

- COD_VALOR_LIMITE = DE_300_À_399

- COD_VALOR_LIMITE = DE_400_À_799

- COD_TEMPO_CONTA = DE_0_A_6_MESES

- COD_TEMPO_CONTA = DE_7_A_12_MESES

- COD_REGIAO_MAILING = RJ-ES

- COD_REGIAO_MAILING = MG

- COD_REGIAO_MAILING = SP_CAPITAL

- COD_FAIXA_ETARIA = ACIMA_DE_60

A conversão desse produto foi de 700 vendas em 9092 contatos, ou seja, 7,69%. Somente separando os registros que possuem as características selecionadas, o número de contatos baixou para 1296 e desses registros 136 vendas, atingindo o percentual de 10,49% e com apenas 14,25% dos contatos.

Pela quantidade reduzida de registros com faixa etária acima de 60 anos, esse atributo não foi considerado. O gráfico 5.2 mostra o desempenho das vendas de títulos de R$35,00.

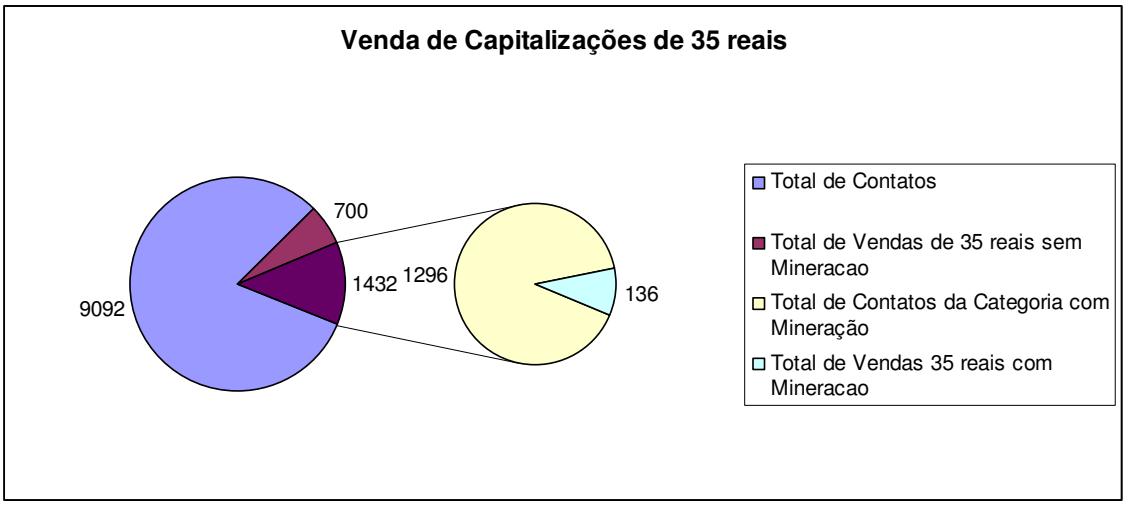

Gráfico 5.2 – Gráfico de Venda de Capitalizações de 35 reais
Fonte: Do autor.

O próximo passo do teste é analisar as vendas sem considerar o produto adquirido, simulando uma situação onde o foco é realizar vendas, independente do valor do título.

Dos 9092 contatos, foram realizadas 2267 vendas, percentual de conversão de 24,93%. Como exigência para a regra ser considerada válida, com valor hipotético de 50 registros, é possível desconsiderar algumas regras que abrangeram menos instâncias. Das regras selecionadas foram separados os seguintes atributos:

- COD_VALOR_LIMITE = DE_1000_À_1999

- COD_VALOR_LIMITE = DE_400_À_799

- COD_VALOR_LIMITE = DE_200_À_299

- COD_VALOR_LIMITE = DE_300_À_399

- COD_VALOR_LIMITE = DE_2000_À_4999

- COD_REGIAO_MAILING = MG

- COD_REGIAO_MAILING = SP_CAPITAL

- COD_REGIAO_MAILING = RJ-ES

- COD_REGIAO_MAILING = NORDESTE_I

- COD_REGIAO_MAILING = SP_INTERIOR

- COD_REGIAO_MAILING = CENTRO-OESTE

- COD_TEMPO_CONTA = DE_0_A_6_MESES

- COD_TEMPO_CONTA = DE_1_A_2_ANOS

- COD_TEMPO_CONTA = DE_7_A_12_MESES

- COD_FAIXA_ETARIA = 40_A_49_ANOS

- COD_FAIXA_ETARIA = 30_A_39_ANOS

Se fossem trabalhados somente os registros com essas características, seriam apenas 2464 contatos efetivos e desses registros seriam 563 vendas atingindo um percentual de conversão de 22,84%, conforme pode ser visualizado no gráfico 5.3;

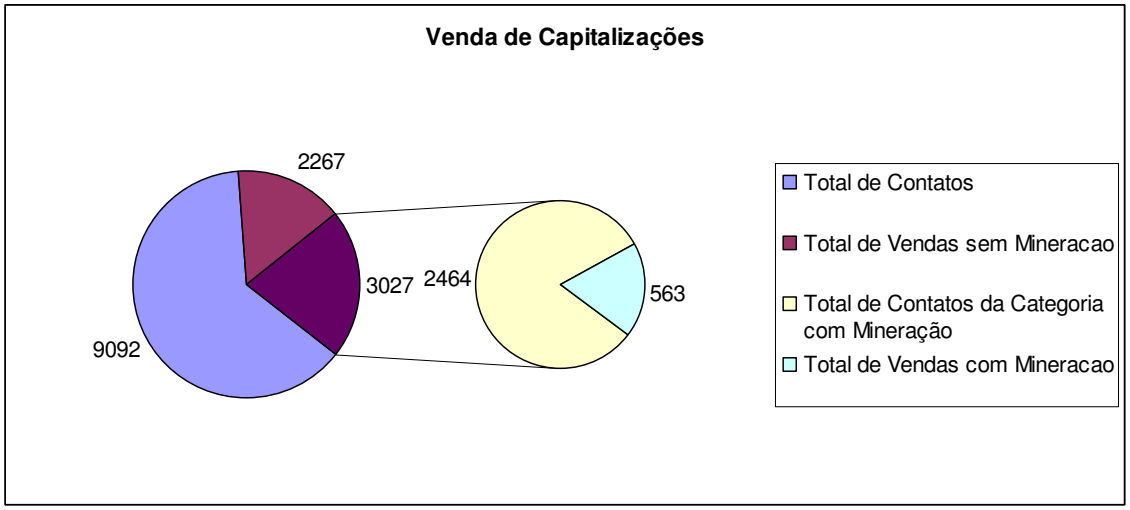

Gráfico 5.3 – Gráfico de Venda de Capitalizações
Fonte: Do autor.

Nesse caso, perde-se 2,09% de conversão, porém se consegue 27,1% dos contatos realizados anteriormente, pois foram apenas 2464 registros, o que também resulta em economia e aumento de precisão na seleção dos *prospects*.

Para tentar reproduzir o mais fielmente o ambiente de *telemarketing*, os testes foram realizados de maneira que alguns atributos, por possuírem menos registros, fossem desprezados como no caso do sexo e do dia de vencimento da fatura.

Dos 9092 registros que existiam na base, deveriam ser cortados aqueles que não faziam parte dos atributos gerados na primeira regra, ou seja, os dados seriam cortados da seguinte maneira:

- Valor limite de 400 a 799, só tem na base 1200, então a base de teste seria de 1200 nomes;

- Valor limite de 1000 1999, só tem na base 800, então a base de teste seria de 2000 nomes (1200+800);

Os próximos atributos podem sobrepor os registros anteriores, ou seja:

- Região RJ/ES, só possui 500 nesses 2000 que haviam sido separados, agora a base de teste possui 500 nomes;

- Região SP capital, só possui 600 nesses 2000 que haviam sido separados, agora a base de teste possui 1100 nomes;

- Região PR, só possui 70 nesses 2000 que haviam sido separados, agora a base de teste possui 1170 nomes;

O que ocorre é que se ainda fosse restringir a base pelo sexo, esta ficaria da seguinte maneira:

- Sexo Masculino, só possui 560 nesses 1170 que haviam sido separados, agora a base de teste seria de 560 nomes;

Se ainda fossem restringidos apenas nomes com vencimento de fatura maior que dia 15, por exemplo, a base de teste poderia ficar da seguinte maneira:

- Vencimento maior do que dia 5, só possuem 230 nesses 560 que haviam sido separados, agora a base seria de 230 nomes;

Pelo fato de existir uma quantidade reduzida de nomes, acreditou-se que isso prejudicaria a comparação dos registros e por isso alguns atributos não foram considerados. Esses mesmos atributos podem ainda representar um grande aumento na taxa de conversão dos registros e por isso outros testes deveriam ser propostos.

Nesse novo caso, os registros são testados da mesma maneira que poderiam ser disponibilizados para atendimento no *Call Center*. Os 230 nomes formariam uma categoria de cliente e poderiam ser liberados para atendimento. No momento que essa quantidade fosse totalmente trabalhada, poderiam ser liberados os registros da outra regra gerada, ou seja, somente aqueles registros que possuem valor limite de 1000 a 1999, que pertençam a Capital Paulista e que possuam vencimento maior do que o dia 22 que formam outra categoria de

cliente. Nesse segundo momento, poderiam existir somente 400 registros, seriam totalmente utilizados e, então, a próxima categoria seria liberada da mesma maneira das anteriores.

Nos próximos testes, cada uma das regras geradas é avaliada separadamente e sua conversão será comparada com a conversão das vendas sem mineração.

Anteriormente, o objetivo era verificar se a conversão de alguns registros era maior do que a conversão de toda a base trabalhada e a quantidade de contatos. Agora, existem várias categorias de clientes, tantas quantas forem às regras geradas na classificação e cada uma dessas categorias possuirá o percentual de conversão obtido.

É importante verificar se existe uma economia de contatos, ou seja, se foi possível diminuir o número de contatos telefônicos, fato que implica em diminuição de custos de telefonia. Juntamente com esse indicador, pode-se considerar também a diminuição do tempo necessário para contatar o número de clientes. Para ser feita uma comparação, se a média de tempo de ligação é de 8 minutos, para contatar 9000 clientes levariam 1200 horas, porém para contatar 1200 clientes é preciso bem menos tempo. Esse tempo economizado poderia ser utilizado para continuar os contatos com clientes de uma categoria definida na mineração além de impactar em diminuição de minutos na fatura telefônica da empresa. A mão de obra necessária nos dois totais de clientes também é outro fator que pode se tornar economia para o *Call Center.*

Para o mesmo teste da mineração dos registros de venda títulos de capitalização de 50 reais, será verificado o valor de conversão dos registros pertencentes a cada categoria formada por cada regra gerada no Weka. A primeira categoria é formada por clientes com limite no cartão de 400 a 799 reais, do sexo masculino, moradores da região do Rio de Janeiro e Espírito Santo e dia de vencimento de fatura maior do que dia 5, dessa categoria existiam 124 registros dentro da base e destes, 33 vendas, totalizando um percentual de conversão de 26,61%.

A próxima categoria é formada por clientes com limite no cartão de 1000 a 1999 reais, moradores da região da capital de São Paulo e com dia de vencimento de fatura maior do que dia 22, dessa categoria existiam 59 registros dentro da base e destes, 5 vendas, totalizando um percentual de conversão de 8,47%.

Para que possa ser feita a totalização da quantidade de registros selecionados e a quantidade de vendas, foi preciso calcular a conversão de todas as regras geradas. Cada uma

das regras foi calculada conforme a ordem que já havia sido apresentada, compondo 5 categorias. O Tabela 5.10, apresenta os resultados das 5 regras selecionadas:

Tabela 5.10 – Tabela de Conversão das 5 categorias selecionadas

	Categoria	Total de registros	Total de Vendas	Percentual de Conversão
	1	124	33	26,61%
	2	59	5	8,47%
	3	115	18	15, 65%
	4	38	11	28,95%
	5	127	25	19,68%
TOTAL	15	463	92	

Fonte: Do autor

Analisando esses registros pode-se concluir que em 463 contatos, atingindo 92 vendas e representando 19,87% de conversão com 5,09% dos contatos, enquanto anteriormente se atingiu 11,60 % de conversão com 21,32% dos contatos. O gráfico 5.4 apresenta a comparação da conversão e a comparação da quantidade de contatos utilizados.

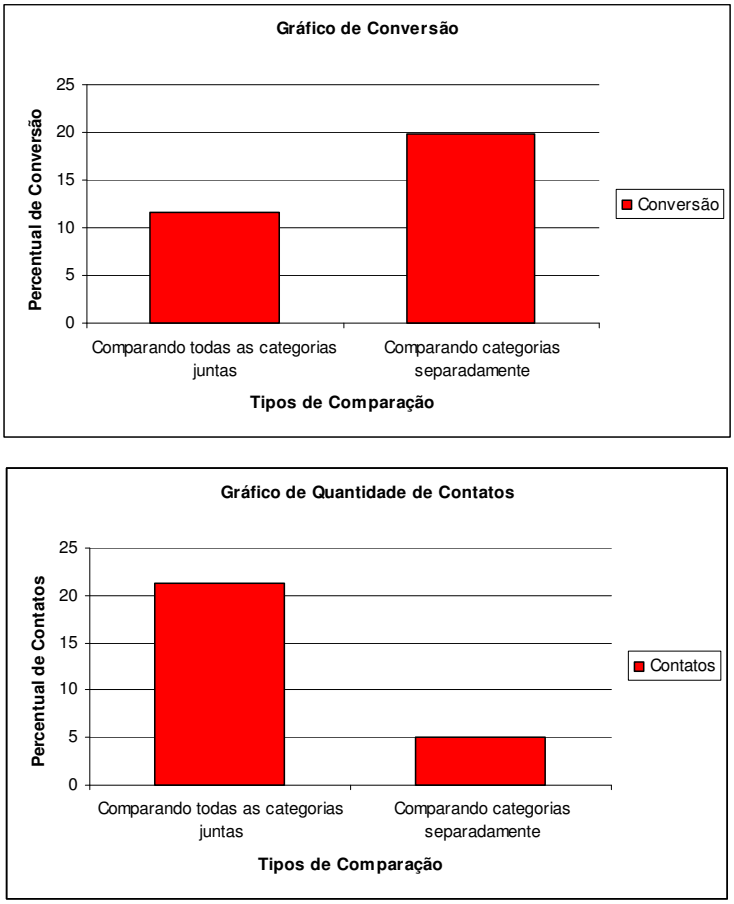

Gráfico 5.4 – Gráficos de Contatos e conversão por categorias
Fonte: Do autor.

Esses números representam que houve uma otimização do trabalho, aumentando a conversão, diminuindo contatos, ou seja, tempo gasto para atingir as vendas e mão de obra necessária para realizar essa quantidade de contatos. Para reafirmar a eficiência da mineração, foram acrescentadas aos totais as outras 8 regras que haviam sido desprezadas pela nível definido. O tabela 5.11 apresenta essas 8 categorias novas:

Tabela 5.11 – Tabela de Conversão das regras desprezadas

Categoria	Total de registros	Total de Vendas	Percentual de Conversão
6	43	8	18,60%
7	25	8	32%
8	61	12	19,67%
9	38	4	10,53%
10	33	13	39,39%
11	35	10	28,57%
12	15	2	13,33%
13	34	8	23,53%

Fonte: Do autor

Agora com todas as regras selecionadas, o total de registros é de 747 contatos, atingindo 157 vendas que representa conversão de 21,01% com 8,22% dos contatos. Além desse aumento do índice de conversão, também se pode notar que algumas das regras que foram desprezadas conseguiram algumas das maiores conversões dessas amostras de teste, como por exemplo, a categoria 10 e 11. O gráfico 5.5 apresenta os totais de contatos e vendas por categorias utilizadas:

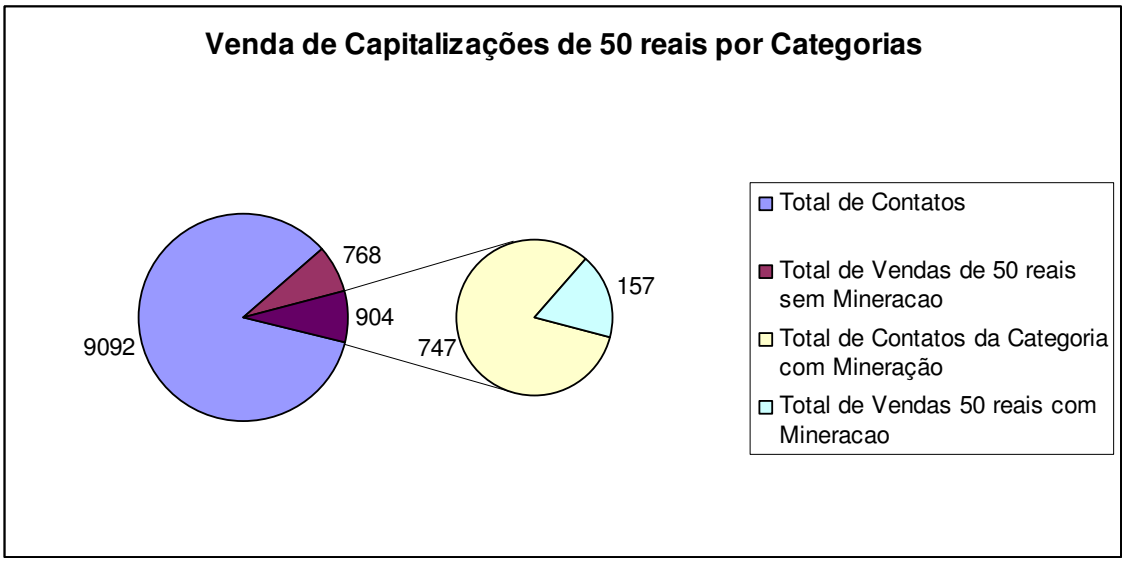

Gráfico 5.5 – Gráfico de Venda de Capitalizações de 50 reais por Categorias
Fonte: Do autor.

Com base nessas informações, é possível realizar outro teste verificando a conversão obtida apenas das categorias que obtiveram conversão na categoria maior que 15%, valor definido por existirem 10 categorias acima desse valor. As categorias selecionadas foram: 1, 3, 4, 5, 6, 7, 8, 10, 11 e 13. A mineração utilizando essas categorias utilizou 635 contatos e atingiu 146 vendas, o que resulta em 22,99% de conversão em 6,98% dos contatos, que é um percentual maior do que os anteriores de 21,01% com 8,22% dos contatos. Se essas regras já fossem conhecidas antes de disponibilizar esses registros para serem contatados, somente 6,98% dos contatos poderiam ser disponibilizados e se conseguiria chegar até 146 vendas. O gráfico 5.6 apresenta o desempenho utilizando as melhores categorias:

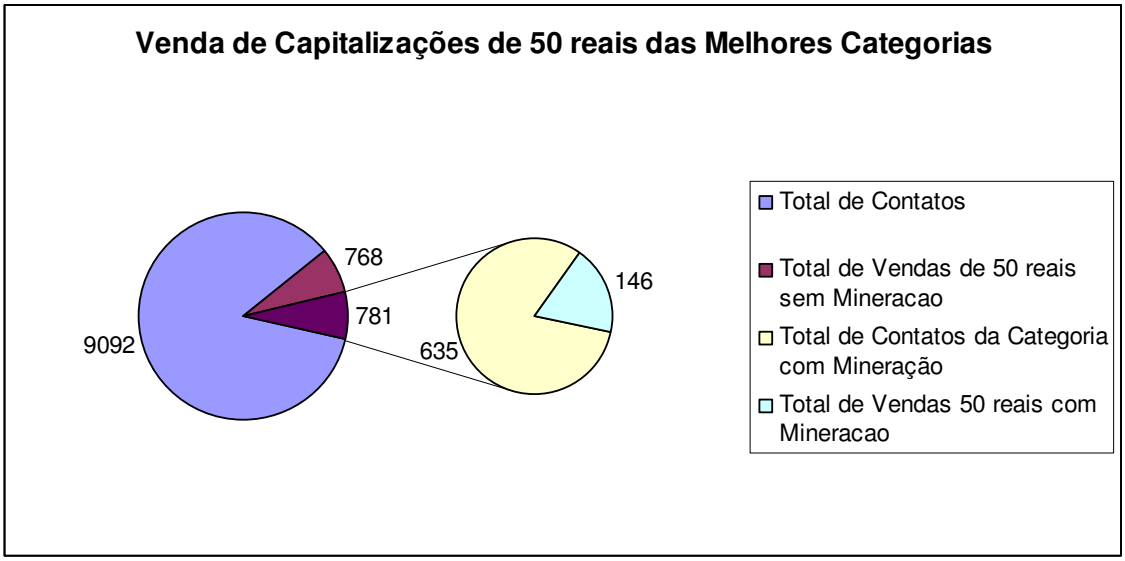

Gráfico 5.6 – Gráfico de Venda de Capitalizações de 50 reais das Melhores Categorias
Fonte: Do autor.

Nos primeiros 635 contatos realizados sem o uso de mineração de dados, observa-se a venda de 37 títulos de 50 reais. Por outro lado, através da utilização de uma árvore de decisão, os 635 contatos selecionados pelas regras geradas, renderam 146 vendas, ou seja, 394% superior.

Outro dado para ser considerado, é que a última venda desses 635 contatos foi fechada em 11 h e 25 min. A venda de número 146 somente foi alcançada no dia 20/09/2007, 3 dias após o início dos contatos.

Esse total de vendas foi atingido, sem que fosse dado foco para venda de títulos de 50 reais, se fosse dado foco nesse tipo de título, mais rapidamente poderia se chegar a um número de vendas maior. Dessa forma, se o responsável pelo *Call Center* necessitasse dar preferência para a venda de títulos de 50 reais, conseguiria 25,34% mais vendas em apenas

635 contatos, pois foram 146 vendas com mineração e 37 com a estratégia da empresa. O restante do tempo poderia ser utilizado para continuar a estratégia em outras bases ou ainda em outros registros dessa mesma base.

6 TESTES DE APLICAÇÃO DA MINERAÇÃO NO *CALL CENTER* EM TEMPO REAL

Este teste se destina a aplicar as regras da mineração diretamente no *Call Center*, ou seja, conforme as categorias definidas nas regras geradas pela classificação, serão disponibilizadas os clientes para serem contatados. Se as regras de classificação indicarem que os clientes que residem na região de Santa Catarina, com idade entre 30 e 39 anos, possuem um índice de compra elevado, somente estes tipos de clientes (categoria) serão enviados aos atendentes para oferecer o título.

Para aplicar esse tipo de teste foi preciso consentimento do setor responsável por este trabalho no *Call Center* para que os testes não atrapalhem o desempenho das vendas.

6.1 Teste de mineração em tempo real

Dando início aos testes, foi selecionado o período com o qual seriam realizados os experimentos. Durante o mês, várias listas de clientes podem estar disponíveis, a empresa opta, muitas vezes, em dividir as listas através de alguma característica ou então já recebe esta lista da empresa contratante de forma segmentada.

Nesse período, só havia disponível uma lista de um tipo de cliente que não é considerado um bom comprador e que deveriam ser contatados. Havia o período de 15 horas e 50 minutos até às 18 horas para aplicar a mineração no dia 23 de Outubro de 2007. Essa lista de clientes se tratava de uma lista apenas com clientes com faixa de valor limite do cartão de crédito entre 400 e 799 reais.

Desde o início do dia estavam sendo contatados todos os clientes, sem prioridade para qualquer característica e não deveria ser dado foco na venda de nenhum tipo específico de título de capitalização.

Da lista de clientes em questão, já haviam sido contatados mais de 3000 clientes. A primeira ação a ser realizada é a mineração desses registros a fim de que a classificação

retorne a árvore de classificação que apontará as melhores regras. Para isso, foram selecionados todos os registros que faziam parte dos clientes que adquiriram títulos de capitalização ou então que recusaram a compra, em um período de 17/09/2007 até 23/10/2007.

É preciso notar que os registros onde o contato não teve sucesso, como nos casos onde o telefone estava ocupado ou o cliente não atendeu a chamada, não foram considerados. O total de registros do arquivo a ser minerado foi de 3333 registros, sendo 907 vendas e 2426 rejeições.

A classificação foi aplicada com o método de meta aprendizagem AdaboostM1 com J48, conforme os outros testes e desses registros foram removidos os atributos SEXO (não estava informado na base), Classificação (todos os registros eram do tipo NÃO INFORMADO) e o produto (já que não havia necessidade de dar foco no produto). A mineração atingiu 2582 instâncias classificadas corretamente totalizando 77,4677% dos registros. No tabela 6.1 é apresentada a matriz de confusão dos testes:

Tabela 6.1 – Matriz de Confusão dos testes diretamente no *Call Center*

	A	B	Classificada como
A	279	628	A = SIM
B	123	2303	B = NAO

Fonte: Do autor

O software desenvolvido se encarregou de separar as melhores regras de toda a árvore de classificação e estão no quadro 6.1:

Quadro 6.1 – Amostra das regras geradas pelo AdaBoostM1

COD_TEMPO_CONTA = DE_7_A_12_MESES | COD_FAIXA_ETARIA = 40_A_49_ANOS | | COD_REGIAO_MAILING = NORDESTE_I: SIM (92.0/44.0) ---------- Acerto: 47.82608695652174%	COD_TEMPO_CONTA = DE_0_A_6_MESES | COD_FAIXA_ETARIA = 30_A_39_ANOS | | COD_REGIAO_MAILING = RJ-ES | | | DIA_VENC_FATURA <= 22: SIM (73.0/30.0) ---------- Acerto: 41.09589041095890%
COD_TEMPO_CONTA = DE_7_A_12_MESES | COD_FAIXA_ETARIA = 50_A_59_ANOS | | COD_REGIAO_MAILING = RJ-ES: SIM (81.0/37.0) ---------- Acerto: 45.67901234567901%	COD_TEMPO_CONTA = DE_7_A_12_MESES | COD_FAIXA_ETARIA = 50_A_59_ANOS | | COD_REGIAO_MAILING = MG: SIM (61.0/24.0) ---------- Acerto: 39.344262295081967%
COD_TEMPO_CONTA = DE_7_A_12_MESES | COD_FAIXA_ETARIA = 40_A_49_ANOS | | COD_REGIAO_MAILING = RJ-ES | | | DIA_VENC_FATURA <= 22: SIM (75.0/31.0) ---------- Acerto: 41.333333333333336%	COD_TEMPO_CONTA = DE_0_A_6_MESES | COD_FAIXA_ETARIA = 30_A_39_ANOS | | COD_REGIAO_MAILING = SP_CAPITAL | | | DIA_VENC_FATURA > 5 | | | | DIA_VENC_FATURA <= 26: SIM (44.0/13.0) ---------- Acerto: 29.545454545454547%

Fonte: Do autor

O próximo passo foi conferir a disponibilidade desses tipos de categorias de clientes dentro da base de clientes, que possui outros registros diferentes dos quais foi aplicada a mineração. Já que estas categorias possuem os melhores índices, teoricamente, bastaria disponibilizar aos atendentes esse tipo de cliente para melhorar as vendas. Segue a análise dos tipos de clientes disponíveis para serem trabalhados, na tabela 6.2:

Tabela 6.2 – Tipos de Clientes e quantidades disponíveis para serem trabalhados

Atributo	Valor	Quantidade
Valor de limite	DE 400 A 799	7524
Tempo de Conta	ACIMA DE 5 ANOS	94
Tempo de Conta	DE 1 A 2 ANOS	7201
Tempo de Conta	DE 4 A 5 ANOS	6
Faixa Etária	18 A 29 ANOS	420
Faixa Etária	30 A 39 ANOS	1499
Faixa Etária	40 A 49 ANOS	1998
Faixa Etária	50 A 59 ANOS	2326
Faixa Etária	ACIMA DE 60	1058
Região	CENTRO-OESTE	200
Região	MG	782
Região	NORDESTE I	1495
Região	NORDESTE II	678
Região	NORTE	356
Região	PR	406
Região	RJ-ES	1553
Região	RS	587
Região	SC	327
Região	SP CAPITAL	638
Região	SP INTERIOR	279

Fonte: Do autor

Porém, não havia como utilizar as regras geradas, pois só existiam clientes com tempo de conta de 1 a 2 anos e mais 100 na faixa acima de 4 anos. Nesse caso, a mineração não poderia ser aplicada.

Mas pelo fato de existirem muitos clientes com tempo de conta de 1 a 2 anos, poderia ser feita a mineração somente nesse tipo de cliente, que ainda não havia sido trabalhado nessa lista. Para isso foram selecionados todos os clientes no mesmo período selecionado anteriormente, que faziam parte de contatos efetivos, que pertenciam a qualquer outra lista trabalhada, mas que também possuíam valor de limite entre 400 e 799 reais e tempo de conta de 1 a 2 anos.

Não poderia ser feita a mineração somente na lista selecionada, pois os nomes com tempo de conta de 1 a 2 anos ainda não haviam sido trabalhados em grande número, por isso foram selecionados de outras listas já trabalhadas.

Dessa categoria de cliente, já haviam sido contatados 388 contatos efetivos, sendo 121 vendas e 267 rejeições. O resultado da mineração foi de 309 instâncias classificadas corretamente, chegando a um total de 79,6392%. A matriz de confusão da tabela 6.3 completa a análise do resultado dessa mineração:

Tabela 6.3 – Matriz de Confusão dos testes diretamente no *Call Center* dos 388 contatos

	A	B	Classificada como
A	60	61	A = SIM
B	18	249	B = NAO

Fonte: Do autor

Separando as melhores regras com o software desenvolvido se chegou até à amostra de regras do quadro 6.2:

Quadro 6.2 – Regras do teste no *Call Center* dos 388 registros

```
COD_REGIAO_MAILING = RJ-ES
|  COD_FAIXA_ETARIA = ACIMA_DE_60: SIM (59.0/27.0)
---------- Acerto: 45.76271186440678%

COD_REGIAO_MAILING = NORDESTE_II: SIM (38.0/15.0)
---------- Acerto: 39.473684210526315%
```

```
COD_REGIAO_MAILING = NORDESTE_I
|  COD_FAIXA_ETARIA = ACIMA_DE_60: SIM (34.0/12.0)
---------- Acerto: 35.294117647058826%
```

```
COD_REGIAO_MAILING = RJ-ES
|  COD_FAIXA_ETARIA = 50_A_59_ANOS
|  |  DIA_VENC_FATURA > 5
|  |  |  DIA_VENC_FATURA <= 26: SIM (25.0/8.0)
---------- Acerto: 32.0%
```

```
COD_REGIAO_MAILING = MG
|  COD_FAIXA_ETARIA = ACIMA_DE_60
|  |  DIA_VENC_FATURA <= 22
|  |  |  DIA_VENC_FATURA <= 12: SIM (23.0/10.0)
---------- Acerto: 43.47826086956522%
```

```
COD_REGIAO_MAILING = RS
|  DIA_VENC_FATURA > 8: SIM (17.0/7.0)
---------- Acerto: 41.1764705882353%
```

```
COD_REGIAO_MAILING = SP_INTERIOR
|  COD_FAIXA_ETARIA = ACIMA_DE_60
|  |  DIA_VENC_FATURA > 12: SIM (13.0/5.0)
---------- Acerto: 38.46153846153846%
```

Fonte: Do autor

Dessas regras então foram selecionadas as três primeiras, que proporcionaram uma quantidade aproximada de 780 clientes. Dessa forma, foram liberados apenas os clientes das regiões RJ-ES, Nordeste I, Nordeste II e com idade acima de 60 anos.

Essa categoria foi liberada a partir de 15 horas e 50 minutos. Seguem os totais de vendas por hora na tabela 6.4 do segmento selecionado para o teste:

Tabela 6.4 – Venda por faixa de horário

Hora	Tipo de Contato	Quantidade
09	VENDAS	12
10	VENDAS	11
11	VENDAS	14
12	VENDAS	18
13	VENDAS	10
14	VENDAS	15
15	VENDAS	11
16	VENDAS	22
17	VENDAS	6
18	VENDAS	7
19	VENDAS	22
20	VENDAS	15
21	VENDAS	10

Fonte: Do autor

Nota-se que a quantidade de vendas onde somente essa categoria estava disponível, na hora 16, foi bem maior do que nos outros horários quando vários tipos de clientes estavam sendo contatados, sendo até 100% maior do que o horário anterior (faixa das 15 horas) e maior que muitos outros horários do dia.

A venda nesse horário igualou-se em quantidade com os melhores horários de venda: ao meio dia e após o horário comercial. O horário das 19 horas que também atingiu um nível alto de vendas obteve contribuição da lista onde foi aplicada a mineração, já que 3 das 22 vendas realizadas na faixa das 19 horas faziam parte do teste de mineração, pois eram registros que foram armazenados na agenda pessoal dos atendentes e foram finalizados nesse horário.

No gráfico 6.1, podem-se visualizar as melhores faixas de horários de vendas obtidas do segmento:

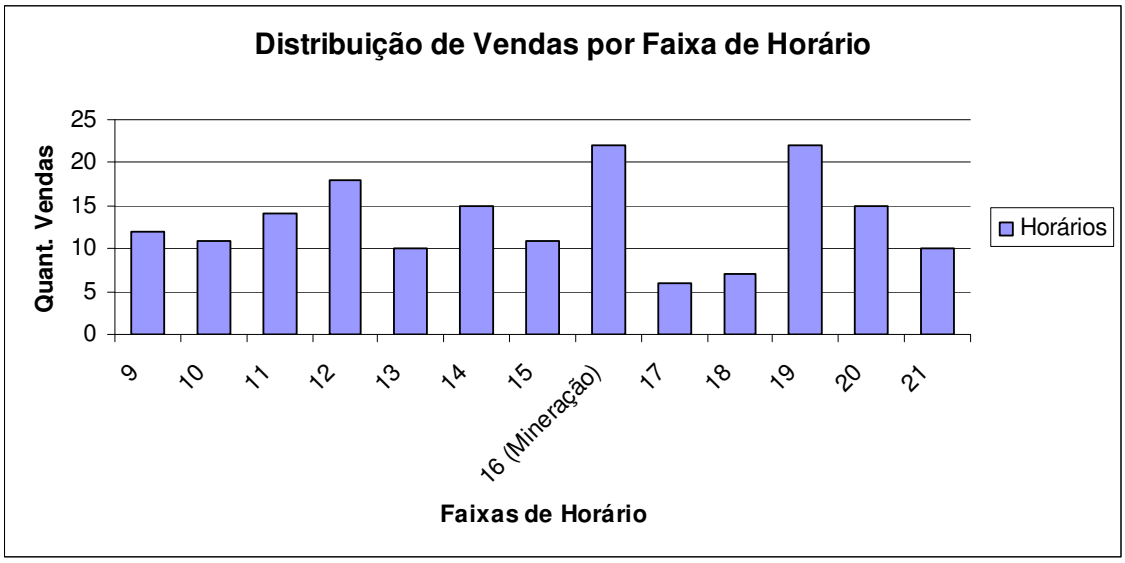

Gráfico 6.1 – Gráfico de quantidade de vendas por faixa de horário
Fonte: Do autor.

Para que seja feito um comparativo e comprovar os bons números do setor de venda de títulos de capitalização, são apresentados no gráfico 6.2 os gráficos de vendas de dois dias antes e depois do dia 23/10/2007, dia em que foi realizado o teste.

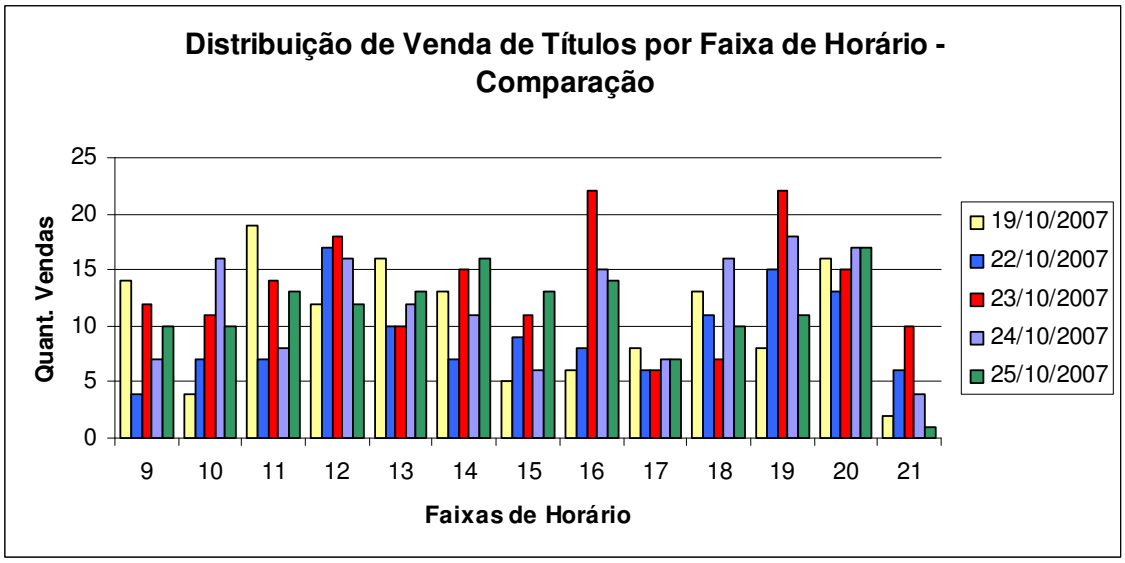

Gráfico 6.2 – Gráficos de vendas de 4 dias próximos a data dos testes no segmento
Fonte: Do autor.

Pode-se notar que as vendas no horário das 15 às 18 horas não possuem a maior média de rendimento. No entanto, no dia 23/10/2007 foram atingidos os melhores índices de

vendas nesse horário e a mineração ainda auxiliaram o horário das 19h do dia 23/10/2007 a chegar ao mesmo nível, pois foram realizadas 3 vendas com os registros classificados na mineração. As vendas desse horário das 16h não foram exclusivamente da lista utilizada para os testes, mas as vendas realizadas permitiram chegar a esse total. Essa lista de clientes atingiu seu melhor índice na faixa das 15h onde foi possível chegar a 6 vendas. Nesse horário somente esta lista estava disponível, porém sem filtro algum de mineração. Dessa maneira, se o rendimento sem mineração seguisse a faixa de horário anterior, o total de vendas da faixa das 16 horas seria mais difícil de ser alcançado.

O horário das 16h também atingiu um bom índice de faturamento, que seria a soma dos valores de títulos vendidos, superior ao esperado. No gráfico 6.3, é apresentado o gráfico por faixa de horário.

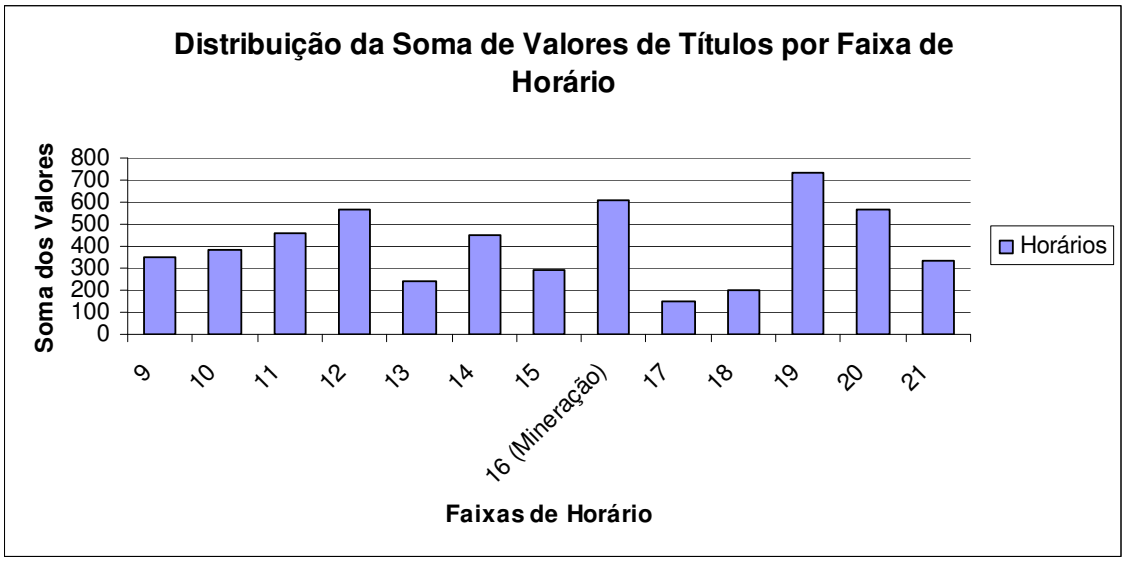

Gráfico 6.3 – Gráfico de faturamento por faixa de horário
Fonte: Do autor.

Até o fim do dia, utilizando a mineração, foram atingidas 19 vendas dessa lista com 746 efetivos de 3127 contatos. Das vendas, 12 delas foram fechadas em um período de 1 hora e 40 minutos de trabalho em um momento em que somente esta lista de clientes estava sendo utilizada (574 contatos). As outras 7 vendas foram realizadas em ligações durante o restante do dia ainda com os registros da mineração, sendo 2 vendas às 18 horas, 3 vendas às 19 horas e 2 vendas às 20 horas.

No gráfico 6.4, é apresentado um gráfico dos resultados do teste:

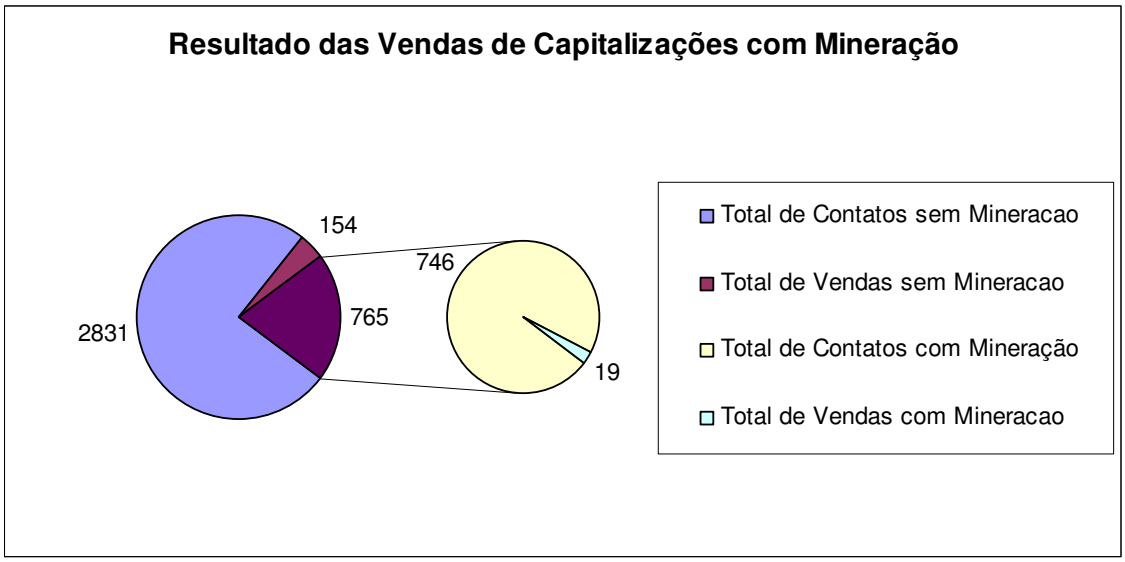

Gráfico 6.4 – Gráfico de quantidade de contatos e venda de títulos com os testes de mineração
Fonte: Do autor.

A taxa de conversão durante a faixa de horário das 16 horas do dia 23/10/2007 foi de 42,59% dos 54 contatos efetivos com 23 vendas, enquanto em horários com alta produção como 12h, foram alcançados 35,29% em 51 contatos efetivos e 18 vendas e até mesmo igualando a conversão de horários como 19 horas. Conforme o *Call Center*, uma variável que interfere na quantidade das vendas é o horário de contato, ou seja, independente do tipo de cliente, em média esse horário possui maior quantidade de vendas.

6.2 Segundo teste no *Call Center*

O segundo teste tem o objetivo de comprovar a eficiência da mineração, para isso continua utilizando a metodologia do primeiro teste. Mediante autorização da empresa, foi possível realizar outro teste com atendimentos realizados por um grupo de atendentes no dia 8 de Novembro de 2007. Outro teste foi iniciado no dia 7 de Novembro de 2007, mas por determinação da empresa, foi preciso ser cancelado.

Esse grupo de atendentes no dia 8 de Novembro tinha disponível uma lista de clientes onde parte dela já havia sido abordada. Já haviam sido realizados 4216 contatos efetivos e destes 279 haviam sido convertidos em vendas, taxa de 6,62% de conversão de contatos em vendas.

A primeira ação a ser tomada foi selecionar no banco de dados todos esses 4216 contatos para ser criado o arquivo arff para realizar a mineração. Para este teste, foram

utilizados os registros que já haviam sido trabalhados para realizar a garimpagem, com o intuito de continuar entrando em contato com a categoria de clientes que mais adquiriu títulos nessa lista de *mailing*. Os clientes que estavam disponíveis para serem contatados nessa listas possuíam diversas combinações de atributos e por este motivo não foi realizado nenhum tipo de filtro para, como por exemplo, somente selecionar os contatos com uma determinada faixa etária. Diversas categorias de clientes já haviam sido contatadas e várias categorias estavam disponíveis.

O Weka foi configurado com as mesmas opções para o AdaboostM1 com a utilização do J48 para realizar a mineração. O resultado do método de meta aprendizagem aplicado chegou a um percentual de 95,5171% de instâncias classificadas corretamente e com a matriz de confusão apresentada na tabela 6.5:

Tabela 6.5 – Matriz de Confusão do segundo teste direto no *Call Center*

	A	B	Classificada como
A	101	178	A = SIM
B	11	3926	B = NAO

Fonte: Do autor

A árvore de 198 folhas foi processada a fim de separar as melhores regras. Para este teste, devido à quantidade de registros disponíveis, foram selecionadas as duas regras que mais registros representam em relação ao total de contatos e que possuíam percentual de acerto elevado em relação às outras regras. As regras selecionadas são apresentadas no quadro 6.3:

Quadro 6.3 – Regras com melhor percentual de acerto e acerto de registros do segundo teste

```
COD_VALOR_LIMITE = DE_1000_A_1999
|  COD_FAIXA_ETARIA = 18_A_29_ANOS: SIM (677.0/303.0)
---------- Acerto: 44.7562776957164%
```

```
COD_VALOR_LIMITE = DE_1000_A_1999
|  COD_FAIXA_ETARIA = 30_A_39_ANOS
|  |  COD_TEMPO_CONTA = DE_0_A_6_MESES: SIM (571.0/278.0)
---------- Acerto: 48.68651488616462%
```

Fonte: Do autor

Para testar a eficiência da mineração cada uma das regras foi utilizada separadamente para utilizar na seleção dos *prospects* da lista em questão. Dessa forma, primeiro foram

abordados os clientes que possuem valor de limite de cartão de crédito entre 1000 e 1999 reais e que ainda estivessem dentro da faixa de 18 a 29 anos. Logo após tenham sido esgotados os contatos dessa categoria, os clientes da categoria da segunda regra (valor limite de 1000 a 1999, faixa etária de 30 a 39 anos e tempo de conta de 0 a 6) foram disponibilizados aos atendentes.

A lista de cliente no mesmo dia, do inicio da manhã até às 15 horas e 46 minutos, realizou 136 contatos efetivos entre 1024 contatos de qualquer tipo e atingiu 2 vendas. Para realizar esses contatos foram utilizados 72 atendentes.

A aplicação das regras se iniciou às 15 horas e 47 minutos, com 270 clientes disponíveis com valor limite de 1000 a 1999 e idade de 18 a 29 anos. Foram realizados todos os contatos, sendo destes 32 contatos efetivos. Os contatos foram esgotados às 16 horas e 25 minutos e se chegou a 2 vendas com taxa de conversão 6,25%. O número de atendentes envolvidos nesses contatos foi de 30.

Para dar continuidade nos testes a segunda categoria de cliente foi disponibilizada no mesmo momento do fim dos registros da categoria anterior. Nesse segundo momento, estavam disponíveis 215 registros da base de dados, que foram todos contatados até 16 horas e 45 minutos, devido ao tempo de atendimento ser menor e pelos problemas de localização dos *prospects*. Foram realizados apenas 18 contatos efetivos e 2 destes foram convertidos em venda com taxa de conversão de 11,11%. O gráfico 6.5 ilustra a comparação do desempenho com e sem a mineração:

Gráfico 6.5 – Gráfico de comparação da conversão do segundo teste em tempo real
Fonte: Do autor.

O gráfico 6.6 ilustra a comparação da quantidade de contatos com e sem a mineração:

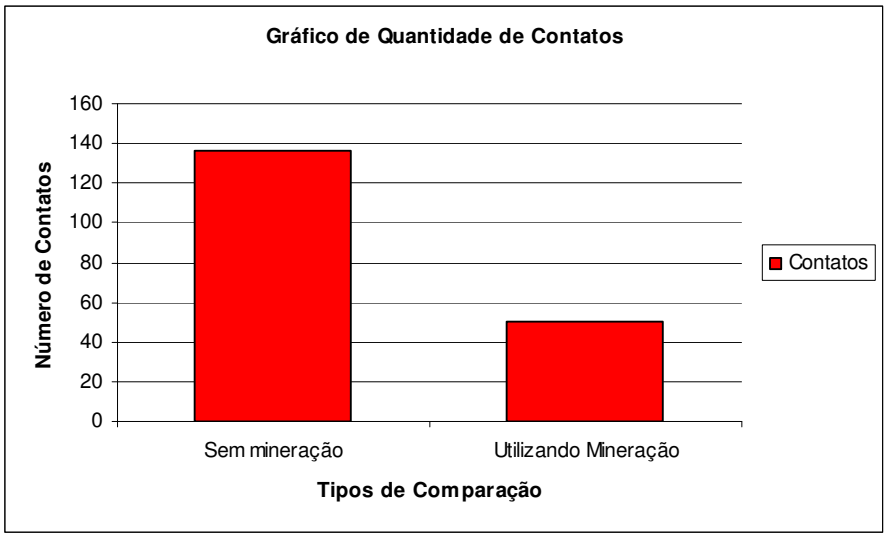

Gráfico 6.6 – Gráfico de comparação da conversão e contatos do segundo teste em tempo real
Fonte: Do autor.

Dessa maneira, foi possível chegar ao dobro de vendas com apenas 50 contatos efetivos, número menor do que os 136 contatos realizados anteriormente quanto não estava sendo aplicada nenhum tipo de mineração de dados.

CONCLUSÃO

Nota-se que o fato do *Call Center* não possuir nenhuma técnica de mineração de dados pode significar um mau aproveitamento dos dados disponíveis. Existem informações importantes que não estão disponíveis na visualização dos dados, que dizem respeito à relação entre as características dos clientes e o resultado do contato. São vários os fatores que influenciam as vendas, inclusive fatores de localização do cliente (quantidade de contatos efetivos) e melhor horário para contato. A combinação de mais de uma característica do cliente pode ser fundamental para definir o perfil da pessoa que mais adquire produtos.

A venda de títulos de capitalização foi selecionada para aplicação das técnicas de KDD pela quantidade de atributos presentes em cada registro de cliente a ser contatado, crescendo a possibilidade de aumentar o grau de qualidade e diversidade das informações geradas. A estrutura utilizada pela empresa facilita muito a aplicação de mineração, por este motivo não foi preciso nenhuma alteração no BD e nem foi necessário deslocar os dados para outro local e realizar alguma modificação, conforme explicado no capítulo 4.

Das técnicas de KDD, a classificação foi selecionada como a técnica a ser utiliza no problema. As árvores de classificação tornam possíveis organizar os atributos e a relação entre estes para os contatos com venda ou recusa. O *software* Weka possui todas as funcionalidades necessárias para aplicar KDD e dá todo o suporte para que os dados sejam garimpados. O desenvolvimento de um *software*, em linguagem Java, que utiliza as bibliotecas do Weka, possibilitou que o usuário possuísse as principais funcionalidades, algumas tarefas de pré-processamento e transformação realizadas de maneira automática e ainda que recebesse os resultados de maneira simplificada, com as regras separadas e filtradas conforme o nível exigido por ele próprio.

Nos testes iniciais do capítulo 5 foi possível conhecer os melhores resultados e como encontrar as melhores configurações dos algoritmos. Na seleção do algoritmo com melhor desempenho, o método de meta aprendizagem AdaboostM1, que utiliza *Boosting*, se comportou de maneira mais eficiente, atingindo melhor quantidade de instâncias classificadas

corretamente e por este motivo foi selecionado para a aplicação dos testes de comparação de desempenho de vendas e teste em tempo real.

Os testes de diminuição de contatos e aumento de vendas nos registros sem mineração, foi aplicado com o fim de comparar se fossem utilizadas técnicas de KDD nas transações já realizadas, se estas obteriam uma economia de contatos, aumento de vendas por contato efetivo e economia de tempo. Em todos os testes e comparativos, pôde ser identificado que muitos contatos não precisariam ser realizados e que o número de vendas poderia ser melhor. Para esse teste, a mineração foi realizada em contatos já realizados e simulou da tomada de decisão do responsável, caso conhecesse os melhores perfis de venda.

Tendo poder desses perfis de clientes, liberando aos atendentes cada um deles de maneira separada, é possível ter um nível de acerto muito maior e vender mais, já que são os tipos de clientes que tem maior histórico de compra. Automaticamente, a quantidade de contatos é menor, visto que muitos clientes não seriam contatados e o tempo necessário para chegar a certo número de vendas também seria menor.

O capítulo 6 apresentou os testes em tempo real, ao invés de realizar simulações da tomada de decisão e como seria a economia e aumento de vendas, foi realizada a garimpagem para descobrir os tipos de clientes que mais compram para cada caso e os utilizou para selecionar os clientes que seriam contatados. Os resultados desses testes comprovaram que menos contatos e menos tempo é utilizado para realizar mais vendas. O desempenho do *Call Center* é melhorado devido ao fato de que muitos clientes não são contatados e que conforme as regras geradas, a priorização dos registros permite que o *Call Center* priorize contato com os melhores tipos de clientes.

Para trabalhos futuros pode ser indicada a continuação dos testes utilizando classificação, selecionando outro algoritmo ou ainda outras configurações, tanto do modelo como do próprio algoritmo. São indicados os testes utilizando outros períodos para a amostra dos dados e até mesmo outros dados e atributos. Testes podem ser feitos em períodos maiores de tempo e aplicando em horários com maior quantidade de localização de clientes. Pode ser feito um estudo mais aprofundado da economia de tempo e economia de mão de obra ao utilizar mineração. Podem ser realizados trabalhos com a utilização de outras técnicas de KDD para encontrar outros resultados. Outra proposta seria a utilização de KDD em ambiente de *Call Center* ativos que utilizam discadores automáticos, que possuem uma quantidade de contatos efetivos maiores e que por si só já são uma vantagem competitiva em relação a outras empresas.

O *software* pode ser melhorado em trabalhos futuros, aumentando sua capacidade de trabalhar com outras técnicas e pode facilitar a inclusão de outros algoritmos, que na versão atual exige conhecimento dos pacotes e das classes do Weka. A seleção dos dados pode ser feita de maneira mais simples e sem a exigência de utilizar SQL, bem como o tratamento dos dados pode ser melhorado.

Ainda como proposta, pode ser utilizada uma outra aplicação de mineração para comparar com o Weka, que foi a única ferramenta utilizada. Essa nova ferramenta pode apresentar melhor os resultados e pode dispensar a criação de uma aplicação para facilitar a visualização dos mesmos.

Com base neste trabalho, além de conhecer os perfis dos compradores em potencial dos produtos, foi possível unir o conhecimento adquirido pelos analistas de informações da empresa de *telemarketing* e o conhecimento descoberto com a mineração. Desse modo, a utilização de técnicas de KDD em um *Call Center* auxiliou na seleção mais eficiente dos *prospects*, aumentando as vendas nos períodos aplicados e reduzindo a quantidade de contatos telefônicos.

REFERÊNCIAS BIBLIOGRÁFICAS

AGRAWAL, Rakesh; IMIELINSKI, Thomas; SWAMI, Arun. Mining Association Rules between Sets of Items in Large Databases. In: ACM SIGMOD International Conference on Management of Data 5/93, 1993, Washington/USA. **Proceedings of SIGMOD 5/93**. Washington/USA, 1993. p. 207-216.

AGRAWAL, Rakesh; SRIKANT, Ramakrishnan. Fast Algorithms for Mining Association Rules. In: 20th International Conference on Very Large Databases (VLDB) CONFERENCE, 1994, Santiago/Chile. **Proceedings of the 20th International Conference on Large Databases**. Santiago/Chile, 1994. p. 487-498.

ATENDEBEM. São Leopoldo - RS. Atendebem Soluções de Atendimento, 2007. Website da empresa de *Call Center*. Disponível em: <http://www.atendebem.com.br>. Acesso em: 03 jun. 2007.

BRACHMAN, Ronald J.; ANAND, Tej. The Process of Knowledge Discovery in Databases. In: FAYYAD et al. **Advances in knowledge discovery and data mining**. Cambridge-Mass:AAAI/MIT Press, 1996. p. 37-57.

CABENA, Peter et al. **Discovering Data Mining from Concept to Implementation**. New Jersey-USA: Prentice Hall PTR, 1997. 193 p.

CARVALHO, Juliano V.; SAMPAIO Marcus C.; MONGIOVI, Giuseppe. Utilização de Técnicas de "Data Mining" para o Reconhecimento de Caracteres Manuscritos. In: XIV SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, 1999, Florianópolis. **XIV Simpósio Brasileiro de Banco de Dados - Anais.** Florianópolis, 1999. p. 235-249.

CORMEN, Thomas H. et al. **Algoritmos:** teoria e prática. 2. ed. Rio de Janeiro: Campus, 2002. p. 21-25

DANTAS, Edmundo Brandão. **Telemarketing:** a chamada para o futuro. 2. ed. São Paulo: Atlas, 1994. 206 p.

DBDESIGNER. FabForce.net, 2007. Apresenta todas as características do projeto, documentação e o software DBDesigner. Disponível em: < http://fabforce.net/dbdesigner4/>. Acesso em: 18 jul. 2007.

DWBRASIL. DW Brasil, 2007. Website sobre banco de dados e Data Warehouse. Disponível em: <http://www.dwbrasil.com.br/html/dmining.html>. Acesso em: 28 mai. 2007.

FAYYAD, Usama M.; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. From Data Mining to Knowledge Discovery: An overview. In: FAYYAD et al. **Advances in knowledge discovery and data mining**. G. Cambridge-Mass:AAAI/MIT Press, 1996. p. 1-27.

FELBER, Edmilson J. W. **Proposta de Uma Ferramenta OLAP em um Data Mart Comercial:** Uma Aplicação Prática na Indústria Calçadista. Novo Hamburgo: 2005. 102 p.

Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Instituto de Ciências Exatas e Tecnológicas, Feevale, 2005.

FRAWLEY, William J.; PIATETSKY-SHAPIRO, Gregory; MATHEUS, Christopher J. Knowledge Discovery in Databases: An overview. In: AI Magazine, 1992, Menlo Park. **American Association for Artificial Intelligence**. Menlo Park, CA, USA, 1992. p. 57-70.

FREITAS, Alex A. Data Mining, In: XIII Simpósio Brasileiro de Banco de Dados. Maringá/Brasil, 1998. 104 p.

HOAGLIN, David C.; MOSTELLER, Frederick; TUKEY, John W. **Análise exploratória de dados**: técnicas robustas: um guia. Lisboa: John Wiley, 1992. 446 p.

LÓPEZ, José M. M.; HERRERO, Jesús G. **Aplicaciones Prácticas Utilizando Microsoft Excel y Weka** . Apostila Técnicas de Análises de Dados. S.l., 2004. 160 p. Disponível em: <http://galahad.plg.inf.uc3m.es/~docweb/ad/transparencias/apuntesAnalisisDatos.pdf>. Acessado em: 02 jun. 2007.

MARTINHAGO, Sergio. **Descoberta de Conhecimento sobre o processo seletivo da UFPR**. Curitiba: 2005. 114 p. Dissertação (Mestrado em Ciências) – Departamento de Matemática, UFPR, 2005.

MONGIOVI, Giuseppe. **T.E.I. Data Mining**. Notas de Aula Data Mining. Campina Grande, 1998. p. 1-102.

OLIVEIRA Fernando L. et al. Utilização de Algoritmos Simbólicos para a Identificação do Número de Caroços do Fruto Pequi, In: IV Encontro de Estudantes de Informática do Estado do Tocantins**,** 2002, Palmas. **Encontro de Estudantes de Informática do Tocantins – Encoinfo**. Palmas, 2002. p. 34-43.

PINHEIRO, Luciane C. **Método de Representação Espacial de Clustering**. Curitiba: 2006. 123 p. Dissertação de Mestrado – Departamento de Informática, UFPR, 2006.

PRODANOV, Cleber C. **Manual de Metodologia Científica**. 3. ed. Novo Hamburgo: Editora Feevale, 2006. 77 p.

QUINLAN, Ross. Induction of Decision Trees. In: SHAVLIK, Jude (ed.); DIETTERICH, Thomas (ed.). **Readings in Machine Learning**, San Mateo, CA: Morgan Kaufmann Publishers, 1990. p. 81-106.

QUINLAN, Ross. **C4.5:** Programs for Machine Learning. San Mateo, CA: Morgan Kaufmann Publishers, 1993. p. 1-109

QUINLAN, Ross. Improved use of continuous attributes in C4.5. In: Journal Of Artificial Intelligence Research 4, 1996, p. 77-90. Disponível em: <http://www.jair.org/media/279/live-279-1538-jair.pdf>. Acessado em: 02 jun. 2007.

SANTOS, Rafael. **Weka na Munheca:** Um guia para uso do Weka em scripts e integração com aplicações em Java. Apostila Princípios e Aplicações de Mineração de Dados. S.l., 2005. 20 p. Disponível em: <http://www.lac.inpe.br/~rafael.santos/CAP/cap359/2005/weka.pdf>. Acessado em: 02 jun. 2007.

Savasere, Ashok; Omiecinsky, Edward; Navathe, Shamkant. An Efficient Algorithm for Mining Association Rules in Large Databases. In: 21st International Conference on Very Large Databases (VLDB) Conference, 1995, Zurich/Suiça. **VLDB Journal**. Zurich/Suíça, 1995. p. 432-444.

SODRÉ, Ulysses. **Transformadas de Laplace**. Notas de Aula Computação, Engenharia Elétrica e Engenharia Civil. S.l., 2003. 39 p.

SRIKANT, Ramakrishnan; AGRAWAL, Rakesh. Mining Quantitative Association Rules in Large Relational Tables. In: ACM SIGMOD International Conference on Management of Data 6/96, 1996, Montreal-Canadá. **Procedings**. Montreal, 1994. p. 1-12.

TUKEY, John W. **Exploratory Data Analysis**. Reading: Addison-Wesley, 1977. p. 95-167

WEKA 3: Data Mining Software in Java. Nova Zelândia. Universidade de Waikato, 2007. Apresenta todas as características do projeto, documentação e o software Weka. Disponível em: <http://www.cs.waikato.ac.nz/ml/weka/index.html>. Acesso em: 28 mar. 2007.

WIEDERHOLD, Gio. On the barriers 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.

WITTEN, Ian H.; FRANK, Eibe. **Data Mining:** practical machine learning tools and techniques. 2. ed. San Francisco: Morgan Kaufmann, 2005. 525 p.