UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

MINERAÇÃO DE DADOS PARA GERAÇÃO DE ÁRVORE DE DECISÃO: APLICAÇÃO VENDAS DE VAREJO

LUCAS DALCOL PEREIRA

BLUMENAU
2017

LUCAS DALCOL PEREIRA

MINERAÇÃO DE DADOS PARA GERAÇÃO DE ÁRVORE DE DECISÃO: APLICAÇÃO EM VENDAS DE VAREJO

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Ciência da Computação do Centro de Ciências Exatas e Naturais da Universidade Regional de Blumenau como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Prof. Mauro Marcelo Mattos, Dr. Eng. - Orientador

BLUMENAU
2017

MINERAÇÃO DE DADOS PARA GERAÇÃO DE ÁRVORE DE DECISÃO: APLICAÇÃO EM VENDAS DE VAREJO

Por

LUCAS DALCOL PEREIRA

Trabalho de Conclusão de Curso aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II pela banca examinadora formada por:

Presidente:	Prof. Mauro Marcelo Mattos, Dr. Eng. – Orientador, FURB
Membro:	Prof. Everaldo Artur Grahl, Mestre Eng. – FURB
Membro:	Prof. Adriano Gonçalves Polidoro, MBA – FURB

Blumenau, 3 de julho de 2017

Dedico este trabalho a minha família e a todas as pessoas que me apoiaram nesta jornada.

AGRADECIMENTOS

Ao meu orientador, Mauro Marcelo Mattos, pelo apoio e dedicação para o desenvolvimento deste trabalho.

Ao meu pai Edson e minha mãe Mylene, que são meus grandes exemplos de caráter e dedicação.

Aos meus irmãos Júnior e Bruna, pela cumplicidade e alegria.

À minha esposa Marina pela sensibilidade, compreensão, apoio e paciência durante a realização deste trabalho.

Aos meus amigos, pelas conversas e risadas.

Aos professores do curso de Ciências da Computação da Universidade Regional de Blumenau, por todo o conhecimento compartilhado.

“No matter what people tell you, words and ideas can change the world”

Robin Williams

RESUMO

O presente trabalho descreve o processo de modelagem e desenvolvimento de um protótipo de aplicação de mineração de dados sobre um substrato de uma base de dados de vendas de varejo. O trabalho descreve as etapas de tabulação, transformação, limpeza e geração dos dados em um arquivo ARFF para a aplicação do algoritmo J48, através da biblioteca Java disponibilizada pela ferramenta Weka. O processo de transformação aplicado é apresentado e a versão final da árvore de decisões é apresentada. Os resultados obtidos demonstram a viabilidade do processo aplicado e confirmam a necessidade do processamento e mineração dos dados para a classificação e determinação dos padrões dos dados para obtenção de informações relevantes à tomada de decisão, sobretudo para obter um ganho de competitividade no mercado de vendas de varejo. Contudo, para um resultado mais efetivo, faz-se necessário ampliar o horizonte temporal dos dados bem como ampliar o escopo, envolvendo, por exemplo, dados sobre as compras realizadas e dados sobre os clientes.

Palavras-chave: Mineração de dados. Weka. ARFF. J48. Vendas de Varejo. Árvore de decisão.

ABSTRACT

The present work describes the process of modeling and developing a data mining application prototype on a substrate of a retail sales database. The work describes the steps of tabulating, transforming, cleaning and generating the data in an ARFF file for the application of the J48 algorithm, through the Java library provided by the Weka tool. The applied transformation process is presented and the final version of the decision tree is presented. The obtained results demonstrate the feasibility of the applied process and confirm the need for data processing and mining for the classification and determination of data standards to obtain information relevant to decision making, especially for gaining competitiveness in the retail sales market. However, for a more effective result, it is necessary to extend the time horizon of the data as well as to extend the scope involving, for example, data on the purchases made and the data on clients.

Key-words: Data mining. Weka. ARFF. J48. Retail sales. Decision tree.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ARFF – Attribute-Relation File Format

KDD – Knowledge Discovery in Databases

Weka – Waikato Environment for Knowledge Analysis

SUMÁRIO

1 INTRODUÇÃO

O volume de dados armazenados atualmente é gigantesco e continua crescendo rapidamente pelo mundo todo (REZENDE, 2003, p. 307). Devido a tamanha quantidade de dados, muita informação e conhecimento úteis podem estar sendo desperdiçados (REZENDE, 2003, p.307). Sendo assim, há a necessidade de se desenvolver novas ferramentas e técnicas de extração de conhecimento a partir destes dados armazenados, tornando o processo de mineração de dados, para obtenção de conhecimento útil e interessante, indispensável para o processo de tomada de decisão (REZENDE, 2003, p. 307).

As estratégias para obtenção de ganho de competitividade entre as empresas devem ser baseadas em informações concretas, visando a minimização de erros para a tomada de decisões por parte dos gestores (DANTAS et al., 2008, p. 1). Estas informações são geradas e inseridas nas bases de dados destas empresas, gerando uma grande gama de dados.

Mesmo com uma base de dados com informações tão importantes, torna-se lenta e custosa a pesquisa destes dados para estratificação e classificação de forma manual. No entanto, existem algumas ferramentas que auxiliam na coleta destes dados, como a ferramenta Waikato Environment for Knowledge Analysis (Weka).

O termo Knowledge Discovery in Databases (KDD) foi formalizado em 1989 em referência ao conceito de identificar conhecimento a partir de base de dados (MACEDO; MATOS, 2010). Segundo Librelotto e Mozzaquatro (2013, p. 29), mineração de dados ou *data mining*, "envolve um conjunto de técnicas e ferramentas computacionais usadas para a identificação desses padrões (conhecimentos) embutidos em grandes massas de dados".

Diante do exposto, este trabalho propõe o desenvolvimento de um protótipo de software de mineração, o qual, aplicando técnicas de KDD, utilize o algoritmo para geração de árvore de decisão da ferramenta de mineração de dados Weka. A aplicação executará sobre um extrato de uma base de dados coletados de uma empresa de vendas de varejo. A expectativa é que a árvore de decisões obtida possa, de alguma forma, contribuir para a tomada de decisões na empresa.

1.1 OBJETIVOS

O objetivo geral deste trabalho é desenvolver um protótipo de software que realize a mineração de dados em informações de vendas de varejo para geração de árvore de decisão.

Os objetivos específicos são:

a) modelar na prática todas as etapas do processo de KDD;

b) disponibilizar uma aplicação que permita a visualização da árvore de decisões produzida;
c) validar o modelo produzido a partir de uma base de dados de exemplo.

1.2 ESTRUTURA

Este trabalho está organizado em capítulos, onde:

a) o capítulo dois apresenta uma introdução e explicação da ferramenta Weka, explicação sobre o arquivo ARFF que é utilizado pela ferramenta para estratificação dos dados, uma introdução ao KDD e suas etapas e são apresentados os trabalhos correlatos e a relação com o trabalho atual;
b) o capítulo três apresenta o desenvolvimento do software, os requisitos funcionais e não funcionais, os diagramas, a implementação do software e a análise dos resultados obtidos;
c) o capítulo quatro apresenta a conclusão obtida com o desenvolvimento deste trabalho e as possíveis extensões para que outros acadêmicos possam utilizar este trabalho como base e dar continuidade.

2 FUNDAMENTAÇÃO TEÓRICA

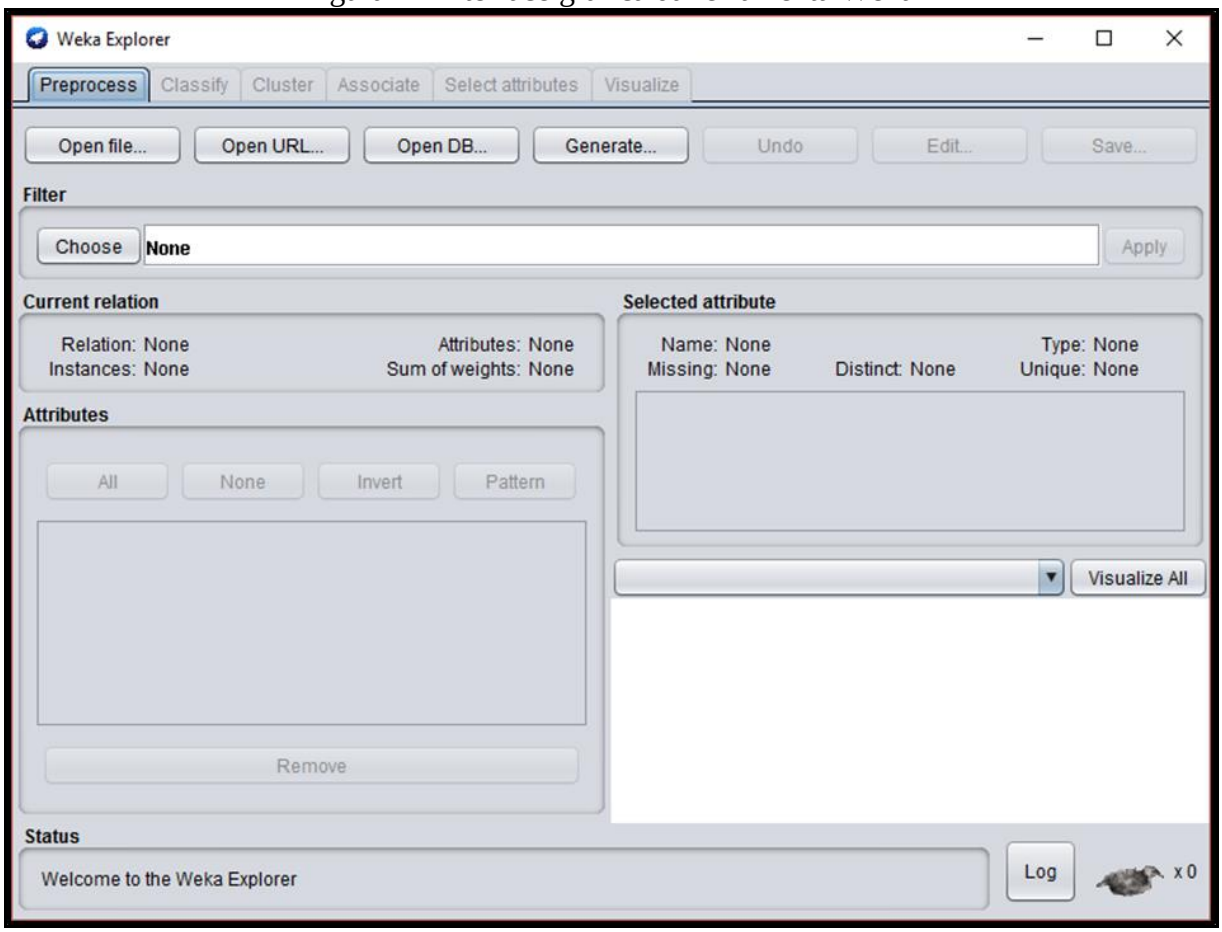
A seguir é apresentada uma introdução à ferramenta Weka, a explicação do formato de arquivo Attribute-Relation File Format (ARFF), que será utilizado neste trabalho para a mineração de dados, a explicação do algoritmo J48 e do processo de Knowledge Discovery in Databases (KDD) e os trabalhos correlatos.

2.1 FERRAMENTA WEKA

Weka é uma ferramenta de KDD que contempla uma série de algoritmos para preparação de dados. Esses algoritmos são voltados ao aprendizado de máquina (mineração) e de validação de resultados (SILVA, 2004). Weka foi desenvolvido na Universidade de Waikato na Nova Zelândia, sendo escrito em Java e possuindo código aberto disponível na Web (SILVA, 2004). Atualmente a ferramenta está na versão estável 3.8 e em desenvolvimento a versão 3.9. Segundo Silva (2004, p. 15), “Grande parte de seus componentes de software são resultantes de teses e dissertações de grupos de pesquisa desta universidade”.

Conforme apresentado na Figura 1, o sistema possui uma interface gráfica amigável (SILVA, 2004). Grande parte de seus recursos são acessíveis via interface gráfica, os demais podem ser utilizados programaticamente através de aplicações externas (SILVA, 2004). Com a aplicação da ferramenta, é possível gerar relatórios com dados analíticos e estatísticos dos registros minerados do banco de dados (SILVA, 2004). Uma limitação é a sua escalabilidade, uma vez que suas versões atuais limitam o volume de dados a ser manipulado à dimensão de memória principal (SILVA, 2004).

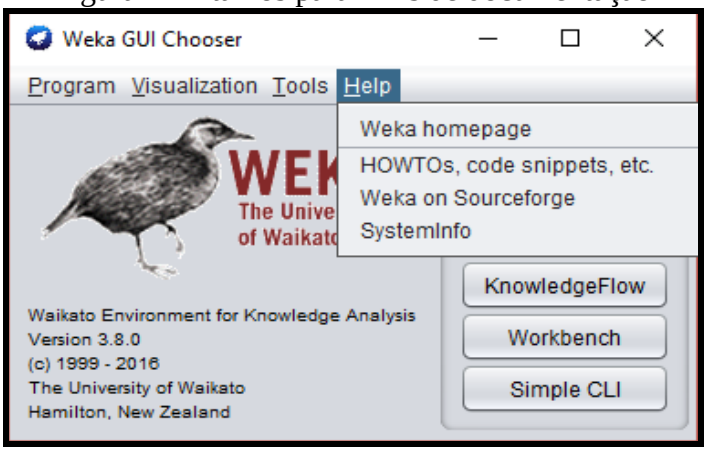
Figura 1 - Interface gráfica da ferramenta Weka



Fonte: Frank, Hall e Witten (2016).

Na ferramenta são disponibilizados links para auxiliar na configuração, parametrização, e utilização da aplicação, conforme é apresentado na Figura 2. Na opção 'HOWTOs, code snippets, etc', o sistema direciona o usuário para a wiki, na qual são apresentados tutoriais, questionamentos de outros usuários sobre eventuais situações e também o fórum para troca de informações entre usuários. Todas as páginas da wiki são colaborativas.

Figura 2 - Atalhos para links de documentação



Fonte: Frank, Hall e Witten (2016).

A ferramenta Weka permite ao usuário:

a) escolher e modificar os dados utilizados;
b) treinar e testar sistemas de aprendizagem;
c) realizar a aplicação de 56 algoritmos de classificação;
d) realizar análise de cluster;
e) configurar regras de associação para os dados;
f) selecionar os atributos mais relevantes;
g) visualizar gráfico 2D interativo dos dados e etc.

A partir da versão 3.2.1 da ferramenta Weka, é disponibilizada além da opção de realizar a aplicação dos algoritmos através do arquivo ARFF, realizar conexão e extrair as informações diretamente do banco de dados.

2.2 ARQUIVO ARFF

ARFF é um formato de arquivo que fora desenvolvido para o projeto de aprendizado de máquina do Departamento de Ciências da Computação da Universidade de Waikato para que fosse utilizado na ferramenta Weka (ARFF, 2016.). O arquivo é construído em duas seções distintas. A primeira seção é o *Header* e a segunda é a *Data*, que são explicados nas seções 2.2.1 e 2.2.2 respectivamente.

2.2.1 Header

O *Header* contém o nome da relação dos dados, a lista de atributos e seus respectivos tipos de dados (ARFF, 2016). O nome da relação é definido na primeira linha do arquivo ARFF, conforme demonstrado no Quadro 1.

Quadro 1 - Definição do nome da relação

```
@relation <relation-name>
```

Fonte: ARFF (2016).

A anotação `@relation` define a propriedade relação e o `<relation-name>` deve ser substituído pelo texto correspondente ao nome da relação.

Os atributos são as colunas que serão utilizadas na seção `Data`, deve ser uma sequência ordenada, quebradas por linha. Cada atributo deve conter um texto sem espaços e o tipo de dado esperado nesta coluna, conforme demonstrado no Quadro 2.

Quadro 2 - Definição do atributo

```
@attribute <attribute-name> <datatype>
```

Fonte: ARFF (2016).

A anotação `@attribute` define que será adicionado um atributo, o `<attribute-name>` deve ser substituído pelo nome desejado do atributo e o `<datatype>` deve ser substituído por um dos seguintes valores:

a) `numeric`: valor numérico;
b) `integer`: será tratado como valor numérico;
c) `real`: será tratado como valor numérico;
d) `string`: texto;
e) `date [<date-format>]`: valor do tipo data, onde é possível definir o formato esperado da data. A formatação deve ser adicionada no `<date-format>` onde são aceitos os mesmos formatos da classe `SimpleDateFormat` da biblioteca do Java. Por padrão é utilizado o formato ISO-8601 (`yyyy-MM-dd'T'HH:mm:ss`) que combina a data e hora.
f) `nominal`: é uma lista de valores possíveis que deve ser adicionado em formato de texto entre chaves e separados por vírgula, por exemplo `{Iris-setosa,Iris-versicolor,Iris-virginica}`.

2.2.2 Data

É a seção no arquivo que determina que na sequência serão listados os dados. Em comparação a uma tabela de banco de dados relacional, seriam os registros. A seção é delimitada pela anotação `@data` e na sequência uma quebra de linha, conforme o Quadro 3.

Quadro 3 - Definição do início dos dados

```
@data
```

Fonte: ARFF (2016).

Logo abaixo da anotação de início dos dados, são listados os registros. Estes registros devem seguir a mesma sequência dos atributos que foram definidos no arquivo e separados por ponto e vírgula. Conforme demonstrado no Quadro 4.

Quadro 4 - Arquivo ARFF

```
@RELATION iris

@ATTRIBUTE sepallength  NUMERIC
@ATTRIBUTE sepalwidth   NUMERIC
@ATTRIBUTE petallength  NUMERIC
@ATTRIBUTE petalwidth   NUMERIC
@ATTRIBUTE class        {Iris-setosa,Iris-versicolor,Iris-virginica}

@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
```

Fonte: ARFF (2016).

2.3 ALGORITMO J48

O algoritmo J48 é implementado em Java em uma versão posterior e ligeiramente melhorada do algoritmo C4.5 revisão 8, que fora implementado na linguagem de programação C (FRANK; HALL; WITTEN, 2016). Segundo Librelotto e Mozzaquatro (2013, p. 30) “Ele tem a finalidade de gerar uma árvore de decisão baseada em um conjunto de dados de treinamento, sendo este modelo usado para classificar as instâncias no conjunto de teste”.

O algoritmo tem grande utilização por especialistas em Mineração de Dados por ser adequado para procedimentos que envolvem variáveis qualitativas contínuas e discretas presentes nas bases de dados (LIBRELOTTO; MOZZAQUATRO, 2013). O algoritmo apresenta o melhor resultado na geração de árvore de decisão, aplicado em dados de treinamento (LIBRELOTTO; MOZZAQUATRO, 2013).

Wu et al. (2007) descreve a implementação do C4.5 em etapas, para melhor compreensão. Wu et al. (2007) cita como exemplo um conjunto de casos S, onde a árvore inicial é criada e então, utilizando o modo dividir-e-conquistar da seguinte forma:

a) Se todos os casos em S pertencem à mesma classe ou S é pequeno, a árvore é uma folha marcada com a classe mais frequente em S;
b) Caso contrário, obtém um teste com base em um único atributo com dois ou mais resultados. Faz deste teste a raiz da árvore com um ramo para cada resultado do teste, partição S nos subconjuntos correspondentes S1, S2, ... de acordo com o

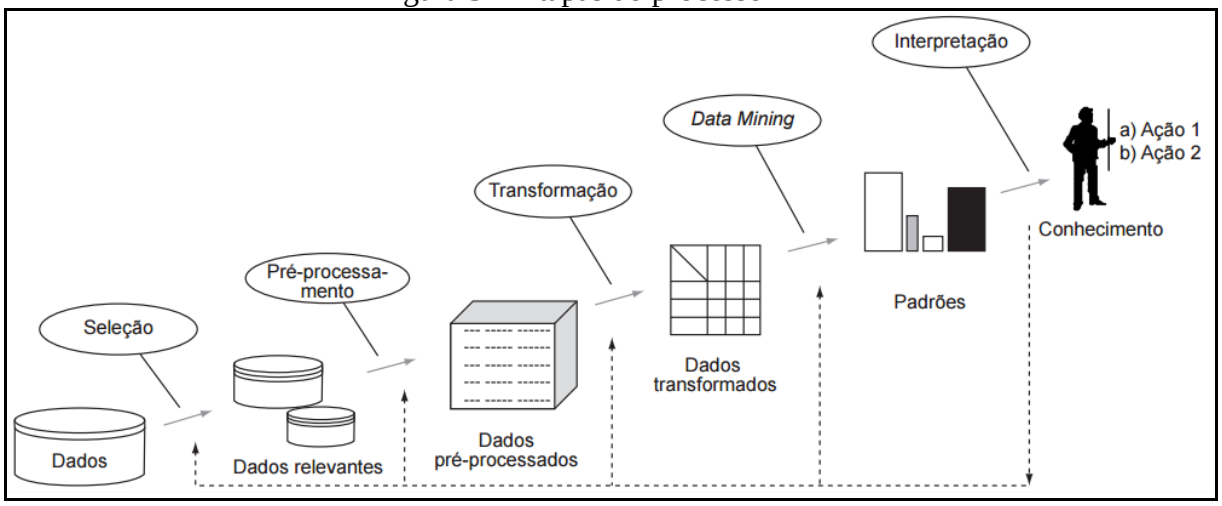
resultado para cada caso e aplica o mesmo procedimento de forma recursiva para cada subconjunto.

Geralmente, há muitos testes que podem ser escolhidos nesta última etapa. O algoritmo usa dois critérios heurísticos para classificar os possíveis testes: ganho de informação e a relação de ganho padrão que divide a informação adquirida pela informação fornecida pelos resultados do teste (WU et al., 2007).

2.4 ETAPAS DO KDD

Segundo Fayyad (1996, p. 39) "KDD é um processo, de várias etapas, não trivial, interativo e iterativo, para identificação de padrões compreensíveis, válidos, novos e potencialmente úteis a partir de grandes conjuntos de dados". O processo é formado por cinco etapas contínuas e integradas que compartilham o conhecimento adquirido a partir da base de dados, conforme demonstrado na Figura 3.

Figura 3 - Etapas do processo KDD



Fonte: Fayyad et al. (2006).

Na etapa de Seleção é definido o domínio sobre o qual se pretende executar o processo de análise e descoberta para então realizar a seleção e coleta do conjunto de dados da base (LIBRELOTTO; MOZZAQUATRO, 2013).

Na etapa de Pré-processamento é realizada a limpeza dos dados e as informações que são consideradas desnecessárias são removidas (LIBRELOTTO; MOZZAQUATRO, 2013). O objetivo desta etapa é reduzir a complexidade do problema, selecionando entre os dados apenas os atributos relevantes, permitindo que os dados que serão disponibilizados para as demais etapas sejam processados mais rapidamente pelo algoritmo (LIBRELOTTO; MOZZAQUATRO, 2013).

Na etapa de Transformação é realizada a normalização, discretização de atributos quantitativos, transformação de atributos qualitativos em quantitativos, entre outros. O objetivo desta etapa é transformar os atributos em um formato adequado para o algoritmo (LIBRELOTTO; MOZZAQUATRO, 2013).

A etapa de *Data Mining* é considerada a principal etapa do processo de KDD, onde é realizada a extração e a descoberta de padrões (LIBRELOTTO; MOZZAQUATRO, 2013).

A técnica de classificação é uma das técnicas de mineração de dados mais estudadas pela comunidade científica de KDD, e o princípio desta técnica é descobrir algum relacionamento entre os atributos preditivos e o atributo meta, de modo a descobrir um padrão que possa ser utilizado para previsões (LIBRELOTTO; MOZZAQUATRO, 2013).

A etapa de Interpretação é realizada através de visualizações, realizando a remoção dos padrões redundantes ou irrelevantes e traduzindo os padrões importantes em condições que podem ser facilmente compreendidas pelos usuários finais (TENFEN, 2003).

Todas as etapas podem retornar para a etapa anterior, conforme forem removendo os dados irrelevantes para o processo e então processar os dados relevantes. O processo de KDD consiste neste ciclo até que seja obtido o resultado válido esperado.

2.5 TRABALHOS CORRELATOS

A seguir são apresentados três trabalhos correlatos com relação ao trabalho proposto. O primeiro é uma aplicação de mineração de dados utilizando a ferramenta Weka (JUSTE, 2013). O segundo, MinerAll, é uma ferramenta para extração e mineração de dados de repositórios de software livre (GEROSA, 2011). O terceiro é uma aplicação dos algoritmos J48 e Apriori em uma base de dados para classificação de perfis de usuários conforme seus indicadores de saúde (LIBRELOTTO; MOZZAQUATRO, 2013).

2.5.1 Mineração de dados utilizando a ferramenta Weka

Juste (2013) relata a aplicação de mineração na base de dados do sistema REDECA. O objetivo é realizar a coleta de informações sobre os atendimentos de crianças e adolescentes, os quais são registrados pelo sistema, para auxiliar na tomada de decisões e planejamentos estratégicos. Na Figura 4 é apresentada a tela inicial do sistema REDECA.

Figura 4 - Sistema REDECA

Fonte: Juste (2013).

Juste (2013) utiliza o arquivo ARFF, gerado obtendo as informações da base de dados, para a aplicação do algoritmo J48 da ferramenta Weka, para realizar a classificação dos dados.

Neste trabalho, Juste (2013) utilizou o processo de KDD e durante a etapa de pré-processamento, onde foram selecionados e preparados os dados, constatou muitos dados com ruídos e incompletos, tornando esta fase muito trabalhosa e extensa. Diante desta grande quantidade de registros incompletos, foi necessário então a geração de dados aleatórios para a realização do trabalho. Juste (2013) ressalta a necessidade de conscientização da importância de realizar o cadastro completo e manter atualizados os registros, para que desta forma seja possível apresentar os resultados esperados.

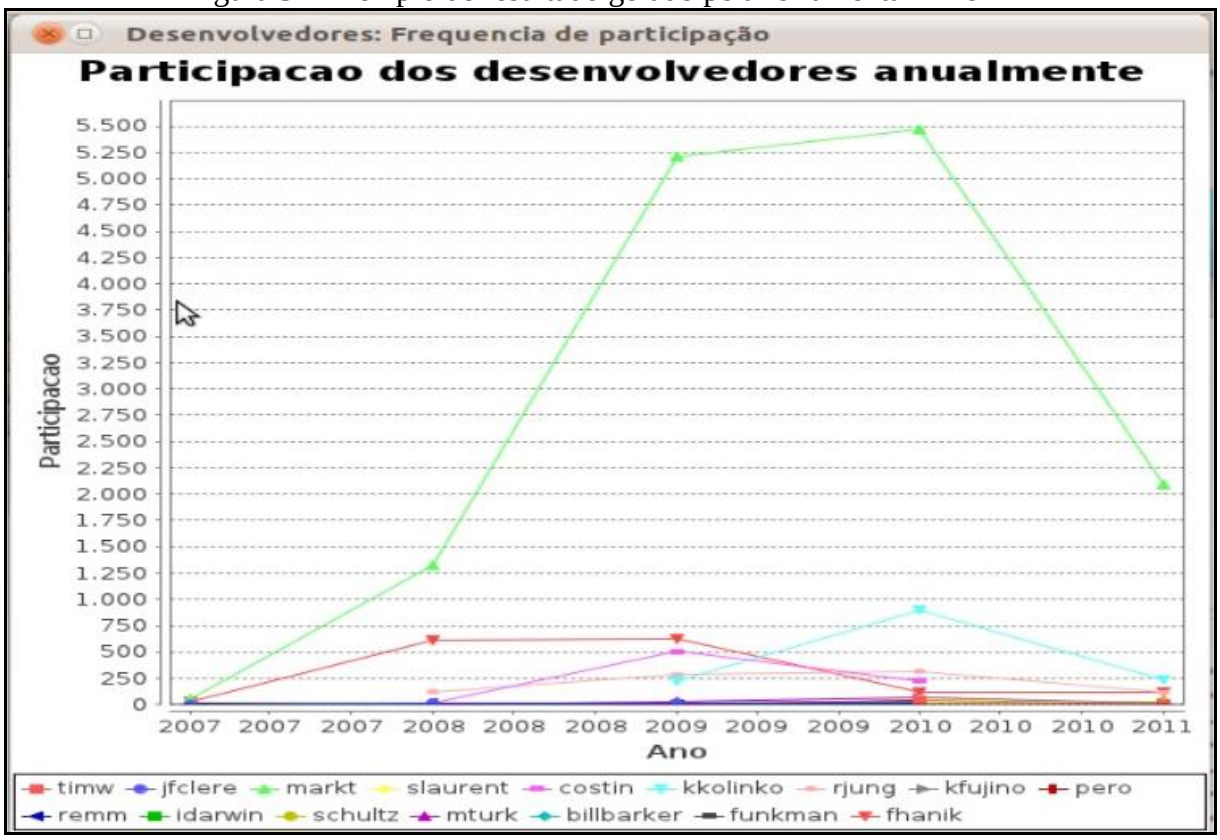
2.5.2 MINERALL

Gerosa (2011) relata o desenvolvimento de uma ferramenta denominada MinerAll, para mineração de dados em arquivos de códigos fonte em repositórios de software livre. Com esta ferramenta o autor pretende realizar a coleta de informações geradas por históricos de alterações de arquivos em repositórios de versionamento de códigos fonte e artefatos. Sendo assim, é gerada uma análise de dependência de mudanças e coletadas informações para geração de gráficos, os quais demonstram por exemplo:

a) quais usuários são mais ativos em determinado projeto, conforme apresentado na Figura 5, onde apresenta por ano os usuários e seu número de alterações realizadas;

b) grau de dependência entre os artefatos.

Figura 5 - Exemplo de resultado gerado pela ferramenta MinerAll



Fonte: Steinmacher (2012).

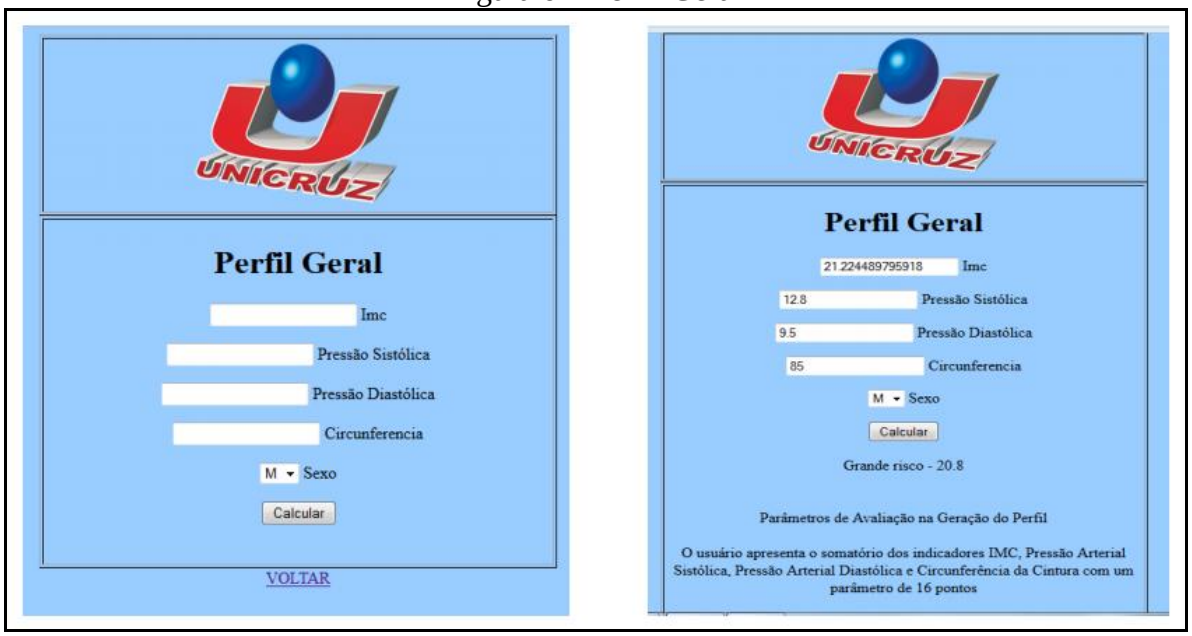
Segundo Gerosa (2011), os processadores de dados criados nesta ferramenta, adquirem os dados pelos objetos de leitura e realizam a mineração sobre estes. Após o processamento dos dados, obtêm-se a saída através de matrizes numéricas de relacionamento. Desta forma, são obtidos os dados necessários para a geração das estatísticas e dos gráficos para apresentação dos resultados da mineração de dados (GEROSA, 2011).

Gerosa (2011, p. 6) afirma sobre a ferramenta desenvolvida que "ela fornece informações importantes para tomada de decisões dos líderes de projetos, como a participação dos membros por exemplo.".

2.5.3 Análise dos algoritmos de mineração J48 e Apriori

Librelotto e Mozzaquatro (2013) descrevem a implementação de um software para cadastro de usuários. Neste cadastro é configurado o perfil de cada usuário, inserindo informações de indicadores de saúde: IMC, pressão sistólica, pressão diastólica, circunferência e sexo, conforme apresentado na Figura 6.

Figura 6 - Perfil Geral

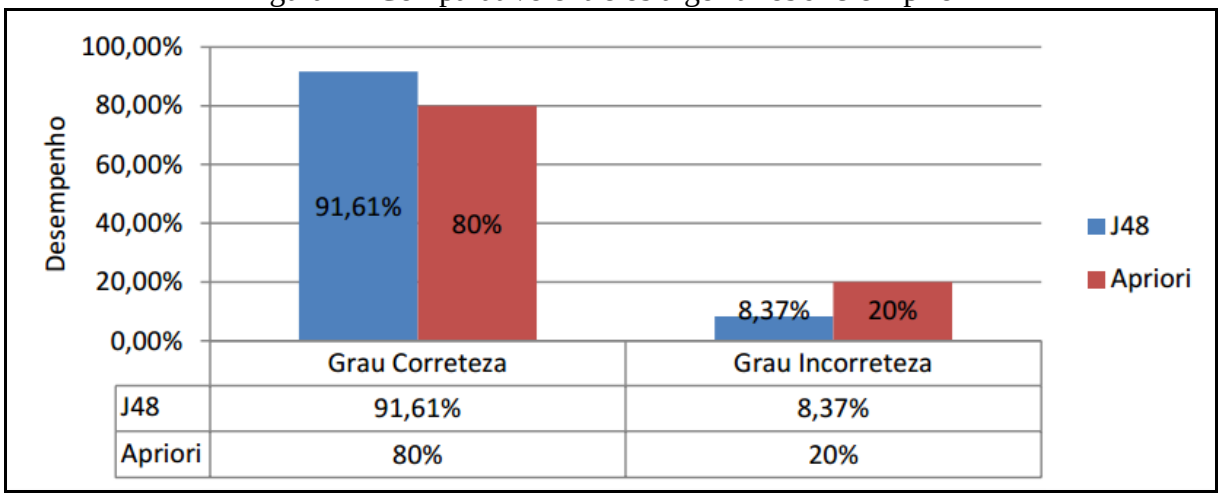


Fonte: Librelotto e Mozzaquatro (2013).

Com as informações dos indicadores de saúde, é realizada a classificação do item baseando-se na relação com os dados da base de conhecimento. Essa classificação é realizada utilizando a mineração de dados com os algoritmos J48 e Apriori.

O ponto principal do trabalho de Librelotto e Mozzaquatro (2013) é a comparação realizada entre os dois algoritmos, apresentando os resultados obtidos e mensurando o grau de assertividade de cada algoritmo. O resultado é apresentado na Figura 7. Librelotto e Mozzaquatro utilizam a palavra *correteza*, neste trabalho fora referenciado como assertividade.

Figura 7 - Comparativo entre os algoritmos J48 e Apriori



Fonte: Librelotto e Mozzaquatro (2013).

Segundo Librelotto e Mozzaquatro (2013), constatou-se que algoritmo J48 apresentou o grau de assertividade 91,61% enquanto o Apriori apresentou 80% de assertividade. Ocorreu

um diferencial de 11,61 pontos percentuais, apontando o algoritmo J48 como o mais eficaz para o objetivo proposto.

3 DESENVOLVIMENTO DO SOFTWARE

A seguir são apresentados os Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF), as etapas e a metodologia de desenvolvimento.

3.1 REQUISITOS

No Quadro 5 são descritos os Requisitos Funcionais (RF).

Quadro 5 - Requisitos Funcionais

Requisito Funcional (RF)
RF01: permitir ao usuário acessar a página de introdução do software
RF02: permitir ao usuário gerar a árvore de decisão
RF03: realizar a carga do arquivo com dados classificados para mineração
RF04: realizar a aplicação do algoritmo de mineração de dados J48
RF05: realizar a apresentação da árvore de decisão gerada pelo algoritmo

Fonte: elaborado pelo autor.

No Quadro 6 são descritos os Requisitos Não Funcionais (RNF).

Quadro 6 - Requisitos Não Funcionais

Requisito Não Funcional (RNF)
RNF01: as telas devem ser implementadas na linguagem de programação JavaScript
RNF02: *webservice* ser implementado utilizando a linguagem de programação Java
RNF03: utilizar o algoritmo J48 da biblioteca Weka

Fonte: elaborado pelo autor.

3.2 ESPECIFICAÇÃO

Nesta seção são apresentadas as ferramentas utilizadas e os diagramas de Atividade, Classes e *Deployment*, elaborados com a ferramenta Draw.io seguindo a especificação Unified Modeling Language (UML).

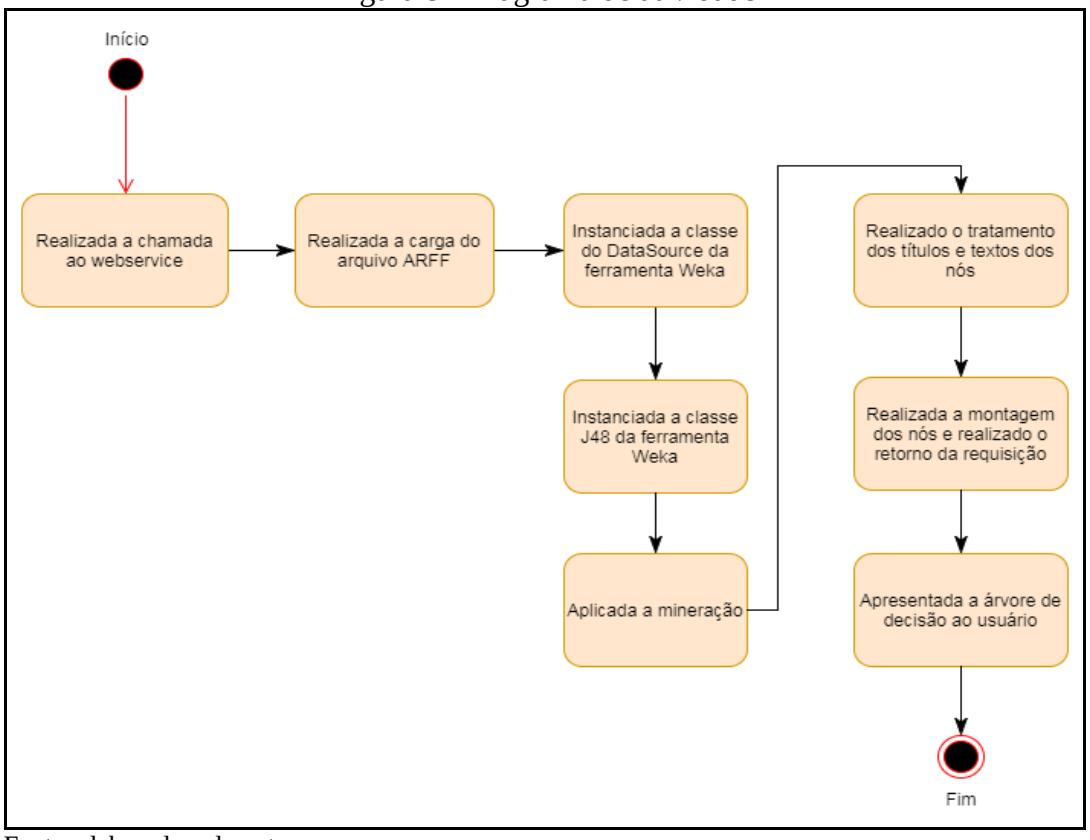
3.2.1 Diagrama de Atividade

Para melhor entendimento do fluxo do software, é apresentado a seguir o diagrama de atividade.

3.2.1.1 Fluxo do software

Na Figura 8 é apresentado o fluxo de atividade do software.

Figura 8 - Diagrama de atividade



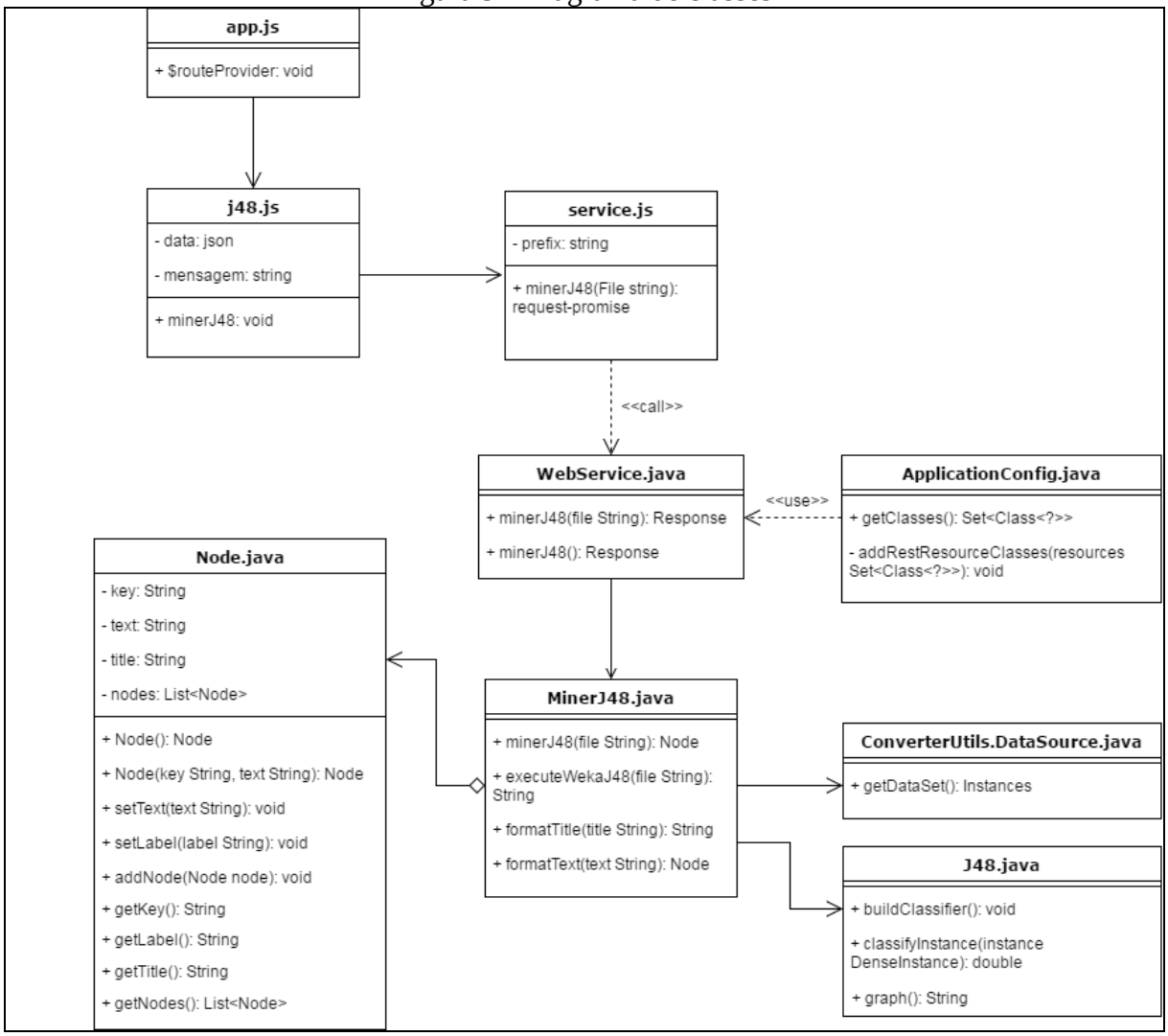
Fonte: elaborado pelo autor.

Ao acessar a página do software é apresentada ao usuário a página principal, `Home`, com informações do projeto de trabalho de conclusão de curso e os menus de acesso com opções de `Home` e `J48`. Ao selecionar o menu `J48`, é apresentada a página correspondente, onde é possível selecionar a opção `Minerar`. Esta opção executa a chamada ao webservice que realiza a carga do arquivo ARFF, instancia o `DataSouce` da ferramenta Weka com os dados do arquivo ARFF. Com o `DataSource` é instanciada a classe `J48` da ferramenta Weka. Realizada a mineração de dados e com o resultado gerado pelo algoritmo é realizado o tratamento dos textos e títulos dos nós e então é realizada a construção do retorno do método, em formato JSON. É retornada a requisição e então apresentado na página `J48` os resultados obtidos em forma de árvore de decisão.

3.2.2 Diagrama de Classes

Nesta seção são apresentadas as classes utilizadas no software, conforme demonstrado na Figura 9.

Figura 9 - Diagrama de classes



Fonte: elaborado pelo autor.

A classe `app.js` é responsável por criar a estrutura de telas e definir qual deve ser apresentada ao usuário conforme a opção selecionada. Ela realiza a inicialização da classe `j48.js`, que é responsável por apresentar ao usuário os dados obtidos do servidor.

A classe `service.js` é responsável por obter a chamada da classe `j48.js` e realizar a requisição ao *webservice*, obter a resposta e retornar os dados.

A classe `ApplicationConfig.java` realiza a inicialização do *webservice* e disponibiliza os métodos que podem ser acessados via requisição.

A classe `WebService.java` contém os métodos que podem ser acessados via requisição. Ao receber a requisição, direciona para a classe `MinerJ48.java` e retorna os dados obtidos.

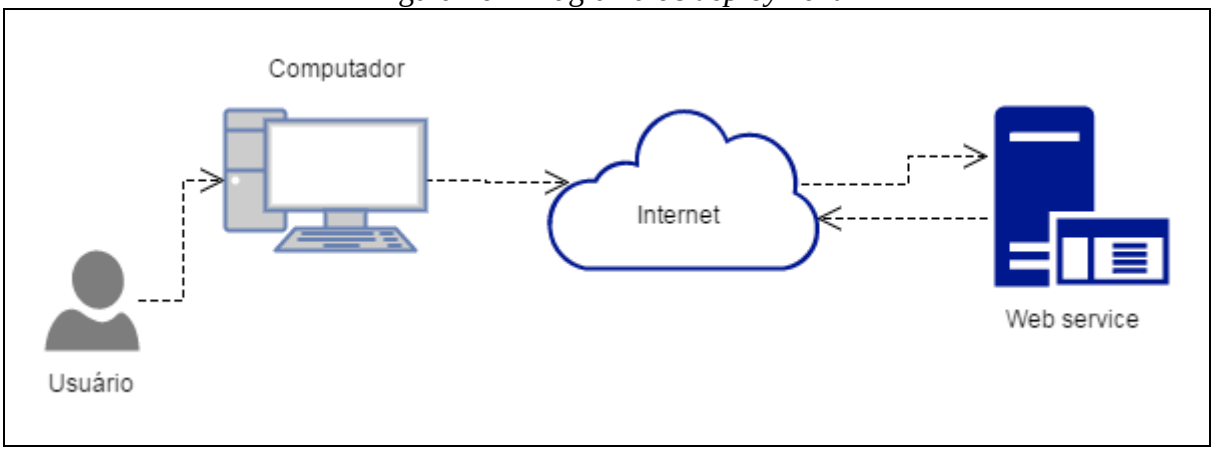
A classe `MinerJ48.java` é responsável por carregar o arquivo de dados, realizar a aplicação do algoritmo de mineração de dados, formatar os dados de saída através da classe `Node.java` e retornar ao *webservice*. Ela utiliza para aplicação dos algoritmos de mineração

de dados as classes `ConverterUtils.DataSource.java`, responsável por montar o data source com os dados do arquivo carregado; e `J48.java`, responsável por aplicar o algoritmo sobre os dados do data source, ambas as classes pertencentes à biblioteca Weka.

3.2.3 Diagrama de *Deployment*

Na Figura 10 é apresentado o diagrama de *deployment* do software e em seguida é descrito o processo realizado para utilização do software.

Figura 10 - Diagrama de *deployment*



Fonte: elaborado pelo autor.

O software é desenvolvido em dois projetos, *front-end* e *back-end*. O *front-end* é a parte visual, onde o usuário acessa através de um browser no computador e visualiza as páginas do software. Ao realizar a ação de minerar no *front-end*, é realizada uma chamada para o *webservice* através da internet. No *webservice* é realizado o processamento e então realiza o retorno ao *front-end*, que por sua vez apresenta as informações ao usuário. Para que o software seja utilizado corretamente, é necessário que ambos os projetos estejam iniciados e o acesso ao *webservice* pela internet esteja disponível.

3.3 IMPLEMENTAÇÃO

A seguir são apresentadas as técnicas e ferramentas utilizadas e a operacionalidade da implementação. Nos itens 3.3.1.1 e 3.3.1.2 é aplicada a etapa de seleção de dados, no item 3.3.1.3 é realizada a etapa de pré-processamento, no item 3.3.1.3.1 é realizada a etapa de transformação e no item 3.3.1.5 é realizada a etapa de mineração de dados do processo de KDD.

3.3.1 Técnicas e ferramentas utilizadas

Para o desenvolvimento do software foram utilizadas as linguagens de programação Java na versão 7 e JavaScript. O ambiente de desenvolvimento foi a NetBeans IDE versão 8.0, disponibilizada pela Oracle Corporation, e Visual Studio Code na versão 1.12.2, disponibilizada pela Microsoft. Fora utilizada a biblioteca open source Weka, desenvolvida pela Universidade de Waikato.

3.3.1.1 Coleta de dados de vendas de varejo

Na Tabela 1 são apresentados alguns registros que foram obtidos da base dados sem qualquer tipo de tratamento e classificação, o arquivo completo possui 1421 registros.

Tabela 1 - Exemplo de dados obtidos

Filial	Data	Hora	Atendimentos	Fluxo	Fluxo vitrine
Shopping A	02/01/17	10	7	10	172
Shopping A	02/01/17	11	4	45	71
Shopping A	02/01/17	12	0	20	141
Shopping A	02/01/17	13	8	33	128
Shopping A	02/01/17	14	5	43	84
Shopping A	02/01/17	15	9	81	236
Shopping A	02/01/17	16	8	77	540
Shopping A	02/01/17	17	16	153	717
Shopping B	03/01/17	11	0	3	25
Shopping B	03/01/17	12	1	3	67
Shopping B	03/01/17	13	0	11	103
Shopping B	03/01/17	14	3	25	92
Shopping B	03/01/17	15	0	18	103
Shopping B	03/01/17	16	2	14	97
Shopping B	03/01/17	17	2	21	165
Shopping B	03/01/17	18	1	7	86

Fonte: elaborada pelo autor.

Conforme a Tabela 1, a coluna Filial corresponde à loja em que foram obtidos os valores do registro. A coluna Data corresponde à data em que fora realizado o registro na base de dados. A coluna Hora corresponde ao horário em que fora realizado o registro. A coluna Atendimentos corresponde ao número de vendas realizadas na data e horário. A coluna Fluxo corresponde à quantidade de clientes que entraram na loja no período indicado. A coluna Fluxo vitrine corresponde à quantidade de pessoas que passaram em frente a vitrine da loja.

3.3.1.2 Coleta de dados metereológicos

Com os dados coletados da base de dados de vendas de varejo, fora possível identificar qual a cidade em que o dado fora coletado e a data do registro. Com estas informações foram

coletados os dados históricos de metereologia para que seja possível validar se a temperatura e chuva influenciam no número de vendas.

Estes dados metereológicos foram obtidos do Banco de Dados Metereológicos para Ensino e Pesquisa (BDMEP) que é populado com dados do Instituto Nacional de Metereologia (INMET). Os dados do sistema BDMEP são gerados em um arquivo em formato Comma-Separated Values (CSV), conforme demonstrado no Quadro 7.

Quadro 7 - Exemplo de dados gerados pelo BDMEP

```
--------------------
BDMEP - INMET
--------------------
Estação           : FLORIANOPOLIS - SC (OMM: 83897)
Latitude  (graus) : -27.58
Longitude (graus) : -48.56
Altitude  (metros): 1.84
Estação Operante
Inicio de operação: 01/12/1921
Periodo solicitado dos dados: 01/01/2017 a 01/02/2017
Os dados listados abaixo são os que encontram-se digitados no BDMEP
Hora em UTC
--------------------
Obs.: Os dados aparecem separados por ; (ponto e vírgula) no formato txt.
      Para o formato planilha XLS, siga as instruções
--------------------
Estacao;Data;Hora;UmidadeRelativa;PressaoAtmEstacao;VelocidadeVentoNebulosidade;
83897;01/01/2017;0000;90;1008.8;1;10;
83897;01/01/2017;1200;78;1007.4;2;8;
83897;01/01/2017;1800;72;1007.5;4;8;
83897;02/01/2017;0000;92;1009.2;0;3;
83897;02/01/2017;1200;68;1009.5;0;4;
83897;02/01/2017;1800;73;1010.2;4;8;
83897;03/01/2017;0000;81;1011.5;4;8;
83897;03/01/2017;1200;80;1013.4;1;6;
83897;03/01/2017;1800;62;1009.9;2;7;
83897;04/01/2017;0000;80;1010;3;10;
83897;04/01/2017;1200;72;1009.2;1;7;
83897;04/01/2017;1800;70;1006.8;4;3;
```

Fonte: elaborado pelo autor.

3.3.1.3 Geração de dados para mineração

Para a mineração de dados, fora realizada a geração de um arquivo em formato ARFF. A estrutura de geração deste arquivo é apresentada no item 2.2.

Para gerar este arquivo foram utilizados os valores coletados da base de dados de vendas de varejo, conforme descrito na seção 3.3.1.1, e adicionados os dados metereológicos, descrito na seção 3.3.1.2.

Inicialmente fora gerado o arquivo conforme demonstrado no Quadro 8.

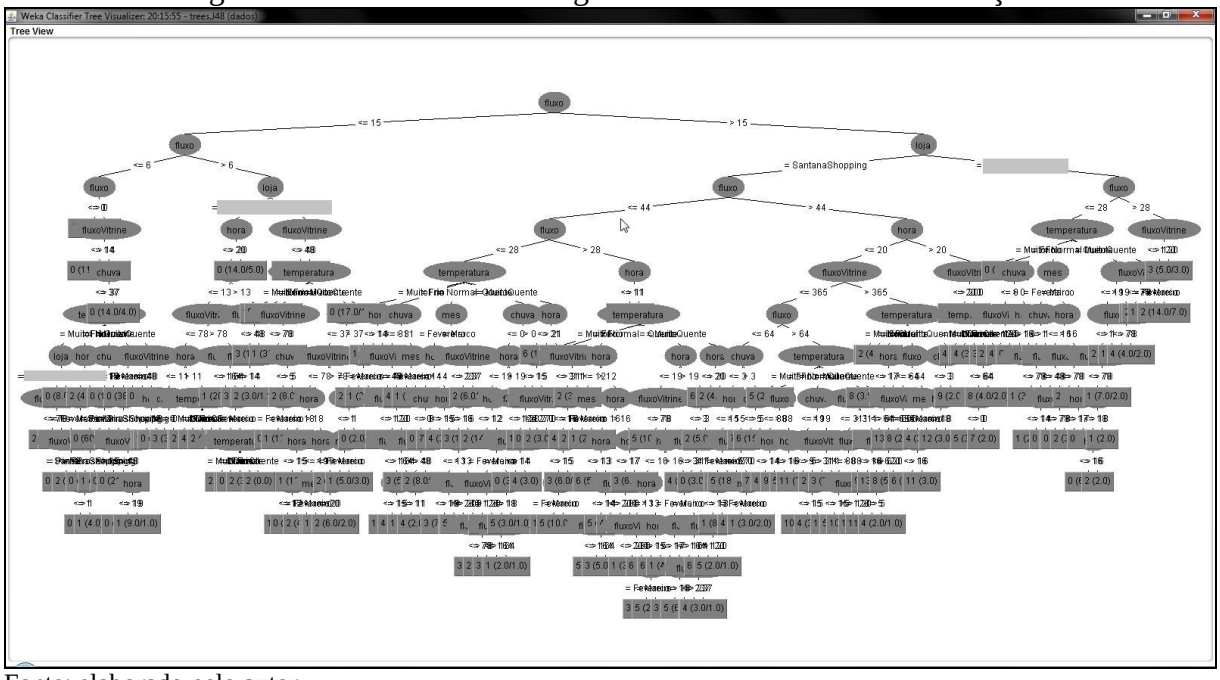
Quadro 8 - Arquivo ARFF sem classificação

```
@relation dados

@attribute loja {ShoppingA, ShoppingB}
@attribute diaSemana {segunda, terca, quarta, quinta, sexta, sabado, domingo}
@attribute diaMes real
@attribute mes {Fevereiro, Marco}
@attribute hora real
@attribute fluxo real
@attribute fluxoVitrine real
@attribute temperatura real
@attribute umidade real
@attribute nebulosidade real
@attribute chuva real
@attribute atendimento {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28}

@data
ShoppingA,quarta,1,Fevereiro,10,15,193,23.4,79,10,1.0,0
ShoppingA,quarta,1,Fevereiro,11,36,67,23.4,79,10,1.0,4
ShoppingA,quarta,1,Fevereiro,12,38,93,22.0,82,10,1.0,4
ShoppingA,quarta,1,Fevereiro,13,33,269,22.0,82,10,1.0,3
ShoppingA,quarta,1,Fevereiro,14,66,229,22.0,82,10,1.0,1
ShoppingA,quarta,1,Fevereiro,15,72,441,22.0,82,10,1.0,2
ShoppingA,quarta,1,Fevereiro,16,54,326,22.0,82,10,1.0,3
ShoppingA,quarta,1,Fevereiro,17,63,23,22.0,82,10,1.0,5
ShoppingA,quarta,1,Fevereiro,18,52,241,26.5,61,10,1.0,3
ShoppingA,quarta,1,Fevereiro,19,51,224,26.5,61,10,1.0,5
ShoppingA,quarta,1,Fevereiro,20,40,388,26.5,61,10,1.0,10
ShoppingA,quarta,1,Fevereiro,21,157,126,26.5,61,10,1.0,7
ShoppingA,quarta,1,Fevereiro,22,69,165,26.5,61,10,1.0,0
```

Fonte: elaborado pelo autor.

Com os dados gerados e sem realizar classificação, fora executado o algoritmo J48 e solicitada a visualização da árvore de decisão pela ferramenta Weka. O resultado foi uma grande quantidade de possibilidades, de difícil identificação, conforme Figura 11.

Figura 11 - Árvore de decisão gerada com dados sem classificação



Fonte: elaborado pelo autor.

Conforme demonstrado na Figura 11, a árvore de decisão gerada é muito complexa, de difícil visualização e identificação das opções. No Apêndice B é demonstrada parte da árvore gerada pela ferramenta Weka com a opção de zoom, para que as informações sejam

visualizadas. Isso se deve ao fato dos dados estarem sendo inseridos como valores numéricos, desta forma apresentando uma opção específica para cada valor. É necessário realizar a classificação dos valores para que seja possível obter informações de forma agrupada, para uma melhor compreensão. Nas seções seguintes são demonstradas as classificações dos dados de temperatura, chuva, hora, fluxo, fluxo vitrine e atendimento.

3.3.1.3.1 Classificação dos dados

Os dados de temperatura do período de 01/01/2017 à 28/02/2017 estavam entre o intervalo de 18ºC e 34ºC. Estes dados foram divididos em quatro categorias, conforme demonstrado no Quadro 9.

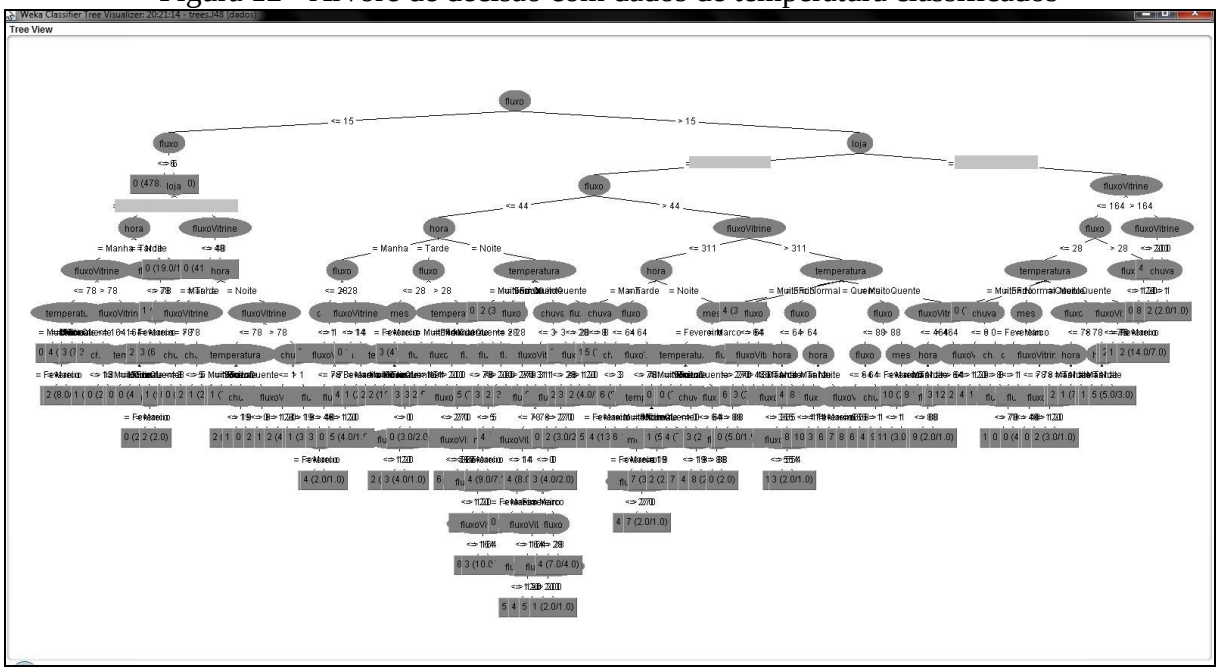
Quadro 9 - Classificação do atributo temperatura

Temperatura	Classificação
18ºC – 21ºC	Frio
21,1ºC – 24ºC	Morno
24,1ºC – 27ºC	Quente
27,1ºC – 34ºC	MuitoQuente

Fonte: elaborada pelo autor.

Na Figura 12 é demonstrado o resultado da árvore de decisão gerada pela ferramenta Weka no arquivo com somente o atributo de temperatura classificado.

Figura 12 - Árvore de decisão com dados de temperatura classificados



Fonte: elaborado pelo autor.

Os dados de chuva foram obtidos em milímetros por dia, variando entre 0mm e 46,6mm. Estes dados foram divididos em seis categorias conforme demonstrados no Quadro 10.

Quadro 10 - Classificação do atributo chuva

Chuva	Classificação
0mm	SemChuva
0,1mm – 1mm	PoucaChuva
1,0mm – 4,6mm	ChuvaLeve
4,7mm – 12,1mm	ChuvaModerada
12,2mm – 27mm	ChuvaForte
27,1mm – 46,6mm	Tempestade

Fonte: elaborada pelo autor.

Os dados de horas, foram classificados conforme demonstrado no Quadro 11.

Quadro 11 - Classificação do atributo hora

Hora	Classificação
09:00 – 12:00	Manha
13:00 – 18:00	Tarde
19:00 – 22:00	Noite

Fonte: elaborada pelo autor.

Os dados de fluxo, foram classificados conforme demonstrado no Quadro 12.

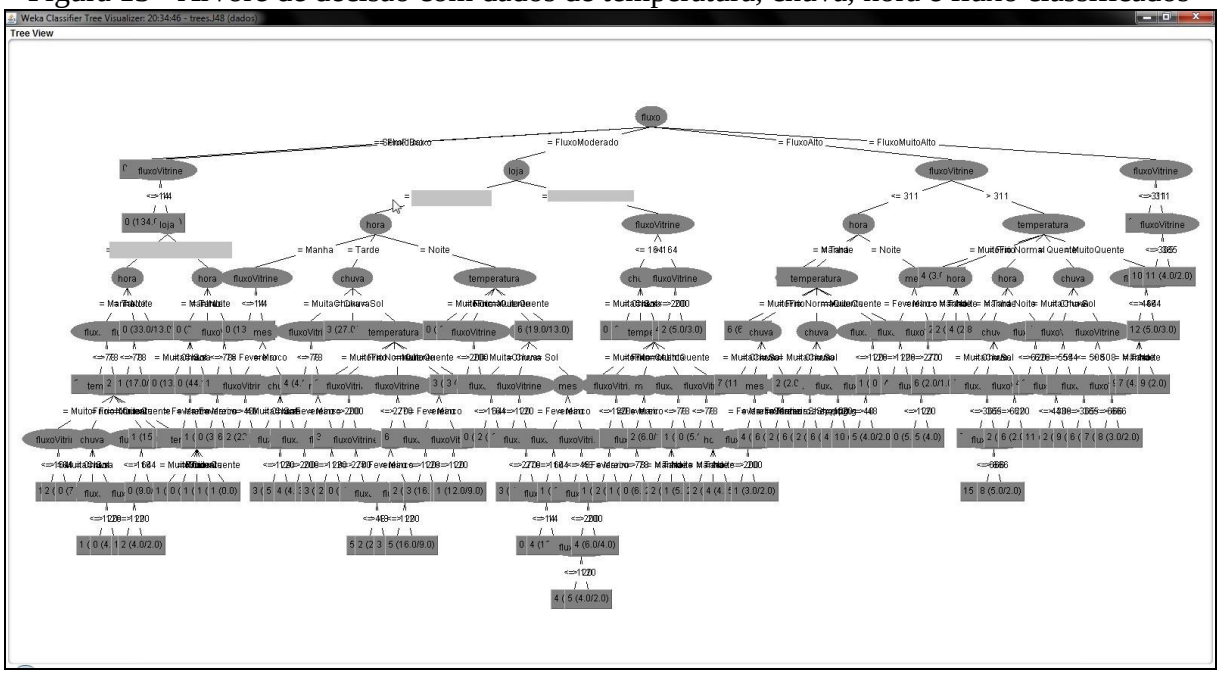
Quadro 12 - Classificação do atributo fluxo

Fluxo	Classificação
0	SemFluxo
1 a 50	FluxoBaixo
51 a 100	FluxoModerado
101 a 200	FluxoAlto

Fonte: elaborada pelo autor.

Com os dados de temperatura, chuva, hora e fluxo classificados, fora gerada novamente a árvore de decisão pela ferramenta Weka. O resultado é apresentado na Figura 13.

Figura 13 - Árvore de decisão com dados de temperatura, chuva, hora e fluxo classificados



Fonte: elaborado pelo autor.

Os dados do atributo fluxoVitrine que representam o volume de pessoas que passam na frente da vitrine são classificados conforme Quadro 13.

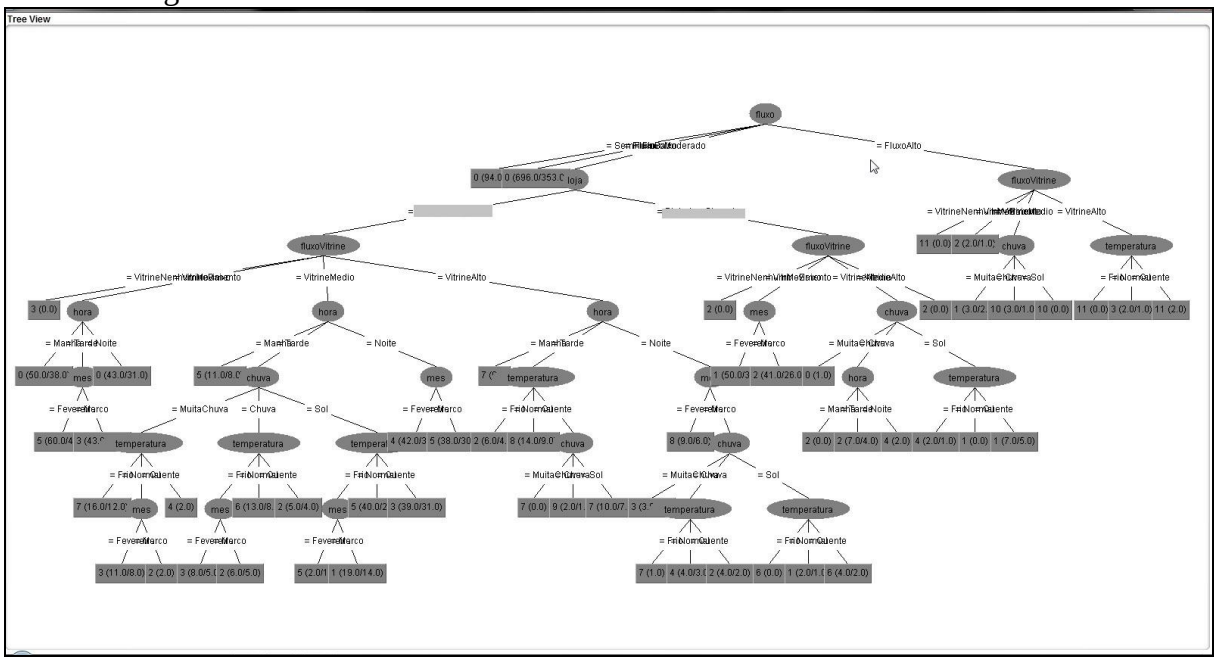
Quadro 13 - Classificação do atributo fluxoVitrine

Fluxo vitrine	Classificação
0	VitrineNenhumMovimento
1 a 90	VitrineBaixo
91 a 300	VitrineMedio
300 a 743	VitrineAlto

Fonte: elaborada pelo autor.

Com a classificação do atributo fluxoVitrine, o resultado da geração da árvore de decisão se transforma em um fluxo um pouco mais simples e legível que os anteriores, conforme demonstrado na Figura 14.

Figura 14 - Árvore de decisão com dados de fluxo de vitrine classificados



Fonte: elaborado pelo autor.

Os dados do atributo atendimento foram classificados conforme Quadro 14.

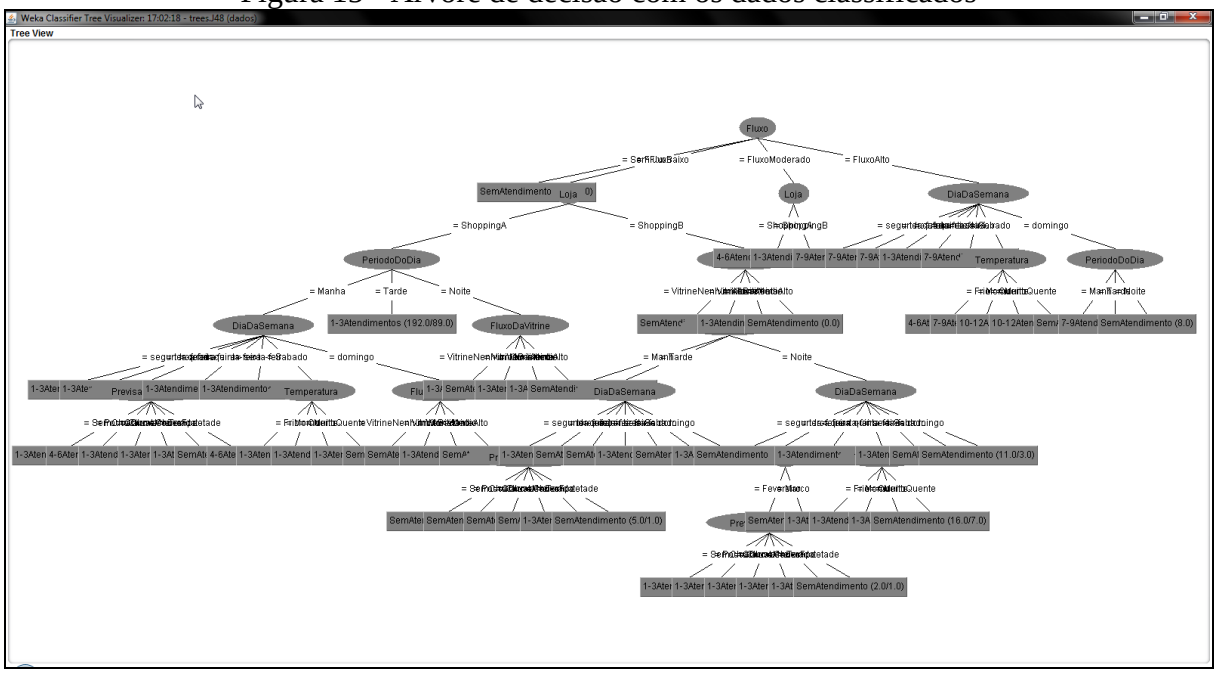
Quadro 14 - Classificação do atributo atendimento

Quantidade de atendimentos	Classificação
0	SemAtendimento
1 a 3	1-3Atendimentos
4 a 6	4-6Atendimentos
7 a 9	7-9Atendimentos
10 a 12	10-12Atendimentos
13 a 15	13-15Atendimentos
19	19Atendimentos

Fonte: elaborada pelo autor.

Desta forma, todos os dados numéricos foram classificados, permitindo a geração da árvore de decisão pela ferramenta Weka conforme apresentado na Figura 15.

Figura 15 - Árvore de decisão com os dados classificados



Fonte: elaborado pelo autor.

Ainda com os dados classificados, e a quantidade de nós e ramos sejam menores a leitura dos dados da árvore de decisão demonstrada no formato horizontal, torna-se difícil. Por este motivo, neste trabalho foram apresentados os dados em formato vertical, para que a leitura seja simplificada, conforme demonstrado no item 3.3.1.5.

O Quadro 15 demonstra uma parte do arquivo ARFF com os dados classificados. No Apêndice A é apresentada uma parte maior do arquivo.

Quadro 15 - Arquivo ARFF com dados classificados

```
@relation dados

@attribute Loja {ShoppingA, ShoppingB}
@attribute DiaDaSemana {segunda-feira, terca-feira, quarta-feira, quinta-feira, sexta-feira, Sabado, domingo}
@attribute Mes {Fevereiro, Marco}
@attribute PeriodoDoDia {Manha, Tarde, Noite}
@attribute Fluxo {SemFluxo, FluxoBaixo, FluxoModerado, FluxoAlto}
@attribute FluxoDaVitrine {VitrineNenhumMovimento, VitrineBaixo, VitrineMedio, VitrineAlto}
@attribute Temperatura {Frio, Morno, Quente, MuitoQuente}
@attribute PrevisaoDoTempo {SemChuva, PoucaChuva, ChuvaLeve, ChuvaModerada, ChuvaForte, Tempestade}
@attribute Atendimentos {SemAtendimento, 1-3Atendimentos, 4-6Atendimentos, 7-9Atendimentos, 10-12Atendimentos, 13-15Atendimentos, 19Atendimentos}

@data
ShoppingA,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,SemAtendimento
ShoppingA,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,4-6Atendimentos
ShoppingA,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,4-6Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineBaixo,Morno,PoucaChuva,4-6Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Quente,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Noite,FluxoModerado,VitrineMedio,Quente,PoucaChuva,4-6Atendimentos
ShoppingA,quarta-feira,Fevereiro,Noite,FluxoBaixo,VitrineAlto,Quente,PoucaChuva,10-12Atendimentos
ShoppingA,quarta-feira,Fevereiro,Noite,FluxoAlto,VitrineMedio,Quente,PoucaChuva,7-9Atendimentos
ShoppingA,quarta-feira,Fevereiro,Noite,FluxoModerado,VitrineMedio,Quente,PoucaChuva,SemAtendimento
ShoppingA,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Frio,ChuvaForte,SemAtendimento
ShoppingA,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Frio,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoAlto,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Morno,ChuvaForte,7-9Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Morno,ChuvaForte,4-6Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoModerado,VitrineAlto,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoAlto,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoAlto,VitrineAlto,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoAlto,VitrineAlto,Morno,ChuvaForte,4-6Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,Morno,ChuvaForte,SemAtendimento
ShoppingA,sexta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Quente,PoucaChuva,SemAtendimento
```

Fonte: elaborado pelo autor.

3.3.1.4 Desenvolvimento do *webservice*

Para o desenvolvimento do *webservice*, fora utilizado o estilo arquitetural Representational State Transfer (REST) utilizando a notação JavaScript Object Notation (JSON) para transferência dos dados. Para a implementação deste *webservice* é necessário definir quais métodos podem ser acessados via requisição Hypertext Transfer Protocol (HTTP), realizado na classe `WebService.java`, conforme demonstrado no Quadro 16.

Quadro 16 - Definição dos métodos do *webservice*

```
21  @Path("service")
22  public class WebService {
23
24      @GET
25      @Path("/minerJ48/{file}")
26      @Produces(MediaType.APPLICATION_JSON)
27      public Response minerJ48(@PathParam("file") String file) {
28          Node node = new MinerJ48().minerJ48(file);
29          return Response.status(200)
30                  .header("Access-Control-Allow-Origin", "*")
31                  .header("Access-Control-Allow-Headers", "origin, content-type, accept, authorization")
32                  .header("Access-Control-Allow-Credentials", "true")
33                  .header("Access-Control-Allow-Methods", "GET, POST, PUT, DELETE, OPTIONS, HEAD")
34                  .header("Access-Control-Max-Age", "1209600")
35                  .entity(node).build();
36      }
```

Fonte: elaborado pelo autor.

Conforme o Quadro 16, na linha 21, é definido através de anotação o identificador `service` para acesso via requisição HTTP. Na linha 25 é definido o caminho restante para a requisição ao método `minerJ48` e o parâmetro que é esperado para esta chamada, neste caso `file`. A definição do tipo de notação em que o objeto de retorno se refere é definida na linha 26, neste caso a notação JSON.

Após a definição da classe e método que pode ser acessado, é necessário realizar a disponibilização para que seja possível realizar a requisição HTTP quando o serviço for iniciado. Para tal, é necessária a implementação da classe `ApplicationConfig.java`, apresentada no Quadro 17. Esta classe realiza a varredura em todo o projeto, identificando as classes e métodos que devem ser habilitados a acessar via requisição HTTP e constrói o *webservice* com estes dados.

Quadro 17 - Construção do *webservice*

```
16  @javax.ws.rs.ApplicationPath("webresources")
17  public class ApplicationConfig extends Application {
18
19      @Override
20      public Set<Class<?>> getClasses() {
21          Set<Class<?>> resources = new java.util.HashSet<Class<?>>();
22          addRestResourceClasses(resources);
23          return resources;
24      }
25
26      /**
27       * Do not modify addRestResourceClasses() method.
28       * It is automatically populated with
29       * all resources defined in the project.
30       * If required, comment out calling this method in getClasses().
31       */
32      private void addRestResourceClasses(Set<Class<?>> resources) {
33          resources.add(com.miner.minerws.WebService.class);
34      }
35
36  }
```

Fonte: elaborado pelo autor.

3.3.1.5 Mineração dos dados

Ao receber uma requisição, o *webservice* direciona a chamada para a classe `MinerJ48.java`, que por sua vez realiza a chamada para o método `executeWekaJ48`, que é demonstrado no Quadro 18.

Esta classe realiza a carga do arquivo ARFF, que fora gerado com os dados classificados, conforme demonstrado no item 3.3.1.3 Geração de dados para mineração. Após a carga do arquivo, é realizada a criação do `ConverterUtils.DataSource.java`, conforme a linha 118 do Quadro 18. Na linha 119 é possível obter um objeto da classe `Instance.java`, onde é necessário identificar o atributo resultado do arquivo, no caso o atributo atendimento.

A partir do data source é possível iniciar a classe `J48.java`, da biblioteca da ferramenta Weka, a qual é responsável por realizar a obtenção dos dados e transformá-los em uma base de conhecimento. Na linha 126 é criado um registro do tipo Instance.java, com 8 atributos, e populados com informações genéricas, exceto com o valor do atributo atendimento. É realizado então, na linha 138 a execução da classificação do arquivo com o registro adicional através do método `classifyInstance` e ao executar este método é gerada a árvore de decisão, que é obtida na linha 140.

Quadro 18 - Método para geração de árvore de decisão

```
116     public static String executeWekaJ48(String file) {
117         try {
118             ConverterUtils.DataSource ds = new ConverterUtils.DataSource(new FileInputStream(file));
119             Instances ins = ds.getDataSet();
120             ins.setClassIndex(8);
121 
122             J48 j48 = new J48();
123             //cria o classificador
124             j48.buildClassifier(ins);
125 
126             Instance novo = new DenseInstance(8);
127             //vincula o arquivo e os dados
128             novo.setDataset(ins);
129             novo.setValue(0, "ShoppingA");
130             novo.setValue(1, "quarta-feira");
131             novo.setValue(2, "Fevereiro");
132             novo.setValue(3, "Tarde");
133             novo.setValue(4, "FluxoModerado");
134             novo.setValue(5, "VitrineMedio");
135             novo.setValue(6, "Morno");
136             novo.setValue(7, "PoucaChuva");
137 
138             double retorno = j48.classifyInstance(novo);
139             System.out.println("retorno: " + retorno);
140             String graph = ((Drawable) j48).graph();
141             System.out.println("#########\n" + j48.numElements() + "\n" + j48.measureTreeSize() + "\n" + graph);
142 
143             return graph;
144         } catch (Exception e) {
                e.printStackTrace();
146         }
147         return "Can't process the file arff";
148     }
```

Fonte: elaborado pelo autor.

A árvore de decisão obtida é em formato `String`, porém é necessário formatá-la para poder apresentar ao usuário. Desta forma é necessário realizar a reestruturação conforme o Quadro 19.

Quadro 19 - Reestruturação do retorno

```
37          for (String item : list) {
38              if (!item.contains("{") && !item.contains("}")) {
39                  if (!item.contains("->")) {
40                      String[] nodeString = item.split("\\[");
41                      String key = nodeString[0].trim();
42                      String text = nodeString[1];
43                      String title = text.substring(text.indexOf("\"") + 1, text.lastIndexOf("\""));
44
45                      if (nodes.containsKey(key)) {
46                          nodes.get(key).setText(formatText(title));
47                      } else {
48                          Node node = new Node(key, formatText(title));
49                          nodes.put(node.getKey(), node);
50
51                          if (root == null) {
52                              root = node;
53                          }
54                      }
55                  } else {
56                      String[] nodeString = item.split("->");
57                      Node nodeParent = nodes.get(nodeString[0].trim());
58
59                      String[] restString = nodeString[1].split("\\[");
60                      String key = restString[0].trim();
61                      String text = restString[1];
62                      String title = text.substring(text.indexOf("\"") + 1, text.lastIndexOf("\""));
63
64                      Node node;
65                      if (nodes.containsKey(key)) {
66                          node = nodes.get(key);
67                      } else {
68                          node = new Node(key, formatTitle(title));
69                      }
```

Fonte: elaborado pelo autor.

Os dados obtidos da árvore de decisão são formatados da seguinte maneira:

a) o nome de cada item é definido entre colchetes;

b) a descrição de cada item é definida entre aspas;

c) as ligações entre um item e o seguinte, são definidos por `->`;

d) a descrição de cada ligação é definida entre aspas.

Desta forma, foi recriada a estrutura em forma de árvore a partir da classe `Node.java`, que é encarregada de armazenar o título de cada item, a descrição das ligações entre seus itens adjacentes e também o armazenamento dos itens inferiores. Para cada título e descrição é necessário o tratamento para formatar em um texto que seja compreensível para o usuário, conforme o Quadro 20. Com o resultado deste método, é retornado o nó raiz da árvore pelo *webservice*.

Quadro 20 - Método para formatar o título dos itens

```
150     public static String formatTitle(String text) {
151         String textChanged;
152         if ("= SemFluxo".equals(text)) {
153             textChanged = "= Sem fluxo (0 clientes)";
154         } else if ("= FluxoBaixo".equals(text)) {
155             textChanged = "= Fluxo baixo (1 a 50 clientes)";
156         } else if ("= FluxoModerado".equals(text)) {
157             textChanged = "= Fluxo moderado (51 a 100 clientes)";
158         } else if ("= FluxoAlto".equals(text)) {
159             textChanged = "= Fluxo alto (101 a 200 clientes)";
160         } else if ("= VitrineNenhumMovimento".equals(text)) {
161             textChanged = "= Sem movimento (0 visitantes)";
162         } else if ("= VitrineBaixo".equals(text)) {
163             textChanged = "= Movimento baixo (1 a 90 visitantes)";
164         } else if ("= VitrineMedio".equals(text)) {
165             textChanged = "= Movimento médio (91 a 300 visitantes)";
166         } else if ("= VitrineAlto".equals(text)) {
167             textChanged = "= Movimento alto (301 a 743 visitantes)";
168         } else if ("= Manha".equals(text)) {
169             textChanged = "= Matutino (10h - 12h)";
170         } else if ("= Tarde".equals(text)) {
171             textChanged = "= Vespertino (13h-18h)";
172         } else if ("= Noite".equals(text)) {
173             textChanged = "= Noturno (19h-22h)";
174         } else if ("= Frio".equals(text)) {
175             textChanged = "= Frio (18°C a 21°C)";
176         } else if ("= Morno".equals(text)) {
177             textChanged = "= Morno (21,1°C a 24°C)";
178         } else if ("= Quente".equals(text)) {
179             textChanged = "= Quente (24,1°C a 27°C)";
180         } else if ("= MuitoQuente".equals(text)) {
181             textChanged = "= Muito quente (27,1°C a 34°C)";
182         } else if ("= ShoppingA".equals(text)) {
183             textChanged = "= Shopping A";
```

Fonte: elaborado pelo autor.

3.3.2 Operacionalidade da implementação

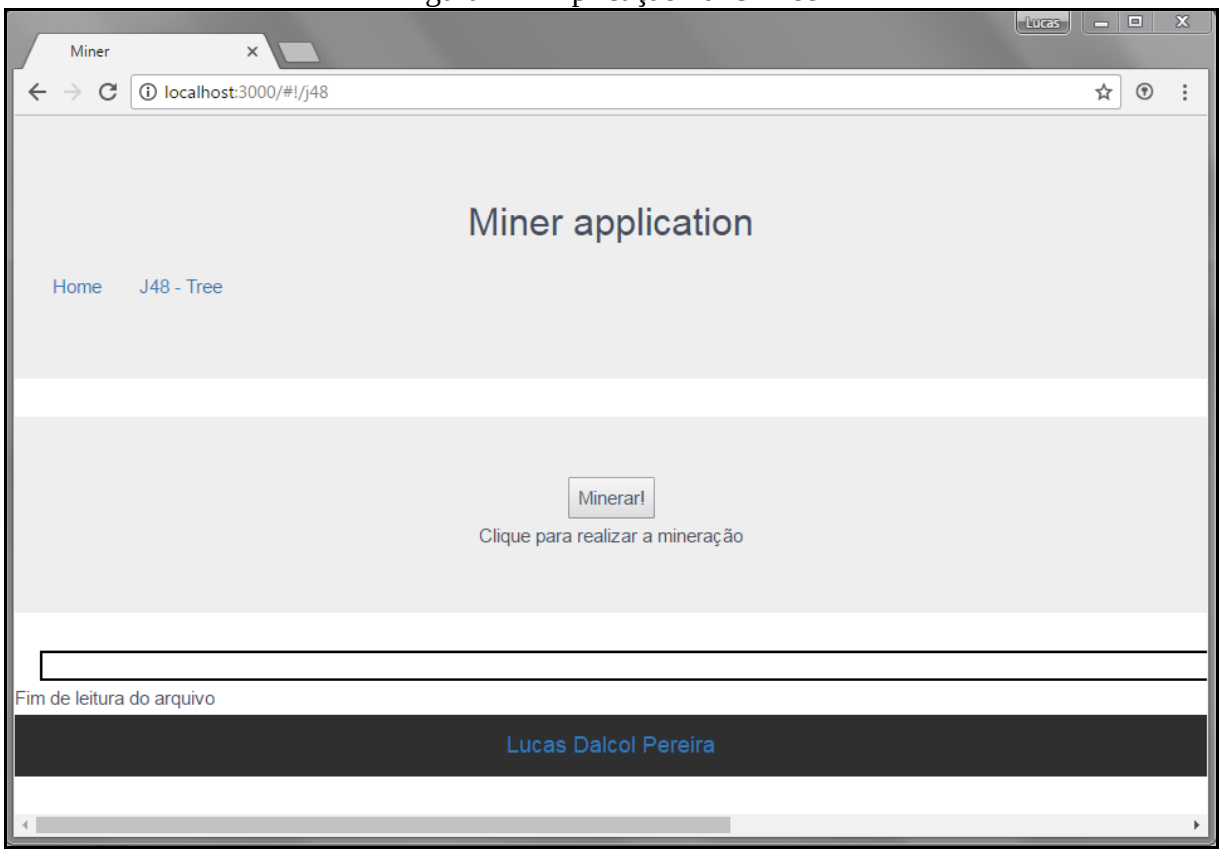
O software apresenta ao usuário uma tela inicial, conforme demonstrado na Figura 16. Onde é possível visualizar duas opções de menu, `Home` e `J48-Tree`. A Figura 16 representa a opção `Home`.

Figura 16 - Aplicação - Home

Fonte: elaborado pelo autor.

A opção J48-Tree apresenta o botão Minerar, para que o usuário inicie a mineração. Conforme demonstrado na Figura 17.

Figura 17 - Aplicação - J48-Tree



Fonte: elaborado pelo autor.

Ao clicar no botão Minerar, é realizada a requisição ao *webservice*, que por sua vez executa o algoritmo de mineração e retorna a árvore gerada. Com os dados da árvore, é demonstrada ao usuário, conforme Figura 18.

Figura 18 - Aplicação - árvore de decisão

Fonte: elaborado pelo autor.

A árvore de decisão completa é apresentada no Apêndice C.

O número de atendimentos possíveis é o resultado esperado, baseando-se nas variáveis como: quantidade de fluxo de vitrine, quantidade de fluxo na loja, temperatura, dia da semana e período. Conforme demonstrado na Figura 18, há um percentual de assertividade para cada resultado de atendimento esperado. O código para a geração deste percentual é apresentado a seguir no Quadro 21.

Quadro 21 - Tratamento do percentual de assertividade

```
} else if (title.contains("Atendimento")) {
    if (title.contains("/")) {
        String[] valores = title.substring(title.indexOf("(") + 1, title.indexOf(")")).split("/");
        double vlPositivo = Double.valueOf(valores[0]);
        double vlNegativo = Double.valueOf(valores[1]);

        double total = vlPositivo + vlNegativo;
        Double assertividade = (vlPositivo*100)/total;
        DecimalFormat decimalFormat = new DecimalFormat("#.00");
        String assertividadeString = decimalFormat.format(assertividade) + "%";

        if (title.startsWith("SemAtendimento")) {

            decimalFormat = new DecimalFormat("#.0");

            titleChanged = String.format("Estimativa de atendimento: sem atendimento, %s de assertividade baseado em "
                    + "%s registros na base de conhecimento.",
                    assertividadeString, decimalFormat.format(total));
        } else {
            String vlInferior = title.substring(0, title.indexOf("-"));
            String vlSuperior = title.substring(title.indexOf("-")+1, title.indexOf("Atend"));

            titleChanged = String.format("Estimativa de atendimento: de %s à %s, %s de assertividade baseado em "
                    + "%s registros na base de conhecimento.",
                    vlInferior, vlSuperior, assertividadeString, decimalFormat.format((vlPositivo + vlNegativo)));
        }
```

Fonte: elaborado pelo autor.

Os valores que são retornados pela geração do algoritmo são adicionados entre parênteses no título do nó. São retornados dois valores separados hífen, sendo o primeiro valor a quantidade de resultados positivos e o segundo a quantidade total de registros que se enquadram na validação do nó. Baseando-se na Figura 19, a cláusula de validação é: se o fluxo for baixo, se a loja for ShoppingA, se o período for matutino e o dia da semana for segunda-feira.

Figura 19 - Percentual de assertividade

```
Se fluxo:
    = Sem fluxo (0 clientes)
        Estimativa de atendimento: sem atendimento, 91,26% de assertividade baseado em 103,0 registros na base de conhecimento.
    = Fluxo baixo (1 a 50 clientes)
        Se loja:
            = Shopping A
                Se período:
                    = Matutino (10h - 12h)
                        Se dia da semana:
                            = segunda-feira
                                Estimativa de atendimento: de 1 à 3, 75,00% de assertividade baseado em 32,00 registros na base de conhecimento.
```

Fonte: elaborado pelo autor.

Para esta validação, retornaram 32 registros, destes 24 registros retornaram positivo para o número estimado de vendas de 1 a 3. O cálculo realizado é o percentual de 24 para 30 registros, resultando em 75%.

Conforme demonstrado na Figura 19, a leitura desta informação deve ser realizada da seguinte forma: Numa segunda-feira (linha 10), no período da manhã (linha 8), no Shopping A (linha 6), com o fluxo baixo (linha 4), a estimativa de vendas é de 1 a 3. O fluxo de leitura deve ser realizado de cima para baixo, onde cada informação iniciada com a palavra `Se` é uma condição ou questionamento. Cada informação iniciada com o símbolo = , é a possível

resposta para o questionamento. Para cada informação iniciada com a palavra Estimativa, é o resultado para as condições anteriores.

No exemplo da Figura 19, é possível identificar que o movimento é baixo na segunda-feira pela manhã, com isto o usuário pode, por exemplo, decidir que não é necessário manter três funcionarios neste período e sim realocá-lo em um período de maior movimento. Desta forma é possível minizar os custos ao manter muitos funcionários para atender em poucos horários de grande movimento, podendo apenas organizar os horários dos funcionários.

3.4 ANÁLISE DOS RESULTADOS

Nesta seção são apresentados os experimentos realizados no software. Na seção 3.4.1 é realizada a análise dos resultados da árvore de decisão gerada no desenvolvimento deste trabalho. Na seção 3.4.2 é apresentada a comparação deste trabalho com os seus correlatos.

3.4.1 Análise da árvore de decisão

Conforme demonstrado no item 3.3.1.3, o processo de transformação e classificação dos dados é essencial para a realização da mineração e deve ser realizado exaustivamente para que os valores esperados sejam obtidos com maior grau de assertividade. Inicialmente o arquivo que fora gerado com os atributos numéricos, não era possível obter uma árvore de decisão compreensível e que fosse relevante para a tomada de decisões. Com a execução destas etapas, foi possível obter uma árvore mais simplificada, com resultados possíveis de compreender visualmente e permitir a obtenção de informações úteis para uma tomada de decisão.

No Quadro 22 é apresentada a relação da árvore de decisão gerada pelo software e a árvore gerada pela ferramenta Weka.

Quadro 22 - Comparação entre a ferramenta Weka e o software desenvolvido

Característica / árvore gerada	**Ferramenta Weka**	**Software desenvolvido**
Utilização do algoritmo J48	Sim	Sim
Utilização do arquivo ARFF gerado	Sim	Sim
Número de nós gerados	90	90
Nível mais profundo da árvore	7	7
Apresentação da árvore de decisão	Sim	Sim
Formato de apresentação da árvore	Horizontal	Vertical
Simplificação dos textos apresentados	Não	Sim

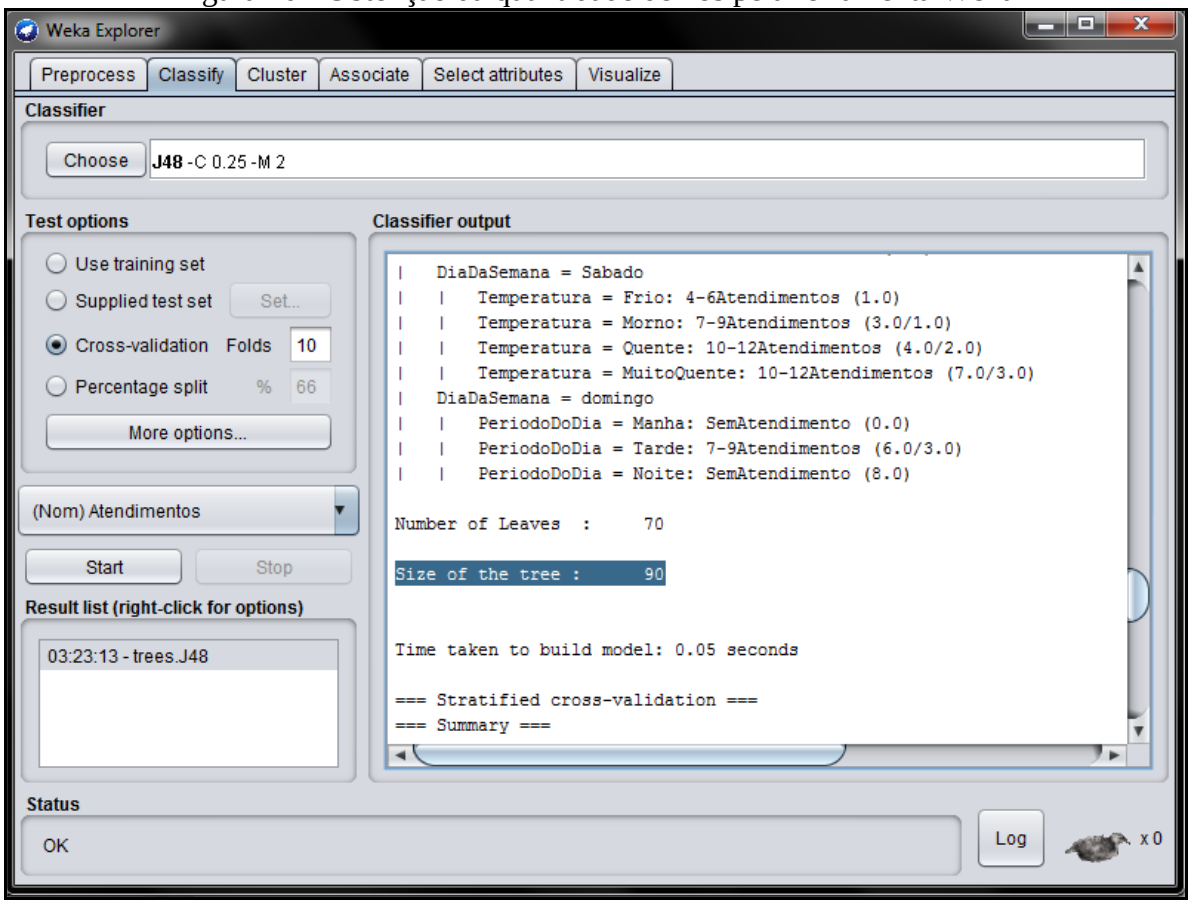
Fonte: elaborado pelo autor.

A primeira característica comparada é a utilização do algoritmo J48, em ambos foi realizada a utilização do algoritmo.

A segunda característica é a utilização do arquivo ARFF gerado por este trabalho, onde ambos utilizaram o mesmo arquivo para a aplicação do algoritmo.

A terceira característica é a quantidade de nós gerados na utilização do algoritmo, onde ambos apresentaram a mesma quantidade de nós. Para obtenção da quantidade de nós gerados pela ferramenta Weka, foi realizada a aplicação do algoritmo pela ferramenta e visualizado no *output* gerado a quantidade de nós, conforme demonstrado na Figura 20.

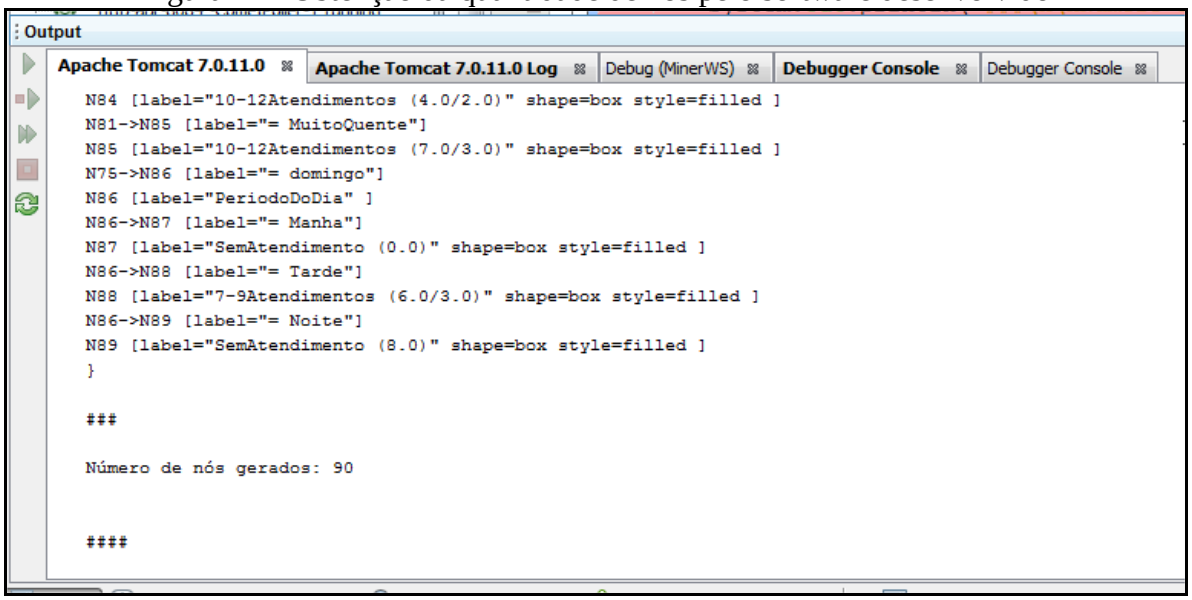
Figura 20 - Obtenção da quantidade de nós pela ferramenta Weka



Fonte: elaborado pelo autor.

Para a obtenção da quantidade de nós gerados pelo software desenvolvido, foi obtido o valor gerado no *output* da aplicação, conforme Figura 21.

Figura 21 - Obtenção da quantidade de nós pelo software desenvolvido



Fonte: elaborado pelo autor.

A quarta característica é o nível mais profundo da árvore gerada, onde ambas apresentaram o mesmo tamanho da árvore. Em ambos os casos foi necessário realizar a contagem do nível de forma manual.

A quinta característica é a geração de árvore de decisão e apresentação ao usuário. Em ambos os casos foi gerada e apresentada corretamente a árvore de decisão.

A sexta característica é o formato em que a árvore é apresentada. No caso da ferramenta Weka, a árvore é apresentada em ramos organizados de forma horizontal e conforme os níveis são aprofundados, é apresentado em uma nova linha abaixo do nó pai. No software desenvolvido, os nós irmãos são apresentados na mesma coluna, ou seja, um abaixo do outro e conforme os níveis são aprofundados, o nó filho é apresentado mais a direita do nó pai. Neste ponto, o software desenvolvido tem vantagem sobre a ferramenta Weka, pois para a visualização completa das informações somente é necessária a utilização do scroll no sentido vertical. No caso da ferramenta Weka é necessário navegar com o mouse arrastando no sentido horizontal e vertical.

A última característica analisada é a simplificação dos textos apresentados para o usuário. No caso da ferramenta Weka são apresentados os mesmos textos dos atributos do arquivo ARFF. No software desenvolvido foi priorizada a apresentação que um usuário que não conheça a metodologia do processo de mineração, consiga visualizar e compreender as informações. No software é realizado o tratamento para a apresentação do percentual de assertividade e na ferramenta Weka apresenta a quantidade de registros positivos e negativos para cada validação.

Portanto, pode-se concluir que o software desenvolvido possui duas características que diferem da ferramenta Weka e que permitem uma melhor compreensão e utilização pelo usuário.

3.4.2 Comparação entre o software desenvolvido e seus correlatos

O Quadro 23 apresenta um comparativo entre o software desenvolvido e os trabalhos correlatos.

Quadro 23 - Comparação entre o software e os trabalhos correlatos

Característica / trabalhos	**Pereira (2017)**	**Juste (2013)**	**Gerosa (2011)**	**Librelotto (2013)**
Gera árvore de decisão	Sim	Sim	Sim	Não
Utiliza arquivo ARFF para mineração	Sim	Sim	Não	Não
Utiliza banco de dados para mineração	Não	Não	Não	Sim
Realiza as etapas de KDD	Sim	Sim	Não	Sim
Utiliza o algoritmo J48	Sim	Sim	Não	Sim
Realiza a geração de gráficos	Não	Não	Sim	Não
Realiza comparação entre algoritmos	Não	Sim	Não	Sim
Utiliza dados reais para base de conhecimento	Sim	Não	Sim	Não
Aplica os resultados em situação real	Não	Não	Não	Não

Fonte: elaborado pelo autor.

Com relação à geração de árvore de decisão, somente o trabalho de Librelotto (2013) não realiza, os demais geram a árvore de decisão. Com relação à forma com que os dados são obtidos para realização da mineração dos dados, Juste (2013) e o software desenvolvido utilizam o arquivo ARFF para mineração, o trabalho de Librelotto (2013) utiliza diretamente o banco de dados, enquanto Gerosa (2011) utiliza arquivos de repositório de projetos.

Quanto à utilização das etapas de KDD e a aplicação do algoritmo J48, somente Gerosa (2011) não as aplica em seu trabalho, porém, é o único trabalho a realizar a geração de gráficos como resultado da mineração para apresentação ao usuário. Quanto à comparação entre algoritmos, somente Juste (2013) e Librelotto (2013) realizam a comparação do algoritmo J48 com outros algoritmos para comprovação de assertividade. Em ambos os trabalhos o algoritmo J48 apresentou melhores resultados. Quanto à utilização de dados reais, o software desenvolvido e o trabalho de Gerosa (2011) utilizaram a base de dados de conhecimento com registros reais. Quanto à aplicação dos resultados gerados em situações reais, nenhum dos trabalhos realizou a aplicação dos softwares desenvolvidos.

4 CONCLUSÕES

Com o gigantesco volume de dados gerado atualmente, torna-se necessário o processo de mineração de dados para obtenção de conhecimento útil para o processo de tomada de decisões. Com a concorrência cada vez mais forte no nicho de vendas de varejo, essas decisões precisam ser tomadas da forma mais rápida e com o maior nível de assertividade possível.

O objetivo de desenvolver um software para auxiliar neste processo permitindo um ganho estratégico para as empresas, foi atingido utilizando o algoritmo J48 disponibilizado pela ferramenta Weka e a geração da árvore de decisão. Nas etapas de pré-processamento, refinamento e transformação foi possível transformar os dados quantitativos em qualitativos, conforme demonstrado nos itens 3.3.1.1, 3.3.1.2 e 3.3.1.3. Através da mineração de dados, demonstrada no item 3.3.1.5, é possível identificar os fatores que influenciam no número de vendas e a quantidade esperada para cada combinação de atributos. Os resultados gerados podem ser observados na Figura 18.

Em função do prazo para a execução do projeto, o resultado deste trabalho não foi submetido a empresas de vendas de varejo para realizar a validação dos dados obtidos, contudo, esta é uma sugestão de continuidade do trabalho.

Neste trabalho somente foram obtidas da base de dados, informações sobre o número de vendas, quantidade de fluxo na loja, quantidade de fluxo na vitrine, a data e hora de cada registro. Para obter informações mais relevantes, seria necessário cruzar estes dados com os dados dos clientes que realizaram as compras. Desta forma será possível identificar o perfil de cada cliente, se são homens ou mulheres, se possuem filhos, frequência de compra, os produtos que são mais vendidos em determinado período e etc.. Com estas informações é possível manter um estoque mais propício à venda em determinado período conforme os dados históricos da base de conhecimento, minimizando os gastos em armazenamento, transporte e compra.

O processo de KDD é contínuo, os dados devem ser regularmente atualizados e aprimorados pelas etapas de pré-processamento e transformação, para que seja possível obter informações mais assertivas.

4.1 EXTENSÕES

Algumas das possíveis extensões para este trabalho são:

a) realizar a validação dos resultados obtidos, em uma empresa real;

b) permitir a seleção do arquivo que deve ser minerado: realizar o tratamento de

forma genérica dos dados para que seja possível selecionar outros arquivos para realizar a mineração;

c) permitir a edição do arquivo para mineração: desenvolver uma tela na aplicação para que o usuário possa importar um arquivo xls com os dados de mineração, gerar um arquivo ARFF e permitir que o usuário consiga editar o arquivo;

d) execução de outros algoritmos para geração da árvore de decisão: permitir ao usuário selecionar o algoritmo que deseja executar.

REFERÊNCIAS

ARFF, 2016. In: **Weka wikispaces**. [S.I.]: Weka The University Of Waikato, 2016. Disponível em: <http://weka.wikispaces.com/ARFF+%28book+version%29>. Acessado em: 20 jun. 2017.

DANTAS, Eric Rommel G. et al. **O uso da descoberta de conhecimento em base de dados para apoiar a tomada de decisões**. In: SIMPÓSIO DE EXCELÊNCIA EM GESTÃO E TECNOLOGIA - SEGET, 5, 2008, João Pessoa - PB. Artigo. Resende - RJ: Aedb, 2008. p. 1 - 10. Disponível em: <http://www.aedb.br/seget/arquivos/artigos08/331_331_Artigo_SEGET_EJDR_Versao_Final_010808.pdf>. Acesso em: 10 mar. 2017.

FAYYAD, Usama M.; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. **From data mining to knowledge discovery**: an overview. San Francisco - CA - UA: American Association For Artificial Intelligence Menlo Park, 1996. 34 p.

FRANK, Eibe; HALL, Mark A.; WITTEN, Ian H.. **The WEKA Workbench:** Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques". 4. ed. Cambridge - United States: Morgan Kaufmann, 2016. (Morgan Kaufmann Series in Data Management Systems). Disponível em: <http://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf>. Acesso em: 08 jun. 2017.

GEROSA, Marco Aurélio et al. **MinerAll**: Uma ferramenta para extração e mineração de dados de repositórios de software livre. 2011. Disponível em: <https://www.academia.edu/955453/MinerAll_Uma_ferramenta_para_extração_e_mineração_de_dados_de_repositórios_de_software_livre>. Acesso em: 14 set. 2016.

JUSTE, Gleice Ebili. **Uma proposta de mineração de dados na base de dados do REDECA utilizando a ferramenta Weka**. 2013. 63 f. TCC (Graduação) - Curso de Graduação, Instituto Municipal de Ensino Superior de Assis, Assis, 2013. Disponível em: <https://cepein.femanet.com.br/BDigital/arqTccs/1011330122.pdf>. Acesso em: 20 set. 2016.

LIBRELOTTO, Solange Rubert; MOZZAQUATRO, Patricia Mariotto. Análise dos algoritmos de mineração J48 e Apriori aplicados na detecção de indicadores da qualidade de vida e saúde. **Revint: Revista Interdisciplinar de Ensino, Pesquisa e Extensão**, Cruz Alta - RS, v. 1, n. 1, p.1-37, jun. 2013. Disponível em: <http://www.revistaeletronica.unicruz.edu.br/index.php/eletronica/article/viewFile/26-37/pdf>. Acesso em: 20 jun. 2017.

MACEDO, Dayana Carla de; MATOS, Simone Nasser. Extração de conhecimento através da mineração de dados. **Revista de Engenharia e Tecnologia**, Curitiba - PR, v. 2, n. 2, p.23-29, ago. 2010. Disponível em: <http://www.revistaret.com.br/ojs-2.2.3/index.php/ret/article/viewFile/38/73>. Acesso em: 20 jun. 2017.

REZENDE, Solange Oliveira. **Sistemas Inteligentes**: Fundamentos e Aplicações. Barueri - SP: Manole Ltda, 2003.

SILVA, Marcelino Pereira dos Santos. **Mineração de dados - conceitos, aplicações e experimentos com Weka.** 2004. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/erirjes/2004/004.pdf>. Acesso em: 10 out. 2016.

TENFEN, Emerson. **A técnica de Knowledge Discovery In Databases (KDD) aplicada nas ocorrências atendidas pela polícia militar**. 2003. 50 f. TCC (Graduação) - Curso de Ciências da Computação, Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau - Furb, Blumenau - Sc, 2003. Disponível em: <http://dsc.inf.furb.br/arquivos/tccs/monografias/2003-1emersontenfenvf.pdf>. Acesso em: 14 jun. 2017.

WU, Xindong et al. **Top 10 algorithms in data mining**. Knowledge And Information Systems, [s.l.], v. 14, n. 1, p.1-37, 4 dez. 2007. Springer Nature. http://dx.doi.org/10.1007/s10115-007-0114-2. Disponível em: <http://www.cs.umd.edu/~samir/498/10Algorithms-08.pdf>. Acesso em: 20 jun. 2017.

APÊNDICE A – Arquivo ARFF utilizado na mineração de dados

A seguir é apresentada parte do arquivo gerado neste trabalho, o arquivo fora separado em dois quadros, Quadro 24 e Quadro 25.

Quadro 24 - Arquivo ARFF gerado pelo software com dados da loja Shopping A

```
@relation dados

@attribute Loja {ShoppingA, ShoppingB}
@attribute DiaDaSemana {segunda-feira, terca-feira, quarta-feira, quinta-feira, sexta-feira, Sabado, domingo}
@attribute Mes {Fevereiro, Marco}
@attribute PeriodoDoDia {Manha, Tarde, Noite}
@attribute Fluxo {SemFluxo, FluxoBaixo, FluxoModerado, FluxoAlto}
@attribute FluxoDaVitrine {VitrineNenhumMovimento, VitrineBaixo, VitrineMedio, VitrineAlto}
@attribute Temperatura {Frio, Morno, Quente, MuitoQuente}
@attribute PrevisaoDoTempo {SemChuva, PoucaChuva, ChuvaLeve, ChuvaModerada, ChuvaForte, Tempestade}
@attribute Atendimentos {SemAtendimento, 1-3Atendimentos, 4-6Atendimentos, 7-9Atendimentos, 10-12Atendimentos, 13-15Atendimentos, 19Atendimentos}

@data
ShoppingA,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,SemAtendimento
ShoppingA,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,4-6Atendimentos
ShoppingA,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,4-6Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineBaixo,Morno,PoucaChuva,4-6Atendimentos
ShoppingA,quarta-feira,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Quente,PoucaChuva,1-3Atendimentos
ShoppingA,quarta-feira,Fevereiro,Noite,FluxoModerado,VitrineMedio,Quente,PoucaChuva,4-6Atendimentos
ShoppingA,quarta-feira,Fevereiro,Noite,FluxoBaixo,VitrineAlto,Quente,PoucaChuva,10-12Atendimentos
ShoppingA,quarta-feira,Fevereiro,Noite,FluxoAlto,VitrineMedio,Quente,PoucaChuva,7-9Atendimentos
ShoppingA,quarta-feira,Fevereiro,Noite,FluxoModerado,VitrineMedio,Quente,PoucaChuva,SemAtendimento
ShoppingA,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Frio,ChuvaForte,SemAtendimento
ShoppingA,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Frio,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoAlto,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Morno,ChuvaForte,7-9Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Morno,ChuvaForte,4-6Atendimentos
ShoppingA,quinta-feira,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoAlto,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoAlto,VitrineAlto,Morno,ChuvaForte,1-3Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoAlto,VitrineAlto,Morno,ChuvaForte,4-6Atendimentos
ShoppingA,quinta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,Morno,ChuvaForte,SemAtendimento
ShoppingA,sexta-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Quente,PoucaChuva,SemAtendimento
ShoppingA,sexta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,PoucaChuva,1-3Atendimentos
ShoppingA,sexta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,sexta-feira,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Morno,PoucaChuva,4-6Atendimentos
ShoppingA,sexta-feira,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Morno,PoucaChuva,7-9Atendimentos
ShoppingA,sexta-feira,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Morno,PoucaChuva,7-9Atendimentos
ShoppingA,sexta-feira,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,sexta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineAlto,Morno,PoucaChuva,1-3Atendimentos
ShoppingA,sexta-feira,Fevereiro,Tarde,FluxoModerado,VitrineMedio,MuitoQuente,PoucaChuva,4-6Atendimentos
ShoppingA,sexta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,PoucaChuva,1-3Atendimentos
ShoppingA,sexta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,PoucaChuva,4-6Atendimentos
ShoppingA,sexta-feira,Fevereiro,Noite,FluxoModerado,VitrineMedio,MuitoQuente,PoucaChuva,7-9Atendimentos
ShoppingA,sexta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,PoucaChuva,SemAtendimento
ShoppingA,Sabado,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Quente,ChuvaLeve,1-3Atendimentos
ShoppingA,Sabado,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,ChuvaLeve,4-6Atendimentos
ShoppingA,Sabado,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Quente,ChuvaLeve,7-9Atendimentos
ShoppingA,Sabado,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,ChuvaLeve,4-6Atendimentos
ShoppingA,Sabado,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Quente,ChuvaLeve,7-9Atendimentos
ShoppingA,Sabado,Fevereiro,Tarde,FluxoModerado,VitrineBaixo,Quente,ChuvaLeve,10-12Atendimentos
ShoppingA,Sabado,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Quente,ChuvaLeve,13-15Atendimentos
ShoppingA,Sabado,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Quente,ChuvaLeve,10-12Atendimentos
ShoppingA,Sabado,Fevereiro,Tarde,FluxoAlto,VitrineAlto,MuitoQuente,ChuvaLeve,10-12Atendimentos
ShoppingA,Sabado,Fevereiro,Noite,FluxoAlto,VitrineAlto,MuitoQuente,ChuvaLeve,10-12Atendimentos
ShoppingA,Sabado,Fevereiro,Noite,FluxoModerado,VitrineAlto,MuitoQuente,ChuvaLeve,7-9Atendimentos
ShoppingA,Sabado,Fevereiro,Noite,FluxoAlto,VitrineAlto,MuitoQuente,ChuvaLeve,7-9Atendimentos
ShoppingA,Sabado,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,MuitoQuente,ChuvaLeve,1-3Atendimentos
ShoppingA,domingo,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,ChuvaModerada,SemAtendimento
ShoppingA,domingo,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,ChuvaModerada,SemAtendimento
ShoppingA,domingo,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Quente,ChuvaModerada,1-3Atendimentos
ShoppingA,domingo,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Quente,ChuvaModerada,4-6Atendimentos
ShoppingA,domingo,Fevereiro,Tarde,FluxoModerado,VitrineAlto,Quente,ChuvaModerada,13-15Atendimentos
ShoppingA,domingo,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Quente,ChuvaModerada,10-12Atendimentos
ShoppingA,domingo,Fevereiro,Tarde,FluxoModerado,VitrineBaixo,Quente,ChuvaModerada,4-6Atendimentos
ShoppingA,domingo,Fevereiro,Tarde,FluxoAlto,VitrineAlto,Quente,ChuvaModerada,7-9Atendimentos
ShoppingA,domingo,Fevereiro,Tarde,FluxoAlto,VitrineAlto,MuitoQuente,ChuvaModerada,7-9Atendimentos
ShoppingA,domingo,Fevereiro,Noite,FluxoModerado,VitrineAlto,MuitoQuente,ChuvaModerada,7-9Atendimentos
ShoppingA,domingo,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,ChuvaModerada,SemAtendimento
ShoppingA,domingo,Fevereiro,Noite,FluxoAlto,VitrineBaixo,MuitoQuente,ChuvaModerada,SemAtendimento
ShoppingA,domingo,Fevereiro,Noite,SemFluxo,VitrineBaixo,MuitoQuente,ChuvaModerada,SemAtendimento
ShoppingA,segunda-feira,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Quente,SemChuva,1-3Atendimentos
ShoppingA,segunda-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,SemChuva,4-6Atendimentos
ShoppingA,segunda-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,SemChuva,1-3Atendimentos
ShoppingA,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,SemChuva,1-3Atendimentos
ShoppingA,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,SemChuva,1-3Atendimentos
ShoppingA,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,SemChuva,7-9Atendimentos
ShoppingA,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,SemChuva,4-6Atendimentos
```

Fonte: elaborado pelo autor.

Quadro 25 - Arquivo ARFF gerado pelo software com dados da loja Shopping B

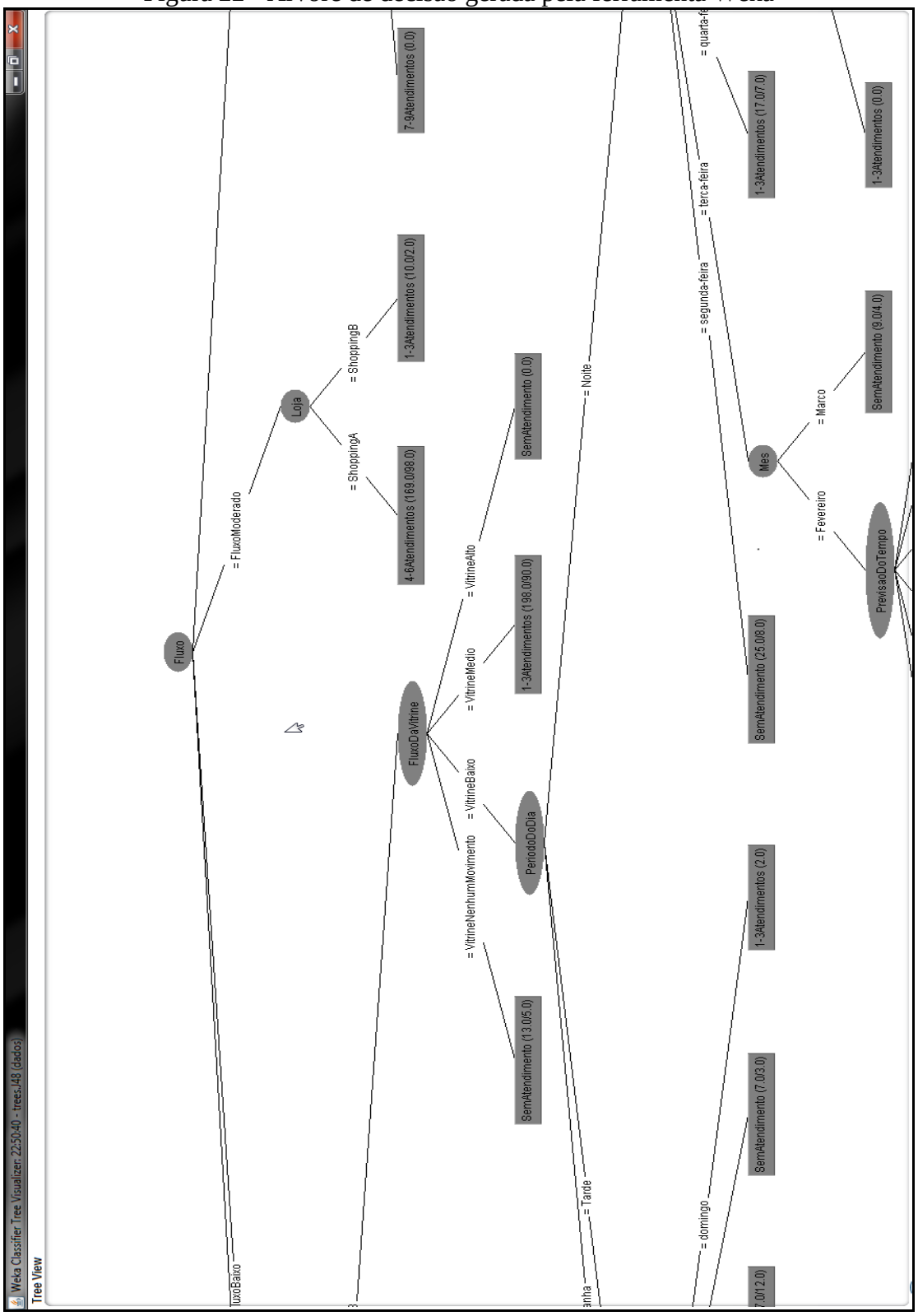
```
ShoppingB,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,SemAtendimento
ShoppingB,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,SemAtendimento
ShoppingB,quarta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,SemAtendimento
ShoppingB,quarta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,SemAtendimento
ShoppingB,quarta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,SemAtendimento
ShoppingB,quarta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,1-3Atendimentos
ShoppingB,quarta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,1-3Atendimentos
ShoppingB,quarta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,SemAtendimento
ShoppingB,quarta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,PoucaChuva,SemAtendimento
ShoppingB,quarta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,Quente,PoucaChuva,1-3Atendimentos
ShoppingB,quarta-feira,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,Quente,PoucaChuva,1-3Atendimentos
ShoppingB,quarta-feira,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,Quente,PoucaChuva,1-3Atendimentos
ShoppingB,quarta-feira,Fevereiro,Noite,SemFluxo,VitrineNenhumMovimento,Quente,PoucaChuva,SemAtendimento
ShoppingB,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Frio,ChuvaForte,SemAtendimento
ShoppingB,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Frio,ChuvaForte,SemAtendimento
ShoppingB,quinta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,ChuvaForte,SemAtendimento
ShoppingB,quinta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Morno,ChuvaForte,SemAtendimento
ShoppingB,quinta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Morno,ChuvaForte,SemAtendimento
ShoppingB,quinta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,ChuvaForte,1-3Atendimentos
ShoppingB,quinta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,ChuvaForte,SemAtendimento
ShoppingB,quinta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Morno,ChuvaForte,1-3Atendimentos
ShoppingB,quinta-feira,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,Morno,ChuvaForte,1-3Atendimentos
ShoppingB,quinta-feira,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,Morno,ChuvaForte,1-3Atendimentos
ShoppingB,quinta-feira,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,Morno,ChuvaForte,SemAtendimento
ShoppingB,quinta-feira,Fevereiro,Noite,SemFluxo,VitrineNenhumMovimento,Morno,ChuvaForte,SemAtendimento
ShoppingB,sexta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,PoucaChuva,SemAtendimento
ShoppingB,sexta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,PoucaChuva,SemAtendimento
ShoppingB,sexta-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,1-3Atendimentos
ShoppingB,sexta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,1-3Atendimentos
ShoppingB,sexta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,1-3Atendimentos
ShoppingB,sexta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,SemAtendimento
ShoppingB,sexta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Morno,PoucaChuva,1-3Atendimentos
ShoppingB,sexta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Morno,PoucaChuva,1-3Atendimentos
ShoppingB,sexta-feira,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,MuitoQuente,PoucaChuva,4-6Atendimentos
ShoppingB,sexta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,PoucaChuva,1-3Atendimentos
ShoppingB,sexta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,PoucaChuva,1-3Atendimentos
ShoppingB,sexta-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,PoucaChuva,4-6Atendimentos
ShoppingB,sexta-feira,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,MuitoQuente,PoucaChuva,1-3Atendimentos
ShoppingB,Sabado,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,ChuvaLeve,SemAtendimento
ShoppingB,Sabado,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,ChuvaLeve,SemAtendimento
ShoppingB,Sabado,Fevereiro,Manha,FluxoBaixo,VitrineMedio,Quente,ChuvaLeve,1-3Atendimentos
ShoppingB,Sabado,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,ChuvaLeve,4-6Atendimentos
ShoppingB,Sabado,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,ChuvaLeve,4-6Atendimentos
ShoppingB,Sabado,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,ChuvaLeve,1-3Atendimentos
ShoppingB,Sabado,Fevereiro,Tarde,FluxoModerado,VitrineMedio,Quente,ChuvaLeve,1-3Atendimentos
ShoppingB,Sabado,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,ChuvaLeve,4-6Atendimentos
ShoppingB,Sabado,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,MuitoQuente,ChuvaLeve,1-3Atendimentos
ShoppingB,Sabado,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,ChuvaLeve,1-3Atendimentos
ShoppingB,Sabado,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,ChuvaLeve,4-6Atendimentos
ShoppingB,Sabado,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,ChuvaLeve,4-6Atendimentos
ShoppingB,Sabado,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,ChuvaLeve,SemAtendimento
ShoppingB,domingo,Fevereiro,Tarde,SemFluxo,VitrineBaixo,Quente,ChuvaModerada,SemAtendimento
ShoppingB,domingo,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,ChuvaModerada,1-3Atendimentos
ShoppingB,domingo,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,ChuvaModerada,1-3Atendimentos
ShoppingB,domingo,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,ChuvaModerada,1-3Atendimentos
ShoppingB,domingo,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,Quente,ChuvaModerada,4-6Atendimentos
ShoppingB,domingo,Fevereiro,Tarde,FluxoBaixo,VitrineMedio,MuitoQuente,ChuvaModerada,1-3Atendimentos
ShoppingB,domingo,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,ChuvaModerada,1-3Atendimentos
ShoppingB,domingo,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,MuitoQuente,ChuvaModerada,SemAtendimento
ShoppingB,segunda-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,SemChuva,1-3Atendimentos
ShoppingB,segunda-feira,Fevereiro,Manha,SemFluxo,VitrineBaixo,Quente,SemChuva,SemAtendimento
ShoppingB,segunda-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Quente,SemChuva,SemAtendimento
ShoppingB,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,SemChuva,1-3Atendimentos
ShoppingB,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,SemChuva,1-3Atendimentos
ShoppingB,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,SemChuva,1-3Atendimentos
ShoppingB,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,SemChuva,SemAtendimento
ShoppingB,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,Quente,SemChuva,1-3Atendimentos
ShoppingB,segunda-feira,Fevereiro,Tarde,FluxoBaixo,VitrineBaixo,MuitoQuente,SemChuva,1-3Atendimentos
ShoppingB,segunda-feira,Fevereiro,Noite,FluxoBaixo,VitrineMedio,MuitoQuente,SemChuva,SemAtendimento
ShoppingB,segunda-feira,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,MuitoQuente,SemChuva,SemAtendimento
ShoppingB,segunda-feira,Fevereiro,Noite,FluxoBaixo,VitrineBaixo,MuitoQuente,SemChuva,SemAtendimento
ShoppingB,segunda-feira,Fevereiro,Noite,SemFluxo,VitrineBaixo,MuitoQuente,SemChuva,SemAtendimento
ShoppingB,terca-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Frio,Tempestade,SemAtendimento
ShoppingB,terca-feira,Fevereiro,Manha,FluxoBaixo,VitrineBaixo,Frio,Tempestade,SemAtendimento
```

Fonte: elaborado pelo autor.

APÊNDICE B – Árvore de decisão gerada pela ferramenta Weka

A seguir é apresentada na Figura 22, parte da árvore gerada pela ferramenta Weka, utilizando a opção de zoom para visualizar as informações.

Figura 22 - Árvore de decisão gerada pela ferramenta Weka



Fonte: elaborado pelo autor.

APÊNDICE C – Árvore de decisão gerada pelo software

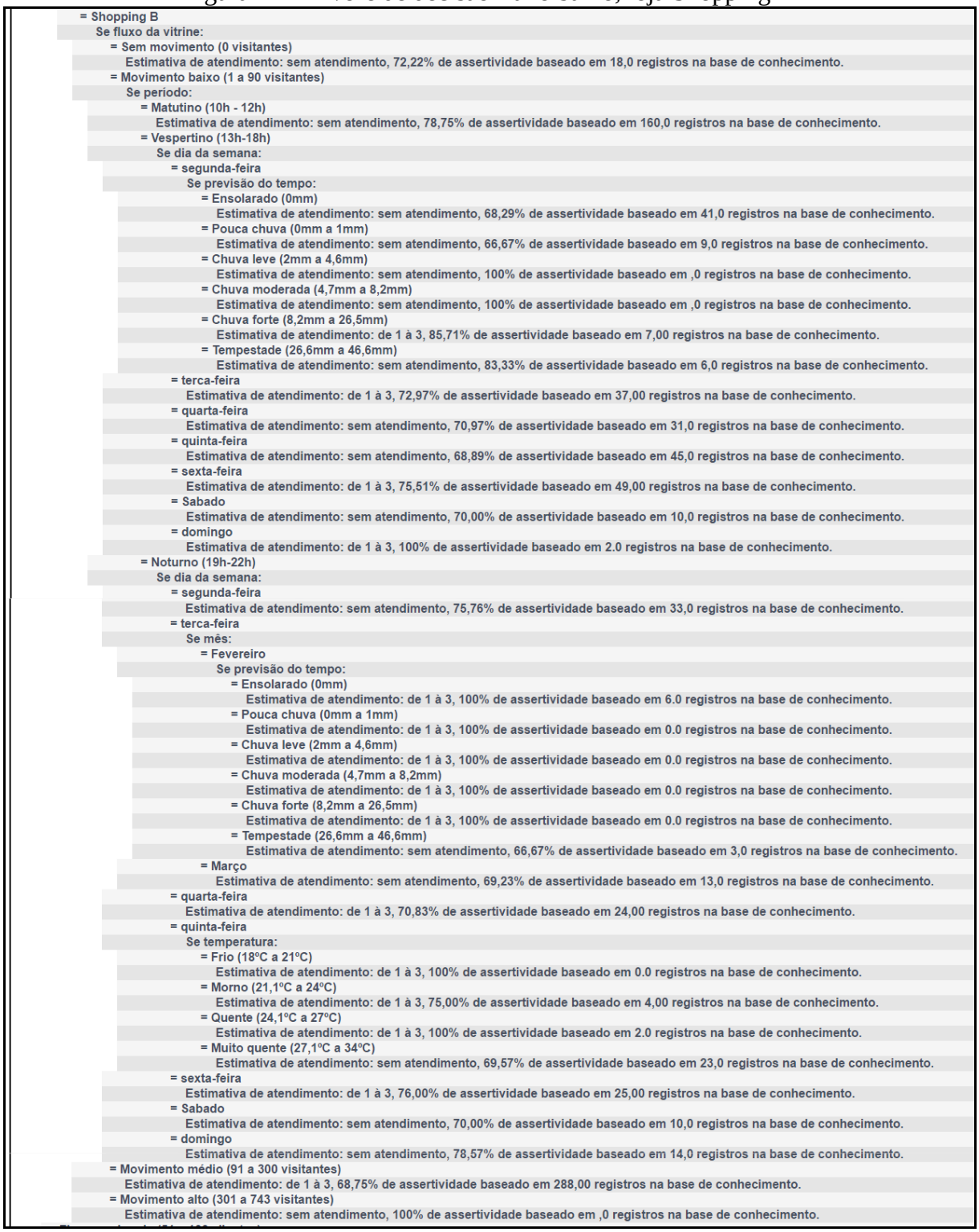
A seguir é apresentada a árvore completa gerada pelo software, dividida em três figuras. A Figura 23 apresenta a árvore de fluxo baixo até o nível da loja Shopping A. A Figura 24 apresenta a árvore de fluxo baixo até o nível da loja Shopping B. A Figura 25 apresenta o fluxo moderado e alto de ambas as lojas.

Figura 23 - Árvore de decisão fluxo baixo, loja Shopping A

```
Se fluxo:
  = Sem fluxo (0 clientes)
    Estimativa de atendimento: sem atendimento, 91,26% de assertividade baseado em 103,0 registros na base de conhecimento.
  = Fluxo baixo (1 a 50 clientes)
    Se loja:
      = Shopping A
        Se período:
          = Matutino (10h - 12h)
            Se dia da semana:
              = segunda-feira
                Estimativa de atendimento: de 1 à 3, 75,00% de assertividade baseado em 32,00 registros na base de conhecimento.
              = terca-feira
                Estimativa de atendimento: de 1 à 3, 69,23% de assertividade baseado em 26,00 registros na base de conhecimento.
              = quarta-feira
                Se previsão do tempo:
                  = Ensolarado (0mm)
                    Estimativa de atendimento: de 1 à 3, 73,68% de assertividade baseado em 19,00 registros na base de conhecimento.
                  = Pouca chuva (0mm a 1mm)
                    Estimativa de atendimento: de 4 à 6, 75,00% de assertividade baseado em 4,00 registros na base de conhecimento.
                  = Chuva leve (2mm a 4,6mm)
                    Estimativa de atendimento: de 1 à 3, 75,00% de assertividade baseado em 4,00 registros na base de conhecimento.
                  = Chuva moderada (4,7mm a 8,2mm)
                    Estimativa de atendimento: de 1 à 3, 100% de assertividade baseado em 0.0 registros na base de conhecimento.
                  = Chuva forte (8,2mm a 26,5mm)
                    Estimativa de atendimento: de 1 à 3, 100% de assertividade baseado em 0.0 registros na base de conhecimento.
                  = Tempestade (26,6mm a 46,6mm)
                    Estimativa de atendimento: sem atendimento, 75,00% de assertividade baseado em 4,0 registros na base de conhecimento.
              = quinta-feira
                Estimativa de atendimento: de 1 à 3, 66,67% de assertividade baseado em 36,00 registros na base de conhecimento.
              = sexta-feira
                Estimativa de atendimento: de 1 à 3, 80,00% de assertividade baseado em 30,00 registros na base de conhecimento.
              = Sabado
                Se temperatura:
                  = Frio (18ºC a 21ºC)
                    Estimativa de atendimento: de 4 à 6, 66,67% de assertividade baseado em 6,00 registros na base de conhecimento.
                  = Morno (21,1ºC a 24ºC)
                    Estimativa de atendimento: de 1 à 3, 76,92% de assertividade baseado em 13,00 registros na base de conhecimento.
                  = Quente (24,1ºC a 27ºC)
                    Estimativa de atendimento: de 1 à 3, 60,00% de assertividade baseado em 10,00 registros na base de conhecimento.
                  = Muito quente (27,1ºC a 34ºC)
                    Estimativa de atendimento: de 1 à 3, 100% de assertividade baseado em 0.0 registros na base de conhecimento.
              = domingo
                Se fluxo da vitrine:
                  = Sem movimento (0 visitantes)
                    Estimativa de atendimento: sem atendimento, 100% de assertividade baseado em ,0 registros na base de conhecimento.
                  = Movimento baixo (1 a 90 visitantes)
                    Estimativa de atendimento: sem atendimento, 94,44% de assertividade baseado em 18,0 registros na base de conhecimento.
                  = Movimento médio (91 a 300 visitantes)
                    Estimativa de atendimento: de 1 à 3, 85,71% de assertividade baseado em 7,00 registros na base de conhecimento.
                  = Movimento alto (301 a 743 visitantes)
                    Estimativa de atendimento: sem atendimento, 100% de assertividade baseado em 1,0 registros na base de conhecimento.
          = Vespertino (13h-18h)
            Estimativa de atendimento: de 1 à 3, 68,33% de assertividade baseado em 281,00 registros na base de conhecimento.
          = Noturno (19h-22h)
            Se fluxo da vitrine:
              = Sem movimento (0 visitantes)
                Estimativa de atendimento: de 1 à 3, 100% de assertividade baseado em 2.0 registros na base de conhecimento.
              = Movimento baixo (1 a 90 visitantes)
                Estimativa de atendimento: sem atendimento, 68,18% de assertividade baseado em 44,0 registros na base de conhecimento.
              = Movimento médio (91 a 300 visitantes)
                Estimativa de atendimento: de 1 à 3, 61,90% de assertividade baseado em 126,00 registros na base de conhecimento.
              = Movimento alto (301 a 743 visitantes)
                Estimativa de atendimento: de 1 à 3, 62,86% de assertividade baseado em 35,00 registros na base de conhecimento.
```

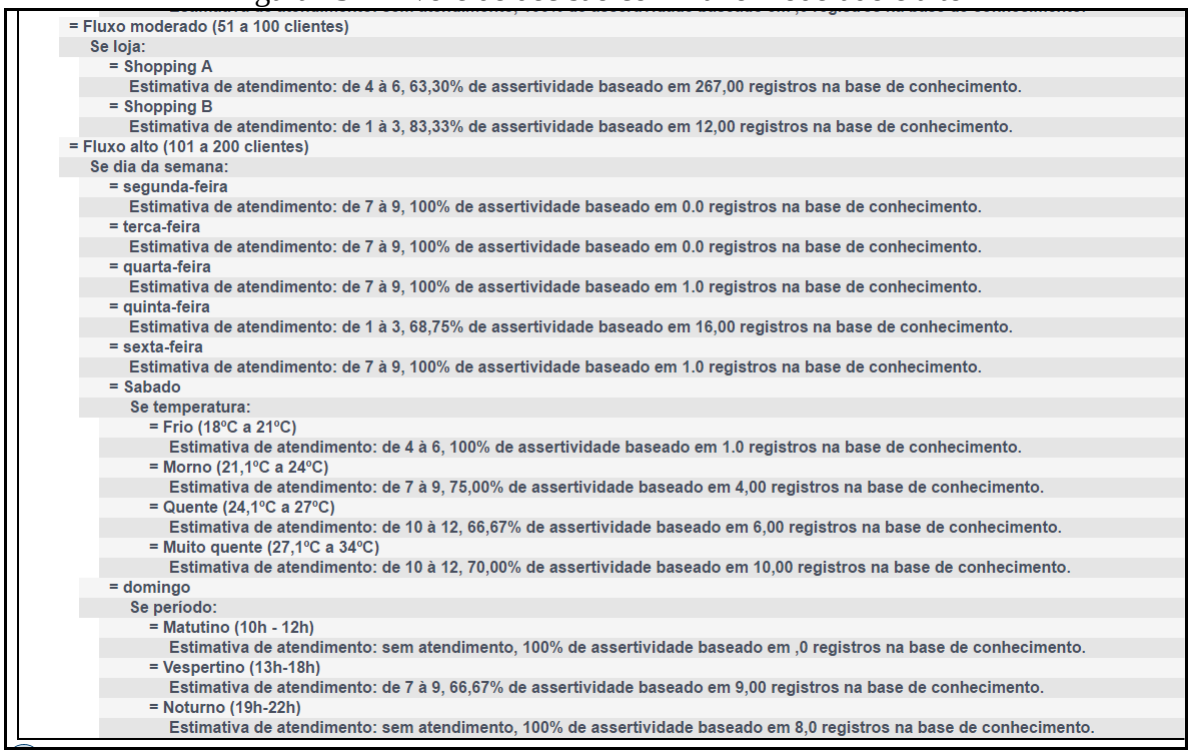
Fonte: elaborado pelo autor.

Figura 24 - Árvore de decisão fluxo baixo, loja Shopping B



Fonte: elaborado pelo autor.

Figura 25 - Árvore de decisão com fluxo moderado e alto



Fonte: elaborado pelo autor.