Universidade Federal de Pernambuco
Centro de informática

Graduação em Ciência da Computação

ISDA.R - Interval Data Analysis for R

Ricardo Jorge de Almeida Queiroz Filho

Trabalho de Graduação

Recife
4 de Julho de 2012

Universidade Federal de Pernambuco
Centro de informática

Ricardo Jorge de Almeida Queiroz Filho

ISDA.R - Interval Data Analysis for R

Trabalho apresentado ao Programa de Graduação em Ciência da Computação do Centro de informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientadora: *Profa. Dra. Renata Maria Cardoso Rodrigues Souza*

Recife
4 de Julho de 2012

Agradecimentos

Agradeço, primeiramente, a minha família por todo apoio e ensinamentos que me proveram durante todo esse tempo. Foi de fundamental importância o incentivo deles nos momentos difíceis e as críticas construtivas quando necessário, uma vez que, através delas, pude refletir e decidir quais eram as melhores abordagens para enfrentar os problemas.

Também é importante agradecer a professora e amiga Renata, que abriu muitas portas em minha vida acadêmica, ela me permitiu, logo no inicio do curso, o ingresso à monitoria de estatística e probabilidade para computação, experiência a qual me ajudou a construir um conhecimento mais sólido no ramo da estatística. Além disso, ela me introduziu à pesquisa, através da iniciação cientifica.

Sou muito grato, também, aos meus colegas de turma, em especial a Átila, Ícaro, Ivan e Leonardo, que foram meus companheiros em praticamente todos os projetos. Viramos muitas noites, ”quebrando a cabeça” para resolvermos os trabalhos acadêmicos postos, acarretando ao longo destes últimos anos uma sólida e inquebrantável amizade.

Finalmente, gostaria de agradecer ao Centro de Informática e a Universidade Federal de Pernambuco que ofereceu toda a infraestrutura técnica e material, imprescindíveis a minha formação, através de seu excelente corpo docente.

Um passo à frente e você não está mais no mesmo lugar.
—CHICO SCIENCE

Resumo

Devido ao crescimento explosivo das bases de dados, a extração e manuseio de informações se tornam mais difíceis. Uma forma para contornar esse problema é resumir essas bases, utilizando-se dos denominados dados simbólicos, que são capazes de representar mais informações que dados clássicos.

O presente trabalho é um pacote na linguagem R, cujo principal intuito é tratar de um tipo de dado simbólico específico que é o intervalar. Esse pacote permitirá tanto a transformação de variáveis clássicas em variáveis que armazenam dados intervalares, quanto à análise desses dados.

Palavras-chave: Data mining, Análise de dados, Dados Simbólicos.

Abstract

Due to the explosive growth of databases, extraction and handling of information becomes more difficult. One way to circumvent this problem is to summarize these bases, using the so-called symbolic data, which are capable of representing more information than classical data.

The present work is a package in R language, whose main purpose is to address a specific symbolic data type that is the "interval". This package will allow both the transformation of classical variables into variables that store data interval, and the analysis of such data.

Keywords: Data mining, Data Analysis, Symbolic data.

Sumário

Lista de Figuras

Lista de Tabelas

CAPÍTULO 1

Introdução

Ninguém consegue realizar nada sem a colaboração de muitos.
—PAOLA RHODEN

Bases de dados estão presentes em praticamente todas as grandes companhias e geralmente crescem atingindo um grande tamanho. Associado a esse crescimento das bases de dados, surge à dificuldade de manusear e extrair informações dessas bases. É evidente que mesmo em situações que as metodologias tradicionais aparentam ser aplicáveis, na prática, muitas vezes, o uso dessas técnicas não são de fato apropriadas.

Uma das maiores razões para as técnicas tradicionais falharem decorre do grande custo computacional necessário para fazer a análise dos dados em base muitos grandes. Mesmo com o poder de processamento crescendo, esse problema seguirá ocorrendo, pois as bases de dados acompanharão esse crescimento. Em consequência disso, apesar de os métodos tradicionais servirem bem para bases de dados menores, é necessário ao analista de dados se utilizar de procedimentos que funcionem melhor em bases maiores.

Uma abordagem para contornar esse problema é resumir essas bases de dados de maneira que o resultado seja um conjunto de dados manuseável. Nesse sentido, podem ser utilizados dados simbólicos.

Tais dados recebem a denominação de "simbólicos", uma vez que eles não são puramente numéricos, pois podem ser representados por números, intervalos, uma distribuição, um conjunto de pesos, uma sequência de valores com pesos como um histograma , etc. Esses dados simbólicos possuem uma estrutura interna que os permitem armazenar mais informações que um dado clássico, sendo, inclusive, capazes de armazenar variações internas e incertezas das variáveis.

Para melhor explicar o acima aduzido, tornar-se-á interessante se consignar o que se segue:

Ao se tratar de uma análise envolvendo a idade de jogadores de futebol, através de dados clássicos, teríamos um valor numérico associado a cada jogador, no entanto, se fossemos examinar tal situação por uma perspectiva em termos de dados simbólicos, poderíamos agrupar os jogadores por times, encontrando, assim, que, em determinado time, as idades dos jogadores teriam os valores entre 19 e 29 anos (intervalo [19,29]), ou seja, uma gama maior de informações seriam encontradas.

O objetivo do presente trabalho é desenvolver um pacote na linguagem R, para tratar de um dado simbólico específico que é o intervalar. Esse pacote, denominado ISDA.R, permitirá tanto a transformação de variáveis clássicas em variáveis que armazene dados intervalares, quanto à análise desses dados.

A linguagem R foi escolhida, uma vez que ela é amplamente utilizada por analistas de todo mundo e por ser gratuita, sendo distribuída sobre a licença GNU GPL. Por sua licença ser GNU ela garante uma série de liberdades ao usuário, dentre elas estão: a de executar o programa para qualquer propósito, a de estudar o programa e a possibilidade de modificá-lo de modo a adequar a suas necessidades.

Além disso, o R é extensível e flexível, pois ao contrário da maioria dos softwares de análise de dados que se utilizam de menus ”point-and-click”ou procedimentos ”caixa-preta”, o R é uma linguagem de programação. Portanto, a criação do pacote ISDA.R permitirá que estudantes, cientistas e analistas, que tenham interesse em análise de dados simbólicos, possam desenvolver seus trabalhos mais rapidamente, utilizando ou modificando o pacote de acordo com seus propósitos.

O capítulo 2 discorre a respeito de análise de dados simbólicos, definindo o que é análise de dados e quais os procedimentos para sua realização. Além disso, é introduzido a significação de dados simbólicos, como são encontrados e como são adaptados os métodos de análise para esse tipo de dados. No final desse capítulo, se fará menção ao estado da arte no que diz respeito a abordagem de análise por dados simbólicos.

O capítulo 4 do trabalho é uma demonstração prática do pacote ISDA.R aplicado a uma base de dados.

Por derradeiro , no último capítulo serão tratadas as conclusões e os trabalhos futuros.

CAPÍTULO 2

Analise de dados simbólicos

A estatística é a tentativa de torturar os números até eles confessarem.
—GENTIL SOUSA JR

2.1 Análise de dados

2.1.1 Introdução

Análise de dados é o processo de inspeção, transformação e construção de modelos com o objetivo de extrair conhecimento de dados. É importante saber que os dados por si só não são conhecimento, mas sim informação crua. O conhecimento é o efeito de se abstrair uma ideia ou noção de alguma coisa, enquanto a informação é o resultado do processamento, manipulação e organização dos dados de modo que represente uma modificação no conhecimento do sistema que a recebe [Beleza, 2009].

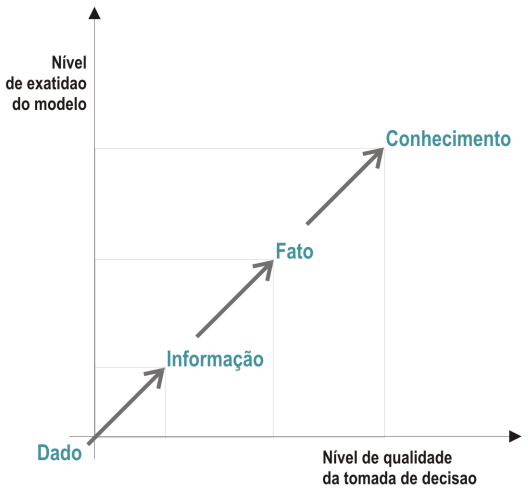

Figura 2.1 Exatidão do modelo x qualidade da tomada de decisão

O processo de transformação de dados para conhecimento é esquematizado pela figura

2.1.1.

O dado se torna informação no momento em que se torna relevante para o problema de decisão. Quando os dados permitem a extração de alguma ideia relacionada ao processo de tomada de decisão, essa informação passará a ser um fato. Fato, portanto, é tudo aquilo que os dados revelam. O fato se torna um conhecimento quando utilizado com sucesso para a conclusão do processo de tomada de decisão.

Considerando um ambiente incerto, a chance de se tomar boas decisões irá aumentar de acordo com a disponibilidade de "boas informações". Portanto, quanto melhor for o processo de descoberta de conhecimento, melhores serão as decisões tomadas. [Arshan, 2012]

2.1.2 O processo de análise de dados

O processo de análise de dados é interativo e iterativo, envolvendo vários passos e decisões. Uma breve introdução a esse procedimento é dada nessa seção. [Fayyad et al., 1996]

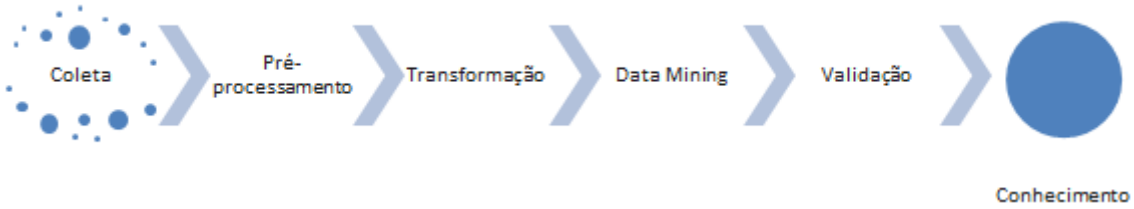

Figura 2.2 Esquema do processo de análise de dados

2.1.2.1 Coleta

Preliminarmente, tornar-se-á imprescindível se ter uma ideia exata do problema que será abordado, definindo-se, por consequência, os objetivos relevantes da pesquisa, para se iniciar a contento a coleta de dados.

A coleta de dados é uma tarefa muito importante no processo de análise de dados, uma vez que, se os dados em que se está trabalhando forem irrelevantes ou tendenciosos, a qualidade da análise será prejudicada. Levando-se em consideração o conceito de população como o conjunto de todos os elementos pertinentes ao estudo, o primeiro passo para se coletar os dados é definir a população em que se está buscando inferir algo. Como na maioria das vezes é impraticável coletar os dados para toda população, é comum se utilizar subconjuntos dessa população chamados de amostra para inferir informações da população. Os dados podem ser coletados de repositórios já existentes ou através de observações ou estudos experimentais. Em estudos experimentais as variáveis de interesse são identificadas e depois um ou mais fatores são controlados com intuito de se perceber como esses fatores influenciam as variáveis. Quando o estudo é observacional não há controle de fatores para in- fluenciar nas variáveis de interesse.

Após os dados serem coletados, muitas vezes são selecionados (de preferência aleatoria-

mente) subconjuntos desses dados e parte dessas variáveis para ser operada na descoberta de conhecimento.

2.1.2.2 Pré-processamento

Nesse passo, é realizada a limpeza dos dados selecionados para o estudo. Os ruídos devem ser descobertos e, caso seja julgado necessário, removidos. Deve-se ,também, decidir quais estratégias a serem adotadas, em caso de dados que estão faltando na base.

2.1.2.3 Transformação

O terceiro procedimento é encontrar maneiras de representar os dados de modo que facilite a análise . Após essa etapa, o número de variáveis levada em consideração pode diminuir e/ou o modo como cada uma é representada pode mudar.

2.1.2.4 Data mining

Consiste em aplicar as técnicas presentes em data mining para extrair conhecimento dos dados previamente coletado. Inicialmente, é realizada uma análise exploratória para extrair in- formações e permitir ao analista a formulação de hipóteses, decisão de quais algoritmos de mineração de dados utilizar para encontrar padrões e encontrar os parâmetros mais apropriados para esses algoritmos. Após a análise exploratória, será iniciada a busca por padrões em uma determinada forma de representação ou em um conjunto de representações (são exemplos de formas de representação: regras ou árvores de classificação, regressões e clustering). Encontrado os padrões, eles serão analisados e caso necessário se pode voltar para um dos passos anteriores.

2.1.2.5 Validação

O último passo no descobrimento de conhecimentos é verificar se os dados minerados são presentes em outras bases de dados, pois nem sempre todos os padrões encontrados são válidos. Esse processo é realizado da seguinte maneira: na etapa de coleta de dados, parte dos dados coletados serve para encontrar os padrões e a outra, disjunta, é escolhida para teste. São aplicados os padrões no conjunto de teste e a saída será comparada com o resultado desejado.

2.2 Dados simbólicos

Dados simbólicos são tipos de dados mais complexos do que dados clássicos, uma vez que possuem estrutura interna. Existem diversos tipos de dados simbólicos. Como exemplo pode-

se citar: os intervalares, os multivalorados (em uma mesma entrada tem mais de um valor) e os modais, que possuem frequências, probabilidades ou distribuições de peso acopladas em seus valores.

A representação de um objeto simbólico u em uma tabela é dada por um vetor $d_u = (\xi_{u1}, .., \xi_{up})$ onde u é o número de linhas da tabela e p é o número de variáveis. Mais genericamente, $d \in (D_1, .., D_p)$ no espaço $D = \times_{j=1}^{p} D_j$ onde D_j é o domínio da variável Y_j. É importante salientar que em uma tabela de dados simbólicos podem ter variáveis cujo domínio são dados clássicos, quando a variável é clássica ela é denotada por x_j e quando simbólica é ξ_j.

Uma das vantagens de se utilizar dados simbólicos é que por serem mais estruturados que dados clássicos, os dados simbólicos são capazes de representar mais informações em menos espaço. Outra vantagem é que os dados simbólicos possuem variação interna e são capazes de representar incertezas de cada observação, enquanto que os dados clássicos se limitam em ter variação entre observações.

A título de demonstração à assertiva acima consignada, exemplificamos com a medição da temperatura de uma determinada região. Com efeito, supondo-se que em um dia seja aferida a média de temperatura em 27 graus, enquanto que em outro dia a média tenha ficado em 30 graus, fica evidente que existe variação entre uma observação e outra. Caso seja levada em consideração outra variável, a temperatura da cidade ao longo do dia, presumindo-se que na cidade em questão a temperatura variou no intervalo [20,31] graus em um dia e no outro entre [22,36] graus, fica visível a presença de variação interna, uma vez que a temperatura em um determinado horário do dia pode estar em qualquer valor dentro do intervalo e existe variação entre as observações já que entre um dia e outro a temperatura pode variar em intervalos diferentes.

2.2.1 Origem

2.2.1.1 Inerentemente simbólicos

Alguns dados são inerentemente simbólicos, isto é, faz parte da natureza deles serem representados por estruturas de dados mais complexas. Um exemplo de dado inerentemente simbólico é o horário que o sol e a lua estão no céu, variável intervalar na qual os limites dos intervalos são representados pelos horários em que esses astros nascem e se põe.

Tabela 2.1 Tabela de horários e fases que a lua está presente no céu

Data	Horário presente no céu	Fase
04/05/2012	[15h49,03h34]	Lua Cheia
05/05/2012	[16h45,04h34]	Lua Cheia Grande
06/05/2012	[17h46, 05h37]	Lua Cheia
07/05/2012	[18h48, 06h41]	Minguante Giboso
08/05/2012	[19h51, 07h45]	Minguante Giboso
09/05/2012	[20h51, 08h45]	Minguante Giboso
...		

2.2.1.2 Agregados

Apesar de existirem dados naturalmente simbólicos, a maioria dos dados atuais é representada classicamente. Deste modo, caso não seja feita nenhuma transformação nesses dados se torna impossível de se trabalhar neles através da analise de dados simbólicos. Portanto, existe a necessidade de tornar os dados que atualmente são representados classicamente em dados simbólicos. Esse procedimento pode ser realizado através da agregação.

A primeira questão quanto à agregação, diz respeito a seguinte indagação - qual será o critério adotado para juntar as observações? Podem ser utilizadas técnicas de clustering para realizar o agrupamento, no entanto, não é obrigatório. Outra indagação que se faz presente é no tocante aquais variáveis dentre as disponíveis no dataset que serão utilizadas para agregar as demais. Julgando que usualmente um dataset contém muitas variáveis, caso não exista o conhecimento de um especialista ou se agrupar por uma determinada variável fizer parte do estudo o processo de seleção das variáveis agrupantes pode ser feito através de tentativa e erro.

Um dos indicadores que o agrupamento é bom é quando o número de dados simbólicos degenerados é muito pequeno. Dados simbólicos degenerados é fruto de um grupo que possui um único elemento e por conta disso não é possível gerar o dado simbólico. Um exemplo de dados agregados é o tamanho do diâmetro dos chapéus de espécies de cogumelo, note que quando tratamos de um elememto dentro de uma espécie de cogumelo podemos representar esses dados com tipos numéricos clássicos, mas quando tratamos de uma população inteira dessa espécie não existe como representar com um único valor numérico, podemos dizer que esse tamanho varia dentro de um intervalo.

2.2.2 Estatística descritiva para dados simbólicos

Apesar de inicialmente os métodos de mineração de dados terem sido elaborados sob os alicerces dos dados clássicos é possível adaptar seus conceitos e métodos para dados simbólicos. Uma ilustração disso é a adaptação da estatística descritiva para dados simbólicos do tipo intervalar, são exemplos de métodos da estatística descritiva a construção histograma, calculo da média , variância,etc. Antes de ser explicado como esses métodos foram adaptados é necessário definir o que é uma 'descrição individual ' e uma 'descrição virtual'.

Descrição individual é o valor de uma variável de um objeto simbólico. O cálculo da frequência de um histograma simbólico envolve contar o número de descrições individuais que tornam verdadeira uma determinada dependência logica nos dados. Uma dependência lógica pode ser representada pela equação 2.1, tal que $x \in X$ (X é o conjunto de todas descrições individuais presentes na tabela) e $A \subseteq D, B \subseteq D$. v retornará um valor binário, 0 se dependência lógica não é verdadeira para x ou 1 se for. [Billard and Diday, 2004]

$$v : [x \in A] \Rightarrow [x \in B] \tag{2.1}$$

Já a descrição virtual de um vetor d é um conjunto de todos elementos x presentes em d que satisfazem todas as dependências logicas em X. Ela é representada pela equação 2.2 como

$vir(d)$ sendo V_x todas regras presentes em X.

$$vir(d) = x \in D; v(x) = 1, \forall v \in V_x \tag{2.2}$$

Deste modo, supondo que há interesse em uma variável $Y_j \equiv Z$ e o valor observado para o objeto u nessa variável é um intervalo $Z(u) = [a_u, b_u]$, para $u \in E = \{1, ..., m\}$ e que os vetores de descrição individuais $x \in vir(d_u)$ são distribuídos uniformemente sobre o intervalo $Z(u)$, temos para cada ξ:

$$P\{x \leq \xi | x \in vir(d_u)\} = \begin{cases} 0, & \text{se } \xi \leq a_u; \\ \frac{\xi - a_u}{a_u - b_u}, & \text{se } a_u \leq \xi \leq b_u; \\ 1, & \text{caso contrário.} \end{cases} \tag{2.3}$$

O vetor de descrição individual x vai ter valores globalmente em $\bigcup_{u \in E} vir(d_u)$ e cada um desses objetos vai ter a mesma probabilidade de ser observado ($\frac{1}{m}$). Teremos, então, que a função empírica de distribuição, $F_Z(\xi)$, é uma função de distribuição de m distribuições uniformes $\{Z(u), u = 1, ..., m\}$. Portanto, da equação 2.3:

$$\begin{aligned} F_Z(\xi) &= \frac{1}{m} \sum_{u \in E} P\{x \leq \xi | x \in vir(d_u)\} \\ &= \frac{1}{m} \left\{ \sum_{\xi \in Z(u)} \left(\frac{\xi - a_u}{b_u - a_u} \right) + |u| \xi \geq b_u| \right\} \end{aligned} \tag{2.4}$$

Caso seja derivada a equação 2.4 em função de ξ será encontrada a função empírica de densidade de Z.

$$f(\xi) = \frac{1}{m} \sum_{u: \xi \in Z(u)} \frac{1}{b_u - a_u} \tag{2.5}$$

Como na equação 2.5 o somatório é apenas sobre objetos u para os quais $\xi \in Z(u)$ é possível escrevê-la de outra forma:

$$f(\xi) = \frac{1}{m} \sum_{u \in E} \frac{I_u(\xi)}{||Z(u)||}, \xi \in \Re \tag{2.6}$$

Onde $I_u(\xi)$ é uma função que indica se ξ está ou não em $Z(u)$, quando afirmativo retornará 1, caso contrário 0. E $||Z(u)||$ é o range do intervalo $Z(u)$.

Agora se torna possível encontrar a média para dados simbólicos intervalares, pois se sabe que a média empírica $\bar{Z}$ em termos da função de densidade empírica é:

$$\bar{Z} = \int_{-\infty}^{\infty} \xi f(\xi) d\xi$$

Substituindo da equação 2.6:

$$\begin{aligned}\bar{Z} &= \frac{1}{m}\sum_{u\in E}\int_{-\infty}^{\infty}\frac{I_u(\xi)}{||Z(u)||}\xi d\xi \\ &= \frac{1}{m}\sum_{u\in E}\frac{1}{b_u-a_u}\int_{\xi\in Z(u)}\xi d\xi \\ &= \frac{1}{2m}\sum_{u\in E}\frac{b_u^2-a_u^2}{b_u-a_u} \\ &= \frac{1}{m}\sum_{u\in E}\frac{b_u+a_u}{2}\end{aligned} \tag{2.7}$$

Se realizado o procedimento análogo da equação 2.7, pode-se encontrar que a variância amostral(S^2) é dada por:

$$S^2 = \frac{1}{3m}\sum_{u\in E}\left(b_u^2+b_ua_u+a_u^2\right) - \frac{1}{4m^2}\left[\sum_{u\in E}b_u+a_u\right]^2 \tag{2.8}$$

Para construir um histograma de dados simbólicos intervalares, é preciso tomar o intervalo $I = [min_{u\in E}a_u, mas+u\in Eb_u]$ em que todos possíveis valores de Z em X estão contidos, e particiona-los em r subintervalos $I_g = [\xi_{g-1},\xi_g), g = 1,...,r-1$ e $I_r = [\xi_{r-1},\xi_r]$. Então, o histograma para Z é a representação gráfica da distribuição de frequência $\left\{(I_g,p_g)\, g = 1,...,r\right\}$ onde:

$$p_g = \frac{1}{m}\sum_{u\in E}\frac{\left|\left|Z(u)\cap I_g\right|\right|}{||Z(u)||} \tag{2.9}$$

Nesse caso p_g representa a probabilidade de um descrição individual x estar no intervalo I_g

2.3 Estado da arte

A partir do final da década de 80, análise de dados simbólicos deixou de ser restrita a um pequeno grupo de pesquisadores para ser uma área de pesquisa bastante relevante marcada por muitas publicações e conferências. Pode-se dizer que o trabalho pioneiro de E. Diday através de papers que tratavam a respeito dos princípios básicos da analise de dados simbólicos [Diday, 1988, Diday, 1987, Diday, 1989, Diday, 1991] se deu inicio ao crescente interesse nessa área.

Posteriormente, houve muitos trabalhos cujos principais objetivos são estender as ferramentas estatísticas e de data mining para dados simbólicos, por exemplo:

- Estatística descritiva básica:Em [De Carvalho, 1995] é tratado a respeito de construção de histograma para distribuição de dados do tipo booleano, enquanto em [Billard and Diday, 2004] são abordados média, variância e histograma para dados do tipo intervalo e multivalorado. Nos trabalhos [Lauro and Gioia, 2006] e [Billard, 2004], são introduzidos métodos para analisar a interdependência e dependência entre variáveis com valores intervalares.

- Medidas de dissimilaridade e verossimilhança: Em [Bock and Diday, 2000],no capitulo 8 são estendidas as medidas de dissimilaridade da dados simbólicos.
  Em [Le-Rademacher and Billard, 2011] é proposta a função de verossimilhança para dados simbólicos,ilustrando sua aplicação ao se encontrar os estimadores de máxima verossimilhança da média e variância para distribuições de dados do tipo intervalo e histograma.

- Análise multidimensional de dados: Em [Gioia and Lauro, 2006] são adaptados os modelos matemáticos que são base de PCA para dados intervalares.

- Clustering: No artigo [de Vargas and Bedregal, 2011], é apresentada uma técnica de clustering fuzzy no qual a entrada e o grau de parentesco podem ser dados intervalares. Em [Bock, 2005], são realizadas otimizações de medidas de dissimilaridade e nos centros das classes integradas à um algoritmo de clustering para dados simbólicos.

- Séries temporais: Em [Arroyo et al., 2011] são analisados vários métodos de previsão para séries temporais de dados do tipo intervalo e histograma, são adaptados filtros de suavização e métodos não paramétricos como o K-NN.

- Visualização de dados: No artigo [Noirhomme-Fraiture, 2002], são propostos os Simple Star,Zoom Star, 3D star que são gráficos radiais para a visualização de objetos simbólicos e o Temporal Star cuja finalidade é visualizar objetos simbólicos variando com o tempo. Em [Irpino et al., 2003], o gráfico proposto para visualização de variáveis intervalares em um plano fatorial.

- etc.

O amadurecimento da área de análise de dados simbólicos culminou na criação de um framework denominado SODAS.

2.3.0.1 SODAS

O SODAS é um software suportado pela EUROSTAT (the statistical office of the European Union) e se encontra atualmente em sua segunda versão denominada SODAS2. Este programa suporta uma grande gama de métodos estatísticos para analisar dados simbólicos além de possuir ferramentas para administrar e manusear os dados. Uma análise de dados no SODAS2 é representada por uma cadeia de blocos, nos quais o primeiro representa a tabela de dados simbólicos e os demais são métodos estatísticos aplicados a ela. Esses blocos serão executados consecutivamente e a saída de um pode servir como a entrada do próximo. Após a execução dessa cadeia novos blocos são adicionados para expor os resultados. [de Baenst-Vandenbroucke and Lechevallier, 2008]

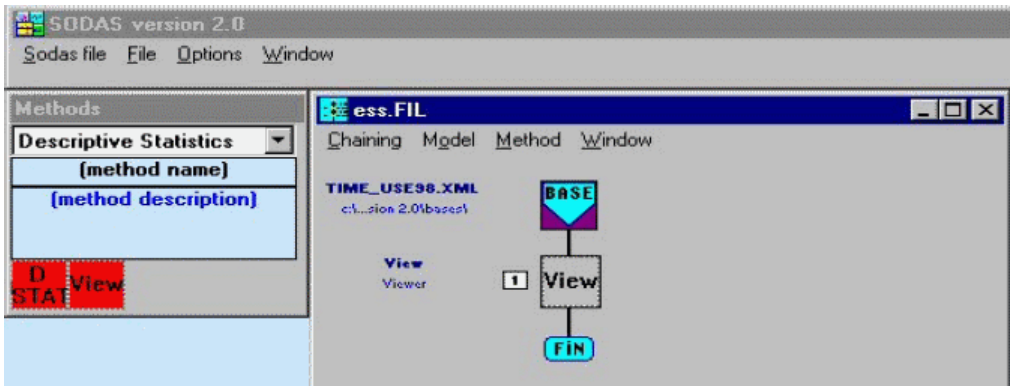

Figura 2.3 Análise de dados no SODAS2

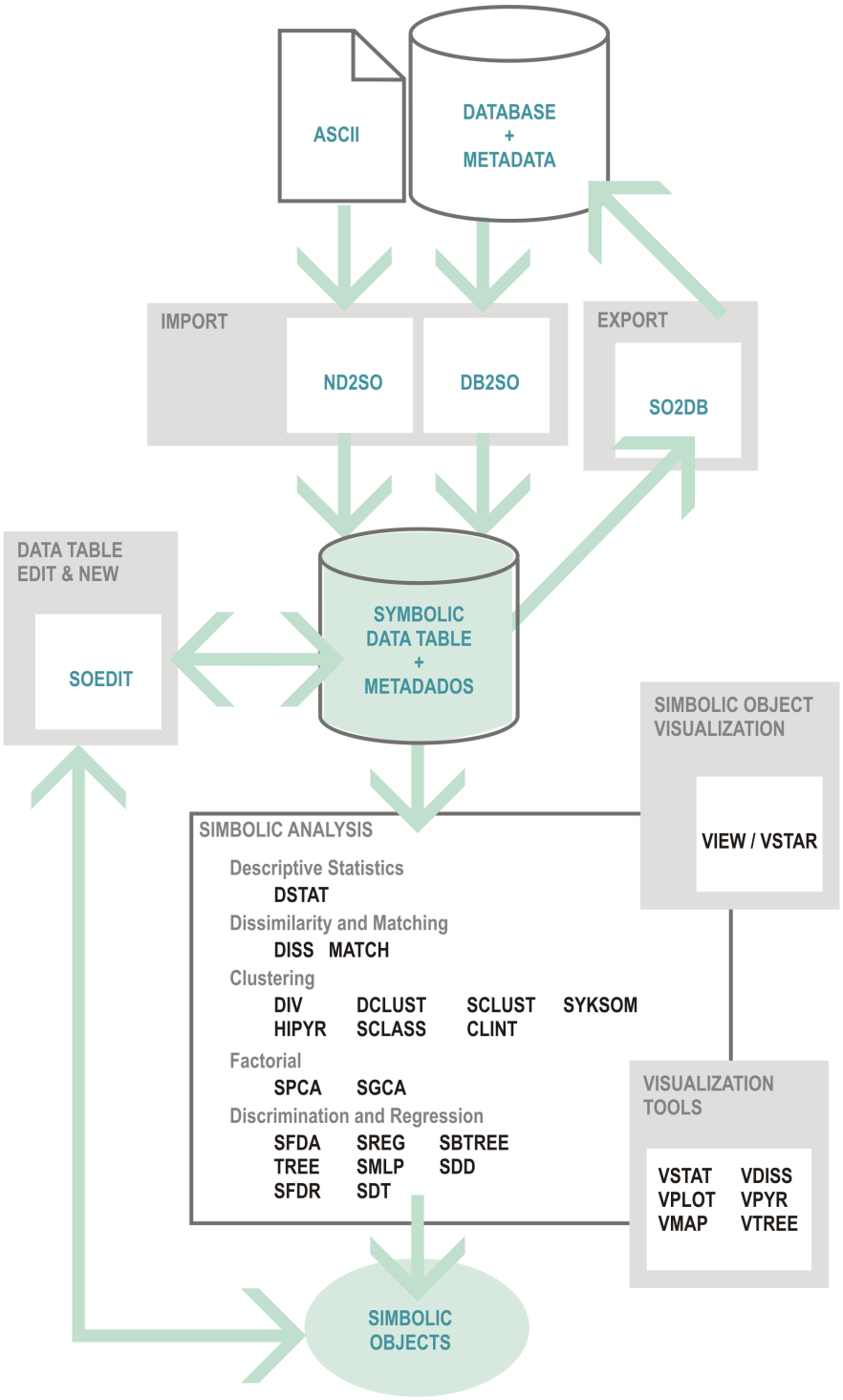

Figura 2.4 Diagrama de módulos do SODAS2

O SODAS2 é dividido em dois módulos principais, de administração dos dados e o de tratamento dos dados. O primeiro módulo é dedicado para a importação, exportação e edição de dados simbólicos e é subdividido em quatro submódulos (ND2SO,DB2SO,SO2DB,SOEDIT) como indica a figura x. O ND2SO e o DB2SO são dois módulos que servem para importar dados simbólicos, a diferença entre os dois é que o ND2SO serve para importar dados simbólicos, que já são simbólicos nativamente, enquanto o DB2SO recebe dados de uma base de dados e transforma em dados simbólicos. O SO2DB faz o caminho inverso do DB2SO, ele transforma dados simbólicos em dados clássicos e exporta para o banco de dados. O SOEDIT serve para editar dados simbólicos que já foram importados.

O segundo módulo é responsável para o tratamento dos dados, ele oferece um conjunto de 21 métodos para realização de análise de dados simbólicos, abrangendo estatística descritiva, análise de fatores, clustering , medidas de dissimiliaridade , discriminantes e regressão e visualização de dados.

CAPÍTULO 3

ISDA.R

Só há um princípio motor: a faculdade de desejar.
— ARISTÓTELES

3.1 Introdução

O pacote ISDA.R está disponível gratuitamente no repositório oficial da linguagem R-Cran (http://cran.r-project.org/web/packages/ISDA.R/index.html) contendo um conjunto de funções que auxiliam o analista de dados a trabalhar com dados intervalares. A ideia por trás desse pacote é agrupar o máximo de operações possíveis sobre o dado intervalar.

Atualmente, o pacote engloba funções para calcular medidas da estatística descritiva, transformações de dados clássicos para dados simbólicos e técnicas de visualização de dados.

3.2 Funções

Nesta seção serão apresentadas as funções presentes no pacote (Intervalo, Objeto Simbólico, Média, Variância, Histograma, Desvio padrão, Moda, Percentil, Resumo, Agregação, Gráfico Intervalar 3D, Gráfico Estrela), explicando para que servem e como funcionam.

3.2.1 Intervalo

A função interval retorna um objeto da classe também denominada de interval. Esse objeto é a estrutura básica do pacote, pois a maioria das funções receberão uma variável interval como parâmetro. É importante frisar que uma variável intervalar não possui obrigatoriamente um único intervalo, o que essa variável guarda é uma coluna de uma tabela que possui um ou mais intervalos. Foi importante definir essa classe por que caso fosse utilizado uma lista simples não seria possível se utilizar da função summary, que será explicada mais adiante.

A função recebe, então, os limites inferiores e superiores da coluna, que contém dados intervalares, e retorna uma variável do tipo intervalo

Para acessar os limites inferiores e superiores se utilizarão os operadores $minValue e $maxValue, respectivamente.

Algorithm 1: interval

Input: min,max
Output: interval

```
1 rval = list(minValue= min, maxValue = max);
2 class(rval)="interval";
3 return(rval);
```

3.2.2 Objeto Simbólico

Um objeto simbólico é uma linha da tabela que contém dados simbólicos. A função getSimbolicObject receberá uma tabela e o índice que o objeto se encontra, e retornará um objeto da classe simbolic object.

Algorithm 2: getSimbolicObject

Input: table,index
Output: simbolicObject

```
1  minr = table[index,seq(1,length(table[1,]),by=2)];
2  maxr = table[index,seq(2,length(table[1,]),by=2)];
3  headerNames = names(minr);
4  simbolicObject = NULL;
5  for i in 1:length(minr) do
6      headerNames[i]= sub("min",,headerNames[i]);
7      simbolicObject = rbind(valores,c(minr[i],maxr[i]));
8  colnames(simbolicObject)= c('min','max');
9  rownames(simbolicObject)=headerNames;
10 class(simbolicObject) = "simbolic object";
11 return(simbolicObject);
```

Na variável minr ficarão armazenados todos os limites inferiores das variáveis e na maxr os limites superiores. headerNames conterá o nome de cada variável. Então, da linha 4 à 7, armazenaremos em cada linha de simbolicObject, uma variável intervalar onde a primeira coluna será o limite inferior e a segunda o limite superior. Caso o nome das variáveis tenha o "min"para indicar que é o limite inferior, o min será retirado(linha 6). Posteriormente, os valores de headerNames serão atribuídos aos nomes das linhas e "min","max"as colunas.

3.2.3 Média

A média é uma medida de posição, ela representa onde se concentram os valores de uma distribuição. Essa função é uma implementação da fórmula 2.7, desse modo a função vai receber uma coluna da tabela (variável tipo interval) e retornará a média dessa variável.

Algorithm 3: meanInterval

Input: interval
Output: mean

```
1 m= length(interval$minValue);
2 mean = (sum(interval$minValue)+ sum(interval$maxValue))/(2*m);
3 return(mean);
```

3.2.4 Variância

A variância é uma medida de dispersão que indica o quanto em média os valores serão diferentes do valor esperado, desse modo quanto maior a for a distância das observações para a média maior será a variância [Bussab and Morettin, 2010]. A função é uma implementação da fórmula2.8, ela recebe uma variável do tipo interval e retorna a variância dessa variável.

Algorithm 4: varianceInterval

Input: interval
Output: variance

```
1  xmin = interval$minValue;
2  xmax = interval$maxValue;
3  nLines= length(xmin);
4  term1temp =0;
5  term2temp =0;
6  for i in 1:m do
7      term1temp = term1temp + ((xmax[i]^2) +(xmin[i]*xmax[i])+(xmin[i]^2));
8      term2temp = term2temp+ (xmin[i]+xmax[i]);
9  term1temp = (1/(3*nLines))*term1temp;
10 term2temp = (1/(4*( nLines ^2)))*(term2temp^2);
11 variance = term1temp-term2temp);
12 return(variance);
```

Basicamente, nas linhas 1 a 3 são armazenados os limites inferiores e superiores dos intervalos e é atribuída a nLines o número de observações contidas na variável interval. Entre a linha 4 e 10 são calculados os dois termos da formula para, na linha 11, subrair o termo 2 do termo 1 de modo a encontrar a variância.

3.2.5 Histograma

O histograma é uma ferramenta de análise exploratória de dados que permite ao usuário maiores informações a respeito da distribuição dos dados. Esse gráfico é constituído por retângulos justapostos ao qual denominamos de classe. A base de cada retângulo é o limite da classe, enquanto a altura é a frequência da classe.

Existem várias abordagens para se definir o número de classes de um histograma, dentre elas podemos citar as fórmulas de Scott e Sturges.

A construção do histograma é decomposta em três funções menores: getBreaks, calcPro-

bInt, buildHist. A primeira função serve para encontrar os limites inferiores e superiores das classes, a segunda é utilizada para calcular a frequência associada a cada classe delimitada anteriormente e a terceira é uma função cuja utilidade é adaptar a saída da segunda função para a construção do histograma através da função nativa do R "hist". Essa decomposição é importante, uma vez que outras funções vão utilizar as delimitações e as frequências das classes.

A função histInterval receberá uma variável interval e o tipo de algoritmo que será utilizado para definir o número de classes sendo os possíveis parâmetros: "ST"para Sturges, "SC"para Scott e "OTHER"se o usuário quiser definir o número de classes manualmente. Caso seja passado como paramêtro "OTHER", o usuário deverá passar também o número de classes desejado. Além desses parâmetros, é de se mencionar o denominado "T", o qual utiliza um método que gera diretamente as classes sem decidir o número de classes a priori.

Algorithm 5: histInterval

Input: interval,type,nclasses
Output: hist

```
1 xmin = interval$minValue;
2 xmax = interval$maxValue;
3 bks= getBreaks(xmin,xmax,type);
4 prob = calcProbInt(bks);
5 return(buildHist(prob,bks)) ;
```

3.2.5.1 Classes

Essa função é utilizada para encontrar os limites de cada classe do histograma, quando é passado o parâmetro type como "SC"ou "ST"ele vai calcular o número de classes usando uma dessas duas fórmulas (Sturges ou Scott), caso passado "OTHER"o número de classes vai ser passado como um parâmetro. A função, então,vai calcular o range das classes subtraindo o maior dos limites superiores pelo menor dos limites inferiores e dividir o valor resultante pelo número de classes. Então, a partir do menor valor do limite inferior será adicionado o valor do range das classes para se encontrar os limites inferiores e superiores de cada classe (e.g. classe1[minValue, minValue+classRange], classe2[minValue + classRange, minValue + 2 * classRange] ... classN[minValue + (n-1) classRange + minValue+n * classRange].

Quando o parâmetro type é "T", será utilizado cada limite inferior e superior como limites das novas classes. Caso existam dois intervalos, por exemplo, intervalo A [2,4] e intervalo B [3,5] as classes resultantes desse algoritmo seriam classe A[2,3], classe B[3,4] e classe C [4,5].

Algorithm 6: getBreaks

```
   Input: xmin,xmax,type,nclasses
   Output: breaks
1  breaks = NULL;
2  if type=='T' then
3      numbers = c(xmin,xmax);
4      temp = table(numbers);
5      name = names(temp);
6      breaks = sort(as.numeric(name));
7  else
8      nclasses = nclasses;
9      if type='ST' then
10         nclasses = ceiling(log2(numberLines)+1);
11     else if type=='SC' then
12         nclasses = (3.5* sdInterval(interval))/(numberLines^(1/3));
13     minValue = min(xmin);
14     maxValue = max(xmax);
15     range = maxValue - minValue;
16     classRange = range/nclasses;
17     breaks = minValue;
18     for i in 1:nclasses do
19         if i != nclasses then
20             breaks= c(breaks,breaks[i]+classRange);
21         else
22             breaks = c(breaks,maxValue);
23 return(breaks)
```

As linhas 2 a 6 representam a porção de código que calcula os limites quando o usuário utilizar o tipo "T"como método para definir as classes. Ele simplesmente concatena os limites inferiores com os superiores, remove os valores repetidos e organiza os valores do vetor resultante em ordem crescente.

O restante do código é utilizado quando o usuário deseja usar a regra de Scott,Sturges ou passar o número de classes como parâmetro. Nesse caso, a função encontrará o range das classes (linha 8 à 11) e iterativamente definirá as classes no restante do código.

3.2.5.2 Probabilidades

Dados os limites das classes e os intervalos presentes na variável interval, denomina-se calcProbInt o método utilizado para calcular a altura de cada retângulo do histograma.

Algorithm 7: calcProbInt

```
   Input: xmin,xmax,bks
   Output: vectResBks
1  vectResBks = NULL;
2  value = 0;
3  lenBks= length(bks);
4  numberLines= length(xmin) ;
5  for i in 1:(lenBks-1) do
6      for j in 1:numberLines do
7          rangeInterval = xmax[j]- xmin[j];
8          intersectionRange =NULL ;
9          if (xmin[j]<=bks[i] and xmax[j]>=bks[i+1]) then
10             intersectionRange= bks[i+1]-bks[i];
11         else if xmin[j]>=bks[i] and xmin[j]<bks[i+1] and xmax[j]>=bks[i+1] then
12             intersectionRange = bks[i+1] - xmin[j];
13         else if xmin[j]<=bks[i] and xmax[j]>bks[i] and xmax[j]<=bks[i+1] then
14             intersectionRange = xmax[j] - bks[i];
15         else if (xmin[j]>=bks[i] and xmax[j]<=bks[i+1]) then
16             intersectionRange = rangeInterval
17         Value = value + intersectionRange/ rangeInterval * numberLines
18     vectResBks = c(vectResBks,value);
19     value = 0;
20 return(vectResBks)
```

Nessa função será iterado por todos os intervalos para calcular a probabilidade deles estarem em cada uma das classes, como pode ser percebidos nas linhas 5 e 6. O primeiro "for"serve para percorrer todas as classes, já o segundo "for", que está dentro do primeiro, vai iterar por todos os intervalos contabilizando probabilidade de cada intervalo estar na classe em questão.

Baseando-se na fórmula 2.9, a probabilidade de um intervalo estar em uma classe é dada pela divisão da intersecção do intervalo com a classe sobre o tamanho do intervalo, isso tudo dividido pelo número de observações. A frequência de cada classe é, portanto, o somatório das probabilidades de cada intervalo estar na classe.

Há quatro condições em que existe intersecção entre a classe e o intervalo, a primeira (linha6) se refere a situação em que a classe está contida no intervalo, como mostrado na figura abaixo.

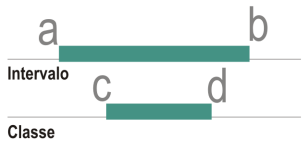

Figura 3.1 Classe contida no intervalo

Nesse caso, a contribuição do intervalo para a frequência dessa classe se dá dividindo o range da classe pelo tamanho do intervalo e número de observações.

A segunda condição acontece quando tanto o limite superior quanto o inferior do intervalo é maior que os da classe, sem que o limite inferior do intervalo seja maior que o superior da classe. Deste modo, a intersecção será notoriamente o intervalo representado por [limite inferior do intervalo, limite superior da classe].

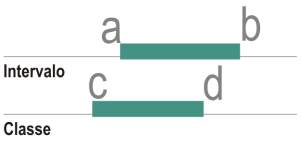

Figura 3.2 Classe à esquerda do intervalo

O terceiro caso é o inverso do segundo, os limites superior e inferior do intervalo são menores que os limites da classe, no entanto, o limite superior do intervalo deve ser maior que o limite inferior da classe. O range da intersecção será, portanto, a subtração do limite superior do intervalo pelo limite inferior da classe.

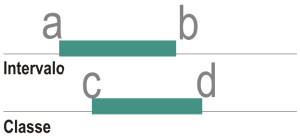

Figura 3.3 Classe à direita do intervalo

Finalmente, o quarto caso acontece quando o intervalo está contido na classe, desse modo, o próprio intervalo representa a intersecção dele com a classe.

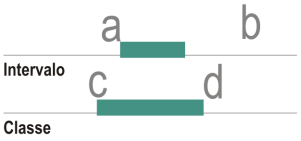

Figura 3.4 Intervalo contido na classe

3.2.5.3 Construção do histograma

O ambiente gráfico do R, incluindo a função para desenhar histogramas, não foi projetado inicialmente para suportar dados simbólicos. Por conta disso foi necessário adaptar a saída de dados da função calcProbInt para servir como entrada para a função hist do R.

Algorithm 8: buildHist

Input: prob,bks
Output: histogram

```
1 prob = round(x*100);
2 vecImpHist <- NULL;
3 for i in 1:(length(bks)-1) do
4     val = (bks[i]+bks[i+1])/2;
5     temp = rep(val, prob[i]);
6     vecImpHist = c(vecImpHist,temp);
7 xlimi=c(min(bks),max(bks));
8 ylimi=c(0,ceiling(max(prob)));
9 hist(vecImpHist,breaks = bks, xlim=xlimi, ylim = ylimi, ylab = "(%)");
```

A função hist recebe um vetor de números e calcula a frequência de cada um para desenhar o histograma.Além disso, o parâmetro breaks é um vetor numérico contendo os limites das classes e xlim e ylim são os limites do eixo x e y, respectivamente.

Para que a função hist funcionasse para intervalos, o vetor de probabilidades retornado pela função calcProbInt deveria conter valores que, quando calculados pela função hist, refletisse a probabilidade de cada classe. Desse modo, para cada classe do histograma foi realizada a média aritmética entre os valores de seus limites e esse valor foi repetido tantas vezes quanto for à probabilidade associado a essa classe (isto é, caso a classe 1 tenha probabilidade de 45%, o valor será repetido 45 vezes).

Os limites do histograma serão para o eixo x o menor e maior valor dos breaks e do eixo y de 0 ao maior valor retornado pela função calcProbInt.

3.2.6 Desvio padrão

Assim como a variância, o desvio padrão é uma medida de dispersão que indica o quanto em média os valores serão diferentes do valor esperado. O desvio padrão é a raiz quadrada da variância, desse modo, a função sdInterval vai receber uma variável interval e retornar o desvio padrão dessa variável.

Algorithm 9: sdInterval

Input: interval
Output: sd

```
1 sd =varianceInterval(interval)^(1/2);
2 return(sd);
```

3.2.7 Moda

A moda é a realização mais frequente dos conjuntos de valores observados [Bussab and Morettin, 2010], desse modo, para encontrar a moda foi utilizada a distribuição do histograma, sendo a classe de maior frequência a moda da variável. Portanto, será passado como parâmetro da função o intervalo e o tipo de algoritmo que será utilizado para calcular o número de classes e será retornado um intervalo que será a moda.

Algorithm 10: modeInterval

Input: interval,type
Output: mode

```
1  xmin = interval$minValue;
2  xmax = interval$maxValue;
3  bks = getBreaks(xmin,xmax,type);
4  prob= calcProbInt(xmin,xmax,bks);
5  maxFreq= max(prob);
6  mode = NULL;
7  for i in 1:length(prob) do
8      if prob[i]== maxFreq then
9          mode = interval(bks[i],bks[i+1]);
10 return(mode);
```

3.2.8 Percentil

Percentil é o valor em que p por cento dos dados estão abaixo [Bussab and Morettin, 2010]. Será utilizada, então, a distribuição de frequência encontrada no histograma para calcular esse valor. Devido à utilização do histograma deverá ser passado como parâmetro o tipo de algoritmo de definição de número de classes, além disso deve ser passado a variável interval e o valor de p.

Algorithm 11: percentileInterval

Input: interval,percentil,type
Output: percentile

```
1  xmin = interval$minValue;
2  xmax = interval$maxValue;
3  bks = getBreaks(xmin,xmax,type);
4  prob= calcProbInt(xmin,xmax,bks);
5  acumulatedDist = prob;
6  for i in 1:(length(prob)-1) do
7      acumulatedDist[i+1] = acumulatedDist[i+1] + acumulatedDist[i] ;
8  count = 1;
9  while acumulatedDist[count]<percentil and count<length(acumulatedDist) do
10     count= count+1;
11 infLimResul = bks[count-1];
12 supLimResul =bks[count];
13 howMuchLess =percentil-acumulatedDist[count-1];
14 result = howMuchLess*(supLimResul- infLimResul])/prob[count];
15 result = result+ bks[count-1];
16 return(result);
```

Nessa função é transformada a distribuição do histograma em uma distribuição acumulada com o intuito de encontrar a classe em que o valor do percentil está contido. Depois de encontrada a classe, será realizada uma regra de três: se nos limites da classe a probabilidade é x, do limite inferior até o valor que queremos encontrar é y, onde o y é o p menos a probabilidade acumulada da classe anterior.

3.2.9 Resumo

A função summary retorna um resumo da variável intervalar passada como parâmetro contendo a média, variância, o desvio padrão e mediana.

Algorithm 12: summary.interval

Input: interval
Output: summary

```
1 mean = meanInterval(interval);
2 variance = varianceInterval(interval);
3 standardDeviance = sdInterval(interval);
4 median = percentileInterval(interval, 0.5, "T");
5 summary = data.frame(mean,variance,standardDeviance,median);
6 class(object) = "summary.interval";
7 return(summary);
```

3.2.10 Agregação

Essa função tem como finalidade agrupar dados clássicos em dados simbólicos. Para agrupar as variáveis de modo a criar dados simbólicos do tipo intervalar é necessário indicar quais variáveis serão agrupadas e quais serão utilizadas para agrupar. Usualmente, tabelas de dados possuem diversas variáveis e caberá ao analista escolher quais serão responsáveis por esses dois papéis. Esta função torna possível ao analista retirar as colunas das variáveis que não interessam, bastando, assim, passar quais variáveis serão utilizadas para agrupar e quais serão agrupadas.

Neste trabalho, a função foi dividida em três, objetivando, assim, tornar mais fácil o entendimento, no entanto, no pacote, tais divisões não existem. No caso, a primeira parte serve para encontrar a tabela que se esta trabalhando, a segunda (findAgroupments) serve para encontrar os grupos e a terceira (group) agrupa de acordo com o tipo de agrupamento passado como parâmetro ("M"para máximo e mínimo, "Q"para distância interquartílica e "P"para a distância do 10-percentil ao 90-percentil).

Algorithm 13: groupVariables

```
   Input: aggregative,aggregated,dataAdress,aggregationType
   Output: matrizResult
 1 data =read.table(dataAdress,header = TRUE);
 2 names = names(data);
 3 variables = c(aggregative,aggregated);
 4 matriz = data[match(variables,names)];
 5 matriz = cbind(matriz,-1);
 6 numLines = dim(matriz)[1];
 7 numColumns = dim(matriz)[2];
 8 matrizResult= NULL;
 9 numAggregative= length(aggregative);
10 numAggregated=length(aggregated);
11 for i in 1:numLines do
12     if matriz[i,numColunasOriginal] == -1 then
13         values =
           findAgroupments(matriz,numAggregated,numAggregative,numColumns);
14         matrizResult=group(values,matrizResult,aggregationType);
15 return(matrizResult)
```

Na linha 2, a função names vai retornar os Headers da tabela para que na linha 4 sejam comparados com os nomes das variáveis de interesse e sejam retornadas essas colunas. Será concatenada uma coluna, cujos valores serão -1 com intuito de marcar as linhas que ainda não foram analisadas em findAgroupments.

3.2.10.1 Encontrar agrupamentos

Algorithm 14: findAgroupments

```
   Input: matriz,numAggregated,numAggregative,numColumns
   Output: values
 1 actualLine = matriz[i,];
 2 values = matrix(nrow=1,ncol = numAggregated);
 3 for w in 1: length(numAggregated) do
 4     values[w] = matriz[i,(w+numAggregative)];
 5 for k in (i+1):numLines do
 6     itsEqual = TRUE;
 7     j = 1;
 8     if matriz[k,numColumns] == -1 then
 9         while (j <=(numAggregative)) and (itsEqual == TRUE) do
10             if actualLine[j] == matriz[k,j] then
11                 if j==(numAggregative) then
12                     newLine = NULL;
13                     for w in 1: length(numAggregated) do
14                         newLine = c(newLine,matriz[k,(w+numAggregative)]);
15                     values=rbind(values,newLine);
16                     matriz[k,numColumns] = 1;
17             else
18                 itsEqual= FALSE;
19             j= j+1;
20 return(values);
```

Da linha 1 a 4, é encontrado o padrão a ser comparado, sendo adicionados os valores das variáveis agrupadas na variável values. O restante do código faz a checagem linha por linha para ver se as variáveis que serão utilizadas para agrupar são iguais as da linha atual, note que se a última coluna da variável matriz for igual a 1 ela será pulada porque já faz parte de outro agrupamento. Caso as variáveis de agrupamento tenham valores semelhantes aos da linha atual, essa linha será marcada como pertencente a um agrupamento (linha 16) e os valores das variáveis agrupadas será concatenado como uma nova linha para a variável values (linha 15).

3.2.10.2 Agrupar

Algorithm 15: group

```
Input: values,matrizResult,aggregationType
Output: matrizResult
lineResult = NULL;
if aggregationType == 'M' then
    for h in 1:numAggregated do
        lineResult = c(lineResult,min(values[,h]));
        lineResult = c(lineResult,max(values[,h]));
else if aggregationType == 'P' then
    for (h in 1:numAggregated)  temp=quantile(values[,h],c(0.1,0.9));
    lineResult = c(lineResult,temp[1]);
    lineResult = c(lineResult,temp[2]);
else if aggregationType =="Q" then
    for (h in 1:numAggregated)  temp=quantile(values[,h],c(0.25,0.75));
    lineResult = c(lineResult,temp[1]);
    lineResult = c(lineResult,temp[2]);
matrizResult =rbind( matrizResult,lineResult);
return(matrizResult);
```

Basicamente, nessa função cada agrupamento utiliza o método de agrupamento escolhido para definir os limites inferiores e superiores do dado intervalar e concatenar à matrizResult as variáveis agrupadas, já transformadas em dados simbólicos.

3.2.11 Gráfico Intervalar 3D

Essa função é responsável por imprimir um gráfico tridimensional de variáveis intervalares, permitindo ao analista a visualização de como três variáveis se distribuem no espaço. São passadas três variáveis interval como parâmetro, e a função intervalGraph3D vai desenhar o relacionamento dessas variáveis. Nesse gráfico, cada variável vai estar associada a um eixo e a junção de duas dessas variáveis será um subespaço de$\Re^3$. Por exemplo, dados dois intervalos Y [3,10] e Z[5,8], a representação dessas variáveis no gráfico será o subespaço cujos valores do eixo Y variam entre 3 e 10 e os valores do eixo Z entre 5 e 8 como mostra na figura 3.5, onde o interior das linhas pontilhadas paralelas ao eixo X é o subespaço que representa a relação entre Y e Z. Nessa mesma linha de raciocínio, caso seja adicionada mais uma variável, se formará no espaço um paralelepípedo que representará o relacionamento entre essas três variáveis. O parâmetro yScale serve para definir qual é a escala do eixo y sobre o eixo x.

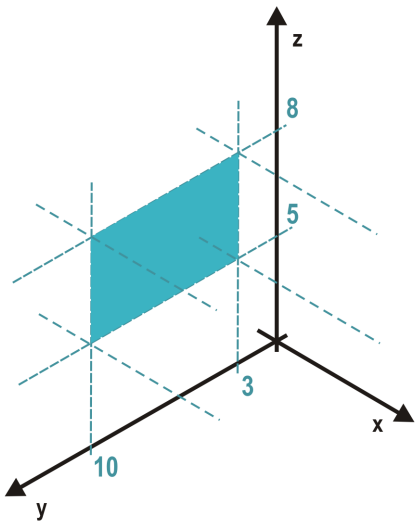

Figura 3.5 Relação entre duas variaveis intervalares

Algorithm 16: intervalGraph3D

Input: xIntervals,yIntervals,zIntervals,yScale
Output: intervalGraph3D

```
1 xmin = xIntervals$minValue;
2 xmax = xIntervals$maxValue;
3 ymin = yIntervals$minValue;
4 ymax = yIntervals$maxValue;
5 zmin = zIntervals$minValue;
6 zmax = zIntervals$maxValue;
7 rr =
  scatterplot3d((min(xmin):max(xmax)),(min(ymin):max(ymax),(min(zmin):max(zmax),
  color = "transparent", box = FALSE, angle = 24, xlim = c((min(xmin),max(xmax)), ylim
  = c(min(ymin), max(ymax)), zlim = c(min(zmin),max(zmax)),scale.y = yScale,grid =
  FALSE);
8 for i in 1:length(xmin) do
9     createsParallelepiped(i);
```

Na linha 7, scatterplot3d servirá como o gráfico base em que os paralelepípedos serão plotados. Ele requer que sejam passados três vetores com as coordenadas do x, y e z para plotar no gráfico, no entanto, não há interesse nesses pontos e, portanto, foi passado como parâmetro para que esses pontos sejam transparentes. A verdadeira finalidade do scatterplot3d é reaproveitar a perspectiva e as ferramentas de "plotagem"que são oferecidas pela função. Além disso, nele será definida a escala entre o eixo y e o x.

3.2.11.1 Desenha paralelepípedos

createsParallelepiped é uma função que está enclausurada dentro de intervalGraph3D, isto é, ela é capaz de acessar as variáveis definidas em intervalGraph3D. Sua finalidade é desenhar os paralelepípedos que representam as relações de três variáveis no espaço.

Algorithm 17: createsParallelepiped

Input: numLine ,...
Output:

```
1 parallelepiped <- rbind(c(xmin[numLine],ymin[numLine],zmax[numLine]),
  c(xmin[numLine],ymin[numLine],zmin[numLine]),
  c(xmax[numLine],ymin[numLine],zmin[numLine]),
  c(xmax[numLine],ymax[numLine],zmin[numLine]),
  c(xmax[numLine],ymax[numLine],zmax[numLine]),
  c(xmin[numLine],ymax[numLine],zmax[numLine]),
  c(xmax[numLine],ymin[numLine],zmax[numLine]),
  c(xmin[numLine],ymax[numLine],zmin[numLine]));
2 rr$points3d(parallelepiped[c(1:6,1,7,3,7,5) ,], type = 'l', lty = 1);
3 rr$points3d(parallelepiped[c(2,8,4,8,6), ], type = 'l', lty = 1);
```

Ele armazena os vértices do paralelepípedo na variável parallelepiped e nas duas linhas seguintes são desenhados os vértices e arestas do paralelepípedo.

3.2.12 Gráfico Estrela

O gráfico estrela é utilizado para representar um objeto simbólico, isto é, uma linha de uma tabela que contém dados simbólicos. É um gráfico radial em que cada eixo representa uma variável do objeto. Dentro de cada eixo haverá um intervalo e a ligação entre eles representa o objeto. Nessa função, será passado como parâmetro o objeto simbólico, o título e o subtítulo do gráfico se o usuário desejar, e as opções de texto (texts,textstextsCol,textSize), para que sejam impressos os textos dos limites dos intervalos, definindo-se, assim, o tamanho e a cor. Também é passado como parâmetro um booleano (linked), visando indicar se os limites dos intervalos serão ligados ou não, juntamente com a cor das ligações (linkedCol).

Algorithm 18: starGraph

Input: simbolicObject,main,sub,texts,textSize,textsCol,linked,linkedCol
Output: starGraphic

```
1  xmin = simbolicObject[,1] xmax = simbolicObject[,2] lenInterval = length(xmin) angles
   = 360/lenInterval;
2  xminChanged = NULL;
3  xmaxChanged = NULL;
4  if min(xmin)<0 then
5      xminChanged = xmin-min(xmin);
6      xmaxChanged = xmax-min(xmin);
7  else
8      xminChanged = xmin;
9      xmaxChanged = xmax;
10 plot(2,3,xlim=c(-max(xmaxChanged),max(xmaxChanged)),ylim=c(-
   max(xmaxChanged),max(xmaxChanged)),type='n',main=main,sub=sub,xaxt="n",yaxt="n");

11 distancePP <-function(x,ang) x/sqrt(1+(tan(i*angles*pi/180)^2));
12 linkInfX = NULL;
13 linkInfY = NULL;
14 linkSupX = NULL;
15 linkSupY = NULL;
16 for i in 1:lenInterval do
17     x= c(distancePP(xminChanged[i],i*angles),distancePP(xmaxChanged[i],i*angles));
18     y= x*tan(i*angles*pi/180);
19     if i*angles>90 and i*angles<270 then
20         x= -x;
21         y=-y;
22     points(x,y,type='l');
23     linkInfX =c(linkInfX,x[1]);
24     linkInfY = c(linkInfY,y[1]);
25     linkSupX = c(linkSupX,x[2]);
26     linkSupY = c(linkSupY,y[2]);
27     if texts then
28         text(x,y,labels=c(xmin[i],xmax[i]),col='blue',cex=textSize);
29 if linked then
30     points(c(linkInfX,linkInfX[1]),c(linkInfY,linkInfY[1]),type='l',col=linkedCol);
31     points(c(linkSupX,linkSupX[1]),c(linkSupY,linkSupY[1]),type='l',col=linkedCol);
```

CAPÍTULO 4

Exemplo prático do uso de ISDA.R

4.1 Introdução

Nesse capítulo é dada uma demonstração da utilização do pacote, para isso será utilizada a base de dados Abalone, disponibilizada em [Warwick J Nash and Ford, 1994]. Essa base contém dados acerca de um tipo de cogumelo (espécies do genêro Haliotis) contendo 4177 observações e 9 váriáveis: Sex, Length, Diameter, Height, WholeWeight, ShuckedWeight, VisceraWeight,ShellWeight,Rings.

4.2 Instalando ISDA.R

Antes de iniciar a utilização de ISDA.R é necessário instalar o pacote, como ele foi submetido e aceito no repositório do R, o processo de instalação é bastante simples, ele consiste em baixar do repositório e carregar o pacote no R. Para isso, devem ser utilizados os seguintes passos:

1. Ao abrir o RGui.exe, clicar em ”Pacotes” e quando o menu abrir ”Instalar pacote(s)”

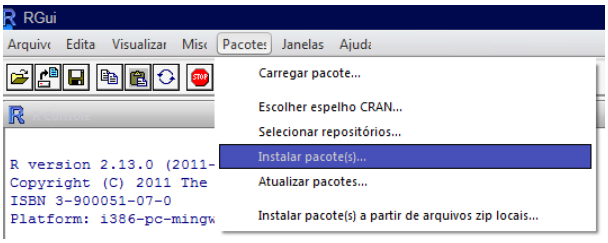

Figura 4.1 Instalando o pacote: passo 1

2. Selecionar um dos servidores da lista

Figura 4.2 Instalando o pacote: passo 2

3. Selecionar o pacote ”ISDA.R” e clicar ”Ok”

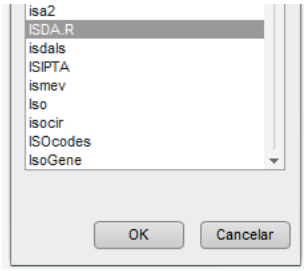

Figura 4.3 Instalando o pacote: passo 3

4. Para concluir basta carregar inserindo o comando library(ISDA.R)

```
> library(ISDA.R)
Carregando pacotes exigidos: scatterplot3d
Mensagens de aviso perdidas:
pacote 'ISDA.R' foi compilado na versão do R 2.13.2
```

Figura 4.4 Instalando o pacote: passo 4

Concluídos esses quatro passos o ambiente já está configurado para se utilizar o ISDA.R

4.3 Utilizando ISDA.R

Como as variáveis da base de dados Abalone são do tipo clássico, para se iniciar o trabalho é necessário transformá-las em variáveis intervalares. Para isso será utilizada a função group-Variables. Para agrupar será utilizada a váriavel Rings, que é o número de anéis e se somada de 3.5 representa a idade do cogumelo. Pode-se utilizar os valores mínimos e máximos, distância interquartilica ou distância percentilica para definir os limites dos agrupamentos, a diferença é

que os dois últimos são menos sensíveis a outliers. Isso pode ser percebido no seguinte exemplo:

Algorithm 19: Agrupando e comparando objetos simbólicos

```
1 variaveisAgrupantes = ”Rings”;
2 variaveisAgrupadas = c(”Length”, ”Diameter”, ”Height”, ”WholeWeight”,
  ”ShuckedWeight”, ”VisceraWeight”, ”’ShellWeight”);
3 agrupamentoPercentil =
  groupVariables(variaveisAgrupantes,variaveisAgrupadas,”abalone.csv”,”P”);
4 agrupamentoMinimosMaximos=
  groupVariables(variaveisAgrupantes,variaveisAgrupadas,”abalone.csv”,”M”);
5 objetoSimbPercentil = getSimbolicObject(agrupamentoPercentil,5);
6 objetoSimbMinimosMaximos = getSimbolicObject(agrupamentoMinimosMaximos,5);
7 starGraphic(objetoSimbPercentil,linked=TRUE,texts=TRUE);
8 starGraphic(objetoSimbMinimosMaximos,linked=TRUE,texts=TRUE);
```

```
                   min    max
Length          0.2550 0.7200
Diameter        0.1950 0.5650
Height          0.0000 1.1300
Whole_weight    0.0800 1.7100
Shucked_weight 0.0315 0.8255
Viscera_weight 0.0135 0.3855
Shell_weight    0.0270 0.4700
attr(,"class")
[1] "Simbolic Object"
```

```
                    min     max
Length          0.39000 0.60000
Diameter        0.30000 0.46500
Height          0.09500 0.15650
Whole_weight    0.27970 1.06315
Shucked_weight 0.12105 0.49480
Viscera_weight 0.05935 0.23200
Shell_weight    0.08535 0.27675
attr(,"class")
[1] "Simbolic Object"
```

Figura 4.5 Objeto simbólico gerado por getSimbolicObject (E) máximo e mínimo, (D) distância do 10 percentil e 90 percentil

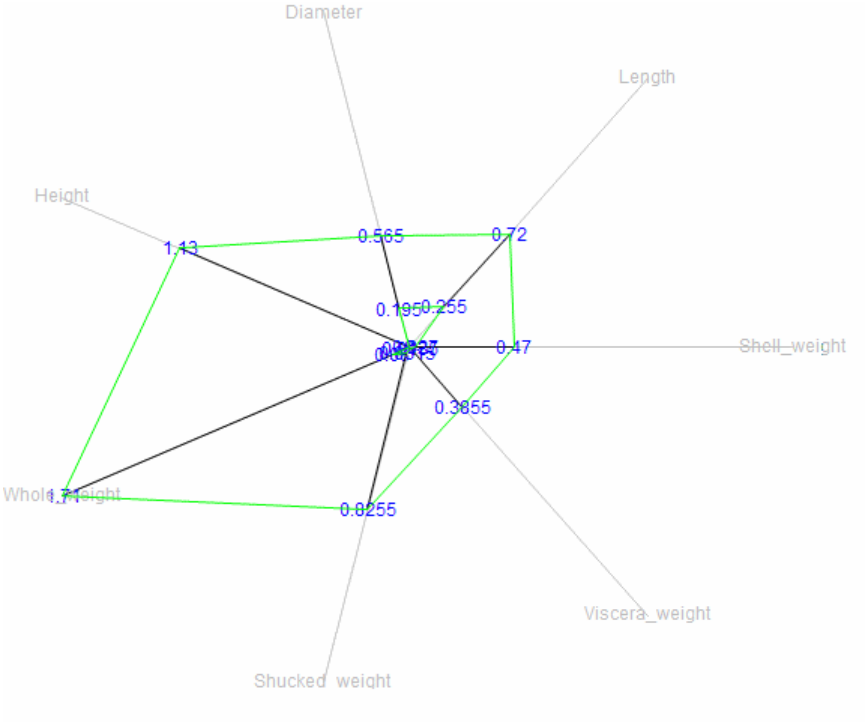

Figura 4.6 Objeto Simbólico gerado através de Máximo e Mínimo

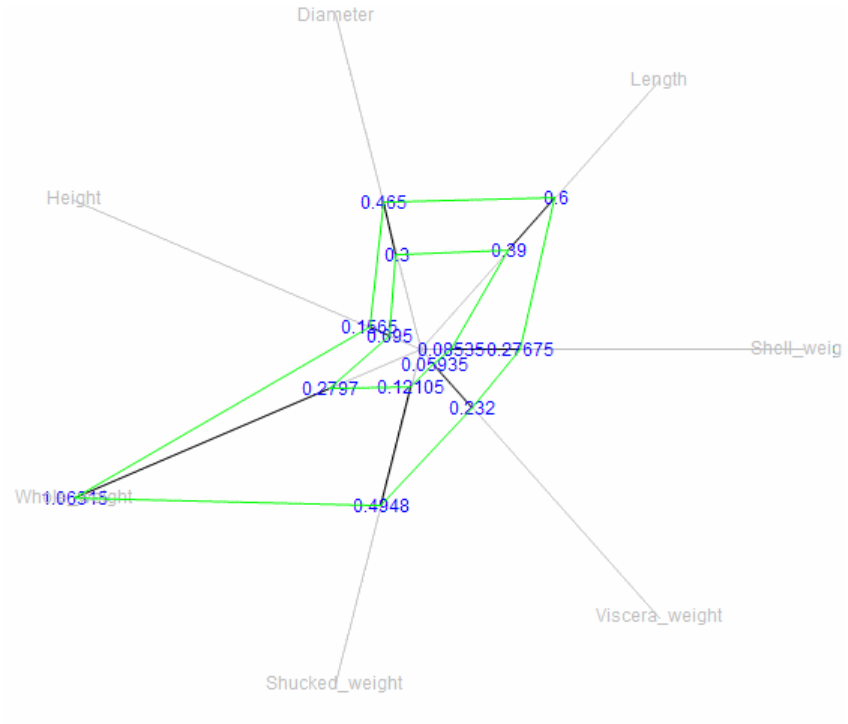

Figura 4.7 Objeto Simbólico gerado através da distância entre o 10-percentil ao 90-percentil

Comparando as figuras 4.6 e 4.7, é perceptível uma grande discrepância entre as duas, notadamente no limite superior da variável Height e nos limites inferiores de praticamente todas variáveis. Isso decorre do fato de que, quando utilizamos "P", estamos cortando os elementos que estão abaixo do 10-percentil e acima do 90-percentil, que provavelmente são exceções em sua população. Em contrapartida, quando utilizamos máximos e mínimos não há perda de informações desses indivíduos, mas são gerados intervalos bem maiores do que deviam por conta dos outliers.

Uma vez transformados os dados clássicos em dados simbólicos do tipo intervalar, é necessário armazenar esses novos dados em uma variável do tipo interval, para se utilizar as demais funções do pacote.

Algorithm 20: Armazenando as variáveis intervalares em variáveis do tipo interval

```
1 height = interval(agrupamentoPercentil[,1],agrupamentoPercentil[,2]);
2 diameter = interval(agrupamentoPercentil[,3],agrupamentoPercentil[,4]);
3 length = interval(agrupamentoPercentil[,5],agrupamentoPercentil[,6]);
```

Agora é possível analisar a relação entre as três váriáveis length, diameter e height através da função intervalGraph3D

Algorithm 21: Constuindo o gráfico tri-dimensional diameter x height x length

```
1 intervalGraph3D(diameter,height,length,1.25,"diameter","height","length");
```

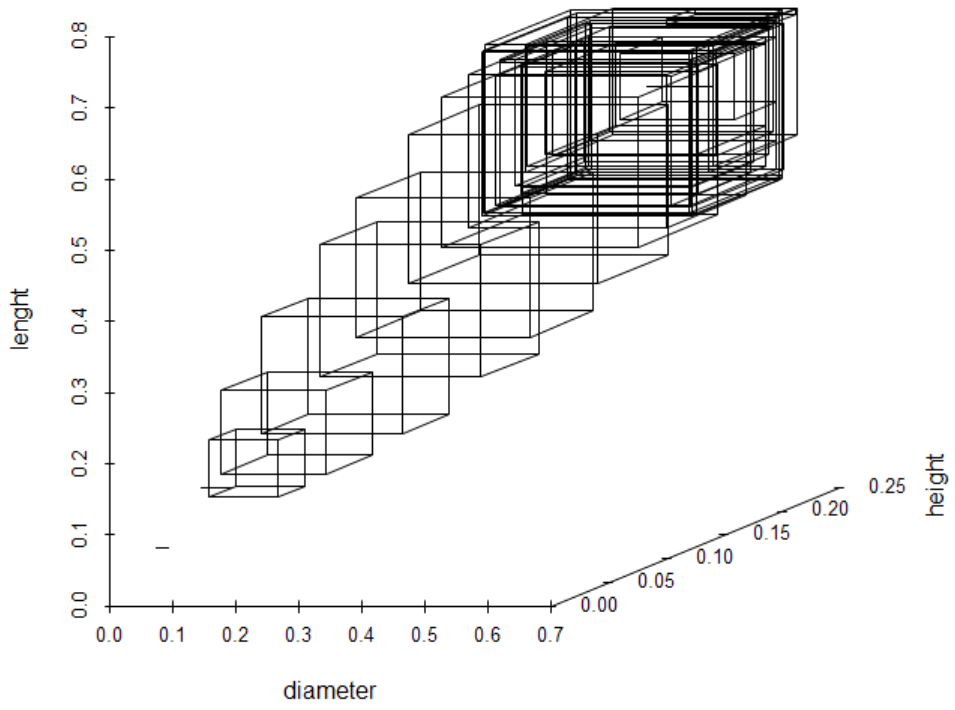

Figura 4.8 intervalGraph3D diameter x height x length

Analisando 4.3, é possível perceber que quanto maior for uma das variáveis, maior serão as demais. Além disso, é visível que as dimensões do cogumelo crescem até chegar a um limite através da concentração dos paralelepípedos no canto superior direito. Como as variáveis foram agrupadas através de "Rings", que representa também a idade do cogumelo, pode-se supor que os paralelepípedos que representam as classes de menor idade estão no canto inferior do gráfico, enquanto as de maior idade no canto superior.

Realizando um estudo mais profundo da variável Height, serão calculadas as medidas estatísticas média, variância, desvio padrão, mediana e moda. Embora exista uma função para cada uma dessas medidas, algumas podem ser calculadas através da função summary

Algorithm 22: Medidas estatistícas para a variável Height

```
1 summary(height);
2 meanInterval(height);
3 varianceInterval(height);
4 modeInterval(height);
5 sdInterval(height);
6 percentileInterval(height,0.5,"T")
```

```
       mean    variance standard_deviance   median
1 0.1438839 0.003310504        0.05753698 0.154901
```

Figura 4.9 Summary da variável height

```
> modeInterval(height,type='T')
$minValue
[1] 0.1565

$maxValue
[1] 0.175

attr(,"class")
[1] "interval"
```

Figura 4.10 Moda da variável height

Como mostrado nas figuras 4.9 e 4.10, os cogumelos terão em média 0.143 milímetro de altura e é de se esperar que cada uma dessas observações varie em média de 0.05 milímetros acima ou abaixo da média como indica o desvio padrão, além disso em metade das observações o valor da altura dos cogumelos estarão acima de 0.154 milímetros. A maior frequência dessa medida para os cogumelos é para a classe que mede entre [0.156,0.175] milímetros. Tais dados podem ser visualizados por construção de histogramas, e serão utilizados para definir o número de classes, a formula de Sturges e o método "T".Nesse caso, a formula de Scott não será utilizada para definir o número de classes, uma vez que o desvio padrão é muito pequeno, o que torna o número de classes igualmente pequeno

Algorithm 23: Construção dos histogramas

```
1 histInterval(height,"T","Histograma de Height", "Height");
2 histInterval(height,'ST',"Histograma de Height", "Height");
```

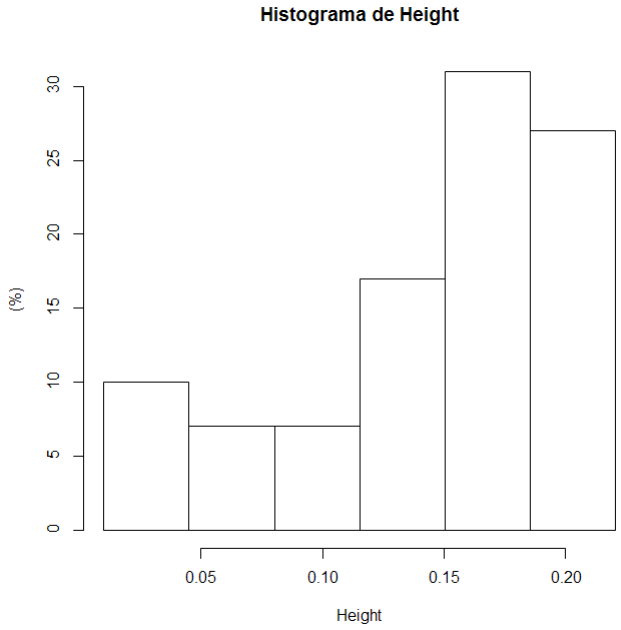

Figura 4.11 Histograma gerado com o número de classes calculado pela fórmula de Sturges

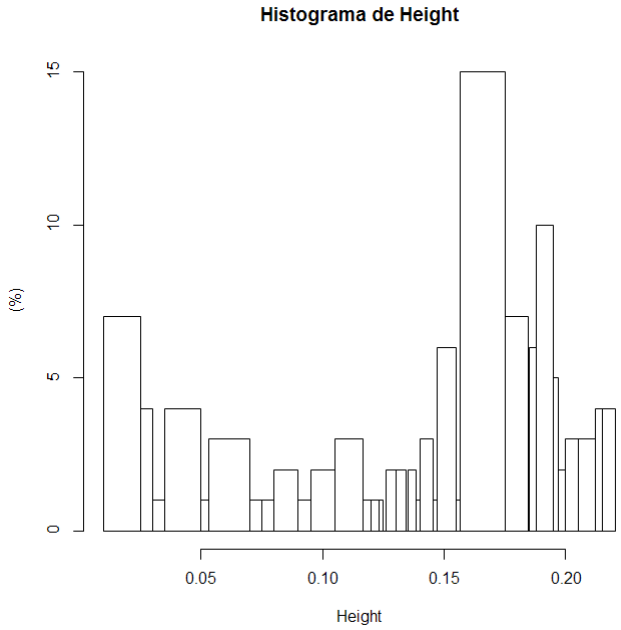

Figura 4.12 Histograma gerado com as classes definidas através do algoritmo ”T”

CAPÍTULO 5

Considerações finais

O presente trabalho foi um esforço para trazer os métodos de análise de dados simbólicos para um ambiente de uma linguagem de programação que é o R-cran.

SODAS2, atualmente o framework mais adotado pelos analistas para a análise de dados simbólicos, que abrange uma grande quantidade de métodos, usa outra abordagem, que é a de construir uma análise de dados através de junção de blocos. Essa abordagem do SODAS2, apesar de ser mais simples de ser usada por pessoas que não tem familiaridade com programação, limita o seu usuário a apenas se utilizar dos métodos presentes no SODAS2.

Desse modo, ISDA.R permite às outras pessoas que estejam trabalhando na área a utilização de suas funções para servir como base na criação de novos algoritmos.

Fica como proposta para trabalhos futuros, a expansão desse pacote para suportar mais métodos de Data mining, como por exemplo: clustering, medidas de dissimilaridade, métodos de regressão, etc.

A principal dificuldade encontrada no decorrer do trabalho foi compreender como os métodos da estatística descritiva foram adaptados para dados simbólicos e o procedimento de submissão do pacote para o repositório oficial da linguagem.

Referências Bibliográficas

[Arroyo et al., 2011] Arroyo, J., González-Rivera, G., Maté, C., and San Roque, A. (2011). Smoothing methods for histogram-valued time series: an application to value-at-risk. *Statistical Analysis and Data Mining*, 4(2):216–228.

[Arshan, 2012] Arshan, H. (2012). Topics in statistical data analysis: Revealing facts from data. Disponível em: http://home.ubalt.edu/ntsbarsh/stat-data/topics.htm [Online; acessado 05-Maio-2012].

[Beleza, 2009] Beleza, M. (2009). Gestão e organização da informação. Disponível em:http://pt.scribd.com/doc/20496449/GESTAO-E-ORGANIZACAO-DA-INFORMACAO [Online; acessado 15-Maio-2012].

[Billard, 2004] Billard, L. (2004). Dependencies in bivariate interval-valued symbolic data. *Classification, Clustering, and Data Mining Applications*, pages 319–324.

[Billard and Diday, 2004] Billard, L. and Diday, E. (2004). Symbolic data analysis: Definition and examples.

[Bock, 2005] Bock, H. (2005). Optimization in symbolic data analysis: dissimilarities, class centers, and clustering. *Data Analysis and Decision Support*, pages 3–10.

[Bock and Diday, 2000] Bock, H. and Diday, E. (2000). *Analysis of symbolic data: exploratory methods for extracting statistical information from complex data*. Springer Verlag.

[Bussab and Morettin, 2010] Bussab, W. and Morettin, P. (2010). *Estatística básica*. Saraiva.

[de Baenst-Vandenbroucke and Lechevallier, 2008] de Baenst-Vandenbroucke, A. and Lechevallier, Y. (2008). Sodas2 software: Overview and methodology. *Symbolic data analysis and the SODAS software*, page 429.

[De Carvalho, 1995] De Carvalho, F. (1995). Histograms in symbolic data analysis. *Annals of Operations Research*, 55(2):299–322.

[de Vargas and Bedregal, 2011] de Vargas, R. and Bedregal, B. (2011). Interval ckmeans: An algorithm for clustering symbolic data. In *Proc. Conf. North American Fuzzy Information Processing Society (NAFIPS 2011)*.

[Diday, 1987] Diday, E. (1987). Introduction à l'approche symbolique en analyse des données. *Actes des journ ees symboliques-num eriques pour l'apprentissage de connaissances a partir des donn ees, Paris*.

[Diday, 1988] Diday, E. (1988). The symbolic approach in clustering and related methods of data analysis. classification and related methods of data analysis. In *Proceedings of the first Conference of the Federation of the classification societies. North Holland*.

[Diday, 1989] Diday, E. (1989). Introduction a l'analyse des donnees symboliques.

[Diday, 1991] Diday, E. (1991). Des objets de l'analyse des données à ceux de l'analyse des connaissances. *Induction Symbolique et Numérique à partir de données, Kodratoff Y. et Diday E. Eds., CEPADUES*.

[Diday, 2008] Diday, E. (2008). The state of the art in symbolic data analysis: overview and future. *Symbolic data analysis and the SODAS software*, pages 3–41.

[Diday and Esposito, 2003] Diday, E. and Esposito, F. (2003). An introduction to symbolic data analysis and the sodas software. *Intelligent Data Analysis*, 7(6):583–601.

[Fayyad et al., 1996] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37.

[Gioia and Lauro, 2006] Gioia, F. and Lauro, C. (2006). Principal component analysis on interval data. *Computational Statistics*, 21(2):343–363.

[Irpino et al., 2003] Irpino, A., Lauro, C., and Verde, R. (2003). Visualizing symbolic data by closed shapes. *Between Data Science and Applied Data Analysis*, pages 244–251.

[Lauro and Gioia, 2006] Lauro, C. and Gioia, F. (2006). Dependence and interdependence analysis for interval-valued variables. *Data Science and Classification*, pages 171–183.

[Le-Rademacher and Billard, 2011] Le-Rademacher, J. and Billard, L. (2011). Likelihood functions and some maximum likelihood estimators for symbolic data. *Journal of Statistical Planning and Inference*, 141(4):1593–1602.

[Le-Rademacher and Billard, 2012] Le-Rademacher, J. and Billard, L. (2012). Symbolic covariance principal component analysis and visualization for interval-valued data. *Journal of Computational and Graphical Statistics*, 21(2):413–432.

[Noirhomme-Fraiture, 2002] Noirhomme-Fraiture, M. (2002). Visualization of large data sets: the zoom star solution. *International Electronic Journal of Symbolic Data Analysis*.

[Warwick J Nash and Ford, 1994] Warwick J Nash, Tracy L Sellers, S. R. T. A. J. C. and Ford, W. B. (1994). The population biology of abalone (haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and islands of bass strait. Disponível em: http://archive.ics.uci.edu/ml/datasets/[Online; acessado 06-Junho-2012].