UNIFACCAMP
Centro Universitário Campo Limpo Paulista

Grafos de Fluxo como Estruturas de Dados para a Representação e Extração de Conhecimento

Emilio Carlos Rodrigues

Janeiro / 2019

Dissertação de Mestrado em Ciência da Computação

Grafos de Fluxo como Estruturas de Dados para a Representação e Extração de Conhecimento

Esse documento corresponde à dissertação apresentada à Banca Examinadora no curso de Mestrado em Ciência da Computação do Centro Universitário Campo Limpo Paulista.

Campo Limpo Paulista, 30 de Janeiro de 2019.

Emilio Carlos Rodrigues

Profa. Dra. Maria do Carmo Nicoletti (Orientadora)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Resumo. *Um Grafo de Fluxo (GF) pode ser abordado como uma estrutura de representação de um conjunto de instâncias de dados de um determinado domínio de conhecimento, que viabiliza a sumarização do conjunto, bem como a agregação à estrutura, de medidas e valores associados ao conjunto. Dessa forma, um GF pode ser caracterizado como uma estrutura de dados bastante informativa, que permite o estabelecimento de processos classificatórios com base na própria estrutura e nas informações que agrega, podendo subsidiar a implementação de técnicas de Aprendizado de Máquina. O trabalho de pesquisa teve por foco o estudo e a investigação de GFs, com o objetivo de identificar vantagens e desvantagens dessa estrutura e do formalismo que a sustenta, quando da contribuição de ambos em processos classificatórios. O trabalho realizado contribuiu com (1) uma proposta de extensão dos GFs, chamada de Grafos de Fluxo Estendidos (GFEs), que amplia o escopo de utilização de GFs por meio da incorporação de métodos de discretização de dados, permitindo seu uso em domínios de dados contínuos, por meio do sistema computacional EFLOWG, desenvolvido durante a pesquisa realizada, e (2), uma proposta de melhoria para os métodos utilizados para extração de classificadores a partir de GFs e GFEs. Em vários domínios de dados, os resultados obtidos pelo classificador estendido mostraram-se promissores quando comparados a resultados obtidos por classificadores induzidos por outros algoritmos mais utilizados na área, como J48, Naïve Bayes, KNN e SVM, disponibilizados junto ao ambiente Weka.*

Palavras-chave. Grafos de Fluxo, Extensão Grafos de Fluxo, Aprendizado de Máquina, Classificadores, Algoritmos de Decisão, Processos Classificatórios.

Abstract*. A Flow Graph (GF) can be approached as a representation structure of a set of data instances of a given knowledge domain, which enables the summarization of the set, as well as the aggregation to the structure, of measures and values associated to the set. In this way, a GF can be characterized as a very informative data structure, which allows the establishment of classification processes based on the structure itself and the information it aggregates, being able to subsidize the implementation of Machine Learning techniques. The research work focused on the study and investigation of GFs, aiming to identify the advantages and disadvantages of this structure and the formalism that underpins it, both of which contribute to classificatory processes. The work accomplished contributed with (1) a proposal to extend the GFs, called Extended Flow Graphs (GFEs), which extends the scope of use of GFs through the incorporation of data discretization methods, allowing their use in continuous data domains, through the EFLOWG computational system developed during the research carried out, and (2) an improvement proposal for the methods used to extract classifiers from GFs and GFEs. In several data domains, results obtained by the extended classifier were promising when compared to results obtained by classifiers induced by other algorithms most used in the area, such as J48, Naïve Bayes, KNN and SVM, available in the Weka environment.*

Keywords. Flow Graphs, Flow Graphs Extension, Machine Learning, Classifiers, Decision Algorithms, Classificatory Processes.

Agradecimentos

Agradeço a Deus, simplesmente por tudo.

Agradeço minha amada esposa, Sabrina, e aos nossos filhotes, Nick e Luke, pelo apoio em todos os momentos dessa jornada.

Agradeço à Profa. Dra. Maria do Carmo Nicoletti pela orientação conduzida, experiências e conhecimento compartilhados.

Agradeço à toda equipe da UNIFACCAMP, em especial, aos docentes, com os quais tive a oportunidade de aprender conteúdos novos e muito interessantes.

Agradeço minha família, amigos e colegas pelos bons momentos.

Gostaria de agradecer, também, a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), pelo suporte dado para a realização deste trabalho e, consequentemente, minha qualificação profissional.

Por fim, agradeço ao Instituto Federal de São Paulo, por permitir meu afastamento das atividades e pela ajuda de custo (nos primeiros meses do Mestrado), ambos concedidos para incentivar minha qualificação.

SUMÁRIO

GLOSSÁRIO

AM – Aprendizado de Máquina

Clustering - Algoritmo de Agrupamento

EFLOWG – *Extended Flow Graphs*

GF - Grafo de Fluxo

GFN - Grafo de Fluxo Normalizado

Inputs – entradas ou dados de entrada

KDD – *Knowledge Discovery in Databases*

MDLP – *Minimum Description Length Principle*

Outputs – saídas ou dados de saída

SRC - Sistema de Representação de Conhecimento

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE ALGORITMOS

LISTA DE ANEXOS

Capítulo 1
Introdução

Os chamados Grafos de Fluxo (GFs), como propostos por Pawlak em um conjunto de artigos ([Pawlak 2003], [Pawlak 2003a], [Pawlak 2003b], [Pawlak 2004b], [Pawlak 2004c], [Pawlak 2005a], [Pawlak 2005b] e [Pawlak 2010]), podem ser abordados como uma estrutura de dados que tem o objetivo de sumarizar um conjunto de instâncias de dados, permitindo a realização de análises voltadas à distribuição de fluxo que ocorre entre seus nós e camadas, e, ao mesmo tempo, servir como um repositório para a extração de conhecimento, via inferência, no contexto do domínio de conhecimento relacionado aos dados que representa. Sob esse aspecto, GFs podem ser considerados estruturas de dados que subsidiam algoritmos de decisão e, portanto, parte integrante do conjunto de técnicas e algoritmos de Aprendizado Indutivo de Máquina (AM) ([Mitchell 1997], [Duda *et al*. 2001], [Bishop 2006] e [Witten *et al.* 2011]).

Foram encontradas na literatura várias definições de grafos de fluxo, associadas a outros conceitos que não aquele definido por Pawlak *i.e.*, a de um dígrafo (grafo direcionado) acíclico que sumariza um conjunto de instâncias de dados. É o caso, por exemplo, do conceito de grafos de fluxo de controle, proposto por F. E. Allen ([Allen 1969, 1970, 1972, 1974]), associados a uma representação baseada em grafos (dígrafos), dos caminhos que podem ser percorridos pelo código de um programa, durante a sua execução, com vistas, principalmente, à otimização do código. Uma variante sua, conhecida como análise do fluxo de dados, se estabeleceu como uma técnica para avaliar e, possivelmente garantir, a confiabilidade de programas de computador (ver [Fosdick & Osterweil 1976]). GFs também diferem, fundamentalmente, dos chamados grafos de fluxo propostos por Ford e Fulkerson ([Ford & Fulkerson 1956]); o algoritmo (ou método) Ford-Fulkerson determina o fluxo máximo em uma rede de fluxo, em que rede de fluxo é um grafo direcionado em que cada arco tem uma capacidade associada e recebe um fluxo; o volume de fluxo em um arco não pode exceder sua capacidade.

Como discutido em [Pawlak 2003], GFs podem ser usados como uma ferramenta matemática para a análise do fluxo de informação em algoritmos de decisão, em

contraste à otimização de fluxo de material, da análise de fluxo clássica. Em muitas situações, GFs definem árvores de fluxo, nas quais alguns ramos podem ser interpretados como regras de decisão e, toda a árvore, como um algoritmo de decisão.

A investigação conduzida e registrada neste documento teve como foco principal a análise das potencialidades, fragilidades e possíveis redundâncias (com relação a formalismos já existentes) da estrutura de Grafo de Fluxos (GFs), em um ambiente de aprendizado indutivo de máquina. As análises buscaram fazer verificações de problemas relacionados ao formalismo que sustenta os GFs e, também, realização de experimentos e análises de resultados utilizando os métodos propostos pelo formalismo de GFs. O objetivo foi o de verificar a viabilidade de uso do formalismo relacionado aos GFs.

Muito embora os primeiros artigos científicos descrevendo a proposta de GFs tenham sido publicados entre os anos 2003–2006, o levantamento bibliográfico, realizado durante o Estudo Dirigido, evidenciou poucos desdobramentos da proposta original e, também, poucas aplicações efetivas em que o formalismo tenha sido utilizado com sucesso em domínios reais.

A pesquisa buscou, também, confirmar algumas suposições inferidas da leitura/estudo de vários dos trabalhos relacionados a GFs, tais como: (1) limitações do formalismo proposto, principalmente a relacionada a tipos de dados utilizados (discretos); (2) falta de rigor em partes do formalismo que subsidia a proposta, (2.1) que traz informações ambíguas e que (2.2) não contempla aspectos importantes da especificação formal da estrutura GF. O Capítulo 4 trata de alguns desses aspectos.

No levantamento bibliográfico realizado sobre GFs foram evidenciadas tanto a existência de algumas propostas de abordagens híbridas utilizando GFs como uma das partes ([Yang *et al.* 2011]), quanto aquelas que abordam GFs associados com algum formalismo já bem estabelecido, como é o caso da abordagem do fluxo da informação em GFs governado pela regra de Bayes ([Pawlak 2004]). Outras propostas baseadas em instâncias também foram encontradas, como no caso de [Sun et al. 2006], em que cada arco armazena identificadores de um conjunto de instâncias de dados que se relacionam (*i.e.*, possuem o mesmo nó de entrada e nó de saída definido pelo arco em questão), possibilitando, ao executar uma varredura sobre o grafo de fluxo, saber exatamente quais instâncias atendem/possuem aquelas características determinadas pelos valores dos nós dos GFs.

Além do Capítulo 1, esse documento está organizado em mais seis capítulos, cujos conteúdos são apresentados, de forma resumida, a seguir.

O Capítulo 2 faz uma breve apresentação da área de AM, de alguns dos conceitos básicos relacionados à área, de algumas áreas de conhecimento, dentre as muitas existentes, que se beneficiam do uso de algoritmos de AM, bem como do tratamento de alguns dos problemas enfrentados por tais algoritmos, particularmente aqueles associados à qualidade e tipo dos dados utilizados. Um GF pode, devido a sua estrutura e às informações que armazena, ser utilizado para subsidiar algoritmos de decisão e, portanto, carrega um forte viés associado a técnicas de aprendizado automático.

O Capítulo 3 apresenta e discute três técnicas, amplamente estabelecidas e difundidas no ambiente de AM, que permitem a realização do particionamento (discretização) de atributos contínuos. Tais técnicas possibilitam que um determinado conjunto de valores contínuos (numéricos) possa ser representado por intervalos de valores. Como vantagem, um atributo contínuo que possui, por exemplo, 150 valores distintos associados, pode ter esse número reduzido, porém, os novos valores associados não seriam exatamente valores numéricos, e sim, intervalos de valores, nos quais estariam enquadrados os antigos valores numéricos associados ao atributo. Essa possibilidade mostrou-se muito interessante e viável para representação, através de GFs, de conjuntos de instâncias de dados que possuem atributos contínuos, pois acaba por diminuir o tamanho do GF, e assim, colabora com melhorias de desempenho computacional e simplificação da estrutura, e do algoritmo de decisão extraído da mesma, como um todo.

O Capítulo 4 tem foco na conceituação básica, notação e formalismo envolvidos no estabelecimento do conceito e propriedades associadas aos GFs. O capítulo usa uma abordagem didática para sedimentar partes das conceituações envolvidas, por meio de exemplos. Também, partes duvidosas e obscuras associadas ao formalismo foram identificadas em várias publicações associadas ao assunto e comentadas por meio de observações. Os exemplos, definições e observações foram numericamente identificados para melhor organização do capítulo.

O Capítulo 5 apresenta o sistema computacional EFLOWG, o qual foi desenvolvido durante o estudo e investigação conduzidos para que pudessem ser testadas, na prática, as teorias e possibilidades investigadas/encontradas. As telas e

elementos principais são apresentados, de forma resumida, com o objetivo de introduzir o EFLOWG ao leitor, para que o mesmo conheça algumas das principais funcionalidades presentes no sistema.

O Capítulo 6 apresenta, inicialmente, a metodologia que foi utilizada para a realização dos experimentos, e trata, basicamente, dos experimentos realizados com os GFs e algoritmos de decisão extraídos dos mesmos, suas variações, resultados, entre outros aspectos. Experimentos com os três diferentes métodos de particionamento, abordados no Capítulo 3, foram também realizados. Uma proposta para extensão dos grafo de fluxo foi proposta, assim como, um novo método para extração de classificadores, baseado no método original. Uma comparação de resultados em relação aos resultados de outros quatro algoritmos, amplamente estabelecidos na área de AM, é apresentada também no Capítulo 6.

O Capítulo 7 apresenta uma investigação empírica sobre o impacto de diferentes sequências de atributos sobre os grafos de fluxo induzidos e os classificadores extraídos destes. O objetivo é o de tentar identificar um padrão na, ou um método de realizar a, ordenação de atributos, de forma que sejam obtidas melhores resultados, tanto em relação às estruturas de grafos de fluxo e classificadores, quanto em relação as acurácias obtidas pelos classificadores nos testes de classificação.

O Capítulo 8, por fim, apresenta a conclusão do trabalho realizado durante a pesquisa, além de, listar possíveis assuntos que podem ser investigados no futuro, gerando assim novos tópicos a serem pesquisados.

Capítulo 2
Aprendizado de Máquina

Aprendizado de Máquina (AM) é uma subárea de pesquisa em Inteligência Artificial com foco, principalmente, na criação e/ou refinamento de algoritmos indutivos voltados ao desenvolvimento de sistemas computacionais que sejam capazes de aprender automaticamente, identificar padrões em instâncias de dados, classificar instâncias de dados, agrupar instâncias com base em suas similaridades, entre outros, a partir de um conjunto de instâncias de dados concretos que foram coletados. É importante enfatizar que, em geral, algoritmos de AM são fortemente influenciados pela maneira como os dados disponíveis são representados e, também, pela estrutura de dados adotada pelo algoritmo para representar o conceito induzido a partir dos dados fornecidos.

É esperado, também, que sistemas computacionais baseados em AM sejam capazes de atualizar suas definições (*i.e.*, seu conhecimento) quando novas instâncias de dados, não consideradas inicialmente, se tornam disponíveis.

Esse capítulo, organizado em sete seções, apresentam e discute vários aspectos relacionados a AM, bem como alguns dos conceitos básicos da área. A Seção 2.1 traz algumas considerações iniciais sobre o ambiente de AM; a Seção 2.2 apresenta, resumidamente, o processo geral de AM; os diferentes tipos de atributo relacionados à área de AM são apresentados na Seção 2.3; os diferentes modelos de AM Supervisionado são tratados na Seção 2.4, e; a Seção 2.5, traz as considerações finais no capítulo.

2.1 Considerações Iniciais

A literatura associada a AM, além de ser ampla, tem um número considerável de focos que abordam tanto aspectos teóricos quanto práticos e, recentemente, tem se expandido em particularizações específicas a muitas de suas áreas de aplicação.

Na literatura, podem ser encontradas inúmeras definições de AM ([Alpaydin 2009], [Mitchell 1997] e [Duda *et al*. 2001]) que, de uma forma ou outra, podem ser traduzidas de uma maneira geral, como sendo uma área de pesquisa que tem por objetivo primeiro, o desenvolvimento de algoritmos computacionais que incorporem aprendizado automático de conhecimento, a partir de instâncias do conhecimento, disponibilizadas nos mais diversos meios (arquivos de dados, imagens, sons, Internet, repositórios de dados, etc.). Como comentado anteriormente, supõe-se que tais algoritmos sejam capazes não só de aprender, mas também, de refinar o conhecimento já aprendido e armazenado, de maneira contínua, à medida que novos dados/informações se tornem disponíveis para o uso ou, então, que sejam implementados de maneira a serem autônomos para buscar dados/informações de interesse.

Com exceção de algumas propostas consideradas de AM, que fazem tradução de linguagens formais em alto nível para alguma de nível mais básico e, para tanto, usam procedimentos dedutivos, toda outra forma de AM empregada é, basicamente, indutiva e, para tal, o processo de aprendizado acontece com base em um conjunto concreto de situações a partir do qual o processo de aprendizado ocorre, via generalização (*i.e.*, indutivo) ([Nicoletti 1994]).

2.1.1 Aprendizado Supervisionado

De acordo com [Theodoridis & Koutroumbas 1999], algoritmos que implementam AM se dividem em três categorias principais nomeadas: (1) aprendizado supervisionado, (2) aprendizado não-supervisionado e (3) aprendizado semissupervisionado. O foco desse trabalho de pesquisa são os algoritmos supervisionados de AM, particularmente os que são baseados em uma estrutura de representação de conhecimento chamada *Grafo de Fluxo* ([Pawlak 2003], [Pawlak 2004a], [Pawlak 2004b], [Pawlak 2006a] e [Pawlak 2005b]).

No aprendizado supervisionado, cada instância de dados é descrita por pares atributo-valor e possuem uma classe associada, que define o tipo, ou o grupo à qual pertence a instância em questão. Os conceitos de instâncias, pares atributo-valor e classes associadas, são abordados com mais detalhes na Subseção 2.2.

No que segue, são brevemente descritas algumas áreas do mundo real que se caracterizam como áreas potenciais para o uso de algoritmos que implementam aprendizado supervisionado.

(1) *Área financeira*: considere a disponibilidade de um conjunto de instâncias de dados contendo informações bancárias sobre clientes (*e.g.*, saldo médio/mês, valor médio de investimentos, empréstimos feitos (S/N), emissão (S/N) de cheque sem fundos, etc.) e, associada a cada instância de dado, a informação (externa), incluída pelo gerente da conta do cliente, informando a *classe* de tal cliente, ou seja, se tal cliente é um cliente de 'baixo risco' ou de 'alto risco'. Com base nesse conjunto de instâncias de dados, um algoritmo supervisionado, direcionado pela *classe* do cliente (alto-risco ou baixo-risco), induz uma expressão geral, descrita utilizando as informações que descrevem os clientes, elencadas acima. Um sistema computacional, usando as expressões gerais induzidas pelo algoritmo de aprendizado que representam um cliente 'baixo-risco' e um cliente 'alto-risco' pode, com base nas informações de um novo cliente, determinar se tal cliente é 'baixo-risco' ou 'alto-risco' em termos de perfil financeiro.

(2) *Área médica*: o aprendizado de máquina tem dado grande suporte à área médica em diversos aspectos e utilizações, em sua maioria, através da análise de imagens geradas por aparelhos de exames específicos. Cardiologistas podem utilizá-lo como ferramenta de apoio para interpretação de diagnósticos, uma vez que esses tipos de algoritmos conseguem analisar automaticamente ECGs (eletrocardiogramas) e indicar o diagnóstico mais provável a partir de um conjunto limitado de diagnósticos ([Deo 2015]). De maneira análoga, é possível classificar padrões contidos em radiografias, mamografias, encefalogramas, ultrassonografias, tomografias computadorizadas indicando a existência ou não de nódulos, tumores, manchas ([Giger *et al.* 2008] e [Theodoridis & Koutroumbas 1999]), problemas neurodegenerativos como Mal de Alzheimer’s e Mal de Parkinson ([de Brujine 2016]), entre outros.

A classificação de um tumor entre benigno e maligno ([Mitchell 1997]) e a previsão de riscos e doenças futuras ([de Brujine 2016] e [Deo 2015]) são outros exemplos onde a utilização do aprendizado supervisionado pode ser empregada. Os

poucos exemplos citados demonstram várias (das inúmeras) aplicações de algoritmos de aprendizado de máquina na área médica.

(3) *Área biológica:* cientistas e pesquisadores têm feito uso do aprendizado de máquina também voltado à área biológica. Na maioria dos casos, estuda-se um problema específico com base em seus dados (*e.g*., células, genes, imagens, etc.), dos quais, com o auxílio do AM, é possível identificar padrões e classificá-los, extraindo assim, conhecimento. O uso de AM pode colaborar para descobertas e definições que levam ao desenvolvimento de remédios e vacinas ([Heinson *et al.* 2017]), possibilita a previsão de sucesso de determinados tratamentos ([Acion *et al.* 2017]), permite a identificação da presença de substâncias ou elementos específicos (*e.g.*, nanomedicamentos, nanodispositivos, entre outros) em pacientes ([de la Iglesia *et al.* 2014]), serve como ferramenta de apoio à descoberta de biomarcadores[1] de doenças ([Swan *et al.* 2015]), possibilita estudos sobre proteínas diversas ([Sastry *et al.* 2017], [Kumar *et al.* 2007], [Diaz *et al.* 2010], [Habibi *et al.* 2014] e [Idicula-Thomas *et al.* 2006]), assim como, estudos sobre câncer ([Menden *et al.* 2013], [Palaniammal & Chandrasekaran 2015], [Rani G. *et al*. 2017], [Kourou *et al.* 2015] e [Raub & Nehmetallah 2017]), entre muitas outras aplicações.

(4) *Área de engenharia*: devido a característica do aprendizado de máquina possibilitar a identificação de padrões de acontecimentos implícitos em dados assim como determinar sua classificação, é ampla sua utilização nas diversas áreas da engenharia.

Na engenharia de materiais, por exemplo, sistemas de aprendizado de máquina foram utilizados para identificação de danos em materiais ([Hattori & Sáez 2013]) e análise de conexões entre certos tipos de materiais colaborando com a classificação de materiais existentes, assim como, com a criação/descoberta de novos materiais ([Pankajakshan *et al.* 2017], [Raccuglia *et al.* 2016] e [Liu *et al.* 2015]). Na engenharia civil, AM também aparece em inúmeras aplicações como, por exemplo, em estudos de

[1] Sintomas comuns que podem indicar a presença de certas doenças.

acidentes de construção e prevenção dos mesmos ([Arciszewski *et al.* 1995]), auxílio no desenvolvimento de ferramentas de gerenciamento para controle de segurança estrutural ([Stone *et al.* 1989]), melhorias em materiais ([Taffese & Sistonen 2017]), entre outras.

(5) *Área de educação*: o AM vem sendo utilizado para que professores tenham uma ferramenta de auxílio no ensino. No estudo da arte para crianças, por exemplo, é possível utilizar AM para colaborar com o processo de ensino-aprendizagem tornando mais interessantes e desafiadoras as aulas ([Morris *et al.* 2013]). Contudo, é fato que aulas interessantes não são garantia de aprendizado, por isso, é importante que os docentes tenham, além de aulas interessantes, uma forma de avaliar se suas estratégias de ensino são eficazes. Para tal, uma nova técnica para fazer a previsão da eficácia da estratégia de ensino foi desenvolvida utilizando, também, AM ([Kushik *et al.* 2016]).

A estratégia utilizada pela modalidade de Ensino a Distância (EaD), muitas vezes, pode ser confusa, pois baseia-se, praticamente, em leitura e vídeos, materiais estes que devem ser consultados pelos estudantes para aquisição do conhecimento. Porém, o processo de aprendizagem destes alunos muitas vezes é incerto, pois em uma sala de aula, o docente consegue observar, perceber facilidades e dificuldades da turma, coisas que não são possíveis no EaD. Mas ainda assim, é possível realizar análises dos *logs* (*i.e.*, dados de utilização) do sistema de EaD em busca de padrões que podem indicar, entre muitas coisas, como foi o processo de aprendizagem de cada aluno ([Stimpson & Cummings 2014]). Quanto mais detalhados os *logs*, mais informações podem ser extraídas durante as análises.

(6) *Tecnologia da Informação*: com a popularização da *Internet of Things* (IoT), e consequentemente, das *Smart Things*, o AM ganhou mais aplicações, pois, para atender bem os moradores de *smart homes* e/ou *smart cities*, dados coletados constantemente devem ser analisados com intuito de o sistema como um todo aprender e compreender as rotinas dos usuários e, com uso desse aprendizado, proporcionar vantagens aos mesmos ([Naouar *et al.* 2016]). Na robótica, o AM é um conceito estabelecido que auxilia em diferentes tarefas, como por exemplo, no processo de descobertas robóticas ([Bratko 2008]), no processo de avaliação de determinadas tarefas realizadas por robôs ([Fard *et al.* 2016]), entre outros. Como última subárea da engenharia a ser citada nesses

exemplos, a utilização do AM na produção industrial vai desde operações de controles de maquinário até mesmo inspeção da qualidade de produtos produzidos ([Wuest *et al.* 2014]).

2.1.2 Classificadores ou Algoritmos de Classificação

Via de regra, a expressão do conceito induzida por algoritmos supervisionados considerando instâncias de dados de um determinado domínio de conhecimento X, são expressões que atribuem uma classe a uma nova instância de dado de X, ainda não classificada, com base apenas em sua descrição e, devido a isso, muitas vezes, tais expressões implementadas como um sistema computacional são caracterizadas como *classificadores*.

Os classificadores, em sua etapa de treinamento (aprendizado inicial), têm como dados de entrada um conjunto de instâncias (ou exemplos). Essas instâncias são descritas por valores de atributos, em que cada atributo representa uma característica relevante da instância. Faz parte também da descrição das instâncias a classe à qual elas pertencem. A classe é uma informação fundamental para que possam ocorrer os processos de treinamento e aprendizado.

Diversos aspectos devem ser considerados, analisados e definidos cuidadosamente, com o objetivo de implementar AM como um sistema computacional, dos quais destacam-se, de acordo com [Nicoletti 1994]:

(1) qual a representação a ser adotada para os dados a serem fornecidos para o algoritmo de AM, *i.e.*, em qual tipo de linguagem as instâncias de dados são representadas (*e.g.*, proposicional, lógica de primeira ordem);

(2) a representação conterá (ou não) informação obtida via supervisão externa;

(3) quais os principais tipos de atributos usados para uma descrição de dados baseada em atributos (atributos nominais, numéricos, discretos, não estruturados (*e.g.*, informação textual), etc.);

(4) formas de tratamento de atributos contínuos;

(5) o domínio de dados em questão é (ou não) caracterizado pela presença de ruído nos dados;

(6) qual tipo de algoritmo empregar, dentre os muitos disponibilizados para implementar AM, considerando os dados disponíveis;

(7) qual técnica de validação utilizar para realizar o aprendizado;

(8) necessidade do uso de Teoria do Domínio no aprendizado;

(9) dados disponibilizados têm valores ausentes de atributos e o tratamento via imputação, e;

(10) dados supervisionados com os números de instâncias de dados desbalanceados entre as classes.

Alguns dos aspectos supracitados são detalhados nas seções que seguem.

2.2 Notação e o Processo Geral de AM

Esta seção, inicialmente, tem foco no estabelecimento de uma notação formal para, então, abordar os principais conceitos envolvidos em AM, com vistas à padronização da nomenclatura ao longo de todo o trabalho. Como mencionado anteriormente, para que o processo de AM possa ser efetivado, é sempre necessário que um conjunto de instâncias de dados seja disponibilizado. Tal conjunto de instâncias de dados é, genericamente, referenciado como *conjunto de treinamento*.

Cada instância do conjunto de treinamento é, via de regra, descrita por valores associados a um conjunto de características, referenciadas como *atributos* e, também, por uma *classe* associada (cujo valor indica qual conceito tal instância representa). É comum conjuntos de instâncias de dados não incluírem a classe à qual cada uma delas pertence.

A representação mais comumente utilizada para descrever uma instância de dados é a de um vetor. Um conjunto de treinamento X, com N instâncias, pode ser representado como o conjunto $X=\{X_1,X_2,...,X_N\}$, em que cada X_i, $1 \leq i \leq N$, é uma instância de dado, descrita por M valores, cada um deles associado, respectivamente, a

cada um dos M atributos, (A_1, A_2, ..., A_M), e também, por uma *classe* associada, dentre um conjunto de P possíveis classes, $C=\{C_1,C_2,...,C_P\}$.

Exemplo 2.1 Considerando a área financeira, uma instância de dado poderia ser descrita por quatro atributos A_1, A_2, A_3, A_4 e uma *classe* associada, em que A_1 representa o saldo médio do cliente (cujo valor é um número real), A_2 representa valor de investimentos (cujo valor é um número real), A_3 representa a situação do cliente ter feito (ou não) empréstimos (atributo nominal cujo valor pode ser S(im) ou N(ão)) e A_4 representa a situação do cliente ter emitido ou não cheque sem fundo (atributo nominal cujo valor poder ser S(im) ou N(ão)). O conjunto de possíveis classes poderia ser representado, no caso, por $C=\{C_1,C_2\}$={baixo_risco,alto_risco}. Um conjunto de treinamento fictício, por exemplo, é mostrado na Tabela 2.1. O mecanismo empregado por um algoritmo de AM, particularmente um que representa o conceito como uma árvore de decisão, considerando os dois valores de classe (baixo_risco (br) e alto_risco (ar)) associados aos dados da Tabela 2.1, é mostrado na Figura 2.1.

Tabela 0.1 Exemplo de um conjunto de treinamento X associado ao domínio financeiro considerado anteriormente, exibindo a descrição de cinco instâncias de dados (I_1, I_2, I_{35}, I_{47}, I_{200}), cada uma delas descrita por quatro atributos (A_1, A_2, A_3 e A_4) e uma classe associada (C_1 = br (baixo_risco) ou C_2 = ar (alto_risco). ID representa a identificação da instância e não é um atributo que participa do processo de AM.

ID	**A_1**	**A_2**	**A_3**	**A_4**	**classe**
1	10000	32.453,55	N	N	br
2	380,39	0,00	N	S	ar
…	…	…	…	…	…
35	2.544,79	351.743,95	N	N	br
…	…	…	…	…	…
47	71,64	1.870,45	S	N	ar
…	…	…	…	…	…
200	200,00	50,00	S	S	ar

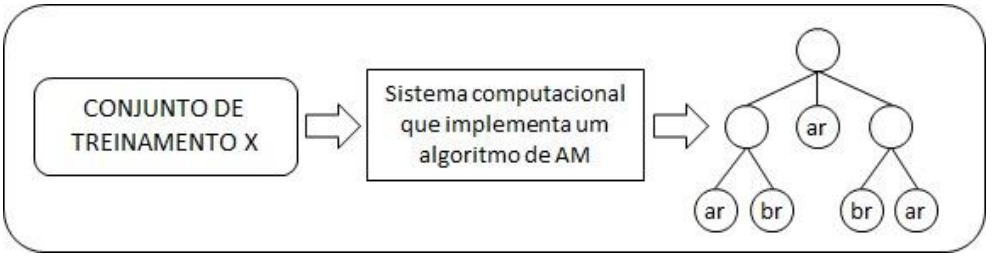


Figura 0.1 Esquema geral do uso de algoritmo de AM para a indução de um conceito, com base no conjunto de treinamento fornecido X (Tabela 2.1). A linguagem de descrição usada pelo algoritmo para representar o conceito por ele induzido é uma árvore de decisão.

2.3 Os Diferentes Tipos de Atributo

Cada atributo que descreve uma instância armazena informação referente a uma determinada característica da instância, expressa como um valor associado. Esses valores associados dão sentido a cada um dos atributos utilizando um tipo de dados específico, i.e., há uma restrição de possíveis valores que o atributo pode assumir. Essa restrição define o domínio do atributo que, por sua vez, define o tipo do atributo. Como pode ser visto no glossário encontrado em [Kluwer Academic Publishers 1998], entre os domínios de dados mais comuns, encontram-se:

(1) *Categórico:* esse tipo de domínio permite a utilização de um número finito de atributos discretos que podem ser do tipo nominal, binário ou ordinal. Um atributo nominal pode assumir valores que não têm ordem entre si (*e.g.* nomes e cores). Um atributo binário assume apenas dois possíveis valores (*e.g.*, Verdadeiro ou Falso; Sim ou Não; 0 ou 1). Por sua vez, um atributo ordinal possui valores discretos cuja interpretação revela que há uma certa ordem estabelecida entre eles (*e.g.*, conceitos em um curso (de A à E) e nível de experiência de um colaborador (Jr., Pl., Sr., Esp., entre outros)). Nota-se que o domínio categórico possui atributos qualitativos e, devido a isso, comparações matemáticas ou lógicas entre eles não são definidas (como é o caso, por exemplo, da tentativa de subtrair o valor de atributo "amarelo" do valor de atributo "azul").

(2) *Contínuo:* o domínio contínuo permite tipos de atributos que representam valores quantitativos, *i.e.*, valores mensuráveis. Números, inteiros e reais, constituem o exemplo típico de atributos contínuos. Esse domínio pode vir a ser um problema para o AM se não houver tratamento adequado, uma vez que, na teoria, um atributo com domínio

contínuo pode possuir infinitos valores associados (*e.g.* peso de pessoas considerando três casas decimais).

2.4 Os Diferentes Modelos de AM Supervisionado

Também conhecidos como paradigmas de AM, os modelos de AM estão relacionados com o algoritmo empregado pelo sistema computacional que o implementa. Entre os principais, encontram-se os modelos:

(1) *Simbólico:* sistemas que empregam o aprendizado simbólico aprendem formando representações simbólicas com base na análise de exemplos e contraexemplos do conceito a ser aprendido. Segundo [Chen 1995], a maioria dos sistemas que utilizam o modelo de aprendizado simbólico produzem regras de produção ou conceitos hierárquicos (*e.g.* regras, árvores de decisão) como saída. Esses tipos de saída são de fácil entendimento e suas implementações geralmente são eficientes, especialmente, se comparadas com aquelas utilizadas por redes neurais e por algoritmos genéticos.

(2) *Baseado em instâncias:* este modelo de aprendizado trata do aprendizado que é feito com base em exemplos conhecidos. O sistema computacional, ao receber uma nova instância, busca por instâncias similares já armazenadas e classifica a nova instância como sendo da mesma classe das similares encontradas. Algoritmos de aprendizado baseado em instâncias, via de regra, mantém o conjunto de treinamento (conjunto de instâncias conhecidas) em memória, fato que, dependendo do volume de instâncias presentes em tal conjunto, vem a ser uma desvantagem desse modelo (para mais informações, ver [Alpaydin 2010]).

(3) *Neural (ou conexionista):* este modelo teve como inspiração o modelo biológico do sistema nervoso que, via de regra, é caracterizado pela alta interconectividade entre os elementos envolvidos. As redes mais comumente usadas são as que implementam classificadores. Abordadas de uma maneira simplista e com foco em redes neurais que são classificadores, uma rede neural é, geralmente, organizada em camadas, em que cada camada tem um determinado número de unidades, cada uma delas identificada como um nó da rede.

A camada inicial da rede, referenciada como camada de entrada (ou *input*), é a camada que recebe os dados a serem usados para treinamento (durante a fase de treinamento da rede) ou, então, para serem classificados, durante a fase de classificação (após o treinamento da rede). A camada de saída (ou *output*) é formada por um conjunto de nós, responsáveis pela atribuição de classe às instâncias sendo classificadas. Camadas entre a camada de entrada e a camada de saída são chamadas de camadas intermediárias. A topologia (ou arquitetura) da rede é definida em uma fase anterior ao aprendizado, via de regra por meio de um processo de tentativa e erro, em que várias topologias (*e.g.*, diferentes números de nós por camadas e diferentes números de camadas, entre outros) são experimentadas. É importante observar, entretanto, que algoritmos que implementam redes neurais construtivas tratam ambos os processos *i.e.*, o de treinamento e o de definição da arquitetura da rede, de maneira articulada, explorando a interdependência entre ambos. A referência [Palma Neto & Nicoletti 2005] apresenta um texto introdutório sobre redes neurais construtivas. A Figura 2.2 mostra o diagrama de uma rede neural.

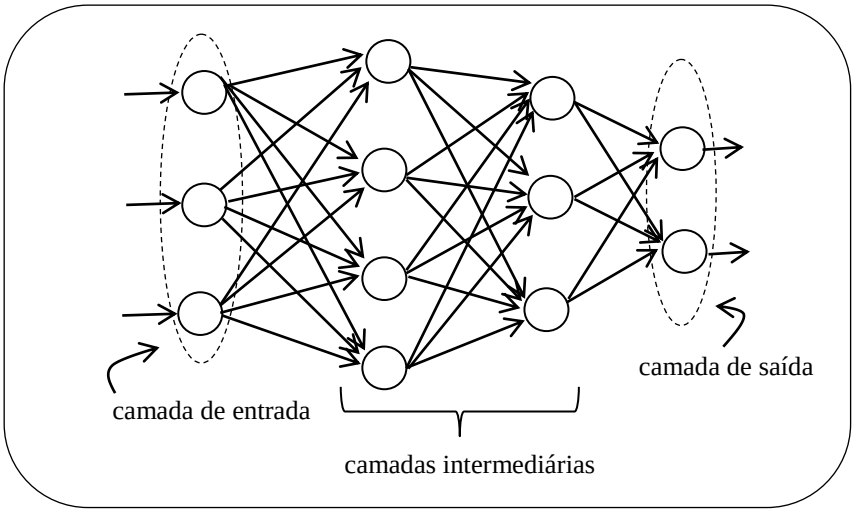


Figura 2.2 Diagrama gral de uma rede neural com duas camadas intermediárias, em que a primeira tem quatro nós e a segunda três.

Sistemas que implementam este modelo de aprendizado são considerados extremamente eficientes e, por isso, são amplamente utilizados em “resoluções de problemas complexos como reconhecimento de voz e de imagem, nos quais várias hipóteses são consideradas, altas taxas de computação são requeridas e os melhores

sistemas disponíveis no momento estão longe de se igualar à performance humana" ([Lippman 1987]). Mitchell, em [Mitchell 1997] comenta que a utilização do modelo neural é apropriada quando:

- as instâncias disponíveis são representadas por muitos pares de atributos-valores;
- a saída da função deve ser um valor discreto, valor real ou um vetor de valores discretos ou reais;
- o conjunto de treinamento pode conter erros;
- uma avaliação rápida da função aprendida é requerida, e;
- a compreensão humana da representação do conceito aprendido não é importante.

(4) *Estatístico:* esse modelo consiste na utilização de métodos estatísticos para realização do aprendizado e é implementado utilizando métodos muito semelhantes aos desenvolvidos por pesquisadores da área de Probabilidade e Estatística. O aprendizado bayesiano é um dos métodos estatísticos mais utilizados neste modelo de AM, pois permite a criação de regras estatísticas que podem predizer valores e/ou riscos com base na análise em casos anteriores ([Alpaydin 2010]).

(5) *Genético:* também referenciado como modelo evolutivo, este modelo é derivado do modelo evolucionário de aprendizado. O modelo gerencia vários elementos, cada um deles considerado uma possível solução do problema, que competem entre si. Os elementos insatisfatórios são descartados e os eficazes sofrem variações com o objetivo de tornarem-se melhores. É possível, assim, fazer uma analogia direta com a teoria de Darwin, segundo a qual, apenas aqueles indivíduos mais adaptados ao ambiente sobrevivem.

De acordo com [Mitchell 2012], os algoritmos deste modelo, exceto variações muito radicais, geralmente obedecem a seguinte estrutura: o algoritmo iterativamente atualiza um *pool* (local) de hipóteses chamado *população*. Em cada uma dessas atualizações os elementos (cromossomos) da população são avaliados de acordo com uma função de *fitness* (*i.e.*, uma função que otimiza uma medida numérica predefinida para o problema em questão). Então, uma nova população é criada selecionando probabilisticamente, da população atual, os elementos que melhor se enquadram. Alguns

deles são selecionados para a próxima geração de população, enquanto outros são utilizados como base para a criação da nova prole de indivíduos por meio do uso de operações genéticas tais como cruzamento e mutação.

2.5 Considerações Finais

Esse capítulo introduziu, de forma sucinta, o conceito de Aprendizado de Máquina (AM) e outros conceitos diretamente relacionados ou derivados.

Inicialmente, foram evidenciadas algumas áreas em que algoritmos de AM podem ser aplicados para exemplificar algumas de suas possibilidades de utilização. Entre os tipos de aprendizado de máquina, o aprendizado supervisionado foi enfatizado no texto, pois o trabalho de mestrado foi desenvolvido com foco em Grafos de Fluxo no contexto de aprendizado supervisionado.

O capítulo a seguir trata sobre o pré-processamento de dados, etapa importante do processo geral para descoberta de conhecimento.

Capítulo 3
Pré-processamento de Dados em Ambiente de AM

Em um ambiente de AM, as instâncias de dados que compõem os conjuntos de treinamento e de teste, nem sempre estão descritas de maneira adequada para poderem ser usadas como entradas para algoritmos de AM. É necessário, portanto, realizar uma série de análises, com o objetivo de identificar problemas nos dados e, se encontrados, corrigi-los. O pré-processamento de dados é uma etapa importante do processo de descoberta de conhecimento (KDD) ([Fayyad *et al.* 1996]), responsável por identificar e corrigir problemas com os dados originais, com o objetivo de adequá-los às etapas que envolvem processos associados a AM.

Este capítulo está organizado da seguinte maneira: a Seção 3.1 traz as considerações iniciais sobre a etapa de pré-processamento de dados e a Seção 3.2 apresenta possíveis ruídos que, por ventura, estejam presentes nos dados, assim como, algumas formas/métodos para detecção desses tipos de problemas. Técnicas para tratamento de ruídos nos dados são apresentadas na Seção 3.3. A Seção 3.4, por sua vez, apresenta o tratamento de atributos contínuos que, muito embora não seja um problema *per se*, pode ser um problema para determinados algoritmos que não foram projetados para tratar tal tipo de dado. Tal tratamento pode ser implementado por meio de várias técnicas. A Seção 3.4, particularmente, aborda três técnicas/algoritmos de particionamento, e por fim, a Seção 3.5 resume os principais pontos enfatizados no capítulo.

3.1 Considerações Iniciais sobre Problemas em Dados

As instâncias de dados utilizadas em processos de AM podem ser originadas de diversas fontes e terem diferentes formas de representação e de coleta (*e.g.*, cadastros, coleta manual, coleta automática, medição por instrumentos eletrônicos). Porém, como

enfatizam vários autores, independentemente da origem dos dados, eles podem ter problemas ([Little & Rubin 1987], [Schafer 1997]), entre os quais, destacam-se: foram registrados de forma incorreta; não pertencem ao domínio ao qual estão associados; têm discrepâncias com relação a dados pertencentes ao domínio de dados em questão; possuem o mesmo significado, porém, estão representados de formas diferentes, e; também, podem estar ausentes ([Monago & Kodratoff 1987], [Twala 2013]).

O não tratamento dos problemas, que eventualmente as instâncias de dados podem ter, tende a causar efeitos indesejados, comprometendo o processo de AM. Dentre estes efeitos colaterais, estão: algoritmos com alto custo computacional dado à quantidade de hipóteses induzidas; indução de hipóteses não representativas, *i.e.*, que não representam a realidade dos dados; indução de hipóteses enviesadas pelos ruídos presentes nos dados, entre outros.

Fica evidente, portanto, a importância do pré-processamento de dados, cujo objetivo é identificar, eliminar e diminuir o número de problemas nos dados, para que, então, estes estejam adequados para utilização no processo de AM.

3.2 Possíveis Problemas nos Dados

Esta seção tem por finalidade apresentar alguns dos principais problemas encontrados nos dados, e também, alguns possíveis tratamentos que podem ser aplicados para resolução e/ou diminuição de tais problemas.

3.2.1 Ruídos nos Dados

O problema de ruído nos dados é um dos desafios enfrentados não apenas por algoritmos de AM, mas também por qualquer outro método/técnica que se utiliza de dados.

Como definido em [Alpaydin 2010]), ruído "é uma anomalia nos dados que pode dificultar o aprendizado de uma classe" assim como, enviesar o resultado obtido por algoritmos de agrupamento e resultados obtidos por métodos de busca, como aqueles implementados por algoritmos genéticos, por exemplo. De acordo com [Zhu & Wu 2004], ruídos podem aumentar tanto o tempo de indução do classificador, como o tamanho do mesmo, afetando assim, o desempenho do processo de AM como um todo.

De acordo com pesquisas ([Lakshminarayan *et al.* 1996], [Kalapanidas *et al.* 2003], [Alpaydin 2009] e [Garcia 2016]), a origem de um ruído nem sempre é conhecida. O ruído pode ter sido originado durante o processo de coleta dos dados, pode ter sido causado por erros randômicos, variação na medição do valor de um atributo por um instrumento (sensor, etc.), informação indisponível no momento do cadastro, problemas com transmissão de dados (via rede ou internet), limitação tecnológica, erros de cálculos, entre outros ([Garcia *et al.* 2013a]).

Segundo [Zhu & Wu 2004], quando o foco é no uso de algoritmos de AM, os ruídos podem ser divididos em dois grupos: ruídos em valores de atributos e ruídos em valores de classes. Independente do grupo do ruído, estes contribuem para a indução de classificadores com resultados inconclusivos ou incorretos.

Geralmente, ruídos aparecem nos dados como valores completamente diferentes do restante dos valores do domínio do atributo em questão, aparentando, assim, nem mesmo pertencer àquele domínio de dados (*e.g.*, atributo idade com o valor associado "M" quando todos os outros valores referentes à idade são valores numéricos do tipo inteiro). De acordo com [Lorena & Carvalho, 2004], dados com rótulos (ou classes) incorretos, dados redundantes (*e.g.*, dois valores diferentes que possuem o mesmo significado), *outliers* (subitem 3.3.3) e dados próximos à borda de decisão (*i.e.*, dados duvidosos a ponto de causar confusão no classificador pelo fato de não estarem claramente definidos) são problemas de ruídos frequentemente encontrados em conjuntos de instâncias de dados, principalmente naqueles compostos por dados reais (não criados manualmente para testes ou outros fins).

Diversos métodos e técnicas podem ser utilizados para detecção/resolução do problema de ruído nos dados. Alguns deles, de acordo com [Hodge & Austin 2004], são: métodos de AM não-supervisionado (agrupamento ou *clustering*) – esses métodos podem ser utilizados quando as instâncias do conjunto de instâncias de dados não possuem classes associadas, *i.e.*, não estão classificadas, rotuladas. Consistem em agrupar as instâncias de dados utilizando distribuição estatística baseada na similaridade entre as mesmas. Dessa forma, são identificadas instâncias de dados que são mais isoladas em relação às instâncias de dados de seus grupos, as quais podem representar ruídos nos dados; modelos de AM supervisionado (classificadores) – esses modelos

realizam testes sobre as instâncias com base em um aprendizado previamente adquirido. Esse aprendizado é feito sobre o conjunto de treinamento (o qual possui instâncias de dados rotuladas/classificadas) e testado sobre o conjunto de testes (o qual serve para auxiliar na verificação da acurácia da hipótese induzida pelo classificador). Após devidamente testada e aceita, espera-se que a hipótese seja capaz de classificar corretamente instâncias de dados. Durante o processo de classificação, podem ser encontradas instâncias de dados que fogem completamente do padrão dos dados, representando assim, anomalias nos dados (*i.e.*, ruídos), e; métodos de AM semi-supervisionado – são métodos de AM que utilizam de características de AM supervisionado e não-supervisionado para poder identificar ruídos nos dados.

Após a detecção de valores com ruído, algumas alternativas podem ser utilizadas para resolução/diminuição do problema de ruídos nos dados. Entre as alternativas, destacam-se a eliminação da instância com ruídos e o preenchimento/substituição (automático ou manual) do valor com ruído. Tais alternativas são discutidas com mais detalhes na Seção 3.3.

3.2.2 Valores Ausentes de Atributos

No mundo real, em diversas situações, conjuntos de instâncias de dados podem possuir valores ausentes, *i.e.*, alguma(s) característica(s) das instâncias de dados em questão, representadas por atributos, podem não possuir valores associados.

É importante esclarecer, porém, que, o fato de haver valores ausentes de atributos em conjuntos de instâncias de dados, é um problema, se e somente se, o algoritmo a ser utilizado for sensível à isso, pois há algoritmos de AM capazes de lidar com atributos com valores ausentes, como por exemplo, o algoritmo de associação *Apriori*, amplamente utilizado no varejo ([Agrawal *et al.* 1994], [Agrawal & Srikant 1994]). Os GFs, investigados nesta dissertação, são sensíveis a esse tipo de problema.

As causas do problema de valores ausentes podem ser as mais variadas, *e.g.* ausência da informação na hora do cadastro, problemas técnicos de transferência de dados via redes de comunicação de dados ([Lakshminarayan *et al.* 1996]), entre outras. Porém, se o algoritmo de AM empregado não for capaz de lidar com valores ausentes, a eficiência do mesmo pode ser prejudicada. As alternativas para tratamento de valores

ausentes são, basicamente, a eliminação daquelas instâncias com valores ausentes e o preenchimento/substituição (automático ou manual) dos valores ausentes (Seção 3.3).

3.2.3 Presença de *Outliers* nos Dados

Presentes apenas em valores numéricos, *outliers* são dados que, apesar de pertencerem a um mesmo domínio de dados, fogem muito do padrão, quando comparados aos outros dados do domínio em questão. *Outliers* são, via de regra, considerados anomalias.

Estatisticamente, como apontado em [John 1995] e [Weisberg 1985], *outliers* são definidos como dados que não seguem o mesmo modelo que o restante dos dados do conjunto aos quais pertencem. A presença de *outliers* no conjunto de treinamento pode influenciar radicalmente os resultados obtidos por processos de AM e, por isso, devem ser tratados com cautela. As origens dos *outliers*, assim como dos ruídos em geral, podem ser diversas, entre elas: falhas mecânicas, mudanças no comportamento do sistema, comportamento fraudulento, erro humano ou erro do instrumento, etc. ([Hodge & Austin 2004]).

Um exemplo da presença de *outliers* pode ser o seguinte: uma empresa possui dez colaboradores, sendo que um deles é o proprietário da empresa. O empresário tem um salário de R$ 15.000,00 (quinze mil reais) mensais, enquanto os outros nove colaboradores da empresa, tem vencimentos de R$ 1.000,00 (um mil reais) mensais cada. Analisando tais dados, observa-se que a média salarial mensal dos colaboradores desta empresa é de R$ 2.400,00 (dois mil e quatrocentos reais), ou seja, mais que o dobro do salário de 90% dos colaboradores. Portanto, nota-se que o valor referente ao salário do proprietário pode ser considerado um *outlier*. Nota-se, também, que o *outlier* está influenciando os resultados e distorcendo a realidade encontrada na empresa.

Porém, é importante sempre verificar, junto a um especialista do domínio de dados, se os *outliers* detectados são, de fato, *outliers*. Se o caso da empresa acima é real, então o *outlier* identificado não poderá ser descartado ou ter seu valor substituído, pois tal ação acabaria por distorcer a realidade dos dados. Portanto, como sugerido anteriormente, é importante ter cautela ao analisar *outliers* e tomar decisões com relação a eles.

Uma das técnicas mais utilizadas (e simples) para detecção de *outliers* é baseada no conceito de quartis, apresentado na Definição 3.1 ([Laurikkala *et al.* 2000], [Hodge & Austin 2004]).

Definição 3.1 Um quartil é definido como um dos três pontos que dividem uma sequência ordenada de dados ou população em quatro partes iguais ([BusinessDictionary 2018]). O primeiro quartil Q_1 (também chamado de quartil inferior) é o número da sequência que define 25% dos dados inferiores, *i.e.*, 25% dos dados da sequência estão abaixo de Q_1. O segundo quartil Q_2 (ou ainda, a mediana) divide a sequência ordenada dos dados ao meio (50%), ou seja, metade dos valores da sequência estão abaixo de Q_2. O terceiro quartil Q_3 (também chamado de quartil superior) define 75% dos dados inferiores (abaixo de Q_3), e também, 25% dos dados superiores (de Q_3 em diante).

Exemplo 3.1: Considere inicialmente um conjunto de instâncias de dados X e que associado ao atributo A que descreve as instâncias de X, o seguinte conjunto de valores foi coletado: {17, 15, 9, 3, 12, 7, 40, 14}. O primeiro passo quando do cálculo dos quartis associados ao conjunto de valores anterior, é ordená-los, gerando assim, uma sequência ordenada de valores em ordem crescente, *i.e.*, 3, 7, 9, 12, 14, 15, 17, 40. O segundo passo consiste em identificar os cinco valores que sumarizam a sequência ordenada, como descrito a seguir:

(1) o menor valor (3);

(2) o maior valor (40);

(3) a mediana da sequência ordenada ((12+14) / 2 = 13) que equivale ao Q_2;

(4) a mediana entre o menor valor e a mediana da sequência ordenada ((7+9) / 2 = 8), que equivale ao Q_1 e;

(5) a mediana entre a mediana da sequência ordenada e o maior valor ((15+17) / 2 = 16), que equivale ao Q_3.

A identificação dos cinco valores que sumarizam a sequência ordenada, e consequentemente, dos três quartis, permite a identificação de quatro grupos na sequência ordenada:

(1) Grupo 1 (G_1 = {3, 7}), formado por valores menores que Q_1 = 8;

(2) Grupo 2 ($G_2 = \{9, 12\}$), formado por valores maiores e iguais a $Q_1 = 8$ e menores que $Q_2 = 13$;

(3) Grupo 3 ($G_3 = \{14, 15\}$), formado por valores maiores e iguais que $Q_2 = 13$ e menores que $Q_3 = 16$, e;

(4) Grupo 4 ($G_4 = \{17, 40\}$), formado por valores maiores ou iguais a $Q_3 = 16$.

O intervalo IQR (interquartil) é definido como o conjunto de valores entre os quartis Q_1 e Q_3, *i.e.*, IQR é formado por valores entre $Q_1 = 9$ e $Q_3 = 16$. O valor de IQR é a diferença entre Q_3 e Q_1, *i.e.*, $IQR = Q_3 – Q_1 = 16 – 8 = 8$.

Multiplicando o valor do IQR (= 8) por 1,5, obtém-se um valor que define limites para identificação de possíveis *outliers*, no caso, $8 \times 1{,}5 = 12$. Os valores da sequência ordenada que são menores que $(Q_1 – 12) = –4$ ou maiores que $(Q_3 + 12) = 28$ são considerados possíveis *outliers*. Nota-se que o único valor considerado um possível *outlier* no conjunto X é o valor 40, uma vez que está muito fora do padrão quando comparado aos demais valores associados ao atributo A.

O algoritmo 3.1 apresenta o pseudocódigo de um algoritmo que calcula a mediana.

```
procedure calculateMedian(ValuesSet, FromPos, ToPos)
input
ValuesSet = {x1, x2, ..., xn}      // set of values
FromPos                            // consider range from FromPos position
// range starts at FromPos position
ToPos                              // consider range up to ToPos position
// range ends at the ToPos position
output
Median                             // median of ValuesSet
01. begin
02.     newValuesSet ← getSubSetFrom(ValuesSet, FromPos,ToPos)
03.     if ( |newValuesSet| is odd ) then
04.         Median ← choose the value at position (|newValuesSet| +1)/2 in ValuesSet
05.     else
06.         v1 ← choose the value at position |newValuesSet| / 2 in newValuesSet
07.         v2 ← choose the value at position (|newValuesSet| / 2) + 1 in newValuesSet
08.         Median ← (v1 + v2) / 2
09.     end if
10.     return Median
11. end
```

Algoritmo 3.1 Pseudocódigo do algoritmo para cálculo de mediana. Produção própria.

O Algoritmo 3.2 apresenta o pseudocódigo do algoritmo utilizado para detecção de possíveis *outliers* em um dado conjunto de valores. Os passos 08 e 09 desse algoritmo fazem uso do procedimento *calculateMedian*, que calcula a mediana de um conjunto de valores e que foi apresentado em Algoritmo 3.1.

```
procedure detectOutliers(A, ValuesSet)
input
        A                               // attribute to be discretized
        ValuesSet = {x1, x2, ..., xn} // set of values associated with A
output
        Possible_outliers               // set of possible outliers
01. begin
02.     Possible_outliers ← ∅
03.     N ← |ValuesSet|
04.     sort(ValuesSet)
05.     median ← calculateMedian(ValueSet, 1, N)
06.     minValue ← lowestValue(ValueSet)
07.     maxValue ← highestValue(ValueSet)
08.     Q1 ← calculateMedian(ValueSet, 1, N/2)
09.     Q3 ← calculateMedian(ValueSet, (N/2)+1, N)
10.     IQR ← (Q3 − Q1)
11.     limit ← IQR × 1.5
12.     for each x ∈ ValuesSet
13.         if ( x < (Q1−limit) or x > (Q3+limit) ) then
14.             Possible_outliers ← Possible_outliers ∪ {x}
15.     end for each
16.     return Possible_outliers
17. end
```

Algoritmo 3.2 Pseudocódigo de algoritmo para detecção de possíveis *outliers* utilizando o conceito de quartis apresentado na Definição 3.1. Produção própria.

Mais uma vez, as alternativas para solução do problema de *outliers* nos dados são baseadas na eliminação da instância e no preenchimento/substituição do *outlier* de forma automática ou manual. A Seção 3.3 traz mais detalhes sobre as alternativas para solução de problemas nos dados.

3.3 Técnicas para Tratamento de Problemas nos Dados

Como sugerido em [Han *et al.* 2012], as técnicas de tratamento de problemas nos dados podem ser resumidas em quatro categorias:

(1) *Limpeza*: as técnicas de limpeza são utilizadas, basicamente, para preenchimento de valores ausentes, tratamento de padrões com ruído, tratar *outliers* e corrigir inconsistências diversas. Tais técnicas buscam corrigir problemas com os dados em si, ignorando outros aspectos, como origem dos dados, etc.

Entre as principais técnicas dessa categoria, afim de adequar um conjunto de instâncias para que o mesmo possa ser utilizado em processos de AM, encontram-se: a remoção da instância com ruído; a substituição do valor com ruído manualmente (processo de imputação); e a substituição do valor com ruído de forma automática por meio do uso de conceitos que podem utilizar valores mais representativos tais como a média, a mediana, ou outra medida dos valores para aquele atributo.

(2) *Integração*: como dito anteriormente, os dados disponíveis podem ter como origem diferentes fontes, o que pode causar problemas de incompatibilidade de informações que possuem a mesma representatividade, por exemplo: na base de dados B1, a altura dos indivíduos é armazenada em metros, enquanto na base de dados B, a altura dos indivíduos é armazenada em centímetros. As técnicas de integração buscam resolver esses tipos de problema encontrados nos conjuntos de dados.

(3) *Redução*: as técnicas de redução estão relacionadas àquelas que buscam minimizar a dimensionalidade e o volume dos dados com o objetivo de fornecer dados mais consistentes e objetivos que possuam a mesma representatividade dos dados originais. Essas técnicas são de extrema importância, uma vez que podem auxiliar na obtenção de algoritmos mais eficientes com base na redução de seus dados, atributos, etc.

(4) *Transformação*: tais técnicas buscam transformar os valores do atributo em questão de forma que sigam um determinado padrão de representação para ser utilizado no atributo. Técnicas de normalização (transformar valores diversos em intervalos pré-definidos), de discretização (transformar atributos contínuos em atributos discretos) e de geração de hierarquia de conceitos são os principais exemplos de técnicas de transformação de dados.

Entre as alternativas de soluções para o problema da presença de ruído nos dados, a mais simples é a eliminação das instâncias de dados afetadas por ruídos, uma vez que tal operação visa descontaminar o conjunto de dados mantendo, assim, sua representatividade. Há casos, entretanto, que a eliminação de instâncias com ruído pode diminuir consideravelmente o número de instâncias de dados do conjunto, distorcendo/desconfigurando a representatividade do conjunto original. Para esses casos, existe uma alternativa: a substituição dos valores com ruídos.

O valor com ruído pode ser substituído por valores que façam com que a representatividade do conjunto de instâncias de dados não seja radicalmente impactada. As medidas estatísticas média, mediana e desvio padrão, são algumas das técnicas que podem ser utilizadas para a substituição (preenchimento automática) de valores com ruídos em instâncias de dados. O valor mais frequente para aquele atributo em questão é uma alternativa de valor para ser utilizado na substituição de valores com ruídos em um conjunto de instâncias de dados. É possível ainda, fazer o preenchimento manual dos valores, alternativa esta que é indicada em casos específicos onde o especialista tenha completo conhecimento do domínio de dados em questão.

Por fim, a combinação de técnicas supervisionadas e não supervisionadas de AM é mais uma alternativa bastante explorada para indução de valores que possam substituir valores com ruído ou ausentes. Um exemplo dessa combinação de técnicas pode ser visto em [Lakshminarayan *et al.* 1996] em que os autores combinaram os algoritmos Autoclass (não supervisionado) e C4.5 (supervisionado) para indução do melhor valor possível para substituição de um valor ou preenchimento de um valor ausente, além de apresentar o uso individual de cada um dos algoritmos para as mesmas finalidades.

3.4 Tratamento de Atributos Contínuos

Em teoria, atributos contínuos podem ter infinitos valores associados, pois são representados, via de regra, por valores reais. Uma das possíveis maneiras de lidar com atributos contínuos em AM é por meio da realização de um processo conhecido como *discretização* dos valores.

O processo de *discretização*, também conhecido como processo de particionamento, é realizado sobre os valores de atributos contínuos e tem, como objetivo, de acordo com [Maslove *et al.* 2013], transformar um atributo contínuo em um atributo discreto (*i.e.*, encontrar padrões representativos entre os diversos valores do atributo contínuo), utilizando técnicas específicas para tal.

Uma vantagem importante, promovida pela discretização dos valores de atributos contínuos que descrevem o conjunto de treinamento, em um ambiente de AM, é a diminuição dos possíveis valores associados a cada um dos atributos que estão sendo

discretizados, fato que implica aumento na velocidade de execução de algoritmos indutivos ([Catllet 1991]).

Entre os diversos métodos propostos para implementar o processo de discretização de um atributo numérico contínuo, estão aqueles baseados na definição de um conjunto de intervalos de valores. Tal conjunto de intervalos é abordado como uma partição do intervalo original, *i.e.*, aquele intervalo definido pelo menor e maior valor (no conjunto de dados) assumidos pelo atributo sendo discretizado.

As três técnicas de particionamento do escopo de atributo utilizadas nos experimentos conduzidos e descritos no Capítulo 6, são referenciadas como:

(1) técnica de particionamento por amplitudes iguais (*Equal Width*);

(2) técnica de particionamento por frequências iguais (*Equal Frequency*), e;

(3) técnica de particionamento por entropia e MDLP ([Fayyad & Irani 1993] e [Kotsiantis & Kanellopoulos 2006]).

As técnicas (1) e (2), de particionamento por amplitudes iguais e de particionamento por frequências iguais, são consideradas técnicas não-supervisionadas de discretização, uma vez que não necessitam de conhecimento extra (valores de classes informados por terceiros e/ou especialistas) para executar a discretização dos valores do atributo contínuo. Já a técnica (3), de particionamento por entropia e MDLP, é considerada uma técnica supervisionada, uma vez que leva em consideração as classes das instâncias de dados que contém os valores do atributo em questão, para realizar o processo de discretização.

As técnicas (1) e (2) dependem da informação do valor (número inteiro) de um parâmetro, geralmente identificado por k, que representa o número de subintervalos que irão compor a partição do intervalo original associado ao atributo sendo discretizado. A técnica (3), por sua vez, pode tanto utilizar o parâmetro k fornecido pelo usuário como definir k automaticamente com base nos valores que descrevem as instâncias (incluindo também o valor da classe associada).

O valor de k, geralmente, é fornecido pelo usuário, porém há técnicas para a definição de um valor adequado para k com base na quantidade de valores do atributo contínuo. Uma das técnicas que permitem o cálculo do valor de k, encontrada em

[Dougherty *et al.* 1995], utiliza a Eq. (3.1), em que *n* representa o número de valores distintos do atributo em questão, presentes no conjunto de treinamento.

$$k = \max\{1, 2\times\log(n)\} \quad (3.1)$$

As subseções que seguem abordam, em detalhes, cada uma das três técnicas de discretização supracitadas.

3.4.1 Discretização por Particionamento do Conjunto de Valores em Intervalos de mesma Amplitude

A discretização por particionamento do conjunto de valores em intervalos de mesma amplitude identifica, em um conjunto de valores ordenados, o valor mínimo (min) e máximo (max) e, então, divide a amplitude do intervalo [min max] pelo valor do parâmetro k (geralmente informado pelo usuário), criando assim uma partição do intervalo associado ao conjunto, em k subintervalos de mesma amplitude ([Catllet 1991] e [Maslove *et al.* 2013]).

Exemplo 3.2 Considere uma situação de discretização de um atributo A cujos possíveis valores, em um determinado conjunto de treinamento, estejam no conjunto V={0,06, 0,15, 0,20, 0,25, 0,42, 0,47, 0,48, 0,54, 0,60, 0,66, 0,71, 0,83, 0,85, 0,90}, |V|=14.

O conjunto V foi apresentado com seus elementos ordenados em ordem crescente de valor, para melhor compreensão do exemplo. Caso V não fosse um conjunto com valores ordenados, seria necessário ordená-los antes da aplicação do método de particionamento por amplitudes iguais.

Como os valores min e max associados ao atributo A são, respectivamente, 0,06 e 0,90, o intervalo de valores associado ao atributo A tem como valor de amplitude 0,90–0,06 = 0,84.

O valor do parâmetro k, que representa o número de *bins* ou subintervalos em que o intervalo será particionado, é geralmente informado pelo usuário. Porém, como mencionado anteriormente, é também possível calcular o número de *bins* por meio do uso da Eq. (3.1), que foi apresentada na Seção 3.4. Usando a Eq. (3.1), e considerando o conjunto V definido anteriormente, o parâmetro k pode ser definido como k = max{1,

$2\times\log(14)\}$ = max{1, 2×1,15}= max = {1, 2,30} = 2,30. Como o valor associado a k deve ser um número inteiro, assume-se k = 2.

É necessário definir um valor limite *lim* que auxiliará na definição dos subintervalos. Seu cálculo é dado pela Eq. (3.2).

$$\text{lim} = \text{amplitude/k} \tag{3.2}$$

Considerando que amplitude = 0,84 e k = 2, o valor limite *lim* é dado por 0,84/2=0,42.

Os valores que definem os subintervalos são chamados *pontos de corte*. O número de pontos de corte que particionam o intervalo de valores é sempre igual a k−1. Como k=2, então é necessário apenas um ponto de corte para dividir os valores de V em k=2 subintervalos, cada qual com o valor limite *lim* = 0,42.

Para este método de discretização, os k−1 pontos de corte *cps* (*cutpoints*) no conjunto de valores V são dados pela Eq. (3.3), em que *menorValor(V)* representa uma função que retorna o menor valor do conjunto de valores V.

$$\text{cps} = \bigcup_{\text{i=1}}^{\text{k}-1} \text{menorValor(V)} + (\text{lim} \times \text{i}) \tag{3.3}$$

O pseudocódigo do algoritmo de discretização por particionamento do conjunto de valores em intervalos de mesma amplitude é apresentado em Algoritmo 3.3.

```
procedure equalWidthBinning(Attrib, ValuesSet, k)
input
        Attrib                          // attribute to be discretized
        ValuesSet = {x1, x2, ..., xn}   // set of values associated with attribute Atrib
        k                               // number of bins
output
        Cutpoints                       // set of values that divide the ValuesSet into k bins
01. begin
02.     if k was not informed then
03.         k = max{1, 2 × log( |ValuesSet| )}
04.     MinValue ← lowestValue(ValuesSet)
05.     MaxValue ← highestValue(ValuesSet)
06.     Lim ← (MaxValue – MinValue) / k
07.     Cutpoints ← ∅
08.     for i from 1 to k–1
09.         Cutpoints ← Cutpoints ∪ {MinValue + Lim × i}
10.     return Cutpoints
11. end
```

Algoritmo 3.3 Algoritmo de particionamento por amplitudes iguais. Produção própria.

Considerando o Algoritmo 3.3, para o conjunto de valores V, k = 2 e *lim* = 0,42, o conjunto de pontos de corte *cps* (inicializado com *cps* = ∅) é dado por

cps = cps ∪ {0,06 + (0,42 × 1) = 0,48} = ∅ ∪ {0,48} = {0,48}.

Assim, a partição P_1 do intervalo original [0,06 0,90] em k = 2 subintervalos de mesma amplitude é dada por P_1 = {[0,06 0,48), [0,48 0,90]}.

Considere o mesmo conjunto V definido anteriormente e seja k = 3 (agora informado pelo usuário). A divisão da amplitude do intervalo que define o escopo do atributo (0,84) pelo valor de k (3) resulta no valor limite *lim* = 0,28. Uma vez que k = 3, são necessários k – 1 = 2 pontos de corte nos valores para particionar o intervalo de valores relativos aos valores de V. O conjunto de pontos de corte *cps* (inicializado com cps = ∅) nos valores de V com base na no valor de *lim* = 0,28 são definidos utilizando a Eq. (3.3), como segue:

(1) cps = cps ∪ {0,06 + (0,28 × 1) = 0,34} = ∅ ∪ {0,34} = {0,34}, e;

(1) cps = cps ∪ {0,06 + (0,28 × 2) = 0,62} = {0,34} ∪ {0,62} = {0,34, 0,62}.

A partição P_2 do intervalo original [0,06 0,90] em k = 3 subintervalos de mesma amplitude é dada por P_2 = {[0.06 0,34), [0,34 0,62), [0,62 0,90]}.

Considere o conjunto V definido anteriormente e k = 4 (informado pelo usuário). O valor limite *lim* é dado por *lim* = 0,84 / 4 = 0,21. Sendo k = 4, k – 1 = 3 pontos de corte deverão ser encontrados no intervalo. É mostrado a seguir como o conjunto de pontos de corte cps (inicializado com cps = ∅) é definido, utilizando, para tal, a Eq. (3.3):

(1) cps = cps ∪ {0,06 + (0,21 × 1) = 0,27} = ∅ ∪ {0,27} = {0,27};

(2) cps = cps ∪ {0,06 + (0,21 × 2) = 0,48} = {0,27} ∪ {0,48} = {0,27, 0,48}, e;

(3) cps = cps ∪ {0,06+(0,21×3) = 0,69} = {0,27, 0,48}∪{0,69} = {0,27, 0,48, 0,69}.

A partição P_3 do intervalo original [0,06 0,90] em k = 4 subintervalos de mesma amplitude é dada por P_3 = {[0.06 0,27), [0,27 0,48), [0,48 0,69), [0,69 0,90]}.

3.4.2 Discretização por Particionamento do Conjunto de Valores por Frequências Iguais

O método de discretização por particionamento do conjunto de valores por frequências iguais define, a partir de um conjunto de valores ordenados, k subintervalos (geralmente k informado pelo usuário) que devem conter, aproximadamente, a mesma frequência ([Catllet 1991], [Dougherty *et al.* 1995] e [Dash *et al.* 2011]).

Se um atributo contínuo tem n valores e esse atributo deve ser particionado em k subintervalos, então, aproximadamente, n/k valores deverão pertencer a cada subintervalo ([Orhan *et al.* 2012] e [Dash et al. 2011]) *i.e.*, frequência = n/k.

É importante observar que nem sempre os subintervalos terão exatamente a frequência desejada, pois podem ocorrer casos em que alguns valores se repetem e, obviamente, devem pertencer ao mesmo subintervalo ([Catllet 1991], [Dougherty *et al.* 1995] e [Dash *et al.* 2011]).

Considerando um conjunto ordenado de valores, um candidato a ponto de corte existe quando o valor atual analisado é menor que o próximo valor. Tal fato não define um ponto de corte real nos valores, mas sim, aponta indícios que pode haver um ponto de corte real que, obviamente, deve ser verificado antes de ser tomado como verdadeiro.

Um ponto de corte real nos valores, por sua vez, particiona, de fato, os valores.

O método de discretização por particionamento do conjunto de valores por frequências iguais busca identificar os pontos de corte reais nos valores respeitando a frequência desejada, ou seja, um ponto de corte real nos valores só é definido caso a frequência (*i.e.*, o número de elementos) do subintervalo atual esteja sendo devidamente respeitada.

Para definir pontos de cortes reais associados aos valores de um atributo, é realizada uma varredura do conjunto de valores, analisando os candidatos a ponto de corte e avaliando a viabilidade de defini-los como pontos de corte reais.

Se um candidato a ponto de corte respeita a frequência desejada, então o mesmo pode ser definido como um ponto de corte real.

A cada definição de um ponto de corte real nos valores, a frequência é recalculada para o restante dos valores. O Exemplo 3.3 mostra detalhes do processo para três situações: k = 2, k = 3 e k = 4.

Observação 3.1 O método de discretização por particionamento do conjunto de valores por frequências iguais, descrito e exemplificado a seguir, foi completamente baseado no código-fonte do algoritmo *Equal Frequency Binning* que compõe os métodos não-supervisionados de discretização do software Weka versão 3.8 ([WEKA 2018]). Tal ação foi necessária devido ao fato de que as referências bibliográficas encontradas não possuíam detalhes sobre tal método, como, por exemplo, a simples definição do termo *frequência*. Dessa forma optou-se por apresentar o método contido no Weka por ser uma versão do método já estabelecida.

Exemplo 3.3 Considere o mesmo conjunto de valores do Exemplo 3.2, *i.e.*, V={0,06, 0,15, 0,20, 0,25, 0,42, 0,47, 0,48, 0,54, 0,60, 0,66, 0,71, 0,83, 0,85, 0,90}, composto por 14 (= |V|) valores. O valor de k pode ser informado pelo usuário ou calculado (como já mostrado no Exemplo 3.2).

Seja k = 2. Logo, é necessário encontrar k – 1 = 2 – 1 = 1 ponto de corte real nos valores de V. Considerando k = 2, a frequência inicial (inicial, pois é recalculada sempre que um ponto de corte é definido) é dada por n/k = 14/2 = 7.

Para a implementação de um procedimento que realiza tal processo de discretização, deve ser considerado um contador (inicializado com o valor 0) responsável por controlar o total de valores por subintervalo, bem como um peso (= |V| = 14), representando o total de valores ainda não considerados. O valor do contador na iteração anterior também deve ser armazenado separadamente, assim como, o último índice (posição da instância) acessado pela iteração. Deve ser considerado e gerenciado, por fim, o número de pontos de corte já encontrados.

Estes dados (variáveis) são atualizados a cada iteração, com exceção da frequência, que é atualizada somente quando um ponto de corte real nos valores é definido.

A Tabela 3.1 apresenta as iterações realizadas pelo método e os valores dos dados necessários para definição do ponto de corte para que o conjunto de valores V seja particionado em k = 2 subintervalos com frequências aproximadamente iguais.

Tabela 3.1 Valores das variáveis ao fim de cada iteração (#I) realizada pelo método de discretização por particionamento do conjunto de valores por frequências iguais, para o conjunto V e k = 2. Cada linha apresenta o valor das variáveis ao final de cada iteração.

# I	freq	$i_0, i_1, ..., i_{\|V\|-2}$	cont	peso	último índice	último	ponto de corte
1	7	0	1	13	0	1	–
2	7	1	2	12	1	2	–
3	7	2	3	11	2	3	–
4	7	3	4	10	3	4	–
5	7	4	5	9	4	5	–
6	7	5	6	8	5	6	–
7	7	6	0	7	-1	0	(0,48+0,54) / 2 = 0,51
8	7	7	1	6	7	1	–
9	7	8	2	5	8	2	–
10	7	9	3	4	9	3	–
11	7	10	4	3	10	4	–
12	7	11	5	2	11	5	–
13	7	12	6	1	12	6	–

A partir da Tabela 3.1, algumas observações são pertinentes:

(1) O termo i_k, iniciado em 0 (*i.e.*, posição 0), representa a posição do valor atual acessado pela iteração e é incrementado a cada iteração;

(2) O contador, que auxilia no gerenciamento número de valores que farão parte do subintervalo atual, é incrementado em 1 a cada iteração;

(3) No caso de definição de um ponto de corte, o contador pode ser zerado ou pode receber o valor (contador − último) justamente para representar que o próximo subintervalo possui 0 ou (contador – ultimo) valor(es). Isso acontece uma vez que o ponto de corte, que define o limite entre os subintervalos, acabou de ser encontrado e pode estar entre o valor atual e próximo valor (definindo, assim, que o valor atual pertence ao subintervalo atual) ou entre o valor anterior e o valor atual (definindo que o valor atual não pertence ao subintervalo atual, e sim, ao próximo subintervalo já iniciando, assim, a definição deste último);

(4) A cada iteração, o peso, que representa o número de valores que ainda não foram acessados pela iteração, é decrementado em 1;

(5) O último índice armazena o valor de i da iteração que está sendo finalizada, exceto quando é definido um ponto de corte, caso no qual o valor do último índice é definido como −1. O último índice é utilizado pelo algoritmo no momento da definição do ponto de corte. É fácil observar na Tabela 3.1 que os valores da coluna último índice (exceto na linha 7) são iguais aos valores de i_k de suas respectivas linhas, pois cada linha representa o valor das variáveis ao final de cada iteração;

(6) O valor da variável último, ao final de cada iteração, é sempre igual ao valor do contador da iteração atual. Assim como o valor do último índice, o valor da variável último pode ser utilizado na definição do ponto de corte;

(7) O ponto de corte foi definido na iteração número 7 quando i = 6. Naquele momento, o valor do contador era 7, ou seja, era maior ou igual ao valor da frequência (=7), *i.e.*, já haviam 7 elementos compondo o subintervalo que estava sendo gerado naquele momento. O valor da posição i = 6 (0,48) era menor que o valor de i+1 = 7 (0,54). Essas condições indicaram um possível ponto de corte real nos dados. Então, foi verificado se o ponto de corte nos valores deveria ser feito entre o valor da posição i e i+1, ou, entre o valor do último índice e último índice+1. Essa decisão depende do valor da frequência:

- se o valor arredondado da frequência é menor que a frequência, então o ponto de corte é definido entre os valores do último índice e último índice+1;
- caso contrário, o ponto de corte é definido entre os valores de i e i+1;

(8) A definição do ponto de corte na iteração 7 influenciou na alteração de algumas variáveis (último índice, contador e, consequentemente, último) de forma diferente das 6 iterações anteriores, além de influenciar a alteração do valor da frequência. Isso ocorre, pois como foi definido um ponto de corte real nos valores e o subintervalo atual possui 7 (= contador) valores, a nova frequência deve considerar apenas os 7 (= peso) valores ainda não acessados pela iteração, e por isso, tem seu valor recalculado/atualizado;

(9) Da iteração 8 em diante o método continua seguindo o padrão procurando os melhores pontos de corte reais nos dados respeitando a frequência desejada;

Ao término das iterações do método de discretização por particionamento do conjunto de valores por frequências iguais, a partição P_4 dos valores de V particionados em k = 2 subintervalos de iguais frequências é dada por P_4={[0,06 0,51), [0,51 0,90]}.

A Tabela 3.2 apresenta as iterações e atualizações de informações (variáveis) para o mesmo conjunto de valores V, porém agora, considerando k = 3.

Tabela 3.2 Valores das informações ao fim de cada iteração (#I) realizada pelo método de discretização por particionamento do conjunto de valores por frequências iguais para o conjunto de valores V e k = 3.

# I	freq	$i_0, i_1, ..., i_{\|V\|-2}$	cont	peso	último índice	último	ponto de corte
1	4,67	0	1	13	0	1	–
2	4,67	1	2	12	1	2	–
3	4,67	2	3	11	2	3	–
4	4,67	3	4	10	3	4	–
5	4,5	4	0	9	–1	0	(0,42 + 0,47) / 2 = 0,44
6	4,5	5	1	8	5	1	–
7	4,5	6	2	7	6	2	–
8	4,5	7	3	6	7	3	–
9	4,5	8	4	5	8	4	–
10	4	9	0	4	–1	0	(0,66 + 0,71) / 2 = 0,69
11	7	10	1	3	10	1	–
12	7	11	2	2	11	2	–
13	7	12	3	1	12	3	–

A partição P_5 dos valores de V em k = 3 subintervalos obtida, de acordo com os pontos de corte da Tabela 3.2, é dada por P_5={[0,06 0,44), [0,44 0,69), [0,69 0,90]}.

A Tabela 3.3, por sua vez, apresenta as iterações e atualizações de informações (variáveis) para o conhecido conjunto de valores V, no caso do número de subintervalos desejados k ser igual a 4.

Tabela 3.3 Valores das informações ao fim de cada iteração (#I) realizada pelo método de discretização por particionamento do conjunto de valores por frequências iguais para o conjunto de valores V e k = 4.

#I	freq	$i_0, i_1, ..., i_{\|V\|-2}$	cont	peso	último índice	último	ponto de corte
1	3,5	0	1	13	0	1	–
2	3,5	1	2	12	1	2	–
3	3,5	2	3	11	2	3	–
4	3,33	3	0	10	–1	0	(0,25+0,42) / 2 = 0,33
5	3,33	4	1	9	4	1	–
6	3,33	5	2	8	5	2	–
7	3,33	6	3	7	6	3	–
8	3,5	7	1	6	7	1	(0,48+0,54) / 2 = 0,51
9	3,5	8	2	5	8	2	–
10	3,5	9	3	4	9	3	–
11	3	10	0	3	–1	0	(0,71+0,83) / 2 = 0,77
12	3	11	1	2	11	1	–
13	3	12	2	1	12	2	–

A partição P_6 dos valores de V em k = 4 subintervalos é dada por P_5={[0,06 0,33), [0,33 0,51), [0,51 0,77), [0,77 0,90]}.

O pseudocódigo do método abordado nesta seção pode ser visto no Algoritmo 3.4.

É importante ressaltar que, o algoritmo apresentado no Algoritmo 3.4 não faz verificações na relação número de *bins*-quantidade de valores, ou seja, não é sempre garantido que o algoritmo seja capaz de particionar os valores na quantidade desejada de *bins*, pois pode não ser possível encontrar k *bins* para n valores (*e.g.*, particionar 10 valores em 2.000 *bins*).

```
procedure equalFrequencyBinning(A, ValuesSet, k)
input
        A                                  // attribute to be discretized
        ValuesSet = { x1, x2, ..., xn }    // set of values associated with attribute Atrib
        k                                  // bins number
output
        Cutpoints                          //set of  values that divide the ValuesSet into k bins
01. begin
02.     Cutpoints ← ∅
03.     Frequency ← |ValuesSet| / k
04.     sort(ValuesSet)
05.     Counter ← 0
06.     for each x ∈ (ValuesSet – {last x ∈ ValuesSet })
07.         Counter ← Counter + 1
08.         if x < next x∈ValuesSet then
09.             if Counter ≥ Frequency then
10.                 if rounded(Frequency) < Frequency then
11.                     Cutpoints ← Cutpoints ∪ { (x + last x) / 2 }
12.                     Counter ← 0
13.                 else
14.                     Cutpoints ← Cutpoints ∪ { (x + next x) / 2 }
15.                     Counter ← 1
16.                 end if
17.                 update other vars
18.                 update Frequency // (to remaining values x∈ValuesSet)
19.             else
20.                 update other vars
21.             end if
22.         end if
23.     end for
24.     return Cutpoints
25. end
```

Algoritmo 3.4 Pseudocódigo do algoritmo de particionamento do escopo de atributo em intervalos com (aproximadamente) a mesma frequência.

3.4.3 Discretização por Particionamento do Conjunto de Valores por Entropia e MDLP

O método de discretização por particionamento do conjunto de valores por entropia e MDLP (*Minimum Description Length Principle*) ([Rissanem 1978]), proposto em [Fayyad & Irani 1993], é um método supervisionado (*i.e.*, depende das classes associadas às instâncias de treinamento) e recursivo, que utiliza entropia para identificar possíveis pontos de corte no conjunto de valores.

Sempre que um possível ponto de corte é identificado em um conjunto de valores S, ele é avaliado por um método baseado no princípio MDL, o qual decide se o possível ponto de corte será aceito ou rejeitado. De acordo com [Fayyad & Irani 1993], "o tamanho de descrição mínima de um objeto é definido pelo número mínimo de bits necessários para especificar de forma única aquele objeto em relação ao universo de objetos".

Para cada ponto de corte aceito pelo princípio MDL no conjunto de valores S, uma partição binária no conjunto é induzida e o método é chamado recursivamente para os dois subconjuntos gerados. Se um possível ponto de corte é rejeitado pelo princípio MDL, o método é interrompido.

Definição 3.2 Uma *partição binária* é uma partição induzida sobre um conjunto de valores S, tal que sejam gerados dois subconjuntos, S_1 e S_2.

Naturalmente, para ocorrer uma partição binária em um conjunto de valores S, é necessário haver um, e somente um, valor que irá induzir a partição binária. Esse valor é chamado de ponto de corte.

Considerando um conjunto S de valores ordenados com n = |S| valores, qualquer valor existente entre dois valores, $v_1 \in S$ e $v_2 \in S$, pode particionar de forma binária o conjunto de valores S. Dessa forma, é possível obter n−1 partições diferentes para S.

Porém, tratando-se de AM, e ainda mais de atributos contínuos, sabe-se que n pode ser um número grande, o que pode prejudicar a performance dos algoritmos com complexidade O(n). Logo, seria inviável para o método de particionamento por entropia e MDL analisar todos os n−1 pontos que podem particionar de forma binária o conjunto de valores ordenados S.

Definição 3.3 Um *ponto de fronteira* T é um valor definido entre dois valores de um conjunto ordenado de valores S quando existem dois exemplos $e_1 \in S$ e $e_2 \in S$ associados a classes distintas, tal que $e_1 < T < e_2$ e não existe qualquer outro exemplo $e' \in S$ tal que $e_1 < e' < e_2$ ([Fayyad & Irani 1993]).

O método de discretização por particionamento do conjunto de valores por entropia e MDLP considera somente pontos de fronteira do conjunto de valores ordenados S. Tal ação se dá pelo fato de que pontos existentes entre valores associados a uma mesma classe não colaboram muito com o objetivo do método e são considerados pontos "ruins" para particionar o conjunto S, uma vez que os pontos de corte reais dos valores devem ser definidos nas fronteiras que separam classes distintas (como discutido em [Fayyad & Irani 1993]). Esta ação colabora, portanto, com a melhoria do desempenho do método como um todo, pois são avaliados apenas pontos que realmente tenham chances de serem definidos como pontos de corte reais nos valores de S.

Para melhor compreensão do método de discretização por particionamento do conjunto de valores por entropia e MDLP, se faz necessário definir o conceito de *entropia*.

Definição 3.4 A *entropia* é uma medida que mede o quão disperso (ou impuro) é um conjunto de valores S, com base na relação entre as instâncias e suas classes associadas. Quanto menor for o valor da entropia, menos disperso (*i.e.*, menos impuro) é o conjunto de valores S.

O cálculo do valor da entropia de um conjunto de valores S pode ser obtido, de acordo com [Fayyad & Irani 1993], pela Eq. (3.4), em que C_i representa o conjunto composto pelas diferentes classes associadas aos valores de S e $P(C_i,S)$ representa a proporção entre o número de valores que possuem a classe i associada e o número de valores pertencentes à S (*i.e.*, $P(C_i, S) = |C_i| \,/\, |S|$).

$$\mathrm{Ent(S)} = -\sum_{i=1}^{k} P(C_i, S)\log\big(P(C_i, S)\big) \tag{3.4}$$

Na Eq. (3.4) o logaritmo na base 2 indica que a entropia mensura a quantidade de informação necessária, em *bits*, para especificar as classes do conjunto de valores S.

Exemplo 3.4 Seja V_1 um conjunto de 10 valores e todos eles associados à uma única classe C. A expressão a seguir calcula o valor da entropia para o conjunto V_1.

$$\mathrm{Ent}(V_1) = -\frac{10}{10}\log_2\frac{10}{10} = 1 \times 0 = 0$$

O conjunto V_1, cujos valores estão todos associados à uma única classe C, é considerado um conjunto puro, uma vez que o valor da entropia a ele associado é 0 (que vem a ser o menor valor possível para entropia). O valor de entropia 0 associado ao conjunto V_1 representa o fato que o conjunto está perfeitamente organizado (no caso, todos os seus elementos pertencem à uma única classe (C)).

Seja um conjunto de valores V_2 também composto por 10 valores. Desses 10 valores, 5 estão associados à classe C_1 e os outros 5 estão associados à classe C_2. A expressão a seguir calcula o valor da entropia para o conjunto de valores V_2.

$$\mathrm{Ent}(V_2) = -\left(\frac{5}{10}\log_2\frac{5}{10}\right) - \left(\frac{5}{10}\log_2\frac{5}{10}\right) = 0{,}5 + 0{,}5 = 1$$

Neste caso, a entropia resulta em 1, uma vez que a relação valores-classe está equilibrada (50% dos valores associados a cada classe disponível). Em outras palavras, o conjunto V_2 é um conjunto impuro (entropia > 0), pois não é possível classificar todos seus valores como pertencentes à uma única classe dentre as classes presentes.

Seja um conjunto de valores V_3 composto, novamente, por 10 valores. Dos 10 valores, 1 está associado à classe C_1, 2 estão associados à classe C_2 e os outros 7 estão associados à classe C_3. Observando os valores e as relações valor-classe do conjunto V_3, é possível notar que o mesmo tem grandes chances de ser um conjunto impuro, fato que pode ser verificado aplicando a Eq. (3.4), para cálculo de entropia para o conjunto V_3.

$$Ent(V_3) = -\left(\frac{1}{10}\log_2\frac{1}{10}\right) - \left(\frac{2}{10}\log_2\frac{2}{10}\right) - \left(\frac{7}{10}\log_2\frac{7}{10}\right) \cong 1{,}15$$

Cada ponto de fronteira, encontrado pelo método de discretização por particionamento do conjunto de valores por entropia e MDLP em um conjunto de valores ordenado S, é analisado para verificar se o mesmo é um candidato a ponto de corte real nos valores de S.

Definição 3.5 A *entropia de informação de classe* refere-se ao valor da entropia calculada sobre um conjunto de valores S particionado por um determinado ponto de fronteira pf. Para seu cálculo, é considerada, tanto a entropia do subconjunto de valores composto por valores menores e iguais ao ponto de fronteira (S_1), como a entropia do conjunto de valores maiores que o ponto de fronteira (S_2). A Eq. (3.5) ([Fayyad & Irani 1993]) pode ser utilizada para cálculo da entropia de informação de classe.

$$E(A, pf, S) = \frac{|S_1|}{|S|}\text{Ent}(S_1) + \frac{|S_2|}{|S|}\text{Ent}(S_2) \tag{3.5}$$

Na Eq. (3.5), a seguinte notação é utilizada:

(1) E(A, pf, S) representa o valor da entropia de informação de classe para o atributo A (que está sendo discretizado), considerando o possível ponto de fronteira pf (encontrado pelo método) sobre o conjunto de valores S;

(2) S representa o conjunto original ordenado de valores;

(3) S_1 e S_2 representam, respectivamente, o subconjunto de valores que são menores ou iguais ao ponto de corte pf e o subconjunto de valores maiores que pf;

(4) $\frac{|S_1|}{|S|}$ e $\frac{|S_2|}{|S|}$ representam a proporção dos valores de S_1 em S e de S_2 em S, respectivamente, e;

(5) Ent(S_1) e Ent(S_2) representam, respectivamente, a entropia de classe do subconjunto de valores S_1 e a entropia de classe do subconjunto de valores S_2.

Exemplo 3.5 Considere o conjunto de valores V_4, já ordenado e composto por 10 valores, apresentado na Tabela 3.4.

Tabela 3.4 Conjunto V_4, composto pelo atributo x (o qual possui 10 valores em ordem crescente) e a classe associada a cada uma das instâncias de dados.

x	classe
1.1	A
1.1	A
1.1	A
1.4	A
1.4	A
1.4	A
1.4	A
1.5	A
1.5	A
1.5	A

A entropia de classe do conjunto de valores V_4 como um todo é dada por:

$$\mathrm{Ent}(V_4) = -\frac{10}{10}\log_2\frac{10}{10} = 1 \times 0 = 0$$

Observa-se, portanto, que V_4 é um conjunto de valores puro (entropia = 0), *i.e.*, todas as instâncias do mesmo podem ser classificadas como pertencentes à uma mesma classe (no caso, a classe A). Neste caso, não é possível (e nem seria viável) encontrar um ponto de fronteira pf para particionar os valores utilizando entropia, uma vez que seria desnecessário particionar valores que estão associados a uma mesma classe.

Considere, agora, o conjunto ordenado de valores V_5, composto por 10 valores, apresentado na Tabela 3.5.

Tabela 3.5 Conjunto de valores V_5, composto pelo atributo x (com 10 valores em ordem crescente) e a classe associada a cada uma das instâncias de dados.

x	classe
1.1	A
1.1	A
1.1	A
1.4	A
1.4	A
1.4	A
1.4	A
1.5	B
1.5	B
1.5	B

A entropia de classe do conjunto de valores V_5 como um todo é dada por:

$$\mathrm{Ent}(V_5) = -\frac{7}{10}\log_2\frac{7}{10} - \frac{3}{10}\log_2\frac{3}{10} \cong 0{,}88$$

Observa-se, portanto, que o conjunto de valores V_5 não é um conjunto puro (entropia > 0), *i.e.*, nem todas as instâncias do mesmo podem ser classificadas como pertencentes à uma única classe. Neste caso, é possível e viável particionar os valores por entropia em dois subintervalos de valores.

O único ponto de fronteira existente no conjunto de valores V_5 encontra-se entre os valores 1,4 e 1,5, uma vez que 1,4 está associado à classe A e 1,5 à classe B.

O ponto de fronteira encontrado entre 1,4 e 1,5 é dado por (1,4 + 1,5) / 2 = 1,45, o qual particiona V_5 em 2 subconjuntos: V_1 = {1.1, 1.1, 1.1, 1.4, 1.4, 1.4, 1.4} e V_2 = {1.5, 1.5, 1.5}. No que segue, são calculadas separadamente as entropias dos subconjuntos V_1 e V_2 da partição $\{V_1, V_2\}$ induzida pelo ponto de fronteira 1.45.

$$\mathrm{Ent}(V_1) = -\frac{7}{7}\log_2\frac{7}{7} = 0, \text{ e}$$

$$\mathrm{Ent}(V_2) = -\frac{3}{3}\log_2\frac{3}{3} = 0.$$

O ponto de corte 1,45 particiona os valores de V_5 em 2 subconjuntos de valores, V_1 e V_2, de forma que a entropia de cada um dos subconjuntos é 0, ou seja, ambos subconjuntos são puros. Isso significa que todos os valores de V_1 podem ser classificados como pertencentes à uma única classe (A) assim como todos os valores de V_2 podem ser classificados como pertencentes à uma única classe (B).

A entropia de informação de classe, considerando o ponto de corte 1,45 em V_5, é dada por:

$$\mathrm{E}(x, 1{,}45, V_5) = \frac{7}{10} \times 0 + \frac{3}{10} \times 0 = 0.$$

De acordo com a entropia de informação de classe calculada acima, o ponto de fronteira 1,45 em V_5 gera uma partição, composta por 2 subintervalos, cuja entropia é 0.

Em um conjunto de valores S, podem haver diversos pontos de fronteira. Porém, como o método de discretização por particionamento do conjunto de valores por entropia e MDLP busca realizar uma partição binária sobre S, somente um ponto de fronteira pode ser escolhido e definido como um possível ponto de corte nos valores de S.

Para definir qual ponto de fronteira será realmente adotado como melhor candidato a ponto de corte para particionar os valores de S, é utilizada uma outra medida conhecida como *ganho de informação* (Definição 3.6) e calculada pela Eq. (3.6).

Definição 3.6 O *ganho de informação* é uma medida que representa o ganho obtido no caso da escolha de um determinado ponto de corte nos valores. Quanto maior o ganho de informação, melhor o ponto de fronteira particiona os valores do conjunto.

$$\text{Ganho}(\text{A}, \text{pf}, \text{S}) = \text{Ent}(\text{S}) - \text{E}(\text{A}, \text{pf}, \text{S}) \tag{3.6}$$

Na Eq. (3.6):

(1) Ganho(A,pf,S) representa o ganho de informação com a escolha do ponto de fronteira pf para o atributo A do conjunto de valores S;

(2) Ent(S) representa a entropia do conjunto de valores ordenados S como um todo, e;

(3) E(A,pf,S) representa o valor da entropia de informação de classe com a escolha do ponto de fronteira pf para o atributo A do conjunto de valores S.

Exemplo 3.6 Considere, novamente, o conjunto V_5 com Ent(V_5) = 0,88. Considere E(x, 1,45, V_5) = 0 para o ponto de corte 1,45. O ganho de informação com o ponto de fronteira pf = 1,45 é dado por:

$$\text{Ganho}(\text{x}, 1{,}45, \text{V}_5) = 0{,}88 - 0 = 0{,}82.$$

O maior ganho de informação (0,82) é o ganho obtido ao particionar o conjunto de valores V_5 utilizando o (único) ponto de fronteira pf = 1,45. Se houvessem dois ou mais pontos de fronteira, aquele com maior ganho de informação associado seria definido como um candidato a ponto de corte real para particionar o conjunto de valores V_5.

Como o maior ganho é o que melhor particiona os valores (Definição 3.6), então o ponto de fronteira pf = 1,45 deve ser definido como um candidato a ponto de corte no conjunto de valores V_5, uma vez que particionaria o conjunto de valores V_5 em 2 subintervalos mantendo uma baixa entropia geral para a partição induzida.

A partição P_1 de V_5, que pode ser induzida pelo candidato a ponto de corte (e também, ponto de fronteira) pf = 1,45, é dada por P_1 = {[1,1 1,45), [1,45 1,5]}.

Porém, é importante notar que o ponto de fronteira pf = 1,45 é um *candidato* a ponto de corte, e não ainda, um ponto de corte propriamente dito. Para se tornar um ponto de corte propriamente dito, ele precisa ser aceito pelo princípio MDL.

Definição 3.7 Um *ponto de corte pc* em um conjunto de valores ordenados S é o ponto de fronteira que melhor particiona S de forma binária e tenha sido aceito pelo princípio MDL.

O método de discretização por particionamento do conjunto de valores por entropia e MDLP encontra o ponto de fronteira pf que melhor particiona o conjunto de valores ordenados S de forma binária. Esse ponto de fronteira pf encontrado é considerado um candidato a ponto de corte. Se o ponto de fronteira for aceito pelo princípio MDL, ele torna-se um ponto de corte pc em S, o qual induz uma partição binária dos valores. A partir deste ponto, são realizadas chamadas recursivas para particionar de forma binária os subconjuntos S_1 e S_2, e assim sucessivamente, até que o critério de parada seja satisfeito. Podem ser utilizados como critério de parada para o método recursivo, o princípio MDL (Definição 3.8) ou um número de *bins* (informado pelo usuário).

Naturalmente, a utilização do número de *bins* como critério de parada é simples: quando k (informado pelo usuário) subintervalos forem obtidos, o método recursivo é interrompido. Porém, tal critério de parada não garante a melhor partição possível dos valores, uma vez que o usuário pode, por exemplo, informar 3 *bins* (subintervalos) como critério de parada, sendo que os valores poderiam ser particionados em muito mais *bins* para atender à realidade presente no conjunto de valores.

Definição 3.8 O princípio *MDL (Minimum Description Length)* tem como função avaliar, dentre as hipóteses disponíveis, qual hipótese especifica um determinado objeto com o menor número de *bits* possíveis ([Fayyad & Irani, 1993]).

A cada definição de ponto de corte no conjunto de valores por meio do uso do conceito de ganho de informação, é verificado se o algoritmo deve aceitar aquele novo ponto de corte, ou se o particionamento atual dos valores (sem o novo ponto de corte) é o que melhor descreve o conjunto de dados, usando para tal, o princípio MDL.

Para ser possível aceitar ou rejeitar um ponto de fronteira pf nos valores de um conjunto S, como apresentado em [Fayyad & Irani 1993], é necessário, antes, calcular o valor de $\Delta(A, pf, S)$, dado pela Eq. (3.7).

$$\Delta(A, pf, S) = \log_2(3^k - 2) - (k\,Ent(S) - k_1\,Ent(S_1) - k_2\,Ent(S_2)). \quad (3.7)$$

Na Eq. (3.7):

(1) k representa o número de classes distintas encontradas no conjunto de valores, e;

(2) k_1 e k_2 representam os números de classes distintas, respectivamente, encontradas nos subconjuntos de valores S_1 e S_2.

De acordo com [Fayyad & Irani 1993], e levando em consideração a Eq. (3.7), o ponto de fronteira pf deve ser aceito se a desigualdade mostrada na Eq. (3.8) for satisfeita, e rejeitado, caso contrário.

$$Ganho(A, pf, S) > \frac{\log_2(N-1)}{N} + \frac{\Delta(A, pf, S)}{N}, \text{em que } N = |S| \quad (3.8)$$

Basicamente, o método de discretização por particionamento do conjunto de valores por entropia e MDLP verifica qual ponto de fronteira particiona o conjunto de valores de forma a obter a menor entropia nos subconjuntos de valores a serem gerados. Quando encontrado este ponto de fronteira, o mesmo é avaliado por meio do princípio MDL, o qual irá aceitá-lo ou rejeitá-lo. Em caso de rejeição, o método é finalizado. Em caso de aceite, o ponto de fronteira pf é definido como ponto de corte pc e o método é chamado novamente, de forma recursiva, para cada um dos subconjuntos induzidos pelo

ponto de corte pc, e assim, sucessivamente, até que o critério MDL avalie que não é mais viável particionar os valores, pois a descrição mínima dos subconjuntos foi obtida.

O Exemplo 3.7 apresenta uma situação na qual o princípio MDL aceita o primeiro ponto de fronteira no conjunto de valores, porém rejeita os outros, encerrando, assim, o método de particionamento.

Exemplo 3.7 Considere:

(1) o conjunto de valores V_5 definido anteriormente;

(2) a entropia de V_5 dada por $Ent(V_5) \cong 0{,}88$;

(3) o ponto de fronteira pf = 1,45 identificado no conjunto por melhor particionar os valores do mesmo;

(4) as entropias dos subconjuntos S_1 e S_2, dadas, respectivamente, por $Ent(S_1) = 0$ e $Ent(S_2) = 0$, no caso de particionar V_5 utilizando o ponto de fronteira pf = 1,45, e;

(5) o ganho de informação dado por $\mathrm{Ganho}(x, 1{,}45, V_5) = 0{,}82$.

Como já citado, o ponto de fronteira 1,45 é o que melhor particiona os valores de V_5. Porém, é necessário verificar se ele é aceito pelo princípio MDL como um ponto de corte real no conjunto. Para tal, é necessário calcular o valor de $\Delta(x, 1{,}45, V_5)$, o qual é dado por:

$$\Delta(x, 1{,}45, V_5) = \log_2(3^2 - 2) - (2 \times 0{,}88 - 1 \times 0 - 1 \times 0) \cong 1{,}05.$$

E, por fim, basta verificar se:

$$\mathrm{Ganho}(x, 1{,}45, V_5) > \frac{\log_2 N - 1}{N} + \frac{\Delta(A, pf, S)}{N} \rightarrow 0{,}82 > 0{,}42.$$

Como 0,82 > 0,42, então o ponto de fronteira pf = 1,45 é aceito pelo princípio MDL tornando-se assim um ponto de corte pc = 1,45 em V_5, e para cada subconjunto S_1 e S_2 induzido por pc, o método de discretização por particionamento do conjunto de valores por entropia e MDLP é executado recursivamente. Porém, como $Ent(S_1) = 0$ e $Ent(S_2) = 0$ não há como definir pontos de fronteira em S_1 nem em S_2, o que provoca o término do algoritmo.

O Exemplo 3.8 apresenta uma nova situação em que são necessários 3 pontos de corte para particionar o conjunto de valores adequadamente utilizando o método proposto em [Fayyad & Irani 1993] e apresentado nesta seção.

Exemplo 3.8 Considere o conjunto de valores ordenados V_6 da Tabela 3.6.

Tabela 3.6 Conjunto de valores ordenados V_6 composto por 10 valores ordenados e suas respectivas classes associadas.

x	classe
1.05	A
1.17	B
1.17	B
1.42	B
1.42	B
1.50	C
1.57	C
1.57	C
1.58	D
1.80	D

O conjunto V_6 é um conjunto impuro, uma vez que $Ent(V_6) > 0$, como mostra o seu cálculo a seguir:

$$\mathrm{Ent}(V_6) = -\frac{1}{10}\log_2\frac{1}{10} - \frac{4}{10}\log_2\frac{4}{10} - \frac{3}{10}\log_2\frac{3}{10} - \frac{2}{10}\log_2\frac{2}{10} \cong 1{,}85.$$

No que segue, todos os possíveis pontos de fronteira de V_6 são verificados. Da mesma forma, são calculadas as entropias dos subconjuntos S_1 e S_2, a entropia de informação de classe e, também, o ganho de informação:

(1) para pf = (1,05 + 1,17) / 2 = 1,11

$$\mathrm{Ent}(S_1) = -\frac{1}{1}\log_2\frac{1}{1} = 0$$

$$\mathrm{Ent}(S_2) = -\frac{4}{9}\log_2\frac{4}{9} - \frac{3}{9}\log_2\frac{3}{9} - \frac{2}{9}\log_2\frac{2}{9} \cong 1{,}53$$

$$\mathrm{E}(x, 1{,}11, V_6) = \frac{1}{10} \times 0 + \frac{9}{10} \times 1{,}53 \cong 1{,}38$$

$$\mathrm{Ganho}(x, 1{,}11, V_6) = 1{,}85 - 1{,}38 = 0{,}47$$

(2) para pf = (1,42 + 1,50) / 2 = 1,46

$$\mathrm{Ent}(S_1) = -\frac{1}{5}\log_2\frac{1}{5} - \frac{4}{5}\log_2\frac{4}{5} \cong 0{,}72$$

$$\mathrm{Ent}(S_2) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} \cong 0{,}97$$

$$\mathrm{E}(x, 1{,}46, V_6) = \frac{5}{10} \times 0{,}72 + \frac{5}{10} \times 0{,}97 \cong 0{,}85$$

$$\mathrm{Ganho}(x, 1{,}46, V_6) = 1{,}85 - 0{,}85 = 1$$

(3) para pf = (1,57 + 1,58) / 2 = 1,575

$$\mathrm{Ent}(S_1) = -\frac{1}{8}\log_2\frac{1}{8} - \frac{4}{8}\log_2\frac{4}{8} - \frac{3}{8}\log_2\frac{3}{8} \cong 1{,}41$$

$$\mathrm{Ent}(S_2) = -\frac{2}{2}\log_2\frac{2}{2} = 0$$

$$\mathrm{E}(x, 1{,}575, V_6) = \frac{8}{10} \times 1{,}41 + \frac{2}{10} \times 0 \cong 1{,}13$$

$$\mathrm{Ganho}(x, 1{,}575, V_6) = 1{,}85 - 1{,}25 = 0{,}60$$

O melhor ponto de fronteira pf para um conjunto de valores S é aquele que proporciona o maior ganho de informação (Definição 3.6). No exemplo, o melhor ponto de fronteira para o conjunto de valores V_6 seria o ponto de fronteira pf=1,46, uma vez que $\mathrm{Ganho}(x, 1{,}46, V_6) = 1$ é o maior ganho de informação dentre os ganhos calculados. É preciso, entretanto, verificar se o método deve aceitar ou rejeitar tal ponto de fronteira utilizando o princípio MDL. Para isso é necessário calcular o valor de $\Delta(x, 1{,}46, V_6)$, o qual é dado por:

$$\Delta(x, 1{,}46, V_6) = \log_2(3^4 - 2) - (4 \times 1{,}85 - 2 \times 0{,}72 - 2 \times 0{,}97) \cong 2{,}28.$$

E, por fim, basta verificar se:

$$\mathrm{Ganho}(x, 1{,}46, V_5) > \frac{\log_2 N - 1}{N} + \frac{\Delta(A, pf, S)}{N} \rightarrow 1 > 0{,}86.$$

Como 1 > 0,86, o ponto de fronteira pf = 1,46 é aceito e, assim, definido como ponto de corte pc = 1,46 no conjunto de valores V_6. Assim, o conjunto de valores V_6 é

particionado induzindo dois subintervalos, os quais compreendem, respectivamente, os subconjuntos V_{61}={1.05, 1.17, 1.17, 1.42 1.42} e V_{62}={1.50, 1.57, 1.57 1.58, 1.80}. Na sequência, o método é aplicado recursivamente para os subconjuntos de valores V_{61} e S_{62}.

A entropia do subconjunto V_{61}, Ent(V_{61}), é dada por:

$$\text{Ent}(V_{61}) = -\frac{1}{5}\log_2\frac{1}{5} - \frac{4}{5}\log_2\frac{4}{5} \cong 0{,}72$$

O conjunto de valores V_{61} possui um único possível ponto de corte, o qual é avaliado a seguir:

(1) para pf = (1,05 + 1,17) / 2 = 1,11

$$\text{Ent}(S_1) = -\frac{1}{1}\log_2\frac{1}{1} = 0$$

$$\text{Ent}(S_2) = -\frac{4}{4}\log_2\frac{4}{4} = 0$$

$$\text{E}(x, 1{,}11, V_{61}) = \frac{1}{5} \times 0 + \frac{4}{5} \times 0 = 0$$

$$\text{Ganho}(x, 1{,}11, V_{61}) = 0{,}72 - 0 = 0{,}72$$

Como cp = 1,11 é o único ponto de fronteira, o maior ganho de informação obtido no conjunto de valores V_{61} é justamente o ganho de informação 0,72 baseado no mesmo. Na sequência é necessário verificar se o MDL aceita o ponto de fronteira pf = 1,11. Para tal, é calculado o valor de $\Delta(x, 1{,}11, V_{61})$, como segue.

$$\Delta(x, 1{,}11, V_{61}) = \log_2(3^2 - 2) - (2 \times 0{,}72 - 1 \times 0 - 1 \times 0) \cong 1{,}36$$

E, por fim, verifica-se se:

$$\text{Ganho}(x, 1{,}11, V_{61}) > \frac{\log_2(N-1)}{N} + \frac{\Delta(A, pf, S)}{N} \rightarrow 0{,}72 > 0{,}67$$

Como 0,72 > 0,67, o ponto de fronteira pf = 1,11 é aceito e definido com ponto de corte cp = 1,11, particionando o subconjunto V_{61} em dois subconjuntos V_{611} e V_{612}. Neste caso, o método é recursivamente acionado para os novos subconjuntos V_{611} e V_{612}.

Vale adiantar que, como cada um dos subconjuntos possui valores associados a uma única classe, o método será interrompido e não haverá mais chamadas recursivas para estes subconjuntos de valores. Resta, portanto, o processamento do subconjunto V_{62}.

O cálculo da entropia do subconjunto V_{62}={1.50, 1.57, 1.57 1.58, 1.80} é mostrado na sequência.

$$\mathrm{Ent}(V_{62}) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0{,}97$$

O único possível ponto de fronteira em V_{62} que é avaliado pelo método em questão é o ponto de fronteira dado encontrado entre os valores 1,57 e 1,58:

(1) para pf = (1,57 + 1,58) / 2 = 1,575

$$Ent(S_1) = -\frac{1}{1}\log_2\frac{1}{1} = 0$$

$$Ent(S_2) = -\frac{4}{4}\log_2\frac{4}{4} = 0$$

$$E(x, 1{,}68, V_{61}) = \frac{3}{5}\times 0 + \frac{2}{5}\times 0 = 0$$

$$Ganho(x, 1{,}575, V_{61}) = 0{,}97 - 0 = 0{,}97$$

O ponto de fronteira pf = 1,575 é o único ponto de fronteira do subconjunto V_{62}, logo, o maior ganho de informação obtido no conjunto de valores V_{62} é justamente o ganho de informação 0,97 baseado no pf = 1,575. Para saber se o princípio MDL aceita o ponto de fronteira pf = 1,575, é calculado $\Delta(\mathrm{x}, 1{,}575, V_{62})$ como segue.

$$\Delta(\mathrm{x}, 1{,}575, \mathrm{V}_{62}) = \log_2(3^2 - 2) - (2 \times 0{,}97 - 1 \times 0 - 1 \times 0) \cong 0{,}87.$$

E, por fim, verifica-se se:

$$\mathrm{Ganho}(\mathrm{x}, 1{,}575, \mathrm{V}_{62}) > \frac{\log_2 \mathrm{N}-1}{\mathrm{N}} + \frac{\Delta(\mathrm{A,pf,S})}{\mathrm{N}} \rightarrow 0{,}97 > 0{,}57.$$

Novamente, como 0,97 > 0,57, o ponto de fronteira pf = 1,575 é aceito e definido como ponto de corte pc = 1,575, particionando o subconjunto V_{62} em dois subconjuntos V_{621} e V_{622}. Neste caso, o método é recursivamente acionado para os novos subconjuntos

V_{621} e V_{622}. Novamente, como cada um dos subconjuntos possui valores associados a uma única classe, o método será finalizado e não haverá mais chamadas recursivas para estes subconjuntos de valores.

Não havendo mais possíveis pontos de fronteira a avaliar, e nem subconjuntos a particionar, o método de discretização por particionamento do conjunto de valores por entropia e MDLP, proposto em [Fayyad & Irani 1993], é finalizado.

3.5 Considerações Finais

Este capítulo abordou os conceitos de ruídos, *outliers* e valores ausentes, problemas que são comumente encontrados em conjuntos de instâncias de dados. Alguns desafios impostos pelos ruídos, *outliers* e valores ausentes nas instâncias dos dados foram comentados e alguns exemplos foram apresentados para facilitar a compreensão do motivo pelo qual esses conceitos são tão importantes e devem ser levados em consideração cuidadosamente. Alternativas para detecção dos problemas nos dados foram também apresentadas, assim como, métodos/técnicas para resolução/diminuição dos mesmos.

O tratamento de dados contínuos foi também introduzido neste capítulo, uma vez que uma base de dados raramente é composta somente por dados discretos. Se o devido tratamento não for realizado sobre os dados, pode ser difícil (e, em alguns casos, até impossível) utilizar de um sistema computacional baseado em AM para extrair conhecimento a partir destes dados. Foram apresentadas, ainda, as técnicas (e algoritmos) de discretização por particionamento do conjunto de valores em intervalos de mesma amplitude, discretização por particionamento do conjunto de valores em intervalos de mesma frequência e discretização por particionamento do conjunto de valores por entropia e MDLP, que são técnicas de particionamento amplamente difundidas e estabelecidas na grande área de IA e suas subáreas.

No próximo capítulo, será apresentada uma forma de representação de conhecimento conhecida como Grafos de Fluxo, que vem a ser uma opção para a representação de conjuntos de instâncias de dados em sistemas computacionais baseados em AM.

Capítulo 4
Grafos de Fluxo: Notação, Principais Conceitos e Variantes

Este capítulo apresenta uma revisão bibliográfica dos principais conceitos e resultados relacionados ao formalismo de representação de conhecimento chamado Grafos de Fluxo (GF), foco desse trabalho de pesquisa em nível de mestrado. O capítulo está organizado em dez seções e se baseia, principalmente, no conteúdo dos seguintes trabalhos científicos relacionados à GFs: [Pawlak 2003], [Pawlak 2003a], [Pawlak 2003b], [Pawlak 2004a], [Pawlak 2004b], [Pawlak 2004c], [Pawlak 2005a], [Pawlak 2005b], [Chan & Tsumoto 2007] e [Pawlak 2010].

O capítulo está organizado como segue: a Seção 4.1 apresenta algumas considerações iniciais sobre os GFs; a Seção 4.2 fica responsável por introduzir a notação e os conceitos gerais relacionados ao formalismo que sustenta os GFs; a Seção 4.3 apresenta a versão normalizada dos GFs; os fatores de certeza, cobertura e força são apresentados na Seção 4.4; a Seção 4.5 apresenta o conceito de Grafos Inversos; a Seção 4.6 define caminhos, caminhos completos e conexões; a Seção 4.7 apresenta a operação de fusão, possível de realização sobre um GF; a relação entre algoritmos de decisão e GFs é tratada na Seção 4.8; a seção 4.9 discorre sobre a dependência entre nós de GFs, e; a Seção 4.10 apresenta as considerações finais do capítulo.

4.1 Considerações Iniciais

Estudos com foco em GF não são tão difundidos quanto aqueles relacionados aos algoritmos mais populares em Aprendizado de Máquina (AM), tais como algoritmos de classificação e de agrupamento (*clustering*), por exemplo. Ainda assim, o material encontrado na literatura disponibiliza informações suficientes para o entendimento dos principais conceitos relacionados aos GFs que, na maioria das vezes, são apresentados de forma didática, acompanhados por exemplos de fácil compreensão.

Os GFs (ver [Pawlak 2003], [Pawlak 2004a], [Pawlak 2004b], [Pawlak 2006a], [Pawlak 2005b] e [Pawlak 2010]) podem ser definidos como estruturas de representação de conhecimento e são utilizados como uma ferramenta matemática para análise de fluxo de informação, representada por um grafo. Ainda, podem ser abordados como um tipo especial de banco de dados em que, em vez de dados sobre objetos individuais, características estatísticas dos objetos são apresentadas em termos de distribuição de fluxo da informação ([Pawlak 2005a]). Relações entre os atributos (de dados) que definem um GF são estabelecidas por meio de arcos que são interpretados como regras de decisão ([Pawlak 2003]). O GF como um todo pode ser interpretado como a representação de um algoritmo de decisão. Sendo assim, a partir da análise da distribuição dos fluxos contidos em um GF é possível induzir um algoritmo de decisão.

Algumas representações de dados (*e.g.* aquelas fundamentadas na Regra de Bayes) têm uma interpretação probabilística, pois seus resultados estão fortemente ligados à probabilidade (ver [Pawlak 2010]). Os resultados das análises realizadas em um GF, por sua vez, têm uma interpretação determinística, uma vez que refletem efetivamente o fluxo dos dados e não apenas uma probabilidade sobre eles. Evidentemente, como qualquer abordagem indutiva para a representação de conhecimento, os fatos representados pelos dados do conjunto de treinamento têm um profundo impacto no desempenho futuro de um GF, quando da classificação de novos dados. A seguir são apresentadas algumas aplicações em que GFs podem ser utilizados para a representação de conhecimento.

(1) *Tabagismo e câncer*: é possível verificar a existência de relações entre tabagismo e câncer a partir de um conjunto de dados contendo registros de pessoas fumantes e não fumantes (ver [Pawlak 2005b]). Cada instância desse conjunto de dados poderia ser descrita por apenas dois atributos: Fumante e Câncer, ambos admitindo como possíveis valores S(sim) e N(ão). A partir deste simples conjunto, com a utilização dos Grafos de Fluxo, é possível verificar as relações entre fumantes e a presença ou ausência de câncer, assim como, entre não-fumantes e a presença ou ausência de câncer. Um ponto-de-vista diferente também é uma possibilidade oferecida pela utilização dos GFs, no qual seria possível analisar a relação da presença de câncer em fumantes e não-fumantes, assim

como a ausência de câncer em fumantes e não-fumantes, bastando para tal, inverter (ver Seção 4.5) o GF original.

(2) *Efetividade de remédios:* um grupo de *n* pessoas participa de um teste para verificação da efetividade de um novo remédio. Entre elas, há pessoas saudáveis e pessoas doentes; as pessoas foram divididas, com base na idade, em três grupos: jovens, meia idade e idosos. Após os testes, os resultados de cada paciente são apresentados. As informações anteriores formam um conjunto de dados, em que cada instância de dado representa um determinado participante, descrito pelos atributos: presença da doença (com valores (S)im ou (N)ão), idade (jovens, meia idade, idosos) e resultado do teste ((P)ositivo ou (N)egativo). A análise desses dados pode evidenciar a relação estabelecida entre os pacientes doentes (ou não), a idade e o sucesso (ou não) do teste (ver [Pawlak 2010]).

(3) *Cabelo, olhos e nacionalidade*: ao se ter acesso a dados referentes à cor do cabelo, cor dos olhos e nacionalidade de *n* indivíduos, GFs podem auxiliar na descoberta de relações entre estas informações de forma que, possa ser determinada, com alta ou baixa precisão (depende dos dados e relações), a nacionalidade de uma pessoa ([Pawlak 2005b]). Obviamente, estes três atributos (cor do cabelo, cor dos olhos e nacionalidade) não cobririam todas as cores e/ou nacionalidades existentes, porém, dada uma população previamente selecionada, isto seria possível. Como comentado em [Pawlak 2005b], é possível induzir se uma pessoa é italiana ou sueca com base na análise da cor do cabelo e dos olhos da pessoa. É possível, também, traçar um perfil desses indivíduos, descobrindo a possível cor de seus cabelos, por meio da combinação dos atributos nacionalidade e cor dos olhos.

(4) *Venda de veículos*: analisando a venda de três modelos de veículos automotivos diferentes, para três grupos diferentes de compradores, por meio de três negociadores, é possível verificar as relações entre os modelos, negociadores e compradores ([Pawlak 2003]). Dessa forma, é possível predizer, com certa precisão, para que tipo de comprador um dado veículo provavelmente será vendido. Isso é possível, pois a análise feita sobre as negociações anteriores pode ter evidenciado essas relações. Analogamente, é possível predizer que modelo de carro que um determinado grupo de compradores prefere, sabendo apenas as características dos compradores e dos negociadores.

(5) *Qualidade de produção:* como descrito em [Pawlak 2005b], três fábricas produzem três tipos de produtos, os quais, naturalmente, nem sempre atendem a qualidade mínima aceitável, ou seja, alguns produtos são considerados defeituosos. A partir de um conjunto de dados em que cada instância é descrita pelos atributos fábrica (com os possíveis valores: fa_1, fa_2 e fa_3), produto (com os possíveis valores: pd_1, pd_2 e pd_3) e qualidade (sendo (b)oa ou (r)uim), é possível descobrir, por meio da representação de tais dados em um GF, as relações entre os atributos e chegar a conclusões que podem ajudar a determinar o fluxo dos dados nesta rede de dados, ou seja, seria possível induzir um conjunto de regras que poderiam predizer a qualidade de um produto, com base na fábrica que o fabricou, ou então, verificar as relações que identificam fábricas que produzem um determinado produto de boa ou má qualidade.

(6) *Blocos de Brinquedo*: como descrito em [Pawlak 2005a], vários blocos de brinquedo, em que os respectivos tamanhos podem ser grandes ou pequenos, cujas formas podem ser quadradas, redondas ou triangulares e cujas cores variam entre vermelho, verde e azul, é possível realizar uma análise para identificar as relações entre esses três atributos *i.e.*, tamanho, forma e cor. Desta maneira, pode-se determinar em que casos (relação de tamanho e forma) os brinquedos poderão ser de uma determinada cor. A partir das cores e formas, também existe a possibilidade de se descobrir quais seriam os tamanhos dos brinquedos em sua maioria.

(7) *Campanha promocional:* uma empresa pode utilizar a abordagem de Grafos de Fluxo para analisar dados e tentar predizer o resultado de campanhas promocionais ([Pawlak 2005b]). Ao coletar os dados de indivíduos sobre a respectiva faixa etária (com os possíveis valores: jovens, meia idade, idosos), local de residência (com possíveis os valores: zona urbana, distrito, zona rural) e se compraria ou não o produto (com os possíveis valores: (S)im ou (N)ão)) em uma determinada pesquisa de campo, é possível criar um conjunto de registros (ou instâncias) de dados que caracterizam (ou não) a intenção de compra de determinados produtos por determinados clientes. Ao representar esses dados com GF, é possível verificar as suas relações de forma que seja possível determinar se um cliente compraria ou não um determinado produto em promoção com base em suas características (faixa etária e local de residência). O GF geraria um conjunto de regras que representaria essas relações, ou seja, poderia ser extraído um

algoritmo de decisão a partir do GF para auxiliar na predição de quem compraria ou não produtos futuramente.

4.2 Notação e Conceitos Gerais

Nesta seção, o conceito de Grafo de Fluxo (GF) é apresentado, e considerações sobre tal conceituação são discutidas e exemplificadas.

Definição 4.1 Um *Grafo de Fluxo (GF)* é um dígrafo (*i.e.*, um grafo finito e direcionado), formalmente representado pela tripla $G=(N,\beta,\varphi)$, em que N representa um conjunto de nós, $\beta \subseteq N \times N$ representa um conjunto de arcos que conectam nós de N e $\varphi:\beta \rightarrow R^+$ é uma função de fluxo, em que R^+ representa o conjunto dos números reais não-negativos ([Pawlak 2005a], [Pawlak 2010] e [Pawlak 2005b]).

Observação 4.1 A Definição 4.1 fala sobre a função de fluxo dos arcos, porém não deixa claro que há também um fluxo, representado por um valor, associado a cada nó que compõe um GF. Esse conceito é definido mais adiante, porém não é, sequer, citado pela Definição 4.1.

Observação 4.2 De acordo com a Definição 4.1, entende-se que arcos de um GF podem conectar quaisquer nós $x \in N$, independentemente de suas camadas e, inclusive, podem possuir o mesmo nó como entrada e saída do arco. Porém, essa definição está incorreta. Considere C o conjunto de camadas de um dado GF. Investigações realizadas, por exemplo, em ([Pawlak 2005a], [Pawlak 2010] e [Pawlak 2005b], mostram que arcos de GFs sempre conectam nós $x \in N$ e $y \in N$, sendo que x deve pertencer à uma camada C_i e y deve pertencer à uma camada C_{i+1}, $1 \leq i \leq |C|-1$. A descrição da indução de GFs apresentada a seguir utiliza o método correto para definição de arcos do GF, de acordo com os artigos investigados.

Dado um conjunto de instâncias de dados X, os GFs são induzidos como segue:

(1) Cada atributo presente em X dá origem à uma camada do GF com o mesmo nome, ou seja, um GF terá tantas camadas quanto for o número de atributos presentes no conjunto de instâncias de dados X;

(2) Cada valor distinto associado a um atributo at_i, no conjunto de instâncias de dados, dá origem à um nó, com o mesmo valor associado, pertencente à camada do GF correspondente ao atributo at_i do conjunto de instâncias de dados;

(2.1) O número de instâncias que possuem esse valor distinto para o atributo at_i é definido como fluxo do nó correspondente no GF;

(3) Cada relação de valores que ocorre entre dois atributos, at_i e at_{i+1}, no conjunto de instâncias de dados, define, no GF, um arco <x,y> que conecta os nós x e y originados a partir dos valores distintos dos atributos at_i e at_{i+1};

(3.1) O número de vezes que uma relação de valores ocorre entre dois valores distintos pertencentes aos atributos at_i e at_{i+1} define o fluxo do arco que liga os nós x (originado a partir de at_i) e y (originado a partir de at_{i+1}).

Observação 4.3 De acordo com a Definição 4.1, em um GF G especificado por $G=(N,\beta\ \varphi)$, N é um conjunto de nós, $\beta \subseteq N \times N$ é um conjunto de arcos que conectam nós de N e $\varphi:\beta \rightarrow R^+$ é uma função de fluxo, em que R^+ representa o conjunto dos números reais não-negativos. O uso da mesma simbologia para representar o fluxo de um nó (*i.e.*, a função φ), introduz uma certa inconsistência, considerando que o fluxo de um nó é uma função $\varphi:N \rightarrow R^+$. A sugestão desse trabalho é que um símbolo diferente de φ seja utilizado para definir o fluxo associado aos nós do GF. Porém, um símbolo diferente não foi proposto neste documento.

Exemplo 4.1 O GF G, exibido na Figura 4.1, será utilizado ao longo deste capítulo e será reutilizado/atualizado à medida que novos conceitos forem introduzidos e discutidos. O GF G representa o fluxo de dados em um conjunto de treinamento com 100 instâncias (ou registros) descritas por três atributos, A1, A2 e A3. Dessas 100 instâncias, 40 têm valor x_1 para o atributo A1, 45 têm valor x_2 para o atributo A1 e 25 têm valor x_3 para o mesmo atributo. Também, das 100 instâncias, 10 têm valor y_1 para o atributo A2, 45 têm o valor y_2 para o atributo A2, 17 possuem valor y_3 para o atributo A2 e 28 possuem o valor y_4 para o mesmo atributo. Por fim, das 100 instâncias, 15 têm valor z_1 para A3 e 85, valor z_2 para o mesmo atributo. As relações entre o número de valores associados a cada atributo, com base no número de instâncias analisadas, permitem a verificação e visualização do fluxo dos dados entre os atributos. Mais

especificamente, as conexões entre os nós que representam os valores dos atributos indicam a *distribuição do fluxo dos dados* entre os nós do GF.

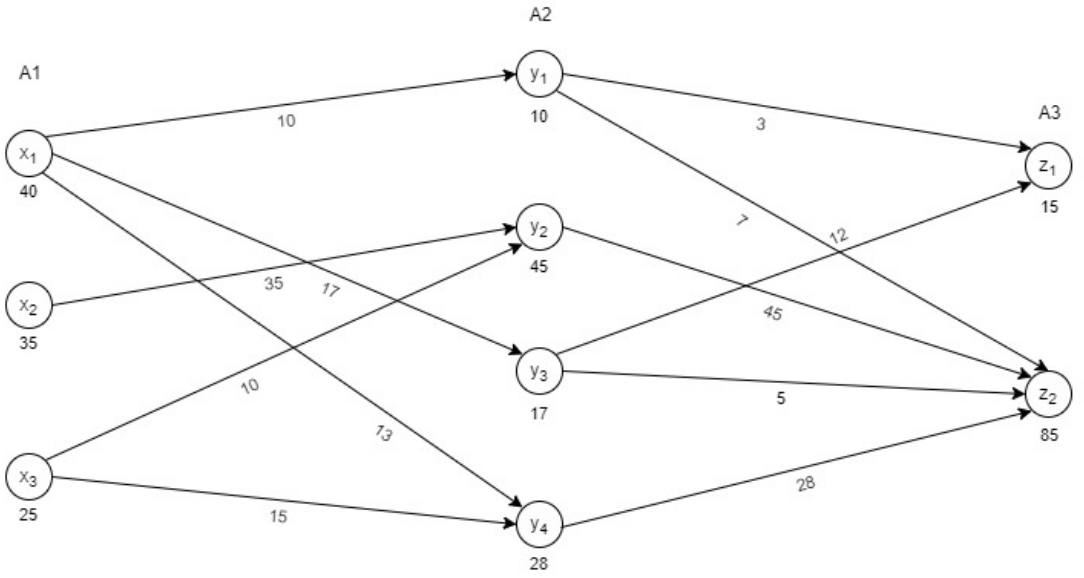


Figura 0.1 Grafo de Fluxo (GF).

Um nó x representa o número de registros que possuem o valor x associado a um determinado atributo de um GF. Esse número é denominado *fluxo de x* e é denotado por φ(x). No grafo mostrado na Figura 4.1, por exemplo, associados ao atributo A1 existem três nós, cada um deles representando um possível valor que tal atributo pode assumir: x_1, x_2 e x_3. O nó x_1 tem fluxo 40 *i.e.*, $\varphi(x_1)=40$, o que significa que, do total de instâncias (100), 40 são descritas por expressões em que o atributo A1 tem valor x_1. Já o nó x_2 é tal que $\varphi(x_2)=35$ e o nó x_3 tem o seu fluxo associado dado por $\varphi(x_3)=25$. Associados ao atributo A2 existem quatro nós que representam os quatro possíveis valores que o atributo A2 pode assumir. Os valores de fluxo desses quatro nós são dados por $\varphi(y_1)=10$, $\varphi(y_2)=45$, $\varphi(y_3)=17$ e $\varphi(y_4)=28$. Associado ao atributo A3 existem dois nós, z_1 e z_2, representando os dois possíveis valores que A3 pode assumir; tais nós, têm como valores de fluxo $\varphi(z_1)=15$ e $\varphi(z_2)=85$.

Note que, da maneira como o GF é construído, a soma dos fluxos associados ao grupo de nós que representam os possíveis valores de um determinado atributo deve ser sempre igual ao número de registros que participam do conjunto inicial de dados. No

caso do GF G exibido na Figura 4.1, por exemplo, $\varphi(x_1)+\varphi(x_2)+\varphi(x_3)=40+35+25=100$; $\varphi(y_1)+\varphi(y_2)+\varphi(y_3)+\varphi(y_4)=10+48+17+28=100$ e $\varphi(z_1)+\varphi(z_2)=15+85=100$.

Seja $<x,y>$ a notação adotada para representar um arco que se inicia no nó x e termina no nó y. Se $<x,y>\in\beta$, existe no GF G um arco conectando os nós x e y. Por convenção, é adotada a terminologia que *y é uma saída de x* e que *x é uma entrada de y*. É possível observar no GF G da Figura 4.1, que o conjunto dos 12 arcos que comparecem em G é dado por:

$\beta = \{<x_1,y_1>,<x_1,y_3>,<x_1,y_4>,<x_2,y_2>,<x_3,y_2>,<x_3,y_4>,<y_1,z_1>,<y_1,z_2>,<y_2,z_2>,<y_3,z_1>,<y_3,z_2>,<y_4,z_2>\}$.

Um nó y pode ter j entradas x_i (i=1,..., j), assim como um nó x pode ter *k* saídas y_i (i=1,..., k). Por exemplo, no GF G da Figura 4.1, o nó y_2 tem 2 entradas (x_2 e x_3) e apenas 1 saída (z_2).

Definição 4.2 O *conjunto de entradas de um nó x* é denotado e definido por $I(x)=\{y\in N \mid <y,x>\in\beta\}$ e corresponde ao conjunto de todos os nós do grafo que são a primeira coordenada do par ordenado que representa o arco de y para x, *i.e.*, $<y,x>$.

Considerando novamente a Figura 4.1, o conjunto das entradas do nó z_2 é dado por $I(z_2)=\{y_1,y_2,y_3,y_4\}$, uma vez que $<y_1,z_2>\in\beta$, $<y_2,z_2>\in\beta$, $<y_3,z_2>\in\beta$ e $<y_4,z_2>\in\beta$.

Definição 4.3 O *conjunto de saídas de um nó x* é denotado e definido por $O(x)=\{y\in N \mid <x,y>\in\beta\}$ e corresponde ao conjunto de todos os nós do grafo que são a segunda coordenada do par ordenado que representa o arco de x para y, *i.e.*, $<x,y>$.

O conjunto das saídas do nó x_1 de G é $O(x_1)=\{y_1,y_3,y_4\}$, pois $<x_1,y_1>\in\beta$, $<x_1,y_3>\in\beta$ e $<x_1,y_4>\in\beta$.

Definição 4.4 A *entrada do GF* G é definida por $I(G)=\{x\in N \mid I(x)=\varnothing\}$, isto é, o conjunto de nós de G tal que o conjunto de entradas de cada um deles é vazio.

Na Figura 4.1, $I(G)=\{x_1,x_2,x_3\}$, dado que x_1, x_2 e x_3 são os únicos nós de G tal que $I(x_1)=\varnothing$, $I(x_2)=\varnothing$ e $I(x_3)=\varnothing$.

Definição 4.5 A *saída do GF* G é definida por $O(G)=\{x \in N \mid O(x)=\varnothing\}$, ou seja, o conjunto daqueles nós de G que têm os respectivos conjuntos de saídas vazios.

Para o GF G da Figura 4.1, $O(G)=\{z_1,z_2\}$, uma vez que z_1 e z_2 são os únicos nós de G tal que $O(z_1)=\varnothing$ e $O(z_2)=\varnothing$.

Definição 4.6 Os nós de entrada e os nós de saída de um GF são considerados *nós externos*, enquanto os outros nós são considerados *nós internos*.

Para o GF da Figura 4.1, os nós y_1, y_2, y_3 e y_4 são os nós internos; todos os outros nós são nós externos.

Definição 4.7 Considere um GF $G=(N,\beta,\varphi)$ como definido em Definição 4.1. Para cada um dos arcos $<x,y> \in \beta$, é possível definir $\varphi(<x,y>)$ como sendo o *fluxo do arco que une x a y*.

No GF da Figura 4.1, $\varphi(<x_1,y_1>)=10$, o que significa que das 40 instâncias de dados, cujo respectivos valores do atributo A1 são todos iguais a x_1, 10 delas têm valor y_1 para o atributo A2. Da mesma forma, $\varphi(<x_1,y_3>)=17$, pois das 40 instâncias de dados que têm atributo A1 com valor x_1, 17 têm valor y_3 para o atributo A2. Por sua vez, $\varphi(<x_1,y_4>)=13$, pois das 40 instâncias de dados, cujos respectivos atributos A1 têm valor x_1, 13 delas têm valor y_4 para A2.

Definição 4.8 Considere um GF $G=(N,\beta,\varphi)$ como definido em Definição 4.1. Para cada nó x de um GF, seu *fluxo de entrada* (*inflow*), notado por $\varphi_+(x)$, é definido como a soma dos fluxos dos arcos que terminam em x, como descreve a Eq. (1).

$$\varphi_+(x) = \sum_{y \in I(x)} \varphi(<y,x>) \qquad (1)$$

A Tabela 4.1 mostra o cálculo dos *inflows* de cada um dos nós que compõem o GF G da Figura 4.1.

Tabela 0.1 Cálculo dos *inflows* dos nós do GF G da Figura 4.1.

nó	**cálculo de *inflow*(nó)**
x_1	$\varphi_+(x_1) = inflow(x_1) = 0$ uma vez que $x_1 \in I(G)$
x_2	$\varphi_+(x_2) = inflow(x_2) = 0$ uma vez que $x_2 \in I(G)$
x_3	$\varphi_+(x_3) = inflow(x_3) = 0$ uma vez que $x_3 \in I(G)$
y_1	$\varphi_+(y_1) = inflow(y_1) = \varphi(<x_1,y_1>) = 10$
y_2	$\varphi_+(y_2) = inflow(y_2) = \varphi(<x_2,y_2>) + \varphi(<x_3,y_2>) = 35+10 = 45$
y_3	$\varphi_+(y_3) = inflow(y_3) = \varphi(<x_1,y_3>) = 17$
y_4	$\varphi_+(y_4) = inflow(y_4) = \varphi(<x_1,y_4>) + \varphi(<x_3,y_4>) = 13+15 = 28$
z_1	$\varphi_+(z_1) = inflow(z_1) = \varphi(<y_1,z_1>) + \varphi(<y_3,z_1>) = 3 + 12 = 15$
z_2	$\varphi_+(z_2) = inflow(z_2) = \varphi(<y_1,z_2>) + \varphi(<y_2,z_2>) + \varphi(<y_3,z_2>) + \varphi(<y_4,z_2>) = 7+45+5+28 = 85$

Definição 4.9 Considere um GF G=(N,β,φ) como definido em Definição 4.1. Para cada nó x de GF, seu *fluxo de saída* (*outflow*), notado por $\varphi_-(x)$, é definido como a soma dos fluxos dos arcos que se originam em x, como descreve a Eq. (2).

$$\varphi_-(x) = \sum_{y \in O(x)} \varphi(<x,y>) \tag{2}$$

A Tabela 4.2 apresenta o cálculo dos *outflows* de cada um dos nós do GF G (Figura 4.1).

Tabela 0.2 Cálculo dos *outflows* dos nós do GF G da Figura 4.1.

nó	**cálculo de *outflow*(nó)**
x_1	$\varphi_-(x_1) = outflow(x_1) = \varphi(<x_1,y_1>) + \varphi(<x_1,y_3>) + \varphi(<x_1,y_4>) = 10+17+13 = 40$
x_2	$\varphi_-(x_2) = outflow(x_2) = \varphi(<x_2,y_2>) = 35$
x_3	$\varphi_-(x_3) = outflow(x_3) = \varphi(<x_3,y_2>) + \varphi(<x_3,y_4>) = 10+15 = 25$
y_1	$\varphi_-(y_1) = outflow(y_1) = \varphi(<y_1,z_1>) + \varphi(<y_1,z_2>) = 3+7 = 10$
y_2	$\varphi_-(y_2) = outflow(y_2) = \varphi(<y_2,y_2>) = 45$
y_3	$\varphi_-(y_3) = outflow(y_3) = \varphi(<y_3,z_1>) + \varphi(<y_3,z_2>) = 12+5 = 17$
y_4	$\varphi_-(y_4) = outflow(y_4) = \varphi(<y_4,z_2>) = 28$
z_1	$\varphi_-(z_1) = outflow(z_1) = 0$ uma vez que $z_1 \in O(G)$
z_2	$\varphi_-(z_2) = outflow(z_2) = 0$ uma vez que $z_2 \in O(G)$

Observação 4.4 Para as definições dos conceitos de *inflow* (Definição 4.8) e *outflow* (Definição 4.9) associados aos nós, valem as mesmas considerações feitas na Observação 4.3.

Definição 4.10 Os cálculos do fluxo de entrada e do fluxo de saída do GF G são similares àqueles associados aos nós. As equações Eq. (3) e Eq. (4) são usadas para obter o fluxo de entrada e de saída, respectivamente, de um GF G:

$$\varphi_{+}(G) = \sum_{x \in I(G)} \varphi_{-}(x) \tag{3}$$

$$\varphi_{-}(G) = \sum_{x \in O(G)} \varphi_{+}(x) \tag{4}$$

Para o GF G do Exemplo 4.1, $I(G)=\{x_1,x_2,x_3\}$ e $O(G)=\{z_1,z_2\}$. Portanto, considerando os valores mostrados nas tabelas 4.1 e 4.2 tem-se que:

$\varphi_{+}(G) = \varphi_{-}(x_1)+\varphi_{-}(x_2)+\varphi_{-}(x_3) = 40+35+25 = 100$ e

$\varphi_{-}(G) = \varphi_{+}(z_1)+\varphi_{+}(z_2) = 15+85 = 100.$

Observação 4.5 Para as definições dos conceitos de fluxo de entrada e fluxo de saída relativos a um GF (Definição 4.10) associados ao grafo como um todo, valem as mesmas considerações feitas na Observação 4.3.

GFs são grafos que buscam modelar um conjunto de dados, em função dos valores associados aos atributos que definem os dados. Sendo assim, para cada *nó interno* x do GF, o fluxo de entrada deve ser igual ao fluxo de saída *i.e.*, $\varphi_{+}(x) = \varphi_{-}(x)$, o que evidencia o fato de que o nó x *não retém dados*.

Se um nó x tem como fluxo de entrada (a soma dos fluxos dos arcos que entram em x) o valor 100, sua saída (a soma dos fluxos dos arcos que saem de x) também deverá ter o valor 100. Essa característica reflete o conceito de *fluxo do nó*, denotado por $\varphi(x)$, visto anteriormente, que se refere ao valor que *passa* pelo nó. Para cada nó *interno* $x \in N$, é fato que $\varphi_{+}(x) = \varphi(x) = \varphi_{-}(x)$ em que $\varphi(x)$ denota o fluxo do nó x. A tabela 4.3 apresenta os valores de φ_{+}, φ e φ_{-} associados a cada nó do GF G.

Tabela 0.3 Os valores de φ, φ_+ e φ_- associados a cada nó do GF G da Figura 4.1.

nó	φ_+**(nó) =** ***inflow*****(nó)**	φ**(nó) =** ***fluxo*****(nó)**	φ_-**(nó) =** ***outflow*****(nó)**
x_1	$\varphi_+(x_1) = 0$, pois $x_1 \in I(G)$	$\varphi(x_1) = 40$	$\varphi_-(x_1) = 40$
x_2	$\varphi_+(x_2) = 0$, pois $x_2 \in I(G)$	$\varphi(x_2) = 35$	$\varphi_-(x_2) = 35$
x_3	$\varphi_+(x_3) = 0$, pois $x_3 \in I(G)$	$\varphi(x_3) = 25$	$\varphi_-(x_3) = 25$
y_1	$\varphi_+(y_1) = 10$	$\varphi(y_1) = 10$	$\varphi_-(y_1) = 10$
y_2	$\varphi_+(y_2) = 45$	$\varphi(y_2) = 45$	$\varphi_-(y_2) = 45$
y_3	$\varphi_+(y_3) = 17$	$\varphi(y_3) = 17$	$\varphi_-(y_3) = 17$
y_4	$\varphi_+(y_4) = 28$	$\varphi(y_4) = 28$	$\varphi_-(y_4) = 28$
z_1	$\varphi_+(z_1) = 15$	$\varphi(z_1) = 15$	$\varphi_-(z_1) = 0$, pois $z_1 \in O(G)$
z_2	$\varphi_+(z_2) = 85$	$\varphi(z_2) = 85$	$\varphi_-(z_2) = 0$, pois $z_2 \in O(G)$

É importante notar, que somente para *nós internos*, $\varphi_+(x) = \varphi(x) = \varphi_-(x)$. Para nós externos, tal igualdade não se verifica, pois nós de entrada possuem fluxo de entrada 0 e nós de saída possuem fluxo de saída 0.

Como as entradas e saídas dos nós e as entradas e saídas dos GFs como um todo são calculadas de forma análoga, as seguintes igualdades se verificam: $\varphi_+(G)=\varphi(G)=\varphi_-(G)$, em que $\varphi(G)$ denota o fluxo do grafo G como um todo. Considerando o GF G do Exemplo 4.1, têm-se que $\varphi_+(G)=40+35+25=100$, $\varphi(G)=100$ e $\varphi_-(G)=15+85=100$.

Exemplo 4.2 O exemplo do GF G do Exemplo 4.1 (apresentado na Figura 4.1) é atualizado e exibido na Figura 4.2, apresentando os conceitos tratados até o momento.

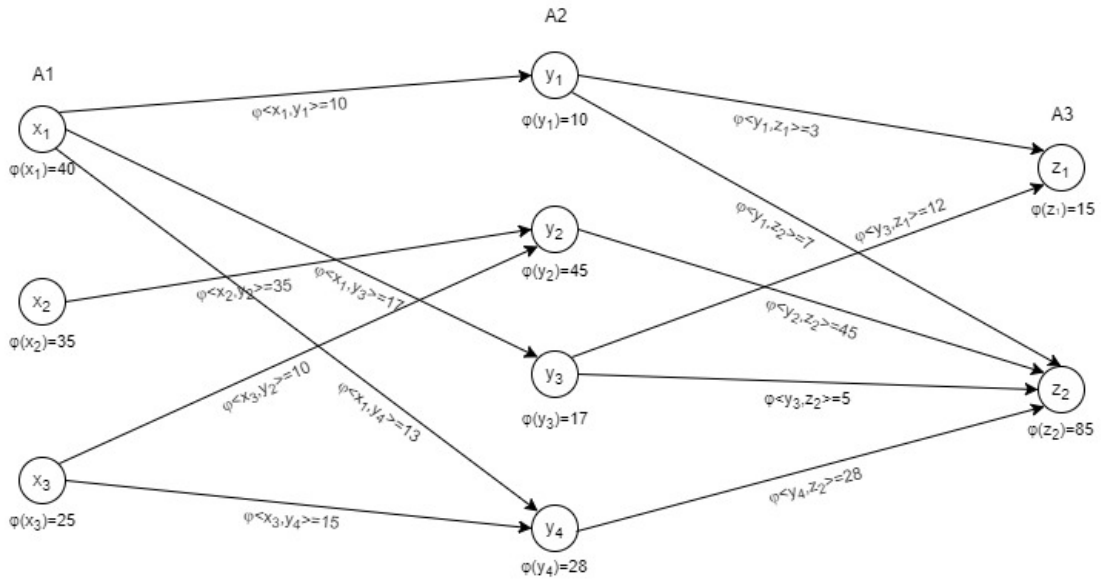


Figura 0.2 Grafo de Fluxo G com especificação dos fluxos de nós e dos fluxos dos arcos.

4.3 Grafos de Fluxo Normalizados

Nesta seção é abordado o conceito de Grafo de Fluxo Normalizado (GFN), uma especialização do conceito de Grafos de Fluxo já abordado na Seção 4.2; no caso de GFNs, o contradomínio da função de fluxo φ está restrito ao intervalo fechado [0 1], como estabelece a Definição 4.11.

Definição 4.11 Um *Grafo de Fluxo Normalizado (GFN)* é um dígrafo (*i.e.*, um grafo finito e direcionado), formalmente representado pela tripla GN=(N,β,σ), em que N representa o conjunto de nós do grafo, $\beta \subseteq N \times N$ é um conjunto de arcos que conectam os nós de N e $\sigma: B \rightarrow [0\ \ 1]$ (ou seja, $0 \leq \sigma \leq 1$) é uma função de fluxo com valores entre 0 e 1. Obviamente, os valores associados aos nós e arcos (*e.g.* fluxo, *inflow* e *outflow*) do GFN GN também são normalizados.

Definição 4.12 Em um GFN GN, como definido em Definição 4.11, o valor σ(x) representa o fluxo do nó x em relação ao grafo como um todo e é definido pela Eq. (5).

$$\sigma(x) = \frac{\varphi(x)}{\varphi(G)} \tag{5}$$

Considerando novamente o GF do Exemplo 4.1, o fluxo do nó x_1 do GF G é 40 ($\varphi(x_1)=40$), pois do total de instâncias de dados (*i.e.* 100), 40 delas possuem valor x_1 para o atributo A1 e o fluxo do GF é 100 ($\varphi(G)=100$). Ao normalizar este fluxo para o nó x_1, considerando a Definição 4.12, tem-se $\sigma(x_1)=40/100=0{,}40$. Da mesma forma, o fluxo normalizado do nó y_2 é dado por $\sigma(y_2)=\varphi(y_2)/\varphi(G)=45/100=0{,}45$. A Tabela 4.4 mostra o fluxo normalizado de todos os nós do GF G, para a construção do Grafo de Fluxo Normalizado GN, associado a G.

Tabela 0.4 Cálculo do fluxo de cada um dos nós do GF G da Figura 4.1 usando Eq. (5), para a construção do GFN GN.

nó	cálculo do fluxo do nó
x_1	$\sigma(x_1) = 40 / 100 = 0{,}400$
x_2	$\sigma(x_2) = 35 / 100 = 0{,}350$
x_3	$\sigma(x_3) = 25 / 100 = 0{,}250$
y_1	$\sigma(y_1) = 10 / 100 = 0{,}100$
y_2	$\sigma(y_2) = 45 / 100 = 0{,}450$
y_3	$\sigma(y_3) = 17 / 100 = 0{,}170$
y_4	$\sigma(y_4) = 28 / 100 = 0{,}280$
z_1	$\sigma(z_1) = 15 / 100 = 0{,}150$
z_2	$\sigma(z_2) = 85 / 100 = 0{,}850$

Definição 4.13 O conjunto de entrada e o conjunto de saída dos nós dos GFs, assim como dos nós dos GFNs, são dados, respectivamente, pelos conjuntos

$$I(x) = \{y \in N \mid \langle y,x \rangle \subseteq \beta\} \text{ e}$$

$$O(x) = \{y \in N \mid \langle x,y \rangle \subseteq \beta\}.$$

Definição 4.14 Em um GFN GN, os conjuntos de entrada e de saída são dados, respectivamente, pelos conjuntos

$$I(GN) = \{x \in N \mid I(x) = \varnothing\} \text{ e}$$

$$O(GN) = \{x \in N \mid O(x) = \varnothing\}.$$

Definição 4.15 Em um GFN, a *força* (*strength*) de um arco $\langle x,y \rangle$, dada por $\sigma(\langle x,y \rangle)$, é uma representação da distribuição do fluxo dos dados que passam pelo arco. A força associada a um arco $\langle x,y \rangle$ do GFN é calculada pela Eq. (6).

$$\sigma(\langle x, y \rangle) = \frac{\varphi(\langle x, y \rangle)}{\varphi(G)} \qquad (6)$$

Definição 4.16 Em um GFN, o fluxo de entrada (*inflow*) de um nó x, *i.e.*, o número de instâncias de dados que entram em x, denotado por $\sigma_+(x)$, é calculado pela Eq. (7).

$$\sigma_{+}(x) = \sum_{y \in I(x)} \sigma(<y, x>) \quad (7)$$

Definição 4.17 Em um GFN, o fluxo de saída (*outflow*) de um nó x, denotado por $\sigma_{-}(x)$, é calculado pela Eq. (8).

$$\sigma_{-}(x) = \sum_{y \in O(x)} \sigma(<x, y>) \quad (8)$$

Em GFNs, de maneira similar aos GFs, os nós não retêm instâncias, ou seja, se n instâncias chegam a um nó x, n instâncias saem do nó x. Em outras palavras, considerando os nós internos de um GFN GN, o número de instâncias que entram em um nó x (inflow(x)), denotado por $\sigma_{+}(x)$, é o mesmo número de instâncias que saem do nó x (outflow(x)), denotado por $\sigma_{-}(x)$. Resumindo, para os nós internos de GN, $\sigma_{+}(x)=\sigma_{-}(x)=\sigma(x)$.

Definição 4.18 Em um GFN GN, o fluxo de entrada (*inflow*) de GN é dado pela Eq. (9).

$$\sigma_{+}(GN) \sum_{x \in I(GN)} \sigma_{-}(x) \quad (9)$$

Definição 4.19 Em um GFN GN, o fluxo de saída (*outflow*) de GN é dado pela Eq. (10).

$$\sigma_{-}(GN) \sum_{x \in O(GN)} \sigma_{+}(x) \quad (10)$$

As instâncias que entram em um GFN GN passam pelos seus nós e, ao final, saem como parte de alguma representação. Esta característica de GFNs é equacionada

pelas igualdades $\sigma_{+}(GN)=\sigma(GN)=\sigma_{-}(GN)$. Por fim, a definição $\sigma(GN)=1$ é válida para todo GFN, pois representa o total de instâncias de dados representadas pelo GFN.

Exemplo 4.3 A Figura 4.3 apresenta o GFN GN obtido com a normalização do GF G da Figura 4.1.

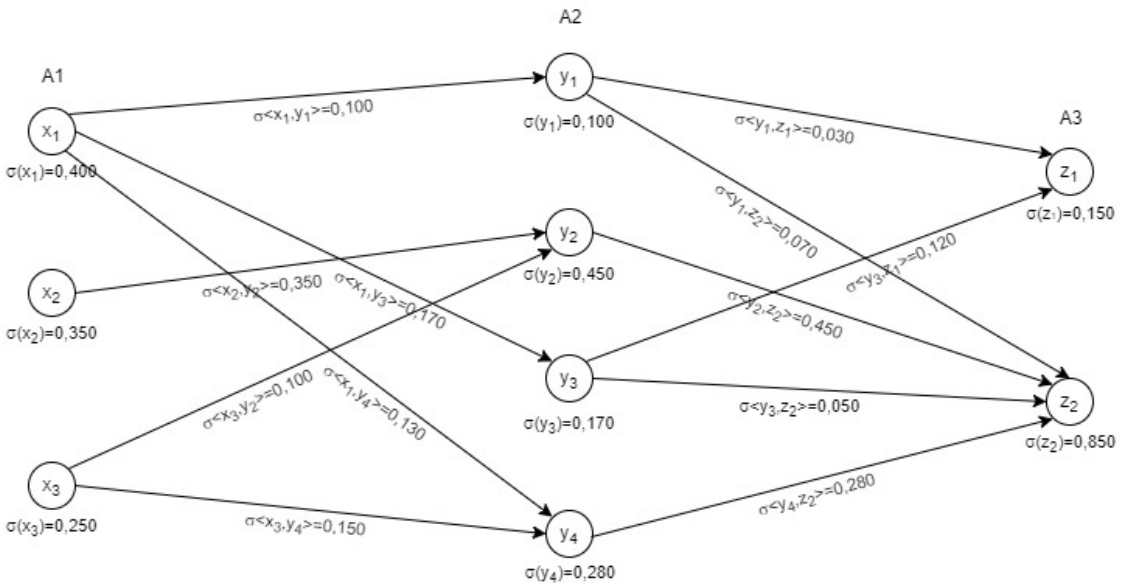


Figura 0.3 Grafo de Fluxo Normalizado (GFN) GN.

Com o objetivo de promover facilidade de leitura e compreensão, a representação numérica adotada neste trabalho, para a apresentação de resultados de cálculos, foi limitada a três dígitos decimais. Devido à representação adotada, algumas pequenas diferenças nos resultados dos cálculos, poderão acontecer e, quando isso acontecer, o sinal $\cong$ será utilizado.

4.4 Certeza, Cobertura e Força

Em um GFN, cada arco $<x,y>\in\beta$ possui quatro *fatores* (ou *coeficientes*) associados, a saber: força (strength), certeza (*certainty*), cobertura (*coverage*) e dependência (*dependency*), cada um deles abordado no que segue.

Definição 4.20 Em um GFN, o *fator de força* de um arco <x,y>, denotado por $\sigma(<x,y>)$, reflete a distribuição do fluxo dos dados que passa pelo arco <x,y> *i.e.*, a

porcentagem, em relação ao total de instâncias de dados que participam da criação do GFN, que passam pelo arco <x,y>. O valor de força pode ser determinado a partir do fluxo do arco (Definição 4.7), do valor do fator de certeza do arco ou do valor do fator de cobertura do arco, como dado pelas Eq. (11) e Eq (12).

$$\sigma(<x, y>) = \varphi(<x, y>) \tag{11}$$

$$\sigma(<x, y>) = \sigma(x)\text{cer}(<x, y>) = \sigma(y)\text{cov}(<x, y>) \tag{12}$$

A Tabela 4.5 apresenta os valores do fator de força para todos os arcos que comparecem no GFN GN e a Figura 4.4 apresenta o GFN GN atualizado com os novos conceitos.

Tabela 0.5 Cálculo do fator de força para os 12 arcos do GFN GN da Figura 4.3.

<x,y>	**σ(<x,y>)**
$<x_1,y_1>$	$\sigma(x_1)\text{cer}(<x_1,y_1>) = \sigma(y_1)\text{cov}(<x_1,y_1>) = 0{,}250\times0{,}400 = 1{,}000\times0{,}100 = 0{,}100$
$<x_1,y_3>$	$\sigma(x_1)\text{cer}(<x_1,y_1>) = \sigma(y_1)\text{cov}(<x_1,y_1>) = 0{,}425\times0{,}400 = 1{,}000\times0{,}170 = 0{,}170$
$<x_1,y_4>$	$\sigma(x_1)\text{cer}(<x_1,y_4>) = \sigma(y_4)\text{cov}(<x_1,y_4>) = 0{,}325\times0{,}400 = 0{,}464\times0{,}280 = 0{,}130$
$<x_2,y_2>$	$\sigma(x_2)\text{cer}(<x_2,y_2>) = \sigma(y_2)\text{cov}(<x_2,y_2>) = 0{,}350\times1{,}000 = 0{,}450\times0{,}777 = 0{,}350$
$<x_3,y_2>$	$\sigma(x_3)\text{cer}(<x_3,y_2>) = \sigma(y_2)\text{cov}(<x_3,y_2>) = 0{,}250\times0{,}400 = 0{,}450\times0{,}222 = 0{,}100$
$<x_3,y_4>$	$\sigma(x_3)\text{cer}(<x_3,y_4>) = \sigma(y_4)\text{cov}(<x_3,y_4>) = 0{,}250\times0{,}600 = 0{,}280\times0{,}535 = 0{,}150$
$<y_1,z_1>$	$\sigma(y_1)\text{cer}(<y_1,z_1>) = \sigma(z_1)\text{cov}(<y_1,y_1>) = 0{,}100\times0{,}300 = 0{,}150\times0{,}200 = 0{,}030$
$<y_1,z_2>$	$\sigma(y_1)\text{cer}(<y_1,z_2>) = \sigma(z_2)\text{cov}(<y_1,z_2>) = 0{,}100\times0{,}700 = 0{,}850\times0{,}082 = 0{,}070$
$<y_2,z_2>$	$\sigma(y_2)\text{cer}(<y_2,z_2>) = \sigma(z_2)\text{cov}(<y_2,z_2>) = 0{,}450\times1{,}000 = 0{,}850\times0{,}529 = 0{,}450$
$<y_3,z_1>$	$\sigma(y_3)\text{cer}(<y_3,z_1>) = \sigma(z_1)\text{cov}(<y_3,z_1>) = 0{,}170\times0{,}705 = 0{,}150\times0{,}800 = 0{,}120$
$<y_3,z_2>$	$\sigma(y_3)\text{cer}(<y_3,z_2>) = \sigma(z_2)\text{cov}(<y_3,z_2>) = 0{,}170\times0{,}294 = 0{,}850\times0{,}058 = 0{,}500$
$<y_4,z_2>$	$\sigma(y_4)\text{cer}(<y_4,z_2>) = \sigma(z_2)\text{cov}(<y_4,z_2>) = 0{,}280\times1{,}000 = 0{,}850\times0{,}329 = 0{,}280$

Definição 4.21 Considere um GFN. O *fator de certeza* de um arco $<x,y>\in\beta$, notado por cer(<x,y>), é gerado com base no nó x do arco <x,y> e corresponde à certeza da distribuição do fluxo entre os valores de atributos e é dado pela Eq. (13).

$$\text{cer}(<x, y>) = \frac{\sigma(<x, y>)}{\sigma(x)} \tag{13}$$

Como exemplo, se cer(<x,y>)=0,95, é certeza (0,95 representa 95% de chance), com base nas instâncias de dados que definiram o GN, que atributos com mesmo valor x tenderão a se relacionar com atributos de valor y.

Considerando o GFN GN exibido na Figura 4.3, o fator de certeza para o arco $\langle x_1,y_1\rangle$ é dado por $cer(\langle x_1,y_1\rangle)=\sigma\langle x_1,y_1\rangle/\sigma(x_1)=0{,}100/0{,}400=0{,}250$. A Tabela 4.6 apresenta o fator de certeza associado a todos os arcos que comparecem em GN.

Tabela 0.6 Cálculo do fator de certeza para os 12 arcos do GFN GN da Figura 4.3.

<x,y>	**cer(<x,y>)**
$\langle x_1,y_1\rangle$	$cer(\langle x_1,y_1\rangle) = \sigma(\langle x_1,y_1\rangle)/\sigma(x_1) = 0{,}100/0{,}400 = 0{,}250$
$\langle x_1,y_3\rangle$	$cer(\langle x_1,y_3\rangle) = \sigma(\langle x_1,y_3\rangle)/\sigma(x_1) = 0{,}170/0{,}400 = 0{,}425$
$\langle x_1,y_4\rangle$	$cer(\langle x_1,y_4\rangle) = \sigma(\langle x_1,y_4\rangle)/\sigma(x_1) = 0{,}130/0{,}400 = 0{,}325$
$\langle x_2,y_2\rangle$	$cer(\langle x_2,y_2\rangle) = \sigma(\langle x_2,y_2\rangle)/\sigma(x_2) = 0{,}350/0{,}350 = 1{,}000$
$\langle x_3,y_2\rangle$	$cer(\langle x_3,y_2\rangle) = \sigma(\langle x_3,y_2\rangle)/\sigma(x_3) = 0{,}100/0{,}250 = 0{,}400$
$\langle x_3,y_4\rangle$	$cer(\langle x_3,y_4\rangle) = \sigma(\langle x_3,y_4\rangle)/\sigma(x_3) = 0{,}150/0{,}250 = 0{,}600$
$\langle y_1,z_1\rangle$	$cer(\langle y_1,z_1\rangle) = \sigma(\langle y_1,z_1\rangle)/\sigma(y_1) = 0{,}030/0{,}100 = 0{,}300$
$\langle y_1,z_2\rangle$	$cer(\langle y_1,z_2\rangle) = \sigma(\langle y_1,z_2\rangle)/\sigma(y_1) = 0{,}070/0{,}100 = 0{,}700$
$\langle y_2,z_2\rangle$	$cer(\langle y_2,z_2\rangle) = \sigma(\langle y_2,z_2\rangle)/\sigma(y_2) = 0{,}450/0{,}450 = 1{,}000$
$\langle y_3,z_1\rangle$	$cer(\langle y_3,z_1\rangle) = \sigma(\langle y_3,z_1\rangle)/\sigma(y_3) = 0{,}120/0{,}170 = 0{,}705$
$\langle y_3,z_2\rangle$	$cer(\langle y_3,z_2\rangle) = \sigma(\langle y_3,z_2\rangle)/\sigma(y_3) = 0{,}050/0{,}170 = 0{,}294$
$\langle y_4,z_2\rangle$	$cer(\langle y_4,z_2\rangle) = \sigma(\langle y_4,z_2\rangle)/\sigma(y_4) = 0{,}280/0{,}280 = 1{,}000$

Definição 4.22 Considere um GFN. O *fator de cobertura* de um arco $\langle x,y\rangle\in\beta$, notado por $cov(\langle x,y\rangle)$, é gerado com foco no nó y do arco $\langle x,y\rangle$ e corresponde à uma medida associada ao valor y de determinado atributo que define o GFN, calculada pela Eq. (14).

$$cov(\langle x,y\rangle) = \frac{\sigma(\langle x,y\rangle)}{\sigma(y)} \qquad (14)$$

Se, por exemplo, $cov(\langle x,y\rangle)=0{,}5$, então há um atributo cujo valor associado x responde por 50% do fluxo de entrada (*inflow*) de y. O restante das entradas e coberturas de y vêm de outros arcos $\langle x,y\rangle\in I(y)$.

Como exemplo, no GFN GN, o fator de cobertura do arco $\langle x_2,y_2\rangle$ é dado por $cov(\langle x_2,y_2\rangle)=\sigma(\langle x_2,y_2\rangle)/\sigma(y_2)=0{,}350/0{,}450=0{,}777$. A Tabela 4.7 apresenta o fator de cobertura dos arcos de GN.

Tabela 0.7 Cálculo do fator de cobertura para os 12 arcos do GFN GN da Figura 4.3.

<x,y>	cov(<x,y>)
$\langle x_1,y_1\rangle$	$cov(\langle x_1,y_1\rangle) = \sigma(\langle x_1,y_1\rangle)/\sigma(y_1) = 0{,}100/0{,}100 = 1{,}000$
$\langle x_1,y_3\rangle$	$cov(\langle x_1,y_3\rangle) = \sigma(\langle x_1,y_3\rangle)/\sigma(y_3) = 0{,}170/0{,}170 = 1{,}000$
$\langle x_1,y_4\rangle$	$cov(\langle x_1,y_4\rangle) = \sigma(\langle x_1,y_4\rangle)/\sigma(y_4) = 0{,}130/0{,}280 = 0{,}464$
$\langle x_2,y_2\rangle$	$cov(\langle x_2,y_2\rangle) = \sigma(\langle x_2,y_2\rangle)/\sigma(y_2) = 0{,}350/0{,}450 = 0{,}777$
$\langle x_3,y_2\rangle$	$cov(\langle x_3,y_2\rangle) = \sigma(\langle x_3,y_2\rangle)/\sigma(y_2) = 0{,}100/0{,}450 = 0{,}222$
$\langle x_3,y_4\rangle$	$cov(\langle x_3,y_4\rangle) = \sigma(\langle x_3,y_4\rangle)/\sigma(y_4) = 0{,}150/0{,}280 = 0{,}535$
$\langle y_1,z_1\rangle$	$cov(\langle y_1,z_1\rangle) = \sigma(\langle y_1,z_1\rangle)/\sigma(z_1) = 0{,}030/0{,}150 = 0{,}200$
$\langle y_1,z_2\rangle$	$cov(\langle y_1,z_2\rangle) = \sigma(\langle y_1,z_2\rangle)/\sigma(z_2) = 0{,}070/0{,}850 = 0{,}082$
$\langle y_2,z_2\rangle$	$cov(\langle y_2,z_2\rangle) = \sigma(\langle y_2,z_2\rangle)/\sigma(z_2) = 0{,}450/0{,}850 = 0{,}529$
$\langle y_3,z_1\rangle$	$cov(\langle y_3,z_1\rangle) = \sigma(\langle y_3,z_1\rangle)/\sigma(z_1) = 0{,}120/0{,}150 = 0{,}800$
$\langle y_3,z_2\rangle$	$cov(\langle y_3,z_2\rangle) = \sigma(\langle y_3,z_2\rangle)/\sigma(z_2) = 0{,}050/0{,}850 = 0{,}058$
$\langle y_4,z_2\rangle$	$cov(\langle y_4,z_2\rangle) = \sigma(\langle y_4,z_2\rangle)/\sigma(z_2) = 0{,}280/0{,}850 = 0{,}329$

Exemplo 4.4 A Figura 4.4 apresenta o GFN GN, usado como exemplo, atualizado com os conceitos apresentados até o momento.

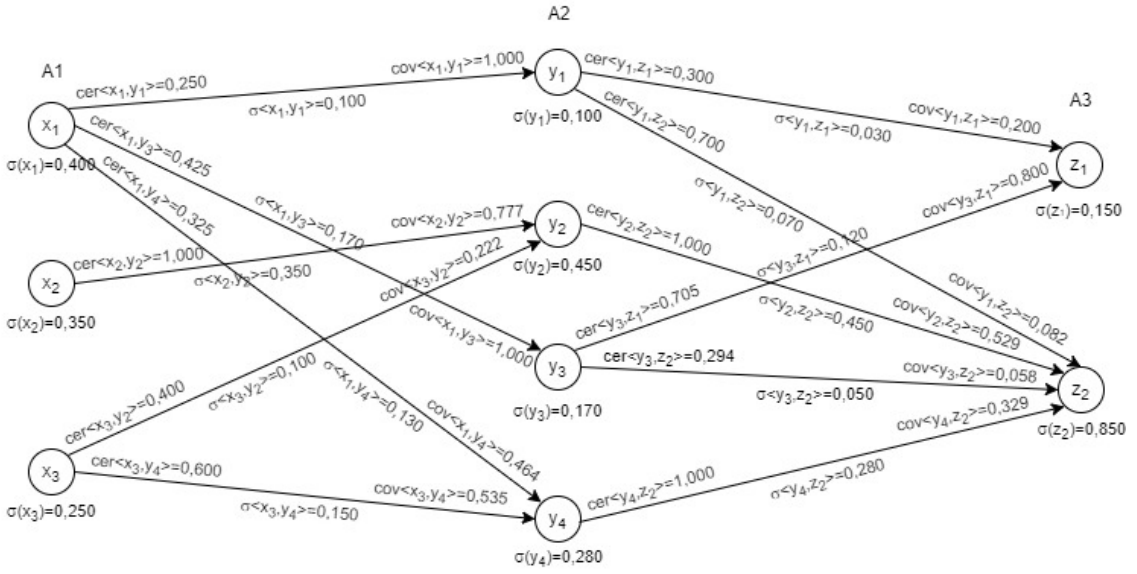


Figura 0.4 Grafo de Fluxo Normalizado (GFN) GN com os valores de fluxos de seus nós e os fatores de certeza, cobertura e força associados a cada um de seus arcos.

4.5 Grafos Inversos

Ao inverter a direção dos arcos de um GFN GN, obtém-se um grafo GN'=(N,β',σ') considerado o grafo inverso de GN (ver Figura 4.5). A análise de GN' viabiliza uma outra perspectiva do GFN original, uma vez que papéis (nós de entrada e nós de saída),

representados pelos nós de GN, são invertidos no grafo inverso GN'. Como comentado em [Pawlak 2010], o foco em GN' permite, a partir dos resultados, buscar *explicações* que os justifiquem. Naturalmente, GN' também é um GFN, porém com entradas e saídas invertidas, assim como o direcionamento de seus arcos. É importante notar que GN' possui os fatores de certeza, de cobertura e de força diferentes do GN original, uma vez que é um novo GFN.

Exemplo 4.5 A Figura 4.5 apresenta o GN', grafo inverso do GFN G, com seus fatores devidamente calculados, os quais são detalhados em Tabela 4.8, Tabela 4.9 e Tabela 4.10.

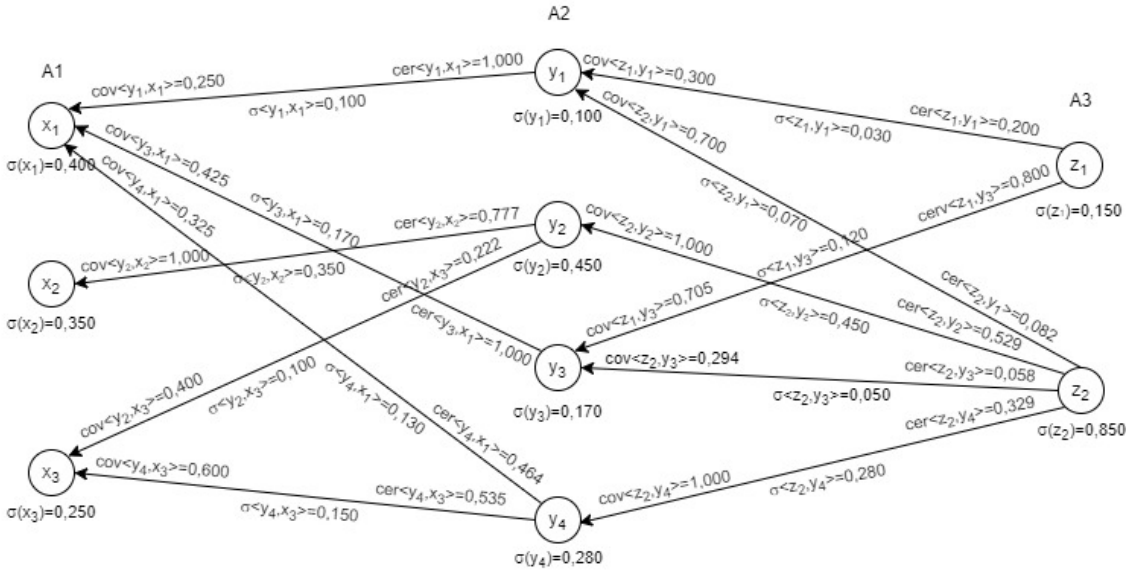


Figura 0.5 O Grafo de Fluxo GN', inverso do GFN GN.

O conjunto de nós N de GN' é o mesmo de GN, uma vez que não foram adicionados nem retirados nós de GN *i.e.*, $N=\{z_1,z_2,y_1,y_2,y_3,y_4,x_1,x_2,x_3\}$.

A inversão do direcionamento dos arcos de GN para a criação de GN' criou um novo conjunto de arcos β' associados a GN', notados por

$\beta'=\{<z_1,y_1>,<z_1,y_3>,<z_2,y_1>,<z_2,y_2>,<z_2,y_3>,<z_2,y_4>,<y_1,x_1>,<y_2,x_2>,<y_2,x_3>,<y_3,x_1>,<y_4,z_1>,<y_4,x_3>\}$.

Outra mudança que pode ser notada após a inversão de arcos de GN para a criação de GN' foi que os nós que compunham a entrada de um nó x de GN se tornaram nós de saída do mesmo nó x em GN', e de forma similar, os nós de saída de um nó x GN

se tornaram nós de entrada do nó x em GN'. Consequentemente, os *inflows* e *outflows* dos nós de GN' são diferentes dos *inflows* e *outflows* dos nós de GN.

A entrada e saída de GN' são dadas, respectivamente, por $I(GN')=\{z_1,z_2\}$ e $O(GN')=\{x_1,x_2,x_3\}$.

Os valores dos fatores de certeza dos arcos de GN' são apresentados na Tabela 4.8, os valores dos fatores de cobertura dos arcos de GN' são apresentados na Tabela 4.9 e os valores dos fatores de força dos arcos de GN' são apresentados na Tabela 4.10.

Tabela 0.5 Cálculo do fator de certeza para os 12 arcos do GFN GN' da Figura 4.5.

<x,y>	**cer(<x,y>)**
$<z_1,y_1>$	$cer(<z_1,y_1>) = \sigma(<z_1,y_1>)/\sigma(z_1) = 0,030/0,150 = 0,200$
$<z_1,y_3>$	$cer(<z_1,y_3>) = \sigma(<z_1,y_3>)/\sigma(z_1) = 0,120/0,150 = 0,800$
$<z_2,y_1>$	$cer(<z_2,y_1>) = \sigma(<z_2,y_1>)/\sigma(z_2) = 0,070/0,850 \cong 0,082$
$<z_2,y_2>$	$cer(<z_2,y_2>) = \sigma(<z_2,y_2>)/\sigma(z_2) = 0,450/0,850 \cong 0,529$
$<z_2,y_3>$	$cer(<z_2,y_3>) = \sigma(<z_2,y_3>)/\sigma(z_2) = 0,050/0,850 \cong 0,058$
$<z_2,y_4>$	$cer(<z_2,y_4>) = \sigma(<z_2,y_4>)/\sigma(z_2) = 0,280/0,850 \cong 0,329$
$<y_1,x_1>$	$cer(<y_1,x_1>) = \sigma(<y_1,x_1>)/\sigma(y_1) = 0,100/0,100 = 1,000$
$<y_2,x_2>$	$cer(<y_2,x_2>) = \sigma(<y_2,x_2>)/\sigma(y_2) = 0,350/0,450 \cong 0,777$
$<y_2,x_3>$	$cer(<y_2,x_3>) = \sigma(<y_2,x_3>)/\sigma(y_2) = 0,100/0,450 \cong 0,222$
$<y_3,x_1>$	$cer(<y_3,x_1>) = \sigma(<y_3,x_1>)/\sigma(y_3) = 0,170/0,170 = 1,000$
$<y_4,x_1>$	$cer(<y_4,x_1>) = \sigma(<y_4,x_1>)/\sigma(y_4) = 0,130/0,280 \cong 0,464$
$<y_4,x_3>$	$cer(<y_4,x_3>) = \sigma(<y_4,x_3>)/\sigma(y_4) = 0,150/0,280 \cong 0,535$

Tabela 0.6 Cálculo do fator de cobertura para os 12 arcos do GFN GN' da Figura 4.5.

<x,y>	**cov(<x,y>)**
$<z_1,y_1>$	$cov(<z_1,y_1>) = \sigma(<z_1,y_1>)/\sigma(y_1) = 0,030/0,100 = 0,300$
$<z_1,y_3>$	$cov(<z_1,y_3>) = \sigma(<z_1,y_3>)/\sigma(y_3) = 0,120/0,170 \cong 0,705$
$<z_2,y_1>$	$cov(<z_2,y_1>) = \sigma(<z_2,y_1>)/\sigma(y_1) = 0,070/0,100 = 0,700$
$<z_2,y_2>$	$cov(<z_2,y_2>) = \sigma(<z_2,y_2>)/\sigma(y_2) = 0,450/0,450 = 1,000$
$<z_2,y_3>$	$cov(<z_2,y_3>) = \sigma(<z_2,y_3>)/\sigma(y_3) = 0,050/0,170 \cong 0,294$
$<z_2,y_4>$	$cov(<z_2,y_4>) = \sigma(<z_2,y_4>)/\sigma(y_4) = 0,280/0,280 = 1,000$
$<y_1,x_1>$	$cov(<y_1,x_1>) = \sigma(<y_1,x_1>)/\sigma(x_1) = 0,100/0,400 = 0,250$
$<y_2,x_2>$	$cov(<y_2,x_2>) = \sigma(<y_2,x_2>)/\sigma(x_2) = 0,350/0,350 = 1,000$
$<y_2,x_3>$	$cov(<y_2,x_3>) = \sigma(<y_2,x_3>)/\sigma(x_3) = 0,100/0,250 = 0,400$
$<y_3,x_1>$	$cov(<y_3,x_1>) = \sigma(<y_3,x_1>)/\sigma(x_1) = 0,170/0,400 = 0,425$
$<y_4,x_1>$	$cov(<y_4,x_1>) = \sigma(<y_4,x_1>)/\sigma(x_1) = 0,130/0,400 = 0,325$
$<y_4,x_3>$	$cov(<y_4,x_3>) = \sigma(<y_4,x_3>)/\sigma(x_3) = 0,150/0,250 = 0,600$

Tabela 0.7 Cálculo do fator de força para os 12 arcos do GFN GN' da Figura 4.5.

<x,y>	**σ(<x,y>)**
$<z_1,y_1>$	$\sigma(z_1)cer(<z_1,y_1>) = \sigma(y_1)cov(<z_1,y_1>) = 0{,}150\times0{,}200 = 0{,}100\times0{,}300 = 0{,}030$
$<z_1,y_3>$	$\sigma(z_1)cer(<z_1,y_3>) = \sigma(y_3)cov(<z_1,y_3>) = 0{,}150\times0{,}800 \cong 0{,}170\times0{,}705 \cong 0{,}120$
$<z_2,y_1>$	$\sigma(z_2)cer(<z_2,y_1>) = \sigma(y_1)cov(<z_2,y_1>) = 0{,}850\times0{,}082 \cong 0{,}100\times0{,}700 \cong 0{,}070$
$<z_2,y_2>$	$\sigma(z_2)cer(<z_2,y_2>) = \sigma(y_2)cov(<z_2,y_2>) = 0{,}850\times0{,}529 \cong 0{,}450\times1{,}000 \cong 0{,}450$
$<z_2,y_3>$	$\sigma(z_2)cer(<z_2,y_3>) = \sigma(y_3)cov(<z_2,y_3>) = 0{,}850\times0{,}058 \cong 0{,}170\times0{,}294 \cong 0{,}450$
$<z_2,y_4>$	$\sigma(z_2)cer(<z_2,y_4>) = \sigma(y_4)cov(<z_2,y_4>) = 0{,}850\times0{,}329 \cong 0{,}280\times1{,}000 \cong 0{,}280$
$<y_1,x_1>$	$\sigma(y_1)cer(<y_1,x_1>) = \sigma(x_1)cov(<y_1,x_1>) = 0{,}100\times1{,}000 = 0{,}400\times0{,}250 = 0{,}100$
$<y_2,x_2>$	$\sigma(y_2)cer(<y_2,x_2>) = \sigma(x_2)cov(<y_2,x_2>) = 0{,}450\times0{,}777 \cong 0{,}350\times1{,}000 \cong 0{,}350$
$<y_2,x_3>$	$\sigma(y_2)cer(<y_2,x_3>) = \sigma(x_3)cov(<y_2,x_3>) = 0{,}450\times0{,}222 \cong 0{,}250\times0{,}400 \cong 0{,}100$
$<y_3,x_1>$	$\sigma(y_3)cer(<y_3,x_1>) = \sigma(x_1)cov(<y_3,x_1>) = 0{,}170\times1{,}000 = 0{,}400\times0{,}425 = 0{,}170$
$<y_4,x_1>$	$\sigma(y_4)cer(<y_4,x_1>) = \sigma(x_1)cov(<y_4,x_1>) = 0{,}280\times0{,}464 \cong 0{,}400\times0{,}325 \cong 0{,}130$
$<y_4,x_3>$	$\sigma(y_4)cer(<y_4,x_3>) = \sigma(x_3)cov(<y_4,x_3>) = 0{,}280\times0{,}535 \cong 0{,}250\times0{,}600 \cong 0{,}150$

4.6 Caminhos e Conexões

As definições contidas nesta subseção são base para o conteúdo contido na próxima subseção.

Definição 4.23 um *caminho* entre um nó x e um nó y, em um GFN G, é dado pela sequência de nós $x_1,\ldots,x_n$, em que $x_1=x$, $x_n=y$ e $<x_i,x_{i+1}>\in\beta$, $1\leq i\leq n-1$ e é denotado por [x,y].

Observação 4.6 De acordo com a Definição 4.23, qualquer arco <x,y>, $x\in N$ e $y\in N$, é um caminho no grafo de fluxo. Apesar de não estar incorreta a definição, estudos baseados em materiais (artigos) sobre grafos de fluxo, criados pelo próprio autor do formalismo, mostram que o autor define o termo caminho como na Definição 4.23 e, posteriormente, utiliza o termo caminho para referenciar um tipo específico de caminho, o qual se origina em um nó de entrada do grafo de fluxo e tem como destino um nó de saída do mesmo. Para evitar ambiguidade na utilização do termo em questão, uma nova definição se fez necessária.

Definição 4.24 Um *caminho completo* entre um nó x e um nó y em um GFN G é dado pela sequência de nós $x_1,\ldots,x_n$, tal que, $x_1\in I(G)$ e $x_n\in O(G)$, em que $x_1=x$, $x_n=y$ e $<x_i,x_{i+1}>\in\beta$, $1\leq i\leq n-1$ e é denotado por [x,y].

A Tabela 4.11 apresenta os caminhos completos do GFN GN (Figura 4.4).

Tabela 0.8 Caminhos completos encontrados no GFN GN da Figura 4.4. Na tabela, o índice na notação do caminho serve como identificação do caminho entre dois nós específicos.

[x,y]	conjunto de arcos
$[x_1,z_1]_1$	$\{<x_1,y_1>,<y_1,z_1>\}$
$[x_1,z_1]_2$	$\{<x_1,y_3>,<y_3,z_1>\}$
$[x_1,z_2]_1$	$\{<x_1,y_1>,<y_1,z_2>\}$
$[x_1,z_2]_2$	$\{<x_1,y_3>,<y_3,z_2>\}$
$[x_1,z_2]_3$	$\{<x_1,y_4>,<y_4,z_2>\}$
$[x_2,z_2]_1$	$\{<x_2,y_2>,<y_2,z_2>\}$
$[x_3,z_2]_1$	$\{<x_3,y_2>,<y_2,z_2>\}$
$[x_3,z_2]_2$	$\{<x_3,y_4>,<y_4,z_2>\}$

Para uma melhor exibição das próximas tabelas, a coluna *conjunto de arcos* (que compõem o caminho completo) será omitida.

Definição 4.25 Em um GFN GN, o fator de certeza associado a um caminho completo [x,y] em GN, denotado por cer([x,y]), é dado pela multiplicação dos fatores de certeza associados aos arcos que participam do caminho completo, como calculado pela Eq. (15).

$$\mathrm{cer}([x,y]) = \prod_{i=1}^{n-1} \mathrm{cer}(<x_i, x_{i+1}>) \qquad (15)$$

A Tabela 4.12 apresenta os caminhos completos do GFN GN da Figura 4.4 e seus respectivos fatores de certeza.

Tabela 0.9 Cálculo do fator de certeza cer([x,y]) para todos caminhos completos do GFN GN da Figura 4.4.

[x,y]	cer([x,y])
$[x_1,z_1]_1$	$\mathrm{cer}(<x_1,y_1>) \times \mathrm{cer}(<y_1,z_1>) = 0{,}250\times0{,}300 = 0{,}075$
$[x_1,z_1]_2$	$\mathrm{cer}(<x_1,y_3>) \times \mathrm{cer}(<y_3,z_1>) = 0{,}425\times0{,}705 \cong 0{,}300$
$[x_1,z_2]_1$	$\mathrm{cer}(<x_1,y_1>) \times \mathrm{cer}(<y_1,z_2>) = 0{,}250\times0{,}700 = 0{,}175$
$[x_1,z_2]_2$	$\mathrm{cer}(<x_1,y_3>) \times \mathrm{cer}(<y_3,z_2>) = 0{,}425\times0{,}294 \cong 0{,}125$
$[x_1,z_2]_3$	$\mathrm{cer}(<x_1,y_4>) \times \mathrm{cer}(<y_4,z_2>) = 0{,}325\times1{,}000 = 0{,}325$
$[x_2,z_2]_1$	$\mathrm{cer}(<x_2,y_2>) \times \mathrm{cer}(<y_2,z_2>) = 1{,}000\times1{,}000 = 1{,}000$
$[x_3,z_2]_1$	$\mathrm{cer}(<x_3,y_2>) \times \mathrm{cer}(<y_2,z_2>) = 0{,}400\times1{,}000 = 0{,}400$
$[x_3,z_2]_2$	$\mathrm{cer}(<x_3,y_4>) \times \mathrm{cer}(<y_4,z_2>) = 0{,}600\times1{,}000 = 0{,}600$

Observação 4.7 Para a Definição 4.25, a Observação 4.3 também se aplica.

Definição 4.26 Em um GFN GN, o fator de cobertura associado a um caminho completo [x,y] em GN, denotado por cov([x,y]), é dado pela multiplicação dos fatores de cobertura associados aos arcos que participam do caminho completo, como calculado pela Eq. (16).

$$\mathrm{cov}([x,y]) = \prod_{i=1}^{n-1} \mathrm{cov}(\langle x_i, x_{i+1} \rangle) \tag{16}$$

A Tabela 4.13 apresenta os caminhos completos de GN (Figura 4.4) e seus respectivos fatores de cobertura.

Tabela 0.10 Cálculo do fator de cobertura cov([x,y]) para todos caminhos completos do GFN GN da Figura 4.4.

[x,y]	cov([x,y])
$[x_1,z_1]_1$	$\mathrm{cov}(\langle x_1,y_1\rangle) \times \mathrm{cov}(\langle y_1,z_1\rangle) = 1{,}000 \times 0{,}200 = 0{,}200$
$[x_1,z_1]_2$	$\mathrm{cov}(\langle x_1,y_3\rangle) \times \mathrm{cov}(\langle y_3,z_1\rangle) = 1{,}000 \times 0{,}800 = 0{,}800$
$[x_1,z_2]_1$	$\mathrm{cov}(\langle x_1,y_1\rangle) \times \mathrm{cov}(\langle y_1,z_2\rangle) = 1{,}000 \times 0{,}082 \cong 0{,}152$
$[x_1,z_2]_2$	$\mathrm{cov}(\langle x_1,y_3\rangle) \times \mathrm{cov}(\langle y_3,z_2\rangle) = 1{,}000 \times 0{,}058 = 0{,}058$
$[x_1,z_2]_3$	$\mathrm{cov}(\langle x_1,y_4\rangle) \times \mathrm{cov}(\langle y_4,z_2\rangle) = 0{,}464 \times 0{,}329 \cong 0{,}152$
$[x_2,z_2]_1$	$\mathrm{cov}(\langle x_2,y_2\rangle) \times \mathrm{cov}(\langle y_2,z_2\rangle) = 0{,}777 \times 0{,}529 \cong 0{,}411$
$[x_3,z_2]_1$	$\mathrm{cov}(\langle x_3,y_2\rangle) \times \mathrm{cov}(\langle y_2,z_2\rangle) = 0{,}222 \times 0{,}529 \cong 0{,}117$
$[x_3,z_2]_2$	$\mathrm{cov}(\langle x_3,y_4\rangle) \times \mathrm{cov}(\langle y_4,z_2\rangle) = 0{,}535 \times 0{,}329 \cong 0{,}176$

Observação 4.8 Para a Definição 4.26, a Observação 4.3 também se aplica.

Definição 4.27 Em um GFN GN, o fator de força associado a um caminho completo [x,y] em GN, denotado por $\sigma([x,y])$, é dado pelas igualdades mostradas na Eq. (17).

$$\sigma([x,y]) = \sigma(x)\mathrm{cer}([x,y]) = \sigma(y)\mathrm{cov}([x,y]) \tag{17}$$

A Tabela 4.14 apresenta o fator de força para os caminhos completos de GN.

Tabela 0.11 Cálculo do fator de força σ([x,y]) para todos caminhos completos do GFN GN da Figura 4.4.

[x,y]	σ([x,y])
$[x_1,z_1]_1$	$\sigma(x_1)cer([x_1,z_1]_1) = \sigma(z_1)cov([x_1,z_1]_1) = 0{,}400\times0{,}075 = 0{,}150\times0{,}200 = 0{,}030$
$[x_1,z_1]_2$	$\sigma(x_1)cer([x_1,z_1]_2) = \sigma(z_1)cov([x_1,z_1]_2) = 0{,}400\times0{,}300 = 0{,}150\times0{,}800 = 0{,}120$
$[x_1,z_2]_1$	$\sigma(x_1)cer([x_1,z_2]_1) = \sigma(z_2)cov([x_1,z_2]_1) = 0{,}400\times0{,}175 = 0{,}850\times0{,}082 \cong 0{,}070$
$[x_1,z_2]_2$	$\sigma(x_1)cer([x_1,z_2]_2) = \sigma(z_2)cov([x_1,z_2]_2) = 0{,}400\times0{,}125 = 0{,}850\times0{,}058 \cong 0{,}050$
$[x_1,z_2]_3$	$\sigma(x_1)cer([x_1,z_2]_3) = \sigma(z_2)cov([x_1,z_2]_3) = 0{,}400\times0{,}325 = 0{,}850\times0{,}152 \cong 0{,}130$
$[x_2,z_2]_1$	$\sigma(x_2)cer([x_2,z_2]_1) = \sigma(z_2)cov([x_2,z_2]_1) = 0{,}350\times1{,}000 = 0{,}850\times0{,}411 \cong 0{,}350$
$[x_3,z_2]_1$	$\sigma(x_3)cer([x_3,z_2]_1) = \sigma(z_2)cov([x_3,z_2]_1) = 0{,}250\times0{,}400 = 0{,}850\times0{,}117 \cong 0{,}100$
$[x_3,z_2]_2$	$\sigma(x_3)cer([x_3,z_2]_2) = \sigma(z_2)cov([x_3,z_2]_2) = 0{,}250\times0{,}600 = 0{,}850\times0{,}176 \cong 0{,}150$

Observação 4.9 Para a Definição 4.27, a Observação 4.3 também se aplica.

Como visto, um caminho completo [x,y] é uma sequência de nós que tem origem em x e tem y como destino. Nos GFNs, podem existir vários caminhos completos.

Definição 4.28 Em um GFN GN, o conjunto de todos os caminhos completos [x,y], que saem de x e chegam em y, é chamado de *conjunto conexão* ou, simplesmente, *conexão*, e é denotado por ⟨x,y⟩.

A Tabela 4.15 exibe as conexões existentes no GFN GN da Figura 4.4.

Tabela 0.12 As conexões do GFN GN da Figura 4.4.

⟨x,y⟩	conjunto de caminhos completos
$\langle x_1,z_1\rangle$	$\{[x_1,z_1]_1,[x_1,z_1]_2\}$
$\langle x_1,z_2\rangle$	$\{[x_1,z_2]_1,[x_1,z_2]_2,[x_1,z_2]_3\}$
$\langle x_2,z_2\rangle$	$\{[x_2,z_2]_1\}$
$\langle x_3,z_2\rangle$	$\{[x_3,z_2]_1,[x_3,z_2]_2\}$

Observação 4.10 Para a Definição 4.28, a Observação 4.3 também se aplica.

Os fatores de certeza, de cobertura e de força também se aplicam às conexões, como definidos no que segue.

Definição 4.29 Em um GFN GN, o *fator de certeza associado à uma conexão* ⟨x,y⟩**,** cer(⟨x,y⟩)**,** é calculado como a soma dos fatores de certeza de cada um dos caminhos completos que pertence à conexão, como especifica a Eq. (18).

$$\text{cer}(\langle x,y\rangle) = \sum_{[x,y]\in\langle x,y\rangle} \text{cer}([x,y]) \tag{18}$$

A Tabela 4.16 apresenta o cálculo do fator de certeza associado a todas as conexões de GN.

Tabela 0.13 As conexões do GFN GN e seus fatores de certeza associados.

⟨x,y⟩	cer(⟨x,y⟩)
$\langle x_1,z_1\rangle$	$\text{cer}([x_1,z_1]_1) + \text{cer}([x_1,z_1]_2) = 0{,}075+0{,}300 = 0{,}375$
$\langle x_1,z_2\rangle$	$\text{cer}([x_1,z_2]_1) + \text{cer}([x_1,z_2]_2) + \text{cer}([x_1,z_2]_3) = 0{,}175+0{,}125+0{,}325 = 0{,}625$
$\langle x_2,z_2\rangle$	$\text{cer}([x_2,z_2]_1) = 1{,}000$
$\langle x_3,z_2\rangle$	$\text{cer}([x_3,z_2]_1) + \text{cer}([x_3,z_2]_2) = 0{,}400+0{,}600 = 1{,}000$

Observação 4.11 Para a Definição 4.29, a Observação 4.3 também se aplica.

Definição 4.30 Em um GFN GN, o *fator de cobertura associado à uma conexão* ⟨x,y⟩, cov(⟨x,y⟩)**,** é calculado como a soma dos fatores de cobertura de cada um dos caminhos completos que pertence à conexão, como especifica a Eq. (19).

$$\text{cov}(\langle x,y\rangle) = \sum_{[x,y]\in\langle x,y\rangle} \text{cov}([x,y]) \tag{19}$$

A Tabela 4.17 apresenta o cálculo do fator de cobertura associado a todas as conexões de GN.

Tabela 0.14 As conexões do GFN GN e seus fatores de cobertura associados.

⟨x,y⟩	cov(⟨x,y⟩)
$\langle x_1,z_1\rangle$	$\text{cov}([x_1,z_1]_1) + \text{cov}([x_1,z_1]_2) = 0{,}200+0{,}800 = 1{,}000$
$\langle x_1,z_2\rangle$	$\text{cov}([x_1,z_2]_1) + \text{cov}([x_1,z_2]_2) + \text{cov}([x_1,z_2]_3) = 0{,}152+0{,}058+0{,}152 = 0{,}362$
$\langle x_2,z_2\rangle$	$\text{cov}([x_2,z_2]_1) = 0{,}411$
$\langle x_3,z_2\rangle$	$\text{cov}([x_3,z_2]_1) + \text{cov}([x_3,z_2]_2) = 0{,}117+0{,}176 = 0{,}293$

Observação 4.12 Para a Definição 4.30, a Observação 4.3 também se aplica.

Definição 4.31 Em um GFN GN, o *fator de força associado à uma conexão* ⟨x,y⟩**,** σ(⟨x,y⟩)**,** é calculado como a soma dos fatores de força de cada um dos caminhos completos que pertence à conexão, como especifica a Eq. (20).

$$\sigma(\langle x,y\rangle) = \sum_{[x,y]\in\langle x,y\rangle} \sigma([x,y]) \qquad (20)$$

A Tabela 4.18 apresenta o cálculo do fator de força associado a todas as conexões de GN.

Tabela 4.18 As conexões do GFN GN e seus fatores de força associados.

$\langle x,y\rangle$	$\sigma(\langle x,y\rangle)$
$\langle x_1,z_1\rangle$	$\sigma([x_1,z_1]_1) + \sigma([x_1,z_1]_2) = 0{,}030+0{,}120 = 0{,}150$
$\langle x_1,z_2\rangle$	$\sigma([x_1,z_2]_1) + \sigma([x_1,z_2]_2) + \sigma([x_1,z_2]_3) = 0{,}070+0{,}050+0{,}130 = 0{,}250$
$\langle x_2,z_2\rangle$	$\sigma([x_2,z_2]_1) = 0{,}350$
$\langle x_3,z_2\rangle$	$\sigma([x_3,z_2]_1) + \sigma([x_3,z_2]_2) = 0{,}100+0{,}150 = 0{,}250$

Observação 4.13 Para a Definição 4.31, a Observação 4.3 também se aplica.

4.7 Fusão

A estrutura de GFN viabiliza o uso de uma importante operação de agregação de nós, chamada *fusão*, com o objetivo de sumarizar a estrutura. O uso de fusão em um GFN permite a criação de uma nova estrutura, GFNf, com a mesma representatividade do GFN, porém com foco nas informações sobre a distribuição de fluxo entre a entrada e saída do grafo ([Pawlak 2010] e [Pawlak 2005b]). A grande diferença entre dois grafos de fluxo normalizados, um GN e um GNf gerado a partir de GN, se deve ao GNf ser formado, apenas, por nós externos *i.e.*, nós de entrada e de saída. Os nós internos são retirados de modo que o GNf evidencie uma estrutura mais simplificada do fluxo dos dados de GN, com foco em suas entradas e saídas.

O processo de criação de um GNf a partir de um GN, utilizando a operação de fusão, substitui, no GN, todas as suas conexões $\langle x,y\rangle$**,** por arcos $<x,y>$. Obviamente, o conjunto β_f de GNf é diferente do conjunto β do GN. A remoção de nós internos induz a definição de arcos $<x,y>\in\beta_f$ que conectam os nós x e y tais que $x\in I(GNf)$ e $y\in O(GNf)$, ou seja, os nós de entrada do GFN original (respeitando, obviamente, a distribuição do fluxo dos dados) passam a estar conectados diretamente, por meio de arcos, aos nós de saída do GFN original.

Os fatores de certeza, de cobertura e de força, definidos para GFNs, também se aplicam a GFNfs. É importante ressaltar, ainda, que como um GNf é abordado como uma sumarização de um GN, a representatividade do GNf deve refletir aquela do GN.

Os valores dos fatores calculados para o GN, entretanto, não são os mesmos que aqueles calculados para o GNf, uma vez que as estruturas e, principalmente, o fluxo dos dados, são diferentes em ambos os grafos. É preciso, portanto, calcular os valores dos fatores do GNf resultante do processo de fusão aplicado a um GN. Porém, se tal cálculo for realizado somente com base em GNf, só serão levados em consideração seus fluxos, o que faz com que a representação GNf deixe de preservar a representatividade do GN a partir do qual se originou.

Para que GNf tenha a mesma representatividade do GN a partir do qual foi originado, é preciso que o cálculo de seus fatores de certeza, de cobertura e de força levem em consideração os fatores do GN. Como foram substituídas as conexões $\langle x,y\rangle$ de GN, por arcos $<x,y>$, em GNf, então $cer(<x,y>)=cer(\langle x,y\rangle)$, ou seja, o valor do fator de certeza do arco $<x,y>$ em GNf, $cer(<x,y>)$, será o valor de certeza da conexão $\langle x,y\rangle$ em GN *i.e.*, $cer(\langle x,y\rangle)$**.** O mesmo deve acontecer com os fatores de cobertura e de força. Sendo assim, é verdade que para um arco $<x,y>$ de GNf, $cov(<x,y>)=cov(\langle x,y\rangle)$, assim como, $\sigma(<x,y>)=\sigma(\langle x,y\rangle)$. Desta forma, o GNf gerado por meio do processo de fusão tem a mesma representatividade do GN original.

Exemplo 4.6 A Figura 4.6 apresenta o GNf gerado a partir da fusão do GN da Figura 4.4. Participam do GNf somente os nós externos de GN. Os arcos que conectam os nós de GNf substituíram as conexões existentes (*e.g.*, a conexão $\langle \mathbf{x_1,z_1} \rangle = \{[x_1,z_1]_1,[x_1,z_1]_2\}$ foi substituída por um arco $<x_1,z_1>$ no GNf). Os fatores de certeza, cobertura e força da conexão $\langle x_1,z_1\rangle$ de GN tornaram-se, respectivamente, os fatores de certeza, cobertura e força do arco $<x_1,z_1>$ de GNf.

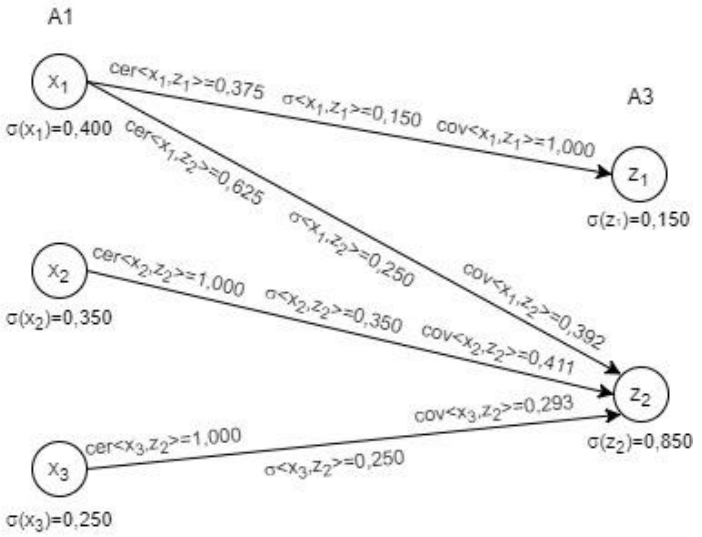


Figura 0.6 Grafo GNf resultante do processo de fusão aplicado ao GN da Figura 4.4.

4.8 Grafos de Fluxo e Algoritmos de Decisão

Grafos de Fluxo representam, de forma simplificada, a distribuição do fluxo das instâncias de dados entre seus nós. Cada nó representa os possíveis valores dos atributos que descrevem as instâncias de dados consideradas.

Definição 4.32 Uma *regra de decisão* $x^* \rightarrow y$ é uma regra formada por dois termos, x* e y, em que x* representa uma sequência de regras de decisão, notada por $x_1,x_2,\ldots,x_{n-2} \rightarrow x_{n-1}$ e $y=x_n$, tal que, se x_1 for verdadeiro, a decisão é verificar se x_2 é verdadeiro e assim por diante até x_{n-1}. Se todas as regras de decisão de x_1 até x_{n-1} incluindo x_{n-1}, forem verdadeiras, então a decisão a ser tomada é dada por x_n (*i.e.* y).

De acordo com a Definição 4.32, dado um arco $\langle x,y\rangle \in \beta$ de um GFN, há uma regra de decisão $x \rightarrow y$, em que x é a sequência de regras de decisão (no caso formada somente por x) e y é a decisão final da regra de decisão. Um caminho completo [x,y], assim como uma conexão ⟨x,y⟩, ambos de um GFN, também podem ser considerados *regras de decisão*.

Definição 4.33 O conjunto de regras de decisão $x^* \rightarrow y$ de um GFN GN, sendo $x \in I(GN)$ e $y \in O(GN)$, é chamado de *algoritmo de decisão* induzido pelo GF e representa o fluxo dos dados do GN desde suas entradas até suas saídas.

A Tabela 4.19 mostra o algoritmo de decisão (AD) extraído a partir do GFN GN da Figura 4.4, referenciado, a partir deste ponto, por AD.

Tabela 4.19 O algoritmo de decisão AD extraído a partir do GFN GN da Figura 4.4.

regras de decisão
$x_1,y_1 \rightarrow z_1$
$x_1,y_1 \rightarrow z_2$
$x_1,y_3 \rightarrow z_1$
$x_1,y_3 \rightarrow z_2$
$x_1,y_4 \rightarrow z_2$
$x_2,y_2 \rightarrow z_2$
$x_3,y_2 \rightarrow z_2$
$x_3,y_4 \rightarrow z_2$

Os fatores de certeza, de cobertura e de força também estão associados às regras de decisão que compõem os ADs extraídos de GFs.

Definição 4.34 O valor do *fator de certeza de uma regra de decisão* $x^* \rightarrow y$ é dado pela Eq. (21).

$$\mathrm{cer}(x^* \rightarrow y) = \frac{\sigma(x)\mathrm{cer}([x,x_n])}{\sigma(x)\mathrm{cer}([x,x_{n-1}])} = \mathrm{cer}(x_n), \text{ em que } x_n = y \qquad (21)$$

Por exemplo, a regra de decisão $x_2,y_2 \rightarrow z_2$, do algoritmo de decisão AD mostrado na Tabela 4.19, tem fator de certeza com valor 1,000, pois

$$\mathrm{cer}(x_2, y_2 \rightarrow z_2) = \frac{\sigma(x_2)\mathrm{cer}([x_2,z_2]_1)}{\sigma(x_2)\mathrm{cer}([x_2,y_2]_1)} = \frac{0{,}350 \times 1{,}000}{0{,}350 \times 1{,}000} = 1{,}000 = \mathrm{cer}(z_2).$$

A Tabela 4.20 apresenta os fatores de certeza para cada uma das regras de decisão do AD mostrado na Tabela 4.19.

Tabela 4.20 Cálculo do valor do fator de certeza, cer(x*→z), para cada uma das regras de decisão do AD da Tabela 4.19.

regra	cer(x*→z)
$x_1,y_1 \rightarrow z_1$	$cer(x_1,y_1 \rightarrow z_1) = \sigma(x_1)cer([x_1,z_1]_1) / \sigma(x_1)cer([x_1,y_1]) = (0{,}400 \times 0{,}075) / (0{,}400 \times 0{,}250) = 0{,}300$
$x_1,y_1 \rightarrow z_2$	$cer(x_1,y_1 \rightarrow z_2) = \sigma(x_1)cer([x_1,z_2]_2) / \sigma(x_1)cer([x_1,y_1]) = (0{,}400 \times 0{,}175) / (0{,}400 \times 0{,}250) = 0{,}700$
$x_1,y_3 \rightarrow z_1$	$cer(x_1,y_3 \rightarrow z_1) = \sigma(x_1)cer([x_1,z_1]_1) / \sigma(x_1)cer([x_1,y_3]) = (0{,}400 \times 0{,}300) / (0{,}400 \times 0{,}425) = 0{,}705$
$x_1,y_3 \rightarrow z_2$	$cer(x_1,y_3 \rightarrow z_2) = \sigma(x_1)cer([x_1,z_2]_2) / \sigma(x_1)cer([x_1,y_3]) = (0{,}400 \times 0{,}125) / (0{,}400 \times 0{,}425) = 0{,}294$
$x_1,y_4 \rightarrow z_2$	$cer(x_1,y_4 \rightarrow z_2) = \sigma(x_1)cer([x_1,z_2]_3) / \sigma(x_1)cer([x_1,y_4]) = (0{,}400 \times 0{,}325) / (0{,}400 \times 0{,}325) = 1{,}000$
$x_2,y_2 \rightarrow z_2$	$cer(x_2,y_2 \rightarrow z_2) = \sigma(x_2)cer([x_2,z_2]_1) / \sigma(x_2)cer([x_2,y_2]) = (0{,}350 \times 1{,}000) / (0{,}350 \times 1{,}000) = 1{,}000$
$x_3,y_2 \rightarrow z_2$	$cer(x_3,y_2 \rightarrow z_2) = \sigma(x_3)cer([x_3,z_2]_1) / \sigma(x_3)cer([x_3,y_2]) = (0{,}250 \times 0{,}400) / (0{,}250 \times 0{,}400) = 1{,}000$
$x_3,y_4 \rightarrow z_2$	$cer(x_3,y_4 \rightarrow z_2) = \sigma(x_3)cer([x_3,z_2]_2) / \sigma(x_3)cer([x_3,y_4]) = (0{,}250 \times 0{,}600) / (0{,}250 \times 0{,}600) = 1{,}000$

Definição 4.35 O valor do *fator de cobertura* de uma regra de decisão (x*→z) é dado pela Eq. (22).

$$\mathrm{cov}(x^* \rightarrow y) = \frac{\sigma(y)\mathrm{cov}([x, x_n])}{\sigma(y)} = \mathrm{cov}([x, x_n]), \text{em que } x_n = y \qquad (22)$$

Por exemplo, a regra de decisão $x_1,y_1 \rightarrow z_1$ do algoritmo de decisão AD, da Tabela 4.19, tem como valor de seu fator de cobertura 0,200, uma vez que:

$$\mathrm{cov}(x_1, y_1 \rightarrow z_1) = \frac{\sigma(z_1)\mathrm{cov}([x_1,z_1]_1)}{\sigma(z_1)} = \frac{0{,}150 \text{ x } 0{,}200}{0{,}150} = 0{,}200 = \mathrm{cov}([x_1, z_1]_1).$$

Os valores do fator de cobertura para as regras de decisão do AD da Tabela 4.19 são mostrados na Tabela 4.21.

Tabela 4.21 Cálculo do valor do fator de cobertura, cov(x*→z), para cada uma das regras de decisão do AD da Tabela 4.19.

regra	cov(x*→z)
$x_1,y_1 \rightarrow z_1$	$cov(x_1,y_1 \rightarrow z_1) = \sigma(z_1)cov([x_1,z_1]_1) / \sigma(z_1) = (0{,}150 \times 0{,}200) / 0{,}150 = 0{,}200$
$x_1,y_1 \rightarrow z_2$	$cov(x_1,y_1 \rightarrow z_2) = \sigma(z_2)cov([x_1,z_2]_2) / \sigma(z_2) = (0{,}850 \times 0{,}082) / 0{,}850 = 0{,}082$
$x_1,y_3 \rightarrow z_1$	$cov(x_1,y_3 \rightarrow z_1) = \sigma(z_1)cov([x_1,z_1]_1) / \sigma(z_1) = (0{,}150 \times 0{,}800) / 0{,}150 = 0{,}800$
$x_1,y_3 \rightarrow z_2$	$cov(x_1,y_3 \rightarrow z_2) = \sigma(z_2)cov([x_1,z_2]_2) / \sigma(z_2) = (0{,}850 \times 0{,}058) / 0{,}850 = 0{,}058$
$x_1,y_4 \rightarrow z_2$	$cov(x_1,y_4 \rightarrow z_2) = \sigma(z_2)cov([x_1,z_2]_3) / \sigma(z_2) = (0{,}850 \times 0{,}152) / 0{,}850 = 0{,}152$
$x_2,y_2 \rightarrow z_2$	$cov(x_2,y_2 \rightarrow z_2) = \sigma(z_2)cov([x_2,z_2]_1) / \sigma(z_2) = (0{,}850 \times 0{,}411) / 0{,}850 = 1{,}000$
$x_3,y_2 \rightarrow z_2$	$cov(x_3,y_2 \rightarrow z_2) = \sigma(z_2)cov([x_3,z_2]_1) / \sigma(z_2) = (0{,}850 \times 0{,}117) / 0{,}850 = 0{,}117$
$x_3,y_4 \rightarrow z_2$	$cov(x_3,y_4 \rightarrow z_2) = \sigma(z_2)cov([x_3,z_2]_2) / \sigma(z_2) = (0{,}850 \times 0{,}176) / 0{,}850 = 0{,}176$

Definição 4.36 O valor do *fator de força* de uma regra de decisão $x^* \rightarrow y$ é dado pela Eq. (23).

$$\sigma(x^* \rightarrow y) = \sigma(x)\ \mathrm{cer}([x, y]) = \sigma(y)\ \mathrm{cov}([x, y]) \qquad (23)$$

Por exemplo, no GFN GN (Figura 4.4), há a regra de decisão $x_1, y_3 \rightarrow z_1$. O valor do fator de força dessa regra de decisão é dado por:

$$\sigma(x_1, y_3 \rightarrow z_1) = \sigma(x_1)\mathrm{cer}([x_1, z_1]_2) = 0{,}400 \times 0{,}300 = 0{,}120 \text{ ou}$$

$$\sigma(x_1, y_3 \rightarrow z_1) = \sigma(z_1)\mathrm{cov}([x_1, z_1]_2) = 0{,}150 \times 0{,}800 = 0{,}120.$$

A Tabela 4.22 apresenta os valores de fatores de força para as regras de decisão do AD da Tabela 4.19.

Tabela 4.22 Cálculo do valor do fator de força certeza, σ(x*→z), para cada uma das regras de decisão do AD da Tabela 4.19.

regra	**σ(x*→z)**
$x_1,y_1 \rightarrow z_1$	$\sigma(x_1,y_1 \rightarrow z_1) = \sigma(x_1)\mathrm{cer}([x_1,z_1]_1) = \sigma(z_1)\mathrm{cov}([x_1,z_1]_1) = (0{,}400 \times 0{,}075) = (0{,}150 \times 0{,}200) = 0{,}030$
$x_1,y_1 \rightarrow z_2$	$\sigma(x_1,y_1 \rightarrow z_2) = \sigma(x_1)\mathrm{cer}([x_1,z_2]_1) = \sigma(z_2)\mathrm{cov}([x_1,z_1]_1) = (0{,}400 \times 0{,}175) = (0{,}850 \times 0{,}082) = 0{,}070$
$x_1,y_3 \rightarrow z_1$	$\sigma(x_1,y_3 \rightarrow z_1) = \sigma(x_1)\mathrm{cer}([x_1,z_1]_2) = \sigma(z_1)\mathrm{cov}([x_1,z_1]_2) = (0{,}400 \times 0{,}300) = (0{,}150 \times 0{,}800) = 0{,}120$
$x_1,y_3 \rightarrow z_2$	$\sigma(x_1,y_3 \rightarrow z_2) = \sigma(x_1)\mathrm{cer}([x_1,z_2]_2) = \sigma(z_2)\mathrm{cov}([x_1,z_2]_2) = (0{,}400 \times 0{,}125) = (0{,}850 \times 0{,}058) = 0{,}050$
$x_1,y_4 \rightarrow z_2$	$\sigma(x_1,y_4 \rightarrow z_2) = \sigma(x_1)\mathrm{cer}([x_1,z_2]_3) = \sigma(z_2)\mathrm{cov}([x_1,z_2]_3) = (0{,}400 \times 0{,}325) = (0{,}850 \times 0{,}152) = 0{,}130$
$x_2,y_2 \rightarrow z_2$	$\sigma(x_2,y_2 \rightarrow z_2) = \sigma(x_2)\mathrm{cer}([x_2,z_2]_1) = \sigma(z_2)\mathrm{cov}([x_2,z_2]_1) = (0{,}350 \times 1{,}000) = (0{,}850 \times 0{,}411) = 0{,}350$
$x_3,y_2 \rightarrow z_2$	$\sigma(x_3,y_2 \rightarrow z_2) = \sigma(x_3)\mathrm{cer}([x_3,z_2]_1) = \sigma(z_2)\mathrm{cov}([x_3,z_2]_1) = (0{,}250 \times 0{,}400) = (0{,}850 \times 0{,}117) = 0{,}100$
$x_3,y_4 \rightarrow z_2$	$\sigma(x_3,y_4 \rightarrow z_2) = \sigma(x_3)\mathrm{cer}([x_3,z_2]_2) = \sigma(z_2)\mathrm{cov}([x_3,z_2]_2) = (0{,}250 \times 0{,}600) = (0{,}850 \times 0{,}176) = 0{,}150$

Exemplo 4.7 A Tabela 4.23 resume as regras de decisão do AD da Tabela 4.19, apresentando os valores dos fatores de certeza, de cobertura e de força para cada uma de suas regras de decisão.

Tabela 4.23 Regras do algoritmo de decisão AD da Tabela 4.19 e respectivos valores dos fatores de certeza, de cobertura e de força.

regra de decisão	**cer(x*→z)**	**cov(x*→z)**	**σ(x*→z)**
$x_1,y_1 \rightarrow z_1$	0,300	0,200	0,030
$x_1,y_1 \rightarrow z_2$	0,700	0,082	0,070
$x_1,y_3 \rightarrow z_1$	0,705	0,800	0,120
$x_1,y_3 \rightarrow z_2$	0,295	0,058	0,050
$x_1,y_4 \rightarrow z_2$	1,000	0,152	0,130
$x_2,y_2 \rightarrow z_2$	1,000	1,000	0,350
$x_3,y_2 \rightarrow z_2$	1,000	0,117	0,100
$x_3,y_4 \rightarrow z_2$	1,000	0,176	0,150

As regras de decisão de AD e seus respectivos fatores evidenciam que:

- Dentre as instâncias de dados descritas por $A1=x_1$ e $A2=y_1$, com certeza 30% delas têm $A3=z_1$ e o restante, 70%, tem $A3=z_2$;
- Dentre as instâncias de dados descritas por $A1=x_1$ e $A2=y_3$, com certeza 70,5% delas têm $A3=z_1$ e o restante, 29,5%, tem $A3=z_2$;
- Dentre as instâncias de dados descritas por $A1=x_1$ e $A2=y_4$, ou $A1=x_2$ e $A2=y_2$, ou $A1=x_3$ e $A2=y_2$, ou $A1=x_3$ e $A2=y_4$, é certeza que todas (100%) têm $A3=z_2$.

Claramente, o algoritmo de decisão extraído a partir do GFN GN é um algoritmo extremamente simples, porém, ainda pode ser melhorado. Regras de decisão que possuem o valor 1 (= 100%) associado ao fator de certeza podem ser sumarizadas com vistas à simplificação do algoritmo de decisão.

Observe o conjunto de regras de decisão RD cujo fator de certeza $cer(x^* \rightarrow z)=1$ (100%) *i.e.*, $RD=\{\{x_1,y_4 \rightarrow z_2\},\{x_2,y_2 \rightarrow z_2\},\{x_3,y_2 \rightarrow z_2\},\{x_3,y_4 \rightarrow z_2\}\}$. É evidente que todas regras de decisão do conjunto RD têm como decisão $A3=z_2$. É possível dividir o conjunto RD em dois subconjuntos RD2 e RD4 tal que RD2 contenha regras de decisão que possuam $A2=y_2$ (*i.e.*, $RD2=\{\{x_2,y_2 \rightarrow z_2\},\{x_3,y_2 \rightarrow z_2\}\}$) e RD4 contenha regras de decisão nas quais $A2=y_4$ (*i.e.*, $RD4=\{\{x_1,y_4 \rightarrow z_2\},\{x_3,y_4 \rightarrow z_2\}\}$).

Nota-se em RD2 que uma regra de decisão com $A2=y_2$ tem decisão z_2. Sendo assim, é fato que, independentemente do valor de A1, se uma instância possui $A2=y_2$, a mesma terá $A3=z_2$. Assumindo essa assertiva como verdadeira, as regras de decisão $x^* \rightarrow y$ que compõem as regras de decisão $x^* \rightarrow z$ contidas em RD2, podem ser substituídas por uma simples regra de decisão, dada por $y \rightarrow z$. No caso de RD2, todas suas regras de decisão $x^* \rightarrow z$ podem ser substituídas pela regra de decisão $y_2 \rightarrow z_2$ *i.e.*, o atributo A1 pode ser ignorado, criando assim uma regra de decisão mais simples que, no entanto, ainda mantém a mesma representatividade das regras de decisão originais para $A2=y_2$.

A mesma ação é possível para as regras de decisão de RD4. Elas também podem ser substituídas por uma única regra de decisão $y \rightarrow z$, de forma a simplificar o algoritmo de decisão. Mais especificamente, as regras de decisão de RD4 podem ser substituídas pela regra de decisão $y_4 \rightarrow z_2$ sem perda de representatividade para o algoritmo de decisão como um todo.

Ao simplificar uma regra de decisão, o fator de cobertura, cov(x*→z), da nova regra de decisão deve ter como valor, a soma dos fatores de cobertura de todas as regras substituídas no processo de simplificação, com limite igual 1 (100%). O mesmo vale para o fator de força, σ(x*→z), da nova regra de decisão.

Obviamente, o algoritmo de decisão simplificado AD_s, constituído pelas seis regras apresentadas na Tabela 4.24, não pode ser induzido *diretamente* do GFN GN, pois modificações foram feitas (*e.g.* para GN, é verdade que $I(GN)=\{x_1,x_2,x_3\}$; porém, se um novo GFN G_s for criado com base em AD_s, a entrada do mesmo será dada por $I(G_s)=\{x_1,y_2,y_4\}$). Entretanto, apesar das alterações, G_s mantém a representatividade de GN, *i.e.*, as regras de decisão de G_s são equivalentes às regras de decisão de AD (extraído a partir de GN).

Exemplo 4.8 A versão simplificada AD_s do algoritmo de decisão AD (Tabela 4.23) é apresentada na Tabela 4.24.

Tabela 4.24 A versão simplificada AD_s do algoritmo de decisão AD.

regras	cer(x*→z)	cov(x*→z)	σ(x*→z)
$x_1,y_1 \rightarrow z_1$	0,300	0,200	0,030
$x_1,y_1 \rightarrow z_2$	0,700	0,082	0,070
$x_1,y_3 \rightarrow z_1$	0,705	0,800	0,120
$x_1,y_3 \rightarrow z_2$	0,295	0,058	0,050
$y_2 \rightarrow z_2$	1,000	1,000	0,450
$y_4 \rightarrow z_2$	1,000	0,328	0,280

4.9 Dependência entre Nós em um GFN

Como citado, no início da Seção 4.4, há ainda um quarto fator (ou coeficiente) em Grafos de Fluxo denominado *dependência entre os nós*, que é aplicado aos arcos de um GFN. Sua função é evidenciar a dependência (negativa ou positiva) ou independência (se for o caso) dos nós que compõem arcos $<x,y> \in \beta$. Esse fator pondera a relação (arco) entre os dois valores associados de atributos distintos (*e.g.* $A1=x_3$ e $A2=y_3$), de forma que o valor obtido possa ser utilizado para auxiliar na definição da viabilidade da regra de decisão em questão. Ao calcular o fator de dependência de um arco <x,y>, é possível visualizar a importância de cada uma das regras de decisão que participam de um AD.

Definição 4.37 Considere um GFN e considere um arco <x,y> de tal grafo. As seguintes desigualdades caracterizam modalidades de dependências entre os nós que definem os arcos:

(a) Se cer(<x,y>)>σ(y) ou cov(<x,y>)>σ(x), então existe uma dependência positiva entre x e y, *i.e.*, x e y são positivamente dependentes.

(b) Se cer(<x,y>)<σ(y) ou cov(<x,y>)<σ(x), então existe uma dependência negativa entre x e y, *i.e.*, x e y são negativamente dependentes.

(c) Se cer(<x,y>)=σ(y) e/ou cov(<x,y>)=σ(x), x e y são independentes.

Definição 4.38 Considere um GFN e considere um arco <x,y>∈β. O cálculo do valor do fator de dependência de <x,y>, denotado por η(<x,y>), é dado pela Eq. (24).

$$\eta(<x,y>) = \frac{\text{cer}(<x,y>) - \sigma(y)}{\text{cer}(<x,y>) + \sigma(y)} = \frac{\text{cov}(<x,y>) - \sigma(x)}{\text{cov}(<x,y>) + \sigma(x)} \qquad (24)$$

Note que o valor de dependência entre os nós que participam de um arco varia no intervalo real [−1 1], ou seja, −1≤η(<x,y>)≤1.

Os valores −1, 0 e 1 para o fator de dependência somente serão possíveis nos casos específicos que seguem:

(a) se cer(<x,y>)=σ(y) e cov(<x,y>)=σ(x), então η(<x,y>)=0;

(b) se cer(<x,y>)=cov(<x,y>)=0, então η(<x,y>)= −1, e;

(c) se σ(y)=σ(x)=0, então η(<x,y>)=1.

Por outro lado, se:

(a) η(<x,y>)=0, então *x* e *y* são *independentes*;

(b) −1≤η(<x,y>)≤0, então *x* e *y* são *negativamente dependentes*, e;

(c) 0≤η(<x,y>)≤1, então *x* e *y* são *positivamente dependentes*.

O fato de η(<x,y>)=0 implica em uma regra de decisão com atributos x e y independentes, o que não é interessante para um algoritmo de decisão por não dar

indícios de influência de um atributo sobre outro. Da mesma forma, conjuntos de regras de decisão que apresentam baixa dependência (*i.e.,* dependência próxima de zero) também não são de grande valia, uma vez que a chance de erro em uma decisão é grande para ambos casos. Como apontado em [Pawlak 2004c], do ponto de vista dos dados, quanto mais próxima de zero for a dependência, menos provável é a relação entre os atributos envolvidos, *i.e.*, não é possível afirmar que o atributo x influencia o valor do atributo y. Do ponto de vista das dependências, as regras realmente importantes são aquelas que possuem fatores de dependência entre nós bem definidos e distantes de zero, positiva ou negativamente. A Tabela 4.25 apresenta os valores dos fatores de dependência entre nós para as regras de decisão do algoritmo de decisão induzido a partir do GNf (ver Figura 4.6). Note que, neste caso específico, $\eta(x \rightarrow z)$ é equivalente à $\eta(<x,y>)$, uma vez que o GNf é formado apenas por nós externos.

Tabela 4.25 As regras de decisão induzidas do GNf e os fatores de dependência entre seus atributos.

regra	$\boldsymbol{\eta(x \rightarrow z)}$
$x_1 \rightarrow z_1$	$\eta(x_1 \rightarrow z_1) = \frac{cer(<x_1,z_1>) - \sigma(z_1)}{cer(<x_1,z_1>) + \sigma(z_1)} = \frac{0,375-0,150}{0,375+0,150} = 0,428$
$x_1 \rightarrow z_2$	$\eta(x_1 \rightarrow z_2) = \frac{cer(<x_1,z_2>) - \sigma(z_2)}{cer(<x_1,z_2>) + \sigma(z_2)} = \frac{0,625-0,850}{0,625+0,850} = -0,152$
$x_2 \rightarrow z_2$	$\eta(x_2 \rightarrow z_2) = \frac{cer(<x_2,z_2>) - \sigma(z_2)}{cer(<x_2,z_2>) + \sigma(z_2)} = \frac{1,000-0,850}{1,000+0,850} = 0,081$
$x_3 \rightarrow z_2$	$\eta(x_3 \rightarrow z_2) = \frac{cer(<x_3,z_2>) - \sigma(z_2)}{cer(<x_3,z_2>) + \sigma(z_2)} = \frac{1,000-0,850}{1,000+0,850} = 0,081$

Os fatores de dependência apresentados na Tabela 4.25 indicam que a regra de decisão mais representativa (*i.e.*, mais distante de zero) é a regra de decisão $x_1 \rightarrow z_1$ seguida pela regra de decisão $x_1 \rightarrow z_2$. As outras regras de decisão possuem valores dos respectivos fatores de dependência próximos do valor zero. A Figura 4.7 exibe o GNf apresentado na Figura 4.6 após a inclusão do fator de dependência entre nós para cada uma de suas regras de decisão.

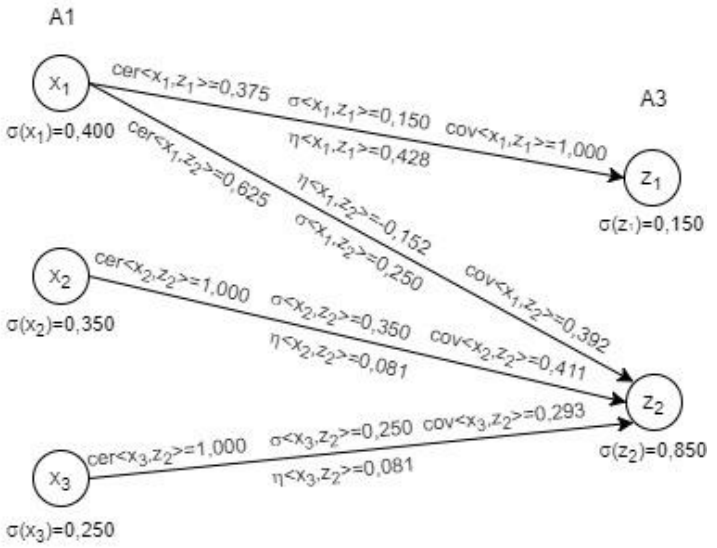


Figura 0.7 O GNf (gerado a partir da fusão do GN) com seus quatro fatores definidos.

Dessa forma, o algoritmo de decisão induzido a partir de GNf não seria um algoritmo de decisão muito representativo do ponto de vista do fator de dependência entre nós. É preciso analisar o fator de certeza das regras de decisão (e seu potencial de classificação) para poder avaliar a viabilidade do algoritmo de decisão, mais especificamente, de suas regras de decisão.

O fator de dependência entre nós pode ser utilizado, também, para escolher regras de decisão menos significativas a serem removidas de um algoritmo de decisão, de forma a torná-lo mais simples e, assim, com melhor poder de classificação (ver [Pawlak 2005b]).

4.10 Considerações Finais

Neste capítulo, foi apresentada uma estrutura de representação de conhecimento denominada Grafos de Fluxo (GF). O capítulo apresenta, inicialmente, um sumário dos principais conceitos e propriedades envolvidos no estabelecimento de uma estrutura GF e adota, para isso, uma abordagem didática subsidiada por exemplos, com vistas à familiarização progressiva do leitor com a notação e com os muitos conceitos formalmente introduzidos.

Devido ao fato da teoria sobre grafos de fluxo ter sido desenvolvida ao longo de um trabalho que levou vários anos, muitos dos conceitos envolvidos tiveram revisões e

reescritas, como podem atestar inúmeras definições encontradas na literatura, que diferem ligeiramente, umas das outras, até mesmo aquelas relativas ao próprio conceito básico de grafos de fluxo. Pequenos problemas, particularmente de consistência, com o formalismo e com as conceituações básicas, foram evidenciados ao longo da leitura e estudo sobre GFs e fazem parte desse documento; cada um deles é identificado como uma Observação que, com vistas à organização e referência futura, foram enumeradas.

O presente capítulo aborda, também, as motivações para modificações e extensões da estrutura de um GF, com vistas à derivação de estruturas associadas, tais como Grafos de Fluxo Normalizados e Grafos Inversos. Métricas associadas à avaliação do poder representacional dessas estruturas são apresentadas e exemplificadas, ao longo de todo o capítulo. A introdução do conceito de caminhos e conexões em GFs, bem como de métricas a eles associadas, estabelecem o fundamento necessário para a proposta de um algoritmo de decisão embasado em estruturas de GFs.

Como evidenciado ao longo de todo o capítulo, os Grafos de Fluxo, propostos por Pawlak, podem ser interpretados como algoritmos de decisão (classificadores), uma vez que suas conexões representam regras de decisão. Essas regras de decisão não possuem significado probabilístico e sim, determinístico, uma vez que representam fielmente a distribuição e estrutura dos dados analisados.

Uma importante vantagem dos Grafos de Fluxo sobre algumas outras teorias e técnicas para representação do conhecimento (Regra de Bayes, por exemplo) é a de não necessitar de informações prévias fornecidas por terceiros (especialistas, etc.) para que os dados possam ser representados de uma maneira estruturada, o que viabiliza o uso de mecanismos associados a fluxos de distribuição de dados para classificá-los.

O uso dos Grafos de Fluxo com conjuntos de instâncias de dados que possuam muitos atributos e muitos valores distintos pode vir a ser um desafio, devido as operações a serem realizadas sobre o mesmo (definição de nós, arcos, caminhos, conexões, extração de regras de decisão, entre outras).

Capítulo 5
O Sistema EFLOWG (*Extended Flow Graphs*)

Durante as investigações realizadas sobre os grafos de fluxo, foi sendo desenvolvido, paralelamente, um sistema computacional que implementa muitas das técnicas e algoritmos que embasam o formalismo GF. O sistema, nomeado EFLOWG (*Extended Flow Graphs*), e suas funcionalidades, foram subsidiados pelos conteúdos dos capítulos anteriores, particularmente aquele do Capítulo 4. Este capítulo apresenta em detalhes as funcionalidades disponibilizadas pelo sistema EFLOWG.

Para facilitar as explicações de cada aspecto e/ou funcionalidade do sistema, alguns elementos foram numerados e eventualmente são referenciados por essas numerações. Também, para facilitar o entendimento geral do sistema, foi utilizado o conjunto de instâncias de dados reais Iris, disponível junto ao repositório UCI ([Dua & Karra Taniskidou 2017]), o qual é amplamente conhecido no ambiente de Aprendizado de Máquina. Porém, para viabilizar a exibição de algumas opções do sistema, foram adicionadas, ao conjunto de instâncias de dados Iris original, 4 instâncias de dados, sendo que 2 delas possuem valores ausentes e as outras 2 possuem valores que o sistema poderá considerar como possíveis *Outliers*.

A organização do capítulo é a seguinte: a Seção 5.1 apresenta informações básicas sobre o sistema EFLOWG; a Seção 5.2 apresenta o layout da tela inicial do sistema, com várias de suas opções e funcionalidades relacionadas ao pré-processamento dos dados; a Seção 5.3 apresenta a tela relacionada especificamente à grafos de fluxo e algoritmos de decisão, e por fim, a Seção 5.4, apresenta uma breve conclusão do capítulo.

5.1 Linguagem de Programação e Ambiente do EFLOWG

O sistema EFLOWG foi desenvolvido na linguagem de programação JAVA ([JAVA 2018]), versão 8, atualização 181. Utilizou-se, para seu desenvolvimento, o SWING ([SWING 2018]), que possui vários componentes inteligentes prontos para o desenvolvimento de sistemas computacionais. Para manipular e organizar tais componentes, assim como para poder desenvolver o sistema propriamente dito, foi utilizado o software NetBeans IDE 8.2 [NETBEANS 2018].

EFLOWG é um sistema computacional *desktop* desenvolvido unicamente para plataforma Windows e destinado para uso exclusivo em computadores e *notebooks*. Não possui recursos de rede ou de acesso à Internet, logo, é para uso (e possui processamento) local. Também, não realiza conexões com Bancos de Dados. Os conjuntos de instâncias de dados, para serem utilizados no EFLOWG, precisam estar gravados em arquivos de dados. Tais arquivos devem estar organizados de acordo com padrão ARFF ([ARFF 2018]), assim como, estar armazenados em local que possam ser acessados pelo sistema, com permissão de leitura.

O sistema EFLOWG foi desenvolvido e testado apenas no ambiente do sistema operacional Windows 10. É dividido em duas guias principais: *Pré-processamento dos Dados* e *Indução de Grafos de Fluxo e Extração de Algoritmos de Decisão*.

Devido ao tamanho de algumas figuras, essas foram inseridas nos anexos na Dissertação. Essa ação permite que as figuras sejam melhor visualizadas e compreendidas. A Figura 5.1, por exemplo, apresenta a tela inicial do EFLOWG, referente à guia *Pré-processamento dos Dados* (n. 1). Na figura, a numeração destacada é utilizada para facilitar, ao leitor, a identificação e localização de alguns objetos do sistema. A figura exibida em Figura 5.1, pode ser visualizada em tamanho maior, no Anexo II.

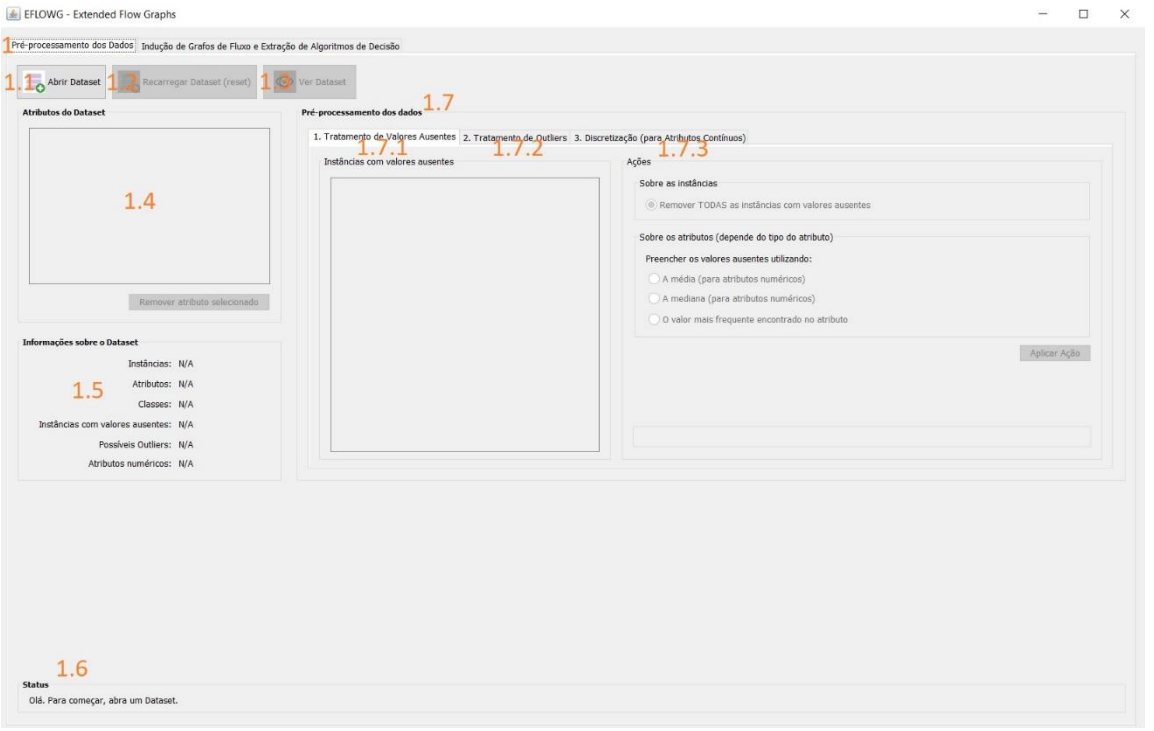


Figura 5.1 Tela inicial do sistema EFLOWG, referente à guia Pré-processamento dos Dados.

Cada uma das partes, de cada uma das abas que compõem o sistema EFLOWG é tratada nas subseções a seguir.

5.2 Guia Pré-processamento dos Dados

Como comentado anteriormente, a tela inicial do sistema EFLOWG é referente à guia "Pré-processamento dos Dados" (n. 1). Nesta tela, os dados são verificados, processados e transformados, para serem utilizados na geração de grafos de fluxo. Há três botões destacados na tela inicial, como é mostrado na Figura 5.2.

Figura 5.2 Os botões "Abrir Dataset", "Recarregar Dataset (reset)" e "Ver Dataset" do sistema EFLOWG.

O botão “Abrir Dataset” (n. 1.1), ao ser clicado, apresenta uma caixa de diálogo, que permite que o arquivo contendo os dados possa ser buscado e aberto (Figura 5.3).

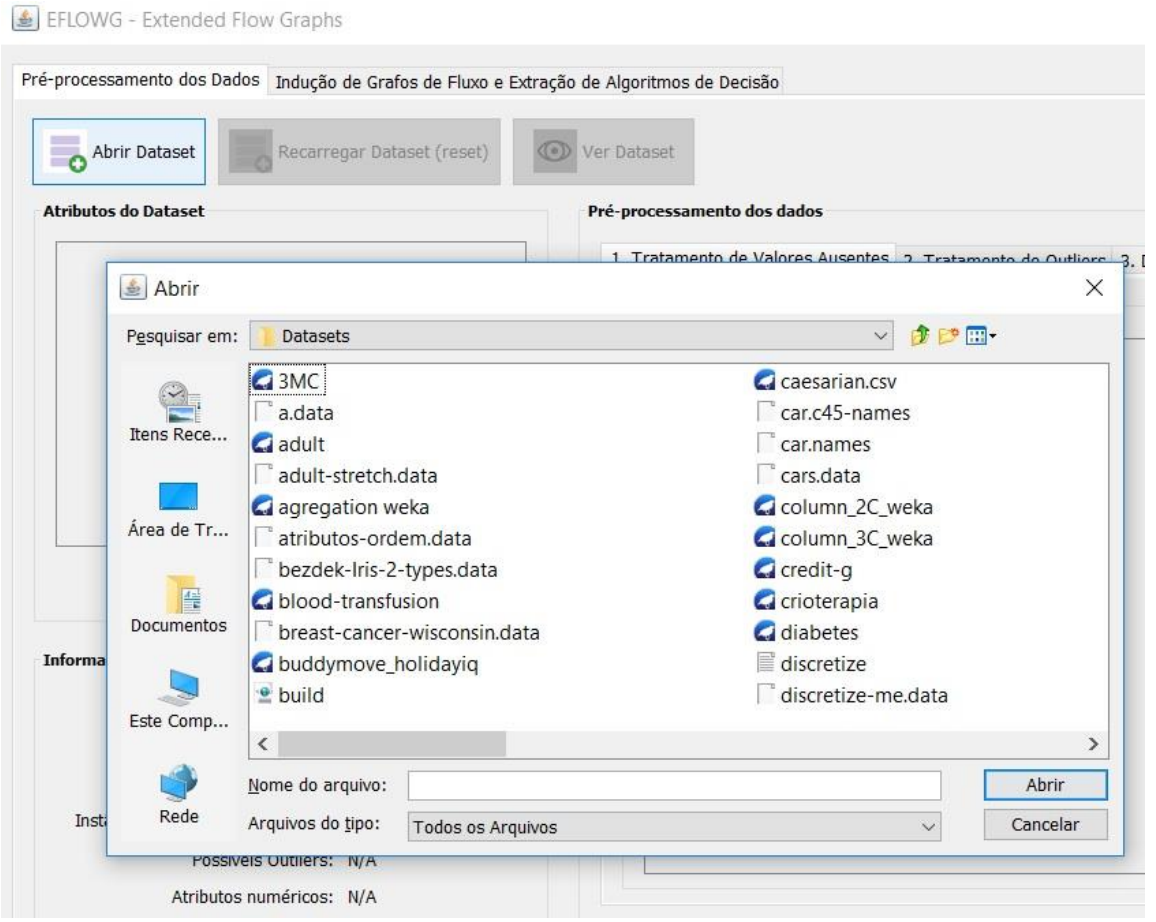


Figura 5.3 Ação executada pelo botão “Abrir Dataset” (n. 1.1) do sistema EFLOWG.

A Figura 5.4 apresenta as instâncias do conjunto de dados Iris no EFLOWG. A mesma figura, porém com tamanho melhor para visualização, é apresentada no Anexo III.

O conjunto de dados Iris original não possui valores ausentes ou Outliers. Porém, foram inseridos 4 registros neste conjunto de dados Iris para que as funcionalidades do EFLOWG, relacionadas ao tratamento desses problemas, pudessem ser apresentadas.

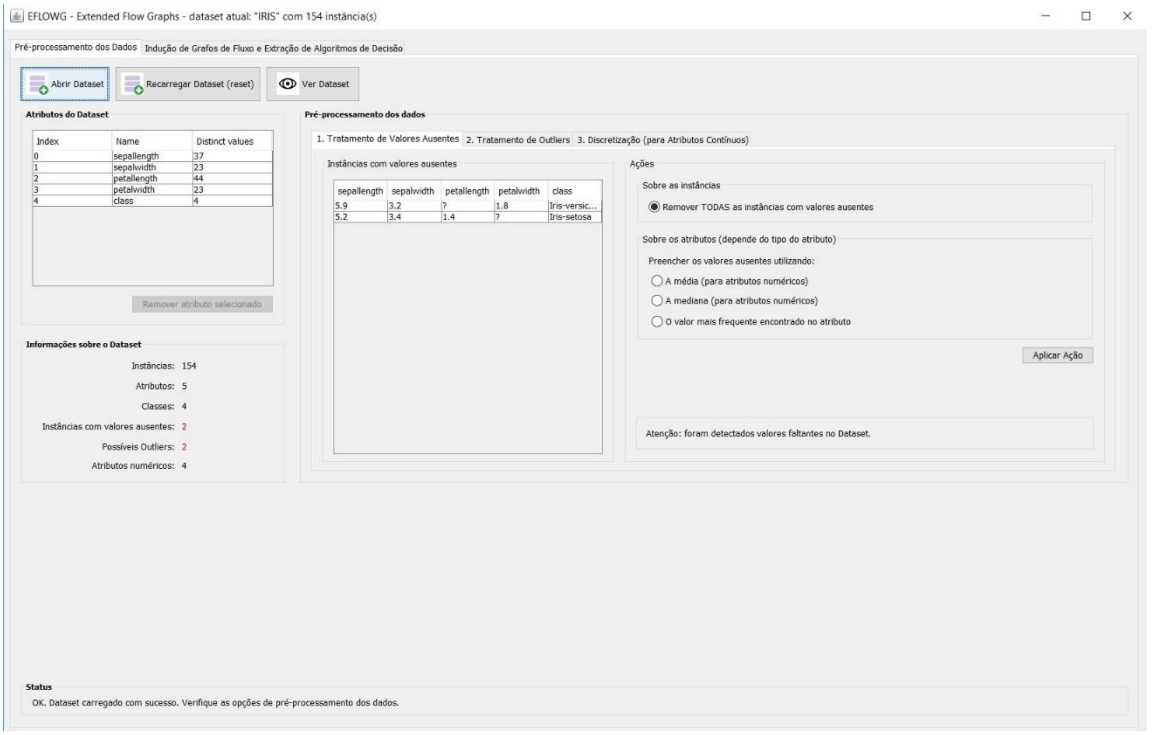


Figura 5.4 Arquivo contendo o conjunto de instâncias de dados Iris aberto no sistema EFLOWG.

O botão "Recarregar Dataset (reset)", (n. 1.2), ao ser acionado, recarrega o arquivo original que contém o conjunto de instâncias de dados (Figura 5.5). Transformações, discretizações, substituições de valores, remoções de instâncias e/ou atributos e ordenações de atributos são perdidas quando este botão é acionado.

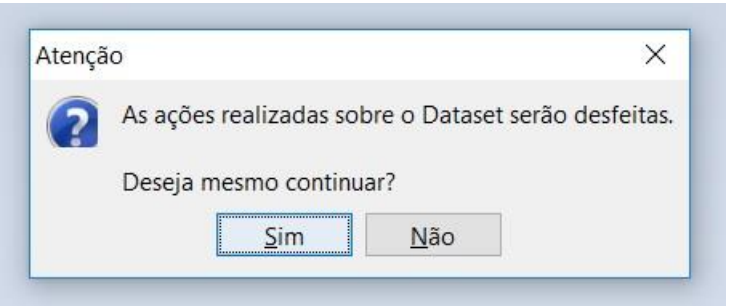
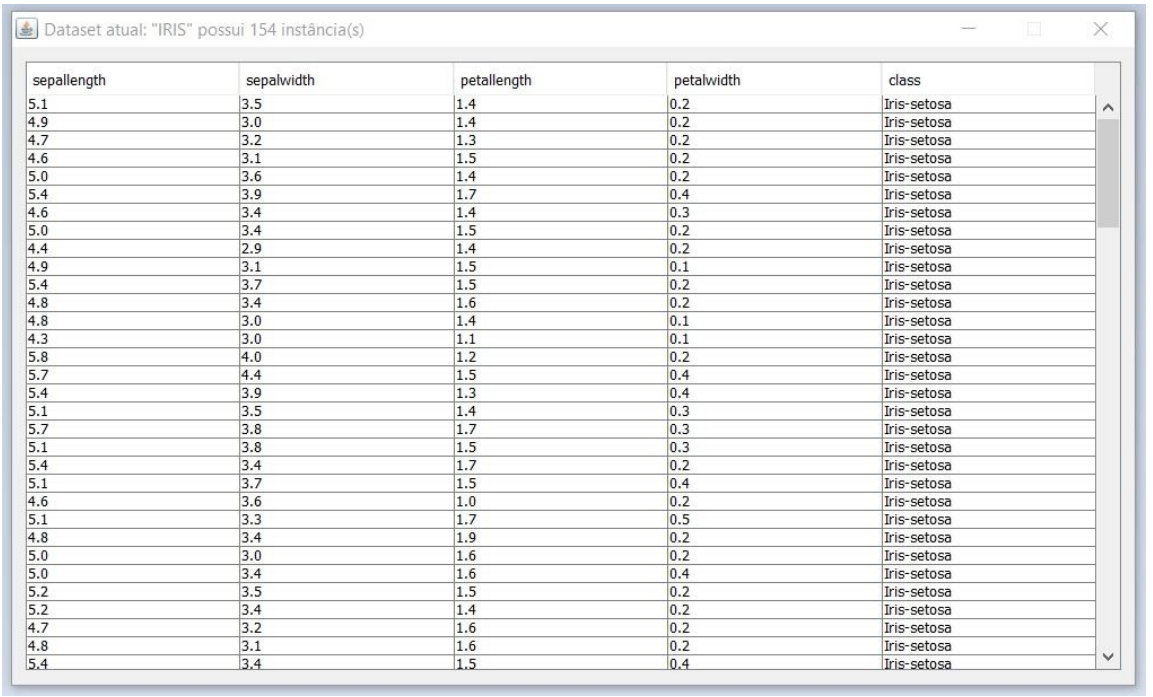


Figura 5.5 Ação executada pelo botão "Recarregar Dataset (reset)" no sistema EFLOWG.

Por sua vez, o botão "Ver Dataset" (n. 1.3) ao ser pressionado, exibe, em uma janela adicional, o conjunto de instâncias de dados em seu estado atual (Figura 5.6).

Caso o conjunto de instâncias de dados tenha sido discretizado, os atributos contínuos apresentarão, no lugar de seus valores originais, subintervalos de valores. A Figura 6.5 pode ser visualizada, com mais detalhes, no Anexo IV.



sepallength	sepalwidth	petallength	petalwidth	class
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
5.4	3.7	1.5	0.2	Iris-setosa
4.8	3.4	1.6	0.2	Iris-setosa
4.8	3.0	1.4	0.1	Iris-setosa
4.3	3.0	1.1	0.1	Iris-setosa
5.8	4.0	1.2	0.2	Iris-setosa
5.7	4.4	1.5	0.4	Iris-setosa
5.4	3.9	1.3	0.4	Iris-setosa
5.1	3.5	1.4	0.3	Iris-setosa
5.7	3.8	1.7	0.3	Iris-setosa
5.1	3.8	1.5	0.3	Iris-setosa
5.4	3.4	1.7	0.2	Iris-setosa
5.1	3.7	1.5	0.4	Iris-setosa
4.6	3.6	1.0	0.2	Iris-setosa
5.1	3.3	1.7	0.5	Iris-setosa
4.8	3.4	1.9	0.2	Iris-setosa
5.0	3.0	1.6	0.2	Iris-setosa
5.0	3.4	1.6	0.4	Iris-setosa
5.2	3.5	1.5	0.2	Iris-setosa
5.2	3.4	1.4	0.2	Iris-setosa
4.7	3.2	1.6	0.2	Iris-setosa
4.8	3.1	1.6	0.2	Iris-setosa
5.4	3.4	1.5	0.4	Iris-setosa

Figura 5.6 Conjunto de instâncias de dados Iris aberto no sistema EFLOWG.

Os atributos, que compõem o conjunto de instâncias de dados que está sendo utilizado, são listados na área “Atributos do Dataset” (n. 1.4), visualizada na Figura 5.7. O índice do atributo, o nome do atributo em questão e a quantidade de valores únicos por atributo são mostrados na tabela desta área. Logo abaixo da tabela, há, ainda, o botão “Remover atributo selecionado” que, quando ativado, remove do conjunto de instâncias de dados, o atributo selecionado na tabela.

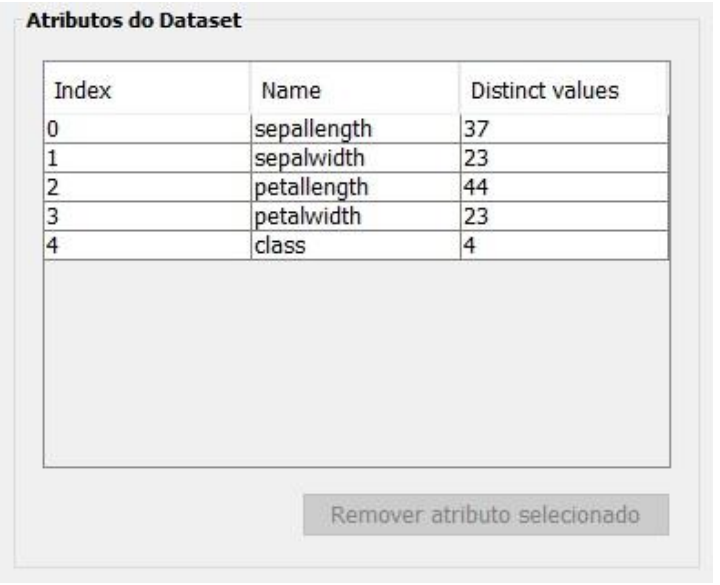


Figura 5.7 A tabela "Atributos do Dataset" e o botão "Remover atributo selecionado" do sistema EFLOWG.

A área "Informações sobre o Dataset" (n. 1.5), exibida na Figura 5.8, traz informações referentes ao conjunto de instâncias de dados sendo utilizado no momento. Essa área disponibiliza as seguintes informações: o número de instâncias, o número de atributos, o número de classes, o número de instâncias com valores ausentes, o número de possíveis *Outliers* e, também, o número de atributos numéricos. A cada atualização de tela, esses valores são recalculados, e assim, as informações desta área refletem, em tempo real, o estado em que se encontra o conjunto de instâncias de dados.

Figura 5.8 A área "Informações sobre o Dataset" do sistema EFLOWG.

O conjunto de instâncias de dados Iris original possui 150 instâncias, não possui valores ausentes e não possui *Outliers*. As informações mostradas na Figura 5.8 são baseadas no conjunto de instâncias de dados Iris utilizado como exemplo que, como comentado anteriormente, foi alterado, por meio da inserção de 2 instâncias de dados que possuem valores ausentes e de outras 2 instâncias de dados que possuem possíveis *Outliers.*

Na Figura 5.8, são apontadas pelo sistema um total de 154 instâncias de dados, sendo que delas, 2 possuem valores ausentes e 1 possui um possível *Outlier*. Como mencionado anteriormente, foram inseridas 2 instâncias de dados com possíveis *Outliers* no conjunto de instâncias de dados Iris do exemplo, porém somente 1 dessas instâncias foi identificada pelo sistema como possível *Outlier*. As razões para a não identificação da segunda instância de dados que possui um possível *Outlier* são apresentadas ainda nesta seção.

A área identificada pelo número 1.6 é a área "Status" (Figura 5.9) do sistema. Ela traz *feedbacks* sobre as operações realizadas e/ou o estado do conjunto de instâncias de dados e, às vezes, dependendo da última ação realizada, traz possíveis ações que podem ser executadas.

Figura 5.9 A área "Status" do sistema EFLOWG logo após a abertura do conjunto de instâncias de dados Iris.

Nas próximas subseções, são apresentadas as ações para o pré-processamento de dados propriamente dito, localizadas sob a guia "Pré-processamento dos Dados" (n. 1.7).

5.2.1 Tratamento de Valores Ausentes

A guia número 1.7.1, “Tratamento de Valores Ausentes”, mostrada na Figura 5.10 (e no Anexo V), traz informações e ações que podem ser executadas com vistas ao tratamento de instâncias de dados que possuem valores ausentes, identificados pelo símbolo “?”.

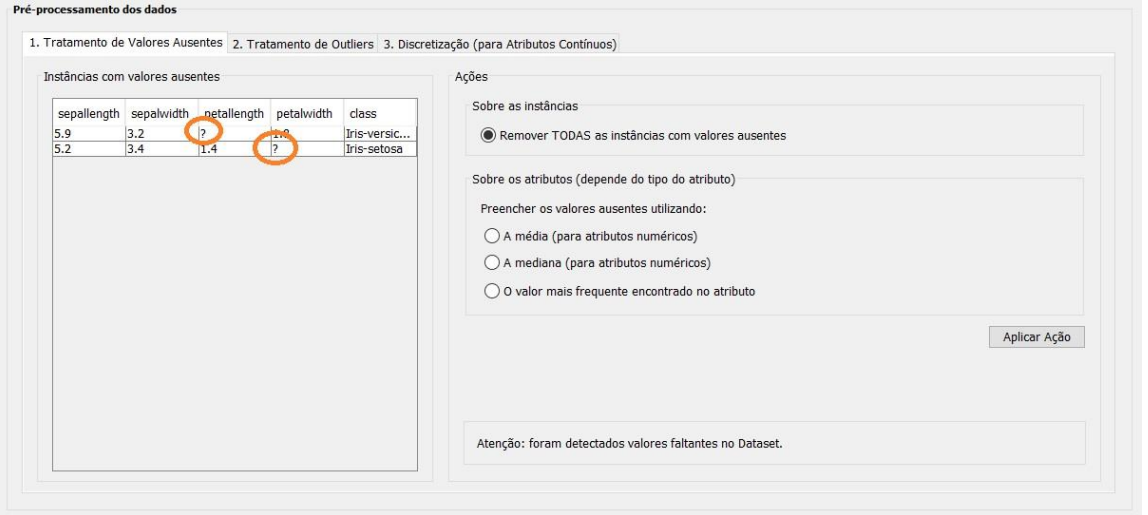


Figura 5.10 A guia “Tratamento de Valores Ausentes” do sistema EFLOWG.

O quadro na tela mostrada na Figura 5.10, caracterizado como “Instâncias com valores ausentes”, mostra as instâncias de dados, do conjunto de dados atual, que possuem valores ausentes. Quando não há instâncias com valores ausentes, nenhuma das ações, contidas na área “Ações” (mais à direita), estará habilitada, uma vez que não teriam utilidade. Por outro lado, se houver pelo menos uma instância de dados com valor(es) ausente(s), as ações relacionadas estarão habilitadas.

A opção “Remover TODAS as instâncias com valores ausentes”, como o próprio nome da ação informa, remove todas as instâncias com valores ausentes do conjunto de instâncias de dados. Já as opções apresentadas no grupo de opções relacionadas aos atributos, podem ser utilizadas dependendo do tipo do atributo selecionado na tabela que apresenta instâncias com valores ausentes. Por exemplo: se o atributo selecionado for um atributo numérico, será possível calcular a média e/ou a mediana de seus valores, assim como encontrar o seu valor mais frequente; porém, se o atributo for nominal, então apenas a opção relacionada ao valor mais frequente é habilitada. O botão “Aplicar

Ação" executa a ação selecionada pelo usuário e, por fim, a pequena área de texto na parte inferior direita da guia apresenta instruções e/ou resultados da operação executada em relação aos valores ausentes.

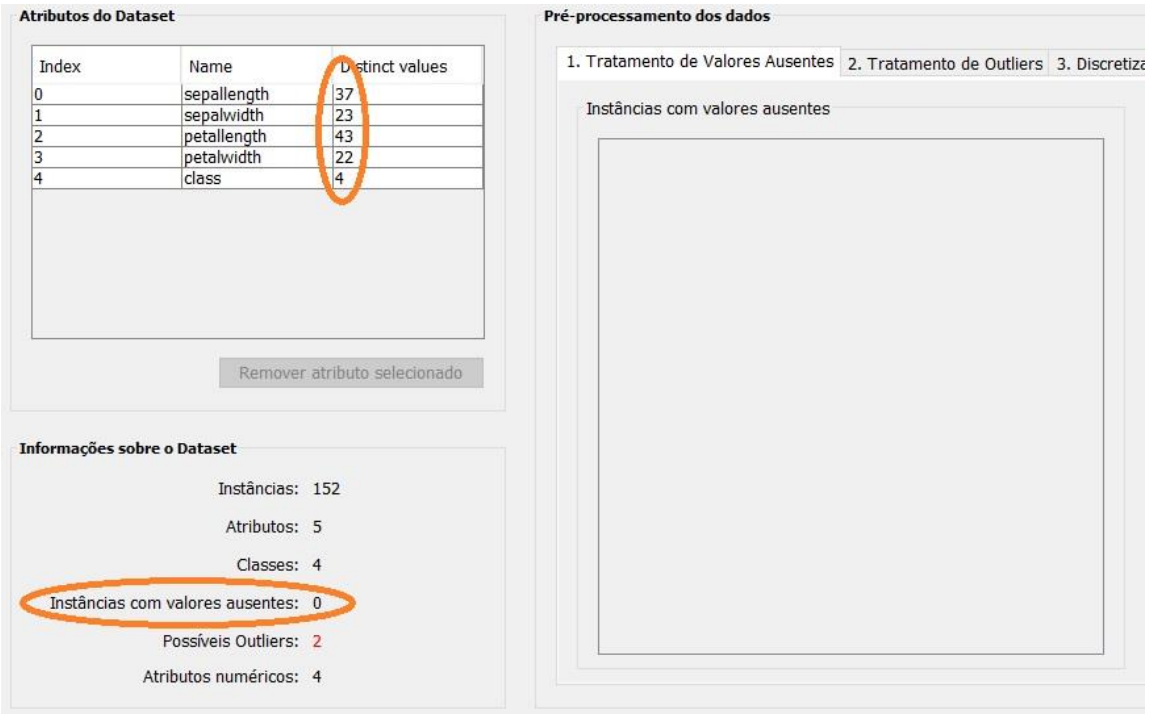


Figura 5.11 Após a execução da ação "Remover TODAS as instância com valores ausentes", são atualizados vários elementos da tela do sistema EFLOWG para que seja refletida a realidade atual do conjunto de instancias de dados.

É importante observar na Figura 5.11, que o número de instâncias caiu de 154 para 152 com a execução da ação que removeu instâncias com valores ausentes, fato que mostra que haviam 2 instâncias de dados, das 154, que possuíam valores ausentes.

Outro ponto importante é que o número de possíveis *Outliers*, encontrados no conjunto de instâncias de dados, aumentou de 1 para 2. Isso ocorreu, pois os valores ausentes são representados pelo símbolo de interrogação ("?") em conjuntos de instâncias de dados padrão ARFF e, devido à sua presença, o sistema não havia considerado um dos atributos que possuía valores ausentes como sendo um atributo numérico (contínuo). Após a remoção das instâncias de dados com valores ausentes, o sistema identificou atributos que eram, de fato, contínuos. Assim, o sistema analisou os

valores associados aos mesmos e detectou valores que podem representar possíveis *Outliers.*

A cada ação executada no sistema, as informações apresentadas nas telas do mesmo são totalmente atualizadas.

As ações sobre os atributos, em relação ao tratamento de valores ausentes, não eliminam instâncias de dados. Essas ações inserem determinados valores nos locais em que se encontram os valores ausentes, substituindo-os.

A Figura 5.12 mostra as instâncias que possuíam valores ausentes, antes e depois, da execução da ação "Preencher os valores ausentes utilizando: A media (para atributos numéricos)" e a Figura 5.13 mostra o resultado após a execução da ação "Preencher os valores ausentes utilizando: A mediana (para atributos numéricos)". Neste caso, para cada atributo que possuía valores ausentes, foram calculadas suas médias e medianas, e dependendo da ação, o valor da média ou mediana preencheu o valor ausente das instâncias de dados. Para ser possível executar as ações e analisar os resultados, o conjunto de instâncias de dados foi recarregado, utilizando o botão "Recarregar Dataset (reset)", antes da execução de cada uma das ações.

Figura 5.12 O conjunto de instancias de dados, antes e depois, da aplicação da ação "Preencher os valores ausentes utilizando: A media (para atributos numéricos").

Figura 5.13 O conjunto de instancias de dados, antes e depois, da aplicação da ação "Preencher os valores ausentes utilizando: A mediana (para atributos numéricos").

As figuras Figura 5.14 e Figura 5.15 mostram o processo que ocorre quando a ação “Preencher os valores ausentes utilizando: o valor mais frequente encontrado no atributo” é executada.

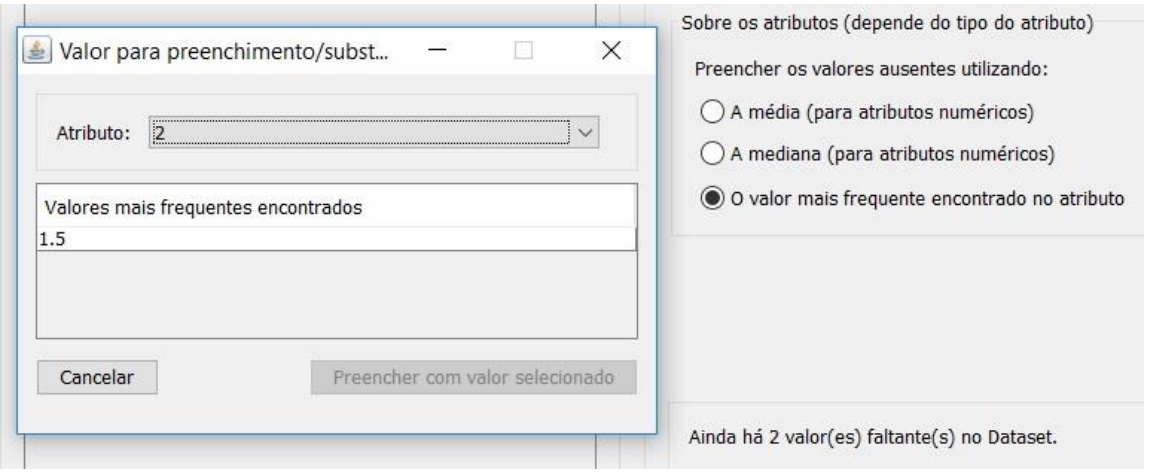


Figura 5.14 A opção “Preencher os valores ausentes utilizando: o valor mais frequente encontrado no atributo” exibe essa caixa de diálogo com os atributos que possuem valores ausentes e, também, os valores mais frequentes para cada atributo selecionado, de forma que o usuário possa decidir qual dos valores mais frequentes do atributo deve ser utilizado para preenchimento dos valores ausentes.

Figura 5.15 Resultado da utilização da opção “Preencher os valores ausentes utilizando: O valor mais frequente encontrado no atributo”.

5.2.2 Tratamento de Possíveis *Outliers*

As opções para o tratamento de possíveis *Outliers* são encontradas na guia “Tratamento de *Outliers*”, indicada na Figura 5.16 (e no Anexo X) pelo número 1.7.2, subguia de “Pré-processamento de Dados”.

Para esta subseção as instâncias de dados que possuíam valores ausentes foram removidas do conjunto de instâncias de dados utilizando a primeira das opções apresentadas na subseção anterior. Também, antes de cada execução dos diferentes

métodos de discretização, o conjunto de instâncias de dados foi recarregado utilizando o botão “Recarregar Data (reset)” (n. 1.2).

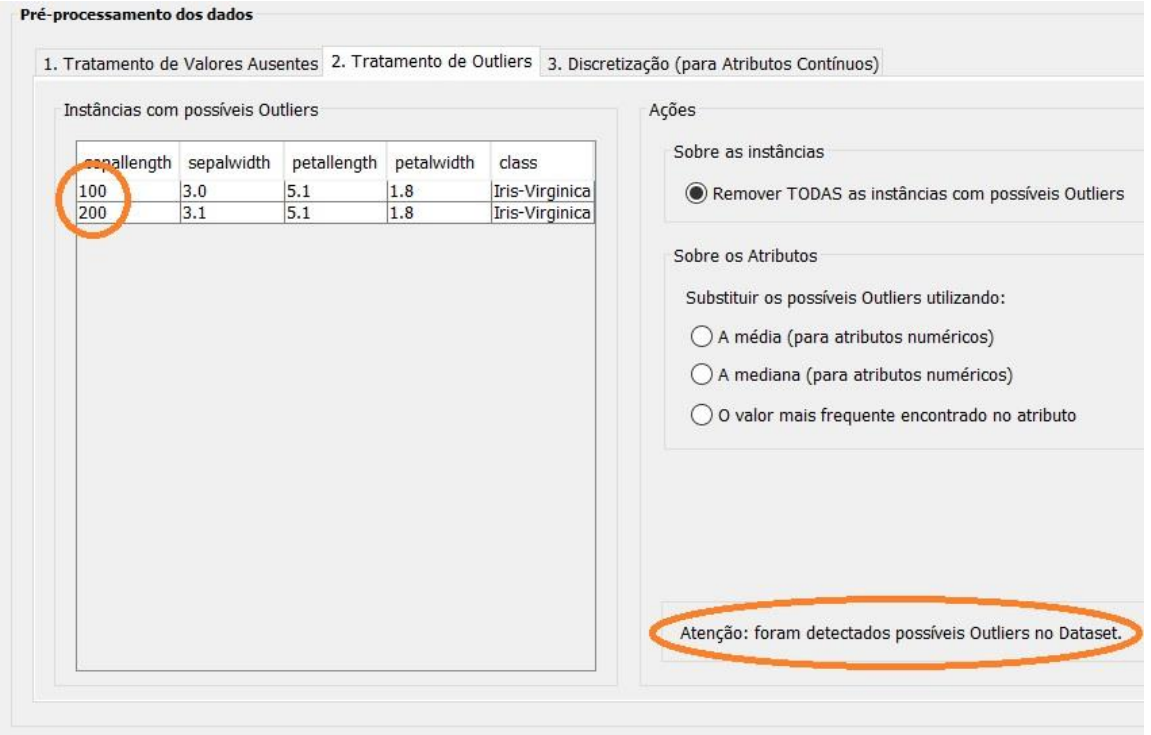


Figura 5.16 Ações disponíveis para o tratamento de possíveis *Outliers* (destacados na figura) identificados no conjunto de instâncias de dados aberto no sistema EFLOWG.

A área “Instâncias com possíveis *Outliers*” apresenta instâncias que podem possuir algum valor detectado pelo sistema como um possível *Outlier* (para mais informações, veja o Capítulo 3). Note que as ações para o tratamento de possíveis *Outliers* só estarão habilitadas caso tenham sido detectados possíveis *Outliers* no conjunto de instâncias de dados.

A ação “Remover TODAS as instâncias com possíveis *Outliers*”, remove, do conjunto de instâncias de dados, todas as instâncias de dados que possuam os possíveis *Outliers* identificados pelo sistema. As opções “A média (para atributos numéricos)”, “A mediana (para atributos numéricos)” e “O valor mais frequente encontrado no atributo” substituem o valor considerado um possível *Outlier* por um novo valor *i.e.*, a média, mediana ou o valor mais frequente associado àquele atributo, respectivamente. O botão “Aplicar Ação”, por sua vez, executa a ação selecionada acima pelo usuário e, por fim, a

área “Status”, referente ao tratamento de possíveis Outliers, apresenta informações sobre a operação executada.

Devido ao fato de que as ações disponíveis para o tratamento de possíveis *Outliers* são, basicamente, as mesmas disponíveis para o tratamento de valores ausentes, não serão apresentados exemplos dos resultados de tais ações sobre o conjunto de instâncias de dados, uma vez que exemplos similares já foram executados e apresentados na subseção anterior (“Tratamento de Valores Ausentes”). A única diferença é que, após as ações que visam tratar o problema de possíveis *Outliers* terem sido executadas, a área “Informações” seria atualizada indicando a ausência de possíveis *Outliers* (Figura 5.17), uma vez que esses problemas teriam sido resolvidos pelo usuário por meio das ações disponíveis no sistema EFLOWG.

Figura 5.17 A área “Informações” após a remoção de todas (*i.e.* 2) instâncias de dados, do conjunto de instâncias de dados, que possuíam possíveis *Outliers.*

No caso da ação selecionada pelo usuário ser "Remover TODAS as instâncias com possíveis *Outliers*", poderia ocorrer uma diminuição do número de instâncias do conjunto de dados.

As demais ações não causam possíveis alterações no número de instâncias do conjunto de instâncias de dados, pois todas elas substituem os possíveis *Outliers* por outros valores (*i.e.*, média, mediana ou valor mais frequente encontrado no atributo em questão).

5.2.3 Discretização de Atributos

A subguia 1.7.3, “Discretização (para Atributos Contínuos)”, apresentada na Figura 5.18, da guia “Pré-processamento de Dados”, fornece opções para execução do processo de discretização de atributos contínuos (para mais informações sobre discretização, veja o Capítulo 3, Seção 3.4).

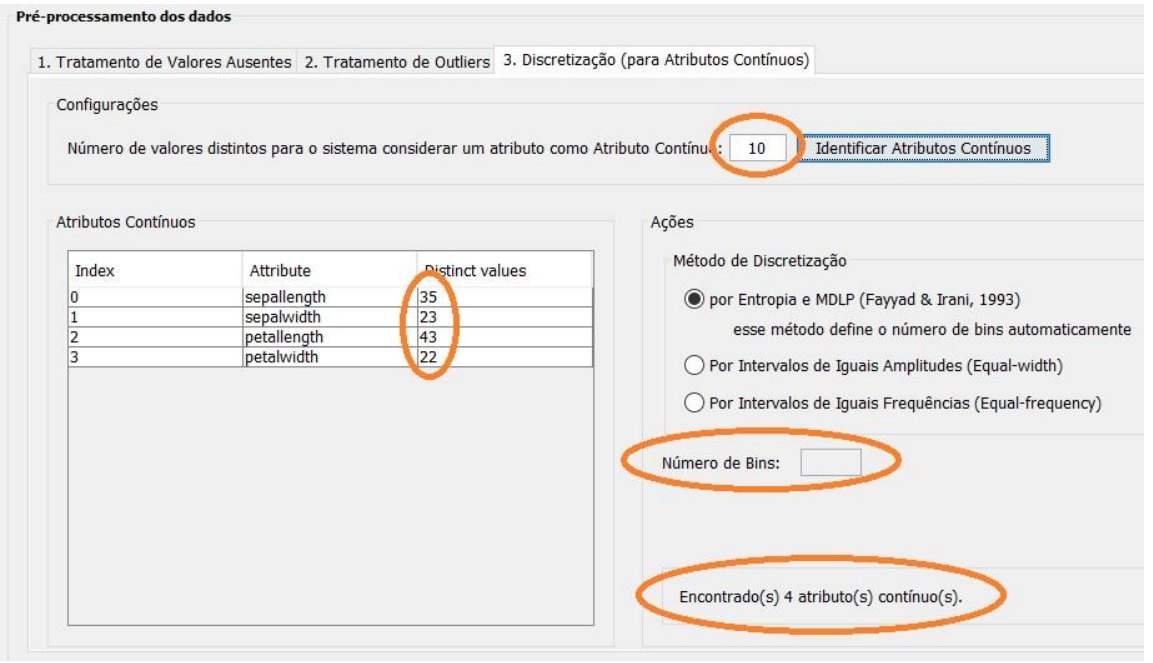


Figura 5.18 Resultado da identificação de atributos contínuos por parte do sistema, atributos estes que, de acordo com as “Configurações”, são aqueles que possuem 10 ou mais valores distintos associados.

Na área “Configurações” é possível informar o número de valores distintos, associados a um determinado atributo numérico do conjunto de instâncias de dados, que define tal atributo como sendo um atributo contínuo. O número de valores distintos *vd* é informado e, ao clicar no botão “Identificar Atributos Contínuos”, o sistema fará uma varredura no conjunto de instâncias de dados buscando atributos numéricos que possuam valores distintos em número igual ou superior a *vd*. Os atributos que satisfazem tal condição são definidos como atributos contínuos e são listados na tabela da área “Atributos Contínuos” (Figura 5.19), e assim, as ações de discretização podem ser aplicadas sobre eles.

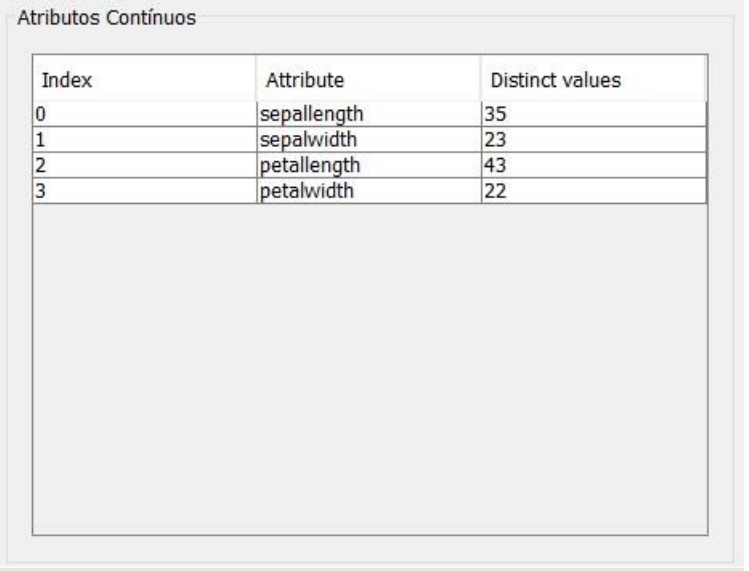


Figura 5.19 Atributos contínuos identificados pelo sistema EFLOWG. No exemplo, qualquer atributo numérico com 10 (valor informado pelo usuário na área "Configurações") ou mais valores distintos foi considerado atributo contínuo.

Entre as ações implementadas para tratamento de atributos contínuos, estão aquelas relacionadas aos 3 métodos de discretização (ou particionamento) apresentados no Capítulo 3 (Seção 3.4).

É importante deixar claro que, dentre as opções disponíveis, duas delas (Iguais Frequências e Iguais Amplitudes), requerem a informação sobre o número de *bins* (subintervalos). O terceiro método de discretização (Entropia e MDLP), por ser um método supervisionado, identifica, indiretamente, qual é o melhor número de *bins* para cada um dos atributos contínuos a ser discretizado.

A Figura 5.20 (ampliada no Anexo VI) apresenta o resultado obtido com a discretização dos atributos contínuos do conjunto de dados Iris (após a eliminação das instâncias com valores ausentes e com possíveis *Outliers)*, utilizado nos exemplos deste capítulo, por meio do método de particionamento por Entropia e MDLP (Fayyad & Irani, 1993), que não necessita da informação sobre o número de *bins* desejado, uma vez que infere o número a partir dos dados.

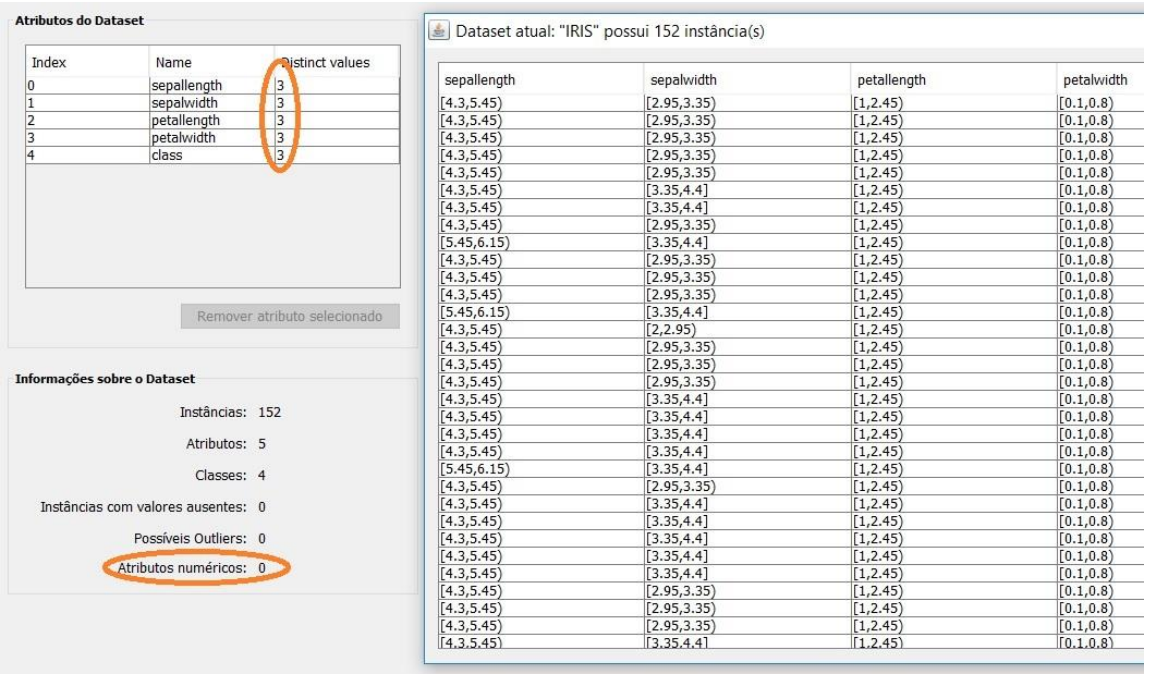


Figura 5.20 Resultado do processo de discretização por Entropia e MLDP aplicado sobre o conjunto de instâncias de dados Iris.

Os valores originais dos atributos contínuos foram substituídos por intervalos de valores, o que provocou uma alteração no número de valores distintos por atributo, como mostrado na área "Atributos do Dataset". Para cada um dos atributos foi obtida uma partição envolvendo 3 subintervalos de valores, como segue.

(1) atributo *sepallength*, partição $P_{sepallength}$={[4.3,5.45),[5.45,6.15),[6.15,7.9]};

(2) atributo *sepalwidth*, partição $P_{sepalwidth}$={[2,2.95),[2.95,3.35),[3.35,4.4]};

(3) atributo *petallength*, $P_{petallength}$={[1,2.45),[2.45,4.75),[4.75,6.9]}; e,

(4) atributo *petalwidth*, partição $P_{petalwidth}$={[0.1,0.8),[0.8,1.75),[1.75,2.5]}.

A Figura 5.21 (ampliada no Anexo VII) mostra os resultados obtidos após a discretização dos atributos contínuos com o uso da técnica de particionamento por intervalos de Amplitudes Iguais (*Equal Width*). Como o valor 3 foi definido como o número de *bins* desejados, para cada atributo contínuo identificado, uma partição definida por 3 subintervalos de valores, foi obtida.

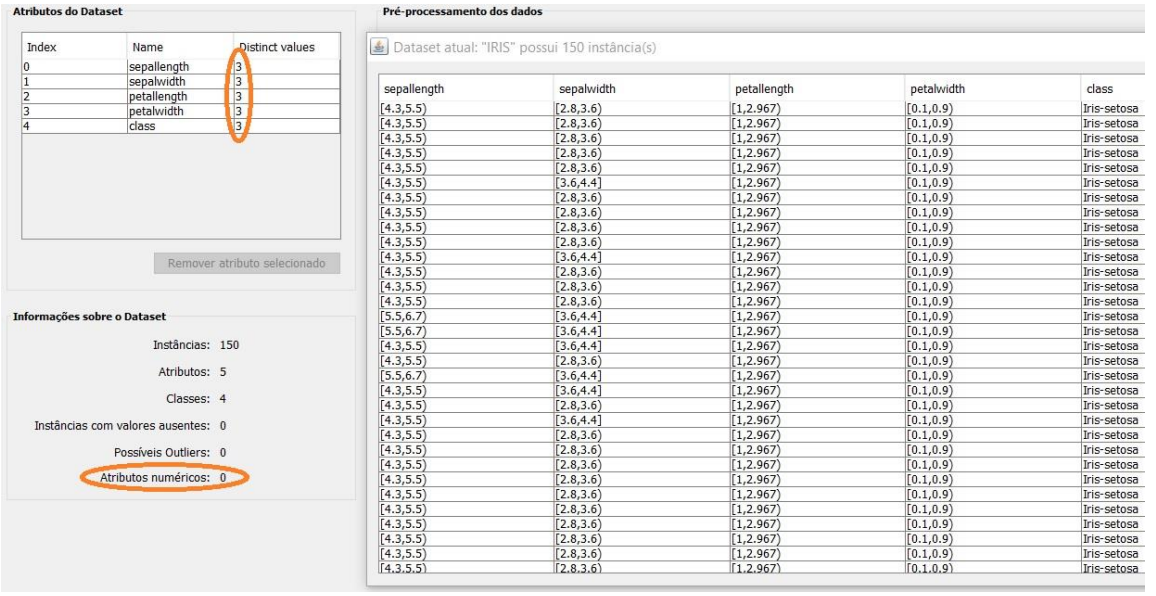


Figura 5.21 Resultado do processo de discretização por intervalos de Amplitudes Iguais (*Equal Width*) aplicado sobre o conjunto de instâncias de dados Iris.

Utilizando a técnica de particionamento por Amplitudes Iguais (*Equal Width*), para cada um dos quatro atributos foi definida uma partição composta por 3 subintervalos, como especificado pelo usuário, antes do uso do processo de discretização.

(1) atributo *sepallength*, partição $P_{sepallength}$={[4.3,5.5),[5.5,6.7),[6.7,7.9]};

(2) atributo *sepalwidth*, partição $P_{sepalwidth}$={[2,2.8),[2.8,3.6),[3.6,4.4]};

(3) atributo *petallength*, partição $P_{petallength}$={[1,2.967),[2.967,4.933),[4.933,6.9]}; e,

(4) atributo *petalwidth*, partição $P_{petalwidth}$={[0.1,0.9),[0.9,1.7),[1.7,2.5]}.

A terceira e última ação disponível para realizar de discretização de atributos contínuos é baseada no particionamento por intervalos de Iguais Frequências (*Equal Frequency*). Esse método requer a informação sobre o número de *bins* desejados. Novamente, o número 3 foi informado como número de *bins* desejado. A Figura 5.22 (ampliada no Anexo VIII) apresenta os resultados obtidos com a execução do método de discretização por Iguais Frequências.

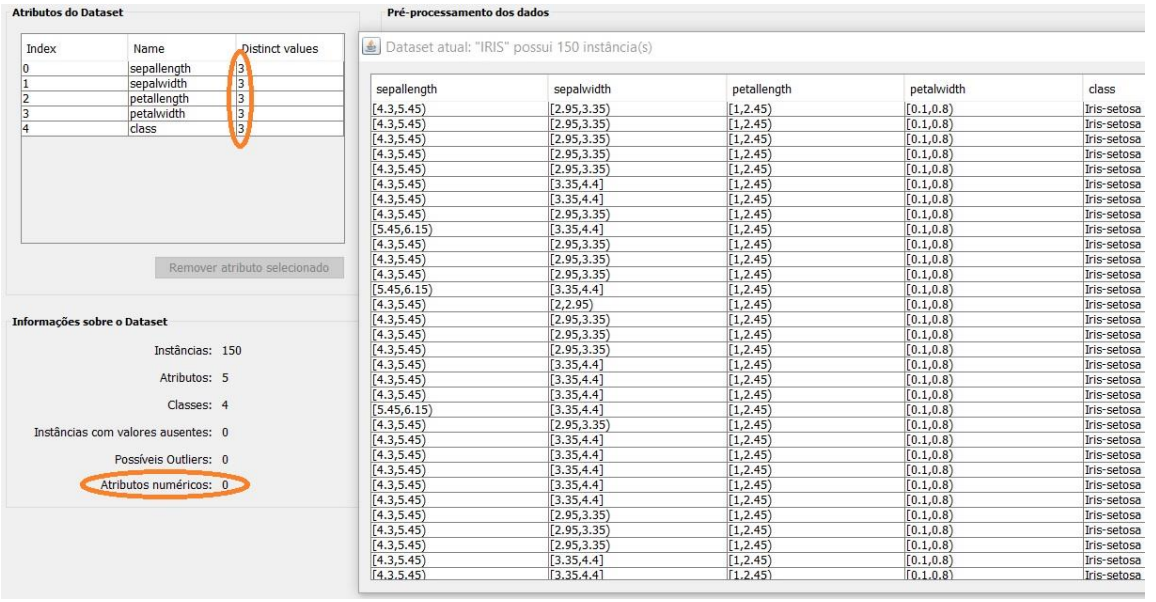


Figura 5.22 Resultado do processo de discretização por intervalos de Iguais Frequências (*Equal Frequency*) aplicado sobre o conjunto de instâncias de dados Iris.

A técnica de particionamento por Iguais Frequências (*Equal Frequency*) produziu, para cada um dos quatro atributos do conjunto Iris, as seguintes partições, cada uma delas formada por 3 subintervalos (bins), como requerido pelo usuário.

(1) atributo *sepallength*, $P_{sepallength}$={[4.3,5.45),[5.45,6.25),[6.25,7.9]};

(2) atributo *sepalwidth*, $P_{sepalwidth}$={[2,2.85),[2.85,3.15),[3.15,4.4]};

(3) atributo *petallength*, $P_{petallength}$={[1,2.45),[2.45,4.85),[4.85,6.9]}; e,

(4) atributo *petalwidth*, a $P_{petalwidth}$={[0.1,0.8),[0.8,1.65),[1.65,2.5]}.

5.3 Indução de Grafos de Fluxo e Extração de Algoritmos de Decisão

A guia “Indução de Grafos de Fluxo e Extração de Algoritmos de Decisão” (n. 2) disponibiliza ações que podem ser aplicadas sobre o conjunto de instâncias de dados; seu foco principal, entretanto, não é mais preparar os dados (o que é feito na guia apresentada na seção anterior), mas sim, disponibilizar formas de manipulação do conjunto de instâncias de dados com objetivo de subsidiar a indução de grafos de fluxo e, a partir deles, extrair algoritmos de decisão (classificadores).

A Figura 5.23 (ampliada no Anexo IX) apresenta a guia “Indução de Grafos de Fluxo e Extração de Algoritmos de Decisão” do sistema EFLOWG. As numerações de alguns de seus elementos, novamente, é para que possam ser referenciados, assim como localizados pelos usuários, mais facilmente.

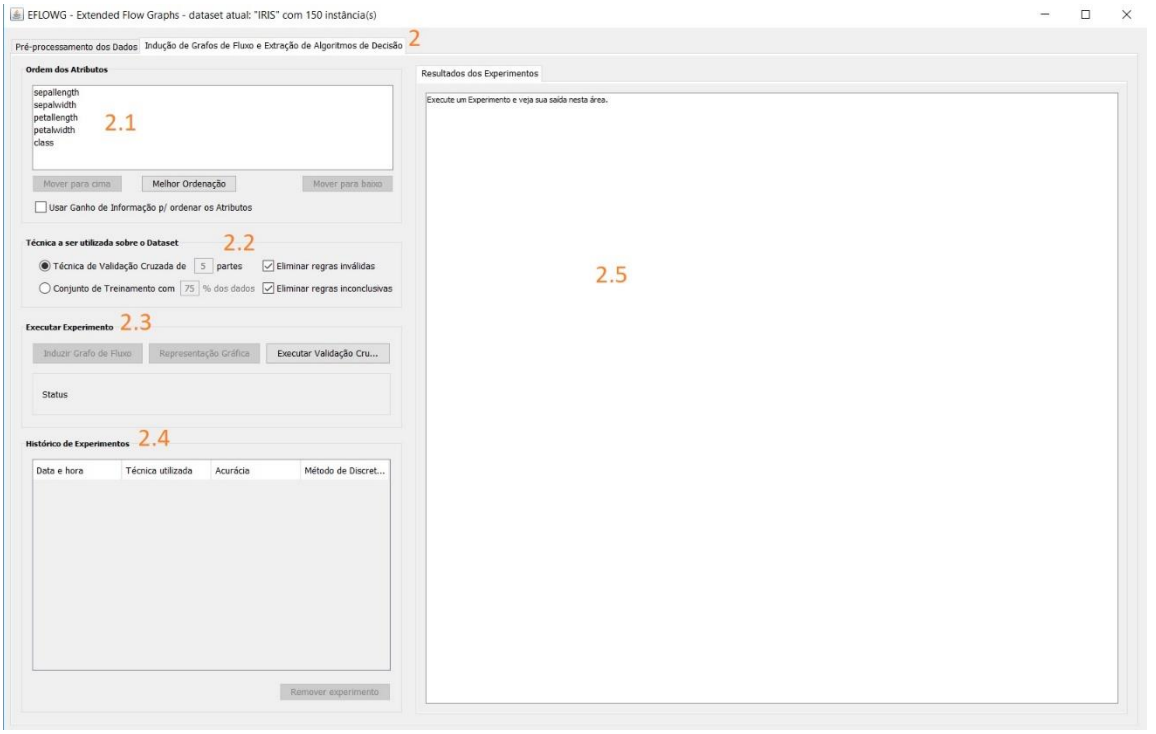


Figura 5.23 A guia “Indução de Grafos de Fluxo e Extração de Algoritmos de Decisão” do sistema EFLOWG.

As opções da área “Ordem dos Atributos” (n. 2.1) permitem que o usuário defina a ordem dos atributos no conjunto de instâncias de dados. As alterações realizadas nesta área refletem imediatamente no conjunto de instâncias de dados e podem influenciar os grafos de fluxo induzidos, uma vez que cada atributo de conjunto de instâncias de dados representa uma camada no grafo de fluxo. Essa opção fez-se necessária, pois, durante experimentos preliminares, foi notado que a ordem de comparecimento dos atributos do conjunto de instâncias de dados pode influenciar a indução de grafos de fluxo melhores (mais simples, objetivos e que melhor sumarizam os dados), e também, a extração de algoritmos de decisão mais precisos (*i.e.* com maior acurácia). Tal assunto é abordado com mais detalhes no Capítulo 6.

A área “Técnica a ser utilizada sobre o Dataset” (Figura 5.24) apresenta os dois métodos disponibilizados para a avaliação do classificador induzido a partir do GF obtido.

Figura 5.24 As técnica que podem ser utilizadas no EFLOWG para indução de grafos de fluxo.

A opção “Técnica de Validação Cruzada de k partes” executa o procedimento de k-validação cruzada, em que as instâncias são divididas em k partes, k grafos de fluxo são induzidos, k algoritmos de decisão são extraídos referentes aos k grafos de fluxo, k testes são realizados e a média geral é obtida e disponibilizada, como resultado, para análises e verificações. Para usar essa opção, o botão a ser pressionado é o botão “Executar Validação Cruzada”, da área “Técnica a ser utilizada sobre o Dataset”, como mostrado na Figura 5.24.

A opção “Conjunto de Treinamento com n% dos dados” dá flexibilidade ao usuário do sistema, na condução de um processo simples de avaliação *ad-hoc*, também envolvendo instâncias de treinamento e de teste, com os percentuais de participação dessas instâncias definidos pelo próprio usuário.

Para executar o algoritmo geral que gera o grafo de fluxo com base na divisão do conjunto de instâncias de dados em conjunto de treinamento e conjunto de teste, extrair o algoritmo de decisão e utilizá-lo como um classificador, pode ser usado o botão “Induzir Grafo de Fluxo”, da área “Técnica a ser utilizada sobre o Dataset”.

O botão “Representação Gráfica” fica habilitado quando a forma de execução dos experimentos selecionada é aquela que divide as instâncias de dados usando porcentagem, pois essa opção gera apenas um grafo de fluxo, o qual pode ser representado graficamente.

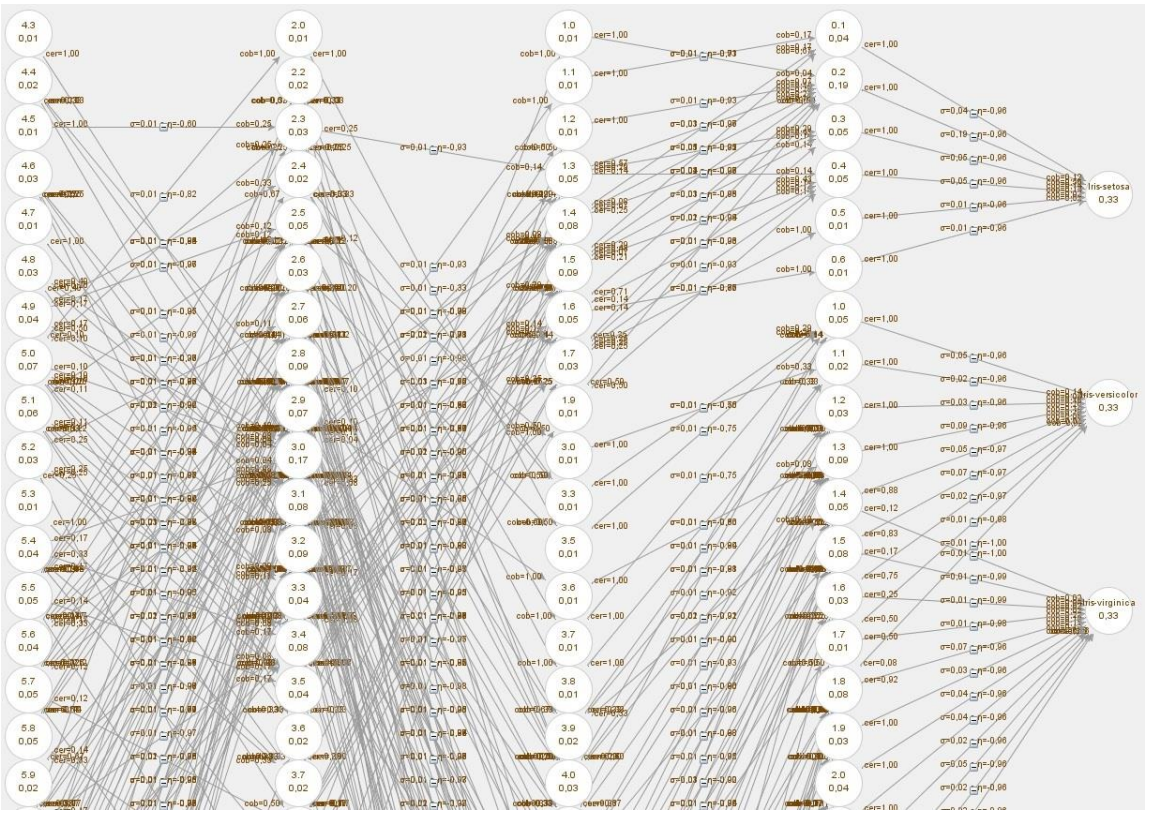


Figura 5.25 Parte da representação gráfica, gerada pelo EFLOWG, do GF induzido a partir do conjunto de instâncias de dados Iris original, que possui 150 instâncias e atributos contínuos.

A Figura 5.25 (ampliada no Anexo X) apresenta a representação gráfica do grafo de fluxo induzido a partir do conjunto de instâncias de dados Iris, em sua forma original, contendo 150 instâncias de dados e atributo contínuos. O objeto desta imagem não é o de apresentar com detalhes o GF, e sim, apresentar a complexidade estrutural (nós e arcos) encontrada no GF induzido a partir do conjunto de instâncias de dados Iris. A Figura 5.26, por sua vez, apresenta o GF induzido a partir do mesmo conjunto de dados (*i.e.*, Iris), porém após a realização do particionamento do escopo dos atributos contínuos por Entropia e MDLP.

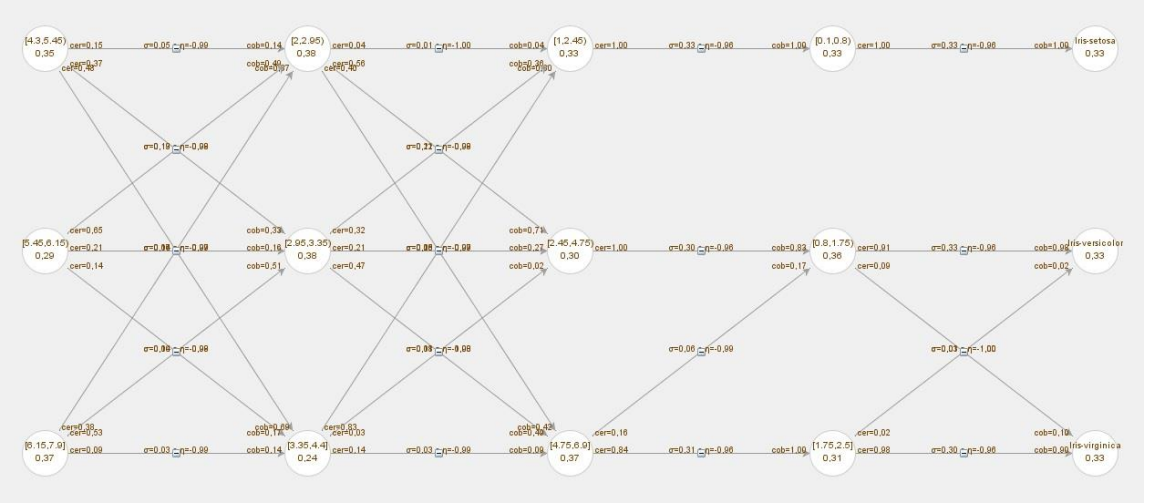


Figura 5.26 Representação gráfica, gerada pelo EFLOWG, do GF induzido a partir do conjunto de instâncias de dados Iris, devidamente discretizado.

A área “Status”, por sua vez, exibe informações sobre as operações realizadas na tela “Indução de Grafos de Fluxo e Extração de Algoritmos de Decisão”. A Figura 5.27 exibe os botões e a área “Status”.

Figura 5.27 Os botões “Criar GF”, “Representação Gráfica” e “Executar” e a área “Status” após a criação de um grafo de fluxo por meio do método de divisão das instâncias de dados por porcentagem, para criação dos conjuntos de treinamento e de teste.

A área “Histórico de Experimentos” (Figura 5.28) armazena, de forma temporária, os resultados dos experimentos realizados com o conjunto de instâncias de dados atual. Para isso, a cada indução de grafos de fluxo utilizando o método de divisão das instâncias de dados por porcentagem ou, a cada execução completa da validação cruzada de k partes, o sistema armazena uma cópia da saída para tais métodos e a disponibiliza para posterior consulta por parte do usuário.

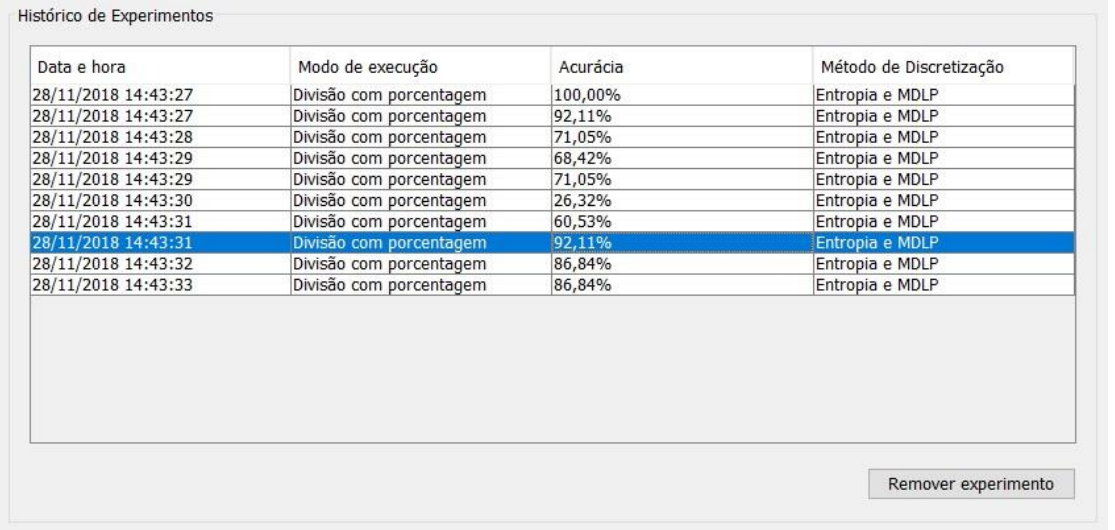


Figura 5.28 A área "Histórico de Experimentos" permite que o usuário visualize, para comparações e análises, as saídas dos experimentos realizados com o conjunto de instâncias de dados atual.

O objetivo da área "Histórico de Experimentos" (n. 2.4) é permitir a visualização das saídas dos experimentos realizados com o conjunto de instâncias de dados atual para comparações e análises, e por essa razão, a tabela que apresenta o histórico dos experimentos é simples, apresentando, de forma objetiva, as seguintes informações sobre os experimentos realizados: data e hora da execução, modo de execução (divisão com porcentagem ou validação cruzada de k partes), acurácia obtida pelo algoritmo de decisão extraído do grafo de fluxo induzido e o método de discretização que foi aplicado sobre o conjunto de instâncias de dados (quando houver). Ao selecionar um experimento realizado, as informações referentes ao experimento realizado são apresentadas na área "Resultados das Experimentos" (n. 2.5).

A área "Resultados das Experimentos" (n. 2.5) é responsável por apresentar os resultados e saídas dos experimentos realizados com grafos de fluxo e algoritmos de decisão no sistema EFLOWG. É preenchida quando os botões "Induzir Grafo de Fluxo" e "Executar Validação Cruzada" são pressionados ou, então, quando o usuário seleciona algum experimento na área "Histórico de Experimentos". A Figura 5.29 mostra a área "Resultados dos Experimentos".

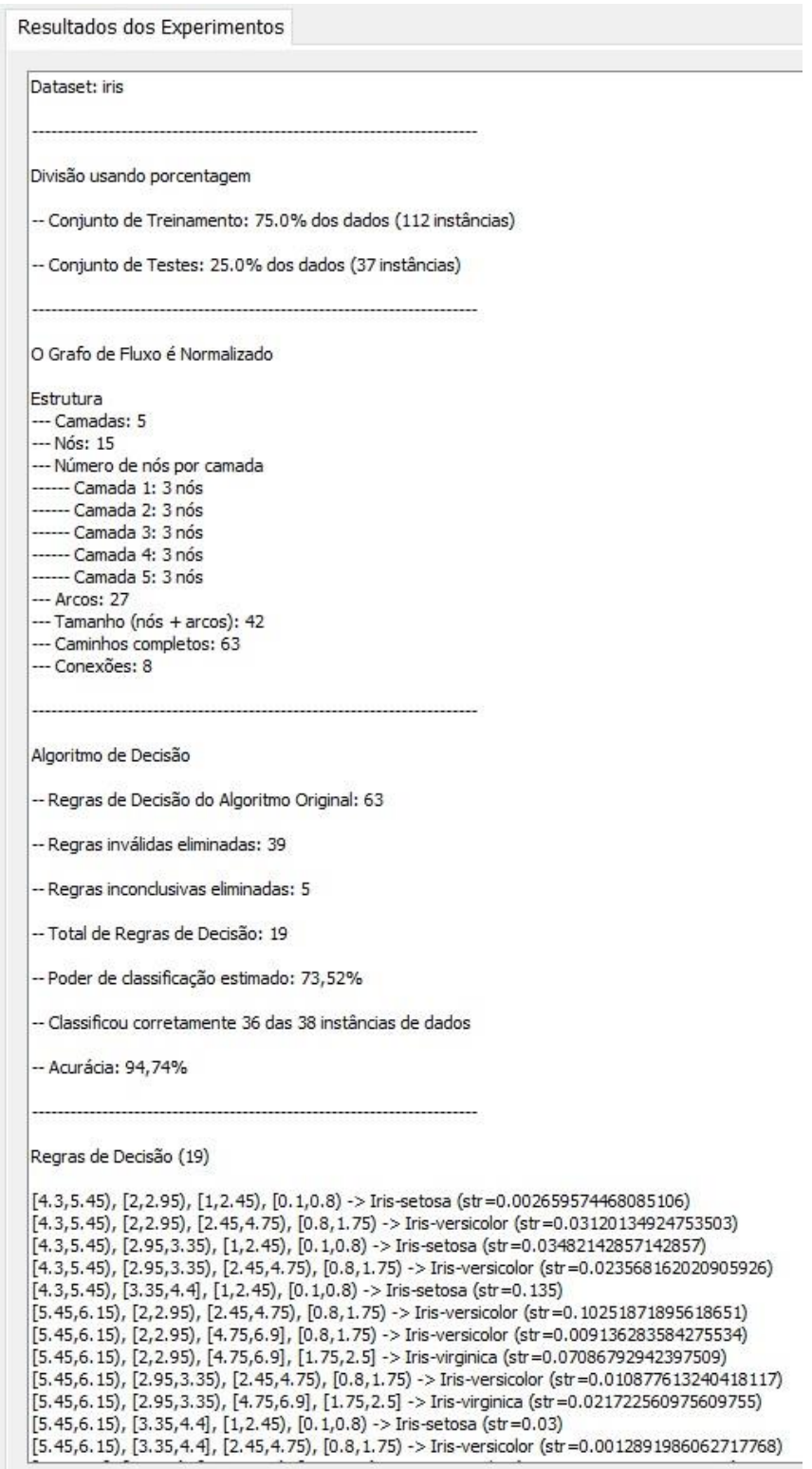


Figura 5.29 A área “Resultados dos de Experimentos” exibe a saída gerada pelos métodos executados no sistema EFLOWG.

5.4 Considerações Finais

Este capítulo pode ser abordado como um breve manual de operação do sistema computacional EFLOWG. Teve por objetivo apresentar uma descrição abrangente do sistema, envolvendo a descrição de suas diversas funcionalidades e, também, de apresentar alguns exemplos de seu uso. O sistema foi desenvolvido e implementado no decorrer da pesquisa realizada, com o objetivo de viabilizar a experimentação tanto do formalismo, como uma representação sumarizada de dados, quanto do uso de algoritmos subsidiados por tal formalismo, para a indução de classificadores. Foi desenvolvido seguindo padrões de usabilidade, como recomendam várias propostas desenvolvidas na área de pesquisa de Interação Humano-Computador ([Booth 1992], [Mayhew 1992], [Nielsen 2003]), com o objetivo de obter um sistema computacional simples, intuitivo, fácil de ser utilizado e não exigente, no que diz respeito à memorização de informações, por parte de seus usuários e, também, que colaborasse com o trabalho em andamento, viabilizando um ambiente que refletisse vários dos aspectos sendo investigados na pesquisa.

Grande parte das funcionalidades disponibilizadas pelo sistema foram apresentadas neste capítulo, algumas com utilização de exemplos. Os botões, interfaces e áreas que compõem o sistema foram apresentados e descritos, o que permite inferir que qualquer eventual dúvida sobre a utilização do sistema possa ser sanada com uma releitura de partes do texto.

Capítulo 6
Metodologia, Descrição dos Experimentos e Análise dos Resultados

O capítulo tem como principal objetivo apresentar os resultados de experimentos realizados utilizando o sistema computacional EFLOWG (ver Capítulo 5), que foi desenvolvido ao longo da pesquisa realizada e embasado pelos formalismos, definições e características das técnicas e algoritmos apresentados nos capítulos anteriores. O capítulo está organizado em seis seções, cada uma delas brevemente descrita nos parágrafos seguintes.

A Seção 6.1 apresenta a metodologia utilizada para a realização dos experimentos, que são apresentados em uma seção posterior. Dados e características, relacionados aos conjuntos de instâncias de dados utilizados nos experimentos, são exibidos nesta seção.

A Seção 6.2 apresenta o método original, proposto por Pawlak, para extração de classificadores a partir de GFs, o qual é testado e avaliado. Alguns problemas encontrados nesse método são discutidos e, por fim, um novo método, baseado no original, é proposto, com objetivo de se obter melhores classificadores a partir de GFs.

A Seção 6.3 analisa a proposta dos GFs originais e realiza alguns experimentos com objetivo de identificar a necessidade de tratamentos de atributos contínuos, para que conjuntos de instâncias de dados, que possuam atributos contínuos, possam ser usados juntamente com GFs.

A Seção 6.4 propõe uma extensão dos GFs originais, chamada de Grafos de Fluxo Estendido, que permite a utilização de GFs sobre conjuntos de instâncias de dados que possuem atributos contínuos. Nesta seção é decidido, também, qual método de discretização vai ser utilizado, por padrão, na extensão proposta.

A Seção 6.5 apresenta os experimentos realizados e resultados obtidos, utilizando os GFs Estendidos e o novo método para extração de classificadores a partir de GFs, de

acordo com a metodologia descrita na Seção 6.1. Também, uma comparação entre os resultados obtidos com o uso dos GFs é realizada com base em resultados de alguns classificadores bem estabelecidos no ambiente de AM.

A Seção 6.6 finaliza o capítulo apresentando um resumo do mesmo.

6.1 Metodologia

Descrita de maneira simplista, a metodologia empregada para a realização dos experimentos foi baseada nas seguintes etapas: (1) levantamento e seleção de conjuntos de dados; (2) indução de GFs a partir de cada um dos conjuntos de dados selecionados; (3) extração de classificadores a partir dos GFs obtidos e (4) avaliação dos classificadores com relação a ambas (estrutura e acurácia).

A metodologia adotada para a realização dos experimentos foi definida e implementada ao longo do desenvolvimento do sistema computacional EFLOWG e de seus vários módulos, com vistas a ampliar seu escopo de atuação, bem como de disponibilizar subsistemas com funcionalidades voltadas ao tratamento de dados, simulação de situações de testes e suporte para a visualização gráfica de resultados, quando possível.

Como visto no Capítulo 5, o sistema EFLOWG, além das múltiplas funcionalidades oferecidas, permite a escolha entre duas opções de uso de GFs em um contexto de representação de dados e de indução de conhecimento: (1) aquela para a indução de um único GF e, consequentemente, de um classificador, a partir de conjuntos de dados e (2) aquela para utilização da técnica de k-validação cruzada, que fornece resultados médios obtidos no processo de indução de GFs, extração de classificadores e testes de classificação.

Considere um conjunto de instâncias de dados X. O processo de k-validação cruzada para k = 5, k informado pelo usuário, executado sobre X, funciona da seguinte maneira:

(1) X é particionado em k = 5 subconjuntos, X_1, X_2, X_3, X_4 e X_5, cada um deles composto por, aproximadamente, o mesmo número de instâncias;

(2) Para cada subconjunto X_i, $1 \leq i \leq k$,

(2.1) X_i é definido como conjunto de teste e $X–X_i$ como conjunto de treinamento;

(2.2) O conjunto de treinamento definido em (2.1) é usado como entrada para o algoritmo de indução de um GF que, uma vez induzido, tem várias de suas características (*e.g.* número de nós, de arcos, de caminhos, etc.) contabilizadas e armazenadas;

(2.3) A partir do GF induzido no passo (2.2), é extraído um classificador, representado por um conjunto de regras de decisão do tipo *if-then*, que é então automaticamente analisado tendo várias de suas características contabilizadas e armazenadas (*e.g.*, número de regras de decisão, número de regras inconclusivas, número de regras inválidas);

(2.4) Um teste de desempenho de classificação das instâncias contidas no conjunto de teste definido em (2.1), utilizando o conjunto de regras obtido em (2.3), é realizado e a acurácia obtida é armazenada;

(3) As médias dos valores associados às características dos GFs induzidos, as médias dos valores das características associadas aos classificadores extraídos a partir dos GFs, bem como as médias de acurácias de tais classificadores, são calculadas e um relatório com as informações obtidas é gerado para o usuário.

O processo de k-validação cruzada foi adotado como procedimento padrão para realização dos experimentos realizados neste trabalho de pesquisa. Quando conveniente ao usuário, entretanto, um processo de avaliação simples, utilizando um único conjunto de teste, está disponibilizado para uso junto ao sistema EFLOWG.

6.1.1 Conjuntos de Instâncias de Dados Utilizados

Nos experimentos conduzidos, foram utilizados 19 conjuntos de dados, cujas principais características estão apresentadas na Tabela 6.1. Para o levantamento e seleção dos conjuntos de instâncias de dados foi utilizado como critério aquele de identificar um grupo de conjuntos de dados cujas características fossem diversificadas em relação ao número de instâncias, tipos de atributo, número de classes existentes e número de instâncias/classe.

Tabela 6.1 Conjuntos de instâncias de dados utilizados nos experimentos. Na tabela, CI: nome do conjunto de instâncias de dados; Ref: referência ao conjunto de instâncias de dados; #NI: número de instâncias de dados em CI; #NA: número de atributos que descrevem as instâncias de dados de CI; #NC: número de classes associadas às instâncias de dados de CI, e; G_id = #NI: distribuição do número de instâncias de dados por classe associada.

CI	Ref	#NI	#NA	#NC	G_id = #NI
Ruspini (Ru)	[Ruspini 1970]	75	2	4	1=20, 2=23, 3=17, 4=15
Mouse-Like (ML)	[GMUM 2018]	1,000	2	3	1=200, 2=200, 3=600
Spherical_6_2 (Sp)	[Bandyopadhway & Maulik 2002]	300	2	6	1=50, 2=50, 3=50, 4=50, 5=50, 6=50
3MC (MC)	[Su *et al.* 2005]	400	2	3	1=120, 2=170, 3=110
Long Square (LS)	[Handl & Knowles 2004]	900	2	6	1=147,2=155,3=150,4=148,5=150,6=150
Iris (Ir)	[Dua & Karra Taniskidou 2017]	150	4	3	1=50, 2=50, 3=50
Fossil (Fo)	[Chernoff 1972]	87	6	3	1=40, 2=34, 3=13
Aggregation (Ag)	[Gionis *et al*. 2007]	788	3	7	1=45, 2=170, 3=102, 4=273, 5=34, 6=130, 7=34
Flame (Fl)	[Fu & Medico 2007]	240	3	2	1=87, 2=153
Seeds (Se)	[Charytanowicz *et al.* 2010]	210	8	3	1=70, 2=70, 3=70
Ecoli (Ec)	[Horton & Nakai 1996]	336	8	8	1=143, 2=77, 3=52, 4=35, 5=20, 6=5, 7=2, 8=2
Blood Transfusion (BT)	[Yeh *et al.* 2008]	748	5	2	1=574, 2=174
Haberman's Survival (HS)	[Haberman 1976]	306	4	2	1=225, 2=81
Vertebral Column (VC)	[Rocha Neto & Barreto 2009]	310	6	3	1=60, 2=150, 3=100
User Knowledge Model. (UK)	[Kahraman *et al*. 2013]	403	5	4	1=50, 2=129, 3=122, 4=102
Caesarian Section (CS)	[Zain Amin & Amir Ali 2018]	80	6	2	1=34, 2=46
Glass (Gl)	[Evett & Spiehler 1987]	214	10	7	1=70, 2=76, 3=17, 4=0, 5=13, 6=9, 7=29
Wine (Wi)	[Aeberhard *et al.* 1992]	178	14	3	1=59, 2=71, 3=48
Cryotherapy (Cr)	[Khozeimeh *et al.* 2017]	90	7	2	1=42, 2=48

Os experimentos realizados e registrados nesse capítulo utilizaram-se da sequência original em que os atributos, que descrevem as instâncias de dados, comparecem nos conjuntos de instâncias de dados.

A seguir, alguns tópicos importantes sobre os GFs e sobre extração de classificadores a partir destes são analisados. Posteriormente, os experimentos, utilizando a metodologia aqui proposta, são realizados e avaliados.

6.2 Extração de Classificadores a Partir de GFs

Essa seção, inicialmente, apresenta algumas considerações sobre o método de extração de, classificadores a partir de GFs, proposto em [Pawlak 2003]. Além disso, são propostas duas melhorias para tal método, gerando assim, um novo método para

extração de classificadores a partir de GFs, de forma a obter classificadores mais robustos e com melhor poder de classificação.

6.2.1 Considerações Iniciais

Como visto no Capítulo 4, todo caminho completo de um GF pode ser interpretado como uma regra de decisão ([Pawlak 2003]). A partir desta afirmação, foram realizados alguns experimentos para investigar o método de extração de regras de decisão a partir de um GF. O intuito foi o de analisar as regras de decisão extraídas dos GFs e de, particularmente, investigar o poder de classificação de tais regras, entre outros aspectos.

Um desses experimentos, utilizou um conjunto de instâncias de dados sintético, criado exclusivamente para este experimento, com 10 instâncias de dados, cada uma delas descrita por 2 atributos e uma classe associada. Tal conjunto de instâncias de dados é apresentado na Tabela 6.2.

Tabela 6.2 Conjunto de instâncias de dados utilizado no experimento para verificação da estrutura dos classificadores extraídos de GFs com base no método proposto por Pawlak, em que cada caminho completo de um GF pode ser interpretado como uma regra de decisão.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
1	1	2	1	2	1	1	2	1	2
B	B	B	B	B	B	B	B	B	B
X	X	Y	X	Y	X	X	Y	X	Y

O GF exibido na Figura 6.1, foi induzido utilizando um conjunto de treinamento com 7 (= 70%) das 10 instâncias do conjunto de instâncias de dados apresentado na Tabela 6.2. As outras 3 instâncias representaram o conjunto de teste. A representação gráfica exibida na Figura 6.1 foi gerada pelo sistema EFLOWG.

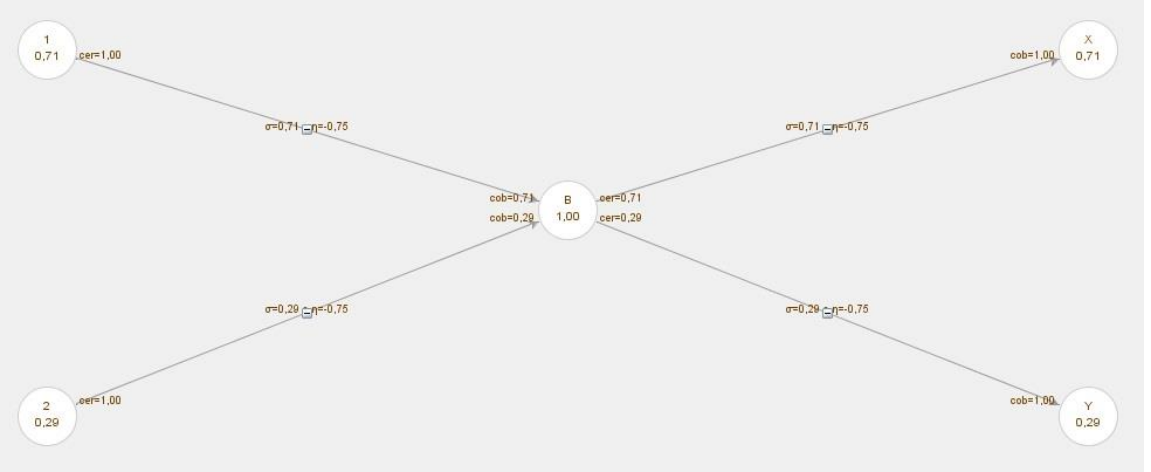


Figura 6.1 GF induzido pelo sistema EFLOWG, a partir de, aproximadamente, 70% das instâncias de dados que compõem o conjunto de instâncias de dados apresentado na Tabela 6.2.

6.2.2 O Método Original para Extração de Algoritmos de Decisão

Seguindo o método proposto por Pawlak (Definição 4.32, Capítulo 4), em que todo caminho completo de um GF pode ser considerado uma regra de decisão, é possível extrair do GF da Figura 6.1, as seguintes regras de decisão:

(1) 1, B $\rightarrow$ X (lê-se: se primeiro atributo tiver o valor 1 associado e o segundo atributo tiver o valor B associado, então X é a decisão a ser tomada)

(2) 1, B $\rightarrow$ Y (lê-se: se primeiro atributo tiver o valor associado 1 e o segundo atributo tiver o valor associado B, então Y é a decisão a ser tomada)

(3) 2, B $\rightarrow$ X (lê-se: se primeiro atributo tiver o valor associado 2 e o segundo atributo tiver o valor associado B, então X é a decisão a ser tomada)

(4) 2, B $\rightarrow$ Y (lê-se: se primeiro atributo tiver o valor associado 2 e o segundo atributo tiver o valor associado B, então Y é a decisão a ser tomada)

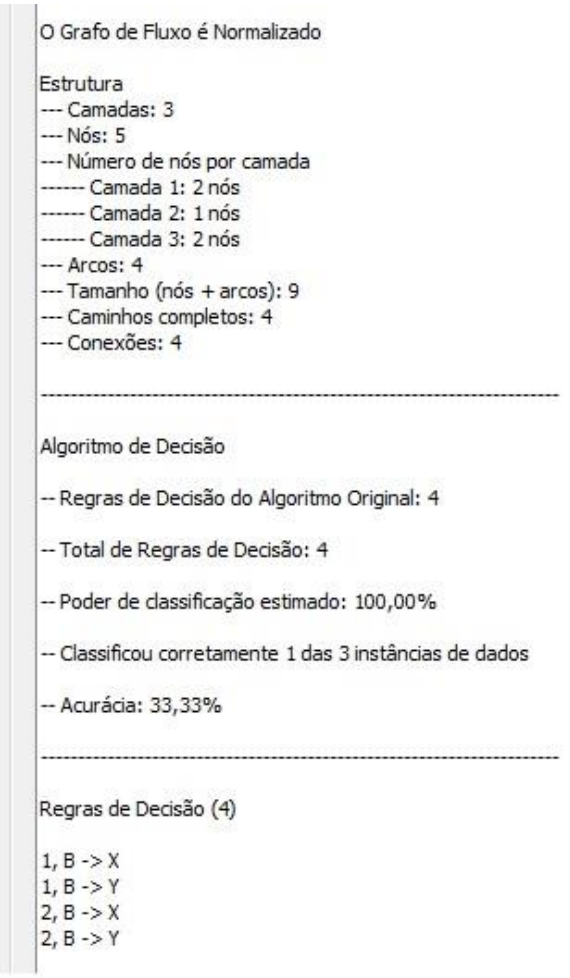


Figura 6.2 Saída do sistema EFLOWG que mostra a acurácia de 33.33% obtida pelo classificador extraído, utilizando o método original de extração de classificadores, do GF apresentado na Figura 6.1.

Porém, ao executar o teste de classificação, o classificador, composto pelas regras de decisão apresentadas acima, apresentou acurácia de 33,33% (Figura 6.2), ou seja, classificou corretamente 1 das 3 instâncias de dados do conjunto de teste. Dada a simplicidade do conjunto de instâncias de dados original, do GF e até mesmo do classificador, a acurácia foi insatisfatória e, por isso, o classificador foi investigado, de forma mais detalhada, a fim de se esclarecer o motivo pelo qual a acurácia foi tão baixa.

6.2.3 Regras Inconclusivas

Analisando o classificador obtido na Subseção 6.2.2, nota-se que há regras de decisão que possuem as mesmas condições, porém levam à diferentes decisões. São elas, as regras (1) e (2), e as regras (3) e (4).

As regras (1) e (2) são utilizadas quando a instância de dados, a ser classificada, possui no primeiro atributo o valor associado 1 e no segundo atributo, o valor associado B. O problema ocorre quando da tomada de decisão, pois há duas regras de decisão com as mesmas condições que levam à decisões (classes) diferentes. O mesmo acontece com relação às regras (3) e (4). Esse foi o primeiro problema encontrado com classificadores extraídos dos GFs da forma proposta pelo autor do formalismo.

As regras de decisão que descrevem os classificadores, que são formadas pelo mesmo antecedente, porém possuem diferentes consequentes, são identificadas nesta dissertação e, também, no sistema EFLOWG, como *regras inconclusivas*.

De acordo com o formalismo dos GFs, apresentado no Capítulo 4, alguns fatores podem ser definidos nos GFs de modo a auxiliar no processo de entendimento da distribuição do fluxo dos dados, e também, de forma a fornecer informações sobre os arcos, caminhos e conexões, que compõem os GFs.

Os fatores de força, certeza, cobertura e dependência entre nós, das regras de decisão, podem ser utilizados para escolher as melhores regras de decisão (*i.e.* àquelas que mais classificam instâncias corretamente), entre várias regras de decisão extraídas de um GF. Entre esses fatores, o fator de força indicaria, justamente, o poder de classificação de uma determinada regra de decisão ([Pawlak 2005b]).

Dessa forma, o método de extração de regras de decisão, implementado no sistema EFLOWG, segue a seguinte regra: caso sejam encontradas regras inconclusivas, tais regras são comparadas entre si e aquela com maior valor associado ao fator de força é mantida; as restantes, são descartadas.

O experimento realizado, logo após a implementação de tal regra, extraiu, do grafo de fluxo em questão, o classificador descrito pelas duas regras que seguem.

(1) 1, B $\rightarrow$ X

(2) 2, B $\rightarrow$ X

É fácil observar que o novo classificador possui um número menor de regras de decisão, fato que pode vir a ser uma vantagem considerando que o custo computacional, exigido para execução dos testes de classificação, foi reduzido pela metade em relação ao custo computacional exigido pelo classificador anterior. O mais importante, porém, é entender o motivo pelo qual essas regras de decisão foram mantidas definindo, assim, um novo classificador. Para isso, basta observar, na Figura 6.1, que o fator de força da regra de decisão 1, B $\rightarrow$ X, str=0.71, é maior que o fator de força da regra de decisão 1, B $\rightarrow$ Y, str=0.29 *i.e.*, 0.71 > 0.29. Da mesma forma, o fator de força da regra 2, B $\rightarrow$ X, str=0.71, é maior que o fator de força da regra 2, B $\rightarrow$ Y, 0.29, *i.e.* 0.71 > 0.29. Assim, foram mantidas somente as regras de decisão que possuíam o maior valor associado ao fator de força (para mais detalhes sobre os fatores relacionados aos elementos dos GFs, ver o Capítulo 4).

Ao executar o experimento, após a atualização que utiliza o fator de força para resolução do problema com regras de decisão inconclusivas, foi obtida acurácia, novamente, de 33,33%, como pode ser visto na Figura 6.3.

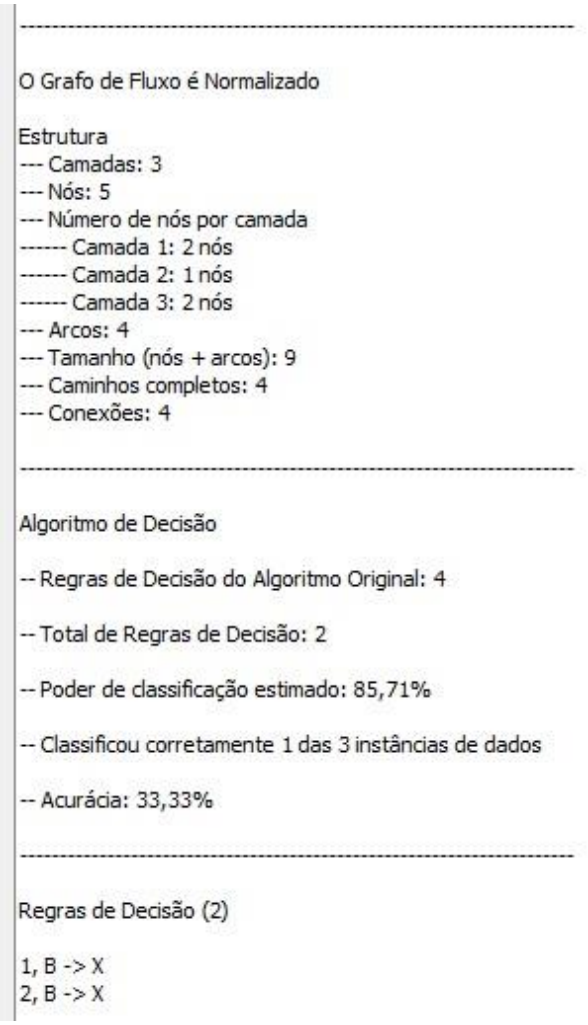


Figura 6.3 Saída do sistema EFLOWG que mostra, novamente, a acurácia de 33.33% obtida pelo classificador após a implementação da atualização que elimina regras inconclusivas.

Ao analisar as instâncias que compunham o conjunto de teste utilizado para o experimento, notou-se que, das 3 instâncias de dados, 2 delas deveriam ter sido classificadas como pertencentes à classe Y, porém não foram.

Ao analisar o algoritmo de decisão, então livre de regras inconclusivas, nota-se que o mesmo não possui, sequer, uma regra de decisão que classifique instâncias de dados como pertencentes à classe Y. Ao eliminar as regras de decisão inconclusivas com base, somente, no fator de força dessas regras, o qual deveria, de acordo com Pawlak ([Pawlak 2003]), apontar regras de decisão que pudessem ter melhor acurácia nas classificações, aconteceu que, algumas regras de decisão que tinham poder de

classificação real de instâncias de dados foram eliminadas do classificador, prejudicando, assim, o classificador e o teste de classificação como um todo. A atualização implementada no sistema EFLOWG, que eliminava regras inconclusivas, foi, então, desfeita, para que fossem investigadas as causas e soluções do novo problema encontrado, ou seja, neste ponto, o EFLOWG voltou a extrair classificadores a partir de GFs da forma original.

6.2.4 Regras Inválidas

Por meio de análise e observações, bem como de experimentos utilizando a técnica original para extração de classificadores, foi detectado um novo problema: a técnica original permite a extração de regras de decisão que não classificam instância de dados alguma do próprio conjunto de treinamento, o que faz com que os classificadores possuam muitas regras de decisão, sendo que, algumas (ou muitas) delas, sejam inúteis.

Tal problema ocorre, pois nem todo caminho completo de um GF possui nós cujos valores associados estão relacionados à uma instância de dados. Logo, pode haver caminhos completos gerados com base, simplesmente, na distribuição de fluxo dos dados entre os nós de diferentes camadas do GF, os quais, ao serem interpretados como regras de decisão, não classificam instância alguma.

Como pode ser observado no GF da Figura 6.1, os nós 1 e 2 da camada de entrada estão conectados ao nó B da segunda camada do GF. É possível ver, também, que o nó B está conectado aos nós X e Y da camada de saída do GF. Não há, no conjunto de instância de dados (Tabela 6.2), uma instância de dados que possa ser descrita pelos valores 1, B, Y e também não há instância que possa descrita pelos valores 2, B, X. Porém, no GF, há uma relação entre 1, B, Y, assim como, há uma relação entre 2, B, X, pelo fato que 1, B, Y é, claramente, um caminho completo no GF, assim como, 2, B, X. Por serem caminhos completos, de acordo com o método original para extração de classificadores, eles podem ser interpretados como regras de decisão; entretanto, são regras de decisão que não classificam qualquer instância de dados.

Regras de decisão que não classificam qualquer das instâncias do próprio conjunto de treinamento foram identificadas nesta dissertação, bem como no sistema EFLOWG, como *regras inválidas*, por não terem poder de classificação.

A eliminação de regras inválidas foi uma opção adotada, implementada e está disponibilizada no sistema EFLOWG. Basicamente, regras de decisão são extraídas dos GFs, utilizando a técnica original e, então, cada uma das regras de decisão é testada tendo como entrada as instâncias do próprio conjunto de treinamento. As regras que não classificam qualquer instância de dados do conjunto de treinamento, são eliminadas, e as regras de decisão válidas são mantidas, dando origem a um novo classificador.

Experimentos com o conjunto de instâncias de dados apresentado na Tabela 6.2, que antes apresentavam acurácias irrelevantes, após a implementação da atualização que elimina regras inválidas, passaram a apresentar acurácias melhores, como pode ser observado na Figura 6.4. Tal fato, sugere a obtenção de uma melhora significativa no classificador, e no processo de classificação, como um todo, após a implementação da função que elimina regras inválidas.

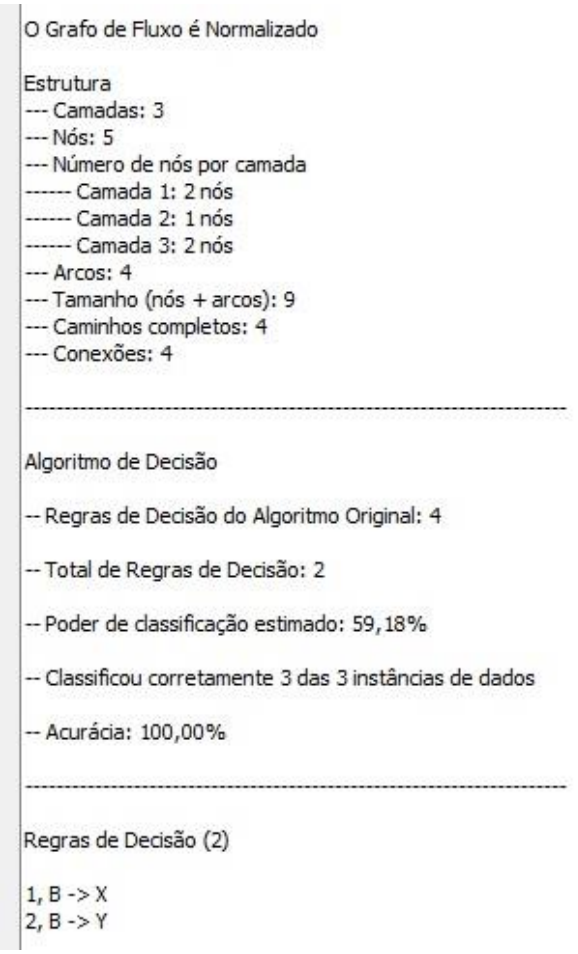


Figura 6.4 Saída do sistema EFLOWG que mostra a acurácia de 100% obtida pelo classificador após a implementação da atualização que elimina regras inválidas.

6.2.5 Novo Método para Extração de Classificadores a partir de GFs

Com a eliminação das regras inválidas do classificador, as regras inconclusivas foram, nos experimentos que utilizaram o GF da Figura 6.1, também, eliminadas. Porém, para garantir que regras inconclusivas não participem dos classificadores ao utilizar outros conjuntos de instâncias de dados, gerando assim, classificadores mais robustos, a atualização discutida na Subseção 6.2.3, a qual faz uso do fator de força das regras para eliminação de regras inconclusivas, foi novamente implementada no sistema EFLOWG.

Dessa forma, o novo método implementado no sistema EFLOWG para extração de classificadores a partir de GFs, pode ser resumido pelos passos a seguir:

(1) Interpretar todos os caminhos completos do GF como regras de decisão que descrevem um classificador associado ao GF;
(2) Testar, utilizando o conjunto de treinamento, o poder de classificação de cada uma das regras de decisão que definem o classificador, interpretadas no passo (1);
    (2.1) Se a regra de decisão não classificar nenhuma instância do conjunto de treinamento (*i.e.*, se for uma regra inválida), ela é eliminada; caso contrário, ela é mantida no classificador.
(3) Detectar regras inconclusivas e manter no classificador somente aquelas com os maiores valores associados ao fator de força.

6.3 Identificando a Necessidade de Tratamento de Atributos com Valores Contínuos ao se Utilizar de GFs

Essa seção faz uma breve análise sobre resultados obtidos, utilizando o método para extração de classificadores a partir de GFs induzidos, a partir dos conjuntos de instâncias de dados apresentados na Tabela 6.1.

Tabela 6.3 Informações sobre GFs induzidos a partir dos conjuntos de instâncias de dados da Tabela 6.1. Na tabela, CI: identificação do conjunto de instâncias de dados; #nós: o número de nós do GF induzido; #arcos: o número de arcos do GF induzido; tamanho: o tamanho do GF induzido (dado por #nós + #arcos), e; #caminhos completos: o número de caminhos que se originam em nós de entrada e têm como destino nós de saída, do GF induzido. Os valores apresentados na tabela referem-se à média dos valores associados aos GFs induzidos durante o processo de 5-validação cruzada.

CI	#nós	#arcos	tamanho	#caminhos completos
Ru	99	109	208	61
ML	1,603	1,600	3,203	800
Sp	453	471	924	255
MC	609	613	1,222	323
LS	1,445	1,439	2,884	720
Ir	120	310	430	2,603
Fo	190	360	550	15,896
Ag	736	1111	1847	989
Fl	268	339	607	203
Se	1033	1149	2182	563
Ec	352	944	1296	559594
BT	162	607	769	5102
HS	91	340	431	3191
VC	1454	1488	2942	362
UK	462	1332	1794	58005
CS	33	63	96	1176
Gl	979	1412	2391	2935770
Wi	1091	1794	2885	5584873
Cr	95	254	349	10527

A partir de cada um dos conjuntos de dados informados na Tabela 6.1, foram induzidos GFs como parte de uma metodologia de avaliação implementada por um processo de 5-validação cruzada.

Os valores apresentados na Tabela 6.3 referem-se às médias das principais características dos GFs induzidos em tal processo a saber, número de nós, número de arcos, tamanho (número de nós + número de arcos) e número de caminhos completos existentes. Fica evidente que, GFs induzidos a partir de conjuntos de instâncias de dados que possuem atributos contínuos são volumosos. Suas estruturas, via de regra, têm um alto número de nós, de arcos e de caminhos completos.

A maneira como a proposta original de GFs trata atributos contínuos (*i.e.*, como se fossem atributos discretos), contribui para que o processo de sumarização do conjunto de dados se torne praticamente inócuo, uma vez que a estrutura do grafo induzido se torna uma representação inconveniente à análise do fluxo dos dados ao longo da estrutura, devido ao alto número de nós, arcos e caminhos presentes. Um exemplo dessa situação

pode ser observado na Figura 6.5, que mostra parte da representação gráfica gerada pelo sistema EFLOWG, do GF induzido a partir do conjunto de instâncias de dados Iris ([Dua & Karra Taniskidou 2017]).

Os classificadores extraídos a partir de GFs induzidos, de acordo com a proposta original, praticamente não generalizam as informações contidas nas instâncias de treinamento, uma vez que, para cada valor contínuo de um determinado atributo, é criado um nó no GF, associado à camada que representa o atributo. Assim sendo, são criados, por atributo (i.e., camada do GF sendo induzido), tantos nós quantos forem os diferentes valores contínuos associados ao atributo em questão, presentes no conjunto de treinamento. A proposta original é, nitidamente, adequada para dados descritos por atributos com valores discretos, e não contínuos. Quando em uso, como parte de um sistema computacional, a proposta original é totalmente inadequada para domínios de dados contínuos, como pode ser facilmente identificado nos valores apresentados na Tabela 6.4, considerando que os resultados apresentados se referem a domínios de dados contínuos, extraídos dos GFs cujas estruturas são apresentadas na Tabela 6.3.

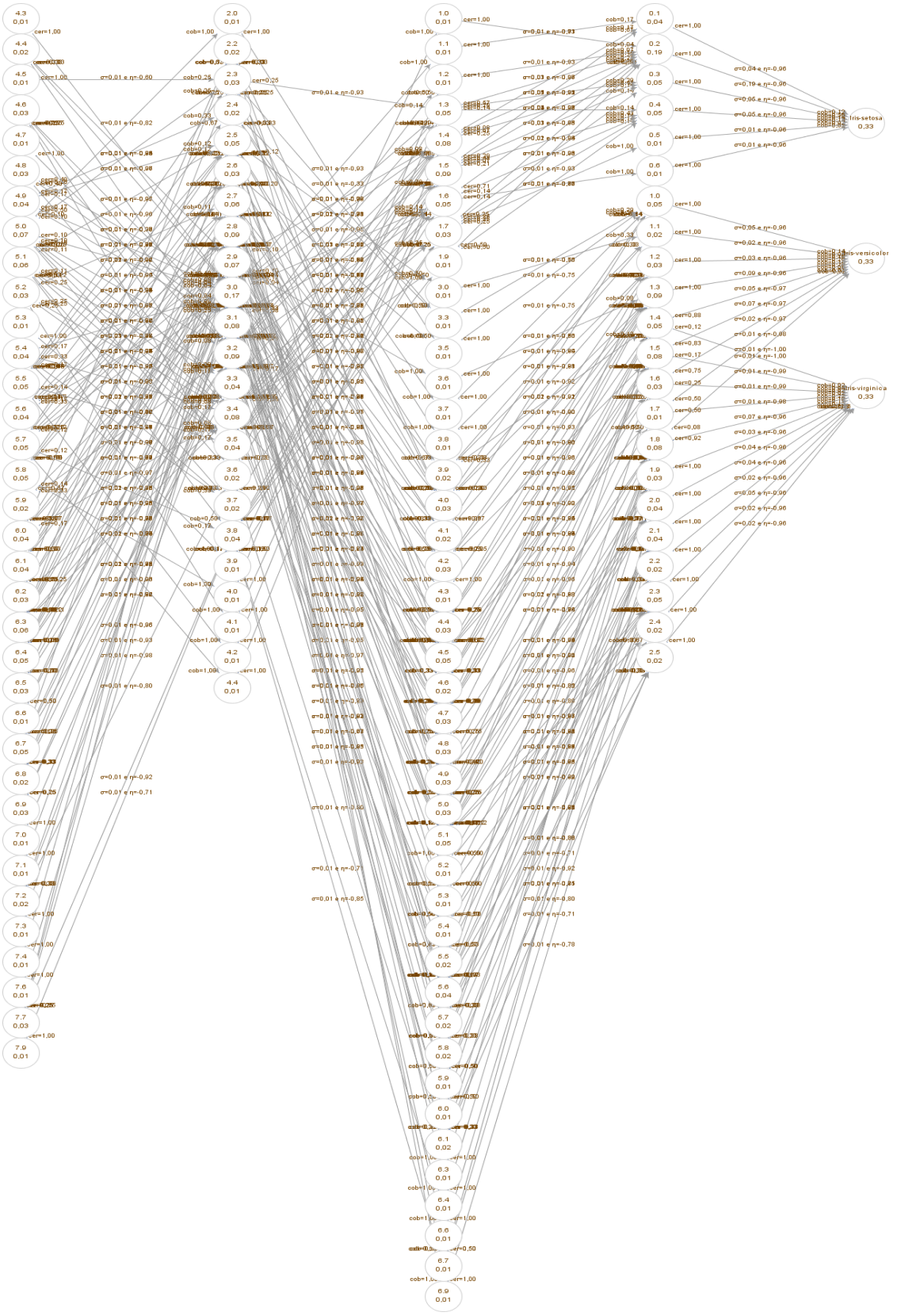


Figura 6.5 Representação gráfica, gerada pelo sistema EFLOWG, do GF induzido a partir do conjunto de instâncias de dados Iris (Ir) para mostrar a complexidade da estrutura gerada com a presença de atributos contínuos.

Tabela 6.4 Informações sobre as características e acurácias dos classificadores extraídos dos GFs, parte de um processo de 5-validação cruzada. Na tabela, CI: identifica o conjunto de instâncias de dados; #TRD: total de regras de decisão do classificador extraído do GF, e; %acurácia: acurácia obtida pelo algoritmo de decisão nos testes de classificação. Os valores apresentados na tabela referem-se à média dos valores obtidos durante o processo de 5-validação cruzada.

CI	# TRD	%acurácia
Ru	61	0.00%
ML	800	0.00%
Sp	255	0.00%
MC	323	0.00%
LS	720	0.00%
Ir	2603	6.00%
Fo	15896	0.00%
Ag	989	0.00%
Fl	203	0.00%
Se	563	0.00%
Ec	270	2.94%
BT	443	36.00%
HS	240	29.03%
VC	248	0.00%
UK	322	0.00%
CS	63	30.00%
Gl	171	0.00%
Wi	142	0.00%
Cr	71	0.00%

Como pode ser observado na Tabela 6.4, a maioria dos resultados foi igual a 0.00%. Tais resultados eram esperados, pois pode acontecer, por exemplo, de haver no conjunto de dados, duas instâncias de dados que possuam, associados a um dado atributo, os valores 1,19 e 1,21. Ao induzir um GF a partir deste conjunto de instâncias de dados, são criados dois nós, com os valores 1,19 e 1,21, respectivamente. Quando o classificador é extraído a partir do GF induzido, o mesmo possui regras de decisão que possuem os valores 1,19 e 1,21. Porém, caso o classificador receba uma instância que possua o valor 1,20, ele não poderá classificá-la, uma vez que ele conhece os valores 1,19 e 1,21, mas não conhece o valor 1,20. Tal fato evidencia que o classificador extraído do GF criou um modelo (*i.e.* adquiriu um aprendizado) muito específico, muito especializado, não sabendo lidar com outros casos, de certa forma, parecidos ou similares. Justamente por este motivo é que os GFs não são indicados para uso com conjuntos de instâncias de dados que possuam atributos contínuos, uma vez que não são capazes de lidar com os mesmos.

Na próxima seção, é descrita e proposta uma extensão de GFs para contemplar a indução e uso de tais estruturas em domínios com dados contínuos; tal proposta foi implementada como um processo de discretização dos atributos contínuos, em uma fase de pré-processamento dos dados, anterior à indução de GFs e de seu uso para a extração de classificadores.

6.4 Grafo de Fluxo Estendido (GFE)

Os GFs originais, propostos por Pawlak, não lidam bem com atributos contínuos, como visto na Seção 6.2. Porém, conjuntos de instâncias de dados do mundo real quase sempre são descritas por alguns, se não todos, atributos contínuos. Como discutido em [Rodrigues & Nicoletti 2018], um cuidadoso levantamento bibliográfico de trabalhos associados a GFs evidencia que o formalismo e resultados teóricos associados a GFs não tiveram impacto e tampouco foram adotados/utilizados em trabalhos cujo foco é aprendizado de máquina. O número de trabalhos acadêmicos que utilizam GFs como estrutura para sumarizar dados de treinamento para posterior extração de classificadores é bem reduzido (ver, por exemplo, [Sun *et al.* 2006], [Yang *et al.* 2011]).

Para resolver, ou no mínimo, suavizar, o problema relacionado a atributos com valores contínuos, uma solução, simples e eficaz, foi proposta e implementada neste trabalho: a de executar, antes da indução do GF, um método de discretização sobre o conjunto de treinamento. Essa solução, chamada de Grafo de Fluxo Estendido (GFE), foi proposta como uma extensão para o formalismo do grafo de fluxo original, uma vez que amplia a possibilidade de utilização dos GFs no ambiente de AM. Tal proposta motivou o desenvolvimento e implementação, junto ao sistema EFLOWG, de uma guia contendo três opções para a realização da discretização de atributos contínuos.

Os experimentos, apresentados a seguir, foram realizados visando o levantamento de dados de forma a auxiliar no processo de identificação de qual método de discretização de atributos contínuos, dentre os três disponibilizados pelo sistema EFLOWG, poderia levar à indução de GFEs que melhor generalizassem as instâncias de dados pertencentes ao conjunto de treinamento, e consequentemente, à extração de classificadores mais robustos e eficientes.

O método utilizado para extração do classificador utilizado nos experimentos, cujos resultados são apresentados nas próximas subseções, é o mesmo proposto na Seção 6.2, o qual lida com regras inválidas e inconclusivas.

6.4.1 Entropia e MDLP

A Tabela 6.5 apresenta as médias de valores obtidos a partir de um processo de 5-validação cruzada, em que algumas características dos GFEs obtidos durante o processo foram contabilizadas e as médias das acurácias dos classificadores extraídos dos GFEs foram calculadas. Para a discretização dos dados, nesse conjunto de experimentos, foi utilizado o método de particionamento do escopo do atributo por Entropia e MDLP [Fayyad & Irani 1993], apresentado e discutido no Capítulo 3.

Tabela 6.5 Informações sobre, as estruturas dos GFEs induzidos a partir dos conjuntos de instâncias de dados da Tabela 6.1, utilizando o método de discretização por Entropia e MDLP ([Fayyad & Irani 1993]) e de desempenho dos classificadores extraídos dos GFs induzidos. Na tabela, #nós: número médio de nós dos GFs induzidos; #arcos: número médio de arcos dos GFEs induzidos; tamanho: tamanho médio dos GFEs dado pela soma de #nós e #arcos; #caminhos completos: número médio de caminhos completos dos GFEs induzidos; #TR: número total de regras de decisão do classificador, e; %acurácia: acurácia obtida pelo algoritmo de decisão em testes de classificação. Os valores apresentados na tabela referem-se à média dos valores obtidos durante o processo de 5-validação cruzada.

	Grafo de Fluxo Estendido				**Classificador**	
CI	**#nós**	**#arcos**	**tamanho**	**#caminhos completos**	**#TR**	**%acurácia**
Ru	12	11	23	7	7	97.33%
ML	11	20	31	31	13	91.50%
Sp	14	12	26	10	6	98.33%
MC	12	15	27	11	9	99.75%
LS	13	16	29	16	10	88.67%
Ir	15	27	42	63	19	93.33%
Fo	26	55	81	425	48	94.44%
Ag	20	49	69	87	34	99.75%
Fl	8	12	20	11	8	85.42%
Se	30	72	102	4360	64	90.48%
Ec	24	43	67	1152	49	82.35%
BT	9	10	19	8	4	76.67%
HS	6	7	13	4	2	72.58%
VC	19	40	59	600	77	83.87%
UK	17	34	51	124	28	91.36%
CS	15	33	48	120	35	75.00%
Gl	35	67	102	27216	73	97.67%
Wi	43	110	153	1658880	137	97.22%
Cr	13	18	31	48	19	83.33%

Os resultados, apresentados na Tabela 6.5, mostram que a proposta de estender o GF original, executando uma etapa de pré-processamento dos dados com base na discretização de atributos contínuos por Entropia e MDLP, gerou melhores resultados nos testes de classificação, quando comparados àqueles exibidos na Tabela 6.4.

6.4.2 Amplitudes Iguais

A Tabela 6.6 apresenta informações sobre os experimentos realizados com os conjuntos de instâncias de dados da Tabela 6.1, utilizando o método de discretização por Intervalos de Amplitudes Iguais (*Equal Width*). Os valores apresentados, novamente, são referentes à médias, uma vez que foi utilizado o processo de 5-validação cruzada para realização do experimento. O número de *bins*, dado obrigatório ao se utilizar tal método de discretização, foi definido como 4, uma vez que, na maioria dos experimentos utilizando Entropia e MDLP, os valores dos atributos contínuos foram sumarizados em partições compostas por 4 subintervalos.

Os resultados apresentados na Tabela 6.6 evidenciam que a utilização do processo de discretização de atributos contínuos por meio do método de discretização por Amplitudes Iguais, produziu resultados insatisfatórios, na maioria das vezes, quando comparados àqueles produzidos pelo método de discretização por Entropia e MDLP.

Dentre os 19 experimentos realizados utilizando a discretização por Amplitudes Iguais, 5 deles, relacionados aos conjuntos de instâncias de dados ML, Ir, Se, BT e UK, apresentaram melhores desempenhos em relação aos resultados apresentados na Tabela 6.4. Em contrapartida, houve grandes quedas em algumas acurácias, como pode ser visto nos resultados relacionados aos conjuntos de instâncias de dados MC, Ls, Fo, Fl e VC.

Tabela 6.6 Informações sobre as estruturas dos GFEs induzidos a partir dos conjuntos de instâncias de dados da Tabela 6.1, utilizando o método de discretização por Intervalos de Amplitudes Iguais (*Equal Width*), e de desempenho dos classificadores extraídos dos GFEs. Na tabela, #nós: número médio de nós dos GFEs induzidos; #arcos: número médio de arcos dos GFEs induzidos; tamanho: soma de #nós e #arcos; #caminhos completos: número de caminhos completos do GFE induzido; #TR: número total de regras de decisão do classificador extraído do GFE induzido, e; %acurácia: acurácia do classificador. O processo de 5-validação cruzada foi utilizado para realização destes experimentos.

	Grafo de Fluxo Estendido				**Classificador**	
CI	**#nós**	**#arcos**	**tamanho**	**#caminhos completos**	**#TR**	**%acurácia**
Ru	12	16	28	16	10	85.33%
ML	11	24	35	32	16	73.70%
Sp	14	18	32	22	10	57.33%
MC	11	21	32	26	14	82.75%
LS	14	22	36	30	11	47.67%
Ir	19	36	55	107	79	82.00%
Fo	31	78	109	1846	1188	67.82%
Ag	15	29	44	52	16	56.09%
Fl	10	20	30	22	14	84.17%
Se	31	83	114	6641	3384	65.24%
Ec	31	75	106	5900	1072	51.49%
BT	18	29	47	49	24	75.40%
HS	14	34	48	88	45	72.22%
VC	25	58	83	1451	665	46.45%
UK	24	74	98	2713	1024	71.22%
CS	14	32	56	115	57	50.00%
Gl	44	107	151	195837	46272	31.78%
Wi	55	165	220	17207012	170	83.33%
Cr	21	53	74	592	342	48.89%

6.4.3 Iguais Frequências

A Tabela 6.7 traz informações sobre os experimentos realizados utilizando o método de discretização por Iguais Frequências (*Equal Frequency*). Os conjuntos de instâncias de dados são os mesmos utilizados nos experimentos anteriores (Tabela 6.1) e o número de *bins* é, novamente, 4. Da mesma forma, a opção referente à 5-validação cruzada foi utilizada para a realização dos experimentos. Mostra, também, que as acurácias obtidas com o método de discretização por Iguais Frequências foram parecidas àquelas obtidas com a utilização do método de discretização por Amplitudes Iguais. Tais resultados foram inferiores àqueles apresentados na Tabela 6.5, relacionados ao método de discretização por Entropia e MDLP.

Tabela 6.7 Informações médias sobre, as estruturas dos GFEs induzidos a partir dos conjuntos de instâncias de dados da Tabela 6.1, utilizando o método de discretização por Intervalos de Iguais Frequências (*Equal Frequency*), e também, sobre os classificadores extraídos dos GFEs correspondentes. Na tabela, #nós: número médio de nós dos GFEs induzidos; #arcos: número médio de arcos dos GFEs induzidos; tamanho: soma de #nós e #arcos; #caminhos completos: número de caminhos completos do GFE induzido; #TR: número total de regras de decisão do classificador extraído do GFE induzido, e; %acurácia: acurácia do classificador. O processo de 5-validação cruzada foi utilizado para realização destes experimentos.

	Grafo de Fluxo Estendido				**Classificador**	
CI	**#nós**	**#arcos**	**tamanho**	**#caminhos completos**	**#TR**	**%acurácia**
Ru	12	19	31	22	12	81.33%
ML	11	23	34	31	16	72.90%
Sp	14	26	40	40	16	57.67%
MC	11	23	34	28	16	79.25%
LS	14	24	38	36	13	44.11%
Ir	19	46	65	230	144	78.67%
Fo	31	92	123	4468	2835	65.52%
Ag	15	31	46	60	16	53.43%
Fl	10	23	33	28	16	86.25%
Se	13	93	114	15788	7142	54.29%
Ec	31	81	112	9407	1708	58.93%
BT	18	43	61	126	63	75.40%
HS	14	40	54	128	64	72.55%
VC	27	89	116	9664	3942	75.48%
UK	24	74	98	2764	1024	73.20%
CS	18	39	57	367	183	62.50%
Gl	46	155	201	3211264	194	28.77%
Wi	55	194	249	89391104	178	82.00%
Cr	23	69	92	2547	1273	31.11%

Dessa forma, foi decidido que a proposta de extensão dos GFs de Pawlak utilizará, por *default*, o método de discretização por Entropia e MDLP. Porém, os outros dois métodos de discretização foram mantidos no sistema EFLOWG, com objetivo de oferecer aos usuários do sistema mais opções para análises, estudos e investigações relacionadas aos GFs e GFEs.

6.5 Experimentos utilizando os Grafos de Fluxo Estendidos

Essa seção apresenta resultados dos experimentos realizados, com base na metodologia proposta na Seção 6.1, utilizando o novo método para extração de classificadores, definido na Seção 6.2, a partir dos Grafos de Fluxo Estendidos, propostos na Seção 6.4.

Após a implementação, junto ao sistema computacional EFLOWG, do novo método para extração de classificadores a partir de GFs, os experimentos, apresentados na Seção 6.3, foram novamente realizados, com o objetivo de se verificar o comportamento e a viabilidade do novo método, juntamente com a nova proposta dos Grafos de Fluxo Estendidos (GFEs). Os conjuntos de instâncias de dados utilizados nestes experimentos são aqueles apresentados na Tabela 6.1. A opção utilizada para indução de GFEs foi a 5-validação cruzada. A Tabela 6.8 exibe a média dos valores dos resultados obtidos nos experimentos.

Tabela 6.8 Informações médias sobre, as estruturas dos GFEs, induzidos a partir dos conjuntos de instâncias de dados da Tabela 6.1, e também, sobre os classificadores extraídos dos GFEs correspondentes, utilizando o novo método de extração de algoritmo de decisão (livre de regras inválidas e de regras inconclusivas), em que: #nós: número de nós; #arcos: número de arcos; #caminhos completos: número de caminhos completos; #RE, representa o número de regras eliminadas (entre inválidas e inconclusivas); #TR: número total de regras de decisão do classificador, e; %acurácia, representa a acurácia obtida pelo classificador em testes de classificação.

	Grafo de Fluxo Estendido				**Classificador**		
CI	**#nós**	**#arcos**	**tamanho**	**#caminhos completos**	**#RE**	**#TR**	**%acurácia**
Ru	12	11	23	7	0	7	97.33%
ML	11	20	31	31	18	13	91.50%
Sp	14	12	26	10	4	6	98.33%
MC	12	15	27	11	2	9	99.75%
LS	13	16	29	16	6	10	88.67%
Ir	15	27	42	63	44	19	93.33%
Fo	26	55	81	425	377	48	94.44%
Ag	20	49	69	87	53	34	99.75%
Fl	8	12	20	11	3	8	85.42%
Se	30	72	102	4360	4296	64	90.48%
Ec	24	43	67	1152	1103	49	82.35%
BT	9	10	19	8	4	4	76.67%
HS	6	7	13	4	2	2	72.58%
VC	19	40	59	600	523	77	83.87%
UK	17	34	51	124	96	28	91.36%
CS	15	33	48	120	85	35	75.00%
Gl	35	67	102	27216	27143	73	97.67%
Wi	43	110	153	1658880	1658743	137	97.22%
Cr	13	18	31	48	29	19	83.33%

O novo método utilizado para extração de classificadores a partir de GFEs mostrou-se eficaz: dos 19 experimentos realizados, e registrados na Tabela 6.8, 11 deles apresentaram acurácia acima de 90%. A menor acurácia obtida, relacionada ao conjunto de instâncias de dados HS, foi de 72,56%, fato este que vem a ser um indicador interessante para um classificador.

Por fim, experimentos utilizando os mesmos conjuntos de instâncias de dados foram realizados, porém, desta vez, no ambiente Weka, versão 3.8, utilizando os algoritmos J48, Naive Bayes, K-Nearest Neighbors e Support Vector Machine. A motivação para realização destes experimentos, foi a de fornecer uma comparação entre as acurácias obtidas pelos GFEs e aquelas obtidas por 4 algoritmos amplamente estabelecidos no ambiente de AM. Em todos os experimentos, o processo de 5-validação cruzada foi utilizado. Os resultados coletados, para fins de comparações, são apresentados na Tabela 6.9, juntamente com os resultados obtidos pelos classificadores extraídos dos GFEs, apresentados na Tabela 6.8. Para facilitar a compreensão dos resultados, os melhores foram destacados com negrito.

Tabela 6.9 Acurácias obtidas pelos classificadores extraídos dos GFEs (GFE), J48 (J48), Naive Bayes (NB), K-Nearest Neighbor (KNN) e Support Vector Machine (SVM), utilizando os conjuntos de instâncias de dados da Tabela 6.1, com vistas à comparações de desempenho.

CI	GFE	J48	NB	KNN	SVM
Ru	97.33%	98.67%	98.67%	**100.00%**	**100.00%**
ML	91.50%	97.90%	98.30%	98.20%	**98.80%**
Sp	98.33%	**100.00%**	**100.00%**	**100.00%**	**100.00%**
MC	99.75%	**100.00%**	**100.00%**	**100.00%**	**100.00%**
LS	88.67%	**99.89%**	99.78%	**99.89%**	99.78%
Ir	93.33%	94.00%	96.00%	94.00%	**96.67%**
Fo	94.44%	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Ag	**99.75%**	99.62%	99.33%	99.33%	95.68%
Fl	85.42%	97.91%	95.83%	**100.00%**	88.33%
Se	90.48%	91.90%	90.95%	92.38%	**93.33%**
Ec	82.35%	82.44%	**85.12%**	80.95%	83.03%
BT	76.67%	**77.27%**	75.13%	71.65%	76.20%
HS	72.58%	70.92%	**75.16%**	68.95%	73.52%
VC	**83.87%**	81.94%	83.22%	76.12%	75.16%
UK	91.36%	**93.05%**	89.58%	84.86%	92.80%
CS	**75.00%**	62.50%	62.50%	57.75%	60.00%
Gl	**97.67%**	97.20%	84.58%	90.19%	79.90%
Wi	97.22%	92.70%	97.17%	95.50%	**98.31%**
Cr	83.33%	**90.00%**	83.33%	88.89%	**90.00%**

Observando a Tabela 6.9, nota-se que os algoritmos J48 e Support Vector Machine obtiveram, cada um, melhores acurácias em relação aos GFEs em 13 dos 19 experimentos realizados. Já os algoritmos Naïve Bayes e K-Nearest Neighbors, produziram, cada um, acurácias melhores que os GFEs em 11 dos 19 experimentos. Os resultados obtidos pelos GFEs conseguiram superar os demais algoritmos em 4 dos 19 experimentos.

Apesar de, no geral, terem obtido acurácias inferiores àquelas obtidas pelos demais algoritmos, os classificadores compostos por regras de decisão extraídas de GFEs, obtiveram, diversas vezes, acurácias próximas daquelas obtidas pelos outros algoritmos. Considerando que os outros algoritmos são bem estabelecidos no ambiente de AM, o resultado foi promissor, e mostra que, estudos mais focados em determinadas partes do formalismo que sustenta os GFs, e também, na extração de classificadores a partir de GFs, possam, talvez, incrementar ainda mais, a acurácia dos classificadores extraídos a partir dos GFEs.

6.6 Considerações Finais

Essa seção apresentou, entre outros itens, a metodologia utilizada, descrição dos experimentos e, também, algumas análises sobre os experimentos utilizando os GFs.

O método original para extração de classificadores a partir de GFs, proposto por Pawlak, foi apresentado, testado e discutido, e um novo método, baseado no original, porém com melhorias, foi proposto.

Outra importante proposta feita no capítulo diz respeito aos Grafos de Fluxo Estendidos (GFEs), os quais possibilitam a utilização de GFs com conjuntos de instâncias de dados que possuem atributos contínuos, o que era, até então, inviável, devido à complexidade das estruturas induzidas e à generalização nos testes de classificação que não ocorrem.

Foram apresentados resultados de testes classificação realizados utilizando os GFEs juntamente com o novo método de extração de classificadores a partir de GFs, os quais apresentaram melhoras interessantes em relação aos métodos originais.

Por fim, os resultados obtidos pelos GFEs, usando as técnicas propostas, foram comparados aos resultados obtidos pelos algoritmos J48, Naive Bayes, K-Nearest Neighbors e Support Vector Machine, estes últimos implementados junto ao ambiente Weka. As comparações mostraram que, mesmo apesar das melhorias implementadas, os resultados obtidos pelos GFEs, apesar de promissores, não mostraram-se, inicialmente, competitivos em relação aos resultados obtidos pelos demais algoritmos. Porém, é

necessária uma análise estatística para verificar se as diferenças, encontradas nos resultados das comparações entre GFEs e os outros algoritmos, são, de fato, relevantes.

O Capítulo 7, a seguir, apresenta uma investigação empírica sobre o impacto de diferentes ordenações de atributos sobre os GFEs e os classificadores extraídos destes.

Capítulo 7
Considerações sobre o Impacto da Ordem dos Atributos sobre Classificadores Extraídos de GFs

7.1 Considerações Iniciais

Durante testes preliminares, realizados com vistas à verificação do funcionamento do sistema EFLOWG e análise das estruturas dos GFs induzidos, observou-se que a ordem em que os atributos comparecem no conjunto de treinamento pode influenciar a estrutura do GF induzido e, consequentemente, os classificadores extraídos.

Este capítulo tem foco em uma investigação inicial empírica sobre o impacto causado por diferentes ordenações das camadas de um grafo de fluxo, uma vez que refletem a ordem com que atributos comparecem nas descrições das instâncias de treinamento.

O capítulo está organizado da seguinte maneira: a Seção 7.2 aborda a descrição de instâncias em relação à ordem de atributos utilizados para tal; a Seção 7.3 apresenta um estudo de caso realizado sobre um conjunto de instâncias de dados sintético (criado artificialmente) e suas possíveis sequências de atributos; a Seção 7.4 realiza um estudo de caso utilizando um conjunto real de instâncias de dados; a Seção 7.5 investiga o impacto da ordenação de atributos baseada no ganho de informação; a Seção 7.6 realiza alguns experimentos de modo a determinar qual sequência de atributos, para cada conjunto de instâncias de dados, produz melhores acurácias, e; por fim, a Seção 7.6 apresenta as considerações finais do Capítulo.

7.2 Descrição de Instâncias e a Influência da Ordem em que Atributos são Considerados

Cada instância de dados, pertencente a um conjunto de instâncias de dados, é descrita por n atributo(s), n > 0, e uma classe associada. A ordem em que os atributos são

dispostos não interfere na descrição das instâncias de dados, uma vez que cada atributo define claramente a característica em questão e possui seus respectivos valores associados. Dessa forma, tem-se que, a ordem de atributos A_1, A_2, A_3 e A_4 descreve uma instância de dados tão bem quanto a ordem de atributos A_3, A_1, A_4 e A_2. Apesar de diferentes, ambas as ordens de atributos apresentadas podem descrever, sem descaracterizar, uma instância de dados.

7.3 Estudo de Caso 1: Conjunto de Instâncias de Dados Sintético

A Tabela 7.1 apresenta um conjunto de instâncias de dados sintético, *i.e.*, gerado artificialmente, que foi utilizado nos experimentos abordados nesta Subseção, relativos ao estudo de como a ordem em que os atributos comparecem na descrição de instâncias de treinamento influencia o classificador, construído como o conjunto de regras de decisão extraídas do GF.

Cada uma das instâncias do conjunto de instâncias de dados considerado é descrita, originalmente, pelos atributos A_1, A_2, A_3 e A_4, nessa ordem, e todas elas possuem classe associada.

Tabela 7.1 O conjunto de 14 instâncias de dados, cada uma delas descrita pelos atributos A_1, A_2, A_3, A_4 e uma classe associada.

#ID	A_1	A_2	A_3	A_4	Classe
1	S	L	H	O	1
2	S	L	H	I	1
3	N	L	H	O	2
4	C	A	H	O	2
5	C	B	N	O	2
6	C	B	N	I	1
7	N	B	N	I	2
8	S	A	H	O	1
9	S	B	N	O	2
10	C	A	N	O	2
11	S	A	N	I	2
12	N	A	H	I	2
13	N	L	N	O	2
14	C	A	H	I	1

Como o conjunto de atributos $A=\{A_1, A_2, A_3, A_4\}$ possui 4 atributos, existem 24 possíveis ordenações diferentes desses quatro atributos, uma vez que representam as possíveis permutações de quatro elementos *i.e.*, $4! = 1 \times 2 \times 3 \times 4 = 24$.

Nos diagramas mostrados em Figura 7.1, Figura 7.2, Figura 7.3 e Figura 7.4, as 24 possibilidades de ordenação dos atributos do conjunto de instâncias de dados da Tabela 7.1 são apresentadas.

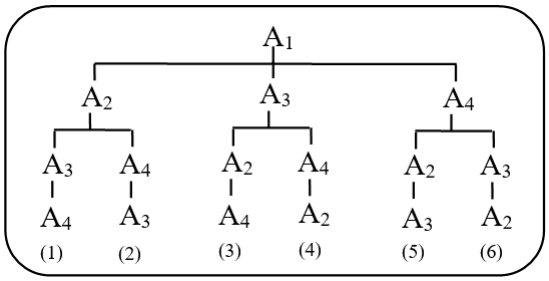


Figura 7.1 As 6, (1), ..., (6), possíveis ordenações de atributos que podem ser definidas tendo A1 como primeiro atributo.

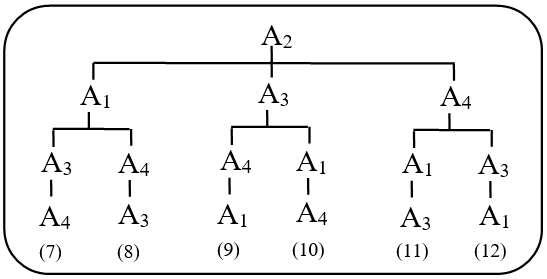


Figura 7.2 As 6, (7), ..., (12), possíveis ordenações de atributos que odem ser definidas tendo A_2 como primeiro atributo.

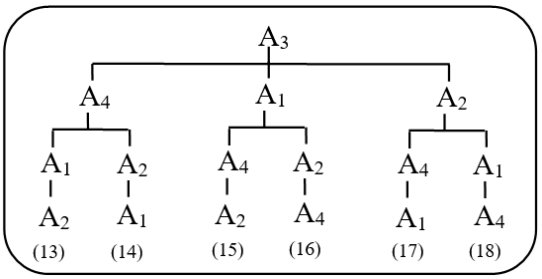


Figura 7.3 As 6, (13), ..., (18), possíveis ordenações de atributos que podem ser definidas tendo A_3 como primeiro atributo.

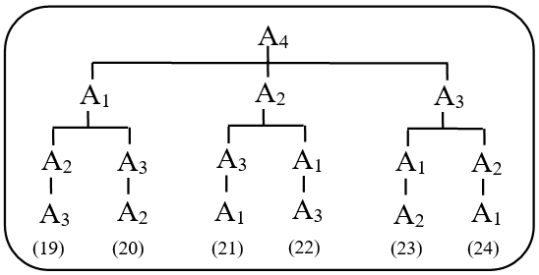


Figura 7.4 As 6, (19), ..., (24), possíveis ordenações de atributos que podem ser definidas tendo A_4 como primeiro atributo.

Foram realizados 24 experimentos relacionados à indução de GFs e extração de classificadores a partir deles, em que as ordens dos atributos do conjunto de treinamento da Tabela 7.1, foram alteradas, de acordo com as sequências (1), ..., (24). Como comentado anteriormente, a alteração da ordem dos atributos do conjunto de treinamento implica alteração da ordem das camadas dos grafos de fluxo induzidos.

A verificação empírica de um possível impacto que a ordem em que os atributos descrevem as instâncias de treinamento poderia ter, no classificador extraído do grafo de fluxo induzido, foi feita por meio de um experimento simples, utilizando as 14 instâncias de dados mostradas na Tabela 7.1. Tal conjunto foi dividido uma única vez em dois subconjuntos, um de treinamento e um de teste. O conjunto de treinamento foi definido como Tr = {#2, #3, #4, #5, #6, #8, #9, #11, #12, #13}, |Tr| = 10, e o de teste, como Te = {#1, #7, #10, #14}, |Te| = 4.

Para cada uma das 24 ordenações de atributo, o experimento consistiu na (1) indução de um GF utilizando, como conjunto de treinamento, 80% das instâncias do conjunto de instâncias de dados, descritas seguindo a ordem estabelecida pela ordenação sendo considerada, seguida pela (2) extração, a partir do GF, de um classificador, representado por um conjunto de regras de decisão e, então, (3) na avaliação do classificador, utilizando o conjunto de teste. Ao término de cada um dos 24 experimentos, foram armazenadas informações sobre o número total de regras de decisão geradas, o número de regras de decisão inválidas e inconclusivas, e o número de regras de decisão consistentes extraídas do respectivo grafo de fluxo. Também foram coletados dados sobre a acurácia (precisão) dessas regras de decisão, obtida utilizando o conjunto de teste. A Tabela 7.2 resume os resultados dos 24 experimento conduzidos.

Tabela 7.2 Resultados obtidos nos experimentos utilizando diferentes ordenações de atributos de acordo com as sequências explicitadas em (1), ..., (24) das figuras Figura 7.1, Figura 7.2, Figura 7.3 e Figura 7.4. Na tabela, seq: o número de identificação da sequência; ordem dos atributos: ordem na qual os atributos foram considerados; #trd: total de regras de decisão encontradas; #rdi: número de regras de decisão inválidas e inconclusivas; #rdc: número de regras de decisão consistentes, e; %acurácia: acurácia (precisão) obtida pelo classificador no teste de classificação. As linhas em negrito apresentam os melhores resultados obtidos.

seq	**ordem dos atributos**				**#trd**	**#rdi**	**#rdc**	**%acu**
(1)	A_1	A_2	A_3	A_4	44	31	13	66.67%
(2)	A_1	A_2	A_4	A_3	56	43	13	66.67%
(3)	A_1	A_3	A_2	A_4	40	28	12	33.33%
(4)	A_1	A_3	A_4	A_2	60	48	12	33.33%
(5)	**A_1**	**A_4**	**A_2**	**A_3**	**60**	**46**	**14**	**100.00%**
(6)	A_1	A_4	A_3	A_2	45	33	12	33.33%
(7)	A_2	A_1	A_3	A_4	46	36	12	33.33%
(8)	A_2	A_1	A_4	A_3	56	43	13	66.67%
(9)	**A_2**	**A_3**	**A_4**	**A_1**	**50**	**36**	**14**	**100.00%**
(10)	A_2	A_3	A_1	A_4	50	38	12	33.33%
(11)	**A_2**	**A_4**	**A_1**	**A_3**	**72**	**58**	**14**	**100.00%**
(12)	**A_2**	**A_4**	**A_3**	**A_1**	**60**	**46**	**14**	**100.00%**
(13)	A_3	A_4	A_1	A_2	44	32	12	33.33%
(14)	A_3	A_4	A_2	A_1	46	33	13	66.67%
(15)	A_3	A_1	A_4	A_2	54	42	12	33.33%
(16)	A_3	A_1	A_2	A_4	48	36	12	33.33%
(17)	**A_3**	**A_2**	**A_4**	**A_1**	**50**	**36**	**14**	**100.00%**
(18)	A_3	A_2	A_1	A_4	46	34	12	33.33%
(19)	A_4	A_1	A_2	A_3	48	35	13	66.67%
(20)	A_4	A_1	A_3	A_2	54	41	13	66.67%
(21)	**A_4**	**A_2**	**A_3**	**A_1**	**50**	**36**	**14**	**100.00%**
(22)	A_4	A_2	A_1	A_3	48	36	12	33.33%
(23)	A_4	A_3	A_1	A_2	48	36	12	33.33%
(24)	A_4	A_3	A_2	A_1	38	25	13	66.67%

Todas as sequências explicitadas possibilitaram a extração de classificadores que classificaram corretamente, no mínimo, 33.33% das instâncias presentes no conjunto de teste. As melhores acurácias (100%), obtidas nas classificações, foram observadas quando do uso das sequencias (5), (9), (11), (12), (17) e (21).

De acordo com os resultados, todas sequências de atributos em que A_2 ou A_4 eram o último atributo, induziram GFs que permitiram a extração de classificadores que obtiveram acurácia máxima de 66.67%. Das 6 vezes que a acurácia obtida foi igual a 100%, em 4 delas, o último atributo era A_1 e em 2 delas, A_3. O atributo A_1, comparecendo como último atributo da sequência, possibilitou, em 2 das 6 sequências, acurácia mínima de 66.67%. Nas outras 4 sequências com A_1 na última posição, foi obtida acurácia de 100%, o que evidencia que tal atributo contribuiu na boa classificação de instâncias quando presente na última posição da sequência de atributos.

O experimento anterior foi refeito, desta vez por meio da execução de um esquema de 5-validação cruzada. De acordo com o esquema, para cada uma das 24 possíveis ordenações de atributos, a sequência constituída pelos processos de: (1) indução do GF usando o conjunto de treinamento; (2) extração do conjunto de regras a partir do GF, para compor o classificador e (3) avaliação do classificador no conjunto de teste, foi executada cinco vezes. Ao final das cinco execuções da sequência foi feita a média das contagens e resultados de acurácia obtidos em cada uma delas.

Como mostra a Tabela 7.3, apenas 2 das 24 sequências, (6) e (8), de atributos levaram a acurácias menores que 57.14%. As sequências (9), (10), (17) e (21), apresentaram as melhores acurácias entre os experimentos realizados, com destaque para a sequência (10), a qual obteve acurácia de 92.86%. Dessas 4 sequências com melhores acurácias, os atributos A_1 e A_4 estão presentes na última, ou penúltima, posição da sequência, em 3 das 4 vezes.

Tabela 7.3 Resultados obtidos nos experimentos utilizando o processo de 5-validação cruzada sobre as ordenações explicitadas em (1), ..., (24) das figuras Figura 7.1, Figura 7.2, Figura 7.3 e Figura 7.4. Na tabela, seq: número de identificação da sequência; ordem dos atributos: ordem na qual os atributos foram considerados; #trd: total de regras de decisão encontradas; #rdi: número de regras de decisão inválidas e inconclusivas; #rdc: número de regras de decisão consistentes e %acurácia: acurácia (precisão) obtida pelo classificador no teste de classificação.

seq	**ordem dos atributos**				**#trd**	**#rdi**	**#rdc**	**%acu**
(1)	A_1	A_2	A_3	A_4	45	33	12	57.14%
(2)	A_1	A_2	A_4	A_3	54	41	13	64.29%
(3)	A_1	A_3	A_2	A_4	53	40	13	71.43%
(4)	A_1	A_3	A_4	A_2	63	50	13	64.29%
(5)	A_1	A_4	A_2	A_3	51	38	13	71.43%
(6)	A_1	A_4	A_3	A_2	47	36	11	21.43%
(7)	A_2	A_1	A_3	A_4	57	44	13	64.29%
(8)	A_2	A_1	A_4	A_3	51	39	12	42.86%
(9)	A_2	A_3	A_4	A_1	47	34	13	85.71%
(10)	**A_2**	**A_3**	**A_1**	**A_4**	**57**	**44**	**13**	**92.86%**
(11)	A_2	A_4	A_1	A_3	55	42	13	78.57%
(12)	A_2	A_4	A_3	A_1	56	43	13	78.57%
(13)	A_3	A_4	A_1	A_2	57	44	13	71.43%
(14)	A_3	A_4	A_2	A_1	46	34	12	57.14%
(15)	A_3	A_1	A_4	A_2	62	50	12	57.14%
(16)	A_3	A_1	A_2	A_4	56	43	13	64.29%
(17)	A_3	A_2	A_4	A_1	47	34	13	85.71%
(18)	A_3	A_2	A_1	A_4	44	12	32	57.14%
(19)	A_4	A_1	A_2	A_3	44	31	13	64.29%
(20)	A_4	A_1	A_3	A_2	52	12	40	57.14%
(21)	A_4	A_2	A_3	A_1	46	33	13	85.71%
(22)	A_4	A_2	A_1	A_3	52	40	12	57.14%
(23)	A_4	A_3	A_1	A_2	55	43	12	57.14%
(24)	A_4	A_3	A_2	A_1	37	25	12	57.14%

O padrão de resultados presente nos dados da Tabela 7.2 não foi mantido quando da execução do processo de 5-validação cruzada. As sequências (5), (9), (11), (12), (17) e (21), que possuíam acurácia de 75% na Tabela 7.2, tiveram, no máximo, acurácia de 85.71% na Tabela 7.3, enquanto as sequências (10) e (13), que tiveram acurácia <= 50% na Tabela 7.2, apresentaram, respectivamente, acurácias de 92.86% e 71.43%, quando realizado o experimento utilizando 5-validação cruzada.

7.4 Estudo de Caso 2: Conjunto de Instâncias de Dados Reais

Esta subseção realiza os experimentos utilizando o conjunto de instâncias de dados Iris, que é um conjunto real de dados. A motivação é poder verificar se o que foi verificado junto a um conjunto de instâncias de dados sintético, se aplica a um conjunto de instâncias de dados reais.

O conjunto de instâncias de dados Iris ([Dua & Karra Taniskidou 2017]), é composto por 150 instâncias de dados, cada uma delas descrita por 4 atributos contínuos e possuindo uma classe associada. O conjunto de atributos do Iris é, originalmente, dado por A={sepallength, sepalwidth, petallength, petalwidth} e o conjunto de classes associadas as instâncias de dados do Iris é dado por C={Iris-setosa, Iris-versicolor, Iris-virginica}. Assim, é possível obter $4! = 1 \times 2 \times 3 \times 4 = 24$ diferentes sequências de atributos a partir do conjunto de atributos A, definido anteriormente.

Para os experimentos, foi utilizada proposta de GF Estendido, em que os atributos contínuos são discretizados antes da geração do GF. O conjunto de treinamento utilizado foi composto por 80% das instâncias. Os resultados dos experimentos realizados, utilizando cada uma das 24 possíveis sequências de atributos do conjunto de instâncias de dados Iris, são apresentados nas Tabela 7.4 e Tabela 7.5. Nos experimentos apresentados na Tabela 7.4, para cada diferente sequência de atributos, foi induzido apenas um GF Estendido, do qual foi extraído um classificador, que, por sua vez, foi testado e teve seus resultados registrados. Já para os experimentos apresentados na Tabela 7.5, foi utilizada a técnica de 5-validação cruzada.

Tabela 7.4 Resultados obtidos nos experimentos utilizando a indução de apenas um GF Estendido, utilizando as possíveis ordenações dos atributos. Na tabela, seq: número de identificação da sequência; ordem dos atributos: ordem na qual os atributos foram considerados; #trd: total de regras de decisão encontradas; #rdi: número de regras de decisão inconclusivas e inválidas; #rdc: número de regras de decisão consistentes, e; %acu: acurácia (precisão) do classificador no conjunto de teste. As linhas em negrito apresentam as melhores acurácias.

seq	**ordem dos atributos**				**#trd**	**#rdi**	**#rdc**	**%acu**
(1)	sepallength	sepalwidth	petallength	petalwidth	60	42	18	93.33%
(2)	sepallength	sepalwidth	petalwidth	petallength	63	44	19	93.33%
(3)	sepallength	petallength	petalwidth	sepalwidth	81	62	19	93.33%
(4)	sepallength	petallength	sepalwidth	petalwidth	105	86	19	90.00%
(5)	sepallength	petalwidth	petallength	sepalwidth	90	71	19	96.67%
(6)	sepallength	petalwidth	sepalwidth	petallength	82	64	18	96.67%
(7)	**sepalwidth**	**sepallength**	**petallength**	**petalwidth**	**48**	**29**	**19**	**100.00%**
(8)	**sepalwidth**	**sepallength**	**petalwidth**	**petallength**	**54**	**35**	**19**	**100.00%**
(9)	sepalwidth	petallength	sepallength	petalwidth	72	54	19	90.00%
(10)	sepalwidth	petallength	petalwidth	sepallength	81	62	19	93.33%
(11)	sepalwidth	petalwidth	petallength	sepallength	72	53	19	86.67%
(12)	sepalwidth	petalwidth	sepallength	petallength	57	39	18	90.00%
(13)	petallength	sepallength	sepalwidth	petalwidth	105	86	19	96.67%
(14)	petallength	sepallength	petalwidth	sepalwidth	153	134	19	90.00%
(15)	petallength	sepalwidth	petalwidth	sepallength	171	152	19	90.00%
(16)	petallength	sepalwidth	sepallength	petalwidth	108	89	19	93.33%
(17)	petallength	petalwidth	sepalwidth	sepallength	84	65	19	93.33%
(18)	petallength	petalwidth	sepallength	sepalwidth	80	62	18	93.33%
(19)	petalwidth	sepallength	petallength	sepalwidth	126	108	18	83.33%
(20)	petalwidth	sepallength	sepalwidth	petallength	105	86	19	93.33%
(21)	petalwidth	sepalwidth	petallength	sepallength	153	134	19	90.00%
(22)	**petalwidth**	**sepalwidth**	**sepallength**	**petallength**	**108**	**89**	**19**	**100.00%**
(23)	petalwidth	petallength	sepalwidth	sepallength	88	70	18	86.67%
(24)	petalwidth	petallength	sepallength	sepalwidth	81	62	19	93.33%

Tabela 7.5 Resultados obtidos nos experimentos utilizando 5-validação cruzada explorando todas as possíveis ordenações dos atributos do conjunto de instâncias de dados Iris. Na tabela, seq: número de identificação da sequência; ordem dos atributos: ordem na qual os atributos foram considerados; #trd: total de regras de decisão encontradas; #rdi: número de regras de decisão inconclusivas e inválidas; #rdc: número de regras de decisão consistentes e %acu: acurácia (precisão) do classificador no conjunto de teste. A linha em negrito representa aquela que obteve o melhor resultado.

seq	**ordem dos atributos**				**#trd**	**#rdi**	**#rdc**	**%acu**
(1)	sepallength	sepalwidth	petallength	petalwidth	23	5	18	93.33%
(2)	sepallength	sepalwidth	petalwidth	petallength	23	5	18	92.67%
(3)	sepallength	petallength	petalwidth	sepalwidth	23	5	18	92.67%
(4)	sepallength	petallength	sepalwidth	petalwidth	23	5	18	93.33%
(5)	sepallength	petalwidth	petallength	sepalwidth	23	5	18	89.33%
(6)	sepallength	petalwidth	sepalwidth	petallength	23	5	18	92.67%
(7)	**sepalwidth**	**sepallength**	**petallength**	**petalwidth**	**23**	**4**	**19**	**94.00%**
(8)	sepalwidth	sepallength	petalwidth	petallength	23	4	19	93.33%
(9)	sepalwidth	petallength	sepallength	petalwidth	23	5	18	92.00%
(10)	sepalwidth	petallength	petalwidth	sepallength	23	5	18	90.67%
(11)	sepalwidth	petalwidth	petallength	sepallength	23	5	18	89.33%
(12)	sepalwidth	petalwidth	sepallength	petallength	23	5	18	92.67%
(13)	petallength	sepallength	sepalwidth	petalwidth	23	5	18	93.33%
(14)	petallength	sepallength	petalwidth	sepalwidth	23	5	18	92.67%
(15)	petallength	sepalwidth	petalwidth	sepallength	23	5	18	90.67%
(16)	petallength	sepalwidth	sepallength	petalwidth	23	5	18	93.33%
(17)	petallength	petalwidth	sepalwidth	sepallength	23	5	18	90.67%
(18)	petallength	petalwidth	sepallength	sepalwidth	23	5	18	92.67%
(19)	petalwidth	sepallength	petallength	sepalwidth	23	5	18	92.67%
(20)	petalwidth	sepallength	sepalwidth	petallength	23	5	18	92.67%
(21)	petalwidth	sepalwidth	petallength	sepallength	23	5	18	90.67%
(22)	petalwidth	sepalwidth	sepallength	petallength	23	5	18	92.67%
(23)	petalwidth	petallength	sepalwidth	sepallength	23	5	18	90.67%
(24)	petalwidth	petallength	sepallength	sepalwidth	23	5	18	92.67%

Nota-se que, nas duas tabelas, Tabela 7.4 e Tabela 7.5, a sequência de atributos (7) se destacou. Isso mostra que, em média, a sequência (7) estará obtendo as melhores acurácias ao se utilizar o conjunto de instâncias de dados Iris.

As acurácias apresentadas na Tabela 7.5 são, no geral, inferiores àquelas apresentadas na Tabela 7.4, pois refletem um resultado médio obtido através da utilização da técnica de 5-validação cruzada. Entretanto, por refletirem um resultado médio, são consideradas mais confiáveis.

7.5 Investigando uma Ordenação de Atributos Subsidiada por Ganho de Informação

O segundo estudo de caso que foi realizado investigou a possibilidade do valor do correspondente ganho de informação associado a cada atributo, poder (ou não)

colaborar, quando da indução do GF e da subsequente extração do classificador. Esse segundo estudo de caso utilizou-se, também, do conjunto de instâncias de dados Iris ([Dua & Karra Taniskidou 2017]).

Foram calculadas as entropias de classe e, também, de cada um dos atributos do conjunto Iris e, a partir dessas entropias, foram calculados os ganhos de informação. Os cálculos relacionados à entropia e ganho de informação podem ser conferidos, com detalhes, no Capítulo 3.

A entropia relacionada a à classe do conjunto Iris é dada por $\text{Ent(I)} \cong 1{,}59$.

A entropia e o ganho de informação de cada um dos atributos do conjunto Iris, são exibidos a seguir:

(1) Atributo *sepallength*:

$$\text{Ent}(sepallength) \cong 0{,}91$$

$$\text{Ganho}(sepallength) \cong \text{Ent(I)} - \text{Ent}(sepallength) \cong 1{,}59 - 0{,}91 \cong 0{,}68$$

(2) Atributo *sepalwidth*:

$$\text{Ent}(sepalwidth) \cong 1{,}21$$

$$\text{Ganho}(sepalwidth) \cong \text{Ent(I)} - \text{Ent}(sepalwidth) = 1{,}59 - 1{,}21 \cong 0{,}38$$

(3) Atributo *petallength*:

$$\text{Ent}(petallength) \cong 0{,}23$$

$$\text{Ganho}(petallength) \cong \text{Ent(I)} - \text{Ent}(petallength) \cong 1{,}59 - 0{,}23 \cong 1{,}36$$

(4) Atributo *petalwidth*:

$$\text{Ent}(petalwidth) \cong 0{,}21$$

$$\text{Ganho}(petalwidth) = \text{Ent(I)} - \text{Ent}(petalwidth) = 1{,}59 - 0{,}21 \cong 1{,}38$$

Com os valores referentes aos ganhos de informação dos atributos do conjunto de instâncias de dados Iris devidamente calculados, 2 experimentos foram realizados. O primeiro deles utilizando uma ordenação de atributos em ordem decrescente de seus respectivos valores de ganho de informação, e o segundo, uma ordenação em ordem crescente.

Primeiro Experimento – Atributos Ordenados por Ordem Decrescente de Ganho de Informação

A sequência de atributos, definida inicialmente, teve como base os valores do ganho de informação em ordem decrescente, de forma que os atributos com os maiores ganhos de informação (*i.e.*, aqueles com menor entropia) fossem os primeiros, como pode ser observado na Tabela 7.6. Os dados relativos ao experimento realizado são referentes àqueles obtidos com o uso de um processo de 5-validação cruzada e estão mostrados na Tabela 7.7.

Tabela 7.6 Sequência de atributos definida pelo ganho de informação em ordem decrescente.

atributo	ganho de informação
petalwidth	1.38
petallength	1.36
sepallength	0.68
sepalwidth	0.38

Tabela 7.7 Resultados obtidos, em média, nos experimentos utilizando o processo de 5-validação cruzada usando a sequência de atributos petalwidth, petallength, sepallength e sepalwidth, definida pelo ganho de informação em ordem decrescente.

Sequência de atributos	petalwidth, petallength, sepallength, sepalwidth
Total de regras de decisão	23
Regras de decisão inconclusivas e inválidas	5
Regras de decisão consistentes	18
Acurácia obtida	82.76%

Observa-se, na Tabela 7.7, que a acurácia obtida pela sequência de atributos definida pelo ganho de informação em ordem decrescente foi de, em média, 82.76%, o que representa um valor insatisfatório se comparado aos melhores resultados obtidos por algumas sequências nos experimentos anteriores (Tabela 7.4 e Tabela 7.5). Portanto, para esse estudo de caso em questão, utilizar uma sequência de atributos com base na ordem decrescente do ganho de informação, não contribuiu com a melhora da acurácia do classificador extraído do GF induzido.

Segundo Experimento – Atributos Ordenados por Ordem Crescente de Ganho de Informação

Uma nova sequência de atributos, com base na ordem crescente de ganho de informação, foi definida para o segundo experimento. Assim, atributos com menor

ganho de informação compareceram antes dos atributos com maior ganho de informação, na descrição das instâncias de treinamento e, consequentemente, nas camadas iniciais do GF a ser induzido. A sequência de atributos baseada na ordem crescente de ganho de informação é exibida na Tabela 7.8.

Tabela 7.8 Sequência de atributos definida pelo ganho de informação em ordem crescente.

atributo	ganho de informação
sepalwidth	0.38
sepallength	0.68
petallength	1.36
petalwidth	1.38

De forma similar, o segundo experimento utilizou o processo de 5-validação cruzada e os seus resultados estão mostrados na Tabela 7.9.

Tabela 7.9 Resultados obtidos, em média, nos experimentos utilizando o processo de 5-validação cruzada usando a sequência de atributos sepalwidth, sepallength, petallength e petalwidth definida pelo ganho de informação em ordem crescente.

Sequência de atributos	sepalwidth, sepallength, petallength, petalwidth
Total de regras de decisão	23
Regras de decisão inconclusivas	5
Regras de decisão consistentes	18
Acurácia obtida	93.33%

A Tabela 7.9 mostra uma acurácia média de 93.33% obtida pelo processo de 5-validação cruzada e ordenação de atributos com base no ganho de informação em ordem crescente. Tal acurácia ficou acima daquela obtida no experimento mostrado na Tabela 7.7, de 82.76%, o qual se utilizou da sequência de atributos com base no ganho de informação em ordem decrescente. Tal fato evidencia que a ordenação dos atributos em ordem crescente de ganho de informação produziu, em média, resultados um pouco melhores que a ordenação de atributos baseada na ordem decrescente de ganho de informação para este conjunto de dados.

7.6 Verificação de Sequências de Atributos em Busca Resultados Satisfatórios

Essa seção apresenta uma avaliação realizada sobre diversas sequências de atributos dos conjuntos de instâncias de dados a fim de identificar aquelas que produzem resultados

satisfatórios (*i.e.* GFs, e consequentemente, classificadores). Os conjuntos de instâncias de dados utilizados nos experimentos dessa seção são aqueles utilizados nos experimentos do Capítulo 6.

Como visto anteriormente, a partir de um conjunto de instâncias de dados que possui n atributos é possível obter n! possíveis sequências de atributos distintas. Porém, como n! pode ser um número alto, o que implicaria alto custo computacional, os experimentos realizados consideraram, no máximo, 50 sequências diferentes de atributos por conjunto de instâncias de dados, selecionadas de forma aleatória.

Para realização da verificação de sequências de atributos em busca de resultados satisfatórios, o seguinte procedimento foi executado para cada conjunto de instâncias de dados apresentados na Tabela 7.X:

(1) Foram identificadas e armazenadas as n! possibilidades de sequências de atributos para o conjunto de instâncias de dados;
(2) Foram selecionadas e armazenadas, de forma aleatória, até 50 sequências de atributos identificadas em (1);
(3) Para cada uma das sequências de atributos armazenadas em (2), foi executado o processo de 5-validação cruzada;
(4) A melhor acurácia obtida em (3) foi armazenada, assim como a sequência de atributos que proporcionou tal acurácia, e;
(5) Ao final, foi disponibilizada a sequência de atributos que produziu a melhor acurácia no processo de 5-validação cruzada.

As sequências de atributos que produziram os resultados mais satisfatórios dentre todas testadas utilizando o procedimento descrito acima, assim como as acurácias obtidas com tais sequências, são apresentadas na Tabela 7.10. Na mesma tabela, são exibidos os resultados que foram obtidos utilizando as sequências originais de atributos, para cada conjunto de instâncias de dados. Esses últimos resultados foram retirados da Tabela 6.8, do Capítulo 6, para comparação de desempenho.

Tabela 7.10 Acurácias obtidas nos experimentos utilizando as sequências de atributos que produziram os resultados mais satisfatórios dentre as testadas, para cada conjunto de instâncias de dados. Na tabela, CI: conjunto de instâncias de dados; SRS: sequência de atributo que produziu o resultados mais satisfatório para o respectivo conjunto de dados; %acuSRS: acurácia obtida ao se utilizar a sequência de atributos apresentada em SRS, e; %acuORG: acurácia obtida ao se utilizar a sequência original de atributos presente nos conjuntos de instancias de dados, para efeitos de comparação de desempenho. Os melhores resultados estão destacados em negrito para melhor compreensão dos dados.

CI	SRS	%acuSRS	%acuORG
(Ru)	x, y	**97.33%**	**97.33%**
(ML)	x, y	**93.70%**	91.50%
(Sp)	y, x	99.67%	**98.33%**
(MC)	x, y	**99.75%**	**99.75%**
(LS)	a1, a0	**99.56%**	88.67%
(Ir)	sepalwidth, sepallength, petallength, petalwidth	**94.00%**	93.33%
(Fo)	at0, at3, at1, at5, at4, at2	88.51%	**94.44%**
(Ag)	y, x	**99.75%**	**99.75%**
(Fl)	y, x	**95.00%**	85.42%
(Se)	at2, at5, at1, at0, at3, at6, at4	**93.81%**	90.48%
(Ec)	alm2, aac, alm1, gvh, chg, mcg, lip	82.14%	**82.35%**
(BT)	RecencyInMonths, FrequencyTimes, Monetary, TimeInMonths	75.53%	**76.67%**
(HS)	ageWhenOperated, operatedIn, posAxiNodes	**75.53%**	72.58%
(VC)	pelvic_incidence, pelvic_radius, lumbar_lordosis_angle, degree_spondylolisthesis, pelvic_tilt, sacral_slope	80.65%	**83.87%**
(UK)	STG, PEG, STR, LPR, SCG	84.86%	**91.36%**
(CS)	Heart, Age, Delivery, Delivery, Blood	73.75%	**75.00%**
(Gl)	Ca, K, Fe, ID, Mg, RI, Ba, Si, Al, Na	**97.67%**	**97.67%**
(Wi)	Alcohol, Malic_acid, Ash, Alcalinity_of_ash, Magnesium, Total_phenols, Flavanoids, Nonflavanoid_phenols, Proanthocyanins, Color_intensity, Hue, OD280_OD315, Prolin	97.19%	**97.22%**
(Cr)	sex, type, time, nWarts, age, area	82.22%	**83.33%**

Dos 19 experimentos realizados, 4 deles produziram, ao se utilizar a sequência de atributos que antes produziu os resultados mais satisfatórios dentre as sequências de atributos testadas, resultados idênticos aqueles obtidos ao se utilizar a sequência original de atributos. Dos outros 15 experimentos restantes, as sequências originais de atributos produziram resultados melhores em 9 ocasiões. Tais resultados mostram que, as sequências de atributos presentes nos conjuntos de instâncias de dados podem impactar os resultados, porém, não foi possível identificar uma regra, um padrão ou uma técnica, que leve sempre aos melhores resultados.

7.7 Considerações Finais

Este capítulo apresentou uma investigação empírica sobre o impacto da ordem dos atributos sobre a indução de GFs Estendidos e sobre a extração de classificadores a partir dos mesmos.

Como visto, diferentes sequências de atributos geram diferentes grafos de fluxo, dos quais podem ser extraídos diferentes classificadores. Com base nessa afirmação, diferentes sequências de atributos, de diferentes conjuntos de instâncias de dados, foram investigadas, a fim de verificar se existe um padrão a ser utilizado na ordenação dos atributos de forma a se obter melhores GFs e classificadores. Porém, não foi possível identificar um padrão através das técnicas e opções utilizadas. Nem mesmo experimentos utilizando sequências de atributos que, em experimentos anteriores mostraram-se satisfatórias, pois obtiveram boas acurácias, não conseguiram, na maioria das vezes, superar as acurácias obtidas pelas sequências originais de atributos, nos conjuntos de instâncias de dados.

Capítulo 8
Conclusões

Este capítulo apresenta as conclusões relacionadas às teorias e ao estudo conduzido sobre Grafos de Fluxo. As contribuições do estudo são também apresentadas, assim como, opções para trabalhos futuros.

8.1 Investigação Realizadas

O estudo conduzido buscou investigar o formalismo relacionado aos Grafos de Fluxo, proposto por Pawlak. Entre os objetivos da investigação, estavam o de se verificar os motivos pelos quais os GFs não se estabeleceram no ambiente de AM, verificar inconsistências e redundâncias no formalismo.

Foi constatado, durante a investigação, que três motivos principais que podem ter contribuído para que os GFs não fossem adotados como uma opção viável para extração de classificadores em AM, sendo eles: (1) ineficiência ao lidar com conjuntos de instâncias de dados que contém atributos contínuos, (2) formalismo e técnicas carentes de clareza nas definições e, (3) custo computacional exigido para extração de classificadores a partir de GFs.

O problema em lidar com atributos contínuos (Capítulo 6) foi resolvido através da investigação de opções que resultaram em uma proposta para extensão dos GFs originais, chamada de Grafos de Fluxo Estendidos (GFEs). Tal proposta, tem como base, realizar a discretização de atributos contínuos presentes nos conjuntos de instâncias de dados, antes da indução do GF. Por padrão, foi decidido que o método a ser utilizado para a discretização é o método de particionamento do escopo do atributo por Entropia e MDLP, proposto em [Fayyad & Irani 1993]. Outros dois métodos, por Amplitudes Iguais e por Iguais Frequências, continuam presentes no sistema computacional desenvolvido e podem ser utilizados para discretização dos atributos, como opção do usuário. Ao se discretizar os atributos contínuos presentes em conjuntos de instâncias de

dados, colabora-se com a indução de GFs menores (em número de nós e arcos), mais simples e que sumarizam melhor a distribuição de fluxo presente entre os nós e arcos do grafo. Tal ação permite, ainda, a extração de classificadores (algoritmos de decisão) que generalizam instâncias de dados, fato que aumenta a chance de acerto em testes de classificação.

O método original para extração de classificadores, proposto por Pawlak, foi estudado, avaliado e experimentado. Em seguida, um novo método para extração de classificadores, que lida com regras de decisão inválidas e regras de decisão inconclusivas, foi proposto, descrito e implementado junto ao sistema EFLOWG. Seus resultados foram apresentados e discutidos.

Problemas encontrados no formalismo relacionado estritamente aos GFs, foram abordados no Capítulo 4. Através de observações presentes no documento, tais problemas foram questionados e, na maioria das vezes, foram propostas correções e/ou melhorias, com vistas a colaborar positivamente com o formalismo em questão.

O problema relacionado com o alto custo computacional, exigido para indução de GFs e extração de classificadores, foi identificado durante testes utilizando o sistema EFLOWG. A abordagem utilizada por Pawlak para extração de classificadores era totalmente inviável, de forma que, conjuntos de instâncias de dados com 6 ou mais atributos constituíam, na maioria das vezes, um grande desafio ao sistema computacional, dadas as inúmeras possibilidades de caminhos completos presentes nos respectivos GFs. O problema foi amenizado, utilizando técnicas mais simples que armazenam somente regras de decisão válidas, o que diminuiu a necessidade de armazenamento temporário de dados em memória, e aumentou o desempenho do sistema e dos processos de indução de GFs, extração de classificadores e testes de classificação, como um todo. Porém, não foi completamente resolvido: conjuntos de instâncias de dados que possuem, por exemplo, mais de 12 ou 13 atributos, continuam sendo um obstáculos (consumo de processamento e memória) para indução de GFEs e, principalmente, extração de classificadores destes GFEs. Como o sistema e os métodos não foram testados em computadores de com maior performance, não é possível afirmar se tal problema está relacionado única e exclusivamente ao *hardware*. Também não foi realizado um estudo sobre a complexidade de tempo dos algoritmos envolvidos no

processo de indução de GFEs, extração de classificadores e testes de classificação, outro fato que não permite que seja afirmado que o problema do custo computacional esteja ligado, novamente, ao *hardware*, *i.e.* pode haver problemas na forma em como foi realizada a implementação dos algoritmos em si.

Como conclusão final, é possível afirmar que, os GFs são interessantes, principalmente, pelo fato ligado à sumarização da distribuição do fluxo dos dados. Neste caso, a representação gráfica fornecida pelos GFs possibilita uma leitura simples e prática da ligação que ocorre entre os nós de suas diferentes camadas, o que, a nível de análise, pode ser interessante. Em relação ao custo computacional, o processo geral de indução de GFs (e GFEs) e extração de classificadores deixou a desejar, se comparado à outros formalismos, pois exige bastante verificações e descobertas de caminhos completos nos GFs, o que, apesar de ser uma tarefa trivial, pode ter longa duração dependendo do *hardware* utilizado e do conjunto de instâncias de dados em questão. As acurácias obtidas nos processos de classificação foram promissoras, porém, como apresentado nos capítulos anteriores, não chegaram a ser competitivas em relação àquelas obtidas por outros classificadores, mesmo após melhorias terem sido propostas e implementadas via GFEs.

Parte da pesquisa conduzida, sobre os GFEs, foi apresentada (ver Anexo I) em uma conferência internacional sobre sistemas inteligentes híbridos, HIS 2018 (Hybrid Intelligent Systems), Quali B2. A conferência aconteceu na cidade do Porto, em Portugal, entre os dias 13 e 15 de dezembro de 2018. O artigo científico, relacionado à HIS 2018, será publicado no formato *Spring-Verlag*.

8.2 Trabalhos Futuros

Algumas opções possíveis de investigação relacionadas aos GFs (e GFEs), para trabalhos futuros, foram identificadas. Entre elas, destacam-se:

(1) Investigação sobre a possibilidade de indução de GFs de forma que estes pudessem representar a distribuição de fluxo existente entre as instâncias que serviram de base para indução do GF, e não apenas a distribuição do fluxo entre nós de diferentes camadas. Se fosse possível, por exemplo, nós seriam

compartilhados somente entre caminhos completos que pudessem, de fato, classificar instâncias de dados, e tal ação poderia gerar GFs mais reduzidos e fiéis às instâncias de dados, o que poderia diminuir o custo computacional e, talvez, poderia impactar nos resultados de testes de classificação de instâncias de dados;

(2) Estudo e investigação mais aprofundados sobre a função e utilização mais adequada dos fatores associados aos arcos em busca de regras de decisão mais robustas, pois na pesquisa ficou claro que, apesar de serem calculados os quatro fatores, o único realmente utilizado foi o fator de força associado aos arcos e regras de decisão;

(3) Pesquisa e investigação em busca de formas automáticas para diminuição da estrutura de GFs induzidos, *i.e.*, realizar uma redução do GF com base em caminhos bem definidos e nos fatores associados aos elementos dos GFs. Se possível, o processo de extração de classificadores poderia ser ainda mais simples, assim como as estruturas dos próprios classificadores;

(4) Redução do número de regras de decisão dos classificadores extraídos de GFs com pouco impacto negativo no poder de classificação do mesmo. A justificativa é a de que, dependendo do conjunto de instâncias de dados, algumas regras de decisão podem classificar unicamente uma instância entre centenas ou milhares, o que levaria elas a terem baixo valor associado ao fator de força. Talvez, seja possível encontrar formas de eliminar essas regras de decisão que possuem valores baixos associados ao fator de força, sem impactar radicalmente o desempenho geral dos testes de classificação;

(5) Investigação da utilização do processamento paralelo para indução de GFs que é possível, de acordo com o autor, porém não foi aplicado neste trabalho, com vistas às melhoras de desempenho do processo em geral;

(6) Utilização de algoritmos de busca para reduzir mais o conjunto de regras de decisão ou selecionar entre as regras inválidas, caso existam, e;

(7) Comparar tamanhos de diferentes modelos obtidos, com objetivo de manter aquele que produza bons resultados e possua tamanho aceitável (*i.e.* viável de utilização na prática).

Referências

[Acion *et al.* 2017] Acion, L., Kelmansky, D., van der Laan, M., Sahker, E., Jones, D., Arndt, S. (2017) Use of a Machine Learning Framework to Predict Substance Use Disorder Treatment Success. PLoS ONE 12(4): e0175383.

[Aeberhard *et al.* 1992] Aeberhard, S., Coomans, D., de Vel, O. (1992) Comparison of Classifiers in High Dimensional Settings. Tech. Rep. no. 92-02. Dept. of Computer Science and Dept. of Mathematics and Statistics, James Cook University of North Queensland.

[Agrawal *et al.* 1994] Agrawal, R., Imielinski, T., Swami, A. (1994) Mining Association Rules Between Sets of Items in Large Databases. In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data (SIGMOD '93), pp. 207-216.

[Agrawal & Srikant 1994] Agrawal, R., Srikant, R. (1994) Fast Algorithms for Mining Association Rules in Large Databases. In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB ’94), pp. 487-499.

[Allen 1969] Allen, F. E. (1969) Program optimization. In Annual Review in Automatic Programming, Pergamon Press, v. 5, pp. 239-307.

[Allen 1970] Allen, F. E. (1970) Control Flow Analysis. SIGPLAN Notices, v. 5, no. 7, pp. 1-19.

[Allen 1972] Allen, F. E. (1972) A Basis for Program Optimization. In Proceedings of IFIP Congress, pp. 385-390.

[Allen 1974] Allen, F. E. (1974) Interprocedural Data Flow Analysis. In Proceedings of IFIP Congress, pp. 398-402.

[Alpaydin 2010] Alpaydin, E. (2010) Introduction to Machine Learning. England: The MIT Press, 2nd Edition.

[An & Cercone 1999] An, A., Cercone, N. (1999) Discretization of Continuous Attributes for Learning Classification Rules. In Proceedings of the 3rd Pacific-Asia

Conference on Knowledge Discovery and Data Mining (PAKDD 1999), pp. 509-514).

[Arciszewski *et al.* 1995] Arciszewski, T., Michalski, R. S., Dybala, T. (1995) STAR Methodology-based Learning About Construction Accidents and their Prevention. Automation in Construction, v. 4, pp. 75–85.

[ARFF 2018] Attribute-Relation File Format (ARFF). Disponível em: <https://www.cs.waikato.ac.nz/ml/weka/arff.html>. Acesso em: 29. Nov. 2018.

[Bandyopadhyay & Maulik 2002] Bandyopadhyay, S., Maulik, U. (2002) Genetic Clustering for Automatic Evolution of Clusters and Application to Image Classification. In Pattern Recognition, v. 35, pp. 1197−1208.

[Bishop 2006] Bishop, C. M. (2006) Pattern Recognition and Machine Learning. Springer Verlag.

[Booth 1992] Booth, Paul. (1992) An Introduction to Human-Computer Interaction. LEA Ltd., 3th ed., 268p.

[Bratko 2008] Bratko, I. (2008) An Assessment of Machine Learning Methods for Robotic Discovery. Journal of Computing and Information Technology, v. 16, pp. 247-254.

[Brodley & Friedl 1996] Brodley, C. E., Friedl, M. A. (1996) Improving Automated Land Cover Mapping by Identifying and Eliminating Mislabeled Observations from Training Data. Published in 1996 International Geoscience and Remote Sensing Symposium (IGARSS '96), pp. 1379-1381, v. 2.

[BusinessDictionary 2018] BusinessDictionary. Disponível em: <http://www.businessdictionary.com/definition/quartile.html>. Acesso em: 05. Jun. 2018.

[Catllet 1991] Catlett J. (1991) On Changing Continuous Attributes Into Ordered Discrete Attributes. Machine Learning - EWSL-91. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence), v. 482.

[Chan & Tsumoto 2007] Chan, C., Tsumoto, S. (2007) On Learning Decision Rules From Flow Graphs. NAFIPS 2007 - 2007 Annual Meeting of the North American Fuzzy Information Processing Society, pp. 655-658.

[Charytanowicz *et al.* 2010] Charytanowicz, M., Niewczas, J., Kulczycki, P., Kowalski, P.A., Łukasik, S., Żak S. (2010) Complete Gradient Clustering Algorithm for Features Analysis of X-Ray Images. In: Piętka E., Kawa J. (eds) Information Technologies in Biomedicine. Advances in Intelligent and Soft Computing, v. 69.

[Chen 1995] Chen, H. (1995) Machine Learning for Information Retrieval: Neural Networks, Symbolic Learning, and Genetic Algorithms. Journal of the American Society for Information Science, v. 46, pp. 194-216.

[Chernoff 1972] Chernoff, H. (1972) Metric Considerations in Cluster Analysis. In Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability: Theory of Statistics, v. 1, pp. 621-629.

[Choudhary *et. al* 2009] Choudhary, A. K., Harding, J. A., Tiwari, M. K. (2009) Data Mining in Manufacturing: a Review Based on the Kind of Knowledge, v. 20, p 501.

[Dash *et al.* 2011] Dash, R., Paramguru, R. L., Dash, Rasmita. (2011) Comparative Analysis of Supervised and Unsupervised Discretization Techniques. International Journal of Advances in Science and Technology, v. 2, p. 28.

[Daza & Acuna 2007] Daza, L., Acuna, E. (2007) An Algorithm for Detecting Noise on Supervised Classification. In Proceedings of World Congress on Engineering and Computer Science 2007 (WCECS 2007).

[de Brujine 2016] de Brujine, M. (2016). Machine Learning Approaches in Medical Image Analysis: From Detection to Diagnosis, v. 33, pp. 94-97.

[de la Iglesia *et al.* 2014] de la Iglesia D., García-Remesal, M., Anguita, A., Muñoz-Mármol, M., Kulikowski, C., Maojo, V. (2014) A Machine Learning Approach to Identify Clinical Trials Involving Nanodrugs and Nanodevices from ClinicalTrials.gov. PLoS One. 2014 Oct 27;9(10): e110331.

[Deo 2015] Deo, R. C. (2015) Machine Learning on Medicine. Circulation, v. 132, pp. 1920-1930.

[Diaz et al. 2010] Diaz, A. A., Tomba, E., Lennarson, R., Richard, R., Bagajewicz, M. J., Harrison, R. G. (2010) Prediction of Protein Solubility in Escherichia Coli using Logistic Regression. Biotechnol Bioeng. 2010 Feb 1;105(2):374-83.

[Domingues & Uchôa 2004] Domingues, M. A., UCHÔA, J. Q. (2004) Representação de Conhecimento Usando Teoria de Conjuntos Aproximados. INFOCOMP, v. 2, pp. 53-57.

[Dougherty *et al.* 1995] Dougherty, J., Kohavi, R., Sahami, M. (1995) Supervised and Unsupervised Discretization of Continuous Features, In Machine Learning. In Proceedings of the Twelfth International Conference on Machine Learning, pp. 194-202.

[Dua & Karra Taniskidou 2017] Dua, D., Karra Taniskidou, E. (2017) UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[Duda *et al.* 2001] Duda, R. O., Hart, P. F., Stork, D. G. (2001) Pattern Classification.

[Evett & Spiehler 1987] Evett, I. W., Spiehler, E. J. (1987) Rule Induction in Forensic Science. KBS in Government, pp. 107-118.

[Fard *et al.* 2016] Fard, M. J., Ameri, S., Chinnam, R. B., Pandya, A. K., Klein, M. D., Ellis, R. D. (2016) Machine Learning Approach for Skill Evaluation in Robotic-Assisted Surgery. Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2016, 19-21 October, 2016, San Francisco, USA.

[Fayyad & Irani 1993] Fayyad, U. M., Irani, K. B. (1993) Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. IJCAI, pp. 1022-1029.

[Fayyad *et al.* 1996] Fayyad, U., Piatetsky-Shapiro, G., Smyth, P. (1996) The KDD process for extracting useful knowledge from volumes of data. Communications of the ACM, v. 39, pp. 27-34.

[Ford & Fulkerson 1956] Ford, L. R., Fulkerson, D. R. (1956) Maximal Flow Through a Network. Classic Papers in Combinatorics, pp. 243-248.

[Fosdick & Osterweil 1976] Fosdick, L. D., Osterweil, L. J. (1976) Data Flow Analysis in Software Reliability. Computing Surveys, v. 8, pp. 305-330.

[Fu & Medico 2007] Fu, L., Medico, E. (2007) FLAME, a Novel Fuzzy Clustering Method for the Analysis of DNA Microarray Data. BMC Bioinformatics, p. 3.

[Gamberger *et al.* 2000] Gamberger, D., N. Lavrac, and S. Dzeroski (2000) Noise Detection and Elimination in Data Preprocessing: Experiments in Medical Domains. Applied Artificial Intelligence, pp. 205–223.

[Garcia *et al.* 2013] Garcia, S., Luengo, J., Sáez, J. A., López, V., Herrera, F. (2013) A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning. IEEE Transactions on Knowledge and data Engineering, v. 25, pp. 734-750.

[Garcia *et al.* 2013a] Garcia, L. P. F., de Carvalho, A. C. P. L. F., Lorena A. C. (2013) Noisy Data Set Identification. In Hybrid Artificial Intelligent Systems (HAIS 2013), pp. 629-638.

[Garcia 2016] Garcia, L. P. F. (2016) Noise Detection in Classification Problems. Tese de Doutorado em Ciências de Computação e Matemática Computacional - Instituto de Ciências Matemáticas e de Computação. University of São Paulo, São Carlos. Acesso em: 2018-01-09.

[Giger *et al.* 2008] Giger, M. L., Chan, H. P., Boone, J. (2008) Anniversary Paper: History and status of CAD and quantitative image analysis: The role of Medical Physics and AAPM. Med. Phys. 35 (12), 5799-5820.

[Gionis *et al.* 2007] Gionis, A., Mannila, H., Tsaparas, P. (2007) Clustering Aggregation. ACM Transactions on Knowledge Discovery from Data (TKDD), pp. 1-30.

[GMUM 2018] Mouse-like Dataset. Disponível em: <http://r.gmum.net/samples/cec.basic.html>. Acesso em: 6. Jan. 2018.

[Habibi *et al.* 2014] Habibi, N., Mohd Hashim, S. Z., Norouzi, A., and Samian, M. R. (2014) A Review of Machine Learning Methods to Predict the Solubility of Overexpressed Recombinant Proteins in Escherichia Coli. BMC Bioinformatics, pp. 1-16.

[Haberman 1976] Haberman, S. J. (1976) Generalized Residuals for Log-Linear Models. In Proceedings of the 9th International Biometrics Conference, pp. 104-122.

[Handl & Knowles 2004] Handl, J., Knowles, J. (2004) Multiobjective Clustering with Automatic Determination of the Number of Clusters, UMIST, Manchester, Tech.

[Hassanien *et. al* 2010] Hassanien A. E., Schaefer G., Darwish A. (2010) Computational Intelligence in Speech and Audio Processing: Recent Advances. In: Gao XZ., Gaspar-Cunha A., Köppen M., Schaefer G., Wang J. (eds) Soft Computing in Industrial Applications. Advances in Intelligent and Soft Computing, v. 75.

[Hattori & Sáez 2013] Hattori, G., Sáez, A. (2013) Damage Identification in Multifield Materials using Neural Networks. Inverse Problems In Science & Engineering, pp. 929-944.

[Heinson *et al.* 2017] Heinson, AI., Gunawardana, Y., Moesker, B., hume, C. C., Vataga. E., Hall, Y, Stylianou, E., McShane, H., Williams, A., Niranjan, M., Woelk, C. H. (2017) Enhancing the Biological Relevance of Machine Learning Classifiers for Reverse Vaccinology. Int J Mol Sci. 2017.

[Hodge & Austin 2004] Hodge, V. J. and Austin, J. (2004) A Survey of Outlier Detection Methodologies. Artificial Intelligence Review, v. 22, pp. 85-126.

[Horton & Nakai 1996] Horton, P., Kenta, N. (1996) A Probabilistic Classification System for Predicting the Cellular Localization Sites of Proteins. In Proceedings of the Fourth International Conference on Intelligent Systems. Intelligent Systems in Molecular Biology, pp. 109-115.

[Idicula-Thomas *et al.* 2006] Idicula-Thomas, S., Kulkarni, A. J., Kulkarni, B. D., Jayaraman, V. K., Balaji, P. V. (2006). A Support Vector Machine-based Method for Predicting the Propensity of a Protein to be Soluble or to Form Inclusion Body on Overexpression in Escherichia Coli. Bioinformatics, 22(3), pp. 278–284.

[InfoEscola 2018] Permutação. Disponível em: <https://www.infoescola.com/matematica/permutacao/>. Acesso em: 04. Out. 2018.

[JAVA 2018] Java. Disponível em: <https://www.java.com/pt_BR/>. Acesso em: 29. Nov. 2018.

[John 1995] John,G.H. (1995). Robust Decision Trees: Removing Outliers from Databases. In Proceedings of the First International Conference on Knowledge Discovery and Data Mining, pp. 174–179.

[Kalapanidas *et al.* 2003] Kalapanidas, E., Avouris, N., Craciun, M., Neagu, D. (2003) Machine Learning Algorithms: a Study on Noise Sensitivity. First Balkan Conference in Informatics, pp. 356-365.

[Kahraman *et al.* 2013] H. T. Kahraman, Sagiroglu, S., Colak, I. (2013) Developing Intuitive Knowledge Classifier and Modeling of Users' Domain Dependent Data in Web, Knowledge Based Systems, v. 37, pp. 283-295.

[Khozeimeh et al. 2017] Khozeimeh, F., Alizadehsani, R., Roshanzamir, M., Khosravi, A., Layegh, P., Nahavandi, S. (2017) An Expert System for Selecting Wart Treatment Method. Computers in Biology and Medicine, v. 81, pp. 167-175.

[Klir & Yuan 1995] Klir, G. J., Yuan, B. (1995) Fuzzy Sets and Fuzzy Logic. Theory and Applications, Prentice Hall Inc., Upper Saddle River.

[Kluwer Academic Publishers 1998] Glossary of Terms, Special Issue on Applications of Machine Learning and the Knowledge Discovery Process, Machine Learning, 30, 271-274 (1998). Disponível em: <http://robotics.stanford.edu/~ronnyk/glossary.html>. Acesso em: 4. Ago. 2017.

[Kotsiantis & Kanellopoulos 2006] Kotsiantis, S., Kanellopoulos, D. (2006) Discretization Techniques: A Recent Survey. GESTS International Transactions on Computer Science and Engineering, v. 32, pp. 47-58.

[Kourou *et al.* 2015] Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V., Fotiadis, D. I. (2015) Machine Learning Applications in Cancer Prognosis and Prediction. Computational and Structural Biotechnology Journal 13, pp. 8–17.

[Kryszkiewicz 1998] Kryszkiewicz, M. (1998) Rough Set Approach to Incomplete Information Systems. Information Sciences, v. 112, pp. 39-49.

[Kumar *et al.* 2007] Kumar, P., Jayaraman, V. K., Kulkarni, B. D. (2007) Granular Support Vector Machine Based Method for Prediction of Solubility of Proteins on Overexpression in Escherichia Coli. In A. Ghosh, R. K. De, and S. K. Pal, editors, Pattern Recognition and Machine Intelligence, Lecture Notes in Computer Science, pp. 406–415.

[Kushik *et al.* 2016] Kushik, N., Yevtushenko, N., Evtushenko, T. (2016) Novel Machine Learning Technique for Predicting Teaching Strategy Effectiveness. International Journal of Information Management.

[Lakshminarayan *et al.* 1996] Lakshminarayan, K., Harp, S. A., Goldman, R., Samad, T. (1996) Imputation of Missing Data Using Machine Learning Techniques. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 140-145.

[Laurikkala *et al.* 2000] Laurikkala, J., Juhola, M., Kentala, E. (2000) Informal Identification of Outliers in Medical Data. Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology IDAMAP-2000 Berlin, 22 August.

[Lisowski & Czyzewski 2015] Lisowski, K., Czyzewski, A. (2015) Pawlak's Flow Graph Extensions for Video Surveillance Systems, In Proceedings of the Federated Conference on Computer Science and Information Systems, pp. 81–87.

[Little & Rubin 1987] Little, R. J. A., Rubin, D. B. (1987) Statistical Analysis with Missing Data.

[Liu et al. 2002] Liu, H., Hussain, F., Tan, C., Dash, M. (2002) Discretization: an Enabling Technique, Data Mining and Knowledge Discovery, v. 6, pp. 393-423.

[Liu *et al.* 2015] Liu, R., Kumar, A., Chen, Z., Agrawal, A., Sundararaghavan, V., Choudhary, A. (2015) A Predictive Machine Learning Approach for Microstructure Optimization and Materials Design. *Scientific Reports* [2045-2322], pp. 11551-11552.

[Lorena & Carvalho 2004] Lorena, Ana C., Carvalho, André C. P. L. F. de. (2004) Evaluation of Noise Reduction Techniques in the Splice Junction Recognition Problem. Genetics and Molecular Biology, v. 27, pp. 665-672.

[Maslove *et al.* 2013] Maslove, D. M., Podchiyska, T., Lowe, H. J. (2013) Discretization of Continuous Features in Clinical Datasets. J Am Med Inform Assoc., pp. 544-553.

[Mayhew 1992] Mayhew, D. J. (1992) Principles and Guidelines in Software User Interface Design. Englewood Cliffs (New Jersey), PTR Prentice Hall. 1992, 619p.

[Menden *et al.* 2013] Menden, M. P., Iorio, F., Garnett, M., McDermott, U., Benes, C. H., Ballester, P. J., Rodriguez, J. S. (2013) Machine Learning Prediction of Cancer Cell Sensitivity to Drugs Based on Genomic and Chemical Properties. PLoS ONE 8(4): e61318.

[Mitchell 1997] Mitchell, T. M. (1997) Machine Learning. McGraw-Hill Education.

[Monago & Kodratoff 1987] Monago, M. M., Kodratoff, Y. (1987) Noise and Knowledge Acquisition. In IJCAI-87, pp. 348–354.

[Morris *et al.* 2013] Morris, D., Fiebrink, R. (2013) Using Machine Learning to Support Pedagogy in the Arts. Personal and Ubiquitous Computing, v. 17, pp. 1631-1635.

[Muralidharan & Sugumaran 2013] V. Muralidharan, V. Sugumaran, Rough Set Based Rule Learning and Fuzzy Classification of Wavelet Features for Fault Diagnosis of Monoblock Centrifugal Pump. In Measurement, v. 46, pp. 3057-3063.

[Naouar *et al.* 2016] Naouar, B., Nesrine, G., Adil, S., Bouchaib, M. (2016) Proactive Intelligent Home System Using Contextual Information and Neural Network Approach. Int. Journal of Engineering Research and Applications, v. 6, pp. 68-73.

[NETBEANS 2018] NetBeans IDE. Disponível em <https://netbeans.org/>. Acesso em: 29. Nov. 2018.

[Nicoletti 1994] Nicoletti, M. D. C. (1994) Ampliando os limites do aprendizado indutivo de máquina através das abordagens construtiva e relacional. Tese de Doutorado, Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos.

[Nicoletti & Uchôa 1997] Nicoletti, M. C., Uchôa, J.Q. (1997) Conjuntos Aproximados Sob a Perspectiva de Função de Pertinência. 3º. Simpósio Brasileiro de Automação Inteligente, pp. 307-312.

[Nicoletti & Uchôa 1998] Nicoletti, M. D. C., Uchôa, J. Q. (1998) O Uso da Teoria de Conjuntos Aproximados na Determinação de Redutos de Conjuntos de Atributos. Relatório Técnico do DC 001/98, São Carlos, DC-UFSCar, 38p.

[Nicoletti & Hruschka Jr. 2006] Nicoletti, M. D. C., Hruschka Jr., E. R. (2006) Fundamentos da Teoria dos Grafos para Computação. EdUFSCar - Série Apontamentos, 224p.

[Nielsen 2003] Nielsen, J. (2003) Usability 101: Introduction to Usability. All Usability.

[Orhan *et al.* 2012] Orhan, U., Hekim, M., Ozer, M. (2012) Epileptic Seizure Detection Using Probability Distribution Based On Equal Frequency Discretization. Journal of Medical Systems, v, 36, pp. 2219-2224.

[Pal & Mitra 2002] Pal, S. K., Mitra, P. (2002) Multispectral Image Segmentation Using the Rough-set-initialized EM Algorithm. IEEE Geoscience and Remote Sensing Society, v. 40, pp. 2495-2501.

[Palaniammal & Chandrasekaran 2015] Palaniammal, V., Chandrasekaran, R. M. (2015) Breast Cancer Classification Using Various Machine Learning Techniques. In Advances in Natural and Applied Sciences, v. 9, p. 65+.

[Pankajakshan *et al.* 2017] Pankajakshan, P., Sanyal, S., de Noord, O. E., Bhattacharya, I., Bhattacharyya, A., Waghmare, U. (2017) Machine Learning and Statistical Analysis for Materials Science: Stability and Transferability of Fingerprint Descriptors and Chemical Insights. Chemistry of materials, v. 29, , pp. 4190-4201.

[Pawlak 1982] Pawlak, Z. (1982) Rough Sets. International Journal of Computer and Information Sciences, v. 11, pp. 341-356.

[Pawlak 1988] Pawlak, Z. (1988) Rough Sets and Information Systems. Podstawy Sterowania, v. 18, pp. 175-200.

[Pawlak *et al.* 1995] Pawlak, Z., Grzymala-Busse, J., Slowinski, R., Ziarko, W. (1995) Rough Sets. Communications of the ACM, v. 38, pp. 89-95.

[Pawlak 2001] Pawlak, Z. (2001) Data Analysis - the Rough Sets Perspective. in: J. Chojcan, J. Łęski (editors), Zbiory rozmyte i ich zastosowania, pp. 173-181.

[Pawlak 2002a] Pawlak, Z. (2002) Rough Sets and Intelligent Data Analysis. Information Sciences, v. 147, pp. 1-12.

[Pawlak 2003] Pawlak, Z. (2003) Flow Graphs and Decision Algorithms. International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing. Lecture Notes in Computer Science, v. 2639, pp. 1-10.

[Pawlak 2003a] Z. Pawlak (2003a) Decision Algorithms and Flow Graphs: a Rough Set Approach. Journal of Telecommunications and Information Technology 3, pp. 98-101.

[Pawlak 2003b] Z. Pawlak (2003) Probability, Truth and Flow Graph. In Electronic Notes in Theoretical Computer Science, v. 82, pp. 1-9.

[Pawlak 2004a] Pawlak, Z. (2004) Decision Rules and Flow Networks. European Journal of Operational Research, v. 152, pp. 184-190.

[Pawlak 2004b] Z. Pawlak (2004) Flow Graphs – A New Paradigm For Data Mining and Knowledge Discovery. JAIST Forum 2004 - Technology Creation Based on Knowledge Science: Theory and Practice, jointly with The 5th International Symposium on Knowledge and Systems Science (Proc. of the KSS2004), pp. 147-153.

[Pawlak 2004c] Z. Pawlak (2004) Data Analysis and Flow Graphs. Journal of Telecommunications and Information Technology 3, pp. 1-5.

[Pawlak 2005a] Pawlak, Z. (2005) Rough Sets and Flow Graphs. Slezak, D. *et al.* (Eds.). Lecture Notes in Artificial Intelligence, v. 3641, pp. 1-11.

[Pawlak 2005b] Pawlak, Z. (2005) Flow Graphs and Data Mining. J. F. Peters and A. Skowron (Eds.). Transactions on Rough Sets III, Lecture Notes in Computer Science, v. 3400, pp. 1-36.

[Pawlak & Skowron 2007] Pawlak, Z., Skowron, A. (2007) Rudiments of Rough Sets. Information Sciences, v. 177, pp. 3–27.

[Pawlak 2010] Pawlak, Z. (2010) Flow Graphs – a New Paradigm for Intelligent Data Analysis. Warsaw University of Technology Digital Library, pp. 1-28.

[Pechenizkiy 2005] Pechenizkiy, M. (2005) The Impact of Feature Extraction on the Performance of a Classifier: kNN, Naïve Bayes and C4.5. In: Kégl B., Lapalme G. (eds) Advances in Artificial Intelligence. AI 2005. Lecture Notes in Computer Science, v. 3501.

[Pechenizkiy *et al.* 2006] Pechenizkiy, M., Tsymbal, A., Puuronen, S., Pechenizkiy, O. (2006) Class Noise and Supervised Learning in Medical Domains: The Effect of Feature Extraction. In 19th IEEE Symposium on Computer-Based Medical Systems (CBMS'06), pp. 708-713.

[Pérez-Díaz *et. al* 2012] Pérez-Díaz, N., Ruano-Ordás, D., Méndez, J. R., Gálvez, J. F., Fdez-Riverola, F. (2012) Rough Sets for Spam Filtering: Selecting Appropriate Decision Rules for Boundary E-mail Classification. In Applied Soft Computing, v. 12, pp. 3671-3682.

[Raccuglia et al. 2016] Raccuglia, P., Elbert, K. C., Adler, P. D. F., Falk, C., Wenny, M. B., Mollo, A., Zeller, M., Friedler, S. A., Schrier, J. Norquist, A. J. (2016) Machine-learning-assisted Materials Discovery Using Failed Experiments. Nature [0028-0836], v. 533, pp. 73-76A.

[Rani G. *et al.* 2016] Rani G., J. J., Gladis, D., Mammen, J. J. (2017) Extraction of Malignancy Status and Validation of Pathological Classification of Breast Cancer Using Machine Learning Approach. Advances in Natural and Applied Sciences, v. 10, p. 154+.

[Raub & Nehmetallah 2017] Raub, C. B., Nehmetallah, G. (2017) Holography, Machine Learning, and Cancer Cells. Cytometry, pp. 754–756.

[Rissanem 1978] Rissanem, J. (1978) Modeling by Shortest Data Description. Automatica, v. 14, pp. 465-471.

[Rissino & Lambert-Torres 2009] Rissino, S., Lambert-Torres, G. (2009) Rough Set Theory - Fundamental Concepts, Principals, Data Extraction, and Applications, Data Mining and Knowledge Discovery in Real Life Applications. Julio Ponce and Adem Karahoca (Ed.), ISBN: 978-3-902613-53-0, InTech, Available from:

(http://www.intechopen.com/books/data_mining_and_knowledge_discovery_in_real_life_applications/rough_set_theory_-_fundamental_concepts__principals__data_extraction__and_applications).

[Rocha Neto & Barreto 2009] Rocha Neto, A. R., Barreto, G. A. (2009) On the Application of Ensembles of Classifiers to the Diagnosis of Pathologies of the Vertebral Column: A Comparative Analysis. IEEE Latin America Transactions, pp. 487-496.

[Rodrigues & Nicoletti 2018] Rodrigues, E. C.; Nicoletti, M. C. (2018) Extending Flow Graphs for Handling Continuous-valued Attributes. In Proceedings of the 18th International Conference on Hybrid Intelligent Systems (HIS 2018), Porto, Portugal. (a publicar).

[Ruspini 1970] Ruspini, E. H. (1970) Numerical Methods for Fuzzy Clustering. Information Science, v. 2, pp. 319-350.

[Sastry *et al.* 2017] Sastry, A., Monk, J., Tegel, H., Uhlen, M., Palsson, B. O., Rockberg, J., Brunk, E. (2017) Machine Learning in Computational Biology to Accelerate High-throughput Protein Expression. Bioinformatics, v. 33, pp. 2487–2495.

[Schafer 1997] Schafer, J. L. (1997) Analysis of Incomplete Multivariate Data, Chapman and Hall, London.

[Shang *et al.* 2009] Shang, L., Yu, S. Y., Jia, X.Y., Ji, Y.S. (2009) Selection and Optimization of Cut-points for Numeric Attribute Values. Computers & Mathematics with Applications, v. 57, pp. 1018-1023. (http://www.sciencedirect.com/science/article/pii/S0898122108005658).

[Shen & Loh 2004] Shen, L., Loh, H. T. (2004) Applying Rough Sets to Market Timing Decisions. In Decision Support Systems, v. 37, pp. 583-597.

[Stimpson & Cummings 2014] Stimpson, A. J., Cummings, M. L. (2014) Assessing Intervention Timing in Computer-Based Education Using Machine Learning Algorithms. In IEEE Access, v. 2, pp. 78-87.

[Stone *et al.* 1989] Stone, J. R., Blockley, D. I., Pilsworth, B. W. (1989) Towards machine learning from case histories. Civil Engineering Systems, v. 6, pp. 129-135.

[Su *et al.* 2005] Su, M. C., Chou, C. H., Hsieh, C. C. (2005) Fuzzy C-Means Algorithm With a Point Symmetry Distance. International Journal of Fuzzy Systems, v. 7, pp. 175-181.

[Sun *et al.* 2006] Sun, J., Liu, H., Zhang, H. (2006) An Extension of Pawlak's Flow Graphs. Proceedings of the First International Conference on Rough Sets and Knowledge Technology. July 24-26, Chongqing, China.

[Swan *et al.* 2015] Swan, A. L., Stekel, D. J., Hodgman, C., Allaway, D., Alqahtani, M. H., Mobasheri, A., Bacardit, J. (2015) A Machine Learning Heuristic to Identify Biologically Relevant and Minimal Biomarker Panels from Omics Data. BMC Genomics, v. 16.

[SWING 2018] Java Swing. Disponível em: <https://docs.oracle.com/javase/8/docs/api/javax/swing/package-summary.html>. Acesso em: 29. Nov. 2018.

[Taffese & Sistonen 2017] Taffese, W. Z., Sistonen, E. (2017) Machine Learning for Durability and Service-life Assessment of Reinforced Concrete Structures: Recent Advances and Future Directions. Automation in Construction, v. 77, pp. 1-14.

[Theodoridis & Koutroumbas 1999] Theodoridis, S., Koutroumbas, K. (1999) Pattern Recognition, USA: Academic Press.

[Twala 2013] Twala, B. (2013) Impact of Noise on Credit Risk Prediction: Does Data Quality Really Matter?. Intelligent Data Analysis, v.17, pp. 1115-1134.

[Uchôa & Nicoletti 1997] Uchôa, J. Q., Nicoletti, M. C. (1997) Elementos da Teoria de Conjuntos Aproximados. Relatório Técnico do DC 001/97, São Carlos, 1997, 25p.

[Weisberg 1985] Weisberg, S. (1985) Applied Linear Regression. John Wiley & Sons.

[WEKA 2018] Weka 3: Data Mining Software in Java. Disponível em <https://www.cs.waikato.ac.nz/ml/weka/>. Acesso em: 20. Dez. 2018.

[Witten *et al.* 2011] Witten, I. H., Frank E., Hall M. A. (2011) Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann.

[Wuest *et al.* 2014] Wuest, T., Irgens, C. Thoben, K. An Approach to Monitoring Quality in Manufacturing Using Supervised Machine Learning on Product State Data. Journal of Intelligent Manufacturing [0956-5515], v. 25, pp.1167-1180.

[Yang *et al.* 2011] Yang, Y.-P. O., Shieh, H.-M., Tzeng, G.-H., Yen, L., Chan, C.-C. (2011) Combine Rough Sets With Flow Graph and Formal Concept Analysis for Business Aviation Decision-making. Journal of Intelligent Systems, v. 36, pp. 347-366.

[Yeh *et al.* 2008] Yeh, I-C., Yang, K.-J., Ting, T.-M. (2008) Knowledge Discovery on RFM Model using Bernoulli Sequence. Expert Systems with Applications, 2008.

[Zadeh 1965] Zadeh, L. A. (1965) Fuzzy Sets. Information and Control, New York, Academic Press, no. 8, pp. 338-353.

[Zain Amin & Amir Ali 2018] Zain Amin, M., Ali, A. (2018) Performance Evaluation of Supervised Machine Learning Classifiers for Predicting Healthcare Operational Decisions. Machine Learning for Operational Decision Making, Wavy Artificial Intelligence Research Foundation, Pakistan.

[Zhang *et. al* 2016] Zhang, Q., Xie, Q., Wang, G. (2016) A Survey on Rough Set Theory and its Applications. In CAAI Transactions on Intelligence Technology, v. 1, no. 4, pp. 323-333.

[Zhu & Wu 2004] Zhu, X., Wu, X. (2004) Class Noise vs. Attribute Noise: A Quantitative Study. Artificial Intelligence Review, v. 22, no. 3, pp. 177-210.

ANEXOS

Anexo I – Certificado de Apresentação de Artigo em Conferência Internacional

18th International Conference on
Hybrid Intelligent Systems
Porto, Portugal December 13-15, 2018

http://www.mirlabs.net/his18

December 13, 2018

Certificate

This is to certify that the following paper was presented during the 18th International Conference on Hybrid Intelligent Systems (HIS 2018) in Porto, Portugal.

Paper ID: 14

Paper title: Extending Flow Graphs for Handling Continuous-valued Attributes

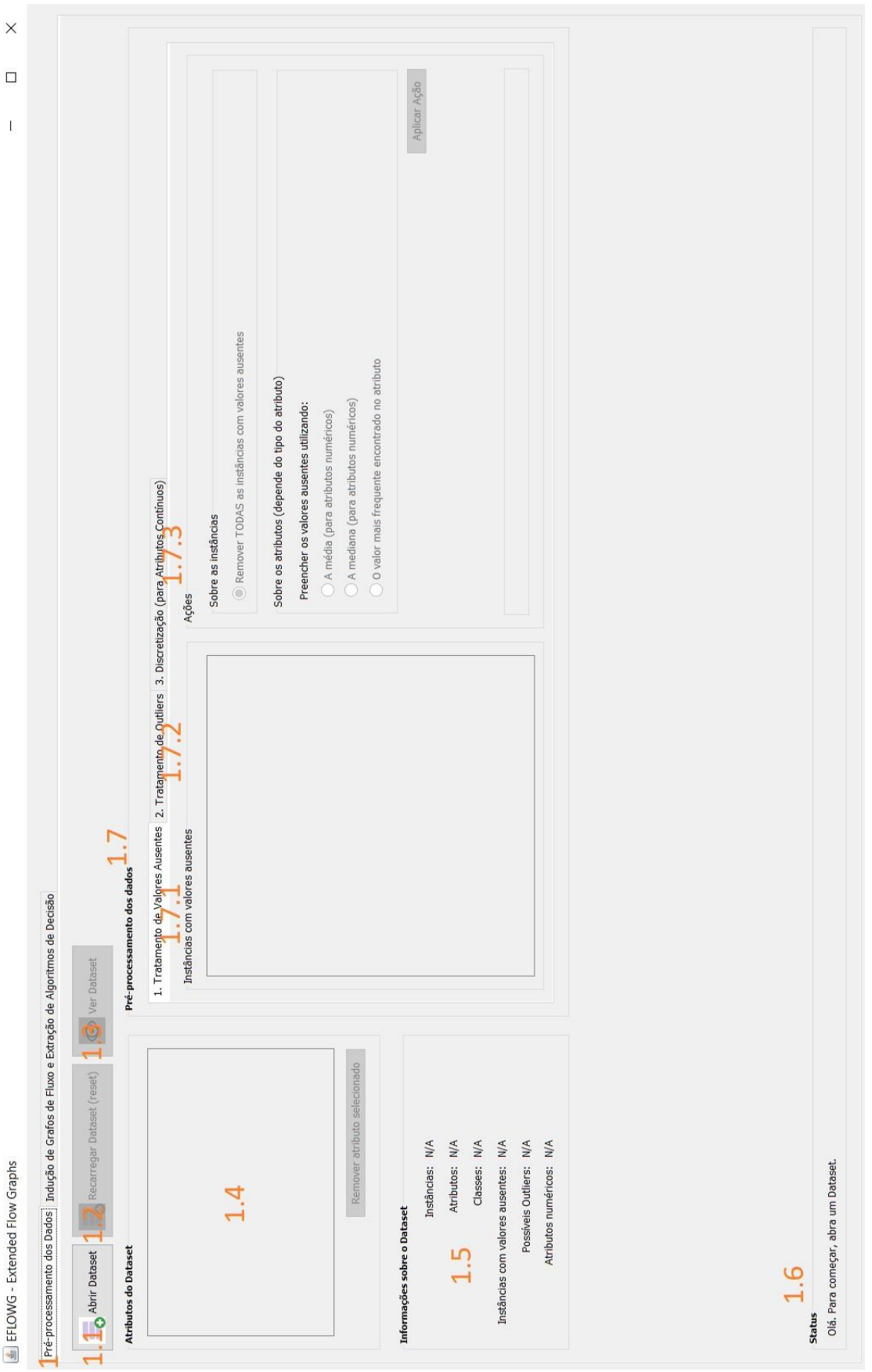
Authors: Emilio Carlos Rodrigues and Maria Do Carmo Nicoletti

Yours sincerely,

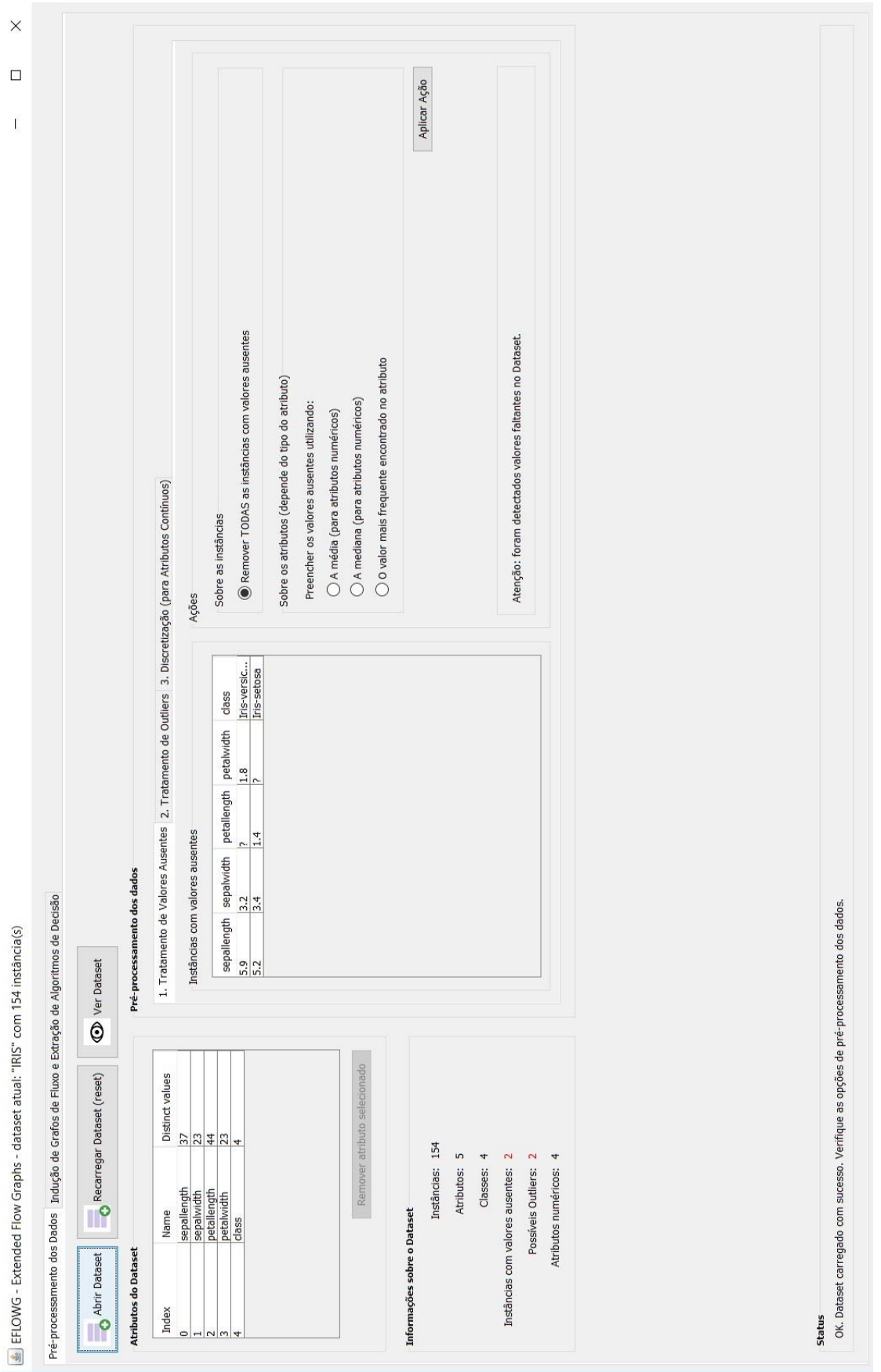
Prof. Dr. Ana Maria Madureira
Instituto Superior de Engenharia do Porto, Portugal
HIS 2018 –General Chair

isep Instituto Superior de Engenharia do Porto

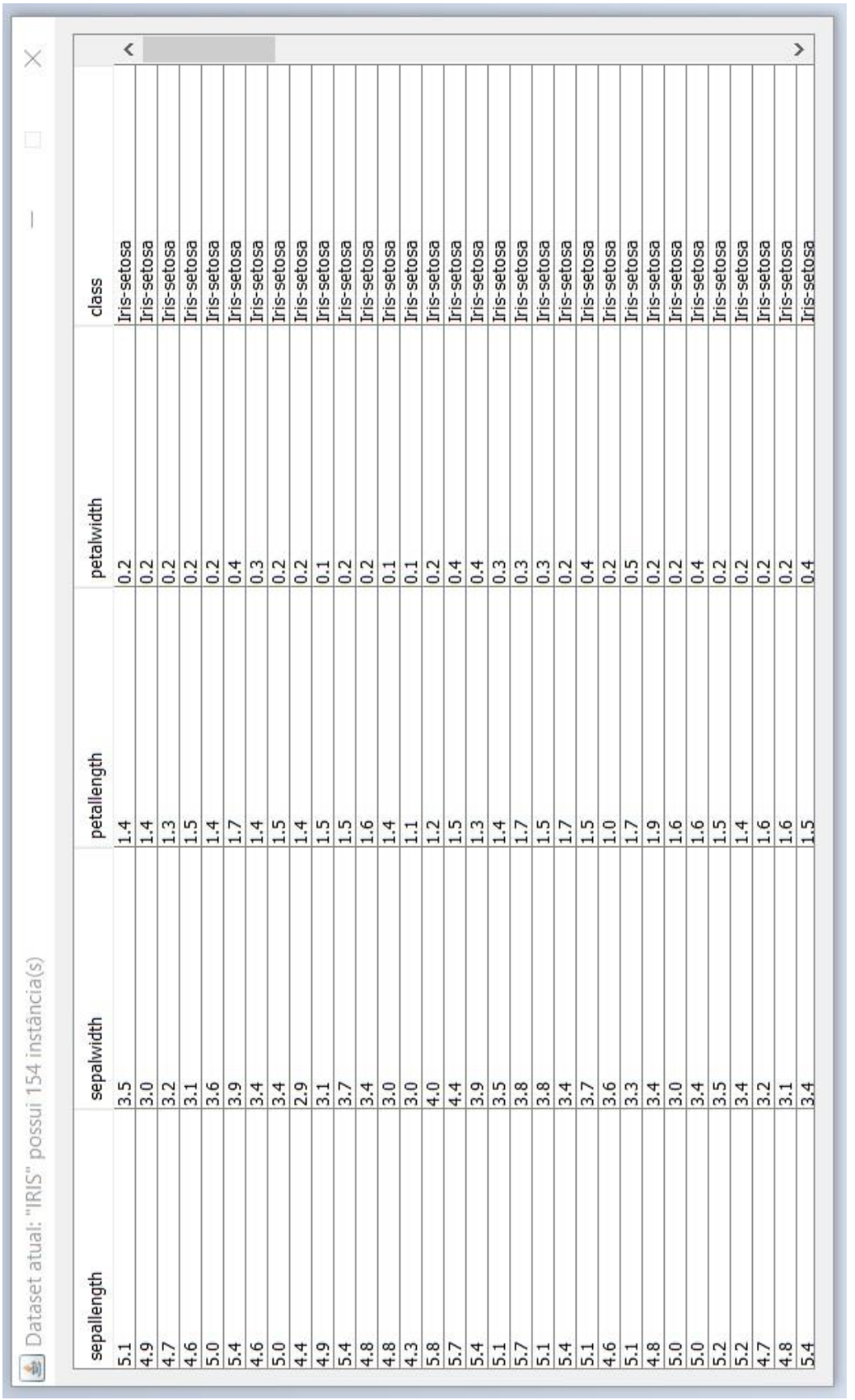
Anexo II – Tela Inicial do sistema EFLOWG



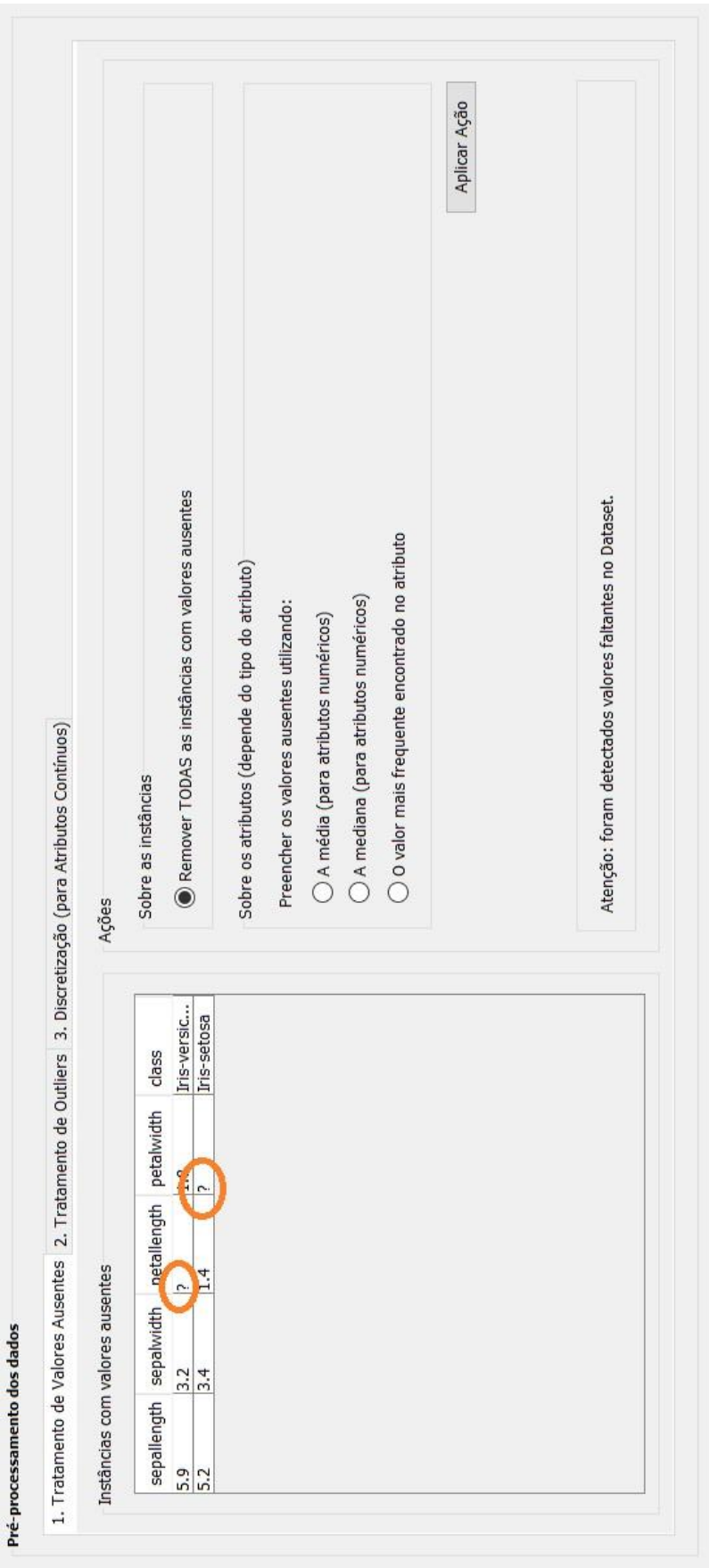
Anexo III – Conjunto de instâncias de dados Iris aberto no sistema EFLOWG



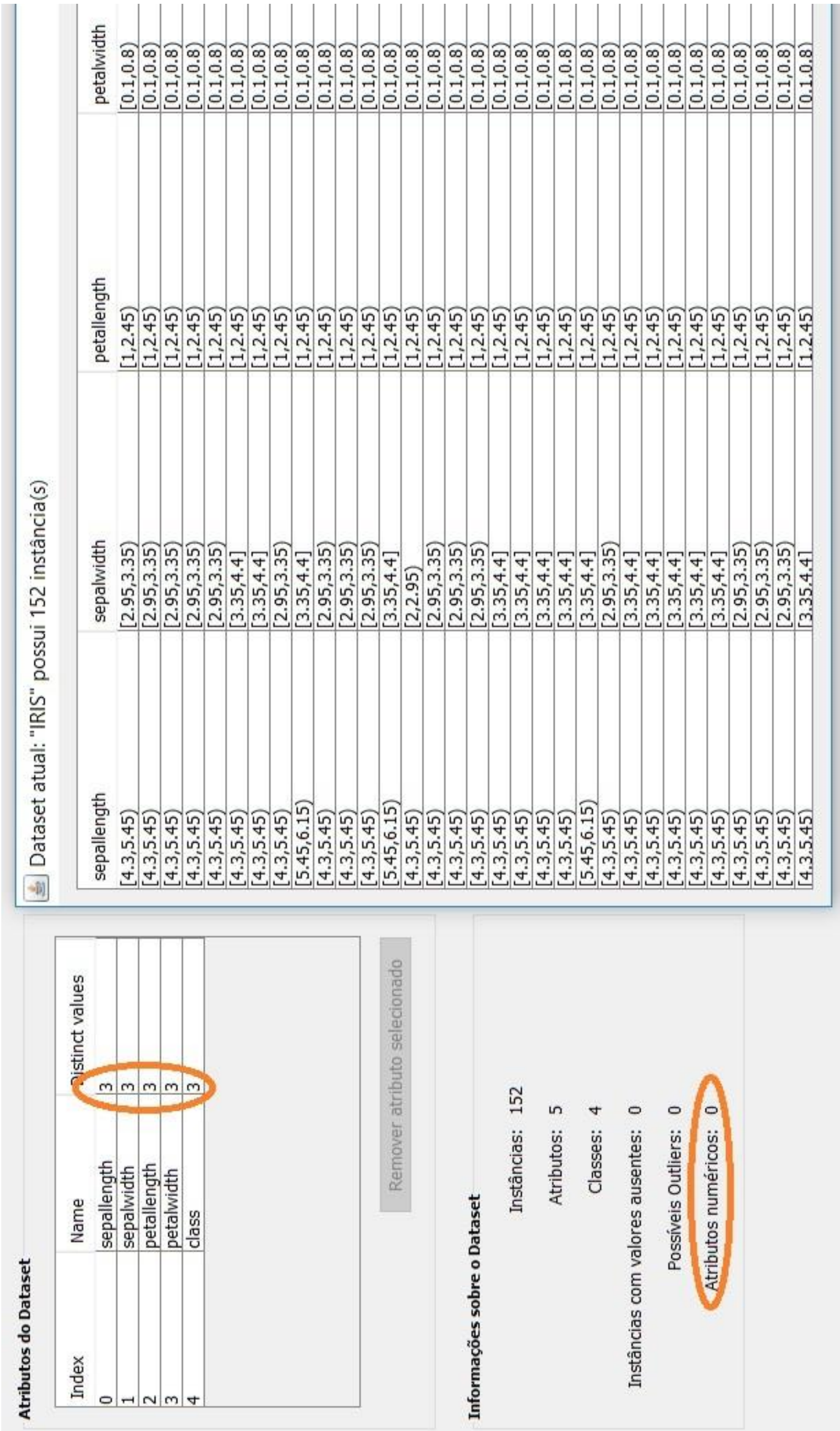
Anexo IV – Visualização das instâncias do conjunto de instância de dados Iris aberto no EFLOWG



Anexo V – A guia “Tratamento de Valores Ausentes” do EFLOWG



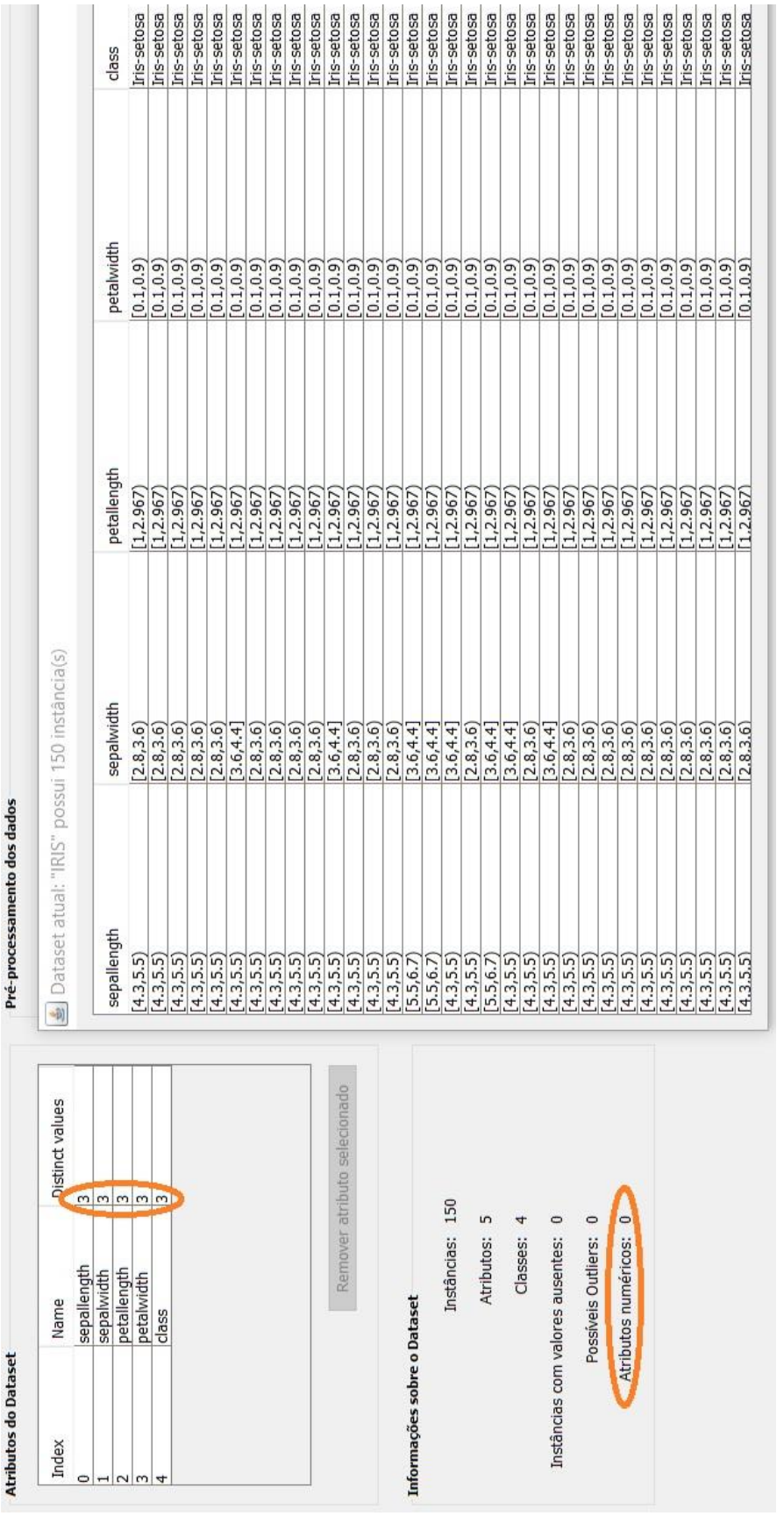
Anexo VI – Resultado obtido com a discretização por Entropia e MDLP sobre o conjunto de instâncias de dados Iris



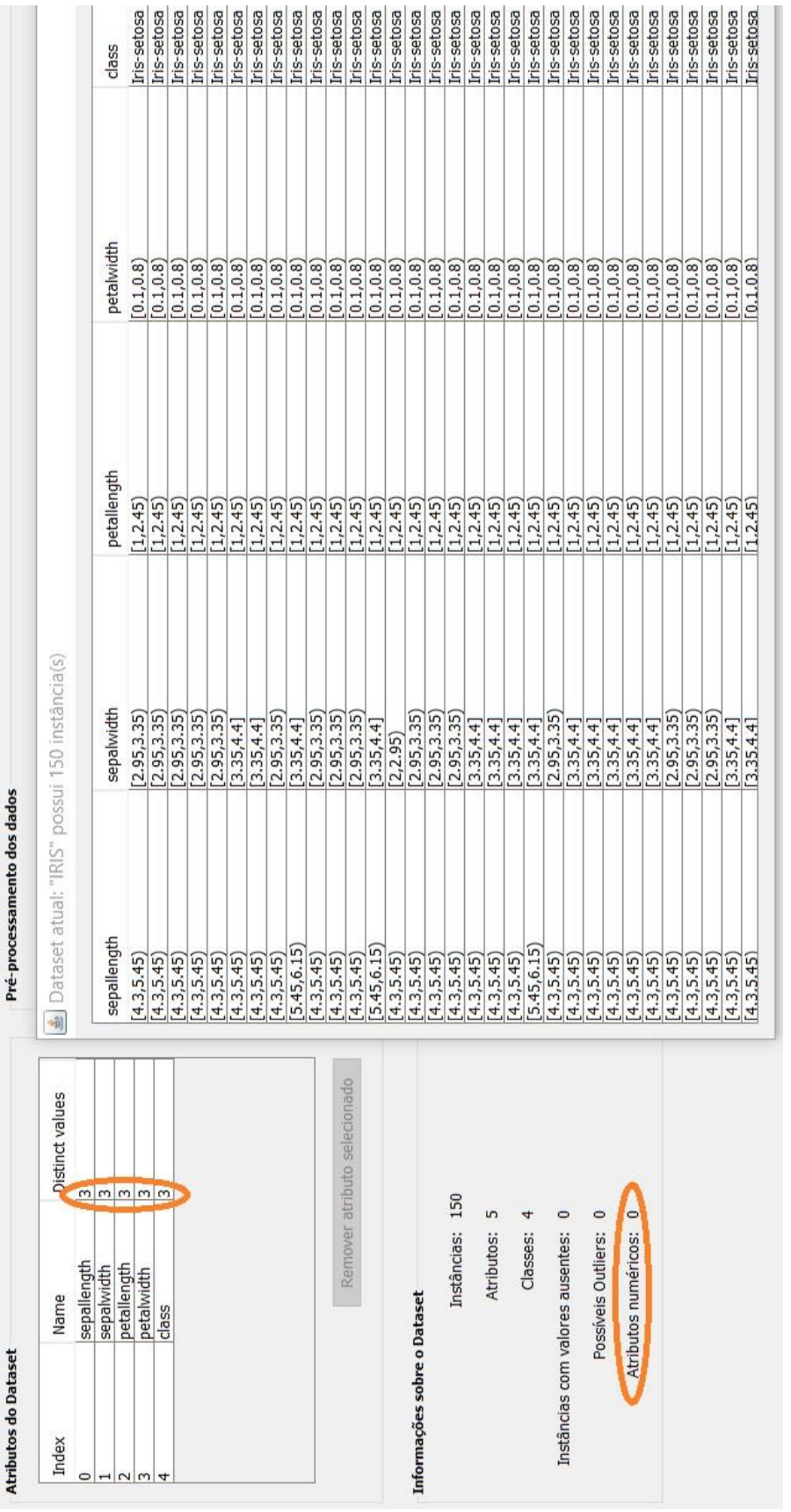
Index	Name	Distinct values
0	sepallength	3
1	sepalwidth	3
2	petallength	3
3	petalwidth	3
4	class	3

sepallength	sepalwidth	petallength	petalwidth
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[5.45,6.15)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[5.45,6.15)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2,2.95)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[5.45,6.15)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[2.95,3.35)	[1,2.45)	[0.1,0.8)
[4.3,5.45)	[3.35,4.4]	[1,2.45)	[0.1,0.8)

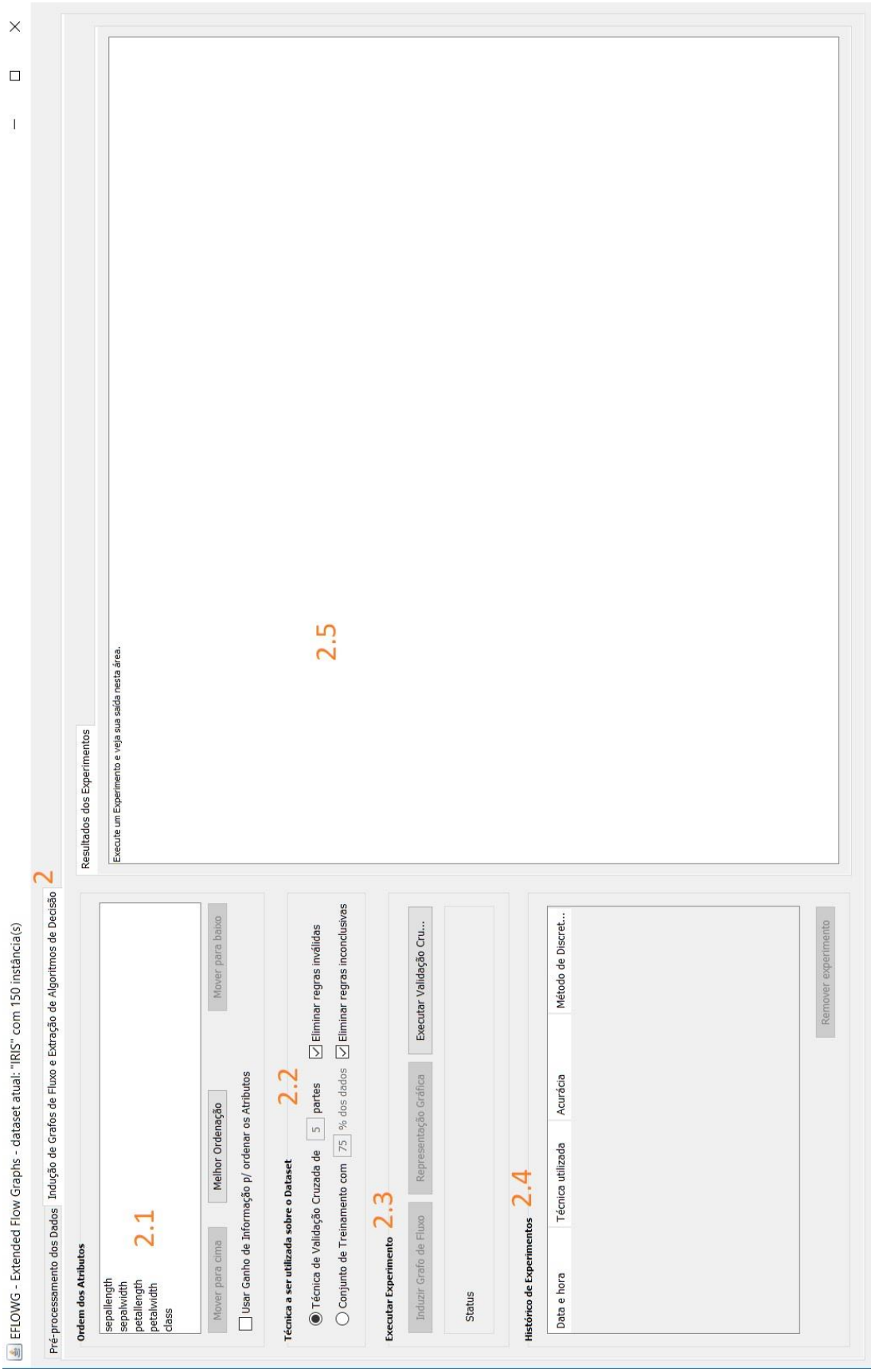
Anexo VII – Resultado obtido com a discretização por Amplitudes Iguais sobre o conjunto de instâncias de dados Iris



Anexo VIII – Resultado obtido com a discretização por Iguais Frequências sobre o conjunto de instâncias de dados Iris



Anexo IX – A guia “Indução de Grafos de Fluxo e Extração de Algoritmos de Decisão” do sistema EFLOWG



Anexo X – Parte da representação gráfica, do GF induzido a partir do conjunto de instâncias de dados Iris, gerada pelo sistema EFLOWG